Pentium® and Pentium Pro Processors and Related Products

Pentium® Pro Processor

Pentium® Processor

OverDrive® Processor

PCIsets

Peripheral Components

intel®

LITERATURE

For additional information on Intel products in the U.S. or Canada, call Intel's Literature Center at (800) 548-4725 or write to:

INTEL LITERATURE SALES
P. O. Box 7641
Mt. Prospect, IL 60056-7641

To order literature outside of the U.S. and Canada contact your local sales office.

Additional information about Intel products is available on Intel's web site: http://www.intel.com.

CURRENT DATABOOKS

Product line databooks contain datasheets, application notes, article reprints, and other design information. All databooks can be ordered individually, and most are available in a pre-packaged set in the U.S. and Canada. Databooks can be ordered in the U.S. and Canada by calling TAB/McGraw-Hill at 1-800-822-8158; outside of the U.S. and Canada contact your local sales office.

Title	Intel Order Number	ISBN
SET OF NINE DATABOOKS (Available in U.S. and Canada)	**231003**	**N/A**
CONTENTS LISTED BELOW FOR INDIVIDUAL ORDERING:		
EMBEDDED MICROCONTROLLERS	270646	1-55512-248-5
EMBEDDED MICROPROCESSORS	272396	1-55512-249-3
FLASH MEMORY (2 volume set)	210830	1-55512-250-7
i960® PROCESSORS AND RELATED PRODUCTS	272084	1-55512-252-3
NETWORKING	297360	1-55512-256-6
OEM BOARDS, SYSTEMS AND SOFTWARE	280407	1-55512-253-1
PACKAGING	240800	1-55512-254-X
PENTIUM® AND PENTIUM PRO PROCESSORS AND RELATED PRODUCTS	241732	1-55512-251-5
PERIPHERAL COMPONENTS	296467	1-55512-255-8
ADDITIONAL LITERATURE: (Not included in handbook set)		
AUTOMOTIVE PRODUCTS	231792	1-55512-257-4
COMPONENTS QUALITY/RELIABILITY	210997	1-55512-258-2
EMBEDDED APPLICATIONS (1995/96)	270648	1-55512-179-9
MILITARY	210461	N/A
SYSTEMS QUALITY/RELIABILITY	231762	1-55512-046-6

A complete set of this information is available on CD-ROM through Intel's Data on Demand program, order number 240897. For information about Intel's Data on Demand ask for item number 240952.

December 1995
Order Number: 000900-001

Intel Application Support Services

World Wide Web [URL: http://www.intel.com/]

Intel's Web site now contains technical and product information that is available 24-hours a day! Also visit Intel's site for financials, history, current news and events, job opportunities, educational news and much, much more!

FaxBack*

Technical and product information are available 24-hours a day! Order documents containing:

- Product Announcements
- Product Literature
- Intel Device Characteristics
- Design/Application Recommendations
- Stepping/Change Notifications
- Quality and Reliability Information

Information on the following subjects are available:

- Microcontroller and Flash
- OEM Branded Systems
- Multibus and iRMX Software/BBS listing
- Multimedia
- Development Tools
- Quality and Reliability/Change Notification
- Microprocessor/PCI/Peripheral
- Intel Architecture Labs

To use FaxBack (for Intel components and systems), dial **(800) 628-2283** or 916-356-3105 (U.S./Canada/APAC/Japan) *or +44{0} 1793-496646 (Europe)* and follow the automated voice-prompt. Document orders will be faxed to the fax number you specify. For information on how the Intel Application Support team can help you, order our Customer Service Agreement, document #1201. Catalogs are updated as needed, so call for the latest information!

Bulletin Board System (BBS)

To use the Intel Application BBS (components and systems), dial **(503) 264-7999** or **(916) 356-3600** (U.S./Canada/APAC/Japan) *or +44{0} 1793-432955 (Europe).* The BBS will support 1200-19200 baud rate modem. *Typical modem configuration: 14.4K baud rate, No Parity, 8 Data Bits, 1 Stop Bit.*

CompuServe *Just type* 'Go Intel'

Intel maintains several forums where people come together to meet their peers, gather information, share discoveries and debate issues. For more information about service fees and access, call CompuServe at 1-800-848-8199 or 614-529-1340 (outside the U.S.). The INTELC forum is setup to support designers using various Intel components.

General Information Help Desk

Dial 1-800-628-8686 or 916-356-7599 (U.S. and Canada) between 5 a.m. and 5 p.m. PST for help with Intel products. For customers not in the U.S. or Canada, please contact your local distributor.

Intel Literature Centers

U.S.	+1-800-548-4725	France	+44{0} 1793 421777
U.S. (from overseas)	+1-708-296-9333	Germany	+44{0} 1793 421333
England	+44{0} 1793 431 155	Japan (fax only)	+81{0} 120 47 88 32

Intel Distributors

Check the back of an Intel data book or request one of the following distributor listing FaxBack documents: #4083 (U.S. Eastern Time Zone), #4084 (U.S. Central Time Zone), #4085 (Mountain Time Zone), #4086 (U.S. Alaska/Pacific Time Zone), #4209 (Europe) or #4403 (Canada).

December 1995
Order Number: 000901-001

intel®

Pentium® and Pentium Pro Processors and Related Products

Microprocessors, PCIsets, Peripheral Components

1996

Information in this document is provided in connection with Intel products. Intel assumes no liability whatsoever, including infringement of any patent or copyright, for sale and use of Intel products except as provided in Intel's Terms and Conditions of Sale for such products.

Intel retains the right to make changes to these specifications at any time, without notice. Microcomputer Products may have minor variations to this specification known as errata.

*Other brands and names are the property of their respective owners.

†Since publication of documents referenced in this document, registration of the Pentium, OverDrive and iCOMP trademarks has been issued to Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

DATASHEET DESIGNATIONS

Intel uses various datasheet markings to designate each phase of the document as it relates to the product. The markings appear in the lower inside corner of each datasheet page. Following are the definitions of each marking:

Datasheet Marking	Description
Product Preview	Contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product becomes available.
Advanced Information	Contains information on products being sampled or in the initial production phase of development.*
Preliminary	Contains preliminary information on new products in production.*
No Marking	Contains information on products in full production.*

* Specifications within these datasheets are subject to change without notice. Verify with your local Intel sales office that you have the latest datasheet before finalizing a design.

November 1995
Order Number: 000902-001

intel®

Pentium® Pro Processor 1

Pentium® Processor
and Related Products 2

Table of Contents

Table of Contents (Continued)

Alphanumeric Index

1

Pentium® Pro Processor

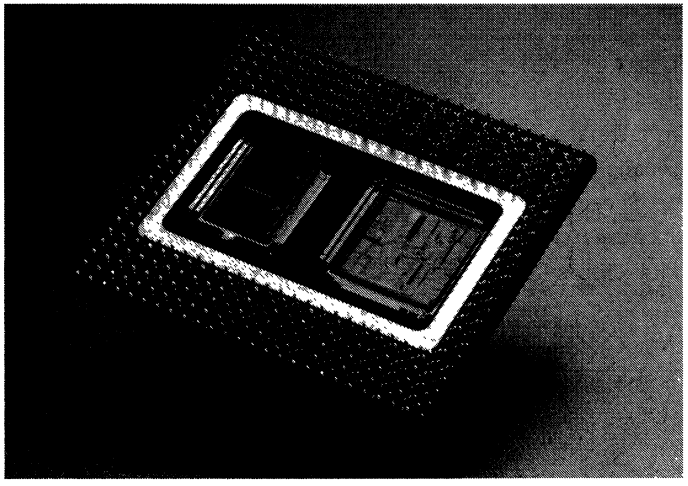
PENTIUM® PRO PROCESSOR AT 150 MHz

- **SPECint95* integer performance of 6.08 and SPECfp95* floating point performance of 5.41 at 150 MHz**
- **Binary compatible with applications running on previous members of the Intel microprocessor family**
- **Optimized for 32-bit applications running on advanced 32-bit operating systems**
- **Dynamic Execution microarchitecture**
- **Single package includes Pentium® Pro Processor CPU, cache and system bus interface**
- **Scalable up to four processors and 4 GB memory**
- **Separate dedicated external system bus, and dedicated internal full-speed cache bus**
- **8K/8K separate data and instruction, non-blocking, level one cache**
- **Integrated 256 KB, non-blocking, level two cache on package**
- **Data integrity and reliability features include ECC, Fault Analysis/Recovery, and Functional Redundancy Checking**
- **Upgradable to a Future OverDrive® Processor**

1

The Pentium® Pro processor family is Intel's next generation of performance for high-end desktops, workstations and servers. The family consists of processors at 150 Mhz and higher and is easily scalable to up to four microprocessors in a multiprocessor system. The Pentium Pro processor delivers more performance than previous generation processors through an innovation called Dynamic Execution. This is the next step beyond the superscalar architecture implemented in the Pentium processor. This makes possible the advanced 3D visualization and interactive capabilities required by today's high-end commercial and technical applications and tomorrow's emerging applications. The Pentium Pro processor also includes advanced data integrity, reliability, and serviceability features for mission critical applications.

The Pentium Pro processor may contain design defects or errors known as errata. Current characterized errata are available upon request.



242764-A4

December 1995
Order Number: 242769-001

PENTIUM® PRO PROCESSOR AT 150 MHz

CONTENTS

CONTENTS

CONTENTS

CONTENTS

1.0. INTRODUCTION

The Pentium Pro processor is the next in the Intel386™, Intel486™, and Pentium family of processors. The Pentium Pro processor implements a *Dynamic Execution* micro-architecture — a unique combination of multiple branch prediction, data flow analysis, and speculative execution.

The Pentium Pro processor is upgradable by a future OverDrive® processor and matching voltage regulator module described in Section 8.

Increasing clock frequencies and silicon density can complicate system designs. The Pentium Pro processor integrates several system components which alleviate some of the previous system burdens. The second level cache, cache controller, and the Advanced Programmable Interrupt Controller (APIC) are some of the components that existed in previous Intel processor family systems which are integrated into this single component. This integration results in the Pentium Pro processor bus more closely resembling a symmetric multi-processing (SMP) system bus rather than resembling a previous generation processor-to-cache bus. This added level of integration and improved performance, results in higher power consumption and a new bus technology. This means it is more important than ever to ensure adherence to this specification.

A significant new feature of the Pentium Pro processor, from a system perspective, is the built-in direct multi-processing support. In order to achieve multi-processing for up to four processors, and maintain the memory and I/O bandwidth to support them, new system designs are needed. In creating a system with multiple processors, it is important to consider the additional power burdens and signal integrity issues of supporting up to 8 loads on a high speed bus.

1.1. TERMINOLOGY

A '#' symbol after a signal name refers to an active low signal. This means that a signal is in the active state (based on the name of the signal) when driven low. For example, when FLUSH# is low a flush has been requested. When NMI is high, a Non-maskable interrupt has occurred. In the case of lines where the name does not imply an active state but describes part of a binary sequence (such as address or data), the '#' symbol implies that the signal is inverted. For example, D[3:0] = 'HLHL' refers to a hex 'A', and D#[3:0] = 'LHLH' also refers to a hex 'A'. (H= High logic level, L= Low logic level)

The word *Preliminary* appears in a few places. Check with your local FAE for recent information.

1.2. REFERENCES

The following are referenced within this specification:

- *Pentium® Pro Processor I/O Buffer Models*— IBIS Format (On world wide web page www.intel.com)
- AP-523, *Pentium® Pro Processor Power Distribution Guidelines* Application Note (Order Number 242764)
- AP-524, *Pentium® Pro Processor GTL+ Layout Guidelines* Application Note (Order Number 242765)
- AP-525, *Pentium® Pro Processor Thermal Design Guidelines* Application Note (Order Number 242766)
- *Pentium® Pro Processor Developer's Manual, Volume 1: Specifications* (Order Number 242690)
- *Pentium® Pro Processor Developer's Manual, Volume 2: Programmer's Reference Manual* (Order Number 242691)
- *Pentium® Pro Processor Developer's Manual, Volume 3: Operating System Developer's Guide* (Order Number 242692)

2.0. PENTIUM® PRO PROCESSOR ARCHITECTURE OVERVIEW

The Pentium Pro processor has a decoupled, 12-stage, superpipelined implementation, trading less work per pipestage for more stages. The Pentium Pro processor also has a pipestage time 33 percent less than the Pentium processor, which helps achieve a higher clock rate on any given process.

The approach used by the Pentium Pro processor removes the constraint of linear instruction sequencing between the traditional "fetch" and "execute" phases, and opens up a wide instruction window using an instruction pool. This approach allows the "execute" phase of the Pentium Pro

processor to have much more visibility into the program's instruction stream so that better scheduling may take place. It requires the instruction "fetch/decode" phase of the Pentium Pro processor to be much more intelligent in terms of predicting program flow. Optimized scheduling requires the fundamental "execute" phase to be replaced by decoupled "dispatch/execute" and "retire" phases. This allows instructions to be started in any order but always be completed in the original program order. The Pentium Pro processor is implemented as three independent engines coupled with an instruction pool as shown in Figure 1.

2.1. Full Core Utilization

The three independent-engine approach was taken to more fully utilize the CPU core. Consider the code fragment in Figure 2:

The first instruction in this example is a load of r1 that, at run time, causes a cache miss. A traditional CPU core must wait for its bus interface unit to read this data from main memory and return it before moving on to instruction 2. This CPU stalls while waiting for this data and is thus being under-utilized.

To avoid this memory latency problem, the Pentium Pro processor "looks-ahead" into its instruction pool at subsequent instructions and will do useful work rather than be stalled. In the example in Figure 2, instruction 2 is not executable since it depends upon the result of instruction 1; however both instructions 3 and 4 are executable. The Pentium Pro processor executes instructions 3 and 4 out-of-order. The results of this out-of-order execution can not be committed to permanent machine state (i.e., the programmer-visible registers) immediately since the original program order must be maintained. The results are instead stored back in the instruction pool awaiting in-order retirement. The core executes instructions depending upon their readiness to execute, and not on their original program order, and is therefore a true dataflow engine. This approach has the side effect that instructions are typically executed out-of-order.

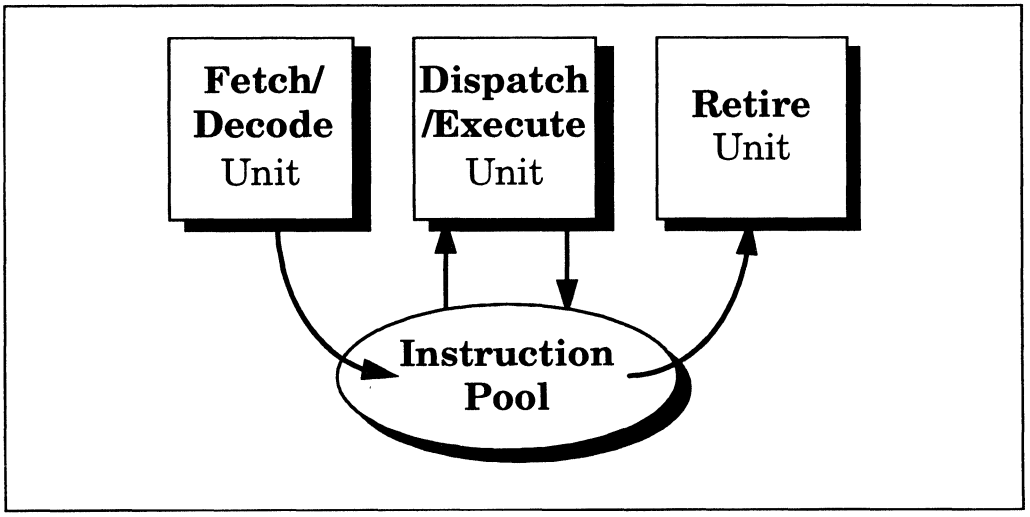


Figure 1. Three Engines Communicating Using an Instruction Pool

```
r1 <= mem [r0]    /* Instruction 1 */
r2 <= r1 + r2     /* Instruction 2 */
r5 <= r5 + 1      /* Instruction 3 */
r6 <= r6 - r3     /* Instruction 4 */
```

Figure 2. A Typical Code Fragment

The cache miss on instruction 1 will take many internal clocks, so the Pentium Pro processor core continues to look ahead for other instructions that could be speculatively executed, and is typically looking 20 to 30 instructions in front of the instruction pointer. Within this 20 to 30 instruction window there will be, on average, five branches that the fetch/decode unit must correctly predict if the dispatch/execute unit is to do useful work. The sparse register set of an Intel Architecture (IA) processor will create many false dependencies on registers so the dispatch/execute unit will rename the IA registers into a larger register set to enable additional forward progress. The retire unit owns the programmer's IA register set and results are only committed to permanent machine state in these registers when it removes completed instructions from the pool in original program order.

Dynamic Execution technology can be summarized as optimally adjusting instruction execution by predicting program flow, having the ability to speculatively execute instructions in any order, and then analyzing the program's dataflow graph to choose the best order to execute the instructions.

2.2. The Pentium® Pro Processor Pipeline

In order to get a closer look at how the Pentium Pro processor implements Dynamic Execution, Figure 3 shows a block diagram including cache and memory interfaces. The "Units" shown in Figure 3 represent groups of stages of the Pentium Pro processor pipeline.

- The FETCH/DECODE unit: An in-order unit that takes as input the user program instruction stream from the instruction cache, and decodes them into a series of micro-operations (μops) that represent the dataflow of that instruction stream. The pre-fetch is speculative.
- The DISPATCH/EXECUTE unit: An out-of-order unit that accepts the dataflow stream, schedules execution of the μops subject to data dependencies and resource availability and temporarily stores the results of these speculative executions.
- The RETIRE unit: An in-order unit that knows how and when to commit ("retire") the temporary, speculative results to permanent architectural state.

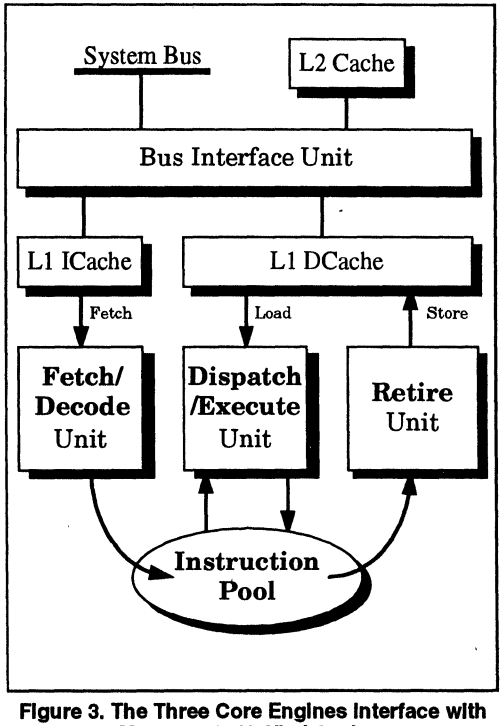


Figure 3. The Three Core Engines Interface with Memory via Unified Caches

- The BUS INTERFACE unit: A partially ordered unit responsible for connecting the three internal units to the real world. The bus interface unit communicates directly with the L2 (second level) cache supporting up to four concurrent cache accesses. The bus interface

unit also controls a transaction bus, with MESI snooping protocol, to system memory.

2.2.1. THE FETCH/DECODE UNIT

Figure 4 shows a more detailed view of the Fetch/Decode Unit.

The ICache is a local instruction cache. The Next_IP unit provides the ICache index, based on inputs from the Branch Target Buffer (BTB), trap/interrupt status, and branch-misprediction indications from the integer execution section.

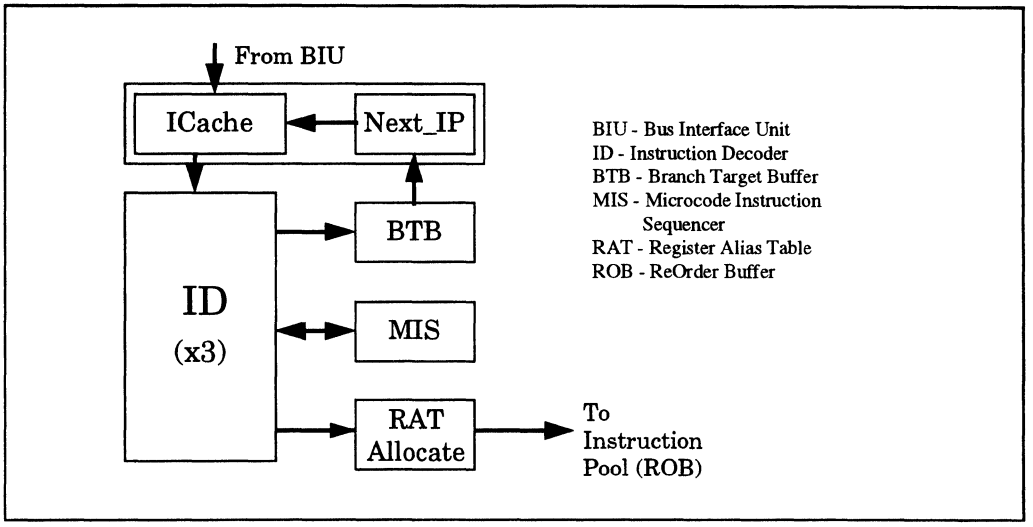


Figure 4. Inside the Fetch/Decode Unit

The ICache fetches the cache line corresponding to the index from the Next_IP, and the next line, and presents 16 aligned bytes to the decoder. The prefetched bytes are rotated so that they are justified for the instruction decoders (ID). The beginning and end of the IA instructions are marked.

Three parallel decoders accept this stream of marked bytes, and proceed to find and decode the IA instructions contained therein. The decoder converts the IA instructions into triadic μops (two logical sources, one logical destination per μop). Most IA instructions are converted directly into single μops, some instructions are decoded into one-to-four μops and the complex instructions require microcode (the box labeled MIS in Figure 4). This microcode is just a set of preprogrammed sequences of normal μops. The μops are queued, and sent to the Register Alias Table (RAT) unit, where the logical IA-based register references are converted into Pentium Pro processor physical register references, and to the Allocator stage, which adds status information to the μops and enters them into the instruction pool. The instruction pool is implemented as an array of Content Addressable Memory called the ReOrder Buffer (ROB).

This is the end of the in-order pipe.

2.2.2. THE DISPATCH/EXECUTE UNIT

The dispatch unit selects μops from the instruction pool depending upon their status. If the status indicates that a μop has all of its operands then the dispatch unit checks to see if the execution resource needed by that μop is also available. If both are true, the *Reservation Station* removes that μop and sends it to the resource where it is executed. The results of the μop are later returned to the pool. There are five ports on the Reservation Station, and the multiple resources are accessed as shown in Figure 5.

The Pentium Pro processor can schedule at a peak rate of 5 µops per clock, one to each resource port, but a sustained rate of 3 µops per clock is typical. The activity of this scheduling process is the out-of-order process; µops are dispatched to the execution resources strictly according to dataflow constraints and resource availability, without regard to the original ordering of the program.

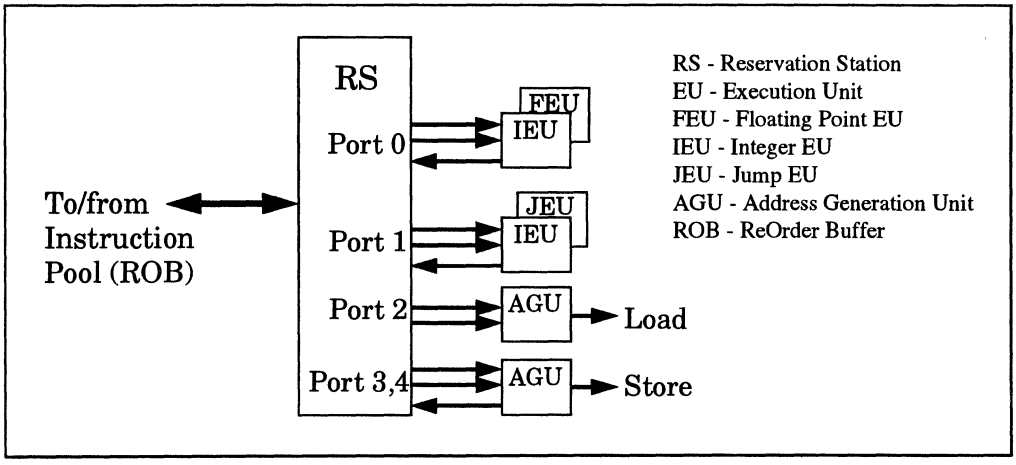


Figure 5. Inside the Dispatch/Execute Unit

Note that the actual algorithm employed by this execution-scheduling process is vitally important to performance. If only one µop per resource becomes data-ready per clock cycle, then there is no choice. But if several are available, it must choose. The Pentium Pro processor uses a pseudo FIFO scheduling algorithm favoring back-to-back µops.

Note that many of the µops are branches. The Branch Target Buffer will correctly predict most of these branches but it can't correctly predict them all. Consider a BTB that is correctly predicting the backward branch at the bottom of a loop; eventually that loop is going to terminate, and when it does, that branch will be mispredicted. Branch µops are tagged (in the in-order pipeline) with their fall-through address and the destination that was predicted for them. When the branch executes, what the branch actually did is compared against what the prediction hardware said it would do. If those coincide, then the branch eventually retires, and most of the speculatively executed work behind it in the instruction pool is good.

But if they do not coincide, then the Jump Execution Unit (JEU) changes the status of all of the µops behind the branch to remove them from the instruction pool. In that case the proper branch destination is provided to the BTB which restarts the whole pipeline from the new target address.

2.2.3. THE RETIRE UNIT

Figure 6 shows a more detailed view of the Retire Unit.

The retire unit is also checking the status of µops in the instruction pool. It is looking for µops that have executed and can be removed from the pool. Once removed, the original architectural target of the µops is written as per the original IA instruction. The retirement unit must not only notice which µops are complete, it must also re-impose the original program order on them. It must also do this in the face of interrupts, traps, faults, breakpoints and mispredictions.

The retirement unit must first read the instruction pool to find the potential candidates for retirement and determine which of these candidates are next in the original program order. Then it writes the results of this cycle's retirements to both the Instruction Pool and the Retirement Register File (RRF). The retirement unit is capable of retiring 3 µops per clock.

2.2.4. THE BUS INTERFACE UNIT

Figure 7 shows a detailed view of the Bus Interface Unit.

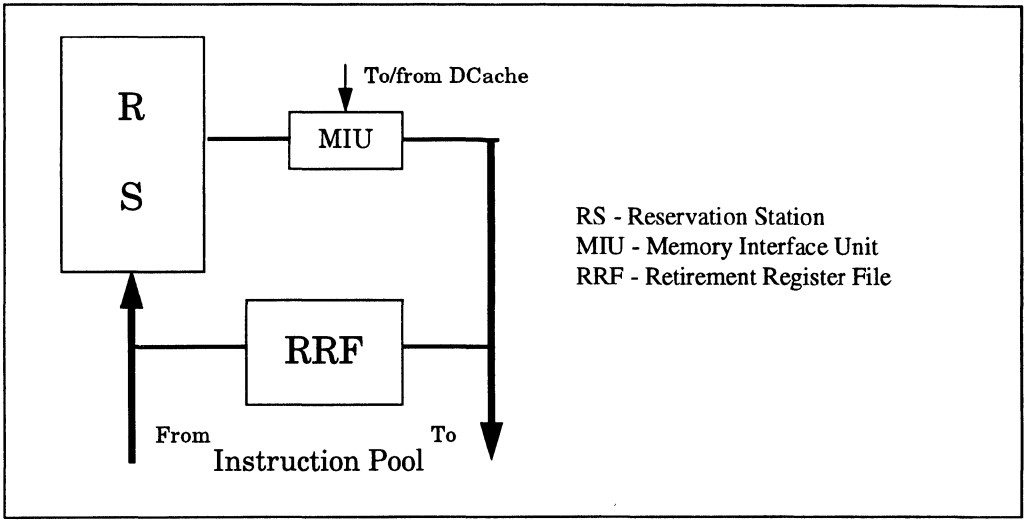


Figure 6. Inside the Retire Unit

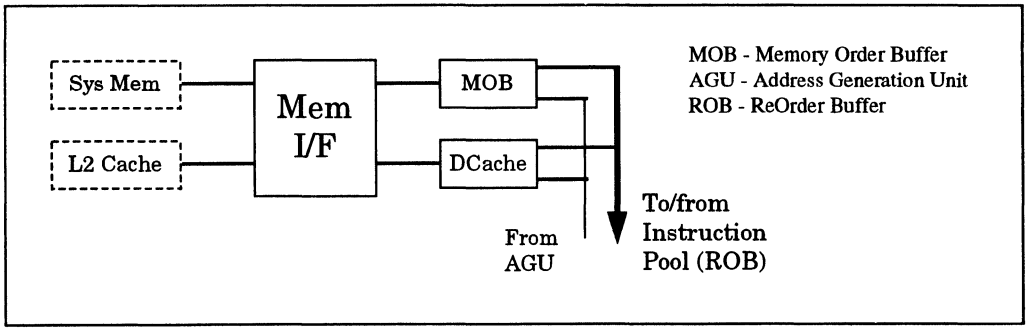


Figure 7. Inside the Bus Interface Unit

There are two types of memory access: loads and stores. Loads only need to specify the memory address to be accessed, the width of the data being retrieved, and the destination register. Loads are encoded into a single µop.

Stores need to provide a memory address, a data width, and the data to be written. Stores therefore require two µops, one to generate the address, and one to generate the data. These µops must later re-combine for the store to complete.

Stores are never performed speculatively since there is no transparent way to undo them. Stores are also never re-ordered among themselves. A store is dispatched only when both the address and the data are available and there are no older stores awaiting dispatch.

A study of the importance of memory access reordering concluded:

- Stores must be constrained from passing other stores, for only a small impact on performance.
- Stores can be constrained from passing loads, for an inconsequential performance loss.

- Constraining loads from passing other loads or stores has a significant impact on performance.

The Memory Order Buffer (MOB) allows loads to pass other loads and stores by acting like a reservation station and re-order buffer. It holds suspended loads and stores and re-dispatches them when a blocking condition (dependency or resource) disappears.

2.3. Architecture Summary

Dynamic Execution is this combination of improved branch prediction, speculative execution and data flow analysis that enables the Pentium Pro processor to deliver its superior performance.

3.0. ELECTRICAL SPECIFICATIONS

3.1. The Pentium® Pro Processor Bus and V_{REF}

Most of the Pentium Pro processor signals use a **variation** of the low voltage GTL signaling technology (Gunning Transceiver Logic).

The Pentium Pro processor bus specification is similar to the GTL specification but has been enhanced to provide larger noise margins and reduced ringing. This is accomplished by increasing the termination voltage level and controlling the edge rates. Because this specification is different from the standard GTL specification, it is refered to as *GTL+* in this document.

The GTL+ signals are open-drain and require external termination to a supply that provides the high signal level. The GTL+ inputs use differential receivers which require a reference signal (V_{REF}). Termination (Usually a resistor on each end of the signal trace) is used to pull the bus up to the high voltage level and to control reflections on the stub-free transmission line. V_{REF} is used by the receivers to determine if a signal is a logical 0 or a logical 1. See **Table 8** for the bus termination voltage specifications for GTL+ , and Section 4 for the GTL+ Interface Specification.

There are 8 V_{REF} pins on the Pentium Pro processor to ensure that internal noise will not affect the performance of the I/O buffers. Pins A1, C7, S7 and Y7 (V_{REF}[3:0]) must be tied together and pins A47, U41, AE47 and AG45 (V_{REF}[7:4]) must be tied together. The two groups may also be tied to each other if desired.

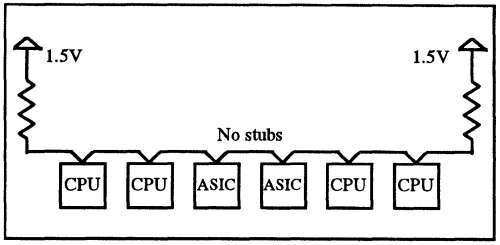


Figure 8. GTL+ Bus Topology

The GTL+ bus depends on incident wave switching. Therefore timing calculations for GTL+ signals are based on *flight time* as opposed to capacitive deratings. Analog signal simulation of the Pentium Pro processor bus including trace lengths is highly recommended when designing a system with a heavily loaded GTL+ bus. See Intel's world wide web page (http:\\www.intel.com) to down-load the buffer models for the Pentium Pro processor in IBIS format.

3.2. Power Management: Stop Grant and Auto HALT

The Pentium Pro processor allows the use of Stop Grant and Auto HALT modes to immediately reduce the power consumed by the device. When enabled, these cause the clock to be stopped to most of the CPU's internal units and thus significantly reduces power consumption by the CPU as a whole.

Stop Grant is entered by asserting the STPCLK# pin of the Pentium Pro processor. When STPCLK# is recognized by the Pentium Pro processor, it will stop execution and will not service interrupts. It will continue snooping the bus. Stop Grant power is specified assuming no snoop hits occur.

Auto HALT is a low power state entered when the Pentium Pro processor executes a halt (HLT) instruction. In this state the Pentium Pro processor behaves as if it executed a halt instruction, and it additionally powers-down most internal units. In Auto HALT, the Pentium Pro processor will recognize all interrupts and snoops. Auto HALT power is specified assuming no snoop hits or interrupts occur.

The low power standby mode of Stop Grant or Auto HALT can be defined by a configuration bit to be either the lowest power achievable by the Pentium Pro processor (Stop Grant power), or a power state in which the clock distribution is left running (Idle power). "Low power standby" *disabled* leaves the core logic running, while "Low power standby" *enabled* allows the Pentium Pro processor to enter its lowest power mode.

3.3. Power and Ground Pins

As future versions of the Pentium Pro processor are released, the operating voltage of the CPU die and of the L2 Cache die may differ from the 150 Mhz Pentium Pro processor. There are two groups of power inputs on the Pentium Pro processor package to support the possible voltage difference between the two die in the package, and one 5V pin to support a fan for the OverDrive processor. There are also 4 pins defined on the package for voltage identification. These pins specify the voltage required by the CPU die. These have been added to cleanly support voltage specification variations on the Pentium Pro processor and future processors. See Section 3.6. for an explanation of the voltage identification (VID) pins.

Future mainstream devices will fall into two groups. Either the CPU die and the L2 Cache die will both run at the same voltage ($V_{CC}P$), or the L2 Cache die will use $V_{CC}S$ (3.3V) while the CPU die runs at another voltage on $V_{CC}P$. When the L2 Cache die is running on the same supply as the CPU die, the $V_{CC}S$ pins will consume no current. To properly support this, the system should distribute 3.3V and a selectable voltage to the Pentium Pro processor socket. Selection may be provided for by socketed regulation or by using the voltage identification pins. Note that it is possible that $V_{CC}P$ and $V_{CC}S$ are both nominally 3.3V. It should not be assumed that these will be able to use the same power supply.

For clean on-chip power distribution, the Pentium Pro processor has 76 V_{CC} (power) and 101 V_{SS} (ground) inputs. The 76 V_{CC} pins are further divided to provide the different voltage levels to the device. $V_{CC}P$ inputs for the CPU die and some L2 die account for 47 of the V_{CC} pins, while 28 $V_{CC}S$ inputs (3.3V) are for use by the on-package L2 Cache die of some processors. One $V_{CC}5$ pin is provided for use by the fan of the OverDrive processor. $V_{CC}5$, $V_{CC}S$ and $V_{CC}P$ must remain electrically separated from each other. On the circuit board, *all* $V_{CC}P$ pins must be connected to a voltage island and *all* $V_{CC}S$ pins must be connected to a separate voltage island (an island is a portion of a power plane that has been divided, or an entire plane). Similarly, *all* V_{SS} pins must be connected to a system ground plane. See **Figure 44** for the locations of power and ground pins

3.4. Decoupling Recommendations

Due to the large number of transistors and high internal clock speeds, the Pentium Pro processor can create large, short duration transient (switching) current surges that occur on internal clock edges which can cause power planes to spike above and below their nominal value if not properly controlled. The Pentium Pro processor is also capable of generating large average current swings between low and full power states, called *Load-Change Transients*, which can cause power planes to sag below their nominal value if bulk decoupling is not adequate. See Figure 9 for an example of these current fluctuations. Care must be taken in the board design to guarantee that the voltage provided to the Pentium Pro processor remains within the specifications listed in this volume. Failure to do so may result in timing violations and/or a reduced lifetime of the component

Adequate decoupling capacitance should be placed near the power pins of the Pentium Pro processor. Low inductance capacitors such as the 1206 package surface mount capacitors are recommended for the best high frequency electrical performance. Forty (40) 1µF 1206-style capacitors with a ±22% tolerance make a good starting point for simulations as this is our recommended decoupling when using a standard Pentium Pro processor Voltage Regulator Module. Inductance should be reduced by connecting capacitors directly to the $V_{CC}P$ and V_{SS} planes with minimal trace length between the component pads and vias to the plane. Be sure to include the effects of board inductance within the simulation. Also, when choosing the capacitors to use, bear in mind the operating temperatures they will see and the tolerance that they are rated at. Type Y5S or better are recommended (±22% tolerance over the temperature range -30°C to +85°C).

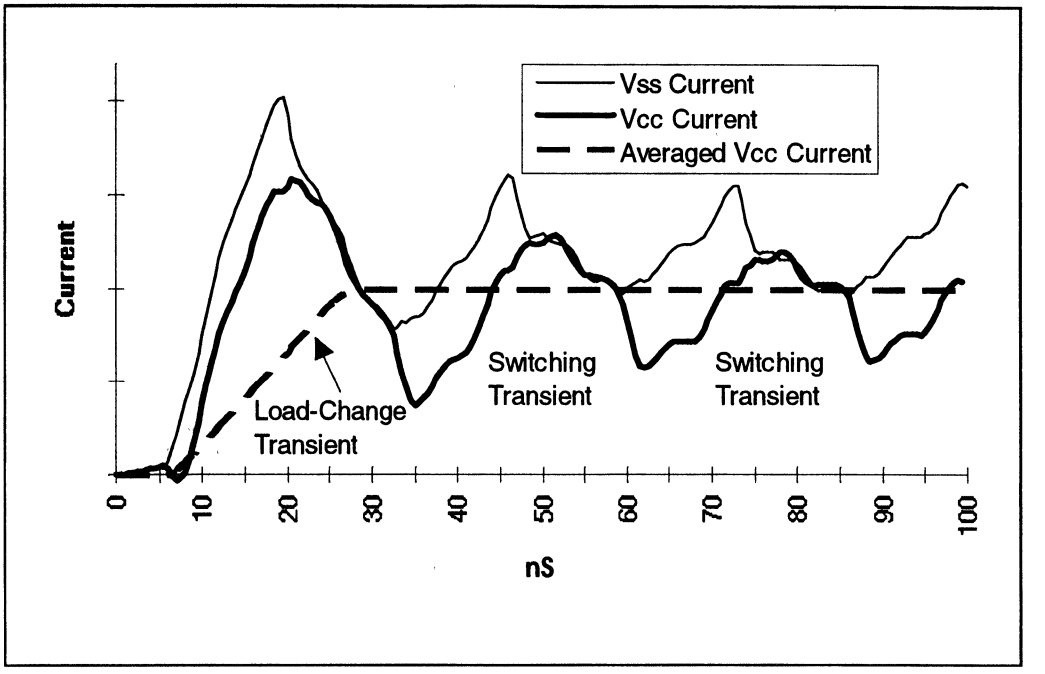


Figure 9. Transient Types

Bulk capacitance with a low Effective Series Resistance (ESR) should also be placed near the Pentium Pro processor in order to handle changes in average current between the low power and normal operating states. About 4000uF of capacitance with an ESR of 5mΩ makes a good starting point for simulations, although more capacitance may be needed to bring the ESR down to this level due to the current technology in the industry. The standard Pentium Pro processor Voltage Regulator Modules already contain this bulk capacitance. Be sure to determine what is available on the market before choosing parameters for the models. Also, include power supply response time and cable inductance in a full simulation.

See *AP-523 Pentium® Pro Processor Power Distribution Guidelines* Application Note (Order Number 242764) for power modeling for the Pentium Pro processor.

3.4.1. $V_{CC}S$ DECOUPLING

Decoupling of ten (10) 1μF ceramic capacitors (type Y5S or better) and a minimum of five 22μF tantalum capacitors is recommended for the $V_{CC}S$ pins. This is to handle the transients that will occur in future devices.

3.4.2. GTL+ DECOUPLING

Although the Pentium Pro GTL+ processor bus receives power external to the Pentium Pro processor, it should be noted that this power supply will also require the same diligent decoupling methodologies as the processor. Notice that the existence of external power entering through the I/O buffers causes V_{SS} current to be higher than the V_{CC} current as evidenced in Figure 9.

3.4.3. PHASE LOCK LOOP (PLL) DECOUPLING

Isolated analog decoupling is required for the internal PLL. This should be equivalent to 0.1μF of ceramic capacitance. The capacitor should be type Y5R or better and should be across the PLL1 and PLL2 pins of the Pentium Pro processor. ("Y5R" implies ±15% tolerance over the temperature range -30°C to +85°C.)

3.5. BCLK Clock Input Guidelines

The BCLK input directly controls the operating speed of the GTL+ bus interface. All GTL+ external timing parameters are specified with respect to the rising edge of the BCLK input. Clock multiplying within the processor is provided by an internal Phase Lock Loop (PLL) which requires a constant frequency BCLK input. Therefore the BCLK frequency cannot be changed dynamically. It can however be changed when RESET# is active assuming that all reset specifications are met for the clock and the configuration signals.

The Pentium Pro processor core frequency must be configured during reset by using the A20M#, IGNNE#, LINT1/NMI, and LINT0/INTR pins. The value on these pins during RESET#, and until two clocks beyond the end of the RESET# pulse, determines the multiplier that the PLL will use for the internal core clock. See the Appendix A for the definition of these pins during reset. At all other times their functionality is defined as the compatibility signals that the pins are named after. These signals have been made 3.3V tolerant so that they may be driven by existing logic devices. This was important for both functions of the pins.

Supplying a bus clock multiplier this way is required in order to increase processor performance without changing the processor design, and to maintain the bus frequency such that system boards can be designed to function properly as CPU frequencies increase.

3.5.1. SETTING THE CORE CLOCK TO BUS CLOCK RATIO

Table 42 lists the configuration pins and the values that must be driven at reset time in order to set the core clock to bus clock ratio. Figure 10 shows the timing relationship required for the clock ratio signals with respect to RESET# and BCLK. CRESET# from an 82453GX (or 82453KX) is shown since its timing is useful for controlling the multiplexing function that is required for sharing the pins.

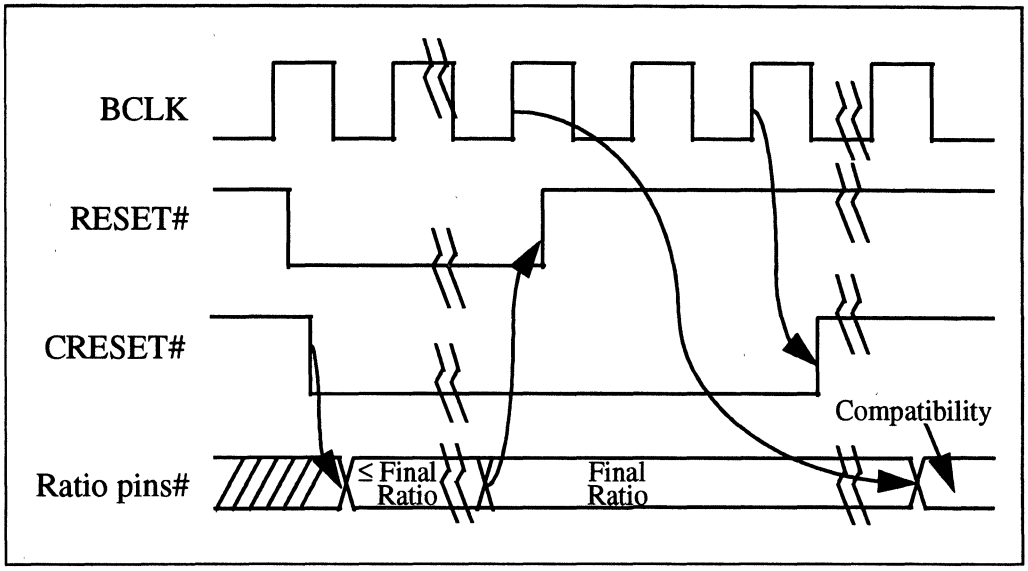


Figure 10. Timing Diagram of Clock Ratio Signals

Using CRESET# (CMOS reset), the circuit in Figure 11 can be used to share the pins. The pins of the processors are bussed together to allow any one of them to be the compatibility processor. The component used as the multiplexer must not have outputs that drive higher than 3.3V in order to meet the Pentium Pro processor's 3.3V tolerant buffer specifications. The multiplexer output current should be limited to 200mA max, in case the VCCP supply to the processor ever fails.

The pull-down resistors between the multiplexer and the processor (1KΩ) force a ratio of 2x into the processor in the event that the Pentium Pro processor powers up before the multiplexer and/or the chipset. This prevents the processor from ever seeing a ratio higher than the final ratio.

If the multiplexer were powered by $V_{CC}P$, CRESET# would still be unknown until the 3.3V supply came up to power the CRESET# driver. A pull-down can be used on CRESET# instead of the four between the multiplexer and the Pentium Pro processor in this case. In this case, the multiplexer must be designed such that the compatibility inputs are truly ignored as their state is unknown.

In any case, the compatibility inputs to the multiplexer must meet the input specifications of the multiplexer. This may require a level translation before the multiplexer inputs unless the inputs and the signals driving them are already compatible.

For FRC mode processors, one multiplexer will be needed per FRC pair, and the multiplexer will need to be clocked using BCLK to meet setup and hold times to the processors. This may require the use of high speed programmable logic.

3.5.2. MIXING PROCESSORS OF DIFFERENT FREQUENCIES

Mixing components of different internal clock frequencies is not fully supported and has not been validated by Intel. One should also note when attempting to mix processors rated at different frequencies in a multi-processor system that a *common* bus clock frequency and a set of multipliers must be found that is acceptable to all processors in the system. Of course, a processor may be run at a core frequency as low as its minimum rating. Operating system support for multi-processing with mixed frequency components should also be considered.

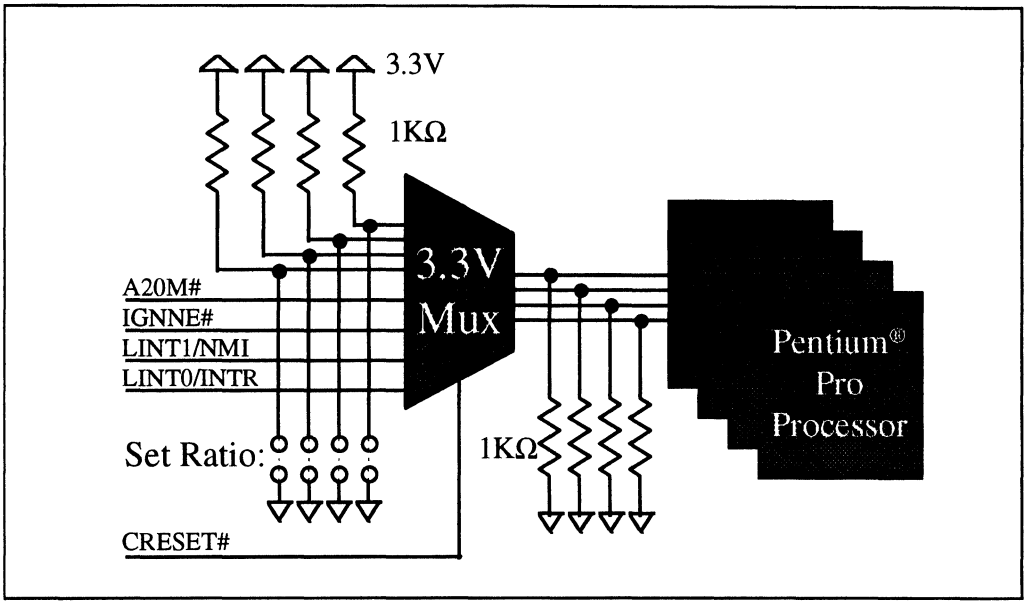


Figure 11. Example Schematic for Clock Ratio Pin Sharing

Note that in order to support different frequency multipliers to each processor, the design shown above would require four multiplexers.

3.6. Voltage Identification

There are four Voltage Identification Pins on the Pentium Pro processor package. These pins can be used to support automatic selection of power supply voltage. These pins are not *signals* but are each either an open circuit in the package or a short circuit to V_{SS}.

The opens and shorts define the voltage required by the processor. This has been added to cleanly support voltage specification variations on future Pentium Pro processors. These pins are named VID0 through VID3 and the definition of these pins is shown in Table 1. A '1' in this table refers to an open pin and '0' refers to a short to ground. **The $V_{CC}P$ power supply should supply the voltage that is requested *or* disable itself.**

Table 1. Voltage Identification Definition [1,2]

VID[3:0]	Voltage Setting	VID[3:0]	Voltage Setting
0000	3.5	1000	2.7
0001	3.4	1001	2.6
0010	3.3	1010	2.5
0011	3.2	1011	2.4
0100	3.1	1100	2.3
0101	3.0	1101	2.2
0110	2.9	1110	2.1
0111	2.8	1111	No CPU Present

NOTES:

1. Nominal setting requiring regulation to ±5% at the Pentium® Pro processor $V_{CC}P$ pins under all conditions. Support not expected for 2.1V-2.32V.
2. 1= Open circuit; 0= Short to V_{SS}

Support for a wider range of VID settings will benefit the system in meeting the power requirements of future Pentium Pro processors. Note that the '1111'

(or all opens) ID can be used to detect the absence of a processor in a given socket as long as the power supply used does not affect these lines.

To use these pins, they may need to be pulled up by an external resistor to another power source. The power source chosen should be one that is guaranteed to be stable whenever the supply to the voltage regulator is stable. This will prevent the possibility of the Pentium Pro processor supply running up to 3.5V in the event of a failure in the supply for the VID lines. Note that the specification for the standard Pentium Pro processor Voltage Regulator Modules allows the use of these signals either as TTL compatible levels or as opens and shorts. Using them as TTL compatible levels will require the use of pull-up resistors to 5V if the input voltage to the regulator is 5V and the use of a voltage divider if the input voltage to the regulator is 12V. The resistors chosen should not cause the current through a VID pin to exceed its specification in Table 3. There must not be any other components on these signals if the VRM uses them as opens and shorts.

3.7. JTAG Connection

The debug port described in the *Pentium® Pro Processor Developer's Manual, Volume 1: Specifications* (Order Number 242690) should be at the start and end of the JTAG chain with TDI to the first component coming from the Debug Port and TDO from the last component going to the Debug Port. The recommended pull-up value for Pentium Pro processor TDO pins is 240Ω.

Due to the voltage levels supported by the Pentium Pro processor JTAG logic, it is recommended that the Pentium Pro processors be first in the JTAG chain and followed by any other 3.3V logic level components within the system. A translation buffer should be used to connect to the rest of the chain unless a 5V component can be used next that is capable of accepting a 3.3V input. Similar considerations must be made for TCK, TMS and TRST#. Components may need these signals buffered to match required logic levels.

In a multi-processor system, be cautious when including empty Pentium Pro Processor sockets in the scan chain. All sockets in the scan chain must have a processor installed to complete the chain or the system must support a method to bypass the empty sockets.

See the *Pentium® Pro Processor Developer's Manual, Volume 1: Specifications* (Order Number 242690) for full information on putting a debug port in the JTAG chain.

3.8. Signal Groups

In order to simplify the following discussion, signals have been combined into groups by buffer type. **All outputs are open drain** and require an external hi-level source provided externally by the termination or a pull-up resistor.

GTL+ input signals have differential input buffers which use V_{REF} as their reference signal. GTL+ output signals require termination to 1.5V. Later in this document, the term "GTL+ Input" refers to the GTL+ input group as well as the GTL+ I/O group when receiving. Similarly, "GTL+ Output" refers to the GTL+ output group as well as the GTL+ I/O group when driving.

The 3.3V tolerant, Clock, APIC and JTAG inputs can each be driven from ground to 3.3V. The 3.3V tolerant, APIC, and JTAG outputs can each be pulled high to as much as 3.3V. See **Table 7** for specifications.

The groups and the signals contained within each group are shown in Table 2. Note that the signals ASZ[1:0]#, ATTR[7:0]#, BE[7:0]#, DEN#, DID[7:0]#, DSZ[1:0]#, EXF[4:0]#, LEN[1:0]#, SMMEM#, and SPLCK# are all GTL+ signals that are shared onto another pin. Therefore they do not appear in this table.

3.8.1. ASYNCHRONOUS VS. SYNCHRONOUS

All GTL+ signals are synchronous. All of the 3.3V tolerant signals can be applied asynchronously, except when running two processors in FRC mode. To run in FRC mode, synchronization logic is required on all signals, (except PWRGOOD) going to both processors. Also note the timing requirements for PICCLK with respect to BCLK. With FRC enabled, PICCLK must be ¼X BCLK and synchronized with respect to BCLK. PICCLK must always lag BCLK by at least 1 ns and no more than 5 ns (PRELIMINARY VALUES).

Table 2. Signal Groups

Group Name	Signals
GTL+ Input	BPRI#, BR[3:1]#[1], DEFER#, RESET#, RS[2:0]#, RSP#, TRDY#
GTL+ Output	PRDY#
GTL+ I/O	A[35:3]#, ADS#, AERR#, AP[1:0]#, BERR#, BINIT#, BNR#, BP[3:2]#, BPM[1:0]#, BR0#, D[63:0]#, DBSY#, DEP[7:0]#, DRDY#, FRCERR, HIT#, HITM#, LOCK#, REQ[4:0]#, RP#
3.3V Tolerant Input	A20M#, FLUSH#, IGNNE#, INIT#, LINT0/INTR, LINT1/NMI, PREQ#, PWRGOOD[2], SMI#, STPCLK#
3.3V Tolerant Output	FERR#, IERR#, THERMTRIP#[3]
Clock[4]	BCLK
APIC Clock[4]	PICCLK
APIC I/O[4]	PICD[1:0]
JTAG Input[4]	TCK, TDI, TMS, TRST#
JTAG Output[4]	TDO
Power/Other[5]	CPUPRES#, PLL1, PLL2, TESTHI, TESTLO, UP#, $V_{CC}P$, $V_{CC}S$, $V_{CC}5$, VID[3:0], V_{REF}[7:0], V_{SS}

NOTES:

1. The BR0# pin is the only BREQ# signal that is bi-directional. The internal BREQ# signals are mapped onto BR# pins after the agent ID is determined.
2. See PWRGOOD in Section 3.9.
3. See THERMTRIP# in Section 3.10.
4. These signals are tolerant to 3.3V. Use a 150Ω pull-up resistor on PICD[1:0] and 240Ω on TDO.
5. CPUPRES# is a ground pin defined to allow a designer to detect the presence of a processor in a socket. (Preliminary)
   PLL1 and PLL2 are for decoupling the internal PLL (See Section 3.4.3.).
   TESTHI pins should be tied to VCCP. A 10K pull-up may be used.
   TESTLO pins should be tied to V_{SS}. A 1K pull-down may be used.
   UP# is an open in the Pentium® Pro processor and tied to V_{SS} in the OverDrive® processor See Section 8.3.2 for usage.
   $V_{CC}P$ is the primary power supply.
   $V_{CC}S$ is the secondary power supply used by some versions of the second level cache.
   $V_{CC}5$ is unused by Pentium Pro processor and is used by the OverDrive processor for fan/heatsink power. See Section 8.
   V_{ID}[3:0] lines are described in Section 3.6.
   V_{REF} [7:0] are the reference voltage pins for the GTL+ buffers.
   V_{SS} is ground.

3.9. PWRGOOD

PWRGOOD requires a special explanation. PWRGOOD is a 3.3V tolerant input. It is expected that this signal will be a *clean* indication that clocks and the system 3.3V, 5V and $V_{CC}P$ supplies are stable and within their specifications. Clean implies that the signal will remain low, (capable of sinking leakage current) without glitches, from the time that the power supplies are turned on until they come within specification. The signal will then transition monotonically to a high (3.3V) state. Figure 12 illustrates the relationship of PWRGOOD to other system signals. PWRGOOD can be driven inactive at any time, but power and clocks must again be stable before the rising edge of PWRGOOD. It must also meet the minimum pulse width specification in Table 12 and be followed by a 1mS RESET# pulse.

This signal must be supplied to the Pentium Pro processor as it is used to protect internal circuits

against voltage sequencing issues. Use of this signal is recommended for added reliability.

This signal does not need to be synchronized for FRC operation.

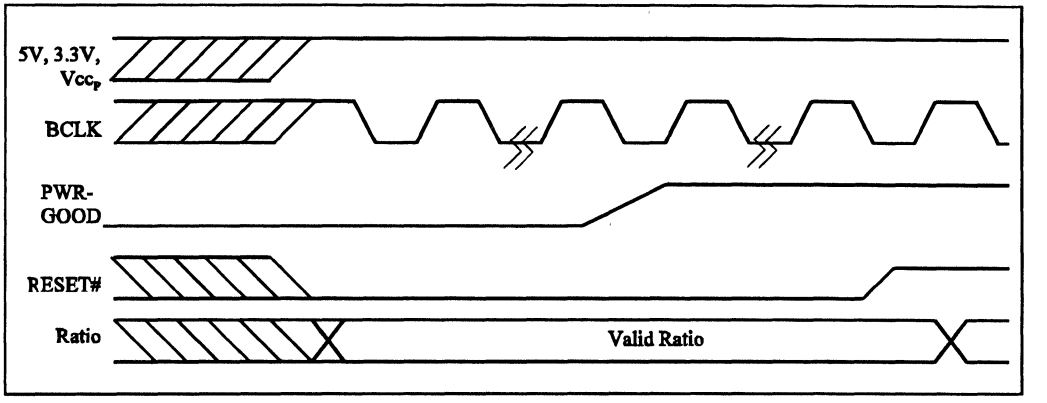


Figure 12. PWRGOOD Relationship at Power-On

3.10. THERMTRIP#

The Pentium Pro processor protects itself from catastrophic overheating by use of an internal thermal sensor. This sensor is set well above the normal operating temperature to ensure that there are no false trips. The processor will stop all execution when the junction temperature exceeds ~135° C. This is signaled to the system by the THERMTRIP# pin. Once activated, the signal remains latched, and the processor stopped, until RESET# goes active. There is no hysteresis built into the thermal sensor itself, so as long as the die temperature drops below the trip level, a RESET# pulse will reset the processor and execution will continue. If the temperature has not dropped beyond the trip level, the processor will continue to drive THERMTRIP# and remain stopped.

3.11. Unused Pins

All RESERVED pins must remain unconnected. All pins named TESTHI must be pulled up, no higher than VCCP, and may be tied directly to $V_{CC}P$. All pins named TESTLO must be pulled low and may be tied directly to V_{SS}.

PICCLK must be driven with a clock input, and the PICD[1:0] lines must each be pulled-up to 3.3V with a separate 150Ω resistor, even when the APIC will not be used.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused GTL+ inputs should be connected to V_{TT}. Unused active low 3.3V tolerant inputs should be connected to 3.3V with a 150Ω resistor and unused active high inputs should be connected to ground (V_{SS}). A resistor must also be used when tying bi-directional signals to power or ground. When tying any signal to power or ground, a resistor will also allow for fully testing the processor after board assembly.

For unused pins, it is suggested that ~10KΩ resistors be used for pull-ups (except for PICD[1:0] discussed above), and ~1KΩ resistors be used as pull-downs. **Never tie a pin directly to a supply other than the processor's own $V_{CC}P$ supply or to V_{SS}.**

3.12. Maximum Ratings

Table 3 contains Pentium Pro stress ratings only. Functional operation at the absolute maximum and minimum is not implied nor guaranteed. The Pentium Pro processor should not receive a clock while subjected to these conditions. Functional operating conditions are given in the AC and DC tables. Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium Pro processor contains protective circuitry to resist damage from static electric

discharge, one should always take precautions to avoid high static voltages or electric fields.

Table 3. Absolute Maximum Ratings[1]

Symbol	Parameter	Min	Max	Unit	Notes
$T_{Storage}$	Storage Temperature	-65	150	°C	
T_{Bias}	Case Temperature under Bias	-65	110	°C	
$V_{CC}P$(Abs)	Primary Supply Voltage with respect to V_{SS}	-0.5	Operating Voltage + 1.4	V	2
$V_{CC}S$(Abs)	3.3V Supply Voltage with respect to V_{SS}	-0.5	4.6	V	
$V_{CC}P$-$V_{CC}S$	Primary Supply Voltage with respect to Secondary Supply	-3.7	Operating Voltage + 0.4	V	2
V_{IN}	GTL+ Buffer DC Input Voltage with respect to V_{SS}	-0.5	$V_{CC}P$+ 0.5 but Not to exceed 4.3	V	3
V_{IN3}	3.3V Tolerant Buffer DC Input Voltage with respect to V_{SS}	-0.5	$V_{CC}P$+ 0.9 but Not to exceed 4.7	V	4
I_I	Max input current		200	mA	5
I_{VID}	Max VID pin current		5	mA	

NOTES:

1. Functional operation at the absolute maximum and minimum is not implied or guaranteed.
2. Operating voltage is the voltage that the component is designed to operate at. See Table 4.
3. Parameter applies to the GTL+ signal groups only.
4. Parameter applies to 3.3V tolerant, APIC, and JTAG signal groups only.
5. Current may flow through the buffer ESD diodes when VIH > VCCP+1.1V, as in a power supply fault condition or while power supplies are sequencing. Thermal stress should be minimized by cycling power off if the $V_{CC}P$ supply fails.

3.13. DC Specifications

Table 4 through Table 7 list the DC specifications associated with the Pentium Pro processor. Specifications are valid only while meeting the processor specifications for case temperature, clock frequency and input voltages. **Care should be taken to read all notes associated with each parameter.** See Section 3.3. for an explanation of voltage plans for Pentium Pro processors. See Section 8.4.1.1. for OverDrive processor information and Section 3.16 for flexible motherboard recommendations.

The DC specifications for the $V_{CC}P$ and $V_{CC}S$ supplies are listed in Table 4 and Table 5.

Most of the signals on the Pentium Pro processor are in the GTL+ signal group. These signals are specified to be terminated to 1.5V. The DC specifications for these signals are listed in Table 6. Care should be taken to read all notes associated with each parameter.

To allow compatibility with other devices, some of the signals are 3.3V tolerant and can therefore be terminated or driven to 3.3V. The DC specifications for these 3.3V tolerant inputs are listed in Table 7. Care should be taken to read all notes associated with each parameter.

Table 4. Voltage Specification

Symbol	Parameter	Min	Typ	Max	Unit	Notes
$V_{CC}P$	Primary V_{CC}	2.945	3.1	3.255	V	3.1 ± 5%, 1
$V_{CC}S$	Secondary V_{CC}	3.135	3.3	3.465	V	3.3 ± 5%
$V_{CC}5$	5V Supply	4.75	5.0	5.25	V	5.0 ± 5%

NOTES:

1. To comply with these guidelines and the industry standard voltage regulator module specifications, the equivalent of forty (40) 1µF±22% capacitors in 1206 packages should be placed near the power pins of the processor. (At least 40µF of capacitance should exist on the power plane with less than 35pH of inductance and 8µΩ of resistance between it and the pins of the processor).

Table 5. Power Specifications [1]

Symbol	Parameter	Min	Typ	Max	Unit	Notes
P_{Max}	150 MHz Thermal Design Power		23.2	29.2	W	2
I_{SGntP}	150 MHz $V_{CC}P$ Stop Grant Current	0.3		1.0	A	3, 4
I_{SGntS}	$V_{CC}S$ Stop Grant Current	0		0	A	
$I_{CC}P$	$V_{CC}P$ Current			9.9	A	4, Tested at max $V_{CC}P$
$I_{CC}S$	$V_{CC}S$ Current			0	A	5
T_C	Operating Case Temperature	0		85	°C	

NOTES:

1. All power measurements taken with CMOS inputs driven to $V_{CC}P$ and to 0V.
2. Maximum values measured at typical V_{CC}. Typical values not tested.
3. Same as Auto HALT current. Max values measured at typical V_{CC}. Minimum values are guaranteed by design/characterization at minimum V_{CC}.
4. All CMOS pins are driven with $V_{IH} = V_{CC}P$ and $V_{IL} = 0V$ during the execution of all I_{CC} and I_{CC}-stopgrant/autohalt tests.
5. The L2 of the current processor will draw no current from the $V_{CC}S$ inputs. $I_{CC}S$ is 0A when the L2 die receives its power from the $V_{CC}P$ pins. See the recommended decoupling in Section 3.4.

Table 6. GTL+ Signal Groups DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL}	Input Low Voltage	-0.3	V_{REF} –0.2	V	1, See Table 8
V_{IH}	Input High Voltage	V_{REF} +0.2	$V_{CC}P$	V	1
V_{OL}	Output Low Voltage	0.30	0.60	V	2
V_{OH}	Output High Voltage	—	—	V	See V_{TT} max in Table 8
I_{OL}	Output Low Current	36	48	mA	2
I_L	Leakage Current		±100	mA	3
I_{REF}	Reference Voltage Current		± 15	mA	4
C_{GTL+}	GTL+ Pin Capacitance		8.5	pF	5

NOTES:

1. V_{REF} worst case, not nominal. Noise on V_{REF} should be accounted for.
2. Parameter measured into a 25Ω resistor to 1.5V. Min. V_{OL} and max. I_{OL} are guaranteed by design/characterization.
3. ($0 \le Vpin \le V_{CC}P$).
4. Total current for all V_{REF} pins. Section 3.1. details the V_{REF} connections.
5. Total of I/O buffer, package parasitics and 0.5 pF for a socket. Capacitance values guaranteed by design for all GTL+ buffers.

Table 7. Non-GTL+[1] Signal Groups DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL}	Input Low Voltage	-0.3	0.8	V	
V_{IH}	Input High Voltage	2.0	3.6	V	
V_{OL}	Output Low Voltage		0.4 0.2	V V	2 3
V_{OH}	Output High Voltage	N/A	N/A	V	All Outputs are Open-Drain
I_L	Input Leakage Current		± 100	mA	4
C_{TOL}	3.3V Tol. Pin Capacitance		10	pF	Except BCLK, TCK, 5
C_{CLK}	BCLK Input Capacitance		9	pF	5
C_{TCK}	TCK Input Capacitance		8	pF	5

NOTES:
1. Table 7 applies to the 3.3V tolerant, APIC, and JTAG signal groups.
2. Parameter measured at 4 mA (for use with TTL inputs).
3. Parameter guaranteed by design at 100 µA(for use with CMOS inputs).
4. ($0 \leq Vpin \leq V_{CC}P$).
5. Total of I/O buffer, package parasitics and 0.5 pF for a socket. Capacitance values are guaranteed by design.

3.14. GTL+ Bus Specifications

The GTL+ bus must be routed in a daisy-chain fashion with termination resistors matching the printed circuit board impedance at each end of every signal trace. These termination resistors are placed between the ends of the signal trace and the V_{TT} voltage supply. The valid high and low levels are determined by the input buffers using a reference voltage called V_{REF}. Table 8 lists the nominal specifications for the GTL+ termination voltage (V_{TT}) and the GTL+ reference voltage (V_{REF}). It is important that the printed circuit board impedance be specified and held to a ±20% tolerance, and that the intrinsic trace capacitance for the GTL+ signal group traces is known. **For more details on GTL+, see Section 4.**

Table 8. GTL+ Bus Voltage Specifications

Symbol	Parameter	Min	Typical	Max	Units	Notes
V_{TT}	Bus Termination Voltage	1.35	1.5	1.65	V	±10%
V_{REF}	Input Reference Voltage	2/3 V_{TT} –2%	2/3 V_{TT}	2/3 V_{TT} +2%	V	±2%, 1

NOTES:
1. V_{REF} should be created from V_{TT} by a voltage divider of 1% resistors.

3.15. AC Specifications

Table 9 through Table 15 list the AC specifications associated with the Pentium Pro processor. Timing Diagrams begin with Figure 14. The AC specifications are broken into categories. Table 9 contains the clock specifications, Table 11 contains the GTL+ specifications, Table 12 is the 3.3V

tolerant Signal group specifications, Table 13 contains timings for the reset conditions, Table 14 covers APIC bus timing, and Table 15 covers Boundary Scan timing.

All AC specifications for the GTL+ signal group are relative to the rising edge of the BCLK input. All GTL+ timings are referenced to V_{REF} for both '0' and '1' logic levels unless otherwise specified.

Care should be taken to read all notes associated with a particular timing parameter.

Table 9. Bus Clock AC Specifications

T#	Parameter	Min	Max	Unit	Figure	Notes
	Core Frequency	100.00	150	MHz		1
	Bus Frequency	50.00	66.67	MHz		1
T1:	BCLK Period	15	20	ns	Figure 14	
T2:	BCLK Period Stability		300	ps		2, 3
T3:	BCLK High Time	4		ns	Figure 14	@>2.0V, 2
T4:	BCLK Low Time	4		ns	Figure 14	@<0.8V, 2
T5:	BCLK Rise Time	0.3	1.5	ns	Figure 14	(0.8V - 2.0V), 2
T6:	BCLK Fall Time	0.3	1.5	ns	Figure 14	(2.0V- 0.8V),2

NOTES:

1. The internal core clock frequency is derived from the bus clock. A clock ratio must be driven into the Pentium® Pro processor on the signals LINT[1:0], A20M# and IGNNE# at reset. See the descriptions for these signals in Appendix A.
2. Not 100% tested. Guaranteed by design/characterization.
3. Measured on rising edge of adjacent BCLKs at 1.5V.
   The jitter present must be accounted for as a component of BCLK skew between devices.
   Clock jitter is measured from one rising edge of the clock signal to the next rising edge at 1.5V. To remain within the clock jitter specifications, all clock periods must be within 300ps of the ideal clock period for a given frequency. For example, a 66.67 MHz clock with a nominal period of 15ns, must not have any single clock period that is greater than 15.3 ns or less than 14.7 ns.

Table 10. GTL+ Signal Groups AC Specifications

RL = 25Ω terminated to 1.5V, V_{REF} = 1.0V					
T# Parameter	**Min**	**Max**	**Unit**	**Figure**	**Notes**
T7A: GTL+ Output Valid Delay H→L	0.55	4.4	ns	Figure 15	t_{PHL}, 1
T7B: GTL+ Output Valid Delay L→H	0.55	3.9	ns	Figure 15	t_{PLH}, 1
T8: GTL+ Input Setup Time	2.2		ns	Figure 16	2, 3, 4
T9: GTL+ Input Hold Time	0.45		ns	Figure 16	4
T10: RESET# Pulse Width	1		ms	Figure 18 Figure 19	5

NOTES:
1. Valid delay timings for these signals are specified into an idealized 25Ω resistor to 1.5V with V_{REF} at 1.0V. Minimum values guaranteed by design.
2. A minimum of 3 clocks must be guaranteed between 2 active-to-inactive transitions of TRDY#.
3. RESET# can be asserted (active) asynchronously, but must be deasserted synchronously.
4. Specification takes into account a 0.3V/ns edge rate and the allowable V_{REF} variation. Guaranteed by design.
5. After V_{CC}, V_{TT}, V_{REF}, BCLK and the clock ratio become stable.

Table 11. GTL+ Signal Groups Ringback Tolerance

Parameter	Min	Unit	Figure	Notes
α: Overshoot	100	mV	Figure 17	1
τ: Minimum Time at High	1.5	ns	Figure 17	1
ρ: Amplitude of Ringback	-100	mV	Figure 17	1
δ: Duration of Squarewave Ringback	N/A	ns	Figure 17	1
ϕ: Final Settling Voltage	100	mV	Figure 17	1

NOTES:
1. Specified for an edge rate of 0.3-0.8V/ns. See Section 4.1.3.1 for the definition of these terms. See Figure 24 and Figure 25 for the generic waveforms. All values determined by design/characterization.

Table 12. 3.3V Tolerant Signal Groups AC Specifications

T#	Parameter	Min	Max	Unit	Figure	Notes
T11:	3.3V Tolerant Output Valid Delay	1	8	ns	Figure 15	1
T12:	3.3V Tolerant Input Setup Time	5		ns	Figure 16	2, 3, 4, 5
T13:	3.3V Tolerant Input Hold Time	1.5		ns	Figure 16	
T14:	3.3V Tolerant Input Pulse Width, except PWRGOOD	2		BCLKs	Figure 15	Active and Inactive states
T15:	PWRGOOD Inactive Pulse Width	10		BCLKs	Figure 15 Figure 19	6

NOTES:
1. Valid delay timings for these signals are specified into 150Ω to 3.3V. See Figure 13 for a capacitive derating curve.
2. These inputs may be driven asynchronously. However, to guarantee recognition on a specific clock, the setup and hold times with respect to BCLK must be met.
3. These signals must be driven synchronously in FRC mode.
4. A20M#, IGNNE#, INIT# and FLUSH# can be asynchronous inputs, but to guarantee recognition of these signals following a synchronizing instruction such as an I/O write instruction, they must be valid with active RS[2:0]# signals of the corresponding synchronizing bus transaction.
5. INTR and NMI are only valid in APIC disable mode. LINT[1:0]# are only valid in APIC enabled mode.
6. When driven inactive, or after Power, V_{REF}, BCLK, and the ratio signals are stable.

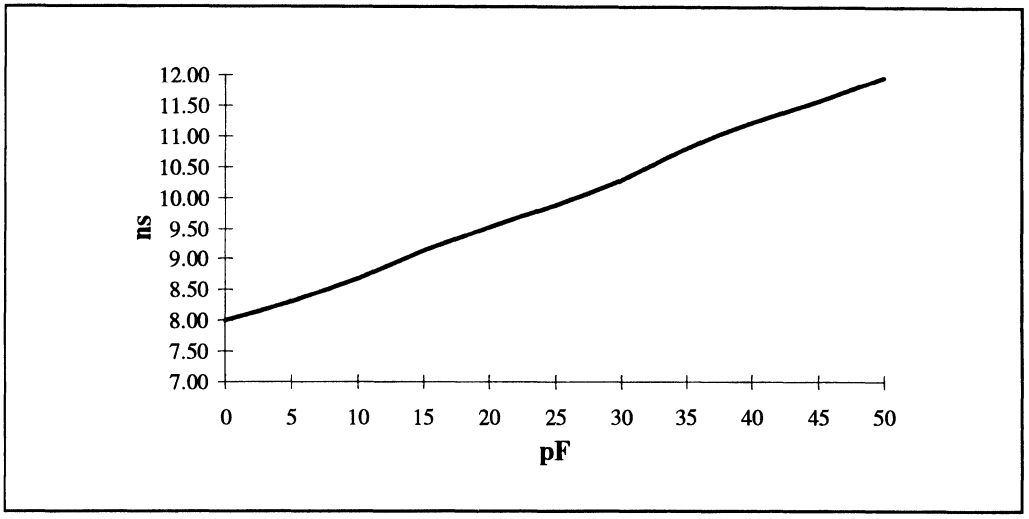


Figure 13. 3.3V Tolerant Group Derating Curve

Table 13. Reset Conditions AC Specifications

T#	Parameter	Min	Max	Unit	Figure	Notes
T16:	Reset Configuration Signals (A[14:5]#, BR0#, FLUSH#, INIT#) Setup Time	4		BCLKs	Figure 18	Before deassertion of RESET#
T17:	Reset Configuration Signals (A[14:5]#, BR0#, FLUSH#, INIT#) Hold Time	2	20	BCLKs	Figure 18	After clock that deasserts RESET#
T18:	Reset Configuration Signals (A20M#, IGNNE#, LINT[1:0]#) Setup Time	1		ms	Figure 18	Before deassertion of RESET#
T19:	Reset Configuration Signals (A20M#, IGNNE#, LINT[1:0]#) Delay Time		5	BCLKs	Figure 18	After assertion of RESET# 1
T20:	Reset Configuration Signals (A20M#, IGNNE#, LINT[1:0]#) Hold Time	2	20	BCLKs	Figure 18 Figure 19	After clock that deasserts RESET#

NOTES:

1. For a reset, the clock ratio defined by these signals must be a safe value (their final or lower multiplier) within this delay unless PWRGOOD is being driven inactive.

Table 14. APIC Clock and APIC I/O AC Specifications

T#	Parameter	Min	Max	Unit	Figure	Notes
T21:	PICCLK Frequency	2	33.3	MHz		1
T22:	PICCLK Period	30	500	ns	Figure 14	
T23:	PICCLK High Time	12		ns	Figure 14	
T24:	PICCLK Low Time	12		ns	Figure 14	
T25:	PICCLK Rise Time	1	5	ns	Figure 14	
T26:	PICCLK Fall Time	1	5	ns	Figure 14	
T27:	PICD[1:0] Setup Time	8		ns	Figure 16	2
T28:	PICD[1:0] Hold Time	2		ns	Figure 16	2
T29:	PICD[1:0] Valid Delay	2.1	10	ns	Figure 15	2, 3, 4

NOTES:

1. With FRC enabled PICCLK must be ¼X BCLK and synchronized with respect to BCLK. PICCLK must always lag BCLK by at least 1 ns and no more than 5 ns (PRELIMINARY VALUES).
2. Referenced to PICCLK Rising Edge.
3. For open drain signals, Valid Delay is synonymous with Float Delay.
4. Valid delay timings for these signals are specified into 150Ω to 3.3V.

Table 15. Boundary Scan Interface AC Specifications

T#	Parameter	Min	Max	Unit	Figure	Notes
T30:	TCK Frequency	—	16	MHz		
T31:	TCK Period	62.5	—	ns	Figure 14	
T32:	TCK High Time	25		ns	Figure 14	@2.0V, 1
T33:	TCK Low Time	25		ns	Figure 14	@0.8V, 1
T34:	TCK Rise Time		5	ns	Figure 14	(0.8V-2.0V), 1, 2
T35:	TCK Fall Time		5	ns	Figure 14	(2.0V-0.8V), 1, 2
T36:	TRST# Pulse Width	40		ns	Figure 21	1, Asynchronous
T37:	TDI, TMS Setup Time	5		ns	Figure 20	3
T38:	TDI, TMS Hold Time	14		ns	Figure 20	3
T39:	TDO Valid Delay	1	10	ns	Figure 20	4, 5
T40:	TDO Float Delay		25	ns	Figure 20	1, 4, 5
T41:	All Non-Test Outputs Valid Delay	2	25	ns	Figure 20	4, 6, 7
T42:	All Non-Test Outputs Float Delay		25	ns	Figure 20	1, 4, 6, 7

T43: All Non-Test Inputs Setup Time	5		ns	Figure 20	3, 6, 7
T44: All Non-Test Inputs Hold Time	13		ns	Figure 20	3, 6, 7

NOTES:

1. Not 100% tested. Guaranteed by design/characterization.
2. 1ns can be added to the maximum TCK rise and fall times for every 1 MHz below 16 MHz.
3. Referenced to TCK rising edge.
4. Referenced to TCK falling edge.
5. Valid delay timing for this signal is specified into 150Ω terminated to 3.3V.
6. Non-Test Outputs and Inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO and TMS). These timings correspond to the response of these signals due to boundary scan operations.
7. During Debug Port operation, use the normal specified timings rather than the boundary scan timings.

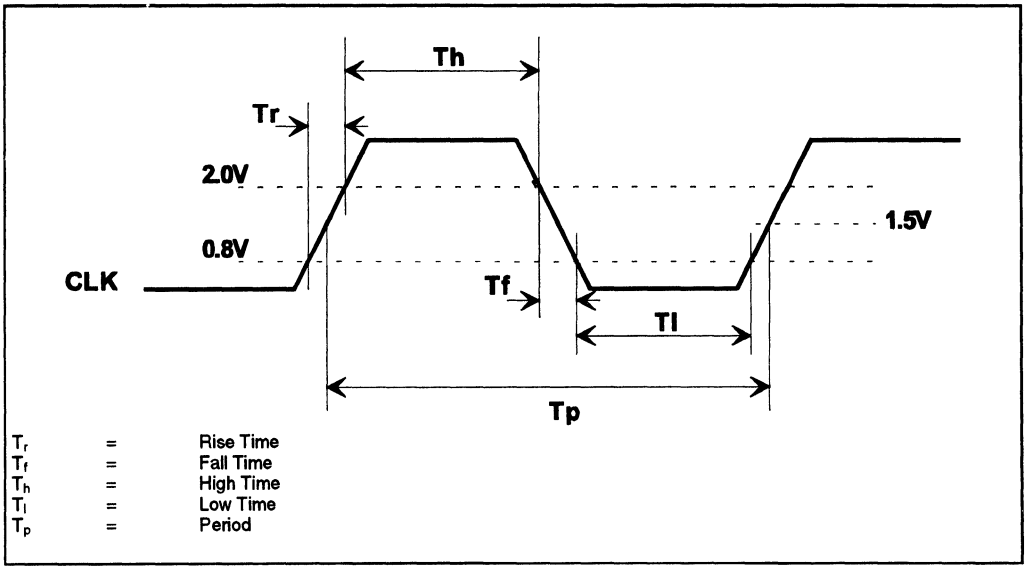
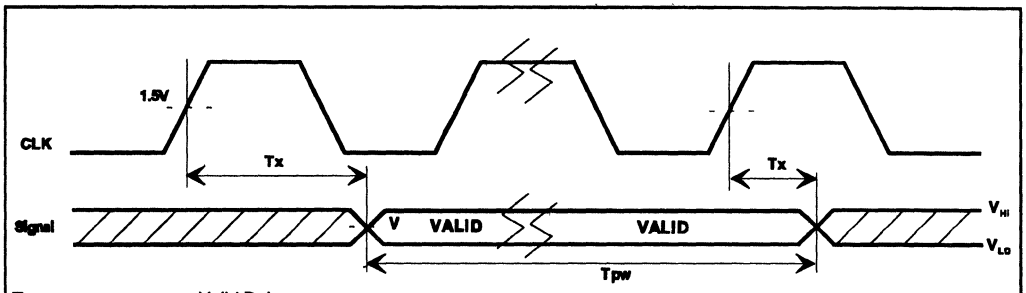
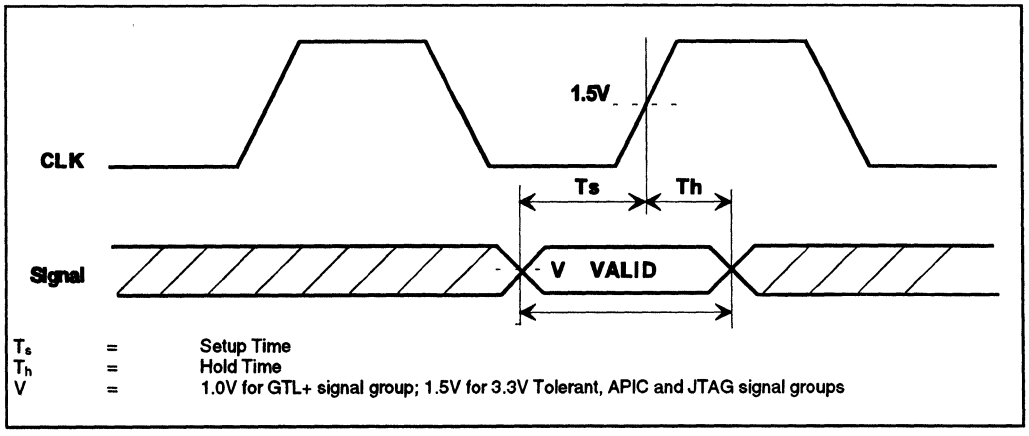


Figure 14. Generic Clock Waveform



Tx	=	Valid Delay
Tpw	=	Pulse Width
V	=	1.0V for GTL+ signal group; 1.5V for 3.3V Tolerant, APIC, and JTAG signal groups
V_{HI}	=	GTL+ signals must achieve a DC high level of at least 1.2V
V_{LO}	=	GTL+ signals must achieve a DC low level of at most 0.8V

Figure 15. Valid Delay Timings



T_s	=	Setup Time
T_h	=	Hold Time
V	=	1.0V for GTL+ signal group; 1.5V for 3.3V Tolerant, APIC and JTAG signal groups

Figure 16. Setup and Hold Timings

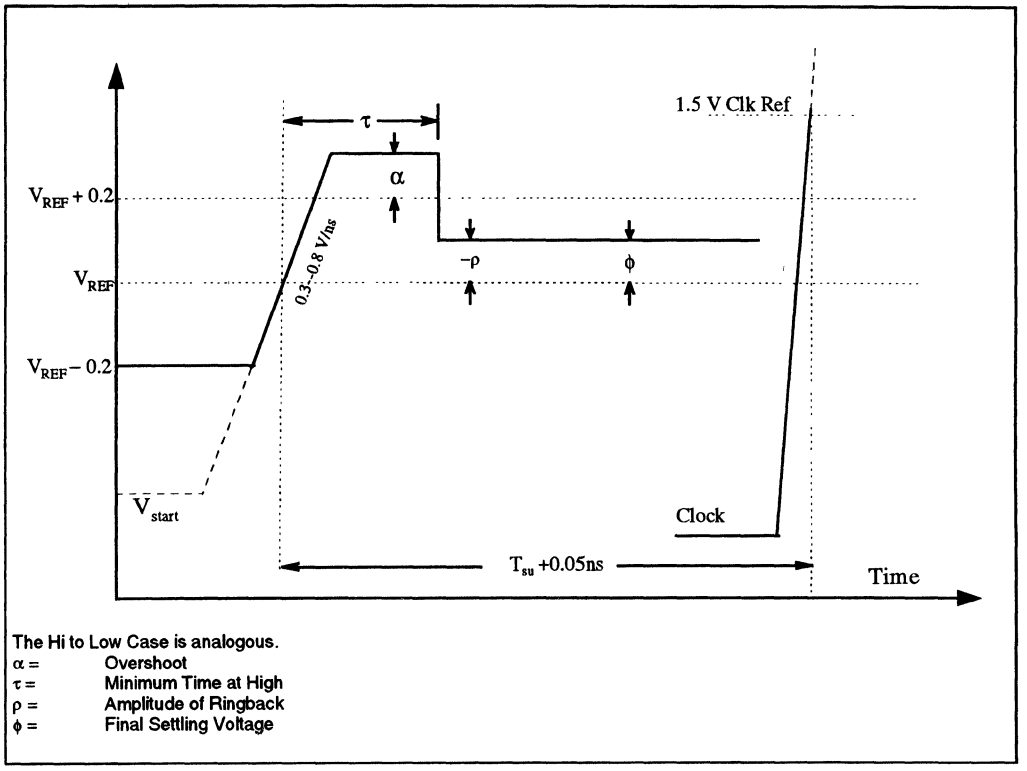
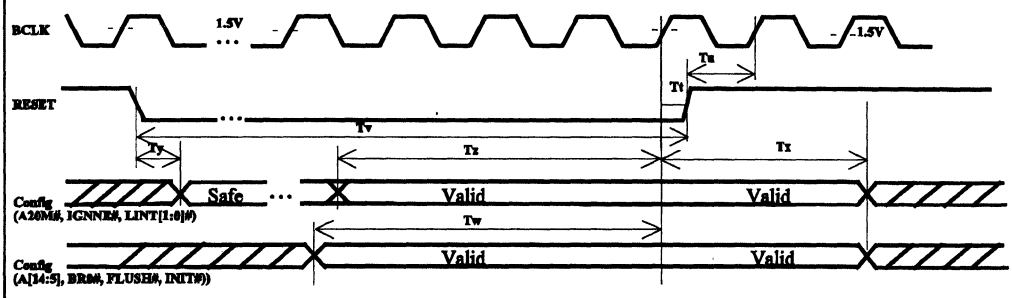
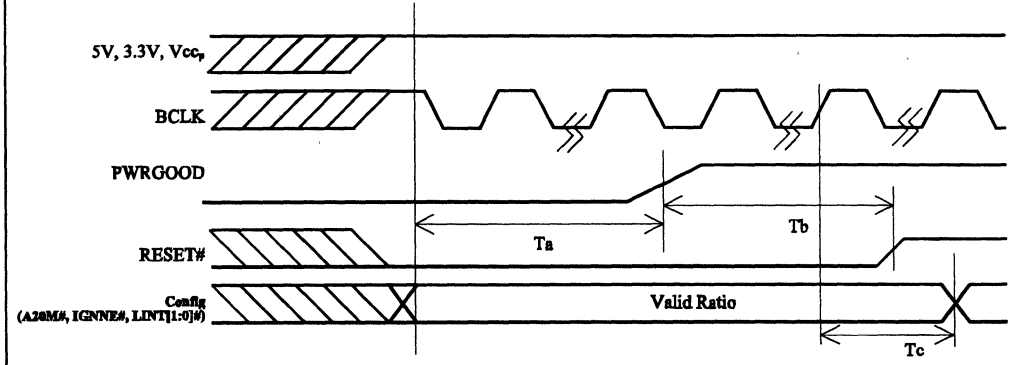


Figure 17. Lo to HI GTL+ Receiver Ringback Tolerance.



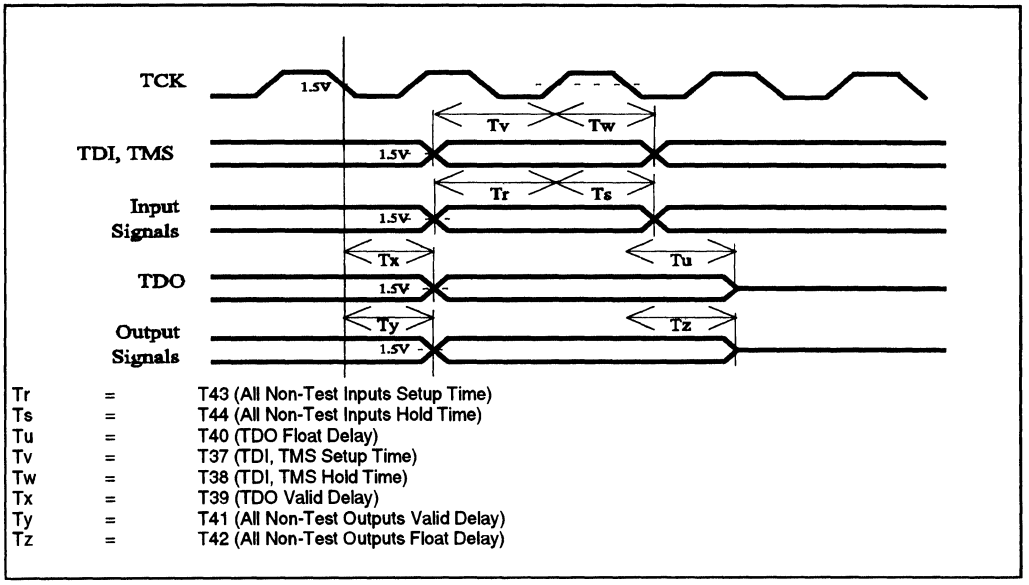
Tt	=	T9 (GTL+ Input Hold Time)
Tu	=	T8 (GTL+ Input Setup Time)
Tv	=	T10 (RESET# Pulse Width)
Tw	=	T16 (Reset Configuration Signals (A[14:5]#, BR0#, FLUSH#, INIT#) Setup Time)
Tx	=	T17 (Reset Configuration Signals (A[14:5]#, BR0#, FLUSH#, INIT#) Hold Time). T20 (Reset Configuration Signals (A20M#, IGNNE#, LINT[1:0]#) Hold Time)
Ty	=	T19 (Reset Configuration Signals (A20M#, IGNNE#, LINT[1:0]#) Delay Time)
Tz	=	T18 (Reset Configuration Signals (A20M#, IGNNE#, LINT[1:0]#) Setup Time)

Figure 18. Reset and Configuration Timings



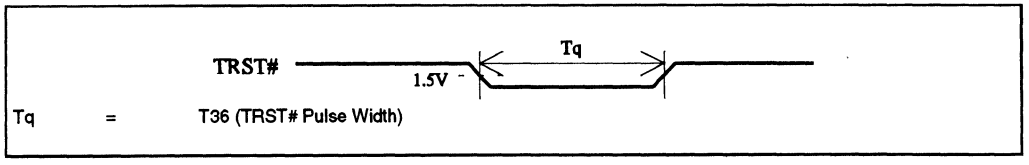
Ta	=	T15 (PWRGOOD Inactive Pulse Width)
Tb	=	T10 (RESET# Pulse Width)
Tc	=	T20 (Reset Configuration Signals (A20M#, IGNNE#, LINT[1:0]#) Hold Time)

Figure 19. Power-On Reset and Configuration Timings



Tr	=	T43 (All Non-Test Inputs Setup Time)
Ts	=	T44 (All Non-Test Inputs Hold Time)
Tu	=	T40 (TDO Float Delay)
Tv	=	T37 (TDI, TMS Setup Time)
Tw	=	T38 (TDI, TMS Hold Time)
Tx	=	T39 (TDO Valid Delay)
Ty	=	T41 (All Non-Test Outputs Valid Delay)
Tz	=	T42 (All Non-Test Outputs Float Delay)

Figure 20. Test Timings (Boundary Scan)



Tq = T36 (TRST# Pulse Width)

Figure 21. Test Reset Timings

3.16. Flexible Motherboard Recommendations

Table 16 provides recommendations for designing a "flexible" motherboard for supporting future Pentium Pro processors. By meeting these recommendations, the same system design should be able to support future standard Pentium Pro processors. If the voltage regulator module is socketed using Header 8, a smaller range of support is required by the voltage regulator module. See Section 8. for information on Header 8. **These values are preliminary!**

The use of a zero-insertion force socket for the processor and the voltage regulator module is recommended. One should also make every attempt to leave margin in the system where possible.

Table 16. Flexible Motherboard (FMB) Power Recommendations[1]

Symbol	Parameter	Low End	High End	Unit	Notes
$V_{CC}P$	Full FMB Primary V_{CC} Socketed VRM Primary V_{CC}	2.4 3.1	3.5 3.5	V V	5% tolerance over range
$V_{CC}S$	FMB Secondary V_{CC}	3.3	3.3	V	5% tolerance
$V_{CC}5$	FMB 5V V_{CC}	5.0	5.0	V	5% tolerance
P_{Max}	FMB Thermal Design power		35	W	
I_{CCP}	Full FMB $V_{CC}P$ Current Socketed VRM $V_{CC}P$ Current	0.3 0.3	12.4 11.2	A A	
I_{CCS}	FMB $V_{CC}S$ Current	0	2.4	A	
I_{CC5}	FMB $V_{CC}5$ Current		340	mA	
C_P	High Frequency $V_{CC}P$ Decoupling		40	µF	40 1µF 1206 package
C_S	High Frequency $V_{CC}S$ Decoupling		10	µF	10 1µF 1206 package
T_C	FMB Operating Case Temperature		85	°C	

NOTE:

1. Values are preliminary, per processor, and are not tested parameters. They are solely recommendations.

4.0 GTL+ Interface Specification

This section defines the new open-drain bus called GTL+. The primary target audience is designers developing systems using GTL+ devices such as the Pentium Pro Processor and the 82450 PCIset. This specification will also be useful for I/O buffer designers developing an I/O cell and package to be used on a GTL+ bus.

This specification is an enhancement to the GTL (Gunning Transceiver Logic) specification. The enhancements were made to allow the interconnect of up to eight devices operating at 66.6 MHz and higher using manufacturing techniques that are standard in the microprocessor industry. The specification enhancements over standard GTL provide better noise margins and reduced ringing. Since this specification is different from the GTL specification, it is referred to as GTL+.

The GTL+ specification defines an open-drain bus with external pull-up resistors providing termination to a termination voltage (V_{TT}). The specification includes a maximum driver output low voltage (V_{OL}) value, output driver edge rate requirements, example AC timings, maximum bus agent loading (capacitance and package stub length), and a receiver threshold (V_{REF}) that is proportional to the termination voltage.

The specification is given in two parts. The first, is the system specification which describes the system environment. The second, is the actual I/O specification, which describes the AC and DC characteristics for an I/O transceiver.

Note that some of the critical distances, such as routing length, are given in electrical length (time) instead of physical length (distance). This is because the system design is dependent on the propagation time of the signal on a printed circuit board trace rather than just the length of the trace. Different PCB materials, package materials and system construction result in different signal propagation velocities. Therefore a given physical length does not correspond to a fixed electrical length. The distance (time) calculation up to the designer.

4.1. System Specification

Figure 22 shows a typical system that a GTL+ device would be placed into. The typical system is shown with two terminations and multiple transceiver agents connected to the bus. The receivers have differential inputs connected to a reference voltage, V_{REF}, which is generated externally by a voltage divider. Typically, one voltage divider exists at each component. Here one is shown for the entire network.

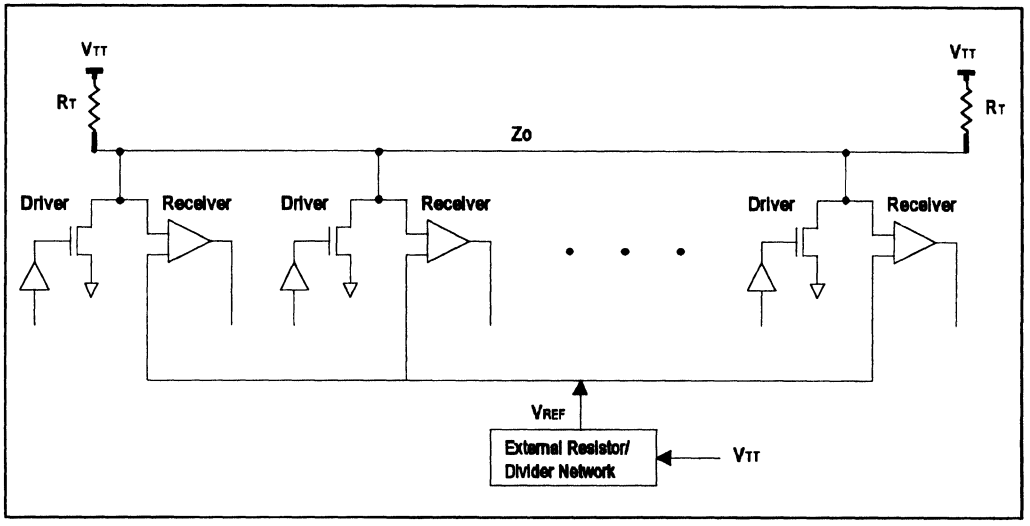


Figure 22. Example Terminated Bus with GTL+ Transceivers

4.1.1. System DC Parameters

The following system DC parameters apply to Figure 22.

Table 17. System DC Parameters

Symbol	Parameter	Value	Tolerance	Notes
V_{TT}	Termination Voltage	1.5V	±10%	
V_{REF}	Input Reference Voltage	2/3 V_{TT}	±2%	1
R_T	Termination Resistance	Z_{EFF} (nominal)	See Note	2, 4
Z_{EFF}	Effective (Loaded) Network Impedance	45-65Ω		2, 3

NOTES:

1. This ±2% tolerance is in addition to the ±10% tolerance of V_{TT}, and could be caused by such factors as voltage divider inaccuracy.
2. $$Z_{EFF} = \frac{Zo\ (nominal)}{(1+Cd/Co)^{1/2}}$$

   Zo = Nominal board impedance; recommended to be 65Ω ±10%. Zo is a function of the trace cross-section, the distance to the reference plane(s), the dielectric constant, ε_r, of the PCB material and the dielectric constant of the solder-mask/air for micro-strip traces.

   Co = Total intrinsic nominal trace capacitance between the first and last bus agents, excluding the termination resistor tails. Co is a function of Zo and εr. For Zo= 65Ω and εr = 4.3, Co is approximately 2.66 pF/in times the network length (first agent to last agent).

   Cd = Sum of the Capacitance of all devices and PCB stubs (if any) attached to the net,
   = PCB Stub Capacitance +Socket Capacitance +Package Stub Capacitance +Die Capacitance.

3. Z_{EFF} of all 8-load nets must remain between 45-65Ω under all conditions, including variations in Zo, Cd, temperature, VCC, etc.
4. To reduce cost, a system would usually employ one value of R_T for all its GTL+ nets, irrespective of the Z_{EFF} of individual nets. The designer may start with the average value of Z_{EFF} in the system. The value of R_T may be adjusted to balance the Hi-to-Lo and Lo-to-Hi noise margins. Increasing the value of R_T tends to slow the rising edge, increasing rising flight time, decreasing the Lo-to-Hi noise margin, and increasing the Hi-to-Lo noise margin by lowering V_{OL}. R_T can be decreased for the opposite effects.

   R_T affects GTL+ rising edge rates and the "apparent clock-to-out" time of a driver in a net as follows: A large R_T causes the standing current in the net to be low when the (open drain) driver is low (on). As the driver switches off, the small current is turned off, launching a relatively small positive-going wave down the net. After a few trips back and forth between the driver and the terminations (undergoing reflections at intervening agents in the meantime) the net voltage finally climbs to V_{TT}. Because the wave launched initially is relatively small in amplitude (than it would have been had R_T been smaller and the standing current larger), the overall rising edge climbs toward V_{TT} at a slower rate. Notice that this effect causes an increase in flight time, and has no influence on the true clock-to-out timing of the driver into the standard 25Ω test load.

4.1.2. Topological Guidelines

The board routing should use layout design rules consistent with high-speed digital design (i.e. minimize trace length and number of vias, minimize trace-to-trace coupling, maintain consistent impedance over the length of a net, maintain consistent impedance from one net to another, ensure sufficient power to ground plane bypassing, etc.). In addition, the signal routing should be done in a *Daisy Chain* topology (such as shown in Figure 8) without any significant stubs. Table 18 describes, more completely, some of these guidelines. Note that the critical distances are measured in electrical length (propagation time) instead of physical length.

Table 18. System Topological Guidelines

Parameter	Description
Maximum Trace Length	To meet a specific Clock cycle time, the maximum trace length between any two agents must be restricted. The flight time (defined later) must be less than or equal to the maximum amount of time which leaves enough time within one clock cycle for the remaining system parameters such as driver clock-out delay (T_{CO}), receiver setup time (T_{SU}), clock jitter and clock skew.
Maximum Stub Length	All signals should use a Daisy Chain routing (i.e. no stubs). It is acknowledged that the package of each device on the net imposes a stub, and that a practical layout using PQFP parts may require SHORT stubs, so a truly stubless network is impossible to achieve, but any stub on the network (including the device package) should be no greater than 250 ps in electrical length.

Table 18. System Topological Guidelines

Parameter	Description
Distributed Loads	Minimum spacing lengths are determined by hold time requirements and clock skew. Maintaining 3" ±30% inter-agent spacing minimizes the variation in noise margins between the various networks, and can provide a significant improvement for the networks. This is only a guideline.

4.1.3. System AC Parameters: Signal Quality

The system AC parameters fall into two categories, Signal Quality and Flight Time. Acceptable signal quality must be maintained over all operating conditions to ensure reliable operation. Signal Quality is defined by three parameters: Overshoot/ Undershoot, Settling Limit, and Ringback. These parameters are illustrated in Figure 23 and are described in Table 19.

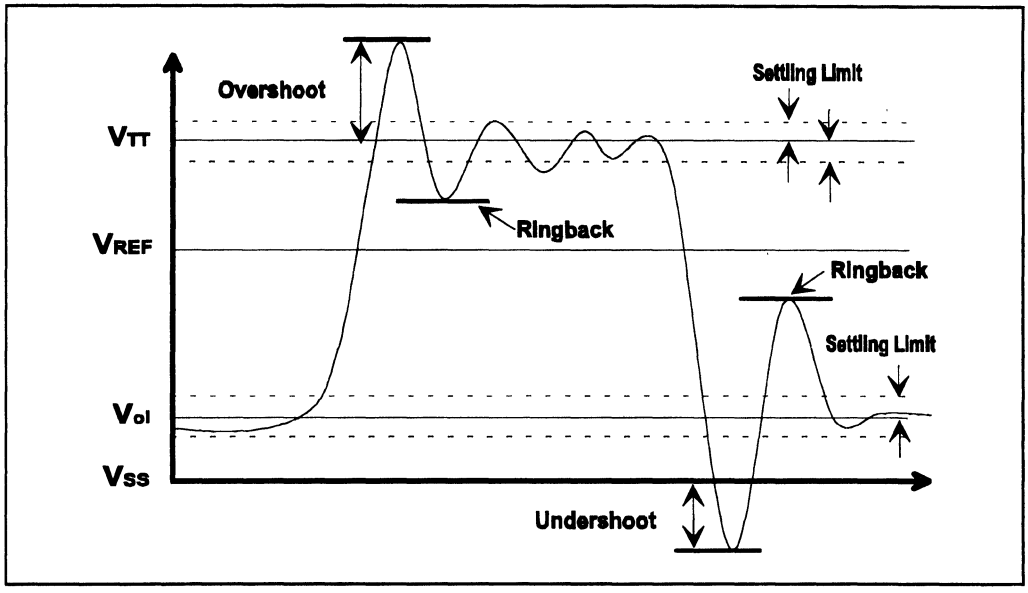


Figure 23. Receiver Waveform Showing Signal Quality Parameters

Table 19. Specifications for Signal Quality

Parameter	Description	Specification
Maximum Signal Overshoot/Undershoot	Maximum Absolute voltage a signal extends above V_{TT} or below V_{SS} (when protection diodes are not present).	0.3V (guideline)

Table 19. Specifications for Signal Quality

Parameter	Description	Specification
Settling Limit	The maximum amount of ringing, at the receiving chip pad, a signal must be limited to before its next transition. This signal should be within 10% of the signal swing to its final value, when either in its high state or low state.	±10% of (V_{OH}-V_{OL}) (guideline)
Maximum Signal Ringback (Nominal)	The maximum amount of ringing allowed for a signal at a receiving chips setup and hold time window before the next clock. This value is dependent upon the specific receiver design. (Normally ringing within the setup and hold windows must not come within 200 mV of V_{REF} although specific devices may allow more ringing and loosen this specification. See Section 4.1.3.1 for more details.)	V_{REF} ±200 mV

The overshoot/undershoot guideline is provided to limit signals transitioning beyond V_{CC} or V_{SS} due to fast signal edge rates. Violating the overshoot/undershoot guideline is acceptable, but since excessive ringback is the harmful effect associated with overshoot/undershoot it will make satisfying the ringback specification very difficult.

Violations of the Settling Limit guideline are acceptable if simulations of 5 to 10 successive transitions do not show the amplitude of the ringing increasing in the subsequent transitions. If a signal has not settled close to its final value before the next logic transition, then the timing delay to V_{REF} of the succeeding transition may vary slightly due to the stored reactive energy in the net inherited from the previous transition. This is akin to "eye" patterns in communication systems caused by inter-symbol interference. The resulting effect is a slight variation in flight time.

4.1.3.1. Ringback Tolerance

The nominal maximum ringback tolerated by GTL+ receivers is stated in Table 19, namely: no closer to V_{REF} than a ±200 mV overdrive zone. This requirement is usually necessary to guarantee that a receiver meets its specified minimum setup time (T_{SU}), since setup time usually degrades as the magnitude of overdrive beyond the switching threshold (V_{REF}) is reduced.

Exceptions to the nominal overdrive requirement can be made when it is known that a particular receiver's setup time (as specified by its manufacturer) is *relatively* insensitive (less than 0.05 ns impact) to well-controlled ringing into the overdrive zone or even to brief re-crossing of the switching threshold, V_{REF}. Such "ringback-tolerant" receivers give the system designer more design freedom, and, if not exploited, at least help maintain high system reliability.

To characterize ringback tolerance, employ the idealized Lo-to-Hi input signal shown in Figure 24. The corresponding waveform for a Hi-to-Lo transition is shown in Figure 25. The object of ringback characterization is to determine the range of values for the different parameters shown on the diagram, which would maintain receiver setup time and correct logic functionality.

These parameters are defined as follows:

τ is *the minimum time that the input must spend, after crossing V_{REF} at the High level, before it can ring back, having overshot $V_{IN\_HIGH\_MIN}$ by at least α, while ρ, δ, and ϕ (defined below) are at some preset values, all without increasing T_{SU} by more than 0.05 ns.* Analogously for Hi-to-Lo transitions.

It is expected that the larger the overshoot α, the smaller the amount of time, τ, needed to maintain setup time to within +0.05 ns of the nominal value. For a given value of α, it is likely that τ will be the longest for the slowest input edge rate of 0.3V/ns. Furthermore, there may be some dependence between τ and lower starting voltages than V_{REF} – 0.2V (for Lo-to-Hi transitions) for the reason described later in the Section on receiver characterization. Analogously for Hi-Lo transitions.

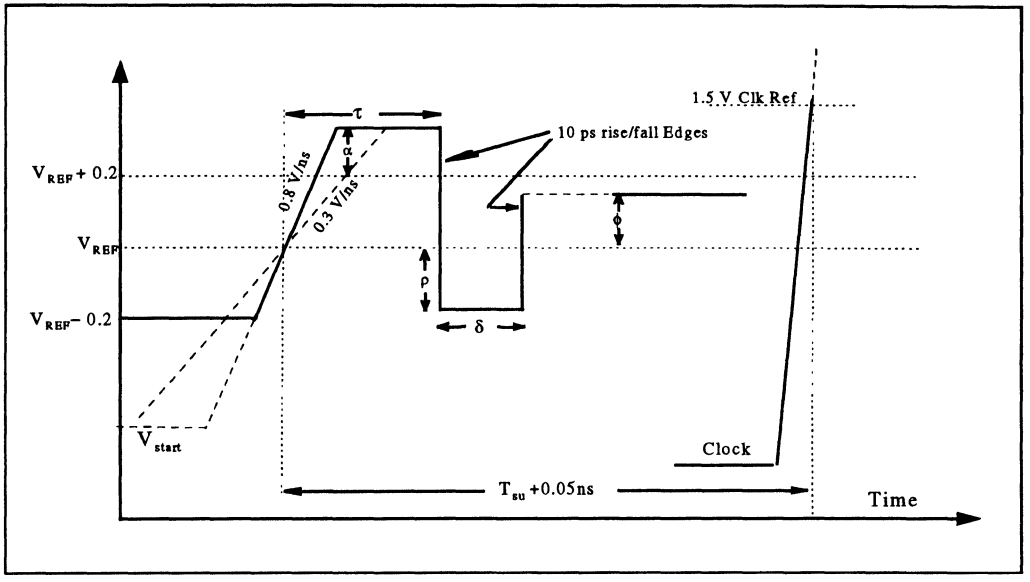


Figure 24. Standard Input Lo-to-HI Waveform for Characterizing Receiver Ringback Tolerance

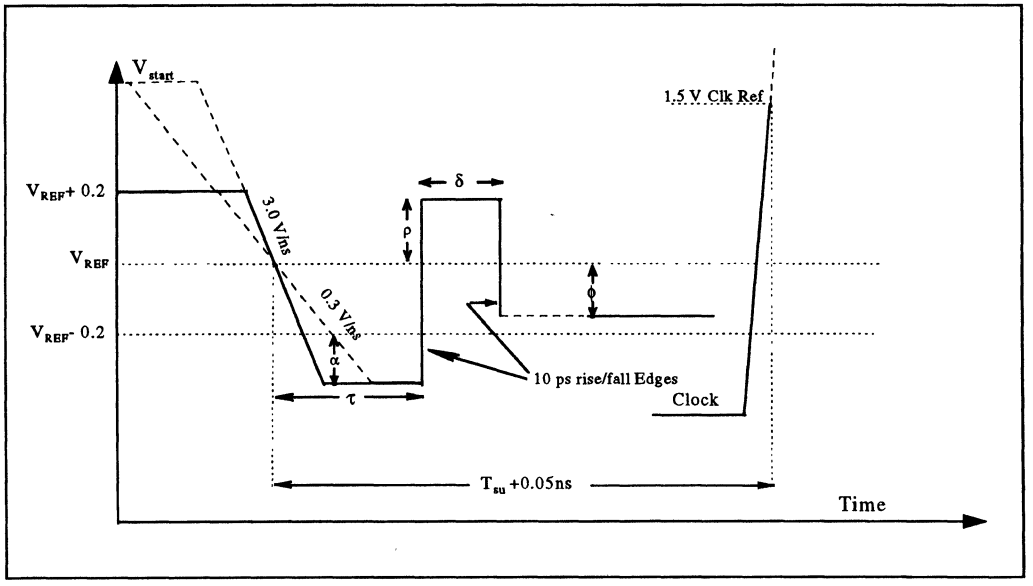


Figure 25. Standard Input HI-to-Lo Waveform for Characterizing Receiver Ringback Tolerance

ρ & δ are respectively, the amplitude and duration of square-wave ringback, below the threshold voltage (V_{REF}), that the receiver can tolerate without increasing T_{SU} by more than 0.05 ns for a given pair of (α, τ) values.

If, for any reason, the receiver cannot tolerate any ringback across the reference threshold (V_{REF}), then ρ would be a negative number, and δ may be infinite. Otherwise, expect an inverse (or near-inverse) relationship between ρ and δ, where the more the ringback, the shorter is the time that the ringback is allowed to last without causing the receiver to detect it.

φ is the final minimum settling voltage, relative to the reference threshold (V_{REF}), that the input should return to after ringback to guarantee a valid logic state at the internal flip-flop input.

φ is a function of the input amplifier gain, its differential mode offset, and its intrinsic maximum level of differential noise.

Specifying the values of α, τ, ρ, δ, and φ is the responsibility of the receiver vendor. The system designer should guarantee that all signals arriving at such a receiver remain in the permissible region specified by the vendor parameters as they correspond to those of the idealized square waves of Figure 24 and Figure 25. For instance, a signal with ringback inside the box delineated by ρ and δ can have a τ equal to or longer than the minimum, and an α equal to or larger than the minimum also.

A receiver that does not tolerate any ringback would show the following values for the above parameters:

$\alpha \geq 0V$, $\tau \geq T_{su}$, $\rho = -200$ mV, δ = undefined, φ = 200 mV.

A receiver which tolerates 50 mV of ringback would show the following values for the above parameters:

$\alpha \geq 0V$, τ = data sheet, $\rho = -150$ mV, δ = data sheet, φ ≥ tens of mV (data sheet).

Finally, a receiver which tolerates ringback across the switching threshold would show the following values for the above parameters:

$\alpha \geq 0$ V, τ = data sheet, $\rho \geq 0$ mV (data sheet), δ = data sheet, φ ≥ tens of mV.

where δ would usually be a brief amount of time, yielding a pulse (or "blip") beyond V_{REF}.

4.1.4. AC Parameters: Flight Time

Signal Propagation Delay is the time between when a signal appears at a driver pin and the time it arrives at a receiver pin. *Flight Time* is often used interchangeably with Signal Propagation Delay but it is actually quite different. Flight time is a term in the timing equation that includes the signal propagation delay, any effects the system has on the T_{CO} of the driver, plus any adjustments to the signal at the receiver needed to guarantee the T_{SU} of the receiver. More precisely, *Flight Time* is defined to be:

> The time difference between when a signal at the **Input pin** of a receiving agent (adjusted to meet the receiver manufacturer's conditions required for AC specifications) crosses V_{REF}, and the time that the **output pin** of the driving agent crosses V_{REF} **were it driving the test load** used by the manufacturer to specify that driver's AC timings.

An example of the simplest Flight Time measurement is shown in Figure 26. The receiver specification assumes that the signal maintains an edge rate greater than or equal to 0.3V/ns at the *receiver chip pad* in the overdrive region from V_{REF} to V_{REF} +200 mV for a rising edge and that there are no signal quality violations after the input crosses V_{REF} at the *pad*. The Flight Time measurement is similar for a simple Hi-to-Lo transition. Notice that timing is measured at the driver and receiver *pins* while signal integrity is observed at the receiver chip *pad*. When signal integrity at the pad violates the guidelines of this specification, and adjustments need to be made to flight time, the adjusted flight time obtained at the chip pad can be assumed to have been obtained at the package pin, usually with a small timing error penalty.

The 0.3V/ns edge rate will be addressed later in this document, since it is related to the conditions used to specify a GTL+ receiver's minimum setup time. What is meant by edge rate is neither instantaneous, nor strictly average. Rather, it can best be described for a rising edge -- by imagining an 0.3V/ns line crossing V_{REF} at the same moment that the signal crosses it, and extending to V_{REF} +200 mV, with the signal staying ahead (earlier in time) of that line at all times, until it reaches V_{REF} +200 mV. Such a requirement would always yield

signals with an average edge rate >0.3V/ns, but which could have instantaneous slopes that are lower or higher than 0.3V/ns, as long as they do not cause a crossing of the inclined line.

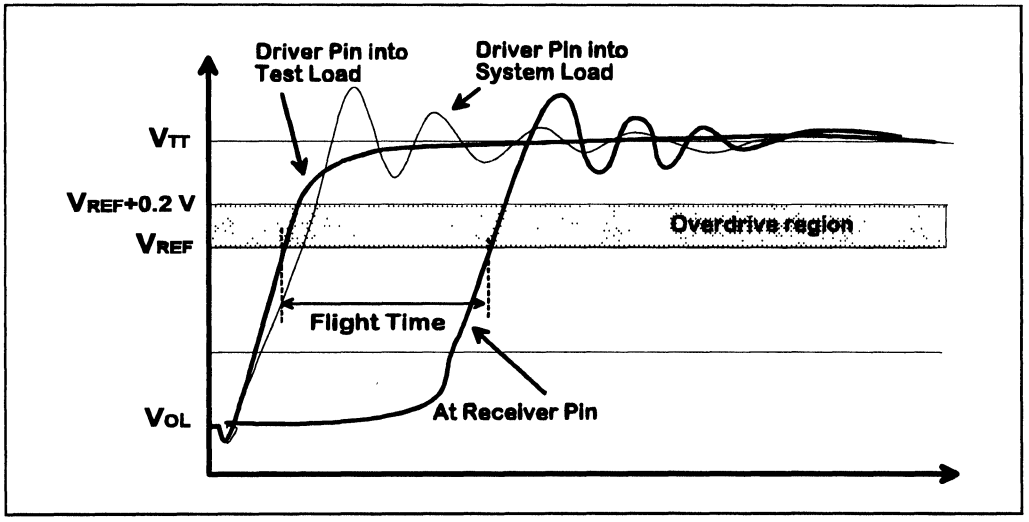


Figure 26. Measuring Nominal Flight Time

If either the rising or falling edge is slower than 0.3V/ns through the overdrive region beyond V_{REF}, (i.e., does not always stay ahead of an 0.3V/ns line), then the flight time for a rising edge is determined by extrapolating back from the signal crossing of V_{REF} +200 mV to V_{REF} using an 0.3V/ns slope as indicated in Figure 27.

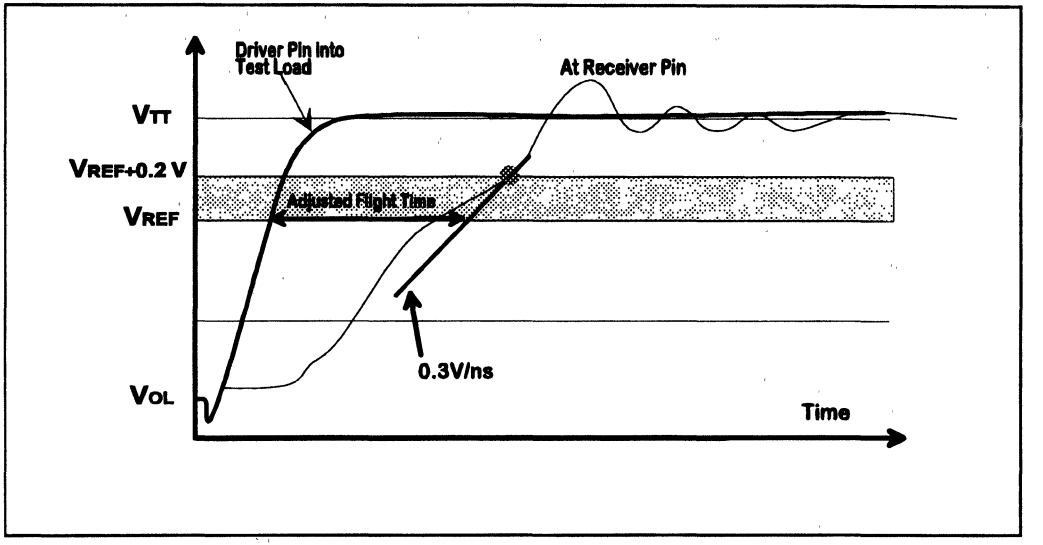


Figure 27. Flight Time of a Rising Edge Slower Than 0.3V/ns

If the signal is not monotonic while traversing the overdrive region (V_{REF} to V_{REF} +200 mV rising, or V_{REF} to V_{REF} – 200 mV falling), or rings back into the overdrive region after crossing V_{REF}, then flight time is determined by extrapolating back from the last crossing of V_{REF} ± 200 mV using a line with a slope of 0.8V/ns (the maximum allowed rising edge rate). This yields a new V_{REF} crossing point to be used for the flight time calculation. Figure 28 represents the situation where the signal is non-monotonic after crossing V_{REF} on the rising edge.

Figure 29 shows a falling edge that rings back into the overdrive region after crossing V_{REF}, and the 0.8V/ns line used to extrapolate flight time. Since strict adherence to the edge rate specification is not required for Hi-to-Lo transitions, and some drivers' falling edges are substantially faster than 0.8V/ns -- at both the fast and slow corners--, care should be taken when using the 0.8V/ns extrapolation. The extrapolation is invalid whenever it yields a V_{REF} crossing that occurs earlier than when the signal's actual edge crosses V_{REF}. In that case, flight time is defined to be the longer of: the time when the input at the receiver crosses V_{REF} initially, or when the line extrapolated (at 0.8V/ns) crosses V_{REF}. Figure 29 illustrates the situation where the extrapolated value would be used.

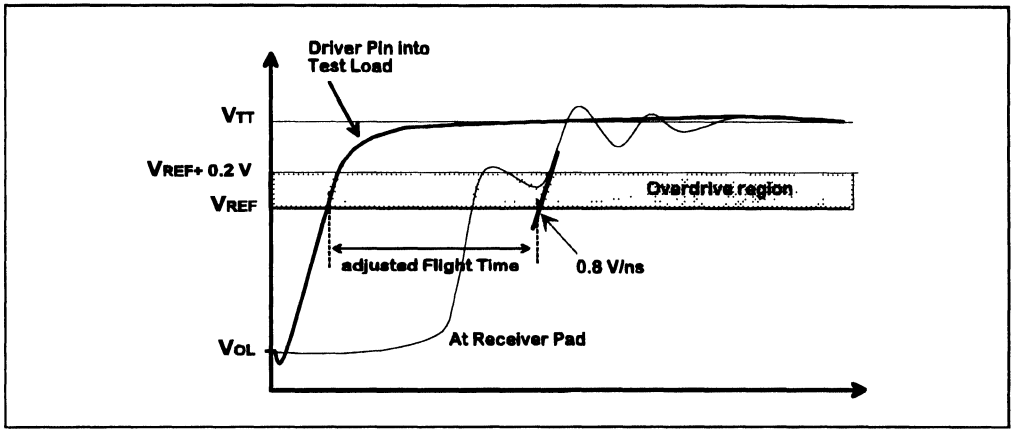


Figure 28. Extrapolated Flight Time of a Non-Monotonic Rising Edge

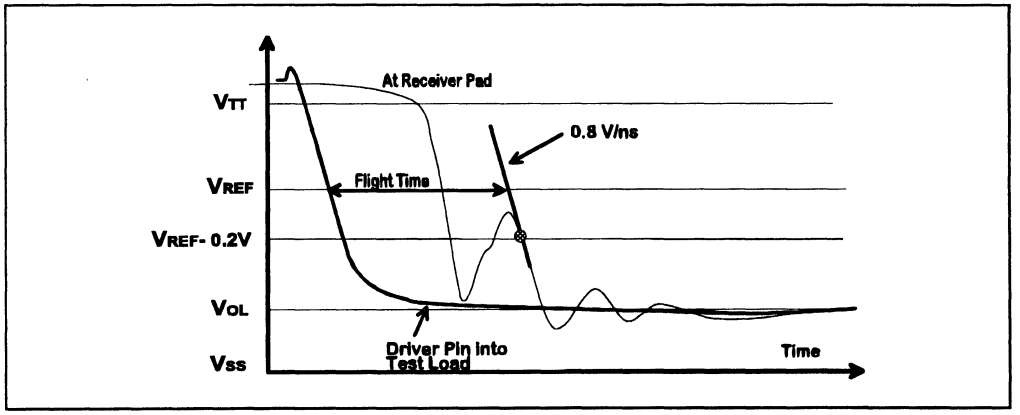


Figure 29. Extrapolated Flight Time of a Non-Monotonic Falling Edge

The maximum acceptable Flight Time is determined on a net-by-net basis, and is usually different for each unique driver-receiver pair. The maximum acceptable Flight Time can be calculated using the following equation (known as the setup time equation):

$$T_{FLIGHT\text{-}MAX} \leq T_{PERIOD\text{-}MIN} - (T_{CO\text{-}MAX} + T_{SU\text{-}MIN} + T_{CLK\_SKEW\text{-}MAX} + T_{CLK\_JITTER\text{-}MAX})$$

Where, $T_{CO\text{-}MAX}$ is the maximum clock-to-out delay of a driving agent, $T_{SU\text{-}MIN}$ is the minimum setup time required by a receiver on the same net, $T_{CLK\_SKEW\text{-}MAX}$ is the maximum anticipated time difference between the driver's and the receiver's clock inputs, and $T_{CLK\_JITTER\text{-}MAX}$ is maximum anticipated edge-to-edge phase jitter. The above equation should be checked for all pairs of devices on all nets of a bus.

The minimum acceptable Flight Time is determined by the following equation (known as the hold time equation):

$$T_{HOLD\text{-}MIN} \leq T_{FLIGHT\text{-}MIN} + T_{CO\text{-}MIN} - T_{CLK\_SKEW\text{-}MAX}$$

Where, $T_{CO\text{-}MIN}$ is the minimum clock-to-out delay of the driving agent, $T_{HOLD\text{-}MIN}$ is the minimum hold time required by the receiver, and $T_{CLK\_SKEW\text{-}MAX}$ is defined above. The Hold time equation is

independent of clock jitter, since data is released by the driver and is required to be held at the receiver on the same clock edge.

4.2. General GTL+ I/O Buffer Specification

This specification identifies the key parameters for the driver, receiver, and package that must be met to operate in the system environment described in the previous section. All specifications must be met over all possible operating conditions including temperature, voltage, and semiconductor process. *This information is included for designers of components for a GTL+ bus.*

4.2.1. I/O Buffer DC Specification

Table 20 contains the I/O Buffer DC parameters.

Table 20. I/O Buffer DC Parameters

Symbol	Parameter	Min	Max	Units	Notes
V_{OL}	Driver Output Low Voltage		0.600	V	1
V_{IH}	Receiver Input High Voltage	V_{REF} +0.2		V	2
V_{IL}	Receiver Input Low Voltage		V_{REF} –0.2	V	2
V_{ILC}	Input Leakage Current		10	µA	3
C_{IN}, C_o	Total Input/Output Capacitance		10	pF	4

NOTES:

1. Measured into a 25Ω test load tied to V_{TT} = 1.5 V, as shown in Figure 32.
2. V_{REF} = 2/3 V_{TT}. (V_{TT} = 1.5 V ±10%), V_{REF} has an additional tolerance of ± 2%.
3. This parameter is for inputs without internal pull-ups or pull downs and 0 ≤ VIN ≤ V_{TT}.
4. Total capacitance, as seen from the attachment node on the network, which includes traces on the PCB, IC socket, component package, driver/receiver capacitance, and ESD structure capacitance.

4.2.2. I/O Buffer AC Specification

Table 21. I/O Buffer AC Parameters

Symbol	Parameter	Min	Max	Unit	Figure	Notes
dV/dt_{EDGE}	Output Signal Edge Rate, rise.	0.3	0.8	V/ns		1, 2, 3
dV/dt_{EDGE}	Output Signal Edge Rate, fall.	0.3	–0.8	V/ns		1, 2, 3
T_{CO}	Output Clock to Data Time		no spec.	ns	Figure 3 3	4, 5
T_{SU}	Input Setup Time		no spec.	ns	Figure 2 4 Figure 2 5	4, 6
T_{HOLD}	Input Hold Time		no spec.	ns		4, 6

NOTES:

1. This is the maximum instantaneous dV/dt over the entire transition range (Hi-to-Lo or Lo-to-Hi) as measured at the driver's output *pin* while driving the Ref8N network, with the driver and its package model located near the center of the network (see Section 4.4).
2. These are design targets. The acceptance of the buffer is also based on the resultant signal quality. In addition to edge rate, the shape of the rising edge can also have a significant effect on the buffer's performance, therefore the driver must also meet the signal quality criteria in the next section. For example, a rising linear ramp of at 0.8V/ns will generally produce worse signal quality (more ringback) than an edge that rolls off as it approaches V_{TT} even though it might have exceeded that rate earlier. Hi-to-Lo edge rates may exceed this specification and produce acceptable results with a corresponding reduction in V_{OL}. For instance, a buffer with a falling edge rate larger than 1.5V/ns can been deemed acceptable because it produced a V_{OL} less than 500 mV. Lo-to-Hi edges must meet both signal quality and maximum edge rate specifications.
3. The minimum edge rate is a design target, and slower edge rates can be acceptable, although there is a timing impact associated with them in the form of an increase in flight time, since the signal at the receiver will no longer meet the required conditions for T_{SU}. Refer to Section 4.1.4 on computing flight time for more details on the effects of edge rates slower than 0.3V/ns.
4. These values are not specific to this Specification, they are dependent on the location of the driver along a network and the system requirements such as the number of agents, the distances between agents, the construction of the PCB (Z_0, ε_r, trace width, trace type, connectors), the sockets being used, if any, and the value of the termination resistors. Good targets for components to be used in an 8-load 66.6 MHz system would be: T_{CO_MAX} = 4.5 ns, T_{CO_MIN} = 1 ns, T_{SU} = 2.5 ns, and T_{HD} = 0.
5. This value is specified at the output pin of the device. T_{CO} should be measured at the test probe point shown in the Figure 34, but the delay caused by the 50Ω transmission line must be subtracted from the measurement to achieve an accurate value for T_{CO} at the output pin of the device. For simulation purposes, the tester load can be represented as a single 25Ω termination resistor connected directly to the pin of the device.
6. See Section 4.2.3 for a description of the procedure for determining the receiver's minimum required setup and hold times.

4.2.2.1. Output Driver Acceptance Criteria

Although Section 4.1.4 describes ways of amending flight time to a receiver when the edge rate is lower than the requirements shown in Table 21, or when there is excessive ringing, it is still preferable to avoid slow edge rates or excessive ringing through good driver and system design, hence the criteria presented in this section.

As mentioned in note 2 of the previous section, the criteria for acceptance of an output driver relate to the edge rate and the signal quality for the Lo-to-Hi transition, and primarily to the signal quality for the Hi-to-Lo transition when the device, with its targeted package, is simulated into the Ref8n network (Figure 36). The edge rate portion of the AC specification is a good initial target, but is

insufficient for guaranteeing acceptable performance.

Since Ref8N is not the worst case network, and is expected to be modeled without many real system effects (e.g., inter-trace crosstalk, DC & AC losses), the required signal quality is slightly different than that specified in Section 4.1.3 of this document.

The signal quality criterion for an acceptable driver design is that the signals produced by the driver (at its fastest corner) at all Ref8N receiver *pads* must remain outside of the shaded areas shown in Figure 30. Simulations must be performed at both device and operating extremes: fast process corner at high VCC and low temperature, and slow process corner at low VCC and high temperature, for both the rising and falling edges. The clock frequency should be at the desired maximum (e.g. 66.6 MHz, or higher), and the simulation results should be analyzed both from a quiescent start (i.e., first cycle in a simulation), and when preceded by at least one previous transition (i.e. subsequent simulation cycles).

The boundaries of the keep-out area for the Lo-to-Hi transition are formed by a vertical line at the start of the receiver setup window (a distance T_{SU}' from the next clock edge), an 0.3V/ns ramp line passing through the intersection between the V_{REF} +100 mV level (the 100 mV is assumed extra noise) and the beginning of the setup window, a horizontal line at V_{REF} +300 mV (which covers 200 mV of specified overdrive, and the 100 mV margin for extra noise coupled to the waveform), and finally a vertical line behind the Clock at T_{HD}'. The keep-out zone for the Hi-to-Lo transition uses analogous boundaries in the other direction. Raising V_{REF} by 100 mV is assumed to be equivalent to having 100 mV of extra noise coupled to the waveform giving it more downward ringback, such coupled noise could come from a variety of sources such as trace-to-trace PCB coupling.

T_{SU}' is the receiver's setup time plus board clock driver and clock distribution skew and jitter, plus an additional number that is inherited from the driver's internal timings (to be described next). Since the I/O buffer designer will most likely be simulating the driver circuit alone, certain delays that add to T_{CO}, such as: on-chip clock phase shift, clock distribution skew, and jitter, plus other data latch or JTAG delays would be missing. It is easier if these numbers are added to T_{SU}, yielding T_{SU}' making the driver simulation simpler. For example, assume T_{SU} to be 2.8 ns, PCB clock generation and distribution skew plus jitter to be 1 ns, and unmodeled delays in the driver to be typically about 0.8 ns, this yields a total $T_{SU}' = 4.6$ ns.

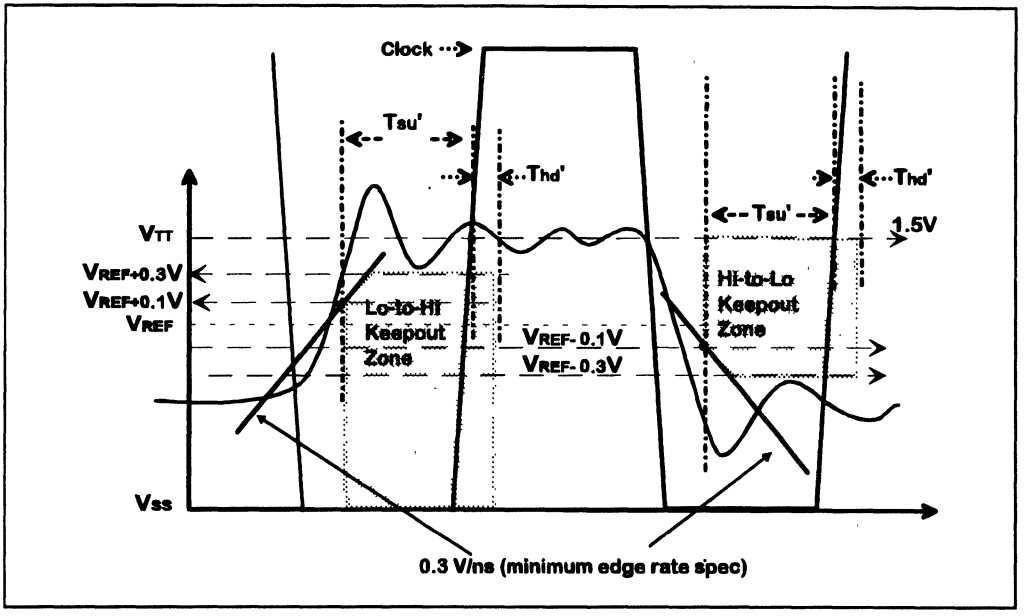


Figure 30. Acceptable Driver Signal Quality

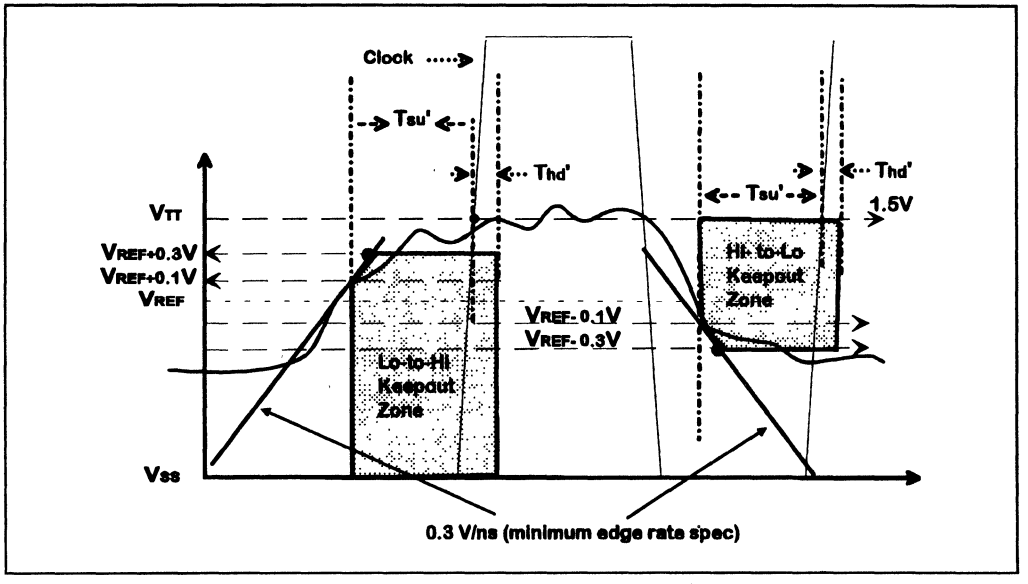


Figure 31. Unacceptable signal, Due to Excessively Slow Edge After Crossing V_{REF}

T_{HD}' is the receiver's hold time plus board clock driver and clock distribution skew *minus* the driver's on-chip clock phase shift, clock distribution skew, and jitter, plus other data latch or JTAG delays (assuming these driver numbers are not included in the driver circuit simulation, as was done for setup in the above paragraph). Note that T_{HD}' may end up being a negative number, i.e. ahead of the clock, rather than after it. That would be acceptable, since that is equivalent to shifting the driver output later in time had these extra delays been added to the driver as opposed to setup and hold.

When using Ref8N to validate a driver design, it is recommended that all relevant combinations of driver and receiver locations be checked.

As with other buffer technologies, such as TTL or CMOS, any given buffer design is not guaranteed to always meet the requirements of all possible system and network topologies. Meeting the acceptance criteria listed in this document helps ensure the I/O buffer can be used in a variety of GTL+ applications, but it is the system designer's responsibility to examine the performance of the buffer in the specific application to ensure that all GTL+ networks meet the signal quality requirements.

4.2.3. Determining Clock-To-Out, Setup and Hold

This section will describe how to determine setup, hold and clock to out timings.

4.2.3.1. Clock-to-Output Time, T_{CO}

T_{CO} is measured using the test load in Figure 34, and is the delay from the 1.5 V crossing point of the clock signal at the clock input *pin* of the device, to the V_{REF} crossing point of the output signal at the output *pin* of the device. For simulation purposes, the test load can be replaced by its electrical equivalent, which is a single 25Ω resistor connected directly to the package pin and terminated to 1.5V.

In a production test environment, it is nearly impossible to measure T_{CO} directly at the output pin of the device, instead, the test is performed a finite distance away from the pin and compensated for the finite distance. The test load circuit shown in Figure 34 takes this into account by making this finite distance a 50-Ω transmission line. To get the exact timings at the output pin, the propagation delay along the transmission line must be subtracted from the measured value at the probe point.

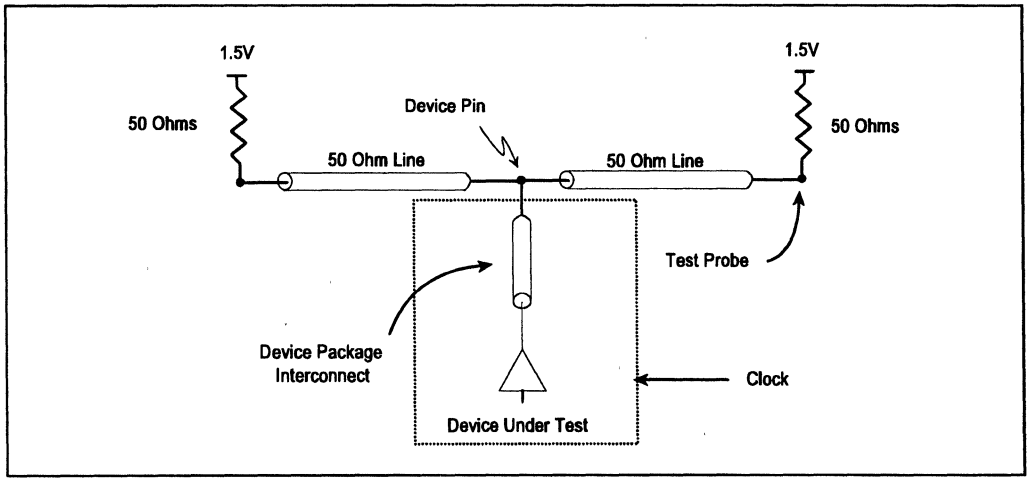


Figure 32. Test Load for Measuring Output AC Timings

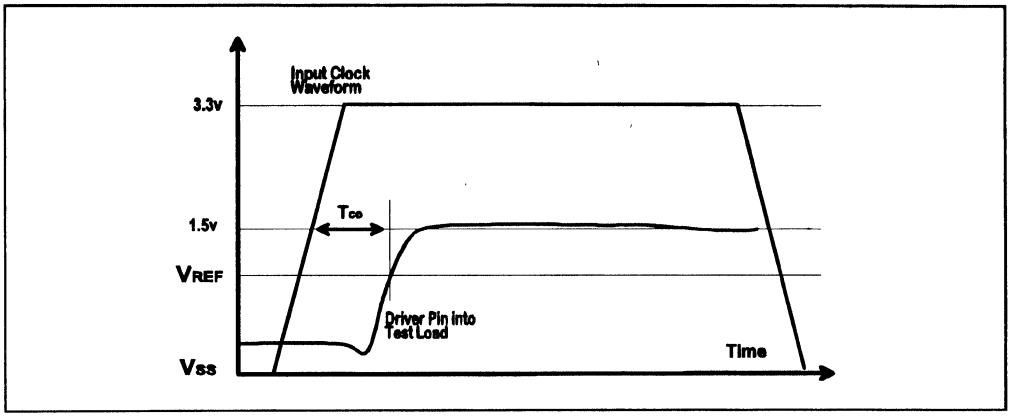


Figure 33. Clock to Output Data Timing (T_{CO})

T_{CO} measurement for a Lo-to-Hi signal transition is shown in Figure 35. The T_{CO} measurement for Hi-to-Lo transitions is similar.

4.2.3.2. Minimum Setup and Hold Times

Setup time for GTL+ (T_{SU}) is defined as:

> The minimum time from the input signal pin crossing of V_{REF} to the clock pin of the receiver crossing the 1.5 V level, which guarantees that the input buffer has captured **new** data at the input pin, given an infinite hold time.

Strictly speaking, setup time must be determined when the input barely meets minimum hold time (see definition of hold time below). However, for current GTL+ systems, hold time should be met well beyond the minimum required in cases where setup is critical. This is because setup is critical when the receiver is far removed from the driver. In such cases, the signal will be held at the receiver for a long time after the clock, since the change needs a long time to propagate from the driver to the receiver.

The recommended procedure for the I/O buffer designer to extract T_{SU} is outlined below. If one employs additional steps, it would be beneficial that any such extra steps be documented with the results of this receiver characterization:

1. The full receiver circuit must be used, comprising the input differential amplifier, any shaping logic gates, and the edge-triggered (or pulse-triggered) flip-flop. The output of the flip-flop must be monitored.

2. The receiver's Lo-to-Hi setup time should be determined using a nominal input waveform like the one shown in Figure 24 (solid line). The Lo-to-Hi input starts at $V_{IN\_LOW\_MAX}$ (V_{REF} - 200 mV) and goes to $V_{IN\_HIGH\_MIN}$ = V_{REF} +200 mV, at a slow edge rate of 0.3V/ns, with the process, temperature, voltage, and $V_{REF\_INTERNAL}$ of the receiver set to the worst (longest T_{SU}) corner values. Here, V_{REF} is the external (system) reference voltage at the device pin. Due to tolerance in V_{TT} (1.5V, ±10%) and the voltage divider generating system V_{REF} from V_{TT} (±2%), V_{REF} can shift around 1 V by a maximum of ±122 mV. When determining setup time, the internal reference voltage $V_{REF\_INTERNAL}$ (at the reference gate of the diff. amp.) must be set to the value which yields the longest setup time. Here, $V_{REF\_INTERNAL}$ = V_{REF} ±(122 mV +V_{NOISE}). Where, V_{NOISE} is the net maximum differential noise amplitude on the component's internal V_{REF} distribution bus (at the amplifier's reference input gate) comprising noise picked up by the connection from the V_{REF} package pin to the input of the amp.

3. Analogously, for the setup time of Hi-to-Lo transitions (Figure 25), the input starts at $V_{IN\_HIGH\_MIN}$ = V_{REF} +200 mV and drops to $V_{IN\_LOW\_MAX}$ = V_{REF} - 200 mV at the rate of 0.3V/ns.

4. For both the 0.3V/ns edge rate and faster edge rates (up to 0.8V/ns for Lo-to-Hi, and 3V/ns for Hi-to-Lo —dashed lines in Figure 24 and Figure 25), one must ensure that lower starting voltages of the input swing (V_{START} in the range 'V_{REF}–200 mV' to 0.5 V for Lo-to-Hi transitions, and 1.5 V to 'V_{REF}+200 mV' for Hi-to-Lo transitions —dashed lines in Figure 25 and Figure 26) do not require T_{SU} to be made longer. This step is needed since a lower starting voltage may cause the input differential amplifier to require more time to switch, due to having been in deeper saturation in the initial state.

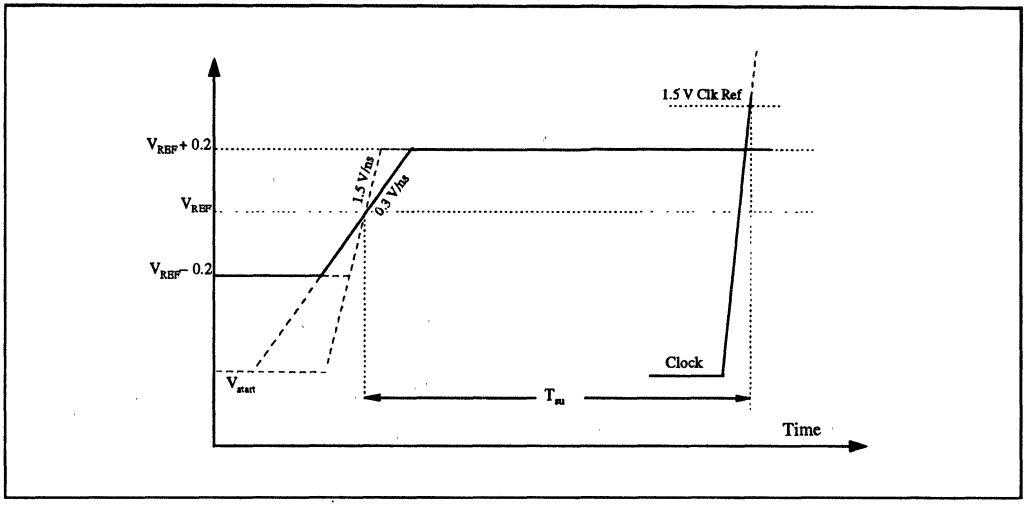


Figure 34. Standard Input Lo-to-Hi Waveform for Characterizing Receiver Setup Time

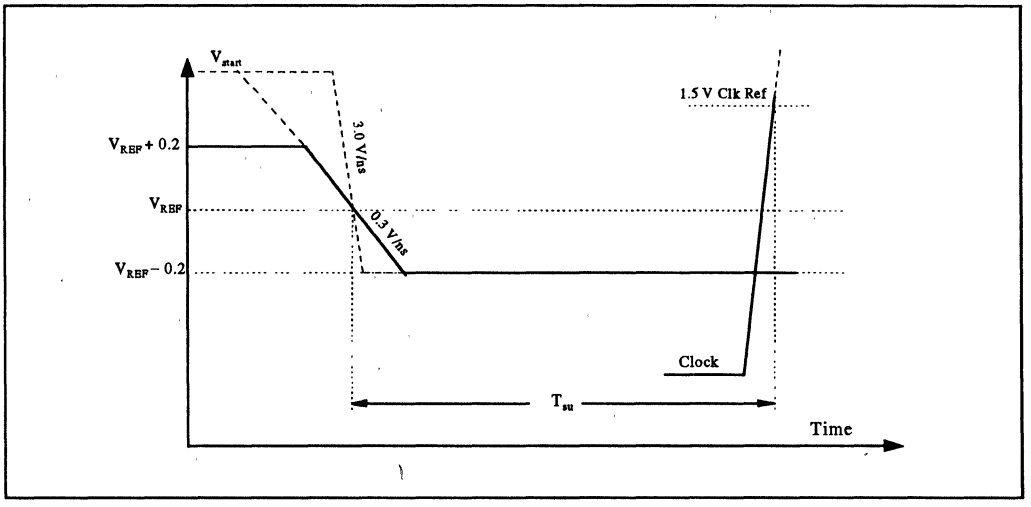


Figure 35. Standard Input Hi-to-Lo Waveform for Characterizing Receiver Setup Time

Hold time for GTL+ , T_{HOLD}, is defined as:

> The minimum time from the clock pin of the receivers crossing of the 1.5 V level to the receiver input signal pin crossing of V_{REF}, which guarantees that the input buffer has captured **new** data at the receiver input signal pin, given an infinite setup time.

Strictly speaking, hold time must be determined when the input barely meets minimum setup time (see definition of setup time above). However, for current GTL+ systems, setup time is expected to be met, well beyond the minimum required in cases where hold is critical. This is because hold is critical when the receiver is very close to the driver. In such cases, the signal will arrive at the receiver shortly after the clock, hence meeting setup time with comfortable margin.

The recommended procedure for extracting T_{HOLD} is outlined below. If one employs additional steps, it would be beneficial that any such extra steps be documented with the results of this receiver characterization:

1. The full receiver circuit must be used, comprising the input differential amplifier, any shaping logic gates, and the edge-triggered (or pulse-triggered) flip-flop. The output of the flip-flop must be monitored.
2. The receiver's Lo-to-Hi hold time should be determined using a nominal input waveform that starts at $V_{IN\_LOW\_MAX}$ (V_{REF} - 200 mV) and goes to V_{TT}, at a fast edge rate of 0.8V/ns, with the process, temperature, voltage, and $V_{REF\_INTERNAL}$ of the receiver set to the fastest (or best) corner values (yielding the longest T_{HOLD}). Here, V_{REF} is the external (system) reference voltage at the device pin. Due to tolerance in V_{TT} (1.5V, ±10%) and the voltage divider generating system V_{REF} from V_{TT} (±2%), V_{REF} can shift around 1V by a maximum of ±122 mV. When determining hold time, the internal reference voltage $V_{REF\_INTERNAL}$ (at the reference gate of the diff. amp.) must be set to the value which yields the worst case hold time. Here, $V_{REF\_INTERNAL}$ = V_{REF} ±(122 mV +V_{NOISE}). Where, V_{NOISE} is the net maximum differential noise amplitude on the component's internal V_{REF} distribution bus (at the amplifier's reference input gate) comprising noise picked up by the connection from the V_{REF} package pin to the input of the amp.
3. Analogously, for the hold time of Hi-to-Lo transitions, the input starts at $V_{IN\_HIGH\_MIN}$ = V_{REF} +200 mV and drops to < 0.5 V at the rate of 3V/ns.

4.2.3.3. Receiver Ringback Tolerance

Refer to Section 4.1.3.1 for a complete description of the definitions and methodology for determining receiver ringback tolerance.

1

4.2.4. System-Based Calculation of Required Input and Output Timings

Below are two sample calculations. The first determines $T_{CO\text{-}MAX}$ and $T_{SU\text{-}MIN}$, while the second determines $T_{HOLD\text{-}MIN}$. These equations can be used for any system by replacing the assumptions listed below, with the actual system constraints.

4.2.4.1. Calculating Target T_{CO}-max, and T_{SU}-Min

$T_{CO\text{-}MAX}$ and $T_{SU\text{-}MIN}$ can be calculated from the Setup Time equation given earlier in Section 4.1.4:

$$T_{FLIGHT\text{-}MAX} \leq T_{PERIOD\text{-}MIN} - (T_{CO\text{-}MAX} + T_{SU\text{-}MIN} + T_{CLK\_SKEW\text{-}MAX} + T_{CLK\_JITTER\text{-}MAX})$$

As an example, for two identical agents located on opposite ends of a network with a flight time of 7.3 ns, and the other assumptions listed below, the following calculations for $T_{CO\text{-}MAX}$ and $T_{SU\text{-}MIN}$ can be done:

Assumptions:

$T_{PERIOD\text{-}MIN}$	15. ns	(66.6 MHz)
$T_{FLIGHT\text{-}MAX}$	7.3 ns	(given flight time)
$T_{CLK\_SKEW\text{-}MAX}$ driver)	0.7 ns	(0.5ns for clock
		(0.2 ns for board skew)
$T_{CLK\_JITTER\text{-}MAX}$	0.2 ns	(Clock phase error)
$T_{CO\text{-}MAX}$	??	(Clock to output data time)
$T_{SU\text{-}MIN}$	??	(Required input setup time)

Calculation:

$$7.3 \leq 15 - (T_{CO\text{-}MAX} + T_{SU\text{-}MIN} + 0.7 + 0.2)$$

$T_{CO\text{-}MAx} + T_{SU\text{-}MIN} \leq 6.8$ ns

The time remaining for $T_{CO\text{-}MAX}$ and $T_{SU\text{-}MIN}$ can be split ~60/40% (recommendation). Therefore, in this example, $T_{CO\text{-}MAX}$ would be 4.0 ns, and $T_{SU\text{-}MIN}$ 2.8 ns.

NOTE

This a numerical example, and does not necessarily apply to any particular device.

Off-end agents will have less distance to the farthest receiver, and therefore will have shorter flight times. T_{CO} values longer than the example above do not necessarily preclude high-frequency (e.g. 66.6 MHz) operation, but will result in placement constraints for the device, such as being required to be placed in the middle of the daisy-chain bus.

4.2.5. Calculating Target $T_{hold\text{-}min}$

To calculate the longest possible minimum required hold time target value, assume that $T_{CO\text{-}MIN}$ is one fourth of $T_{CO\text{-}MAX}$, and use the hold time equation given earlier. Note that Clock Jitter is not a part of the equation, since data is released by the driver and must be held at the receiver relative to the same clock edge:

$T_{HOLD\text{-}MIn} \leq T_{FLIGHT\text{-}MIN} + T_{CO\text{-}MIN} - T_{CLK\_SKEW\text{-}MAX}$

Assumptions:

$T_{CO\text{-}MAX}$	4.0 ns	(Max clock to data time)
$T_{CO\text{-}MIN}$	1.0 ns	(Assumed ¼ of max)
$T_{CLK\_SKEW\text{-}MAX}$	0.7 ns	(Driver to receiver skew)
$T_{FLIGHT\text{-}MIN}$	0.1 ns	(Min of 0.5" at 0.2 ns/inch)
$T_{HOLD\text{-}MIN}$	??	(Minimum signal hold time)

Calculation:

$T_{HOLD\text{-}MIN} \leq 0.1 + 1.0 - 0.7$

$T_{HOLD\text{-}MIN} \leq 0.4$ ns.

NOTE

This a numerical example, and does not necessarily apply to any particular device.

4.3. Package Specification

This information is also included for designers of components for a GTL+ bus. The package that the I/O transceiver will be placed into must adhere to two critical parameters. They are package trace length, (the electrical distance from the pin to the die), and package capacitance. The specifications for package trace length and package capacitance are not explicit, but are implied by the system and I/O buffer specifications.

4.3.1. Package Trace Length

The System specification requires that all signals be routed in a daisy chain fashion, and that no stub in the network exceed 250 ps in electrical length. The stub includes any printed circuit board (PCB) routing to the pin of the package from the "Daisy Chain" net, as well as a socket if necessary, and the trace length of the package interconnect (i.e. the electrical length from the pin, through the package, across a bond wire if necessary, and to the die). For example, for a PGA package, which allows PCB routing both to and from a pin and is soldered to the PCB, the maximum package trace length cannot exceed 250 ps. If the PGA package is socketed, the maximum package trace length would be ~225 ps since a typical PGA socket is around 25 ps in electrical length. For a QFP package, which typically requires a short stub on the PCB from the pad landing to a via (~50 ps), the package lead frame length should be less than ~200 ps.

4.3.2. Package Capacitance

The maximum package pin capacitance is a function of the Input/Output capacitance of the I/O transceiver. The I/O Buffer specification requires the total of the package capacitance, output driver, input receiver and ESD structures, as seen from the pin, to be less than 10 pF. Thus, the larger the I/O transceiver capacitance, the smaller the allowable package capacitance.

4.4. Ref8N Network

The Ref8N network shown below, which represents an eight-node reference network (hence the name Ref8N), is used to characterize I/O drivers' behavior into a known environment. This network is not a worst case, but a representative sample of a typical

system environment. A SPICE deck of the network is also given.

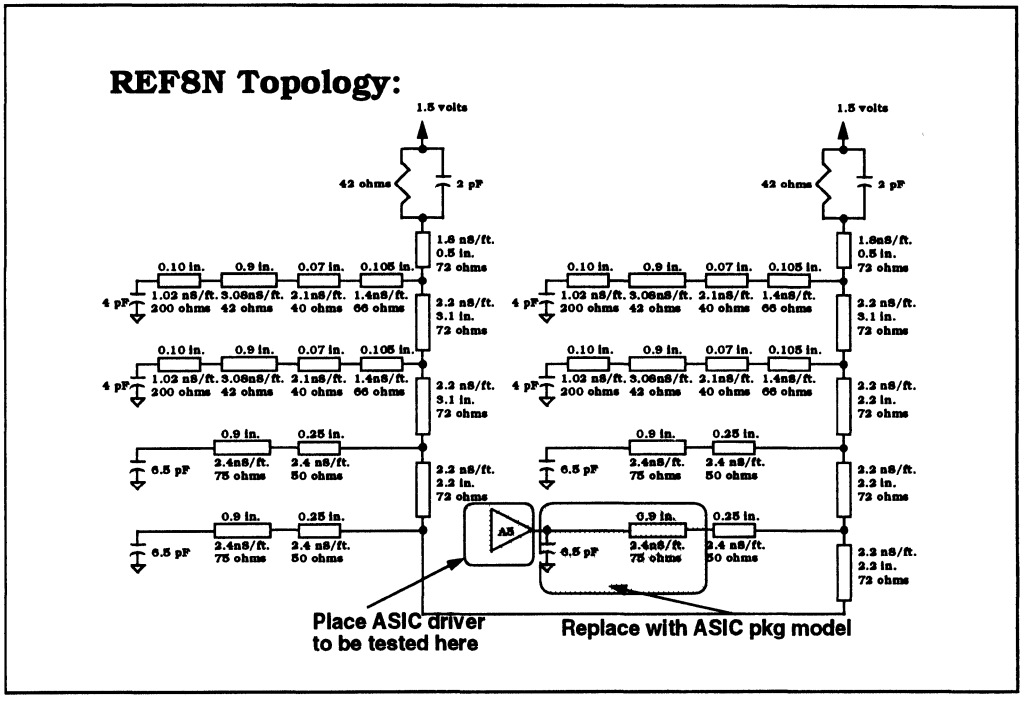


Figure 36. Ref8N Topology

4.4.1. Ref8N HSPICE Netlist:

```
$REF8N, Rev 1.1

Vpu vpu GND DC(vtt)

rterm PU1 vpu (R=42)                    $ Pull-up termination resistance
crterm PU1 vpu 2PF                      $ Pull-up termination capacitance
TPU PU1 0 line1 0 Z0=72 TD=.075NS       $ PCB link from terminator to load 1

X1 line1 load1 socket                   $ Socket model
T1 load1 0 load1a 0 Z0=42 TD=230PS      $ CPU package model
T2 load1a 0 CPU_1 0 Z0=200 TD=8.5PS     $ Bondwire
CCPU_1 CPU_1 0 4PF                      $ CPU input capacitance

T3 line1 0 line2 0 Z0=72 TD=568PS       $ PCB trace between packages
```

```
x2 line2 load2 socket                 $ Socket model
T4 load2 0 load2a 0 Z0=42 TD= 230ps   $ CPU worst case package
T5 load2a 0 p6_2 0 Z0=200 TD=8.5ps    $ Bondwire
CCPU_2 p6_2 0 4pf                     $ CPU input capacitance

T6 line2 0 line3 0 Z0=72 TD=568ps     $ PCB trace between packages
T7 line3 0 load3 0 Z0=50 TD=50ps      $ PCB trace from via to landing pad
T8 load3 0 asic_1 0 Z0=75 TD=180PS    $ ASIC package
CASIC_1 asic_1 0 6.5PF                $ ASIC input capacitance (die capacitance)

T9 line3 0 line4 0 Z0=72 TD=403PS     $ PCB trace between packages
T10 line4 0 load4 0 Z0=50 TD=50PS     $ PCB trace from via to landing pad
T11 load4 0 asic_2 0 Z0=75 TD=180PS   $ ASIC package
CASIC_2 asic_2 0 6.5PF                $ ASIC input capacitance (die capacitance)

T12 line4 0 line5 0 Z0=72 TD=403PS    $ PCB trace between packages
T13 line5 0 load5 0 Z0=50 TD=50PS     $ PCB trace from via to landing pad
T14 load5 0 asic_3 0 Z0=75 TD=180PS   $ Replace this line and the next line with
CASIC_3 asic_3 0 6.5PF                $ the equivalent model for your package.
                                      $ (This model should include the package
                                      $ pin, package trace, bond wire and any die
                                      $ capacitance that is not already included
                                      $ in your driver model.)

T15 line5 0 line6 0 Z0=72 TD=403PS    $ PCB trace between packages
T16 line6 0 load6 0 Z0=50 TD=50PS     $ PCB trace from via to landing pad
T17 load6 0 asic_4 0 Z0=75 TD=180PS   $ ASIC package
CASIC_4 asic_4 0 6.5PF                $ ASIC input capacitance

T18 line6 0 line7 0 Z0=72 TD=403PS    $ PCB trace between packages
X3 line7 load7 socket                 $ Socket model
T19 load7 0 load7a 0 Z0=42 TD=230PS   $ CPU worst case package
T20 load7a 0 p6_3 0 Z0=200 TD=8.5PS   $ Bondwire
CCPU_3 p6_3 0 4PF                     $ CPU input capacitance

T21 line7 0 line8 0 Z0=72 TD=568PS    $ PCB trace between packages
X4 line8 load8 socket                 $ Socket model
T22 load8 0 load8a 0 Z0=42 TD=230PS   $ CPU worst case package
T23 load8a 0 p6_4 0 Z0=200 TD=8.5PS   $ Bondwire
CCPU_4 p6_4 0 4PF                     $ CPU input capacitance

T24 line8 0 R_TERM 0 Z0=72 TD=75PS    $ PCB trace to termination resistor
Rterm1 R_TERM vpu (R=42)               $ Pull-up termination resistance
CRTERM1 R_TERM vpu (C=2PF)            $ Pull-up termination capacitance

Rout bond asic_3.001

.subckt socket in out                  $ Socket model
```

```
TX out 0 jim 0 Z0=40 TD=12.25PS
ty jim 0 in 0 Z0=66 TD=12.25ps
.ENDS
```

5.0 3.3V Tolerant Signal Quality Specifications

The signals that are 3.3V tolerant should also meet signal quality specifications to guarantee that the components read data properly and to ensure that incoming signals do not affect the long term reliability of the component. There are three signal quality parameters defined for the 3.3V tolerant signals. They are Overshoot/Undershoot, Ringback and Settling Limit. All three signal quality parameters are shown in Figure 37. The *Pentium® Pro Processor I/O Buffer Models—IBIS Format (On world wide web page www.intel.com)* contain models for simulating 3.3V tolerant signal distribution.

5.1. OVERSHOOT/UNDERSHOOT GUIDELINES

Overshoot (or undershoot) is the absolute value of the maximum voltage above the nominal high voltage or below VSS. The overshoot/undershoot guideline limits transitions beyond $V_{CC}P$ or V_{SS} due to the fast signal edge rates. See Figure 37. The processor can be damaged by repeated overshoot events on 3.3V tolerant buffers if the charge is large enough (i.e. if the overshoot is great enough). However, excessive ringback is the dominant harmful effect resulting from overshoot or undershoot (i.e. violating the overshoot/undershoot guideline will make satisfying the ringback specification difficult). The **overshoot/undershoot guideline is 0.8V** and assumes the absence of diodes on the input. These guidelines should be verified in simulations **without the on-chip ESD protection diodes present** because the diodes will begin clamping the 3.3V tolerant signals beginning at approximately 1.5V above $V_{CC}P$ and 0.5V below VSS. If signals are not reaching the clamping voltage, then this is not an issue. A system should not rely on the diodes for overshoot/undershoot protection as this will negatively affect the life of the components and make meeting the ringback specification very difficult.

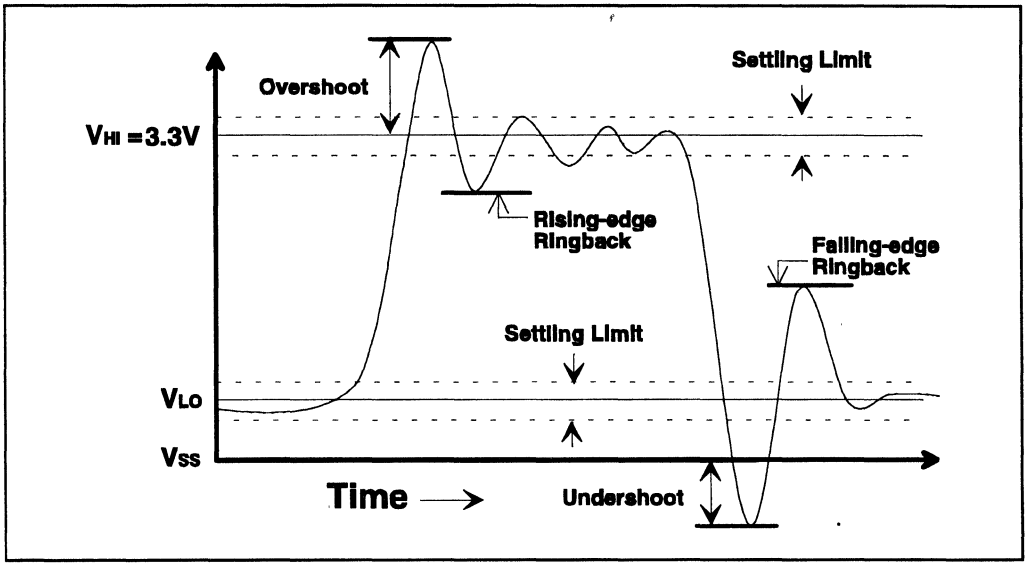


Figure 37. 3.3V Tolerant Signal Overshoot/Undershoot and Ringback

5.2. RINGBACK SPECIFICATION

Ringback refers to the amount of reflection seen after a signal has undergone a transition. The ringback specification is the voltage that the signal rings back to after achieving its farthest excursion. See Figure 37 for an illustration of ringback. Excessive ringback can cause false signal

detection or extend the propagation delay. The ringback specification applies to the input pin of each receiving agent. Violations of the signal Ringback specification are not allowed under any circumstances.

Ringback can be simulated with or without the input protection diodes that can be added to the input buffer model. However, signals that reach the clamping voltage should be evaluated further. See Table 22 for the signal ringback specifications for Non-GTL+ signals

Table 22. Signal Ringback Specifications

Transition	Maximum Ringback (with Input diodes present)
0→1	2.5V
1→0	0.8V

5.3. SETTLING LIMIT GUIDELINE

A Settling Limit defines the maximum amount of ringing at the receiving pin that a signal must be limited to before its next transition. The amount allowed is 10% of the total signal swing (V_{HI}-V_{LO}) above and below its final value. A signal should be within the settling limits of its final value, when either in its high state of low state, before it transitions again.

Signals that are not within their settling limit before transitioning are at risk of unwanted oscillations which could jeopardize signal integrity. Simulations to verify Settling Limit may be done either with or without the input protection diodes present. Violation of the Settling Limit guideline is acceptable if simulations of 5-10 successive transitions do not show the amplitude of the ringing increasing in the subsequent transitions.

6.0. THERMAL SPECIFICATIONS

Table 5 specifies the Pentium Pro processor power dissipation. It is highly recommended that systems be designed to dissipate at least ***35-40W*** per processor to allow the same design to accommodate higher frequency or otherwise enhanced members of the Pentium Pro processor family.

6.1. Thermal Parameters

This section defines the terms used for Pentium Pro processor thermal analysis.

6.1.1. AMBIENT TEMPERATURE

Ambient temperature, T_A, is the temperature of the ambient air surrounding the package. In a system environment, ambient temperature is the temperature of the air upstream from the package and in its close vicinity; or in an active cooling system, it is the inlet air to the active cooling device.

6.1.2. CASE TEMPERATURE

To ensure functionality and reliability, the Pentium Pro processor is specified for proper operation when T_C (case temperature) is within the specified range in Table 5. Special care is required when measuring the case temperature to ensure an accurate temperature measurement. Thermocouples are often used to measure T_C. Before any temperature measurements, the thermocouples must be calibrated. When measuring the temperature of a surface which is at a different temperature from the surrounding ambient air, errors could be introduced in the measurements if not handled properly. The measurement errors could be due to having a poor thermal contact between the thermocouple junction and the surface, heat loss by radiation, or by conduction through thermocouple leads. To minimize the measurement errors, the following approach is recommended:

- Use a 35 gauge K-type thermocouple or equivalent.
- Attach the thermocouple bead or junction to the package top surface at a location corresponding to the center of the Pentium Pro processor die. (Location A in Figure 38) Using the center of the Pentium Pro processor die gives a more accurate measurement and less variation as the boundary condition changes
- Attach the thermocouple bead or junction at a 90° angle by an adhesive bond (such as thermal grease or heat-tolerant tape) to the package top surface as shown in Figure 39. When a heat sink is attached, a hole should be drilled through the heat sink to allow probing the Pentium Pro processor package above the

center of the Pentium Pro processor die. The hole diameter should be no larger than 0.150."

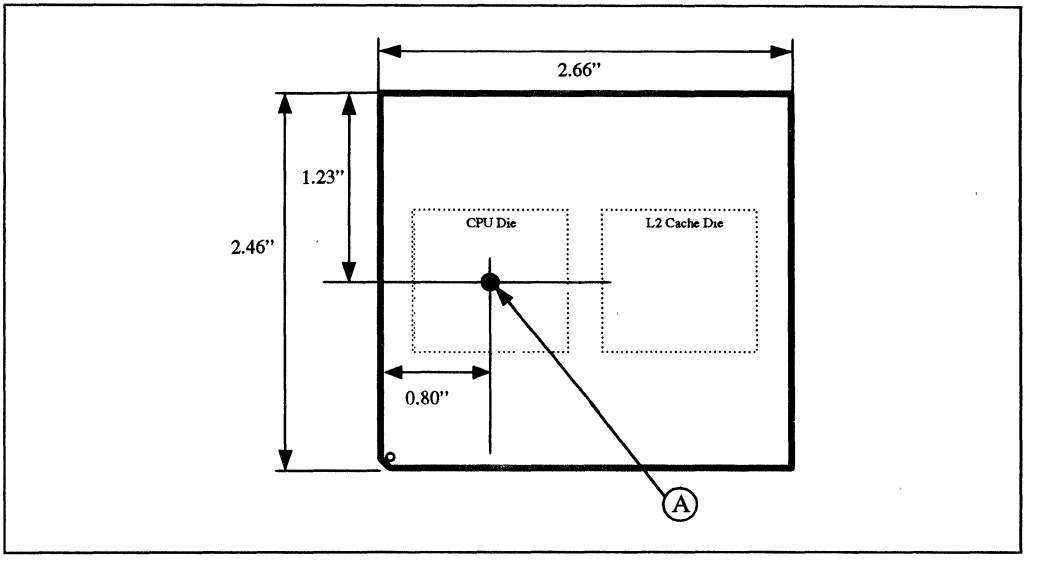


Figure 38. Location of Case Temperature Measurement (Top-side View)

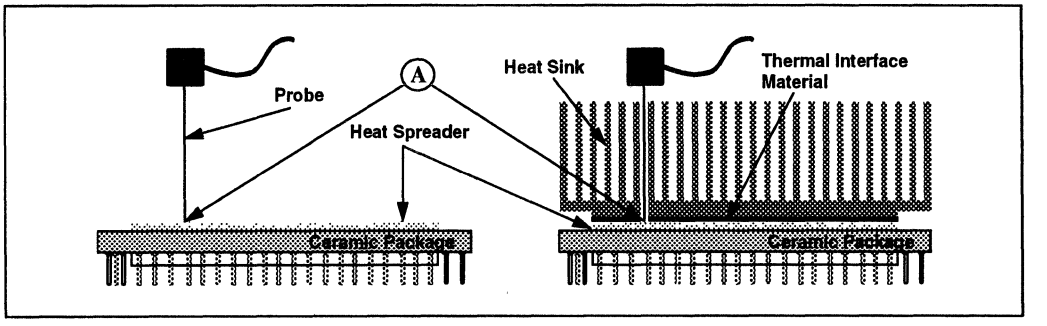


Figure 39. Thermocouple Placement

6.1.3. THERMAL RESISTANCE

The thermal resistance value for the case-to-ambient, Θ_{CA}, is used as a measure of the cooling solution's thermal performance. Θ_{CA} is comprised of the case-to-sink thermal resistance, Θ_{CS}, and the sink-to-ambient thermal resistance, Θ_{SA}. Θ_{CS} is a measure of the thermal resistance along the heat flow path from the top of the IC package to the bottom of the thermal cooling solution. This value is strongly dependent on the material, conductivity, and thickness of the thermal interface used. Θ_{SA} is a measure of the thermal resistance from the top of the cooling solution to the local ambient air. Θ_{SA} values depend on the material, thermal conductivity, and geometry of the thermal cooling solution as well as on the airflow rates.

The parameters are defined by the following relationships (See also Figure 40.):

$$Q_{CA} = (T_C - T_A) / P_D$$

$Q_{CA} = \Theta_{CS} + \Theta_{SA}$

Where:

Q_{CA} = Case-to-Ambient thermal resistance (°C/W)

Q_{CS} = Case-to-Sink thermal resistance (°C/W)

Q_{SA} = Sink-to-Ambient thermal resistance (°C/W)

T_C = Case temperature at the pre-defined location (°C)

T_A = Ambient temperature (°C)

P_D = Device power dissipation (W)

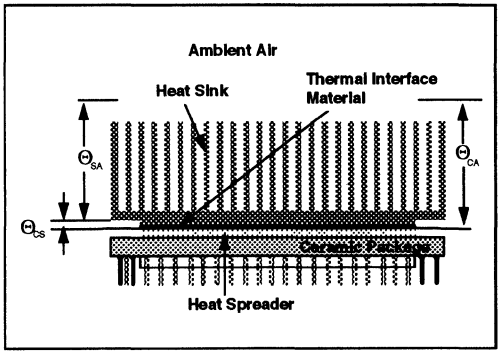


Figure 40. Thermal Resistance Relationships

6.2. Thermal Analysis

The following is an analysis of the thermal properties of the Pentium Pro processor package. This thermal analysis of the Pentium Pro processor package was based on the following **assumptions**. Note that this analysis varies for different versions of the Pentium Pro processor.

- Power dissipation is assumed to be a constant 29.2W.
- Maximum case temperature is assumed to be 85°C.

Table 23 below lists the case-to-ambient thermal resistances of the Pentium Pro processor for different air flow rates and heat sink heights. Table 24 shows the T_A required given a 29.2W processor, and a T_C of 85°C. Table 24 was produced by using the relationships of Section 6.1.3. and the data of Table 23.

Table 23. Case-To-Ambient Thermal Resistance

	ΘCA [°C/W] vs. Airflow [Linear Feet per Minute] and Heat Sink Height[1]					
Airflow (LFM):	**100**	**200**	**400**	**600**	**800**	**1000**
With 0.5" Heat Sink [2]	—	3.16	2.04	1.66	1.41	1.29
With 1.0" Heat Sink [2]	2.55	1.66	1.08	0.94	0.80	0.76
With 1.5" Heat Sink [2]	1.66	1.31	0.90	0.78	0.71	0.67
With 2.0" Heat Sink [2]	1.47	1.23	0.87	0.75	0.69	0.65

NOTES:

1. All data taken at sea level. For altitudes above sea level, it is recommended that a derating factor of 1°C/1000 feet be used.
2. Heat Sink: 2.235" square omni-directional pin, aluminum heat sink with a pin thickness of 0.085", a pin spacing of 0.13" and a base thickness of 0.15". See Figure 41. A thin layer of thermal grease (Thermoset TC208 with thermal conductivity of 1.2W/m-°K) was used as the interface material between the heat sink and the package.

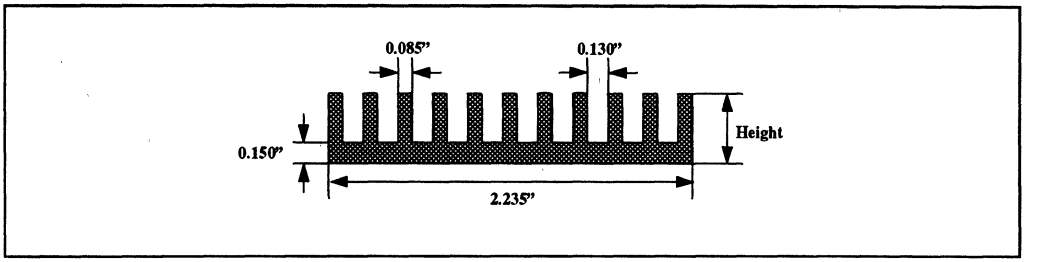


Figure 41. Analysis Heat Sink Dimensions

Table 24. Ambient Temperature Required at Heat Sink Height1

	T_A vs. Airflow [Linear Feet per Minute] and Heat Sink Height[1]					
Airflow (LFM):	**100**	**200**	**400**	**600**	**800**	**1000**
With 0.5" Heat Sink [2]	—	-8	25	36	43	47
With 1.0" Heat Sink [2]	10	36	53	57	61	62
With 1.5" Heat Sink [2]	36	46	58	62	64	65
With 2.0" Heat Sink [2]	42	49	59	63	64	66

NOTES:

1. All data taken at sea level. For altitudes above sea level, it is recommended that a derating factor of 1°C/1000 feet be used.
2. Heat Sink: 2.235" square omni-directional pin, aluminum heat sink with a pin thickness of 0.085", a pin spacing of 0.13" and a base thickness of 0.15". See Figure 41. A thin layer of thermal grease (Thermoset TC208 with thermal conductivity of 1.2W/m-°K) was used as the interface material between the heat sink and the package.

7.0. MECHANICAL SPECIFICATIONS

The Pentium Pro processor is packaged in a modified staggered 387 pin ceramic pin grid array (SPGA) with a gold plated Copper-Tungsten (CuW) heat spreader on top. Mechanical specifications and the pin assignments follow.

7.1. Dimensions

The mechanical specifications are provided in Table 25. Figure 42 shows the bottom and side views with package dimensions for the Pentium Pro processor and Figure 43 shows the top view with dimensions. Figure 44 is the top view of the Pentium Pro processor with VCCP, VCCS, $V_{CC}5$, and VSS locations shown. **Be sure to read Section 8 for the mechanical constraints for the OverDrive processor. Also, investigate the tools that will be used to debug the system before laying out the system.**

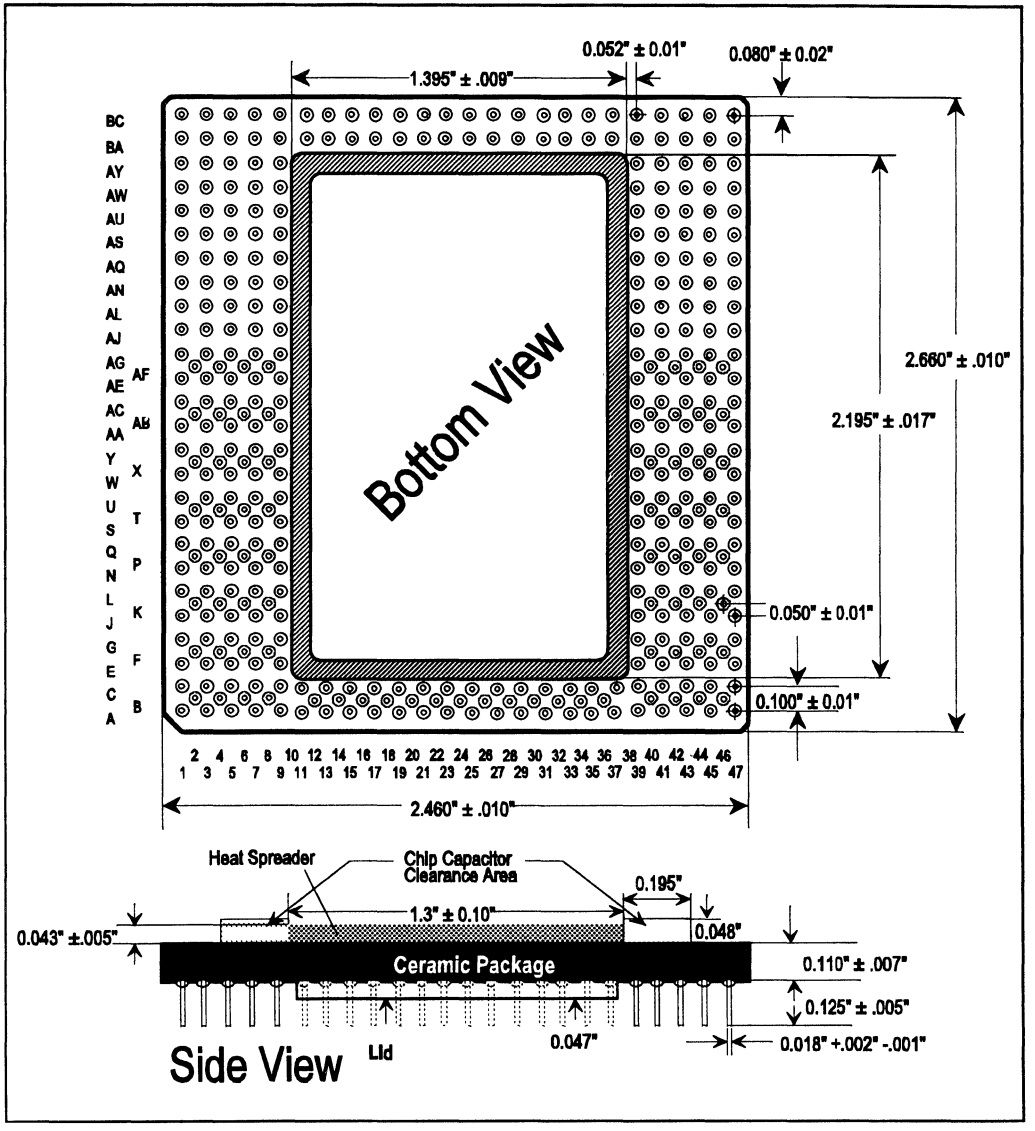


Figure 42. Package Dimensions (Bottom View)

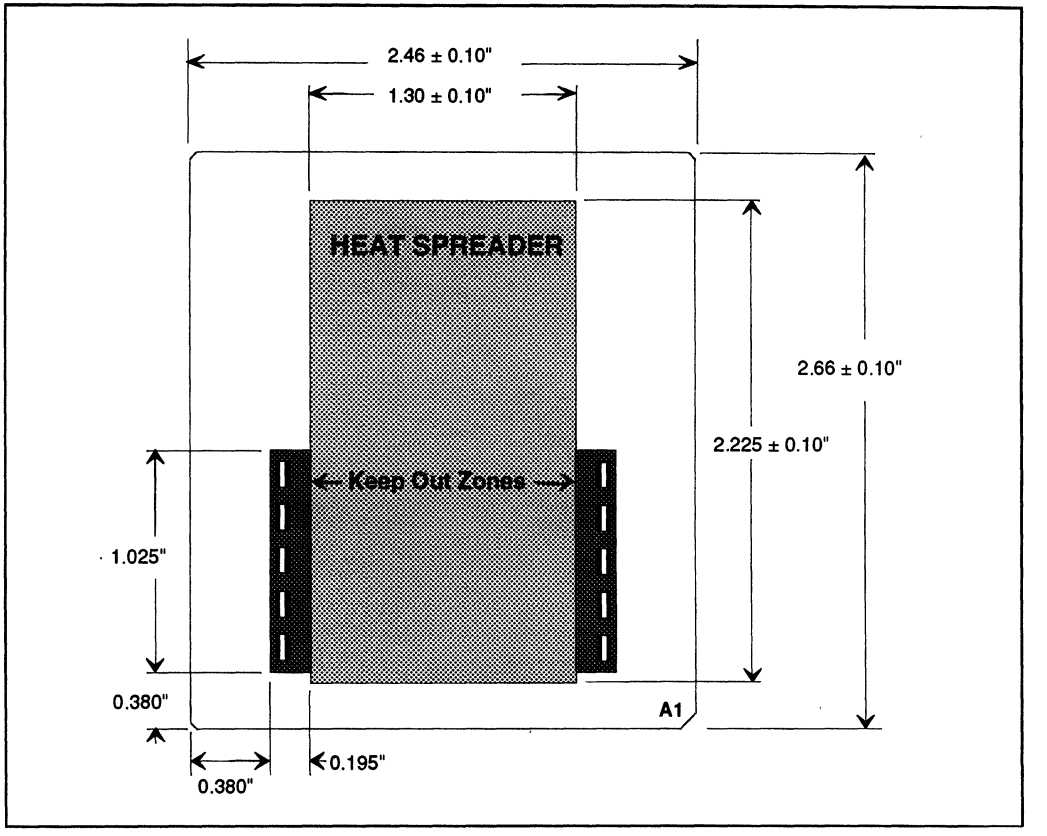


Figure 43. Top View of Keep Out Zones and Heat Spreader

Table 25. Pentium® Pro Processor Package

Parameter	Value
Package Type	PGA
Total Pins	387
Pin Array	Modified Staggered
Package Size	2.66" x 2.46" (7.76cm x 6.25cm)
Heat Spreader Size	2.225" x 1.3" x 0.04" (5.65cm x 3.3cm x 0.1cm)
Approximate Weight	90 grams

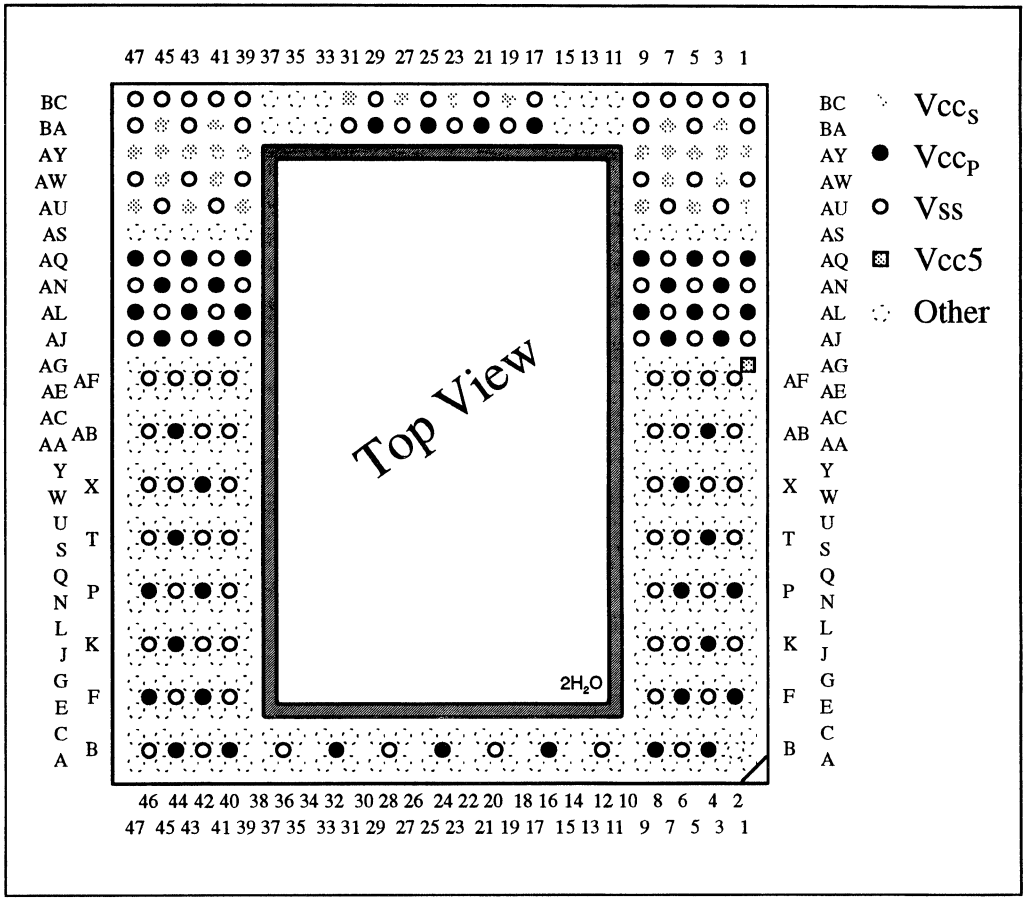


Figure 44. Pentium® Pro Processor Top View with Power Pin Locations

7.2. Pinout

Table 26 is the pin listing in pin number order. Table 27 is the pin listing in pin name order. Please see Section 3.8. to determine a signal's I/O type. Bus signals are described in Appendix A and the other pins are described in Section 3 and in Table 2.

Table 26. Pin Listing in Pin # Order

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
A1	$V_{REF}0$	B24	$V_{CC}P$	C47	D21#
A3	STPCLK#	B28	V_{SS}	E1	A29#
A5	TCK	B32	$V_{CC}P$	E3	A30#
A7	TRST#	B36	V_{SS}	E5	A32#
A9	IGNNE#	B40	$V_{CC}P$	E7	A33#
A11	A20M#	B42	V_{SS}	E9	A34#
A13	TDI	B44	$V_{CC}P$	E39	D22#
A15	FLUSH#	B46	V_{SS}	E41	D23#
A17	THERMTRIP#	C1	A35#	E43	D25#
A19	BCLK	C3	IERR#	E45	D24#
A21	RESERVED	C5	BERR#	E47	D26#
A23	TESTHI	C7	$V_{REF}1$	F2	$V_{CC}P$
A25	TESTHI	C9	FRCERR	F4	V_{SS}
A27	D1#	C11	INIT#	F6	$V_{CC}P$
A29	D3#	C13	TDO	F8	V_{SS}
A31	D5#	C15	TMS	F40	V_{SS}
A33	D8#	C17	FERR#	F42	$V_{CC}P$
A35	D9#	C19	PLL1	F44	V_{SS}
A37	D14#	C21	TESTLO	F46	$V_{CC}P$
A39	D10#	C23	PLL2	G1	A22#
A41	D11#	C25	D0#	G3	A24#
A43	D13#	C27	D2#	G5	A27#
A45	D16#	C29	D4#	G7	A26#
A47	$V_{REF}4$	C31	D6#	G9	A31#
B2	CPUPRES#	C33	D7#	G39	D27#
B4	$V_{CC}P$	C35	D12#	G41	D29#
B6	V_{SS}	C37	D15#	G43	D30#
B8	$V_{CC}P$	C39	D17#	G45	D28#
B12	V_{SS}	C41	D20#	G47	D31#
B16	$V_{CC}P$	C43	D18#	J1	A19#
B20	V_{SS}	C45	D19#	J3	A21#

Table 26. Pin Listing in Pin # Order

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
J5	A20#	N39	D44#	S45	D53#
J7	A23#	N41	D45#	S47	D50#
J9	A28#	N43	D47#	T2	V_{SS}
J39	D32#	N45	D42#	T4	V_{CCP}
J41	D35#	N47	D41#	T6	V_{SS}
J43	D38#	P2	V_{CCP}	T8	V_{SS}
J45	D33#	P4	V_{SS}	T40	V_{SS}
J47	D34#	P6	V_{CCP}	T42	V_{SS}
K2	V_{SS}	P8	V_{SS}	T44	V_{CCP}
K4	V_{CCP}	P40	V_{SS}	T46	V_{SS}
K6	V_{SS}	P42	V_{CCP}	U1	AP0#
K8	V_{SS}	P44	V_{SS}	U3	RSP#
K40	V_{SS}	P46	V_{CCP}	U5	BPRI#
K42	V_{SS}	Q1	A9#	U7	BNR#
K44	V_{CCP}	Q3	A7#	U9	BR3#
K46	V_{SS}	Q5	A5#	U39	DEP7#
L1	RESERVED	Q7	A8#	U41	$V_{REF}6$
L3	A16#	Q9	A10#	U43	D60#
L5	A15#	Q39	D51#	U45	D56#
L7	A18#	Q41	D52#	U47	D55#
L9	A25#	Q43	D49#	W1	SMI#
L39	D37#	Q45	D48#	W3	BR1#
L41	D40#	Q47	D46#	W5	REQ4#
L43	D43#	S1	A6#	W7	REQ1#
L45	D36#	S3	A4#	W9	REQ0#
L47	D39#	S5	A3#	W39	DEP2#
N1	A12#	S7	$V_{REF}2$	W41	DEP4#
N3	A14#	S9	AP1#	W43	D63#
N5	A11#	S39	D59#	W45	D61#
N7	A13#	S41	D57#	W47	D58#
N9	A17#	S43	D54#	X2	V_{SS}

Table 26. Pin Listing in Pin # Order

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
X4	V_{SS}	AB40	V_{SS}	AF46	V_{SS}
X6	$V_{CC}P$	AB42	V_{SS}	AG1	$V_{CC}5$
X8	V_{SS}	AB44	$V_{CC}P$	AG3	UP#
X40	V_{SS}	AB46	V_{SS}	AG5	RESERVED
X42	$V_{CC}P$	AC1	RESERVED	AG7	PWRGOOD
X44	V_{SS}	AC3	HIT#	AG9	RESERVED
X46	V_{SS}	AC5	BR0#	AG39	RESERVED
Y1	REQ3#	AC7	RP#	AG41	LINT1/NMI
Y3	REQ2#	AC9	RS0#	AG43	LINT0/INTR
Y5	DEFER#	AC39	BP3#	AG45	$V_{REF}7$
Y7	$V_{REF}3$	AC41	BPM0#	AG47	RESERVED
Y9	TRDY#	AC43	BINIT#	AJ1	V_{SS}
Y39	PRDY#	AC45	DEP0#	AJ3	$V_{CC}P$
Y41	RESET#	AC47	DEP3#	AJ5	V_{SS}
Y43	DEP1#	AE1	RESERVED	AJ7	$V_{CC}P$
Y45	DEP6#	AE3	ADS#	AJ9	V_{SS}
Y47	D62#	AE5	RS1#	AJ39	V_{SS}
AA1	BR2#	AE7	RS2#	AJ41	$V_{CC}P$
AA3	DRDY#	AE9	AERR#	AJ43	V_{SS}
AA5	DBSY#	AE39	TESTHI	AJ45	$V_{CC}P$
AA7	HITM#	AE41	PICD1	AJ47	V_{SS}
AA9	LOCK#	AE43	BP2#	AL1	$V_{CC}P$
AA39	BPM1#	AE45	RESERVED	AL3	V_{SS}
AA41	PICD0	AE47	$V_{REF}5$	AL5	$V_{CC}P$
AA43	PICCLK	AF2	V_{SS}	AL7	V_{SS}
AA45	PREQ#	AF4	V_{SS}	AL9	$V_{CC}P$
AA47	DEP5#	AF6	V_{SS}	AL39	$V_{CC}P$
AB2	V_{SS}	AF8	V_{SS}	AL41	V_{SS}
AB4	$V_{CC}P$	AF40	V_{SS}	AL43	$V_{CC}P$
AB6	V_{SS}	AF42	V_{SS}	AL45	V_{SS}
AB8	V_{SS}	AF44	V_{SS}	AL47	$V_{CC}P$

Table 26. Pin Listing in Pin # Order

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
AN1	V_{SS}	AU3	V_{SS}	BA5	V_{SS}
AN3	V_{CCP}	AU5	V_{CCS}	BA7	V_{CCS}
AN5	V_{SS}	AU7	V_{SS}	BA9	V_{SS}
AN7	V_{CCP}	AU9	V_{CCS}	BA11	RESERVED
AN9	V_{SS}	AU39	V_{CCS}	BA13	TESTLO
AN39	V_{SS}	AU41	V_{SS}	BA15	TESTLO
AN41	V_{CCP}	AU43	V_{CCS}	BA17	V_{CCP}
AN43	V_{SS}	AU45	V_{SS}	BA19	V_{SS}
AN45	V_{CCP}	AU47	V_{CCS}	BA21	V_{CCP}
AN47	V_{SS}	AW1	V_{SS}	BA23	V_{SS}
AQ1	V_{CCP}	AW3	V_{CCS}	BA25	V_{CCP}
AQ3	V_{SS}	AW5	V_{SS}	BA27	V_{SS}
AQ5	V_{CCP}	AW7	V_{CCS}	BA29	V_{CCP}
AQ7	V_{SS}	AW9	V_{SS}	BA31	V_{SS}
AQ9	V_{CCP}	AW39	V_{SS}	BA33	TESTLO
AQ39	V_{CCP}	AW41	V_{CCS}	BA35	RESERVED
AQ41	V_{SS}	AW43	V_{SS}	BA37	TESTLO
AQ43	V_{CCP}	AW45	V_{CCS}	BA39	V_{SS}
AQ45	V_{SS}	AW47	V_{SS}	BA41	V_{CCS}
AQ47	V_{CCP}	AY1	V_{CCS}	BA43	V_{SS}
AS1	VID0	AY3	V_{CCS}	BA45	V_{CCS}
AS3	VID1	AY5	V_{CCS}	BA47	V_{SS}
AS5	VID2	AY7	V_{CCS}	BC1	V_{SS}
AS7	VID3	AY9	V_{CCS}	BC3	V_{SS}
AS9	RESERVED	AY39	V_{CCS}	BC5	V_{SS}
AS39	TESTLO	AY41	V_{CCS}	BC7	V_{SS}
AS41	TESTLO	AY43	V_{CCS}	BC9	V_{SS}
AS43	TESTLO	AY45	V_{CCS}	BC11	RESERVED
AS45	TESTLO	AY47	V_{CCS}	BC13	TESTLO
AS47	RESERVED	BA1	V_{SS}	BC15	TESTLO
AU1	V_{CCS}	BA3	V_{CCS}	BC17	V_{SS}

Table 26. Pin Listing in Pin # Order

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
BC19	$V_{CC}S$	BC29	V_{SS}	BC39	V_{SS}
BC21	V_{SS}	BC31	$V_{CC}S$	BC41	V_{SS}
BC23	$V_{CC}S$	BC33	TESTLO	BC43	V_{SS}
BC25	V_{SS}	BC35	RESERVED	BC45	V_{SS}
BC27	$V_{CC}S$	BC37	TESTLO	BC47	V_{SS}

Table 27. Pin Listing in Alphabetic Order

Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #
A3#	S5	A33#	E7	D10#	A39
A4#	S3	A34#	E9	D11#	A41
A5#	Q5	A35#	C1	D12#	C35
A6#	S1	ADS#	AE3	D13#	A43
A7#	Q3	AERR#	AE9	D14#	A37
A8#	Q7	AP0#	U1	D15#	C37
A9#	Q1	AP1#	S9	D16#	A45
A10#	Q9	BCLK	A19	D17#	C39
A11#	N5	BERR#	C5	D18#	C43
A12#	N1	BINIT#	AC43	D19#	C45
A13#	N7	BNR#	U7	D20#	C41
A14#	N3	BP2#	AE43	D21#	C47
A15#	L5	BP3#	AC39	D22#	E39
A16#	L3	BPM0#	AC41	D23#	E41
A17#	N9	BPM1#	AA39	D24#	E45
A18#	L7	BPRI#	U5	D25#	E43
A19#	J1	BR0#	AC5	D26#	E47
A20#	J5	BR1#	W3	D27#	G39
A20M#	A11	BR2#	AA1	D28#	G45
A21#	J3	BR3#	U9	D29#	G41
A22#	G1	CPUPRES#	B2	D30#	G43
A23#	J7	D0#	C25	D31#	G47
A24#	G3	D1#	A27	D32#	J39
A25#	L9	D2#	C27	D33#	J45
A26#	G7	D3#	A29	D34#	J47
A27#	G5	D4#	C29	D35#	J41
A28#	J9	D5#	A31	D36#	L45
A29#	E1	D6#	C31	D37#	L39
A30#	E3	D7#	C33	D38#	J43
A31#	G9	D8#	A33	D39#	L47
A32#	E5	D9#	A35	D40#	L41

Table 27. Pin Listing in Alphabetic Order

Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #
D41#	N47	DEP6#	Y45	RESERVED	AE45
D42#	N45	DEP7#	U39	RESERVED	AG5
D43#	L43	DRDY#	AA3	RESERVED	AG9
D44#	N39	FERR#	C17	RESERVED	AG39
D45#	N41	FLUSH#	A15	RESERVED	AG47
D46#	Q47	FRCERR	C9	RESERVED	AS9
D47#	N43	HIT#	AC3	RESERVED	AS47
D48#	Q45	HITM#	AA7	RESERVED	BA11
D49#	Q43	IERR#	C3	RESERVED	BA35
D50#	S47	IGNNE#	A9	RESERVED	BC11
D51#	Q39	INIT#	C11	RESERVED	BC35
D52#	Q41	LINT0/INTR	AG43	RESET#	Y41
D53#	S45	LINT1/NMI	AG41	RP#	AC7
D54#	S43	LOCK#	AA9	RS0#	AC9
D55#	U47	PICCLK	AA43	RS1#	AE5
D56#	U45	PICD0	AA41	RS2#	AE7
D57#	S41	PICD1	AE41	RSP#	U3
D58#	W47	PLL1	C19	SMI#	W1
D59#	S39	PLL2	C23	STPCLK#	A3
D60#	U43	PRDY#	Y39	TCK	A5
D61#	W45	PREQ#	AA45	TDI	A13
D62#	Y47	PWRGOOD	AG7	TDO	C13
D63#	W43	REQ0#	W9	TESTHI	A23
DBSY#	AA5	REQ1#	W7	TESTHI	A25
DEFER#	Y5	REQ2#	Y3	TESTHI	AE39
DEP0#	AC45	REQ3#	Y1	TESTLO	C21
DEP1#	Y43	REQ4#	W5	TESTLO	AS39
DEP2#	W39	RESERVED	A21	TESTLO	AS41
DEP3#	AC47	RESERVED	L1	TESTLO	AS43
DEP4#	W41	RESERVED	AC1	TESTLO	AS45
DEP5#	AA47	RESERVED	AE1	TESTLO	BA13

Table 27. Pin Listing in Alphabetic Order

Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #
TESTLO	BA15	$V_{CC}P$	T44	$V_{CC}S$	AU9
TESTLO	BA33	$V_{CC}P$	X6	$V_{CC}S$	AU39
TESTLO	BA37	$V_{CC}P$	X42	$V_{CC}S$	AU43
TESTLO	BC13	$V_{CC}P$	AB4	$V_{CC}S$	AU47
TESTLO	BC15	$V_{CC}P$	AB44	$V_{CC}S$	AW3
TESTLO	BC33	$V_{CC}P$	AJ3	$V_{CC}S$	AW7
TESTLO	BC37	$V_{CC}P$	AJ7	$V_{CC}S$	AW41
THERMTRIP#	A17	$V_{CC}P$	AJ41	$V_{CC}S$	AW45
TMS	C15	$V_{CC}P$	AJ45	$V_{CC}S$	AY1
TRDY#	Y9	$V_{CC}P$	AL1	$V_{CC}S$	AY3
TRST#	A7	$V_{CC}P$	AL5	$V_{CC}S$	AY5
UP#	AG3	$V_{CC}P$	AL9	$V_{CC}S$	AY7
$V_{CC}5$	AG1	$V_{CC}P$	AL39	$V_{CC}S$	AY9
$V_{CC}P$	B4	$V_{CC}P$	AL43	$V_{CC}S$	AY39
$V_{CC}P$	B8	$V_{CC}P$	AL47	$V_{CC}S$	AY41
$V_{CC}P$	B16	$V_{CC}P$	AN3	$V_{CC}S$	AY43
$V_{CC}P$	B24	$V_{CC}P$	AN7	$V_{CC}S$	AY45
$V_{CC}P$	B32	$V_{CC}P$	AN41	$V_{CC}S$	AY47
$V_{CC}P$	B40	$V_{CC}P$	AN45	$V_{CC}S$	BA3
$V_{CC}P$	B44	$V_{CC}P$	AQ1	$V_{CC}S$	BA7
$V_{CC}P$	F2	$V_{CC}P$	AQ5	$V_{CC}S$	BA41
$V_{CC}P$	F6	$V_{CC}P$	AQ9	$V_{CC}S$	BA45
$V_{CC}P$	F42	$V_{CC}P$	AQ39	$V_{CC}S$	BC19
$V_{CC}P$	F46	$V_{CC}P$	AQ43	$V_{CC}S$	BC23
$V_{CC}P$	K4	$V_{CC}P$	AQ47	$V_{CC}S$	BC27
$V_{CC}P$	K44	$V_{CC}P$	BA17	$V_{CC}S$	BC31
$V_{CC}P$	P2	$V_{CC}P$	BA21	VID0	AS1
$V_{CC}P$	P6	$V_{CC}P$	BA25	VID1	AS3
$V_{CC}P$	P42	$V_{CC}P$	BA29	VID2	AS5
$V_{CC}P$	P46	$V_{CC}S$	AU1	VID3	AS7
$V_{CC}P$	T4	$V_{CC}S$	AU5	$V_{REF}0$	A1

Table 27. Pin Listing in Alphabetic Order

Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #
$V_{REF}1$	C7	V_{SS}	T40	V_{SS}	AL41
$V_{REF}2$	S7	V_{SS}	T42	V_{SS}	AL45
$V_{REF}3$	Y7	V_{SS}	T46	V_{SS}	AN1
$V_{REF}4$	A47	V_{SS}	X2	V_{SS}	AN5
$V_{REF}5$	AE47	V_{SS}	X4	V_{SS}	AN9
$V_{REF}6$	U41	V_{SS}	X8	V_{SS}	AN39
$V_{REF}7$	AG45	V_{SS}	X40	V_{SS}	AN43
V_{SS}	B6	V_{SS}	X44	V_{SS}	AN47
V_{SS}	B12	V_{SS}	X46	V_{SS}	AQ3
V_{SS}	B20	V_{SS}	AB2	V_{SS}	AQ7
V_{SS}	B28	V_{SS}	AB6	V_{SS}	AQ41
V_{SS}	B36	V_{SS}	AB8	V_{SS}	AQ45
V_{SS}	B42	V_{SS}	AB40	V_{SS}	AU3
V_{SS}	B46	V_{SS}	AB42	V_{SS}	AU7
V_{SS}	F4	V_{SS}	AB46	V_{SS}	AU41
V_{SS}	F8	V_{SS}	AF2	V_{SS}	AU45
V_{SS}	F40	V_{SS}	AF4	V_{SS}	AW1
V_{SS}	F44	V_{SS}	AF6	V_{SS}	AW5
V_{SS}	K2	V_{SS}	AF8	V_{SS}	AW9
V_{SS}	K6	V_{SS}	AF40	V_{SS}	AW39
V_{SS}	K8	V_{SS}	AF42	V_{SS}	AW43
V_{SS}	K40	V_{SS}	AF44	V_{SS}	AW47
V_{SS}	K42	V_{SS}	AF46	V_{SS}	BA1
V_{SS}	K46	V_{SS}	AJ1	V_{SS}	BA5
V_{SS}	P4	V_{SS}	AJ5	V_{SS}	BA9
V_{SS}	P8	V_{SS}	AJ9	V_{SS}	BA19
V_{SS}	P40	V_{SS}	AJ39	V_{SS}	BA23
V_{SS}	P44	V_{SS}	AJ43	V_{SS}	BA27
V_{SS}	T2	V_{SS}	AJ47	V_{SS}	BA31
V_{SS}	T6	V_{SS}	AL3	V_{SS}	BA39
V_{SS}	T8	V_{SS}	AL7	V_{SS}	BA43

Table 27. Pin Listing in Alphabetic Order

Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #
V_{SS}	BA47	V_{SS}	BC9	V_{SS}	BC39
V_{SS}	BC1	V_{SS}	BC17	V_{SS}	BC41
V_{SS}	BC3	V_{SS}	BC21	V_{SS}	BC43
V_{SS}	BC5	V_{SS}	BC25	V_{SS}	BC45
V_{SS}	BC7	V_{SS}	BC29	V_{SS}	BC47

8.0. OVERDRIVE® PROCESSOR SOCKET SPECIFICATION

8.1. Introduction

Intel will offer future OverDrive processors for the Pentium Pro processor. This OverDrive processor will be based on a faster, future Intel processor core.

The future OverDrive processor for Pentium Pro processor-based systems is a processor upgrade that will make all software run faster on an existing Pentium Pro processor system. The OverDrive processor is binary compatible with the Pentium Pro processor. The OverDrive processor is intended for use as a replacement upgrade for single and dual processor Pentium Pro processor designs. The OverDrive processor will be equipped with an integral fan/heatsink and retention clips. Intel plans to ship OverDrive processors with a matched Voltage Regulator Module (OverDrive processor VRM).

To support processor upgrades, a Zero Insertion Force (ZIF) socket (Socket 8) and a Voltage Regulator Module connector (Header 8) have been defined along with the Pentium Pro processor. Header 8 can be populated with an OEM Pentium Pro processor VRM or with the OverDrive processor VRM which Intel plans to ship with the OverDrive processor as part of the retail package.

The OverDrive processor will also support Voltage Identification as described in Section 3.6. The four Voltage ID outputs (VID0-VID3) can be used to design a programmable power supply that will meet the power requirements of both the Pentium Pro and OverDrive processors via the Header 8 described in this section, or on the motherboard. If you plan to use VID to design a programmable supply for the OverDrive processor, please contact Intel for additional information.

A single socket system should include Socket 8 and Header 8. When this system configuration is upgraded, the Pentium Pro processor and its VRM are replaced with a future OverDrive processor for Pentium Pro processor-based systems and its matching OverDrive processor VRM. The OverDrive processor VRM is capable of delivering the lower voltage and higher current required by the upgrade. Other voltage regulation configurations are described in Section 8.3.2.

8.1.1. TERMINOLOGY

Header 8: 40-pin Voltage Regulator Module (VRM) connector defined to contain the OEM VRM and OverDrive processor VRM.

OverDrive processor: A future OverDrive processor for Pentium Pro processor-based systems.

OverDrive processor VRM: A VRM designed to provide the specific voltage required by the future OverDrive processor for Pentium Pro processor-based systems.

Socket 8: 387-pin SPGA Zero Insertion Force (ZIF) socket defined to contain either a Pentium Pro or OverDrive processor.

8.2. Mechanical Specifications

This section specifies the mechanical features of Socket 8 and Header 8. This section includes the pinout, surrounding space requirements, and standardized clip attachment features.

Figure 45 shows a mechanical representation of the OverDrive processor in Socket 8 and the OverDrive processor VRM in Header 8.

8.2.1. VENDOR CONTACTS FOR SOCKET 8 AND HEADER 8

Contact your local Intel representative for a list of participating Socket 8 and Header 8 suppliers.

8.2.2. SOCKET 8 DEFINITION

Socket 8 is a 387-pin, modified staggered pin grid array (SPGA), Zero Insertion Force (ZIF) socket. Thepinout is identical to the Pentium Pro processor.

The pinout is identical to the Pentium Pro processor. Two pins are used to support the on-package fan/heatsink included on the OverDrive processor and indicate the presence of the OverDrive processor. The OverDrive processor package is oriented in Socket 8 by the asymmetric use of interstitial pins. Standardized heat sink clip attachment tabs are also defined as part of Socket 8 (Section 8.2.2.3.).

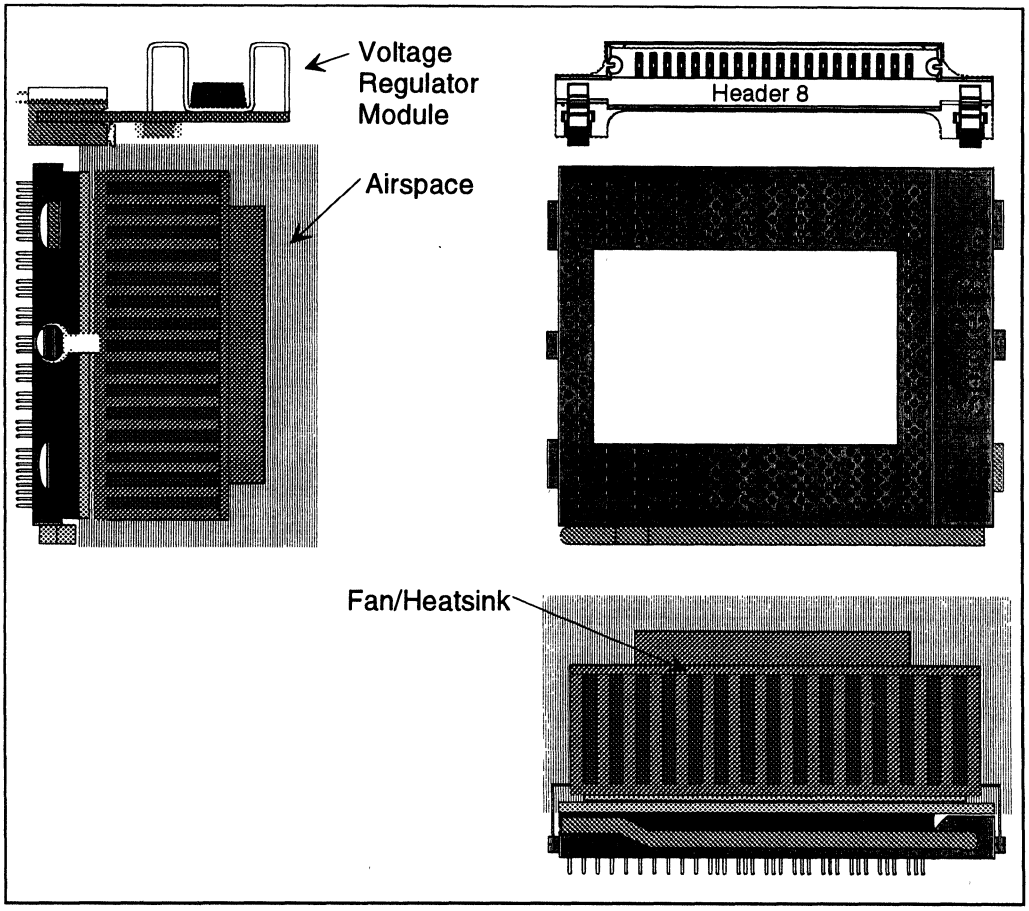


Figure 45. Socket 8 Shown with the Fan/heatsink Cooling Solution, Clip Attachment Features and Adjacent Voltage Regulator Module

8.2.2.1. Socket 8 Pinout

Socket 8 is shown in Figure 46 along with the VRM (Header 8) connector. Refer to Section 7, for pin listings of the Pentium Pro processor. The OverDrive processor pinout is identical to the Pentium Pro processor pinout.

Descriptions of the upgrade specific pins are presented in Table 28. Note the location of pin A1 in relation to the cam shelf position. If the socket has the cam shelf located in a different position, then correct insertion of the OverDrive processor may not be possible. See Section 8.2.2.2. for space requirements.

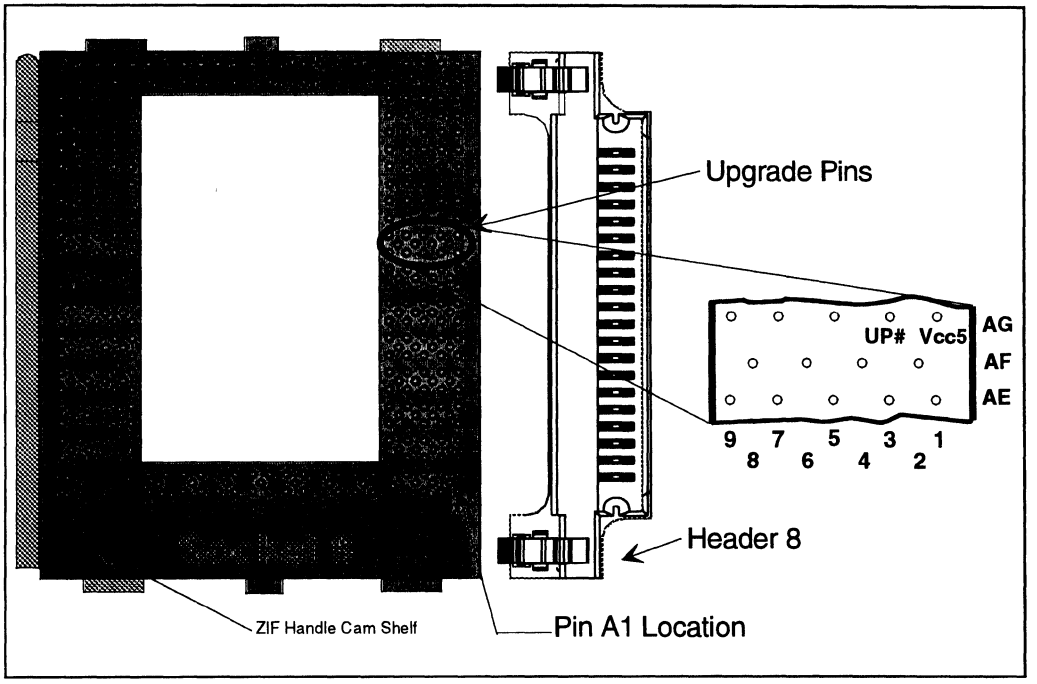


Figure 46. OverDrive® Processor Pinout

Table 28. OverDrive® Processor Signal Descriptions

Pin Name	Pin #	I/O	Function
$V_{CC}5$	AG1	Input	+5V Supply required for OverDrive processor fan/heatsink.
UP#	AG3	Output	This output is tied to V_{SS} in the OverDrive processor to indicate the presence of an upgrade processor. This output is an open in the Pentium® Pro processor.

NOTES:

1. Refer to Section 8.3. for a functional description of the above signals.

8.2.2.2. Socket 8 Space Requirements

The OverDrive processor will be equipped with a fan/heatsink thermal management device. The package envelope dimensions for the OverDrive processor with attached fan/heatsink are shown in Figure 47. Clearance is required around the fan/heatsink to ensure unimpeded air flow for proper cooling (refer to Section 8.5.1.1. for details). Figure 48 shows the Socket 8 space requirements for the OverDrive processor. All dimensions are in inches.

"Keep out zones," also shown in Figure 48, have been established around the heat sink clip attachment tabs to prevent damage to surface mounted components during clip installation and removal. The keep out zones extend upwards from the surface of the motherboard to the top of the heat sink. The lateral limits of the keep out zones extend 0.1 inch from the perimeter of each tab.

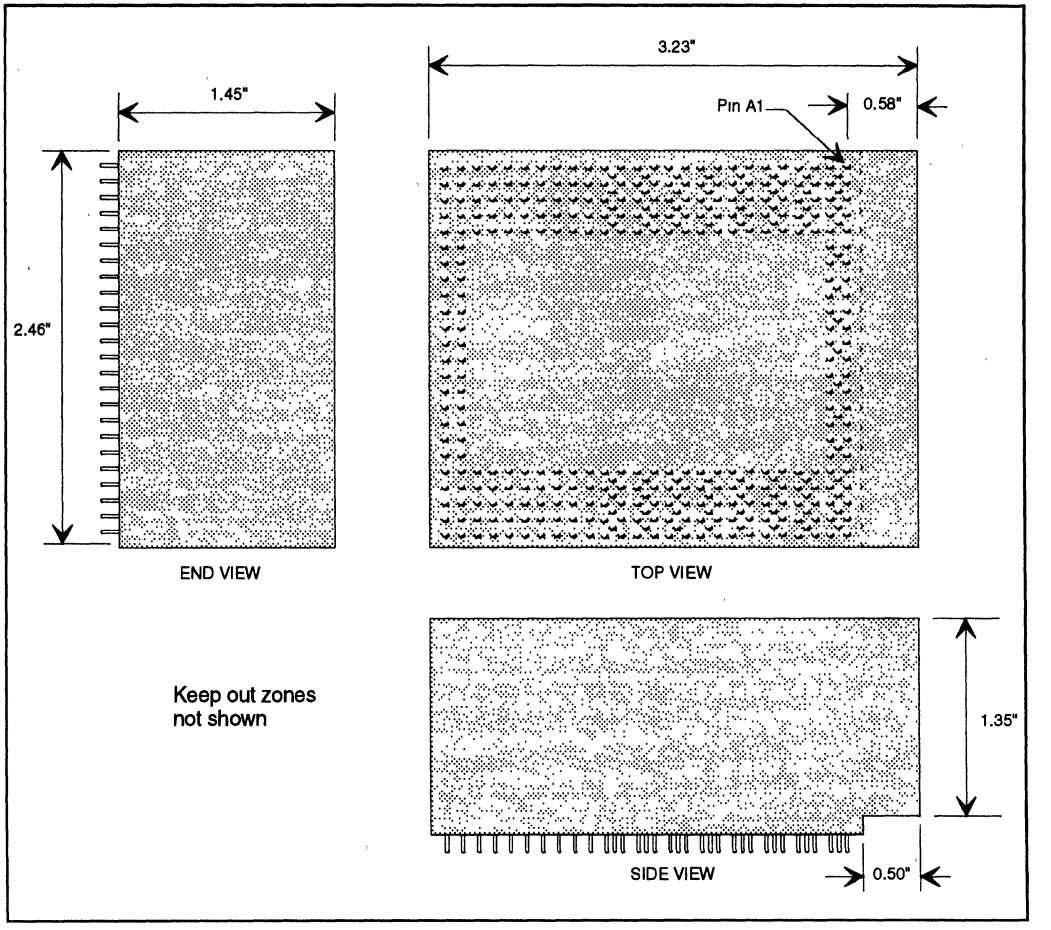


Figure 47. OverDrive® Processor Envelope Dimensions

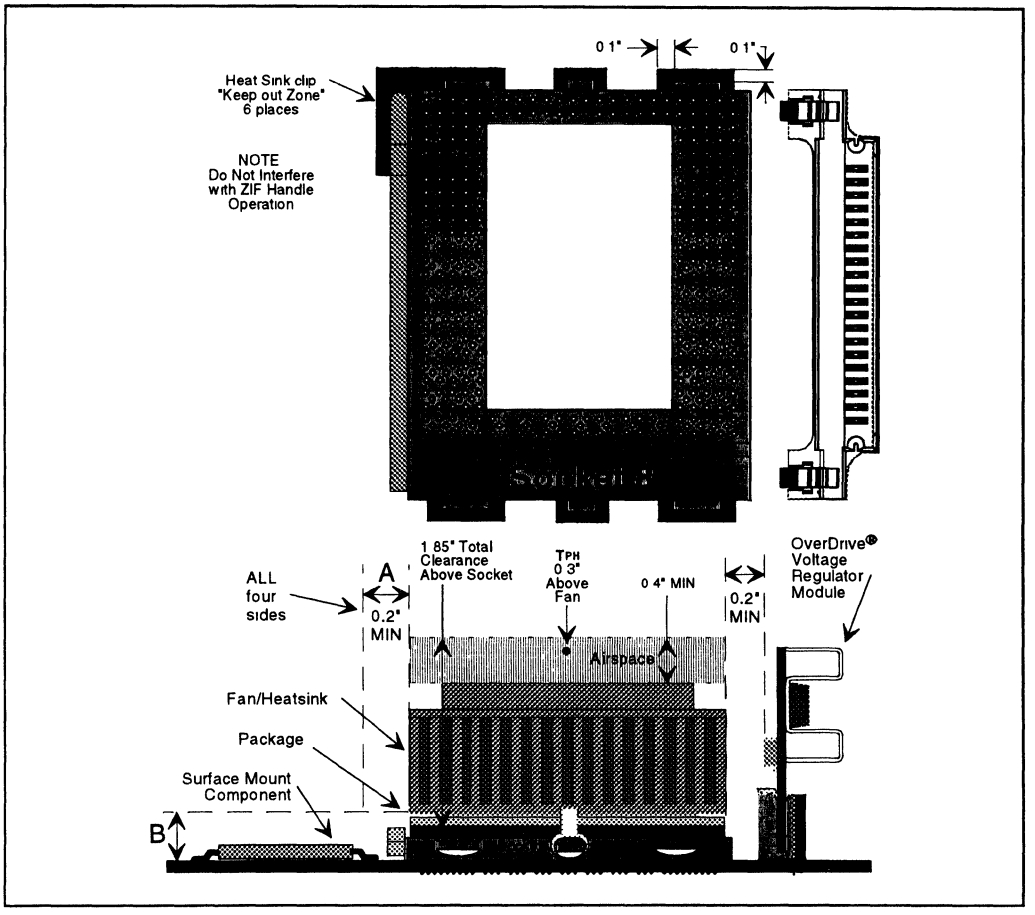


Figure 48. Space Requirements for the OverDrive® Processor

Immovable objects must not be located less than 1.85 inches above the seating plane of the ZIF socket. Removable objects must also not be located less than the 1.85 inches above the seating plane of the ZIF socket required for the processor and fan/heatsink. These requirements also apply to the area above the cam shelf.

As shown in Figure 48 it is acceptable to allow any device (i.e. add-in cards, surface mount device, chassis etc.) to enter within the free space distance of 0.2" from the chip package if it is not taller than the level of the heat sink base. In other words, if a component is taller than height 'B', it cannot be closer to the chip package than distance 'A'. This applies to all four sides of the chip package (the handle side of the ZIF socket will generally meet this specification since its width is typically larger than distance 'A' (0.2")).

For designs which use Header 8, the header itself can violate the 0.2" airspace around the OverDrive processor package. A VRM (either Pentium Pro processor VRM or OverDrive processor VRM), once installed in Header 8, and any components on the module, MUST NOT violate the 0.2" airspace. Also, the header must not interfere with the installation of the Pentium Pro or OverDrive processors, and must not interfere with the operation of the ZIF socket lever. Alternately,

Socket 8, and the installed processor must not interfere with the installation and removal of a VRM in Header 8.

NOTE

Components placed close to Socket 8 must not impede access to and operation of the handle of the ZIF socket lever. Adequate clearance must be provided within the proximity of the ZIF socket lever to provide fingertip access to the lever for normal operation, and to allow raising the lever to the full open position.

8.2.2.3. Socket 8 Clip Attachment Tabs

Standardized clip attachment tabs will be provided on Socket 8. These will allow clips to secure the OverDrive processor to the socket to enhance shock and vibration protection. OEMs may utilize the attachment tabs for their own thermal solutions. As an option, OEMs may use customized attachment features providing that the additional features do not interfere with the standard tabs used by the upgrade.

Details of the clip attachment tabs and overall dimensions of Intel qualified sockets may be obtained from participating socket suppliers.

8.2.3. OVERDRIVE® PROCESSOR VOLTAGE REGULATOR MODULE DEFINITION

Header 8 is a 2-row, 40-pin shrouded header designed to accommodate a Pentium Pro processor VRM, OverDrive processor VRM, or a programmable VRM. The OverDrive processor VRM is used to convert the standard 5.0V supply to the OverDrive processor core operating voltage. Integral OverDrive processor VRM hold down tabs are included as part of the header definition for enhanced shock and vibration protection.

OEMs who plan to design a custom VRM PC Board to fit into Header 8 should refer to the AP-523, *Pentium® Pro Processor Power Distribution Guidelines* Application Note (Order Number 242764).

8.2.3.1. OverDrive® Processor VRM Requirement

When upgrading with an OverDrive processor, Intel suggests the use of its matched Voltage Regulator Module, which Intel plans to ship with the OverDrive processor retail package.

If the OEM includes on-board voltage regulation and the Header 8 for the OverDrive processor VRM, the on-board voltage regulator must be shut off via the UP# output of the CPU. When the OverDrive processor is installed, and the UP# signal is driven LOW, the on-board VR must never power on. This will ensure that there is no contention between the OverDrive processor VRM and the on-board regulator.

8.2.3.2. OverDrive® Processor VRM Location

It is recommended that Header 8 be located within approximately 1 inch of Socket 8 to facilitate end user installation. For optimum electrical performance, the Header 8 should be as close as possible to Socket 8. The location must not interfere with the operation of the ZIF socket handle or heatsink attachment clips. To allow system design flexibility, Header 8 placement is optional, but it is recommended that Header 8 NOT be placed on the same side of the ZIF socket as the handle.

8.2.3.3. OverDrive® Processor VRM Pinout

The OverDrive processor VRM pinout and pin description is presented in Figure 49 and Table 29, respectively.

8.2.3.4. OverDrive® Processor VRM Space Requirements

Figure 50 describes the maximum OverDrive processor VRM envelope. No part of the OverDrive processor VRM will extend beyond the defined space.

Pin #	Signal Name	Pin #	Signal Name
A1	5Vin	B1	5Vin
A2	5Vin	B2	5Vin
A3	5Vin	B3	5Vin
A4	12Vin	B4	12Vin
A5	Reserved	B5	Reserved
A6	Reserved	B6	OUTEN
A7	VID0	B7	VID1
A8	VID2	B8	VID3
A9	UP#	B9	PwrGood
A10	V_{CCP}	B10	Vss
A11	Vss	B11	V_{CCP}
A12	V_{CCP}	B12	Vss
A13	Vss	B13	V_{CCP}
A14	V_{CCP}	B14	Vss
A15	Vss	B15	V_{CCP}
A16	V_{CCP}	B16	Vss
A17	Vss	B17	V_{CCP}
A18	V_{CCP}	B18	Vss
A19	Vss	B19	V_{CCP}
A20	V_{CCP}	B20	Vss

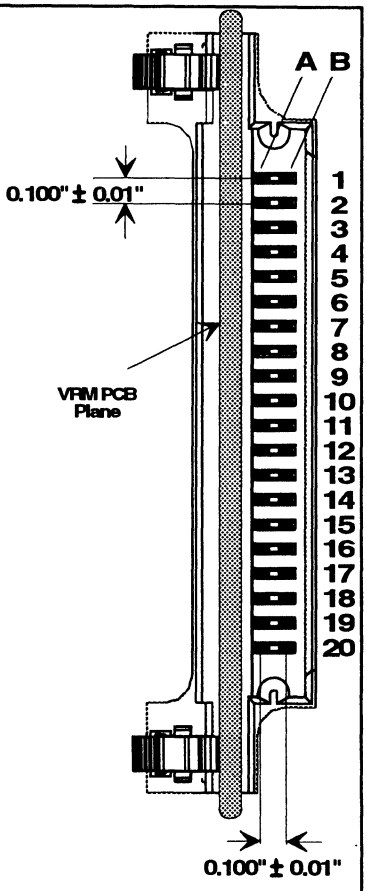


Figure 49. Header 8 Pinout

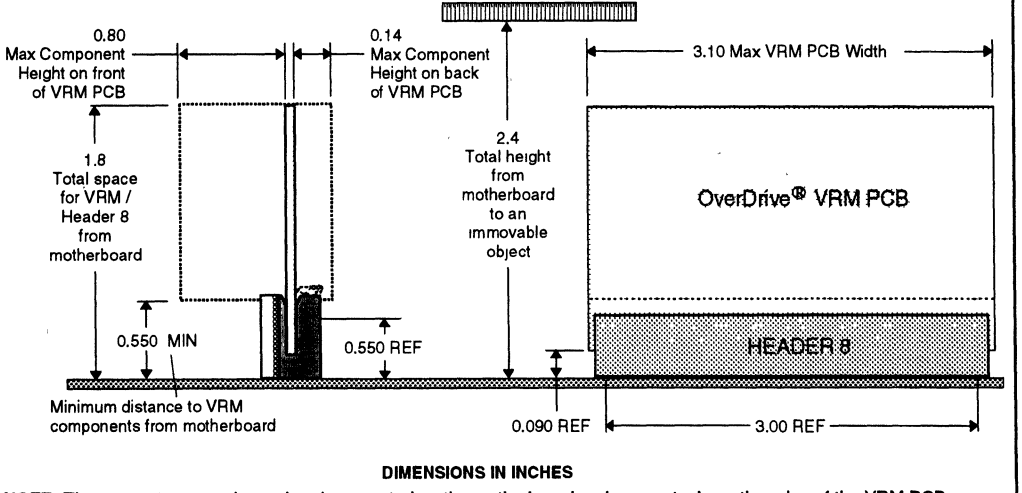
Table 29. Header 8 Pin Reference

Pin Name	I/O	Usage	Function
12Vin	Input	Required	+12V±5% Supply
5Vin	Input	Required	+5V±5% Supply [1]
V_{SS}	Input	Required	Ground Reference
OUTEN	Input	Optional	When driven high this input will enable the OEM VRM output and float the OverDrive® processor VRM output. When this input is driven low, the output of the OEM module will float and the OverDrive processor VRM output

			will be enabled.
PWRGOOD	Output	Optional	Power Good is driven high upon the VRM output reaching valid levels. This output requires an external pull-up resistor (~10KΩ).
RES		No connect	Reserved for future use.
UP#	Input	Required	This signal is held high via an external pull-up resistor on the open collector output of the Pentium® Pro processor, and is driven low by the grounded output of the OverDrive processor.
$V_{CC}P$	Output	Required	Voltage Regulator Module core voltage output. Voltage level for the OverDrive processor will be lower than for the Pentium Pro processor.
VID3-VID0	Inputs	Optional	Used by the Pentium Pro processor VRM to determine what output voltage to provide to the CPU. The OverDrive processor VRM does not require these pins to be connected as it will be voltage matched in advance to the OverDrive processor. Refer to Table 1 for Voltage ID pin decoding.

NOTES:

1. The OverDrive® processor Voltage Regulator Module requires both 5V and 12V. Routing for the 5V VRM supply must support the full requirements of the OverDrive processor VRM given in Table 32 even if the 12V supply is utilized for the OEM VRM.



NOTE: The connector comprises a header mounted on the motherboard and a receptacle on the edge of the VRM PCB.

Figure 50. OverDrive® Processor Voltage Regulator Module Envelope

8.3. Functional Operation of OverDrive® Processor Signals

8.3.1. FAN/HEATSINK POWER ($V_{CC}5$)

This 5V supply provides power to the fan of the fan/heatsink assembly. See Table 31 for $V_{CC}5$ specifications.

8.3.2. UPGRADE PRESENT SIGNAL (UP#)

The Upgrade Present signal is used to prevent operation of voltage regulators providing a potentially harmful voltage to the OverDrive processor, and to prevent contention between on-board regulation and the OverDrive processor VRM. UP# is an open collector output, held high using a pull-up resistor on the motherboard tied to +5 Volts.

There are several system voltage regulation design options to support both the Pentium Pro processor and its OverDrive processor. The use of the UP# signal for each case is described below:

— Case 1: *Header 8 only*

If the system is designed with voltage regulation from the Header 8 only, then the UP# signal must be connected between the CPU socket (Socket 8) and the VRM connector (Header 8). The Pentium Pro processor VRM should internally connect the UP# input directly to the VRM OUTEN input. If the Pentium Pro processor is replaced with an OverDrive processor and the OEM VRM is NOT replaced with the OverDrive processor VRM, the original voltage regulator will never enable its outputs because the lower voltage OverDrive processor could be damaged. Refer to Figure 51.

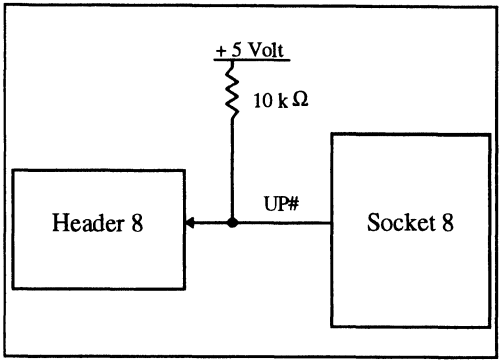


Figure 51. Upgrade Presence Detect Schematic—Case 1

— Case 2: *Header 8 AND alternate voltage source*

If the system is designed with alternate voltage source and a Header 8 for future upgrade support, then the UP# signal must be connected between Socket 8, Header 8, and the alternate voltage source. The Pentium Pro processor voltage regulator should use the UP# signal to disable the voltage output when detected low (indicating that an OverDrive processor has been installed). The OverDrive processor VRM, when installed into the Header 8 will use the UP# signal to enable its outputs (when detected low). When the Pentium Pro processor is replaced with an OverDrive processor and the OverDrive processor VRM is installed, the original voltage regulator must never enable its outputs because the lower voltage OverDrive processor could be damaged. Refer to Figure 52.

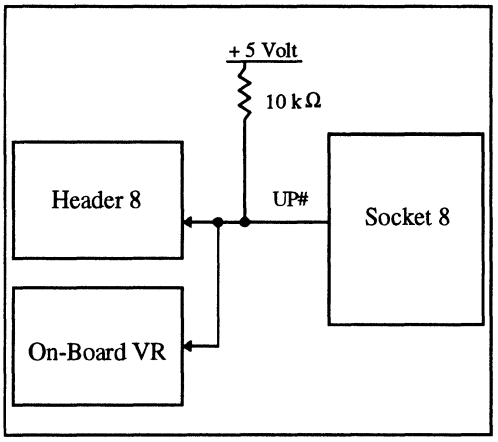


Figure 52. Upgrade Presence Detect Schematic—Case 2

— Case 3: *Alternate voltage source only*

If the system is designed with only a programmable voltage source using the VID3-VID0 pins, then the UP# signal need not be used.

NOTE

The programmable voltage source needs to be able to provide the OverDrive processor with its required power. Refer to Figure 53.

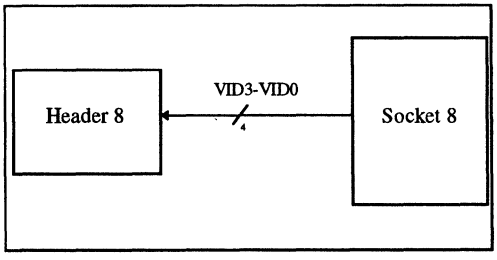


Figure 53. Upgrade Presence Detect Schematic—Case 3

8.3.3. BIOS CONSIDERATIONS

Please refer to the *Pentium® Pro Processor Developers Manual: Volume 3, Programmer's Reference Manual* (Order Number 242691) for test information for BIOS requirements.

It is the responsibility of the BIOS to detect the type of CPU in the system and program the support hardware accordingly. In most cases, the BIOS does this by reading the CPU signature, comparing it to known signatures, and, upon finding a match, executing the corresponding hardware initialization code.

The CPUID instruction is used to determine several processor parameters. Following execution of the CPUID instruction, bits 12 and 13 of the EAX register can be used to determine if the processor is an OEM or an OverDrive processor. An OverDrive processor is present if bit 13=0 and bit 12=1.

NOTE

Contact your BIOS vendor to ensure that the above requirements have been included.

8.3.3.1. OverDrive® Processor CPUID

Following power-on RESET or the CPUID instruction, the EAX register contains the values shown in Table 30.

Table 30. OverDrive® Processor CPUID

Type [13:12]	Family [11:8]	Model [7:4]	Stepping [3:0]
1	6	3	X

8.3.3.2. Common Causes of Upgradability Problems Due to BIOS

CPU signature detection has been a common cause of current upgradability problems due to BIOS. A few precautions within the BIOS can help to eliminate future upgradability problems with Pentium Pro processor-based systems. When programming or modifying a BIOS, be aware of the impact of future OverDrive processors. The following recommendations should prevent problems in the future:

- Always use the CPU signature and feature flags to identify the processor, including future processors.
- Never use timing loops for delays.
- If an OverDrive processor is detected, report the presence of an "OverDrive processor" to the end-user.

- If an OverDrive processor is detected, don't test on-chip cache sizes or organization. The OverDrive processor cache parameters differ from those of the Pentium Pro processor.
- If an OverDrive processor is detected, don't use the Pentium Pro processor model specific registers and test registers. OverDrive processor MSRs differ from those of the Pentium Pro processor.

8.4. OverDrive® Processor Electrical Specifications

This section describes the electrical requirements for the OverDrive processor.

NOTE

ZIF socket electrical parameters may differ from LIF socket parameters; therefore, be sure to use the appropriate ZIF socket parameters for electrical design simulations.

8.4.1. DC SPECIFICATIONS

8.4.1.1. OverDrive® Processor DC Specifications

Table 31 lists the DC specifications for the OverDrive processor that are either different from or in addition to the Pentium Pro processor specifications.

8.4.1.2. OverDrive® Processor VRM DC Specifications

The DC specifications for the OverDrive processor VRM are presented in Table 32.

Table 31. OverDrive® Processor DC Specifications

Symbol	Parameter	Min	Typ	Max	Unit	Notes
$I_{CC}P$	Primary I_{CC} Current	0.100		11.2	A	1
$V_{CC}P$	Primary V_{CC} Voltage	2.375	2.5	2.625		$V_{CC}P$ = 2.5V±5% 2
$I_{CC}S$	Secondary I_{CC} Current			0	A	
VCCS	Secondary V_{CC} Voltage	3.145	3.3	3.465		$V_{CC}S$ = 3.3V ±5%
$I_{CC}5$	Fan/heatsink Current			340	mA	
$V_{CC}5$	Fan/heatsink Voltage	4.75	5	5.25		$V_{CC}5$ = 5V ± 5%
P_{MAX}	Maximum Thermal Design Power			26.7	W	1

NOTES:

1. This specification applies to the future OverDrive® processor for 150 MHz Pentium® Pro processor-based systems.
2. This is the TARGET OverDrive processor Voltage. It is recommended that the Voltage Identification be used to determine processor voltage for programmable voltage sources and implement a voltage range which adequately covers the OverDrive processor Target Voltage (~2.4-2.7V).

Table 32. OverDrive® Processor VRM Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL}	Control Signal Input Low Voltage	-0.3	0.8	V	
V_{IH}	Control Signal Input High Voltage	2.0	$V_{CC}5$+0.3	V	
V_{OL}	Control Signal Output Low Voltage		0.4V	V	
$V_{OH}5$	Control Signal Output High Voltage	2.4	$V_{CC}5$+0.3	V	PWRGOOD
$I_{CC}5$	5.0V Power Supply Current	0.100	7.0	A	VRM input current
$I_{CC}12$	12.0V Power Supply Current		150	mA	VRM input current
I_{OUT}	VRM Output Current		11.2	A	
L_{MB}	Total inductance between VRM output and processor pins		2.5	nH	
R_{MB}	Total resistance between VRM output and processor pins		2.1	mW	1
di_{CC}/dt	Worst Case Input ($I_{CC}5$) Load Change		100	mA/ µS	
T_{VOUT}	Valid Input Supply to Output Delay		10	ms	

NOTES:

1. Maximum total resistance from VRM output to CPU pins cannot exceed 2.1 mΩ. For example, a breakdown of the resistive path might be 0.45 mΩ for VRM header, 1.0 mW for motherboard power plane resistance, and 0.65 mΩ for ZIF socket.

8.4.2. OVERDRIVE® PROCESSOR DECOUPLING REQUIREMENTS

No additional decoupling capacitance is required to support the OverDrive processor beyond what is necessary for the Pentium Pro processor. Any incremental decoupling, both bulk and high speed, required by the OverDrive processor will be provided on the processor package. It is strongly recommended that liberal, low inductance decoupling capacitance be placed near Socket 8 following the guidelines in Note 1 of Table 4 and the AP-523, *Pentium® Pro Processor Power Distribution Guidelines* Application Note (Order Number 242764). Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

8.4.3. AC SPECIFICATIONS

Except for internal CPU core Clock frequency, the OverDrive processor will operate within the same AC specifications as the Pentium Pro processor.

8.5. Thermal Specifications

This section describes the cooling solution utilized by the OverDrive processor and the cooling requirements for both the processor and VRM. Heat dissipation by the OverDrive processor will be no greater than the Pentium Pro processor, as described in Section 6.

8.5.1. OVERDRIVE® PROCESSOR COOLING REQUIREMENTS

The OverDrive processor will be cooled with a fan/heatsink cooling solution. The OverDrive

processor will operate properly when the preheat temperature, T_{PH}, is a maximum of 50°C (T_{PH} is the temperature of the air entering the fan/heatsink, measured 0.3" above the center of the fan — See Figure 48). When the preheat temperature requirement is met, the fan/heatsink will keep the case temperature, T_C, within the specified range, provided airflow through the fan/heatsink is unimpeded (see Space Requirements, Section 8.2.2.2.).

It is strongly recommended that testing be conducted to determine if the fan inlet temperature requirement is met at the system maximum ambient operating temperature.

NOTE

The OverDrive processor will operate properly when the preheat temperature, T_{PH}, is a maximum of 50°C (T_{PH} is the temperature of the air entering the fan/heatsink, measured 0.3" above the center of the fan — See Figure 48.)

8.5.1.1. Fan/Heatsink Cooling Solution

A height of 0.4" airspace above the fan/heatsink unit and a distance of 0.2" around all four sides of the OverDrive processor is REQUIRED to ensure that the airflow through the fan/heatsink is not blocked. The fan/heatsink will reside within the boundaries of the surface of the chip. Blocking the airflow to the fan/heatsink reduces the cooling efficiency and decreases fan life. Figure 48 illustrates an acceptable airspace clearance above the fan/heatsink and around the OverDrive processor package.

8.5.2. OEM PROCESSOR COOLING REQUIREMENTS

The OEM processor cooling solution must not impede the upgradability of the system. For example:

- If an OEM fan/heatsink is used, then electrical connections between the OEM fan/heatsink and system must be through an end user separable connector.
- If an OEM fan/heatsink is used, removal of the assembly must not interfere with the operation of the OverDrive processor, for example, by activating cooling failure protection mechanisms employed by the OEM.
- Custom attachment features in addition to the features covered in Section 8.2.2.3. must not interfere with attachment of the upgrade retention clips.

8.5.3. OVERDRIVE® PROCESSOR VRM COOLING REQUIREMENTS

The OverDrive processor Voltage Regulator Module will be shipped with a passive heat sink. Voltage regulator case temperature must not exceed 105°C. The ambient temperature, T_A, required to properly cool the VRM can be estimated from the following section.

8.5.4. THERMAL EQUATIONS AND DATA

The OverDrive processor Voltage Regulator Module requires that T_C does not exceed 105°C. T_C is measured on the surface of the hottest component of the VRM. To calculate T_A values for the VRMs at different flow rates, the following equations and data may be used:

$$T_A = T_C - (P \times \Theta_{CA})$$

Where,

T_A and T_C =	Ambient and Case temperature, respectively. (°C)
Θ_{CA} =	Case-to-Ambient Thermal Resistance (°C/Watt)
P =	Maximum Power Consumption (Watt)

Table 33. Upgrade Power Dissipation for Thermal Design

Parameter	Typ [1]	Max [1]	Unit	Notes
VRM Power Dissipation	6	7	Watts	OverDrive Processor VRM
T_C, Max		105	°C	Voltage Regulator Maximum Case Temperature

NOTES:

1. Specification for the OverDrive® Processor Voltage Regulator Module. A Pentium® Pro processor OEM Module is specific to the design and may differ.

Table 34. OverDrive® Processor Thermal Resistance and Maximum Ambient Temperature

	Airflow - Ft./Min (M/Sec) [1]				
	100 (0.50)	150 (0.75)	200 (1.01)	250 (1.26)	300 (1.52)
OverDrive processor T_A, Max (°C)	Fan/Heatsink requires Ambient of 50°C or less regardless of external airflow.				
OverDrive processor VRM ΘCA (°C/W)	9.8	8.3	6.8	6.4	6.0
OverDrive processor VRM T_A, Max (°C) [2]	36	47	57	60	63

NOTES:

1. Airflow direction parallel to long axis of VRM PCB.
2. T_{CASE} = 105°C, Power as per Table 33.

8.6. Criteria for OverDrive® Processor

This section provides PC system designers with information on the engineering criteria required to ensure that a system is upgradable. The diagrams and checklists will aid the OEM to check specific criteria. Several design tools are available through Intel field representatives which will help the OEM meet the criteria. Refer to Section 8.6.1 for a list of documents.

The criteria are divided into 5 different categories:

- Electrical Criteria
- Thermal Criteria
- Mechanical Criteria
- Functional Criteria
- End User Criteria

8.6.1. RELATED DOCUMENTS

All references to related documents within this section imply the latest published revision of the related document, unless specifically stated otherwise. Contact your local Intel Sales representative for latest revisions of the related documents.

Processor and Motherboard Documentation:

- *Pentium® Pro Processor Developer's Manual: Programmer's Reference Manual* (Order Number 242691)

8.6.2. ELECTRICAL CRITERIA

The criteria in this section concentrates on the CPU and VRM, and covers pin to plane continuity, signal connections, signal timing and quality, and voltage transients.

8.6.2.1. OverDrive® Processor Electrical Criteria

The electrical criteria for the OverDrive processor is split into three tables. Most of the criteria refer directly to previous sections of this document.

The criteria for the OverDrive processor that only apply to motherboards and systems which employ a Header 8 are presented in Table 35. See Table 37 for criteria that apply regardless of a Header 8.

The criteria for the OverDrive processor that apply to all motherboards and systems are presented in Table 37.

Table 35. Electrical Test Criteria for Systems Employing Header 8

Criteria	Refer To:	Comment
5Vin Tolerance Header 8 Input	Table 32	Measured Under the following Loading Conditions: Max I_{CC5} at Steady-State Min I_{CC5} at Steady-State Fast Switch between Max and Min I_{CC5} Refer to Table 32 for OverDrive processor VRM I_{CC5} specification.
Pentium® Pro processor V_{CCP} Specification	Table 4	Measured Under the following Loading Conditions: Max I_{CCP} at Steady-State Min I_{CCP} at Steady-State Fast Switch between Max and Min I_{CCP} Refer to Table 5 for Pentium Pro processor I_{CCP} specification.
VRM RES pins	Table 29	Must not be connected.
VRM control signals (5Vin, VSS, PWRGOOD, UP#, V_{CCP}, and VID3-VID0)	Table 29	Must be connected as specified. OUTEN is optional.
VRM control input signal quality	Table 32	VRM control input signals must meet the DC specifications of the VRM.
Maximum Total LMB	Table 32	Inductance between VRM output and CPU socket pins.
Maximum Total RMB	Table 32	Resistance between VRM output and CPU socket pins.

Table 36. Electrical Test Criteria for Systems Not Employing Header 8

Criteria	Refer To:	Comment
V_{CCP} Primary CPU V_{CC} Voltage	Table 31 including note 2	Measured Under the following Loading Conditions: Max I_{CCP} at Steady-State Min I_{CCP} at Steady-State Fast Switch between Max and Min I_{CCP} Refer to Table 31 for OverDrive® processor I_{CCP} specification.

Table 37. Electrical Test Criteria for all Systems

Criteria	Refer To:	Comment
V_{CC5} Secondary CPU V_{CC} Voltage	Table 31	Loading Conditions: • Max I_{CC5} at Steady-State • Min I_{CC5} at Steady-State • Fast Switch between Max and Min I_{CC5} Refer to Table 31 for OverDrive® processor I_{CC5} specification.
V_{CC5}	Table 31	Fan/Heatsink Voltage
V_{CC} continuity to Socket 8	Table 26	0.5W or less for any single pin from Socket 8 V_{CC} pins to V_{CC} supply. Applies to both primary and secondary pins and their respective supplies.
VSS continuity to Socket 8	Table 26	0.5W or less for any single pin From Socket 8 VSS pins to VSS supply.
RESERVED Pins	Table 26	Must not be connected.
Input signal quality	Section 5.2	Must meet specification of the Pentium® Pro processor.
AC timing specifications	Section 3.15	Must meet all AC specifications of the Pentium Pro Processor.

8.6.2.2. Pentium® Pro Processor Electrical Criteria

Motherboards and systems will be tested to the specifications of the Pentium Pro processor in Section 3.

8.6.3. THERMAL CRITERIA

8.6.3.1. OverDrive® Processor Cooling Requirements (Systems Testing Only)

The maximum preheat temperature, T_{PH}, for the OverDrive processor must not be greater than specified in Section 8.5.1. T_{PH} is the temperature of the air entering the fan heatsink and is measured 0.3 inches (0.76 cm) above the center of the fan. Thermal testing should be performed at the OEM specified maximum system operating temperature (not less than 32°C), and under worst case thermal loading. Worst case thermal loading requires every I/O bus expansion slot to be filled with the longest typical add-in card that will not violate the required clearance for airflow around the OverDrive processor (refer to Section 8.2.2.2. for these requirements). These add-in cards represent typical power dissipation per type and form factor (Full length PCI, VL, ISA, and ½ length PCI dissipate 10W; ¾ length ISA dissipates 7.5W, ½ length ISA dissipates 5W, and ¼ length ISA dissipates 3.3W).

8.6.3.2. Pentium® Pro Processor Cooling Requirements (Systems Testing Only)

The Pentium Pro processor case temperature must meet the specifications of the Pentium Pro processor. Thermal testing should be performed under worst case thermal loading (Refer to 8.6.3.1. for loading description), and with a cooling solution representative of the OEM's cooling solution. Refer to Table 5 for the Pentium Pro processor case temperature specification.

8.6.3.3. Voltage Regulator Modules (Systems Employing a Header 8 Only)

The case temperature of the voltage regulator on the OverDrive processor VRM must not exceed the specification of Table 38.

Table 38. Thermal Test Criteria

Criteria	Refer To:	Comment
T_{PH}	Section 8.5.1.	Air temperature entering the fan/heatsink of the OverDrive® processor. Measured 0.3 inches (0.76 cm) above the center of the fan/heatsink.
Pentium® Pro processor Case Temperature	Table 5	T_C must meet the specifications of the Pentium Pro Processor. Measured with a cooling solution representative of the OEM's.
Voltage Regulator Case Temperature	Table 33	

8.6.4. MECHANICAL CRITERIA

8.6.4.1. OverDrive® Processor Clearance and Airspace Requirements

Refer to Figure 48 for a drawing of the various clearance and airspace requirements

Table 39. Mechanical Test Criteria for the OverDrive® Processor

Criteria	Refer To:	Comment
Minimum airspace from top surface of socket to any object.	Figure 48	See "Total Clearance Above Socket" in Figure 40.
Minimum airspace around all 4 sides of the OverDrive processor fan/heatsink.	Figure 48	Required from the CPU package side to the top of the vertical clearance area. See "A" in Figure 40.
Minimum airspace around heatsink clip tabs.	Figure 48	Extend from the motherboard surface to the top of the fan/heatsink. See "Keep Out Zones" in Figure 40.
ZIF socket lever operation.	Figure 48	Must operate from fully closed to fully open position with no interference.

8.6.4.2. OverDrive® Processor VRM Clearance and Airspace Requirements

Refer to Figure 50 for a drawing of the various clearance and airspace requirements of the OverDrive processor VRM. Nothing must intrude into the space envelope, including airspace region, defined in Figure 50 with the exception of Header 8 itself.

8.6.5. FUNCTIONAL CRITERIA

The OverDrive processor is intended to replace the original Pentium Pro processor. The system must boot properly without error messages when the OverDrive processor is installed.

8.6.5.1. Software Compatibility

System hardware and software that operates properly with the original Pentium Pro processor

must operate properly with the OverDrive processor.

Table 40. Functional Test Criteria

Criteria	Refer To:	Comment
Software Compatibility		No incompatibilities resulting from upgrade installation.
BIOS Functionality	Section 8.3.3.	CPU Type Reported on Screen must be reported correctly or not at all. Intel recommends reporting "OverDrive Processor". Never Use Timing Loops. Do not test the cache, or use model specific registers when the upgrade is detected.

8.6.5.2. BIOS Functionality

The BIOS must continue to operate correctly with the OverDrive processor installed in the system. Always use the CPU Signature and Feature flags to identify if an OverDrive processor is installed. Please refer to the *Pentium® Pro Processor Developer's Manual: Volume 3, Programmer's Reference Manual:* (Order Number 242691) for the BIOS recommendations.

8.6.6. END USER CRITERIA

8.6.6.1. Qualified OverDrive® Processor Components

To ensure processor upgradability, a system should employ the following Intel-qualified OverDrive processor components. For a list of qualified components contact your Intel sales representative, or if in the US, contact Intel FaxBACK Information Service at (800) 525-3019.

- Genuine Intel OEM CPU
- Socket 8, 387-hole ZIF
- Header 8, 40-pin shrouded (Systems and Motherboards employing Header 8 solution only.) OR programmable voltage regulator capable of providing the voltage and current required by the OverDrive processor.

8.6.6.2. Visibility and Installation

Socket 8 and Header 8 must be visible upon removal of the system cover. Otherwise, the OEM must include diagrams or other indicators visible upon removal of the system cover or clear instructions in the user's manual to guide the end user to the CPU socket and the VRM header. Special tools, other than a screw driver, must not be required for an upgrade installation.

8.6.6.3. Jumper Configuration

End user configured jumpers are not recommended. If design requires jumpers or switches to upgrade the system, a detailed jumper description in the manual is required. The jumpers must be easy to locate and set. Jumper identification should be silk-screened on the motherboard if possible. Jumper tables on the inside of the system case are recommended.

8.6.6.4. BIOS Changes

BIOS changes or additional software must not be required to upgrade the system with the OverDrive processor.

8.6.6.5. Documentation

The system documentation must include installation instructions, with illustrations of the system, Socket 8 and Header 8 location, and any heatsink clip's operation and orientation instructions. Furthermore, there must be no documentation anywhere stating that the warranty is void if the OEM processor is removed.

8.6.6.6. Upgrade Removal

The upgrade process must be reversible such that upon re-installation of the original CPU, the system

must retain original functionality and the cooling solution must return to its original effectiveness.

APPENDIX A

This appendix provides an alphabetical listing of all Pentium Pro processor signals. **Pins that do not appear here are not considered bus signals and are described in Table 2.**

A.1 A[35:3]# (I/O)

The A[35:3]# signals are the address signals. They are driven during the two-clock Request Phase by the request initiator. The signals in the two clocks are referenced Aa[35:3]# and Ab[35:3]#. During both clocks, A[35:24]# signals are protected with the AP1# parity signal, and A[23:3]# signals are protected with the AP0# parity signal.

The Aa[35:3]# signals are interpreted based on information carried during the first Request Phase clock on the REQa[4:0]# signals.

For memory transactions as defined by REQa[4:0]# = {XX01X,XX10X,XX11X}, the Aa[35:3]# signals define a 2^{36}-byte physical memory address space. The cacheable agents in the system observe the Aa[35:3]# signals and begin an internal snoop. The memory agents in the system observe the Aa[35:3]# signals and begin address decode to determine if they are responsible for the transaction completion. Aa[4:3]# signals define the critical word, the first data chunk to be transferred on the data bus. Cache line transactions use the standard burst order described in *Pentium® Pro Processor Developer's Manual, Volume 1: Specifications (Order* Number 242690) to transfer the remaining three data chunks.

For Pentium Pro processor IO transactions as defined by REQa[4:0]# = 1000X, the signals Aa[16:3]# define a 64K+3 byte physical IO space. The IO agents in the system observe the signals and begin address decode to determine if they are responsible for the transaction completion. Aa[35:17]# are always zero. Aa16# is zero unless the IO space being accessed is the first three bytes of a 64KByte address range.

For deferred reply transactions as defined by REQa[4:0]# = 00000, Aa[23:16]# carry the deferred ID. This signal is the same deferred ID supplied by the request initiator of the original transaction on Ab[23:16]#/DID[7:0]# signals. Pentium Pro processor bus agents that support deferred replies sample the deferred ID and perform an internal match against any outstanding transactions waiting for deferred replies. During a deferred reply, Aa[35:24]# and Aa[15:3]# are reserved.

For the branch-trace message transaction as defined by REQa[4:0]# = 01001 and for special and interrupt acknowledge transactions, as defined by REQa[4:0]# = 01000, the Aa[35:3]# signals are reserved and undefined.

During the second clock of the Request Phase, Ab[35:3]# signals perform identical signal functions for all transactions. For ease of description, these functions are described using new signal names. Ab[31:24]# are renamed the attribute signals ATTR[7:0]#. Ab[23:16]# are renamed the Deferred ID signals DID[7:0]#. Ab[15:8]# are renamed the eight-byte enable signals BE[7:0]#. Ab[7:3]# are renamed the extended function signals EXF[4:0]#.

Table 41. Request Phase Decode

Ab[31:24]#	Ab[23:16]#	Ab[15:8]#	Ab[7:3]#
ATTR[7:0]#	DID[7:0]#	BE[7:0]#	EXF[4:0]#

On the active-to-inactive transition of RESET#, each Pentium Pro processor bus agent samples A[35:3]# signals to determine its power-on configuration. Two clocks after RESET# is sampled deasserted, these signals begin normal operation.

A.2 A20M# (I)

The A20M# signal is the address-20 mask signal in the PC Compatibility group. If the A20M# input signal is asserted, the Pentium Pro processor masks physical address bit 20 (A20#) before looking up a line in any internal cache and before driving a read/write transaction on the bus. Asserting A20M# emulates the 8086 processor's address wrap around at the one Mbyte boundary. Only assert A20M# when the processor is in real mode. The effect of asserting A20M# in protected mode is undefined and may be implemented differently in future processors.

Snoop requests and cache-line writeback transactions are unaffected by A20M# input. Address 20 is not masked when the processor samples external addresses to perform internal snooping.

A20M# is an asynchronous input. However, to guarantee recognition of this signal following an I/O write instruction, A20M# must be valid with active RS[2:0]# signals of the corresponding I/O Write bus transaction. In FRC mode, A20M# must be synchronous to BCLK.

During active RESET#, the Pentium Pro processor begins sampling the A20M#, IGNNE#, and LINT[1:0] values to determine the ratio of core-clock frequency to bus-clock frequency. After the PLL-lock time, the core clock becomes stable and is locked to the external bus clock. On the active-to-inactive transition of RESET#, the Pentium Pro processor latches A20M# and IGNNE# and freezes the frequency ratio internally. Normal operation on the two signals continues two clocks after RESET# is sampled inactive.

A.3 ADS# (I/O)

The ADS# signal is the address Strobe signal. It is asserted by the current bus owner for one clock to indicate a new Request Phase. A new Request Phase can only begin if the In-order Queue has less than the maximum number of entries defined by the power-on configuration (1 or 8), the Request Phase is not being stalled by an active BNR# sequence and the ADS# associated with the previous Request Phase is sampled inactive. Along with the ADS#, the request initiator drives A[35:3]#, REQ[4:0]#, AP[1:0]#, and RP# signals for two clocks. During the second Request Phase clock, ADS# must be inactive. RP# provides parity protection for REQ[4:0]# and ADS# signals during both clocks. If the transaction is part of a bus locked operation, LOCK# must be active with ADS#.

If the request initiator continues to own the bus after the first Request Phase, it can issue a new request every three clocks. If the request initiator needs to release the bus ownership after the Request Phase, it can deactivate its BREQn#/ BPRI# arbitration signal as early as with the activation of ADS#.

All bus agents observe the ADS# activation to begin parity checking, protocol checking, address decode, internal snoop, or deferred reply ID match operations associated with the new transaction. On sampling the asserted ADS#, all agents load the new transaction in the In-order Queue and update internal counters. The Error, Snoop, Response, and Data Phase of the transaction are defined with respect to ADS# assertion.

Table 42. Bus Clock Ratios Versus Pin Logic Levels

Ratio of Core Clock to Bus Clock	LINT[1]/NMI	LINT[0]/INTR	IGNNE#	A20M#
2	L	L	L	L
2	H	H	H	H
3	L	L	H	L
4	L	L	L	H
5	L	L	H	H
5/2	L	H	L	L
7/2	L	H	H	L
9/2	L	H	L	H
11/2	L	H	H	H
RESERVED	ALL OTHER COMBINATIONS			

A.4 AERR# (I/O)

The AERR# signal is the address parity error signal. Assuming the AERR# driver is enabled during the power-on configuration, a bus agent can drive AERR# active for exactly one clock during the Error Phase of a transaction. AERR# must be inactive for a minimum of two clocks. The Error Phase is always three clocks from the beginning of the Request Phase.

On observing active ADS#, all agents begin parity and protocol checks for the signals valid in the two Request Phase clocks. Parity is checked on AP[1:0]# and RP# signals. AP1# protects A[35:24]#, AP0# protects A[23:3]# and RP# protects REQ[4:0]#. A parity error without a protocol violation is signaled by AERR# assertion.

If AERR# observation is enabled during power-on configuration, AERR# assertion in a valid Error Phase aborts the transaction. All bus agents remove the transaction from the In-order Queue and update internal counters. The Snoop Phase, Response Phase, and Data Phase of the transaction are aborted. All signals in these phases must be deasserted two clocks after AERR# is asserted, even if the signals have been asserted before AERR# has been observed. Specifically if the Snoop Phase associated with the aborted transaction is driven in the next clock, the snoop results, including a STALL condition (HIT# and HITM# asserted for one clock), are ignored. All bus agents must also begin an arbitration reset sequence and deassert BREQn#/BPRI# arbitration signals on sampling AERR# active. A current bus owner in the middle of a bus lock operation must keep LOCK# asserted and assert its arbitration request BPRI#/BREQn# after keeping it inactive for two clocks to retain its bus ownership and guarantee lock atomicity. All other agents, including the current bus owner not in the middle of a bus lock operation, must wait at least 4 clocks before asserting BPRI#/BREQn# and beginning a new arbitration.

If AERR# observation is enabled, the request initiator can retry the transaction up to n times until it reaches the retry limit defined by its implementation. (The Pentium Pro processor retries once.) After n retries, the request initiator treats the error as a hard error. The request initiator asserts BERR# or enters the Machine Check Exception handler, as defined by the system configuration.

If AERR# observation is disabled during power-on configuration, AERR# assertion is ignored by all bus agents except a central agent. Based on the Machine Check Architecture of the system, the central agent can ignore AERR#, assert NMI to execute NMI handler, or assert BINIT# to reset the bus units of all agents and execute an MCE handler.

A.5 AP[1:0]# (I/O)

The AP[1:0]# signals are the address parity signals. They are driven by the request initiator during the two Request Phase clocks along with ADS#, A[35:3]#, REQ[4:0]#, and RP#. AP1# covers A[35:24]#. AP0# covers A[23:3]#. A correct parity signal is high if an even number of covered signals are low and low if an odd number of covered signals are low. This rule allows parity to be high when all the covered signals are high.

Provided "AERR# drive" is enabled during the power-on configuration, all bus agents begin parity checking on observing active ADS# and determine if there is a parity error. On observing a parity error on any one of the two Request Phase clocks, the bus agent asserts AERR# during the Error Phase of the transaction.

A.6 ASZ[1:0]# (I/O)

The ASZ[1:0]# signals are the memory address-space size signals. They are driven by the request initiator during the first Request Phase clock on the REQa[4:3]# pins. The ASZ[1:0]# signals are valid only when REQa[1:0]# signals equal 01B, 10B, or 11B, indicating a memory access transaction. The ASZ[1:0]# decode is defined in Table 43.

Table 43. ASZ[1:0]# Signal Decode

ASZ[1:0]#		Description
0	0	0 <= A[35:3]# < 4 GB
0	1	4 GB <= A[35:3]# < 64 GB
1	X	Reserved

If the memory access is within the 0-to-(4GByte -1) address space, ASZ[1:0]# must be 00B. If the memory access is within the 4Gbyte-to-(64GByte - 1) address space, ASZ[1:0]# must be 01B. All observing bus agents that support the 4Gbyte (32 bit) address space must respond to the transaction only when ASZ[1:0]# equals 00. All observing bus agents that support the 64GByte (36- bit) address

space must respond to the transaction when ASZ[1:0]# equals 00B or 01B.

A.7 ATTR[7:0]# (I/O)

The ATTR[7:0]# signals are the attribute signals. They are driven by the request initiator during the second Request Phase clock on the Ab[31:24]# pins. The ATTR[7:0]# signals are valid for all transactions. The ATTR[7:3]# are reserved and undefined. The ATTR[2:0]# are driven based on the Memory Range Register attributes and the Page Table attributes. Table 45. defines ATTR[3:0]# signals.

A.8 BCLK (I)

The BCLK (clock) signal is the Execution Control group input signal. It determines the bus frequency. All agents drive their outputs and latch their inputs on the BCLK rising edge.

The BCLK signal indirectly determines the Pentium Pro processor's internal clock frequency. Each Pentium Pro processor derives its internal clock from BCLK by multiplying the BCLK frequency by a ratio as defined and allowed by the power-on configuration. See Table 42.

All external timing parameters are specified with respect to the BCLK signal.

A.9 BE[7:0]# (I/O)

The BE[7:0]# signals are the byte-enable signals. They are driven by the request initiator during the second Request Phase clock on the Ab[15:8]# pins. These signals carry various information depending on the REQ[4:0]# value.

For memory or I/O transactions (REQa[4:0]# = {10000B, 10001B, XX01XB, XX10XB, XX11XB}) the byte-enable signals indicate that valid data is requested or being transferred on the corresponding byte on the 64 bit data bus. BE0# indicates D[7:0]# is valid, BE1# indicates D[15:8]# is valid,..., BE7# indicates D[63:56]# is valid.

For Special transactions ((REQa[4:0]# = 01000B) and (REQb[1:0]# = 01B)), the BE[7:0]# signals carry special cycle encodings as defined in Table 44. All other encodings are reserved.

Table 44. Special Transaction Encoding on BE[7:0]#

BE[7:0]#	Special Cycle
0000 0000	Reserved
0000 0001	Shutdown
0000 0010	Flush
0000 0011	Halt
0000 0100	Sync
0000 0101	Flush Acknowledge
00000 0110	Stop Clock Acknowledge
00000 0111	SMI Acknowledge
Other	Reserved

For Deferred Reply, Interrupt Acknowledge, and Branch Trace Message transactions, the BE[7:0]# signals are undefined.

Table 45. ATTR[7:0]# Field Descriptions

ATTR[7:3]#	ATTR[2]#	ATTR[1:0]#			
XXXXX	X	11	10	01	00
Reserved	Potentially Speculatable	Write-Back	Write-Protect	Write-Through	UnCacheable

A.10 BERR# (I/O)

The BERR# signal is the Error group Bus Error signal. It is asserted to indicate an unrecoverable error without a bus protocol violation.

The BERR# protocol is as follows: If an agent detects an unrecoverable error for which BERR# is a valid error response and BERR# is sampled inactive, it asserts BERR# for three clocks. An agent can assert BERR# only after observing that the signal is inactive. An agent asserting BERR# must deassert the signal in two clocks if it observes that another agent began asserting BERR# in the previous clock.

BERR# assertion conditions are defined by the system configuration. Configuration options enable the BERR# driver as follows:

- Enabled or disabled
- Asserted optionally for internal errors along with IERR#
- Optionally asserted by the request initiator of a bus transaction after it observes an error
- Asserted by any bus agent when it observes an error in a bus transaction

BERR# sampling conditions are also defined by the system configuration. Configuration options enable the BERR# receiver to be enabled or disabled. When the bus agent samples an active BERR# signal and if MCE is enabled, the Pentium Pro processor enters the Machine Check Handler. If MCE is disabled, typically the central agent forwards BERR# as an NMI to one of the processors. The Pentium Pro processor does not support BERR# sampling (always disabled).

A.11 BINIT# (I/O)

The BINIT# signal is the bus initialization signal. If the BINIT# driver is enabled during the power on configuration, BINIT# is asserted to signal any bus condition that prevents reliable future information.

The BINIT# protocol is as follows: If an agent detects an error for which BINIT# is a valid error response, and BINIT# is sampled inactive, it asserts BINIT# for three clocks. An agent can assert BINIT# only after observing that the signal is inactive. An agent asserting BINIT# must deassert the signal in two clocks if it observes that another agent began asserting BINIT# in the previous clock.

If BINIT# observation is enabled during power-on configuration, and BINIT# is sampled asserted, all bus state machines are reset. All agents reset their rotating ID for bus arbitration to the state after reset, and internal count information is lost. The L1 and L2 caches are not affected.

If BINIT# observation is disabled during power-on configuration, BINIT# is ignored by all bus agents except a central agent that must handle the error in a manner appropriate to the system architecture.

A.12 BNR# (I/O)

The BNR# signal is the Block Next Request signal in the Arbitration group. The BNR# signal is used to assert a bus stall by any bus agent who is unable to accept new bus transactions to avoid an internal transaction queue overflow. During a bus stall, the current bus owner cannot issue any new transactions.

Since multiple agents might need to request a bus stall at the same time, BNR# is a wire-OR signal. In order to avoid wire-OR glitches associated with simultaneous edge transitions driven by multiple drivers, BNR# is activated on specific clock edges and sampled on specific clock edges. A valid bus stall involves assertion of BNR# for one clock on a well-defined clock edge (T1), followed by de-assertion of BNR# for one clock on the next clock edge (T1+1). BNR# can first be sampled on the second clock edge (T1+1) and must always be ignored on the third clock edge (T1+2). An extension of a bus stall requires one clock active (T1+2), one clock inactive (T1+3) BNR# sequence with BNR# sampling points every two clocks (T1+1, T1+3,...).

After the RESET# active-to-inactive transition, bus agents might need to perform hardware initialization of their bus unit logic. Bus agents intending to create a request stall must assert BNR# in the clock after RESET# is sampled inactive.

After BINIT# assertion, all bus agents go through a similar hardware initialization and can create a request stall by asserting BNR# four clocks after BINIT# assertion is sampled.

On the first BNR# sampling clock that BNR# is sampled inactive, the current bus owner is allowed to issue one new request. Any bus agent can immediately reassert BNR# (four clocks from the previous assertion or two clocks from the previous de-assertion) to create a new bus stall. This

throttling mechanism enables independent control on every new request generation.

If BNR# is deasserted on two consecutive sampling points, new requests can be freely generated on the bus. After receiving a new transaction, a bus agent can require an address stall due to an anticipated transaction-queue overflow condition. In response, the bus agent can assert BNR#, three clocks from active ADS# assertion and create a bus stall. Once a bus stall is created, the bus remains stalled until BNR# is sampled asserted on subsequent sampling points.

A.13 BP[3:2]# (I/O)

The BP[3:2]# signals are the System Support group Breakpoint signals. They are outputs from the Pentium Pro processor that indicate the status of breakpoints.

A.14 BPM[1:0]# (I/O)

The BPM[1:0]# signals are more System Support group breakpoint and performance monitor signals. They are outputs from the Pentium Pro processor that indicate the status of breakpoints and programmable counters used for monitoring Pentium Pro processor performance.

A.15 BPRI# (I)

The BPRI# signal is the Priority-agent Bus Request signal. The priority agent arbitrates for the bus by asserting BPRI#. The priority agent is always be the next bus owner. Observing BPRI# active causes the current symmetric owner to stop issuing new requests, unless such requests are part of an ongoing locked operation.

If LOCK# is sampled inactive two clocks from BPRI# driven asserted, the priority agent can issue a new request within four clocks of asserting BPRI#. The priority agent can further reduce its arbitration latency to two clocks if it samples active ADS# and inactive LOCK# on the clock in which BPRI# was driven active and to three clocks if it samples active ADS# and inactive LOCK# on the clock in which BPRI# was sampled active. If LOCK# is sampled active, the priority agent must wait for LOCK# deasserted and gains bus ownership in two clocks after LOCK# is sampled deasserted. The priority agent can keep BPRI# asserted until all of its requests are completed and can release the bus by de-asserting BPRI# as early as the same clock edge on which it issues the last request.

On observation of active AERR#, RESET#, or BINIT#, BPRI# must be deasserted in the next clock. BPRI# can be reasserted in the clock after sampling the RESET# active-to-inactive transition or three clocks after sampling BINIT# active and RESET# inactive. On AERR# assertion, if the priority agent is in the middle of a bus-locked operation, BPRI# must be re-asserted after two clocks, otherwise BPRI# must stay inactive for at least 4 clocks.

After the RESET# inactive transition, Pentium Pro processor bus agents begin BPRI# and BNR# sampling on BNR# sample points. When both BNR# and BPRI# are observed inactive on a BNR# sampling point, the APIC units in Pentium Pro processors on a common APIC bus are synchronized.

A.16 BR0#(I/O), BR[3:1]# (I)

The BR[3:0]# pins are the physical bus request pins that drive the BREQ[3:0]# signals in the system. The BREQ[3:0]# signals are interconnected in a rotating manner to individual processor pins. Table 46 gives the rotating interconnect between the processor and bus signals.

Table 46. BR[3:0]# Signals Rotating Interconnect

Bus Signal	Agent 0 Pins	Agent 1 Pins	Agent 2Pins	Agent 3 Pins
BREQ0#	BR0#	BR3#	BR2#	BR1#
BREQ1#	BR1#	BR0#	BR3#	BR2#
BREQ2#	BR2#	BR1#	BR0#	BR3#
BREQ3#	BR3#	BR2#	BR1#	BR0#

During power-up configuration, the central agent must assert the BR0# bus signal. All symmetric agents sample their BR[3:0]# pins on active-to-inactive transition of RESET#. The pin on which the agent samples an active level determines its agent ID. All agents then configure their pins to match the appropriate bus signal protocol, as shown in Table 47.

Table 47. BR[3:0]# Signal Agent IDs

Pin Sampled Active on RESET#	Agent ID
BR0#	0
BR3#	1
BR2#	2
BR1#	3

A.17 BREQ[3:0]# (I/O)

The BREQ[3:0]# signals are the Symmetric-agent Arbitration Bus signals (called bus request). A symmetric agent n arbitrates for the bus by asserting its BREQn# signal. Agent n drives BREQn# as an output and receives the remaining BREQ[3:0]# signals as inputs.

The symmetric agents support distributed arbitration based on a round-robin mechanism. The rotating ID is an internal state used by all symmetric agents to track the agent with the lowest priority at the next arbitration event. At power-on, the rotating ID is initialized to three, allowing agent 0 to be the highest priority symmetric agent. After a new arbitration event, the rotating ID of all symmetric agents is updated to the agent ID of the symmetric owner. This update gives the new symmetric owner lowest priority in the next arbitration event.

A new arbitration event occurs either when a symmetric agent asserts its BREQn# on an Idle bus (all BREQ[3:0]# previously inactive), or the current symmetric owner de-asserts BREQm# to release the bus ownership to a new bus owner n. On a new arbitration event, based on BREQ[3:0]#, and the rotating ID, all symmetric agents simultaneously determine the new symmetric owner. The symmetric owner can park on the bus (hold the bus) provided that no other symmetric agent is requesting its use. The symmetric owner parks by keeping its BREQn# signal active. On sampling active BREQm# asserted by another symmetric agent, the symmetric owner de-asserts BREQn# as soon as possible to release the bus. A symmetric owner stops issuing new requests that are not part of an existing locked operation upon observing BPRI# active.

A symmetric agent can not deassert BREQn# until it becomes a symmetric owner. A symmetric agent can reassert BREQn# after keeping it inactive for one clock.

On observation of active AERR#, RESET#, or BINIT#, the BREQ[3:0]# signals must be deasserted in the next clock. BREQ[3:0]# can be reasserted in the clock after sampling the RESET# active-to-inactive transition or three clocks after sampling BINIT# active and RESET# inactive. On AERR# assertion, if bus agent n is in the middle of a bus-locked operation, BREQn# must be re-asserted after two clocks, otherwise BREQ[3:0]# must stay inactive for at least 4 clocks.

A.18 D[63:0]# (I/O)

The D[63:0]# signals are the data signals. They are driven during the Data Phase by the agent responsible for driving the data. These signals provide a 64-bit data path between various Pentium Pro processor bus agents. 32-byte line transfers require four data transfer clocks with valid data on all eight bytes. Partial transfers require one data transfer clock with valid data on the byte(s) indicated by active byte enables BE[7:0]#. Data

signals not valid for a particular transfer must still have correct ECC (if data bus ECC is selected). If BE0# is asserted, D[7:0]# transfers the least significant byte. If BE7# is asserted, D[63:56]# transfers the most significant byte.

The data driver asserts DRDY# to indicate a valid data transfer. If the Data Phase involves more than one clock the data driver also asserts DBSY# at the beginning of the Data Phase and de-asserts DBSY# no earlier than on the same clock that it performs the last data transfer.

A.19 DBSY# (I/O)

The DBSY# signal is the Data-bus Busy signal. It indicates that the data bus is busy. It is asserted by the agent responsible for driving the data during the Data Phase, provided the Data Phase involves more than one clock. DBSY# is asserted at the beginning of the Data Phase and may be deasserted on or after the clock on which the last data is driven. The data bus is released one clock after DBSY# is deasserted.

When normal read data is being returned, the Data Phase begins with the Response Phase. Thus the agent returning read data can assert DBSY# when the transaction reaches the top of the In-order Queue and it is ready to return response on RS[2:0]# signals. In response to a write request, the agent driving the write data must drive DBSY# active after the write transaction reaches the top of the In-order Queue and it sees active TRDY# with inactive DBSY# indicating that the target is ready to receive data. For an implicit writeback response, the snoop agent must assert DBSY# active after the target memory agent of the implicit writeback asserts TRDY#. Implicit writeback TRDY# assertion begins after the transaction reaches the top of the In-order Queue, and TRDY# de-assertion associated with the write portion of the transaction, if any is completed. In this case, the memory agent guarantees assertion of implicit writeback response in the same clock in which the snooping agent asserts DBSY#.

A.20 DEFER# (I)

The DEFER# signal is the defer signal. It is asserted by an agent during the Snoop Phase to indicate that the transaction cannot be guaranteed in-order completion. Assertion of DEFER# is normally the responsibility of the addressed memory agent or I/O agent. For systems that involve resources on a system bus other than the Pentium Pro processor bus, a bridge agent can accept the DEFER# assertion responsibility on behalf of the addressed agent.

When HITM# and DEFER# are both active during the Snoop Phase, HITM# is given priority and the transaction must be completed with implicit writeback response. If HITM# is inactive, and DEFER# active, the agent asserting DEFER# must complete the transaction with a Deferred or Retry response.

If DEFER# is inactive, or HITM# is active, then the transaction is committed for in-order completion and snoop ownership is transferred normally between the requesting agent, the snooping agents, and the response agent.

If DEFER# is active with HITM# inactive, the transaction commitment is deferred. If the defer agent completes the transaction with a retry response, the requesting agent must retry the transaction. If the defer agent returns a deferred response, the requesting agent must freeze snoop state transitions associated with the deferred transaction and issues of new order-dependent transactions until the corresponding deferred reply transaction. In the meantime, the ownership of the deferred address is transferred to the defer agent and it must guarantee management of conflicting transactions issued to the same address.

If DEFER# is active in response to a newly issued bus-lock transaction, the entire bus-locked operation is re-initiated regardless of HITM#. This feature is useful for a bridge agent in response to a split bus-locked operation. It is recommended that the bridge agent extend the Snoop Phase of the first transaction in a split locked operation until it can either guarantee ownership of all system resources to enable successful completion of the split sequence or assert DEFER# followed by a Retry Response to abort the split sequence.

A.21 DEN# (I/O)

The DEN# signal is the defer-enable signal. It is driven to the bus on the second clock of the Request Phase on the EXF1#/Ab4# pin. DEN# is asserted to indicate that the transaction can be deferred by the responding agent.

A.22 DEP[7:0]# (I/O)

The DEP[7:0]# signals are the data bus ECC protection signals. They are driven during the Data Phase by the agent responsible for driving D[63:0]#. The DEP[7:0]# signals provide optional ECC protection for the data bus. During power-on configuration, DEP[7:0]# signals can be enabled for either ECC checking or no checking.

The ECC error correcting code can detect and correct single-bit errors and detect double-bit or nibble errors. The *Pentium® Pro Processor Developer's Manual, Volume 1: Specifications* (Order Number 242690) provides more information about ECC.

DEP[7:0]# provide valid ECC for the entire data bus on each data clock, regardless of which bytes are valid. If checking is enabled, receiving agents check the ECC signals for all 64 data signals.

A.23 DID[7:0]# (I/O)

The DID[7:0]# signals are Deferred Identifier signals. They are transferred using A[23:16]# signals by the request initiator. They are transferred on Ab[23:16]# during the second clock of the Request Phase on all transactions, but only defined for deferrable transactions (DEN# asserted). DID[7:0]# is also transferred on Aa[23:16]# during the first clock of the Request Phase for Deferred Reply transactions.

The deferred identifier defines the token supplied by the request initiator. DID[7:4]# carry the request initiators' agent identifier and DID[3:0]# carry a transaction identifier associated with the request. This configuration limits the bus specification to 16 bus masters with each one of the bus masters capable of making up to sixteen requests.

Every deferrable transaction issued on the Pentium Pro processor bus which has not been guaranteed completion (has not successfully passed its Snoop Result Phase) will have a unique Deferred ID. This includes all outstanding transactions which have not had their snoop result reported, or have had their snoop results deferred. After a deferrable transaction passes its Snoop Result Phase without DEFER# asserted, its Deferred ID may be reused. Similarly, the deferred ID of a transaction which was deferred may be reused after the completion of the snoop window of the deferred reply.

DID[7]# indicates the agent type. Symmetric agents use 0. Priority agents use 1. DID[6:4]# indicates the agent ID. Symmetric agents use their arbitration ID. The Pentium Pro processor has four symmetric agents, so does not assert DID[6]#. DID[3:0]# indicates the transaction ID for an agent. The transaction ID must be unique for all transactions issued by an agent which have not reported their snoop results.

Table 48. DID[7:0]# Encoding

DID[7]	DID[6:4]	DID[3:0]
Agent Type	Agent ID	Transaction ID

The Deferred Reply agent transmits the DID[7:0]# (Ab[23:16]#) signals received during the original transaction on the Aa[23:16]# signals during the Deferred Reply transaction. This process enables the original request initiator to make an identifier match and wake up the original request waiting for completion.

A.24 DRDY# (I/O)

The DRDY# signal is the Data Phase data-ready signal. The data driver asserts DRDY# on each data transfer, indicating valid data on the data bus. In a multi-cycle data transfer, DRDY# can be deasserted to insert idle clocks in the Data Phase. During a line transfer, DRDY# is active for four clocks. During a partial 1-to-8 byte transfer, DRDY# is active for one clock. If a data transfer is exactly one clock, then the entire Data Phase may consist of only one clock active DRDY# and inactive DBSY#. If DBSY# is asserted for a 1-to-8 byte transfer, then the data bus is not released until one clock after DBSY# is deasserted.

A.25 DSZ[1:0]# (I/O)

The DSZ[1:0]# signals are the data-size signals. They are transferred on REQb[4:3]# signals in the second clock of Request Phase by the requesting agent. The DSZ[1:0]# signals define the data transfer capability of the requesting agent. For the Pentium Pro processor, DSZ#= 00, always.

A.26 EXF[4:0]# (I/O)

The EXF[4:0]# signals are the Extended Function signals and are transferred on the Ab[7:3]# signals by the request initiator during the second clock of the Request Phase. The signals specify any special functional requirement associated with the transaction based on the requester mode or capability. The signals are defined in Table 49.

Table 49. EXF[4:0]# Signal Definitions

EXF	NAME	External Functionality	When Activated
EXF4#	SMMEM#	SMM Mode	After entering SMM mode
EXF3#	SPLCK#	Split Lock	The first transaction of a split bus lock operation
EXF2#	Reserved	Reserved	
EXF1#	DEN#	Defer Enable	The transactions for which Defer or Retry Response is acceptable.
EXF0#	Reserved	Reserved	

A.27 FERR# (O)

The FERR# signal is the PC Compatibility group Floating-point Error signal. The Pentium Pro processor asserts FERR# when it detects an unmasked floating-point error. FERR# is similar to the ERROR# signal on the Intel387™ coprocessor. FERR# is included for compatibility with systems using DOS-type floating-point error reporting.

A.28 FLUSH# (I)

When the FLUSH# input signal is asserted, the Pentium Pro processor bus agent writes back all internal cache lines in the Modified state and invalidates all internal cache lines. At the completion of a flush operation, the Pentium Pro processor issues a Flush Acknowledge transaction to indicate that the cache flush operation is complete. The Pentium Pro processor stops caching any new data while the FLUSH# signal remains asserted.

FLUSH# is an asynchronous input. However, to guarantee recognition of this signal following an I/O write instruction, FLUSH# must be valid along with RS[2:0]# in the Response Phase of the corresponding I/O Write bus transaction. In FRC mode, FLUSH# must be synchronous to BCLK.

On active-to-inactive transition of RESET#, each Pentium Pro processor bus agent samples FLUSH# signals to determine its power-on configuration. Two clocks after RESET# is sampled deasserted, these signals begin normal operation.

A.29 FRCERR (I/O)

The FRCERR signal is the Error group Functional-redundancy-check Error signal. If two Pentium Pro processors are configured in an FRC pair, as a single "logical" processor, then the checker processor asserts FRCERR if it detects a mismatch between its internally sampled outputs and the master processor's outputs. The checker's FRCERR output pin is connected to the master's FRCERR input pin.

For point-to-point connections, the checker always compares against the master's outputs. For bussed single-driver signals, the checker compares against the signal when the master is the only allowed driver. For bussed multiple-driver Wire-OR signals, the checker compares against the signal only if the master is expected to drive the signal low.

FRCERR is also toggled during the Pentium Pro processor's reset action. A Pentium Pro processor asserts FRCERR for approximately 1 second after RESET's active-to-inactive transition if it executes its built-in self-test (BIST). When BIST execution completes, the Pentium Pro processor de-asserts FRCERR if BIST completed successfully and continues to assert FRCERR if BIST fails. If the Pentium Pro processor does not execute the BIST action, then it keeps FRCERR asserted for approximately 20 clocks and then de-asserts it.

The Pentium® Pro Processor Developer's Manual, Volume 1: Specifications (Order Number 242690) describes how a Pentium Pro processor can be configured as a master or a checker.

A.30 HIT# (I/O), HITM# (I/O)

The HIT# and HITM# signals are Snoop-hit and Hit-modified signals. They are snoop results asserted by any Pentium Pro processor bus agent in the Snoop Phase.

Any bus agent can assert both HIT# and HITM# together for one clock in the Snoop Phase to indicate that it requires a snoop stall. When a stall condition is sampled, all bus agents extend the Snoop Phase by two clocks. The stall can be continued by reasserting HIT# and HITM# together every other clock for one clock.

A caching agent must assert HITM# for one clock in the Snoop Phase if the transaction hits a Modified line, and the snooping agent must perform an implicit writeback to update main memory. The snooping agent with the Modified line makes a transition to Shared state if the original transaction is Read Line or Read Partial, otherwise it transitions to Invalid state. A Deferred Reply transaction may have HITM# asserted to indicate the return of unexpected data.

A snooping agent must assert HIT# for one clock during the Snoop Phase if the line does not hit a Modified line in its writeback cache and if at the end of the transaction it plans to keep the line in Shared state. Multiple caching agents can assert HIT# in the same Snoop Phase. If the requesting agent observes HIT# active during the Snoop Phase it can not cache the line in Exclusive or Modified state.

On observing a snoop stall, the agents asserting HIT# and HITM# independently reassert the signal after one inactive clock so that the correct snoop result is available, in case the Snoop Phase terminates after the two clock extension.

A.31 IERR# (O)

The IERR# signal is the Error group Internal Error signal. A Pentium Pro processor asserts IERR# when it observes an internal error. It keeps IERR# asserted until it is turned off as part of the Machine Check Error or the NMI handler in software, or with RESET#, BINIT#, and INIT# assertion.

An internal error can be handled in several ways inside the processor based on its power-on configuration. If Machine Check Exception (MCE) is enabled, IERR# causes an MCE entry. IERR# can also be directed on the BERR# pin to indicate an error. Usually BERR# is sampled back by all processors to enter MCE or it can be redirected as an NMI by the central agent.

A.32 IGNNE# (I)

The IGNNE# signal is the Intel Architecture Compatability group Ignore Numeric Error signal. If IGNNE# is asserted, the Pentium Pro processor ignores a numeric error and continues to execute non-control floating-point instructions. If IGNNE# is deasserted, the Pentium Pro processor freezes on a non-control floating-point instruction if a previous instruction caused an error.

IGNNE# has no effect when the NE bit in control register 0 is set.

IGNNE# is an asynchronous input. However, to guarantee recognition of this signal following an I/O write instruction, IGNNE# must be valid along with RS[2:0]# in the Response Phase of the corresponding I/O Write bus transaction. In FRC mode, IGNNE# must be synchronous to BCLK.

During active RESET#, the Pentium Pro processor begins sampling the A20M#, IGNNE# and LINT[1:0] values to determine the ratio of core-clock frequency to bus-clock frequency. See Table 42. After the PLL-lock time, the core clock becomes stable and is locked to the external bus clock. On the active-to-inactive transition of RESET#, the Pentium Pro processor latches A20M# and IGNNE# and freezes the frequency ratio internally. Normal operation on the two signals continues two clocks after RESET# inactive is sampled.

A.33 INIT# (I)

The INIT# signal is the Execution Control group initialization signal. Active INIT# input resets integer registers inside all Pentium Pro processors without affecting their internal (L1 or L2) caches or their floating-point registers. Each Pentium Pro processor begins execution at the power-on reset vector configured during power-on configuration regardless of whether INIT# has gone inactive. The processor continues to handle snoop requests during INIT# assertion.

INIT# can be used to help performance of DOS extenders written for the Intel 80286 processor. INIT# provides a method to switch from protected mode to real mode while maintaining the contents

of the internal caches and floating-point state. INIT# can not be used in lieu of RESET# after power-up.

On active-to-inactive transition of RESET#, each Pentium Pro processor bus agent samples INIT# signals to determine its power-on configuration. Two clocks after RESET# is sampled deasserted, these signals begin normal operation.

INIT# is an asynchronous input. In FRC mode, INIT# must be synchronous to BCLK.

A.34 INTR (I)

The INTR signal is the Interrupt Request signal. The INTR input indicates that an external interrupt has been generated. The interrupt is maskable using the IF bit in the EFLAGS register. If the IF bit is set, the Pentium Pro processor vectors to the interrupt handler after the current instruction execution is completed. Upon recognizing the interrupt request, the Pentium Pro processor issues a single Interrupt Acknowledge (INTA) bus transaction. INTR must remain active until the INTA bus transaction to guarantee its recognition.

INTR is sampled on every rising BCLK edge. INTR is an asynchronous input but recognition of INTR is guaranteed in a specific clock if it is asserted synchronously and meets the setup and hold times. INTR must also be deasserted for a minimum of two clocks to guarantee its inactive recognition. In FRC mode, INTR must be synchronous to BCLK. On power-up the LINT[1:0] signals are used for power-on-configuration of clock ratios. Both these signals must be software configured by programming the APIC register space to be used either as NMI/INTR or LINT[1:0] in the BIOS. Because APIC is enabled after reset, LINT[1:0] is the default configuration.

A.35 LEN[1:0]# (I/O)

The LEN[1:0]# signals are data-length signals. They are transmitted using REQb[1:0]# signals by the request initiator in the second clock of Request Phase. LEN[1:0]# define the length of the data transfer requested by the request initiator as defined in Table 50. The LEN[1:0]#, HITM#, and RS[2:0]# signals together define the length of the actual data transfer.

Table 50. LEN[1:0]# Data Transfer Lengths

LEN[1:0]#	Request Initiator's Data Transfer Length
00	0-8 Bytes
01	16 Bytes
10	32 Bytes
11	Reserved

A.36 LINT[1:0] (I)

The LINT[1:0] signals are the Execution Control group Local Interrupt signals. When APIC is disabled, the LINT0 signal becomes INTR, a maskable interrupt request signal, and LINT1 becomes NMI, a non-maskable interrupt. INTR and NMI are backward compatible with the same signals for the Pentium processor. Both signals are asynchronous inputs. In FRC mode, LINT[1:0] must be synchronous to BCLK.

During active RESET#, the Pentium Pro processor continuously samples the A20M# and IGNNE# values to determine the ratio of core-clock frequency to bus-clock frequency. After the PLL-lock time, the core clock becomes stable and is locked to the external bus clock. On the active-to-inactive transition of RESET#, the Pentium Pro processor freezes the frequency ratio internally. Normal operation on the two signals continues two clocks after RESET# inactive is sampled. Both these signals must be software configured by programming the APIC register space to be used either as NMI/INTR or LINT[1:0] in the BIOS. Because APIC is enabled after reset, LINT[1:0] is the default configuration.

LINT[1:0] is also used for core-to-bus frequency ratio configuration. See Table 42.

A.37 LOCK# (I/O)

The LOCK# signal is the Arbitration group bus lock signal. For a locked sequence of transactions, LOCK# is asserted from the first transaction's Request Phase through the last transaction's Response Phase. A locked operation can be prematurely aborted (and LOCK# deasserted) if AERR# or DEFER# is asserted during the first bus transaction of the sequence. The sequence can also be prematurely aborted if a hard error (such as

a hard failure response or AERR# assertion beyond the retry limit) occurs on any one of the transactions during the locked operation.

When the priority agent asserts BPRI# to arbitrate for bus ownership, it waits until it observes LOCK# deasserted. This enables symmetric agents to retain bus ownership throughout the bus locked operation and guarantee the atomicity of lock. If AERR# is asserted up to the retry limit during an ongoing locked operation, the arbitration protocol ensures that the lock owner receives the bus ownership after arbitration logic is reset. This result is accomplished by requiring the lock owner to reactivate its arbitration request one clock ahead of other agents' arbitration request. LOCK# is kept asserted throughout the arbitration reset sequence.

A.38 NMI (I)

The NMI signal is the Non-maskable Interrupt signal. It is the state of the LINT1 signal when APIC is disabled. Asserting NMI causes an interrupt with an internally supplied vector value of 2. An external interrupt-acknowledge transaction is not generated. If NMI is asserted during the execution of an NMI service routine, it remains pending and is recognized after the IRET is executed by the NMI service routine. At most, one assertion of NMI is held pending.

NMI is rising-edge sensitive. Recognition of NMI is guaranteed in a specific clock if it is asserted synchronously and meets the setup and hold times. If asserted asynchronously, active and inactive pulse widths must be a minimum of two clocks. In FRC mode, NMI must be synchronous to BCLK.

A.39 PICCLK (I)

The PICCLK signal is the Execution Control group APIC Clock signal. It is an input clock to the Pentium Pro processor for synchronous operation of the APIC bus. PICCLK must be synchronous to BCLK in FRC mode.

A.40 PICD[1:0] (I/O)

The PICD[1:0] signals are the Execution Control group APIC Data signals. They are used for bi-directional serial message passing on the APIC bus.

A.41 PWRGOOD (I)

PWRGOOD is driven to the Pentium Pro processor by the system to indicate that the clocks and power supplies are within their specification. See Section 3.9 for additional details. This signal will not affect FRC operation.

A.42 REQ[4:0]# (I/O)

The REQ[4:0]# signals are the Request Command signals. They are asserted by the current bus owner in both clocks of the Request Phase. In the first clock, the REQa[4:0]# signals define the transaction type to a level of detail that is sufficient to begin a snoop request. In the second clock, REQb[4:0]# signals carry additional information to define the complete transaction type. REQb[4:2]# is reserved. REQb[1:0]# signals transmit LEN[1:0]# (the data transfer length information). In both clocks, REQ[4:0]# and ADS# are protected by parity RP#.

All receiving agents observe the REQ[4:0]# signals to determine the transaction type and participate in the transaction as necessary, as shown in Table 51.

Table 51. Transaction Types Defined by REQa#/REQb# Signals

	REQa[4:0]#					REQb[4:0]#				
Transaction	**4**	**3**	**2**	**1**	**0**	**4**	**3**	**2**	**1**	**0**
Deferred Reply	0	0	0	0	0	X	X	X	X	X
Rsvd (Ignore)	0	0	0	0	1	X	X	X	X	X
Interrupt Acknowledge	0	1	0	0	0	DSZ#		X	0	0
Special Transactions	0	1	0	0	0	DSZ#		X	0	1
Rsvd (Central agent response)	0	1	0	0	0	DSZ#		X	1	X
Branch Trace Message	0	1	0	0	1	DSZ#		X	0	0
Rsvd (Central agent response)	0	1	0	0	1	DSZ#		X	0	1
Rsvd (Central agent response)	0	1	0	0	1	DSZ#		X	1	X
I/O Read	1	0	0	0	0	DSZ#		X	LEN#	
I/O Write	1	0	0	0	1	DSZ#		X	LEN#	
Rsvd (Ignore)	1	1	0	0	X	DSZ#		X	X	X
Memory Read & Invalidate	ASZ#		0	1	0	DSZ#		X	LEN#	
Rsvd (Memory Write)	ASZ#		0	1	1	DSZ#		X	LEN#	
Memory Code Read	ASZ#		1	D/C#=0	0	DSZ#		X	LEN#	
Memory Data Read	ASZ#		1	D/C#=1	0	DSZ#		X	LEN#	
Memory Write (may not be retried)	ASZ#		1	W/WB#=0	1	DSZ#		X	LEN#	
Memory Write (may not be retried)	ASZ#		1	W/WB#=1	1	DSZ#		X	LEN#	

A.43 RESET# (I)

The RESET# signal is the Execution Control group reset signal. Asserting RESET# resets all Pentium Pro processors to known states and invalidates their L1 and L2 caches without writing back Modified (M state) lines. For a power-on type reset, RESET# must stay active for at least one millisecond after $V_{CC}P$ and CLK have reached their proper DC and AC specifications. On observing active RESET#, all bus agents must deassert their outputs within two clocks.

A number of bus signals are sampled at the active-to-inactive transition of RESET# for the power-on configuration. The configuration options are described in the *Pentium® Pro Processor Developer's Manual, Volume 1: Specifications*

(Order Number 242690) and in the pertinent signal descriptions in this appendix.

Unless its outputs are tristated during power-on configuration, after active-to-inactive transition of RESET#, the Pentium Pro processor optionally executes its built-in self-test (BIST) and begins program execution at reset-vector 0_000F_FFF0H or 0_FFFF_FFF0H.

A.44 RP# (I/O)

The RP# signal is the Request Parity signal. It is driven by the request initiator in both clocks of the Request Phase. RP# provides parity protection on ADS# and REQ[4:0]#. When a Pentium Pro processor bus agent observes an RP# parity error on any one of the two Request Phase clocks, it must assert AERR# in the Error Phase, provided "AERR# drive" is enabled during the power-on configuration.

A correct parity signal is high if an even number of covered signals are low and low if an odd number of covered signals are low. This definition allows parity to be high when all covered signals are high.

A.45 RS[2:0]# (I)

The RS[2:0]# signals are the Response Status signals. They are driven by the response agent (the agent responsible for completion of the transaction at the top of the In-order Queue). Assertion of RS[2:0]# to a non-zero value for one clock completes the Response Phase for a transaction. The response encodings are shown Table 52. Only certain response combinations are valid, based on the snoop result signaled during the transaction's Snoop Phase.

The RS[2:0]# assertion for a transaction is initiated when all of the following conditions are met:

- All bus agents have observed the Snoop Phase completion of the transaction.
- The transaction is at the top of the In-order Queue.
- RS[2:0]# are sampled in the Idle state

The response driven depends on the transaction as described below:

- The response agent returns a hard-failure response for any transaction in which the response agent observes a hard error.
- The response agent returns a Normal with data response for a read transaction with HITM# and DEFER# deasserted in the Snoop Phase, when the addressed agent is ready to return data and samples inactive DBSY#.
- The response agent returns a Normal without data response for a write transaction with HITM# and DEFER# deasserted in the Snoop Phase, when the addressed agent samples TRDY# active and DBSY# inactive, and it is ready to complete the transaction.

Table 52. Transaction Response Encodings

RS[2:0]	Description	HITM#	DEFER#
000	Idle State.	N/A	N/A
001	Retry Response. The transaction is canceled and must be retried by the initiator.	0	1
010	Defer Response. The transaction is suspended. The defer agent will complete it with a defer reply	0	1
011	RReserved.	0	1
100	Hard Failure. The transaction received a hard error. Exception handling is required.	X	X
101	NNormal without data	0	0

110	II Implicit WriteBack Response. Snooping agent will transfer the modified cache line on the data bus.	1	X
111	Normal with data.	0	0

- The response agent must return an Implicit writeback response in the next clock for a read transaction with HITM# asserted in the Snoop Phase, when the addressed agent samples TRDY# active and DBSY# inactive.
- The addressed agent must return an Implicit writeback response in the clock after the following sequence is sampled for a write transaction with HITM# asserted:
  1. TRDY# active and DBSY# inactive
  2. Followed by TRDY# inactive
  3. Followed by TRDY# active and DBSY# inactive
- The defer agent can return a Deferred, Retry, or Split response anytime for a read transaction with HITM# deasserted and DEFER# asserted.
- The defer agent can return Deferred, Retry, or Split response when it samples TRDY# active and DBSY# inactive for a write transaction with HITM# deasserted and DEFER# asserted.

A.46 SMI# (I)

System Management Interrupt is asserted asynchronously by system logic. On accepting a System Management Interrupt, the Pentium Pro processor saves the current state and enters SMM mode. It issues an SMI Acknowledge Bus transaction and then begins program execution from the SMM handler.

A.47 RSP# (I)

The RSP# signal is the Response Parity signal. It is driven by the response agent during assertion of RS[2:0]#. RSP# provides parity protection for RS[2:0]#.

A correct parity signal is high if an even number of covered signals are low and low if an odd number of covered signals are low. During Idle state of RS[2:0]# (RS[2:0]#=000), RSP# is also high since it is not driven by any agent guaranteeing correct parity.

Pentium Pro processor bus agents can check RSP# at all times and if a parity error is observed, treat it as a protocol violation error. If the BINIT# driver is enabled during configuration, the agent observing RSP# parity error can assert BINIT#.

A.48 SMMEM# (I/O)

The SMMEM# signal is the System Management Mode Memory signal. It is driven on the second clock of the Request Phase on the EXF4#/Ab7# signal. It is asserted by the Pentium Pro processor to indicate that the processor is in System Management Mode and is executing out of SMRAM space.

A.49 SPLCK# (I/O)

The SPLCK# signal is the Split Lock signal. It is driven in the second clock of the Request Phase on the EXF3#/Ab6# signal of the first transaction of a locked operation. It is driven to indicate that the locked operation will consist of four locked transactions. Note that SPLCK# is asserted only for locked operations and only in the first transaction of the locked operation.

A.50 STPCLK# (I)

The STPCLK# signal is the Stop Clock signal. When asserted, the Pentium Pro processor enters a low power state, the stop-clock state. The processor issues a Stop Clock Acknowledge special transaction, and stops providing internal clock signals to all units except the bus unit and the APIC unit. The processor continues to snoop bus transactions and service interrupts while in stop clock state. When STPCLK# is deasserted, the processor restarts its internal clock to all units and resumes execution. The assertion of STPCLK# has no effect on the bus clock.

STPCLK# is an asynchronous input. In FRC mode, STPCLK# must be synchronous to BCLK.

A.51 TCK (I)

The TCK signal is the System Support group Test Clock signal. TCK provides the clock input for the test bus (also known as the test access port). Make certain that TCK is active before initializing the TAP.

A.52 TDI(I)

The TDI signal is the System Support group test-data-in signal. TDI transfers serial test data into the Pentium Pro processor. TDI provides the serial input needed for JTAG support.

A.53 TDO (O)

The TDO signal is the System Support group test-data-out signal. TDO transfers serial test data out from the Pentium Pro processor. TDO provides the serial output needed for JTAG support.

A.54 TMS (I)

The TMS signal is an additional System Support group JTAG-support signal.

A.55 TRDY (I)

The TRDY# signal is the target Ready signal. It is asserted by the target in the Response Phase to indicate that the target is ready to receive write or implicit writeback data transfer. This enables the request initiator or the snooping agent to begin the appropriate data transfer. There will be no data transfer after a TRDY# assertion if a write has zero length indicated in the Request Phase. The data transfer is optional if an implicit writeback occurs for a transaction which writes a full cache line (the Pentium Pro processor will perform the implicit writeback).

TRDY# for a write transaction is driven by the addressed agent when:

- When the transaction has a write or writeback data transfer
- It has a free buffer available to receive the write data
- A minimum of 3 clocks after ADS# for the transaction
- The transaction reaches the top-of-the-In-order Queue
- A minimum of 1 clock after RS[2:0]# active assertion for transaction "n-1". (After the transaction reaches the top of the In-order Queue).

TRDY# for an implicit writeback is driven by the addressed agent when:

- The transaction has an implicit writeback data transfer indicated in the Snoop Result Phase.
- It has a free cache line buffer to receive the cache line writeback
- If the transaction also has a request initiated transfer, that the request initiated TRDY# was asserted and then deasserted (TRDY# must be deasserted for at least one clock between the TRDY# for the write and the TRDY# for the implicit writeback),
- A minimum of 1 clock after RS[2:0]# active assertion for transaction "n-1". After the transaction reaches the top of the In-order Queue).

TRDY# for a write or an implicit writeback may be deasserted when:

- Inactive DBSY# and active TRDY# are observed.
- DBSY# is observed inactive on the clock TRDY# is asserted.
- A minimum of three clocks can be guaranteed between two active-to-inactive transitions of TRDY#
- The response is driven on RS[2:0]#.
- Inactive DBSY# and active TRDY# are observed for a write, and TRDY# is required for an implicit writeback.

TRST (I)

The TRST# signal resets the JTAG logic.

intel®

AP-525

APPLICATION NOTE

1

Pentium® Pro Processor Thermal Design Guidelines

December 1995

Order Number: 242766-001

Pentium® PRO PROCESSOR THERMAL DESIGN GUIDELINES

CONTENTS

CONTENTS

PAGE

FIGURES

TABLES

1.0. INTRODUCTION

In a system environment, the processor's temperature is a function of both the system and component thermal characteristics. The system level thermal constraints include the local ambient temperature at the processor and the airflow over the processor(s), as well as the physical constraints at and above the processor(s). The processor's case temperature depends on the component's power dissipation and size, the effective thermal conductivity of the packaging material, the type of interconnection to the printed circuit board (PCB), the presence of a thermal cooling solution, and the thermal conductivity and power density of the PCB.

All of these parameters are aggravated by the continued push of technology to increase performance levels (higher operating speeds, MHz) and packaging density (more transistors). As operating frequencies increase and packaging size decreases, the power density increases and the thermal cooling solution space and airflow become more constrained. The result is an increased importance on system design to ensure that thermal design requirements are met for each component in the system.

1.1. Document Goals

The Pentium® Pro processor is the next generation in the Intel386™, Intel486™, and Pentium families of microprocessors. The Pentium Pro microprocessor generates sufficient heat to require some attention in order to meet the case temperature specification in system designs. The goal of this document is to provide an understanding of the thermal characteristics of the Pentium Pro processor, and to discuss guidelines for meeting the thermal requirements imposed on single and multiple processor systems.

In the future, Intel will provide an OverDrive® processor to upgrade single and dual Pentium Pro processor systems. The OverDrive processor will ship with an integrated fan/heat sink for thermal management. Guidelines for meeting the thermal specifications of the OverDrive processor are also included in this document.

1.2. References

The *Pentium® Pro Processor Developer's Manual, Volume 1* (Order Number 242690) is referenced throughout this document.

2.0. IMPORTANCE OF THERMAL MANAGEMENT

The objective of thermal management is to ensure that the temperature of all components in a system is maintained within functional and absolute maximum limits. The functional temperature limit is the range within which the electrical circuits can be expected to meet their specified performance requirements. Operation outside the functional limit can degrade system performance, or cause logic errors, component and/or system damage. The absolute maximum temperature limit is the highest temperature to which the component may be safely exposed. Temperatures exceeding the maximum operating limits may result in irreversible changes in the operating characteristics of the component.

3.0. PENTIUM® PRO PROCESSOR PACKAGE SPECIFICATIONS

The Pentium Pro processor is packaged in a 2.46" x 2.66", 387 pin ceramic pin grid array (PGA) with a gold plated copper tungsten heat spreader. The pins are arranged in a modified staggered array. Please see the *Pentium® Pro Processor Developer's Manual, Volume 1* for complete dimensions.

4.0. PENTIUM® PRO PROCESSOR POWER SPECIFICATIONS

The Pentium Pro processor power dissipation can be found in the *Pentium® Pro Processor Developer's Manual, Volume 1*. It is highly recommended that systems be designed to dissipate 40W per processor, as this will allow the same design to accommodate higher frequency or otherwise enhanced members of the Pentium Pro family.

To ensure proper operation and reliability of the Pentium Pro processor, the maximum device case temperature must remain within the specification in the *Pentium® Pro Processor Developer's Manual, Volume 1*. A typical case temperature is 85°C. Considering the power dissipation levels and typical ambient environments of 35°C to 45°C, the Pentium Pro processor's specified case temperature cannot be maintained without additional thermal enhancement to dissipate the heat generated. The OverDrive processor for upgrading Pentium Pro processor systems will be equipped with an integrated fan/heat sink, and will remain within its specified temperature limits provided the fan/heat sink air inlet temperature does not exceed the specified temperature in the *Pentium® Pro*

Processor Developer's Manual, Volume 1. This air temperature is typically 50°C.

The thermal characterization data described in later sections illustrates that both a thermal cooling device and system airflow are needed. The size and type (passive or active) of thermal cooling device and the amount of system airflow are interrelated and can be varied to meet specific system design constraints. In typical systems, the thermal solution size and type is limited by board layout, spacing, and component placement. Airflow is determined by the size and number of fans along with their placement in relation to the components and the airflow channels within the system. Acoustic noise constraints may also limit the size and/or types of fans that can be used in a particular design.

To develop a reliable, cost-effective thermal solution, all of the above variables must be considered. Thermal characterization and simulation should be carried out at the entire system level, accounting for the thermal requirements of each component.

5.0. THERMAL PARAMETERS

Component power dissipation results in a rise in temperature relative to the temperature of a reference point. The amount of rise in temperature depends on the net thermal resistance between the component's package and the reference point. Thermal resistance is the key factor in determining the power handling capability of any electronic package.

5.1. Case Temperature

To ensure functionality and reliability, the Pentium Pro processor is specified for proper operation when case temperature (T_C) is within a specified range: typically 0°C to 85°C. Special care is required when measuring the case temperature to ensure an accurate temperature measurement. Thermocouples are often used to measure T_C. When measuring the temperature of a surface which is at a different temperature from the surrounding ambient air, errors could easily be introduced into the measurements. Measurement errors may occur due to a poor thermal contact between the thermocouple junction and the surface, or heat loss by radiation or conduction through the thermocouple leads. To minimize the measurement errors, the following approach is recommended:

- Use a 35 gauge K-type thermocouple or equivalent.
- Ensure that the thermocouple has been properly calibrated.
- Attach the thermocouple bead or junction to the package top surface at a location corresponding to the center of the Pentium Pro processor die (location Ⓐ in Figure 1). Using the center of the Pentium Pro processor die gives a more accurate measurement and less variation as the boundary condition changes.
- Attach the thermocouple bead or junction at a 90° angle by an adhesive bond to the package top surface as shown in Figure 2. When a heat sink is attached, a hole should be drilled through the heat sink to allow a probe to reach the package directly above the center of the Pentium Pro processor die. The hole diameter should be no larger than 0.150".

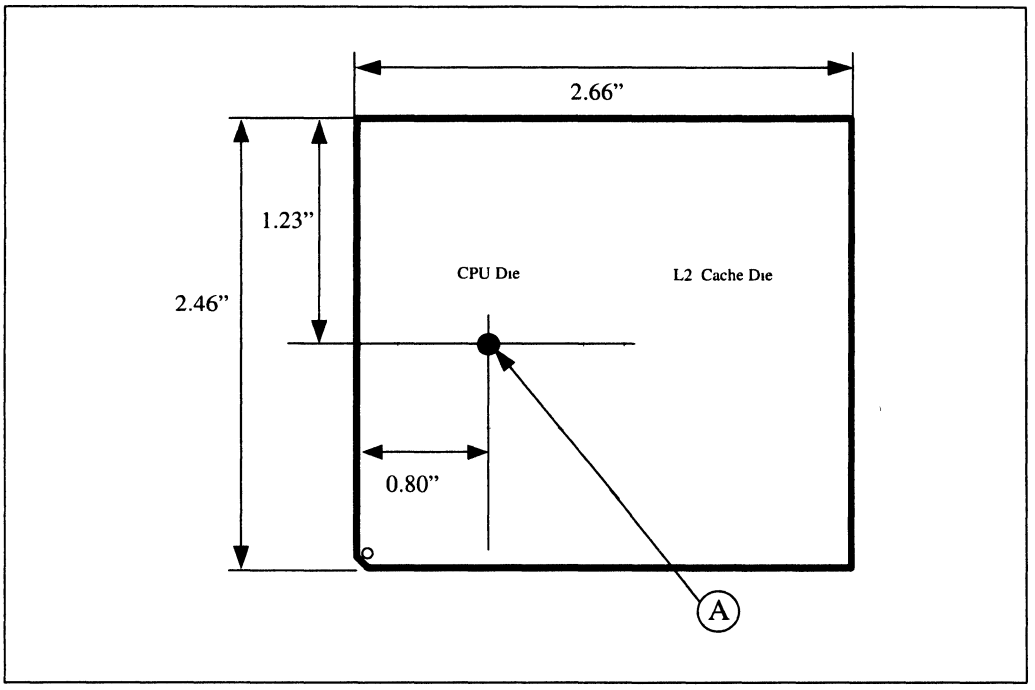


Figure 1. Location of Case Temperature Measurement (Top-Side View)

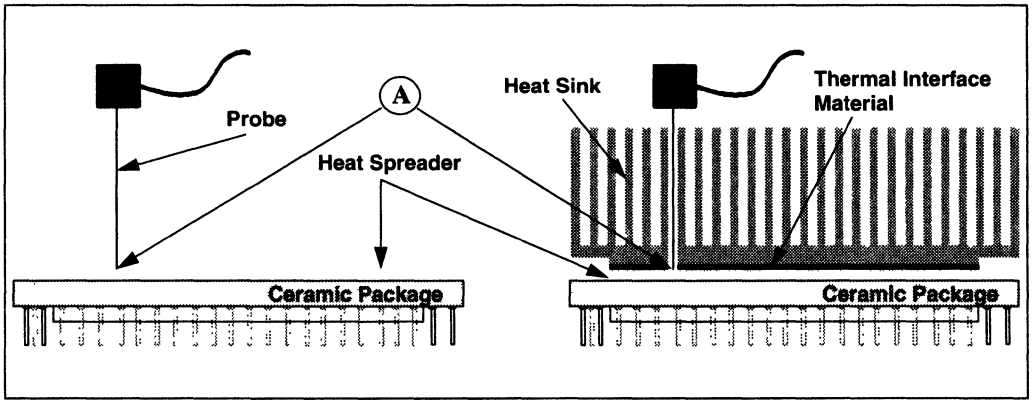


Figure 2. Thermocouple Placement

5.2. Ambient Temperature

Ambient temperature (T_A) is the temperature of the ambient air surrounding the package. In a system environment, ambient temperature is the temperature of the air upstream of the package and in its close vicinity. If an active cooling solution is used, T_A is the temperature at the inlet to the active cooling device.

The OverDrive processor for upgrading Pentium Pro processor-based systems specifies a T_A for its integrated fan/heat sink assembly. However, an ambient temperature is not directly specified for the Pentium Pro processor. The only restriction is that case temperature (T_C) is met. To determine the allowable T_A values, the following equations may be used:

$$T_C = T_A + (P_D * \Theta_{CA})$$

Where: T_A = Ambient temperature (°C)
T_C = Case temperature of the device (°C)
P_D = Total power dissipated by the dies (W)
Θ_{CA} = Case-to-ambient thermal resist. (°C/W)

5.3. Thermal Resistance

The case-to-ambient thermal resistance value (Θ_{CA}) is used as a measure of the cooling solution's thermal performance. Θ_{CA} is comprised of the case-to-sink thermal resistance (Θ_{CS}) and the sink-to-ambient thermal resistance (Θ_{SA}). Θ_{CS} is a measure of the thermal resistance along the heat flow path from the top of the IC package to the bottom of the thermal cooling solution. This value is strongly dependent on the material, conductivity, and thickness of the thermal interface used. Θ_{SA} is a measure of the thermal resistance from the bottom of the heat sink to the local ambient air. Θ_{SA} values depend on the material, thermal conductivity, and geometry of the thermal cooling solution as well as on the airflow rates.

The parameters are defined by the following relationships (see Figure 3):

$$\Theta_{CA} = (T_C - T_A) / P_D$$

$$\Theta_{CA} = \Theta_{CS} + \Theta_{SA}$$

Where: Θ_{CA} = Case-to-ambient thermal resist. (°C/W)
Θ_{CS} = Case-to-sink thermal resistance (°C/W)
Θ_{SA} = Sink-to-ambient thermal resist. (°C/W)
T_C = Case temperature of the device (°C)
T_A = Ambient temperature (°C)
P_D = Total power dissipated by dies (W)

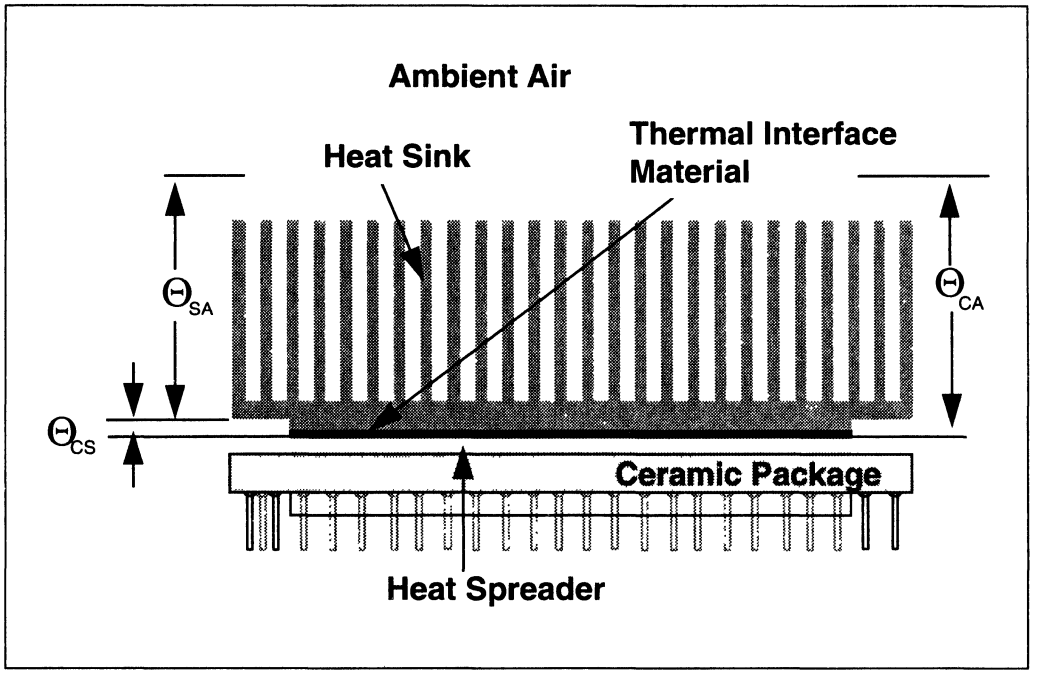


Figure 3. Thermal Resistance Relationships

5.3.1. ANALYSIS ASSUMPTIONS

The following example thermal analysis of a Pentium Pro processor package is based on the following assumptions:

- Power dissipation of the Pentium Pro processor die is 24W
- Power dissipation of the L2 cache die is 4W
- Maximum case temperature is 85°C

The actual specifications can be found in the *Pentium® Pro Processor Developer's Manual, Volume 1*. Table 1 below lists the Pentium Pro processor's case-to-ambient thermal resistance for different airflow rates and heat sink heights. Table 2 translates this to the ambient temperature required for different airflow rates and heat sink heights. This information is also charted in Figure 5. In addition, Figure 6 charts power dissipation versus heat sink height.

Table 1. Case-to Ambient Thermal Resistance

	Θ_{CA} [° C/W] vs. Airflow [Linear Feet per Minute] and Heat Sink Height[1]					
Airflow (LFM):	**100**	**200**	**400**	**600**	**800**	**1000**
With 0.5" Heat sink[2]	----	3.16	2.04	1.66	1.41	1.29
With 1.0" Heat sink[2]	2.55	1.66	1.08	0.94	0.80	0.76
With 1.5" Heat sink[2]	1.66	1.31	0.90	0.78	0.71	0.67
With 2.0" Heat sink[2]	1.47	1.23	0.87	0.75	0.69	0.65

NOTES:

1. All data taken at sea level. For altitudes above sea level, it is recommended that a derating factor of 1°C/1000 feet be used.
2. Heat sink: 2.235" square omni-directional pin, aluminum heat sink with a pin thickness of 0.085", a pin spacing of 0.13" and a base thickness of 0.15". See Figure 4. A thin layer of thermal grease (Thermoset TC208 with thermal conductivity of 1.2 W/m-°K) was used as the interface material between the heat sink and the package.

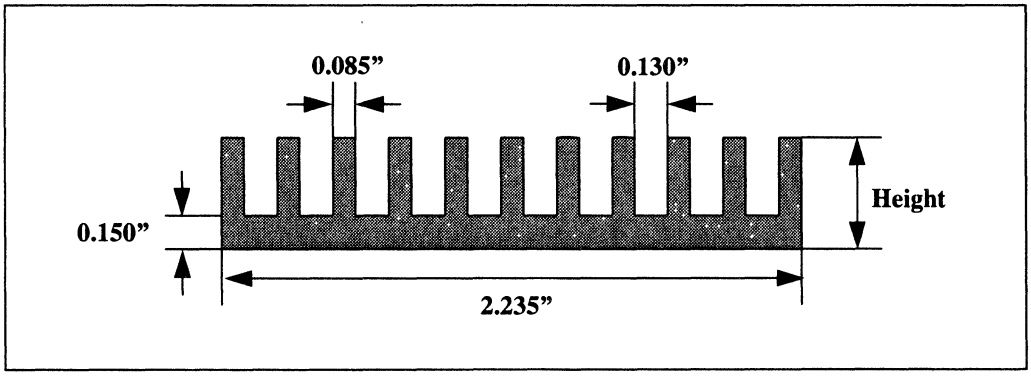


Figure 4. Side View Of Omni-Directional Pin Fin Heat Sink

Table 2. Case-to Ambient Thermal Resistance

	T_A vs. Airflow [Linear Feet per Minute] and Heat Sink Height[1]					
Airflow (LFM):	**100**	**200**	**400**	**600**	**800**	**1000**
With 0.5" Heat sink[2]	----	-4	27	38	45	48
With 1.0" Heat sink[2]	13	38	54	58	62	63
With 1.5" Heat sink[2]	38	48	59	63	65	66
With 2.0" Heat sink[2]	43	50	60	64	65	66

NOTES:

1. All data taken at sea level. For altitudes above sea level, it is recommended that a derating factor of 1°C/1000 feet be used.
2. Heat sink: 2.235" square omni-directional pin, aluminum heat sink with a pin thickness of 0.085", a pin spacing of 0.13" and a base thickness of 0.15". See Figure 4. A thin layer of thermal grease (Thermoset TC208 with thermal conductivity of 1.2 W/m-°K) was used as the interface material between the heat sink and the package.

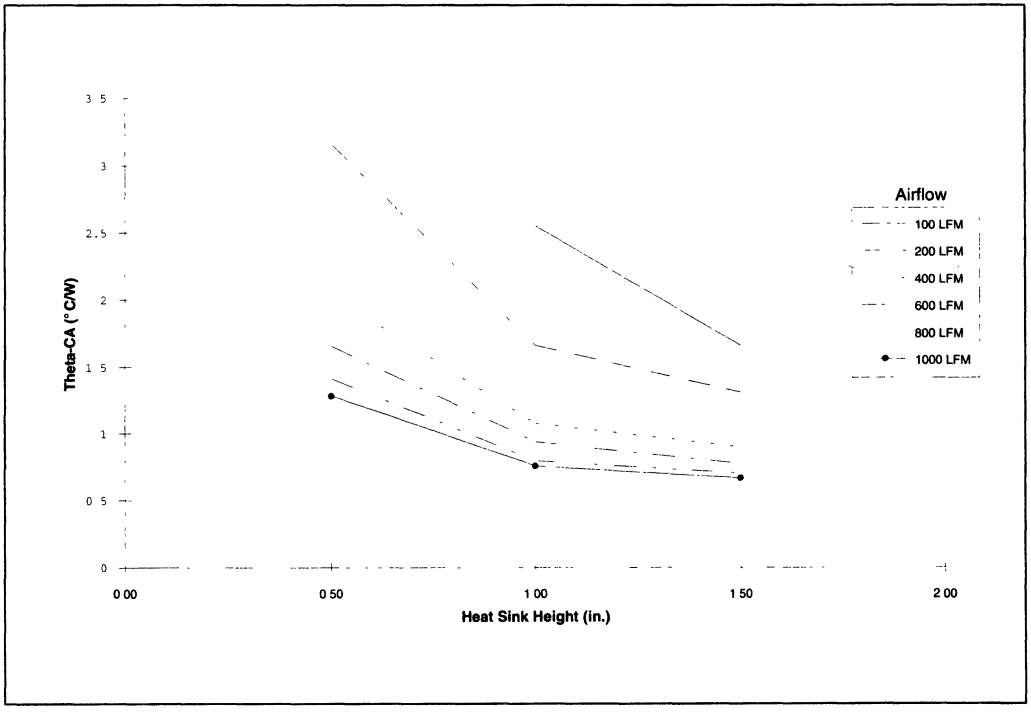


Figure 5. Θ_{CA} Versus Heat Sink Height

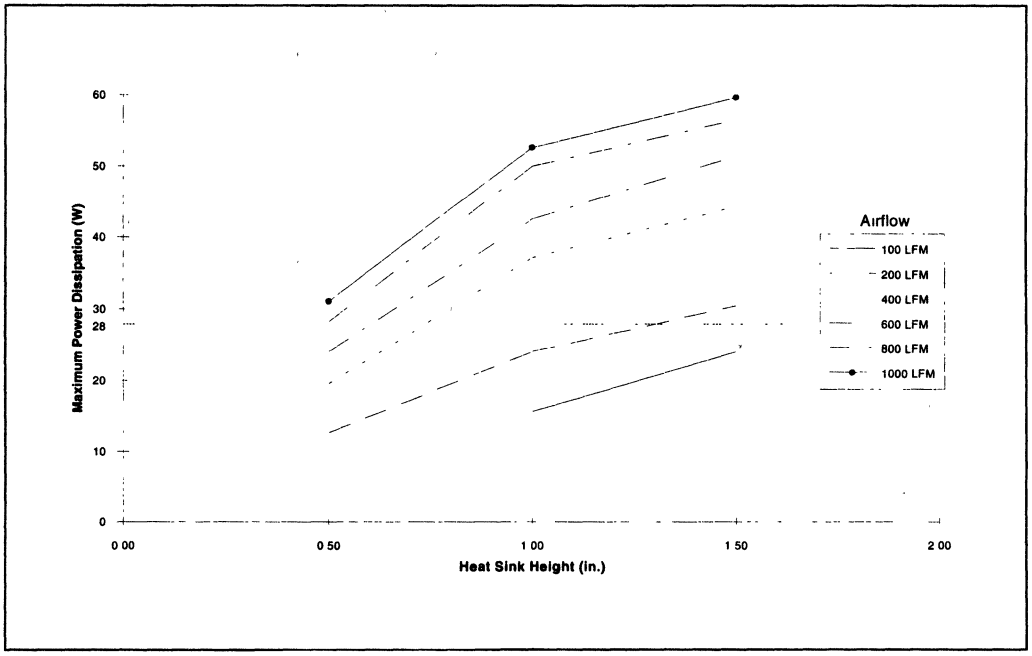


Figure 6. Power Dissipation Versus Heat Sink Height (T_A=45°C)

6.0. DESIGNING FOR THERMAL PERFORMANCE

The Pentium Pro processor specifies a maximum case temperature, T_C. This case temperature limit, along with external ambient temperature and the Pentium Pro processor's power specification, can be used to determine the case-to-ambient thermal resistance of the cooling solution required to keep the Pentium Pro processor within its operational limits.

Figure 7 shows a simple model for use in calculating various thermal parameters based on known values. This model can theoretically be extended to any number of processors.

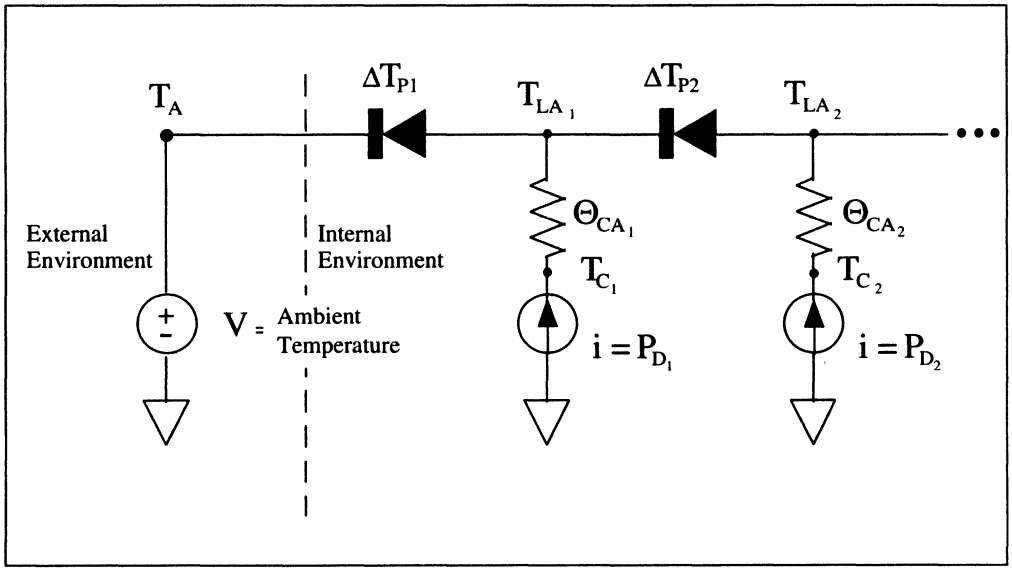


Figure 7. Simple Thermal Model

The equations governing this simple model are:

$$T_{C_1} = T_A + \Delta T_{P_1} + (P_{D_1} * \Theta_{CA_1})$$
$$T_{C_2} = T_A + \Delta T_{P_1} + \Delta T_{P_2} + (P_{D_2} * \Theta_{CA_2})$$

where: T_A = External ambient temperature (°C)
T_{LA} = Local ambient temperature (°C)
T_C = Case temperature of the device (°C)
P_D = Power dissipated by the device under test
Θ_{CA} = Case-to-ambient thermal resist. (°C/W)
ΔT_P = Temperature rise between elements

6.1. Airflow Management

To maximize the amount of air that flows over the processor and minimize the local ambient temperature (T_{LA}) near the processor, it is important to manage the amount of air that flows within the system as well as how it flows. Total system air flow can be increased by adding one or more fans to the system, or by increasing the output of an existing system's fan(s). An important consideration in airflow management is the temperature of the air flowing over the processor(s). Heating effects from add-in boards, DRAM, and disk drives greatly reduce the cooling efficiency of this air, as does recirculation of warm interior air through the system fan. Care must be taken to minimize the heating effects of peripheral components, and to eliminate warm air recirculation.

For example, a clear air path from the external system vents to the system fan(s) will enable the warm air from the Pentium Pro processors to be efficiently pulled out of the system. If no air path exists across the processors, the warm air from the Pentium Pro processors will not be removed from the system, resulting in localized heating ("hot spots") around the processors. Figure 8 shows two examples of air exchange through a PC style chassis. The system on the left is an example of good air exchange, incorporating both the power supply fan, and an additional system fan. The system on the right shows a poorly vented system, using only the power supply fan to move the air, resulting in inadequate air flow. Recirculation of warm air is most common between the system fan and chassis, and between the system fan intake and the drive bays behind the front bezel. These paths may be eliminated by mounting the fan flush to the chassis, obstructing the flow between the drive bays and fan inlet, and by providing generous intake vents in both the chassis and the front bezel.

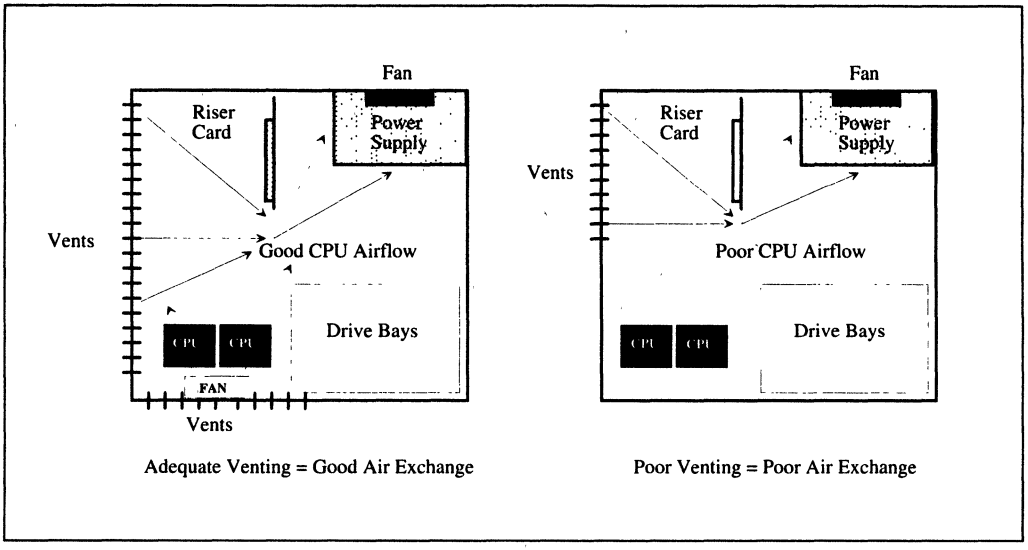


Figure 8. Example of Air Exchange Through a PC Chassis

6.2. Extruded Heat Sink Solutions

One method used to improve case-to-ambient thermal performance is to increase the surface area of the device, by attaching a metallic heat sink to the ceramic package. Heat sinks are generally extruded from blocks of metal, usually aluminum (due to its low price/performance ratio). To maximize the thermal conduction, the thermal resistance from the heat sink to the air can be reduced by maximizing the airflow through the heat sink fins as well as by maximizing the surface area of the heat sink itself.

6.2.1. DESIGN EXAMPLE #1

In Example #1, two Pentium Pro processors are placed side-by-side in front of a dedicated 80mm fan. The fan pulls air from outside the system chassis and pushes the air over the processors, providing approximately 200 LFM of laminar airflow over the processors, as shown in Figure 9. Using the simple thermal model in Figure 7 and assuming a maximum external ambient temperature of 45°C, a maximum case temperature of 85°C, a maximum power dissipation of 28 Watts, and a local processor ambient of 50°C, the following equation can be used to calculate the Θ_{CA} required for either of the two processors in this particular system:

$$\Theta_{CA} = (T_C - T_{LA})/P_D$$

Where: $T_{LA} = T_A + \Delta T_P$

Solving the equation shows that the case-to-ambient thermal resistance required in this particular Pentium Pro processor system is 1.25°C/W. Using Table 1, it can be seen that a 1.0" heat sink with an airflow between 200 and 400 LFM will meet the Pentium Pro processor maximum case temperature limit of 85°C. Linearly extrapolating the values in Table 1 gives an airflow of approximately 340 LFM. To keep the airflow within the 200 LFM assumption, a heat sink approximately 1.9" in height would be required.

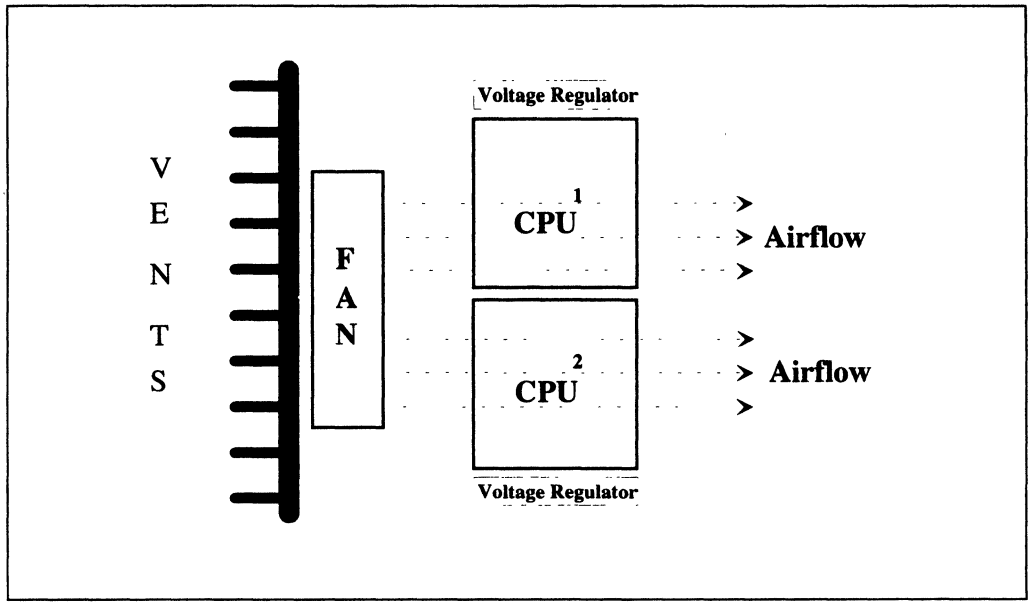


Figure 9. Top View of Design Example #1

6.2.2. DESIGN EXAMPLE #2

In Example #2, the processors are placed in a row directly in front of the system fan. There is a dedicated 80 mm fan directly in front of the first processor providing approximately 600 LFM of laminar airflow (see Figure 10). Using the simple thermal model in Figure 7 and assuming a maximum external ambient temperature of 45°C, a maximum case temperature of 85°C, a maximum power dissipation of 28 Watts, and a local processor ambient of 50°C, the following equation can be used to calculate the Θ_{CA_1} required to cool the first processor in this system:

$$\Theta_{CA_1} = (T_{C_1} - T_{LA_1})/P_{D_1}$$

Where: $T_{LA_1} = T_A + \Delta T_{P_1}$

Solving the equation shows that the case-to-ambient thermal resistance required to cool the first Pentium Pro processor in this system is 1.25°C/W. Referring to Table 1, it can be seen that with 600 LFM, a heat sink between 0.5" and 1.0" meets the Pentium Pro maximum case temperature limit of 85°C. Linearly extrapolating the values in Table 1 gives an a heat sink height of approximately 0.87".

Heat from the first processor will elevate the air temperature over the second processor. For the purpose of this example, it will be assumed that the temperature elevation due to the first processor is 15°C, resulting in an air temperature of 60°C at the second processor (the exact temperature rise due to the first processor is system dependent. Individual designs must be characterized to determine the temperature rise at the second processor).

In addition, the first Pentium Pro processor will disrupt and block the airflow from the dedicated fan, affecting the air reaching the second processor. Individual system designs must be characterized to understand the airflow over the second processor. Also, since the airflow is not necessarily laminar airflow, the values in Table 1 may not be accurate. For the purpose of this example, a 400 LFM airflow over the second processor will be assumed. Using the simple electrical model in Figure 7, the following equation can be used to calculate the Θ_{CA_2} required to cool the second processor in this system:

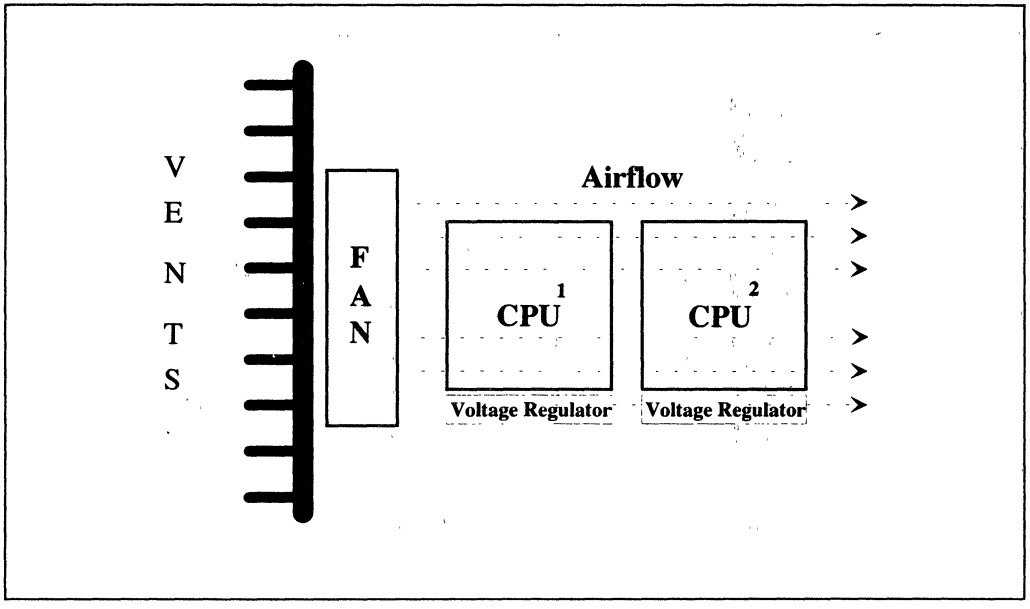


Figure 10. Top View of Design Example #2

$$T_{C_2} = T_{LA_1} + \Delta T_{P_2} + (P_{D_2} * \Theta_{CA_2})$$

Where: T_P = Air temperature at the device under test
$85°C = 60°C + (28 \text{ Watts} * \Theta_{CA_2})$
$\Theta_{CA_2} = 0.89°C/\text{Watt}$

Using $\Theta_{CA} = 0.89°C/\text{Watt}$ and Table 1, it can be seen that a 1.5" heat sink with an airflow of 400 LFM will meet the Pentium Pro processor maximum case temperature limit of 85°C.

6.3. Fans

Fans are often needed to assist in moving the air inside a chassis. The airflow rate of a fan is usually directly related to the acoustic noise level of the fan and system. Maximum acceptable noise levels may limit the fan output or the number of fans selected for a system.

Fan/heat sink assemblies are one type of advanced solution which can be used to cool the Pentium Pro microprocessor. The OverDrive processor for upgrading Pentium Pro processor-based systems will use an integrated fan/heat sink. Intel has worked with fan/heat sink vendors and computer manufacturers to make fan/heat sink cooling solutions available in the industry for use with the Pentium Pro processor in systems. Please consult such a vendor to acquire the proper solution for your needs.

6.3.1. DESIGN EXAMPLE #3

Example #3 shows a single Pentium Pro processor cooled with a fan/heat sink assembly (see Figure 11). The simple thermal model in Figure 7 can be used to calculate the required Θ_{CA} to cool a 28 Watt Pentium Pro processor. Assuming a maximum external ambient temperature of 45°C, a maximum case temperature of 85°C, and a temperature rise from the external ambient to the Pentium Pro processor of 5°C, the following equation can be used:

$$\Theta_{CA_1} = (T_{C_1} - T_{LA_1})/P_{D_1}$$

Where: $T_{LA_1} = T_A + \Delta T_{P_1}$

Solving the equation shows that the fan/heat sink assembly must be able to produce a case-to-ambient thermal resistance of 1.25°C/W to cool the Pentium Pro processor in this system. This calculation can be extended to include multiple Pentium Pro processors in a system.

Note that the OverDrive processor is designed to meet its thermal requirements if the fan/heat sink inlet air temperature is 50°C or less.

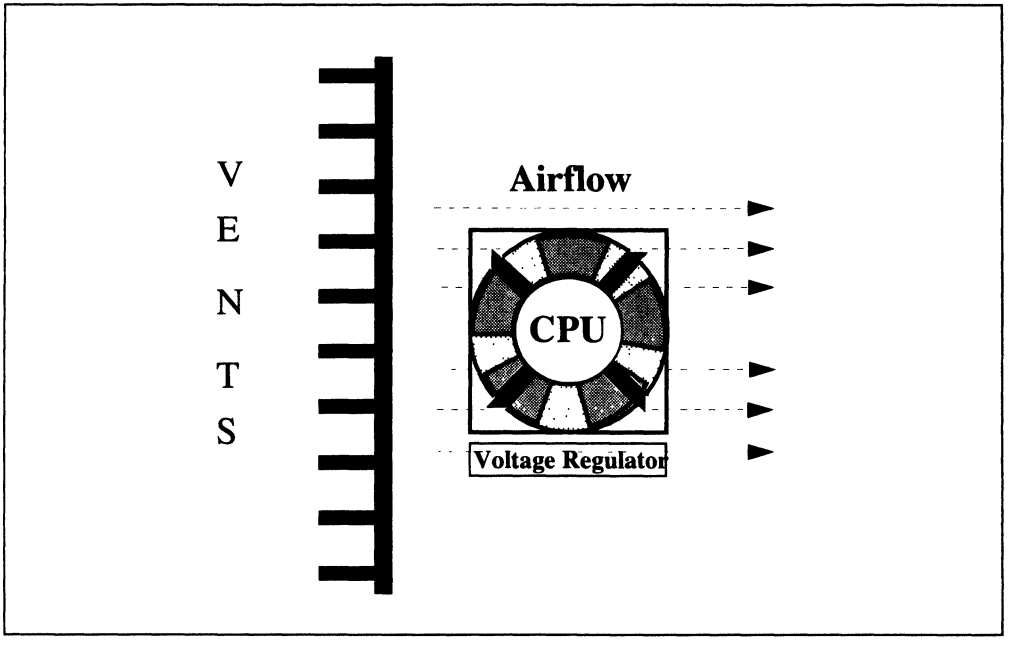


Figure 11. Top View of Design Example #3

6.4. Alternative Cooling Solutions

In addition to extruded heat sinks and system fans, other solutions exist for cooling integrated circuit devices. For example, ducted blowers, heat pipes and liquid cooling are all are capable of dissipating additional heat. Due to their varying attributes, each of these solutions may be appropriate for a particular system implementation.

6.4.1. DESIGN EXAMPLE #4

The Pentium Pro processor is designed to provide scaleable performance in a multiprocessor system; up to four processors can be used in such a configuration. In order to minimize interconnect delays between the processors in an MP environment, the processors must be connected in close proximity. Example #4 demonstrates the use of low cost, high density, extruded heat sinks enclosed in a duct to provide the thermal headroom necessary to cool four Pentium Pro processors while maintaining case temperatures of 85°C. Note that since the OverDrive processor will upgrade single or dual processor Pentium Pro processor-based systems, cooling solution compatibility with the OverDrive processor need only be considered for those system configurations.

6.4.1.1. AIRFLOW CHARACTERIZATION

In this example, two Pentium Pro processors are placed side by side upstream, and two side by side downstream to make a quad processor unit. Each processor's power is locally supplied by a DC/DC converter (see Figure 12). Extruded fin heat sinks with a high aspect ratio (i.e. 12:1 fin height to gap width), 1.2" tall, and with dimensions defined by the plan area of the CPU package are mounted on each of the Pentium Pro processors. The outline of the duct is designed to ensure that the air from the blower is directed through the heat sinks. Air is purged from the system via vents located behind the downstream processors. A single 120 mm blower, mounted in front of the CPU module, provides the airflow through the assembly. For this example, it is recommended that the intake and outlet vents in the chassis be at least 50 percent open.

Assuming a cooling target of 40 Watts per processor (with the processors as the maximum heat-generating parts in the assembly), this duct was designed to isolate the processors from the effects of system heating (such as from add-in cards), and to maximize the processor

cooling temperature budget. Air provided by the blower is split into two parallel paths to cool the processors. A one dimensional model based on drag and frictional losses was developed, using airflow resistance elements to evaluate the operating points of various air moving devices. Although the duct is not symmetrical, assuming equal amounts of airflow through the two parallel paths provides a reasonable approximation of the operating point (the actual airflow delivered by a device attached to the duct in a system environment). Fan and blower manufacturers provide performance curves, characterized by the amount of airflow that the device can deliver at different resistances to the flow (represented by static pressure). At maximum static pressure, the amount of airflow delivered is usually zero, and airflow (measured in cubic feet per minute, or CFM) at no static pressure is the maximum theoretically achievable. The practically realizable airflow through the ducted environment is dependent on the heat sinks' resistance to airflow. The amount of airflow delivered by a fan or blower is the intersection point between the performance curve of the device (provided by the device's manufacturer) and the system resistance curve, which is calculated by the friction and drag contributions of the various resistance elements in the flow path. Energy is spent to move air through bends, expansions, and contractions (which contribute to drag), and friction from surfaces of duct walls and heat sink fins. Shown in Figure 13 are the resistance characteristics for a single air moving device in conjunction with a single high density extruded heat sink (60 mil thick fins, 90 mil fin gap). Maximizing friction and minimizing the drag contribution is desirable.

For comparison purposes, a typical 120mm system fan is also plotted. Axial fans are typically characterized by a high volumetric flow rate, but low pressure drop in the system. In contrast, a blower has a significantly higher pressure drop characteristic, with a lower flow rate, for identical dimensions. The system resistance curve (friction and drag contributions for the ducted cooling scheme) has a significant slope. The fan shown in this example is typically rated at 100 CFM, and the blower is rated at 25 CFM. In the absence of the analysis above, it is common to assume that a system fan provides significantly higher airflow, which translates to improved cooling; however, the effective airflow rate delivered by the 100 CFM fan is about 8 CFM, while the blower provides twice this effective flow rate. This translates to 14 CFM, or 7 CFM through each of the two parallel paths within the duct.

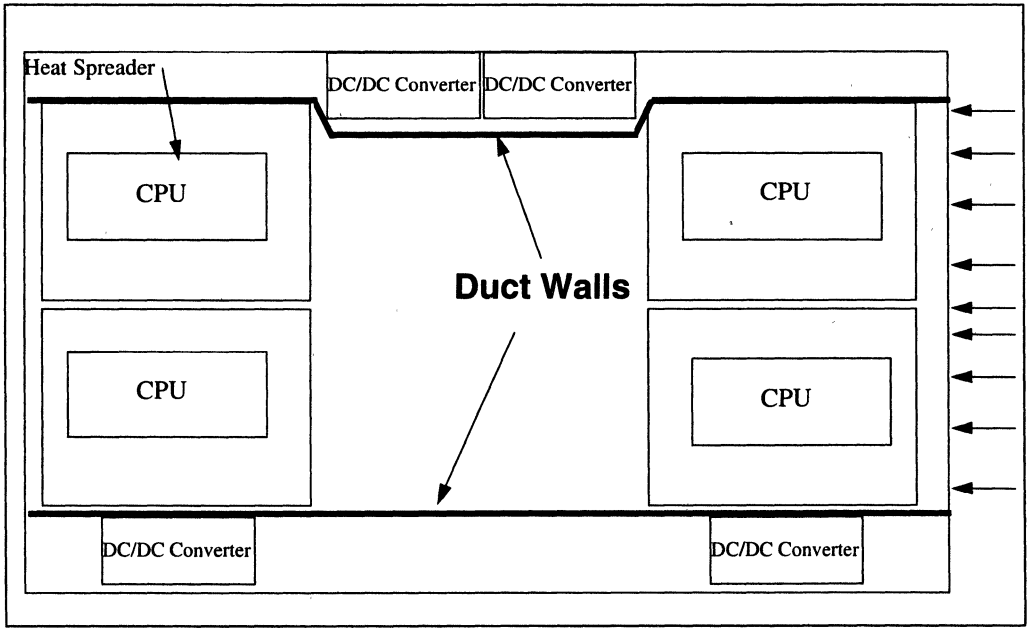


Figure 12. Top View of Design Example #4

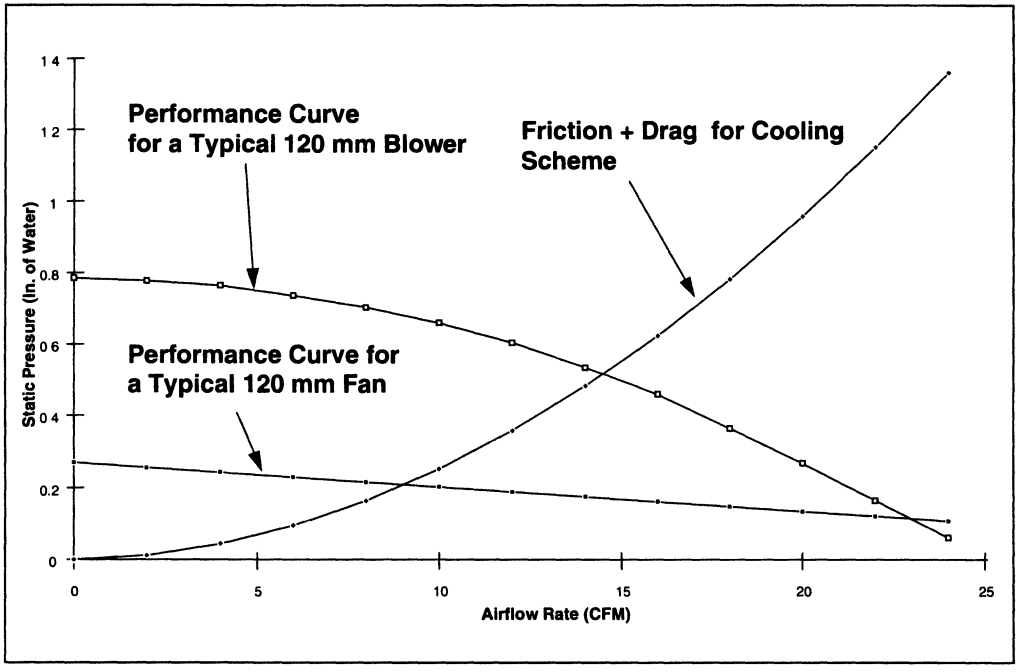


Figure 13. Effective Flow Rate for Blower and Fan for the Ducted Cooling Scheme

6.4.1.2. THERMAL PERFORMANCE CHARACTERIZATION

The physical constraints of this layout require that at least one Pentium Pro processor be placed downstream of the other. In addition, all the air entering the duct must pass through both the upstream and downstream heat sinks. The power dissipation capability of this cooling scheme can be assessed using flat plate heat transfer correlation.

Figure 14 shows temperature predictions at different air flow rates through the heat sinks, assuming a power dissipation of 30 Watts per Pentium Pro processor. The overall flow rate from the blower, based on the performance curves above, is assumed to be split equally between two parallel paths on the module. Since the upstream and downstream heat sinks are at the opposite ends of the module, it is reasonable to expect the air entering the downstream heat sinks has an average mixed temperature, which will be higher than the ambient temperature of the upstream heat sinks. This difference is particularly significant at low flow rates. Figure 15 shows the case temperature values plotted against different air flow rates through the high density heat sinks, using this one dimensional model, at sea level. As the airflow rate increases, more air is available to purge the warm air from the system, reducing the temperature of the system. 1 CFM of airflow through the heat sinks roughly corresponds to 100 LFM of airflow, and 25 CFM to 2,300 LFM. To produce such flow rates in an open air test environment such as a wind tunnel or unducted flow, a minimum of 2 to 3 times the airflow around the heat sink is required. Observe that the temperature drops off rapidly with increasing flow rate, reaching an asymptotic value. This suggests that higher airflow for a given heat sink will not necessarily translate to significant gains in the reduction of case temperature. These calculations, along with the airflow performance curves, demonstrate that it is not fundamentally beneficial to increase airflow. Increasing airflow results in an increase in the pressure drop requirement, translating to large, noisy blowers which are not conducive to an office environment. Based on an operating point of 14 CFM for a single blower (i.e. 7 CFM through each parallel flow path), the calculated temperatures for the upstream and downstream Pentium Pro processors in this cooling arrangement are 53°C and 61°C at a 35°C ambient. Clearly both the upstream and downstream processors have significant thermal headroom. The extent of the upstream processor's heat

on the air which reaches the downstream processor is kept at a minimum due to the mixing and high airflow rates achieved in the blower cooling arrangement. Also note that an arrangement which uses a 120 mm fan with a duct cannot maintain the case temperature of the downstream processor below the recommended temperature of 85°C, and hence is not recommended as a cooling option for a quad processor arrangement.

Figure 15 shows the power versus case temperature tradeoff for this ducted blower approach. The conventional approach of using a system fan with low aspect ratio heat sinks that are typical in the industry is also included for comparison. The classical approach of widely spaced fins results in a heat sink which is ineffective in cooling high performance systems with processors in close proximity. This blower solution, however, can provide scalability to 40 Watts without any changes in the design.

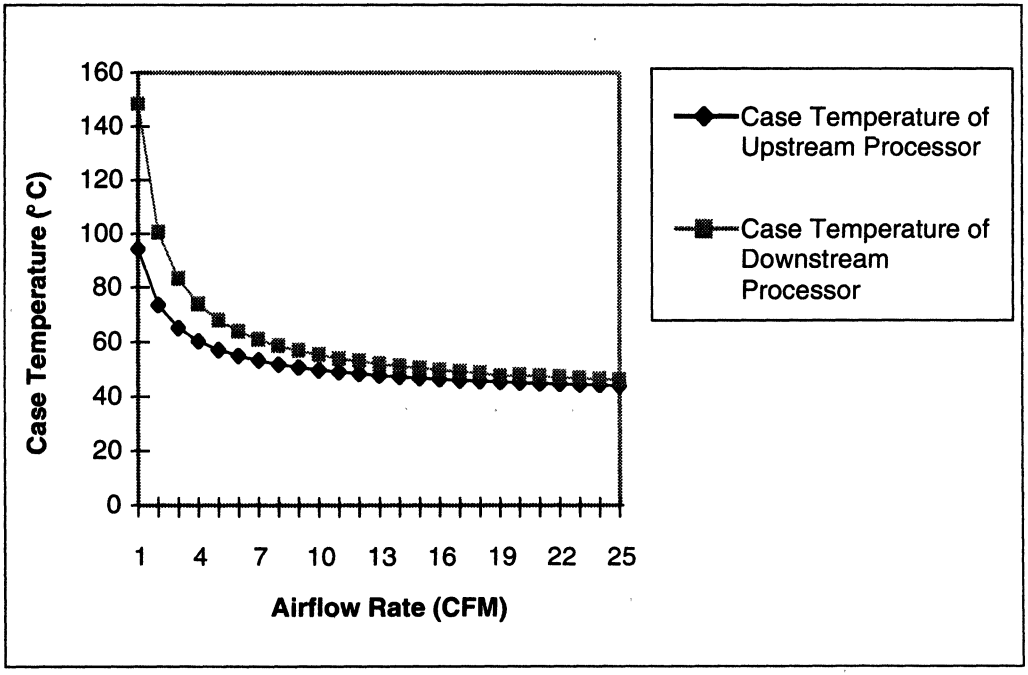


Figure 14. Case Temperature vs. Airflow at 30W

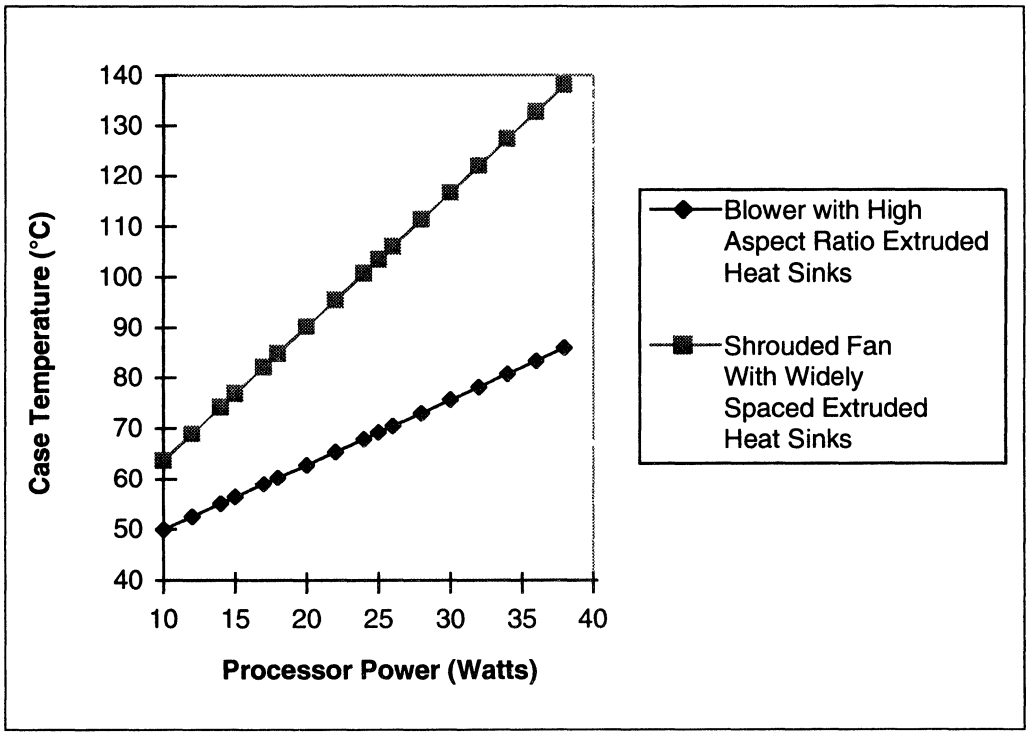


Figure 15. Case Temperature vs. Processor Power for Design Example #4

7.0. CONCLUSION

As the complexity of today's microprocessors continues to increase, so do the power dissipation requirements. Care must be taken to ensure that the additional power is properly dissipated. Heat can be dissipated using passive heat sinks, fans and/or active cooling devices. Further cooling can be achieved through the use of ducting solutions.

The simplest and most cost effective method is to use an extruded heat sink and a system fan. The size of the heat sink and the output of the fan can be varied to balance size and space constraints with acoustic noise. As shown, a 1.9" high heat sink can be used with a dedicated 200 LFM system fan to cool a Pentium Pro processor dissipating 28 Watts. In another example, it was shown that a fan/heat sink assembly having a Θ_{CA} of 1.25°C/W can also be used to cool a single Pentium Pro processor dissipating 28 Watts, as long as the local ambient temperature does not exceed 50°C. This will also be the requirement for the OverDrive processor for upgrading Pentium Pro processor-based systems.

Both of these solutions provide adequate cooling to maintain the Pentium Pro processor's case temperature at or below 85°C. The addition of ducting allows cooling of up to four Pentium Pro processors. By maintaining the Pentium Pro processor's case temperature and OverDrive processor's fan/heat sink air inlet temperature at the values specified in the *Pentium® Pro Processor Developer's Manual, Volume 1*, a system can guarantee proper functionality and reliability of these processors.

intel®

2

Pentium® Processor and Related Products

THE PENTIUM® FAMILY-A TECHNICAL OVERVIEW

Intel's Pentium® processor family combines the performance traditionally associated with minicomputers and workstations with the flexibility and compatibility that characterize the personal computer platform. Designed to meet the needs of today's and tomorrow's sophisticated software applications, the Pentium processor extends the range of Intel's microprocessor architecture to new heights, blurring previous distinctions between hardware platforms and creating an entirely new realm of possibilities for notebook computers, desktop PCs, and servers.

This article begins by presenting an overview of the Pentium processor. It then details the key technological features that enable the Intel solution to meet the market's evolving requirements for high performance, continued software compatibility, and advanced functionality.

THE WORLD'S BEST PERFORMANCE FOR ALL PC SOFTWARE

The Pentium processor family includes the highest performing members of Intel's family of microprocessors.

Pentium® Processor 75/90/100/120/133 Core Frequency	External Bus Interface	iCOMP® Index
133 MHz	66 MHz	1110
120 MHz	60 MHz	1000
100 MHz	66/50 MHz	815
90 MHz	60 MHz	735
75 MHz	50 MHz	610

While incorporating new features and improvements made possible by advances in semiconductor technology, the Pentium processor is fully software compatible with previous members of the Intel microprocessor family-thereby preserving the value of users' software investments. The Pentium processor meets the demands of computing in a number of areas: advanced operating systems, such as DOS*, Windows*, OS/2*, and UNIX*; computing-intensive graphics applications, such as 3-D modeling, computer-aided design/engineering (CAD/CAE), large scale financial analysis, high-throughput client/server, handwriting, and voice recognition; network applications; virtual reality; electronic mail that combines many of the above areas; and new applications yet to be developed.

The Pentium processor family was designed using an advanced process technology and has features that are less than a micron (one-millionth of a meter) in size. The Pentium processor (510\60, 567\66) was developed utilizing 5V, 0.8 micron technology, while the Pentium processor (610\75, 735\90, 815\100. 1000\120, 1110\133) was designed using 3.3V, 0.6 micron and 3.3V, 0.35 micron technology.

The increasingly improved Pentium processor family brings the users CPUs with higher frequencies, while the system bus frequencies range from 50 MHz to 66 MHz, allowing cost effective system designs.

THE PENTIUM® PROCESSOR: TECHNICAL INNOVATIONS

A number of innovative product features contribute to the Pentium processor's unique combination of high performance, compatibility, data integrity and upgradability. These include:

- Superscalar Architecture
- Separate 8K Code and Data Caches
- Writeback MESI Protocol in the Data Cache
- Dynamic Branch Prediction
- Pipelined Floating-point Unit
- Improved Instruction Execution Time
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Functional Redundancy Checking
- Execution Tracing
- Performance Monitoring

November 1995
Order Number: 242423-002

- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions
- Upgradable With a Future Pentium OverDrive® Processor
- Multiprocessor Support

In addition to the features listed above, the Pentium processor 75/90/100/120/133 offers the following enhancements over the Pentium processor 60/66

- Dual Processing Support
- SL Power Management Features
- Fractional Bus Operation
- On-chip Local APIC Device

Superscalar Architecture

The Pentium processor's superscalar architecture enables the processor to achieve superior performance by executing more than one instruction per clock cycle. The term "superscalar" refers to a microprocessor architecture that contains more than one execution unit. These execution units, or piplines are where the CPU processes the data and instructions that are fed to it by the rest of the system.

The Pentium processor's superscalar implementation represents a natural progression from previous generations of processors in the 32-bit Intel architecture. The Intel486™ processor for instance, is able to execute many instructions in one clock cycle, while previous generations of Intel microprocessors require multiple clock cycles to execute a single instruction.

The ability to execute multiple instructions per clock cycle is due to the fact that the Pentium processor has two pipelines that can execute two instructions simultaneously. The Pentium processor's dual pipelines execute integer instructions in five stages: *prefetch, decode1, decode2, execute* and *writeback*. This permits several instructions to be in various stages of execution, thus increasing processing performance.

The Pentium processor also uses hardwired instructions to replace many of the microcoded instructions used in previous microprocessor generations. Hardwired instructions are simple and commonly used, and can be executed by the processor's hardware without requiring microcode.

This improves performance without affecting compatibility. In the case of more complex instructions, the Pentium processor's enhanced microcode further boosts performance by employing both dual integer pipelines to execute instructions.

Separate 8K Code and Data Caches

Pentium processors include separated code and data caches integrated on-chip to meet performance goals. On-chip caches increase performance by acting as temporary storage places for commonly-used instructions and data, replacing the need to go off-chip to the system's main memory to fetch information. The separate caches reduce bus conflicts and are available more often when they are needed.

The Pentium processor's code and data caches each contain 8 Kbytes of information and both are organized as two-way set associative caches-meaning that they save time by searching only pre-specified 32-byte segments rather than the entire cache. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses.

The Pentium processor's data cache is configurable to be "writeback" or "write through" on a line-by-line basis and follows the MESI (Modified, Exclusive, Shared, Invalid) protocol. The "writeback" method transfers data to the cache without going out to main memory. Data is written to main memory only when it is removed from the cache. In contrast, the "write through" method transfer data to the external memory each time the processor writes data to the cache. The "writeback" technique increases performance by reducing bus utilization and preventing unnecessary bottlenecks in the system.

To ensure that data in the cache and in main memory are consistent, the data cache implements the MESI protocol during reads and writes. This is especially important in a multiprocessor environment.

Dynamic Branch Prediction

Branch prediction is an advanced computing technique that boosts performance by keeping the execution pipelines full. It is accomplished by predetermining the most likely set of instructions to be executed.

To understand the concept better, consider a typical application program. After each pass through a software loop, the program performs a conditional test to determine whether to return to the beginning of the loop or to exit and continue on to the next execution step. These two paths are called branches. Dynamic branch prediction forecasts which branch the software will require, based on the assumption that the previous taken branch will be used again. Pentium processors make prediction by using a Branch Target Buffer (BTB). Pentium processors also implement two prefetch buffers, one to prefetch code in a linear fashion and the other to prefetch code according to the addresses in the BTB. As a result, the needed code is always prefetched before it is required for execution. In addition, the Pentium processors support more sophisticated algorithms by using two level branch prediction.

Pipe-lined Floating-Point Unit

The 32-bit compute-intensive software applications require a high degree of floating-point processing power to handle mathematical calculations. As the floating-point requirements of personal computer software have steadily increased, advances in microprocessor technology have been introduced to satisfy these needs. The Intel486 DX processor, for example, was the first Intel microprocessor to integrate math coprocessing functions on-chip; previous-generation Intel processors used off-chip math coprocessors when floating-point calculations were required.

The Pentium processor family takes math computational ability to the next performance level by using an enhanced on-chip floating-point unit that incorporates sophisticated eight-stage pipeline and hardwired functions. A three-stage floating-point instruction pipeline is appended to the integer pipelines. Most floating-point instructions begin execution in one of the integer pipelines, then move on to the floating-point pipeline. In addition, common floating-point functions, such as, add, multiply and divide, are hardwired for faster execution.

Enhanced 64-Bit Data Bus

The data bus is the highway that carries information between the processor and the memory subsystem. Because of its external 64-bit data bus, the Pentium processor can transfer data to and from memory at rates up to 528 Mbytes/second, a more than five-fold increase over the peak transfer rate of the 66 MHz Intel DX2™ microprocessor (105 Mbytes/second). This wider data bus facilitates high-speed processing by maintaining the flow of instructions and data to the processor's superscalar execution unit.

In addition to having a wider data bus, the Pentium processor implements bus cycle pipelining to increase bus bandwidth. Bus cycle pipelining allows a second cycle to start before the first one is completed. This gives the memory subsystem more time to decode the address, which allows slower and less-expensive memory components to be used, resulting in a lower overall system cost. Burst reads and writes, parity on address and data, and a simple cycle identification all contribute to providing greater bandwidth and improved system reliability.

The Pentium processor also has two write buffers, one corresponding to each pipeline, to enhance the performance of consecutive writes to memory. Write buffers improve performance by allowing the processor to proceed with the next pair of instructions, even though one of the current instructions needs to write to memory while the bus is busy.

Data Integrity and Error Detection Features

Protecting important data and ensuring its integrity has become increasingly important as mission-critical applications continue to proliferate. To ensure the Pentium processors' reliability, Intel ran millions of simulations and tests. In addition, designers have added significant data integrity and error detection capability. Data parity checking is supported on byte-by-byte basis. Address parity checking, and internal parity checking features have been added along with a new exception, the machine check exception.

Internal error detection places parity bits on the internal code and data caches, translation look aside buffers, microcode, and branch target buffer. This feature helps to detect errors in a manner that remains transparent to both the user and the system.

Furthermore, the Pentium processors have implemented functional redundancy checking to provide maximum error detection of the processor and the interface to the processor. When functional redundancy checking is used, two Pentium processors act as "master" and "checker" respectively. The "checker" is used to

execute in lock step with the "master" processor. The two chips run in tandem, and the "checker" samples the "master's" outputs and compares those values with the values it computes internally. The "checker" asserts an error signal if a discrepancy is discovered, and the system is notified.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this situation, the Pentium processors have increased test and debug capability. The Pentium processors implement IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processors have specified four breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match.

Performance Monitoring

Performance monitoring is a feature of the Pentium processor family that enables system designers and software application developers to optimize their hardware and software products by identifying potential code bottlenecks. Designers can observe and count clocks for internal processor events that affect the performance of data reads and writes, cache hits and misses, interrupts, and bus utilization. this allows them to measure the effect that their code has on both the Pentium processor architecture and their product, and to fine-tune their application or system for optimal performance, due to the greater synergy between the Pentium processor, its host system, and application software.

SL Enhanced Power Management Features

The Pentium processor (610\75, 735\90, 815\100, 1000\120, 1110\133) incorporate SL technology features for superior power-management capabilities. These features operate at two levels: the microprocessor and the system. Power management at the processor level involves putting the processor into low power state during non-processor intensive tasks, such as word processing, or into "sleep mode", which is a very low-power state, when the computer is not in use. At the system level, Intel's SL technology uses system management mode (SMM) to control the way power is used by the computer and its peripherals. This mode provides intelligent system management that allow the microprocessor to slow down, suspend, or completely shut down various system components so as to maximize energy savings. All members of the Pentium processor family include SMM.

Virtual Memory Extensions

The Pentium processors' Memory Management Unit offers the optional extensions from the traditional memory page size of 4 Kbytes, to the architecture which allow 2-Mbyte and 4-Mbyte page sizes. This feature, which is transparent to the application software, was provided to reduce the frequency of page swapping in complex graphics applications, frame buffers, and operating system kernels, where the increased page size allows users to map large, previously unwieldy objects. The larger page enables an increased page hit rate, resulting in higher performance.

Upgradable With a Future Pentium® OverDrive® Processor

The Pentium processor has been designed for upgradability using Intel's upgrade technology. This innovation protects user investments by adding performance that helps to maintain the productivity levels of Intel processor-based systems over their entire life spans. Many Pentium processor-based systems have been designed to support this upgradability. The Future Pentium OverDrive processor will speed up typical applications by 40% to 70%.

Multiprocessor Support

The Pentium processor is ideal for the increasing implementation of multiprocessing systems. Multiprocessing applications that combine two or more Pentium processors are well served by the processors' advanced architecture, separate on-chip code and data caches, chipsets for controlling external caches, and sophisticated data integrity features.

As previously discussed, the Pentium processor family uses the MESI protocol to maintain cache consistency among several processors. The Pentium processor also ensures that instructions are seen by the system in the order that they were programmed. This strong ordering helps software designed to run on a single-processor system to work correctly in a multiprocessing environment.

Symmetric dual processing in a system is supported with two Pentium processors 75/90/100/120/133. The two processors appear to the system as a single Pentium processor. Operating systems with dual processing support properly schedule computing tasks between the two processors. This scheduling of tasks is transparent to software applications and the end-users. Logic built into the processors support a "glueless" interface for easy system design. Through a private bus, the two Pentium processors 75/90/100/120/133 arbitrate for the external bus and maintain cache coherency.

On-chip Local APIC Device

The Pentium processor 75/90/100/120/133 incorporates a local Advanced Programmable Interrupt Controller designed to handle interrupts in a multiprocessing environment. This implementation is capable of supporting a multiprocessing interrupt scheme with an external I/O APIC. The I/O APIC is a device which captures all system interrupts and directs them to the appropriate processors via various programmable distribution schemes. For instance, in the multiprocessing environment, the APIC can distribute static and dynamic symmetric interrupts across all processors. It is also capable of routing the dynamic interrupts to the lowest-priority processor. These are just two examples among many features of APIC architecture.

In a dual processor configuration, the local APIC may be used with an additional device similar to the I/O APIC. An external device provides the APIC system clock. Interrupts which are local to each CPU go through the APIC on each chip.

INCREASED PERFORMANCE: BY THE NUMBERS

The iCOMP (Intel COmparative Microprocessor Performance) index provides a simple relative measure of microprocessor performance. It is not a benchmark, but a collection of benchmarks used to calculate an index of relative processor performance intended to help end-users to decide which Intel microprocessor best meets their desktop computing needs. The iCOMP index rating is based on the technical categories that encompass four separate aspects of both 16- and 32-bit CPU performance: integer, floating-point, graphics and video performance. Each category is weighted based on the estimated percentage of time it enters into the processing picture. The higher the iCOMP index rating, the higher the relative performance of the microprocessor.

Figure 1 illustrates the iCOMP index ratings for ten Intel microprocessors. The Intel Pentium processor 133 MHz with an iCOMP index rating 1110 yields 2.5 times the performance of the 100 MHz IntelDX4® processor, which has an iCOMP index rating of 435.

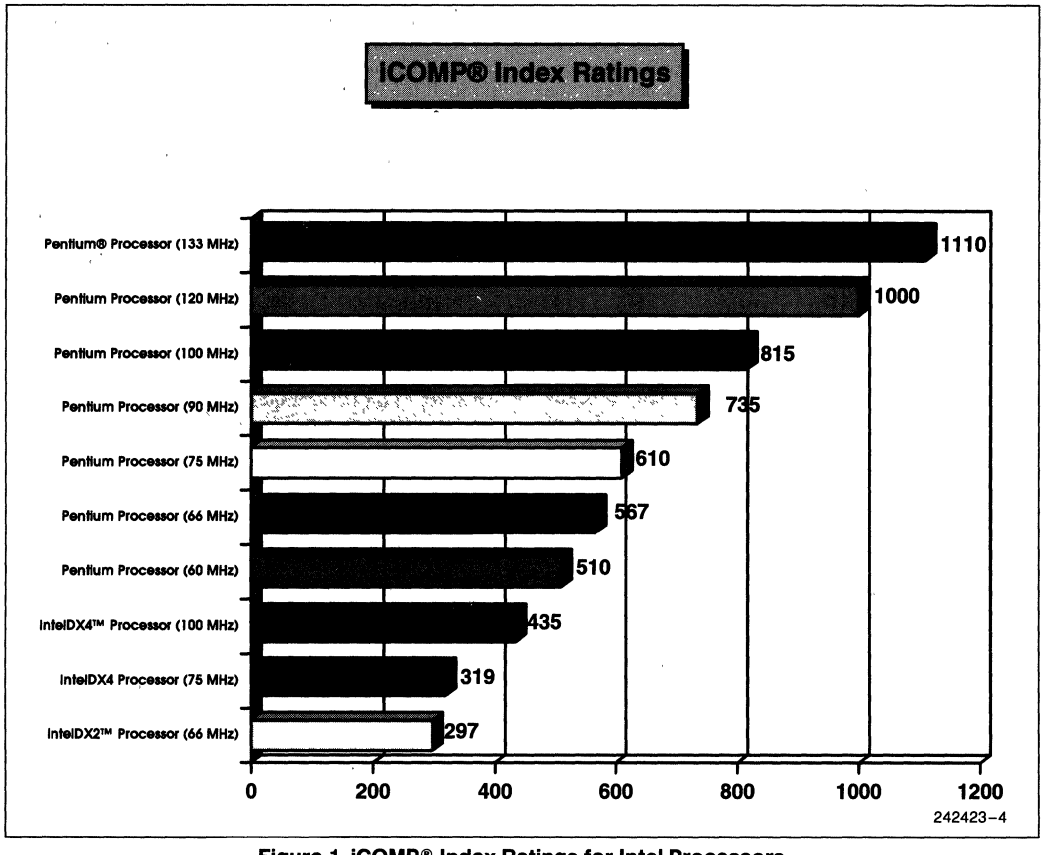


Figure 1. iCOMP® Index Ratings for Intel Processors

Modern industry standard benchmarks were chosen to accurately demonstrate the superior performance of the Intel Pentium Pro processor family. Processor-intensive benchmarks such as SPECint95*, SPECint92* and several 32-bit benchmarks highlight workstation-level performance. SPECint95 and SPECint92 are two very effective benchmarks for evaluating CPU performance because of its large size and application representative instruction mix.

SPECint92 is a processor-intensive UNIX benchmark that evaluates desktop performance using a representative mix of application instructions. The SPECfp92* UNIX benchmark is a useful measure of floating-point performance. Because today's commercial applications are comprised almost exclusively of integer-intensive programs, SPECint92 represents an appropriate instruction mix for commercial applications and is a much more effective benchmark to predict 32-bit business performance than SPECfp92.

SPEC95* was designed to provide measures of performance for comparing compute-intensive workloads on different computer systems. SPEC95 consists of two suites of benchmarks: CINT95* for measuring and comparing compute-intensive integer performance, and CFP95* for measuring and comparing compute-intensive floating-point performance. The two suites provide

component-level benchmarks that measure the performance of the computer's processor, memory architecture and compiler. SPEC* benchmarks are selected from existing application and benchmark source code running across multiple platforms. Each benchmark is tested on different platforms to obtain fair performance results across competing hardware and software systems.

SPEC95 is the third major version of the SPEC benchmark suites, which in 1989 became the first widely accepted standard for comparing compute-intensive performance across various architectures. The new release replaces SPEC92*, which will be gradually phased out between now and June 1996, when SPEC will stop publishing SPEC92 results and stop selling the benchmark suite. Performance results from SPEC95 cannot be compared to those from SPEC92, since new benchmarks have been added and existing ones changed.

The CINT95 suite, written in C language, contains eight CPU-intensive integer benchmarks. It is used to measure and calculate the following metrics:

- SPECint95 – The geometric mean of eight normalized ratios (one for each integer benchmark) when compiled with aggressive optimization for each benchmark.
- SPECint_base95* – The geometric mean of eight normalized ratios when compiled with the conservative optimization for each benchmark.

The CFP95 suite, written in FORTRAN language, contains 10 CPU-intensive floating point benchmarks. It is used to measure and calculate the following metrics:

- SPECfp95* – The geometric mean of 10 normalized ratios (one for each floating point benchmark) when compiled with aggressive optimization for each benchmark.
- SPECfp_base95* – The geometric mean of 10 normalized ratios when compiled with conservative optimization for each benchmark.

Because today's commercial applications are comprised almost exclusively of integer-intensive programs, SPECint95 represents an appropriate instruction mix for commercial applications and is a much more effective benchmark to predict 32-bit business performance than SPECfp95.

Figures 2 and 3 show the SPECint95 and SPECfp95 performance of the Pentium processor 100 MHz, 120 MHz and 133 MHz under the 1 MB second-level cache ("L2 cache") configurations. Figures 4 and 5 show the SPECint92 and SPECfp92 performance of Pentium processor 100 MHz, 120 MHz and 133 MHz under 1 MB second-level cache ("L2 cache") configurations. The SPECint92 and SPECfp92 numbers use the latest compiler technology (see the lastest Intel Pentium Pro Processor Performance Brief).

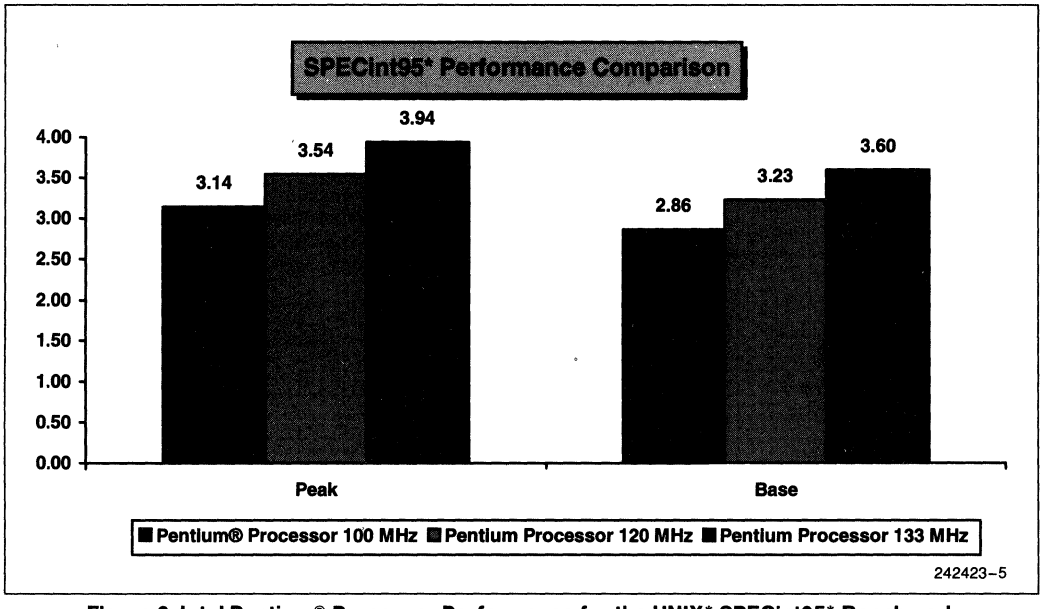


Figure 2. Intel Pentium® Processor Performance for the UNIX* SPECint95* Benchmark

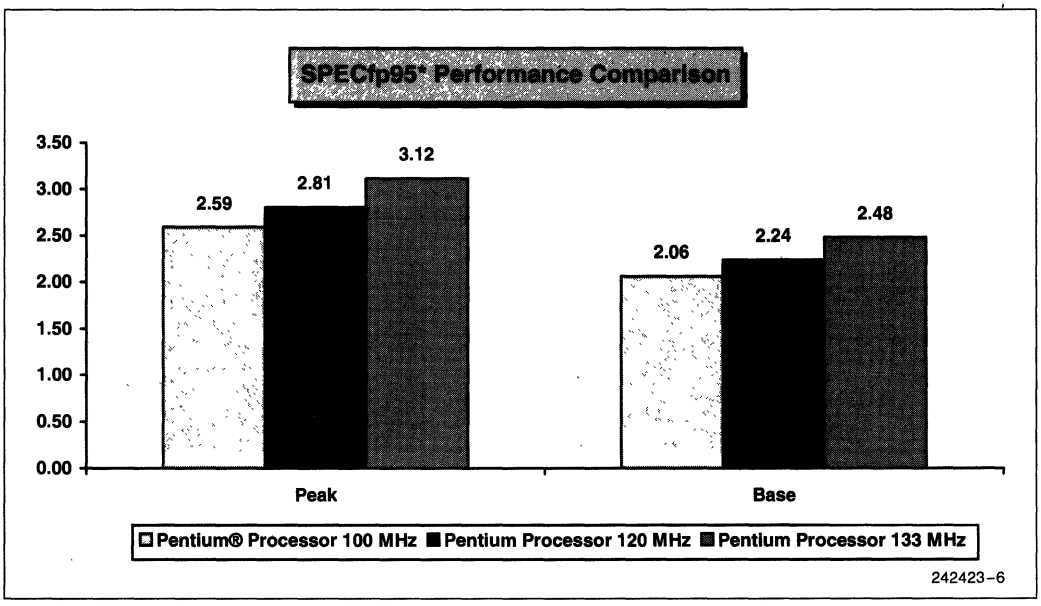


Figure 3. Intel Pentium® Processor Performance for the UNIX* SPECfp95* Benchmark

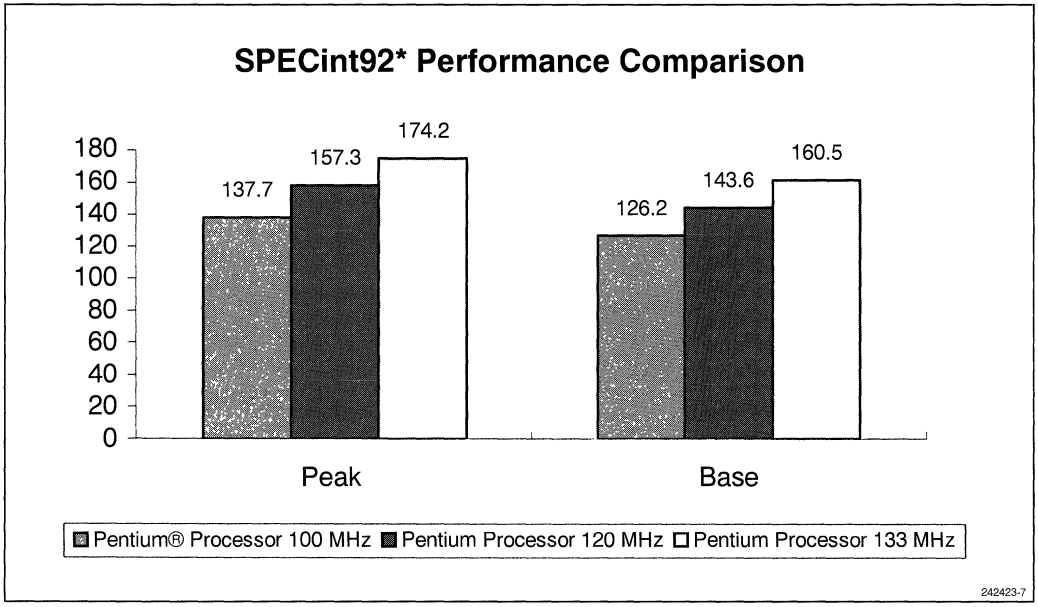


Figure 4. Intel Pentium Processor Performance for UNIX* SPECint92* Benchmark

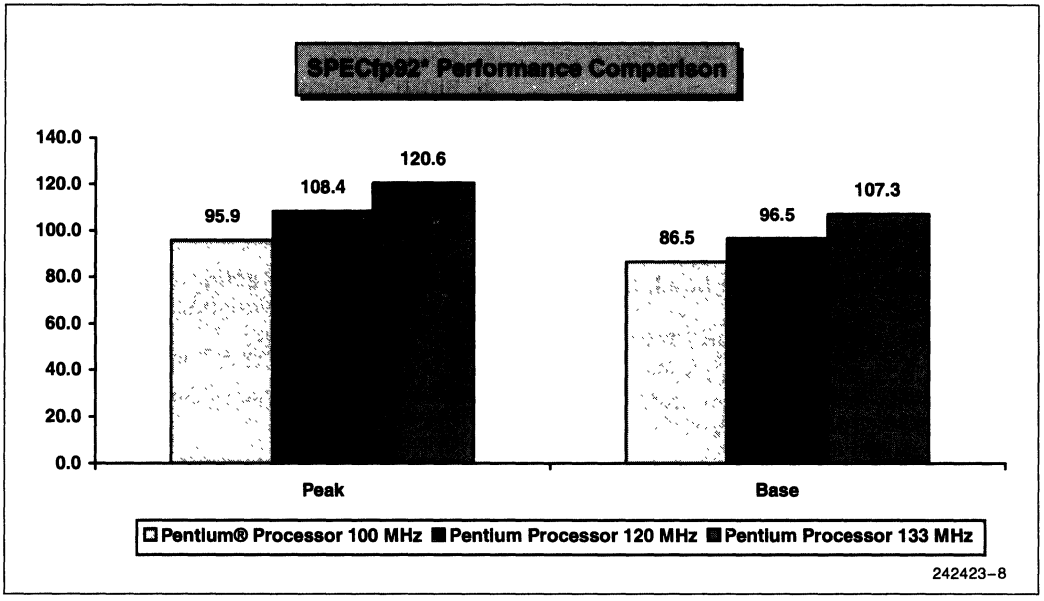


Figure 5. Intel Pentium® Processor Performance for the UNIX* SPECfp92* Benchmark

HIGH PERFORMANCE WITH FULL COMPATIBILITY

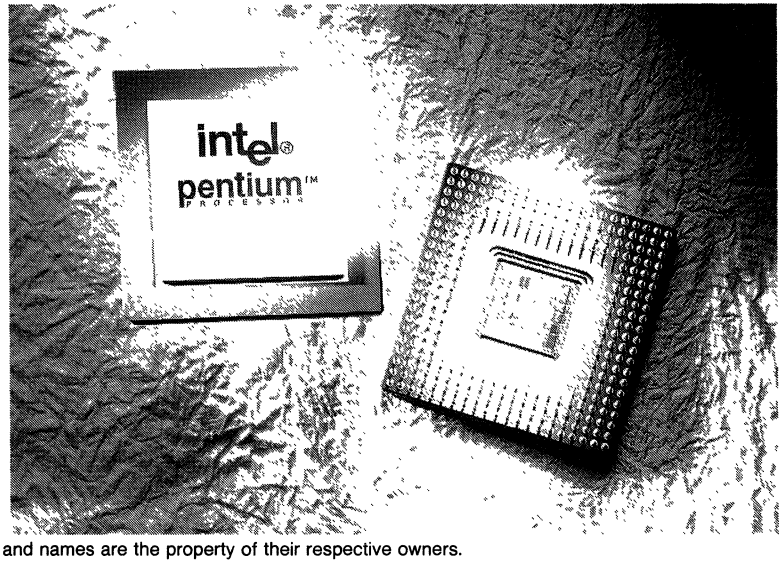
The Pentium processor family provides extremely high performance because it incorporates the latest state-of-the-art design principles. With its superscalar architecture, separate code and data caches, dynamic branch prediction, and an enhanced floating-point unit, the Pentium processor can meet the performance needs of today's and tomorrow's software applications. Meanwhile it maintains complete compatibility with the installed base of software currently running on all the Intel family members.

The Pentium processors' combination of performance and compatibility uniquely positions it to meet the needs of the expanding development of notebook, desktop, and server application. Not only will users experience dramatic performance improvements while running their current software, but they can also anticipate that new applications will take even further advantage of the Pentium processors' high performance features.

PENTIUM® PROCESSOR AT iCOMP INDEX 510\60 MHz
PENTIUM® PROCESSOR AT iCOMP INDEX 567\66 MHz

- **Binary Compatible with Large Software Base**
  - **— DOS*, OS/2*, UNIX*, and WINDOWS***
- **32-Bit Microprocessor**
  - **— 32-Bit Addressing**
  - **— 64-Bit Data Bus**
- **Superscalar Architecture**
  - **— Two Pipelined Integer Units**
  - **— Capable of under One Clock per Instruction**
  - **— Pipelined Floating Point Unit**
- **Separate Code and Data Caches**
  - **— 8K Code, 8K Write Back Data**
  - **— 2-Way 32-Byte Line Size**
  - **— Software Transparent**
  - **— MESI Cache Consistency Protocol**
- **Advanced Design Features**
  - **— Branch Prediction**
  - **— Virtual Mode Extensions**
- **273-Pin Grid Array Package**
- **BiCMOS Silicon Technology**
- **Increased Page Size**
  - **— 4M for Increased TLB Hit Rate**
- **Multi-Processor Support**
  - **— Multiprocessor Instructions**
  - **— Support for Second Level Cache**
- **Internal Error Detection**
  - **— Functional Redundancy Checking**
  - **— Built in Self Test**
  - **— Parity Testing and Checking**
- **IEEE 1149.1 Boundary Scan Compatibility**
- **Performance Monitoring**
  - **— Counts Occurrence of Internal Events**
  - **— Traces Execution through Pipelines**

The Pentium® processor (510\60, 567\66) provides the next generation of power for high-end workstations and servers. The Pentium processor (510\60, 567\66) is compatible with the entire installed base of applications for DOS*, Windows*, OS/2*, and UNIX*. The Pentium processor's superscalar architecture can execute two instructions per clock cycle. Branch Prediction and separate caches also increase performance. The pipelined floating point unit of the Pentium processor (510\60, 567\66) delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor (510\60, 567\66) has 3.1 million transistors and is built on Intel's 0.8 Micron BiCMOS silicon technology. The Pentium processor may contain design defects or errors known as errata. Current characterized errata are available upon request.



241595-1

*Other brands and names are the property of their respective owners.

November 1995
Order Number: 241595-003

Pentium® Processor (510\60, 567\66)

CONTENTS

CONTENTS

1.0 MICROPROCESSOR ARCHITECTURE OVERVIEW

The Pentium® processor (510\60, 567\66) is the next generation member of the Intel386™ and Intel486™ microprocessor family. It is 100% binary compatible with the 8086/88, 80286, Intel386 DX CPU, Intel386 SX CPU, Intel486 DX CPU, Intel486 SX and the Intel486 DX2 CPUs.

The Pentium processor (510\60, 567\66) contains all of the features of the Intel486 CPU, and provides significant enhancements and additions including the following:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Improved Instruction Execution Time
- Separate 8K Code and Data Caches
- Writeback MESI Protocol in the Data Cache
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Functional Redundancy Checking
- Execution Tracing
- Performance Monitoring
- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions

The application instruction set of the Pentium processor (510\60, 567\66) includes the complete Intel486 CPU instruction set with extensions to accommodate some of the additional functionality of the Pentium processor (510\60, 567\66). All application software written for the Intel386 and Intel486 microprocessors will run on the Pentium processor (510\60, 567\66) without modification. The on-chip memory management unit (MMU) is completely compatible with the Intel386 and Intel486 CPUs.

The Pentium processor (510\60, 567\66) implements several enhancements to increase performance. The two instruction pipelines and floating-point unit on the Pentium processor (510\60, 567/66) are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, 2 floating point instructions) in one clock.

Branch prediction is implemented in the Pentium processor (510\60, 567\66). To support this, the Pentium processor (510\60, 567\66) implements two prefetch buffers, one to prefetch code in a linear fashion, and one that prefetches code according to the BTB so the needed code is almost always prefetched before it is needed for execution.

The floating-point unit has been completely redesigned over the Intel486 CPU. Faster algorithms provide up to 10X speed-up for common operations including add, multiply, and load.

The Pentium processor (510\60, 567\66) includes separate code and data caches integrated on chip to meet its performance goals. Each cache is 8 Kbytes in size, with a 32-byte line size and is 2-way set associative. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses. The data cache is configurable to be writeback or writethrough on a line by line basis and follows the MESI protocol. The data cache tags are triple ported to support two data transfers and an inquire cycle in the same clock. The code cache is an inherently write protected cache. The code cache tags are also triple ported to support snooping and split line accesses. Individual pages can be configured as cacheable or non-cacheable by software or hardware. The caches can be enabled or disabled by software or hardware.

The Pentium processor (510\60, 567\66) has increased the data bus to 64-bits to improve the data transfer rate. Burst read and burst writeback cycles are supported by the Pentium processor (510\60, 567\66). In addition, bus cycle pipelining has been added to allow two bus cycles to be in progress simultaneously. The Pentium processor (510\60, 567\66) Memory Management Unit contains optional extensions to the architecture which allow 4 Mbyte page sizes.

The Pentium processor (510\60, 567\66) has added significant data integrity and error detection capability. Data parity checking is still supported on a byte by byte basis. Address parity checking, and internal parity checking features have been added along with a new exception, the machine check exception. In addition, the Pentium processor (510\60, 567\66) has implemented functional redundancy checking to provide maximum error detection of the processor and the interface to the processor. When functional redundancy checking is used, a second processor, the "checker" is used to execute in lock step with the "master" processor. The checker samples the master's outputs and compares those values with the values it computes internally, and asserts an error signal if a mismatch occurs.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this, the Pentium processor (510\60, 567\66) has increased test and debug capability. Like many of the Intel486 CPUs, the Pentium processor (510\60, 567\66) implements IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processor (510\60, 567\66) has specified 4 breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match. Execution tracing provides external indications when an instruction has completed execution in either of the two internal pipelines, or when a branch has been taken.

System management mode has been implemented along with some extensions to the SMM architecture. Enhancements to the Virtual 8086 mode have been made to increase performance by reducing the number of times it is necessary to trap to a virtual 8086 monitor.

Figure 1-1 shows a block diagram of the Pentium processor (510\60, 567\66).

The block diagram shows the two instruction pipelines, the "u" pipe and the "v" pipe. The u-pipe can execute all integer and floating point instructions. The v-pipe can execute simple integer instructions and the FXCH floating point instructions.

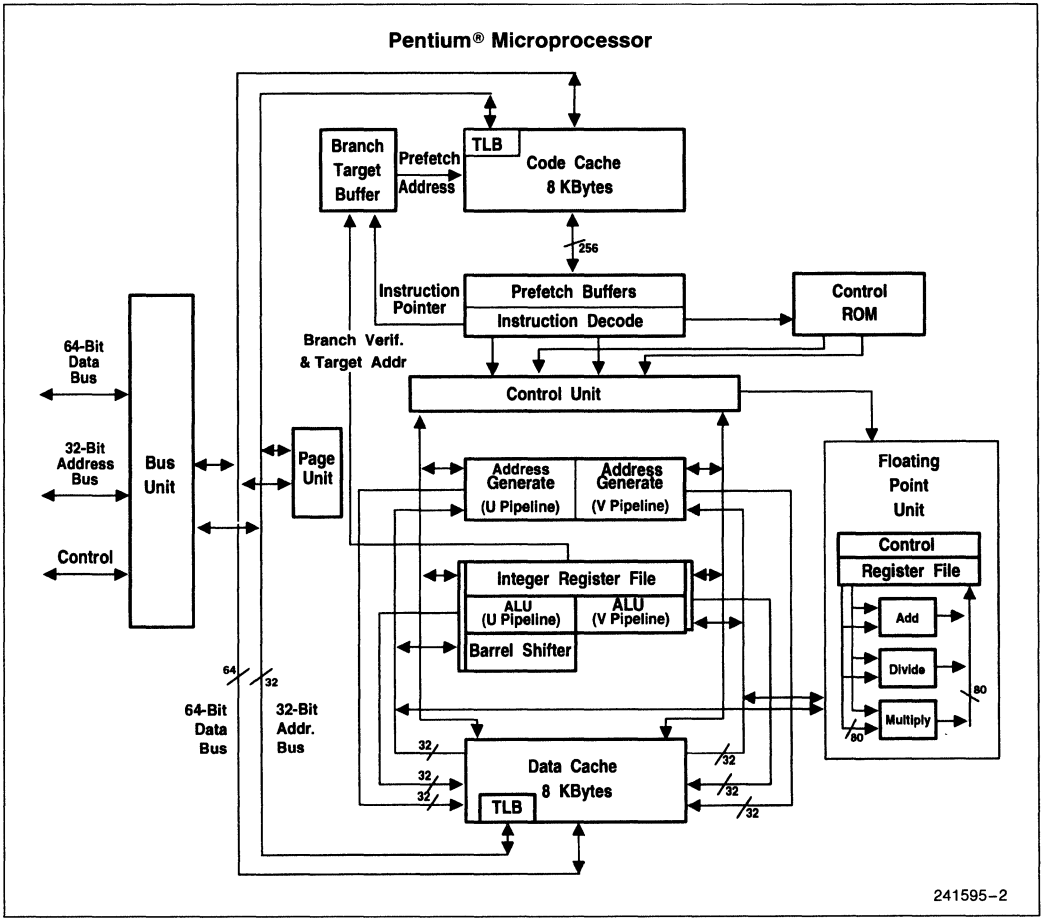


Figure 1-1. Pentium® Processor (510\60, 567\66) Block Diagram

The separate caches are shown, the code cache and data cache. The data cache has two ports, one for each of the two pipes (the tags are triple ported to allow simultaneous inquire cycles). The data cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium processor (510\60, 567\66). Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so the Pentium processor (510\60, 567\66) can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that must be performed to implement the Pentium processor (510\60, 567\66) architecture. The control ROM unit has direct control over both pipelines.

The Pentium processor (510\60, 567\66) contains a pipelined floating point unit that provides a significant floating point performance advantage over previous generations of the Pentium processor (510\60, 567\66).

The architectural features introduced in this chapter are more fully described in the *Pentium® Processor Family Developer's Manual*, Volume 3.

2.0 PINOUT

2.1 Pinout and Pin Descriptions

2.1.1 Pentium® PROCESSOR (510\60, 567\66) PINOUT

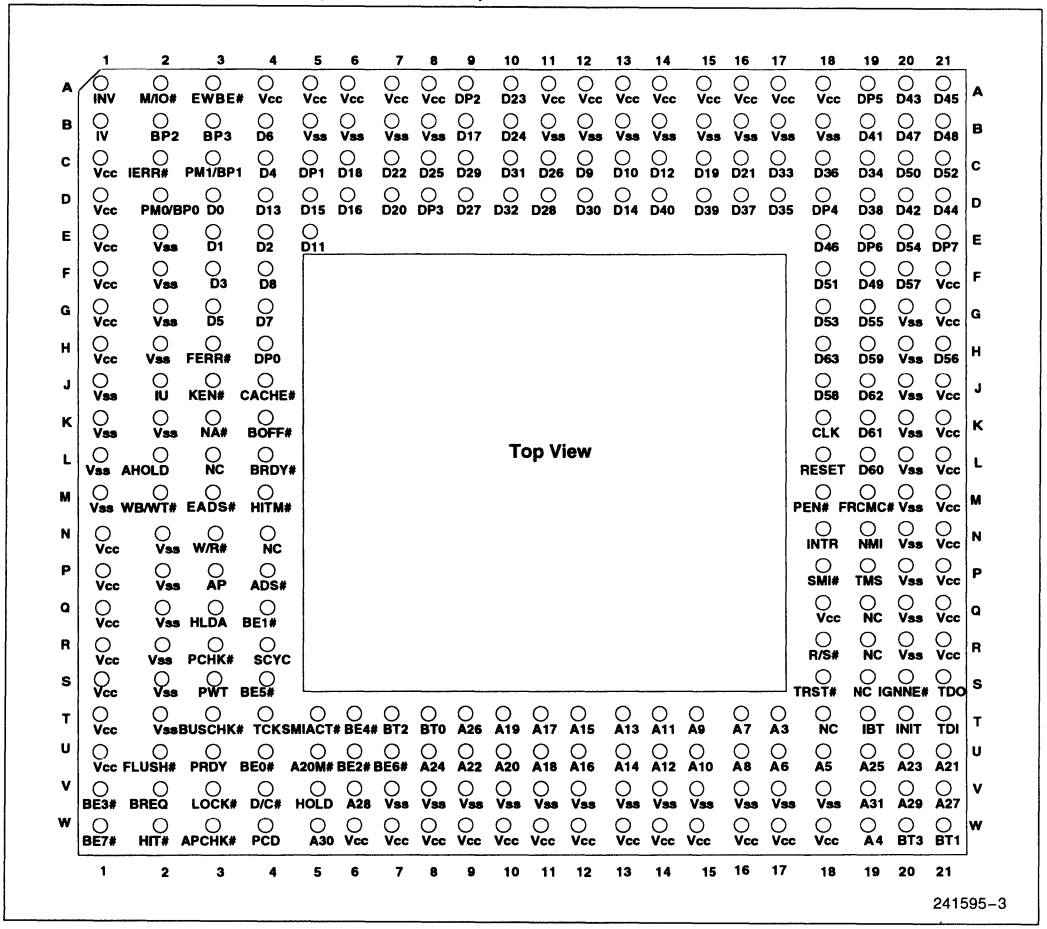


Figure 2-1. Pentium® Processor (510\60, 567\66) Pinout (Top View)

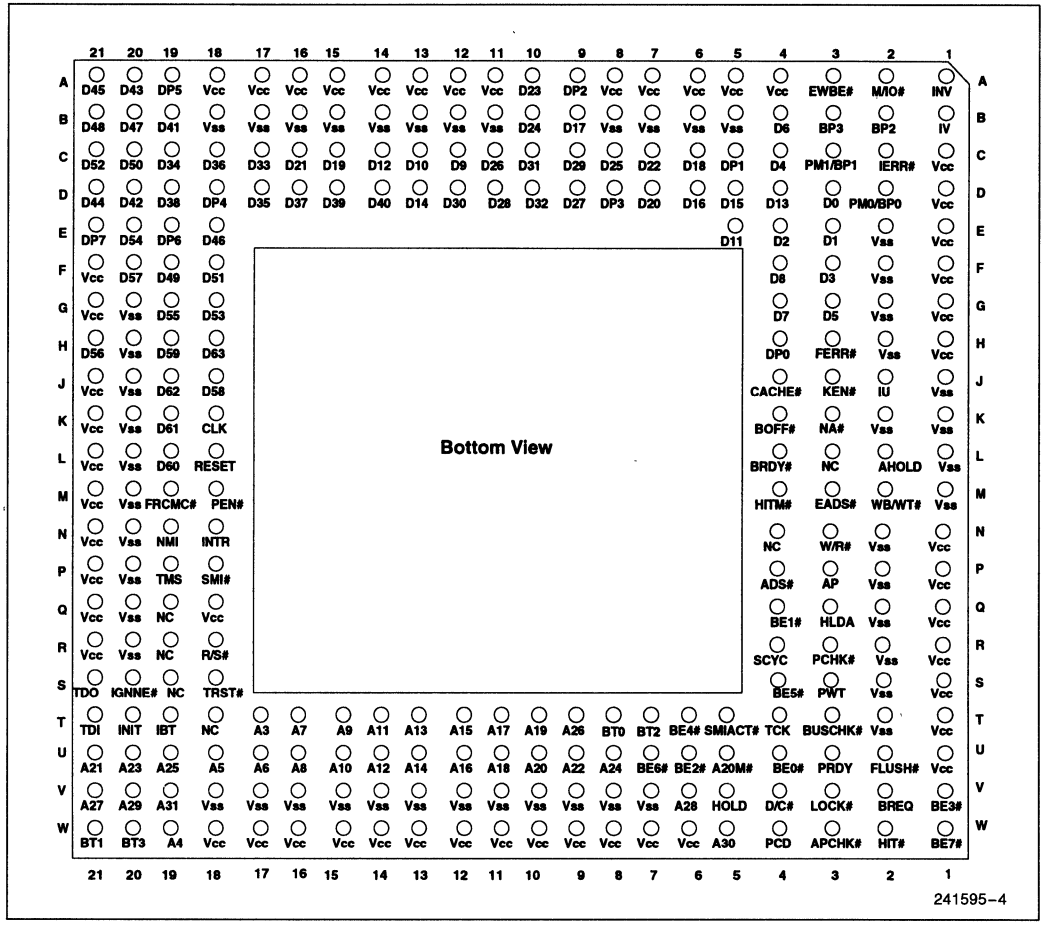


Figure 2-2. Pentium® Processor (510\60, 567\66) Pinout (Bottom View)

Table 2-1. Pentium® Processor (510\60, 567\66) Pin Cross Reference Table by Pin Name

Signal	Location	Signal	Location	Signal	Location	Signal	Location
A3	T17	BE2#	U06	D18	C06	D54	E20
A4	W19	BE3#	V01	D19	C15	D55	G19
A5	U18	BE4#	T06	D20	D07	D56	H21
A6	U17	BE5#	S04	D21	C16	D57	F20
A7	T16	BE6#	U07	D22	C07	D58	J18
A8	U16	BE7#	W01	D23	A10	D59	H19
A9	T15	BOFF#	K04	D24	B10	D60	L19
A10	U15	BP2	B02	D25	C08	D61	K19
A11	T14	BP3	B03	D26	C11	D62	J19
A12	U14	BRDY#	L04	D27	D09	D63	H18
A13	T13	BREQ	V02	D28	D11	D/C#	V04
A14	U13	BT0	T08	D29	C09	DP0	H04
A15	T12	BT1	W21	D30	D12	DP1	C05
A16	U12	BT2	T07	D31	C10	DP2	A9
A17	T11	BT3	W20	D32	D10	DP3	D08
A18	U11	BUSCHK#	T03	D33	C17	DP4	D18
A19	T10	CACHE#	J04	D34	C19	DP5	A19
A20	U10	CLK	K18	D35	D17	DP6	E19
A21	U21	D0	D03	D36	C18	DP7	E21
A22	U09	D1	E03	D37	D16	EADS#	M03
A23	U20	D2	E04	D38	D19	EWBE#	A03
A24	U08	D3	F03	D39	D15	FERR#	H03
A25	U19	D4	C04	D40	D14	FLUSH#	U02
A26	T09	D5	G03	D41	B19	FRCMC#	M19
A27	V21	D6	B04	D42	D20	HIT#	W02
A28	V06	D7	G04	D43	A20	HITM#	M04
A29	V20	D8	F04	D44	D21	HLDA	Q03
A30	W05	D9	C12	D45	A21	HOLD	V05
A31	V19	D10	C13	D46	E18	IBT	T19
A20M#	U05	D11	E05	D47	B20	IERR#	C02
ADS#	P04	D12	C14	D48	B21	IGNNE#	S20
AHOLD	L02	D13	D04	D49	F19	INIT	T20
AP	P03	D14	D13	D50	C20	INTR	N18
APCHK#	W03	D15	D05	D51	F18	INV	A01
BE0#	U04	D16	D06	D52	C21	IU	J02
BE1#	Q04	D17	B09	D53	G18	IV	B01

Table 2-1. Pentium® Processor (510\60, 567\66) Pin Cross Reference Table by Pin Name (Continued)

Signal	Location
KEN#	J03
LOCK#	V03
M/IO#	A02
NA#	K03
NMI	N19
PCD	W04
PCHK#	R03
PEN#	M18
PM0/BP0	D02
PM1/BP1	C03
PRDY	U03
PWT	S03

Signal	Location
RESET	L18
R/S#	R18
SCYC	R04
SMI#	P18
SMIACT#	T05
TCK	T04
TDI	T21
TDO	S21
TMS	P19
TRST#	S18
WB/WT#	M02
W/R#	N03

Signal	Location
NC	L03, N04, Q19, R19, S19, T18
V_{CC}	A04, A05, A06, A07, A08, A11, A12, A13, A14, A15, A16, A17, A18, C01, D01, E01, F01, F21, G01, G21, H01, J21 K21, L21, M21, N01, N21, P01, P21, Q01, Q18, Q21, R01, R21, S01, T01, U01, W06, W07, W08, W09, W10, W11, W12, W13, W14, W15, W16, W17, W18

Signal	Location
V_{SS}	B05, B06, B07, B08, B11, B12, B13, B14, B15, B16, B17, B18, E02, F02, G02, G20, H02, H20, J01, J20, K01, K02, K20, L01, L20, M01, M20, N02, N20, P02, P20, Q02, Q20, R02, R20, S02, T02, V07, V08, V09, V10, V11, V12, V13, V14, V15, V16, V17, V18

2.2 Design Notes

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active LOW inputs should be connected to V_{CC}. Unused active HIGH inputs should be connected to GND.

No Connect (NC) pins must remain unconnected. Connection of NC pins may result in component failure or incompatibility with processor steppings.

NOTE:
The No Connect pin located at L03 (BRDYC#) along with BUSCHK# are sampled by the Pentium processor (510\60, 567\66) at RESET to configure the I/O buffers of the processor for use with the 82496 Cache Controller/82491 Cache SRAM secondary cache as a chip set (refer to the *Pentium® Processor Family Developer's Manual,* Volume 2 for further information).

2.3 Quick Pin Reference

This section gives a brief functional description of each of the pins. For a detailed description, see the Hardware Interface chapter in the *Pentium® Processor Family Developer's Manual,* Volume 1. **Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior.** In this section, the pins are arranged in alphabetical order. The functional grouping of each pin is listed at the end of this chapter.

The # symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage. When a # symbol is not present after the signal name, the signal is active, or asserted at the high voltage level.

Table 2-2. Quick Pin Reference

Symbol	Type*	Name and Function
A20M#	I	When the *address bit 20 mask* pin is asserted, the Pentium® Processor (510\60, 567\66) emulates the address wraparound at one Mbyte which occurs on the 8086. When A20M# is asserted, the Pentium processor (510\60, 567\66) masks physical address bit 20 (A20) before performing a lookup to the internal caches or driving a memory cycle on the bus. The effect of A20M# is undefined in protected mode. A20M# must be asserted only when the processor is in real mode.
A31–A3	I/O	As outputs, the *address* lines of the processor along with the byte enables define the physical area of memory or I/O accessed. The external system drives the inquire address to the processor on A31–A5.
ADS#	O	The *address status* indicates that a new valid bus cycle is currently being driven by the Pentium processor (510\60, 567\66).
AHOLD	I/O	In response to the assertion of *address hold*, the Pentium processor (510\60, 567\66) will stop driving the address lines (A31–A3), and AP in the next clock. The rest of the bus will remain active so data can be returned or driven for previously issued bus cycles.
AP	I/O	*Address parity* is driven by the Pentium processor (510\60, 567\66) with even parity information on all Pentium processor (510\60, 567\66) generated cycles in the same clock that the address is driven. Even parity must be driven back to the Pentium processor (510\60, 567\66) during inquire cycles on this pin in the same clock as EADS# to ensure that the correct parity check status is indicated by the Pentium processor (510\60, 567\66).
APCHK#	O	The *address parity check* status pin is asserted two clocks after EADS# is sampled active if the Pentium processor (510\60, 567\66) has detected a parity error on the address bus during inquire cycles. APCHK# will remain active for one clock each time a parity error is detected.
BE7#–BE0#	O	The *byte enable* pins are used to determine which bytes must be written to external memory, or which bytes were requested by the CPU for the current cycle. The byte enables are driven in the same clock as the address lines (A31–3).
BOFF#	I	The *backoff* input is used to abort all outstanding bus cycles that have not yet completed. In response to BOFF#, the Pentium processor (510\60, 567\66) will float all pins normally floated during bus hold in the next clock. The processor remains in bus hold until BOFF# is negated at which time the Pentium processor (510\60, 567\66) restarts the aborted bus cycle(s) in their entirety.
BP[3:2] PM/BP[1:0]	O	The *breakpoint* pins (BP3–0) correspond to the debug registers, DR3–DR0. These pins externally indicate a breakpoint match when the debug registers are programmed to test for breakpoint matches. BP1 and BP0 are multiplexed with the Performance Monitoring pins (PM1 and PM0). The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of reset configured for performance monitoring.
BRDY#	I	The *burst ready* input indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted the Pentium processor (510\60, 567\66) data in response to a write request. This signal is sampled in the T2, T12 and T2P bus states.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
BREQ	O	The *bus request* output indicates to the external system that the Pentium processor (510\60, 567\66) has internally generated a bus request. This signal is always driven whether or not the Pentium processor (510\60, 567\66) is driving its bus.
BT3–BT0	O	The *branch trace* outputs provide bits 2-0 of the branch target linear address (BT2-BT0) and the default operand size (BT3) during a branch trace message special cycle.
BUSCHK#	I	The *bus check* input allows the system to signal an unsuccessful completion of a bus cycle. If this pin is sampled active, the Pentium processor (510\60, 567\66) will latch the address and control signals in the machine check registers. If in addition, the MCE bit in CR4 is set, the Pentium processor (510\60, 567\66) will vector to the machine check exception.
CACHE#	O	For Pentium processor (510\60, 567\66)-initiated cycles the *cache* pin indicates internal cacheability of the cycle (if a read), and indicates a burst writeback cycle (if a write). If this pin is driven inactive during a read cycle, Pentium processor (510\60, 567\66) will not cache the returned data, regardless of the state of the KEN# pin. This pin is also used to determine the cycle length (number of transfers in the cycle).
CLK	I	The *clock* input provides the fundamental timing for the Pentium processor (510\60, 567\66). Its frequency is the internal operating frequency of the Pentium processor (510\60, 567\66) and requires TTL levels. All external timing parameters except TDI, TDO, TMS and TRST# are specified with respect to the rising edge of CLK.
D/C#	O	The *Data/Code* output is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. D/C# distinguishes between data and code or special cycles.
D63–D0	I/O	These are the 64 *data lines* for the processor. Lines D7-D0 define the least significant byte of the data bus; lines D63-D56 define the most significant byte of the data bus. When the CPU is driving the data lines, they are driven during the T2, T12, or T2P clocks for that cycle. During reads, the CPU samples the data bus when BRDY# is returned.
DP7–DP0	I/O	These are the *data parity* pins for the processor. There is one for each byte of the data bus. They are driven by the Pentium processor (510\60, 567\66) with even parity information on writes in the same clock as write data. Even parity information must be driven back to the Pentium processor (510\60, 567\66) on these pins in the same clock as the data to ensure that the correct parity check status is indicated by the Pentium processor (510\60, 567\66). DP7 applies to D63–D56, DP0 applies to D7–D0.
EADS#	I	This signal indicates that a *valid external address* has been driven onto the Pentium processor (510\60, 567\66) address pins to be used for an inquire cycle.
EWBE#	I	The *external write buffer empty* input, when inactive (high), indicates that a write cycle is pending in the external system. When the Pentium processor (510\60, 567\66) generates a write, and EWBE# is sampled inactive, the Pentium processor (510\60, 567\66) will hold off all subsequent writes to all E or M-state lines in the data cache until all write cycles have completed, as indicated by EWBE# being active.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
FERR#	O	The *floating point error* pin is driven active when an unmasked floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS type floating point error reporting.
FLUSH#	I	When asserted, the *cache flush* input forces the Pentium processor (510\60, 567\66) to writeback all modified lines in the data cache and invalidate its internal caches. A Flush Acknowledge special cycle will be generated by the Pentium processor (510\60, 567\66) indicating completion of the writeback and invalidation. If FLUSH# is sampled low when RESET transitions from high to low, tristate test mode is entered.
FRCMC#	I	The *Functional Redundancy Checking Master/Checker* mode input is used to determine whether the Pentium processor (510\60, 567\66) is configured in master mode or checker mode. When configured as a master, the Pentium processor (510\60, 567\66) drives its output pins as required by the bus protocol. When configured as a checker, the Pentium processor (510\60, 567\66) tristates all outputs (except IERR# and TDO) and samples the output pins. The configuration as a master/checker is set after RESET and may not be changed other than by a subsequent RESET.
HIT#	O	The *hit* indication is driven to reflect the outcome of an inquire cycle. If an inquire cycle hits a valid line in either the Pentium processor (510\60, 567\66) data or instruction cache, this pin is asserted two clocks after EADS# is sampled asserted. If the inquire cycle misses Pentium processor (510\60, 567\66) cache, this pin is negated two clocks after EADS#. This pin changes its value only as a result of an inquire cycle and retains its value between the cycles.
HITM#	O	The *hit to a modified line* output is driven to reflect the outcome of an inquire cycle. It is asserted after inquire cycles which resulted in a hit to a modified line in the data cache. It is used to inhibit another bus master from accessing the data until the line is completely written back.
HLDA	O	The *bus hold acknowledge* pin goes active in response to a hold request driven to the processor on the HOLD pin. It indicates that the Pentium processor (510\60, 567\66) has floated most of the output pins and relinquished the bus to another local bus master. When leaving bus hold, HLDA will be driven inactive and the Pentium processor (510\60, 567\66) will resume driving the bus. If the Pentium processor (510\60, 567\66) has bus cycle pending, it will be driven in the same clock that HLDA is deasserted.
HOLD	I	In response to the *bus hold request*, the Pentium processor (510\60, 567\66) will float most of its output and input/output pins and assert HLDA after completing all outstanding bus cycles. The Pentium processor (510\60, 567\66) will maintain its bus in this state until HOLD is deasserted. HOLD is not recognized during LOCK cycles. The Pentium processor (510\60, 567\66) will recognize HOLD during reset.
IBT	O	The *instruction branch taken* pin is driven active (high) for one clock to indicate that a branch was taken. This output is always driven by the Pentium processor (510\60, 567\66).

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
IERR#	O	The *internal error* pin is used to indicate two types of errors, internal parity errors and functional redundancy errors. If a parity error occurs on a read from an internal array, the Pentium processor (510\60, 567\66) will assert the IERR# pin for one clock and then shutdown. If the Pentium processor (510\60, 567\66) is configured as a checker and a mismatch occurs between the value sampled on the pins and the corresponding value computed internally, the Pentium processor (510\60, 567\66) will assert IERR# two clocks after the mismatched value is returned.
IGNNE#	I	This is the *ignore numeric error* input. This pin has no effect when the NE bit in CR0 is set to 1. When the CR0.NE bit is 0, and the IGNNE# pin is asserted, the Pentium processor (510\60, 567\66) will ignore any pending unmasked numeric exception and continue executing floating point instructions for the entire duration that this pin is asserted. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating point instruction is one of FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (510\60, 567\66) will execute the instruction in spite of the pending exception. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating point instruction is one other than FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (510\60, 567\66) will stop execution and wait for an external interrupt.
INIT	I	The Pentium processor (510\60, 567\66) *initialization* input pin forces the Pentium processor (510\60, 567\66) to begin execution in a known state. The processor state after INIT is the same as the state after RESET except that the internal caches, write buffers, and floating point registers retain the values they had prior to INIT. INIT may NOT be used in lieu of RESET after power-up. If INIT is sampled high when RESET transitions from high to low the Pentium processor (510\60, 567\66) will perform built-in self test prior to the start of program execution.
INTR	I	An active *maskable interrupt* input indicates that an external interrupt has been generated. If the IF bit in the EFLAGS register is set, the Pentium processor (510\60, 567\66) will generate two locked interrupt acknowledge bus cycles and vector to an interrupt handler after the current instruction execution is completed. INTR must remain active until the first interrupt acknowledge cycle is generated to assure that the interrupt is recognized.
INV	I	The *invalidation* input determines the final cache line state (S or I) in case of an inquire cycle hit. It is sampled together with the address for the inquire cycle in the clock EADS# is sampled active.
IU	O	The *u-pipe instruction complete* output is driven active (high) for 1 clock to indicate that an instruction in the u-pipeline has completed execution. This pin is always driven by the Pentium processor (510\60, 567\66).
IV	O	The *v-pipe instruction complete* output is driven active (high) for one clock to indicate that an instruction in the v-pipeline has completed execution. This pin is always driven by the Pentium processor (510\60, 567\66).
KEN#	I	The *cache enable* pin is used to determine whether the current cycle is cacheable or not and is consequently used to determine cycle length. When the Pentium processor (510\60, 567\66) generates a cycle that can be cached (CACHE# asserted) and KEN# is active, the cycle will be transformed into a burst line fill cycle.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
LOCK#	O	The *bus lock* pin indicates that the current bus cycle is locked. The Pentium processor (510\60, 567\66) will not allow a bus hold when LOCK# is asserted (but AHOLD and BOFF# are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the BRDY# is returned for the last locked bus cycle. LOCK# is guaranteed to be deasserted for at least one clock between back to back locked cycles.
M/IO#	O	The *Memory/Input-Output* is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. M/IO# distinguishes between memory and I/O cycles.
NA#	I	An active *next address* input indicates that the external memory system is ready to accept a new bus cycle although all data transfers for the current cycle have not yet completed. The Pentium processor (510\60, 567\66) will drive out a pending cycle two clocks after NA# is asserted. The Pentium processor (510\60, 567\66) supports up to 2 outstanding bus cycles.
NMI	I	The *non-maskable interrupt* request signal indicates that an external non-maskable interrupt has been generated.
PCD	O	The *page cache disable* pin reflects the state of the PCD bit in CR3, the Page Directory Entry, or the Page Table Entry. The purpose of PCD is to provide an external cacheability indication on a page by page basis.
PCHK#	O	The *parity check* output indicates the result of a parity check on a data read. It is driven with parity status two clocks after BRDY# is returned. PCHK# remains low one clock for each clock in which a parity error was detected. Parity is checked only for the bytes on which valid data is returned.
PEN#	I	The *parity enable* input (along with CR4.MCE) determines whether a machine check exception will be taken as a result of a data parity error on a read cycle. If this pin is sampled active in the clock a data parity error is detected, the Pentium processor (510\60, 567\66) will latch the address and control signals of the cycle with the parity error in the machine check registers. If in addition the machine check enable bit in CR4 is set to "1", the Pentium processor (510\60, 567\66) will vector to the machine check exception before the beginning of the next instruction.
PM/BP[1:0]BP[3:2]	O	These pins function as part of the Performance Monitoring feature. The breakpoint pins BP[1:0] are multiplexed with the Performance Monitoring pins PM[1:0]. The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of reset configured for performance monitoring.
PRDY	O	The PRDY output pin indicates that the processor has stopped normal execution in response to the R/S# pin going active, or Probe Mode being entered.
PWT	O	The *page write through* pin reflects the state of the PWT bit in CR3, the Page Directory Entry, or the Page Table Entry. The PWT pin is used to provide an external writeback indication on a page by page basis.
R/S#	I	The R/S# input is an asynchronous, edge sensitive interrupt used to stop the normal execution of the processor and place it into an idle state. A high to low transition on the R/S# pin will interrupt the processor and cause it to stop execution at the next instruction boundary.

Table 2-2. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
RESET	I	*Reset* forces the Pentium processor (510\60, 567\66) to begin execution at a known state. All the Pentium processor (510\60, 567\66) internal caches will be invalidated upon the RESET. Modified lines in the data cache are not written back. FLUSH#, FRCMC# and INIT are sampled when RESET transitions from high to low to determine if tristate test mode or checker mode will be entered, or if BIST will be run.
SCYC	O	The *split cycle* output is asserted during misaligned LOCKed transfers to indicate that more than two cycles will be locked together. This signal is defined for locked cycles only. It is undefined for cycles which are not locked.
SMI#	I	The system Management Interrupt causes a system management interrupt request to be latched internally. When the latched SMI# is recognized on an instruction boundary, the processor enters System Management Mode.
SMIACT#	O	An active *system management interrupt active* output indicates that the processor is operating in System Management Mode (SMM).
TCK	I	The *testability clock* input provides the clocking function for the Pentium processor (510\60, 567\66) boundary scan in accordance with the IEEE Boundary Scan interface (Standard 1149.1). It is used to clock state information and data into and out of the Pentium processor (510\60, 567\66) during boundary scan.
TDI	I	The *test data input* is a serial input for the test logic. TAP instructions and data are shifted into the Pentium processor (510\60, 567\66) on the TDI pin on the rising edge of TCK when the TAP controller is in an appropriate state.
TDO	O	The *test data output* is a serial output of the test logic. TAP instructions and data are shifted out of the Pentium processor (510\60, 567\66) on the TDO pin on the falling edge of TCK when the TAP controller is in an appropriate state.
TMS	I	The value of the *test mode select* input signal sampled at the rising edge of TCK controls the sequence of TAP controller state changes.
TRST#	I	When asserted, the *test reset* input allows the TAP controller to be asynchronously initialized.
W/R#	O	*Write/Read* is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. W/R# distinguishes between write and read cycles.
WB/WT#	I	The writeback/writethrough input allows a data cache line to be defined as write back or write through on a line by line basis. As a result, it determines whether a cache line is initially in the S or E state in the data cache.

NOTE:
*The pins are classified as Input or Output based on their function in Master Mode. See the Functional Redundancy Checking section in the "Error Detection" chapter of the *Pentium® Processor Family Developer's Manual*, Volume 1, for further information.

2.4 Pin Reference Tables

Table 2-3. Output Pins

Name	Active Level	When Floated
ADS#	LOW	Bus Hold, BOFF#
APCHK#	LOW	
BE7#-BE0#	LOW	Bus Hold, BOFF#
BREQ	HIGH	
BT3-BT0	n/a	
CACHE#	LOW	Bus Hold, BOFF#
FERR#	LOW	
HIT#	LOW	
HITM#	LOW	
HLDA	HIGH	
IBT	HIGH	
IERR#	LOW	
IU	HIGH	
IV	HIGH	
LOCK#	LOW	Bus Hold, BOFF#
M/IO#, D/C#, W/R#	n/a	Bus Hold, BOFF#
PCHK#	LOW	
BP3-2, PM1/BP1, PM0/BP0	HIGH	
PRDY	HIGH	
PWT, PCD	HIGH	Bus Hold, BOFF#
SCYC	HIGH	Bus Hold, BOFF#
SMIACT#	LOW	
TDO	n/a	All states except Shift-DR and Shift-IR

NOTE:
All output and input/output pins are floated during tri-state test mode and checker mode (except IERR#).

Table 2-4. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal resistor	Qualified
A20M#	LOW	Asynchronous		
AHOLD	HIGH	Synchronous		
BOFF#	LOW	Synchronous		
BRDY#	LOW	Synchronous		Bus State T2, T12, T2P
BUSCHK#	LOW	Synchronous	Pullup	BRDY#
CLK	n/a			
EADS#	LOW	Synchronous		
EWBE#	LOW	Synchronous		BRDY#
FLUSH#	LOW	Asynchronous		
FRCMC#	LOW	Asynchronous		
HOLD	HIGH	Synchronous		
IGNNE#	LOW	Asynchronous		
INIT	HIGH	Asynchronous		
INTR	HIGH	Asynchronous		
INV	HIGH	Synchronous		EADS#
KEN#	LOW	Synchronous		First BRDY#/ NA#
NA#	LOW	Synchronous		Bus State T2, TD, T2P
NMI	HIGH	Asynchronous		
PEN#	LOW	Synchronous		BRDY#
R/S#	n/a	Asynchronous	Pullup	
RESET	HIGH	Asynchronous		
SMI#	LOW	Asynchronous	Pullup	
TCK	n/a		Pullup	
TDI	n/a	Synchronous/TCK	Pullup	TCK
TMS	n/a	Synchronous/TCK	Pullup	TCK
TRST#	LOW	Asynchronous	Pullup	
WB/WT#	n/a	Synchronous		First BRDY#/ NA#

Table 2-5. Input/Output Pins

Name	Active Level	When Floated	Qualified (when an input)
A31–A3	n/a	Address hold, Bus Hold, BOFF#	EADS#
AP	n/a	Address hold, Bus Hold, BOFF#	EADS#
D63–D0	n/a	Bus Hold, BOFF#	BRDY#
DP7–DP0	n/a	Bus Hold, BOFF#	BRDY#

NOTE:
All output and input/output pins are floated during tristate test mode (except TDO) and checker mode (except IERR# and TDO).

2.5 Pin Grouping According to Function

Table 2-6 organizes the pins with respect to their function.

Table 2-6. Pin Functional Grouping

Function	Pins
Clock	CLK
Initialization	RESET, INIT
Address Bus	A31-A3, BE7#-BE0#
Address Mask	A20M#
Data Bus	D63-D0
Address Parity	AP, APCHK#
Data Parity	DP7-DP0, PCHK#, PEN#
Internal Parity Error	IERR#
System Error	BUSCHK#
Bus Cycle Definition	M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK#
Bus Control	ADS#, BRDY#, NA#
Page Cacheability	PCD, PWT
Cache Control	KEN#, WB/WT#
Cache Snooping/ Consistency	AHOLD, EADS#, HIT#, HITM#, INV
Cache Flush	FLUSH#
Write Ordering	EWBE#
Bus Arbitration	BOFF#, BREQ, HOLD, HLDA
Interrupts	INTR, NMI
Floating Point Error Reporting	FERR#, IGNNE#
System Management Mode	SMI#, SMIACT#
Functional Redundancy Checking	FRCMC# (IERR#)
TAP Port	TCK, TMS, TDI, TDO, TRST#
Breakpoint/ Performance Monitoring	PM0/BP0, PM1/BP1, BP3-2
Execution Tracing	BT3-BT0, IU, IV, IBT
Probe Mode	R/S#, PRDY

2.6 Output Pin Grouping According to when Driven

This section groups the output pins according to when they are driven.

Group 1

The following output pins are driven active at the beginning of a bus cycle with ADS#. A31-A3 and AP are guaranteed to remain valid until AHOLD is asserted or until the earlier of the clock after NA# or the last BRDY#. The remaining pins are guaranteed to remain valid until the earlier of the clock after NA# or the last BRDY#:

A31-A3, AP, BE7#-0#, CACHE#, M/IO#, W/R#, D/C#, SCYC, PWT, PCD.

Group 2

As outputs, the following pins are driven in T2, T12, and T2P. As inputs, these pins are sampled with BRDY#:

D63-0, DP7-0.

Group 3

These are the status output pins. They are always driven:

BREQ, HIT#, HITM#, IU, IV, IBT, BT3-BT0, PM0/BP0, PM1/BP1, BP3, BP2, PRDY, SMIACT#.

Group 4

These are the glitch free status output pins.

APCHK#, FERR#, HLDA, IERR#, LOCK#, PCHK#.

3.0 ELECTRICAL SPECIFICATIONS

3.1 Power and Ground

For clean on-chip power distribution, the Pentium processor (510\60, 567\66) has 50 V_{CC} (power) and 49 V_{SS} (ground) inputs. Power and ground connections must be made to all external V_{CC} and V_{SS} pins of the Pentium processor (510\60, 567\66). On the circuit board, all V_{CC} pins must be connected to a V_{CC} plane. All V_{SS} pins must be connected to a V_{SS} plane.

3.2 Decoupling Recommendations

Liberal decoupling capacitance should be placed near the Pentium processor (510\60, 567\66). The Pentium processor (510\60, 567\66) driving its large address and data buses at high frequencies can cause transient power surges, particularly when driving large capacitive loads.

Low inductance capacitors (i.e. surface mount capacitors) and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by connecting capacitors directly to the V_{CC} and V_{SS} planes, with minimal trace length between the component pads and vias to the plane. Capacitors specifically for PGA packages are also commercially available.

These capacitors should be evenly distributed among each component. Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

3.3 Connection Specifications

All NC pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active high inputs should be connected to ground.

3.4 Maximum Ratings

Table 3-1 is a stress rating only. Functional operation at the maximums is not guaranteed. Functional operating conditions are given in the A.C. and D.C. specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium processor (510\60, 567\66) contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Table 3-1. Absolute Maximum Ratings

Case temperature under bias	−65°C to +110°C
Storage temperature	−65°C to +150°C
Voltage on any pin with respect to ground	−0.5 V_{CC} to V_{CC} + 0.5 (V)
Supply voltage with respect to V_{SS}	−0.5V to +6.5V

3.5 D.C. Specifications

Table 3-2 lists the D.C. specifications associated with the Pentium processor (510\60, 567\66).

Table 3-2. Pentium® Processor (510\60, 567\66) D.C. Specifications
V_{CC} = See Notes 10, 11; T_{case} = See Notes 12, 13

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL}	Input Low Voltage	−0.3	+0.8	V	TTL Level
V_{IH}	Input High Voltage	2.0	V_{CC} + 0.3	V	TTL Level
V_{OL}	Output Low Voltage		0.45	V	TTL Level, (1)
V_{OH}	Output High Voltage	2.4		V	TTL Level, (2)
I_{CC}	Power Supply Current		3200 2910	mA mA	66 MHz, (7), (8) 60 MHz, (7), (9)
I_{LI}	Input Leakage Current		±15	uA	$0 \le V_{IN} \le V_{CC}$, (4)
I_{LO}	Output Leakage Current		±15	uA	$0 \le V_{OUT} \le V_{CC}$ Tristate, (4)
I_{IL}	Input Leakage Current		−400	uA	V_{IN} = 0.45V, (5)
I_{IH}	Input Leakage Current		200	uA	V_{IN} = 2.4V, (6)
C_{IN}	Input Capacitance		15	pF	
C_O	Output Capacitance		20	pF	
$C_{I/O}$	I/O Capacitance		25	pF	
C_{CLK}	CLK Input Capacitance		8	pF	
C_{TIN}	Test Input Capacitance		15	pF	
C_{TOUT}	Test Output Capacitance		20	pF	
C_{TCK}	Test Clock Capacitance		8	pF	

NOTES:
1. Parameter measured at 4 mA load.
2. Parameter measured at 1 mA load.
4. This parameter is for input without pullup or pulldown.
5. This parameter is for input with pullup.
6. This parameter is for input with pulldown.
7. Worst case average I_{CC} for a mix of test patterns.
8. (16W max.) Typical Pentium processor (510\60, 567\66) supply current is 2600 mA (13W) at 66 MHz.
9. (14.6W max.) Typical Pentium processor (510\60, 567\66) supply current is 2370 mA (11.9W) at 60 MHz.
10. V_{CC} = 5V ± 5% at 60 MHz.
11. V_{CC} = 4.9V to 5.40V at 66 MHz.
12. T_{case} = 0°C to +80°C at 60 MHz.
13. T_{case} = 0°C to +70°C at 66 MHz.

3.6 A.C. Specifications

The 66 MHz and 60 MHz A.C. specifications given in Tables 3-3 and 3-4 consist of output delays, input setup requirements and input hold requirements. All A.C. specifications (with the exception of those for the TAP signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5 volts for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor (510\60, 567\66) operation.

Care should be taken to read all notes associated with a particular timing parameter. In addition, the following list of notes apply to the timing specification tables in general and are not associated with any one timing. They are 2, 5, 6, and 14.

Table 3-3. 66 MHz Pentium® Processor (510\60, 567\66) A.C. Specifications

V_{CC} = 4.9V to 5.40V; T_{case} = 0°C to +70°C; C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	33.33	66.66	MHz		1x CLK
t_1	CLK Period	15		ns	3.1	
t_{1a}	CLK Period Stability		±250	ps		(18), (19), (20), (21)
t_2	CLK High Time	4		ns	3.1	@2V, (1)
t_3	CLK Low Time	4		ns	3.1	@0.8V, (1)
t_4	CLK Fall Time	0.15	1.5	ns	3.1	(2.0V–0.8V), (1)
t_5	CLK Rise Time	0.15	1.5	ns	3.1	(0.8V–2.0V), (1)
t_6	ADS#, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Valid Delay	1.5	8.0	ns	3.2	
t_{6a}	AP Valid Delay	1.5	9.5	ns	3.2	
t_7	ADS#, AP, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10	ns	3.3	(1)
t_8	PCHK#, APCHK#, IERR#, FERR# Valid Delay	1.5	8.3	ns	3.2	(4)
t_9	BREQ, HLDA, SMIACT# Valid Delay	1.5	8.0	ns	3.2	(4)
t_{10}	HIT#, HITM# Valid Delay	1.5	8.0	ns	3.2	
t_{11}	PM0–1, BP0–3, IU, IV, IBT Valid Delay	1.5	10	ns	3.2	
t_{11a}	PRDY Valid Delay	1.5	8.0	ns	3.2	
t_{12}	D0–D63, DP0–7 Write Data Valid Delay	1.5	9	ns	3.2	
t_{13}	D0–63, DP0–7 Write Data Float Delay		10	ns	3.3	(1)
t_{14}	A5–A31 Setup Time	6.5		ns	3.4	
t_{15}	A5–A31 Hold Time	1.5		ns	3.4	
t_{16}	EADS#, INV, AP Setup Time	5		ns	3.4	
t_{17}	EADS#, INV, AP Hold Time	1.5		ns	3.4	
t_{18}	KEN#, WB/WT# Setup Time	5		ns	3.4	
t_{18a}	NA# Setup Time	4.5		ns	3.4	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.5		ns	3.4	
t_{20}	BRDY# Setup Time	5		ns	3.4	
t_{21}	BRDY# Hold Time	1.5		ns	3.4	
t_{22}	AHOLD, BOFF# Setup Time	5.5		ns	3.4	
t_{23}	AHOLD, BOFF# Hold Time	1.5		ns	3.4	

Table 3-3. 66 MHz Pentium® Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 4.9V to 5.40V; T_{case} = 0°C to +70°C; C_L = 0 pF (Continued)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5		ns	3.4	
t_{25}	BUSCHK#, EWBE#, HOLD, PEN# Hold Time	1.5		ns	3.4	
t_{26}	A20M#, INTR, Setup Time	5		ns	3.4	(12), (16)
t_{27}	A20M#, INTR, Hold Time	1.5		ns	3.4	(13)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5		ns	3.4	(16), (17)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.5		ns	3.4	
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2		CLKs		(15), (17)
t_{31}	R/S# Setup Time	5		ns	3.4	(12), (16), (17)
t_{32}	R/S# Hold Time	1.5		ns	3.4	(13)
t_{33}	R/S# Pulse Width, Async.	2		CLKs		(15), (17)
t_{34}	D0–D63 Read Data Setup Time	3.8		ns	3.4	
t_{34a}	DP0–7 Read Data Setup Time	3.8		ns	3.4	
t_{35}	D0–D63, DP0–7 Read Data Hold Time	2		ns	3.4	
t_{36}	RESET Setup Time	5		ns	3.5	(11), (12), (16)
t_{37}	RESET Hold Time	1.5		ns	3.5	(11), (13)
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	3.5	(11)
t_{39}	RESET Active After V_{CC} & CLK Stable	1		ms	3.5	power up, (11)
t_{40}	Pentium® processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5		ns	3.5	(12), (16), (17)
t_{41}	Pentium processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.5		ns	3.5	(13)
t_{42}	Pentium processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2		CLKs	3.5	(16)
t_{43}	Pentium processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time, Async.	2		CLKs	3.5	
t_{44}	TCK Frequency		16	MHz		
t_{45}	TCK Period	62.5		ns	3.1	

Table 3-3. 66 MHz Pentium® Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 4.9V to 5.40V; T_{case} = 0°C to +70°C; C_L = 0 pF (Continued)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{46}	TCK High Time	25		ns	3.1	@2V, (1)
t_{47}	TCK Low Time	25		ns	3.1	@0.8V, (1)
t_{48}	TCK Fall Time		5	ns	3.1	(2.0V–0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5	ns	3.1	(0.8V–2.0V), (1), (8), (9)
t_{50}	TRST# Pulse Width	40		ns	3.7	Asynchronous, (1)
t_{51}	TDI, TMS Setup Time	5		ns	3.6	(7)
t_{52}	TDI, TMS Hold Time	13		ns	3.6	(7)
t_{53}	TDO Valid Delay	3	20	ns	3.6	(8)
t_{54}	TDO Float Delay		25	ns	3.6	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3	20	ns	3.6	(3), (8), (10)
t_{56}	All Non-Test Outputs Float Delay		25	ns	3.6	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5		ns	3.6	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13		ns	3.6	(3), (7), (10)

NOTES:
1. Not 100% tested. Guaranteed by design/characterization.
2. TTL input test waveforms are assumed to be 0 to 3 Volt transitions with 1Volt/ns rise and fall times.
3. Non-Test Outputs and Inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch free outputs. Glitch free signals monotonically transition without false transitions (i.e. glitches).
5. 0.8 V/ns ≤ CLK input rise/fall time ≤ 8 V/ns.
6. 0.3 V/ns ≤ Input rise/fall time ≤ 5 V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 16 MHz.
10. During probe mode operation, use the normal specified timings. Do not use the boundary scan timings ($t_{55\text{-}58}$).
11. FRCMC# should be tied to V_{CC} (high) to ensure proper operation of the Pentium processor (510\60, 567\66) as a master Pentium processor (510\60, 567\66).
12. Setup time is required to guarantee recognition on a specific clock.
13. Hold time is required to guarantee recognition on a specific clock.
14. All TTL timings are referenced from 1.5 V.
15. To guarantee proper asynchronous recognition, the signal must have been deasserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously.
17. When driven asynchronously, NMI, FLUSH#, R/S#, INIT, and SMI# must be deasserted (inactive) for a minimum of 2 clocks before being returned active.
18. Functionality is guaranteed by design/characterization.
19. Measured on rising edge of adjacent CLKs at 1.5V.
20. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency.
21. The amount of jitter present must be accounted for as a component of CLK skew between devices.

Table 3-4. 60 MHz Pentium® Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 5V ±5%; T_{case} = 0°C to +80°C; C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	33.33	60	MHz		1x CLK
t_1	CLK Period	16.67		ns	3.1	
t_{1a}	CLK Period Stability		±250	ps		(18), (19), (20), (21)
t_2	CLK High Time	4		ns	3.1	@2V, (1)
t_3	CLK Low Time	4		ns	3.1	@0.8V,(1)
t_4	CLK Fall Time	0.15	1.5	ns	3.1	(2.0V–0.8V), (1)
t_5	CLK Rise Time	0.15	1.5	ns	3.1	(0.8V–2.0V), (1)
t_6	ADS#, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Valid Delay	1.5	9.0	ns	3.2	
t_{6a}	AP Valid Delay	1.5	10.5	ns	3.2	
t_7	ADS#, AP, A3–A31, BT0–3, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		11	ns	3.3	(1)
t_8	PCHK#, APCHK#, IERR#, FERR# Valid Delay	1.5	9.3	ns	3.2	(4)
t_9	BREQ, HLDA, SMIACT# Valid Delay	1.5	9.0	ns	3.2	(4)
t_{10}	HIT#, HITM# Valid Delay	1.5	9.0	ns	3.2	
t_{11}	PM0–1, BP0–3, IU, IV, IBT Valid Delay	1.5	11	ns	3.2	
t_{11a}	PRDY Valid Delay	1.5	9.0	ns	3.2	
t_{12}	D0–D63, DP0–7 Write Data Valid Delay	1.5	10	ns	3.2	
t_{13}	D0–D63, DP0–7 Write Data Float Delay		11	ns	3.3	(1)
t_{14}	A5–A31 Setup Time	7		ns	3.4	
t_{15}	A5–A31 Hold Time	1.5		ns	3.4	
t_{16}	EADS#, INV, AP Setup Time	5.5		ns	3.4	
t_{17}	EADS#, INV, AP Hold Time	1.5		ns	3.4	
t_{18}	KEN#, WB/WT# Setup Time	5.5		ns	3.4	
t_{18a}	NA# Setup Time	5.0		ns	3.4	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.5		ns	3.4	
t_{20}	BRDY# Setup Time	5.5		ns	3.4	
t_{21}	BRDY# Hold Time	1.5		ns	3.4	
t_{22}	AHOLD, BOFF# Setup Time	6		ns	3.4	
t_{23}	AHOLD, BOFF# Hold Time	1.5		ns	3.4	
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.5		ns	3.4	
t_{25}	BUSCHK#, EWBE#, HOLD, PEN# Hold Time	1.5		ns	3.4	

Table 3-4. 60 MHz Pentium® Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 5V ±5%; T_{CASE} = 0°C to +80°C; C_L = 0 pF (Continued)

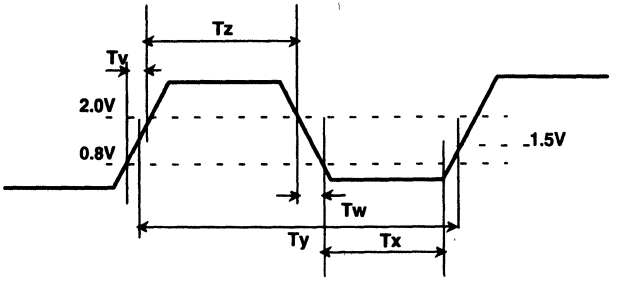
Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{26}	A20M#, INTR, Setup Time	5.5		ns	3.4	(12), (16)
t_{27}	A20M#, INTR, Hold Time	1.5		ns	3.4	(13)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.5		ns	3.4	(16), (17)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.5		ns	3.4	
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2		CLKs		(15), (17)
t_{31}	R/S# Setup Time	5.5		ns	3.4	(12), (16), (17)
t_{32}	R/S# Hold Time	1.5		ns	3.4	(13)
t_{33}	R/S# Pulse Width, Async.	2		CLKs		(15), (17)
t_{34}	D0-D63 Read Data Setup Time	4.3		ns	3.4	
t_{34a}	DP0-7 Read Data Setup Time	4.3		ns	3.4	
t_{35}	D0-D63, DP0-7 Read Data Hold Time	2		ns	3.4	
t_{36}	RESET Setup Time	5.5		ns	3.5	(11), (12), (16)
t_{37}	RESET Hold Time	1.5		ns	3.5	(11), (13)
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	3.5	(11)
t_{39}	RESET Active after V_{CC} & CLK Stable	1		ms	3.5	Power Up, (11)
t_{40}	Pentium® Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.5		ns	3.5	(12), (16), (17)
t_{41}	Pentium Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.5		ns	3.5	(13)
t_{42}	Pentium Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2		CLKs	3.5	(16)
t_{43}	Pentium Processor (510\60, 567\66) Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time, Async.	2		CLKs	3.5	
t_{44}	TCK Frequency		16	MHz		
t_{45}	TCK Period	62.5		ns	3.1	
t_{46}	TCK High Time	25		ns	3.1	@2V, (1)
t_{47}	TCK Low Time	25		ns	3.1	@0.8V, (1)
t_{48}	TCK Fall Time		5	ns	3.1	(2.0V-0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5	ns	3.1	(0.8V-2.0V), (1), (8), (9)

Table 3-4. 60 MHz Pentium® Processor (510\60, 567\66) A.C. Specifications
V_{CC} = 5V ±5%; T_{CASE} = 0°C to +80°C; C_L = 0 pF (Continued)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{50}	TRST# Pulse Width	40		ns	3.7	Async, (1)
t_{51}	TDI, TMS Setup Time	5		ns	3.6	(7)
t_{52}	TDI, TMS Hold Time	13		ns	3.6	(7)
t_{53}	TDO Valid Delay	3	20	ns	3.6	(8)
t_{54}	TDO Float Delay		25	ns	3.6	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3	20	ns	3.6	(3), (8), (10)
t_{56}	All Non-Test Outputs Float Delay		25	ns	3.6	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5		ns	3.6	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13		ns	3.6	(3), (7), (10)

NOTES:
1. Not 100% tested. Guaranteed by design/characterization.
2. TTL input test waveforms are assumed to be 0 to 3 Volt transitions with 1Volt/ns rise and fall times.
3. Non-Test Outputs and Inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch free outputs. Glitch free signals monotonically transition without false transitions (i.e. glitches).
5. 0.8 V/ns ≤ CLK input rise/fall time ≤ 8 V/ns.
6. 0.3 V/ns ≤ Input rise/fall time ≤ 5 V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 16 MHz.
10. During probe mode operation, use the normal specified timings. Do not use the boundary scan timings ($t_{55\text{-}58}$).
11. FRCMC# should be tied to V_{CC} (high) to ensure proper operation of the Pentium processor (510\60, 567\66) as a master Pentium processor (510\60, 567\66).
12. Setup time is required to guarantee recognition on a specific clock.
13. Hold time is required to guarantee recognition on a specific clock.
14. All TTL timings are referenced from 1.5 V.
15. To guarantee proper asynchronous recognition, the signal must have been deasserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously.
17. When driven asynchronously, NMI, FLUSH#, R/S#, INIT, and SMI# must be deasserted (inactive) for a minimum of 2 clocks before being returned active.
18. Functionality is guaranteed by design/characterization.
19. Measured on rising edge of adjacent CLKs at 1.5V.
20. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency.
21. The amount of jitter present must be accounted for as a component of CLK skew between devices.

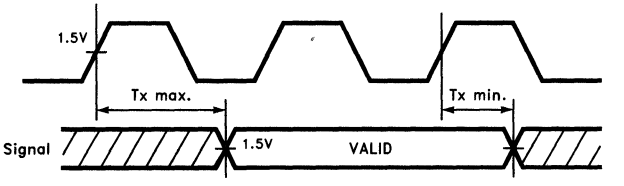
Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.



Tv = t5, t49
Tw = t4, t48
Tx = t3, t47
Ty = t1, t45
Tz = t2, t46

241595-5

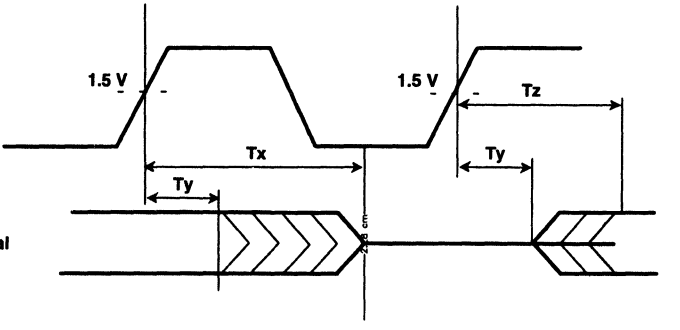
Figure 3-1. Clock Waveform



241595-6

Tx = t6, t6a, t8, t9, t10, t11, t11a, t12

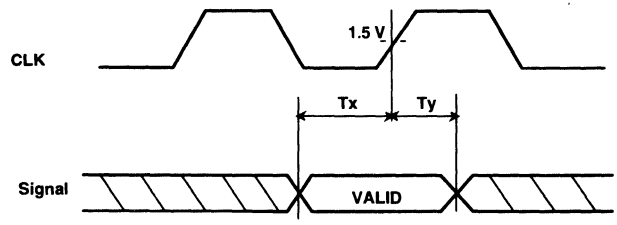
Figure 3-2. Valid Delay Timings



Tx = t7, t13
Ty = t6min, t12min

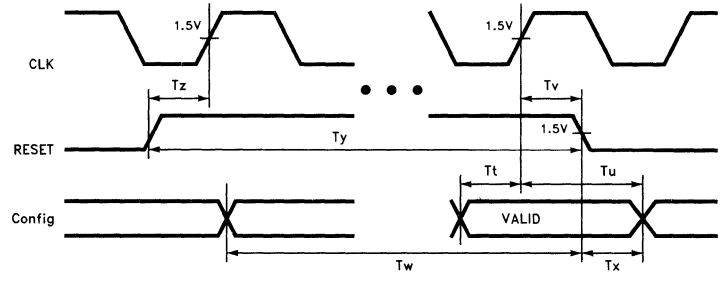
241595-7

Figure 3-3. Float Delay Timings



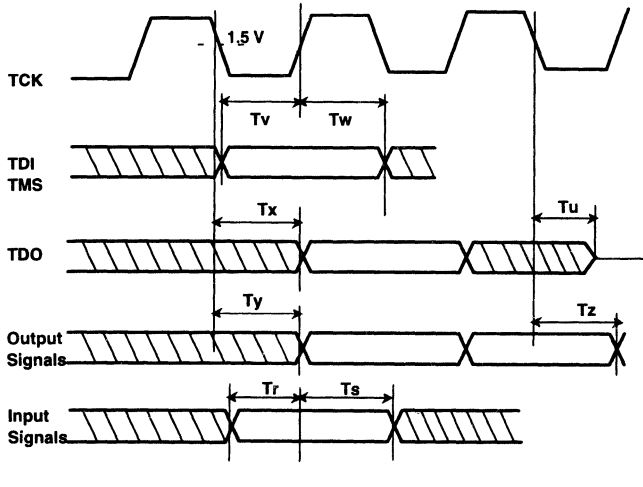
Tx = t14, t16, t18, t18a, t20, t22, t24, t26, t28, t31, t34, t34a
Ty = t15, t17, t19, t21, t23, t25, t27, t29, t32, t35

Figure 3-4. Setup and Hold Timings



Tt = t40
Tu = t41
Tv = t37
Tw = t42
Tx = t43
Ty = t38, t39
Tz = t36

Figure 3-5. Reset and Configuration Timings



Tr = t57
Ts = t58
Tu = t54
Tv = t51
Tw = t52
Tx = t53
Ty = t55
Tz = t56

Figure 3-6. Test Timings

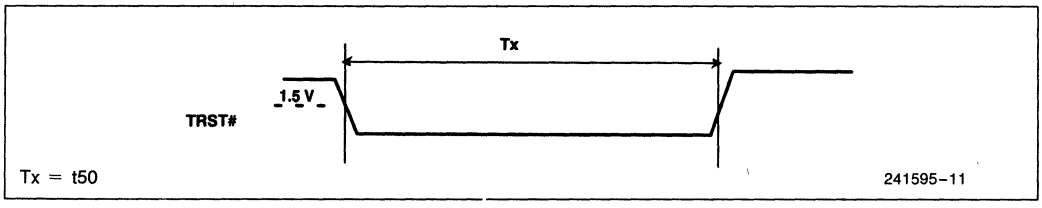


Figure 3-7. Test Reset Timings

4.0 MECHANICAL SPECIFICATIONS

The Pentium processor (510\60, 567\66) is packaged in a 273 pin ceramic pin grid array (PGA). The pins are arranged in a 21 by 21 matrix and the package dimensions are 2.16″ × 2.16″ (Table 4-1).

Figure 4-1 shows the package dimensions for the Pentium processor (510\60, 567\66). The mechanical specifications are provided in Table 4-2.

Table 4-1. Pentium® Processor (510\60, 567\66) Package Information Summary

	Package Type	Total Pins	Pin Array	Package Size	Estimated Wattage
Pentium® Processor (510\60, 567\66)	PGA	273	21 × 21	2.16″ × 2.16″ 5.49 cm × 5.49 cm	16

NOTE:
See D.C. Specifications for more detailed power specifications.

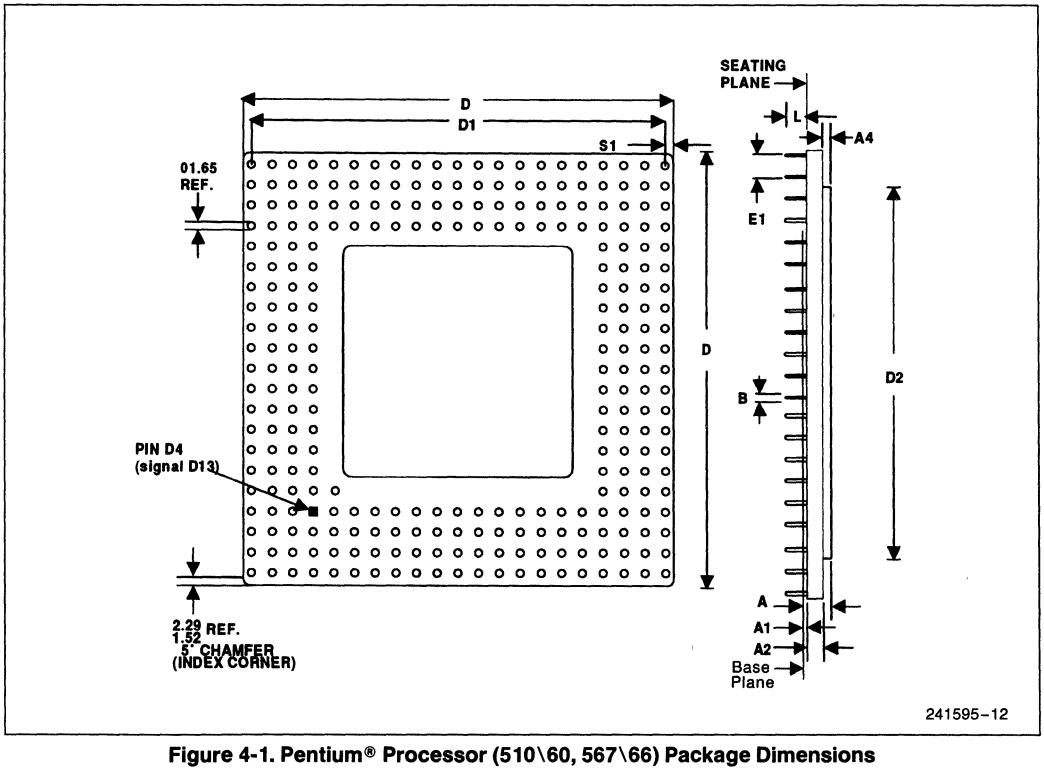


Figure 4-1. Pentium® Processor (510\60, 567\66) Package Dimensions

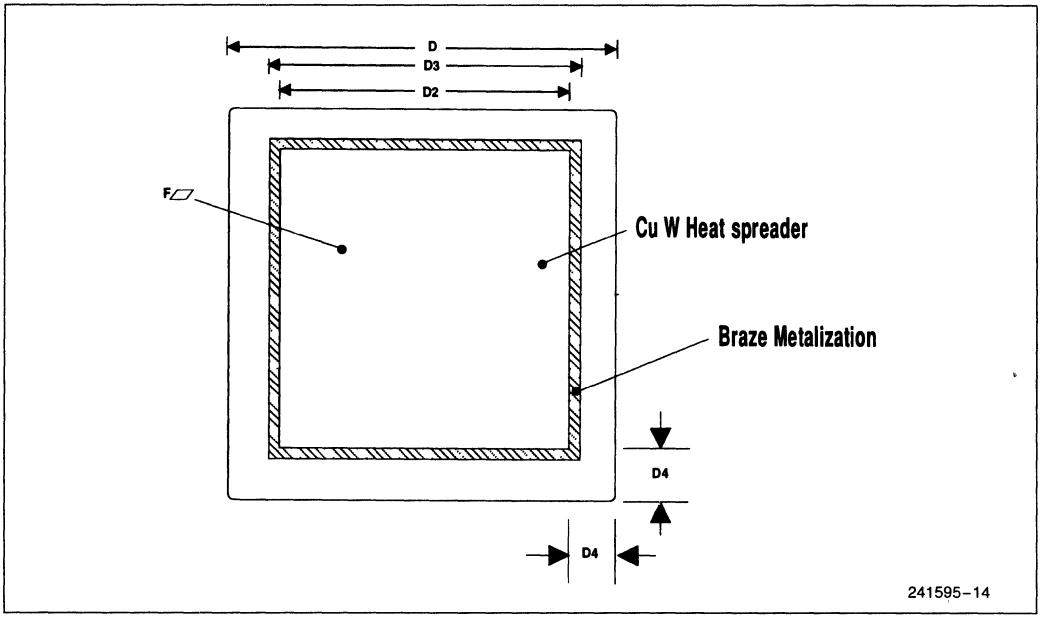


Figure 4-2. Pentium® Processor (510\60, 567\66) Package Dimensions

Table 4-2. Pentium® Processor (510\60, 567\66) Mechanical Specifications

Family: Ceramic Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A	3.91	4.70	Solid Lid	0.154	0.185	Solid Lid
A1	0.38	0.43	Solid Lid	0.015	0.017	Solid Lid
A2	2.62	4.30		0.103	0.117	
A4	0.97	1.22		0.038	0.048	
B	0.43	0.51		0.017	0.020	
D	54.66	55.07		2.152	2.168	
D1	50.67	50.93		1.995	2.005	
D2	37.85	38.35	Spreader Size	1.490	1.510	Spreader Size
D3	40.335	40.945	Braze	1.588	1.612	Braze
D4	8.382			0.330		
E1	2.29	2.79		0.090	0.110	
F	0.127		Flatness of spreader measured diagonally		0.005	Flatness of spreader measured diagonally
L	2.54	3.30		0.120	0.130	
N	273		Total Pins	273		Total Pins
S1	1.651	2.16		0.065	0.085	

5.0 THERMAL SPECIFICATIONS

The Pentium processor (510\60, 567\66) is specified for proper operation when T_C (case temperature) is within the specified range. To verify that the proper T_C is maintained, it should be measured at the center of the top surface (opposite of the pins) of the device in question. To minimize the measurement errors, it is recommended to use the following approach:

- Use 36 gauge or finer diameter k, t, or j type thermocouples. The laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using high thermal conductivity cements. The laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90 degrees angle as shown in Figure 5-1. When a heat sink is attached, a hole (no larger than 0.15″) should be drilled through the heat sink to allow probing the center of the package as shown in Figure 5-1.
- If the case temperature is measured with a heat sink attached to the package, drill a hole through the heat sink to route the thermocouple wire out.

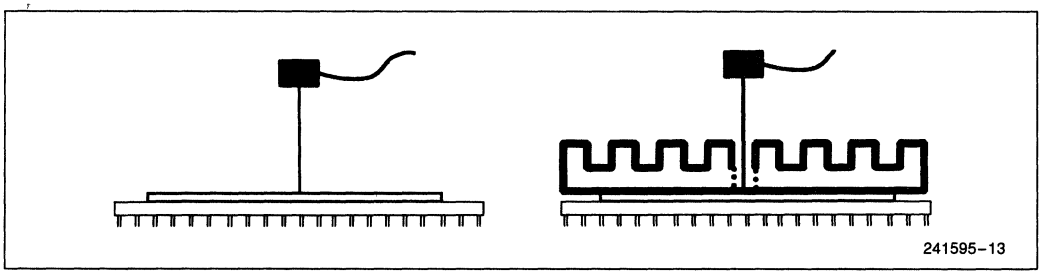


Figure 5-1. Technique for Measuring T_{case}

An ambient temperature T_A is not specified directly. The only restriction is that T_C is met. To determine the allowable T_A values, the following equations may be used:

$$T_J = TC + (P * \theta_{JC})$$

$$T_A = T_J - (P * \theta_{JA})$$

$$\theta_{CA} = \theta_{JA} - \theta_{JC}$$

$$T_A = T_C - (P * \theta_{CA})$$

where, T_J, T_A, and T_C = Junction, Ambient and Case Temperature, respectively.

θ_{JC}, θ_{JA}, and θ_{CA} = Junction-to-Case, Junction-to-Ambient, and Case-to-Ambient Thermal Resistance, respectively.

P = Maximum Power Consumption

Table 5-1 lists the θ_{JC} and θ_{CA} values for the Pentium processor (510\60, 567\66).

Table 5-1. Junction-to-Case and Case-to-Ambient Thermal Resistances for the Pentium® Processor (510\60, 567\66) (With and Without a Heat Sink)

	θ_{JC}	θ_{CA} vs Airflow (ft/min)					
		0	200	400	600	800	1000
With 0.25″ Heat Sink	0.6	8.3	5.4	3.5	2.6	2.1	1.8
With 0.35″ Heat Sink	0.6	7.4	4.5	3.0	2.2	1.8	1.6
With 0.65″ Heat Sink	0.6	5.9	3.0	1.9	1.5	1.2	1.1
Without Heat Sink	1.2	10.5	7.9	5.5	3.8	2.8	2.4

NOTES:
1. Heat Sink: 2.1 sq. in. base, omni-directional pin Al heat sink with 0.050 in. pin width, 0.143 in pin-to-pin center spacing and 0.150 in. base thickness. Heat sinks are attached to the package with a 2 to 4 mil thick layer of typical thermal grease. The thermal conductivity of this grease is about 1.2 w/m °C.
2. θ_{CA} values shown in Table 5-1 are typical values. The actual θ_{CA} values depend on the air flow in the system (which is typically unsteady, non uniform and turbulent) and thermal interactions between Pentium® processor (510\60, 567\66) and surrounding components though PCB and the ambient.

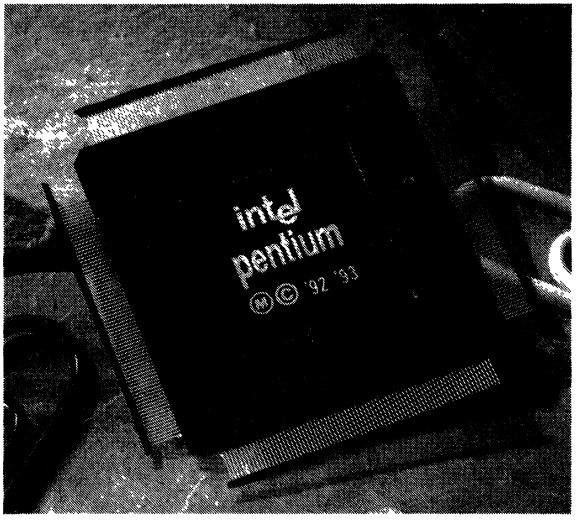
PENTIUM® PROCESSOR AT iCOMP® INDEX 610\75 MHz

- **Compatible with Large Software Base**
  - **MS-DOS, Windows, OS/2, UNIX**
- **32-Bit CPU with 64-Bit Data Bus**
- **Superscalar Architecture**
  - **Two Pipelined Integer Units Are Capable of 2 Instructions/Clock**
  - **Pipelined Floating Point Unit**
- **Separate Code and Data Caches**
  - **8K Code, 8K Writeback Data**
  - **MESI Cache Protocol**
- **Advanced Design Features**
  - **Branch Prediction**
  - **Virtual Mode Extensions**
- **3.3V BiCMOS Silicon Technology**
- **4M Pages for Increased TLB Hit Rate**
- **IEEE 1149.1 Boundary Scan**
- **Internal Error Detection Features**
- **SL Enhanced Power Management Features**
  - **System Management Mode**
  - **Clock Control**
- **Fractional Bus Operation**
  - **75-MHz Core/50-MHz Bus**

2

The Pentium® processor is fully compatible with the entire installed base of applications for DOS, Windows, OS/2, and UNIX, and all other software that runs on any earlier Intel 8086 family product. The Pentium processor's superscalar architecture can execute two instructions per clock cycle. Branch prediction and separate caches also increase performance. The pipelined floating-point unit delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor (610\75) has 3.3 million transistors, is built on Intel's advanced 3.3V BiCMOS silicon technology, and has full SL Enhanced power management features, including System Management Mode (SMM) and clock control. The additional SL Enhanced features, 3.3V operation, and the TCP package, which are not available in the Pentium processor (510\60, 567\66), make the Pentium processor (610\75) TCP ideal for enabling mobile Pentium processor designs.

The Pentium processor may contain design defects or errors known as errata. Current characterized errata are available upon request.



242323-21

November 1995
Order Number: 242323-002

PENTIUM® PROCESSOR AT iCOMP® INDEX 610\75 MHz

CONTENTS

CONTENTS

1.0 INTRODUCTION

Intel is now manufacturing its latest version of the Pentium® processor family that is designed specifically for mobile systems, with a core frequency of 75 MHz and a bus frequency of 50 MHz. The Pentium processor (610\75) is provided in the TCP (Tape Carrier Package) and SPGA packages, and has all of the advanced features of the Pentium processor (735\90, 815\100).

The new Pentium processor (610\75) TCP package has several features which allow high-performance notebooks to be designed with the Pentium processor, including the following:

- TCP package dimensions are ideal for small form-factor designs.
- The TCP package has superior thermal resistance characteristics.
- 3.3V V_{CC} reduces power consumption by half (in both the TCP and SPGA packages).
- The SL Enhanced feature set, which was initially implemented in the Intel486™ CPU.

The architecture and internal features of the Pentium processor (610\75) TCP and SPGA packages are identical to those of the Pentium processor (735\90, 815\100), although several features have been eliminated for the Pentium processor (610\75) TCP, as described in section 1.1.

This document should be used in conjunction with the Pentium processor documents listed below.

List of related documents:

- *Pentium® Processor Family Developer's Manual, Volume 1* (Order Number: 241428)
- *Pentium® Processor Family Developer's Manual, Volume 3* (Order Number: 241430)

1.1 Pentium® Processor (610\75) SPGA Specifications and Dif ferences from the TCP Package

This section provides references to the Pentium processor (610\75) SPGA specifications and describes the major differences between the Pentium processor (610\75) SPGA and TCP packages.

All Pentium processor (610\75) SPGA specifications, with the exception of power consumption, are identical to the Pentium processor (735\90, 815\100) specifications provided in the *Pentium® Processor Family Developer's Manual, Volume 1*. See Tables 8 and 11 in section 4.2 for the Pentium processor (610\75) SPGA and TCP power specifications.

The following features have been eliminated for the Pentium processor (610\75) TCP: the Upgrade feature, the Dual Processing (DP) feature, and the Master/Checker functional redundancy feature. Table 1 lists the corresponding pins which exist on the Pentium processor (610\75) SPGA but have been removed on the Pentium processor (610\75) TCP.

Table 1. SPGA Signals Removed in TCP

Signal	Function
ADSC#	Additional Address Status. This signal is mainly used for large or standalone L2 cache memory subsystem support required for high-performance desktop or server models.
BRDYC#	Additional Burst Ready. This signal is mainly used for large or standalone L2 cache memory subsystem support required for high-performance desktop or server models.
CPUTYP	CPU Type. This signal is used for dual processing systems.
D/P#	Dual/Primary processor identification. This signal is only used for an Upgrade processor.
FRCMC#	Functional Redundancy Checking. This signal is only used for error detection via processor redundancy, and requires two Pentium® processors (master/checker).
PBGNT#	Private Bus Grant. This signal is only used for dual processing systems.
PBREQ# PHIT#	Private Bus Request. This signal is used only for dual processing systems. Private Hit. This signal is only used for dual processing systems.
PHITM#	Private Modified Hit. This signal is only used for dual processing systems.

The I/O buffer models provided in section 4.4 of this document apply to both the Pentium processor (610\75) TCP and SPGA packages, although the capacitance (C_p) and inductance (L_p) parameter values differ between the two packages. Also, the thermal parameters, T_{CASE} max and θ_{CA}, differ between the TCP and SPGA packages. For Pentium processor (610\75) SPGA values, refer to Chapters 24 and 26 of the *Pentium® Processor Family Developer's Manual, Volume 1*.

2.0 MICROPROCESSOR ARCHITECTURE OVERVIEW

The Pentium processor at iCOMP® rating 610\75 MHz extends the Intel Pentium family of microprocessors. It is compatible with the 8086/88, 80286, Intel386™ DX CPU, Intel386 SX CPU, Intel486 DX CPU, Intel486 SX CPU, Intel486 DX2 CPUs, the Pentium processor at iCOMP Index 510\60 MHz and iCOMP Index 567\66 MHz, and the Pentium processor at iCOMP Index 735\90 MHz and iCOMP Index 815\100 MHz.

The Pentium processor family consists of the new Pentium processor at iCOMP rating 610\75 MHz, described in this document, the original Pentium processor (510\60, 567\66), and the Pentium processor (735\90, 815\100). The name "Pentium processor (610\75)" will be used in this document to refer to the Pentium processor at iCOMP rating 610\75 MHz. "Pentium Processor" will be used in this document to refer to the entire Pentium processor family in general.

The Pentium processor family architecture contains all of the features of the Intel486 CPU family, and provides significant enhancements and additions including the following:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Improved Instruction Execution Time
- Separate 8K Code and 8K Data Caches
- Writeback MESI Protocol in the Data Cache
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Execution Tracing
- Performance Monitoring
- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions

2.1 Pentium® Processor Family Architecture

The application instruction set of the Pentium processor family includes the complete Intel486 CPU family instruction set with extensions to accommodate some of the additional functionality of the Pentium processors. All application software written for the Intel386 and Intel486 family microprocessors will run on the Pentium processors without modification. The on-chip memory management unit (MMU) is completely compatible with the Intel386 family and Intel486 family of CPUs.

The Pentium processors implement several enhancements to increase performance. The two instruction pipelines and floating-point unit on Pentium processors are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, two floating-point instructions) in one clock.

Branch prediction is implemented in the Pentium processors. To s upport this, Pentium processors implement two prefetch buffers, one to prefetch code in a linear fashion, and one that prefetches code according to the BTB so the needed code is almost always prefetched before it is needed for execution.

The floating-point unit has been completely redesigned over the Intel486 CPU. Faster algorithms provide up to 10X speed-up for common operations including add, multiply, and load.

Pentium processors include separate code and data caches integrated on-chip to meet performance goals. Each cache is 8 Kbytes in size, with a 32-byte line size and is 2-way set associative. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses. The data cache is configurable to be writeback or writethrough on a line-by-line basis and follows the MESI protocol. The data cache tags are triple ported to support two data transfers and an inquire cycle in the same clock. The code cache is an inherently write-protected cache. The code cache tags are also triple ported to support snooping and split line accesses. Individual pages can be configured as cacheable or non-cacheable by software or hardware. The caches can be enabled or disabled by software or hardware.

The Pentium processors have increased the data bus to 64 bits to improve the data transfer rate. Burst read and burst writeback cycles are supported by the Pentium processors. In addition, bus cycle pipelining has been added to allow two bus cycles to be in progress simultaneously. The Pentium processors' Memory Management Unit contains optional extensions to the architecture which allow 2-Mbyte and 4-Mbyte page sizes.

The Pentium processors have added significant data integrity and error detection capability. Data parity checking is still supported on a byte-by-byte basis. Address parity checking, and internal parity checking features have been added along with a new exception, the machine check exception.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this, the Pentium processors have increased test and debug capability. The Pentium processors implement IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processors have specified 4 breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match. Execution tracing provides external indications when an instruction has completed execution in either of the two internal pipelines, or when a branch has been taken.

System Management Mode (SMM) has been implemented along with some extensions to the SMM architecture. Enhancements to the virtual 8086 mode have been made to increase performance by reducing the number of times it is necessary to trap to a virtual 8086 monitor.

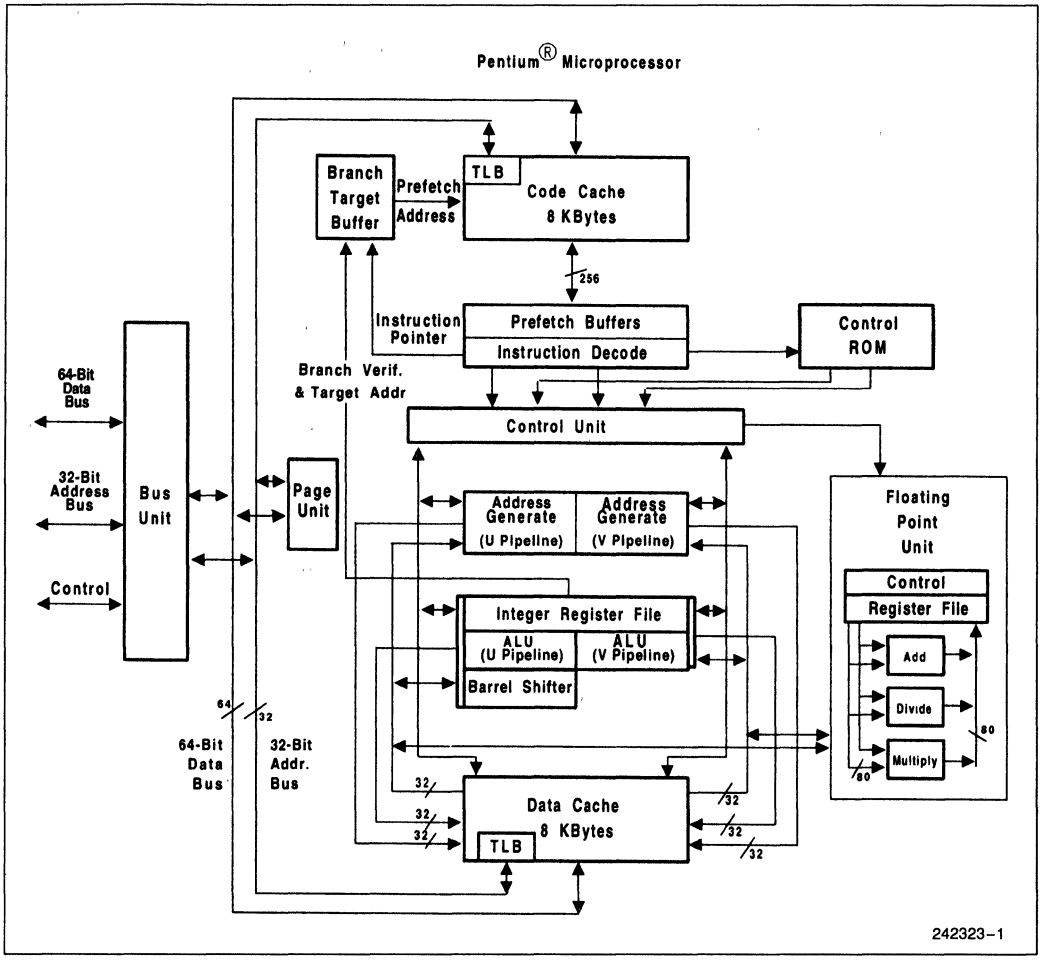


Figure 1. Pentium® Processor Block Diagram

The block diagram shows the two instruction pipelines, the "u" pipe and the "v" pipe. The u-pipe can execute all integer and floating point instructions. The v-pipe can execute simple integer instructions and the FXCH floating-point instructions.

The separate caches are shown, the code cache and data cache. The data cache has two ports, one for each of the two pipes (the tags are triple ported to allow simultaneous inquire cycles). The data cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium processor. Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so the Pentium processor can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that must be performed to implement the Pentium processor architecture. The control ROM unit has direct control over both pipelines.

The Pentium processors contain a pipelined floating-point unit that provides a significant floating-point performance advantage over previous generations of processors.

The architectural features introduced in this section are more fully described in the *Pentium® Processor Family Developer's Manual, Volume 3.*

3.0 TCP PINOUT

3.1 TCP Pinout and Pin Descriptions

3.1.1 Pentium® Processor (610\75) TCP PINOUT

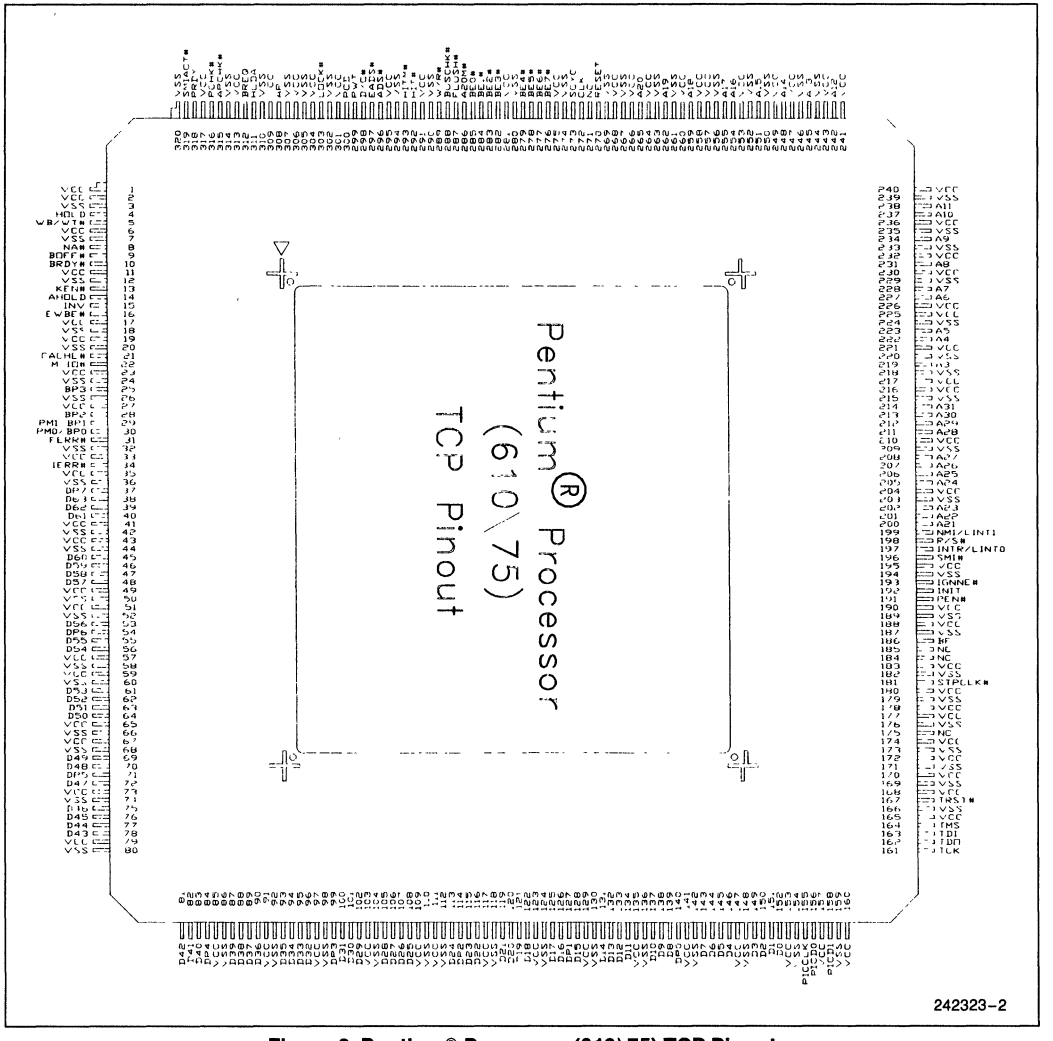


Figure 2. Pentium® Processor (610\75) TCP Pinout

3.1.2 PIN CROSS REFERENCE TABLE FOR Pentium® Processor (610\75) TCP

Table 2. TCP Pin Cross Reference by Pin Name

Address									
A3	219	A9	234	A15	251	A21	200	A27	208
A4	222	A10	237	A16	254	A22	201	A28	211
A5	223	A11	238	A17	255	A23	202	A29	212
A6	227	A12	242	A18	259	A24	205	A30	213
A7	228	A13	245	A19	262	A25	206	A31	214
A8	231	A14	248	A20	265	A26	207		
Data									
D0	152	D13	132	D26	107	D39	87	D52	62
D1	151	D14	131	D27	106	D40	83	D53	61
D2	150	D15	128	D28	105	D41	82	D54	56
D3	149	D16	126	D29	102	D42	81	D55	55
D4	146	D17	125	D30	101	D43	78	D56	53
D5	145	D18	122	D31	100	D44	77	D57	48
D6	144	D19	121	D32	96	D45	76	D58	47
D7	143	D20	120	D33	95	D46	75	D59	46
D8	139	D21	119	D34	94	D47	72	D60	45
D9	138	D22	116	D35	93	D48	70	D61	40
D10	137	D23	115	D36	90	D49	69	D62	39
D11	134	D24	113	D37	89	D50	64	D63	38
D12	133	D25	108	D38	88	D51	63		

Table 2. TCP Pin Cross Reference by Pin Name (Continued)

Control							
A20M#	286	BREQ	312	HITM#	293	PM1/BP1	29
ADS#	296	BUSCHK#	288	HLDA	311	PRDY	318
AHOLD	14	CACHE#	21	HOLD	4	PWT	299
AP	308	D/C#	298	IERR#	34	R/S#	198
APCHK#	315	DP0	140	IGNNE#	193	RESET	270
BE0#	285	DP1	127	INIT	192	SCYC	273
BE1#	284	DP2	114	INTR/LINT0	197	SMI#	196
BE2#	283	DP3	99	INV	15	SMIACT#	319
BE3#	282	DP4	84	KEN#	13	TCK	161
BE4#	279	DP5	71	LOCK#	303	TDI	163
BE5#	278	DP6	54	M/IO#	22	TDO	162
BE6#	277	DP7	37	NA#	8	TMS	164
BE7#	276	EADS#	297	NMI/LINT1	199	TRST#	167
BOFF#	9	EWBE#	16	PCD	300	W/R#	289
BP2	28	FERR#	31	PCHK#	316	WB/WT#	5
BP3	25	FLUSH#	287	PEN#	191		
BRDY#	10	HIT#	292	PM0/BP0	30		
APIC				**Clock Control**			
PICCLK	155	PICD1	158	BF	186	STPCLK#	181
PICD0 [DPEN#]	156	[APICEN]		CLK	272		

Table 2. TCP Pin Cross Reference by Pin Name (Continued)

V_{CC}								
1*	35	73	123	168*	190*	230	257*	295
2	41*	79	129	170*	195*	232*	258	301
6*	43	85	135	172*	204	236	260*	304*
11*	49*	91	141	174*	210	240*	264	306
17*	51	97	147	177*	216	241	266*	309*
19	57*	103	153*	178	217*	243*	268*	313
23	59	109	157*	18 0*	221	247	275	317*
27*	65*	111*	160	183*	225*	249*	281	
33*	67	117	165*	188*	226	253	291	
V_{SS}								
3	50	104	166	209	250	302		
7	52	110	169	215	252	305		
12	58	112	171	218	256	307		
18	60	118	173	220	261	310		
20	66	124	176	224	263	314		
24	68	130	179	229	267	320		
26	74	136	182	233	269			
32	80	142	187	235	274			
36	86	148	189	239	280			
42	92	154	194	244	290			
44	98	159	203	246	294			
NC								
175	184	185	271					

NOTE:
*These V_{CC} pins are 3.3V supplies for the Pentium processor (610\75) TCP but will be lower voltage pins on future offerings of this microprocessor family. All other V_{CC} pins will remain at 3.3V.

3.2 Design Notes

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active HIGH inputs should be connected to GND (V_{SS}).

No Connect (NC) pins must remain unconnected. Connection of NC pins may result in component failure or incompatibility with processor steppings.

3.3 Quick Pin Reference

This section gives a brief functional description of each of the pins. For a detailed description, see the "Hardware Interface" chapter in the *Pentium® Processor Family Developer's Manual, Volume 1.*

Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior.

The # symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage. When a # symbol is not present after the signal name, the signal is active, or asserted at the high voltage level.

Table 3. Quick Pin Reference

Symbol	Type	Name and Function
A20M#	I	When the **address bit 20 mask** pin is asserted, the Pentium® processor (610\75) emulates the address wraparound at 1 Mbyte which occurs on the 8086. When A20M# is asserted, the Pentium processor (610\75) masks physical address bit 20 (A20) before performing a lookup to the internal caches or driving a memory cycle on the bus. The effect of A20M# is undefined in protected mode. A20M# must be asserted only when the processor is in real mode.
A31–A3	I/O	As outputs, the **address** lines of the processor along with the byte enables define the physical area of memory or I/O accessed. The external system drives the inquire address to the processor on A31-A5.
ADS#	O	The **address status** indicates that a new valid bus cycle is currently being driven by the Pentium processor (610\75).
AHOLD	I	In response to the assertion of **address hold,** the Pentium processor (610\75) will stop driving the address lines (A31-A3), and AP in the next clock. The rest of the bus will remain active so data can be returned or driven for previously issued bus cycles.
AP	I/O	**Address parity** is driven by the Pentium processor (610\75) with even parity information on all Pentium processor (610\75) generated cycles in the same clock that the address is driven. Even parity must be driven back to the Pentium processor (610\75) during inquire cycles on this pin in the same clock as EADS# to ensure that correct parity check status is indicated by the Pentium processor (610\75).
APCHK#	O	The **address parity check** status pin is asserted two clocks after EADS# is sampled active if the Pentium processor (610\75) has detected a parity error on the address bus during inquire cycles. APCHK# will remain active for one clock each time a parity error is detected.
[APICEN] PICD1	I	The **Advanced Programmable Interrupt Controller Enable** pin enables or disables the on-chip APIC interrupt controller. If sampled high at the falling edge of RESET, the APIC is enabled. APICEN shares a pin with the **Programmable Interrupt Controller Data 1** signal.
BE7#–BE5# BE4#-BE0#	O I/O	The **byte enable** pins are used to determine which bytes must be written to external memory, or which bytes were requested by the CPU for the current cycle. The byte enables are driven in the same clock as the address lines (A31–3). The lower four byte enable pins (BE3#–BE0#) are used on the Pentium processor (610\75) also as APIC ID inputs and are sampled at RESET for that function.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
[BF]	I	**Bus Frequency** determines the bus-to-core frequency ratio. BF is sampled at RESET, and cannot be changed until another non-warm (1 ms) assertion of RESET. Additionally, BF must not change values while RESET is active. For proper operation of the Pentium processor (610\75) this pin should be strapped high or low. When BF is strapped to V_{CC}, the processor will operate at a 2 to 3 bus to core frequency ratio. When BF is strapped to V_{SS}, the processor will operate at a 1 to 2 bus to core frequency ratio. If BF is left floating, the Pentium processor (610\75) defaults to a 2 to 3 bus ratio. Note the Pentium processor (610\75) will not operate at a 1 to 2 bus to core frequency ratio.
BOFF#	I	The **backoff** input is used to abort all outstanding bus cycles that have not yet completed. In response to BOFF#, the Pentium processor (610\75) will float all pins normally floated during bus hold in the next clock. The processor remains in bus hold until BOFF# is negated, at which time the Pentium processor (610\75) restarts the aborted bus cycle(s) in their entirety.
BP[3:2] PM/BP[1:0]	O	The **breakpoint** pins (BP30–0) correspond to the debug registers, DR3–DR0. These pins externally indicate a breakpoint match when the debug registers are programmed to test for breakpoint matches. BP1 and BP0 are multiplexed with the **performance monitoring** pins (PM1 and PM0). The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
BRDY#	I	The **burst ready** input indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted the Pentium processor (610\75) data in response to a write request. This signal is sampled in the T2, T12 and T2P bus states.
BREQ	O	The **bus request** output indicates to the external system that the Pentium processor (610\75) has internally generated a bus request. This signal is always driven whether or not the Pentium processor (610\75) is driving its bus.
BUSCHK#	I	The **bus check** input allows the system to signal an unsuccessful completion of a bus cycle. If this pin is sampled active, the Pentium processor (610\75) will latch the address and control signals in the machine check registers. If, in addition, the MCE bit in CR4 is set, the Pentium processor (610\75) will vector to the machine check exception.
CACHE#	O	For Pentium processor (610\75)-initiated cycles the **cache** pin indicates internal cacheability of the cycle (if a read), and indicates a burst writeback cycle (if a write). If this pin is driven inactive during a read cycle, the Pentium processor (610\75) will not cache the returned data, regardless of the state of the KEN# pin. This pin is also used to determine the cycle length (number of transfers in the cycle).

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
CLK	I	The **clock** input provides the fundamental timing for the Pentium processor (610\75).Its frequency is the operating frequency of the Pentium processor (610\75) external bus and requires TTL levels. All external timing parameters except TDI, TDO, TMS, TRST#, and PICD0–1 are specified with respect to the rising edge of CLK.
D/C#	O	The **data/code** output is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. D/C# distinguishes between data and code or special cycles.
D63–D0	I/O	These are the 64 **data lines** for the processor. Lines D7–D0 define the least significant byte of the data bus; lines D63–D56 define the most significant byte of the data bus. When the CPU is driving the data lines, they are driven during the T2, T12, or T2P clocks for that cycle. During reads, the CPU samples the data bus when BRDY# is returned.
DP7–DP0	I/O	These are the **data parity** pins for the processor. There is one for each byte of the data bus. They are driven by the Pentium processor (610\75) with even parity information on writes in the same clock as write data. Even parity information must be driven back to the Pentium processor (610\75) on these pins in the same clock as the data to ensure that the correct parity check status is indicated by the Pentium processor (610\75). DP7 applies to D63–D56; DP0 applies to D7–D0.
[DPEN#] PICD0	I/O	**Dual processing enable** is an output of the Dual processor and an input of the Primary processor. The Dual processor drives DPEN# low to the Primary processor at RESET to indicate that the Primary processor should enable dual processor mode. Since the dual processing feature is not supported on the Pentium processor (610\75) TCP package, DPEN# should never be asserted (low) at RESET. DPEN# shares a pin with PICD0.
EADS#	I	This signal indicates that a valid **external address** has been driven onto the Pentium processor (610\75) address pins to be used for an inquire cycle.
EWBE#	I	The **external write buffer empty** input, when inactive (high), indicates that a write cycle is pending in the external system. When the Pentium processor (610\75) generates a write, and EWBE# is sampled inactive, the Pentium processor (610\75) will hold off all subsequent writes to all E- or M-state lines in the data cache until all write cycles have completed, as indicated by EWBE# being active.
FERR#	O	The **floating point error** pin is driven active when an unmasked floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS-type floating point error reporting.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
FLUSH#	I	When asserted, the **cache flush** input forces the Pentium processor (610\75) to write back all modified lines in the data cache and invalidate its internal caches. A Flush Acknowledge special cycle will be generated by the Pentium processor (610\75) indicating completion of the writeback and invalidation. If FLUSH# is sampled low when RESET transitions from high to low, tristate test mode is entered.
HIT#	O	The **hit** indication is driven to reflect the outcome of an inquire cycle. If an inquire cycle hits a valid line in either the Pentium processor (610\75) data or instruction cache, this pin is asserted two clocks after EADS# is sampled asserted. If the inquire cycle misses the Pentium processor (610\75) cache, this pin is negated two clocks after EADS#. This pin changes its value only as a result of an inquire cycle and retains its value between the cycles.
HITM#	O	The **hit to a modified line** output is driven to reflect the outcome of an inquire cycle. It is asserted after inquire cycles which resulted in a hit to a modified line in the data cache. It is used to inhibit another bus master from accessing the data until the line is completely written back.
HLDA	O	The **bus hold acknowledge** pin goes active in response to a hold request driven to the processor on the HOLD pin. It indicates that the Pentium processor (610\75) has floated most of the output pins and relinquished the bus to another local bus master. When leaving bus hold, HLDA will be driven inactive and the Pentium processor (610\75) will resume driving the bus. If the Pentium processor (610\75) has a bus cycle pending, it will be driven in the same clock that HLDA is de-asserted.
HOLD	I	In response to the **bus hold request,** the Pentium processor (610\75) will float most of its output and input/output pins and assert HLDA after completing all outstanding bus cycles. The Pentium processor (610\75) will maintain its bus in this state until HOLD is de-asserted. HOLD is not recognized during LOCK cycles. The Pentium processor (610\75) will recognize HOLD during reset.
IERR#	O	The **internal error** pin is used to indicate internal parity errors. If a parity error occurs on a read from an internal array, the Pentium processor (610\75) will assert the IERR# pin for one clock and then shutdown.
IGNNE#	I	This is the **ignore numeric error** input. This pin has no effect when the NE bit in CR0 is set to 1. When the CR0.NE bit is 0, and the IGNNE# pin is asserted, the Pentium processor (610\75) will ignore any pending unmasked numeric exception and continue executing floating-point instructions for the entire duration that this pin is asserted. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked nume ric exception exists (SW.ES = 1), and the floating-point instruction is one of FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (610\75) will execute the instruction in spite of the pending exception. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating-point instruction is one other than FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor (610\75) will stop execution and wait for an external interrupt.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
INIT	I	The Pentium processor (610\75) **initialization** input pin forces the Pentium processor (610\75) to begin execution in a known state. The processor state after INIT is the same as the state after RESET except that the internal caches, write buffers, and floating point registers retain the values they had prior to INIT. INIT may NOT be used in lieu of RESET after power up. If INIT is sampled high when RESET transitions from high to low, the Pentium processor (610\75) will perform built-in self test prior to the start of program execution.
INTR/LINT0	I	An active **maskable interrupt** input indicates that an external interrupt has been generated. If the IF bit in the EFLAGS register is set, the Pentium processor (610\75) will generate two locked interrupt acknowledge bus cycles and vector to an interrupt handler after the current instruction execution is completed. INTR must remain active until the first interrupt acknowledge cycle is generated to assure that the interrupt is recognized. If the local APIC is enabled, this pin becomes local interrupt 0.
INV	I	The **invalidation** input determines the final cache line state (S or I) in case of an inquire cycle hit. It is sampled together with the address for the inquire cycle in the clock EADS# is sampled active.
KEN#	I	The **cache enable** pin is used to determine whether the current cycle is cacheable or not and is consequently used to determine cycle length. When the Pentium processor (610\75) generates a cycle that can be cached (CACHE# asserted) and KEN# is active, the cycle will be transformed into a burst line fill cycle.
LINT0/INTR	I	If the APIC is enabled, this pin is **local interrupt 0.** If the APIC is disabled, this pin is **interrupt**.
LINT1/NMI	I	If the APIC is enabled, this pin is **local interrupt 1.** If the APIC is disabled, this pin is **non-maskable interrupt**.
LOCK#	O	The **bus lock** pin indicates that the current bus cycle is locked. The Pentium processor (610\75) will not allow a bus hold when LOCK# is asserted (but AHOLD and BOFF# are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the BRDY# is returned for the last locked bus cycle. LOCK# is guaranteed to be de-asserted for at least one clock between back-to-back locked cycles.
M/IO#	O	The **memory/input-output** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. M/IO# distinguishes between memory and I/O cycles.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
NA#	I	An active **next address** input indicates that the external memory system is ready to accept a new bus cycle although all data transfers for the current cycle have not yet completed. The Pentium processor (610\75) will issue ADS# for a pending cycle two clocks after NA# is asserted. The Pentium processor (610\75) supports up to 2 outstanding bus cycles.
NMI/LINT1	I	The **non-maskable interrupt** request signal indicates that an external non-maskable interrupt has been generated. If the local APIC is enabled, this pin becomes local interrupt 1.
PCD	O	The **page cache disable** pin reflects the state of the PCD bit in CR3, the Page Directory Entry, or the Page Table Entry. The purpose of PCD is to provide an external cacheability indication on a page-by-page basis.
PCHK#	O	The **parity check** output indicates the result of a parity check on a data read. It is driven with parity status two clocks after BRDY# is returned. PCHK# remains low one clock for each clock in which a parity error was detected. Parity is checked only for the bytes on which valid data is returned.
PEN#	I	The **parity enable** input (along with CR4.MCE) determines whether a machine check exception will be taken as a result of a data parity error on a read cycle. If this pin is sampled active in the clock a data parity error is detected, the Pentium processor (610\75) will latch the address and control signals of the cycle with the parity error in the machine check registers. If, in addition, the machine check enable bit in CR4 is set to "1", the Pentium processor (610\75) will vector to the machine check exception before the beginning of the next instruction.
PICCLK	I	The APIC interrupt controller serial data bus clock is driven into the **programmable interrupt controller clock** input of the Pentium processor (610\75).
PICD0-1 [DPEN#] [APICEN]	I/O	**Programmable interrupt controller data lines 0–1** of the Pentium processor (610\75) comprise the data portion of the APIC 3-wire bus. They are open-drain outputs that require external pull-up resistors. These signals share pins with DPEN# and APICEN.
PM/BP[1:0]	O	These pins function as part of the performance monitoring feature. The breakpoint 1-0 pins are multiplexed with the **performance monitoring 1–0 pins.** The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
PRDY	O	The **probe ready** output pin indicates that the processor has stopped normal execution in response to the R/S# pin going active, or Probe Mode being entered.
PWT	O	The **page writethrough** pin reflects the state of the PWT bit in CR3, the page directory entry, or the page table entry. The PWT pin is used to provide an external writeback indication on a page-by-page basis.

Table 3. Quick Pin Reference (Continued)

Symbol	Type*	Name and Function
R/S#	I	The **run/stop** input is an asynchronous, edge-sensitive interrupt used to stop the normal execution of the processor and place it into an idle state. A high to low transition on the R/S# pin will interrupt the processor and cause it to stop execution at the next instruction boundary.
RESET	I	**RESET** forces the Pentium processor (610\75) to begin execution at a known state. All the Pentium processor (610\75) internal caches will be invalidated upon the RESET. Modified lines in the data cache are not written back. FLUSH# and INIT are sampled when RESET transitions from high to low to determine if tristate test mode will be entered, or if BIST will be run.
SCYC	O	The **split cycle** output is asserted during misaligned LOCKed transfers to indicate that more than two cycles will be locked together. This signal is defined for locked cycles only. It is undefined for cycles which are not locked.
SMI#	I	The **system management interrupt** causes a system management interrupt request to be latched internally. When the latched SMI# is recognized on an instruction boundary, the processor enters System Management Mode.
SMIACT#	O	An active **system management interrupt active** output indicates that the processor is operating in System Management Mode.
STPCLK#	I	Assertion of the **stop clock** input signifies a request to stop the internal clock of the Pentium processor (610\75) thereby causing the core to consume less power. When the CPU recognizes STPCLK#, the processor will stop execution on the next instruction boundary, unless superseded by a higher priority interrupt, and generate a Stop Grant Acknowledge cycle. When STPCLK# is asserted, the Pentium processor (610\75) will still respond to external snoop requests.
TCK	I	The **testability clock** input provides the clocking function for the Pentium processor (610\75) boundary scan in accordance with the IEEE Boundary Scan interface (Standard 1149.1). It is used to clock state information and data into and out of the Pentium processor (610\75) during boundary scan.
TDI	I	The **test data input** is a serial input for the test logic. TAP instructions and data are shifted into the Pentium processor (610\75) on the TDI pin on the rising edge of TCK when the TAP controller is in an appropriate state.
TDO	O	The **test data output** is a serial output of the test logic. TAP instructions and data are shifted out of the Pentium processor (610\75) on the TDO pin on TCK's falling edge when the TAP controller is in an appropriate state.
TMS	I	The value of the **test mode select** input signal sampled at the rising edge of TCK controls the sequence of TAP controller state changes.
TRST#	I	When asserted, the **test reset** input allows the TAP controller to be asynchronously initialized.

Table 3. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
V_{CC} V_{SS}	I I	The Pentium processor (610\75) has 79 3.3V **power** inputs. The Pentium processor (610\75) has 72 **ground** inputs.
W/R#	O	**Write/read** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. W/R# distinguishes between write and read cycles.
WB/WT#	I	The **writeback/writethrough** input allows a data cache line to be defined as writeback or writethrough on a line-by-line basis. As a result, it determines whether a cache line is initially in the S or E state in the data cache.

3.4 Pin Reference Tables

Table 4. Output Pins

Name	Active Level	When Floated
ADS#	Low	Bus Hold, BOFF#
APCHK#	Low	
BE7#-BE5#	Low	Bus Hold, BOFF#
BR EQ	High	
CACHE#	Low	Bus Hold, BOFF#
FERR#	Low	
HIT#	Low	
HITM#	Low	
HLDA	High	
IERR#	Low	
LOCK#	Low	Bus Hold, BOFF#
M/IO#, D/C#, W/R#	n/a	Bus Hold, BOFF#
PCHK#	Low	
BP3-2, PM1/BP1, PM0/BP0	High	
PRDY	High	
PWT, PCD	High	Bus Hold, BOFF#
SCYC	High	Bus Hold, BOFF#
SMIACT#	Low	
TDO	n/a	All states except Shift-DR and Shift-IR

NOTE:
All output and input/output pins are floated during tristate test mode (except TDO).

Table 5. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal resistor	Qualified
A20M#	Low	Asynchronous		
AHOLD	High	Synchronous		
BF	High	Synchronous/RESET	Pullup	
BOFF#	Low	Synchronous		
BRDY#	Low	Synchronous		Bus State T2, T12, T2P
BUSCHK#	Low	Synchronous	Pullup	BRDY#
CLK	n/a			
EADS#	Low	Synchronous		
EWBE#	Low	Synchronous		BRDY#
FLUSH#	Low	Asynchronous		
HOLD	High	Synchronous		
IGNNE#	Low	Asynchronous		
INIT	High	Asynchronous		
INTR	High	Asynchronous		
INV	High	Synchronous		EADS#
KEN#	Low	Synchronous		First BRDY#/NA#
NA#	Low	Synchronous		Bus State T2,TD,T2P
NMI	High	Asynchronous		
PEN#	Low	Synchronous		BRDY#
PICCLK	High	Asynchronous	Pullup	
R/S#	n/a	Asynchronous	Pullup	
RESET	High	Asynchronous		
SMI#	Low	Asynchronous	Pullup	
STPCLK#	Low	Asynchronous	Pullup	
TCK	n/a		Pullup	
TDI	n/a	Synchronous/TCK	Pullup	TCK
TMS	n/a	Synchronous/TCK	Pullup	TCK
TRST#	Low	Asynchronous	Pullup	
WB/WT#	n/a	Synchronous		First BRDY#/NA#

Table 6. Input/Output Pins

Name	Active Level	When Floated	Qualified (when an input)	Internal Resistor
A31-A3	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
AP	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
BE4#-BE0#	Low	Bus Hold, BOFF#	RESET	Pulldown*
D63-D0	n/a	Bus Hold, BOFF#	BRDY#	
DP7-DP0	n/a	Bus Hold, BOFF#	BRDY#	
PICD0[DPEN#]				Pullup
PICD1[APICEN]				Pulldown

NOTES:
All output and input/output pins are floated during tristate test mode (except TDO).
*BE3#–BE0# have pulldowns during RESET only.

3.5 Pin Grouping According to Function

Table 7 organizes the pins with respect to their function.

Table 7. Pin Functional Grouping

Function	Pins
Clock	CLK
Initialization	RESET, INIT
Address Bus	A31-A3, BE7#-BE0#
Address Mask	A20M#
Data Bus	D63-D0
Address Parity	AP, APCHK#
APIC Support	PICCLK, PICD0-1
Data Parity	DP7-DP0, PCHK#, PEN#
Internal Parity Error	IERR#
System Error	BUSCHK#
Bus Cycle Definition	M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK#
Bus Control	ADS#, BRDY#, NA#
Page Cacheability	PCD, PWT
Cache Control	KEN#, WB/WT#
Cache Snooping/Consistency	AHOLD, EADS#, HIT#, HITM#, INV
Cache Flush	FLUSH#
Write Ordering	EWBE#
Bus Arbitration	BOFF#, BREQ, HOLD, HLDA
Interrupts	INTR, NMI
Floating Point Error Reporting	FERR#, IGNNE#
System Management Mode	SMI#, SMIACT#
TAP Port	TCK, TMS, TDI, TDO, TRST#
Breakpoint/Performance Monitoring	PM0/BP0, PM1/BP1, BP3-2
Clock Control	STPCLK#
Probe Mode	R/S#, PRDY

4.0 Pentium® Processor (610\75) TCP ELECTRICAL SPECIFICATIONS

4.1 Absolute Maximum Ratings

The following values are stress ratings only. Functional operation at the maximum ratings is not implied or guaranteed. Functional operating conditions are given in the AC and DC specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium processor (610\75) contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Case temperature under bias ..... −65°C to 110°C

Storage temperature ..... −65°C to +150°C

3V Supply voltage with respect to V_{SS} ..... −0.5V to +4.6V

3V Only Buffer DC Input Voltage..... −0.5V to V_{CC} +0.5; not to exceed 4.6V(2)

5V Safe Buffer DC Input Voltage..... −0.5V to 6.5V(1,3)

NOTES:
1. Applies to CLK and PICCLK.
2. Applies to all Pentium processor (610\75) inputs except CLK and PICCLK.
3. See Table 9.

WARNING

Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

4.2 DC Specifications

Tables 8, 9, and 10 list the DC specifications which apply to the Pentium processor (610\7 5). The Pentium processor (610\75) is a 3.3V part internally. The CLK and PICCLK inputs may be a 3.3V or 5V inputs. Since the 3.3V (5V safe) input levels defined in Table 9 are the same as the 5V TTL levels, the CLK and PICCLK inputs are compatible with existing 5V clock drivers. The power dissipation specification in Table 11 is provided for design of thermal solutions during operation in a sustained maximum level. This is the worst-case power the device would dissipate in a system for a sustain ed period of time. This number is used for design of a thermal solution for the device.

Table 8. 3.3V DC Specifications

T_{CASE} = 0 to 95°C; V_{CC} = 3.3V ±5%

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL3}	Input Low Voltage	−0.3	0.8	V	TTL Level (3)
V_{IH3}	Input High Voltage	2.0	V_{CC}+0.3	V	TTL Level(3)
V_{OL3}	Output Low Voltage		0.4	V	TTL Level(1) (3)
V_{OH3}	Output High Voltage	2.4		V	TTL Level(2) (3)
I_{CC3}	Power Supply Current		2650	mA	@75 MHz(4)

NOTES:
1. Parameter measured at 4 mA.
2. Parameter measured at 3 mA.
3. 33V TTL levels apply to all signals except CLK and PICCLK.
4. This value should be used for power supply design. It was determined usi ng a worst-case instruction mix and V_{CC} + 5%. Power supply transient response and decoupling capacitors must be sufficient to handle the instantaneous current changes occurring during transitions from stop clock to full active modes. For more information, refer to section 4.3.2.

Table 9. 3.3V (5V Safe) DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL5}	Input Low Voltage	−0.3	0.8	V	TTL Level (1)
V_{IH5}	Input High Voltage	2.0	5.55	V	TTL Level (1)

NOTES:
1.Applies to CLK and PICCLK only.

Table 10. Input and Output Characteristics

Symbol	Parameter	Min	Max	Unit	Notes
C_{IN}	Input Capacitance		15	pF	(4)
C_O	Output Capacitance		20	pF	(4)
$C_{I/O}$	I/O Capacitance		25	pF	(4)
C_{CLK}	CLK Input Capacitance		15	pF	(4)
C_{TIN}	Test Input Capacitance		15	pF	(4)
C_{TOUT}	Test Output Capacitance		20	pF	(4)
C_{TCK}	Test Clock Capacitance		15	pF	(4)
I_{LI}	Input Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{LO}	Output Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{IH}	Input Leakage Current		200	μA	$V_{IN} = 2.4V$(3)
I_{IL}	Input Leakage Current		−400	μA	$V_{IN} = 0.4V$(2)

NOTES:
1. This parameter is for input without pull up or pull down.
2. This parameter is for input with pull up.
3. This parameter is for input with pull down.
4. Guaranteed by design.

Table 11. Power Dissipation Requirements for Thermal Solution Design

Parameter	Typical(1)	Max(2)	Unit	Notes
Active Power Dissipation	3	8.0	Watts	@ 75 MHz
Stop Grant and Auto Halt Powerdown Power Dissipation		1.2	Watts	@ 75 MHz(3)
Stop Clock Power Dissipation	0.02	≤.05	Watts	(4) (5)

NOTES:
1. This is the typical power dissipation in a system. This value was t he average value measured in a system using a typical device at V_{CC} = 3.3V running typical applications. This value is highly dependent upon the specific system configuration.
2. Systems must be designed to thermally dissipate the maximum active power dissipation. It is determined using a worst-case instruction mix with V_{CC} = 3.3V. The use of nominal V_{CC} in this measurement takes into account the thermal time constant of the package.
3. Stop Grant/Auto Halt Powerdown Power Dissipation is determined by asserting the STPCLK# pin or executing the HALT instruction.
4. Stop Clock Power Dissipation is determined by asserting the STPCLK# pin and then removing the external CLK input.
5. Complete characterization of the specification was still in process at the time of print. Please contact Intel for the latest information. The final specification may be less than 50 mW.

2

4.3 AC Specifications

The AC specifications of the Pentium processor (610\75) consist of setup times, hold times, and valid delays at 0 pF.

WARNING

Do not exceed the Pentium processor (610\75) internal maximum frequency of 75 MHz by either selecting the 1/2 bus fraction or providing a clock greater than 50 MHz.

4.3.1 POWER AND GROUND

For clean on-chip power distribution, the Pentium processor (610\75) has 79 V_{CC} (power) and 72 V_{ss} (ground) inputs. Power and ground connections must be made to all external V_{CC} and V_{ss} pins of the Pentium processor (610\75). On the circuit board all V_{CC} pins must be connected to a 3.3V V_{CC} plane. All V_{ss} pins must be connected to a V_{ss} plane.

4.3.2 DECOUPLING RECOMMENDATIONS

Liberal decoupling capacitance should be placed near the Pentium processor (610\75). The Pentium processor (610\75) driving its large address and data buses at high frequencies can cause transient power surges, particularly when driving large capacitive loads.

Low inductance capacitors and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by shortening circuit board traces between the Pentium processor (610\75) and decoupling capacitors as much as possible.

These capacitors should be evenly distributed around each component on the 3.3V plane. Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

For the Pentium processor (610\ 75), the power consumption can transition from a low level of power to a much higher level (or high to low power) very rapidly. A typical example would be entering or exiting the Stop Grant state. Another example would be executing a HALT instruction, causing the Pentium processor (610\75) to enter the Auto HALT Powerdown state, or transitioning from HALT to the Normal state. All of these examples may cause abrupt changes in the power being consumed by the Pentium processor (610\75). Note that the Auto HALT Powerdown feature is always enabled even when other power management features are not implemented.

Bulk storage capacitors with a low ESR (Effective Series Resistance) in the 10 μf to 100 μf range are required to maintain a regulated supply voltage during the interval between the time the current load changes and the point that the regulated power supply output can react to the change in load. In order to reduce the ESR, it may be necessary to place several bulk storage capacitors in parallel.

These capacitors should be placed near the Pentium processor (610\75) (on the 3.3V plane) to ensure that the supply voltage stays within specified limits during changes in the supply current during operation.

4.3.3 CONNECTION SPECIFICATIONS

All NC pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active high inputs should be connected to ground.

4.3.4 AC TIMINGS FOR A 50-MHZ BUS

The AC specifications given in Table 12 consist of output delays, input setup requirements and input hold requirements for a 50-MHz external bus. All AC specifications (with the exception of those for the TAP signals and APIC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentiu m processor (610\75) operation.

Table 12. Pentium® Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation

V_{CC} = 3.3V± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	25.0	50.0	MHz		Max Core Freq. = 75 MHz @ 2/3
t_{1a}	CLK Period	20.0	40.0	nS	3	
t_{1b}	CLK Period Stability		250	pS		(1), (25)
t_2	CLK High Time	4.0		nS	3	@2V, (1)
t_3	CLK Low Time	4.0		nS	3	@0.8V, (1)
t_4	CLK Fall Time	0.15	1.5	nS	3	(2.0V–0.8V), (1), (5)
t_5	CLK Rise Time	0.15	1.5	nS	3	(0.8V–2.0V), (1), (5)
t_{6a}	PWT, PCD, M/IO#, CACHE# Valid Delay	1.0	7.0	nS	4	
t_{6b}	AP Valid Delay	1.0	8.5	nS	4	
t_{6c}	A3–A31, BE0–7# Valid Delay	0.7	7.0	nS	4	
t_{6d}	D/C#, SCYC, LOCK# Valid Delay	0.9	7.0	nS		
t_{6e}	ADS# Valid Delay	0.8	7.0	nS		
t_{6f}	W/R# Valid Delay	0.5	7.0	nS		

Table 12. Pentium® Processor (610\75) TCP
AC Specifications for 50-MHz Bus Operation (Continued)
V_{CC} = 3.3V ± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_7	ADS#, AP, A3-A31, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	5	(1)
t_8	APCHK#, IERR#, FERR#, PCHK# Valid Delay	1.0	8.3	nS	4	(4)
t_{9a}	BREQ, HLDA, SMIACT# Valid Delay	1.0	8.0	nS	4	(4)
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	4	
t_{10b}	HITM# Valid Delay	0.5	6.0	nS	4	
t_{11a}	PM0–1, BP0–3 Valid Delay	1.0	10.0	nS	4	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	4	
t_{12}	D0–D63, DP0–7 Write Data Valid Delay	1.3	8.5	nS	4	
t_{13}	D0–D63, DP0–3 Write Data Float Delay		10.0	nS	5	(1)
t_{14}	A5-A31 Setup Time	6.5		nS	6	(26)
t_{15}	A5-A31 Hold Time	1.0		nS	6	
t_{16a}	INV, AP Setup Time	5.0		nS	6	
t_{16b}	EADS# Setup Time	6.0		nS	6	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	6	
t_{18a}	KEN# Setup Time	5.0		nS	6	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	6	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	6	
t_{20}	BRDY# Setup Time	5.0		nS	6	
t_{21}	BRDY# Hold Time	1.0		nS	6	
t_{22}	BOFF# Setup Time	5.5		nS	6	
t_{22a}	AHOLD Setup Time	6.0		nS	6	
t_{23}	AHOLD, BOFF# Hold Time	1.1		nS	6	

Table 12. Pentium® Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation (Continued)

V_{CC} = 3.3V ± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	6	
t_{25}	BUSCHK#, EWBE #, PEN# Hold Time	1.0		nS	6	
t_{25a}	HOLD Hold Time	1.5		nS	6	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	6	(12), (16)
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	6	(13)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	6	(12), (16), (17)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.1		nS	6	(13)
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs	6	(15), (17)
t_{31}	R/S# Setup Time	5.0		nS	6	(12), (16), (17)
t_{32}	R/S# Hold Time	1.0		nS	6	(13)
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs	6	(15), (17)
t_{34}	D0–D63, DP0–7 Read Data Setup Time	3.8		nS	6	
t_{35}	D0–D63, DP0–7 Read Data Hold Time	2.0		nS	6	
t_{36}	RESET Setup Time	5.0		nS	7	(11), (12), (16)
t_{37}	RESET Hold Time	1.0		nS	7	(11), (13)
t_{38}	RESET Pulse Width, Vcc & CLK Stable	15		CLKs	7	(11), (17)
t_{39}	RESET Active After Vcc & CLK Stable	1.0		mS	7	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#) Setup Time	5.0		nS	7	(12), (16), (17)
t_{41}	Reset Configuration Signals (INIT, FLUSH#) Hold Time	1.0		nS	7	(13)

Table 12. Pentium® Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation (Continued)

V_{CC} = 3.3V ± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{42a}	Reset Configuration Signals (INIT, FLUSH#) Setup Time, Async.	2.0		CLKs	7	To RESET falling edge (16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, BRDY#, BUSCHK#) Hold Time, Async.	2.0		CLKs	7	To RESET falling edge (27)
t_{42c}	Reset Configuration Signal (BRDY#, BUSCHK#) Setup Time, Async.	3.0		CLKs	7	To RESET falling edge (27)
t_{42d}	Reset Configuration Signal BRDY# Hold Time, RESET driven synchronously	1.0		ns		To RESET falling edge (1), (27)
t_{43a}	BF Setup Time	1.0		ms	7	(22) to RESET falling edge
t_{43b}	BF Hold Time	2.0		CLKs	7	(22) to RESET falling edge
t_{43c}	APICEN Setup Time	2.0		CLKs	7	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	7	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		ns	3	
t_{46}	TCK High Time	25.0		ns	3	@2V, (1)
t_{47}	TCK Low Time	25.0		ns	3	@0.8V, (1)
t_{48}	TCK Fall Time		5.0	ns	3	(2.0V–0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5.0	ns	3	(0.8V–2.0V), (1), (8), (9)
t_{50}	TRST# Pulse Width	40.0		ns	9	(1), Asynchronous
t_{51}	TDI, TMS Setup Time	5.0		ns	8	(7)
t_{52}	TDI, TMS Hold Time	13.0		ns	8	(7)
t_{53}	TDO Valid Delay	3.0	20.0	ns	8	(8)
t_{54}	TDO Float Delay		25.0	ns	8	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	ns	8	(3), (8), (10)

Table 12. Pentium® Processor (610\75) TCP AC Specifications for 50-MHz Bus Operation (Continued)

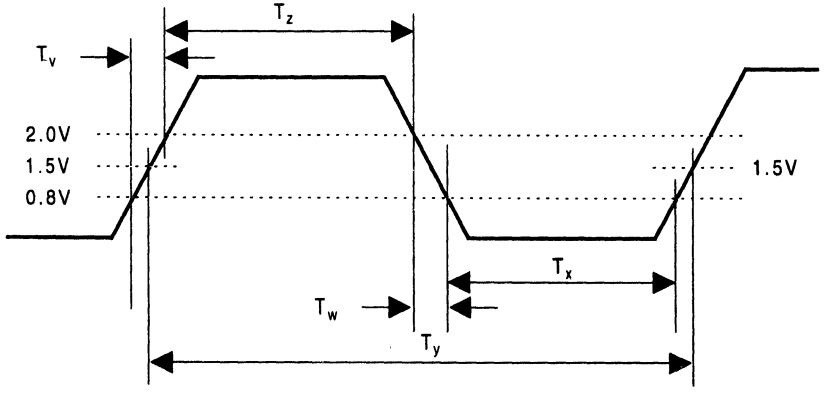
V_{CC} = 3.3V± 5%, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{56}	All Non-Test Outputs Float Delay		25.0	ns	8	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5.0		ns	8	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13.0		ns	8	(3), (7), (10)
APIC AC Specifications						
t_{60a}	PICCLK Frequency	2.0	16.66	MHz		
t_{60b}	PICCLK Period	60.0	500.0	ns	3	
t_{60c}	PICCLK High Time	9.0		ns	3	
t_{60d}	PICCLK Low Time	9.0		ns	3	
t_{60e}	PICCLK Rise Time	1.0	5.0	ns	3	
t_{60f}	PICCLK Fall Time	1.0	5.0	ns	3	
t_{60g}	PICD0-1 Setup Time	3.0		ns	6	to PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		ns	6	to PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	ns	4	from PICCLK, (28)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	ns	4	from PICCLK, (28)

NOTES:
Notes 2, 6, and 14 are general and apply to all standard TTL signals used with the Pentium Processor family.
Notes 11, 18, 19, 20, 23, and 24 do not apply to the TCP package and have been removed in this document.
1. Not 100% tested. Guaranteed by design.
2. TTL input test waveforms are assumed to be 0 to 3V transitions with 1V/ns rise and fall times.
3. Non-test outputs and inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch-free outputs. Glitch-free signals monotonically transition without false transitions (i.e., glitches).
5. 0.8V/ns ≤ CLK input rise/fall time ≤ 8V/ns.
6. 0.3V/ns ≤ input rise/fall time ≤ 5V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 33 MHz.
10. During probe mode operation, do not use the boundary scan timings (t_{55-58}).
12. Setup time is required to guarantee recognition on a specific clock.
13. Hold time is required to guarantee recognition on a specific clock.
14. All TTL timings are referenced from 1.5V.
15. To guarantee proper asynchronous recognition, the signal must have been de-asserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously.
17. When driven asynchronously, RESET, NMI, FLUSH#, R/S#, INIT, and SMI# must be de-asserted (inactive) for a minimum of 2 clocks before being returned active.

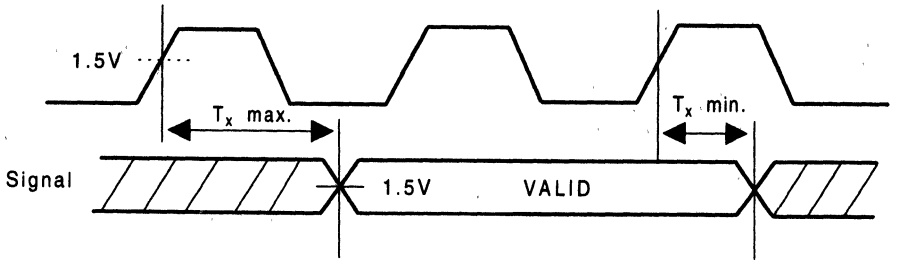
21. The D/C#, M/IO#, W/R#, CACHE#, and A5–A31 signals are sampled only on the CLK that ADS# is active.
22. BF should be strapped to V_{CC} or V_{SS}.
25. These signals are measured on the rising edge of adjacent CLKs at 1.5V. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency. The amount of jitter present must be accounted for as a component of CLK skew between devices.
26. Timing t_{14} is required for external snooping (e.g., address setup to the CLK in which EADS# is sampled active).
27. BUSCHK# is used as a reset configuration signal to select buffer size.
28. This assumes an external pullup resistor to V_{CC} and a lumped capacitive load. The pullup resistor must be between 150 ohms and 1K ohms, the capacitance must be between 20 pF and 240 pF, and the RC product must be between 3 ns and 36 ns.

** Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.



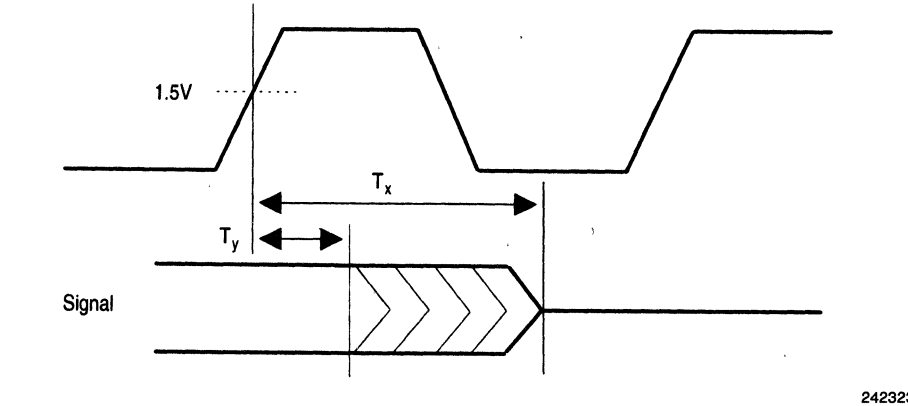
T_v = t5, t49, t60e
T_w = t4, t8,t60f
T_x = t3, t47, t60d
T_y = t1, t45, t60b
T_z = t2, t46, t60c

Figure 3. Clock Waveform



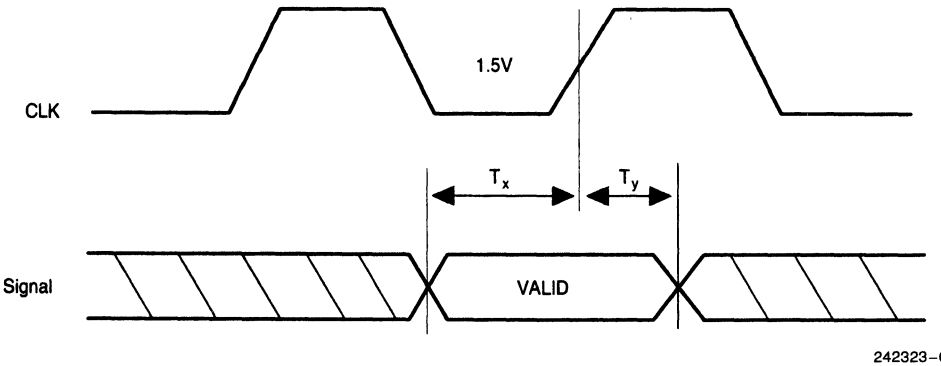
T_x = t6, t8, t9, t10, t11, t12, t60i

Figure 4. Valid Delay Timings



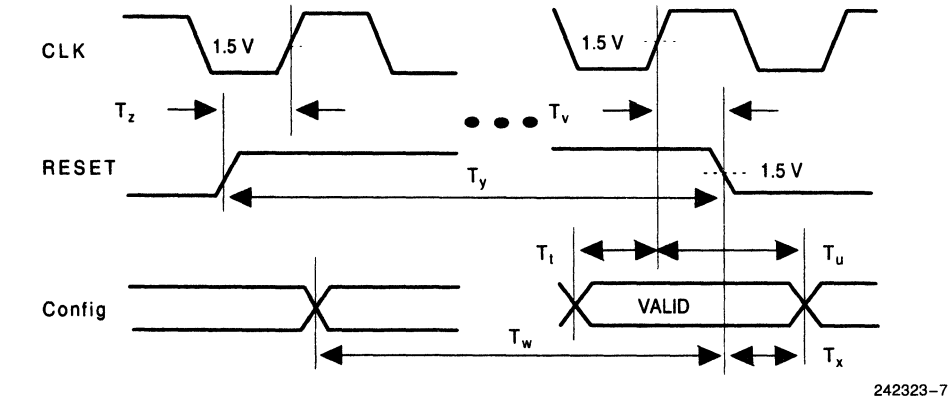
T_x = t7, t13
T_y = t6 min, t12 min

Figure 5. Float Delay Timings



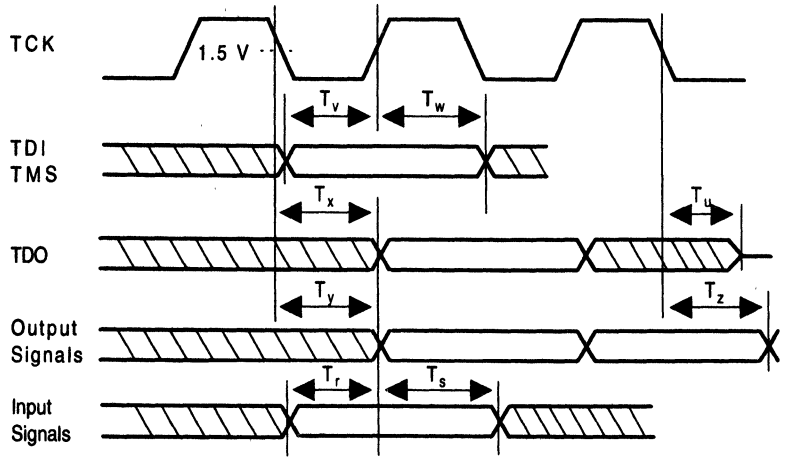
T_x = t14, t16, t18, t20, t22, t24, t26, t28, t31, t34, t60g (to PICCLK)
T_y = t15, t17, t19, t21, t23, t25, t27, t29, t32, t35, t60h (to PICCLK)

Figure 6. Setup and Hold Timings



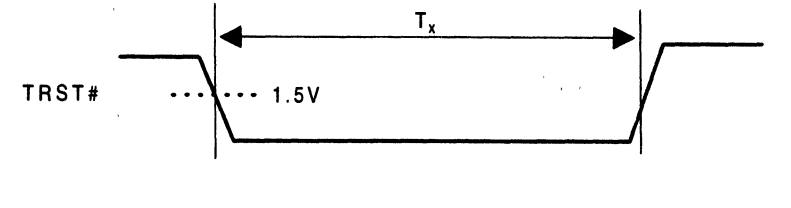
T_t = t40
T_u = t41
T_v = t37
T_w = t42, t43a, t43c
T_x = t43b, t43d
T_y = t38, t39
T_z = t36

Figure 7. Reset and Configuration Timings



T_r = t57
T_s = t58
T_u = t54
T_v = t51
T_w = t52
T_x = t53
T_y = t55
T_z = t56

Figure 8. Test Timings



T_x = t50

Figure 9. Test Reset Timings

4.4 I/O Buffer Models

This section describes the I/O buffer models of the Pentium processor (610\75).

The first order I/O buffer model is a simplified representation of the complex input and output buffers used in the Pentium processor (610\75). Figures 10 and 11 show the structure of the input buffer model and Figure 12 shows the output buffer model. Tables 13 and 14 show the parameters used to specify these models.

Although simplified, these buffer models will accurately model flight time and signal quality. For these parameters, there is very little added accuracy in a complete transistor model.

The following two models represent the input buffer models. The first model, Figure 10, represents all of the input buffers of the Pentium processor (610\75) except for a special group of input buffers. The second model, Figure 11, represents these special buffers. These buffers are the inputs: AHOLD, EADS#, KEN#, WB/WT#, INV, NA#, EWBE#, BOFF#, CLK, and PICCLK.

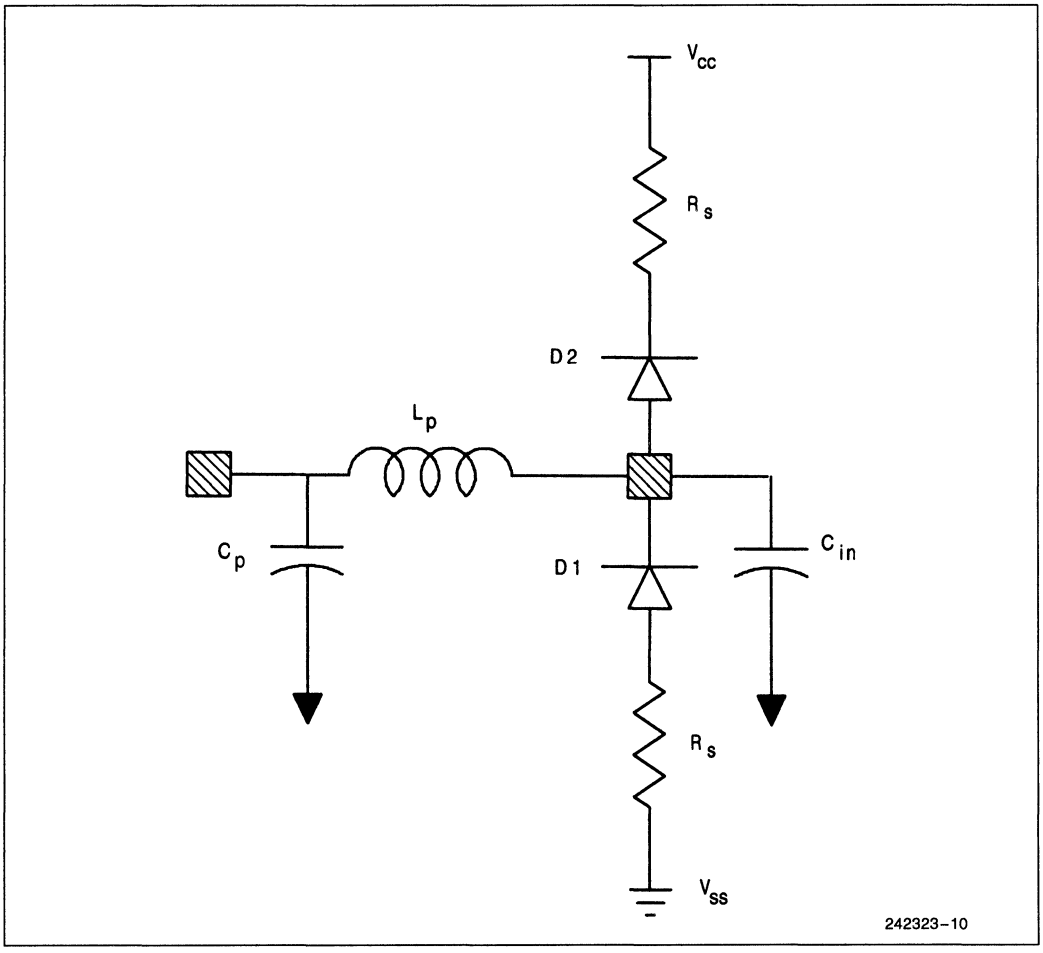


Figure 10. Input Buffer Model, Except Special Group

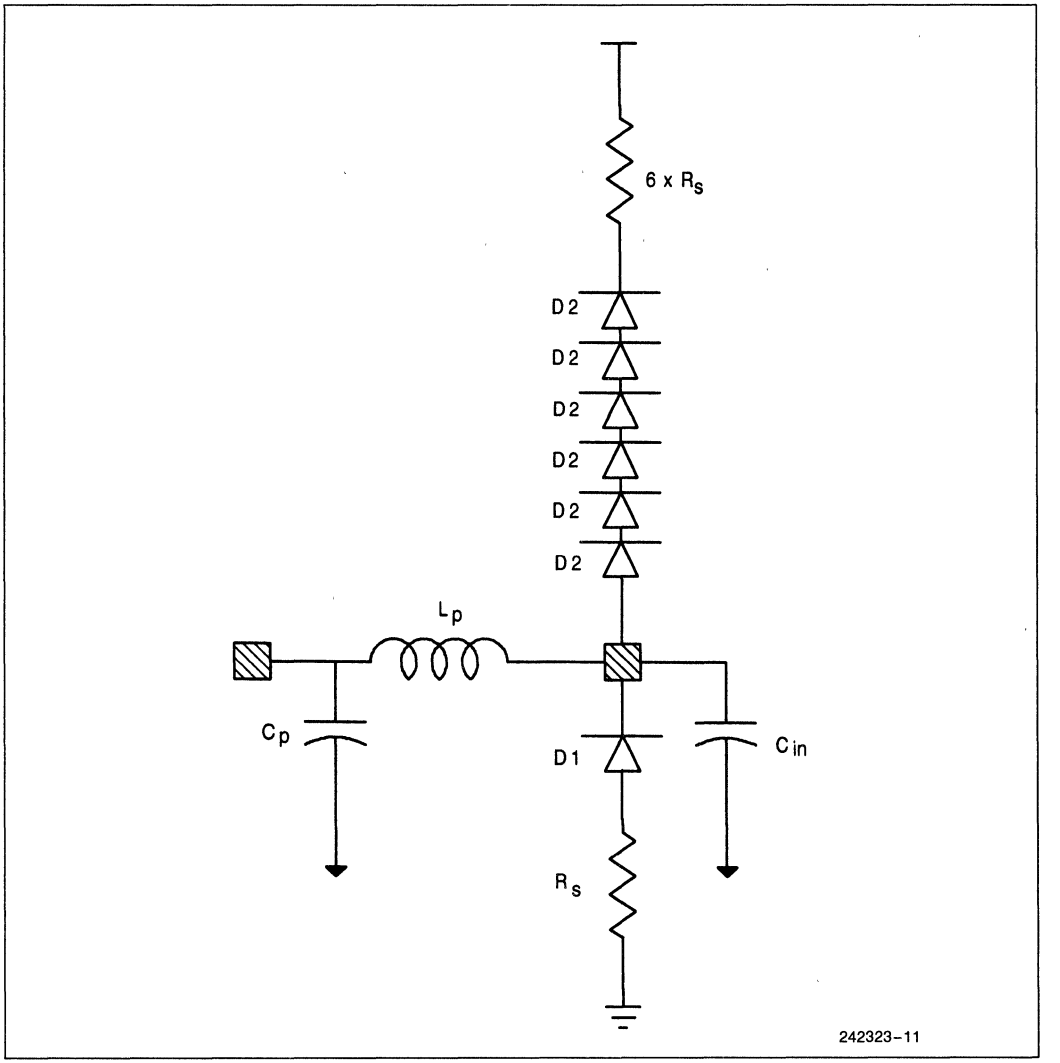


Figure 11. Input Buffer Model for Special Group

Table 13. Parameters Used in the Specification of the First Order Input Buffer Model

Parameter	Description
Cin	Minimum and Maximum value of the capacitance of the input buffer model.
Lp	Minimum and Maximum value of the package inductance.
Cp	Minimum and Maximum value of the package capacitance.
Rs	Diode Series Resistance
D1, D2	Ideal Diodes

Figure 12 shows the structure of the output buffer model. This model is used for all of the output buffers of the Pentium processor (610\75).

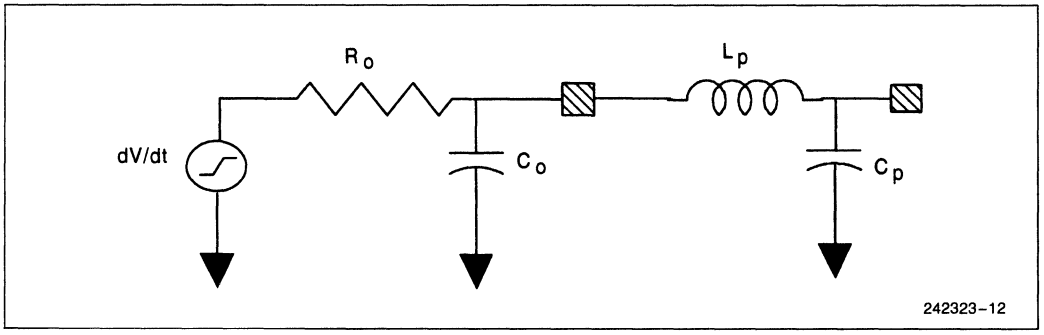


Figure 12. First Order Output Buffer Model

Table 14. Parameters Used in the Specification of the First Order Output Buffer Model

Parameter	Description
dV/dt	Minimum and maximum value of the rate of change of the open circuit voltage source used in the output buffer model.
Ro	Minimum and maximum value of the output impedance of the output buffer model.
Co	Minimum and Maximum value of the capacitance of the output buffer model.
Lp	Minimum and Maximum value of the package inductance.
Cp	Minimum and Maximum value of the package capacitance.

In addition to the input and output buffer parameters, input protection diode models are provided for added accuracy. These diodes have been optimized to provide ESD protection and provide some level of clamping. Although the diodes are not required for simulation, it may be more difficult to meet specifications without them.

Note, however, some signal quality specifications require that the diodes be removed from the input model. The series resistors (Rs) are a part of the diode model. Remove these when removing the diodes from the input model.

4.4.1 BUFFER MODEL PARAMETERS

This section gives the parameters for each Pentium processor (610\75) input, output, and bidirectional signal, as well as the settings for the configurable buffers.

Some pins on the Pentium processor (610\75) have selectable buffer sizes. These pins use the configurable output buffer EB2. Table 15 shows the drive level for BRDY# required at the falling edge of RESET to select the buffer strength. The buffer sizes selected should be the appropriate size required; otherwise AC timings might not be met, or too much overshoot and ringback may occur. There are no other selection choices; all of the configurable buffers get set to the same size at the same time.

Table 15. Buffer Selection Chart

Environment	BRDY#	Buffer Selection
Typical Stand Alone Component	1	EB2
Loaded Component	0	EB2A

NOTES:
For correct buffer selection, the BUSCHK# signal must be held inactive (high) at the falling edge of RESET.
For the Pentium processor (610\75) SPGA version, BRDYC# is used to configure selectable buffer sizes.

Please refer to Table 16 for the groupings of the buffers.

Table 16. Signal to Buffer Type

Signals	Type	Driver Buffer Type	Receiver Buffer Type
CLK	I		ER0
A20M#, AHOLD, BF, BOFF#, BRDY#, BUSCHK#, EADS#, EWBE#, FLUSH#, HOLD, IGNNE#, INIT, INTR, INV, KEN#, NA#, NMI, PEN#, PICCLK, R/S#, RESET, SMI#, STPCLK#, TCK, TDI, TMS, TRST#, WB/WT#	I		ER1
APCHK#, BE[7:5]#, BP[3:2], BREQ, FERR#, IERR#, PCD, PCHK#, PM0/BP0, PM1/BP1, PRDY, PWT, SMIACT#, TDO, U/O#	O	ED1	
A[31:21], AP, BE[4:0]#, CACHE#, D/C#, D[63:0], DP[8:0], HLDA, LOCK#, M/IO#, SCYC	I/O	EB1	EB1
A[20:3], ADS#, HITM#, W/R#	I/O	EB2A	EB2
HIT#	I/O	EB3	EB3
PID0, PICD1	I/O	EB4	EB4

The input, output and bidirectional buffer values are listed in Table 17. This table contains listings for all three types, do not get them confused during simulation. When a bidirectional pin is operating as an input, just use the Cin, Cp and Lp values; if it is operating as a driver, use all of the data parameters.

Table 17. Input, Output and Bidirectional Buffer Model Parameters

Buffer Type	Transition	dV/dt (V/nsec)		Ro (Ohms)		Cp (pF)		Lp (nH)		Co/Cin (pF)	
		min	max	min	max	min	max	min	max	min	max
ER0	Rising					0.3	0.4	3.9	5.0	0.8	1.2
(input)	Falling					0.3	0.4	3.9	5.0	0.8	1.2
ER1	Rising					0.2	0.5	3.1	6.0	0.8	1.2
(input)	Falling					0.2	0.5	3.1	6.0	0.8	1.2
ED1	Rising	3/3.0	3.7/0.9	21.6	53.1	0.3	0.6	3.7	6.6	2.0	2.6
(output)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.3	0.6	3.7	6.6	2.0	2.6
EB1	Rising	3/3.0	3.7/	21.6.9	53.1	0.2	0.5	2.9	6.1	2.0	2.6
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.5	2.9	6.1	2.0	2.6
EB2	Rising	3/3.0	3.7/0.9	21.6	53.1	0.2	0.5	3.1	6.4	9.1	9.7
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.5	3.1	6.4	9.1	9.7
EB2A	Rising	3/2.4	3.7/0.9	10.1	22.4	0.2	0.5	3.1	6.4	9.1	9.7
(bidir)	Falling	3/2.4	3.7/0.9	9.0	21.2	0.2	0.5	3.1	6.4	9.1	9.7
EB3	Rising	3/3.0	3.7/0.9	21.6	53.1	0.2	0.4	3.2	4.1	3.3	3.9
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.4	3.2	4.1	3.3	3.9
EB4	Rising	3/3.0	3.7/0.9	21.6	53.1	0.3	0.4	4.0	4.1	5.0	7.0
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.3	0.4	4.0	4.1	5.0	7.0

Table 18. Input Buffer Model Parameters: D (Diodes)

Symbol	Parameter	D1	D2
IS	Saturation Current	1.4e-14A	2.78e-16A
N	Emission Coefficient	1.19	1.00
RS	Series Resistance	6.5 ohms	6.5 ohms
TT	Transit Time	3 ns	6 ns
VJ	PN Potential	0.983V	0.967V
CJ0	Zero Bias PN Capacitance	0.281 pF	0.365 pF
M	PN Grading Coefficient	0.385	0.376

4.4.2 SIGNAL QUALITY SPECIFICATIONS

Signals driven by the system into the Pentium processor (610\75) must meet signal quality specifications to guarantee that the components read data properly and to ensure that incoming signals do not affect the reliability of the component. There are two signal quality parameters: Ringback and Settling Time.

4.4.2.1 Ringback

Excessive ringback can contribute to long-term reliability degradation of the Pentium processor (610\75), and can cause false signal detection. Ringback is simulated at the input pin of a component using the input buffer model. Ringback can be simulated with or without the diodes that are in the input buffer model.

Ringback is the absolute value of the maximum voltage at the receiving pin below V_{CC} (or above V_{SS}) relative to V_{CC} (or V_{SS}) level after the signal has reached its maximum voltage level. The input diodes are assumed present.

> Maximum Ringback on Inputs = 0.8V (with diodes)

If simulated without the input diodes, follow the Maximum Overshoot/Undershoot specification. By meeting the overshoot/undershoot specification, the signal is guaranteed not to ringback excessively.

If simulated with the diodes present in the input model, follow the maximum ringback specification.

Overshoot (Undershoot) is the absolute value of the maximum voltage above V_{CC} (below V_{SS}). The guideline assumes the absence of diodes on the input.

- Maximum Overshoot/Undershoot on 5V 82497 Cache Controller, and 82492 Cache SRAM Inputs (CLK and PICCLK only) = 1.6V above V_{CC5} (without diodes)
- Maximum Overshoot/Undershoot on 3.3V Pentium processor (610\75) Inputs (not CLK and PICCLK) = 1.4V above V_{CC3} (without diodes)

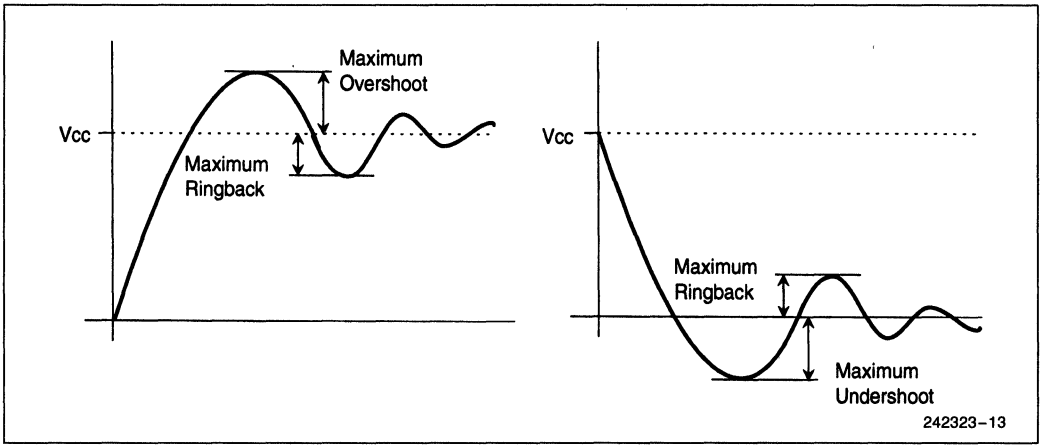


Figure 13. Overshoot/Undershoot and Ringback Guidelines

4.4.2.2 Settling Time

The settling time is defined as the time a signal requires at the receiver to settle within 10% of V_{CC} or V_{SS}. Settling time is the maximum time allowed for a signal to reach within 10% of its final value.

Most available simulation tools are unable to simulate settling time so that it accurately reflects silicon measurements. On a physical board, second-order effects and other effects serve to dampen the signal at the receiver. Because of all these concerns, settling time is a recommendation or a tool for layout tuning and not a specification.

Settling time is simulated at the slow corner, to make sure that there is no impact on the flight times of the signals if the waveform has not settled. Settling time may be simulated with the diodes included or excluded from the input buffer model. If diodes

are included, settling time recommendation will be easier to meet.

Although simulated settling time has not shown good correlation with physical, measured settling time, settling time simulations can still be used as a tool to tune layouts.

Use the following procedure to verify board simulation and tuning with concerns for settling time.

1. Simulate settling time at the slow corner for a particular signal.
2. If settling time violations occur, simulate signal trace with D.C. diodes in place at the receiver pin. The D.C. diode behaves almost identically to the actual (non-linear) diode on the part as long as excessive overshoot does not occur.
3. If settling time violations still occur, simulate flight times for 5 consecutive cycles for that particular signal.
4. If flight time values are consistent over the 5 simulations, settling time should not be a concern. If however, flight times are not consistent over the 5 simulations, tuning of the layout is required.
5. Note that, for signals that are allocated 2 cycles for flight time, the recommended settling time is doubled.

A typical design method would include a settling time that ensures a signal is within 10% of V_{CC} or V_{SS} for at least 2.5 ns prior to the end of the CLK period.

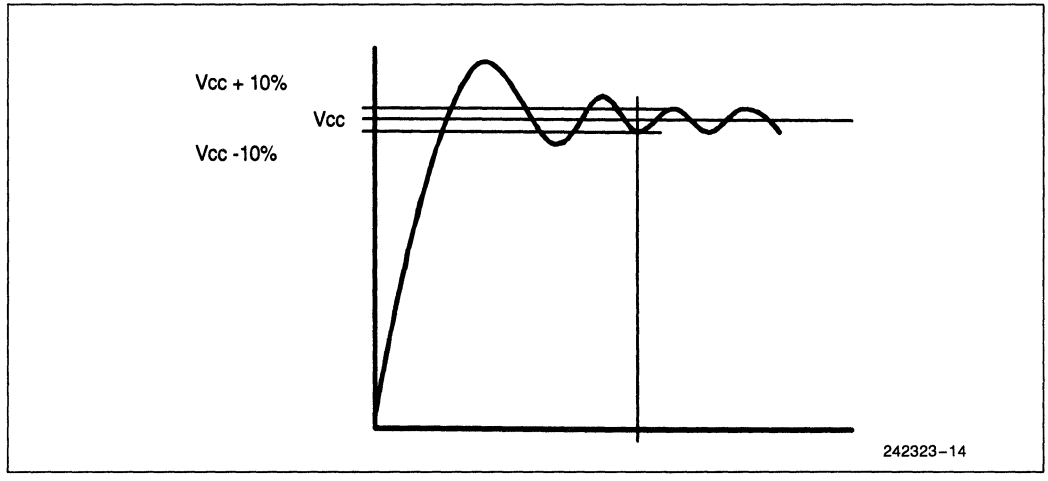


Figure 14. Settling Time

5.0 Pentium® Processor (610\75) TCP MECHANICAL SPECIFICATIONS

Today's portable computers face the challenge of meeting desktop performance in an environment that is constrained by thermal, mechanical, and electrical design considerations. These considerations have driven the development and implementation of Intel's Tape Carrier Package (TCP). The Intel TCP package has been designed to offer a high pin count, low profile, reduced footprint package with uncompromised thermal and electrical performance. Intel continues to provide packaging solutions that meet our rigorous criteria for quality and performance, and this new entry into the Intel package portfolio is no exception.

Key features of the TCP package include: surface mount technology design, lead pitch of 0.25 mm, polyimide body size of 24 mm and polyimide up for pick&place handling. TCP components are shipped with the leads flat in slide carriers, and are designed to be excised and lead formed at the customer manufacturing site. Recommendations for the manufacture of this package are included in the Pentium™ Processor (610\75) *Tape Carrier Package User' s Guide*.

Figure 15 shows a cross-sectional view of the TCP package as mounted on the Printed Circuit Board. Figures 16 and 17 show the TCP as shipped in its slide carrier, and key dimensions of the carrier and package. Figure 18 shows a blow up detail of the package in cross-section. Figure 19 shows an enlarged view of the outer lead bond area of the package.

Tables 19 and 20 provide Pentium processor (610\75) TCP package dimensions.

5.1 TCP Package Mechanical Diagrams

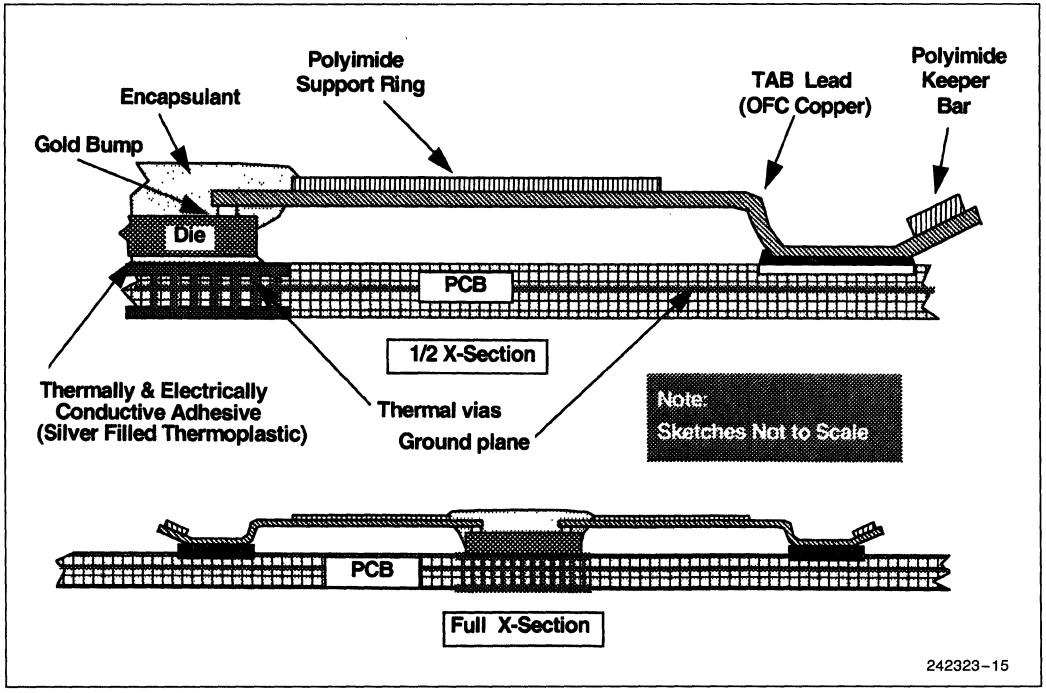


Figure 15. Cross-Sectional View of the Mounted TCP Package

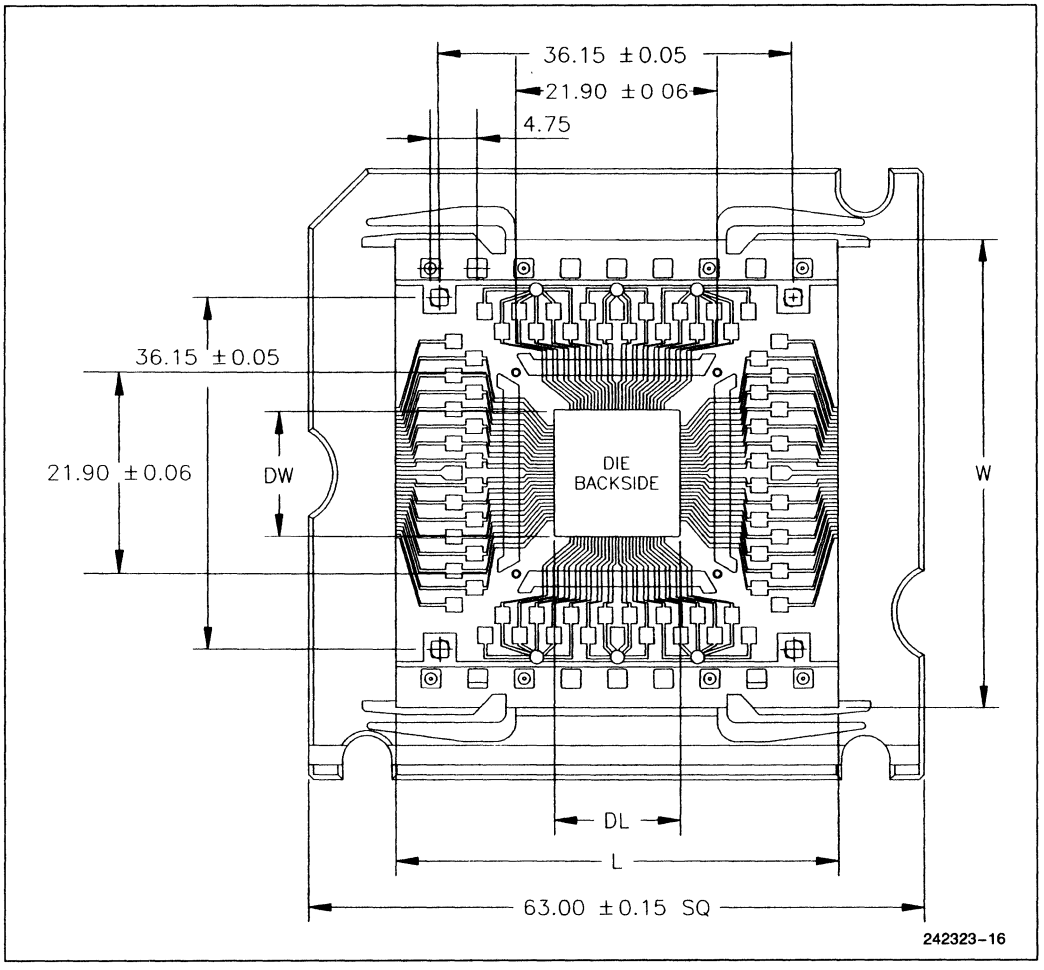


Figure 16. One TCP Site in Carrier (Bottom View of Die)

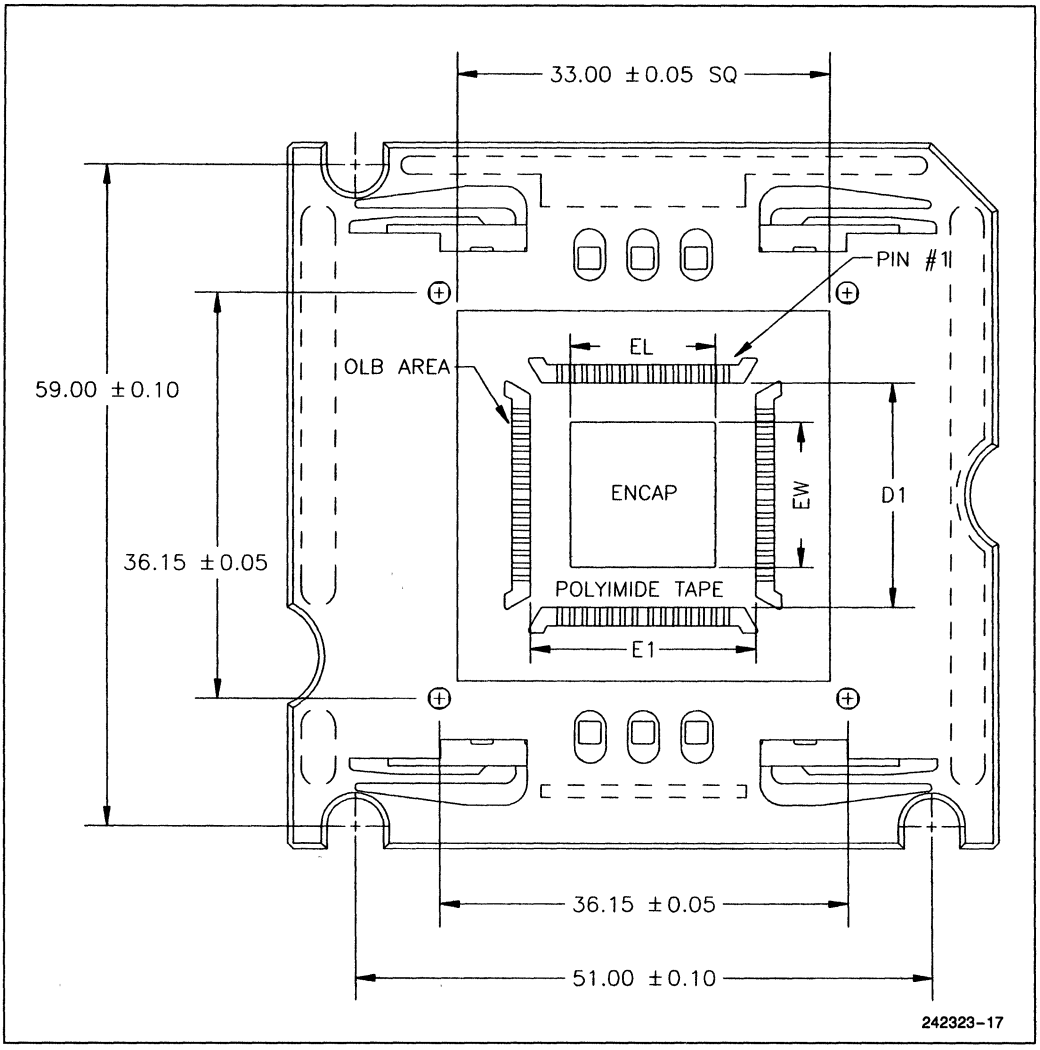


Figure 17. One TCP Site in Carrier (Top View of Die)

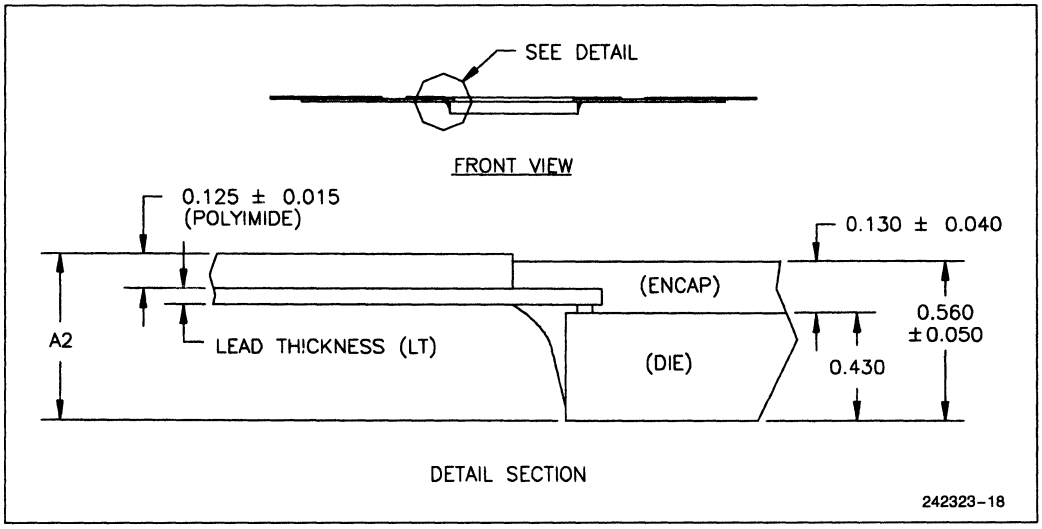


Figure 18. One TCP Site (Cross-Sectional Detail)

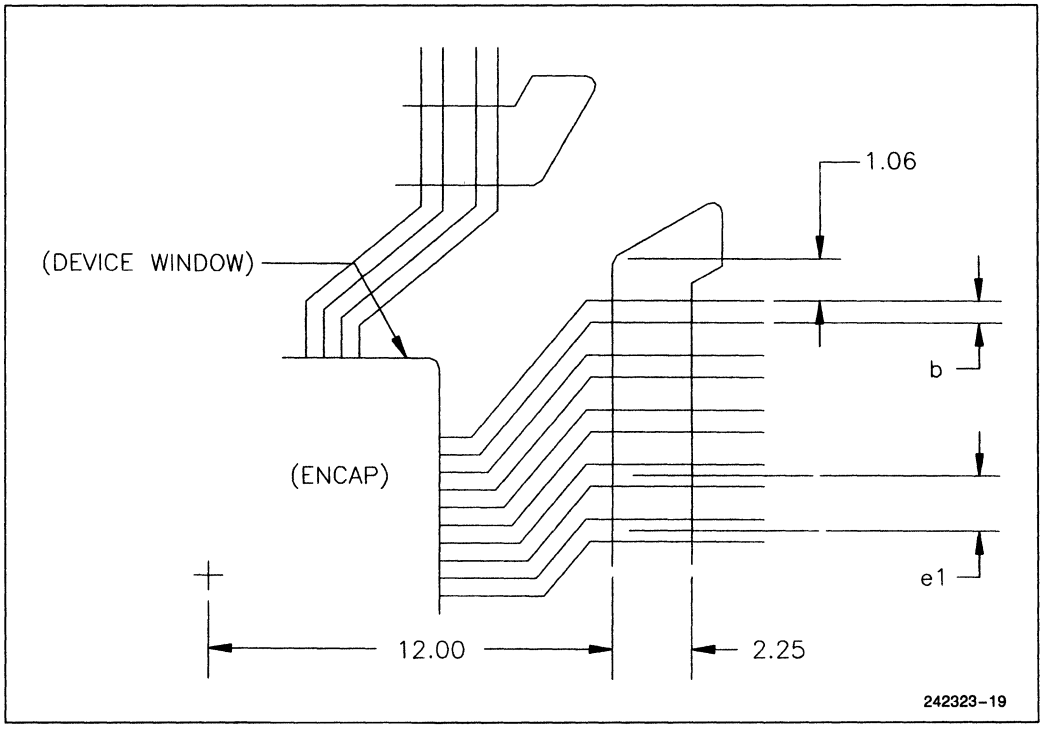


Figure 19. Outer Lead Bond (OLB) Window Detail

Table 19. TCP Key Dimensions

Symbol	Description	Dimension
N	Leadcount	320 leads
W	Tape Width	48.18 ±0.12
L	Site Length	(43.94) reference only
e1	Outer Lead Pitch	0.25 nominal
b	Outer Lead Width	0.10 ±0.01
D1,E1	Package Body Size	24.0 ±0.1
A2	Package Height	75 MHz/90 MHz–0.615 ±0.030 120 MHz–0.605 ±0.030
DL	Die Length	75 MHz/90 MHz–12.769 ±0.015 120 MHz–9.929 ±0.015
DW	Die Width	75 MHz/90 MHz–11.755 ±0.015 120 MHz–9.152 ±0.015
LT	Lead Thickness	75 MHz/90 MHz–0.035 mm 120 MHz–0.025 mm
EL	Encap Length	75 MHz/90 MHz–(13.40 mm) reference only 120 MHz–(10.56 mm) reference only
EW	Encap Width	75 MHz/90 MHz–(12.39 mm) reference only 120 MHz–(9.78 mm) reference only

NOTES:
Dimensions are in millimeters unless otherwise noted.
Dimensions in parentheses are for reference only.

Table 20. Mounted TCP Package Dimensions

Description	Dimension
Package Height	0.75 max.
Terminal Dimension	29.5 nom.
Package Weight	0.5 g max.

NOTE:
Dimensions are in millimeters unless otherwise noted.
Package terminal dimension (lead tip-to-lead tip) assumes the use of a keeper bar.

6.0 Pentium® Processor (610\75) TCP THERMAL SPECIFICATIONS

The Pentium processor (610\75) is specified for proper operation when the case temperature, T_{CASE}, (T_C) is within the specified range of 0°C to 95°C.

6.1 Measuring Thermal Values

To verify that the proper T_C (case temperature) is maintained for the Pentium processor (610\75), it should be measured at the center of the package top surface (encapsulant). To minimize any measurement errors, the following techniques are recommended:

- Use 36 gauge or finer diameter K, T, or J type thermocouples. Intel's laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using highly thermally conductive cements. Intel's laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90° angle as shown in Figure 20.

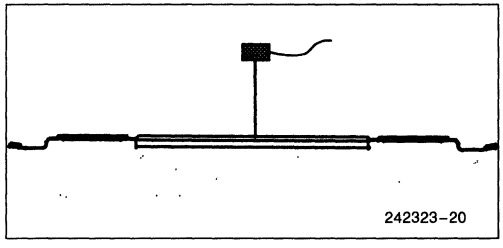


Figure 20. Technique for Measuring Case Temperature (T_C)

6.2 Thermal Equations

For the Pentium processor (610\75), an ambient temperature (T_A) is not specified directly. The only requirement is that the case temperature (T_C) is met. The ambient temperature can be calculated from the following equations:

$$T_J = T_C + P \times \theta_{JC}$$
$$T_A = T_J - P \times \theta_{JA}$$
$$T_A = T_C - (P \times \theta_{CA}$$
$$T_C = T_A + P \times [\theta_{JA} - \theta_{JC}]$$
$$\theta_{CA} = \theta_{JA} - \theta_{JC}$$

where,

T_A and T_C are ambient and case temperatures (°C)
θ_{CA} = Case-to-Ambient thermal resistance (°C/W)
θ_{JA} = Junction-to-Ambient thermal resistance (°C/W)
θ_{JC} = Junction-to-Case thermal resistance (°C/W)
P = maximum power consumption (Watts)

P (maximum power consumption) is specified in section 4.2.

6.3 TCP Thermal Characteristics

The primary heat transfer path from the die of the Tape Carrier Package (TCP) is through the back side of the die and into the PC board. There are two thermal paths traveling from the PC board to the ambient air. One is the spread of heat within the board and the dissipation of heat by the board to the ambient air. The other is the transfer of heat through the board and to the opposite side where thermal enhancements (e.g., heat sinks, pipes) are attached. To prevent the possibility of damaging the TCP component, the thermal enhancements should be attached to the opposite side of the TCP site not directly mounted to the package surface.

6.4 PC Board Enhancements

Copper planes, thermal pads, and vias are design options that can be used to improve heat transfer from the PC board to the ambient air. Tables 21 and 22 present thermal resistance data for copper plane thickness and via effects. It should be noted that although thicker copper planes will reduce the θ_{ca} of a system without any thermal enhancements, they have less effect on the θ_{ca} of a system with thermal enhancements. However, placing vias under the die will reduce the θ_{ca} of a system with and without thermal enhancements.

Table 21. Thermal Resistance vs. Copper Plane Thickness with and without Enhancements

Copper Plane Thickness*	θ_{CA} (°C/W) No Enhancements	θ_{CA} (°C/W) With Heat Pipe
1 oz. Cu	18	8
3 oz. Cu	14	8

NOTES:
*225 vias underneath the die
(1 oz = 1.3 ml)

Table 22. Thermal Resistance vs. Thermal Vias underneath the Die

No. of Vias Under the Die*	θ_{CA} (°C/W) No Enhancements
0	15
144	13

NOTE:
*3 oz. copper planes in test boards

6.4.1 STANDARD TEST BOARD CONFIGURATION

All Tape Carrier Package (TCP) thermal measurements provided in the following tables were taken with the component soldered to a 2″ x 2″ test board outline. This six-layer board contains 225 vias (underneath the die) in the die attach pad which are connected to two 3 oz. copper planes located at layers two and five. For the Pentium processor (610\75) TCP, the vias in the die attach pad should be connected without thermal reliefs to the ground plane(s). The die is attached to the die attach pad using a thermally and electrically conductive adhesive. This test board was designed to optimize the heat spreading into the board and the heat transfer through to the opposite side of the board.

NOTE:
Thermal resistance values should be used as guidelines only, and are highly system dependent. Final system verification should always refer to the case temperature specification.

Table 23. Pentium® Processor (610\75) TCP Package Thermal Resistance without Enhancements

	θ_{JC} (°C/W)	θ_{CA} (°C/W)
Thermal Resistance without Enhancements	0.8	13.9

Table 24. Pentium® Processor (610\75) TCP Package Thermal Resistance with Enhancements (without Airflow)

Thermal Enhancements	θ_{CA} (°C/W)	Notes
Heat sink	11.7	1.2″ ×1.2″ ×0.35
Al Plate	8.7	4″ ×4″ ×0.030″
Al Plate with Heat Pipe	7.8	0.3×1″ ×4″

Table 25. Pentium® Processor (610\75) TCP Package Thermal Resistance with Enhancements (with Airflow)

Thermal Enhancements	θ_{CA} (°C/W)	Notes
Heat sink with Fan @ 1.7 CFM	5.0	1.2″ ×1.2″ ×0.35″ HS 1″ ×1″ ×0.4″ Fan
Heat sink with Airflow @ 400 LFM	5.1	1.2″ ×1.2″ ×0.35″ HS
Heat sink with Airflow @ 600 LFM	4.3	1.2″ ×1.2″ ×0.35″ HS

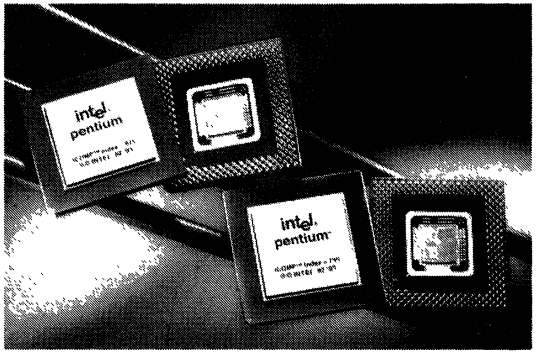
HS = heat sink
LFM = Linear Feet/Minute
CFM = Cubic Feet/Minute

PENTIUM® PROCESSOR at iCOMP® INDEX 610\75 MHz, 735\90 MHz, 815\100 MHz, 1000\120 MHz, and 1110\133 MHz

- **Compatible with Large Software Base**
  - **MS-DOS‡, Windows‡, OS/2‡, UNIX‡**
- **32-Bit CPU with 64-Bit Data Bus**
- **Superscalar Architecture**
  - **Two Pipelined Integer Units Are Capable of 2 Instructions/Clock**
  - **Pipelined Floating Point Unit**
- **Separate Code and Data Caches**
  - **8K Code, 8K Write Back Data**
  - **MESI Cache Protocol**
- **Advanced Design Features**
  - **Branch Prediction**
  - **Virtual Mode Extensions**
- **3.3V BiCMOS Silicon Technology**
- **4M Pages for Increased TLB Hit Rate**
- **IEEE 1149.1 Boundary Scan**
- **Dual Processing Configuration**
- **Multi-Processor Support**
  - **Multiprocessor Instructions**
  - **Support for Second Level Cache**
- **On-Chip Local APIC Controller**
  - **MP Interrupt Management**
  - **8259 Compatible**
- **Internal Error Detection Features**
- **Upgradable with a Future Pentium® OverDrive® Processor**
- **Power Management Features**
  - **System Management Mode**
  - **Clock Control**
- **Fractional Bus Operation**
  - **133-MHz Core/66-MHz Bus**
  - **120-MHz Core/60-MHz Bus**
  - **100-MHz Core/66-MHz Bus**
  - **100-MHz Core/50-MHz Bus**
  - **90-MHz Core/60-MHz Bus**
  - **75-MHz Core/50-MHz Bus**

The Pentium® processor 75/90/100/120/133 extends the Pentium processor family, providing performance needed for mainstream desktop applications as well as for workstations and servers. The Pentium processor is compatible with the entire installed base of applications for DOS, Windows, OS/2, and UNIX. The Pentium processor 75/90/100/120/133 superscalar architecture can execute two instructions per clock cycle. Branch prediction and separate caches also increase performance. The pipelined floating point unit delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor 75/90/100/120/133 has 3.3 million transistors and is built on Intel's advanced 3.3V BiCMOS silicon technology. The Pentium processor 75/90/100/120/133 has on-chip dual processing support, a local multiprocessor interrupt controller, and SL power management features.

The Pentium processor may contain design defects or errors known as errata. Current characterized errata are available upon request.



241997-17

‡Other brands and trademarks are the property of their respective owners.
†Since publication of documents referenced in this document, registration of the Pentium, OverDrive, and iCOMP trademarks has been issued to Intel Corporation.

November 1995
Order Number: 241997-006

PENTIUM® PROCESSOR at iCOMP® INDEX 610\75 MHz, 735\90 MHz, 815\100 MHz, 1000\120 MHz, and 1110\133 MHz

CONTENTS

CONTENTS

1.0 MICROPROCESSOR ARCHITECTURE OVERVIEW

The Pentium® processor at iCOMP® rating 610\75 MHz, iCOMP rating 735\90 MHz, and iCOMP rating 815\100 MHz, iCOMP rating 1000\120, and iCOMP rating 1110\133 extends the Intel Pentium family of microprocessors. It is 100% binary compatible with the 8086/88, 80286, Intel386™ DX CPU, Intel386 SX CPU, Intel486™ DX CPU, Intel486 SX CPU, Intel486 DX2 CPUs, and Pentium processor at iCOMP Index 510\60 MHz and iCOMP Index 567\66 MHz.

The Pentium processor family consists of the Pentium processor at iCOMP rating 610\75 MHz, iCOMP rating 735\90 MHz, iCOMP rating 815\100 MHz, iCOMP rating 1000\120, and iCOMP rating 1110\133 (product order code 80502), described in this document, and the original Pentium processor 60/66 (order code 80501). The name "Pentium processor 75/90/100/120/133" will be used in this document to refer to the Pentium processor at iCOMP rating 610\75 MHz, iCOMP rating 735\90 MHz, iCOMP rating 815\100 MHz, iCOMP rating 1000\120 MHz, and iCOMP rating 1110\133 MHz. Also, the name "Pentium processor 60/66" will be used to refer to the original 60- and 66-MHz version product.

The Pentium processor family architecture contains all of the features of the Intel486 CPU family, and provides significant enhancements and additions including the following:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Improved Instruction Execution Time
- Separate 8K Code and 8K Data Caches
- Writeback MESI Protocol in the Data Cache
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Functional Redundancy Checking
- Execution Tracing
- Performance Monitoring
- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions

In addition to the features listed above, the Pentium processor 75/90/100/120/133 offers the following enhancements over the Pentium processor 60/66:

- iCOMP performance rating of 1110 at 133 MHz in single processor configuration
- iCOMP performance rating of 1000 at 120 MHz in single processor configuration
- iCOMP performance rating of 815 at 100 MHz in single processor configuration
- iCOMP performance rating of 735 at 90 MHz in single processor configuration
- iCOMP performance rating of 610 at 75 MHz in single processor configuration
- Dual processing support
- SL power management features
- Fractional bus operation
- On-chip local APIC device

1.1 Pentium® Processor Family Architecture

The application instruction set of the Pentium processor family includes the complete Intel486 CPU family instruction set with extensions to accommodate some of the additional functionality of the Pentium processors. All application software written for the Intel386 and Intel486 family microprocessors will run on the Pentium processors without modification. The on-chip memory management unit (MMU) is completely compatible with the Intel386 family and Intel486 family of CPUs.

The Pentium processors implement several enhancements to increase performance. The two instruction pipelines and floating-point unit on Pentium processors are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, two floating-point instructions) in one clock.

Branch prediction is implemented in the Pentium processors. To support this, Pentium processors implement two prefetch buffers, one to prefetch code in a linear fashion, and one that prefetches code according to the BTB so the needed code is almost always prefetched before it is needed for execution.

The floating-point unit has been completely redesigned over the Intel486 CPU. Faster algorithms provide up to 10X speed-up for common operations including add, multiply, and load.

Pentium processors include separate code and data caches integrated on-chip to meet performance goals. Each cache is 8 Kbytes in size, with a 32-byte line size and is 2-way set associative. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses. The data cache is configurable to be write back or write through on a line-by-line basis and follows the MESI protocol. The data cache tags are triple ported to support two data transfers and an inquire cycle in the same clock. The code cache is an inherently write-protected cache. The code cache tags are also triple ported to support snooping and split line accesses. Individual pages can be configured as cacheable or non-cacheable by software or hardware. The caches can be enabled or disabled by software or hardware.

The Pentium processors have increased the data bus to 64 bits to improve the data transfer rate. Burst read and burst write back cycles are supported by the Pentium processors. In addition, bus cycle pipelining has been added to allow two bus cycles to be in progress simultaneously. The Pentium processors' Memory Management Unit contains optional extensions to the architecture which allow 2-Mbyte and 4-Mbyte page sizes.

The Pentium processors have added significant data integrity and error detection capability. Data parity checking is still supported on a byte-by-byte basis. Address parity checking, and internal parity checking features have been added along with a new exception, the machine check exception. In addition, the Pentium processors have implemented functional redundancy checking to provide maximum error detection of the processor and the interface to the processor. When functional redundancy checking is used, a second processor, the "checker" is used to execute in lock step with the "master" processor. The checker samples the master's outputs and compares those values with the values it computes internally, and asserts an error signal if a mismatch occurs.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this, the Pentium processors have increased test and debug capability. The Pentium processors implement IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processors have specified 4 breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match. Execution tracing provides external indications when an instruction has completed execution in either of the two internal pipelines, or when a branch has been taken.

System Management Mode (SMM) has been implemented along with some extensions to the SMM architecture. Enhancements to the virtual 8086 mode have been made to increase performance by reducing the number of times it is necessary to trap to a virtual 8086 monitor.

Figure 1 shows a block diagram of the Pentium processor 75/90/100/120/133.

For Pentium Processor (610\75) designs which use the TCP package, Intel document 242323 must be referenced for correct TCP pinout, mechanical, thermal, and AC specifications.

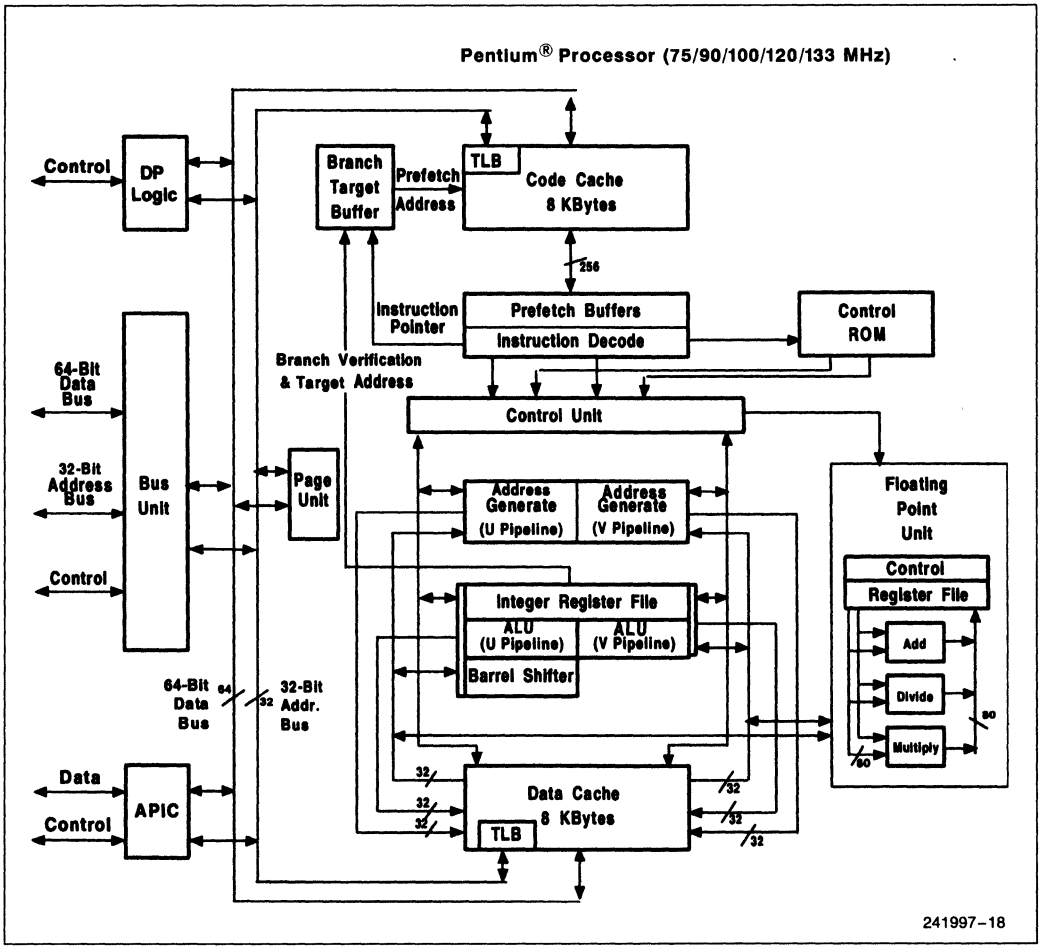


Figure 1. Pentium® Processor Block Diagram

The block diagram shows the two instruction pipelines, the "u" pipe and the "v" pipe. The u-pipe can execute all integer and floating point instructions. The v-pipe can execute simple integer instructions and the FXCH floating-point instructions.

The separate caches are shown, the code cache and data cache. The data cache has two ports, one for each of the two pipes (the tags are triple ported to allow simultaneous inquire cycles). The data cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium processor. Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so the Pentium processors can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that

must be performed to implement the Pentium processor architecture. The control ROM unit has direct control over both pipelines.

The Pentium processors contain a pipelined floating-point unit that provides a significant floating-point performance advantage over previous generations of processors.

The architectural features introduced in this chapter are more fully described in the *Pentium® Processor Family Developer's Manual.*

1.2 Pentium® Processor 75/90/100/120/133

In addition to the architecture described above for the Pentium processor family, the Pentium processor 75/90/100/120/133 has additional features which are described in this section.

The Pentium processor 75/90/100/120/133 offers higher performance and higher operating frequencies than the Pentium processor 60/66.

Pentium® Processor 75/90/100/120/133 Core Frequency	External Bus Interface	iCOMP® Index
133 MHz	66 MHz	1110
120 MHz	60 MHz	1000
100 MHz	66/50 MHz	815
90 MHz	60 MHz	735
75 MHz	50 MHz	610

Symmetric dual processing in a system is supported with two Pentium processors 75/90/100/120/133. The two processors appear to the system as a single Pentium processor 75/90/100/120/133. Operating systems with dual processing support properly schedule computing tasks between the two processors. This scheduling of tasks is transparent to software applications and the end-user. Logic built into the processors support a "glueless" interface for easy system design. Through a private bus, the two Pentium processors 75/90/100/120/133 arbitrate for the external bus and maintain cache coherency. **Dual processing is supported in a system only if both processors are operating at identical core and bus frequencies.**

In this document, in order to distinguish between two Pentium processors 75/90/100/120/133 in dual processing mode, one CPU will be designated as the Primary processor with the other being the Dual processor. Note that this is a different concept than that of "master" and "checker" processors described above in the discussion on functional redundancy.

Due to the advanced 3.3V BiCMOS process that it is produced on, the Pentium processor 75/90/100/120/133 dissipates less power than the Pentium processor 60/66. In addition to the SMM features described above, the Pentium processor 75/90/100/120/133 supports clock control. When the clock to the Pentium processor 75/90/100/120/133 is stopped, power dissipation is virtually eliminated. The combination of these improvements makes the Pentium processor 75/90/100/120/133 a good choice for energy-efficient desktop designs.

Supporting an upgrade socket (Socket 5) in the system will provide end-user upgradability by the addition of a future Pentium OverDrive processor. Typical applications will realize a 40%–70% performance increase by addition of a future Pentium OverDrive processor.

Socket 7 has been defined as the upgrade socket for the Pentium processor (1110/133) in addition to the Pentium processors 75/90/100/120. The flexibility of the Socket 7 definition makes it backward compatible with Socket 5 and should be used for all new Pentium processor-based system designs.

The Pentium processor 75/90/100/120/133 supports fractional bus operation. This allows the internal processor core to operate at high frequencies, while communicating with the external bus at lower frequencies. The external bus frequency operates at a selectable one-half or two-thirds fraction of the internal core frequency.

The Pentium processor 75/90/100/120/133 contains an on-chip Advanced Programmable Interrupt Controller (APIC). This APIC implementation supports multiprocessor interrupt management (with symmetric interrupt distribution across all processors), multiple I/O subsystem support, 8259A compatibility, and inter-processor interrupt support.

2.0 PINOUT

2.1 Pinout and Pin Descriptions

2.1.1 PENTIUM® PROCESSOR 75/90/100/120/133 PINOUT

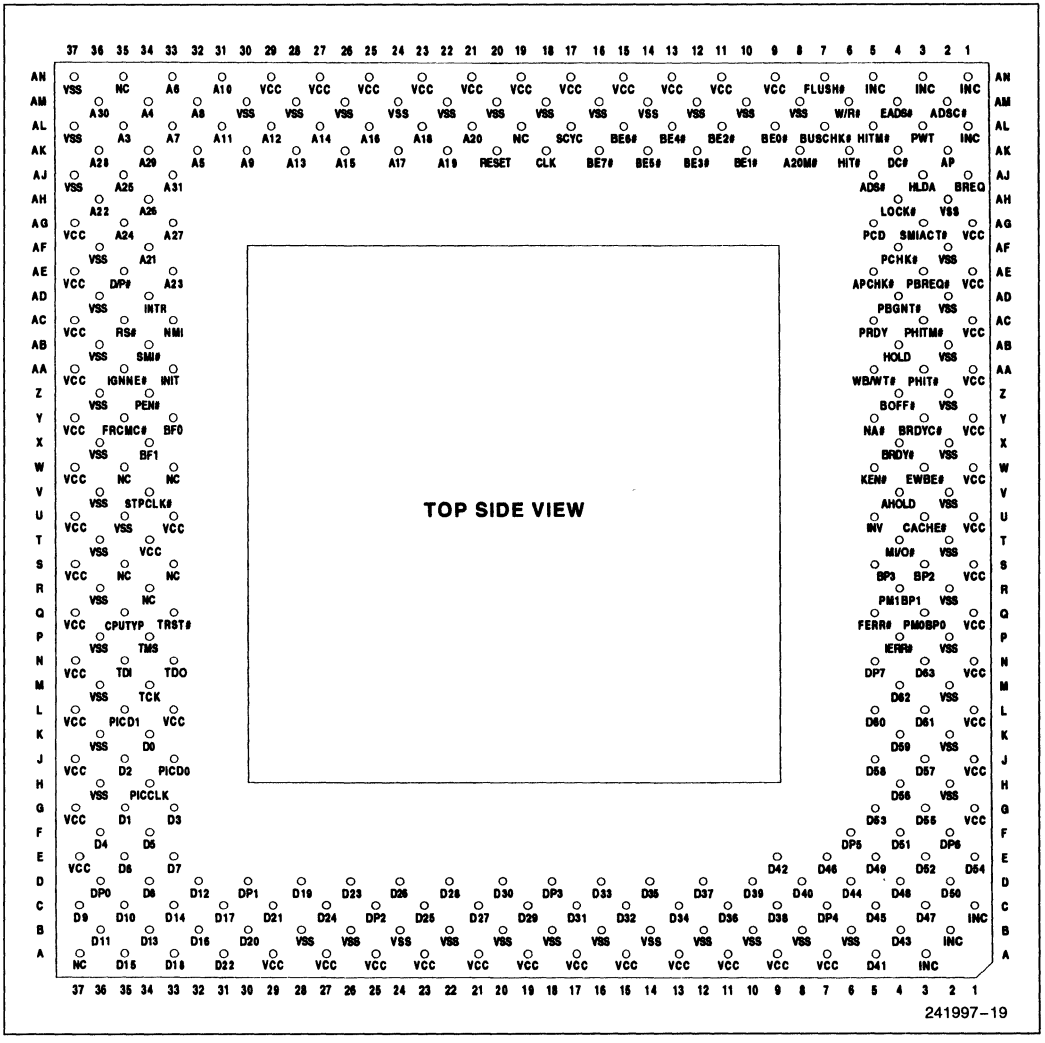


Figure 2. Pentium® Processor 75/90/100/120/133 Pinout (Top Side View)

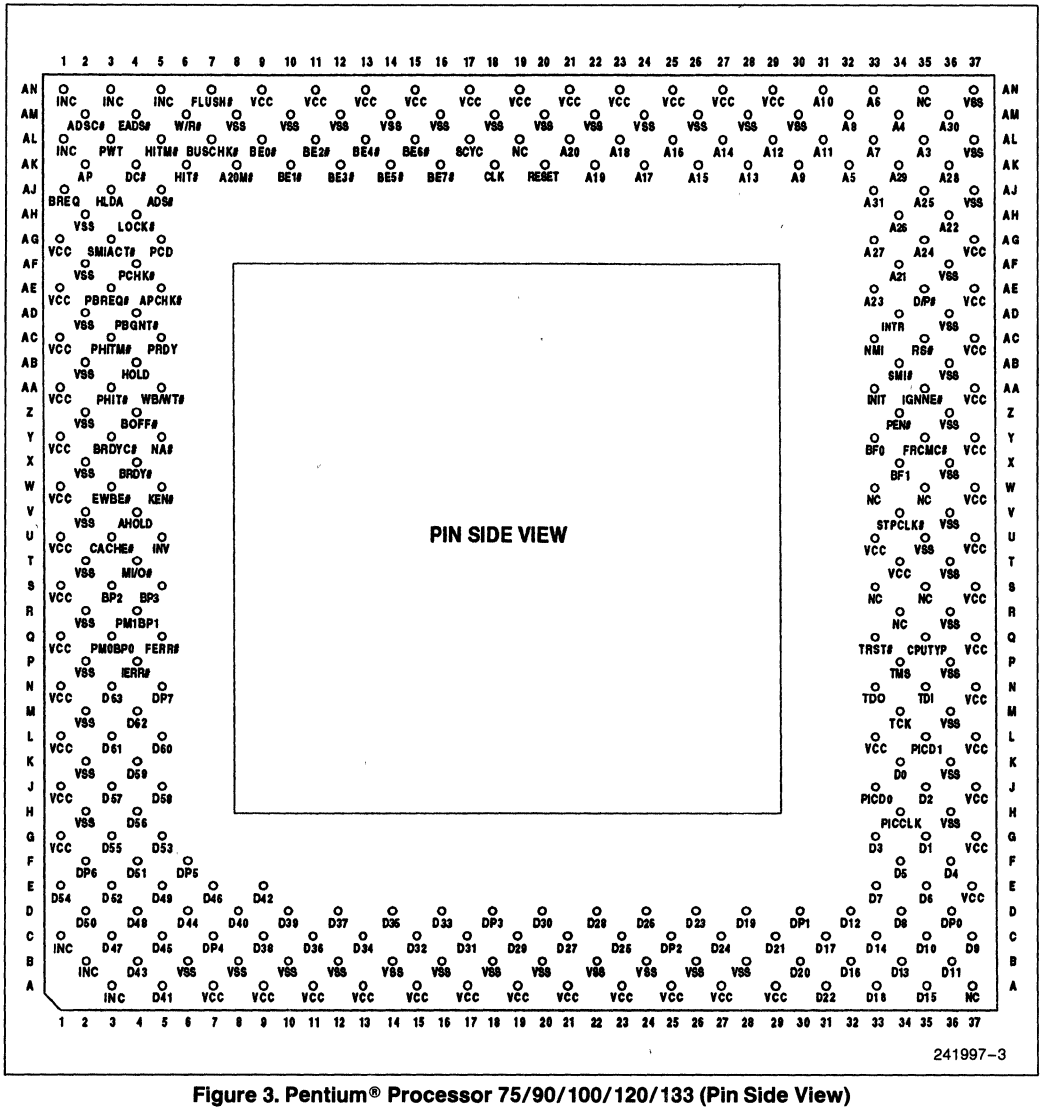


Figure 3. Pentium® Processor 75/90/100/120/133 (Pin Side View)

2.1.2 PIN CROSS REFERENCE TABLE FOR PENTIUM® PROCESSOR 75/90/100/120/133

Table 1. Pin Cross Reference by Pin Name

Address									
A3	AL35	A9	AK30	A15	AK26	A21	AF34	A27	AG33
A4	AM34	A10	AN31	A16	AL25	A22	AH36	A28	AK36
A5	AK32	A11	AL31	A17	AK24	A23	AE33	A29	AK34
A6	AN33	A12	AL29	A18	AL23	A24	AG35	A30	AM36
A7	AL33	A13	AK28	A19	AK22	A25	AJ35	A31	AJ33
A8	AM32	A14	AL27	A20	AL21	A26	AH34		
Data									
D0	K34	D13	B34	D26	D24	D39	D10	D52	E03
D1	G35	D14	C33	D27	C21	D40	D08	D53	G05
D2	J35	D15	A35	D28	D22	D41	A05	D54	E01
D3	G33	D16	B32	D29	C19	D42	E09	D55	G03
D4	F36	D17	C31	D30	D20	D43	B04	D56	H04
D5	F34	D18	A33	D31	C17	D44	D06	D57	J03
D6	E35	D19	D28	D32	C15	D45	C05	D58	J05
D7	E33	D20	B30	D33	D16	D46	E07	D59	K04
D8	D34	D21	C29	D34	C13	D47	C03	D60	L05
D9	C37	D22	A31	D35	D14	D48	D04	D61	L03
D10	C35	D23	D26	D36	C11	D49	E05	D62	M04
D11	B36	D24	C27	D37	D12	D50	D02	D63	N03
D12	D32	D25	C23	D38	C09	D51	F04		

Table 1. Pin Cross Reference by Pin Name (Contd.)

Control							
A20M#	AK08	BRDYC#	Y03	FLUSH#	AN07	PEN#	Z34
ADS#	AJ05	BREQ	AJ01	FRCMC#	Y35	PM0/BP0	Q03
ADSC#	AM02	BUSCHK#	AL07	HIT#	AK06	PM1/BP1	R04
AHOLD	V04	CACHE#	U03	HITM#	AL05	PRDY	AC05
AP	AK02	CPUTYP	Q35	HLDA	AJ03	PWT	AL03
APCHK#	AE05	D/C#	AK04	HOLD	AB04	R/S#	AC35
BE0#	AL09	D/P#	AE35	IERR#	P04	RESET	AK20
BE1#	AK10	DP0	D36	IGNNE#	AA35	SCYC	AL17
BE2#	AL11	DP1	D30	INIT	AA33	SMI#	AB34
BE3#	AK12	DP2	C25	INTR/LINT0	AD34	SMIACT#	AG03
BE4#	AL13	DP3	D18	INV	U05	TCK	M34
BE5#	AK14	DP4	C07	KEN#	W05	TDI	N35
BE6#	AL15	DP5	F06	LOCK#	AH04	TDO	N33
BE7#	AK16	DP6	F02	M/IO#	T04	TMS	P34
BOFF#	Z04	DP7	N05	NA#	Y05	TRST#	Q33
BP2	S03	EADS#	AM04	NMI/LINT1	AC33	W/R#	AM06
BP3	S05	EWBE#	W03	PCD	AG05	WB/WT#	AA05
BRDY#	X04	FERR#	Q05	PCHK#	AF04		

APIC		Clock Control		Dual Processor Private Interface	
PICCLK	H34	CLK	AK18	PBGNT#	AD04
PICD0	J33	BF0	Y33	PBREQ#	AE03
[DPEN#]		BF1	X34	PHIT#	AA03
PICD1	L35	STPCLK#	V34	PHITM#	AC03
[APICEN]					

Table 1. Pin Cross Reference by Pin Name (Contd.)

V_{CC}								
A07	A19	E37	L33	S01	W01	AC01	AN09	AN21
A09	A21	G01	L37	S37	W37	AC37	AN11	AN23
A11	A23	G37	N01	T34	Y01	AE01	AN13	AN25
A13	A25	J01	N37	U01	Y37	AE37	AN15	AN27
A15	A27	J37	Q01	U33	AA01	AG01	AN17	AN29
A17	A29	L01	Q37	U37	AA37	AG37	AN19	
V_{SS}								
B06	B22	M02	U35	AB36	AM08	AM24		
B08	B24	M36	V02	AD02	AM10	AM26		
B10	B26	P02	V36	AD36	AM12	AM28		
B12	B28	P36	X02	AF02	AM14	AM30		
B14	H02	R02	X36	AF36	AM16	AN37		
B16	H36	R36	Z02	AH02	AM18			
B18	K02	T02	Z36	AJ37	AM20			
B20	K36	T36	AB02	AL37	AM22			
NC/INC								
A03	C01	S35	W35	AL01	AN01	AN05		
A37	R34	W33		AL19	AN03	AN35		
B02	S33							

2.2 Design Notes

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active HIGH inputs should be connected to GND.

No Connect (NC) pins must remain unconnected. Connection of NC pins may result in component failure or incompatibility with processor steppings.

2.3 Quick Pin Reference

This section gives a brief functional description of each of the pins. For a detailed description, see the "Hardware Interface" chapter in the *Pentium®† Family User's Manual,* Volume 1. **Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior.**

The # symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage. When a # symbol is not present after the signal name, the signal is active, or asserted at the high voltage level.

The following pins exist on the Pentium processor 60/66 but have been removed from the Pentium processor 75/90/100/120/133:

- IBT, IU, IV, BT0-3

The following pins become I/O pins when two Pentium processors 75/90/100/120/133 are operating in a dual processing environment:

- ADS#, CACHE#, HIT#, HITM#, HLDA#, LOCK#, M/IO#, D/C#, W/R#, SCYC

Table 2. Quick Pin Reference

Symbol	Type*	Name and Function
A20M#	I	When the **address bit 20 mask** pin is asserted, the Pentium® processor 75/90/100/120/133 emulates the address wraparound at 1 Mbyte which occurs on the 8086. When A20M# is asserted, the Pentium processor 75/90/100/120/133 masks physical address bit 20 (A20) before performing a lookup to the internal caches or driving a memory cycle on the bus. The effect of A20M# is undefined in protected mode. A20M# must be asserted only when the processor is in real mode. A20M# is internally masked by the Pentium processor 75/90/100/120/133 when configured as a Dual processor.
A31-A3	I/O	As outputs, the **address** lines of the processor along with the byte enables define the physical area of memory or I/O accessed. The external system drives the inquire address to the processor on A31-A5.
ADS#	O	The **address status** indicates that a new valid bus cycle is currently being driven by the Pentium processor 75/90/100/120/133.
ADSC#	O	ADSC# is functionally identical to ADS#.
AHOLD	I	In response to the assertion of **address hold,** the Pentium processor 75/90/100/120/133 will stop driving the address lines (A31-A3), and AP in the next clock. The rest of the bus will remain active so data can be returned or driven for previously issued bus cycles.
AP	I/O	**Address parity** is driven by the Pentium processor 75/90/100/120/133 with even parity information on all Pentium processor 75/90/100/120/133 generated cycles in the same clock that the address is driven. Even parity must be driven back to the Pentium processor 75/90/100/120/133 during inquire cycles on this pin in the same clock as EADS# to ensure that correct parity check status is indicated by the Pentium processor 75/90/100/120/133.
APCHK#	O	The **address parity check** status pin is asserted two clocks after EADS# is sampled active if the Pentium processor 75/90/100/120/133 has detected a parity error on the address bus during inquire cycles. APCHK# will remain active for one clock each time a parity error is detected (including during dual processing private snooping).
[APICEN] PICD1	I	**Advanced Programmable Interrupt Controller Enable** is a new pin that enables or disables the on-chip APIC interrupt controller. If sampled high at the falling edge of RESET, the APIC is enabled. APICEN shares a pin with the **Programmable Interrupt Controller Data** 1 signal.
BE7#-BE5# BE4#-BE0#	O I/O	The **byte enable** pins are used to determine which bytes must be written to external memory, or which bytes were requested by the CPU for the current cycle. The byte enables are driven in the same clock as the address lines (A31-3). Unlike the Pentium processor 60/66, the lower 4-byte enables (BE3#-BE0#) are used on the Pentium processor 75/90/100/120/133 as APIC ID inputs and are sampled at RESET. After RESET, these behave exactly like the Pentium processor 60/66 byte enables. In dual processing mode, BE4# is used as an input during Flush cycles.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
BF[1:0]	I	**Bus Frequency** determines the bus-to-core frequency ratio. BF is sampled at RESET, and cannot be changed until another non-warm (1 ms) assertion of RESET. Additionally, BF must not change values while RESET is active. For proper operation of the Pentium processor 75/90/100/120/133 this pin should be strapped high or low. When BF is strapped to V_{CC}, the processor will operate at a 2/3 bus/core frequency ratio. When BF is strapped to V_{SS}, the processor will operate at a 1/2 bus/core frequency ratio. If BF is left floating, the Pentium processor 75/90/100/120/133 defaults to a 2/3 bus ratio. Note that core operation at either 75 MHz or 90 MHz does not allow 1/2 bus/core frequency, while core operation at 120 MHz and 133 MHz does not allow 2/3 bus core frequency.
BOFF#	I	The **backoff** input is used to abort all outstanding bus cycles that have not yet completed. In response to BOFF#, the Pentium processor 75/90/100/120/133 will float all pins normally floated during bus hold in the next clock. The processor remains in bus hold until BOFF# is negated, at which time the Pentium processor 75/90/100/120/133 restarts the aborted bus cycle(s) in their entirety.
BP[3:2] PM/BP[1:0]	O	The **breakpoint** pins (BP3-0) correspond to the debug registers, DR3-DR0. These pins externally indicate a breakpoint match when the debug registers are programmed to test for breakpoint matches. BP1 and BP0 are multiplexed with the **performance monitoring** pins (PM1 and PM0). The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
BRDY#	I	The **burst ready** input indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted the Pentium processor 75/90/100/120/133 data in response to a write request. This signal is sampled in the T2, T12 and T2P bus states.
BRDYC#	I	This signal has the same functionality as BRDY#.
BREQ	O	The **bus request** output indicates to the external system that the Pentium processor 75/90/100/120/133 has internally generated a bus request. This signal is always driven whether or not the Pentium processor 75/90/100/120/133 is driving its bus.
BUSCHK#	I	The **bus check** input allows the system to signal an unsuccessful completion of a bus cycle. If this pin is sampled active, the Pentium processor 75/90/100/120/133 will latch the address and control signals in the machine check registers. If, in addition, the MCE bit in CR4 is set, the Pentium processor 75/90/100/120/133 will vector to the machine check exception. **NOTE:** To assure that the BUSCHK# will always be recognized, STPCLK# must be deasserted any time BUSCHK# is asserted by the system, before the system allows another external bus cycle. If BUSCHK# is asserted by the system for a snoop cycle while STPCLK# remains asserted, usually (if MCE = 1) the processor will vector to the exception after STPCLK# is deasserted. But if another snoop to the same line occurs during STPCLK# assertion, the processor can lose the BUSCHK# request.
CACHE#	O	For Pentium processor 75/90/100/120/133-initiated cycles the **cache** pin indicates internal cacheability of the cycle (if a read), and indicates a burst write back cycle (if a write). If this pin is driven inactive during a read cycle, the Pentium processor 75/90/100/120/133 will not cache the returned data, regardless of the state of the KEN# pin. This pin is also used to determine the cycle length (number of transfers in the cycle).

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
CLK	I	The **clock** input provides the fundamental timing for the Pentium processor 75/90/100/120/133. Its frequency is the operating frequency of the Pentium processor 75/90/100/120/133 external bus, and requires TTL levels. All external timing parameters except TDI, TDO, TMS, TRST#, and PICD0-1 are specified with respect to the rising edge of CLK. **NOTE:** **It is recommended that CLK begin toggling within 150 ms after V_{CC} reaches its proper operating level. This recommendation is only to ensure long-term reliability of the device.**
CPUTYP	I	**CPU type** distinguishes the Primary processor from the Dual processor. In a single processor environment, or when the Pentium processor 75/90/100/120/133 is acting as the Primary processor in a dual processing system, CPUTYP should be strapped to V_{SS}. The Dual processor should have CPUTYP strapped to V_{CC}. For the future Pentium OverDrive processor, CPUTYP will be used to determine whether the bootup handshake protocol will be used (in a dual socket system) or not (in a single socket system).
D/C#	O	The **data/code** output is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. D/C# distinguishes between data and code or special cycles.
D/P#	O	The **dual/primary** processor indication. The Primary processor drives this pin low when it is driving the bus, otherwise it drives this pin high. D/P# is always driven. D/P# can be sampled for the current cycle with ADS# (like a status pin). This pin is defined only on the Primary processor. Dual processing is supported in a system only if both processors are operating at identical core and bus frequencies. Within these restrictions, two processors of different steppings may operate together in a system.
D63-D0	I/O	These are the 64 **data lines** for the processor. Lines D7-D0 define the least significant byte of the data bus; lines D63-D56 define the most significant byte of the data bus. When the CPU is driving the data lines, they are driven during the T2, T12, or T2P clocks for that cycle. During reads, the CPU samples the data bus when BRDY# is returned.
DP7-DP0	I/O	These are the **data parity** pins for the processor. There is one for each byte of the data bus. They are driven by the Pentium processor 75/90/100/120/133 with even parity information on writes in the same clock as write data. Even parity information must be driven back to the Pentium processor 75/90/100/120/133 on these pins in the same clock as the data to ensure that the correct parity check status is indicated by the Pentium processor 75/90/100/120/133. DP7 applies to D63-56, DP0 applies to D7-0.
[DPEN#] PICD0	I/O	**Dual processing enable** is an output of the Dual processor and an input of the Primary processor. The Dual processor drives DPEN# low to the Primary processor at RESET to indicate that the Primary processor should enable dual processor mode. DPEN# may be sampled by the system at the falling edge of RESET to determine if Socket 5 is occupied. DPEN# shares a pin with PICD0.
EADS#	I	This signal indicates that a valid **external address** has been driven onto the Pentium processor 75/90/100/120/133 address pins to be used for an inquire cycle.
EWBE#	I	The **external write buffer empty** input, when inactive (high), indicates that a write cycle is pending in the external system. When the Pentium processor 75/90/100/120/133 generates a write, and EWBE# is sampled inactive, the Pentium processor 75/90/100/120/133 will hold off all subsequent writes to all E- or M-state lines in the data cache until all write cycles have completed, as indicated by EWBE# being active.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
FERR#	O	The **floating point error** pin is driven active when an unmasked floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS type floating point error reporting. FERR# is never driven active by the Dual processor.
FLUSH#	I	When asserted, the **cache flush** input forces the Pentium processor 75/90/100/120/133 to write back all modified lines in the data cache and invalidate its internal caches. A Flush Acknowledge special cycle will be generated by the Pentium processor 75/90/100/120/133 indicating completion of the write back and invalidation. If FLUSH# is sampled low when RESET transitions from high to low, tristate test mode is entered. If two Pentium processors 75/90/100/120/133 are operating in dual processing mode in a system and FLUSH# is asserted, the Dual processor will perform a flush first (without a flush acknowledge cycle), then the Primary processor will perform a flush followed by a flush acknowledge cycle. **NOTE:** If the FLUSH# signal is asserted in dual processing mode, it must be deasserted at least one clock prior to BRDY# of the FLUSH Acknowledge cycle to avoid DP arbitration problems.
FRCMC#	I	The **functional redundancy checking master/checker** mode input is used to determine whether the Pentium processor 75/90/100/120/133 is configured in master mode or checker mode. When configured as a master, the Pentium processor 75/90/100/120/133 drives its output pins as required by the bus protocol. When configured as a checker, the Pentium processor 75/90/100/120/133 tristates all outputs (except IERR# and TDO) and samples the output pins. The configuration as a master/checker is set after RESET and may not be changed other than by a subsequent RESET.
HIT#	O	The **hit** indication is driven to reflect the outcome of an inquire cycle. If an inquire cycle hits a valid line in either the Pentium processor 75/90/100/120/133 data or instruction cache, this pin is asserted two clocks after EADS# is sampled asserted. If the inquire cycle misses the Pentium processor 75/90/100/120/133 cache, this pin is negated two clocks after EADS#. This pin changes its value only as a result of an inquire cycle and retains its value between the cycles.
HITM#	O	The **hit to a modified line** output is driven to reflect the outcome of an inquire cycle. It is asserted after inquire cycles which resulted in a hit to a modified line in the data cache. It is used to inhibit another bus master from accessing the data until the line is completely written back.
HLDA	O	The **bus hold acknowledge** pin goes active in response to a hold request driven to the processor on the HOLD pin. It indicates that the Pentium processor 75/90/100/120/133 has floated most of the output pins and relinquished the bus to another local bus master. When leaving bus hold, HLDA will be driven inactive and the Pentium processor 75/90/100/120/133 will resume driving the bus. If the Pentium processor 75/90/100/120/133 has a bus cycle pending, it will be driven in the same clock that HLDA is de-asserted.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
HOLD	I	In response to the **bus hold request,** the Pentium processor 75/90/100/120/133 will float most of its output and input/output pins and assert HLDA after completing all outstanding bus cycles. The Pentium processor 75/90/100/120/133 will maintain its bus in this state until HOLD is de-asserted. HOLD is not recognized during LOCK cycles. The Pentium processor 75/90/100/120/133 will recognize HOLD during reset.
IERR#	O	The **internal error** pin is used to indicate two types of errors, internal parity errors and functional redundancy errors. If a parity error occurs on a read from an internal array, the Pentium processor 75/90/100/120/133 will assert the IERR# pin for one clock and then shutdown. If the Pentium processor 75/90/100/120/133 is configured as a checker and a mismatch occurs between the value sampled on the pins and the corresponding value computed internally, the Pentium processor 75/90/100/120/133 will assert IERR# two clocks after the mismatched value is returned.
IGNNE#	I	This is the **ignore numeric error** input. This pin has no effect when the NE bit in CR0 is set to 1. When the CR0.NE bit is 0, and the IGNNE# pin is asserted, the Pentium processor 75/90/100/120/133 will ignore any pending unmasked numeric exception and continue executing floating-point instructions for the entire duration that this pin is asserted. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating point instruction is one of FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor 75/90/100/120/133 will execute the instruction in spite of the pending exception. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating-point instruction is one other than FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the Pentium processor 75/90/100/120/133 will stop execution and wait for an external interrupt. IGNNE# is internally masked when the Pentium processor 75/90/100/120/133 is configured as a Dual processor.
INIT	I	The Pentium processor 75/90/100/120/133 **initialization** input pin forces the Pentium processor 75/90/100/120/133 to begin execution in a known state. The processor state after INIT is the same as the state after RESET except that the internal caches, write buffers, and floating point registers retain the values they had prior to INIT. INIT may NOT be used in lieu of RESET after power-up. If INIT is sampled high when RESET transitions from high to low, the Pentium processor 75/90/100/120/133 will perform built-in self test prior to the start of program execution.
INTR/LINT0	I	An active **maskable interrupt** input indicates that an external interrupt has been generated. If the IF bit in the EFLAGS register is set, the Pentium processor 75/90/100/120/133 will generate two locked interrupt acknowledge bus cycles and vector to an interrupt handler after the current instruction execution is completed. INTR must remain active until the first interrupt acknowledge cycle is generated to assure that the interrupt is recognized. If the local APIC is enabled, this pin becomes **local interrupt 0.**

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
INV	I	The **invalidation** input determines the final cache line state (S or I) in case of an inquire cycle hit. It is sampled together with the address for the inquire cycle in the clock EADS# is sampled active.
KEN#	I	The **cache enable** pin is used to determine whether the current cycle is cacheable or not and is consequently used to determine cycle length. When the Pentium processor 75/90/100/120/133 generates a cycle that can be cached (CACHE# asserted) and KEN# is active, the cycle will be transformed into a burst line fill cycle.
LINT0/INTR	I	If the APIC is enabled, this pin is **local interrupt 0.** If the APIC is disabled, this pin is **interrupt.**
LINT1/NMI	I	If the APIC is enabled, this pin is **local interrupt 1.** If the APIC is disabled, this pin is **non-maskable interrupt.**
LOCK#	O	The **bus lock** pin indicates that the current bus cycle is locked. The Pentium processor 75/90/100/120/133 will not allow a bus hold when LOCK# is asserted (but AHOLD and BOFF# are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the BRDY# is returned for the last locked bus cycle. LOCK# is guaranteed to be de-asserted for at least one clock between back-to-back locked cycles.
M/IO#	O	The **memory/input-output** is one o f the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. M/IO# distinguishes between memory and I/O cycles.
NA#	I	An active **next address** input indicates that the external memory system is ready to accept a new bus cycle although all data transfers for the current cycle have not yet completed. The Pentium processor 75/90/100/120/133 will issue ADS# for a pending cycle two clocks after NA# is asserted. The Pentium processor 75/90/100/120/133 supports up to 2 outstanding bus cycles.
NMI/LINT1	I	The **non-maskable interrupt** request signal indicates that an external non-maskable interrupt has been generated. If the local APIC is enabled, this pin becomes **local interrupt 1.**
PBGNT#	I/O	**Private bus grant** is the grant line that is used when two Pentium processors 75/90/100/120/133 are configured in dual processing mode, in order to perform private bus arbitration. PBGNT# should be left unconnected if only one Pentium processor 75/90/100/120/133 exists in a system.
PBREQ#	I/O	**Private bus request** is the request line that is used when two Pentium processors 75/90/100/120/133 are configured in dual processing mode, in order to perform private bus arbitration. PBREQ# should be left unconnected if only one Pentium processor 75/90/100/120/133 exists in a system.
PCD	O	The **page cache disable** pin reflects the state of the PCD bit in CR3, the Page Directory Entry, or the Page Table Entry. The purpose of PCD is to provide an external cacheability indication on a page by page basis.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
PCHK#	O	The **parity check** output indicates the result of a parity check on a data read. It is driven with parity status two clocks after BRDY# is returned. PCHK# remains low one clock for each clock in which a parity error was detected. Parity is checked only for the bytes on which valid data is returned. When two Pentium processors 75/90/100/120/133 are operating in dual processing mode, PCHK# may be driven two or three clocks after BRDY# is returned.
PEN#	I	The **parity enable** input (along with CR4.MCE) determines whether a machine check exception will be taken as a result of a data parity error on a read cycle. If this pin is sampled active in the clock a data parity error is detected, the Pentium processor 75/90/100/120/133 will latch the address and control signals of the cycle with the parity error in the machine check registers. If, in addition, the machine check enable bit in CR4 is set to "1", the Pentium processor 75/90/100/120/133 will vector to the machine check exception before the beginning of the next instruction.
PHIT#	I/O	**Private hit** is a hit indication used when two Pentium processors 75/90/100/120/133 are configured in dual processing mode, in order to maintain local cache coherency. PHIT# should be left unconnected if only one Pentium processor 75/90/100/120/133 exists in a system.
PHITM#	I/O	**Private modified hit** is a hit indication used when two Pentium processors 75/90/100/120/133 are configured in dual processing mode, in order to maintain local cache coherency. PHITM# should be left unconnected if only one Pentium processor 75/90/100/120/133 exists in a system.
PICCLK	I	The APIC interrupt controller serial data bus clock is driven into the **programmable interrupt controller clock** input of the Pentium processor 75/90/100/120/133.
PICD0-1 [DPEN#] [APICEN]	I/O	**Programmable interrupt controller data lines 0-1** of the Pentium processor 75/90/100/120/133 comprise the data portion of the APIC 3-wire bus. They are open-drain outputs that require external pull-up resistors. These signals share pins with DPEN# and APICEN.
PM/BP[1:0]	O	These pins function as part of the performance monitoring feature. The breakpoint 1-0 pins are multiplexed with the **performance monitoring 1-0** pins. The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
PRDY	O	The **probe ready** output pin indicates that the processor has stopped normal execution in response to the R/S# pin going active, or Probe Mode being entered.
PWT	O	The **page write through** pin reflects the state of the PWT bit in CR3, the page directory entry, or the page table entry. The PWT pin is used to provide an external write back indication on a page-by-page basis.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
R/S#	I	The **run/stop** input is an asynchronous, edge-sensitive interrupt used to stop the normal execution of the processor and place it into an idle state. A high to low transition on the R/S# pin will interrupt the processor and cause it to stop execution at the next instruction boundary.
RESET	I	**RESET** forces the Pentium processor 75/90/100/120/133 to begin execution at a known state. All the Pentium processor 75/90/100/120/133 internal caches will be invalidated upon the RESET. Modified lines in the data cache are not written back. FLUSH#, FRCMC# and INIT are sampled when RESET transitions from high to low to determine if tristate test mode or checker mode will be entered, or if BIST will be run.
SCYC	O	The **split cycle** output is asserted during misaligned LOCKed transfers to indicate that more than two cycles will be locked together. This signal is defined for locked cycles only. It is undefined for cycles which are not locked.
SMI#	I	The **system management interrupt** causes a system management interrupt request to be latched internally. When the latched SMI# is recognized on an instruction boundary, the processor enters System Management Mode.
SMIACT#	O	An active **system management interrupt active** output indicates that the processor is operating in System Management Mode.
STPCLK#	I	Assertion of the **stop clock** input signifies a request to stop the internal clock of the Pentium processor 75/90/100/120/133 thereby causing the core to consume less power. When the CPU recognizes STPCLK#, the processor will stop execution on the next instruction boundary, unless superseded by a higher priority interrupt, and generate a stop grant acknowledge cycle. When STPCLK# is asserted, the Pentium processor 75/90/100/120/133 will still respond to interprocessor and external snoop requests.
TCK	I	The **testability clock** input provides the clocking function for the Pentium processor 75/90/100/120/133 boundary scan in accordance with the IEEE Boundary Scan interface (Standard 1149.1). It is used to clock state information and data into and out of the Pentium processor 75/90/100/120/133 during boundary scan.

Table 2. Quick Pin Reference (Contd.)

Symbol	Type*	Name and Function
TDI	I	The **test data input** is a serial input for the test logic. TAP instructions and data are shifted into the Pentium processor 75/90/100/120/133 on the TDI pin on the rising edge of TCK when the TAP controller is in an appropriate state.
TDO	O	The **test data output** is a serial output of the test logic. TAP instructions and data are shifted out of the Pentium processor 75/90/100/120/133 on the TDO pin on TCK's falling edge when the TAP controller is in an appropriate state.
TMS	I	The value of the **test mode select** input signal sampled at the rising edge of TCK controls the sequence of TAP controller state changes.
TRST#	I	When asserted, the **test reset** input allows the TAP controller to be asynchronously initialized.
V_{CC}	I	The Pentium processor 75/90/100/120/133 has 53 3.3V **power** inputs.
V_{SS}	I	The Pentium processor 75/90/100/120/133 has 53 **ground** inputs.
W/R#	O	**Write/read** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. W/R# distinguishes between write and read cycles.
WB/WT#	I	The **write back/write through** input allows a data cache line to be defined as write back or write through on a line-by-line basis. As a result, it determines whether a cache line is initially in the S or E state in the data cache.

* The pins are classified as Input or Output based on their function in Master Mode. See the Functional Redundancy Checking section in the "Error Detection" chapter of the *Pentium® Processor Family Developer's Manual,* Vol. 1, for further information.

2.4 Pin Reference Tables

Table 3. Output Pins

Name	Active Level	When Floated
ADS#*	Low	Bus Hold, BOFF#
ADSC#	Low	Bus Hold, BOFF#
APCHK#	Low	
BE7#-BE5#	Low	Bus Hold, BOFF#
BREQ	High	
CACHE#*	Low	Bus Hold, BOFF#
D/P#**	n/a	
FERR#**	Low	
HIT#*	Low	
HITM#*	Low	
HLD A*	High	
IERR#	Low	
LOCK#*	Low	Bus Hold, BOFF#
M/IO#*, D/C#*, W/R#*	n/a	Bus Hold, BOFF#
PCHK#	Low	
BP3-2, PM1/BP1, PM0/BP0	High	
PRDY	High	
PWT, PCD	High	Bus Hold, BOFF#
SCYC*	High	Bus Hold, BOFF#
SMIACT#	Low	
TDO	n/a	All states except Shift-DR and Shift-IR

NOTES:
All output and input/output pins are floated during tristate test mode and checker mode (except IERR#).
* These are I/O signals when two Pentium® processors 75/90/100/120/133 are operating in dual processing mode.
** These signals are undefined when the CPU is configured as a Dual Processor.

Table 4. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal Resistor	Qualified
A20M#*	Low	Asynchronous		
AHOLD	High	Synchronous		
BF	High	Synchronous/RESET	Pullup	
BOFF#	Low	Synchronous		
BRDY#	Low	Synchronous		Bus State T2, T12, T2P
BRDYC#	Low	Synchronous	Pullup	Bus State T2, T12, T2P
BUSCHK#	Low	Synchronous	Pullup	BRDY#
CLK	n/a			
CPUTYP	High	Synchronous/RESET		
EADS#	Low	Synchronous		
EWBE#	Low	Synchronous		BRDY#
FLUSH#	Low	Asynchronous		
FRCMC#	Low	Asynchronous		
HOLD	High	Synchronous		
IGNNE#*	Low	Asynchronous		
INIT	High	Asynchronous		
INTR	High	Asynchronous		
INV	High	Synchronous		EADS#
KEN#	Low	Synchronous		First BRDY#/NA#
NA#	Low	Synchronous		Bus State T2,TD,T2P
NMI	High	Asynchronous		
PEN#	Low	Synchronous		BRDY#
PICCLK	High	Asynchronous	Pullup	
R/S#	n/a	Asynchronous	Pullup	
RESET	High	Asynchronous		
SMI#	Low	Asynchronous	Pullup	
STPCLK#	Low	Asynchronous	Pullup	
TCK	n/a		Pullup	
TDI	n/a	Synchronous/TCK	Pullup	TCK
TMS	n/a	Synchronous/TCK	Pullup	TCK
TRST#	Low	Asynchronous	Pullup	
WB/WT#	n/a	Synchronous		First BRDY#/NA#

* Undefined when the CPU is configured as a Dual processor.

Table 5. Input/Output Pins

Name	Active Level	When Floated	Qualified (when an input)	Internal Resistor
A31-A3	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
AP	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
BE4#-BE0#	Low	Address Hold, Bus Hold, BOFF#	RESET	Pulldown*
D63-D0	n/a	Bus Hold, BOFF#	BRDY#	
DP7-DP0	n/a	Bus Hold, BOFF#	BRDY#	
PICD0[DPEN#]				Pullup
PICD1[APICEN]				Pulldown

NOTES:
All output and input/output pins are floated during tristate test mode (except TDO) and checker mode (except IERR# and TDO).
* BE3#-BE0# have Pulldowns during RESET only.

Table 6. Inter-Processor I/O Pins

Name	Active Level	Internal Resistor
PHIT#	Low	Pullup
PHITM#	Low	Pullup
PBGNT#	Low	Pullup
PBREQ#	Low	Pullup

NOTE:
For proper inter-processor operation, the system cannot load these signals.

2.5 Pin Grouping According to Function

Table 7 organizes the pins with respect to their function.

Table 7. Pin Functional Grouping

Function	Pins
Clock	CLK
Initialization	RESET, INIT, BF1–BF0
Address Bus	A31-A3, BE7#–BE0#
Address Mask	A20M#
Data Bus	D63-D0
Address Parity	AP, APCHK#
APIC Support	PICCLK, PICD0-1
Data Parity	DP7-DP0, PCHK#, PEN#
Internal Parity Error	IERR#
System Error	BUSCHK#
Bus Cycle Definition	M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK#
Bus Control	ADS#, ADSC#, BRDY#, BRDYC#, NA#
Page Cacheability	PCD, PWT
Cache Control	KEN#, WB/WT#
Cache Snooping/Consistency	AHOLD, EADS#, HIT#, HITM#, INV
Cache Flush	FLUSH#
Write Ordering	EWBE#
Bus Arbitration	BOFF#, BREQ, HOLD, HLDA
Dual Processing Private Bus Control	PBGNT#, PBREQ#, PHIT#, PHITM#
Interrupts	INTR, NMI
Floating Point Error Reporting	FERR#, IGNNE#
System Management Mode	SMI#, SMIACT#
Functional Redundancy Checking	FRCMC# (IERR#)
TAP Port	TCK, TMS, TDI, TDO, TRST#
Breakpoint/Performance Monitoring	PM0/BP0, PM1/BP1, BP3-2
Power Management	STPCLK#
Miscellaneous Dual Processing	CPUTYP, D/P#
Probe Mode	R/S#, PRDY

3.0 ELECTRICAL SPECIFICATIONS

This section describes the electrical differences between the Pentium processor 60/66 and the Pentium processor 75/90/100/120/133 and the DC and AC specifications.

3.1 Electrical Differences Between Pentium® Processor 75/90/100/120/133 and Pentium Processor 60/66

Pentium® Processor 60/66 Electrical Characteristic	Difference in Pentium Processor 75/90/100/120/133
5V Power Supply	3.3V Power Supply*
5V TTL Inputs/Outputs	3.3V Inputs/Outputs
Pentium Processor 60/66 Buffer Models	Pentium Processor 75/90/100/120/133 Buffer Models

* The upgrade socket specifies two 5V inputs (section 6.0.).

The sections that follow will briefly point out some ways to design with these electrical differences.

3.1.1 3.3V POWER SUPPLY

The Pentium processor 75/90/100/120/133 has all V_{CC} 3.3V inputs. By connecting all Pentium processor 60/66 V_{CC} inputs to a common and dedicated power plane, that plane can be converted to 3.3V for the Pentium processor 75/90/100/120/133.

The CLK and PICCLK inputs can tolerate a 5V input signal. This allows the Pentium processor 75/90/100/120/133 to use 5V or 3.3V clock drivers.

3.1.2 3.3V INPUTS AND OUTPUTS

The inputs and outputs of the Pentium processor 75/90/100/120/133 are 3.3V JEDEC standard levels. Both inputs and outputs are also TTL-compatible, although the inputs cannot tolerate voltage swings above the 3.3V V_{IN} max.

For Pentium processor 75/90/100/120/133 outputs, if the Pentium processor 60/66 system support components use TTL-compatible inputs, they will interface to the Pentium processor 75/90/100/120/133 without extra logic. This is because the Pentium processor 75/90/100/120/133 drives according to the 5V TTL specification (but not beyond 3.3V).

For Pentium processor 75/90/100/120/133 inputs, the voltage must not exceed the 3.3V V_{IH3} maximum specification. System support components can consist of 3.3V devices or open-collector devices. 3.3V support components may interface to the Pentium processor 60/66 since they typically meet 5V TTL specifications. In an open-collector configuration, the external resistor may be biased with the CPU V_{CC}; as the CPU's V_{CC} changes from 5V to 3.3V, so does this signal's maximum drive.

The CLK and PICCLK inputs of the Pentium processor 75/90/100/120/133 are 5V tolerant, so they are electrically identical to the Pentium processor 60/66 clock input. This allows a Pentium processor 60/66 clock driver to drive the Pentium processor 75/90/100/120/133.

All pins, other than the CLK and PICCLK inputs, are 3.3V-only. If an 8259A interrupt controller is used, for example, the system must provide level converters between the 8259A and the Pentium processor 75/90/100/120/133.

3.1.3 3.3V PENTIUM® PROCESSOR 75/90/100/120/133 BUFFER MODELS

The structure of the buffer models of the Pentium processor 75/90/100/120/133 is the same as that of the Pentium processor 60/66, but the values of the components change since the Pentium processor 75/90/100/120/133 buffers are 3.3V buffers on a different process.

Despite this difference, the simulation results of Pentium processor 75/90/100/120/133 buffers and Pentium processor 60/66 buffers look nearly identical. Since the 0pF AC specifications of the Pentium processor 75/90/100/120/133 are derived from the Pentium processor 60/66 specifications, the system should see little difference between the AC behavior of the Pentium processor 75/90/100/120/133 and the Pentium processor 60/66.

To meet specifications, simulate the AC timings with Pentium processor 75/90/100/120/133 buffer models. Pay special attention to the new signal quality restrictions imposed by 3.3V buffers.

3.2 Absolute Maximum Ratings

The values listed below are stress ratings only. Functional operation at the maximums is not implied or guaranteed. Functional operating conditions are given in the AC and DC specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium processor 75/90/100/120/133 contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Case temperature under bias ..... −65°C to 110°C

Storage temperature ..... −65°C to 150°C

3V Supply voltage with respect to V_{SS} ..... −0.5V to +4.6V

3V Only Buffer DC Input Voltage −0.5V to V_{CC} +0.5; not to exceed V_{CC3} max(2)

5V Safe Buffer DC Input Voltage ..... −0.5V to 6.5V(1,3)

NOTES:

1. Applies to CLK and PICCLK.
2. Applies to all Pentium processor 75/90/100/120/133 inputs except CLK and PICCLK.
3. See overshoot/undershoot transient spec.

** WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

3.3 DC Specifications

Tables 8, 9, and 10 list the DC specifications which apply to the Pentium processor 75/90/100/120/133. The Pentium processor 75/90/100/120/133 is a 3.3V part internally. The CLK and PICCLK inputs may be a 3.3V or 5V inputs. Since the 3.3V (5V-safe) input levels defined in Table 9 are the same as the 5V TTL levels, the CLK and PICCLK inputs are compatible with existing 5V clock drivers. The power dissipation specification in Table 11 is provided for design of thermal solutions during operation in a sustained maximum level. This is the worst case power the device would dissipate in a system. This number is used for design of a thermal solution for the device.

Table 8. 3.3V DC Preliminary Specifications

T_{CASE} = 0 to 70°C; 3.135V < V_{CC} < 3.6V

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL3}	Input Low Voltage	−0.3	0.8	V	TTL Level(3)
V_{IH3}	Input High Voltage	2.0	V_{CC}+0.3	V	TTL Level(3)
V_{OL3}	Output Low Voltage		0.4	V	TTL Level(1, 3)
V_{OH3}	Output High Voltage	2.4		V	TTL Level(2, 3)
I_{CC3}	Power Supply Current		3400	mA	@133 MHz
			3730	mA	@120 MHz(4, 5)
			3250	mA	@100 MHz(4)
			2950	mA	@90 MHz(4)
			2650	mA	@75 MHz(4)

NOTES:

1. Parameter measured at 4 mA.
2. Parameter measured at 3 mA.
3. 3.3V TTL levels apply to all signals except CLK and PICCLK.
4. This value should be used for power supply design. It was determined using a worst case instruction mix and V_{CC} = 3.6V. Power supply transient response and decoupling capacitors must be sufficient to handle the instantaneous current changes occurring during transitions from stop clock to full active modes. For more information, refer to section 3.4.3.
5. Please also check stepping information.

Table 9. 3.3V (5V-Safe) DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL5}	Input Low Voltage	−0.3	0.8	V	TTL Level(1)
V_{IH5}	Input High Voltage	2.0	5.55	V	TTL Level(1)

NOTES:
1. Applies to CLK and PICCLK only.

Table 10. Input and Output Characteristics

Symbol	Parameter	Min	Max	Unit	Notes
C_{IN}	Input Capacitance		15	pF	4
C_O	Output Capacitance		20	pF	4
$C_{I/O}$	I/O Capacitance		25	pF	4
C_{CLK}	CLK Input Capacitance		15	pF	4
C_{TIN}	Test Input Capacitance		15	pF	4
C_{TOUT}	Test Output Capacitance		20	pF	4
C_{TCK}	Test Clock Capacitance		15	pF	4
I_{LI}	Input Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{LO}	Output Leakage Current		±15	μA	$0 < V_{IN} < V_{CC3}$(1)
I_{IH}	Input Leakage Current		200	μA	$V_{IN} = 2.4V$(3)
I_{IL}	Input Leakage Current		−400	μA	$V_{IN} = 0.4V$(2)

NOTES:
1. This parameter is for input without pullup or pulldown.
2. This parameter is for input with pullup.
3. This parameter is for input with pulldown.
4. Guaranteed by design.

Table 11. Preliminary Power Dissipation Requirements for Thermal Solution Design

Parameter	Typical[1]	Max[2]	Unit	Notes
Active Power Dissipation	4.3	11.2	Watts	@133 MHz
	5.06	12.81[6]	Watts	@120 MHz[6]
	3.9	10.1	Watts	@100 MHz
	3.5	9.0	Watts	@90 MHz
	3.0	8.0	Watts	@75 MHz
Stop Grant and Auto Halt Powerdown Power Dissipation		1.7	Watts	@133 MHz[3]
		1.76[6]	Watts	@120 MHz[3, 6]
		1.55	Watts	@100 MHz[3]
		1.40	Watts	@90 MHz[3]
		1.20	Watts	@75 MHz
Stop Clock Power Dissipation	0.02	<0.3	Watts	(4, 5)

NOTES:
1. This is the typical power dissipation in a system. This value was the average value measured in a system using a typical device at V_{CC} = 3.3V running typical applications. This value is highly dependent upon the specific system configuration.
2. Systems must be designed to thermally dissipate the maximum active power dissipation. It is determined using worst case instruction mix with V_{CC} = 3.3V and also takes into account the thermal time constants of the package.
3. Stop Grant/Auto Halt Powerdown Power Dissipation is determined by asserting the STPCLK# pin or executing the HALT instruction.
4. Stop Clock Power Dissipation is determined by asserting the STPCLK# pin and then removing the external CLK input.
5. Complete characterization of this specification was still in process at the time of print. Please contact Intel for the latest information. The final specification will be less than 0.1W.
6. This is determined using worst case instruction mix with V_{CC} = 3.52V and also takes into account the thermal time constants of the package.

3.4 AC Specifications

The AC specifications of the Pentium processor 75/90/100/120/133 consist of setup times, hold times, and valid delays at 0 pF.

3.4.1 PRIVATE BUS

When two Pentium processors 75/90/100/120/133 are operating in dual processor mode, a "private bus" exists to arbitrate for the CPU bus and maintain local cache coherency. The private bus consists of two pinout changes:

1. Five pins are added: PBREQ#, PBGNT#, PHIT#, PHITM#, D/P#.
2. Ten output pins become I/O pins: ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, HIT#, HITM#, HLDA, SCYC.

The new pins are given AC specifications of valid delays at 0 pF, setup times, and hold times. Simulate with these parameters and their respective I/O buffer models to guarantee that proper timings are met.

The AC specification gives input setup and hold times for the ten signals that become I/O pins. These setup and hold times must only be met when a dual processor is present in the system.

3.4.2 POWER AND GROUND

For clean on-chip power distribution, the Pentium processor 75/90/100/120/133 has 53 V_{CC} (power) and 53 V_{SS} (ground) inputs. Power and ground connections must be made to all external V_{CC} and V_{SS} pins of the Pentium processor 75/90/100/120/133. On the circuit board all V_{CC} pins must be connected to a 3.3V V_{CC} plane. All V_{SS} pins must be connected to a V_{SS} plane.

3.4.3 DECOUPLING RECOMMENDATIONS

Liberal decoupling capacitance should be placed near the Pentium processor 75/90/100/120/133. The Pentium processor 75/90/100/120/133 driving its large address and data buses at high frequencies can cause transient power surges, particularly when driving large capacitive loads.

Low inductance capacitors and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by shortening circuit board traces between the Pentium processor 75/90/100/120/133 and decoupling capacitors as much as possible.

These capacitors should be evenly distributed around each component on the 3.3V plane. Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

For the Pentium processor 75/90/100/120/133, the power consumption can transition from a low level of power to a much higher level (or high to low power) very rapidly. A typical example would be entering or exiting the Stop Grant state. Another example would be executing a HALT instruction, causing the Pentium processor 75/90/100/120/133 to enter the Auto HALT Powerdown state, or transitioning from HALT to the Normal state. All of these examples may cause abrupt changes in the power being consumed by the Pentium processor 75/90/100/120/133. Note that the Auto HALT Powerdown feature is always enabled even when other power management features are not implemented.

Bulk storage capacitors with a low ESR (Effective Series Resistance) in the 10 to 100 μf range are required to maintain a regulated supply voltage during the interval between the time the current load changes and the point that the regulated power supply output can react to the change in load. In order to reduce the ESR, it may be necessary to place several bulk storage capacitors in parallel.

These capacitors should be placed near the Pentium processor 75/90/100/120/133 (on the 3.3V plane) to ensure that the supply voltage stays within specified limits during changes in the supply current during operation.

3.4.4 CONNECTION SPECIFICATIONS

All NC and INC pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to V_{CC}. Unused active high inputs should be connected to ground.

2

3.4.5 AC TIMING TABLES

3.4.5.1 AC Timing Table for a 50-MHz Bus

The AC specifications given in Tables 12 and 13 consist of output delays, input setup requirements and input hold requirements for a 50-MHz external bus. All AC specifications (with the exception of those for the TAP signals and APIC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor 75/90/100/120/133 operation.

Table 12. Pentium® Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	25.0	50.0	MHz		Max Core Freq = 100 MHz @1/2
t_{1a}	CLK Period	20.0	40.0	nS	4	
t_{1b}	CLK Period Stability		±250	pS		1, 25
t_2	CLK High Time	4.0		nS	4	@2V,(1)
t_3	CLK Low Time	4.0		nS	4	@0.8V,(1)
t_4	CLK Fall Time	0.15	1.5	nS	4	(2.0V–0.8V),(1,5)
t_5	CLK Rise Time	0.15	1.5	nS	4	(0.8V–2.0V), (1,5)
t_{6a}	ADS#, ADSC#, PWT, PCD, BE0-7#, M/IO#, D/C#, CACHE#, SCYC, W/R# Valid Delay	1.0	7.0	nS	5	

Table 12. Pentium® Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{6b}	AP Valid Delay	1.0	8.5	nS	5	
t_{6c}	A3-A31, LOCK# Valid Delay	1.1	7.0	nS	5	
t_7	ADS#, ADSC#, AP, A3-A31, PWT, PCD, BE0-7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	6	1
t_8	APCHK#, IERR#, FERR#, PCHK# Valid Delay	1.0	8.3	nS	5	4
t_{9a}	BREQ, HLDA, SMIACT# Valid Delay	1.0	8.0	nS	5	4
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	5	
t_{10b}	HITM# Valid Delay	1.1	6.0	nS	5	
t_{11a}	PM0-1, BP0-3 Valid Delay	1.0	10.0	nS	5	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	5	
t_{12}	D0-D63, DP0-7 Write Data Valid Delay	1.3	8.5	nS	5	
t_{13}	D0-D63, DP0-3 Write Data Float Delay		10.0	nS	6	1
t_{14}	A5-A31 Setup Time	6.5		nS	7	26
t_{15}	A5-A31 Hold Time	1.0		nS	7	
t_{16a}	INV, AP Setup Time	5.0		nS	7	
t_{16b}	EADS# Setup Time	6.0		nS	7	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	7	
t_{18a}	KEN# Setup Time	5.0		nS	7	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	7	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	7	
t_{20}	BRDY#, BRDYC# Setup Time	5.0		nS	7	
t_{21}	BRDY#, BRDYC# Hold Time	1.0		nS	7	
t_{22}	BOFF# Setup Time	5.5		nS	7	
t_{22a}	AHOLD Setup Time	6.0		nS	7	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	7	

Table 12. Pentium® Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
3.135 < V_{CC} < 3.6V, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	7	
t_{25}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	7	
t_{25a}	HOLD Hold Time	1.5		nS	7	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	7	12, 16
t_{27}	A20M#, INTR, STPCLK# Hold Time	1 .0		nS	7	13
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	7	12, 16, 17
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	7	13
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs	7	15, 17
t_{31}	R/S# Setup Time	5.0		nS	7	12, 16, 17
t_{32}	R/S# Hold Time	1.0		nS	7	13
t_{33}	R/S# Pulse Width, Async.	2. 0		CLKs	7	15, 17
t_{34}	D0-D63, DP0-7 Read Data Setup Time	3.8		nS	7	
t_{35}	D0-D63, DP0-7 Read Data Hold Time	1.5		nS	7	
t_{36}	RESET Setup Time	5.0		nS	8	11, 12, 16
t_{37}	RESET Hold Time	1.0		nS	8	11, 13
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	8	11, 17
t_{39}	RESET Active After V_{CC} & CLK Stable	1.0		mS	8	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.0		nS	8	12, 16, 17
t_{41}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.0		nS	8	13

Table 12. Pentium® Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{42a}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2.0		CLKs	8	To RESET falling edge(16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#, BRDYC#, BUSCHK#) Hold Time, Async.	2.0		CLKs	8	To RESET falling edge(27)
t_{42c}	Reset Configuration Signals (BRDYC#, BUSCHK#) Setup Time, Async.	3.0		CLKs	8	To RESET falling edge(27)
t_{42d}	Reset Configuration Signal BRDYC# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge(1,27)
t_{43a}	BF, CPUTYP Setup Time	1.0		mS	8	To RESET falling edge(22)
t_{43b}	BF, CPUTYP Hold Time	2.0		CLKs	8	To RESET falling edge(22)
t_{43c}	APICEN Setup Time	2.0		CLKs	8	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	8	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		nS	4	
t_{46}	TCK High Time	25.0		nS	4	@2V(1)
t_{47}	TCK Low Time	25.0		nS	4	@0.8V(1)
t_{48}	TCK Fall Time		5.0	nS	4	(2.0V–0.8V)(1,8,9)
t_{49}	TCK Rise Time		5.0	nS	4	(0.8V–2.0V)(1,8,9)
t_{50}	TRST# Pulse Width	40.0		nS	10	Asynchronous(1)
t_{51}	TDI, TMS Setup Time	5.0		nS	9	7
t_{52}	TDI, TMS Hold Time	13.0		nS	9	7
t_{53}	TDO Valid Delay	3.0	20.0	nS	9	8
t_{54}	TDO Float Delay		25.0	nS	9	1, 8
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	9	3, 8, 10

Table 12. Pentium® Processor 610\75, 815\100 AC Specifications for 50-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	9	1, 3, 8, 10
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	9	3, 7, 10
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	9	3, 7, 10
	APIC AC Specifications					
t_{60a}	PICCLK Frequency	2.0	16.66	MHz		
t_{60b}	PICCLK Period	60.0	500.0	nS	4	
t_{60c}	PICCLK High Time	15.0		nS	4	
t_{60d}	PICCLK Low Time	15.0		nS	4	
t_{60e}	PICCLK Rise Time	0.15	25	nS	4	
t_{60f}	PICCLK Fall Time	0.15	25	nS	4	
t_{60g}	PICD0-1 Setup Time	3.0		nS	7	To PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		nS	7	To PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	nS	5	From PICCLK(28,29)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	nS	5	From PICCLK(28,29)

Table 13. Pentium® Processor 610\75, 815\100 Dual Processor Mode AC Specifications for 50 MHz Bus Operation

$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{80}	PBREQ#, PBGNT# PHIT#, PHITM# Flight Time	0	2.0	nS	11	30
t_{83a}	A5-A31 Setup Time	6 .5		nS	7	18, 21, 26
t_{83b}	D/C#, W/R#, CACHE#, LOCK#, SCYC Setup Time	6.0		nS	7	18, 21
t_{83c}	ADS#, M/IO# Setup Time	8.0		nS	7	18, 21
t_{83d}	HIT#, HITM# Setup Time	8.0		nS	7	18, 21
t_{83e}	HLDA Setup Time	6.0		nS	7	18, 21
t_{84}	ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, A5-A31, HLDA, HIT#, HITM#, SCYC Hold Time	1.0		nS	7	18, 21
t_{85}	DPEN# Valid Time		10.0	CLKs		18, 19, 23
t_{86}	DPEN# Hold Time	2.0		CLKs		18, 20, 23
t_{87}	APIC ID (BE0#-BE3#) Setup Time	2.0		CLKs	8	To RESET falling edge(23)
t_{88}	APIC ID (BE0#-BE3#) Hold Time	2.0		CLKs	8	From RESET falling edge(23)
t_{89}	D/P# Valid Delay	1.0	8.0	nS	5	Primary Processor Only

3.4.5.2 AC Timing Tables for a 60-MHz Bus

The AC specifications given in Tables 14 and 15 consist of output delays, input setup requirements and input hold requirements for a 60-MHz external bus. All AC specifications (with the exception of those for the TAP signals and APIC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor 75/90/100/120/133 operation.

Table 14. Pentium® Processor 735\90, 1000\120 AC Specifications for 60-MHz Bus Operation
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	30.0	60.0	MHz	4	
t_{1a}	CLK Period	16.67	33.33	nS	4	
t_{1b}	CLK Period Stability		±250	pS	4	Adjacent Clocks,(1,25)
t_2	CLK High Time	4.0		nS	4	@2V(1)
t_3	CLK Low Time	4.0		nS	4	@0.8V(1)
t_4	CLK Fall Time	0.15	1.5	nS	4	(2.0V–0.8V)(1,5)
t_5	CLK Rise Time	0.15	1.5	nS	4	(0.8V–2.0V)(1,5)
t_{6a}	ADS#, ADSC#, PWT, PCD, BE0-7#, M/IO#, D/C#, CACHE#, SCYC, W/R# Valid Delay	1.0	7.0	nS	5	
t_{6b}	AP Valid Delay	1.0	8.5	nS	5	
t_{6c}	LOCK# Valid Delay	1.1	7.0	nS	5	
t_{6e}	A3–A31 Valid Delay	1.1	6.3	nS	5	
t_7	ADS#, ADSC#, AP, A3-A31, PWT, PCD, BE0-7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	5	1
t_{8a}	APCHK#, IERR#, FERR# Valid Delay	1.0	8.3	nS	5	4
t_{8b}	PCHK# Valid Delay	1.0	7.0	nS	5	4
t_{9a}	BREQ, HLDA Valid Delay	1.0	8.0	nS	5	4
t_{9b}	SMIACT# Valid Delay	1.0	7.6	nS	5	
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	5	
t_{10b}	HITM# Valid Delay	1.1	6.0	nS	5	
t_{11a}	PM0-1, BP0-3 Valid Delay	1.0	10.0	nS	5	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	5	
t_{12}	D0-D63, DP0-7 Write Data Valid Delay	1.3	7.5	nS	5	

Table 14. Pentium® Processor 735\90, 1000\120 AC Specifications for 60-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{13}	D0-D63, DP0-3 Write Data Float Delay		10.0	nS	6	1
t_{14}	A5-A31 Setup Time	6.0		nS	7	26
t_{15}	A5-A31 Hold Time	1.0		nS	7	
t_{16a}	INV, AP Setup Time	5.0		nS	7	
t_{16b}	EADS# Setup Time	5.5		nS	7	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	7	
t_{18a}	KEN# Setup Time	5.0		nS	7	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	7	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	7	
t_{20}	BRDY#, BRDYC# Setup Time	5.0		nS	7	
t_{21}	BRDY#, BRDYC# Hold Time	1.0		nS	7	
t_{22}	AHOLD, BOFF# Setup Time	5.5		nS	7	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	7	
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	7	
t_{25}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	7	
t_{25a}	HOLD Hold Time	1.5		nS	7	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	7	12, 16
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	7	13
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	7	12, 16, 17
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	7	13
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs		15, 17
t_{31}	R/S# Setup Time	5.0		nS	7	12, 16, 17
t_{32}	R/S# Hold Time	1.0		nS	7	13
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs	7	15, 17

Table 14. Pentium® Processor 735\90, 1000\120 AC Specifications for 60-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{34}	D0-D63, DP0-7 Read Data Setup Time	3.0		nS	7	
t_{35}	D0-D63, DP0-7 Read Data Hold Time	1.5		nS	8	
t_{36}	RESET Setup Time	5.0		nS	8	11, 12, 16
t_{37}	RESET Hold Time	1.0		nS	8	11, 13
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	8	11, 17
t_{39}	RESET Active After V_{CC} & CLK Stable	1.0		mS	8	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.0		nS	8	12, 16, 17
t_{41}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.0		nS	8	13
t_{42a}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2.0		CLKs	8	To RESET falling edge(16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#, BRDYC#, BUSCHK#) Hold Time, Async.	2.0		CLKs	8	To RESET falling edge (27)
t_{42c}	Reset Configuration Signals (BRDYC#, BUSCHK#) Setup Time, Async.	3.0		CLKs	8	To RESET falling edge(27)
t_{42d}	Reset Configuration Signal BRDYC# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge(1,27)
t_{43a}	BF, CPUTYP Setup Time	1.0		mS	8	To RESET falling edge(22)
t_{43b}	BF, CPUTYP Hold Time	2.0		CLKs	8	To RESET falling edge(22)
t_{43c}	APICEN Setup Time	2.0		CLKs	8	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	8	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz	8	
t_{45}	TCK Period	62.5		nS	4	
t_{46}	TCK High Time	25.0		nS	4	@2V(1)
t_{47}	TCK Low Time	25.0		nS	4	@0.8V(1)

Table 14. Pentium® Processor 735\90, 1000\120 AC Specifications for 60-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{48}	TCK Fall Time		5.0	nS	4	(2.0V–0.8V)(1,8,9)
t_{49}	TCK Rise Time		5.0	nS	4	(0.8V–2.0V)(1,8,9)
t_{50}	TRST# Pulse Width	40.0		nS	10	Asynchronous(1)
t_{51}	TDI, TMS Setup Time	5.0		nS	9	7
t_{52}	TDI, TMS Hold Time	13.0		nS	9	7
t_{53}	TDO Valid Delay	3.0	20.0	nS	9	8
t_{54}	TDO Float Delay		25.0	nS	9	1, 8
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	9	3, 8, 10
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	9	1, 3, 8, 10
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	9	3, 7, 10
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	9	3, 7, 10
	APIC AC Specifications					
t_{60a}	PICCLK Frequency	2.0	16.66	MHz	4	
t_{60b}	PICCLK Period	60.0	500.0	nS	4	
t_{60c}	PICCLK High Time	15.0		nS	4	
t_{60d}	PICCLK Low Time	15.0		nS	4	
t_{60e}	PICCLK Rise Time	0.15	2.5	nS	4	
t_{60f}	PICCLK Fall Time	0.15	2.5	nS	4	
t_{60g}	PICD0-1 Setup Time	3.0		nS	7	To PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		nS	7	To PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	nS	5	From PICCLK(28,29)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	nS	5	From PICCLK(28,29)

Table 15. Pentium® Processor 735\90, 1000\120 Dual Processor Mode AC Specifications for 60-MHz Bus Operation

$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{80}	PBREQ#, PBGNT#, PHIT#, PHITM# Flight Time	0	2.0	nS	11	30
t_{83a}	A5-A31 Setup Time	3.9		nS	7	18, 21, 26
t_{83b}	D/C#, W/R#, CACHE#, LOCK#, SCYC Setup Time	4.0		nS	7	18, 21
t_{83c}	ADS#, M/IO# Setup Time	6.0		nS	7	18, 21
t_{83d}	HIT#, HITM# Setup Time	6.0		nS	7	18, 21
t_{83e}	HLDA Setup Time	6.0		nS	7	18, 21
t_{84}	ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, A5-A31, HLDA, HIT#, HITM#, SCYC Hold Time	1.0		nS	7	18, 21
t_{85}	DPEN# Valid Time		10.0	CLKs		18, 19, 23
t_{86}	DPEN# Hold Time	2.0		CLKs		18, 20, 23
t_{87}	APIC ID (BE0#-BE3#) Setup Time	2.0		CLKs	8	To RESET falling edge(23)
t_{88}	APIC ID (BE0#-BE3#) Hold Time	2.0		CLKs	8	From RESET falling edge(23)
t_{89}	D/P# Valid Delay	1.0	8.0	nS	5	Primary Processor Only

3.4.5.3 AC Timing Tables for a 66-MHz Bus

The AC specifications given in Tables 16 and 17 consist of output delays, input setup requirements and input hold requirements for a 66-MHz external bus. All AC specifications (with the exception of those for the TAP signals and AP IC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct Pentium processor 75/90/100/120/133 operation.

Table 16. Pentium® Processor 815\100, 1110\133 AC Specifications for 66-MHz Bus Operation
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	33.33	66.6	MHz		
t_{1a}	CLK Period	15.0	30.0	nS	4	
t_{1b}	CLK Period Stability		±250	pS		Adjacent Clocks(1,25)
t_2	CLK High Time	4.0		nS	4	@2V(1)
t_3	CLK Low Time	4.0		nS	4	@0.8V(1)
t_4	CLK Fall Time	0.15	1.5	nS	5	(2.0V–0.8V)(1)
t_5	CLK Rise Time	0.15	1.5	nS	4	(0.8V–2.0V)(1)
t_{6a}	ADSC#, PWT, PCD, BE0-7#, D/C#, W/R#, CACHE#, SCYC Valid Delay	1.0	7.0	nS	5	
t_{6b}	AP Valid Delay	1.0	8.5	nS	5	
t_{6c}	LOCK# Valid Delay	1.1	7.0	nS	5	
t_{6d}	ADS#, Valid Delay	1.0	6.0	nS	5	
t_{6e}	A3-A31 Valid Delay	1.1	6.3	nS	5	
t_{6f}	M/IO# Valid Delay	1.0	5.9	nS	5	
t_7	ADS#, ADSC#, AP, A3-A31, PWT, PCD, BE0-7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	6	1
t_{8a}	APCHK#, IERR#, FERR# Valid Delay	1.0	8.3	nS	5	4
t_{8b}	PCHK# Valid Delay	1.0	7.0	nS	5	4
t_{9a}	BREQ Valid Delay	1.0	8.0	nS	5	4
t_{9b}	SMIACT# Valid Delay	1.0	7.3	nS	5	4
t_{9c}	HLDA Valid Delay	1.0	6.8	nS	5	
t_{10a}	HIT# Valid Delay	1.0	6.8	nS	5	
t_{10b}	HITM# Valid Delay	1.1	6.0	nS	5	
t_{11a}	PM0-1, BP0-3 Valid Delay	1.0	10.0	nS	5	
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	5	
t_{12}	D0-D63, DP0-7 Write Data Valid Delay	1.3	7.5	nS	5	
t_{13}	D0-D63, DP0-3 Write Data Float Delay		10.0	nS	6	1
t_{14}	A5-A31 Setup Time	6.0		nS	7	26
t_{15}	A5-A31 Hold Time	1.0		nS	7	

Table 16. Pentium® Processor 815\100, 1110\133 AC Specifications for 66-MHz Bus Operation (Contd.)
3.135 < V_{CC} < 3.6V, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{16a}	INV, AP Setup Time	5.0		nS	7	
t_{16b}	EADS# Setup Time	5.0		nS	7	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	7	
t_{18a}	KEN# Setup Time	5.0		nS	7	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	7	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	7	
t_{20}	BRDY#, BRDYC# Setup Time	5.0		nS	7	
t_{21}	BRDY#, BRDYC# Hold Time	1.0		nS	7	
t_{22}	AHOLD, BOFF# Setup Time	5.5		nS	7	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	7	
t_{24a}	BUSCHK#, EWBE#, HOLD Setup Time	5.0		nS	7	
t_{24b}	PEN# Setup Time	4.8		nS	7	
t_{25a}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	7	
t_{25b}	HOLD Hold Time	1.5		nS	7	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	7	12, 16
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	7	13
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	7	12, 16, 17
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	7	13
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs		15, 17
t_{31}	R/S# Setup Time	5.0		nS	7	12, 16, 17
t_{32}	R/S# Hold Time	1.0		nS	7	13
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs		15, 17
t_{34}	D0-D63, DP0-7 Read Data Setup Time	2.8		nS	7	
t_{35}	D0-D63, DP0-7 Read Data Hold Time	1.5		nS	7	
t_{36}	RESET Setup Time	5.0		nS	8	11, 12, 16

Table 16. Pentium® Processor 815\100, 1110\133 AC Specifications for 66-MHz Bus Operation (Contd.)
3.135 < V_{CC} < 3.6V, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{37}	RESET Hold Time	1.0		nS	8	11, 13
t_{38}	RESET Pulse Width, V_{CC} & CL K Stable	15.0		CLKs	8	11, 17
t_{39}	RESET Active After V_{CC} & CLK Stable	1.0		mS	8	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time	5.0		nS	8	12, 16, 17
t_{41}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Hold Time	1.0		nS	8	13
t_{42a}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#) Setup Time, Async.	2.0		CLKs	8	To RESET falling edge(16)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, FRCMC#, BRDYC#, BUSCHK#) Hold Time, Async.	2.0		CLKs	8	To RESET falling edge(27)
t_{42c}	Reset Configuration Signals (BRDYC#, BUSCHK#) Setup Time, Async.	3.0		CLKs	8	To RESET falling edge(27)
t_{42d}	Reset Configuration Signal BRDYC# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge(1,27)
t_{43a}	BF, CPUTYP Setup Time	1.0		mS	8	To RESET falling edge(22)
t_{43b}	BF, CPUTYP Hold Time	2.0		CLKs	8	To RESET falling edge(22)
t_{43c}	APICEN Setup Time	2.0		CLKs	8	To RESET falling edge
t_{43d}	APICEN Hold Time	2.0		CLKs	8	To RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		nS	4	
t_{46}	TCK High Time	25.0		nS	4	@2V(1)
t_{47}	TCK Low Time	25.0		nS	4	@0.8V(1)
t_{48}	TCK Fall Time		5.0	nS	4	(2.0V–0.8V)(1,8,9)
t_{49}	TCK Rise Time		5.0	nS	4	(0.8V–2.0V)(1,8,9)
t_{50}	TRST# Pulse Width	40.0		nS	10	Asynchronous(1)
t_{51}	TDI, TMS Setup Time	5.0		nS	9	7

Table 16. Pentium® Processor 815\100, 1110\133 AC Specifications for 66-MHz Bus Operation (Contd.)
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{52}	TDI, TMS Hold Time	13.0		nS	9	7
t_{53}	TDO Valid Delay	3.0	20.0	nS	9	8
t_{54}	TDO Float Delay		25.0	nS	9	1, 8
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	9	3, 8, 10
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	9	1, 3, 8, 10
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	9	3, 7, 10
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	9	3, 7, 10
APIC AC Specifications						
t_{60a}	PICCLK Frequency	2.0	16.66	MHz		
t_{60b}	PICCLK Period	60.0	500.0	nS	4	
t_{60c}	PICCLK High Time	15.0		nS	4	
t_{60d}	PICCLK Low Time	15.0		nS	4	
t_{60e}	PICCLK Rise Time	0.15	2.5	nS	4	
t_{60f}	PICCLK Fall Time	0.15	2.5	nS	4	
t_{60g}	PICD0-1 Setup Time	3.0		nS	7	To PICCLK
t_{60h}	PICD0-1 Hold Time	2.5		nS	7	To PICCLK
t_{60i}	PICD0-1 Valid Delay (LtoH)	4.0	38.0	nS	5	From PICCLK(28,29)
t_{60j}	PICD0-1 Valid Delay (HtoL)	4.0	22.0	nS	5	From PICCLK(28,29)

Table 17. Pentium® Processor 815\100, 1110\133 Dual Processor Mode AC Specifications for 66-MHz Bus Operation

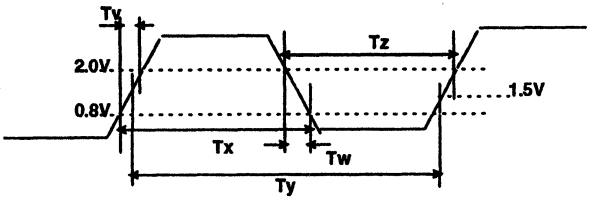
$3.135 < V_{CC} < 3.6V$, T_{CASE} = 0 to 70°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{80}	PBREQ#, PBGNT#, PHIT#, PHITM# Flight Time	0	2.0	nS	11	30
t_{83a}	A5-A31 Setup Time	3.7		nS	7	18, 21, 26
t_{83b}	D/C#, W/R#, CACHE#, LOCK#, SCYC Setup Time					
t_{83c}	ADS#, M/IO# Setup Time	5.8		nS	7	18, 21
t_{83d}	HIT#, HITM# Setup Time	6.0		nS	7	18, 21
t_{83e}	HLDA Setup Time	6.0		nS	7	18, 21
t_{84}	ADS#, D/C#, W/R#, M/IO#, CACHE#, LOCK#, A5-A31, HLDA, HIT#, HITM#, SCYC Hold Time	1.0		nS	7	18, 21
t_{85}	DPEN# Valid Time		10.0	CLKs		18, 19, 23
t_{86}	DPEN# Hold Time	2.0		CLKs		18, 20, 23
t_{87}	APIC ID (BE0#-BE3#) Setup Time	2.0		CLKs	8	To RESET falling edge(23)
t_{88}	APIC ID (BE0#-BE3#) Hold Time	2.0		CLKs	8	From RESET falling edge(23)
t_{89}	D/P# Valid Delay	1.0	8.0	nS	5	Primary Processor Only

NOTES:
Notes 2, 6, and 14 are general and apply to all standard TTL signals used with the Pentium® Processor family.
1. Not 100% tested. Guaranteed by design/characterization.
2. TTL input test waveforms are assumed to be 0 to 3V transitions with 1V/nS rise and fall times.
3. Non-test outputs and inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch-free outputs. Glitch-free signals monotonically transition without false transitions (i.e., glitches).
5. 0.8V/ns ≤ CLK input rise/fall time ≤ 8V/ns.
6. 0.3V/ns ≤ input rise/fall time ≤ 5V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 33 MHz.
10. During probe mode operation, do not use the boundary scan timings ($t_{55\text{-}58}$).
11. FRCMC# should be tied to V_{CC} (high) to ensure proper operation of the Pentium processor 75/90/100/120/133 as a primary processor.
12. Setup time is required to guarantee recognition on a specific clock. Pentium processor 75/90/100/120/133 must meet this specification for dual processor operation for the FLUSH# and RESET signals.

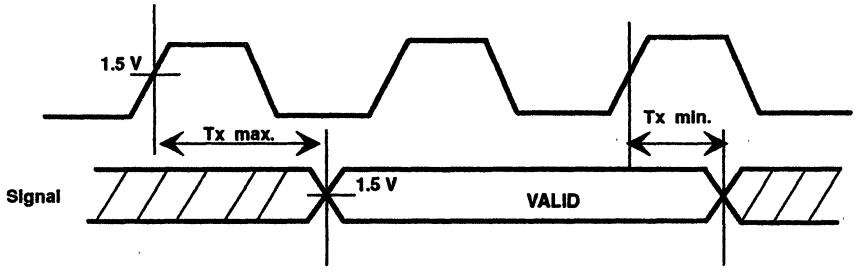
13. Hold time is required to guarantee recognition on a specific clock. Pentium processor 75/90/100/120/133 must meet this specification for dual processor operation for the FLUSH# and RESET signals.
14. All TTL timings are referenced from 1.5V.
15. To guarantee proper asynchronous recognition, the signal must have been de-asserted (inactive) for a minimum of 2 clocks before being returned active and must meet the minimum pulse width.
16. This input may be driven asynchronously. However, when operating two processors in dual processing mode, FLUSH# and RESET must be asserted synchronously to both processors.
17 When driven asynchronously, RESET, NMI, FLUSH#, R/S#, INIT, and SMI# must be de-asserted (inactive) for a minimum of 2 clocks before being returned active.
18. Timings are valid only when dual processor is present.
19. Maximum time DPEN# is valid from rising edge of RESET
20. Minimum time DPEN# is valid after falling edge of RESET.
21. The D/C#, M/IO#, W/R#, CACHE#, and A5-A31 signals are sampled only on the CLK that ADS# is active.
22. BF and CPUTYP should be strapped to V_{CC} or V_{SS}.
23. RESET is synchronous in dual processing mode and functional redundancy checking mode. All signals which have a setup or hold time with respect to a falling or rising edge of RESET in UP mode, should be measured with respect to the first processor clock edge in which RESET is sampled either active or inactive in dual processing and functional redundancy checking modes.
24. The PHIT# and PHITM# signals operate at the core frequency (75, 90, 100, 120 or 133 MHz).
25. These signals are measured on the rising edge of adjacent CLKs at 1.5V. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency. The amount of jitter present must be accounted for as a component of CLK skew between devices.
26. In dual processing mode, timing t_{14} is replaced by t_{83a}. Timing t_{14} is required for external snooping (e.g., address setup to the CLK in which EADS# is sampled active) in both uniprocessor and dual processor modes.
27. BRDYC# and BUSCHK# are used as reset configuration signals to select buffer size.
28. This assumes an external pullup resistor to V_{CC} and a lumped capacitive load such that the maximum RC product does not exceed R = 150Ω, C = 240 pF.
29. This assumes an external pullup resistor to V_{CC} and a lumped capacitive load such that the minimum RC product does not fall below R = 150Ω, C = 20 pF.
30. This is a flight time specification, that includes both flight time and clock skew. The flight time is the time from where the unloaded driver crosses 1.5V (50% of min V_{CC}), to where the receiver crosses the 1.5V level (50% of min V_{CC}). See Figure 11.

** Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.



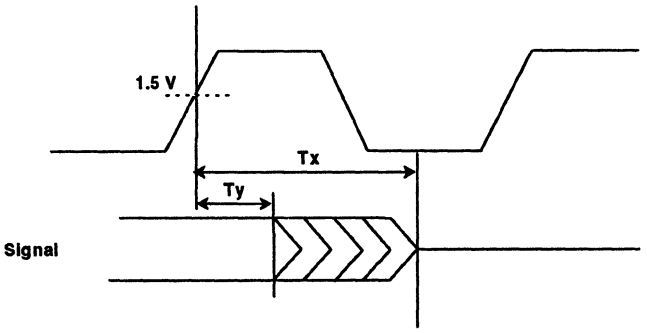
Tv = t5, t49, t60e; Tw = t4, t48, t60f; Tx = t3, t47, t60d
Ty = t1, t45, t60b; Tz = t2, t46, t60c

Figure 4. Clock Waveform



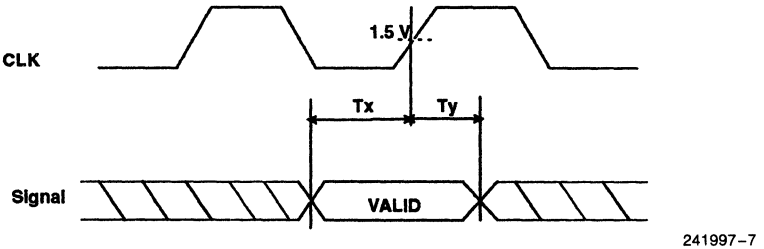
Tx = t6, t8, t9, t10, t11, t12, t60i, t80, t89

Figure 5. Valid Delay Timings



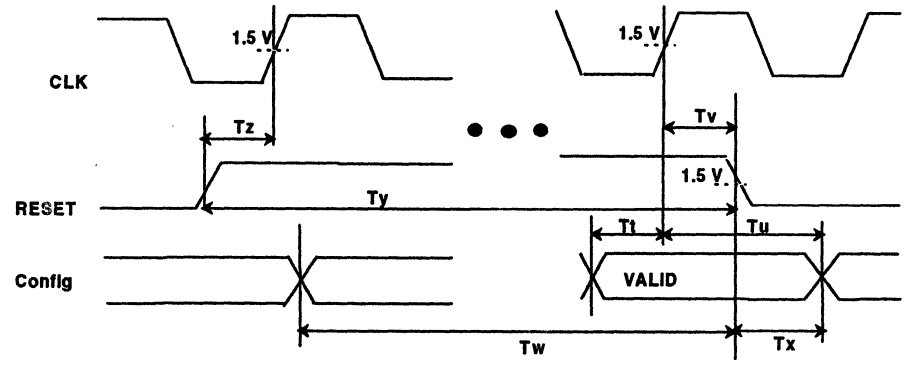
Tx = t7, t13; Ty = t6min, t12min

Figure 6. Float Delay Timings



Tx = t14, t16, t18, t20, t22, t24, t26, t28, t31, t34, t60g (to PICCLK),t81, t83
Ty = t15, t17, t19, t21, t23, t25, t27, t29, t32, t35, t60h (to PICCLK), t82, t84

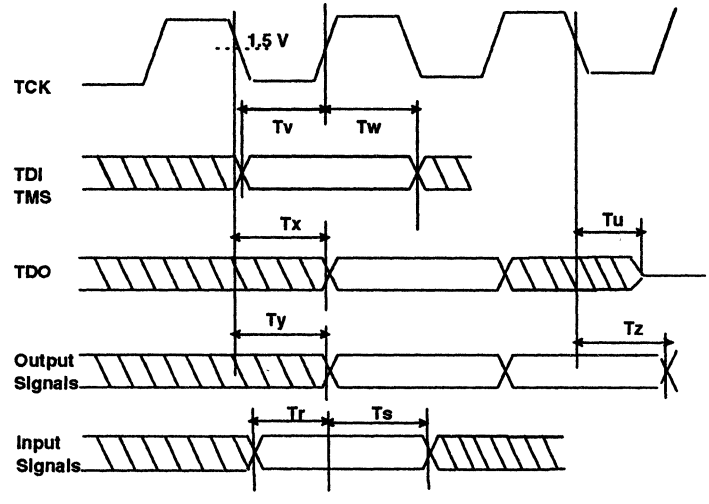
Figure 7. Setup and Hold Timings



241997-8

Tt = t40, Tu = t41, Tv = t37, T w =t42, t43a, t43c, t87, Tx = t43b, t43d, t88, Ty = t38, t39, Tz = t36

Figure 8. Reset and Configuration Timings



241997-9

Tr = t57, Ts = t58, Tu = t54, Tv = t51, Tw = t52, Tx = t53, Ty = t55, Tz = t56

Figure 9. Test Timings

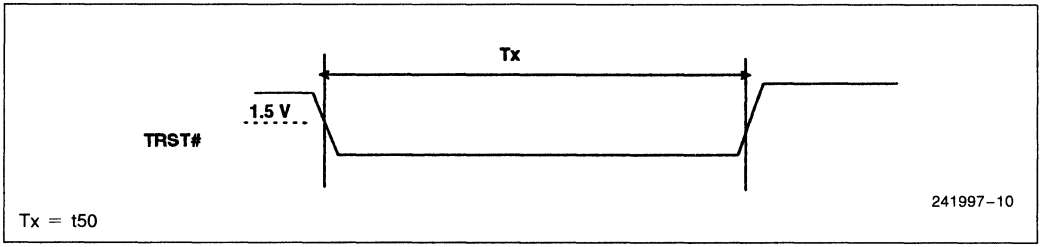


Figure 10. Test Reset Timings

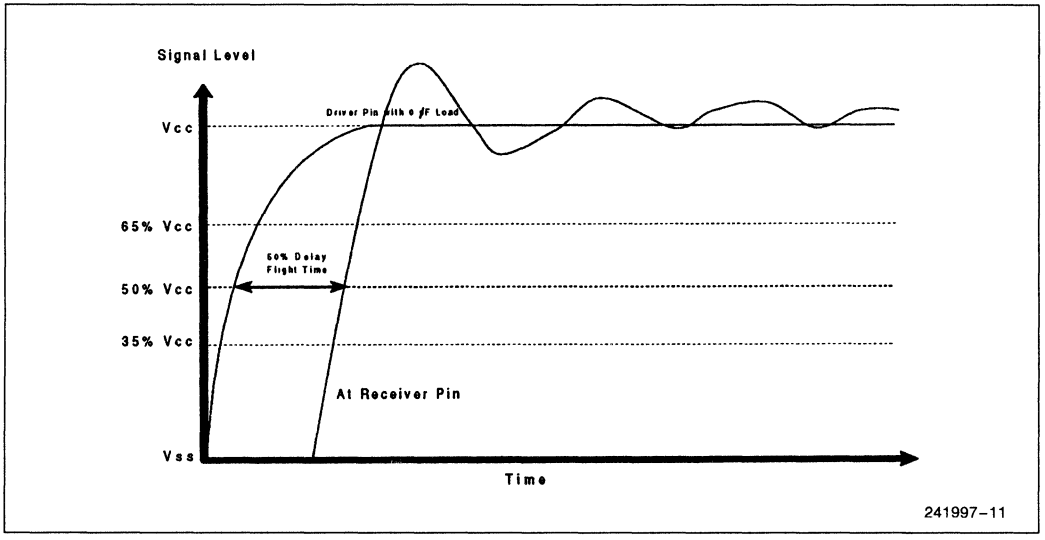


Figure 11. 50% V_{CC} Measurement of Flight Time

4.0 MECHANICAL SPECIFICATIONS

The Pentium processor 75/90/100/120/133 is packaged in a 296-pin staggered pin grid array package. The pins are arranged in a 37 x 37 matrix and the package dimensions are 1.95″ x 1.95″ (Table 18). A 1.25″ x 1.25″ copper tungsten heat spreader may be attached to the top of the ceramic. This package design with spreader is being phased out in favor of a package which has no attached spreader. In this section, both spreader and non-spreader packages are shown.

The mechanical specifications for the Pentium processor 75/90/100/120/133 are provided in Table 19. Figure 12 shows the package dimensions.

Table 18. Package Information Summary

	Package Type	Total Pins	Pin Array	Package Size
Pentium® Processor 75/90/100/120/133	SPGA	296	37 x 37	1.95″ x 1.95″ 4.95 cm x 4.95 cm

Table 19. Package Dimensions with Spreader

Family: Ceramic Staggered Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A	3.91	4.70	Solid Lid	0.154	0.185	Solid Lid
A1	0.33	0.43	Solid Lid	0.013	0.017	Solid Lid
A2	2.62	2.97		0.103	0.117	
B	0.43	0.51		0.017	0.020	
D	49.28	49.91		1.940	1.965	
D1	45.47	45.97		1.790	1.810	
D2	31.50	32.00	Square	1.240	1.260	Square
D3	33.99	34.59		1.338	1.362	
D4	8.00	9.91		0.315	0.390	
E1	2.41	2.67		0.095	0.105	
E2	1.14	1.40		0.045	0.055	
F		0.127	Diagonal		0.005	Diagonal
L	3.05	3.30		0.120	0.130	
N	296			296		
S1	1.52	2.54		0.060	0.100	

Table 20. Package Dimensions without Spreader

Family: 296-Pin Ceramic Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A	3.27	3.83	Ceramic Lid	0.129	0.151	Ceramic Lid
A1	0.66	0.86	Ceramic Lid	0.026	0.034	Ceramic Lid
A2	2.62	2.97		0.103	0.117	
B	0.43	0.51		0.017	0.020	
D	49.28	49.78		1.940	1.960	
D1	45.59	45.85		1.795	1.805	
D3	24.00	24.25	Includes Fillet	0.945	0.955	Includes Fillet
e1	2.29	2.79		0.090	0.110	
F		0.127	Flatness of the top of the package, measured diagonally		0.005	Flatness of the top of the package, measured diagonally
L	3.05	3.30		0.120	1.130	
N	296		Total Pins	296		Total Pins
S1	1.52	2.54		0.060	0.100	

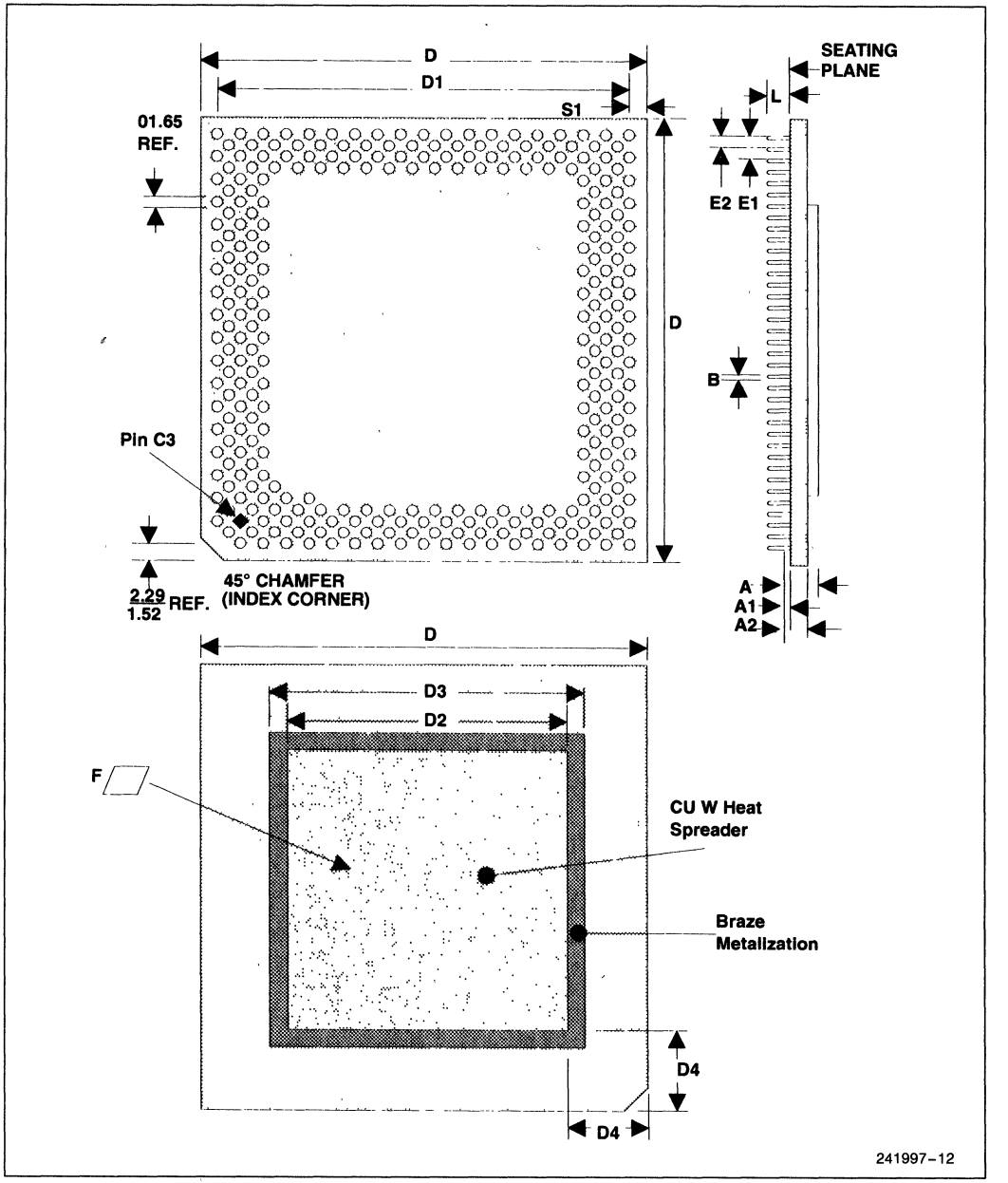


Figure 12. Pentium® Processor 75/90/100/120/133 Package Dimensions (with Spreader)

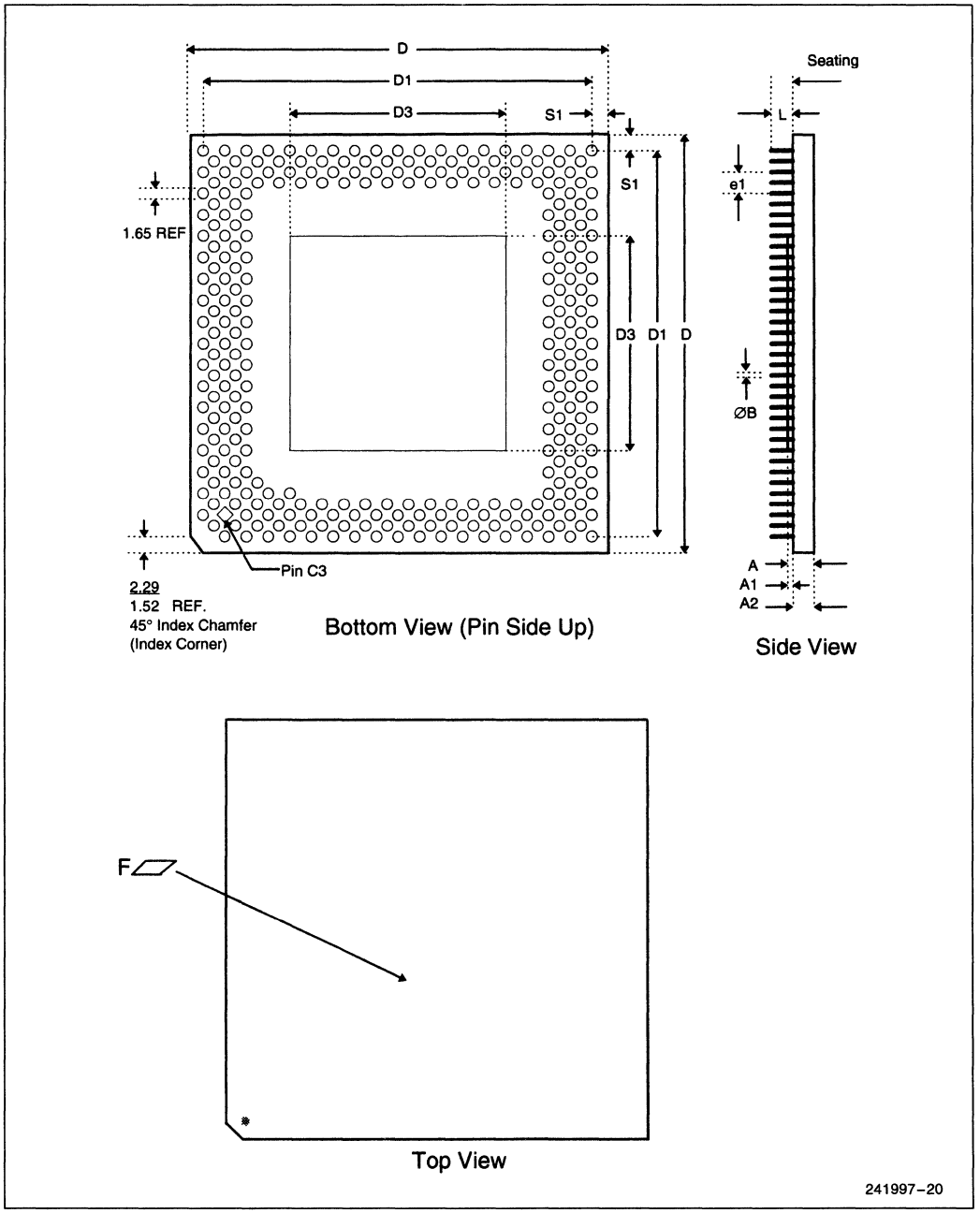


Figure 13. Pentium® Processor 75/90/100/120/133 Package Dimensions (without Spreader)

5.0 THERMAL SPECIFICATIONS

Due to the advanced 3.3V BiCMOS process that it is produced on, the Pentium processor 75/90/100/120/133 dissipates less power than the Pentium processor 60/66.

The Pentium processor 75/90/100/120/133 is specified for proper operation when case temperature, T_{CASE}, (T_C) is within the specified range of 0°C to 70°C.

5.1 Measuring Thermal Values

To verify that the proper T_C (case temperature) is maintained, it should be measured at the center of the package top surface (opposite of the pins). The measurement is made in the same way with or without a heat sink attached. When a heat sink is attached a hole (smaller than 0.150″ diameter) should be drilled through the heat sink to allow probing the center of the package. See Figure 13 for an illustration of how to measure T_C.

To minimize the measurement errors, it is recommended to use the following approach:

- Use 36-gauge or finer diameter K, T, or J type thermocouples. The laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using high thermal conductivity cements. The laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90-degree angle as shown in Figure 14.
- The hole size should be smaller than 0.150″ in diameter.

5.1.1 THERMAL EQUATIONS AND DATA

For the Pentium processor 75/90/100/120/133, an ambient temperature, T_A (air temperature around the processor), is not specified directly. The only restriction is that T_C is met. To calculate T_A values, the following equations may be used:

$$T_A = T_C - (P * \theta_{CA})$$

$$\theta_{CA} = \theta_{JA} - \theta_{JC}$$

where:

T_A and T_C = ambient and case temperature. (°C)

θ_{CA} = case-to-ambient thermal resistance. (°C/Watt)

θ_{JA} = junction-to-ambient thermal resistance. (°C/Watt)

θ_{JC} = junction-to-case thermal resistance. (°C/Watt)

P = maximum power consumption (Watt)

Table 21 lists the θ_{CA} values for the Pentium processor 75/90/100/120/133 with passive heat sinks. Figure 15 shows Table 21 in graphic format.

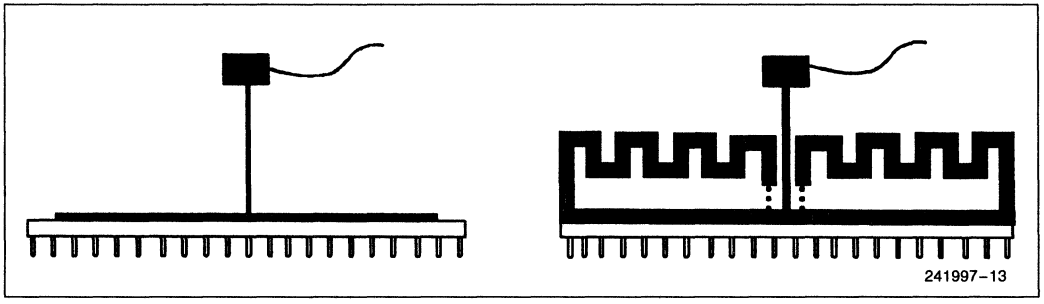


Figure 14. Technique for Measuring TC*

*Though the figure shows the package with a heat spreader, the same technique applies to measuring TC of the package without a heat spreader.

Table 21. Thermal Resistances for Packages with Spreader

Heat Sink in Inches	θ_{JC} (°C/Watt)	θ_{CA}(°C/Watt) vs. Laminar Airflow (linear ft/min)					
		0	100	200	400	600	800
1.95x1.95x0.25	0.9	8.7	7.6	6.2	4.0	3.2	2.6
1.95x1.95x0.35	0.9	8.4	7.1	5.6	3.6	2.9	2.4
1.95x1.95x0.45	0.9	8.0	6.6	4.9	3.2	2.5	2.1
1.95x1.95x0.55	0.9	7.7	6.1	4.3	2.8	2.2	1.9
1.95x1.95x0.65	0.9	7.3	5.6	3.9	2.6	2.0	1.7
1.95x1.95x0.80	0.9	6.6	4.9	3.5	2.2	1.8	1.6
1.95x1.95 x1.00	0.9	5.9	4.2	3.2	2.2	1.7	1.4
1.95x1.95x1.20	0.9	5.5	3.9	2.9	2.0	1.6	1.4
1.95x1.95x1.40	0.9	5.0	3.5	2.6	1.8	1.5	1.3
Without Heat Sink	1.4	11.4	10.5	8.7	5.7	4.5	3.8

2

Table 22. Thermal Resistances for Packages without Spreader

Heat Sink in Inches	θ_{JC} (°C/Watt)	θ_{CA}(°C/Watt) vs. Laminar Airflow (linear ft/min)					
		0	100	200	400	600	800
0.25	0.8	9.1	8.0	6.6	4.4	3.6	3.0
0.35	0.8	8.8	7.5	6.0	4.0	3.3	2.8
0.45	0.8	8.4	7.0	5.3	3.6	2.9	2.5
0.55	0.8	8.1	6.5	4.7	3.2	2.6	2.3
0.65	0.8	7.7	6.0	4.3	3.0	2.4	2.1
0.80	0.8	7.0	5.3	3.9	2.8	2.2	2.0
1.00	0.8	6.3	4.6	3.6	2.6	2.1	1.8
1.20	0.8	5.9	4.3	3.3	2.4	2.0	1.8
1.40	0.8	5.4	3.9	3.0	2.2	1.9	1.7
Without Heat Sink	1.3	14.4	13.1	11.7	8.8	7.4	6.5

NOTES:
Heat sinks are omni directional pin aluminum alloy.
Features were based on standard extrusion practices for a given height
Pin size ranged from 50 to 129 mils
Pin spacing ranged from 93 to 175 mils
Based thickness ranged from 79 to 200 mils
Heat sink attach was 0.005" of thermal grease.
Attach thickness of 0.002" will improve performance approximately 0.3°C/Watt

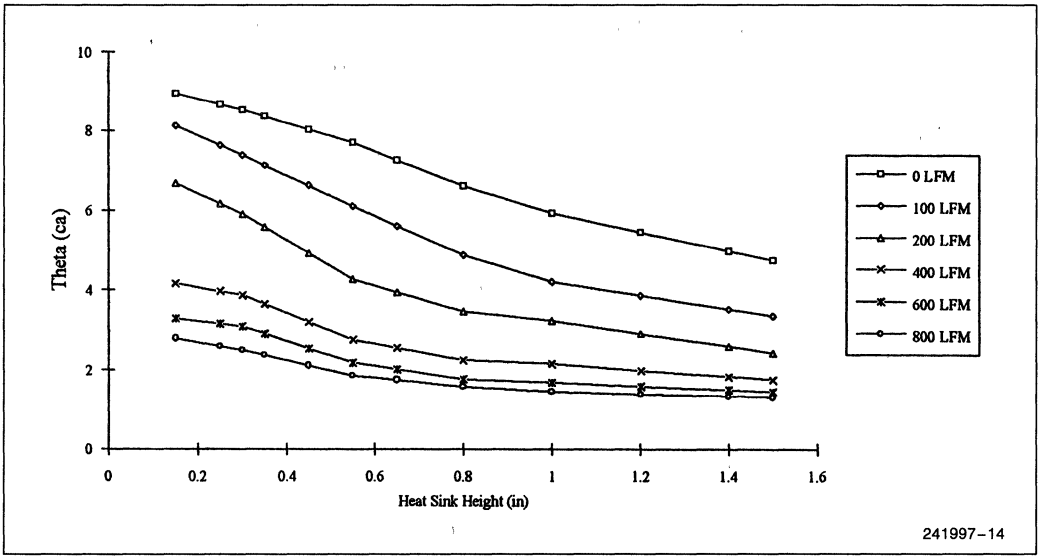


Figure 15. Thermal Resistance vs. Heatsink Height (Spreader Package)

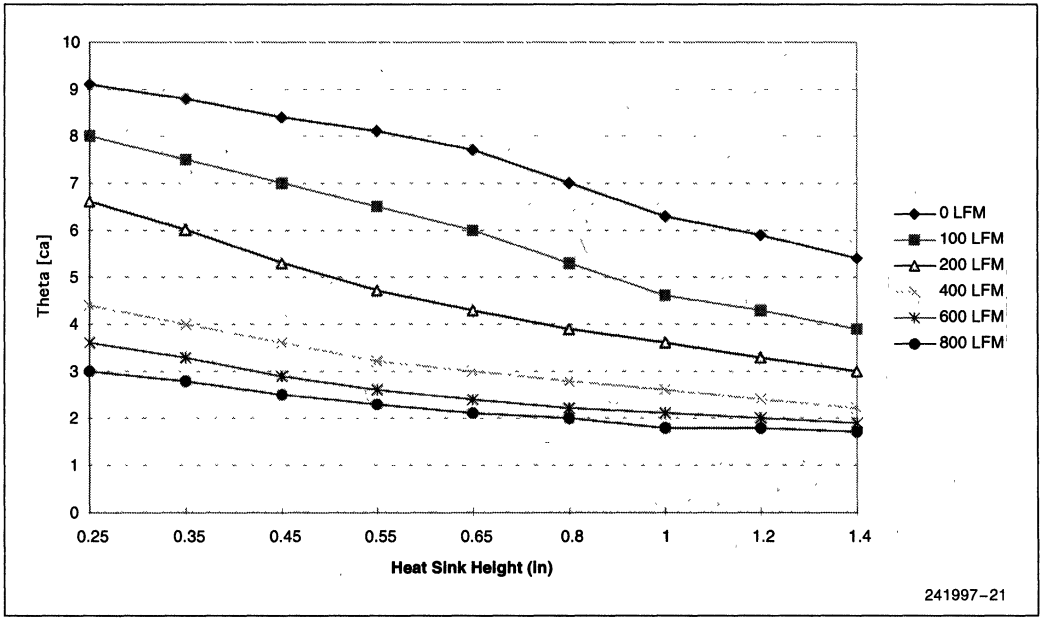


Figure 16. Thermal Resistance vs. Heatsink Height (Non-Spreader Package)

6.0 FUTURE PENTIUM® OverDrive® PROCESSOR SOCKET SPECIFICATION

6.1 Introduction

The future OverDrive processors are end-user single-chip CPU upgrade products for Pentium processor-based systems. The future OverDrive processors will speed up most software applications by 40% to 70% and are binary compatible with the Pentium processor.

Two upgrade sockets have been defined for the Pentium processor-based system as part of the processor architecture. Socket 5 has been defined for the Pentium processor (610\75, 735\90, 815\100, 1000\120)-based systems. Upgradability can be supported by implementing either a single socket or a dual socket strategy. A single socket system will include a 320-pin SPGA Socket 5. When this system configuration is upgraded, the Pentium processor is simply replaced by the future OverDrive processor. A dual socket system will include a 296-pin SPGA socket for the Pentium processor and a 320-pin SPGA Socket 5 for the second processor. In dual socket systems, Socket 5 can be filled with either the Dual processor or the future OverDrive processors. The rest of this section describes the Socket 5 specifications.

Socket 7 has been defined as the upgrade socket for the Pentium processor (1110\133) in addition to the Pentium processor (610\75, 735\90, 815\100, 1000\120). The flexibility of the Socket 7 definition makes it backward compatible with Socket 5 and should be used for all new Pentium processor-based system designs. The Socket 7 support requires key changes from Socket 5 designs; split voltage planes, voltage regulator module header, 3.3V clocks, BIOS updates, additional decoupling, etc. This information is not provided in this datasheet. Contact Intel for further information regarding the Socket 7 specifications.

6.1.1 UPGRADE OBJECTIVES

Systems using the Pentium processor (610\75, 735\90, 815\100, 1000\120), and equipped with only one processor socket, must use Socket 5 to accept the future OverDrive processor. Systems equipped with two processor sockets must use Socket 5 as the second socket to contain either the Pentium processor Dual processor or the future OverDrive processor.

Inclusion of Socket 5 in Pentium processor systems provides the end-user with an easy and cost-effective way to increase system performance. The majority of upgrade installations which take advantage of Socket 5 will be performed by end users and resellers. Therefore, it is important that the design be "end-user easy," and that the amount of training and technical expertise required to install the upgrade processors be minimized. Upgrade installation instructions should be clearly described in the system user's manual. In addition, by making installation simple and foolproof, PC manufacturers can reduce the risk of system damage, warranty claims and service calls. The three main characterizations of end user easy designs are:

- Accessible socket location
- Clear indication of upgrade component orientation
- Minimization of insertion force

The future OverDrive processor will support all Intel chip sets that are supported by the Pentium processor, including 82430NX PCIset and 82430FX PCIset.

6.1.2 INTEL PLATFORM SUPPORT LABS

The Intel Platform Support Labs ensure that Pentium processor (610\75, 735\90, 815\100, 1000\120) and Pentium processor (1110\133)-based personal computers meet design criteria for reliable and straightforward CPU upgradability with the future OverDrive processor. Evaluation performed at the Intel Platforms Support Labs confirms that Pentium processor and future OverDrive processor specifications for mechanical, thermal, electrical, functional, and end-user installation attributes have been met.

The OEM submits motherboard and system designs to one of Intel's worldwide Platform Support Labs for evaluation. The OEM benefits from engineering feedback on Pentium processor and future OverDrive processor support. Contact your local Intel representative for more information on the Intel Platform Support Labs.

6.2 Future Pentium® OverDrive® Processor (Socket 5) Pinout

This section contains pinouts of the future Pentium OverDrive Socket (Socket 5) when used as a single-socket turbo upgrade.

6.2.1 PIN DIAGRAMS

6.2.1.1 Socket 5 Pinout

For systems with a single socket for the Pentium processor 75/90/100/120/133 and future Pentium OverDrive processor, the following pinout *must* be followed for the single socket location. Note that to be Intel Verified for a future Pentium OverDrive processor, this must be a ZIF socket. The socket footprint contains V_{CC}, V_{CC5}, and V_{SS} pins that are internal no connects on the Pentium processor 75/90/100/120/133. These pins *must* be connected to the appropriate PCB power and ground layers to ensure future Pentium OverDrive processor compatibility.

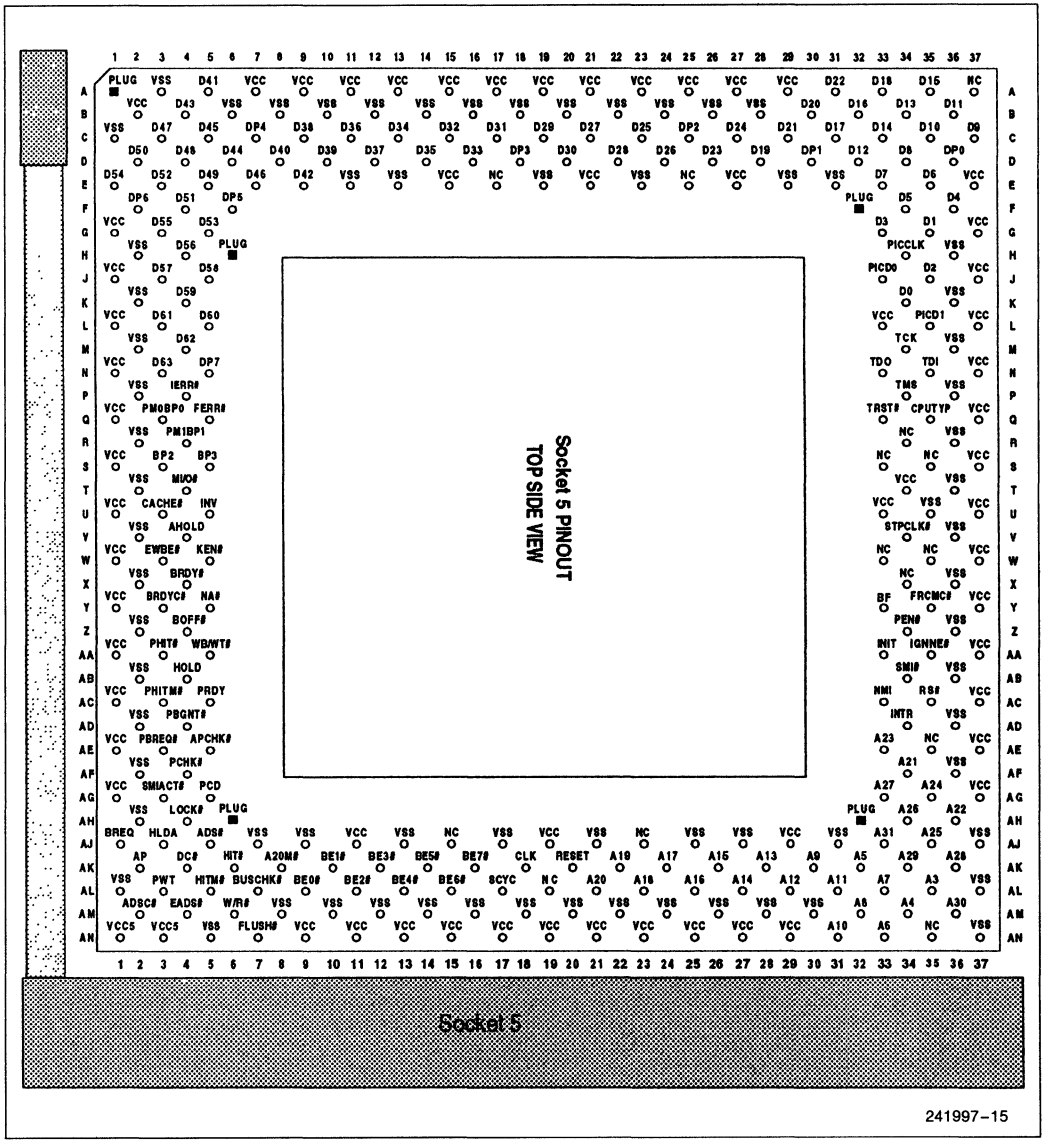


Figure 17. Socket 5 Pinout Top Side View

NOTE:

The "Socket 5 PINOUT TOP SIDE VIEW" text orientation on the top side view drawing in this section represents the orientation of the ink mark on the actual packages. (Note that the text shown in this section is not the actual text which will be marked on the packages).

Table 23. Pentium® Processor 75/90/100/120/133 vs. Socket 5 Pins

Pentium® Processor 75/90/100/120/133 Signal	Socket 5 Signal	Pin Number
INC	V_{SS}	A03
D/P#	NC	AE35
NO PIN	V_{SS}	AJ07
NO PIN	V_{SS}	AJ09
NO PIN	V_{CC}	AJ11
NO PIN	V_{SS}	AJ13
NO PIN	NC	AJ15
NO PIN	V_{SS}	AJ17
NO PIN	V_{CC}	AJ19
NO PIN	V_{SS}	AJ21
NO PIN	NC	AJ23
NO PIN	V_{SS}	AJ25
NO PIN	V_{SS}	AJ27
NO PIN	V_{CC}	AJ29
NO PIN	V_{SS}	AJ31
INC	V_{CC5}	AN01
INC	V_{CC5}	AN03
INC	V_{CC}	B02
NO PIN	V_{SS}	E11
NO PIN	V_{SS}	E13
NO PIN	V_{CC}	E15
NO PIN	NC	E17
NO PIN	V_{SS}	E19
NO PIN	V_{CC}	E21
NO PIN	V_{SS}	E23
NO PIN	NC	E25
NO PIN	V_{CC}	E27
NO PIN	V_{SS}	E29
NO PIN	V_{SS}	E31

NOTE:
All INCs are internal no connects. These signals are guaranteed to remain internally not connected in the Pentium processor 75/90/100/120/133.

6.3 Electrical Specifications

The future Pentium OverDrive processor will have the same AC specifications, power and ground specifications and decoupling recommendations as the Pentium processor 75/90/100/120/133.

6.3.1 V_{CC5} PIN DEFINITION

The future Pentium OverDrive processor pinout contains two 5V V_{CC} pins (V_{CC5}) used to provide power to the fan/heatsink. These pins should be connected to +5V ±5% regardless of the system design. Failure to connect V_{CC5} to 5V may cause the component to shut down.

6.4 Absolute Maximum Ratings of Upgrade

The on-chip Voltage Regulation and fan/heatsink devices included with the future Pentium OverDrive processor require different stress ratings than the Pentium processor 75/90/100/120/133. The voltage regulator is surface mounted on the future Pentium OverDrive processor and is, therefore, an integral part of the assembly. The future Pentium OverDrive processor storage temperature ratings are tightened as a result. The fan is a detachable unit, and the storage temperature is stated separately in the table below. Functional operation of the future Pentium OverDrive processor remains 0°C to 70°C.

Future Pentium® OverDrive® Processor and Voltage Regulator Assembly

Storage Temperature . . . . . −30°C to 100°C

Case Temperature Under Bias . . . . . −30°C to 100°C

Fan

Storage Temperature . . . . . −30°C to 75°C

Case Temperature Under Bias . . . . . −30°C to 75°C

** WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

6.4.1 DC SPECIFICATIONS

The future Pentium OverDrive processor will have compatible DC specifications to the Pentium processor 75/90/100/120/133, except that I_{CC3} (Power Supply Current), I_{CC5} (Fan/Heatsink Current), and V_{CC} are the following:

Table 24. Future Pentium® OverDrive® Processor I_{CC} Specification

V_{CC5} = 5V ±5%, T_{CASE} = 0 to 70°C

Symbol	Parameter	Min	Max	Unit
I_{CC3}(1)	Power Supply Current		4330	mA
I_{CC5}	Fan/Heatsink Current		200	mA

NOTE:
1. V_{CC} = 3.135V to 3.6V

6.5 Mechanical Specifications

The future Pentium OverDrive processor will be packaged in a 320-pin ceramic staggered pin grid array (SPGA). The pins will be arranged in a 37 x 37 matrix and the package dimensions will be 1.95" x 1.95" (4.95 cm x 4.95 cm).

Table 25. Processor Package Information Summary

	Package Type	Total Pins	Pin Array	Package Size
Future Pentium® OverDrive® Processor	SPGA	320	37 x 37	1.95" x 1.95" 4.95 cm x 4.95 cm

Table 26. Future Pentium® OverDrive® Processor Package Dimensions

Family: Ceramic Staggered Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A*		33.88	Solid Lid		1.334	Solid Lid
A1	0.33	0.43	Solid Lid	0.013	0.017	Solid Lid
A2	2.62	2.97		0.103	0.117	
A4		20.32			0.800	
A5	10.16		Air Space	0.400		Air Space
B	0.43	0.51		0.017	0.020	
D	49.28	49.91		1.940	1.965	
D1	45.47	45.97		1.790	1.810	
E1	2.41	2.67		0.095	0.105	
E2	1.14	1.40		0.045	0.055	
L	3.05	3.30		0.120	0.130	
N	320		SPGA pins	320		SPGA pins
S1	1.52	2.54		0.060	0.100	

NOTES:
* Assumes the minimum air space above the fan/heatsink
A 0.2″ clearance around three of four sides of the package is also required to allow free airflow through the fan/heatsink.

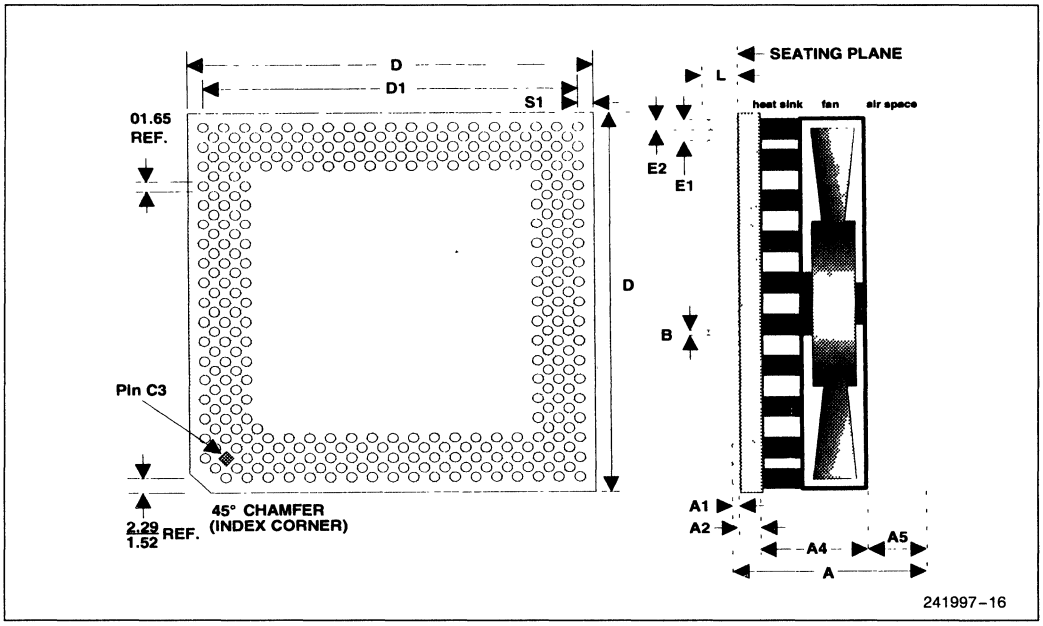


Figure 18. Future Pentium® OverDrive® Processor Package Dimensions

6.6 Thermal Specifications

The future Pentium OverDrive processor will be cooled with a fan/heatsink cooling solution. The future Pentium OverDrive processor with a fan/heatsink is specified for proper operation when T_A (air temperature entering the fan/heatsink) is a maximum of 45°C. When the $T_A(max) \leq 45°C$ specification is met, the fan/heatsink will keep T_C (case temperature) within the specified range of 0°C to 70°C provided airflow through the fan/heatsink is unimpeded.

6.7 Upgradability with Socket 5/ Socket 7

6.7.1 INTRODUCTION

- Built-in upgradability for Pentium processor 75/90/100/120 based systems with Socket 5
  - — Supports the future Pentium OverDrive processor 320-pin socket (Socket 5)
- Built-in upgradability for Pentium processor 133 based systems with Socket 7
  - — Supports the future Pentium OverDrive processor Socket 7

6.7.2 SOCKET 5 VENDORS

OEMs should contact Intel for the most current list of Intel-qualified socket vendors. For a complete list of Qualified Sockets and Vendor Order Numbers, call the Intel Faxback number for your geographical area and have document #7209 automatically faxed to you. Figure 19 shows preliminary dimensions for AMP and Yamaichi sockets. OEMs should directly contact the socket vendors for the most current socket information. Figure 20 shows the upgrade processor's orientation in Socket 5.

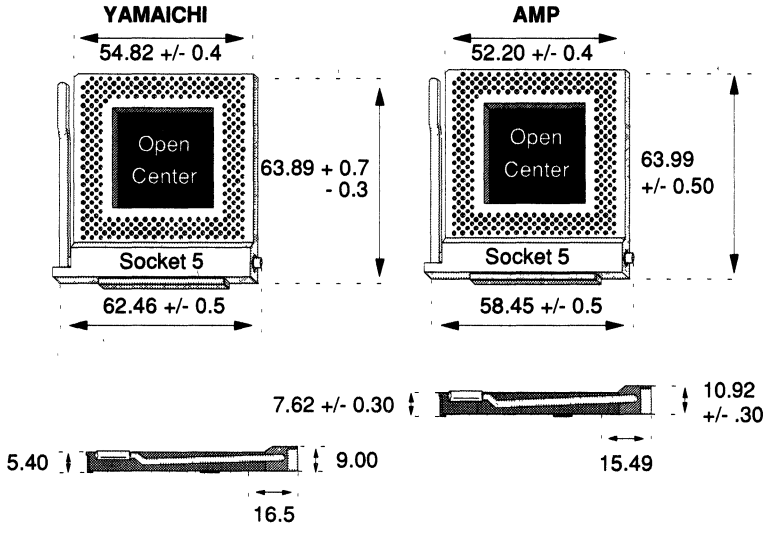
To order Socket 5 from AMP and Yamaichi, the phone numbers and part numbers are as follows:

AMP: 1-800-522-6752
part#: 916513-1

Yamaichi: 1-800-769-0797
part#: NP210-320K13625

6.7.3 SOCKET 7

Socket 7 information is not provided in this datasheet. Contact Intel for further information regarding Socket 7 specifications.



NOTE:
All dimensions are in mm.

Figure 19. Socket 5 Footprint Dimensions

WARNING:
See socket manufacturer for the most current information.

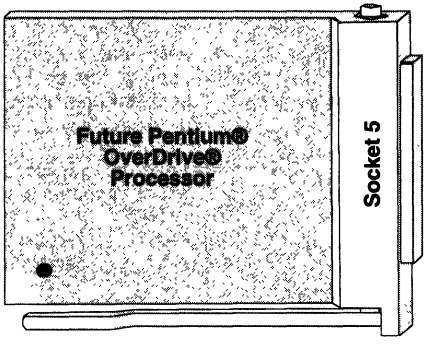


Figure 20. Socket 5 Chip Orientation

6.8 Testability

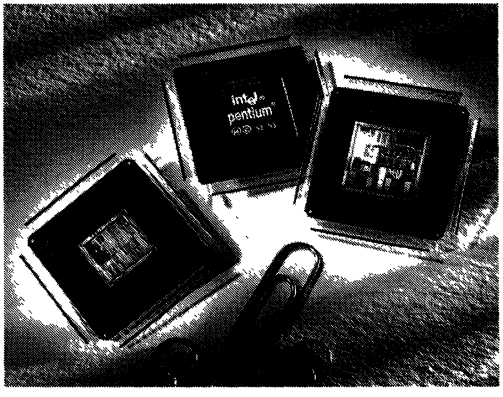
6.8.1 BOUNDARY SCAN

The future Pentium OverDrive processor supports the IEEE Standard 1149.1 boundary scan using the Test Access Port (TAP) and TAP Controller. The boundary scan register for the future Pentium OverDrive processor contains a cell for each pin. The turbo upgrade component will have a different bit order than the Pentium processor 75/90/100/120/133. If the TAP port on your system will be used by an end user following installation of the future Pentium OverDrive processor, please contact Intel for the bit order of the upgrade processor boundary scan register.

PENTIUM® PROCESSORS AT iCOMP® INDEX 1000\120, 735\90, 610\75 MHz WITH VOLTAGE REDUCTION TECHNOLOGY

- **Compatible with Large Software Base**
  - **MS-DOS‡, Windows‡, OS/2‡, UNIX‡**
- **32-Bit CPU with 64-Bit Data Bus**
- **Superscalar Architecture**
  - **Two Pipelined Integer Units Are Capable of 2 Instructions/Clock**
  - **Pipelined Floating Point Unit**
- **Separate Code and Data Caches**
  - **8K Code, 8K Writeback Data**
  - **MESI Cache Protocol**
- **Advanced Design Features**
  - **Branch Prediction**
  - **Virtual Mode Extensions**
- **Low Voltage BiCMOS Silicon Technology**
- **4M Pages for Increased TLB Hit Rate**
- **IEEE 1149.1 Boundary Scan**
- **Internal Error Detection Features**
- **SL Enhanced Power Management Features**
  - **System Management Mode**
  - **Clock Control**
- **Voltage Reduction Technology**
  - **2.9V V_{CC} for core supply**
  - **3.3V V_{CC} for I/O buffer supply**
- **Fractional Bus Operation**
  - **75-MHz Core/50-MHz Bus**
  - **90-MHz Core/60-MHz Bus**
  - **120-MHz Core/60-MHz Bus**

The Pentium® processor is fully compatible with the entire installed base of applications for DOS‡, Windows‡, OS/2‡, and UNIX‡, and all other software that runs on any earlier Intel 8086 family product. The Pentium processor's superscalar architecture can execute two instructions per clock cycle. Branch prediction and separate caches also increase performance. The pipelined floating-point unit delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor with voltage reduction technology has 3.3 million transistors. It is built on Intel's advanced low voltage BiCMOS silicon technology, and has full SL Enhanced power management features, including System Management Mode (SMM) and clock control. The additional SL Enhanced features, 2.9V core operation along with 3.3V I/O buffer operation, and the option of the TCP package, which are not available in the Pentium processor (510\60, 567\66), make the Pentium processor with voltage reduction technology ideal for enabling mobile Pentium processor designs. The Pentium processor may contain design defects or errors known as errata. Current characterized errata are available upon request.



242557-1

‡Other brands and trademarks are the property of their respective owners.

October 1995
Order Number: 242557-002

PENTIUM® PROCESSORS AT iCOMP® INDEX 1000\120, 735\90, 610\75 MHz WITH VOLTAGE REDUCTION

CONTENTS

CONTENTS

CONTENTS

PAGE

CONTENTS

PAGE

1.0 INTRODUCTION

Intel is manufacturing a reduced power version of the latest Pentium® processor, the Pentium processor at iCOMP® index (610\75 MHz, 735\90 MHz, 1000\120 MHz) with voltage reduction technology, targeting the mobile market. Voltage reduction technology allows the processor to "talk" to industry standard 3.3V components while its inner core, operating at 2.9 volts, consumes less power to promote a longer battery life. The Pentium processor with voltage reduction technology is offered in the Tape Carrier Package (TCP) and the SPGA package. It has all the advanced features of the 3.3V Pentium except for the differences listed in sections 3.1 and 7.1.1.

The Pentium processor with voltage reduction technology has several features which allow high-performance notebooks to be designed with the Pentium processor, including the following:

- The Pentium processor with voltage reduction technology TCP package dimensions are ideal for small form-factor designs.
- The Pentium processor with voltage reduction technology TCP package has superior thermal resistance characteristics.
- 2.9V core and 3.3V I/O buffer V_{CC} inputs reduce power consumption significantly, while maintaining 3.3V compatibility externally.
- The SL Enhanced feature set, which was initially implemented in the Intel486™ CPU.

The architecture and internal features of the Pentium processor with voltage reduction technology (TCP and SPGA) are identical to those of the Pentium processor (610\75, 735\90, 815\100, 1000\120), except several features not used in mobile applications have been eliminated to streamline it for mobile applications. The TCP Pentium processor with voltage reduction technology specifications are detailed in section 3 to section 6. All SPGA Pentium processor with voltage reduction technology specifications, with the exception of the differences described in section 7, are identical to the Pentium processor (610\75, 735\90, 815\100, 1000\120) specifications provided in the *Pentium® Processor Family Developer's Manual, Volume 1: Pentium® Processors*.

This document should be used in conjunction with the Pentium processor documents listed below.

List of related documents:

- *Pentium® Processor Family Developer's Manual, Vol. 1: Pentium® Processors* (Order Number: 241428)
- *Pentium® Processor Family Developer's Manual, Vol. 3: Architecture and Programming Manual* (Order Number: 241430)

2.0 MICROPROCESSOR ARCHITECTURE OVERVIEW

The Pentium processor with voltage reduction technology extends the Intel Pentium family of microprocessors. It is compatible with the 8086/88, 80286, Intel386™ DX, Intel386 SX, Intel486™ DX, Intel486 DX2, Intel486 SX and the Pentium processors at iCOMP® Index 510\60 MHz, 567\66 MHz, 610\75 MHz, 735\90 MHz and 815\100 MHz.

The Pentium processor family consists of the Pentium processor with voltage reduction technology described in this document, the original Pentium processor (510\60, 567\66), and the Pentium processor (610\75, 735\90, 815\100, 1000\120). "Pentium processor" will be used in this document to refer to the entire Pentium processor family in general.

The Pentium processor family architecture contains all of the features of the Intel486 CPU family, and provides significant enhancements and additions including the following:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Improved Instruction Execution Time
- Separate 8K Code and 8K Data Caches
- Writeback MESI Protocol in the Data Cache
- 64-Bit Data Bus
- Bus Cycle Pipelining
- Address Parity
- Internal Parity Checking
- Execution Tracing
- Performance Monitoring
- IEEE 1149.1 Boundary Scan
- System Management Mode
- Virtual Mode Extensions

2.1 Pentium® Processor Family Architecture

The application instruction set of the Pentium processor family includes the complete Intel486 CPU family instruction set with extensions to accommodate some of the additional functionality of the Pentium processors. All application software written for the Intel386 and Intel486 family microprocessors will run on the Pentium processors without modification. The on-chip memory management unit (MMU) is completely compatible with the Intel386 family and Intel486 family of CPUs.

The Pentium processors implement several enhancements to increase performance. The two instruction pipelines and floating-point unit on Pentium processors are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, two floating-point instructions) in one clock.

Branch prediction is implemented in the Pentium processors. To support this, Pentium processors implement two prefetch buffers, one to prefetch code in a linear fashion, and one that prefetches code according to the Branch Target Buffer (BTB) so the needed code is almost always prefetched before it is needed for execution.

The floating-point unit has been completely redesigned over the Intel486 CPU. Faster algorithms provide up to 10X speed-up for common operations including add, multiply and load.

Pentium processors include separate code and data caches integrated on-chip to meet performance goals. Each cache is 8 Kbytes in size, with a 32-byte line size and is 2-way set associative. Each cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to physical addresses. The data cache is configurable to be writeback or writethrough on a line-by-line basis and follows the MESI protocol. The data cache tags are triple ported to support two data transfers and an inquire cycle in the same clock. The code cache is an inherently write-protected cache. The code cache tags are also triple ported to support snooping and split line accesses. Individual pages can be configured as cacheable or non-cacheable by software or hardware. The caches can be enabled or disabled by software or hardware.

The Pentium processors have increased the data bus to 64 bits to improve the data transfer rate. Burst read and burst writeback cycles are supported by the Pentium processors. In addition, bus cycle pipelining has been added to allow two bus cycles to be in progress simultaneously. The Pentium processors' Memory Management Unit contains optional extensions to the architecture which allow 2-Mbyte and 4-Mbyte page sizes.

The Pentium processors have added significant data integrity and error detection capability. Data parity checking is still supported on a byte-by-byte basis. Address parity checking and internal parity checking features have been added along with a new exception, the machine check exception.

As more and more functions are integrated on chip, the complexity of board level testing is increased. To address this, the Pentium processors have increased test and debug capability. The Pentium processors implement IEEE Boundary Scan (Standard 1149.1). In addition, the Pentium processors have specified four breakpoint pins that correspond to each of the debug registers and externally indicate a breakpoint match. Execution tracing provides external indications when an instruction has completed execution in either of the two internal pipelines, or when a branch has been taken.

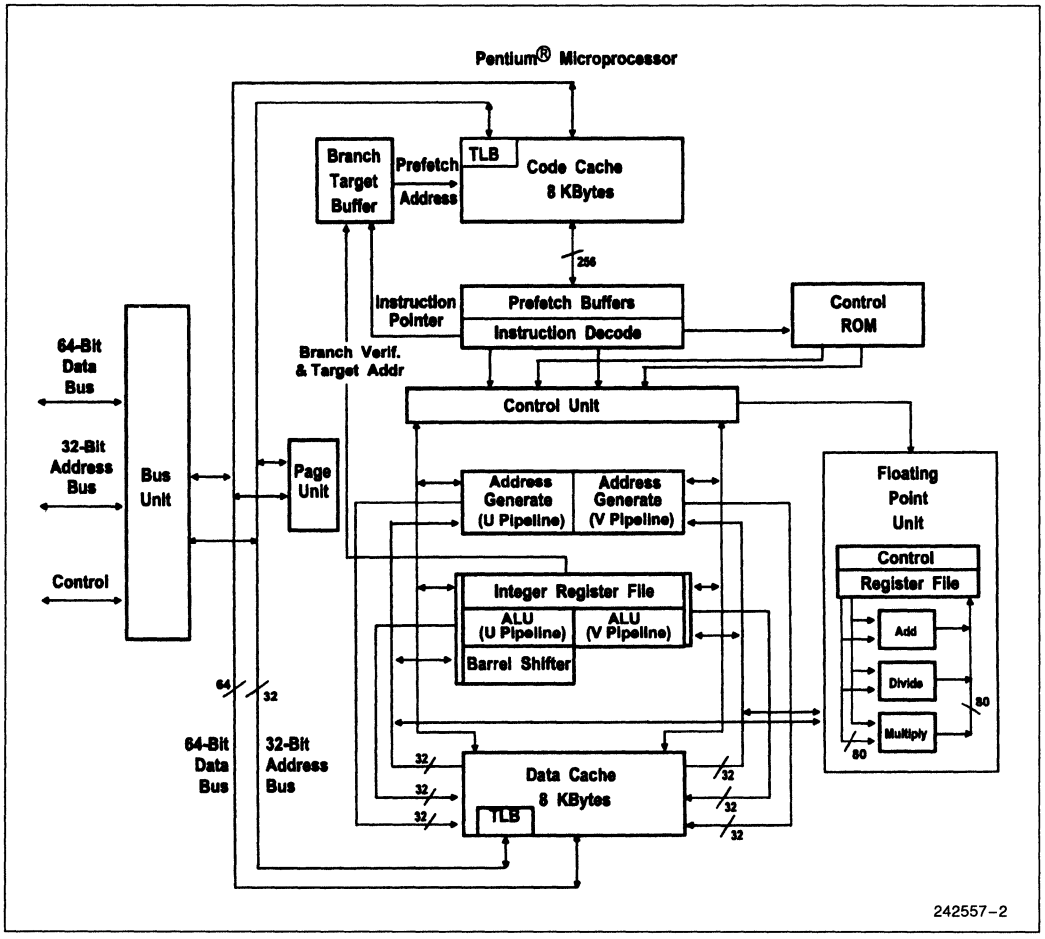


Figure 1. Pentium® Processor Block Diagram

System Management Mode (SMM) has been implemented along with some extensions to the SMM architecture. Enhancements to the virtual 8086 mode have been made to increase performance by reducing the number of times it is necessary to trap to a virtual 8086 monitor.

The block diagram shows the two instruction pipelines, the "u" pipe and "v" pipe. The u-pipe can execute all integer and floating point instructions. The v-pipe can execute simple integer instructions and the FXCH floating-point instructions.

The separate caches are shown, the code cache and data cache. The data cache has two ports, one for each of the two pipes (the tags are triple ported to allow simultaneous inquire cycles). The data cache has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium processor. Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so the Pentium processor can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that must be performed to implement the Pentium processor architecture. The control ROM unit has direct control over both pipelines.

The Pentium processors contain a pipelined floating-point unit that provides a significant floating-point performance advantage over previous generations of processors.

The architectural features introduced in this section are more fully described in the *Pentium® Processor Family Developer's Manual, Volume 3.*

3.0. TCP PINOUT

3.1. Pentium® Processor with Voltage Reduction Technology Differences from the SPGA 3.3V Pentium Processor

To better streamline the part for mobile applications, the following features have been eliminated for the TCP and SPGA Pentium processor with voltage reduction technology: Upgrade, Dual Processing (DP), APIC and Master/Checker functional redundancy. Table 1 lists the corresponding pins which exist on the SPGA 3.3V Pentium processor but have been removed on the TCP and SPGA Pentium processor with voltage reduction technology.

Table 1. Signals Removed in Pentium® Processor with Voltage Reduction Technology

Signal	Function
ADSC#	Additional Address Status. This signal is mainly used for large or standalone L2 cache memory subsystem support required for high-performance desktop or server models.
BRDYC#	Additional Burst Ready. This signal is mainly used for large or standalone L2 cache memory subsystem support required for high-performance desktop or server models.
CPUTYP	CPU Type. This signal is used for dual processing systems.
D/P#	Dual/Primary processor identification. This signal is only used for an Upgrade processor.
FRCMC#	Functional Redundancy Checking. This signal is only used for error detection via processor redundancy, and requires two Pentium® processors (master/checker).
PBGNT#	Private Bus Grant. This signal is only used for dual processing systems.
PBREQ#	Private Bus Request. This signal is used only for dual processing systems.
PHIT#	Private Hit. This signal is only used for dual processing systems.
PHITM#	Private Modified Hit. This signal is only used for dual processing systems.
PICCLK	APIC Clock. This signal is the APIC interrupt controller serial data bus clock.
PICD0 [DPEN#]	APIC's Programmable Interrupt Controller Data line 0. PICD0 shares a pin with DPEN# (Dual Processing Enable).
PICD1 [APICEN]	APIC's Programmable Interrupt Controller Data line 1. PICD1 shares a pin with APICEN (APIC Enable (on RESET)).

3.2. TCP Pinout and Pin Descriptions

3.2.1. PENTIUM® PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY TCP PINOUT

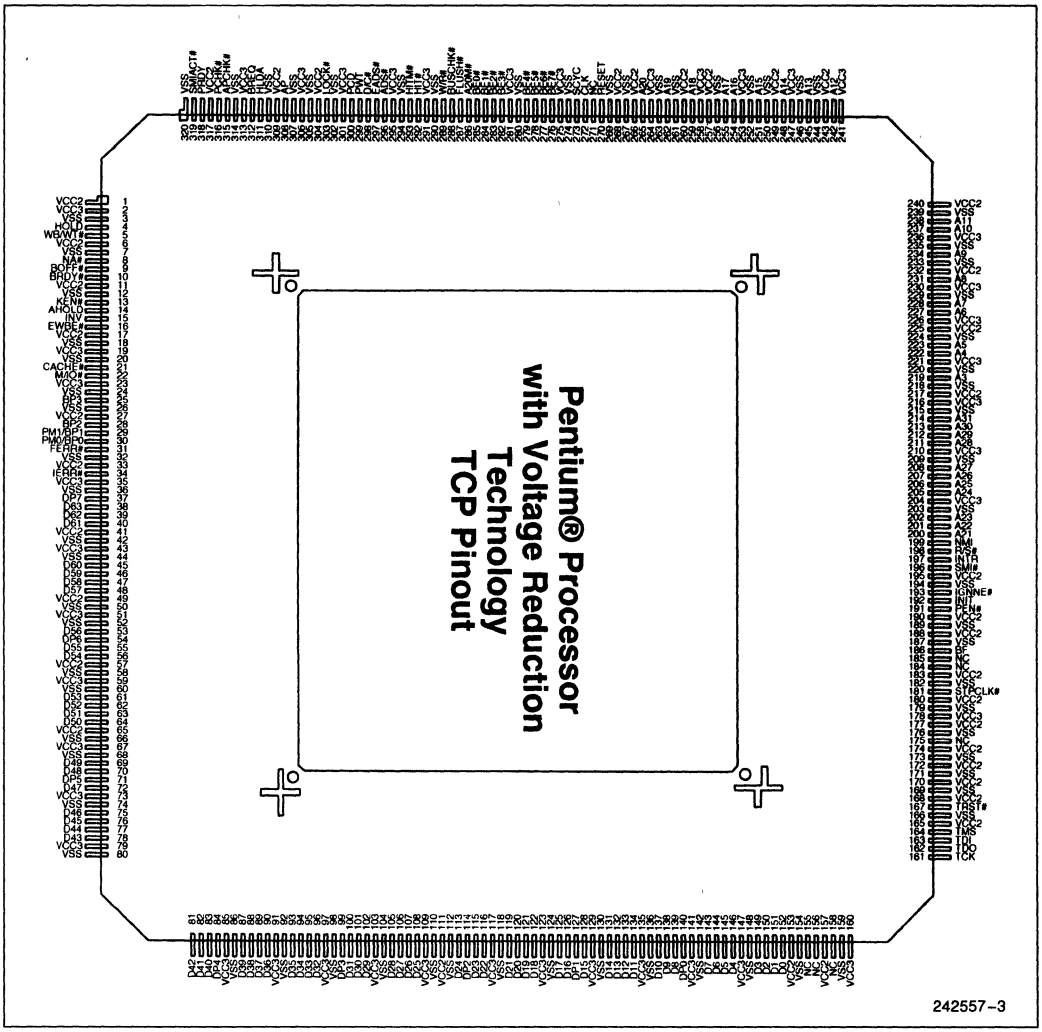


Figure 2. Pentium® Processor with Voltage Reduction Technology TCP Pinout

3.2.2. TCP PIN CROSS REFERENCE TABLE FOR PENTIUM® PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY

Table 2. TCP Pin Cross Reference by Pin Name

Address									
A3	219	A9	234	A15	251	A21	200	A27	208
A4	222	A10	237	A16	254	A22	201	A28	211
A5	223	A11	238	A17	255	A23	202	A29	212
A6	227	A12	242	A18	259	A24	205	A30	213
A7	228	A13	245	A19	262	A25	206	A31	214
A8	231	A14	248	A20	265	A26	207		
Data									
D0	152	D13	132	D26	107	D39	87	D52	62
D1	151	D14	131	D27	106	D40	83	D53	61
D2	150	D15	128	D28	105	D41	82	D54	56
D3	149	D16	126	D29	102	D42	81	D55	55
D4	146	D17	125	D30	101	D43	78	D56	53
D5	145	D18	122	D31	100	D44	77	D57	48
D6	144	D19	121	D32	96	D45	76	D58	47
D7	143	D20	120	D33	95	D46	75	D59	46
D8	139	D21	119	D34	94	D47	72	D60	45
D9	138	D22	116	D35	93	D48	70	D61	40
D10	137	D23	115	D36	90	D49	69	D62	39
D11	134	D24	113	D37	89	D50	64	D63	38
D12	133	D25	108	D38	88	D51	63		

Table 2. TCP Pin Cross Reference by Pin Name (Continued)

Control							
A20M#	286	BREQ	312	HITM#	293	PM1/BP1	29
ADS#	296	BUSCHK#	288	HLDA	311	PRDY	318
AHOLD	14	CACHE#	21	HOLD	4	PWT	299
AP	308	D/C#	298	IERR#	34	R/S#	198
APCHK#	315	DP0	140	IGNNE#	193	RESET	270
BE0#	285	DP1	127	INIT	192	SCYC	273
BE1#	284	DP2	114	INTR/LINT0	197	SMI#	196
BE2#	283	DP3	99	INV	15	SMIACT#	319
BE3#	282	DP4	84	KEN#	13	TCK	161
BE4#	279	DP5	71	LOCK#	303	TDI	163
BE5#	278	DP6	54	M/IO#	22	TDO	162
BE6#	277	DP7	37	NA#	8	TMS	164
BE7#	276	EADS#	297	NMI/LINT1	199	TRST#	167
BOFF#	9	EWBE#	16	PCD	300	W/R#	289
BP2	28	FERR#	31	PCHK#	316	WB/WT#	5
BP3	25	FLUSH#	287	PEN#	191		
BRDY#	10	HIT#	292	PM0/BP0	30		
				Clock Control			
				BF	186	STPCLK#	181
				CLK	272		

Table 2. TCP Pin Cross Reference by Pin Name (Continued)

$V_{CC}2^*$			
1	111	183	257
6	153	188	260
11	157	190	266
17	165	195	268
27	168	217	304
33	170	225	309
41	172	232	317
49	174	240	
57	177	243	
65	180	249	
$V_{CC}3^{**}$			
2	91	178	258
19	97	204	264
23	103	210	275
35	109	216	281
43	117	221	291
51	123	226	295
59	129	230	301
67	135	236	306
73	141	241	313
79	147	247	
85	160	253	

NOTE:
*These $V_{CC}2$ pins are 2.9V inputs for the TCP Pentium® processor with voltage reduction technology, but may change to a different voltage on future offerings of this microprocessor family.
**All $V_{CC}3$ pins will remain at 3.3V power inputs for TCP Pentium processor.

Table 2. TCP Pin Cross Reference by Pin Name (Continued)

V_{SS}						
3	50	104	166	209	250	302
7	52	110	169	215	252	305
12	58	112	171	218	256	307
18	60	118	173	220	261	310
20	66	124	176	224	263	314
24	68	130	179	229	267	320
26	74	136	182	233	269	
32	80	142	187	235	274	
36	86	148	189	239	280	
42	92	154	194	244	290	
44	98	159	203	246	294	
NC						
155	156	158	175	184	185	271

Table 3. TCP Pin Cross Reference by Pin Number (Pins 1-136)

Pin #	Signal	Pin #	Signal	Pin #	Signal	Pin #	Signal
1	VCC2	35	VCC3	69	D49	103	VCC3
2	VCC3	36	VSS	70	D48	104	VSS
3	VSS	37	DP7	71	DP5	105	D28
4	HOLD	38	D63	72	D47	106	D27
5	WB/WT#	39	D62	73	VCC3	107	D26
6	VCC2	40	D61	74	VSS	108	D25
7	VSS	41	VCC2	75	D46	109	VCC3
8	NA#	42	VSS	76	D45	110	VSS
9	BOFF#	43	VCC3	77	D44	111	VCC2
10	BRDY#	44	VSS	78	D43	112	VSS
11	VCC2	45	D60	79	VCC3	113	D24
12	VSS	46	D59	80	VSS	114	DP2
13	KEN#	47	D58	81	D42	115	D23
14	AHOLD	48	D57	82	D41	116	D22
15	INV	49	VCC2	83	D40	117	VCC3
16	EWBE#	50	VSS	84	DP4	118	VSS
17	VCC2	51	VCC3	85	VCC3	119	D21
18	VSS	52	VSS	86	VSS	120	D20
19	VCC3	53	D56	87	D39	121	D19
20	VSS	54	DP6	88	D38	122	D18
21	CACHE#	55	D55	89	D37	123	VCC3
22	M/IO#	56	D54	90	D36	124	VSS
23	VCC3	57	VCC2	91	VCC3	125	D17
24	VSS	58	VSS	92	VSS	126	D16
25	BP3	59	VCC3	93	D35	127	DP1
26	VSS	60	VSS	94	D34	128	D15
27	VCC2	61	D53	95	D33	129	VCC3
28	BP2	62	D52	96	D32	130	VSS
29	PM1/BP1	63	D51	97	VCC3	131	D14
30	PM0/BP0	64	D50	98	VSS	132	D13
31	FERR#	65	VCC2	99	DP3	133	D12
32	VSS	66	VSS	100	D31	134	D11
33	VCC2	67	VCC3	101	D30	135	VCC3
34	IERR#	68	VSS	102	D29	136	VSS

Table 3. TCP Pin Cross Reference by Pin Number (Pins 137-272) (Continued)

Pin #	Signal	Pin #	Signal	Pin #	Signal	Pin #	Signal
137	D10	171	VSS	205	A24	239	VSS
138	D9	172	VCC2	206	A25	240	VCC2
139	D8	173	VSS	207	A26	241	VCC3
140	DP0	174	VCC2	208	A27	242	A12
141	VCC3	175	NC	209	VSS	243	VCC2
142	VSS	176	VSS	210	VCC3	244	VSS
143	D7	177	VCC2	211	A28	245	A13
144	D6	178	VCC3	212	A29	246	VSS
145	D5	179	VSS	213	A30	247	VCC3
146	D4	180	VCC2	214	A31	248	A14
147	VCC3	181	STPCLK#	215	VSS	249	VCC2
148	VSS	182	VSS	216	VCC3	250	VSS
149	D3	183	VCC2	217	VCC2	251	A15
150	D2	184	NC	218	VSS	252	VSS
151	D1	185	NC	219	A3	253	VCC3
152	D0	186	BF	220	VSS	254	A16
153	VCC2	187	VSS	221	VCC3	255	A17
154	VSS	188	VCC2	222	A4	256	VSS
155	NC	189	VSS	223	A5	257	VCC2
156	NC	190	VCC2	224	VSS	258	VCC3
157	VCC2	191	PEN#	225	VCC2	259	A18
158	NC	192	INIT	226	VCC3	260	VCC2
159	VSS	193	IGNNE#	227	A6	261	VSS
160	VCC3	194	VSS	228	A7	262	A19
161	TCK	195	VCC2	229	VSS	263	VSS
162	TDO	196	SMI#	230	VCC3	264	VCC3
163	TDI	197	INTR	231	A8	265	A20
164	TMS	198	R/S#	232	VCC2	266	VCC2
165	VCC2	199	NMI	233	VSS	267	VSS
166	VSS	200	A21	234	A9	268	VCC2
167	TRST#	201	A22	235	VSS	269	VSS
168	VCC2	202	A23	236	VCC3	270	RESET
169	VSS	203	VSS	237	A10	271	NC
170	VCC2	204	VCC3	238	A11	272	CLK

Table 3. TCP Pin Cross Reference by Pin Number (Pins 273-320) (Continued)

Pin #	Signal	Pin #	Signal	Pin #	Signal	Pin #	Signal
273	SCYC	285	BE0#	297	EADS#	309	VCC2
274	VSS	286	A20M#	298	D/C#	310	VSS
275	VCC3	287	FLUSH#	299	PWT	311	HLDA
276	BE7#	288	BUSCHK#	300	PCD	312	BREQ
277	BE6#	289	W/R#	301	VCC3	313	VCC3
278	BE5#	290	VSS	302	VSS	314	VSS
279	BE4#	291	VCC3	303	LOCK#	315	APCHK#
280	VSS	292	HIT#	304	VCC2	316	PCHK#
281	VCC3	293	HITM#	305	VSS	317	VCC2
282	BE3#	294	VSS	306	VCC3	318	PRDY
283	BE2#	295	VCC3	307	VSS	319	SMIACT#
284	BE1#	296	ADS#	308	AP	320	VSS

NOTE:
1. VCC2 pins are 2.9V power inputs to the core.
2. VCC3 pins are 3.3V power inputs to the core.

3.3. Design Notes

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to $V_{CC}3$. Unused active HIGH inputs should be connected to GND (V_{SS}).

No Connect (NC) pins must remain unconnected. Connection of NC pins may result in component failure or incompatibility with processor steppings.

3.4. Quick Pin Reference

This section gives a brief functional description of each of the pins. For a detailed description, see the "Hardware Interface" chapter in the *Pentium® Processor Family Developer's Manual, Volume 1.*

Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior.

The # symbol at the end of a signal name indicates that the active or asserted state occurs when the signal is at a low voltage. When a # symbol is not present after the signal name, the signal is active, or asserted at the high voltage level.

Table 4. Quick Pin Reference

Symbol	Type	Name and Function
A20M#	I	When the **address bit 20 mask** pin is asserted, the Pentium® processor emulates the address wraparound at 1 Mbyte which occurs on the 8086. When A20M# is asserted, the processor masks physical address bit 20 (A20) before performing a lookup to the internal caches or driving a memory cycle on the bus. The effect of A20M# is undefined in protected mode. A20M# must be asserted only when the processor is in real mode.
A31–A3	I/O	As outputs, the **address** lines of the processor along with the byte enables define the physical area of memory or I/O accessed. The external system drives the inquire address to the processor on A31–A5.
ADS#	O	The **address status** indicates that a new valid bus cycle is currently being driven by the processor.
AHOLD	I	In response to the assertion of **address hold**, the processor will stop driving the address lines (A31–A3), and AP in the next clock. The rest of the bus will remain active so data can be returned or driven for previously issued bus cycles.
AP	I/O	**Address parity** is driven by the processor with even parity information on all processor generated cycles in the same clock that the address is driven. Even parity must be driven back to the processor during inquire cycles on this pin in the same clock as EADS# to ensure that correct parity check status is indicated.
APCHK#	O	The **address parity check** status pin is asserted two clocks after EADS# is sampled active if the processor has detected a parity error on the address bus during inquire cycles. APCHK# will remain active for one clock each time a parity error is detected.
BE7#–BE5# BE4#–BE0#	O I/O	The **byte enable** pins are used to determine which bytes must be written to external memory, or which bytes were requested by the CPU for the current cycle. The byte enables are driven in the same clock as the address lines (A31–3).
BF	I	**Bus Frequency** determines the bus-to-core ratio. BF is sampled at RESET, and cannot be changed until another non-warm (1 ms) assertion of RESET. Additionally, BF must not change values while RESET is active. For proper operation of the processor, this pin should be strapped high or low. When BF is strapped to VCC, the processor will operate at a $\frac{2}{3}$ bus/core frequency ratio. When BF is strapped to VSS, the processor will operate to a $\frac{1}{2}$ bus/core frequency ratio. If BF is left floating, the processor defaults to a $\frac{2}{3}$ bus/core ratio.
BOFF#	I	The **backoff** input is used to abort all outstanding bus cycles that have not yet completed. In response to BOFF#, the processor will float all pins normally floated during bus hold in the next clock. The processor remains in bus hold until BOFF# is negated, at which time the Pentium processor restarts the aborted bus cycle(s) in their entirety.
BP[3:2] PM/BP[1:0]	O	The **breakpoint** pins (BP3–0) correspond to the debug registers, DR3–DR0. These pins externally indicate a breakpoint match when the debug registers are programmed to test for breakpoint matches. BP1 and BP0 are multiplexed with the **performance monitoring** pins (PM1 and PM0). The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.

Table 4. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
BRDY#	I	The **burst ready** input indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted the processor data in response to a write request. This signal is sampled in the T2, T12 and T2P bus states.
BREQ	O	The **bus request** output indicates to the external system that the processor has internally generated a bus request. This signal is always driven whether or not the processor is driving its bus.
BUSCHK#	I	The **bus check** input allows the system to signal an unsuccessful completion of a bus cycle. If this pin is sampled active, the processor will latch the address and control signals in the machine check registers. If, in addition, the MCE bit in CR4 is set, the processor will vector to the machine check exception.
CACHE#	O	For processor-initiated cycles, the **cache** pin indicates internal cacheability of the cycle (if a read), and indicates a burst writeback cycle (if a write). If this pin is driven inactive during a read cycle, the processor will not cache the returned data, regardless of the state of the KEN# pin. This pin is also used to determine the cycle length (number of transfers in the cycle).
CLK	I	The **clock** input provides the fundamental timing for the processor. Its frequency is the operating frequency of the processor external bus and requires TTL levels. All external timing parameters except TDI, TDO, TMS, TRST# and PICD0–1 are specified with respect to the rising edge of CLK. It is recommended that CLK begin 150 ms after V_{CC} reaches its proper operating level. This recommendation is only to assure the long term reliability of the device.
D/C#	O	The **data/code** output is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. D/C# distinguishes between data and code or special cycles.
D63–D0	I/O	These are the 64 **data lines** for the processor. Lines D7–D0 define the least significant byte of the data bus; lines D63–D56 define the most significant byte of the data bus. When the CPU is driving the data lines, they are driven during the T2, T12 or T2P clocks for that cycle. During reads, the CPU samples the data bus when BRDY# is returned.
DP7–DP0	I/O	These are the **data parity** pins for the processor. There is one for each byte of the data bus. They are driven by the processor with even parity information on writes in the same clock as write data. Even parity information must be driven back to the Pentium processor with voltage reduction technology on these pins in the same clock as the data to ensure that the correct parity check status is indicated by the processor. DP7 applies to D63–D56; DP0 applies to D7–D0.
EADS#	I	This signal indicates that a valid **external address** has been driven onto the processor address pins to be used for an inquire cycle.

Table 4. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
EWBE#	I	The **external write buffer empty** input, when inactive (high), indicates that a write cycle is pending in the external system. When the processor generates a write and EWBE# is sampled inactive, the processor will hold off all subsequent writes to all E- or M-state lines in the data cache until all write cycles have completed, as indicated by EWBE# being active.
FERR#	O	The **floating point error** pin is driven active when an unmasked floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS-type floating point error reporting.
FLUSH#	I	When asserted, the **cache flush** input forces the processor to write back all modified lines in the data cache and invalidate its internal caches. A Flush Acknowledge special cycle will be generated by the processor indicating completion of the writeback and invalidation. If FLUSH# is sampled low when RESET transitions from high to low, tristate test mode is entered.
HIT#	O	The **hit** indication is driven to reflect the outcome of an inquire cycle. If an inquire cycle hits a valid line in either the data or instruction cache, this pin is asserted two clocks after EADS# is sampled asserted. If the inquire cycle misses the cache, this pin is negated two clocks after EADS#. This pin changes its value only as a result of an inquire cycle and retains its value between the cycles.
HITM#	O	The **hit to a modified line** output is driven to reflect the outcome of an inquire cycle. It is asserted after inquire cycles which resulted in a hit to a modified line in the data cache. It is used to inhibit another bus master from accessing the data until the line is completely written back.
HLDA	O	The **bus hold acknowledge** pin goes active in response to a hold request driven to the processor on the HOLD pin. It indicates that the processor has floated most of the output pins and relinquished the bus to another local bus master. When leaving bus hold, HLDA will be driven inactive and the processor will resume driving the bus. If the processor has a bus cycle pending, it will be driven in the same clock that HLDA is de-asserted.
HOLD	I	In response to the **bus hold request**, the processor will float most of its output and input/output pins and assert HLDA after completing all outstanding bus cycles. The processor will maintain its bus in this state until HOLD is de-asserted. HOLD is not recognized during LOCK cycles. The processor will recognize HOLD during reset.
IERR#	O	The **internal error** pin is used to indicate internal parity errors. If a parity error occurs on a read from an internal array, the processor will assert the IERR# pin for one clock and then shutdown.

Table 4. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
IGNNE#	I	This is the **ignore numeric error** input. This pin has no effect when the NE bit in CR0 is set to 1. When the CR0.NE bit is 0, and the IGNNE# pin is asserted, the processor will ignore any pending unmasked numeric exception and continue executing floating-point instructions for the entire duration that this pin is asserted. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating-point instruction is one of FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the processor will execute the instruction in spite of the pending exception. When the CR0.NE bit is 0, IGNNE# is not asserted, a pending unmasked numeric exception exists (SW.ES = 1), and the floating-point instruction is one other than FINIT, FCLEX, FSTENV, FSAVE, FSTSW, FSTCW, FENI, FDISI, or FSETPM, the processor will stop execution and wait for an external interrupt.
INIT	I	The processor **initialization** input pin forces the processor to begin execution in a known state. The processor state after INIT is the same as the state after RESET except that the internal caches, write buffers, and floating point registers retain the values they had prior to INIT. INIT may NOT be used in lieu of RESET after power up. If INIT is sampled high when RESET transitions from high to low, the processor will perform built-in self test prior to the start of program execution.
INTR	I	An active **maskable interrupt** input indicates that an external interrupt has been generated. If the IF bit in the EFLAGS register is set, the processor will generate two locked interrupt acknowledge bus cycles and vector to an interrupt handler after the current instruction execution is completed. INTR must remain active until the first interrupt acknowledge cycle is generated to assure that the interrupt is recognized.
INV	I	The **invalidation** input determines the final cache line state (S or I) in case of an inquire cycle hit. It is sampled together with the address for the inquire cycle in the clock EADS# is sampled active.
KEN#	I	The **cache enable** pin is used to determine whether the current cycle is cacheable or not and is consequently used to determine cycle length. When the processor generates a cycle that can be cached (CACHE# asserted) and KEN# is active, the cycle will be transformed into a burst line fill cycle.
LOCK#	O	The **bus lock** pin indicates that the current bus cycle is locked. The processor will not allow a bus hold when LOCK# is asserted (but AHOLD and BOFF# are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the BRDY# is returned for the last locked bus cycle. LOCK# is guaranteed to be de-asserted for at least one clock between back-to-back locked cycles.
M/IO#	O	The **memory/input-output** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. M/IO# distinguishes between memory and I/O cycles.

Table 4. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
NA#	I	An active **next address** input indicates that the external memory system is ready to accept a new bus cycle although all data transfers for the current cycle have not yet completed. The processor will issue ADS# for a pending cycle two clocks after NA# is asserted. The processor supports up to two outstanding bus cycles.
NMI	I	The **non-maskable interrupt** request signal indicates that an external non-maskable interrupt has been generated.
PCD	O	The **page cache disable** pin reflects the state of the PCD bit in CR3; Page Directory Entry or Page Table Entry. The purpose of PCD is to provide an external cacheability indication on a page-by-page basis.
PCHK#	O	The **parity check** output indicates the result of a parity check on a data read. It is driven with parity status two clocks after BRDY# is returned. PCHK# remains low one clock for each clock in which a parity error was detected. Parity is checked only for the bytes on which valid data is returned.
PEN#	I	The **parity enable** input (along with CR4.MCE) determines whether a machine check exception will be taken as a result of a data parity error on a read cycle. If this pin is sampled active in the clock, a data parity error is detected. The processor will latch the address and control signals of the cycle with the parity error in the machine check registers. If, in addition, the machine check enable bit in CR4 is set to 1, the processor will vector to the machine check exception before the beginning of the next instruction.
PM/BP[1:0]	O	These pins function as part of the performance monitoring feature. The breakpoint 1–0 pins are multiplexed with the **performance monitoring 1–0** pins. The PB1 and PB0 bits in the Debug Mode Control Register determine if the pins are configured as breakpoint or performance monitoring pins. The pins come out of RESET configured for performance monitoring.
PRDY	O	The **probe ready** output pin indicates that the processor has stopped normal execution in response to the R/S# pin going active or Probe Mode being entered.
PWT	O	The **page writethrough** pin reflects the state of the PWT bit in CR3, the page directory entry, or the page table entry. The PWT pin is used to provide an external writeback indication on a page-by-page basis.
R/S#	I	The **run/stop** input is an asynchronous, edge-sensitive interrupt used to stop the normal execution of the processor and place it into an idle state. A high to low transition on the R/S# pin will interrupt the processor and cause it to stop execution at the next instruction boundary.
RESET	I	**RESET** forces the processor to begin execution at a known state. All the processor internal caches will be invalidated upon the RESET. Modified lines in the data cache are not written back. FLUSH# and INIT are sampled when RESET transitions from high to low to determine if tristate test mode will be entered or if BIST will be run.
SCYC	O	The **split cycle** output is asserted during misaligned LOCKed transfers to indicate that more than two cycles will be locked together. This signal is defined for locked cycles only. It is undefined for cycles which are not locked.

Table 4. Quick Pin Reference (Continued)

Symbol	Type	Name and Function
SMI#	I	The **system management interrupt** causes a system management interrupt request to be latched internally. When the latched SMI# is recognized on an instruction boundary, the processor enters System Management Mode.
SMIACT#	O	An active **system management interrupt active** output indicates that the processor is operating in System Management Mode.
STPCLK#	I	Assertion of the **stop clock** input signifies a request to stop the internal clock of the Pentium processor with voltage reduction technology thereby causing the core to consume less power. When the CPU recognizes STPCLK#, the processor will stop execution on the next instruction boundary, unless superseded by a higher priority interrupt, and generate a Stop Grant Acknowledge cycle. When STPCLK# is asserted, the processor will still respond to external snoop requests.
TCK	I	The **testability clock** input provides the clocking function for the processor boundary scan in accordance with the IEEE Boundary Scan interface (Standard 1149.1). It is used to clock state information and data into and out of the processor during boundary scan.
TDI	I	The **test data input** is a serial input for the test logic. TAP instructions and data are shifted into the processor on the TDI pin on the rising edge of TCK when the TAP controller is in an appropriate state.
TDO	O	The **test data output** is a serial output of the test logic. TAP instructions and data are shifted out of the processor on the TDO pin on TCK's falling edge when the TAP controller is in an appropriate state.
TMS	I	The value of the **test mode select** input signal sampled at the rising edge of TCK controls the sequence of TAP controller state changes.
TRST#	I	When asserted, the **test reset** input allows the TAP controller to be asynchronously initialized.
$V_{CC}2$	I	These pins are the 2.9V power inputs to the Pentium processor with voltage reduction technology.
$V_{CC}3$	I	These pins are the 3.3V power inputs to the Pentium processor with voltage reduction technology.
V_{SS}	I	These pins are the ground inputs to the Pentium processor with voltage reduction technology.
W/R#	O	**Write/read** is one of the primary bus cycle definition pins. It is driven valid in the same clock as the ADS# signal is asserted. W/R# distinguishes between write and read cycles.
WB/WT#	I	The **writeback/writethrough** input allows a data cache line to be defined as writeback or writethrough on a line-by-line basis. As a result, it determines whether a cache line is initially in the S or E state in the data cache.

3.5. Pin Reference Tables

Table 5. Output Pins

Name	Active Level	When Floated
ADS#	Low	Bus Hold, BOFF#
APCHK#	Low	
BE7#–BE5#	Low	Bus Hold, BOFF#
BREQ	High	
CACHE#	Low	Bus Hold, BOFF#
FERR#	Low	
HIT#	Low	
HITM#	Low	
HLDA	High	
IERR#	Low	
LOCK#	Low	Bus Hold, BOFF#
M/IO#, D/C#, W/R#	n/a	Bus Hold, BOFF#
PCHK#	Low	
BP3–2, PM1/BP1, PM0/BP0	High	
PRDY	High	
PWT, PCD	High	Bus Hold, BOFF#
SCYC	High	Bus Hold, BOFF#
SMIACT#	Low	
TDO	n/a	All states except Shift-DR and Shift-IR

NOTE:
All output and input/output pins are floated during tristate test mode (except TDO).

Table 6. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal Resistor	Qualified
A20M#	Low	Asynchronous		
AHOLD	High	Synchronous		
BF	High	Synchronous/RESET	Pullup	
BOFF#	Low	Synchronous		
BRDY#	Low	Synchronous	Pullup	Bus State T2, T12, T2P
BUSCHK#	Low	Synchronous	Pullup	BRDY#
CLK	n/a			
EADS#	Low	Synchronous		
EWBE#	Low	Synchronous		BRDY#
FLUSH#	Low	Asynchronous		
HOLD	High	Synchronous		
IGNNE#	Low	Asynchronous		
INIT	High	Asynchronous		
INTR	High	Asynchronous		
INV	High	Synchronous		EADS#
KEN#	Low	Synchronous		First BRDY#/NA#
NA#	Low	Synchronous		Bus State T2,TD,T2P
NMI	High	Asynchronous		
PEN#	Low	Synchronous		BRDY#
R/S#	n/a	Asynchronous	Pullup	
RESET	High	Asynchronous		
SMI#	Low	Asynchronous	Pullup	
STPCLK#	Low	Asynchronous	Pullup	
TCK	n/a		Pullup	
TDI	n/a	Synchronous/TCK	Pullup	TCK
TMS	n/a	Synchronous/TCK	Pullup	TCK
TRST#	Low	Asynchronous	Pullup	
WB/WT#	n/a	Synchronous		First BRDY#/NA#

Table 7. Input/Output Pins

Name	Active Level	When Floated	Qualified (when an Input)	Internal Resistor
A31–A3	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
AP	n/a	Address Hold, Bus Hold, BOFF#	EADS#	
BE4#–BE0#	Low	Bus Hold, BOFF#	RESET	Pulldown*
D63–D0	n/a	Bus Hold, BOFF#	BRDY#	
DP7–DP0	n/a	Bus Hold, BOFF#	BRDY#	

NOTES:
All output and input/output pins are floated during tristate test mode (except TDO).
*BE3#–BE0# have pulldowns during RESET only.

3.6. Pin Grouping According to Function

Table 8 organizes the pins with respect to their function.

Table 8. Pin Functional Grouping

Function	Pins
Clock	CLK
Initialization	RESET, INIT
Address Bus	A31-A3, BE7#-BE0#
Address Mask	A20M#
Data Bus	D63-D0
Address Parity	AP, APCHK#
Data Parity	DP7-DP0, PCHK#, PEN#
Internal Parity Error	IERR#
System Error	BUSCHK#
Bus Cycle Definition	M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK#
Bus Control	ADS#, BRDY#, NA#
Page Cacheability	PCD, PWT
Cache Control	KEN#, WB/WT#
Cache Snooping/Consistency	AHOLD, EADS#, HIT#, HITM#, INV
Cache Flush	FLUSH#
Write Ordering	EWBE#
Bus Arbitration	BOFF#, BREQ, HOLD, HLDA
Interrupts	INTR, NMI
Floating Point Error Reporting	FERR#, IGNNE#
System Management Mode	SMI#, SMIACT#
TAP Port	TCK, TMS, TDI, TDO, TRST#
Breakpoint/Performance Monitoring	PM0/BP0, PM1/BP1, BP3-2
Clock Control	STPCLK#
Probe Mode	R/S#, PRDY

4.0. TCP PENTIUM® PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY ELECTRICAL SPECIFICATIONS

4.1. Maximum Ratings

The following values are stress ratings only. Functional operation at the maximum ratings is not implied nor guaranteed. Functional operating conditions are given in the AC and DC specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Pentium processor with voltage reduction technology contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Case temperature under bias ..... −65°C to 110°C

Storage temperature ..... −65°C to 150°C

3V Supply voltage
with respect to V_{SS} ..... −0.5V to +4.6V

2.9V Supply voltage
with respect to V_{SS} ..... −0.5V to +4.1V

3V Only Buffer DC Input Voltage .... −0.5V to $V_{CC}3$ ..... +0.5; not to exceed 4.6V (2)

5V Safe Buffer
DC Input Voltage ..... −0.5V to 6.5V (1,3)

NOTES:
1. Applies to CLK.
2. Applies to all Pentium processors with voltage reduction technology inputs except CLK.
3. See Table 10.

WARNING

Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

4.2. DC Specifications

Tables 9, 10 and 11 list the DC specifications which apply to the TCP Pentium processor with voltage reduction technology. The Pentium processor with voltage reduction technology core operates at 2.9V internally while the I/O interface operates at 3.3V. The CLK input may be 3.3V or 5V. Since the 3.3V (5V safe) input levels defined in Table 10 are the same as the 5V TTL levels, the CLK input is compatible with existing 5V clock drivers. The power dissipation specification in Table 12 is provided for design of thermal solutions during operation in a sustained maximum level. This is the worst-case power the device would dissipate in a system for a sustained period of time. This number is used for design of a thermal solution for the device.

4.2.1. POWER SEQUENCING

There is no specific sequence required for powering up or powering down the $V_{CC}2$ and $V_{CC}3$ power supplies. However, for compatibility with future mobile processors, it is recommended that the $V_{CC}2$ and $V_{CC}3$ power supplies be either both on or both off within one second of each other.

Table 9. 3.3V DC Specifications

T_{CASE} = 0 to 95°C; $V_{CC}2$ = 2.9V ± 165mV; $V_{CC}3$ = 3.3V ± 165mV

Symbol	Parameter	Min	Max	Unit	Notes
$V_{IL}3$	Input Low Voltage	−0.3	0.8	V	TTL Level (3)
$V_{IH}3$	Input High Voltage	2.0	$V_{CC}3+0.3$	V	TTL Level (3)
$V_{OL}3$	Output Low Voltage		0.4	V	TTL Level (1) (3)
$V_{OH}3$	Output High Voltage	2.4		V	TTL Level (2) (3)
$I_{CC}2$	Power Supply Current from 2.9V core supply		2096 2515 2500	mA mA mA	@75 MHz (4) @90 MHz (4) @120 MHz (4) (5)
$I_{CC}3$	Power Supply Current from 3.3V I/O buffer supply		26 318 320	mA mA mA	@75 MHz (4) @90 MHz (4) @120 MHz (4)

NOTES:
1. Parameter measured at 4 mA.
2. Parameter measured at 3 mA.
3. 3.3V TTL levels apply to all signals except CLK.
4. This value should be used for power supply design. It was determined using a worst-case instruction mix and V_{CC} + 165mV. Power supply transient response and decoupling capacitors must be sufficient to handle the instantaneous current changes occurring during transitions from stop clock to full active modes. For more information, refer to section 4.3.2.
5. The lower power number is due to a process improvement.

Table 10. 3.3V (5V Safe) DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
$V_{IL}5$	Input Low Voltage	−0.3	0.8	V	TTL Level (1)
$V_{IH}5$	Input High Voltage	2.0	5.55	V	TTL Level (1)

NOTES:
1. Applies to CLK only.

Table 11. Input and Output Characteristics

Symbol	Parameter	Min	Max	Unit	Notes
C_{IN}	Input Capacitance		15	pF	(4)
C_O	Output Capacitance		20	pF	(4)
$C_{I/O}$	I/O Capacitance		25	pF	(4)
C_{CLK}	CLK Input Capacitance		15	pF	(4)
C_{TIN}	Test Input Capacitance		15	pF	(4)
C_{TOUT}	Test Output Capacitance		20	pF	(4)
C_{TCK}	Test Clock Capacitance		15	pF	(4)
I_{LI}	Input Leakage Current		±15	μA	$0 < V_{IN} < V_{CC}3$ (1)
I_{LO}	Output Leakage Current		±15	μA	$0 < V_{IN} < V_{CC}3$ (1)
I_{IH} I_{IL}	Input Leakage Current Input Leakage Current		200 −400	μA μA	V_{IN} = 2.4V (3) V_{IN} = 0.4V (2)

2

NOTES:
1. This parameter is for input without pull up or pull down.
2. This parameter is for input with pull up.
3. This parameter is for input with pull down.
4. Guaranteed by design.

Table 12. Power Dissipation Requirements for Thermal Solution Design

Parameter	Typical(1)	Max(2)	Unit	Notes
Active Power Dissipation	2.0–3.0 2.5–3.5 2.5–3.5	6.0 7.3 7.1	Watts Watts Watts	@75 MHz @90 MHz @120 MHz (5)
Stop Grant and Auto Halt Powerdown Power Dissipation		1.0 1.2 1.2	Watts Watts Watts	@75 MHz (3) @90 MHz (3) @120 MHz (3)
Stop Clock Power Dissipation	.02 .02	0.05 0.05 0.05	Watts Watts Watts	@75 MHz (4) @90 MHz (4) @120 MHz (4)

NOTES:
1. This is the typical power dissipation in a system. This value was the average value measured in a system using a typical device at $V_{CC}2$ = 2.9V and $V_{CC}3$ = 3.3V running typical applications. This value is highly dependent upon the specific system configuration.
2. Systems must be designed to thermally dissipate the maximum active power dissipation. It is determined using a worst-case instruction mix with $V_{CC}2$ = 2.9V and $V_{CC}3$ = 3.3V. The use of nominal V_{CC} in this measurement takes into account the thermal time constant of the package.
3. Stop Grant/Auto Halt Powerdown Power Dissipation is determined by asserting the STPCLK# pin or executing the HALT instruction.
4. Stop Clock Power Dissipation is determined by asserting the STPCLK# pin and then removing the external CLK input.
5. The lower power number is due to a process improvement.

4.3. AC Specifications

The AC specifications of the TCP Pentium processor with voltage reduction technology consist of setup times, hold times, and valid delays at 0 pF. All TCP Pentium processor with voltage reduction technology AC specifications are valid for $V_{CC}2$ = 2.9V ± 165mV, $V_{CC}3$ = 3.3V ± 165mV and Tcase = 0 to 95°C.

WARNING

Do not exceed the 75-MHz Pentium processor with voltage reduction technology internal maximum frequency of 75 MHz by either selecting the ½ bus fraction or providing a clock greater than 50 MHz.

Do not exceed the 90-MHz Pentium processor with voltage reduction technology internal maximum frequency of 90 MHz by either selecting the ½ bus fraction or providing a clock greater than 60 MHz.

4.3.1. POWER AND GROUND

For clean on-chip power distribution, the TCP Pentium processor with voltage reduction technology has 37 $V_{CC}2$ (2.9V power), 42 $V_{CC}3$ (3.3V power) and 72 V_{SS} (ground) inputs. Power and ground connections must be made to all external $V_{CC}2$, $V_{CC}3$ and V_{SS} pins of the Pentium processor with voltage reduction technology. On the circuit board all $V_{CC}2$ pins must be connected to a 2.9V V_{CC2} plane (or island) and all $V_{CC}3$ pins must be connected to a 3.3V $V_{CC}3$ plane. All V_{SS} pins must be connected to a V_{SS} plane. Please refer to Table 2 for the list of $V_{CC}2$, $V_{CC}3$ and V_{SS} pins.

4.3.2. DECOUPLING RECOMMENDATIONS

Transient power surges can occur as the processor is executing instruction sequences or driving large loads. To mitigate these high frequency transients, liberal high frequency decoupling capacitors should be placed near the processor.

Low inductance capacitors and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by shortening circuit board traces between the processor and decoupling capacitors as much as possible.

These capacitors should be evenly distributed around each component on the 3.3V plane and the 2.9V plane (or island). Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

Power transients also occur as the processor rapidly transitions from a low level of power consumption to a much higher level (or high to low power). A typical example would be entering or exiting the Stop Grant state. Another example would be executing a HALT instruction, causing the processor to enter the Auto HALT Powerdown state, or transitioning from HALT to the Normal state. All of these examples may cause abrupt changes in the power being consumed by the processor. Note that the Auto HALT Powerdown feature is always enabled even when other power management features are not implemented.

Bulk storage capacitors with a low ESR (Effective Series Resistance) in the 10 to 100 μF range are required to maintain a regulated supply voltage during the interval between the time the current load changes and the point that the regulated power supply output can react to the change in load. In order to reduce the ESR, it may be necessary to place several bulk storage capacitors in parallel.

These capacitors should be placed near the processor (on the 3.3V plane and the 2.9V plane (or island)) to ensure that these supply voltages stay within specified limits during changes in the supply current during operation.

For more detailed informations, please contact Intel or refer to the *Pentium® Processor with Voltage Reduction Technology: Power Supply Design Considerations for Mobile Systems* application note (Order Number 242558).

4.3.3. CONNECTION SPECIFICATIONS

All NC pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to $V_{CC}3$. Unused active high inputs should be connected to ground.

4.3.4. AC TIMINGS FOR A 50-MHZ BUS

The AC specifications given in Table 13 consist of output delays, input setup requirements and input hold requirements for a 50-MHz external bus. All AC specifications (with the exception of those for the TAP signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct operation.

Table 13. TCP Pentium® Processor with Voltage Reduction Technology AC Specifications for 50-MHz Bus Operation

$V_{CC}2$ = 2.9V ±165mV, $V_{CC}3$ = 3.3V ±165mV, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	25.0	50.0	MHz		
t_{1a}	CLK Period	20.0	40.0	nS	3	
t_{1b}	CLK Period Stability		±250	pS		(1), (19)
t_2	CLK High Time	4.0		nS	3	@2V, (1)
t_3	CLK Low Time	4.0		nS	3	@0.8V, (1)
t_4	CLK Fall Time	0.15	1.5	nS	3	(2.0V–0.8V), (1), (5)
t_5	CLK Rise Time	0.15	1.5	nS	3	(0.8V–2.0V), (1), (5)
t_{6a}	ADS#, PWT, PCD, BE0–7#, M/IO#, D/C#, CACHE#, SCYC, W/R# Valid Delay	1.0	7.0	nS	4	(22)
t_{6b}	AP Valid Delay	1.0	8.5	nS	4	(22)
t_{6c}	A3–A31, LOCK# Valid Delay	1.1	7.0	nS	4	(22)

Table 13. TCP Pentium® Processor with Voltage Reduction Technology AC Specifications for 50-MHz Bus Operation (Continued)

$V_{CC}2$ = 2.9V ±165mV, $V_{CC}3$ = 3.3V ±165mV, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_7	ADS#, AP, A3–A31, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	5	(1), (22)
t_8	APCHK#, IERR#, FERR#, PCHK# Valid Delay	1.0	8.3	nS	4	(4), (22)
t_{9a}	BREQ, HLDA, SMIACT# Valid Delay	1.0	8.0	nS	4	(4), (22)
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	4	(22)
t_{10b}	HITM# Valid Delay	1.1	6.6	nS	4	(22)
t_{11a}	PM0–1, BP0–3 Valid Delay	1.0	10.0	nS	4	(22)
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	4	(22)
t_{12}	D0–D63, DP0–7 Write Data Valid Delay	1.3	8.5	nS	4	(22)
t_{13}	D0–D63, DP0–3 Write Data Float Delay		10.0	nS	5	(1)
t_{14}	A5–A31 Setup Time	6.5		nS	6	(20)
t_{15}	A5–A31 Hold Time	1.0		nS	6	
t_{16a}	INV, AP Setup Time	5.0		nS	6	
t_{16b}	EADS# Setup Time	6.0		nS	6	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	6	
t_{18a}	KEN# Setup Time	5.0		nS	6	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	6	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	6	
t_{20}	BRDY# Setup Time	5.0		nS	6	
t_{21}	BRDY# Hold Time	1.0		nS	6	
t_{22}	BOFF# Setup Time	5.5		nS	6	
t_{22a}	AHOLD Setup Time	6.0		nS	6	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	6	

Table 13. TCP Pentium® Processor with Voltage Reduction Technology AC Specifications for 50-MHz Bus Operation (Continued)

$V_{CC}2$ = 2.9V ±165mV, $V_{CC}3$ = 3.3V ±165mV, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	6	
t_{25}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	6	
t_{25a}	HOLD Hold Time	1.5		nS	6	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	6	(11), (15)
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	6	(12)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	6	(11), (15), (16)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	6	(12)
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs	6	(14), (16)
t_{31}	R/S# Setup Time	5.0		nS	6	(11), (15), (16)
t_{32}	R/S# Hold Time	1.0		nS	6	(12)
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs	6	(14), (16)
t_{34}	D0–D63, DP0–7 Read Data Setup Time	3.8		nS	6	
t_{35}	D0–D63, DP0–7 Read Data Hold Time	1.5		nS	6	
t_{36}	RESET Setup Time	5.0		nS	7	(11), (15)
t_{37}	RESET Hold Time	1.0		nS	7	(12)
t_{38}	RESET Pulse Width, V_{CC} and CLK Stable	15		CLKs	7	(16)
t_{39}	RESET Active After V_{CC} and CLK Stable	1.0		mS	7	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#) Setup Time	5.0		nS	7	(11), (15), (16)
t_{41}	Reset Configuration Signals (INIT, FLUSH#) Hold Time	1.0		nS	7	(12)

Table 13. TCP Pentium® Processor with Voltage Reduction Technology AC Specifications for 50-MHz Bus Operation (Continued)

$V_{CC}2$ = 2.9V ±165mV, $V_{CC}3$ = 3.3V ±165mV, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{42a}	Reset Configuration Signals (INIT, FLUSH#) Setup Time, Async.	2.0		CLKs	7	To RESET falling edge (15)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, BRDY#, BUSCHK#) Hold Time, Async.	2.0		CLKs	7	To RESET falling edge (21)
t_{42c}	Reset Configuration Signal (BRDY#, BUSCHK#) Setup Time, Async.	3.0		CLKs	7	To RESET falling edge (21)
t_{42d}	Reset Configuration Signal BRDY# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge (1), (21)
t_{43a}	BF Setup Time	1.0		mS	7	(18) to RESET falling edge
t_{43b}	BF Hold Time	2.0		CLKs	7	(18) to RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		nS	3	
t_{46}	TCK High Time	25.0		nS	3	@2V, (1)
t_{47}	TCK Low Time	25.0		nS	3	@0.8V, (1)
t_{48}	TCK Fall Time		5.0	nS	3	(2.0V–0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5.0	nS	3	(0.8V–2.0V), (1), (8), (9)
t_{50}	TRST# Pulse Width	40.0		nS	9	(1), Asynchronous
t_{51}	TDI, TMS Setup Time	5.0		nS	8	(7)
t_{52}	TDI, TMS Hold Time	13.0		nS	8	(7)
t_{53}	TDO Valid Delay	3.0	20.0	nS	8	(8)
t_{54}	TDO Float Delay		25.0	nS	8	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	8	(3), (8), (10)

Table 13. TCP Pentium® Processor with Voltage Reduction Technology AC Specifications for 50-MHz Bus Operation (Continued)

$V_{CC}2$ = 2.9V ±165mV, $V_{CC}3$ = 3.3V ±165mV, T_{CASE} = 0°C to 95°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	8	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	8	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	8	(3), (7), (10)

NOTES:
Notes 2, 6 and 14 are general and apply to all standard TTL signals used with the Pentium® processor family.

1. Not 100 percent tested. Guaranteed by design.
2. TTL input test waveforms are assumed to be 0 to 3V transitions with 1V/nS rise and fall times.
3. Non-test outputs and inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch-free outputs. Glitch-free signals monotonically transition without false transitions (i.e., glitches).
5. 0.8V/ns ≤ CLK input rise/fall time ≤ 8V/ns.
6. 0.3V/ns ≤ input rise/fall time ≤ 5V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 33 MHz.
10. During probe mode operation, do not use the boundary scan timings (t_{55-58}).
11. Setup time is required to guarantee recognition on a specific clock.
12. Hold time is required to guarantee recognition on a specific clock.
13. All TTL timings are referenced from 1.5V.
14. To guarantee proper asynchronous recognition, the signal must have been de-asserted (inactive) for a minimum of two clocks before being returned active and must meet the minimum pulse width.
15. This input may be driven asynchronously.
16. When driven asynchronously, RESET, NMI, FLUSH#, R/S#, INIT, and SMI# must be de-asserted (inactive) for a minimum of two clocks before being returned active.
17. The D/C#, M/IO#, W/R#, CACHE#, and A5–A31 signals are sampled only on the CLK that ADS# is active.
18. BF should be strapped to $V_{CC}3$ or left floating.
19. These signals are measured on the rising edge of adjacent CLKs at 1.5V. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency. The amount of jitter present must be accounted for as a component of CLK skew between devices.
20. Timing (t_{14}) is required for external snooping (e.g., address setup to the CLK in which EADS# is sampled active).
21. BUSCHK# is used as a reset configuration signal to select buffer size.
22. Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.

4.3.5. AC TIMINGS FOR A 60-MHZ BUS

The AC specifications given in Tables 14 and 15 consist of output delays, input setup requirements and input hold requirements for the 60-MHz external bus. All AC specifications (with the exception of those for the TAP signals and APIC signals) are relative to the rising edge of the CLK input.

All timings are referenced to 1.5V for both "0" and "1" logic levels unless otherwise specified. Within the sampling window, a synchronous input must be stable for correct operation.

Table 14. Differences in the TCP Pentium® Processor 735\90 and 1000\120 with Voltage Reduction Technology AC Specifications for 60-MHz Bus Operation

Symbol	Parameter	735\90		1000\120		Unit	Figure	Notes
		Min	Max	Min	Max			
t_{6c}	A3–A31, LOCK# Valid Delay	1.1	7.0			nS	4	(22)
t_{6c}	LOCK# Valid Delay			1.1	7.0	nS	4	(22)
t_{6e}	A3–A31, Valid Delay			1.1	6.3	nS	4	(22)
t_{35}	D0–D63, DP0–7 Read Data Hold Time	2.0		1.5		nS	6	

Table 15. TCP Pentium® Processor (735\90, 1000\120) with Voltage Reduction Technology AC Specifications for 60-MHz Bus Operation

$V_{CC}2$ = 2.9V ± 165mV, $V_{CC}3$ = 3.3V ± 165mV, TCP T_{CASE} = 0°C to 95°C, SPGA T_{CASE} = 0°C to 85°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	30.0	60.0	MHz		
t_{1a}	CLK Period	16.67	33.33	nS	3	
t_{1b}	CLK Period Stability		±250	pS		(1), (19)
t_2	CLK High Time	4.0		nS	3	@2V, (1)
t_3	CLK Low Time	4.0		nS	3	@0.8V, (1)
t_4	CLK Fall Time	0.15	1.5	nS	3	(2.0V–0.8V), (1), (5)
t_5	CLK Rise Time	0.15	1.5	nS	3	(0.8V–2.0V), (1), (5)
t_{6a}	ADS#, PWT, PCD, BE0–7#, M/IO#, D/C#, CACHE#, SCYC, W/R# Valid Delay	1.0	7.0	nS	4	(22)
t_{6b}	AP Valid Delay	1.0	8.5	nS	4	(22)
t_7	ADS#, AP, A3–A31, PWT, PCD, BE0–7#, M/IO#, D/C#, W/R#, CACHE#, SCYC, LOCK# Float Delay		10.0	nS	5	(1), (22)
t_{8a}	APCHK#, IERR#, FERR# Valid Delay	1.0	8.3	nS	4	(4), (22)
t_{8b}	PCHK# Valid Delay	1.0	7.0	nS	4	(4), (22)
t_{9a}	BREQ, HLDA Valid Delay	1.0	8.0	nS	4	(4), (22)
t_{9b}	SMIACT# Valid Delay	1.0	7.6	nS	4	(22)
t_{10a}	HIT# Valid Delay	1.0	8.0	nS	4	(22)
t_{10b}	HITM# Valid Delay	1.1	6.0	nS	4	(22)
t_{11a}	PM0–1, BP0–3 Valid Delay	1.0	10.0	nS	4	(22)
t_{11b}	PRDY Valid Delay	1.0	8.0	nS	4	(22)
t_{12}	D0–D63, DP0–7 Write Data Valid Delay	1.3	7.5	nS	4	(22)
t_{13}	D0–D63, DP0–3 Write Data Float Delay		10.0	nS	5	(1)
t_{14}	A5–A31 Setup Time	6.0		nS	6	(20)

Table 15. TCP Pentium® Processor (735\90, 1000\120) with Voltage Reduction Technology AC Specifications for 60-MHz Bus Operation (Continued)

$V_{CC}2$ = 2.9V ± 165mV, $V_{CC}3$ = 3.3V ± 165mV, TCP T_{CASE} = 0°C to 95°C, SPGA T_{CASE} = 0°C to 85°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{15}	A5–A31 Hold Time	1.0		nS	6	
t_{16a}	INV, AP Setup Time	5.0		nS	6	
t_{16b}	EADS# Setup Time	5.5		nS	6	
t_{17}	EADS#, INV, AP Hold Time	1.0		nS	6	
t_{18a}	KEN# Setup Time	5.0		nS	6	
t_{18b}	NA#, WB/WT# Setup Time	4.5		nS	6	
t_{19}	KEN#, WB/WT#, NA# Hold Time	1.0		nS	6	
t_{20}	BRDY# Setup Time	5.0		nS	6	
t_{21}	BRDY# Hold Time	1.0		nS	6	
t_{22}	AHOLD, BOFF# Setup Time	5.5		nS	6	
t_{23}	AHOLD, BOFF# Hold Time	1.0		nS	6	
t_{24}	BUSCHK#, EWBE#, HOLD, PEN# Setup Time	5.0		nS	6	
t_{25}	BUSCHK#, EWBE#, PEN# Hold Time	1.0		nS	6	
t_{25a}	HOLD Hold Time	1.5		nS	6	
t_{26}	A20M#, INTR, STPCLK# Setup Time	5.0		nS	6	(11), (15)
t_{27}	A20M#, INTR, STPCLK# Hold Time	1.0		nS	6	(12)
t_{28}	INIT, FLUSH#, NMI, SMI#, IGNNE# Setup Time	5.0		nS	6	(11), (15), (16)
t_{29}	INIT, FLUSH#, NMI, SMI#, IGNNE# Hold Time	1.0		nS	6	(12)
t_{30}	INIT, FLUSH#, NMI, SMI#, IGNNE# Pulse Width, Async	2.0		CLKs		(14), (16)
t_{31}	R/S# Setup Time	5.0		nS	6	(11), (15), (16)
t_{32}	R/S# Hold Time	1.0		nS	6	(12)
t_{33}	R/S# Pulse Width, Async.	2.0		CLKs		(14), (16)
t_{34}	D0–D63, DP0–7 Read Data Setup Time	3.0		nS	6	

Table 15. TCP Pentium® Processor (735\90, 1000\120) with Voltage Reduction Technology AC Specifications for 60-MHz Bus Operation (Continued)

$V_{CC}2$ = 2.9V ± 165mV, $V_{CC}3$ = 3.3V ± 165mV, TCP T_{CASE} = 0°C to 95°C, SPGA T_{CASE} = 0°C to 85°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{36}	RESET Setup Time	5.0		nS	7	(11), (15)
t_{37}	RESET Hold Time	1.0		nS	7	(12)
t_{38}	RESET Pulse Width, V_{CC} & CLK Stable	15		CLKs	7	(16)
t_{39}	RESET Active After V_{CC} & CLK Stable	1.0		mS	7	Power up
t_{40}	Reset Configuration Signals (INIT, FLUSH#) Setup Time	5.0		nS	7	(11), (15), (16)
t_{41}	Reset Configuration Signals (INIT, FLUSH#) Hold Time	1.0		nS	7	(12)
t_{42a}	Reset Configuration Signals (INIT, FLUSH#) Setup Time, Async.	2.0		CLKs	7	To RESET falling edge (15)
t_{42b}	Reset Configuration Signals (INIT, FLUSH#, BRDY#, BUSCHK#) Hold Time, Async.	2.0		CLKs	7	To RESET falling edge (21)
t_{42c}	Reset Configuration Signal (BRDY#, BUSCHK#) Setup Time, Async.	3.0		CLKs	7	To RESET falling edge (21)
t_{42d}	Reset Configuration Signal BRDY# Hold Time, RESET driven synchronously	1.0		nS		To RESET falling edge (1), (21)
t_{43a}	BF Setup Time	1.0		mS	7	(18) to RESET falling edge
t_{43b}	BF Hold Time	2.0		CLKs	7	(18) to RESET falling edge
t_{44}	TCK Frequency		16.0	MHz		
t_{45}	TCK Period	62.5		nS	3	
t_{46}	TCK High Time	25.0		nS	3	@2V, (1)
t_{47}	TCK Low Time	25.0		nS	3	@0.8V, (1)
t_{48}	TCK Fall Time		5.0	nS	3	(2.0V–0.8V), (1), (8), (9)
t_{49}	TCK Rise Time		5.0	nS	3	(0.8V–2.0V), (1), (8), (9)

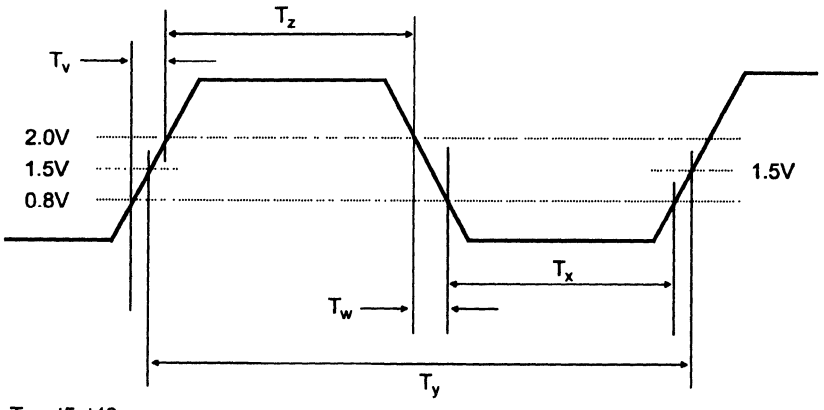
Table 15. TCP Pentium® Processor (735\90, 1000\120) with Voltage Reduction Technology AC Specifications for 60-MHz Bus Operation (Continued)

$V_{CC}2$ = 2.9V ± 165mV, $V_{CC}3$ = 3.3V ± 165mV, TCP T_{CASE} = 0°C to 95°C, SPGA T_{CASE} = 0°C to 85°C, C_L = 0 pF

Symbol	Parameter	Min	Max	Unit	Figure	Notes
t_{50}	TRST# Pulse Width	40.0		nS	9	(1), Asynchronous
t_{51}	TDI, TMS Setup Time	5.0		nS	8	(7)
t_{52}	TDI, TMS Hold Time	13.0		nS	8	(7)
t_{53}	TDO Valid Delay	3.0	20.0	nS	8	(8)
t_{54}	TDO Float Delay		25.0	nS	8	(1), (8)
t_{55}	All Non-Test Outputs Valid Delay	3.0	20.0	nS	8	(3), (8), (10)
t_{56}	All Non-Test Outputs Float Delay		25.0	nS	8	(1), (3), (8), (10)
t_{57}	All Non-Test Inputs Setup Time	5.0		nS	8	(3), (7), (10)
t_{58}	All Non-Test Inputs Hold Time	13.0		nS	8	(3), (7), (10)

NOTES:
Notes 2, 6 and 14 are general and apply to all standard TTL signals used with the Pentium® processor family.

1. Not 100 percent tested. Guaranteed by design.
2. TTL input test waveforms are assumed to be 0 to 3V transitions with 1V/nS rise and fall times.
3. Non-test outputs and inputs are the normal output or input signals (besides TCK, TRST#, TDI, TDO, and TMS). These timings correspond to the response of these signals due to boundary scan operations.
4. APCHK#, FERR#, HLDA, IERR#, LOCK#, and PCHK# are glitch-free outputs. Glitch-free signals monotonically transition without false transitions (i.e., glitches).
5. 0.8V/ns ≤ CLK input rise/fall time ≤ 8V/ns.
6. 0.3V/ns ≤ input rise/fall time ≤ 5V/ns.
7. Referenced to TCK rising edge.
8. Referenced to TCK falling edge.
9. 1 ns can be added to the maximum TCK rise and fall times for every 10 MHz of frequency below 33 MHz.
10. During probe mode operation, do not use the boundary scan timings (t_{55-58}).
11. Setup time is required to guarantee recognition on a specific clock.
12. Hold time is required to guarantee recognition on a specific clock.
13. All TTL timings are referenced from 1.5V.
14. To guarantee proper asynchronous recognition, the signal must have been de-asserted (inactive) for a minimum of two clocks before being returned active and must meet the minimum pulse width.
15. This input may be driven asynchronously.
16. When driven asynchronously, RESET, NMI, FLUSH#, R/S#, INIT, and SMI# must be de-asserted (inactive) for a minimum of two clocks before being returned active.
17. The D/C#, M/IO#, W/R#, CACHE#, and A5-A31 signals are sampled only on the CLK that ADS# is active.
18. BF should be strapped to $V_{CC}3$ or left floating.
19. These signals are measured on the rising edge of adjacent CLKs at 1.5V. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency. The amount of jitter present must be accounted for as a component of CLK skew between devices.
20. Timing (t_{14}) is required for external snooping (e.g., address setup to the CLK in which EADS# is sampled active).
21. BUSCHK# is used as a reset configuration signal to select buffer size.
22. Each valid delay is specified for a 0 pF load. The system designer should use I/O buffer modeling to account for signal flight time delays.



T_v = t5, t49

T_w = t4, t48

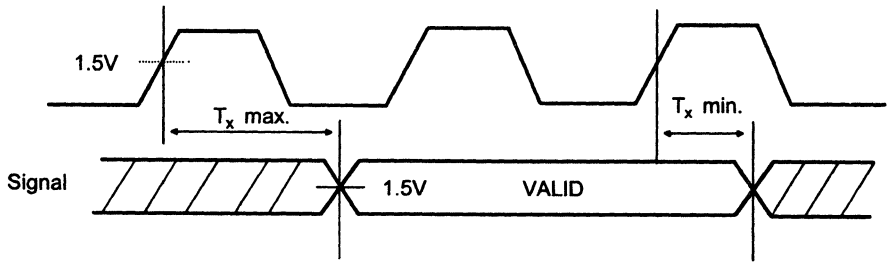
T_x = t3, t47

T_y = t1, t45

T_z = t2, t46

242557-4

Figure 3. Clock Waveform



T_x = t6, t8, t9, t10, t11, t12

242557-5

Figure 4. Valid Delay Timings

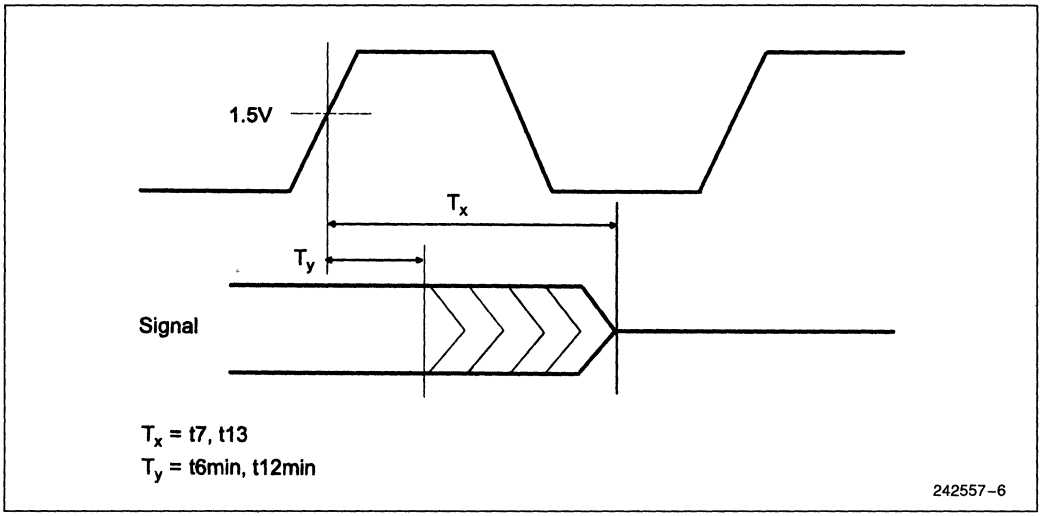


Figure 5. Float Delay Timings

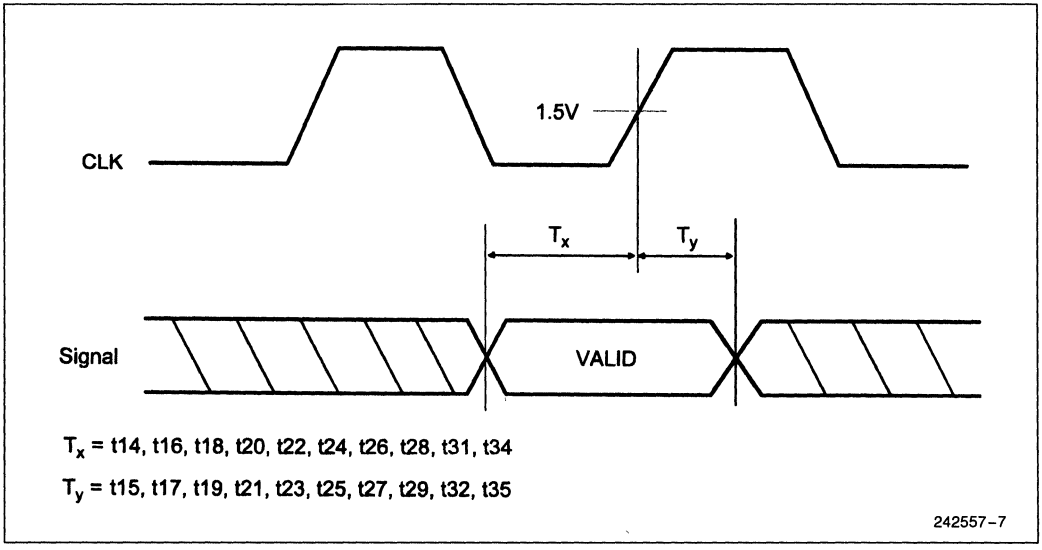
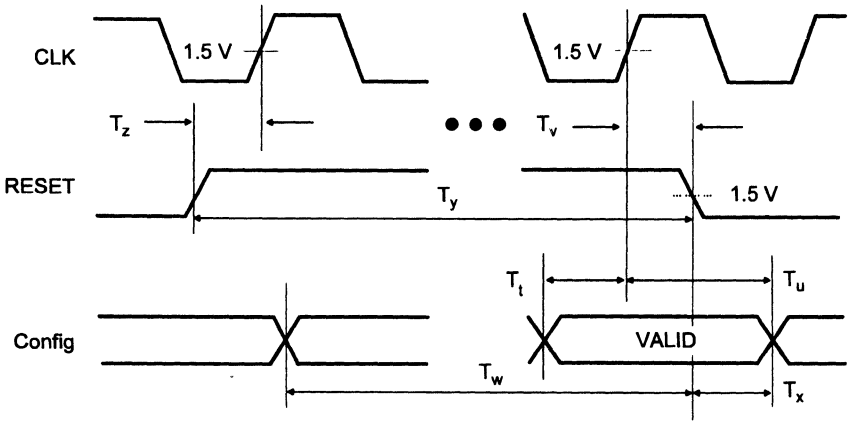


Figure 6. Setup and Hold Timings



T_t = t40

T_u = t41

T_v = t37

T_w = t42, t43a

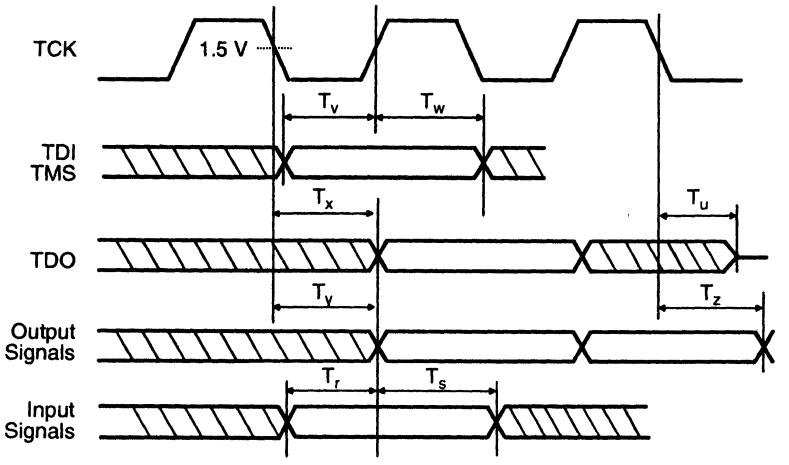
T_x = t43b

T_y = t38, t39

T_z = t36

242557-8

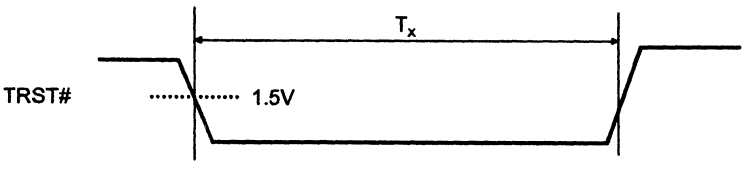
Figure 7. Reset and Configuration Timings



T_r = t57
T_s = t58
T_u = t54
T_v = t51
T_w = t52
T_x = t53
T_y = t55
T_z = t56

242557-9

Figure 8. Test Timings



T_x = t50

242557-10

Figure 9. Test Reset Timings

4.4. I/O Buffer Models

This section describes the I/O buffer models of the Pentium processor with voltage reduction technology.

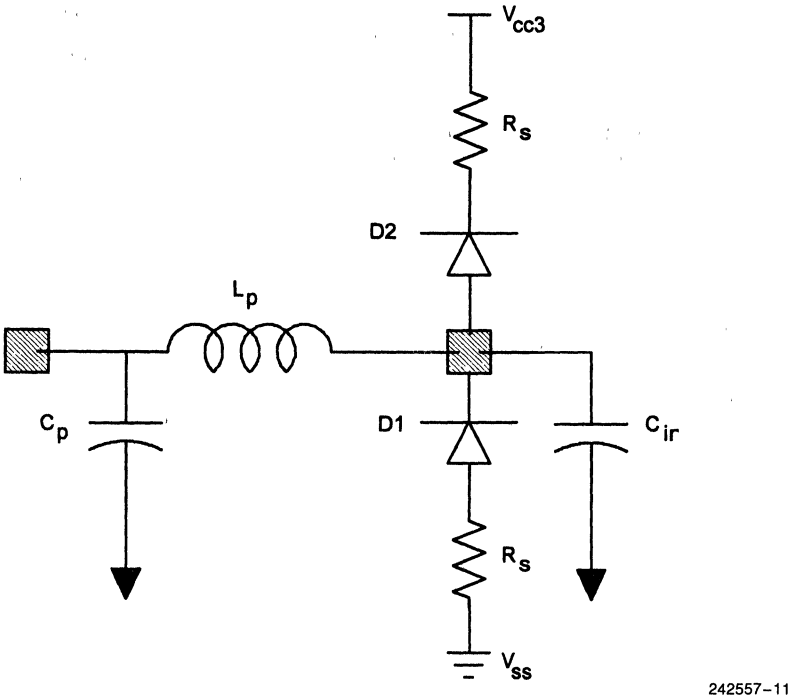
The first order I/O buffer model is a simplified representation of the complex input and output buffers used in the Pentium processor with voltage reduction technology. Figures 10 and 11 show the structure of the input buffer model and Figure 12 shows the output buffer model. Tables 16 and 17 show the parameters used to specify these models.

Although simplified, these buffer models will accurately model flight time and signal quality. For these parameters, there is very little added accuracy in a complete transistor model.

The following two models represent the input buffer models. The first model, Figure 10, represents all of the input buffers except for a special group of input buffers. The second model, Figure 11, represents these special buffers. These buffers are the inputs: AHOLD, EADS#, KEN#, WB/WT#, INV, NA#, EWBE#, BOFF# and CLK.

In addition to the input and output buffer parameters, input protection diode models are provided for added accuracy. These diodes have been optimized to provide ESD protection and provide some level of clamping. Although the diodes are not required for simulation, it may be more difficult to meet specifications without them.

Note, however, some signal quality specifications require that the diodes be removed from the input model. The series resistors (Rs) are a part of the diode model. Remove these when removing the diodes from the input model.



NOTE:
V_{CC} refers to the I/O buffer $V_{CC}3$.

Figure 10. Input Buffer Model, Except Special Group

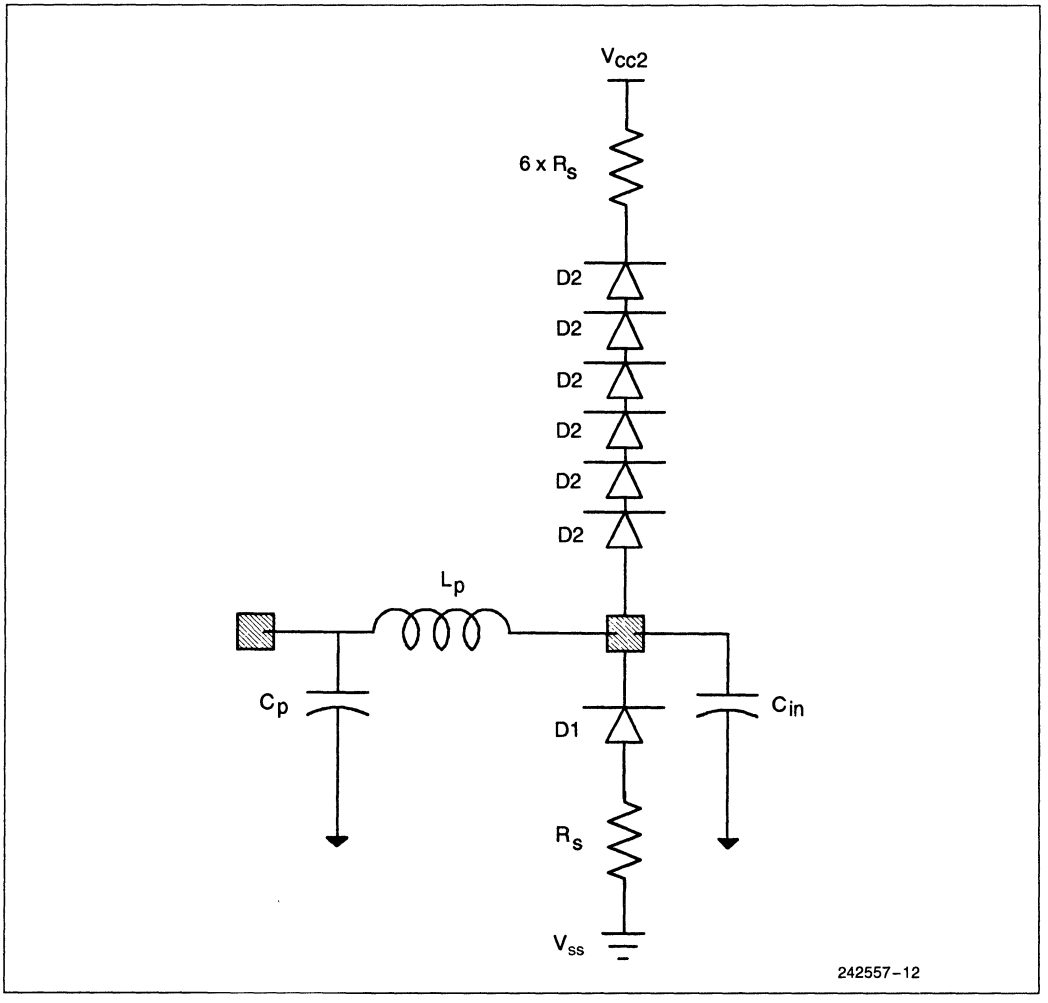


Figure 11. Input Buffer Model for Special Group

Table 16. Parameters Used in the Specification of the First Order Input Buffer Model

Parameter	Description
C_{IN}	Minimum and Maximum value of the capacitance of the input buffer model
Lp	Minimum and Maximum value of the package inductance
Cp	Minimum and Maximum value of the package capacitance
Rs	Diode Series Resistance
D1, D2	Ideal Diodes

Figure 12 shows the structure of the output buffer model. This model is used for all of the output buffers of the Pentium processor with voltage reduction technology.

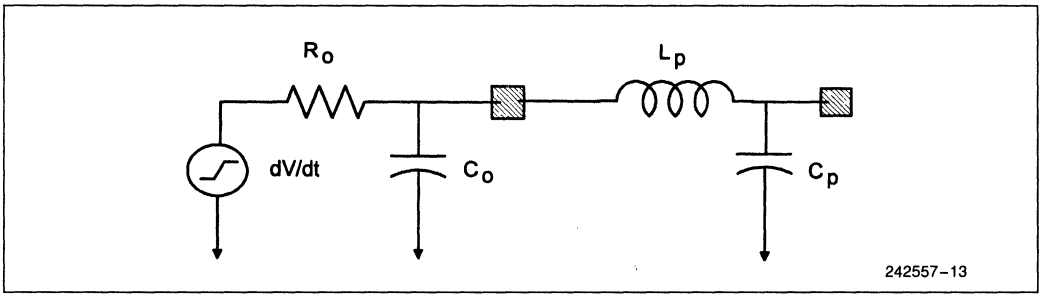


Figure 12. First Order Output Buffer Model

Table 17. Parameters Used in the Specification of the First Order Output Buffer Model

Parameter	Description
dV/dt	Minimum and maximum value of the rate of change of the open circuit voltage source used in the output buffer model
R_o	Minimum and maximum value of the output impedance of the output buffer model
C_o	Minimum and Maximum value of the capacitance of the output buffer model
Lp	Minimum and Maximum value of the package inductance
Cp	Minimum and Maximum value of the package capacitance

4.4.1. BUFFER MODEL PARAMETERS

This section gives the parameters for each TCP Pentium processor with voltage reduction technology input, output and bidirectional signal, as well as the settings for the configurable buffers.

Some pins on the TCP Pentium processor with voltage reduction technology have selectable buffer sizes. These pins use the configurable output buffer EB2. Table 18 shows the drive level for BRDY# required at the falling edge of RESET to select the buffer strength. The buffer sizes selected should be the appropriate size required; otherwise AC timings might not be met, or too much overshoot and ringback may occur. There are no other selection choices; all of the configurable buffers get set to the same size at the same time.

The input, output and bidirectional buffer values of the TCP Pentium processor with voltage reduction technology are listed in Table 20. This table contains listings for all three types, do not get them confused during simulation. When a bidirectional pin is operating as an input, use the C_{IN}, Cp and Lp values; if it is operating as a driver, use all of the data parameters.

Please refer to Table 19 for the groupings of the buffers.

Table 18. Buffer Selection Chart

Environment	BRDY#	Buffer Selection
Typical Stand Alone Component	1	EB2
Loaded Component	0	EB2A

NOTE:
For correct buffer selection, the BUSCHK# signal must be held inactive (high) at the falling edge of RESET.

Table 19. TCP Signal to Buffer Type

Signals	Type	Driver Buffer Type	Receiver Buffer Type
CLK	I		ER0
A20M#, AHOLD, BF, BOFF#, BRDY#, BUSCHK#, EADS#, EWBE#, FLUSH#, HOLD, IGNNE#, INIT, INTR, INV, KEN#, NA#, NMI, PEN#, R/S#, RESET, SMI#, STPCLK#, TCK, TDI, TMS, TRST#, WB/WT#	I		ER1
APCHK#, BE[7:5]#, BP[3:2], BREQ, FERR#, IERR#, PCD, PCHK#, PM0/BP0, PM1/BP1, PRDY, PWT, SMIACT#, TDO, U/O#	O	ED1	
A[31:21], AP, BE[4:0]#, CACHE#, D/C#, D[63:0], DP[8:0], HLDA, LOCK#, M/IO#, SCYC	I/O	EB1	EB1
A[20:3], ADS#, HITM#, W/R#	I/O	EB2/EB2A	EB2/EB2A
HIT#	I/O	EB3	EB3

Table 20. Pentium® Processor with Voltage Reduction Technology TCP Input, Output and Bidirectional Buffer Model Parameters

Buffer Type	Transition	dV/dt (V/nsec)		R_O (Ohms)		Cp (pF)		Lp (nH)		C_O/C_{IN} (pF)	
		min	max	min	max	min	max	min	max	min	max
ER0	Rising					0.3	0.4	3.9	5.0	0.8	1.2
(input)	Falling					0.3	0.4	3.9	5.0	0.8	1.2
ER1	Rising					0.2	0.5	3.1	6.0	0.8	1.2
(input)	Falling					0.2	0.5	3.1	6.0	0.8	1.2
ED1	Rising	3/3.0	3.7/0.9	21.6	53.1	0.3	0.6	3.7	6.6	2.0	2.6
(output)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.3	0.6	3.7	6.6	2.0	2.6
EB1	Rising	3/3.0	3.7/0.9	21.6	53.1	0.2	0.5	2.9	6.1	2.0	2.6
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.5	2.9	6.1	2.0	2.6
EB2	Rising	3/3.0	3.7/0.9	21.6	53.1	0.2	0.5	3.1	6.4	9.1	9.7
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.5	3.1	6.4	9.1	9.7
EB2A	Rising	3/2.4	3.7/0.9	10.1	22.4	0.2	0.5	3.1	6.4	9.1	9.7
(bidir)	Falling	3/2.4	3.7/0.9	9.0	21.2	0.2	0.5	3.1	6.4	9.1	9.7
EB3	Rising	3/3.0	3.7/0.9	21.6	53.1	0.2	0.4	3.2	4.1	3.3	3.9
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.2	0.4	3.2	4.1	3.3	3.9
EB4 (1)	Rising	3/3.0	3.7/0.9	21.6	53.1	0.3	0.4	4.0	4.1	5.0	7.0
(bidir)	Falling	3/2.8	3.7/0.8	17.5	50.7	0.3	0.4	4.0	4.1	5.0	7.0

NOTE:
1. EB4 applies to the Pentium® processor 610\75 and 735\90.

Table 21. Input Buffer Model Parameters: D (Diodes)

Symbol	Parameter	D1	D2
IS	Saturation Current	1.4e-14A	2.78e-16A
N	Emission Coefficient	1.19	1.00
RS	Series Resistance	6.5 ohms	6.5 ohms
TT	Transit Time	3 ns	6 ns
VJ	PN Potential	0.983V	0.967V
CJ0	Zero Bias PN Capacitance	0.281 pF	0.365 pF
M	PN Grading Coefficient	0.385	0.376

4.4.2. SIGNAL QUALITY SPECIFICATIONS

Signals driven by the system into the Pentium processor with voltage reduction technology must meet signal quality specifications to guarantee that the components read data properly and to ensure that incoming signals do not affect the reliability of the component. There are two signal quality parameters: Ringback and Settling Time.

4.4.2.1. Ringback

Excessive ringback can contribute to long-term reliability degradation of the Pentium processor with voltage reduction technology, and can cause false signal detection. Ringback is simulated at the input pin of a component using the input buffer model. Ringback can be simulated with or without the diodes that are in the input buffer model.

Ringback is the absolute value of the maximum voltage at the receiving pin below $V_{CC}3$ (or above V_{SS}) relative to $V_{CC}3$ (or V_{SS}) level after the signal has reached its maximum voltage level. The input diodes are assumed present.

Maximum Ringback on Inputs = 0.8V(with diodes)

If simulated without the input diodes, follow the Maximum Overshoot/Undershoot specification. By meeting the overshoot/undershoot specification, the signal is guaranteed not to ringback excessively.

If simulated with the diodes present in the input model, follow the maximum ringback specification.

Overshoot (Undershoot) is the absolute value of the maximum voltage above $V_{CC}3$ (below V_{SS}). The guideline assumes the absence of diodes on the input.

- Maximum Overshoot/Undershoot on 5V 82497 Cache Controller, and 82492 Cache SRAM Inputs (CLK and PICCLK only) = 1.6V above $V_{CC}5$ (without diodes)
- Maximum Overshoot/Undershoot on 3.3V Pentium processor with voltage reduction technology Inputs (not CLK) = 1.4V above $V_{CC}3$ (without diodes)

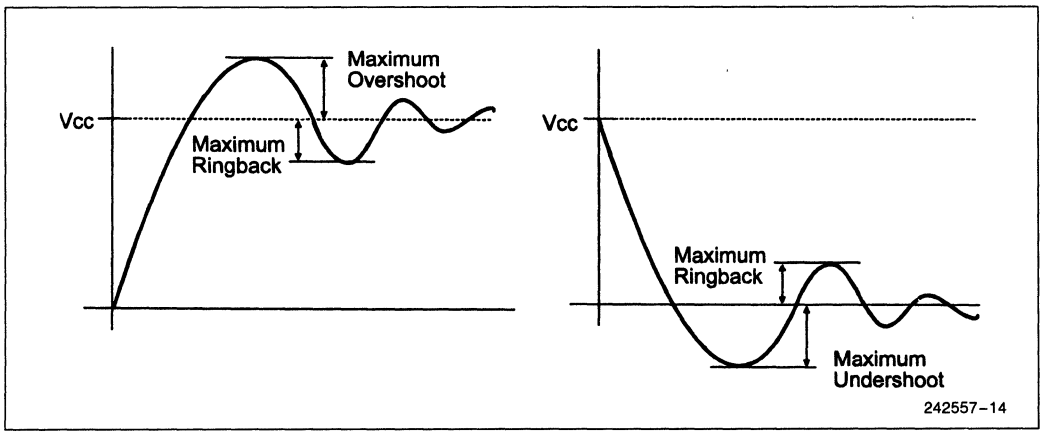


Figure 13. Overshoot/Undershoot and Ringback Guidelines

4.4.2.2. Settling Time

The settling time is defined as the time a signal requires at the receiver to settle within 10 percent of $V_{CC}3$ or V_{SS}. Settling time is the maximum time allowed for a signal to reach within 10 percent of its final value.

Most available simulation tools are unable to simulate settling time so that it accurately reflects silicon measurements. On a physical board, second-order effects and other effects serve to dampen the signal at the receiver. Because of all these concerns, settling time is a recommendation or a tool for layout tuning and not a specification.

Settling time is simulated at the slow corner, to make sure that there is no impact on the flight times of the signals if the waveform has not settled. Settling time may be simulated with the diodes included or excluded from the input buffer model. If diodes are included, settling time recommendation will be easier to meet.

Although simulated settling time has not shown good correlation with physical, measured settling time, settling time simulations can still be used as a tool to tune layouts.

Use the following procedure to verify board simulation and tuning with concerns for settling time.

1. Simulate settling time at the slow corner for a particular signal.
2. If settling time violations occur, simulate signal trace with D.C. diodes in place at the receiver pin. The D.C. diode behaves almost identically to the actual (non-linear) diode on the part as long as excessive overshoot does not occur.
3. If settling time violations still occur, simulate flight times for five consecutive cycles for that particular signal.
4. If flight time values are consistent over the five simulations, settling time should not be a concern. If however, flight times are not consistent over the five simulations, tuning of the layout is required.
5. Note that, for signals that are allocated two cycles for flight time, the recommended settling time is doubled.

A typical design method would include a settling time that ensures a signal is within 10 percent of $V_{CC}3$ or V_{SS} for at least 2.5 ns prior to the end of the CLK period.

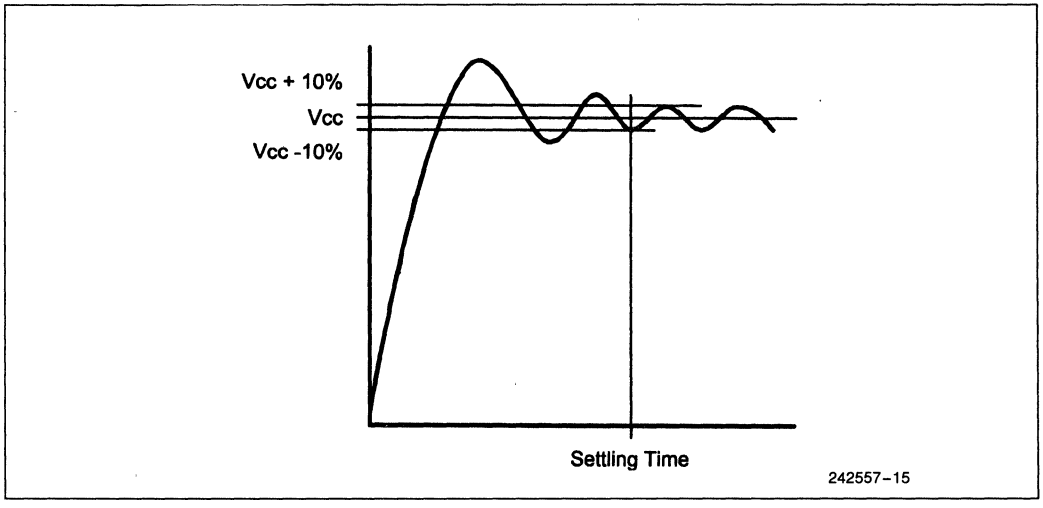


Figure 14. Settling Time

5.0. TCP PENTIUM® PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY MECHANICAL SPECIFICATIONS

Today's portable computers face the challenge of meeting desktop performance in an environment that is constrained by thermal, mechanical and electrical design considerations. These considerations have driven the development and implementation of Intel's Tape Carrier Package (TCP). The Intel TCP package has been designed to offer a high pin count, low profile, reduced footprint package with uncompromised thermal and electrical performance. Intel continues to provide packaging solutions that meet our rigorous criteria for quality and performance, and this new entry into the Intel package portfolio is no exception.

Key features of the TCP package include: surface mount technology design, lead pitch of 0.25 mm, polyimide body size of 24 mm and polyimide up for pick-and-place handling. TCP components are shipped with the leads flat in slide carriers, and are designed to be excised and lead formed at the customer manufacturing site. Recommendations for the manufacture of this package are included in the Pentium processor *Tape Carrier Package User's Guide*.

Figure 15 shows a cross-sectional view of the TCP package as mounted on the Printed Circuit Board. Figures 16 and 17 show the TCP as shipped in its slide carrier, and key dimensions of the carrier and package. Figure 18 shows a blow up detail of the package in cross-section. Figure 19 shows an enlarged view of the outer lead bond area of the package.

Tables 22 and 23 provide the Pentium processor with voltage reduction technology TCP package dimensions.

2

5.1. TCP Package Mechanical Diagrams

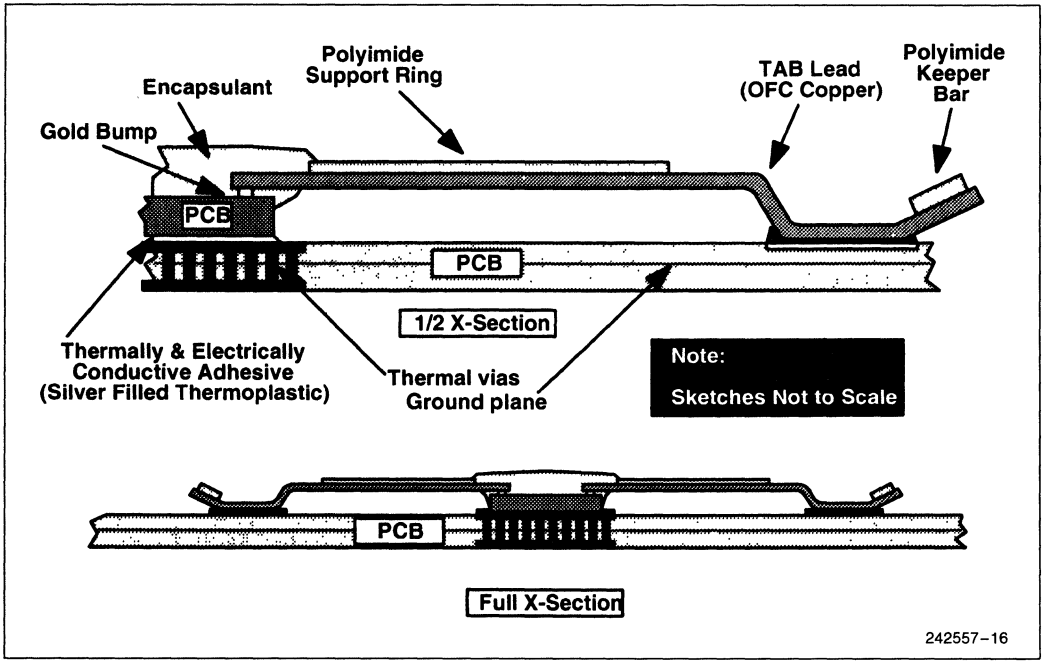


Figure 15. Cross-Sectional View of the Mounted TCP Package

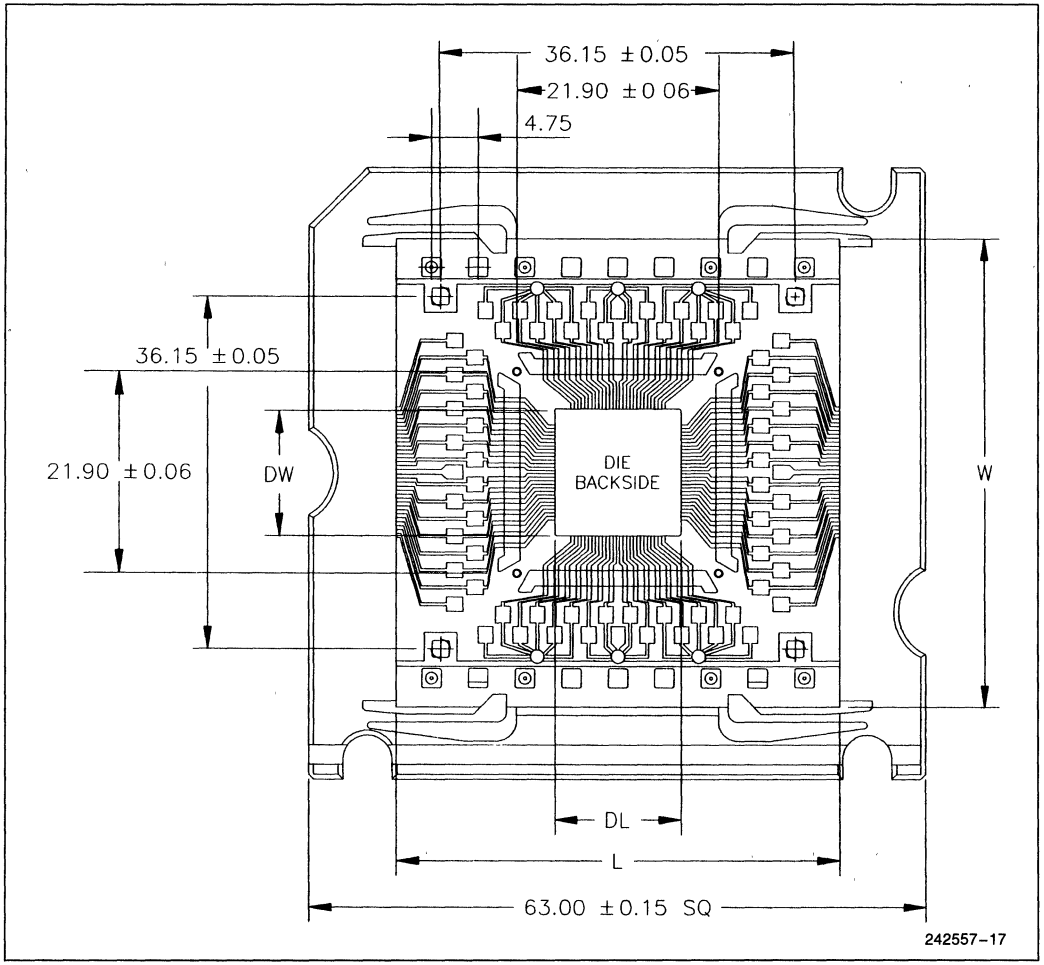


Figure 16. One TCP Site in Carrier (Bottom View of Die)

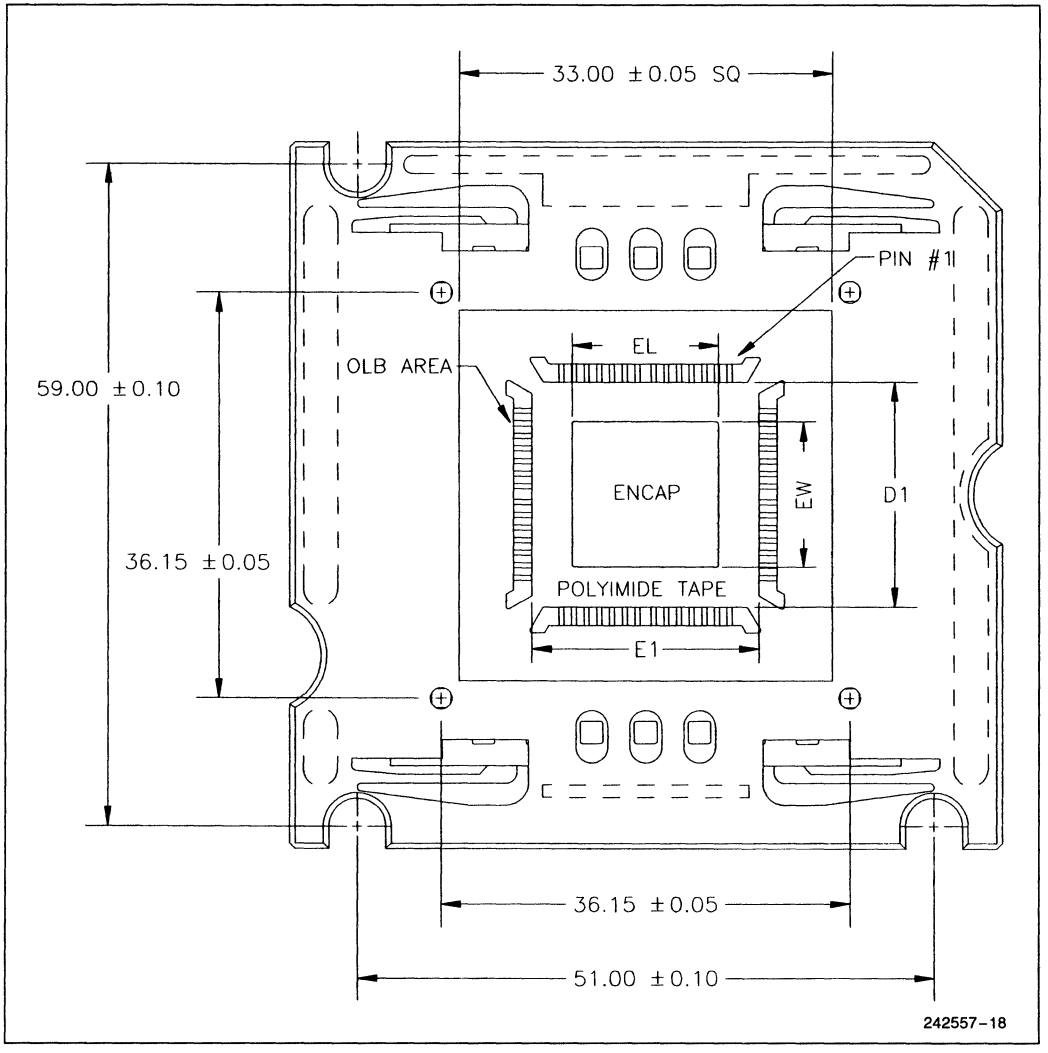


Figure 17. One TCP Site in Carrier (Top View of Die)

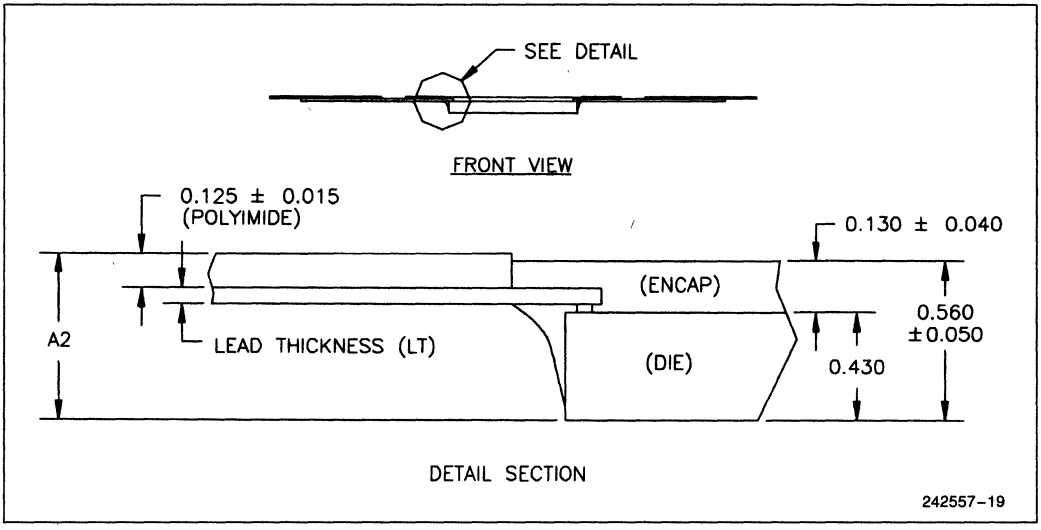


Figure 18. One TCP Site (Cross-Sectional Detail)

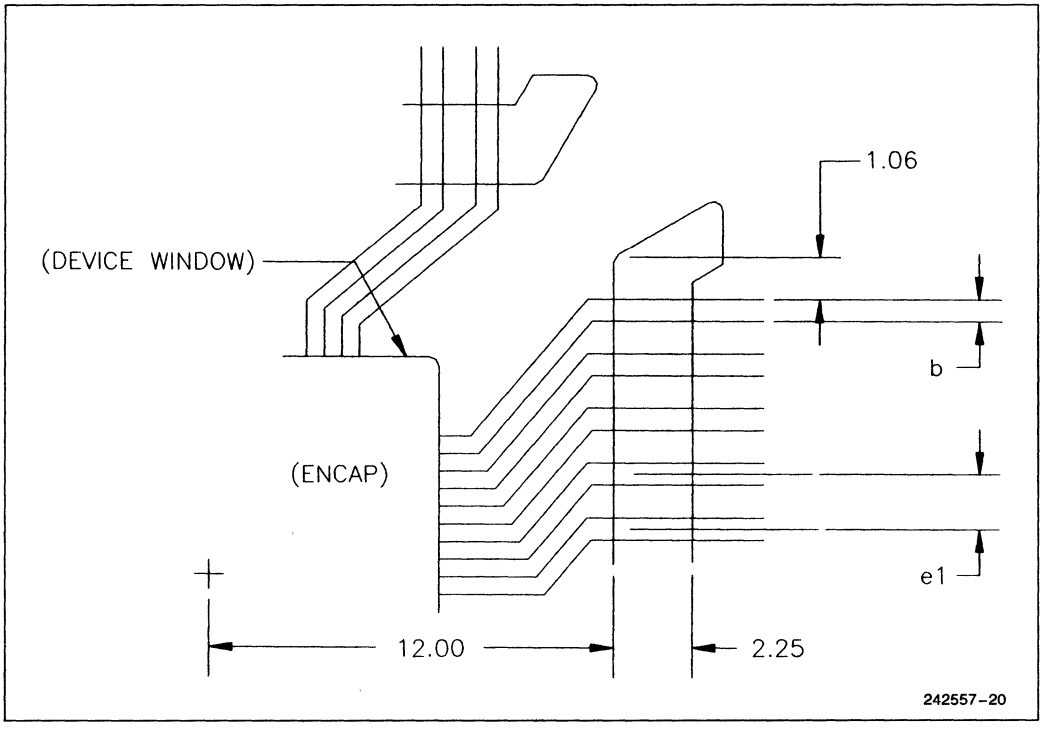


Figure 19. Outer Lead Bond (OLB) Window Detail

Table 22. TCP Key Dimensions

Symbol	Description	Dimension
N	Leadcount	320 leads
W	Tape Width	48.18 ± 0.12
L	Site Length	(43.94) reference only
e1	Outer Lead Pitch	0.25 nominal
b	Outer Lead Width	0.10 ± 0.01
D1,E1	Package Body Size	24.0 ± 0.1
A2	Package Height	75 MHz/90 MHz–0.615 ± 0.030 120 MHz–0.605 ± 0.030
DL	Die Length	75 MHz/90 MHz–12.769 ± 0.015 120 MHz–9.929 ± 0.015
DW	Die Width	75 MHz/90 MHz–11.755 ± 0.015 120 MHz–9.152 ± 0.015
LT	Lead Thickness	75 MHz/90 MHz–0.035 mm 120 MHz–0.025 mm
EL	Encap Length	75 MHz/90 MHz–(13.40 mm) reference only 120 MHz–(10.56 mm) reference only
EW	Encap Width	75 MHz/90 MHz–(12.39 mm) reference only 120 MHz–(9.78 mm) reference only

NOTES:
Dimensions are in millimeters unless otherwise noted.
Dimensions in parentheses are for reference only.

Table 23. Mounted TCP Package Dimensions

Description	Dimension
Package Height	0.75 maximum
Terminal Dimension	29.5 nominal
Package Weight	0.5 g maximum

NOTES:
Dimensions are in millimeters unless otherwise noted.
Package terminal dimension (lead tip-to-lead tip) assumes the use of a keeper bar.

6.0. TCP PENTIUM® PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY THERMAL SPECIFICATIONS

The TCP Pentium processor with voltage reduction technology is specified for proper operation when the case temperature, T_{CASE}, (T_C) is within the specified range of 0 °C to 95 °C.

6.1. Measuring Thermal Values

To verify that the proper T_C (case temperature) is maintained for the Pentium processor, it should be measured at the center of the package top surface (encapsulant). To minimize any measurement errors, the following techniques are recommended:

- Use 36 gauge or finer diameter K, T, or J type thermocouples. Intel's laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using highly thermally conductive cements. Intel's laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90° angle as shown in Figure 20.

6.2. Thermal Equations

For the Pentium processor with voltage reduction technology, an ambient temperature (T_A) is not specified directly. The only requirement is that the case temperature (T_C) is met. The ambient temperature can be calculated from the following equations:

$$T_J = T_C + P \times \theta_{JC}$$
$$T_A = T_J - P \times \theta_{JA}$$
$$T_A = T_C - (P \times \theta_{CA})$$
$$T_C = T_A + P \times [\theta_{JA} - \theta_{JC}]$$
$$\theta_{CA} = \theta_{JA} - \theta_{JC}$$

where,

T_A and T_C are ambient and case temperatures (C)

θ_{CA} = Case-to-Ambient thermal resistance (C/W)

θ_{JA} = Junction-to-Ambient thermal resistance (C/W)

θ_{JC} = Junction-to-Case thermal resistance (C/W)

P = maximum power consumption (Watts)

P (maximum power consumption) is specified in section 4.2.

6.3. TCP Thermal Characteristics

The primary heat transfer path from the die of the Tape Carrier Package (TCP) is through the back side of the die and into the PC board. There are two thermal paths traveling from the PC board to the ambient air. One is the spread of heat within the board and the dissipation of heat by the board to the ambient air. The other is the transfer of heat through the board and to the opposite side where thermal enhancements (e.g., heat sinks, pipes) are attached. Solder-side heat sinking, compared to TCP component-side heat sinking, is the prefered method due to reduced risk of die damage, easier mechanical implementation and larger surface area for attachment. However, component-side heat sinking is possible. The design requirements in a component-side thermal solution are: no direct loading of inner lead bonds on the TCP, a maximum force of 4.5 kgf on the center of a clear TCP, no direct loading of the TAB tape or outer lead bonds and controlled board deflection.

6.4. PC Board Enhancements

Copper planes, thermal pads, and vias are design options that can be used to improve heat transfer from the PC board to the ambient air. Tables 24 and 25 present thermal resistance data for copper plane thickness and via effects. It should be noted that although thicker copper planes will reduce the θ_{CA} of a system without any thermal enhancements, they have less effect on the θ_{CA} of a system with thermal enhancements. However, placing vias under the die will reduce the θ_{CA} of a system with and without thermal enhancements.

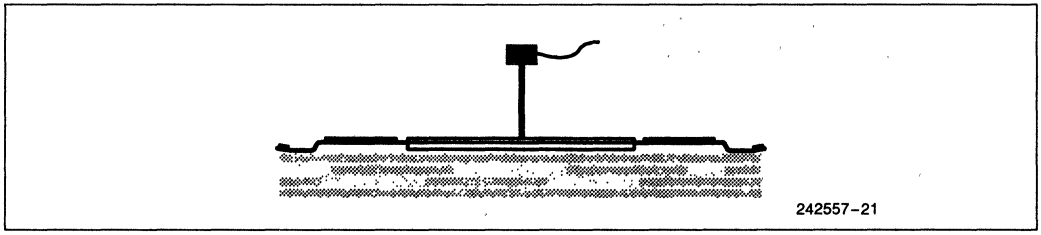


Figure 20. Technique for Measuring Case Temperature (T_C)

Table 24. Thermal Resistance vs. Copper Plane Thickness with and without Enhancements

Copper Plane Thickness*	θ_{CA} (°C/W) No Enhancements	θ_{CA} (°C/W) With Heat Pipe
1 oz. Cu	18	8
3 oz. Cu	14	8

NOTES:
*225 vias underneath the die
(1 oz = 1.3 ml)

Table 25. Thermal Resistance vs. Thermal Vias underneath the Die

Thermal Via Configuration	θ_{CA} (°C/W) No Enhancements
No thermal vias	15
20 mil drill on 40 mil pitch	13

Table 26. Pentium® Processor with Voltage Reduction Technology TCP Package Thermal Resistance without Enhancements

	θ_{JC} (°C/W)	θ_{CA} (°C/W)
Thermal Resistance without Enhancements	0.8	13.9

Table 27. Pentium® Processor with Voltage Reduction Technology TCP Package Thermal Resistance with Enhancements (without Airflow)

Thermal Enhancements	θ_{CA} (°C/W)	Notes
Heat sink	11.7	1.2″ × 1.2″ × .35″
Al Plate	8.7	4″ × 4″ × .030″
Al Plate with Heat Pipe	7.8	0.3″ × 1″ × 4″

Table 28. Pentium® Processor with Voltage Reduction Technology TCP Package Thermal Resistance with Enhancements (with Airflow)

Thermal Enhancements	θ_{CA} (°C/W)	Notes
Heat sink with Fan @ 1.7 CFM	5.0	1.2″ × 1.2″ × .35″ HS 1″ × 1″ × .4″ Fan
Heat sink with Airflow @ 400 LFM	5.1	1.2″ × 1.2″ × .35″ HS
Heat sink with Airflow @ 600 LFM	4.3	1.2″ × 1.2″ × .35″ HS

HS = heat sink
LFM = Linear Feet/Minute
CFM = Cubic Feet/Minute

6.4.1. STANDARD TEST BOARD CONFIGURATION

All Tape Carrier Package (TCP) thermal measurements provided in the following tables were taken with the component soldered to a 2″ x 2″ test board outline. This six-layer board contains 13.5 mil drill on 40 mil pitch vias (underneath the die) in the die attach pad which are connected to two 3 oz. copper planes located at layers two and five. For the TCP Pentium processor with voltage reduction technology, the vias in the die attach pad should be connected without thermal reliefs to the ground plane(s). The die is attached to the die attach pad using a thermally and electrically conductive adhesive. This test board was designed to optimize the heat spreading into the board and the heat transfer through to the opposite side of the board.

NOTE

Thermal resistance values should be used as guidelines only, and are highly system dependent. Final system verification should always refer to the case temperature specification.

7.0. SPGA PENTIUM® PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY SPECIFICATIONS

7.1. SPGA Pentium® Processor with Voltage Reduction Technology Differences from 3.3V Pentium Processor

All SPGA Pentium processor with voltage reduction technology Specifications, except the differences described in this section, are identical to those of the 3.3V Pentium processor.

7.1.1 Features Removed

The following features have been removed for the Pentium processor with voltage reduction technology: Upgrade, Dual Processing (DP), APIC and Master/Checker functional redundancy. Table 1 lists the corresponding pins which exist on the 3.3V Pentium processor but have been removed on the Pentium processor with voltage reduction technology.

7.1.2. Maximum Rating

The following values are stress ratings only. Functional operation at the maximum ratings is not implied nor guaranteed. Functional operating conditions are given in the AC and DC specification tables.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the SPGA Pentium processor with voltage reduction technology contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

Case temperature under bias ..... −65°C to 110°C

Storage temperature ..... −65°C to 150°C

3V Supply voltage with respect to V_{SS} ..... −0.5V to +4.6V

2.9V Supply voltage with respect to V_{SS} ..... −0.5V to +4.1V

3V Only Buffer DC Input Voltage .... −0.5V to $V_{CC}3$ ..... +0.5; not to exceed 4.6V (2)

5V Safe Buffer DC Input Voltage ..... −0.5V to 6.5V (1,3)

NOTES:
1. Applies to CLK.
2. Applies to all SPGA Pentium processor with voltage reduction technology inputs except CLK.
3. See Table 30.

WARNING

Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

7.1.3. DC Specifications

Tables 29, 30 and 31 list the DC specifications which apply to the SPGA Pentium processor with voltage reduction technology. The SPGA Pentium processor with voltage reduction technology core operates at 2.9V internally while the I/O interface operates at 3.3V. The CLK input may be at 3.3V or 5V. Since the 3.3V (5V safe) input levels defined in Table 30 are the same as the 5V TTL levels, the CLK input is compatible with existing 5V clock drivers. The power dissipation specification in Table 32 is provided for design of thermal solutions during operation in a sustained maximum level. This is the worst-case power the device would dissipate in a system for a sustained period of time. This number is used for design of a thermal solution for the device.

Table 29. 3.3V DC Specifications

T_{CASE} = 0 to85°C; $V_{CC}2$ = 2.9V ± 165mV; $V_{CC}3$ = 3.3V ± 165mV

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL3}	Input Low Voltage	−0.3	0.8	V	TTL Level (3)
V_{IH3}	Input High Voltage	2.0	$V_{CC}3$ + 0.3	V	TTL Level (3)
V_{OL3}	Output Low Voltage		0.4	V	TTL Level (1) (3)
V_{OH3}	Output High Voltage	2.4		V	TTL Level (2) (3)
$I_{CC}2$	Power Supply Current from 2.9V core supply		2096 2515 2500	mA mA mA	@75 MHz (4) @90 MHz (4) @120 MHz (4) (5)
$I_{CC}3$	Power Supply Current from 3.3V I/O buffer supply		265 318 320	mA mA mA	@75 MHz (4) @90 MHz (4) @120 MHz (4)

NOTES:
1. Parameter measured at 4 mA.
2. Parameter measured at 3 mA.
3. 3.3V TTL levels apply to all signals except CLK.
4. This value should be used for power supply design. It was estimated for a worst-case instruction mix and $V_{CC}2$ = 2.9V ± 165mV and $V_{CC}3$ = 3.3V ± 165mV. Power supply transient response and decoupling capacitors must be sufficient to handle the instantaneous current changes occurring during transitions from stop clock to full active modes.
5. The lower power number is due to a process improvement.

Table 30. 3.3V (5V Safe) DC Specifications

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL5}	Input Low Voltage	−0.3	0.8	V	TTL Level (1)
V_{IH5}	Input High Voltage	2.0	5.55	V	TTL Level (1)

NOTES:
1. Applies to CLK only.

Table 31. Input and Output Characteristics

Symbol	Parameter	Min	Max	Unit	Notes
C_{IN}	Input Capacitance		15	pF	(4)
C_O	Output Capacitance		20	pF	(4)
$C_{I/O}$	I/O Capacitance		25	pF	(4)
C_{CLK}	CLK Input Capacitance		15	pF	(4)
C_{TIN}	Test Input Capacitance		15	pF	(4)
C_{TOUT}	Test Output Capacitance		20	pF	(4)
C_{TCK}	Test Clock Capacitance		15	pF	(4)
I_{LI}	Input Leakage Current		±15	μA	$0 < V_{IN} < V_{CC}3$ (1)
I_{LO}	Output Leakage Current		±15	μA	$0 < V_{IN} < V_{CC}3$ (1)
I_{IH}	Input Leakage Current		200	μA	V_{IN} = 2.4V (3)
I_{IL}	Input Leakage Current		−400	μA	V_{IN} = 0.4V (2)

NOTES:
1. This parameter is for input without pull up or pull down.
2. This parameter is for input with pull up.
3. This parameter is for input with pull down.
4. Guaranteed by design.

Table 32. Power Dissipation Requirements for Thermal Solution Design

Parameter	Typical(1)	Max(2)	Unit	Notes
Active Power Dissipation	2.0-3.0	6.0	Watts	@75 MHz
	2.5-3.5	7.3	Watts	@90 MHz
	2.5-3.5	7.1	Watts	@120 MHz (5)
Stop Grant and Auto Halt		1.0	Watts	@75 MHz
Powerdown Power Dissipation		1.2	Watts	@90 MHz (3)
		1.2	Watts	@120 MHz (3)
Stop Clock Power Dissipation	.02	0.05	Watts	(4)

NOTES:
1. This is the typical power dissipation in a system. This value was the average value measured in a system using a typical device at $V_{CC}2$ = 2.9V and $V_{CC}3$ = 3.3V running typical applications. This value is highly dependent upon the specific system configuration.
2. Systems must be designed to thermally dissipate the maximum active power dissipation. It is determined using a worst-case instruction mix with $V_{CC}2$ = 2.9V and $V_{CC}3$ = 3.3V. The use of nominal V_{CC} in this measurement takes into account the thermal time constant of the package.
3. Stop Grant/Auto Halt Powerdown Power Dissipation is determined by asserting the STPCLK# pin or executing the HALT instruction.
4. Stop Clock Power Dissipation is determined by asserting the STPCLK# pin and then removing the external CLK input.
5. The lower power number is due to a process improvement.

7.1.3.1. Power Sequencing

There is no specific sequence required for powering up or powering down the $V_{CC}2$ and $V_{CC}3$ power supplies. However, for compatibility with future mobile processors, it is recommended that the $V_{CC}2$ and $V_{CC}3$ power supplies be either both on or both off within one second of each other.

7.1.4. AC Specifications

The AC specifications of the SPGA Pentium processor with voltage reduction technology consist of set-up times, hold times, and valid delays at 0 pF. All SPGA Pentium processor with voltage reduction technology AC specifications are valid for $V_{CC}2$ = 2.9V + 165mV, $V_{CC}3$ = 3.3V + 165mV, and Tcase = 0 to 85°C.

WARNING

Do not exceed the 75-MHz Pentium processor with voltage reduction technology internal maximum frequency of 75 MHz by either selecting the ½ bus fraction or providing a clock greater than 50 MHz.

Do not exceed the 90-MHz Pentium processor with voltage reduction technology internal maximum frequency of 90 MHz by either selecting the ½ bus fraction or providing a clock greater than 60 MHz.

7.1.4.1. Power and Ground

For clean on-chip power distribution, the SPGA Pentium processor with voltage reduction technology has 25 $V_{CC}2$ (2.9V power), 28 $V_{CC}3$ (3.3V power) and 53 V_{SS} (ground) inputs. Power and ground connections must be made to all external $V_{CC}2$, $V_{CC}3$ and V_{SS} pins of the SPGA Pentium processor with voltage reduction technology. On the circuit board all $V_{CC}2$ pins must be connected to a 2.9V $V_{CC}2$ plane (or island) and all $V_{CC}3$ pins must be connected to a 3.3V $V_{CC}3$ plane. All V_{SS} pins must be connected to a V_{SS} plane. Refer to Table 35 for a listing of $V_{CC}2$ and $V_{CC}3$.

7.1.4.2. Decoupling Recommendations

Transient power surges can occur as the processor is executing instruction sequences or driving large loads. To mitigate these high frequency transients, liberal high frequency decoupling capacitors should be placed near the processor.

Low inductance capacitors and interconnects are recommended for best high frequency electrical performance. Inductance can be reduced by shortening circuit board traces between the processor and decoupling capacitors as much as possible.

These capacitors should be evenly distributed around each component on the 3.3V plane and the 2.9V plane (or island). Capacitor values should be chosen to ensure they eliminate both low and high frequency noise components.

Power transients also occur as the processor rapidly transitions from a low level of power consumption to a much higher level (or high to low power). A typical example would be entering or exiting the Stop Grant state. Another example would be executing a HALT instruction, causing the processor to enter the Auto HALT Powerdown state, or transitioning from HALT to the Normal state. All of these examples may cause abrupt changes in the power being consumed by the processor. Note that the Auto HALT Powerdown feature is always enabled even when other power management features are not implemented.

Bulk storage capacitors with a low ESR (Effective Series Resistance) in the 10 to 100 μF range are required to maintain a regulated supply voltage during the interval between the time the current load changes and the point that the regulated power supply output can react to the change in load. In order to reduce the ESR, it may be necessary to place several bulk storage capacitors in parallel.

These capacitors should be placed near the processor (on the 3.3V plane and the 2.9V plane (or island)) to ensure that these supply voltages stay within specified limits during changes in the supply current during operation.

For more detailed informations, please contact Intel or refer to the *Pentium Processor with Voltage Reduction Technology: Power Supply Design Considerations for Mobile Systems* application note (Order Number 242558).

7.1.4.3. Connection Specifications

All NC pins must remain unconnected. Refer to Table 35 for a listing of NC pins.

All RESERVED pins must remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Unused active low inputs should be connected to $V_{CC}3$. Unused active high inputs should be connected to ground.

7.1.4.4. AC Timings

Table 33 contains the SPGA Pentium processor with voltage reduction technology AC timing changes for 50-MHz bus operation. Table 34 contains the SPGA Pentium processor with voltage reduction technology AC timing changes for 60-MHz bus operation.

7.1.5. Thermal Specifications

The SPGA Pentium processor with voltage reduction technology is specified for proper operation when the case temperature, T_{CASE} (T_C) is within the specified range of 0 °C to 85 °C.

7.1.6. SPGA Package Differences

The SPGA Pentium processor with voltage reduction technology package has a pin array that is mechanically identical to the SPGA version of the 3.3V Pentium, but some pins need to be connected differently. Also, there are small differences in the package dimensions.

7.1.6.1. Pinout

Table 35 lists the SPGA Pentium processor with voltage reduction technology pins that are different from the SPGA 3.3V Pentium processor. Figure 21 depicts the pin side SPGA pinout diagram. The $V_{CC}2$ pins are 3.3V V_{CC} pins for the 3.3V Pentium processor, but will be 2.9V $V_{CC}2$ pins for the SPGA Pentium processor with voltage reduction technology. The NC pins correspond to the unused (for mobile) functions listed in Table 1. They should be left unconnected. Connection of these pins may result in component failure or incompatibility with processor steppings. For a brief functional description of the remaining pins, please refer to Tables 3 and 4. For Input and Output pins reference, please refer to Table 5, 6 and 7.

7.1.6.2. Package Dimensions

The Pentium processor with voltage reduction technology implements an SPGA package that removes the Heat spreader from the top of the package. The package is mechanically equivalent to the package used on the 3.3V Pentium processor C2 stepping except that the SPGA Pentium processor with voltage reduction technology will use the metal lid instead of a ceramic lid, and has the dimensions shown in Figure 22.

7.1.7. I/O Buffer Models

The I/O buffer models provided in section 4.4 of this document apply to both the TCP and SPGA Pentium processor with voltage reduction technology packages, although the capacitance (C_p) and inductance (L_p) parameter values differ between the two packages. For SPGA Pentium processor with voltage reduction technology values, refer to Chapter 24 of the *Pentium® Processor Family Developer's Manual, Volume 1: Pentium® Processors*.

Table 33. SPGA Pentium® Processor with Voltage Reduction Technology AC Timing Changes for 50-MHz Bus Operation

$V_{CC}2$ = 2.9V +165mV, $V_{CC}3$ = 3.3V +165mV, T_{case} = 0°C to 85°C, C_L = 0pF

Symbol	Parameter	Min	Max	Unit
t_{6a}	W/R# Valid Delay	1.0	7.9	ns
t_{6a}	BE0-7# Valid Delay	1.0	8.1	ns
t_{6c}	LOCK# Valid Delay	1.1	7.9	ns
t_{6a}	PWT Valid Delay	1.0	7.5	ns
t_{6a}	CACHE# Valid Delay	1.0	7.3	ns
t_{6c}	A3-A31 Valid Delay	1.1	8.3	ns
t_{6d}	ADS# Valid Delay	1.0	7.4	ns
t_{10b}	HITM# Valid Delay	1.1	6.6	ns
t_{12}	DP, DBUS Valid Delay	1.3	9.2	ns

Table 34. SPGA Pentium® Processor with Voltage Reduction Technology AC Timing Changes for 60-MHz Bus Operation

$V_{CC}2$ = 2.9V +165mV, $V_{CC}3$ = 3.3V +165mV, T_{case} = 0°C to 85°C, CL = 0pF

Symbol	Parameter	Min	Max	Unit
t_{6c}	A3-A31 Valid Delay	1.1	7.7	ns
t_{12}	DP, DBUS Valid Delay	1.3	7.8	ns

Table 35. SPGA Pentium® Processor with Voltage Reduction Technology $V_{CC}2$ and $V_{CC}3$ Pins

$V_{CC}2$*				
A17	A07	Q01	AA01	AN11
A15	G01	S01	AC01	AN13
A13	J01	U01	AE01	AN15
A11	L01	W01	AG01	AN17
A09	N01	Y01	AN09	AN19

$V_{CC}3$*						
A19	A27	AE37	AN25	G37	N37	U33
A21	A29	AG37	AN27	J37	Q37	U37
A23	AA37	AN21	AN29	L33	S37	W37
A25	AC37	AN23	E37	L37	T34	Y37

NC**				
A37	AE03	AN35	Q35	W33
AA03	AE35	H34	R34	W35
AC03	AL19	J33	S33	Y03
AD04	AM02	L35	S35	Y35

NOTE:
*These $V_{CC}2$ pins are 3.3V V_{CC} pins for the SPGA 3.3V Pentium® processor. For the SPGA Pentium processor with voltage reduction technology, these pins are 2.9V $V_{CC}2$ supplies for the SPGA core.
**These NC pins should be left unconnected. Connection of these pins may result in component failure or incompatibility with processor steppings.

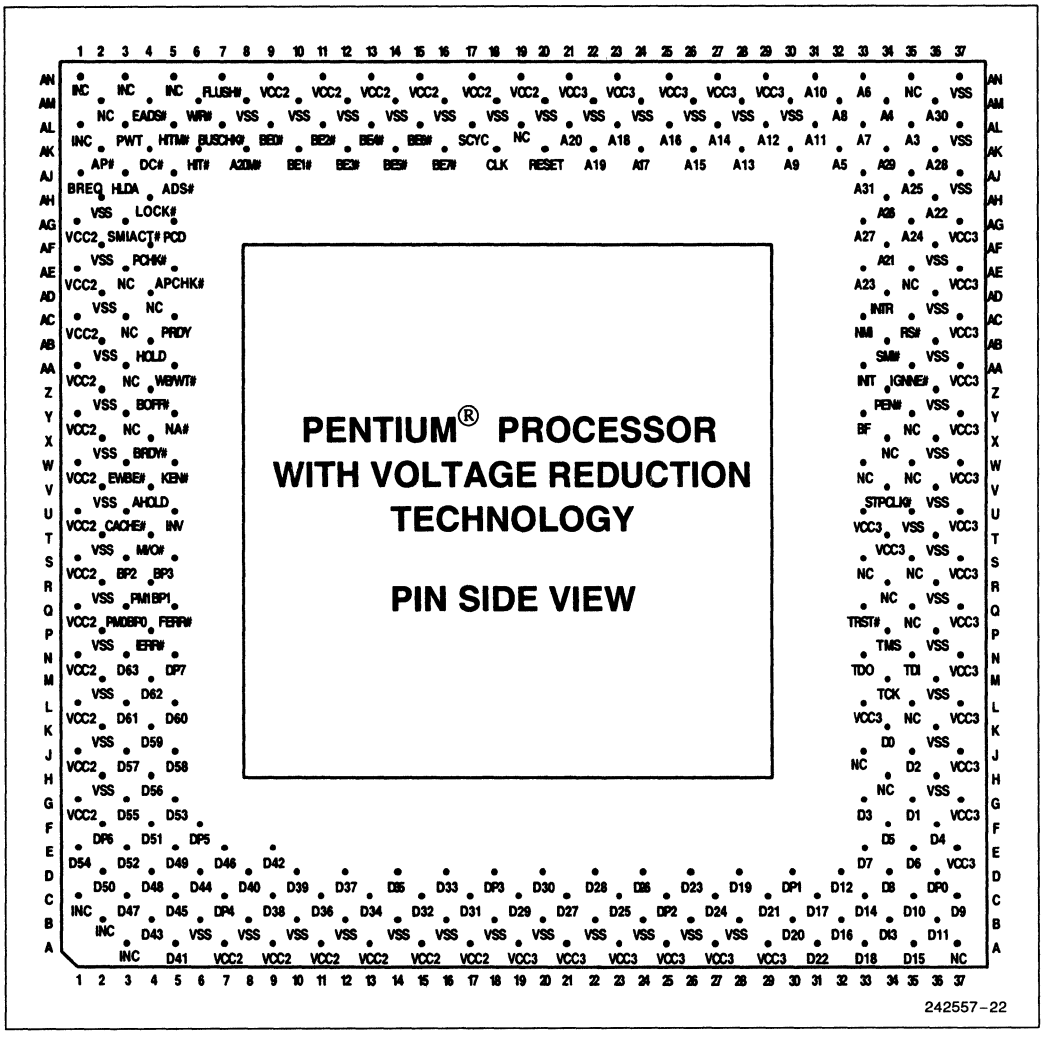


Figure 21. SPGA Pentium® Processor with Voltage Reduction Technology Pinout

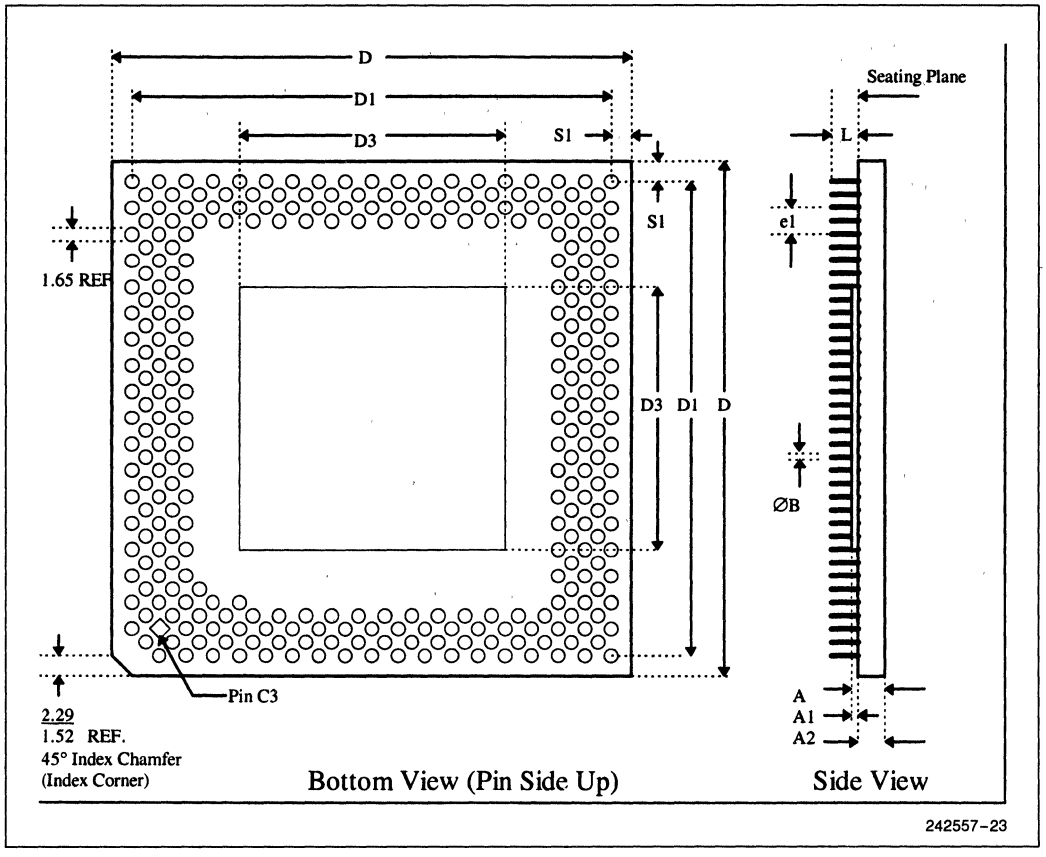


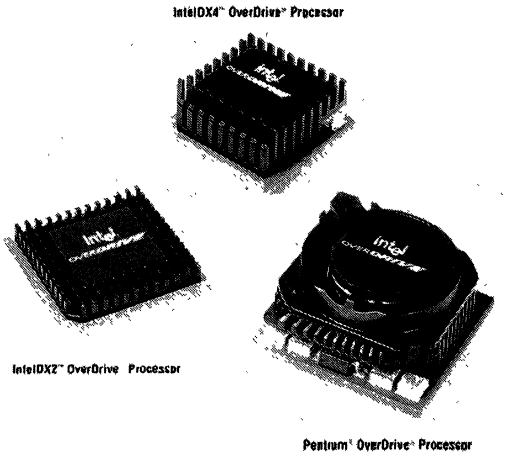
Figure 22. 296-Pin Ceramic Pin Grid Array Package

Table 36. 296-Pin Ceramic Pin Grid Array: The Package Dimensional Specification

Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A	3.27	3.83	Ceramic Lid	0.129	0.151	Ceramic Lid
A1	0.66	0.86	Ceramic Lid	0.026	0.034	Ceramic Lid
A2	2.62	2.97		0.103	0.117	
B	0.43	0.51		0.017	0.020	
D	49.28	49.78		1.940	1.960	
D1	45.59	45.85		1.795	1.805	
D3	24.00	24.25	Includes Fillet	0.945	0.955	Includes Fillet
e1	2.29	2.79		0.090	0.110	
L	3.05	3.30		0.120	1.130	
N	296		Total Pins	296		Total Pins
S1	1.52	2.54		0.060	0.100	

INTEL OverDrive® PROCESSORS

- **Powerful Processor Upgrades for most Intel486™ Microprocessor-Based Systems**
  - **— Significantly Accelerates All Software Applications**
- **Intel OverDrive® Processor Family Includes:**
  - **— Pentium® OverDrive Processor**
  - **— IntelDX4™ OverDrive Processor**
  - **— IntelDX2™ OverDrive Processor**
- **Designed to Upgrade Systems Based on:**
  - **— Intel486™ SX Processors**
  - **— Intel486™ DX Processors**
  - **— IntelSX2™ Processors**
  - **— IntelDX2™ Processors**
- **Large Installed Base of Thousands of Applications**
- **Incorporates SMM Power Saving Features**



290436-61

The Pentium® OverDrive® processor upgrades most Intel486 processor-based systems to Pentium processor technology. It is the recommended upgrade option for IntelSX2™ and IntelDX2™ CPU-based systems, and the superior upgrade option for Intel486 SX and DX CPU-based systems. It features a true Pentium processor core (superscalar architecture, branch prediction and faster floating point unit), silicon enhancements (separate code and data caches, 16 KB each and 32-bit bus interface), and package innovations (on-package voltage regulation and fan heat sink).

The IntelDX4™ OverDrive processor is an upgrade for most Intel486 SX and DX CPU-based systems. It features Intel's speed-tripling technology, enhanced 16 KB on-chip cache memory and a math coprocessor.

The IntelDX2 OverDrive processor is an entry-level upgrade for most Intel486 SX and DX CPU-based systems. It features Intel's speed-doubling technology, on-chip math coprocessor and 8 KB on-chip cache memory.

*Other brands and names are the property of their respective owners.

December 1995
Order Number: 290436-006

Intel OverDrive® Processors

CONTENTS

CONTENTS

PAGE

CONTENTS

CONTENTS

1.0 INTRODUCTION

This data sheet describes the Intel OverDrive processors, a family of CPU upgrades for Intel486 processor-based systems. This family includes the IntelDX2 OverDrive processor, the IntelDX4 OverDrive processor and the Pentium OverDrive processor. These processor upgrades significantly accelerate all software applications, thereby increasing overall PC performance.

It is important to note that this data sheet is intended to be used in conjunction with the Intel486 Microprocessor Family Datasheet—which describes the Intel Family Architecture and functionality (Order # 242202-003). All enhancements or differences between the OverDrive processor and the original processor (i.e., IntelDX2 or IntelDX4 OverDrive vs. Intel486 DX processor, Pentium vs. Pentium OverDrive processors) are described in this data sheet.

Intel486 SX, Intel486 DX, IntelSX2, or IntelDX2 processor-based systems that are compatible to the Intel OverDrive processor(s) must be designed to both the original processor specifications and the Intel OverDrive processor(s) specifications.

1.1 Product Overview

The following sections provide an overview of each of the OverDrive processors. Refer to the specific product section(s) for more detailed information.

Figure 1-1 lists some of the key features of each OverDrive processor. Figure 1-2 describes the upgrade choices available for an existing Intel486 SX or DX system.

2

IntelDX2™ OverDrive® Processor	• For Up To 33 MHz Intel486 SX or DX Processor-Based Systems • Based on IntelDX2 Technology, which Includes Speed Doubling and Floating Point Math Unit • Intel486 DX Processor and Intel487 Math CoProcessor Pinouts • 8 KByte Integrated Instruction and Data Cache
IntelDX4™ OverDrive Processor	• For Up To 33 MHz Intel486 SX or DX Processor-Based Systems • Based on IntelDX4 Speed Tripling Technology, which Includes Floating Point Math Unit • Intel486 DX Processor and Intel487 Math CoProcessor Pinouts • 16 KByte Integrated Instruction and Data Cache
Pentium® OverDrive Processor	• For Up To 33 MHz Intel486 SX or DX Processor-Based Systems • For Up To 66 MHz IntelDX2 or 50 MHz IntelSX2 Processor-Based Systems • Based on Intel Pentium Processor Technology, which includes Superscalar, Faster Floating Point, Branch Prediction • Separate Code and Data Caches, 16 KByte each, Data Cache has Write-Back Capability • Integrated Fan Heat Sink

290436-27

Figure 1-1. Key Features

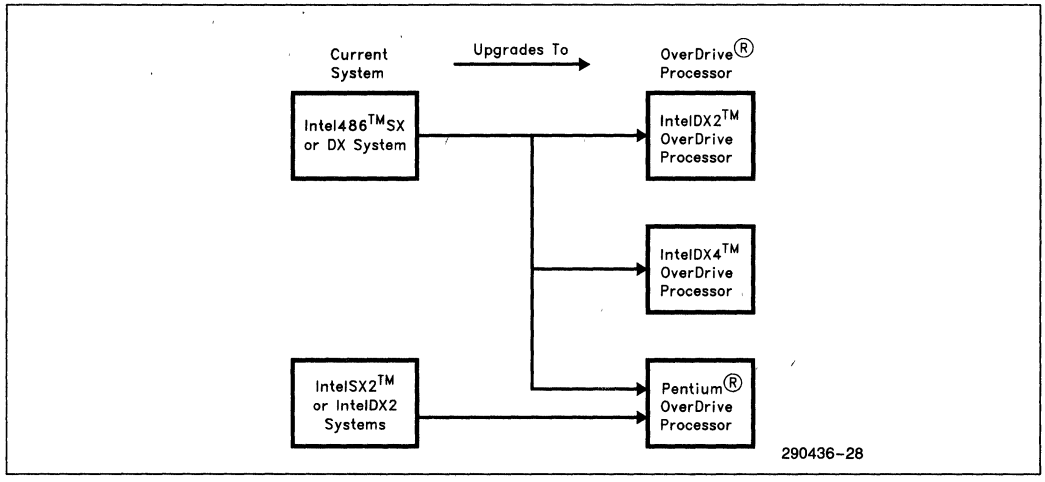


Figure 1-2. Upgrade Choices

1.1.1 IntelDX2™ OverDrive® PROCESSOR

The IntelDX2 OverDrive processor is the entry-level processor upgrade designed for most Intel486 SX and Intel486 DX processor-based systems. Based on the IntelDX2 processor, it features the Intel speed doubling technology. This accelerates both integer and floating point software, to deliver performance equivalent to a similarly configured IntelDX2 processor-based system.

The IntelDX2 OverDrive processor integrates an integer unit, a floating point math coprocessor unit, a memory management unit and an 8 KByte cache on a single chip. The speed doubling technology allows the processor to operate internally at twice the speed of the system bus; up to a maximum of 66 MHz for a 33 MHz system.

The IntelDX2 OverDrive processor comes in two package offerings; 168-lead Pin Grid Array (PGA) and 169-lead PGA. It is designed to be installed into the OverDrive processor socket of Intel486 SX and DX processor-based systems. It can also replace the existing processor in single-socket systems.

1.1.2 IntelDX4™ OverDrive® PROCESSOR

The IntelDX4 OverDrive processor is an upgrade designed for most Intel486 SX and Intel486 DX processor-based systems. Utilizing the Intel speed tripling technology, the IntelDX4 OverDrive processor accelerates both integer and floating point software, achieving performance comparable to a similarly configured IntelDX4 processor-based system.

The IntelDX4 OverDrive processor integrates an integer unit, a floating point math coprocessor unit, a memory management unit and a 16-KByte cache on a single chip. The speed tripling technology allows the processor to operate internally at three times the speed of the system bus; up to a maximum of 100 MHz for a 33 MHz system.

The IntelDX4 OverDrive processor comes in two package offerings; 168-lead Pin Grid Array (PGA) and 169-lead PGA. It is designed to be installed into the OverDrive processor socket of Intel486 SX and DX processor-based systems. It can also replace the existing processor in most single-socket systems.

1.1.3 Pentium® OverDrive® PROCESSOR

The Pentium OverDrive processor is the highest performance CPU upgrade available for systems based on the Intel486 family of CPUs, bringing Pentium processor technology (including Superscalar Architecture, Branch Prediction, faster floating-point unit, and separate data and code caches) to most Intel486 processor-based systems. It is the recommended upgrade option for most IntelSX2 and IntelDX2 processor-based systems, and the superior upgrade option for most Intel486 SX and DX processor-based systems.

Inclusion of the Pentium OverDrive processor socket in systems based on the Intel486 family of microprocessors provides the end user with an easy and cost-effective way to increase system performance for most Intel486 processor-based systems. The majority of upgrade installations which take advantage of the Pentium OverDrive processor socket will be performed by end users and resellers. Therefore, it is important that the design be "end user easy", and that the amount of training and technical expertise required to install the OverDrive processors be minimized. Upgrade installation instructions should be clearly described in the system user's manual. In addition, by making installation simple and foolproof, PC manufacturers can reduce the risk of system damage, warranty claims and service calls. Feedback from Intel's upgrade customers highlight three main characteristics of end user easy designs: accessible socket location, clear indication of upgrade component orientation, and minimization of insertion force. Recommendations regarding designing for easy upgradability appear in Appendix C.

1.2 Pinouts

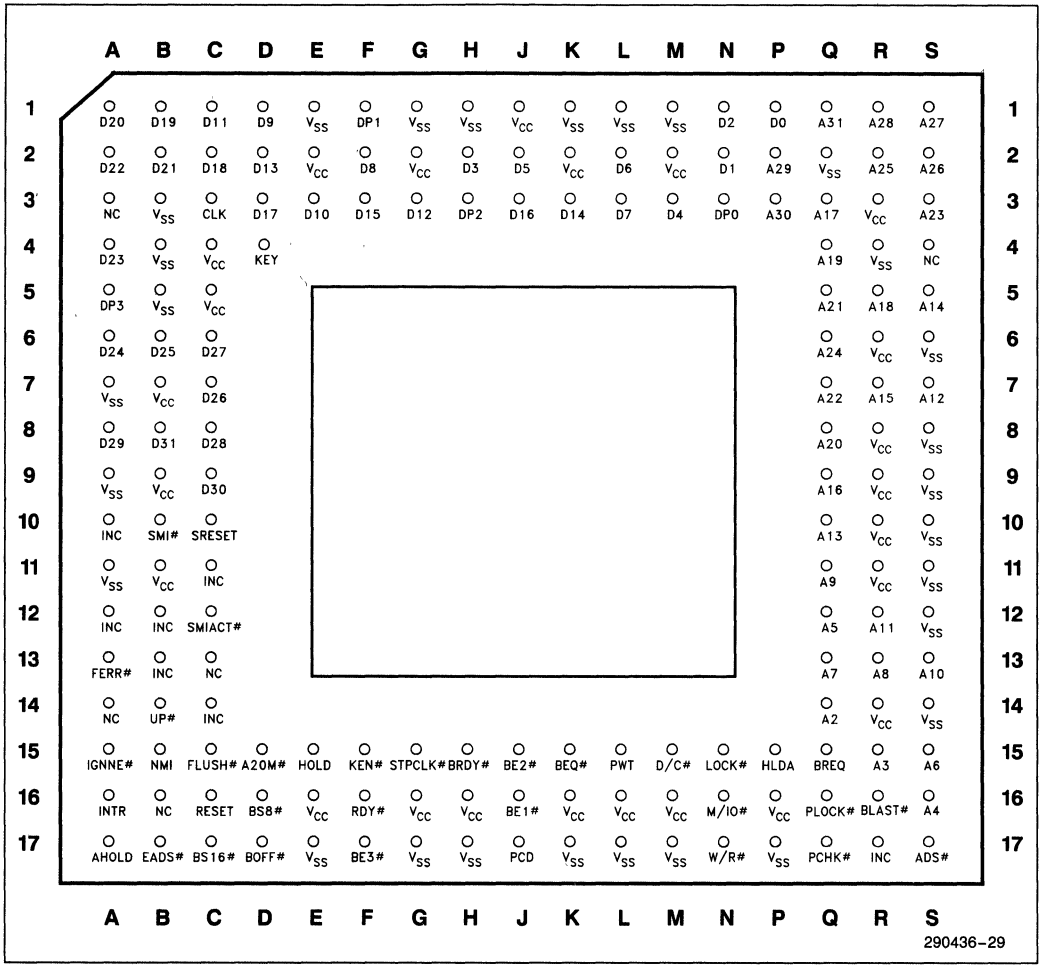
1.2.1 168/169 PIN SOCKET

Refer to Figures 1-3 and 1-4 for an illustration of each of the two PGA packages. Figure 1-3 shows the 169-lead PGA package, while Figure 1-4 illustrates the 168-lead PGA package.

Table 1-1 cross-references the pin number to pin function for the 169-lead PGA package. Table 1-2 is a cross-reference for the 168-lead package.

Table 5-1 in Section 5 gives a brief description of the function of each pin.

Refer to each specific OverDrive processor section for a description of any differences from the pinouts described in this section.



	A	B	C	D	E	F	G	H	J	K	L	M	N	P	Q	R	S	
1	D20	D19	D11	D9	V_{SS}	DP1	V_{SS}	V_{SS}	V_{CC}	V_{SS}	V_{SS}	V_{SS}	D2	D0	A31	A28	A27	1
2	D22	D21	D18	D13	V_{CC}	D8	V_{CC}	D3	D5	V_{CC}	D6	V_{CC}	D1	A29	V_{SS}	A25	A26	2
3	NC	V_{SS}	CLK	D17	D10	D15	D12	DP2	D16	D14	D7	D4	DP0	A30	A17	V_{CC}	A23	3
4	D23	V_{SS}	V_{CC}	KEY											A19	V_{SS}	NC	4
5	DP3	V_{SS}	V_{CC}												A21	A18	A14	5
6	D24	D25	D27												A24	V_{CC}	V_{SS}	6
7	V_{SS}	V_{CC}	D26												A22	A15	A12	7
8	D29	D31	D28												A20	V_{CC}	V_{SS}	8
9	V_{SS}	V_{CC}	D30												A16	V_{CC}	V_{SS}	9
10	INC	SMI#	SRESET												A13	V_{CC}	V_{SS}	10
11	V_{SS}	V_{CC}	INC												A9	V_{CC}	V_{SS}	11
12	INC	INC	SMIACT#												A5	A11	V_{SS}	12
13	FERR#	INC	NC												A7	A8	A10	13
14	NC	UP#	INC												A2	V_{CC}	V_{SS}	14
15	IGNNE#	NMI	FLUSH#	A20M#	HOLD	KEN#	STPCLK#	BRDY#	BE2#	BEQ#	PWT	D/C#	LOCK#	HLDA	BREQ	A3	A6	15
16	INTR	NC	RESET	BS8#	V_{CC}	RDY#	V_{CC}	V_{CC}	BE1#	V_{CC}	V_{CC}	V_{CC}	M/IO#	V_{CC}	PLOCK#	BLAST#	A4	16
17	AHOLD	EADS#	BS16#	BOFF#	V_{SS}	BE3#	V_{SS}	V_{SS}	PCD	V_{SS}	V_{SS}	V_{SS}	W/R#	V_{SS}	PCHK#	INC	ADS#	17
	A	B	C	D	E	F	G	H	J	K	L	M	N	P	Q	R	S	

290436-29

Figure 1-3. 169-Lead PGA Bottom View Pinout (ODP)

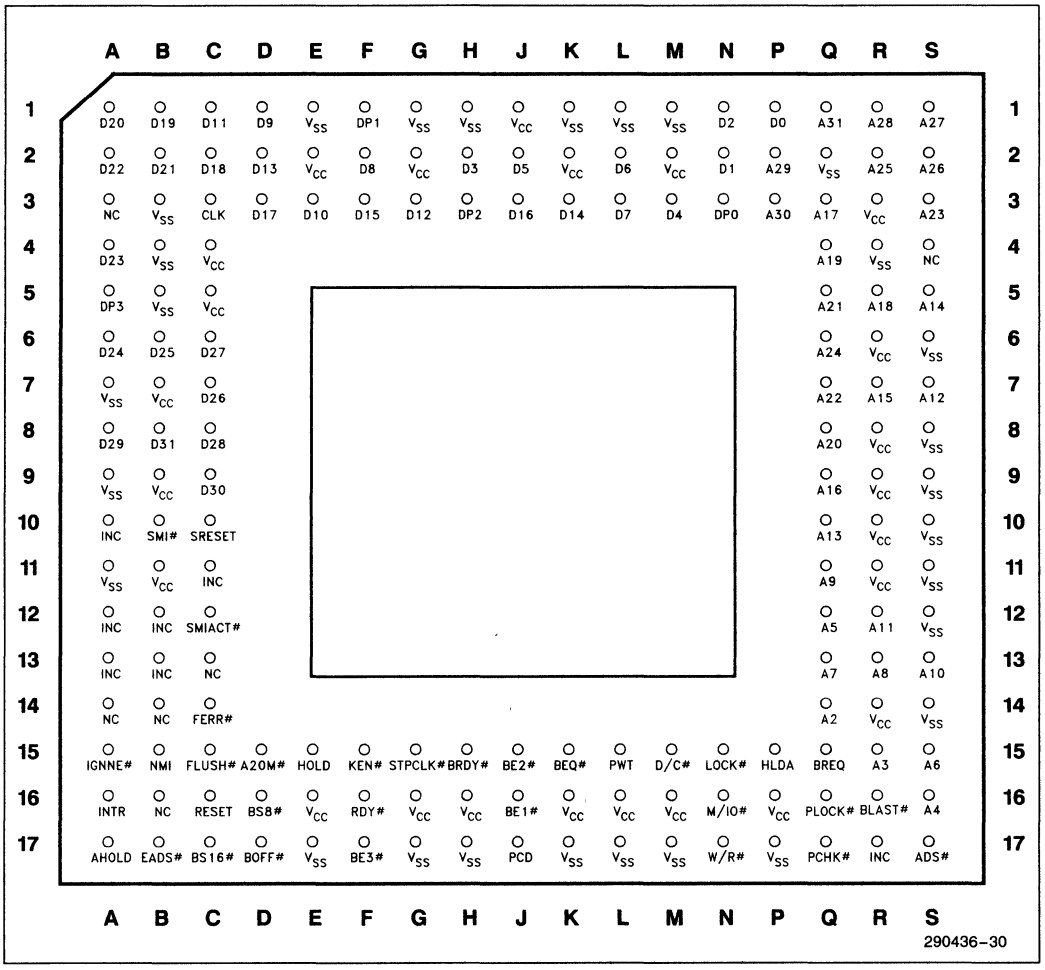


Figure 1-4. 168-Lead PGA Bottom View Pinout (ODPR)

Table 1-1. 169-Lead PGA Pin Cross Reference by Pin Name (ODP)

Address		Data		Control		Control		NC	V_{CC}	V_{SS}
A_2	Q14	D_0	P1	A20M#	D15	PLOCK#	Q16	A3	B7	A7
A_3	R15	D_1	N2	ADS#	S17	PWT	L15	A14	B9	A9
A_4	S16	D_2	N1	AHOLD	A17	RDY#	F16	B16	B11	A11
A_5	Q12	D_3	H2	BE0#	K15	RESET	C16	C13	C4	B3
A_6	S15	D_4	M3	BE1#	J16	SMI#	B10		C5	B4
A_7	Q13	D_5	J2	BE2#	J15	SMIACT#	C12		E2	B5
A_8	R13	D_6	L2	BE3#	F17	SRESET	C10		E16	E1
A_9	Q11	D_7	L3	BLAST#	R16	STPCLK#	G15		G2	E17
A_{10}	S13	D_8	F2	BOFF#	D17	UP#	B14		G16	G1
A_{11}	R12	D_9	D1	BRDY#	H15	W/R#	N17		H16	G17
A_{12}	S7	D_{10}	E3	BREQ#	Q15				J1	H1
A_{13}	Q10	D_{11}	C1	BS8#	D16				K2	H17
A_{14}	S5	D_{12}	G3	BS16#	C17				K16	K1
A_{15}	R7	D_{13}	D2	CLK	C3	**Position**			L16	K17
A_{16}	Q9	D_{14}	K3	D/C#	M15				M2	L1
A_{17}	Q3	D_{15}	F3	DP0	N3	KEY	D4		M16	L17
A_{18}	R5	D_{16}	J3	DP1	F1	PLUG	D5		P16	M1
A_{19}	Q4	D_{17}	D3	DP2	H3	PLUG	D13		R3	M17
A_{20}	Q8	D_{18}	C2	DP3	A5	PLUG	D14		R6	P17
A_{21}	Q5	D_{19}	B1	EADS#	B17	PLUG	E4	**INC**	R8	Q2
A_{22}	Q7	D_{20}	A1	FERR#	A13	PLUG	E14		R9	R4
A_{23}	S3	D_{21}	B2	FLUSH#	C15	PLUG	N4	A10	R10	S6
A_{24}	Q6	D_{22}	A2	HLDA	P15	PLUG	N14	A12	R11	S8
A_{25}	R2	D_{23}	A4	HOLD	E15	PLUG	P4	B12	R14	S9
A_{26}	S2	D_{24}	A6	IGNNE#	A15	PLUG	P5	B13		S10
A_{27}	S1	D_{25}	B6	INTR	A16	PLUG	P13	C11		S11
A_{28}	R1	D_{26}	C7	KEN#	F15	PLUG	P14	C14		S12
A_{29}	P2	D_{27}	C6	LOCK#	N15			R17		S14
A_{30}	P3	D_{28}	C8	M/IO#	N16			S4		
A_{31}	Q1	D_{29}	A8	NMI	B15					
		D_{30}	C9	PCD	J17					
		D_{31}	B8	PCHK#	Q17					

NOTES:
1. All NC pins must remain unconnected.
2. Refer to each specific OverDrive section for differences in pin functions.
3. NC = No Connection.
4. INC = Internal No Connect.

Table 1-2. 168-Lead PGA Pin Cross Reference by Pin Name (ODPR)

Address		Data		Control		Control		N/C	V_{CC}	V_{SS}
A_2	Q14	D_0	P1	A20M#	D15	PLOCK#	Q16	A3	B7	A7
A_3	R15	D_1	N2	ADS#	S17	PWT	L15	A14	B9	A9
A_4	S16	D_2	N1	AHOLD	A17	RDY#	F16	B14	B11	A11
A_5	Q12	D_3	H2	BE0#	K15	RESET	C16	B16	C4	B3
A_6	S15	D_4	M3	BE1#	J16	SMI#	B10	C13	C5	B4
A_7	Q13	D_5	J2	BE2#	J15	SMIACT#	C12		E2	B5
A_8	R13	D_6	L2	BE3#	F17	SRESET	C10		E16	E1
A_9	Q11	D_7	L3	BLAST#	R16	STPCLK#	G15		G2	E17
A_{10}	S13	D_8	F2	BOFF#	D17	UP#	C11		G16	G1
A_{11}	R12	D_9	D1	BRDY#	H15	W/R#	N17		H16	G17
A_{12}	S7	D_{10}	E3	BREQ#	Q15				J1	H1
A_{13}	Q10	D_{11}	C1	BS8#	D16				K2	H17
A_{14}	S5	D_{12}	G3	BS16#	C17				K16	K1
A_{15}	R7	D_{13}	D2	CLK	C3				L16	K17
A_{16}	Q9	D_{14}	K3	D/C#	M15				M2	L1
A_{17}	Q3	D_{15}	F3	DP0	N3				M16	L17
A_{18}	R5	D_{16}	J3	DP1	F1				P16	M1
A_{19}	Q4	D_{17}	D3	DP2	H3				R3	M17
A_{20}	Q8	D_{18}	C2	DP3	A5			**INC**	R6	P17
A_{21}	Q5	D_{19}	B1	EADS#	B17				R8	Q2
A_{22}	Q7	D_{20}	A1	FERR#	C14			A10	R9	R4
A_{23}	S3	D_{21}	B2	FLUSH#	C15			A12	R10	S6
A_{24}	Q6	D_{22}	A2	HLDA	P15			A13	R11	S8
A_{25}	R2	D_{23}	A4	HOLD	E15			B12	R14	S9
A_{26}	S2	D_{24}	A6	IGNNE#	A15			B13		S10
A_{27}	S1	D_{25}	B6	INTR	A16			R17		S11
A_{28}	R1	D_{26}	C7	KEN#	F15			S4		S12
A_{29}	P2	D_{27}	C6	LOCK#	N15					S14
A_{30}	P3	D_{28}	C8	M/IO#	N16					
A_{31}	Q1	D_{29}	A8	NMI	B15					
		D_{30}	C9	PCD	J17					
		D_{31}	B8	PCHK#	Q17					

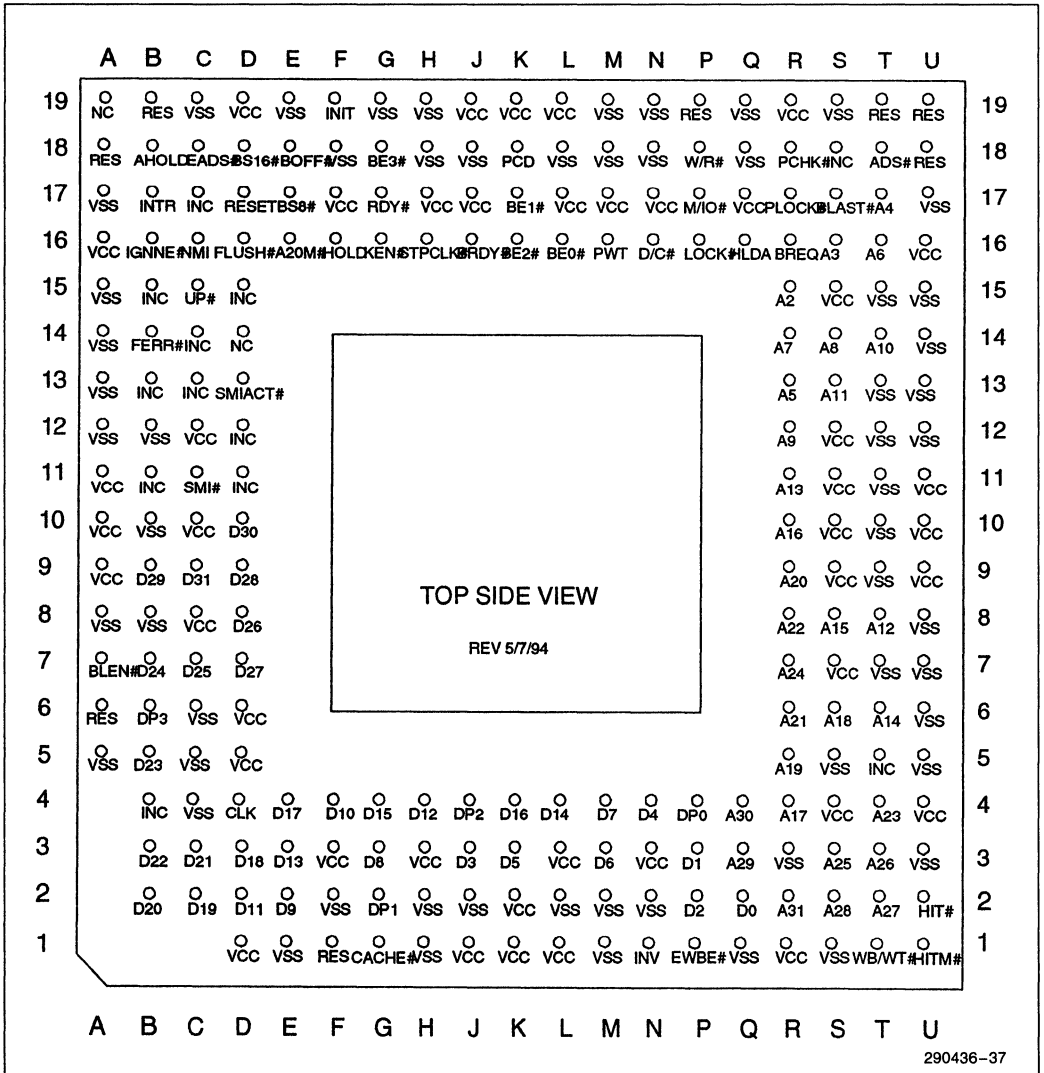
NOTES:
1. All NC pins must remain unconnected.
2. Refer to each specific OverDrive section for differences in pin functions.
3. NC = No Connection.
4. INC = Internal No Connect.

1.2.2 Pentium® OverDrive® PROCESSOR SPECIFICATIONS

The Intel Pentium OverDrive processor socket specifies 237 contacts. The 237 contacts correspond to a standard 240 pin socket with one inside "KEY" contact, one outer "KEY" contact and four 'orientation' contacts plugged on the outside corner. The inside "KEY" contact provides backward compatibility for the IntelDX2 and IntelDX4 OverDrive processors for Intel486 SX, Intel486 DX, IntelSX2, and IntelDX2 processor-based systems. The Pentium OverDrive processor itself (not the socket) **does not have** any "KEY" pins. The five contacts plugged on the outside corner ensure proper orientation for the Pentium OverDrive processor. The Pentium OverDrive processor pinout is shown in Figures 1-5 and 1-6.

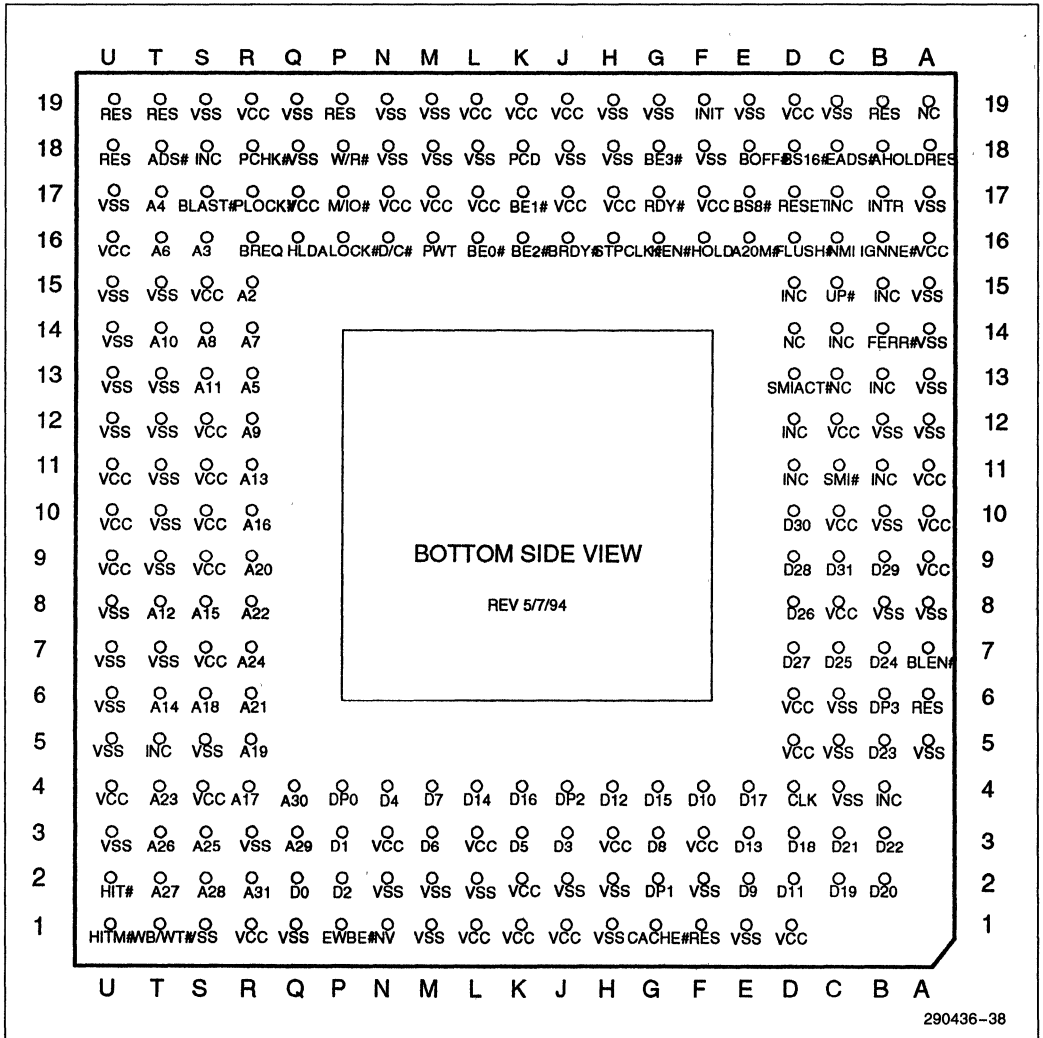
Please note that the boundary scan pins (**TCK, TDO,TDI,** and **TMS**), and all testability pins have been removed from the production version of the Pentium OverDrive processor. An engineering sample will be available that will allow the use of boundary scan and testability functions. For more information on boundary scan and testability pins, please contact Intel.

1.2.2.1 Pentium® OverDrive® Processor Pinout



***NOTE:**
Pin A1 is located in the lower left corner to make the labels on the top plate of Intel processors orient right side up. All "RES" and "NC" pins should be left unconnected to insure proper operation.

Figure 1-5. Pentium® OverDrive® Processor Pinout (Top Side View)



***NOTE:**
Pin A1 is located in the lower left corner to make the labels on the top plate of Intel processors orient right side up. All "RES" and "NC" pins should be left unconnected to insure proper operation.

Figure 1-6. Pentium® OverDrive® Processor Pinout (Bottom Side View)

1.2.2.2 Pin Cross Reference

Table 1-3. Pentium® OverDrive® Processor Pin Cross Reference

Address		Data		Control				N/C	V_{CC}		V_{SS}	
A2	R15	D0	Q2	A20M#	E16	INV	N1	A19	A9	L1	A5	M18
A3	S16	D1	P3	ADS#	T18	KEN#	G16	D14	A10	L3	A8	M19
A4	T17	D2	P2	AHOLD	B18	LOCK#	P16		A11	L17	A12	N2
A5	R13	D3	J3	BE0#	L16	M/IO#	P17	**INC**	A16	L19	A13	N18
A6	T16	D4	N4	BE1#	K17	NMI	C16		C8	M17	A14	N19
A7	R14	D5	K3	BE2#	K16	PCD	K18	B11	C10	N3	A15	Q1
A8	S14	D6	M3	BE3#	G18	PCHK#	R18	B13	C12	N17	A17	Q18
A9	R12	D7	M4	BLAST#	S17	PLOCK#	R17	B4	D1	Q17	B8	Q19
A10	T14	D8	G3	BLEN#	A7	PWT	M16	B15	D5	R1	B10	R3
A11	S13	D9	E2	BOFF#	E18	RDY#	G17	C17	D6	R19	B12	S1
A12	T8	D10	F4	BRDY#	J16	RESET	D17	C13	D19	S4	C4	S5
A13	R11	D11	D2	BREQ	R16	SMI#	C11	C14	F3	S7	C5	S19
A14	T6	D12	H4	BS8#	E17	SMIACT#	D13	D11	F17	S9	C6	T7
A15	S8	D13	E3	BS16#	D18	STPCLK#	H16	D12	H3	S10	C19	T9
A16	R10	D14	L4	CACHE#	G1	UP#	C15	D15	H17	S11	E1	T10
A17	R4	D15	G4	CLK	D4	W/R#	P18	S18	J1	S12	E19	T11
A18	S6	D16	K4	D/C#	N16	WB/WT#	T1	T5	J17	S15	F2	T12
A19	R5	D17	E4	DP0	P4				J19	U4	F18	T13
A20	R9	D18	D3	DP1	G2				K1	U9	G19	T15
A21	R6	D19	C2	DP2	J4				K2*	U10	H1	U3
A22	R8	D20	B2	DP3	B6			**RES**	K19	U11	H2	U5
A23	T4	D21	C3	EADS#	C18					U16	H18	U6
A24	R7	D22	B3	EWBE#	P1			A6			H19	U7
A25	S3	D23	B5	FERR#	B14			A18			J2	U8
A26	T3	D24	B7	FLUSH#	D16			B19			J18	U12
A27	T2	D25	C7	HIT#	U2			F1			L2	U13
A28	S2	D26	D8	HITM#	U1			P19			L18	U14
A29	Q3	D27	D7	HLDA	Q16			T19			M1	U15
A30	Q4	D28	D9	HOLD	F16			U18			M2	U17
A31	R2	D29	B9	IGNNE#	B16			U19				
		D30	D10	INIT	F19							
		D31	C9	INTR	B17							

*If designing for single socket compatibility with future Pentium OverDrive processors, pin K2 may be connected to V_{CC} via a circuit to limit the current through the pin. Please contact Intel for more information about compatibility with future Pentium OverDrive processors.

NOTE:
The Pentium OverDrive processor socket provides orientation guides for IntelDX2 OverDrive processors one "KEY" pin in location E5. The Pentium OverDrive processor does not employ this inside "KEY" pin, which is left for backwards compatibility, but relies on the keying mechanism in the A1 corner to ensure proper orientation.

2.0 IntelDX2™ OverDrive® PROCESSOR FOR Intel486™ SX AND DX MICROPROCESSOR-BASED SYSTEMS

- **Processor Upgrade for most Intel486™ SX and DX Processor-Based Systems**
  - **— Single-Chip Upgrade**
  - **— Increases Both Integer and Floating Point Performance**
- **Two Package Variations to Support Systems with and without an OverDrive® Processor Socket**
- **169-Lead Pin Grid Array Package**
  - **— Pin Compatible with Intel487™ SX Math CoProcessor**
  - **— 169th Alignment Pin Ensures Proper Chip Orientation**
- **168-Lead Pin Grid Array Package**
  - **— Pin Compatible with Intel486™ DX Processor**
- **Utilizes IntelDX2 Speed-Doubling Technology**
  - **— Processor Core Runs at Twice the Frequency of the System Bus**
  - **— Compatible with 33, 25, 20 and 16 MHz Systems**
- **Floating Point Math Unit Included On-Chip**
- **High Integration Enables On-Chip**
  - **— 8 KByte Code and Data Cache**
  - **— Paged, Virtual Memory Management**
- **Binary Compatible with Large Installed Software Base**
  - **— MS-DOS, OS/2™, Windows**
  - **— UNIX System V/386**
  - **— IRMX, IRMK™ Kernals**
- **High Performance Design**
  - **— Core Clock Speed up to 66 MHz**
  - **— 106 Mbyte/sec Burst Bus**
  - **— CHMOS V Process Technology**
- **Complete 32-Bit Architecture**
  - **— Address and Data Busses**
  - **— Registers**
  - **— 8-, 16-, 32-Bit Data Types**
- **Compatible with Intel SL Enhanced Features**

The IntelDX2 OverDrive processor is the entry-level processor upgrade option offering excellent price/performance for cost-conscious users of most Intel486 SX and DX processor-based systems. Based on Intel's IntelDX2 technology, the IntelDX2 OverDrive processor integrates an integer unit, a floating point unit, a memory management unit, SL Enhanced features and an 8 KByte cache on a single chip.

Using the IntelDX2 processor's speed doubling technology, the IntelDX2 OverDrive processor operates internally at twice the speed of the system bus. This allows users of Intel486 SX and DX microprocessor-based systems to double the frequency of their computer's processor by adding a single chip, without upgrading any other system components. For example, adding an IntelDX2 OverDrive processor to an Intel486 DX 33 MHz system will double the processor's internal operating speed to 66 MHz.

The IntelDX2 OverDrive processor is based on the IntelDX2 microprocessor technology. This technology doubles the clock speed of the internal processor core, while interfacing with the system at the same external clock speed. When installed in a 33 MHz Intel486 SX or DX microprocessor-based system, the internal processor core, integer unit, floating point unit and cache operate at 66 MHz, while the speed of the external bus remains at 33 MHz. This provides increased processor performance while maintaining compatibility with the existing system design.

The IntelDX2 OverDrive processor is currently available in four product versions, which consist of two speed options (50 MHz and 66 MHz) and two package options (168-lead Pin Grid Array (PGA) and 169-lead PGA).

The 50 MHz IntelDX2 OverDrive processor is designed to upgrade 25 MHz Intel486 DX microprocessor-based systems and 16 MHz, 20 MHz and 25 MHz Intel486 SX microprocessor-based systems. The 66 MHz IntelDX2 OverDrive processor is designed to upgrade 33 MHz Intel486 SX and DX microprocessor-based systems. Table 2-1 illustrates the speed and pinout configurations for each system type.

These products come with a (0.25″ high) heat sink attached to the standard 169-lead PGA or 168-lead PGA package to aid in heat dissipation. All IntelDX2 OverDrive processors are binary compatible with a large base of software based on DOS, OS/2, Windows and Unix operating systems.

For more detailed information about the operation of the IntelDX2 OverDrive processor, refer to the IntelDX2 microprocessor data book (Order #241731-001).

Table 2-1. IntelDX2 OverDrive Processor to System Reference Table

OverDrive® Processor Part Number	OverDrive Processor Pinout	Systems Upgraded
DX2ODP50	169	Intel486™ SX -16 MHz -20 MHz -25 MHz
DX2ODP66	169	Intel486 SX -33 MHz
DX2ODPR50	168	Intel486 DX -25 MHz
DX2ODPR66	168	Intel 486 DX -33 MHz

2.1 Socket Configurations

Both single-socket and two-socket system configurations can be upgraded with the IntelDX2 OverDrive processor. In a single-socket Intel486 microprocessor-based system, this is done by replacing the original processor with the OverDrive processor. In a two-socket system, the IntelDX2 OverDrive processor can simply be placed into the empty OverDrive processor socket.

2.2 169-Lead PGA Device (DX2ODP)

The 169-lead version of the IntelDX2 OverDrive processor is currently available in two speeds; 50 MHz (DX2ODP50) and 66 MHz (DX2ODP66). The 169-lead versions are designed to be used in most Intel486 SX processor-based system and contain a key pin to assure proper orientation of the device (refer to Table 2-1). The OverDrive processor is simply inserted into the OverDrive processor socket, while the original processor remains in its socket.

Figure 1-3 shows the bottom-view (pin-side) pinout diagram of the 169-lead Pin Grid Array (PGA) package. Table 1-1 cross references the device's pin numbers to the pin names.

2.3 168-Lead PGA Device (DX2ODPR)

The 168-lead version of the IntelDX2 OverDrive processor is currently available in two speeds; 50 MHz (DX2ODPR50) and 66 MHz (DX2ODPR66). The 168-lead versions are designed to be used in most Intel486 DX processor-based system (refer to Table 2-1). The existing processor is removed and the upgrade processor is simply inserted into the same socket.

Figure 1-4 shows the bottom-view (pin-side) pinout diagram of the 168-lead Pin Grid Array (PGA) package. Table 1-2 cross references the device's pin numbers to the pin names.

3.0 IntelDX4™ OverDrive® PROCESSOR FOR Intel486™ SX AND DX MICROPROCESSOR-BASED SYSTEMS

- **Processor Upgrade for most Intel486™ SX and DX Processor-Based Systems**
  - **— Single-Chip Upgrade**
  - **— Increases Both Integer and Floating Point Performance**
- **Two Package Variations to Support Systems with and without an OverDrive® Processor Socket**
- **169-Lead Pin Grid Array Package**
  - **— Pin Compatible with Intel487™ SX Math CoProcessor**
  - **— 169th Alignment Pin Ensures Proper Chip Orientation**
- **168-Lead Pin Grid Array Package**
  - **— Pin Compatible with Intel486™ DX Processor**
- **High Integration Enables On-Chip**
  - **— 16 KByte Code and Data Cache**
  - **— Paged, Virtual Memory Management**
- **Floating Point Math Unit Included On-Chip**
- **Utilizes IntelDX4 Speed-Tripling Technology**
  - **— Processor Core Runs at Three Times the Frequency of the System Bus**
  - **— Compatible with 33, 25, 20 and 16 MHz Systems**
- **Binary Compatible with Large Installed Software and Operating System Base**
  - **— MS-DOS, OS/2™, Windows**
  - **— UNIX System V/386**
  - **— IRMX, IRMK™ Kernals**
- **High Performance Design**
  - **— Core Clock Speed up to 100 MHz**
  - **— CHMOS V Process Technology**
- **Complete 32-Bit Architecture**
  - **— Address and Data Busses**
  - **— Registers**
  - **— 8-, 16-, 32-Bit Data Types**
- **SL Enhanced Intel486™ Microprocessor Features Included On-Chip**

2

The IntelDX4 OverDrive processor is an upgrade for most Intel486 SX and DX microprocessor-based systems. It operates at a maximum internal core frequency of 100 MHz and is available in two package versions. When installed in a system, the IntelDX4 OverDrive processor significantly increases both the integer and floating point performance.

The IntelDX4 OverDrive processor offers several new features not found in the IntelDX2 OverDrive processors. It has 16 KByte on-chip cache and the internal core operates at 3x (speed tripled) the external clock frequency. The underlying technology behind the IntelDX4 OverDrive processor is the IntelDX4 microprocessor core with on-package voltage regulation. This allows the OverDrive processor to plug directly into existing 5V systems. Like the IntelDX2 OverDrive processor the IntelDX4 OverDrive processor supports System Management Mode (SMM) and Stop Clock Mode. The SMM and Stop Clock Mode, identical to those implemented in SL Enhanced Intel486 SX and DX microprocessors, make the IntelDX4 OverDrive processor compatible with the advanced power management, system security and device emulation features of SL Enhanced systems.

The IntelDX4 OverDrive processor is based on the IntelDX4 microprocessor technology. This technology triples the clock speed of the internal processor core, while interfacing with the system at the same external clock speed. When installed in a 33 MHz Intel486 SX or DX microprocessor-based system, the internal processor core, integer unit, floating point unit and cache operate at 100 MHz, while the speed of the external bus remains at 33 MHz. This provides increased processor performance while maintaining compatibility with the existing system design. In addition, the internal cache has been doubled to 16 KBytes.

The IntelDX4 OverDrive processor is currently available in four product versions, which consist of two speed options (75 MHz and 100 MHz) and two package options (168-lead Pin Grid Array (PGA) and 169-lead PGA).

The 100 MHz OverDrive processors are designed to upgrade most 33 MHz Intel486 SX and DX microprocessor-based systems. The 75 MHz OverDrive processors are designed to upgrade most 25 MHz Intel486 DX microprocessor-based systems and 16 MHz, 20 MHz and 25 MHz Intel486 SX microprocessor-based systems. Table 3-1 illustrates this. The speed tripling technology will triple the internal speed of the processor to three times the bus speed of the existing system.

These products come with a (0.6" high) heat sink attached to the standard 169-lead PGA or 168-lead PGA package to aid in heat dissipation. Refer to Sections 14.0 and 15.0 for clearance and thermal requirements. All IntelDX4 OverDrive processors are binary compatible with a large base of software based on DOS, OS/2, Windows and Unix operating systems.

For more detailed information about the operation of the IntelDX4 OverDrive processor, refer to the IntelDX4 microprocessor data book (Order #241944-001).

Table 3-1. IntelDX4 OverDrive Processor to System Reference Table

OverDrive® Processor Part Number	OverDrive Processor Pinout	Systems Upgraded
DX4ODP75	169	Intel486™ SX -16 MHz -20 MHz -25 MHz
DX4ODP100	169	Intel486 SX -33 MHz
DX4ODPR75	168	Intel486 DX -25 MHz
DX4ODPR100	168	Intel 486 DX -33 MHz

3.1 Socket Configurations

Both single-socket and two-socket system configurations can be upgraded with the IntelDX4 OverDrive processor. In a single-socket Intel486 microprocessor-based system, this is done by replacing the original processor with the OverDrive processor. In a two-socket system, the IntelDX4 OverDrive processor can simply be placed into the empty OverDrive processor socket.

3.2 169-Lead PGA Device (DX4ODP)

The 169-lead version of the IntelDX4 OverDrive processor is currently available in two speeds; 75 MHz (DX4ODP75) and 100 MHz (DX4ODP100). The 169-lead versions are designed to be used in most Intel486 SX processor-based system and contain a key pin to assure proper orientation of the device (refer to Table 3-1). The processor is simply inserted into the OverDrive processor socket, while the original processor remains in its socket.

Figure 1-3 shows the bottom-view (pin-side) pinout diagram of the 169-lead Pin Grid Array (PGA) package. Table 1-1 cross references the device's pin numbers to the pin names.

3.3 168-Lead PGA Device (DX4ODPR)

The 168-lead version of the IntelDX4 OverDrive processor is currently available in two speeds; 75 MHz (DX4ODPR75) and 100 MHz (DX4ODPR100). The 168-lead versions are designed to be used in most Intel486 DX processor-based system (refer to Table 3-1). The existing processor is removed and the upgrade processor is simply inserted into the same socket.

Figure 1-4 shows the bottom-view (pin-side) pinout diagram of the 168-lead Pin Grid Array (PGA) package. Table 1-2 cross references the device's pin numbers to the pin names.

4.0 INTEL Pentium® OverDrive® PROCESSOR

- **Powerful CPU Upgrade for most Intel486™ CPU-Based Systems**
  - **Makes Intel Procesor-Based Systems Run Faster**
  - **Significantly Accelerates All Software Applications**
- **Designed for Systems Based on:**
  - **Intel486 SX Processors**
  - **Intel486 DX Processors**
  - **IntelSX2™ Processors**
  - **IntelDX2™ Processors**
- **Compatible with Installed Base of Thousands of Applications**
- **Based on Intel Pentium® Processor Technology**
  - **Superscalar Architecture**
  - **Branch Prediction**
  - **Faster Floating Point Unit**
- **Enhancements to Core Pentium Processor Silicon**
  - **Separate Code and Data Caches**
  - **16 KB Code Cache**
  - **16 KB Write-Back Data Cache**
  - **32-Bit Bus Interface**
- **Package Innovations**
  - **On-Package Voltage Regulation**
  - **Integrated Fan Heat Sink**
- **Incorporates SMM Power Saving Features**

The Pentium OverDrive processor is Intel's highest performance CPU upgrade for systems based on the Intel486 family of CPUs. It is the recommended upgrade option for most IntelSX2 and IntelDX2 CPU-based systems and the superior upgrade option for most Intel486 SX and DX CPU-based systems. The Pentium processor's superscalar architecture (which allows more than one instruction per clock cycle to be executed), the 32 KB enhanced on-chip cache memory and faster floating point unit provide a significant performance boost across a wide range of applications. The specially-designed bus interface unit enables the Pentium OverDrive processor to operate internally at 64 bits while working seamlessly with the 32-bit Intel486 architecture.

The Pentium OverDrive processor may contain certain design defects or errors known as errata. Current characterized errata are available on request.

4.1 Product Description

The Pentium OverDrive processor is designed to upgrade most systems based on Intel486 SX, Intel486 DX, IntelSX2, and IntelDX2 processor-based systems and is based on Intel's Pentium processor technology. It is 100% binary compatible with the 8086/88, 80286, Intel386 DX, Intel386 SX, Intel486 DX, Intel486 SX, IntelSX2, and the IntelDX2 processor family.

The Pentium OverDrive processor provides significant improvements over the Intel486 CPU including:

- Superscalar Architecture
- Dynamic Branch Prediction
- Pipelined Floating-Point Unit
- Separate 16K Code and 16K Data Caches
- Improved Instruction Execution Times
- Write back MESI Protocol implemented in Data Cache
- System Management Mode

The Pentium OverDrive processor significantly increases the integer performance, and can attain up to 2x floating-point performance relative to an equivalent frequency IntelDX2 processor. The bus frequencies for the Pentium OverDrive processor are 25 MHz and 33 MHz.

The Pentium OverDrive processor has two pipelines and a floating-point unit that are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, 2 floating point instructions) in one clock.

The floating-point unit has been completely redesigned over the IntelDX2 processor. Faster algorithms provide at least 3X internal speed-up for common floating point operations including ADD, MUL, and LOAD. With instruction scheduling and overlapped (pipelined) execution, these three performance enhancements can allow many math intensive applications to achieve a 2X performance boost.

The Pentium OverDrive processor implements 32-bit address and data busses.

The Pentium OverDrive processor has separate code and data caches, both are 16 KBytes each. The data cache has write-back capabilities.

The Pentium OverDrive also has an integrated fan heat sink. The heat sink contains logic circuitry which senses if the speed of the fan is insufficient to cool the processor and will reduce the internal frequency of the processor to that of the internal bus. This will allow the processor to run indefinitely without damage.

NOTE:

1. Refer to the *Pentium Processor Data Book* for more information on instruction execution timing and pairing.

5.0 PIN DESCRIPTIONS

5.1 Pins Common to All OverDrive® Processors

Tables 5-1 through 5-4 list pin descriptions of the signals present on the Intel DX2, IntelDX4 OverDrive processors, and are not unique on the Pentium OverDrive processor.

Table 5-1. Pin Descriptions

Symbol	Type	Name and Function
CLK	I	*Clock* provides the fundamental timing for the bus interface unit and is multiplied by two (2x) for the IntelDX2 OverDrive Processors or three (3x) for the IntelDX4 OverDrive Processor to provide the internal frequency for the Intel OverDrive processor. All external timing parameters are specified with respect to the rising edge of CLK.
ADDRESS BUS		
A31–A4 A2–A3	I/O O	A31–A2 are the *address lines* of the processor. A31–A2, together with the byte enables BE0#–BE3#, define the physical area of memory or input/output space accessed. Address lines A31–A4 are used to drive addresses into the processor to perform cache line invalidations. Input signals must meet setup and hold times t_{22} and t_{23}. A31–A2 are not driven during bus or address hold.
BE0–3#	O	The *byte enable* signals indicate active bytes during read and write cycles. During the first cycle of a cache fill, the external system should assume that all byte enables are active. BE3# applies to D24–D31, BE2# applies to D16–D23, BE1# applies to D8–D15 and BE0# applies to D0–D7. BE0#–BE3# are active LOW and are not driven during bus hold.
DATA BUS		
D31–D0	I/O	These are the *data lines* for the Intel OverDrive processor. Lines D0–D7 define the least significant byte of the data bus while lines D24–D31 define the most significant byte of the data bus. These signals must meet setup and hold times t_{22} and t_{23} for proper operation on reads. These pins are driven during the second and subsequent clocks of write cycles.
DATA PARITY		
DP0–DP3	I/O	There is one *data parity* pin for each byte of the data bus. Data parity is generated on all write data cycles with the same timing as the data driven by the Intel OverDrive processor. Even parity information must be driven back into the microprocessor on the data parity pins with the same timing as read information to insure that the correct parity check status is indicated by the Intel OverDrive processor. The signals read on these pins do not affect program execution. Input signals must meet setup and hold times t_{22} and t_{23}. DP0–DP3 should be connected to V_{CC} through a pullup resistor in systems which do not use parity. DP0–DP3 are active HIGH and are driven during the second and subsequent clocks of write cycles.

Table 5-1. Pin Descriptions (Continued)

Symbol	Type	Name and Function
DATA PARITY (Continued)		
PCHK#	O	*Parity Status* is driven on the PCHK# pin the clock after ready for read operations. The parity status is for data sampled at the end of the previous clock. A parity error is indicated by PCHK# being LOW. Parity status is only checked for enabled bytes as indicated by the byte enable and bus size signals. PCHK# is valid only in the clock immediately after read data is returned to the microprocessor. At all other times PCHK# is inactive (HIGH). PCHK# is never floated.
BUS CYCLE DEFINITION		
M/IO# D/C# W/R#	O O O	The *memory/input-output, data/control* and *write/read* lines are the primary bus definition signals. These signals are driven valid as the ADS# signal is asserted.

M/IO#	D/C#	W/R#	Bus Cycle Initiated
0	0	0	Interrupt Acknowledge
0	0	1	Halt/Special Cycle
0	1	0	I/O Read
0	1	1	I/O Write
1	0	0	Code Read
1	0	1	Reserved
1	1	0	Memory Read
1	1	1	Memory Write

The bus definition signals are not driven during bus hold and follow the timing of the address bus. Refer to Section 7.2.11 for a description of the special bus cycles.

LOCK#	O	The *bus lock* pin indicates that the current bus cycle is locked. The Intel OverDrive processor will not allow a bus hold when LOCK# is asserted (but address holds are allowed). LOCK# goes active in the first clock of the first locked bus cycle and goes inactive after the last clock of the last locked bus cycle. The last locked cycle ends when RDY# is returned. LOCK# is active LOW and is not driven during bus hold. Locked read cycles will not be transformed into cache fill cycles if KEN# is returned active.
PLOCK#	O	The *pseudo-lock* pin indicates that the current bus transaction requires more than one bus cycle to complete. Examples of such operations are floating point long reads and writes (64 bits), segment table descriptor reads (64 bits), in addition to cache line fills (128 bits). The Intel OverDrive processor will drive PLOCK# active until the addresses for the last bus cycle of the transaction have been driven regardless of whether RDY# or BRDY# have been returned. Normally PLOCK# and BLAST# are inverse of each other. However during the first bus cycle of a 64-bit floating point write, both PLOCK# and BLAST# will be asserted. PLOCK# is a function of the BS8#, BS16# and KEN# inputs. PLOCK# should be sampled only in the clock RDY# is returned. PLOCK# is active LOW and is not driven during bus hold.
BUS CONTROL		
ADS#	O	The *address status* output indicates that a valid bus cycle definition and address are available on the cycle definition lines and address bus. ADS# is driven active in the same clock as the addresses are driven. ADS# is active LOW and is not driven during bus hold.

Table 5-1. Pin Descriptions (Continued)

Symbol	Type	Name and Function
BUS CONTROL (Continued)		
RDY#	I	The *non-burst ready* input indicates that the current bus cycle is complete. RDY# indicates that the external system has presented valid data on the data pins in response to a read or that the external system has accepted data from the Intel OverDrive processor in response to a write. RDY# is ignored when the bus is idle and at the end of the first clock of the bus cycle. RDY# is active during address hold. Data can be returned to the processor while AHOLD is active. RDY# is active LOW, and is not provided with an internal pullup resistor. RDY# must satisfy setup and hold times t_{16} and t_{17} for proper chip operation.
BURST CONTROL		
BRDY#	I	The *burst ready input* performs the same function during a burst cycle that RDY# performs during a non-burst cycle. BRDY# indicates that the external system has presented valid data in response to a read or that the external system has accepted data in response to a write. BRDY# is ignored when the bus is idle and at the end of the first clock in a bus cycle. BRDY# is sampled in the second and subsequent clocks of a burst cycle. The data presented on the data bus will be strobed into the microprocessor when BRDY# is sampled active. If RDY# is returned simultaneously with BRDY#, BRDY# is ignored and the burst cycle is prematurely interrupted BRDY# is active LOW and is provided with a small pullup resistor. BRDY# must satisfy the setup and hold times t_{16} and t_{17}.
BLAST#	O	The *burst last* signal indicates that the next time BRDY# is returned the burst bus cycle is complete. BLAST# is active for both burst and non-burst bus cycles. BLAST# is active LOW and is not driven during bus hold.
INTERRUPTS		
RESET	I	The **RESET** input forces the processor to begin execution at a known state. Reset is asynchronous, but must meet setup and hold times t20 and t21 for recognition in any specific clock. The processor cannot begin execution of instructions until at least 1 ms after V_{CC} and CLK have reached their proper AC and DC specifications. However, for soft resets, RESET should remain active for at least 15 CLK periods. The RESET pin should remain active during this time to ensure proper processor operation. RESET is active HIGH. RESET sets the SMBASE descriptor to a default address of 30000H. If the system uses SMBASE relocation, then the SRESET pin should be used for soft resets.
SRESET	I	The SRESET pin duplicates all the functionality of the RESET pin with the following two exceptions: 1. The SMBASE register will retain its previous value. 2. If UP# (I) is asserted, SRESET will not have an effect on the host microprocessor. For soft resets, SRESET should remain active for at least 15 CLK periods. SRESET is active HIGH. SRESET is asynchronous but must meet setup and hold times t_{20} and t_{21} for recognition in any specific clock.

Table 5-1. Pin Descriptions (Continued)

Symbol	Type	Name and Function
INTERRUPTS (Continued)		
SMI#	I	The **System Management Interrupt** input is used to invoke the System Management Mode (SMM). SMI# is a falling edge triggered signal which forces the processor into SMM at the completion of the current instruction. SMI# is recognized on an instruction boundary and at each iteration for repeat string instructions. SMI# does not break LOCKed bus cycles and cannot interrupt a currently executing SMM. The processor will latch the falling edge of one pending SMI# signal while the processor is executing an existing SMI. The nested SMI will not be recognized until after the execution of a Resume (RSM) instruction.
SMIACT#	O	The **System Management Interrupt ACTive** is an active low output, indicating that the processor is operating in SMM. It is asserted when the processor begins to execute the SMI state save sequence and will remain active LOW until the processor executes the last state restore cycle out of SMRAM.
STPCLK#	I	The **SToP CLocK request** input signal indicates a request has been made to turn off the CLK input. When the processor recognizes a STPCLK#, the processor will stop execution on the next instruction boundary, unless superseded by a higher priority interrupt, empty all internal pipelines and the write buffers and generate a Stop Grant acknowledge bus cycle. STPCLK# is active LOW and is provided with an internal pull-up resistor. STPCLK# is asynchronous but setup and hold times t_{20} and t_{21} must be met to ensure recognition in any specific clock.
INTR	I	The *maskable interrupt* indicates that an external interrupt has been generated. If the internal interrupt flag is set in EFLAGS, active interrupt processing will be initiated. The Intel OverDrive processor will generate two locked interrupt acknowledge bus cycles in response to the INTR pin going active. INTR must remain active until the interrupt acknowledges have been performed to assure that the interrupt is recognized. INTR is active HIGH and is not provided with an internal pulldown resistor. INTR is asynchronous, but must meet setup and hold times t_{20} and t_{21} for recognition in any specific clock.
NMI	I	The *non-maskable interrupt* request signal indicates that an external non-maskable interrupt has been generated. NMI is rising edge sensitive. NMI must be held LOW for at least four CLK periods before this rising edge. NMI is not provided with an internal pulldown resistor. NMI is asynchronous, but must meet setup and hold times t_{20} and t_{21} for recognition in any specific clock.
BUS ARBITRATION		
BREQ	O	The *internal cycle pending* signal indicates that the Intel OverDrive processor has internally generated a bus request. BREQ is generated whether or not the Intel OverDrive processor is driving the bus. BREQ is active HIGH and is never floated.
HOLD	I	The *bus hold request* allows another bus master complete control of the Intel OverDrive processor bus. In response to HOLD going active the Intel OverDrive processor will float most of its output and input/output pins. HLDA will be asserted after completing the current bus cycle, burst cycle or sequence of locked cycles. The Intel OverDrive processor will remain in this state until HOLD is deasserted. HOLD is active high and is not provided with an internal pulldown resistor. HOLD must satisfy setup and hold times t_{18} and t_{19} for proper operation.

Table 5-1. Pin Descriptions (Continued)

Symbol	Type	Name and Function
BUS ARBITRATION (Continued)		
HLDA	O	*Hold acknowledge* goes active in response to a hold request presented on the HOLD pin. HLDA indicates that the Intel OverDrive processor has given the bus to another local bus master. HLDA is driven active in the same clock that the Intel OverDrive processor floats its bus. HLDA is driven inactive when leaving bus hold. HLDA is active HIGH and remains driven during bus hold.
BOFF#	I	The *backoff* input forces the Intel OverDrive processor to float its bus in the next clock. The microprocessor will float all pins normally floated during bus hold but HLDA will not be asserted in response to BOFF#. BOFF# has higher priority than RDY# or BRDY#; if both are returned in the same clock, BOFF# takes effect. The microprocessor remains in bus hold until BOFF# is negated. If a bus cycle was in progress when BOFF# was asserted the cycle will be restarted. BOFF# is active LOW and must meet setup and hold times t_{18} and t_{19} for proper operation.
CACHE INVALIDATION		
AHOLD	I	The *address hold* request allows another bus master access to the Intel OverDrive processor's address bus for a cache invalidation cycle. The Intel OverDrive processor will stop driving its address bus in the clock following AHOLD going active. Only the address bus will be floated during address hold, the remainder of the bus will remain active. AHOLD is active HIGH and is provided with a small internal pulldown resistor. For proper operation AHOLD must meet setup and hold times t_{18} and t_{19}.
EADS#	I	This signal indicates that a *valid external address* has been driven onto the Intel OverDrive processor address pins. This address will be used to perform an internal cache invalidation cycle. EADS# is active LOW and is provided with an internal pullup resistor. EADS# must satisfy setup and hold times t_{12} and t_{13} for proper operation.
CACHE CONTROL		
KEN#	I	The *cache enable* pin is used to determine whether the current cycle is cacheable. When the Intel OverDrive processor generates a cycle that can be cached and KEN# is active, the cycle will become a cache line fill cycle. Returning KEN# active one clock before RDY# during the last read in the cache line fill will cause the line to be placed in the on-chip cache. KEN# is active LOW and is provided with a small internal pullup resistor. KEN# must satisfy setup and hold times t_{14} and t_{15} for proper operation.
FLUSH#	I	The *cache flush* input forces the Intel OverDrive processor to flush its entire internal cache. FLUSH# is active low and need only be asserted for one clock. FLUSH# is asynchronous but setup and hold times t_{20} and t_{21} must be met for recognition in any specific clock. FLUSH# being sampled low in the clock before the falling edge of RESET causes the Intel OverDrive processor to enter the tri-state test mode.
PAGE CACHEABILITY		
PWT PCD	O O	The *page write-through* and *page cache disable* pins reflect the state of the page attribute bits, PWT and PCD, in the page table entry or page directory entry. If paging is disabled or for cycles that are not paged, PWT and PCD reflect the state of the PWT and PCD bits in control register 3. PWT and PCD have the same timing as the cycle definition pins (M/IO#, D/C# and W/R#). PWT and PCD are active HIGH and are not driven during bus hold. PCD is masked by the cache disable bit (CD) in Control Register 0.

Table 5-1. Pin Descriptions (Continued)

Symbol	Type	Name and Function
NUMERIC ERROR REPORTING		
FERR#	O	The *floating point error* pin is driven active when a floating point error occurs. FERR# is similar to the ERROR# pin on the Intel387™ math coprocessor. FERR# is included for compatibility with systems using DOS type floating point error reporting. FERR# will not go active if FP errors are masked in FPU register. FERR# is active LOW, and is not floated during bus hold.
IGNNE#	I	When the *ignore numeric error* pin is asserted the Intel OverDrive processor will ignore a numeric error and continue executing non-control floating point instructions, but FERR# will still be activated by the Intel OverDrive processor. When IGNNE# is deasserted the Intel OverDrive processor will freeze on a non-control floating point instruction, if a previous floating point instruction caused an error. IGNNE# has no effect when the NE bit in control register 0 is set. IGNNE# is active LOW and is provided with a small internal pullup resistor. IGNNE# is asynchronous but setup and hold times t_{20} and t_{21} must be met to insure recognition on any specific clock.
BUS SIZE CONTROL		
BS16# BS8#	I I	The *bus size 16* and *bus size 8* pins (bus sizing pins) cause the Intel OverDrive processor to run multiple bus cycles to complete a request from devices that cannot provide or accept 32 bits of data in a single cycle. The bus sizing pins are sampled every clock. The state of these pins in the clock before ready is used by the Intel OverDrive processor to determine the bus size. These signals are active LOW and are provided with internal pullup resistors. These inputs must satisfy setup and hold times t_{14} and t_{15} for proper operation.
ADDRESS MASK		
A20M#	I	When the *address bit 20 mask* pin is asserted, the Intel OverDrive processor masks physical address bit 20 (A20) before performing a lookup to the internal cache or driving a memory cycle on the bus. A20M# emulates the address wraparound at one Mbyte which occurs on the 8086. A20M# is active LOW and should be asserted only when the processor is in real mode. This pin is asynchronous but should meet setup and hold times t_{20} and t_{21} for recognition in any specific clock. For proper operation, A20M# should be sampled high at the falling edge of RESET.
i486 DX AND i486 SX PROCESSOR INTERFACE		
UP#(1,2)	O	The *upgrade present* pin is used to signal the Intel486 Microprocessor to float its outputs and get off the bus. It is active low and is never floated. UP# is driven low at power-up and remains active for the entire duration of the Upgrade Processor operation.
KEY PIN		
KEY(2)		The KEY pin is an electrically non-functional pin which is used to ensure correct Upgrade Processor orientation in a 169-pin socket.

NOTE:
1. The UP# pin was previously named the MP# pin in the i486 SX Microprocessor/i487 SX Math CoProcessor data book. The functionality is the same, only the name has changed.
2. The UP# input pin and KEY pin are not defined on the OverDrive processor for replacement of PGA Intel486 DX Microprocessor (ODPR).

Table 5-2. Output Pins

Name	Active Level	When Floated
BREQ	HIGH	
HLDA	HIGH	
BE0#–BE3#	LOW	Bus Hold
PWT, PCD	HIGH	Bus Hold
W/R#, D/C#, M/IO#	HIGH	Bus Hold
LOCK#	LOW	Bus Hold
PLOCK#	LOW	Bus Hold
ADS#	LOW	Bus Hold
BLAST#	LOW	Bus Hold
PCHK#	LOW	
FERR#	LOW	
SMIACT#	LOW	
UP#	LOW	
A2–A3	HIGH	Bus, Address Hold

Table 5-3. Input Pins

Name	Active Level	Synchronous/ Asynchronous
CLK		
RESET	HIGH	Asynchronous
HOLD	HIGH	Synchronous
AHOLD	HIGH	Synchronous
EADS#	LOW	Synchronous
BOFF#	LOW	Synchronous
FLUSH#	LOW	Asynchronous
A20M#	LOW	Asynchronous
BS16#, BS8#	LOW	Synchronous
KEN#	LOW	Synchronous
RDY#	LOW	Synchronous
BRDY#	LOW	Synchronous
INTR	HIGH	Asynchronous
NMI	HIGH	Asynchronous
SRESET	HIGH	Asynchronous
SMI#	LOW	Asynchronous
STPCLK#	LOW	Asynchronous
IGNNE#	LOW	Asynchronous

Table 5-4. Input/Output Pins

Name	Active Level	When Floated
D0–D31	HIGH	Bus Hold
DP0–DP3	HIGH	Bus Hold
A4–A31	HIGH	Bus, Address Hold

5.2 Pentium® OverDrive® Processor Pin Descriptions

This section provides a summary of the Pentium OverDrive processor pins and how they function. For more information on the pinout differences between the Write-Back Enhanced IntelDX2 processor and the Pentium OverDrive processor, please see Appendix B.

5.2.1 SIGNAL DESCRIPTIONS

Table 5-5 provides a brief pin description.

Table 5-5. Signal Description

Symbol	Type	Name and Function
INTERRUPTS		
INIT	I	The **INIT** pin is the Pentium OverDrive processor initialization pin. Since the SL-enhanced Intel486 processors implement this functionality on the **SRESET** pin, the corresponding pin (D11) on the Pentium OverDrive Processor is defined as an **INC** pin so that pin F19 and pin D11 may be tied together on the motherboard. **INIT** will force the Pentium OverDrive Processor to begin execution in a known state. The processor state after **INIT** is the same as the state after **RESET** except that the internal caches, floating point registers, and the SMM base register retain whatever values they had prior to **INIT. INIT** may NOT be used in lieu of **RESET** after power-up. **INIT** can not be used to cause the processor to enter BIST or Tri-State Test Mode. INIT is an edge triggered interrupt and is processed at instruction boundaries. Since **INIT** is an interrupt, **ADS#** may be driven even if **INIT** is active. Recognition of **INIT** is guaranteed in a specific clock if it is asserted synchronously and meets setup and hold times. To guarantee recognition if **INIT** is asserted asynchronously, it must have been deasserted for a minimum of two clocks before being returned active to the processor and remain asserted for a minimum pulse width of two clocks. **INIT** must remain active for three clocks prior to the **BRDY#** or **RDY#** of I/O write cycle to guarantee the processor recognizes and performs **INIT** right after I/O write instruction. **INIT** is asynchronous but must meet setup and hold times t_{20} and t_{21} to be recognized on any one clock. **INIT** is supplied with an internal pull-down.
CACHE CONTROL		
WB/WT#	I	This pin allows a cache line to be defined as *write back or write through* on a line by line basis. As a result, it controls the MESI state that the line is saved in. This pin is sampled in the clock in which the first **BRDY#** or **RDY#** is returned for a read cycle or a write through cycle. However, it must the meet setup and hold times at every clock. Cache lines are not allocated on write through cycles. This pin is also used as a configuration pin at the falling edge of **RESET** only. If **WB/WT#** is driven low, or left unconnected, the processor will operate in a **standard bus (write through only)** mode. This means that the Pentium OverDrive processor will run in an IntelDX2 processor compatible write through cache mode. If **WB/WT#** is driven high, the processor will operate in enhanced bus **(write back)** mode. **INIT** can not be used to change the mode of the processor with **WB/WT#.** During **RESET, WB/WT#** should be held at its desired value for two clocks before and after the falling edge of **RESET. WB/WT#** is ignored when the processor is in **standard bus** mode. Please see Appendix A for a detailed description of the two modes of operation. **WB/WT#** has an internal pull down resistor to force the processor into the **standard bus** mode if left unconnected. The pin timings must meet setup and hold times t_{38} and t_{39} on every clock edge.

Table 5-5. Signal Description (Continued)

Symbol	Type	Name and Function
CACHE INQUIRE		
EWBE#	I	The *External Write Buffer Empty* pin, when inactive (HIGH), indicates that a write through cycle is pending in the external system. When the processor generates a non-write back cycle, and **EWBE#** is sampled inactive, the processor will hold off all subsequent writes to all E- or M- state lines until all write-through cycles have completed, as indicated by **EWBE#** going active. This ensures that writes are visible from outside the processor in the same order as they were generated by software. When the Pentium OverDrive processor serializes instruction execution through the use of a serializing instruction, it waits for **EWBE#** to go active before fetching and executing the next instruction. **EWBE#** is sampled with each **BRDY#** or **RDY#** of a write through cycle. If sampled inactive, the processor will repeatedly sample **EWBE#** in each clock until it is found active. Once sampled active, it will ignore **EWBE#** until the next **BRDY#** or **RDY#** of a write cycle. **EWBE#** is not sampled and has no effect on processor operation while the processor is in standard bus mode. If unused, **EWBE#** should be tied LOW or left unconnected. **EWBE#** is active low and must meet setup and hold times t_{38} and t_{39} and is supplied with an internal pull-down.
HIT#	O	This pin participates in an inquire cycle. If an inquire cycle hits a valid line in the processor, this pin is asserted two clocks after **EADS#** has been driven to the processor. If the inquire cycle misses in the processor cache, this pin is negated two clocks after **EADS#.** This pin changes its value only as a result of inquire cycle as described above and retains its value between any two inquire cycles.
HITM#	O	**HITM#** is asserted when an inquire cycle hits a modified line in the Pentium OverDrive processor. It can be used to inhibit another bus master from accessing the data until the line is completely written back. **HITM#** is asserted two clocks after an **EADS#** assertion hits a modified line in the processor cache and deasserts after the last **BRDY#** or **RDY#** of the corresponding write back is returned. **HITM#** is guaranteed to be deasserted before the next **ADS#** following a write back cycle. If an INVD instruction occurs at the same time as an external snoop, **HITM#** may be asserted and deasserted without a corresponding **ADS#** for a write back cycle.
CACHE BURST CONTROL		
BLEN#	I	**BLEN#** controls if write back cycles will be attempted to run as burst cycles. When **BLEN#** is HIGH, the processor will write out a dirty line as four separate writes, each with its own **ADS#** and **BLAST#.** When **BLEN#** is LOW, a write back is done as a 16 byte burst. **BLEN#** is a constant input, meaning that it will have to be tied HIGH or LOW. As a result, it does not have setup and hold time specifications. **BLEN#** is has no effect and is not sampled when the processor is in **standard bus** mode. **BLEN#** is supplied with an internal pull-up resistor.
CACHE#	O	The **CACHE#** pin is used to indicate a cache operation. **CACHE#** will be active along with the first **ADS#** until the first **RDY#/BRDY#** and is undefined during any other time period. On cacheable read accesses, **CACHE#** will be asserted when **PCD** is low, except on locked cycles. **CACHE#** will always be asserted in the beginning of cacheable reads (line fills and code prefetches) and can be used as an indication that the processor intends to perform a linefill. For write cycles, **CACHE#** is active for write backs only. The beginning of a replacement write back can be uniquely identified by the presence of **ADS#, W/R#** and **CACHE#** together. The beginning of a snoop write back is marked by **ADS#, W/R#, CACHE#** and **HITM#** being active together.

Table 5-5. Signal Description (Continued)

Symbol	Type	Name and Function
INC PINS		
INC		The **INC** pin is defined to be an *internal no-connect*. This means that the pin is not connected internally, and may be used for the routing of external signals. It will never be used for any other function, and is guaranteed to remain an **INC** pin. Any voltage level applied to an **INC** pin must remain within the processor V_{CC} specifications. Most **INC** pins have a specified use in creating a design that supports multiple processors in one socket. For more information, please see Section 9.

5.2.2 OUTPUT PINS

Table 5-6 lists all the output pins, indicating their active level, and when they are floated.

Table 5-6. Output Pins

Name	Active Level	When Floated
BREQ	HIGH	
HLDA	HIGH	
BE3#–BE0#	LOW	Bus Hold, Backoff
PWT, PCD	HIGH	Bus Hold, Backoff
W/R#, D/C#, M/IO#	LOW	Bus Hold, Backoff
LOCK#	LOW	Bus Hold, Backoff
PLOCK#	LOW	Bus Hold, Backoff
ADS#	LOW	Bus Hold, Backoff
BLAST#	LOW	Bus Hold, Backoff
PCHK#	LOW	
FERR#	LOW	
SMIACT#	LOW	
A3–A2	HIGH	Bus Hold, Address Hold and Backoff
HIT#, HITM#	LOW	
CACHE#	LOW	Bus Hold, Backoff
UP#	LOW	

5.2.3 INPUT PINS

Table 5-7 lists all input pins, indicating their active level, and whether they are synchronous or asynchronous inputs.

Table 5-7. Input Pins

Name	Active Level	Synchronous/ Asynchronous	Internal Resistor	Qualified
CLK				
RESET	HIGH	Asynchronous		
HOLD	HIGH	Synchronous		
AHOLD	HIGH	Synchronous	PULLDOWN	
EADS#	LOW	Synchronous	PULLUP	
BOFF#	LOW	Synchronous	PULLUP	
FLUSH#	LOW	Asynchronous	PULLUP	
A20M#	LOW	Asynchronous	PULLUP	
BS16#, BS8#	LOW	Synchronous	PULLUP	
KEN#	LOW	Synchronous	PULLUP	
RDY#	LOW	Synchronous		
BRDY#	LOW	Synchronous	PULLUP	
INTR	HIGH	Asynchronous		
NMI	HIGH	Asynchronous		
IGNNE#	LOW	Asynchronous	PULLUP	
BLEN#	LOW	Tie High or Low	PULLUP	
EWBE#	LOW	Synchronous	PULLDOWN	BRDY#/RDY#
INIT	HIGH	Asynchronous	PULLDOWN	
INV	HIGH	Synchronous	PULLUP	EADS#
STPCLK#	LOW	Asynchronous	PULLUP	
SMI#	LOW	Asynchronous	PULLUP	
WB/WT#	BOTH	Synchronous	PULLDOWN	FIRST RDY#/RDY#

5.2.4 INPUT/OUTPUT PINS

Table 5-8 lists all the input/output pins, indicating their active level, and when they are floated.

Table 5-8. Input/Output Pins

Name	Active Level	When Floated
D31–D0	HIGH	Bus Hold, Backoff
DP3–DP0	HIGH	Bus Hold, Backoff
A31–A4	HIGH	Bus, Address Hold, Backoff

5.3 Architecture Block Diagram

5.3.1 IntelDX2™ AND IntelDX4™ OverDrive® PROCESSORS

Figure 5-1 shows a block diagram of the IntelDX2 and IntelDX4 OverDrive Processor Architecture. There are a few minor architectural differences between each of the OverDrive processors. These differences are summarized below, with respect to Figure 5-1.

The IntelDX4 OverDrive processor contains a clock tripling circuit, as opposed to the clock doubling circuit used in the IntelDX2 OverDrive processors. This is located in the upper left of the diagram.

The IntelDX4 OverDrive processor contains a 16 KByte cache, as opposed to the 8 KByte cache used in the IntelDX2 OverDrive processors. This is located in the center of the diagram.

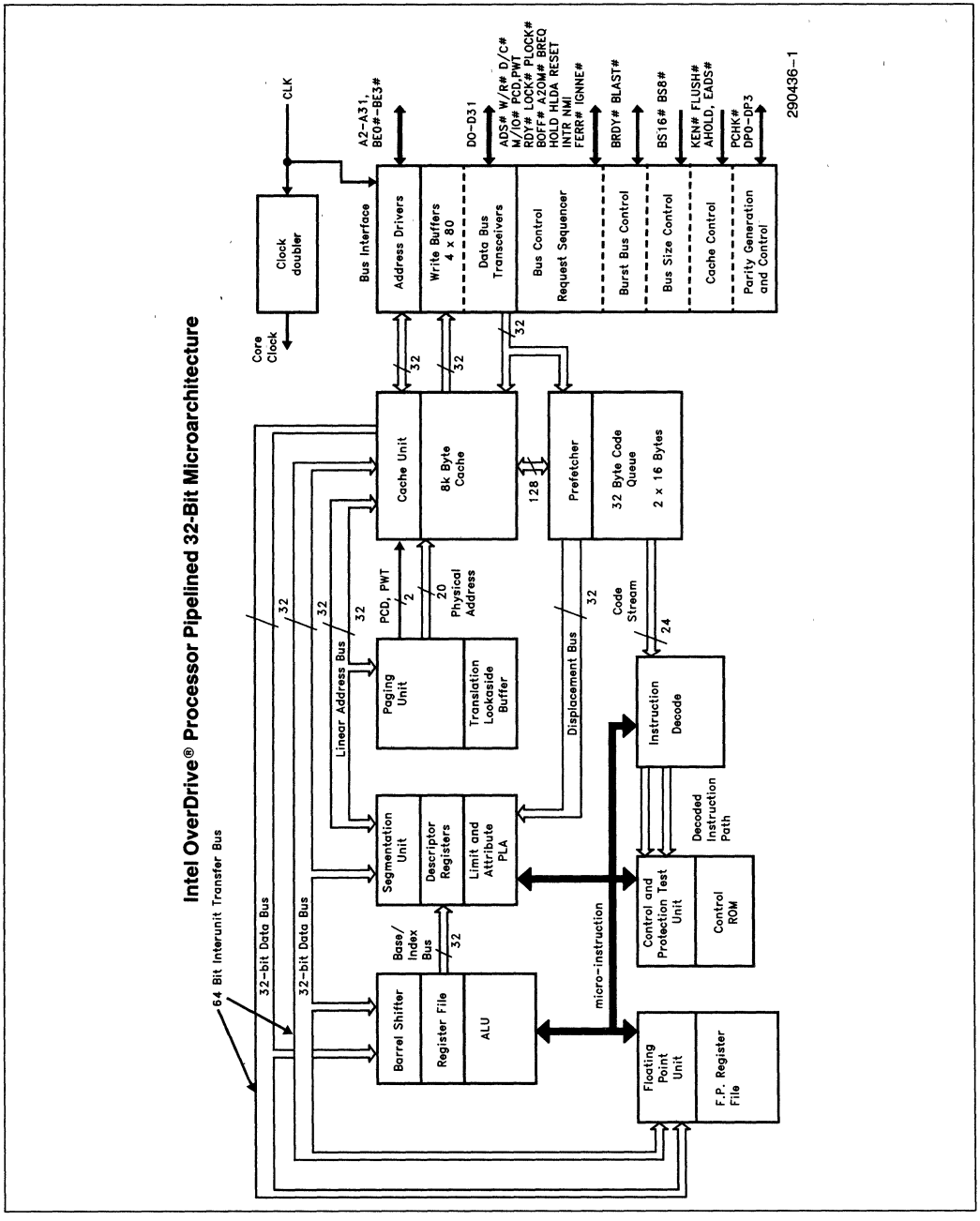


Figure 5-1. OverDrive® Processor Architecture Block Diagram

5.3.2 INTEL Pentium® OverDrive® PROCESSOR

Figure 5-2 shows the Pentium OverDrive processor's two pipelines and floating-point unit that are capable of independent operation. Each pipeline issues frequently used instructions in a single clock. Together, the dual pipes can issue two integer instructions in one clock, or one floating point instruction (under certain circumstances, 2 floating point instructions) in one clock.

The floating-point unit has been completely redesigned over the IntelDX2 processor. Faster algorithms provide at least 3X internal speed-up for common floating point operations including ADD, MUL, and LOAD. With instruction scheduling and overlapped (pipelined) execution, these three performance enhancements can allow many math intensive applications to achieve a 2X performance boost.

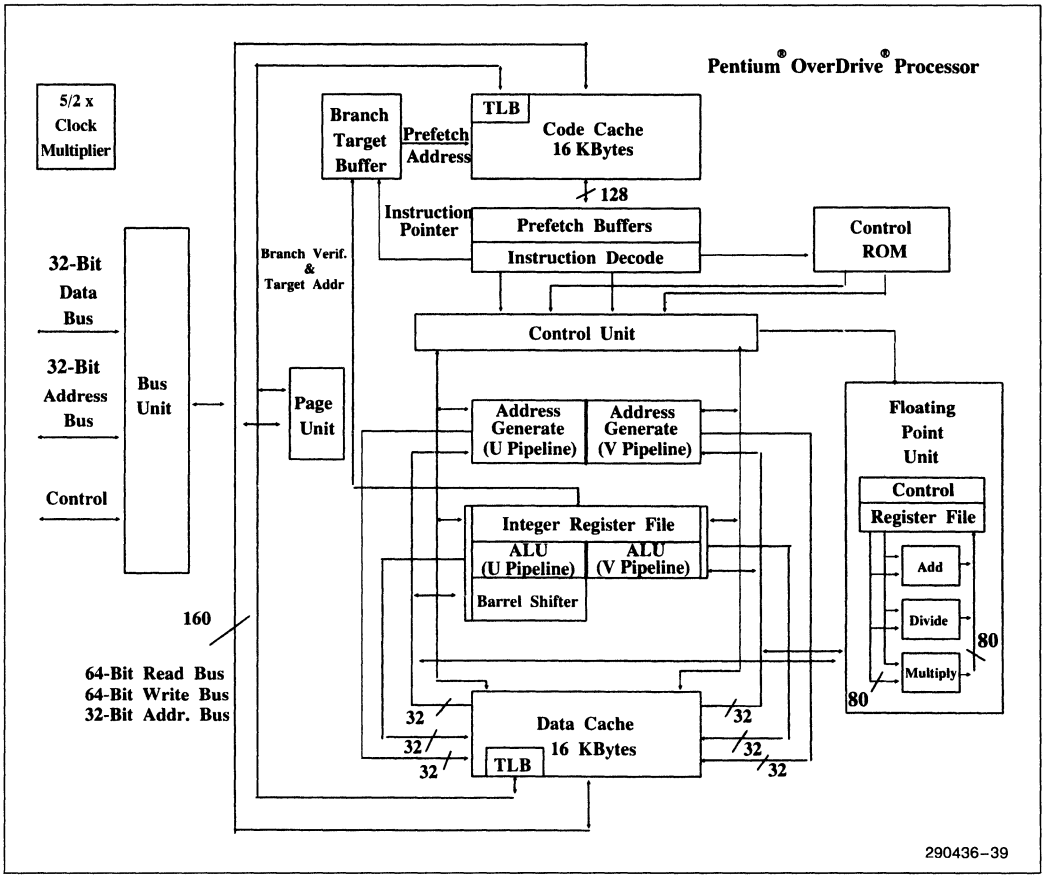


Figure 5-2. Pentium® OverDrive® Processor Block Diagram

The Pentium OverDrive processor implements 32-bit address and data busses.

The block diagram shows that the Pentium OverDrive processor contains two instruction pipelines, the "u" pipe and the "v" pipe. Both the u- and the v-pipes execute integer instructions, while only the u-pipe executes floating point instructions. The one exception is the FXCH instruction which may also be executed in the v-pipe. Therefore, the Pentium OverDrive processor is capable of executing two integer instructions in each clock, or one floating point instruction in each clock. Floating point instructions can be paired in certain conditions(1). Each pipeline has its own address generation logic, arithmetic logic unit and data cache interface.

NOTE:

1. Refer to the *Pentium Processor Data Book* for more information on instruction execution timing and pairing.

Note that there are two separate caches, a code cache and a data cache. The data cache has three tag ports, one for each of the two pipes and one dedicated to handle snoops from other processors. It has a dedicated Translation Lookaside Buffer (TLB) to translate linear addresses to the physical addresses used by the data cache.

The code cache, branch target buffer and prefetch buffers are responsible for getting raw instructions into the execution units of the Pentium OverDrive processor. Instructions are fetched from the code cache or from the external bus. Branch addresses are remembered by the branch target buffer. The code cache TLB translates linear addresses to physical addresses used by the code cache.

The decode unit decodes the prefetched instructions so that the Pentium OverDrive processor can execute the instruction. The control ROM contains the microcode which controls the sequence of operations that must be performed to implement the Pentium OverDrive processor architecture. The control ROM unit has direct control over both pipelines.

6.0 DIFFERENCES IN FUNCTIONALITY BETWEEN THE IntelDX2™ AND IntelDX4™ OverDrive® PROCESSOR FAMILY AND THE Intel486™ SX AND Intel486™ DX PROCESSORS

The IntelDX2 and IntelDX4 OverDrive processors are an enhanced family of Intel486 microprocessors. There are, however, four functional differences. First, the IntelDX2 and IntelDX4 OverDrive processors have an internal clock doubling (IntelDX2) or clock tripling (IntelDX4) circuit which decreases the time required to execute instructions. Second, the IntelDX2 and IntelDX4 OverDrive processor family does not support the JTAG boundary scan test feature. Third, the IntelDX2 and IntelDX4 OverDrive processors have different processor revision identifications than the Intel486 SX or Intel486 DX processors. Finally, the IntelDX4 OverDrive processor contains a 16 KByte cache, as opposed to the 8 KByte cache on the IntelDX2 OverDrive processors. These four differences are described in the following sections, according to how they affect the processor functionality.

6.1 Hardware Interface

The bus of the Intel OverDrive processors has been designed to be identical to the Intel486 Microprocessor bus. Although the external clock is internally doubled or tripled, and data and instructions are manipulated in the processor core at twice or three times the external frequency, the external bus is functionally identical to that of the Intel486 processor.

The four boundary scan test signals (TCK, Test clock; TMS, Test Mode select; TDI, Test Data Input; TDO, Test Data Output), defined for some Intel486 processors, are not specified for the Intel486 DX2 OverDrive processor.

The UP# (Upgrade Present) signal, which is defined as an input for some Intel486 processors, is an output signal on the Intel OverDrive processor. The UP# pin on the Intel OverDrive processor provides a logical low output signal which can be used to enable logic to recognize and configure the system for the Intel OverDrive processor. This signal is identical to the MP# output defined for the Intel487 SX Math CoProcessor.

The DX register always contains the component identifier at the conclusion of RESET. The Intel OverDrive processor has a different revision identifier in the DL register than the Intel486 SX or Intel486 DX microprocessors (refer to Section 11.1). When the OverDrive processor is installed in a system the component identifier is supplied by the OverDrive processor, rather than the original processor. The stepping identification portion of the component identification will change with different revisions of the OverDrive processor. The designer should only assume that the component identification for the OverDrive processor will be 043x for the IntelDX2 OverDrive processor and 148x or 048x for the IntelDX4 OverDrive processor, where "x" is the stepping identifier.

6.2 Testability

As detailed in Section 6.1, the Intel OverDrive processor does not support the JTAG boundary scan testability feature.

6.3 Instruction Set Summary

The Intel OverDrive processor supports all Intel486 extensions to the 8086/80186/80286 instruction set. In general, instructions will run faster on the Intel OverDrive processors than on the Intel486 microprocessor. Specifically, an instruction that only uses memory from the on-chip cache executes at the full core clock rate while all bus accesses execute at the bus clock rate. To calculate the elapsed time of an instruction, the number of clock counts for that instruction must be multiplied by the clock period for the system. The instruction set clock count summary tables from the Intel486 SX and Intel486 DX Microprocessor Data Sheets can be used for the OverDrive processor with the following modifications:

— Clock counts for a cache hit: This value represents the number of internal processor core clocks for an instruction that requires no external bus accesses or the base core clocks for an instruction requiring external bus accesses.

— Penalty clock counts for a cache miss: This value represents the worst-case approximation of the additional number of external clock counts that are required for an instruction which must access the external bus for data (a cache miss). This number must be multiplied by 2 (for the IntelSX2 and IntelDX2 OverDrive processors) or 3 (for the IntelDX4 OverDrive processor) to convert it to an equal number of internal processor core clock counts and added to the base core clocks to compute the number of core clocks for this instruction.

The actual number of core clocks for an instruction with a cache miss may be less than the base clock counts (from the cache hit column) plus the penalty clock counts (2 times the cache miss column number for the IntelSX2 and IntelDX2, 3 times the cache miss column number for the IntelDX4). The clock counts in the cache miss penalty column can be a cumulative value of external bus clocks (for data reads) and internal clocks for manipulating the data which has been loaded from the external bus. The number of clocks which are related to external bus accesses are correctly represented in terms of internal core clocks by multiplying by two. However, the clock counts related to internal data manipulation should not be multiplied by two. Therefore the total number of processor core clock counts for an instruction with a cache miss represents a worst-case approximation.

To calculate the execution time for an OverDrive processor instruction, multiply the total processor core clock counts by the core clock period. For example, in a 25 MHz system upgraded with a 50 MHz IntelDX2 OverDrive processor, the core clock period is 20 ns (1/50 MHz).

Additionally, the assumptions specified below should be understood in order to estimate instruction execution time.

A cache miss will force the OverDrive processor to run an external bus cycle. The Intel486 microprocessor 32-bit burst bus is defined as r−b−w.

Where:

r = The number of bus clocks in the first cycle of a burst read or the number of clocks per data cycle is a non-burst read.

b = The number of bus clocks for the second and subsequent cycles in a burst read.

w = The number of bus clocks for a write.

The fastest bus the OverDrive processor can support is 2−1−2 assuming 0 waits states. The clock counts in the cache miss penalty column assume a 2−1−2 bus. For slower busses add r−2 clocks to the cache miss penalty for the first dword accessed. Other factors also affect instruction clock counts.

Instruction Clock Count Assumptions

1. The external bus is available for reads or writes at all times. Else add bus clocks to reads until the bus is available
2. Accesses are aligned. Add three core clocks to each misaligned access.
3. Cache fills complete before subsequent accesses to the same line. If a read misses the cache during a cache fill due to a previous read or prefetch, the read must wait for the cache fill to complete. If a read or write accesses a cache line still being filled, it must wait for the fill to complete.
4. If an effective address is calculated, the base register is not the destination register of the preceding instruction. If the base register is the destination register of the preceding instruction add 1 to the core clock counts shown. Back-to-back PUSH and POP instructions are not affected by this rule.
5. An effective address calculation uses one base register and does not use an index register. However, if the effective address calculation uses an index register. 1 core clock may be added to the clock shown.
6. The target of a jump is in the cache. If not, add r clocks for accessing the destination instruction of a jump. If the destination instruction is not completely contained in the first dword read, add a maximum of 3b bus clocks. If the destination instruction is not completely contained in the first 16 byte burst, add a maximum of another r+3b bus clocks.
7. If no write buffer delay, w bus clocks are added only in the case in which all write buffers are full.
8. Displacement and immediate not used together. If displacement and immediate used together, 1 core clock may be added to the core clock count shown.
9. No invalidate cycles. Add a delay of 1 bus clock for each invalidate cycle if the invalidate cycle contends for the internal cache/external bus when the OverDrive processor needs to use it.
10. Page translation hits in TLB. A TLB miss will add 13, 21 or 28 bus clocks + 1 possible core clock to the instruction depending on whether the Accessed and/or Dirty bit in neither, one or both of the page entries needs to be set in memory. This assumes that neither page entry is in the data cache and a page fault does not occur on the address translation.
11. No exceptions are detected during instruction execution. Refer to interrupt core Clock Counts Table for extra clocks if an interrupt is detected.
12. Instructions that read multiple consecutive data items (i.e., task switch, POPA, etc.) and miss the cache are assumed to start the first access on a 16-byte boundary. If not, an extra cache line fill may be necessary which may add up to (r+3b) bus clocks to the cache miss penalty.

7.0 DIFFERENCES BETWEEN THE Intel486™ FAMILY AND THE Pentium® OverDrive® PROCESSOR

This section covers the differences between the Intel486 family of microprocessors and the Pentium OverDrive processor.

7.1 Software

The Pentium OverDrive processor is compatible with the entire installed base of applications for MS-DOS*, Windows*, OS/2*, and UNIX*. In addition to being binary compatible with the Intel architecture based processors which preceded the Pentium processor, it can also run programs which have been compiled to utilize Pentium processor instructions. This allows pairing of instructions to take advantage of the Superscalar Architecture and adds floating point instructions if replacing an Intel486 SX or IntelSX2 processor.

7.2 Hardware

The Pentium OverDrive processor is in a 235-Pin Grid Array package while the Intel486 family uses a 168-Pin Grid Array package. These extra pins are used for extra power and ground pins, including separate power and ground for the fan heatsink, also the internal cache write-back signals are on these pins.

8.0 DIFFERENCES BETWEEN THE Pentium® PROCESSOR AND THE Pentium® OverDrive® PROCESSOR

There are several differences between the Pentium processor and the Pentium OverDrive processors. These differences are covered in the next two sections.

8.1 Software

The following paragraphs are provided as a reference for software differences between the Pentium OverDrive processor and the Pentium processor. Unless otherwise stated, it can be assumed that the Pentium OverDrive processor will have the same software characteristics as the Pentium processor. For more information on the software characteristics of the Pentium Processor, please see the *Pentium Processor Data Book* or *Programmers Reference Manual.*

The Pentium OverDrive processor does not support machine check exception. As a result, CR4.MCE is not useful. It will be forced to a zero by the hardware. However, any attempt by the software to set this bit to 1 will not create a general protection exception.

The Pentium OverDrive processor allows two different page sizes: 4 KB and 4 MB. Please contact Intel for more information on the use of 4 MB pages.

The behavior of INVD, WBINVD and INVPLG instructions are similar to the Pentium processor. If the processor is in **the enhanced bus mode,** the WBINVD instruction will write back all dirty lines first, flush the cache and run two special cycles (the write back special cycle followed by the flush special cycle) on the bus. INVD will flush the cache and run one special cycle (The flush special cycle) on the bus. INVD will not write back the dirty lines, if any. All three instructions are privileged level 0 and should be executed by BIOS or operating system code only.

On the IntelDX2 processor, when paging is disabled, and/or when instructions that are not affected by paging are executed, the **PCD** and **PWT** pins are driven with values from CR3.PCD and CR3.PWT bits, respectively. On the Pentium OverDrive processor, when paging is disabled and/or when instructions that are not affected by paging are executed, the **PWT** pin will be drive LOW and the **PCD** pin will reflect the value of the CR0.CD bit.

When the CPUID instruction is performed with EAX = 1, the Pentium OverDrive processor will return the following values to the EDX register. These bits indicate what features the processor has.

The various bit positions have the following meaning:

Table 8-1. Feature Bit Assignment

Bit	Value	Meaning
0	1	FPU: Floating Point Unit On-Chip
1	1	VME: Virtual-8086 Mode Enhancements
2	1	DE: Debugging Extensions
3	1	PSE: Page Size Extension
4	1	TSC: Time Stamp Counter
5	1	MSR: Pentium Processor-Style MSR
6	R	Reserved
7	R	Reserved
8	1	CX8: CMPXCHG8B Instruction
9–31	R	Reserved

A value of "R" means that the corresponding bit is reserved and the software should not depend on its value.

8.2 Hardware

This section covers the hardware differences between the Pentium processor and the Pentium OverDrive processor.

The Pentium processor has a 64-bit data bus while the Pentium OverDrive processor has a 32-bit data bus. This is required in order for the Pentium OverDrive processor to work in an Intel486 architecture based system.

The size of the data bus requires a similar decrease in the internal cache line size. A burst memory access is four bus widths of data this is the length of each line in the cache. For an Intel486 based architecture system that is 16-Bytes, for a Pentium processor based system this is 32-bytes. The internal cache in the Pentium processor is expecting 32-bytes. This would require two burst reads. Therefore the cache line size was reduced to 16-bytes wide.

The cache on the Pentium processor is split into two 8 KByte caches. One is for data and the other is the instruction or code cache. The Pentium OverDrive processor keeps the idea of two separate caches but increases the size to 16 KBytes each.

The Pentium OverDrive processor does not have the JTAG boundry scan capabilities that the Pentium processor has.

The package is different. The Pentium processor is packaged in a 273-Pin Grid Array while the Pentium OverDrive processor uses a 235-Pin Grid Array package.

The Pentium OverDrive processor has the integrated fan heatsink attached. This integrated fan informs the processor if the fan has slowed or stopped. This reduces the internal frequency to 1x multiplier from the previous 2.5x. The part can run indefinitely at the lower frequency without incurring any damage. This allows the Pentium OverDrive processor to continue in the system until a replacement fan can be installed.

9.0 IntelDX2™ AND IntelDX4™ OverDrive® PROCESSOR CIRCUIT DESIGN

9.1 Upgrade Circuit for Intel486™ Processor-Based Systems with UP#

Figure 9-1 shows the IntelDX2 and IntelDX4 OverDrive processor socket circuit for Intel486 processor-based systems using UP#. The Upgrade Present input, UP# pin, allows the Intel486 processor to directly recognize when the Intel OverDrive processor socket is populated. When the UP# pin is driven active to the Intel486 processor, the Intel486 processor tri-states all of its output pins and enters power-down mode.

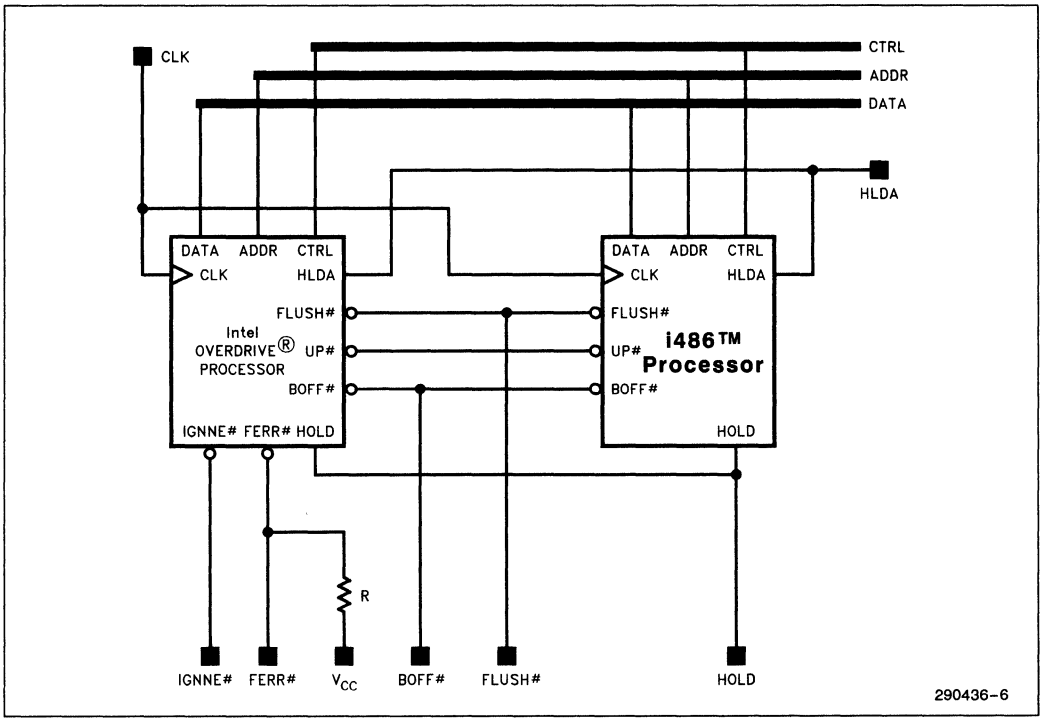


Figure 9-1. Intel OverDrive® Socket Circuit Diagram for Systems Based on Intel486™ Processors That Have the UP# Input Pin

10.0 Pentium® OverDrive® PROCESSOR DESIGN CONSIDERATIONS

10.1 Hardware Design Considerations

10.1.1 INTRODUCTION

This section describes the organization of the Translation Lookaside Buffers (TLBs), write buffers and the buffering scheme used in the Pentium OverDrive processor. Other important Pentium OverDrive Processor design specific details are also included as well as the use of the **WB/WT#** pin as an input to determine the fundamental cache operation mode of the processor.

10.1.1.1 Cache Consistency Cycles

The external system can check and invalidate cache lines in the internal processor cache using inquire cycles (snooping). Snooping allows the external system to keep cache coherency throughout the system. Snoop cycles may be performed using **AHOLD, BOFF#**, or **HLDA** and then asserting **EADS#** to inform the processor that the snoop address is available on the bus. The following table summarizes the snoop mechanism initiated by any of these three control signals.

The snoop cycle begins by checking whether a particular cache line has been "cached" and invalidates the line based on the state of the **INV** pin. If the Pentium OverDrive processor is configured in the Standard Bus Mode, the processor will always invalidate the cache line on snoop hits. If the processor is configured in the Enhanced Bus Mode, the system must drive **INV** high to invalidate a particular cache line. The Pentium OverDrive processor will invalidate the line and write back an E-state line if the system snoop hits either S-state, E-state, or M-state line, provided **INV** was driven high during **EADS#** assertion. If **INV** is driven low, a modified line will be written back to memory and will remain in the cache as a write-back line. If **INV** is driven low, a shared line also will continue to remain in the cache as a shared line.

Table 10-1. Snoop Cycles under AHOLD, BOFF#, or HLDA

AHOLD	Tri-states the address bus. **ADS#** will be asserted under **AHOLD** only to initiate a snoop writeback cycle. An ongoing burst cycle will complete under an **AHOLD.** For non-burst cycles, a specific non-burst transfer (**ADS# - RDY#** transfer) will complete under **AHOLD** and will be fractured (interrupted) before the next assertion of **ADS#.** A snoop writeback cycle will be reordered ahead of a fractured non-burst cycle. Should an **ADS#** be required to start the next cycle while **AHOLD** is asserted, the processor will only perform snoop write back cycles. If the processor issues and **ADS#** while **AHOLD** is asserted, it is the responsibility of the system to determine the address of the snoop write back from the snoop address driven with **EADS#.** An interrupted non-burst cycle will be completed only after the snoop writeback cycle is completed, provided there are no other snoop writeback cycles scheduled and **AHOLD** is deasserted. If **BLEN#** is driven inactive (disabling bursted writes), the processor will drive four individual **ADS# - RDY#** cycles to complete the cycle while **AHOLD** is active.
BOFF#	Overrides **AHOLD;** takes effect in the next clock. Ongoing bus cycles will stop in the clock following **BOFF#** being asserted and resumes when **BOFF#** is deasserted, in the same manner as the standard Intel486 processor bus. A snoop writeback will be reordered ahead of the backed off cycle. The snoop writeback cycle begins after **BOFF#** is deasserted followed by any backed off cycle.
HOLD	**HOLD** will be acknowledged only between bus cycles, except for a non-cacheable, non-bursted code prefetch cycle. In a non-cacheable, non-bursted code prefetch cycle, **HOLD** is acknowledged after the system returns **RDY#** or if **BOFF#** is asserted. Once **HLDA** is active, the processor blocks all bus activities until the system releases the bus (by de-asserting **HOLD**).

After asserting **AHOLD** or **BOFF#**, the external master driving a snoop cycle must wait at least two clocks before asserting **EADS#**. If snooping is done after **HLDA** assertion, then the master performing a snoop must wait for at least one clock cycle before driving the snoop addresses and asserting **EADS#**. **INV** should be driven low during bus master read operations to minimize invalidations. **INV** should be driven high to invalidate a cache line during bus master write operations. The Pentium OverDrive processor asserts **HIT#** and **HITM#** if the cycle hits an M state line in the cache or **HIT#** only if the cycle hits an E or S state line. These output signals become valid two clock periods after **EADS#** is valid on the bus. **HITM#** will remain asserted at least until the last **RDY#** or **BRDY#** of the snoop writeback cycle is returned. The **HIT#** signal will continue to drive the result of the last snoop until the next external snoop occurs. Most timing diagrams in the following sections do not include the **HIT#** signal since it is not necessary for single processor system designs. Snoop operations may interrupt an ongoing bus operation in both the Standard Bus Mode and Enhanced Bus Mode.

The Pentium OverDrive processor can accept EADS# in every clock period while in the Standard Bus Mode. In the Enhanced Bus Mode, the processor can accept EADS# every other clock period until the external snoop hits an M-state line. Any **EADS#** assertion after the **EADS#** that hit a modified line will be ignored. The processor will not accept any further **EADS#** assertions until the snoop writeback operation is completed and **HITM#** is deasserted. Figure 10-1 shows the allowable **EADS#** window for the different snooping mechanisms (**AHOLD, HOLD, BOFF#**). **For the Enhanced bus mode, EADS# must not be asserted outside the windows presented in the diagrams below.**

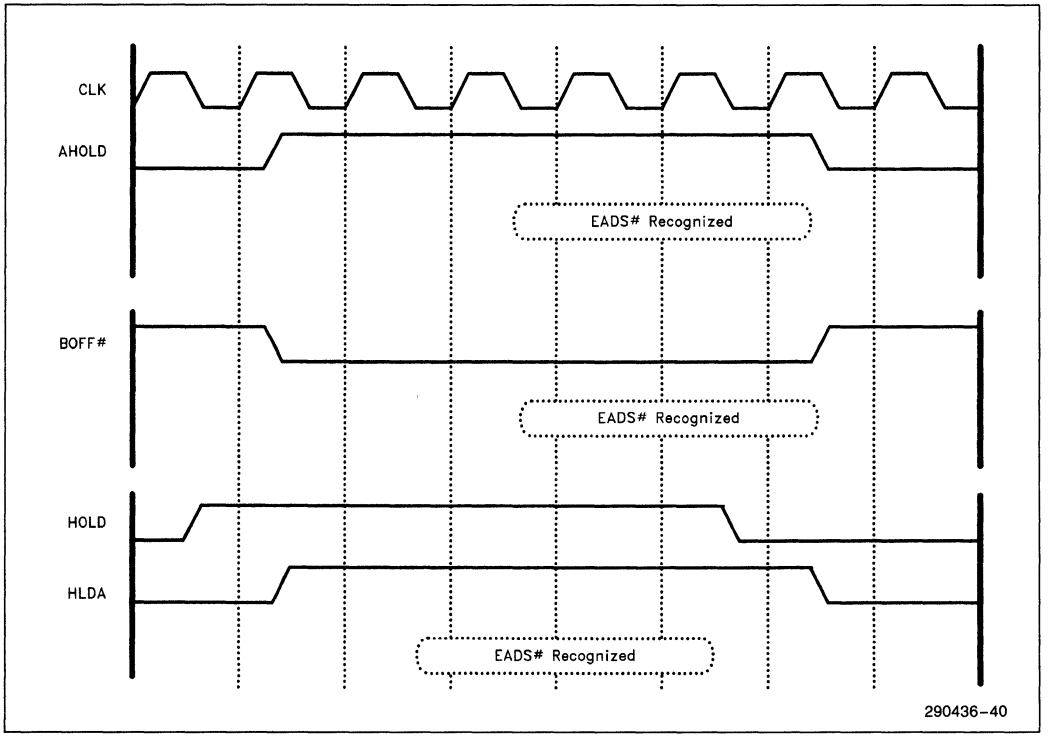


Figure 10-1. EADS# Snooping Window

Figure 10-2 shows that the processor can accept an **EADS#** assertion only every other clock while in enhanced bus mode.

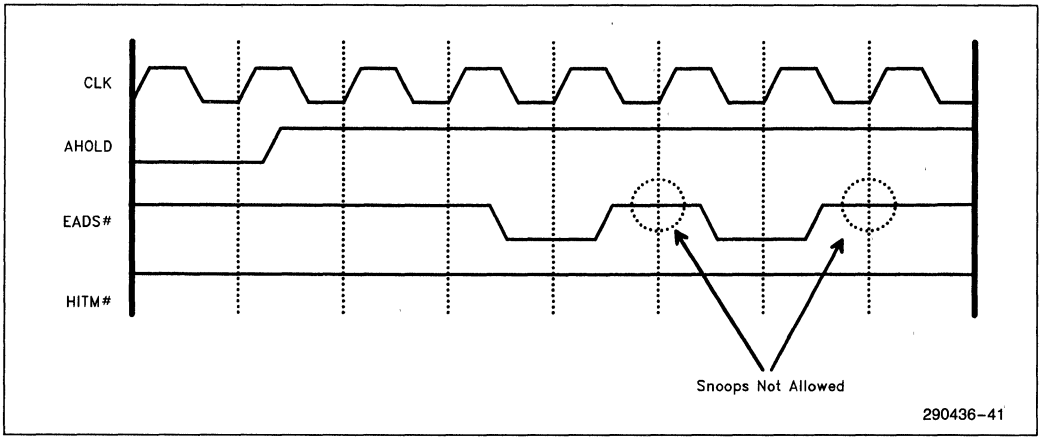


Figure 10-2. EADS# Snooping Frequency in Enhanced Bus Mode

10.1.1.2 Write Back Cycles

Writeback cycles may be bursted or non-bursted by returning **RDY#** or **BRDY#**. All writeback operations write 16 bytes of data to memory corresponding to the modified line in the cache. **BS8# and BS16# are not allowed during writeback cycles.**

The state of **BLEN#** determines how a 16 byte line is written back. When **BLEN#** is LOW, the write back is done as a 16-byte burst transfer. **BRDY#** or **RDY#** may terminate each transfer. At the fourth transfer, **BLAST#** will signify the end of the write back. A write back when **BLEN#** is LOW but **RDY#** is used rather than **BRDY#** is shown in Figure 10-4. When **BLEN#** is HIGH, the 16-byte write back will be done as four consecutive 4 byte writes, each with its own **ADS#** and **BLAST#**. (See Figure 10-5).

BLEN# is not a dynamic pin since it cannot be toggled on a cycle per cycle basis. It cannot be changed once power has been applied to the system, so it should be tied HIGH or LOW.

The **CACHE#** pin is used to indicate that a cache operation is taking place. **CACHE#** is active with the first **ADS#** for both write backs and line fills, and is terminated by the first **RDY#/BRDY#**. An occurrence of **ADS#** = 0, **W/R#** = 1, and **CACHE#** = 0 indicates that a replacement write back is starting. The occurrence of **HITM#** = 0 during the above operation signifies that a snoop write back is occurring. Write back cycles begin at address 0x0 of the 16-byte line being pushed out. The burst order is the standard Intel486 Processor order of 0x0, 0x4, 0x8 and 0xC. If the write back is done as four separate writes, then each write will push out four bytes starting at byte 0x0. **PCD** and **CACHE#** are low during write back cycles, while **KEN#** is ignored.

BS8#, BS16# are ignored during write back cycles. After the last **BRDY#/RDY#** of a write back cycle is asserted, the Pentium OverDrive Processor will wait at least one **CLK** before issuing the next **ADS#**. In other words, a dead clock is inserted by the processor after the last transfer in a write back cycle. The dead clock is between the last **BRDY#/RDY#**, and the next **ADS#**. The dead clock appears only after the write back is complete, so there are no dead clocks between individual transfers of a write back. This dead clock time is used by the Pentium OverDrive processor to complete internal cache operations.

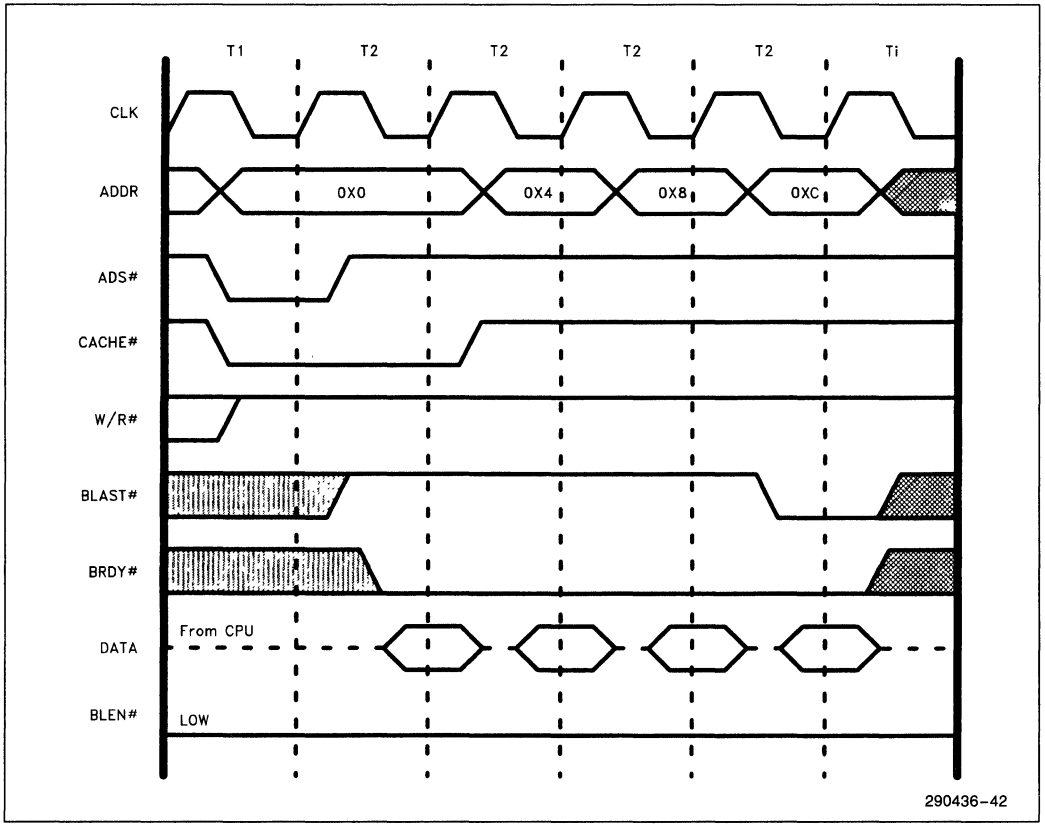


Figure 10-3. Write Back Operation—BLEN# Active

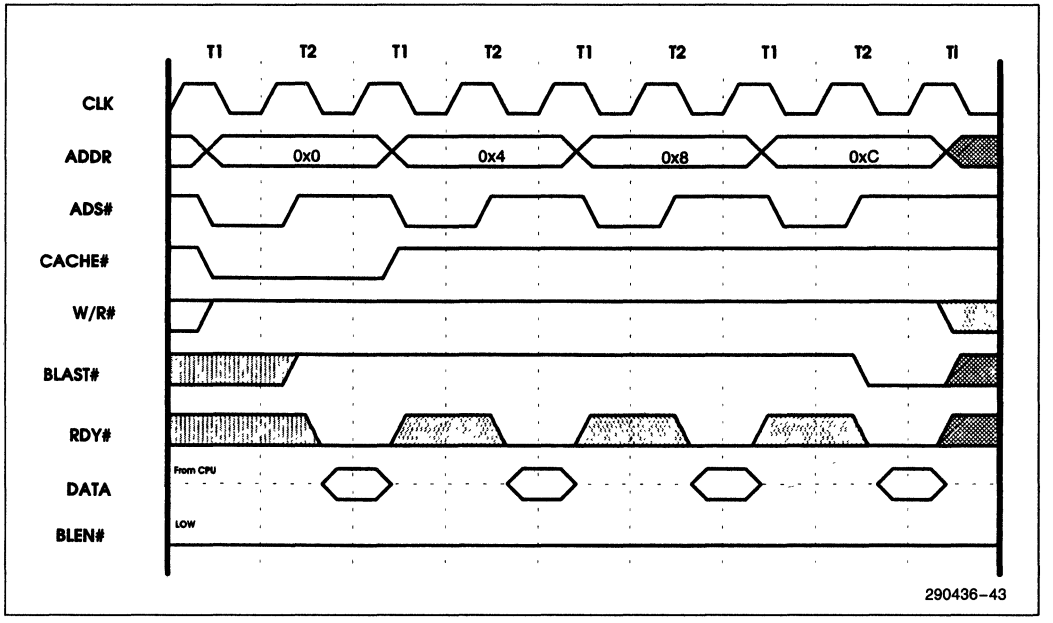


Figure 10-4. Write Back with BLEN# Active Using RDY#

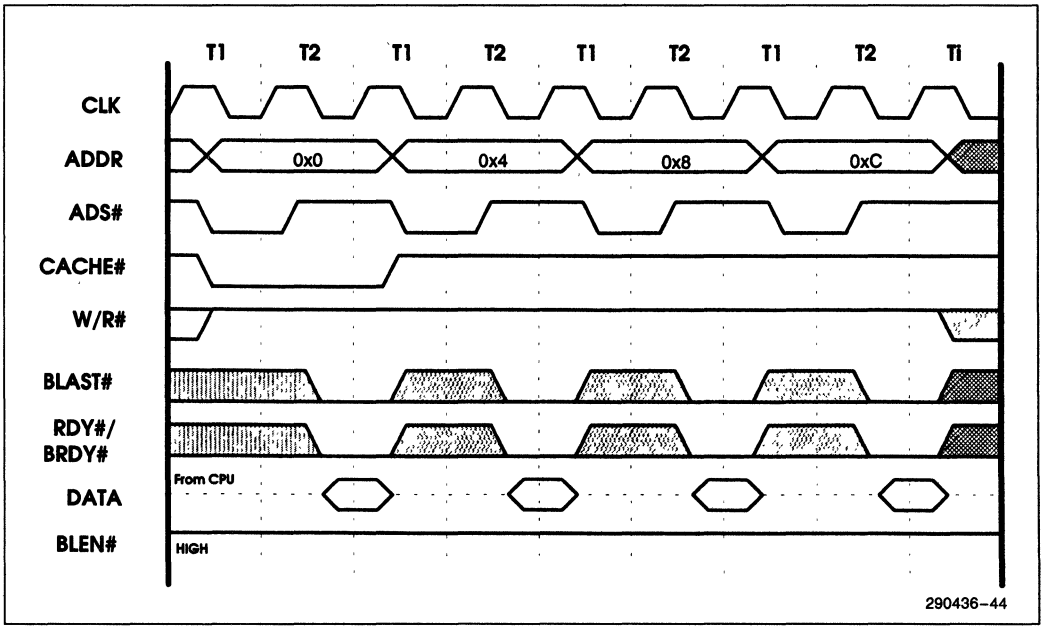


Figure 10-5. Write Back as Four 4-Byte Transfers (BLEN# Inactive)

10.1.2 WB/WT# AS AN INITIALIZATION INPUT

To make the Pentium OverDrive processor more compatible with the IntelDX2 processor, several small enhancements have been added to the processor to help distinguish different processor modes of operation. These two modes will be referred to as **"standard bus mode" (Write through processor cache only)** and "enhanced bus mode" (Write back processor enabled and are discussed in the following paragraphs.

In order to allow the Pentium OverDrive processor to operate in more of an IntelDX2 processor manner, the **WB/WT#** pin is used as an initialization input to configure the operating mode of the processor. At the falling edge of **RESET,** the processor can be configured to operate in a write through only (IntelDX2 CPU compatible) L1 cache mode (**standard bus** mode), or in a write back L1 cache mode (enhanced bus mode). Once a mode is selected, the processor will continue to operate in the selected mode and can only be changed to a different mode by starting the **RESET** process again. Assertion of **INIT** will not change the operating mode of the processor. **WB/WT#** has an internal pulldown that will force any design that leaves **WB/WT#** unconnected into the write through mode of operation. Table 10-2 lists the two modes of operation and the differences between them.

For more information on the effect of the mode operation on the various signals mentioned above, please see the *Intel486 Microprocessor Family Data Book*. Unless otherwise mentioned, all other functions of the Pentium OverDrive processor remain identical in both operating modes.

10.1.3 INIT FUNCTIONALITY

INIT behaves like an edge triggered interrupt on the Pentium OverDrive processor while in both enhanced bus mode (**WB/WT#** = HIGH at **RESET**) and standard bus mode (**WB/WT#** = LOW at **RESET**). Therefore, when **INIT** is asserted, there is a high probability that one or more bus cycles will be run by the processor while **INIT** is HIGH. The following figures demonstrate two scenarios of how the processor can issue an **ADS#** while **INIT** is asserted. Although the figures assume that an initial I/O write causes the **INIT** to be asserted by the support logic, these cases apply whenever **INIT** is HIGH. The first figure describes cycles being run by the processor before **INIT** is recognized. The second details cycles being run after **INIT** has been recognized.

Figure 10-6 shows the **INIT** signal being triggered by an I/O write. Even though **INIT** is asserted immediately, there is a pending prefetch which executes before the **INIT** is recognized.

Table 10-2. Effects of WB/WT# Initialization

State of WB/WT# at Reset Falling	Affect on Processor Operation
WB/WT# = LOW	Processor is in **Standard Bus** Mode ** IntelDX2 Processor Compatible ** 1: No Special **FLUSH#** Acknowledge Cycles are run on the bus after the assertion of the FLUSH# pin. 2: When **FLUSH#** is asserted, the caches will be invalidated in 15–20 system **CLKs**. 3: All Write Back specific inputs are ignored—**(BLEN#, EWBE#, WB/WT#, INV)** 4: **EADS#** is sampled at any time.
WB/WT# = HIGH	Processor is in **Enhanced Bus** Mode ** Intel486 Processor Write Back Bus Operation ** 1: The special **FLUSH#** Acknowledge Cycles will be run on the bus after the assertion of the **FLUSH#** and all the cache write backs (if any) are complete. 2: Write backs will be performed if a cache flush is requested (i.e.: **FLUSH#, WBINVD** inst . . .). The flush will take about 2000+ clocks. The system must watch for the FLUSH# special cycles to determine the end of the flush. 3: **WB/WT#** is sampled on a line by line basis to determine the storage state of a cache line on reads and writes. 4: The **BLEN#, EWBE#** and **INV** are no longer ignored. 5: **EADS#** is sampled only when the processor is an a **HOLD, AHOLD,** or **BOFF#** state. 6: **PLOCK#** is inactive and driven HIGH.

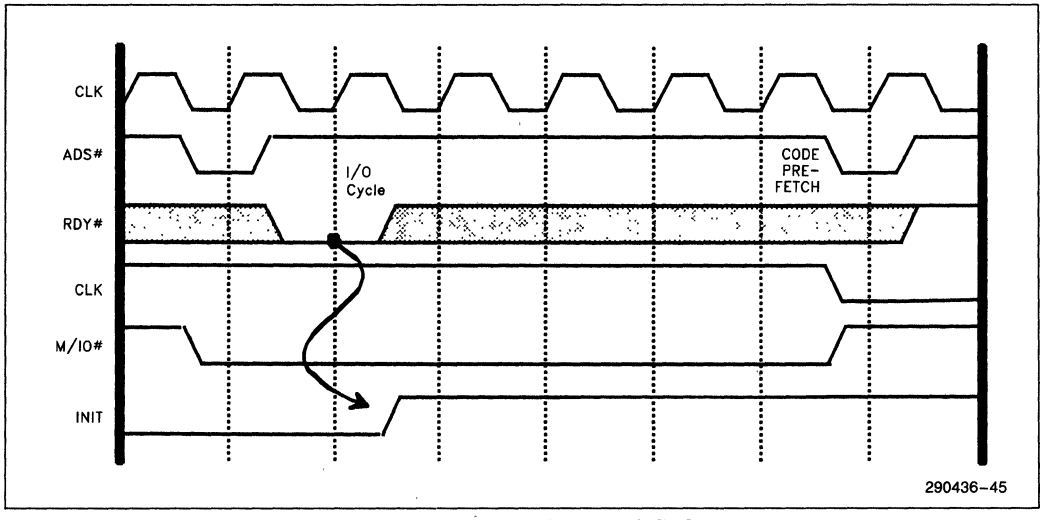


Figure 10-6. ADS# Issued during INIT: Case 1

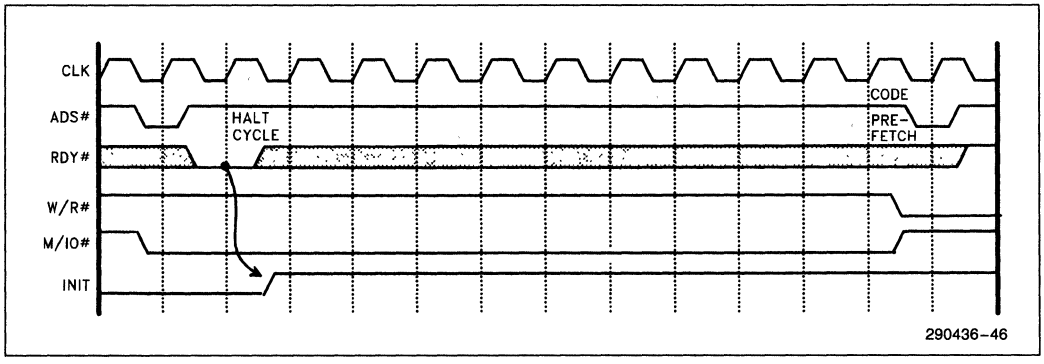


Figure 10-7. ADS# Issued during INIT: Case 2

Figure 10-7 assumes that an I/O cycle was used to generate the **INIT,** but the system waited until all bus activity had stopped (by monitoring the HALT special cycle) before asserting the **INIT** pin. In this case, the processor recognizes **INIT,** and starts a prefetch before **INIT** is deasserted. The prefetch may be from any location, and not necessarily the F..F0h address, even though **INIT** has been recognized.

On other Intel486 write through cache processors (Write-Back Enhanced IntelDX2 in standard bus mode and SL-Enhanced CPUs), the **SRESET** pin is executed immediately, in a manner similar to the **RESET** pin. This means that no cycles will be run after **SRESET** is asserted on these processors. With the Pentium OverDrive processor, it is the responsibility of the system to ensure that any cycles that are issued while INIT is active are completed properly to prevent data corruption, lost bus cycles or system lock-ups.

10.1.4 INSTRUCTION PREFETCH

The Pentium OverDrive processor contains a prefetch buffer of several bytes, and can prefetch a significant number of bytes beyond the end of the last executed instruction. In addition, the processor implements a dynamic branch prediction algorithm which speculatively runs code fetch cycles to addresses corresponding to instructions executed some time in the past. Such code fetch cycles are run based on past execution history, regardless of whether the instructions retrieved are relevant to the currently executing instruction sequence.

The effect of both mechanisms is that the Pentium OverDrive processor may run code fetch bus cycles to retrieve instructions which are never executed. Although the opcodes retrieved are discarded, the system must complete the code fetch bus cycle by returning **RDY#/BRDY#.** It is particularly important that the system return **RDY#/BRDY#** for all code fetch cycles, regardless of the address.

Furthermore, it is possible that the processor may run speculative code fetch cycles to addresses beyond the end of the current code segment. Although the processor may prefetch beyond the CS limit, it will not attempt to execute beyond the CS limit, it will raise a GP fault instead. Thus, segmentation cannot be used to prevent speculative code fetches to inaccessible areas of memory. On the other hand, the processor will never run code fetch cycles to inaccessible pages, so the paging mechanism guards against both the fetch and execution of instructions in inaccessible pages.

If the processor has been placed in a halt state with the **HLT** instruction, and the processor has issued the halt cycle, or the processor has run a shutdown cycle, one of the methods of exiting this condition is to assert an interrupt. Once the interrupt is asserted, the Pentium OverDrive processor may issue a prefetch before the interrupt is acknowledged with an interrupt cycle, or the action desired by the interrupt is performed. For example, if the processor is in a halt state, and the **FLUSH#** interrupt is asserted, the processor could exit the halt state, perform a prefetch, and then start driving the write backs for the flush operation. Prefetches of this type may be run after any interrupt, including **INTR.**

For memory reads and writes, both segmentation and paging prevent the generation of bus cycles to inaccessible regions of memory.

10.2 Upgrade Circuit Design

10.2.1 DUAL PROCESSOR SITE DESIGN

The Pentium OverDrive processor can reside on the same processor bus as an Intel486 processors. The Pentium OverDrive processor specifies a **UP#** output (Upgrade Present) pin which should be connected directly to the **UP#** input pin of the Intel486 microprocessor. When the Pentium OverDrive processor occupies the Pentium OverDrive processor socket, the UP# signal (active low) forces the Intel486 microprocessor to tri-state all outputs and reduce power consumption. When the Pentium OverDrive processor is not present, a pull-up resistor, internal to the Intel486 microprocessor, drives **UP#** inactive and allows the Intel486 microprocessor to control the processor bus.

2

10.2.2 SINGLE PROCESSOR SITE DESIGN

A single processor site is defined as a system design that can accept all Intel486 processors in a single socket location. Doing a single socket design requires that certain pins are connected via the **INC** pins of the Pentium OverDrive processor. See Section 9 for more details on how to design a single socket processor site compatible other Intel486 processors.

10.2.3 CIRCUIT CONSIDERATIONS FOR WRITE BACK CACHE SUPPORT

The Pentium OverDrive processor is specified to support the MESI write back protocol for the on-chip cache. To support the write back protocol, seven new signals , are defined for the Pentium OverDrive processor, four of which are present on the Write-Back Enhanced IntelDX2 processor. The new signals defined for MESI write back capability are **WB/WT#, INV, HIT#, HITM#, CACHE#, EWBE#,** and **BLEN#.** Another new pin, **INIT,** is used to facilitate warm resets. These new signals are defined in detail in Section 5.2. For more information on designing a system for processor write back cache support that is compatible with the Write-Back Enhanced IntelDX2 processor, please see Appendix B.

10.3 Software Considerations

10.3.1 EXTERNAL BUS CYCLE ORDERING

10.3.1.1 Write Buffers and Memory Ordering

The Pentium OverDrive processor has write buffers to enhance the performance of consecutive writes to memory. Writes in these buffers are driven out on the external bus in the order they were generated by the processor core. No reads (as a result of cache miss) are reordered around previously generated writes sitting in the write buffers (Unlike the IntelDX2 processor). The implication of this is that the write buffers will be emptied before a subsequent processor generated bus cycle is run on the external bus.

It should be noted that only memory writes are buffered and I/O writes are not. There is no guarantee of synchronization between completion of memory writes on the bus and instruction execution after the write. The OUT instruction or a serializing instruction needs to be executed to synchronize writes with the next instruction. Please refer to the *Pentium Processor Programmers Reference Manual* for information on serializing instructions.

No re-ordering of read cycles occurs on the Pentium OverDrive processor. Specifically, the write buffers are emptied before the IN instruction is executed.

10.3.1.2 External Event Synchronization

When the value of **NMI, INTR, FLUSH#, SMI#** or **INIT** changes as the result of executing an OUT instruction, these inputs must be at a valid state three clocks before **RDY#/BRDY#** is returned to ensure that the new value will be recognized before the next instruction is executed.

Note that if an OUT instruction is used to modify **A20M#**, this will not affect previously prefetched instructions. A serializing instruction must be executed to guarantee recognition of **A20M#** before a specific instruction.

10.3.2 MODEL SPECIFIC REGISTERS

The Pentium OverDrive processor defines certain Model Specific Registers that are used in execution tracing, performance monitoring and testing. They are unique to the Pentium OverDrive processor and may or may not be implemented in the same way in future processors.

Please contact Intel for more information on the model specific registers.

10.3.3 EXCEPTION PRIORITIES

Exceptions are serviced and recognized on the boundary between instructions. The instruction pointer pushed onto the stack for the interrupt handler points to the next instruction. The priority among simultaneous exceptions is as follows (interrupt vector numbers are shown in decimal in parentheses):

Trap on the previous instruction:

- Breakpoint (#3)

External Interrupts:

- **FLUSH#**
- **SMI#**
- **INIT**
- **NMI**
- **INTR**

Floating Point Errors:

- **FERR#**

External Interrupt:

- **STPCLK#**

Faults on Fetching Next Instruction:

- Code Seg Limit Violation (#13), Page Fault on prefetch (#14) (the relative priority unpredictable)

Faults in Decoding the next Instruction:

- Invalid Opcode (#6), Device Not Available (#7)
- General Protection Fault for Instruction Length > 15B (#13)

Faults on Executing an Instruction (These may occur in a manner that varies from implementation to implementation as necessary to insure functional correctness. They are not listed in any particular order):

- General Detect (#1)
- FP Error (from previous FP instruction) (#16)
- Interrupt on Overflow (#4)
- Bound (#5)
- Invalid TSS (#10)

- Segment Not Present (#11)
- Stack Exception (#12)
- General Protection (#13)
- Data Page Fault (#14)
- Alignment Check (#17)

10.3.3.1 External Interrupt Considerations

The Pentium OverDrive processor recognizes the following external interrupts: **FLUSH#, SMI#, INIT, NMI, INTR** and **STPCLK#.** They are listed in priority order, however, they are subject to the considerations listed below.

FLUSH#

Note that unlike the Intel486 CPU which invalidates its cache a small fixed number of clocks after **FLUSH#** is asserted, the **FLUSH#** pin on the Pentium OverDrive processor is an interrupt and therefore is only recognized at the boundary between instructions. While **FLUSH#** is being serviced, all the dirty lines in the data cache are written back to main memory. This may take several thousand clocks. During this time no instructions are executed and no other interrupts are recognized. If the processor is in the HALT or Shutdown state, **FLUSH#** is still recognized. The processor will return to the HALT or Shutdown state after servicing the **FLUSH#.**

11.0 BIOS AND SOFTWARE

The following should be considered when designing a system for upgrade with an Intel OverDrive processor.

11.1 Intel OverDrive® Processor Detection

The component identifier and the stepping/revision identifier for the Intel OverDrive processors is readable in the DH and DL registers, respectively, immediately after RESET. The value loaded into each register is defined in Table 11-1. The "x" value defines the device stepping.

Table 11-1. CPU ID Values

Processor	DH Reg.	DL Reg.
Intel486DX	04h	0xh, 1xh
Intel486SX	04h	2xh
IntelSX2 OverDrive	04h	5xh
IntelDX2 OverDrive	04h	3xh
IntelDX4 OverDrive	14h, 04h	8xh
Pentium OverDrive	15h	3xh

As it is difficult to differentiate between Intel486 DX processor and some of the Intel OverDrive processors in software, it is recommended that the BIOS save the contents of the DX register immediately after RESET. This will allow the information to be used later, if required, to identify an Intel OverDrive processor in the system.

Alternately, for those OverDrive processors supporting it, the CPUID instruction can be used to identify the processor. Refer to the Intel486 Microprocessor Data Book for additional information on the CPUID instruction and its use.

NOTE:
Initialization routines for IntelSX2 OverDrive processor and Intel486 SX processor-based systems should check for the presence of a floating point unit and set the CR0 register accordingly (refer to the Intel486 SX Microprocessor Data Book for specific details). In addition, the BIOS should check for the presence of the 16 KByte cache in the IntelDX4 OverDrive processor.

11.2 Timing Dependent Loops

The Intel OverDrive processors execute instructions at two times (for the IntelSX2 and IntelDX2 OverDrive processors) or three times (for the IntelDX4 OverDrive processor) the frequency of the input clock. Thus, software (or instruction based) timing loops will execute faster on the Intel OverDrive processor than on the Intel486 DX or Intel486 SX processor (at the same input clock frequency). Instructions such as NOP, LOOP, and JMP $+2, have

been used by BIOS to implement timing loops that are required, for example, to enforce recovery time between consecutive accesses for I/O devices. These instruction based timing loop implementations may require modification for systems intended to be upgradable with the Intel OverDrive processors.

In order to avoid any incompatibilities, it is recommended that timing requirements be implemented in hardware rather than in software. This provides transparency and also does not require any change in BIOS or I/O device drivers in the future when moving to higher processor clock speeds. As an example, a timing routine may be implemented as follows: The software performs a dummy I/O instruction to an unused I/O port. The hardware for the bus controller logic recognizes this I/O instruction and delays the termination of the I/O cycle to the processor by keeping RDY# or BRDY# deasserted for the appropriate amount of time.

11.3 Test Register Access on the Pentium® OverDrive® Processor

The IntelDX2 processor has test registers which allow OEM's to test the functionality of different areas of the component. These test registers are accessed on the Intel486 processor family using the "MOV reg, TRx and MOV TRx, reg" instructions. These instructions are not available on the Pentium OverDrive processor. Any attempt to execute them will cause a invalid opcode exception. The Pentium OverDrive processor uses the Model Specific Registers (MSR's) to implement on chip testing. These MSR's are accessed using the RDMSR and WRMSR instructions. BIOS must recognize this fundamental difference between the Pentium OverDrive processor and the IntelDX2 processor and act accordingly.

12.0 Pentium® OverDrive® PROCESSOR TESTABILITY

12.1 Introduction

This section describes the features which are included in the Pentium OverDrive processor for the purpose of enhancing the testability of the part. The capabilities of the Intel486 processor test hooks are included in the processor, however they are implemented differently. In addition, new test features were added to assure timely testing and production of a system product. All features described here are also present in the Pentium processor.

Internal component testing through the Built In Self Test (BIST) feature of the Pentium OverDrive processor provides 100% single stuck at fault coverage of the microcode ROM and large PLAs. Some testing of the instruction cache, data cache, Translation Lookaside Buffers (TLBs), and Branch Target Buffer (BTB) is also performed. In addition, the constant ROMs are checked.

The production version of the Pentium OverDrive processor will not include the boundary scan or testability pins.

Several test registers are also included in the Pentium OverDrive processor to simplify access to all on-chip caches and TLBs. These test registers on the processor are not compatible with the definitions for the test registers on the IntelDX2 processor, and will be provided by Intel at a later date. For the latest information on these test registers, please contact Intel.

The following list summarizes the Pentium OverDrive processor testability features:

- Built In Self Test
- Cache and TLB Test Registers
- Tristate Test Mode

Table 12-1. Pentium® OverDrive® Processor RESET Modes

RESET	INIT	Type of Reset	Effect on I/D Caches	Effect on FP Registers	Effect on SMM Base Register
0	0	None	Not Applicable	Not Applicable	Not Applicable
0	1	Warm Reset	None	None	None
1	X	Cold Reset (w/ BIST)	Invalidated	Initialized	Invalidated
1	X	Cold Reset (w/o BIST)	Invalidated	Undefined	Invalidated

12.2 Pentium® OverDrive® Processor Reset Pins/BIST Initiation

Two pins, **RESET** and **INIT,** are used to reset the Pentium OverDrive processor in different manners. The following table shows the different types of resets that can be initiated using these pins.

Toggling either the **RESET** pin or the **INIT** pin individually forces the Pentium OverDrive processor to begin execution at address 0FFFFFFF0h. The internal instruction cache and data cache are invalidated when **RESET** is asserted (modified lines in the data cache are NOT written back). The instruction cache and data cache are not altered when the INIT pin is asserted without **RESET.** In neither case are the floating point registers altered. In both cases, the BTB, the segment descriptor cache and both TLBs are all invalidated.

Reset with self test is initiated by holding the **AHOLD** pin HIGH for 2 clocks before and 2 clocks after **RESET** is driven from HIGH to LOW. The instruction cache and data cache are invalidated and the floating point registers are initialized. The processor begins execution at address 0xFFFFFFF0h. The BTB, the segment descriptor cache and both TLBs are all invalidated before execution begins.

At the conclusion of reset, with or without self test, the DX register will contain a component identifier. The upper byte will contain 15h and the lower byte will contain a stepping identifier.

Table 12-2 defines the processor state after **RESET, INIT** and **RESET** with BIST (built in self test).

2

Table 12-2. Register State after RESET, INIT and BIST
(Register States are given in Hexadecimal Format)

Storage Element	RESET (no BIST)	RESET (BIST)	INIT
EAX	0	0 if pass	0
EDX	1530 + stepping	1530 + stepping	1530 + stepping
ECX, EBX, ESP EBP, ESI, EDI	0	0	0
EFLAGS	2	2	2
EIP	0FFF0	0FFF0	0FFF0
CS	selector = F000	selector = F000	selector = F000
	base = FFFF0000	base = FFFF0000	base = FFFF0000
	limit = FFFF	limit = FFFF	limit = FFFF
DS,ES,FE,GS,SS	selector = 0	selector = 0	selector = 0
	base = 0	base = 0	base = 0
	limit = FFFF	limit = FFFF	limit = FFFF

Table 12-2. Register State after RESET, INIT and BIST
(Register States are given in Hexadecimal Format)

Storage Element	RESET (no BIST)	RESET (BIST)	INIT
IDTR	base = 0	base = 0	base = 0
	limit = FFF	limit = FFF	limit = FFF
GDTR,LDTR,TR	undefined	undefined	undefined
CR0	60000010	60000010	Note 1
CR2,3,4	0	0	0
DR3–0	0	0	0
DR6	FFFF0FF0	FFFF0FF0	FFFF0FF0
DR7	00000400	00000400	00000400
Time Stamp Counter	0	0	UNCHANGED
Control and Event Select	0	0	UNCHANGED
TR12	0	0	UNCHANGED
All Other MSR's	undefined	undefined	UNCHANGED
CW	undefined	37F	UNCHANGED
SW	undefined	0	UNCHANGED
TW	undefined	FFFF	UNCHANGED
FIP,FEA,FCS, FDS,FOP	undefined	0	UNCHANGED
FSTACK	undefined	undefined	UNCHANGED
Data and Code Cache	invalid	invalid	UNCHANGED
Code Cache TLB, Data Cache TLB, BTB, SDC	invalid	invalid	invalid

NOTE:
1. CD and NW are unchanged, bit 4 is set to 1, all other bits are cleared.
   State of output pins after RESET:
   High: **LOCK#, ADS#, PCHK#, HIT#, HITM#, FERR#, SMIACT#**
   Low: **HLDA, BREQ**
   High Impedance: **D31–D0**
   Undefined: **A31–A3, BE3#–BE0#, W/R#, M/IO#, D/C#, PCD, PWT, CACHE#**

12.3 Built In Self Test (BIST)

Self test is initiated by holding the **AHOLD** pin HIGH for the clock before **RESET** changes from HIGH to LOW. If asserted asynchronously, AHOLD must be asserted two clocks before and two clocks after **RESET** to guarantee recognition.

No bus cycles are run by the Pentium OverDrive processor during self test. The duration of self test is approximately 2^{19} internal clocks. Approximately 70% of the devices in the processor are tested by BIST.

The Pentium OverDrive processor BIST consists of two parts: hardware self test and microcode self test.

During the hardware portion of BIST, the microcode and all large PLAs are tested. All possible input combinations of the microcode ROM and PLAs are tested.

The constant ROMs, BTB, TLBs, and all caches are tested by the microcode portion of BIST. The array tests (caches, TLBs, and BTB) have two passes. On the first pass, data patterns are written to arrays, read back and checked for mismatches. The second pass writes the complement of the initial data pattern, reads it back, and checks for mismatches. The constant ROMs are tested by using the microcode to add various constants and check the result against a stored value.

Upon completion of BIST, the cumulative result of all tests are stored in the EAX register. If EAX contains 0x0h, then all checks passed; any non-zero result indicates a faulty unit.

During BIST, **EADS#** should not be used to perform snoops, otherwise false **HITM#** indications, with no corresponding write back cycles, can occur.

12.4 Tri-State Test Mode

The Pentium OverDrive processor provides the ability to float all its outputs and bi-directional pins. This includes pins that are floated during bus hold as well as some pins that are not normally floated during normal operation. When the Pentium OverDrive processor is in tri-state test mode, external testing can be used to test on board connections.

The tri-state test mode is invoked by driving **FLUSH#** low for 2 clocks before and 2 clocks after **RESET** going low. The outputs are guaranteed to tri-state no later than 10 clocks after **RESET** goes low. The processor will remain in tri-state test mode until the next **RESET.**

12.5 Cache, TLB and BTB Test Registers

The Pentium OverDrive processor contains several test registers. The purpose of the test registers is to provide direct access to the processor caches, TLBs, and BTB, so user programs can easily exercise these structures. Because the architecture of the caches, TLBs, and BTB is different, a different set of test registers (along with a different test mechanism) is required for each. Most test registers can be shared between the code and data caches.

Since much of the testability hardware is used for other purposes during normal operation of the Pentium OverDrive processor, some restrictions may exist on what software may do while testability operations are being run.

Please contact Intel for more information on the Pentium OverDrive processor Cache, TLB, and BTB Test Registers.

12.6 Fan Protection Mechanism and THermal ERRor Bit

The Pentium OverDrive processor employs an active fan/heatsink unit to assist in cooling the processor. Another integral part of this cooling solution is the ability for software to poll the status of the fan to determine if the fan has fallen to a speed that is unacceptable to cool the processor. Should the fan fall into a speed range that is too slow, a control register will record the event. (For more information, please contact Intel.)

13.0 ELECTRICAL DATA

The following sections describe recommended electrical connections for the Intel OverDrive processor, and its electrical specifications.

13.1 Power and Grounding

13.1.1 POWER CONNECTIONS

Power and ground connections must be made to all external V_{CC} and GND pins of the Intel OverDrive processor. On the circuit board, all V_{CC} pins must be connected on a V_{CC} plane. All V_{SS} pins must be likewise connected on a GND plane.

13.1.2 Intel OverDrive® PROCESSOR DECOUPLING CAPACITORS

Because of the fast internal switching speeds of the Intel OverDrive processor, it is important that the Intel OverDrive processor use a liberal amount of decoupling capacitors. For proper V_{CC} transient response, Intel recommends that a system design employ at least 4 each of 47 μF bulk capacitors and 9 each of 0.1 μF and 0.01 μF capacitors. It is recommended that surface mount capacitors be used for decoupling the Intel OverDrive processor. This style of capacitor introduces less inductance than leaded capacitors, so fewer are needed to achieve the same results. The capacitors should be added around the Intel OverDrive processor in a manner that ensures they are evenly spread about and close to the processor location.

13.1.3 OTHER CONNECTION RECOMMENDATIONS

N.C. pins should always remain unconnected.

For reliable operation, always connect unused inputs to an appropriate signal level. Active LOW inputs should be connected to V_{CC} through a pullup resistor. Pullups in the range of 20 KΩ are recommended. Active HIGH inputs should be connected to GND.

13.2 Maximum Ratings

Table 13-1 lists the absolute maximum ratings for each of the OverDrive processors. This table is a stress rating only, and functional operation at the maximums is not guaranteed. Functional operating conditions are given in Section 13.3, DC Specifications, and Section 13.4, AC Specifications.

Extended exposure to the maximum ratings may affect device reliability. Furthermore, although the Intel OverDrive processors contains protective circuitry to resist damage from static electric discharge, always take precautions to avoid high static voltages or electric fields.

13.3 DC Specifications

The DC specifications for each of the OverDrive processors are contained in the tables in Sections 13.3.1, 13.3.2, and 13.3.3. For additional information, refer to the appropriate Intel microprocessor handbook.

Table 13-1. Absolute Maximum Ratings

	IntelDX2™ OverDrive®	IntelDX4™ OverDrive®	Pentium® OverDrive®
Case Temperature under Bias	−65°C to +110°C	−30°C to +110°C	−30°C to +110°C
Fan/Heat sink Temperature under Bias	N/A	N/A	−5°C to +60°C
Storage Temperature	−65°C to +150°C	−30°C to +125°C	−30°C to +125°C
Fan/Heat sink Storage Temperature	N/A	N/A	−40°C to +70°C
Voltage on any Pin with Respect to Ground	−0.5V to (V_{CC} + 0.5V)	−0.5V to (V_{CC} + 0.5V)	−0.5V to (V_{CC} + 0.5V)
Supply Voltage with Respect to V_{SS}	−0.5V to +6.5V	−0.5V to +6.5V	−0.5V to +6.5V

13.3.1 IntelDX2™ OverDrive® PROCESSOR DC SPECIFICATIONS

Table 13-2 details the DC Specifications of the IntelDX2 OverDrive processor.

Table 13-2. DC Specifications for the IntelDX2™ OverDrive® Processor

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL}	Input Low Voltage	−0.3	+0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC} +0.3	V	
V_{OL}	Output Low Voltage		0.45	V	(Note 2)
V_{OH}	Output High Voltage	2.4		V	(Note 3)
I_{CC}	Power Supply Current CLK = 33 MHz CLK = 25 MHz		 1200 950	 mA	 (Note 4)
I_{LI}	Input Leakage Current		±15	μA	(Note 5)
I_{IH}	Input Leakage Current		200	μA	(Note 6)
I_{IL}	Input Leakage Current		−400	μA	(Note 7)
I_{LO}	Output Leakage Current		±15	μA	
C_{IN}	Input Capacitance		13	pF	F_C = 1 MHz(8)
C_O	I/O or Output Capacitance		17	pF	F_C = 1 MHz(8)
C_{CLK}	CLK Capacitance		15	pF	F_C = 1 MHz(8)

NOTES:
1. The function operating temperature range is:
   OverDrive processor—25 MHz, T_{sink} = 0°C to +95°C
   OverDrive processor—33 MHz, T_{sink} = 0°C to +95°C
2. This parameter is measured at:
   Address, Data, BEn 4.0 mA
   Definition, Control 5.0 mA
3. This parameter is measured at:
   Address, Data, BEn −1.0 mA
   Definition, Control −0.9 mA
4. Typical supply current:
   775 mA @ CLK = 25 MHz
   975 mA @ CLK = 33 MHz
5. This parameter is for inputs without internal pullups or pulldowns and $0 \leq V_{IN} \leq V_{CC}$.
6. This parameter is for inputs with internal pulldowns and V_{IH} = 2.4V.
7. This parameter is for inputs with internal pullups and V_{IL} = 0.45V.
8. Not 100% tested.

13.3.2 IntelDX4™ OverDrive® PROCESSOR DC SPECIFICATIONS

Table 13-3 details the DC Specifications of the IntelDX4 OverDrive processor.

Table 13-3. DC Specifications for the IntelDX4™ OverDrive® Processor

Functional operating range: V_{CC} = 5V + 5%, T_{SINK} = 0°C to +95°C.

Symbol	Parameter	Min	Max	Unit	Notes
V_{IL}	Input Low Voltage	−0.3	+0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC} + 0.3	V	
V_{OL}	Output Low Voltage		0.45	V	(Note 1)
V_{OH}	Output High Voltage	2.4		V	I_{OH} = −2 mA
I_{CC}	Power Supply Current CLK = 25/75 MHz CLK = 33/100 MHz		 1200 1550	 mA	(Note 2)
I_{CC} Stop Grant	Power Supply Current in Stop Grant State CLK = 25/75 MHz CLK = 33/100 MHz		 85 110	 mA mA	(Note 3)
I_{CC} Stop Clock	Power Supply Current in Stop Clock State		20	mA	(Note 4)
I_{LI}	Input Leakage Current		±15	μA	(Note 5)
I_{IH}	Input Leakage Current		200	μA	(Note 6)
I_{IL}	Input Leakage Current		−400	μA	(Note 7)
I_{LO}	Output Leakage Current		±15	μA	
C_{IN}	Input Capacitance		13	pF	F_C = MHz(8)
C_O	I/O or Output Capacitance		17	pF	F_C = MHz(8)
C_{CLK}	CLK Capacitance		15	pF	F_C = MHz(8)

NOTES:
1. This parameter is measured at:
   4.0 mA: Address, Data, BEn
   5.0 mA: Definition, Control
2. The maximum and typical values shown here are design estimates. Typical supply current:
   I_{CC} = 835 mA @ CLK = 25 MHz
   I_{CC} = 1085 mA @ CLK = 33 MHz
3. The I_{CC} Stop Grant specification refers to the I_{CC} value once the IntelDX4 OverDrive processor enters the Stop Grant or Halt Auto Powerdown State.
4. The I_{CC} Stop Clock specification refers to the I_{CC} value once the IntelDX4 OverDrive processor enters the Stop Clock State. V_{IH} and V_{IL} levels must be V_{CC} and 0V, respectively, in order to meet the I_{CC} Stop Clock specification.
5. This parameter is for inputs without pullups or pulldowns and $0 \le V_{IN} \le V_{CC}$.
6. This parameter is for inputs with pulldowns and V_{IH} = 2.4V.
7. This parameter is for inputs with pullups and V_{IL} = 0.45V.
8. Not 100% tested.

13.3.3 Pentium® OverDrive® PROCESSOR DC SPECIFICATIONS

Table 13-4 provides the DC operating conditions for the Pentium OverDrive processor.

Functional operating range: V_{CC} = 5V +5%; $T_{A(IN)}$ = 10°C to +55°C @33 MHz and 25 MHz.

Table 13-4. DC Specifications for the Pentium® OverDrive® Processor

Symbol	Parameter	Min	Max	Units	Notes
V_{IL}	Input Low Voltage	−0.3	+0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC} + 0.3	V	
V_{OL}	Output Low Voltage		0.45	V	(Note 1)
V_{OH}	Output High Voltage	2.4		V	(Note 2)
I_{CC}	Power Supply Current CLK = 25 MHz CLK = 33 MHz		 2200 2600	 mA	
I_{LI}	Input Leakage Current		±15	μA	(Note 3)
I_{IH}	Input Leakage Current		200	μA	(Note 4)
I_{IL}	Input Leakage Current		−400	μA	(Note 5)
I_{LO}	Output Leakage Current		±15	μA	
C_{IN}	Input Capacitance		13	pF	F_c = 1 MHz (6)
C_O	I/O or Output Capacitance		17	pF	F_c = 1 MHz (6)
C_{CLK}	CLK Capacitance		15	pF	F_c = 1 MHz (6)

NOTES:
1. This parameter is measured at:
   Address, Data, BEn 4.0 mA
   Definition, Control 5.0 mA
2. This parameter is measured at:
   Address, Data, BEn −1.0 mA
   Definition, Control −0.9 mA
3. This parameter is for inputs without pullups or pulldowns and $0 < V_{IN} < V_{CC}$.
4. This parameter is for inputs with pulldowns and V_{IH} = 2.4V.
5. This parameter is for inputs with pullups and V_{IL} = 0.45V.
6. Not 100% tested.

13.4 AC Specifications

The AC specifications for each of the OverDrive processors are contained in the tables in Sections 13.4.1, 13.4.2 and 13.4.3. These specifications consist of output delays, input setup requirements and input hold requirements. All AC specifications are relative to the rising edge of the CLK signal.

AC specification measurements are defined by Figures 13-1 through 13-6. All timings are referenced to 1.5V, unless otherwise specified. Inputs must be driven to the voltage levels indicated by Figure 13-3 when AC specifications are measured. Intel OverDrive processor output delays are specified with minimum and maximum limits, measured as shown. The minimum Intel OverDrive processor delay times are hold times provided to external circuitry. Intel OverDrive processor input setup and hold times are specified as minimums, defining the smallest acceptable sampling window. Within the sampling window, a synchronous signal must be stable for correct Intel OverDrive processor operation.

Table 13-5 defines the AC timing specifications for a 33 MHz system. Table 13-7 defines the AC timing specifications for a 25 MHz system. Table 13-8 defines the AC timing specifications for a 20 MHz system. Table 13-8 defines the AC timing specifications for a 16 MHz system.

Each Intel OverDrive processor meets the AC specifications for the processor it is upgrading. For example, a 100 MHz IntelDX4 OverDrive processor meets the system AC timing specifications for the 33 MHz processor it is upgrading.

Refer to Sections 13.4.1 through 13.4.3 for any timing differences from those specified in the following tables.

For additional information, refer to the appropriate Intel microprocessor handbook.

13.4.1 IntelDX2™ OverDrive® PROCESSOR AC SPECIFICATIONS

The IntelDX2 OverDrive processor can be placed into an existing 16 MHz, 20 MHz, 25 MHz or 33 MHz Intel486 system, doubling the internal processor speed to 32 MHz, 40 MHz, 50 MHz or 66 MHz, respectively.

Tables 13-5 through 13-8 contain the AC timing specifications for the processors in those systems.

13.4.2 IntelDX4™ OverDrive® PROCESSOR AC SPECIFICATIONS

The IntelDX4 OverDrive processor can be placed into an existing 16 MHz, 20 MHz, 25 MHz or 33 MHz Intel486 system, tripling the internal processor speed to 48 MHz, 60 MHz, 75 MHz or 100 MHz, respectively.

Tables 13-5 through 13-8 contain the AC timing specifications for the processors in those systems.

Table 13-5. 33 MHz Intel Processor Characteristics(1)

V_{CC} = 5V ±5%; T_{sink} = See Note 6; C_l = 50 pF unless otherwise specified(3)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	8	33	MHz		1X Clock Driven to OverDrive processor
t_1	CLK Period	30	125	ns	13-1	
t_{1a}	CLK Period Stability		0.1%	Δ		Adjacent Clocks
t_2	CLK High Time	11		ns	13-1	at 2V
t_3	CLK Low Time	11		ns	13-1	at 0.8V
t_4	CLK Fall Time		3	ns	13-1	2V to 0.8V
t_5	CLK Rise Time		3	ns	13-1	0.8V to 2V
t_6	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK#, SMIACT#, FERR#, BREQ, HLDA Valid Delay	3	14	ns	13-5	(Note 4)
t_7	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK# Float Delay		20	ns	13-6	(Note 2)
t_8	PCHK# Valid Delay	3	22	ns	13-4	(Note 4)
t_{8a}	BLAST#, PLOCK# Valid Delay	3	20	ns	13-5	(Note 4)
t_9	BLAST#, PLOCK# Float Delay		20	ns	13-6	(Note 2)
t_{10}	D0–D31, DP0–3 Write Data Valid Delay	3	18	ns	13-5	(Note 4)
t_{11}	D0–D31, DP0–3 Write Data Float Delay		20	ns	13-6	(Note 2)
t_{12}	EADS# Setup Time	5		ns	13-2	
t_{13}	EADS# Hold Time	3		ns	13-2	
t_{14}	KEN#, BS16#, BS8# Setup Time	5		ns	13-2	
t_{15}	KEN#, BS16#, BS8# Hold Time	3		ns	13-2	
t_{16}	RDY#, BRDY# Setup Time	5		ns	13-3	
t_{17}	RDY#, BRDY# Hold Time	3		ns	13-3	
t_{18}	HOLD, AHOLD Setup Time	6		ns	13-2	
t_{18a}	BOFF# Setup Time	7		ns	13-2	
t_{19}	HOLD, AHOLD, BOFF# Hold Time	3		ns	13-2	
t_{20}	RESET, FLUSH#, A20M#, NMI, INTR, SMI#, STPCLK#, SRESET, IGNNE# Setup Time	5		ns	13-2	
t_{21}	RESET, FLUSH#, A20M#, NMI, INTR, SMI#, STPCLK#, SRESET, IGNNE# Hold Time	3		ns	13-2	
t_{22}	D0–D31, DP0–3, A4–A31 Read Setup Time	5		ns	13-2, 13-3	
t_{23}	D0–D31, DP0–3, A4–A31 Read Hold Time	3		ns	13-2, 13-3	

NOTES:
1. To be used for 66 MHz IntelDX2 and 100 MHz IntelDX4 OverDrive processors.
2. Not 100% tested. Guaranteed by design characterization.
3. All timing specifications assume C_L = 50 pF.
4. The minimum Intel OverDrive processor output valid delays are hold times provided to external circuitry.
5. A reset pulse width of 15 CLK cycles is required for warm resets. Power-up resets require RESET to be asserted for at least 1 ms after V_{CC} and CLK are stable.
6. T_{SINK} temperatures are:
   IntelDX2 OverDrive processor: 0°C to +95°C
   IntelDX4 OverDrive processor: 0°C to +95°C

Table 13-6. 25 MHz Intel Processor Characteristics(1)

V_{CC} = 5V ±5%; T_{sink} = See Note 6; C_l = 50 pF unless otherwise specified(3)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	8	25	MHz		1X Clock Driven to OverDrive Processor
t_1	CLK Period	40	125	ns	13-1	
t_{1a}	CLK Period Stability		0.1%	Δ		Adjacent Clocks
t_2	CLK High Time	14		ns	13-1	at 2V
t_3	CLK Low Time	14		ns	13-1	at 0.8V
t_4	CLK Fall Time		4	ns	13-1	2V to 0.8V
t_5	CLK Rise Time		4	ns	13-1	0.8V to 2V
t_6	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK#, FERR#, BREQ, HLDA, SMIACT#, Valid Delay	3	19	ns	13-5	(Note 4)
t_7	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK# Float Delay		28	ns	13-6	(Note 2)
t_8	PCHK# Valid Delay	3	24	ns	13-4	(Note 4)
t_{8a}	BLAST#, PLOCK# Valid Delay	3	24	ns	13-5	(Note 4)
t_9	BLAST#, PLOCK# Float Delay		28	ns	13-6	(Note 2)
t_{10}	D0–D31, DP0–3 Write Data Valid Delay	3	20	ns	13-5	(Note 4)
t_{11}	D0–D31, DP0–3 Write Data Float Delay		28	ns	13-6	(Note 2)
t_{12}	EADS# Setup Time	8		ns	13-2	
t_{13}	EADS# Hold Time	3		ns	13-2	
t_{14}	KEN#, BS16#, BS8# Setup Time	8		ns	13-2	
t_{15}	KEN#, BS16#, BS8# Hold Time	3		ns	13-2	
t_{16}	RDY#, BRDY# Setup Time	8		ns	13-3	
t_{17}	RDY#, BRDY# Hold Time	3		ns	13-3	
t_{18}	HOLD, AHOLD, BOFF# Setup Time	8		ns	13-2	
t_{19}	HOLD, AHOLD, BOFF# Hold Time	3		ns	13-2	
t_{20}	RESET, FLUSH#, A20M#, NMI, SMI#, STPCLK#, SRESET, INTR, IGNNE# Setup Time	8		ns	13-2	
t_{21}	RESET, FLUSH#, A20M#, NMI, SMI#, STPCLK#, SRESET, INTR, IGNNE# Hold Time	3		ns	13-2	
t_{22}	D0–D31, DP0–3, A4–A31 Read Setup Time	5		ns	13-2, 13-3	
t_{23}	D0–D31, DP0–3, A4–A31 Read Hold Time	3		ns	13-2, 13-3	

NOTES:
1. To be used for 50 MHz or 60 MHz IntelDX2 and 75 MHz or 100 MHz IntelDX4 OverDrive processors.
2. Not 100% tested. Guaranteed by design characterization.
3. All timing specifications assume C_L = 50 pF.
4. The minimum Intel OverDrive processor output valid delays are hold times provided to external circuitry.
5. A reset pulse width of 15 CLK cycles is required for warm resets. Power-up resets require RESET to be asserted for at least 1 ms after V_{CC} and CLK are stable.
6. T_{SINK} temperatures are:
   IntelDX2 OverDrive processor: 0°C to +95°C
   IntelDX4 OverDrive processor: 0°C to +95°C

Table 13-7. 20 MHz Intel Processor Characteristics(1)

V_{CC} = 5V ±5%; T_{SINK} = See Note 6; C_l = 50 pF unless otherwise specified(3)

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	8	20	MHz		1X Clock Driven to OverDrive Processor
t_1	CLK Period	50	125	ns	13-1	
t_{1a}	CLK Period Stability		0.1%	Δ		Adjacent Clocks
t_2	CLK High Time	16		ns	13-1	at 2V
t_3	CLK Low Time	16		ns	13-1	at 0.8V
t_4	CLK Fall Time		6	ns	13-1	2V to 0.8V
t_5	CLK Rise Time		6	ns	13-1	0.8V to 2V
t_6	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK#, FERR#, BREQ, HLDA, SMIACT# Valid Delay	3	23	ns	13-5	(Note 4)
t_7	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK# Float Delay		37	ns	13-6	(Note 2)
t_8	PCHK# Valid Delay	3	28	ns	13-4	(Note 4)
t_{8a}	BLAST#, PLOCK# Valid Delay	3	28	ns	13-5	(Note 4)
t_9	BLAST#, PLOCK# Float Delay		37	ns	13-6	(Note 2)
t_{10}	D0–D31, DP0–3 Write Data Valid Delay	3	26	ns	13-5	(Note 4)
t_{11}	D0–D31, DP0–3 Write Data Float Delay		37	ns	13-6	(Note 2)
t_{12}	EADS# Setup Time	10		ns	13-2	
t_{13}	EADS# Hold Time	3		ns	13-2	
t_{14}	KEN#, BS16#, BS8# Setup Time	10		ns	13-2	
t_{15}	KEN#, BS16#, BS8# Hold Time	3		ns	13-2	
t_{16}	RDY#, BRDY# Setup Time	10		ns	13-3	
t_{17}	RDY#, BRDY# Hold Time	3		ns	13-3	
t_{18}	HOLD, AHOLD, Setup Time	12		ns	13-2	
t_{19}	HOLD, AHOLD, BOFF# Hold Time	3		ns	13-2	
t_{20}	RESET, FLUSH#, A20M#, NMI, SMI#, STPCLK#, SRESET, INTR, IGNNE# Setup Time	12		ns	13-2	(Note 5)
t_{21}	RESET, FLUSH#, A20M#, NMI, SMI#, STPCLK#, SRESET, INTR, IGNNE# Hold Time	3		ns	13-2	(Note 5)
t_{22}	D0–D31, DP0–3, A4–A31 Read Setup Time	6		ns	13-2, 13-3	
t_{23}	D0–D31, DP0–3, A4–A31 Read Hold Time	3		ns	13-2, 13-3	

NOTES:
1. To be used 50 MHz or 60 MHz IntelDX2 and 75 MHz or 100 MHz IntelDX4 OverDrive processors.
2. Not 100% tested. Guaranteed by design characterization.
3. All timing specifications assume C_L = 50 pF.
4. The minimum Intel OverDrive processor output valid delays are hold times provided to external circuitry.
5. A reset pulse width of 15 CLK cycles is required for warm resets. Power-up resets require RESET to be asserted for at least 1 ms after V_{CC} and CLK are stable.
6. T_{SINK} temperatures are:
   IntelDX2 OverDrive processor: 0°C to +95°C
   IntelDX4 OverDrive processor: 0°C to +95°C

Table 13-8. 16 MHz Intel Processor Characteristics(1)

V_{CC} = 5V ±5%; T_{SINK} = See Note 6; C_l = 50 pF unless otherwise specified

Symbol	Parameter	Min	Max	Unit	Figure	Notes
	Frequency	8	16	MHz		1X Clock Driven to OverDrive Processor
t_1	CLK Period	62.5	125	ns	13-1	
t_{1a}	CLK Period Stability		0.1%	Δ		Adjacent Clocks
t_2	CLK High Time	20		ns	13-1	at 2V
t_3	CLK Low Time	20		ns	13-1	at 0.8V
t_4	CLK Fall Time		8	ns	13-1	2V to 0.8V
t_5	CLK Rise Time		8	ns	13-1	0.8V to 2V
t_6	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK#, FERR#, BREQ, HLDA, SMIACT# Valid Delay	3	26	ns	13-5	(Note 4)
t_7	A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK# Float Delay		42	ns	13-6	(Note 2)
t_8	PCHK# Valid Delay	3	35	ns	13-4	(Note 4)
t_{8a}	BLAST#, PLOCK# Valid Delay	3	35	ns	13-5	(Note 4)
t_9	BLAST#, PLOCK# Float Delay		42	ns	13-6	(Note 2)
t_{10}	D0–D31, DP0–3 Write Data Valid Delay	3	30	ns	13-5	(Note 4)
t_{11}	D0–D31, DP0–3 Write Data Float Delay		42	ns	13-6	(Note 2)
t_{12}	EADS# Setup Time	12		ns	13-2	
t_{13}	EADS# Hold Time	4		ns	13-2	
t_{14}	KEN#, BS16#, BS8# Setup Time	12		ns	13-2	
t_{15}	KEN#, BS16#, BS8# Hold Time	4		ns	13-2	
t_{16}	RDY#, BRDY# Setup Time	12		ns	13-3	
t_{17}	RDY#, BRDY# Hold Time	4		ns	13-3	
t_{18}	HOLD, AHOLD, BOFF# Setup Time	12		ns	13-2	
t_{19}	HOLD, AHOLD, BOFF# Hold Time	4		ns	13-2	
t_{20}	RESET, FLUSH#, A20M#, NMI, SMI#, STPCLK#, SRESET, INTR, IGNNE# Setup Time	14		ns	13-2	(Note 5)
t_{21}	RESET, FLUSH#, A20M#, NMI, SMI#, STPCLK#, SRESET, INTR, IGNNE# Hold Time	4		ns	13-2	(Note 5)
t_{22}	D0–D31, DP0–3, A4–A31 Read Setup Time	10		ns	13-2, 13-3	
t_{23}	D0–D31, DP0–3, A4–A31 Read Hold Time	4		ns	13-2, 13-3	

NOTES:
1. To be used for 50 MHz or 60 MHz IntelDX2 and 75 MHz or 100 MHz IntelDX4 OverDrive processors.
2. Not 100% tested. Guaranteed by design characterization.
3. All timing specifications assume C_L = 50 pF.
4. The minimum Intel OverDrive processor output valid delays are hold times provided to external circuitry.
5. A reset pulse width of 15 CLK cycles is required for warm resets. Power-up resets require RESET to be asserted for at least 1 ms after V_{CC} and CLK are stable.
6. T_{SINK} temperatures are:
   IntelDX2 OverDrive processor: 0°C to +95°C
   IntelDX4 OverDrive processor: 0°C to +95°C

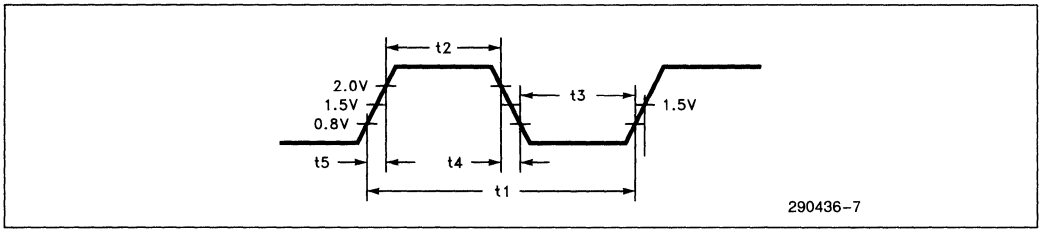


Figure 13-1. CLK Waveforms

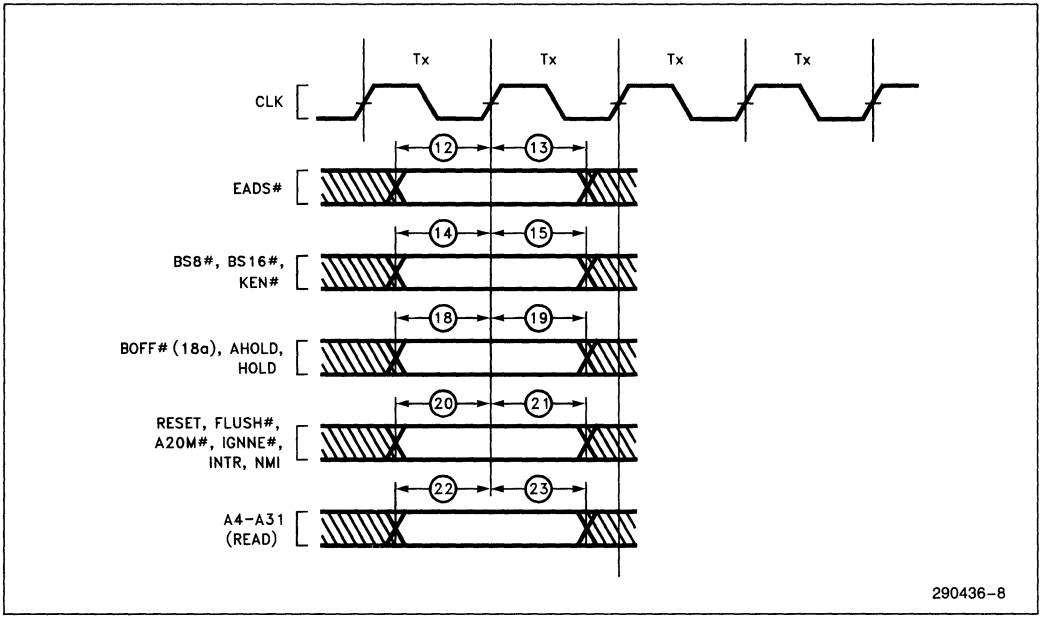


Figure 13-2. Input Setup and Hold Timing

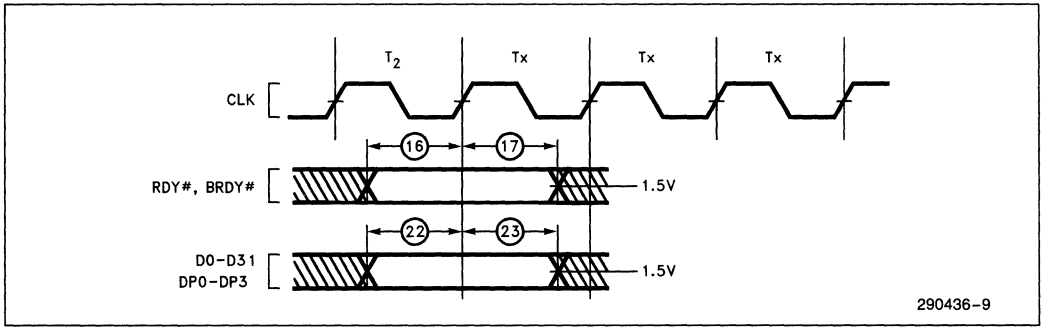


Figure 13-3. Input Setup and Hold Timing

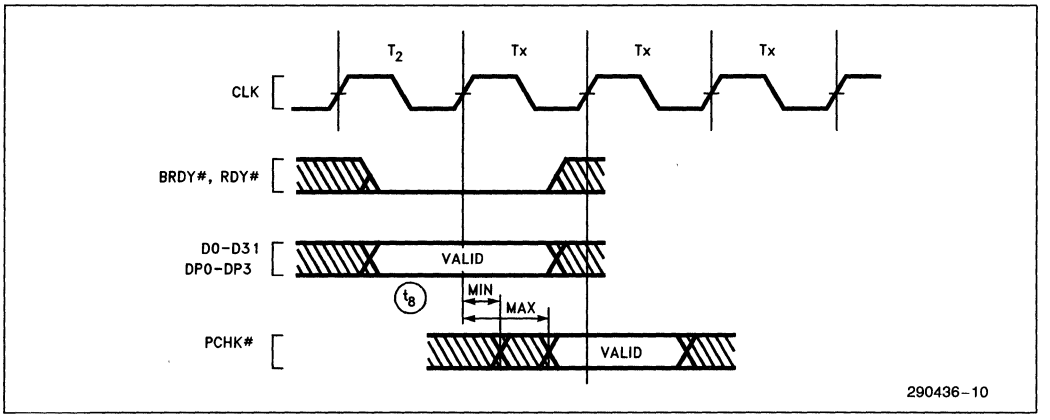


Figure 13-4. PCHK# Valid Delay Timing

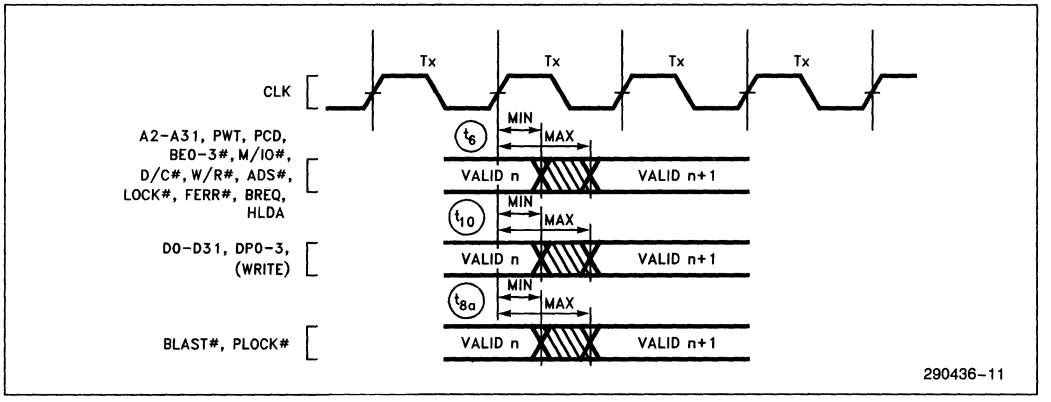


Figure 13-5. Output Valid Delay Timing

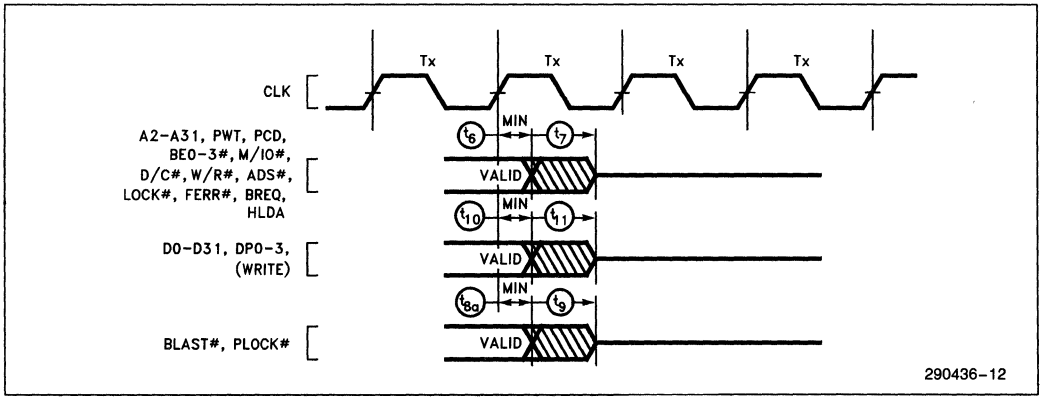


Figure 13-6. Maximum Float Delay Timing

13.4.3 Pentium® OverDrive® PROCESSOR AC SPECIFICATIONS

Tables 13-9 and 13-10 provide the AC specifications for the Pentium OverDrive processor at external clock frequencies of 25 MHz and 33 MHz respectively. They consist of output delays, input setup requirements, and input hold requirements. All AC specifications are relative to the rising edge of the input system clock unless otherwise specified. Internal core frequencies will be a multiple of the system bus frequency.

13.4.3.1 V_{CC} Transient Specification

Due to the on-board voltage regulator, the V_{CC} of the Pentium OverDrive processor is allowed to exceed the DC Voltage specifications (V_{CC} = 5V + 5%) when the processor creates a large current transient, as would be the case in a full operation to Autohalt transition (2.5A to 200 mA I_{CC} change). The width of the pulse that exceeds 5V + 5% should be no wider than 1ms, and can not exceed 5.5V. V_{CC} is not allowed to go below the DC specification of 5V − 5% at any time. This specification applies to the Pentium OverDrive processor only and can not be applied to any other Intel processors. Figure 13-7 shows an example of the V_{CC} transient specification.

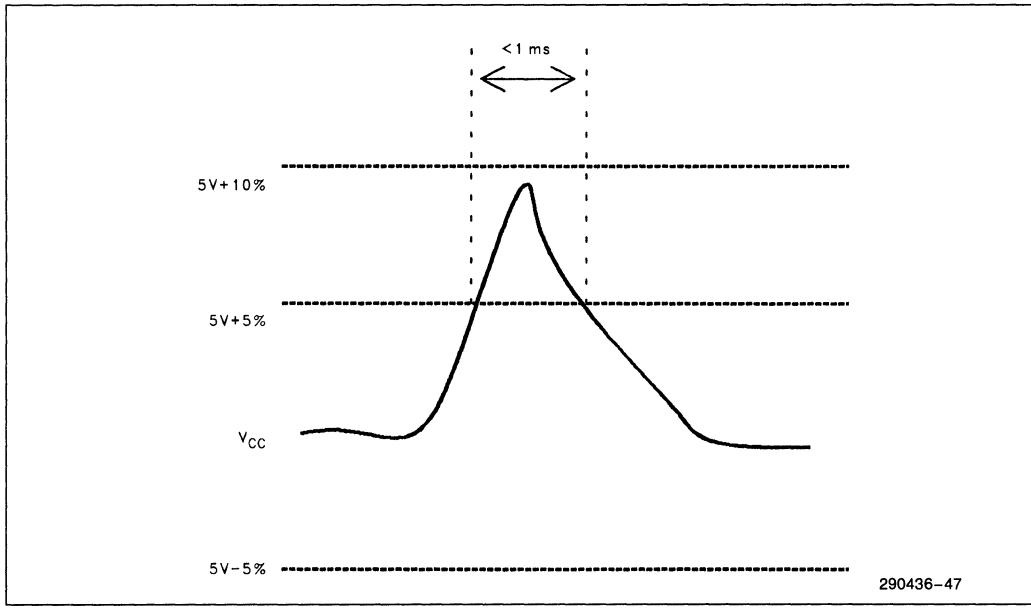


Figure 13-7. V_{CC} Transient Example

Table 13-9. Pentium® OverDrive® Processor—25 MHz AC Characteristics

V_{CC} = 5V + 5%; $T_{A(IN)}$ = 10°C to +55°C; C_L = 50 pF (1) Unless Otherwise Specified

Symbol	Parameter	Min	Max	Units	Figure	Notes
	Frequency	8	25	MHz		1X Clock Input to Processor
t_1	**CLK** Period	40	125	ns	13-8	
t_{1a}	CLK Period Stability		250	ps		Adjacent Clocks
t_2	**CLK** High Time	14		ns	13-8	at 2V
t_3	**CLK** Low Time	11		ns	13-8	at 0.8V
t_4	**CLK** Fall Time		4	ns	13-8	2V to 0.8V
t_5	**CLK** Rise Time		4	ns	13-8	0.8V to 2V
t_6	**A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK#, FERR#, BREQ, HLDA, SMIACT#**, Valid Delay	3	19	ns	13-10	
t_7	**A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK#** Float Delay		28	ns	13-10	After Clock Edge(2)
t_8	**PCHK#** Valid Delay	3	24	ns	13-10	
t_{8a}	**BLAST#, PLOCK#** Valid Delay	3	24	ns	13-10	
t_9	**BLAST#, PLOCK#** Float Delay		28	ns	13-10	After Clock Edge(2)
t_{10}	**D0–D31, DP0–3** Write Data Valid Delay	3	20	ns	13-10	
t_{11}	**D0–D31, DP0–3** Write Data Float Delay		28	ns	13-10	After Clock Edge(2)
t_{12}	**EADS#** Setup Time	8		ns	13-9	
t_{13}	**EADS#** Hold Time	3		ns	13-9	
t_{14}	**KEN#, BS16#, BS8#** Setup Time	8		ns	13-9	

Table 13-9. Pentium® OverDrive® Processor—25 MHz AC Characteristics (Continued)
V_{CC} = 5V + 5%; $T_{A(IN)}$ = 10°C to +55°C; C_L = 50 pF (1) Unless Otherwise Specified

Symbol	Parameter	Min	Max	Units	Figure	Notes
t_{15}	**KEN#, BS16#, BS8#** Hold Time	3		ns	13-9	
t_{16}	**RDY#, BRDY#,** Setup Time	8		ns	13-9	
t_{17}	**RDY#, BRDY#,** Hold Time	3		ns	13-9	
t_{18}	**HOLD, AHOLD** Setup Time	8		ns	13-9	
t_{18a}	**BOFF#, SMI#** Setup Time	8		ns	13-9	
t_{19}	**HOLD, AHOLD, BOFF#, SMI#** Hold Time	3		ns	13-9	
t_{20}	**RESET, FLUSH#, A20M#, NMI, INTR, IGNNE#, INIT** Setup Time	8		ns	13-9	
t_{21}	**RESET, FLUSH#, A20M#, NMI, INTR, IGNNE#, INIT Hold Time**	3		ns	13-9	
t_{22}	**D0–D31, DP0–3, A4–A31** Read Setup Time	5		ns	13-9	
t_{23}	**D0–D31, DP0–3, A4–A31** Read Hold Time	3		ns	13-9	
t_{38}	**WB/WT#** and **EWBE#** Setup Time	8		ns	13-9	
t_{39}	**WB/WT#** and **EWBE#** Hold Time	3		ns	13-9	
t_{40}	**INV Setup Time**	8		ns	13-9	
t_{41}	**INV** Hold Time	3		ns	13-9	
t_{42}	**HIT#, HITM#** Valid Delay	3	19	ns	13-10	
t_{43}	**HIT#, HITM#** Float Delay		28	ns	13-10	Only during Three State Test Mode
t_{44}	**CACHE#** Valid Delay	3	19	ns	13-10	
t_{45}	**CACHE#** Float Delay		28	ns	13-10	
t_{46}	**STPCLK#** Setup Time	5		ns	13-9	
t_{47}	**STPCLK#** Hold Time	3		ns	13-9	

NOTES:
1. All timing specifications assume C_L = 50 pF. Section 14.3.1 provides the charts that may be used to determine the delay due to derating, depending on the lumped capacitive loading, that must be added to these specification values.
2. Not 100% tested, guaranteed by design characterization.

Table 13-10. Pentium® OverDrive® Processor—33 MHz AC Characteristics

V_{CC} = 5V + 5%; $T_{A(IN)}$ = 10°C to +55°C; C_L = 50 pF (1) Unless Otherwise Specified

Symbol	Parameter	Min	Max	Units	Figure	Notes
	Frequency	8	33	MHz		1X Clock Input to Processor
t_1	**CLK** Period	30	125	ns	13-8	
t_{1a}	**CLK** Period Stability		250	ps		Adjacent Clocks
t_2	**CLK** High Time	11		ns	13-8	At 2V
t_3	**CLK** Low Time	8		ns	13-8	At 0.8V
t_4	**CLK** Fall Time		3	ns	13-8	2V to 0.8V
t_5	**CLK** Rise Time		3	ns	13-8	0.8V to 2V
t_6	**A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK#, FERR#, BREQ, HLDA, SMIACT#,** Valid Delay	3	14	ns	13-10	
t_7	**A2–A31, PWT, PCD, BE0–3#, M/IO#, D/C#, W/R#, ADS#, LOCK# BP3, BP2,** Float Delay		20	ns	13-10	After Clock Edge(2)
t_8	**PCHK#** Valid Delay	3	14	ns	13-10	
t_{8a}	**BLAST#, PLOCK#** Valid Delay	3	14	ns	13-10	
t_9	**BLAST#, PLOCK#** Float Delay		20	ns	13-10	After Clock Edge(2)
t_{10}	**D0–D31, DP0–3** Write Data Valid Delay	3	14	ns	13-10	
t_{11}	**D0–D31, DP0–3** Write Data Float Delay		20	ns	13-10	After Clock Edge(2)

Table 13-10. Pentium® OverDrive® Processor—33 MHz AC Characteristics (Continued)
V_{CC} = 5V + 5%; $T_{A(IN)}$ = 10°C TO +55°C; C_L = 50 pF (1) Unless Otherwise Specified

Symbol	Parameter	Min	Max	Units	Figure	Notes
t_{12}	**EADS#** Setup Time	5		ns	13-9	
t_{13}	**EADS#** Hold Time	3		ns	13-9	
t_{14}	**KEN#, BS16#, BS8#** Setup Time	5		ns	13-9	
t_{15}	**KEN#, BS16#, BS8#** Hold Time	3		ns	13-9	
t_{16}	**RDY#, BRDY#,** Setup Time	5		ns	13-9	
t_{17}	**RDY#, BRDY#,** Hold Time	3		ns	13-9	
t_{18}	**HOLD, AHOLD** Setup Time	6		ns	13-9	
t_{18a}	**BOFF#, SMI#** Setup Time	7		ns	13-9	
t_{19}	**HOLD, AHOLD, BOFF#, SMI#** Hold Time	3		ns	13-9	
t_{20}	**RESET, FLUSH#, A20M#, NMI, INTR, IGNNE#, INIT** Setup Time	5		ns	13-9	
t_{21}	**RESET, FLUSH#, A20M#, NMI, INTR, IGNNE#, INIT** Hold Time	3		ns	13-9	
t_{22}	**D0–D31, DP0–3, A4–A31** Read Setup Time	5		ns	13-9	
t_{23}	**D0–D31, DP0–3, A4–A31** Read Hold Time	3		ns	13-9	
t_{38}	**WB/WT#** and **EWBE#** Setup Time	5		ns	13-9	
t_{39}	**WB/WT#** and **EWBE#** Hold Time	3		ns	13-9	
t_{40}	**INV** Setup Time	5		ns	13-9	
t_{41}	**INV** Hold Time	3		ns	13-9	
t_{42}	**HIT#, HITM#** Valid Delay	3	14	ns	13-10	
t_{43}	**HIT#, HITM#** Float Delay		20	ns	13-10	Only during Three State Test Mode
t_{44}	**CACHE#** Valid Delay	3	14	ns	13-10	
t_{45}	**CACHE#** Float Delay		20	ns	13-10	
t_{46}	**STPCLK#** Setup Time	5		ns	13-9	
t_{47}	**STPCLK#** Hold Time	3		ns	13-9	

NOTES:
1. All signal timings except Boundary Scan timing specifications assume C_L = 50 pF. Section 14.3.1 provides the charts that may be used to determine the delay due to derating, depending on the lumped capacitive loading, that must be added to these specification values.
2. Not 100% tested, guaranteed by design characterization.

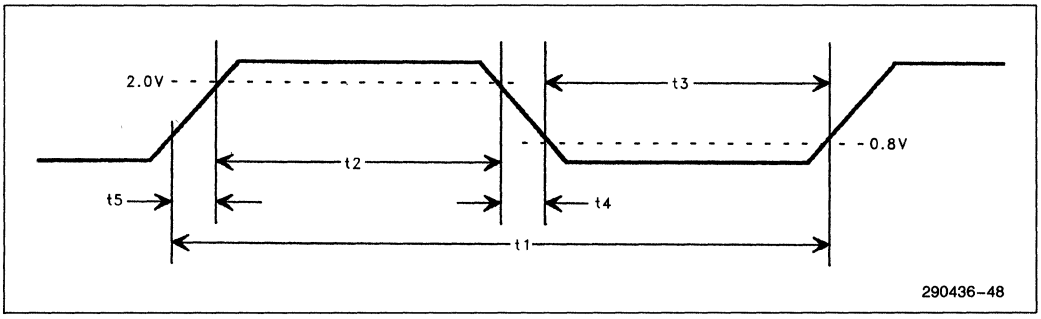


Figure 13-8. CLK Waveform

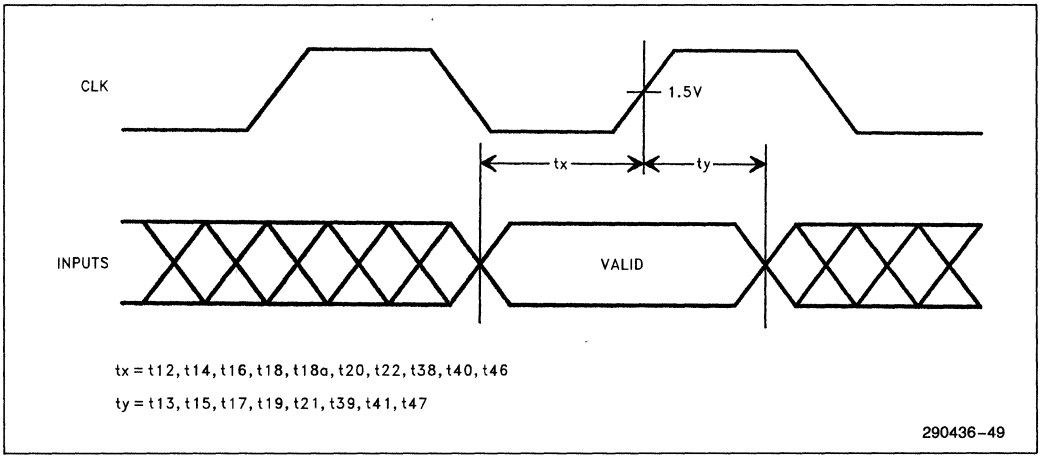


Figure 13-9. SETUP and HOLD Timings

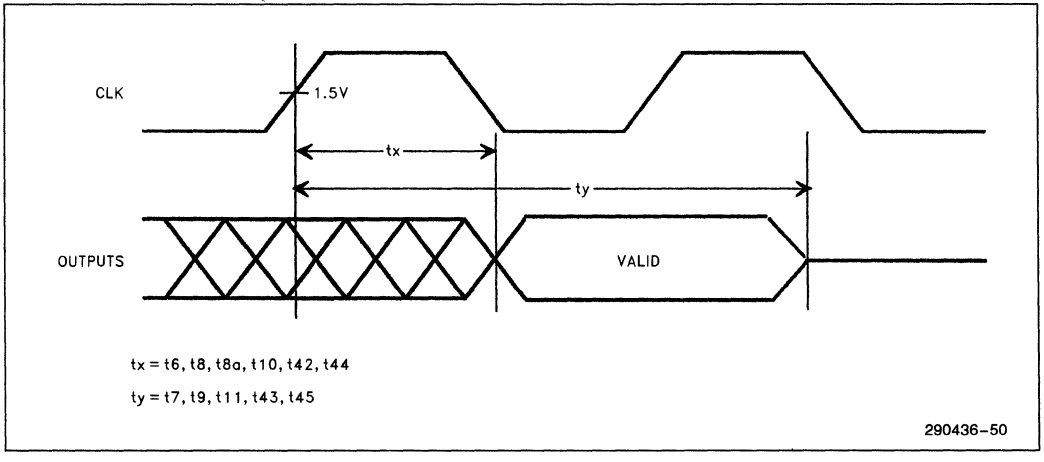


Figure 13-10. Valid and Float Delay Timings

13.4.3.2 Derating I/O Specifications

Figures 13-11 and 13-12 can be used to determine the amount of derating necessary for a given amount of lumped capacitive load. This delay due to derating must be added accordingly to the specification values listed in Tables 13-9 and 13-10 for the Pentium OverDrive processor. These values are design estimates.

Refer to the Pentium Processor Data Book for more information on instruction execution timing and pairing.

A generic discussion on the operation of cache memories can be found in the Intel Cache Tutorial available from your Intel sales representative or from Intel's Literature department, order #296543-002.

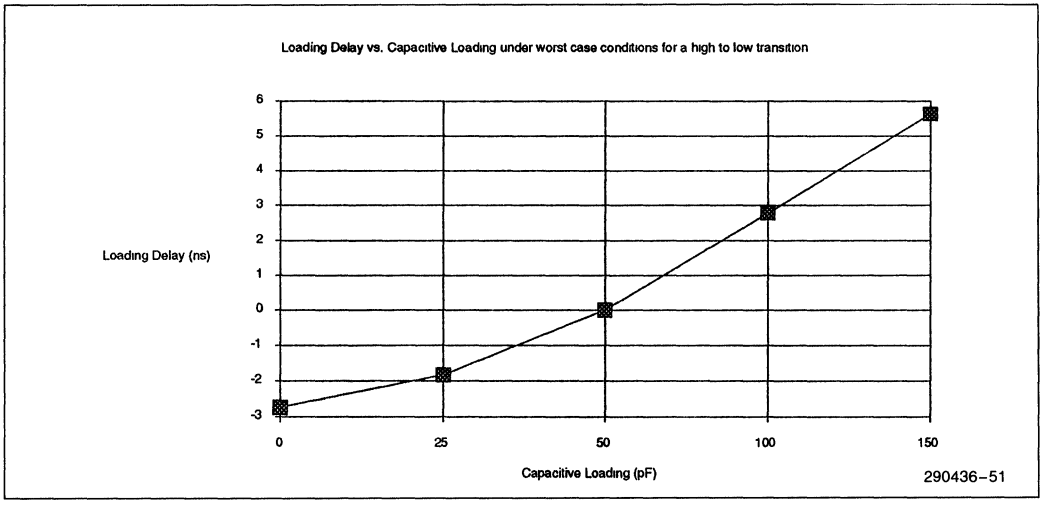


Figure 13-11. Loading Delay vs Load Capacitance (High to Low Transition)

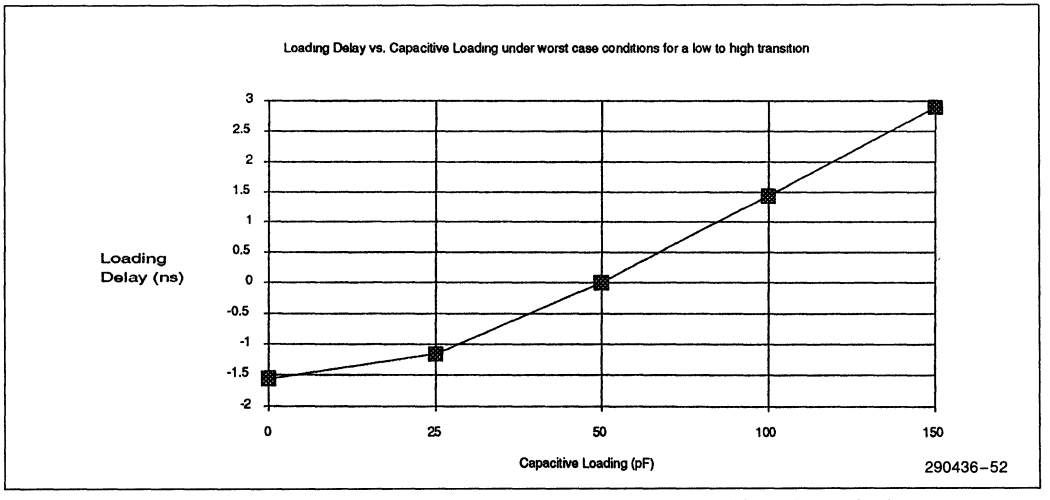


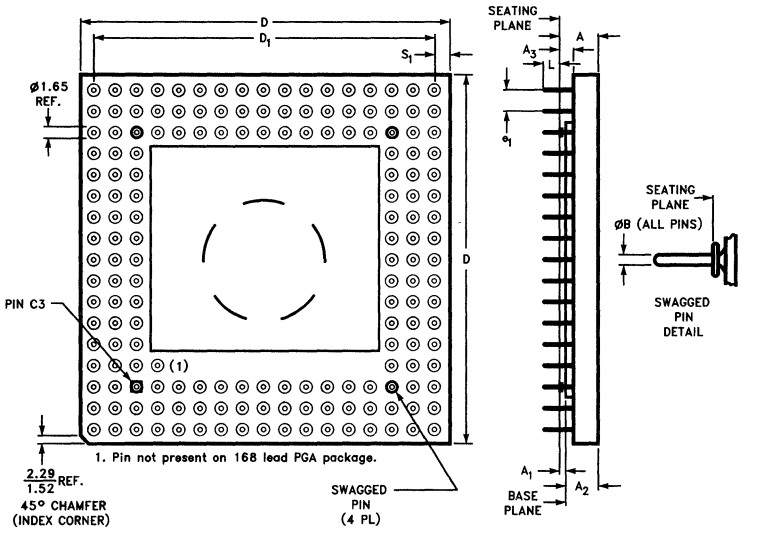
Figure 13-12. Loading Delay vs Load Capacitance (Low to High Transition)

14.0 MECHANICAL DATA

The following sections describe the physical dimensions of the OverDrive processor packages and heat sinks.

14.1 Package Dimensions for the IntelDX2™ and IntelDX4™ OverDrive® Processors

Figure 14-1 describes the physical dimensions of the PGA packages (168-lead PGA and 169-lead PGA) used with the IntelDX2 and IntelDX4 OverDrive processors.



Family: Ceramic Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A	3.56	4.57		0.140	0.180	
A_1	0.64	1.14	SOLID LID	0.025	0.045	SOLID LID
A_2	2.8	3.5	SOLID LID	0.110	0.140	SOLID LID
A_3	1.14	1.40		0.045	0.055	
B	0.43	0.51		0.017	0.020	
D	44.07	44.83		1.735	1.765	
D_1	40.51	40.77		1.595	1.605	
e_1	2.29	2.79		0.090	0.110	
L	2.54	3.30		0.100	0.130	
N	168, 169			168, 169		
S_1	1.52	2.54		0.060	0.100	
ISSUE	IWS REV X 7/15/88					

Figure 14-1. OverDrive® Processor Package Dimensions

14.2 Heat Sink Dimensions

There are two different passive heat sinks and one fan heat sink used on the Intel OverDrive processors. The IntelDX2 OverDrive processor uses the 0.25″ heat sink. The IntelDX4 OverDrive processor uses the 0.6″ heat sink. The Pentium OverDrive processor uses an integrated fan heat sink. All three heat sinks are described in the following sections.

14.2.1 0.25″ PASSIVE HEAT SINK

Figure 14-2 describes the physical dimensions of the 0.25″ heat sink used with the IntelDX2 OverDrive processor. Table 14-1 lists the physical dimensions.

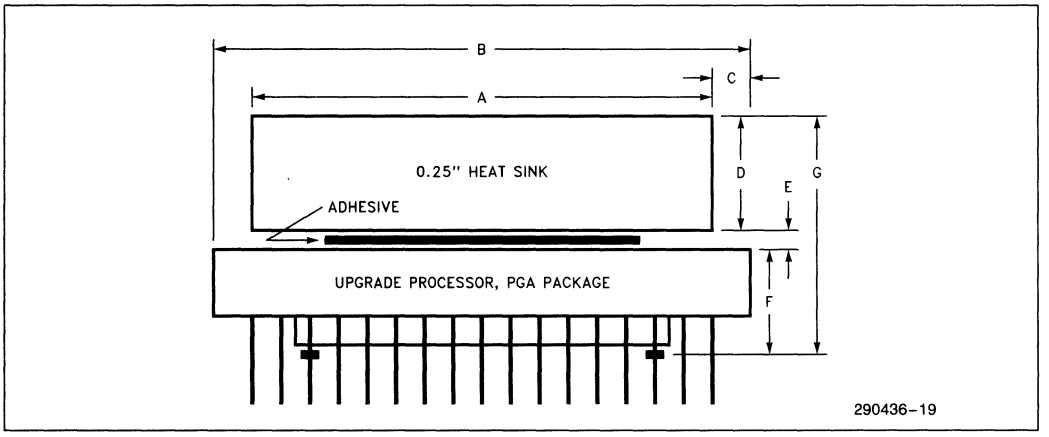


Figure 14-2. Dimensions, IntelDX2™ OverDrive® Processor with 0.25″ Heat Sink

Table 14-1. 0.25″ Heat Sink Dimensions

Dimension (inches)	Minimum	Maximum
A. Heat Sink Width	1.520	1.550
B. PGA Package Width	1.735	1.765
C. Heat Sink Edge Gap	0.065	0.155
D. Heat Sink Height	0.212	0.260
E. Adhesive Thickness	0.008	0.012
F. Package Height from Stand-Offs	0.140	0.180
G. Total Height from Package Stand-Offs to Top of Heat Sink	0.360	0.452

14.2.2 0.6" PASSIVE HEAT SINK

Figure 14-3 describes the physical dimensions of the 0.6" heat sink used with the IntelDX4 OverDrive processors. The maximum and minimum dimensions for the PGA package with heat sink are shown in Table 14-2. As the table shows, the maximum height of the IntelDX4 OverDrive processor from the pin stand-offs to the top of the heat sink, including the adhesive thickness, is 0.780 inches. A minimum clearance of 0.25" should be allowed above the top of the heat sink.

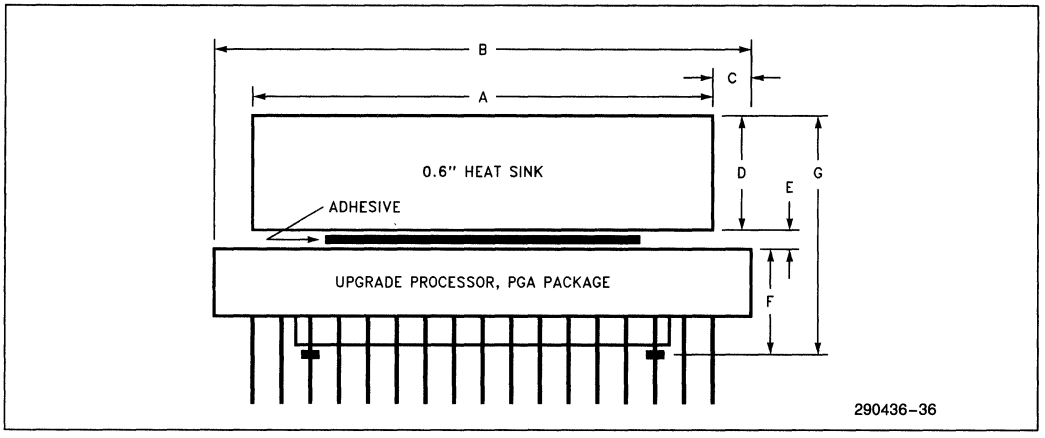


Figure 14-3. Dimensions, IntelDX4™ OverDrive® Processor with 0.6" Heat Sink

Table 14-2. 0.6" Heat Sink Dimensions

Dimension (inches)	Minimum	Maximum
A. Heat Sink Width	1.520	1.550
B. PGA Package Width	1.735	1.765
C. Heat Sink Edge Gap	0.065	0.155
D. Heat Sink Height	0.580	0.600
E. Adhesive Thickness	0.006	0.012
F. Package Height from Stand-Offs	0.140	0.180
G. Total Height from Package Stand-Offs to Top of Heat Sink	0.720	0.780

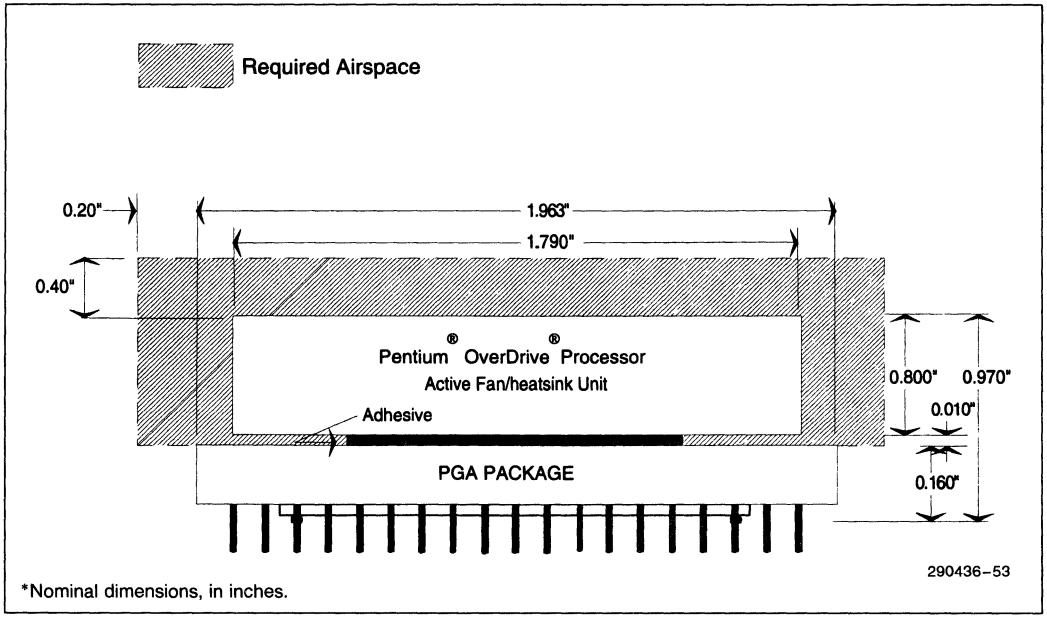
14.2.3 Pentium® OverDrive® PROCESSOR ACTIVE HEAT SINK

The Pentium OverDrive processor is designed to fit in a standard 240-lead (19 x 19) PGA socket with four corner pins removed. The Pentium OverDrive processor will use an active heat sink, and therefore requires more vertical clearance. For more discussion on the Pentium OverDrive processor active heat sink, please see the following section.

The maximum and minimum dimensions of the Pentium OverDrive processor package with the active heat sink are shown in Table 14-3. The active heat sink unit is divided into the size of the actual heat sink, and the required free space above the heat sink. The total height required for the Pentium OverDrive processor from the motherboard will depend on the height of the PGA socket. The total external height given in the table below is only measured from the PGA pin stand-offs. Table 14-3 also details the minimum clearance needed around the PGA package.

Table 14-3. Pentium® OverDrive® Processor, 235-Pin, PGA Package Dimensions with Active Heat Sink Attached

Component (inches)	Length and Width		Height	
	Min	Max	Min	Max
PGA Package	1.950	1.975	0.140	0.180
Adhesive	N/A	N/A	0.008	0.012
Fan/Heat Sink	1.77	1.82	0.790	0.810
Required Airspace	0.200	N/A	0.400	N/A
External Pkg. Total	1.950	1.975	0.938	1.002
Min. Ext. w/ Airspace Fixture	2.150		1.338	



*Nominal dimensions, in inches.

Figure 14-4. 235-Pin, PGA Package with Active Heat Sink Attached

14.2.3.1 ACTIVE HEAT SINK DETAILS

Since the Pentium OverDrive processor dissipates more power than the Intel486 Family, it requires a larger cooling capacity. To accomplish the task of cooling the Pentium OverDrive processor, an active heat sink is attached to the top of the part. The active heat sink will use a heat sink/fan combination to provide airflow at high velocity to the Pentium OverDrive processor. No external connections (Power, etc . . .) will be required for the active heat sink. All the needed connections will be made through the pins of the processor. The amount of extra power needed for the fan is taken into account in the I_{CC} numbers of the processor.

The fan/heat sink unit also supports an integrated thermal protection mechanism that will allow the fan to signal the processor if the speed of the fan should become insufficient to cool the processor. Should this occur, the processor will modify its internal core frequency to match the **CLK** input in a manner that is transparent to the external system. The fan/heat sink has been designed so that should the fan stop, it will have the capability to properly cool the processor in a still air environment. The fan unit is removable so that the unit may be easily replaced. If the fan is removed, or power to the fan is lost, the processor will treat these conditions as if the fan has failed. Figure 14-5 below gives a functional representation of the Pentium OverDrive processor and heat sink unit.

As can be seen in the mechanical dimensions in Table 14-3, the actual height required by the heat sink is less than the total space allotted. Since the Pentium OverDrive processor employs an active heat sink, a certain amount of space is required above the heat sink unit to ensure that the airflow is not blocked. Figure 14-6 shows unacceptable blocking of the airflow for the Pentium OverDrive processor heat sink unit. Figure 14-7 details the minimum space needed around the PGA package to ensure proper heat sink airflow.

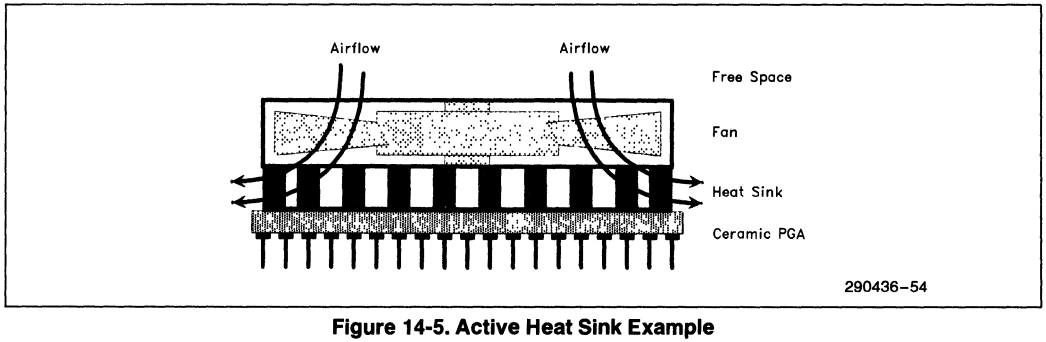


Figure 14-5. Active Heat Sink Example

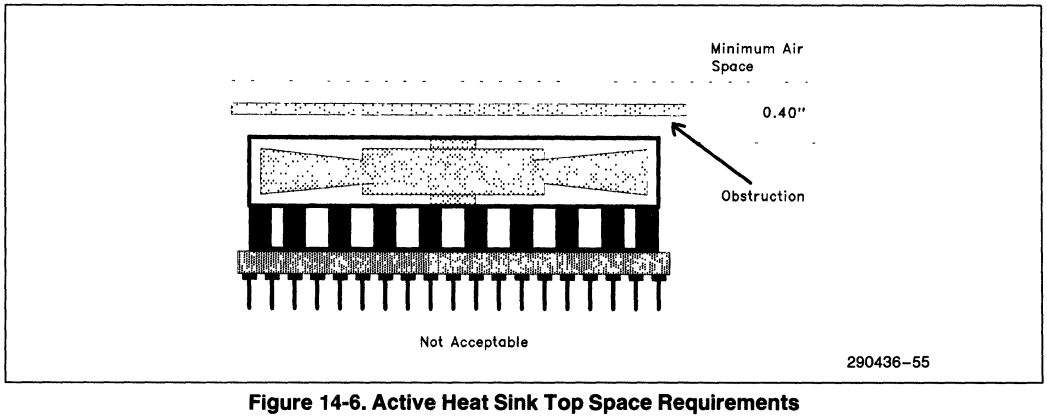


Figure 14-6. Active Heat Sink Top Space Requirements

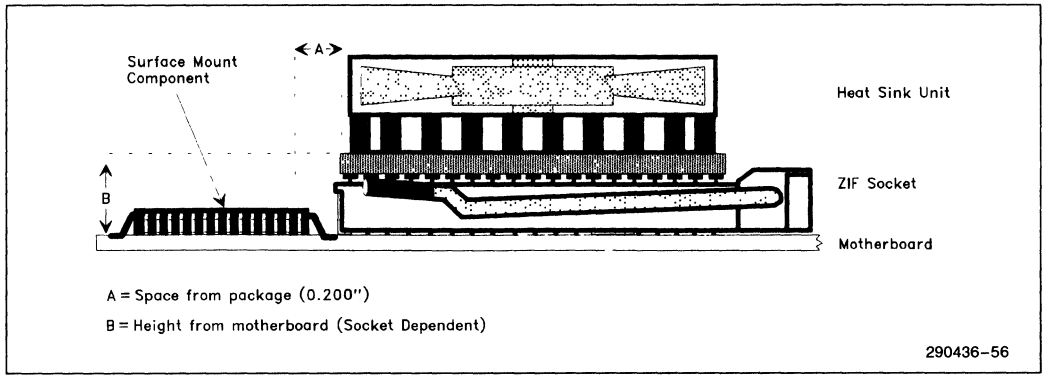


Figure 14-7. Required Free Space from Sides of PGA Package

As shown in Figure 14-7, it is acceptable to allow any device to enter within the free space distance of 0.2" from the PGA package if it is not taller than the level of the heat sink base. In other words, if a component is taller than height "B", it can not be closer to the PGA package than distance "A". This applies to all four sides of the PGA package, although the back and handle sides of a ZIF socket will generally automatically meet this specification since they have widths larger than distance "A".

14.3 Pentium® OverDrive® Processor Socket

14.3.1 SOCKET BACKWARD COMPATIBILITY

The Pentium OverDrive processor socket is designed specifically for the requirements of the Pentium OverDrive processor. In addition the socket can accept and is pin compatible with the IntelDX2 and IntelDX4 OverDrive processors. This added compatibility may be useful during system troubleshooting and debug.

The Pentium OverDrive processor defines a fourth row of contacts around the outside of the 169 contacts defined for the IntelDX2 and IntelDX4 OverDrive processors. The three inner rows of the socket are 100% compatible with the IntelDX2 and IntelDX4 OverDrive processors. For backward compatibility, the inner row key pin location (E5) must be included in any socket that is to accept the Pentium OverDrive processor. For proper operation of the Pentium OverDrive processor, all the power and ground pins in the outer row of pins must be connected.

14.3.2 SOCKET 3 PINOUT

Socket 3 is the ZIF (Zero Insertion Force) socket recommended for the Pentium OverDrive processor. To ensure proper orientation, four corner pins have been removed from the outer row of pins. Additionally, the three inner rows of pins are compatible with the IntelDX2 and IntelDX4 OverDrive processors. This includes the "key" pin in the inside corner. Figure 14-8 shows an example of the pinout of Socket 3.

Socket 2, a previous ZIF socket definition for the Pentium OverDrive processor, is compatible to the IndelDX2, IntelDX4, and Pentium OverDrive processors. This definition has been replaced with the socket 3 definition so it will not be discussed.

Product Highlights

- Distinctive socket with the "Socket 3" marking
- Rapid end user access to Socket 3
- Keyed ZIF socket for easy and correct Pentium OverDrive processor installation

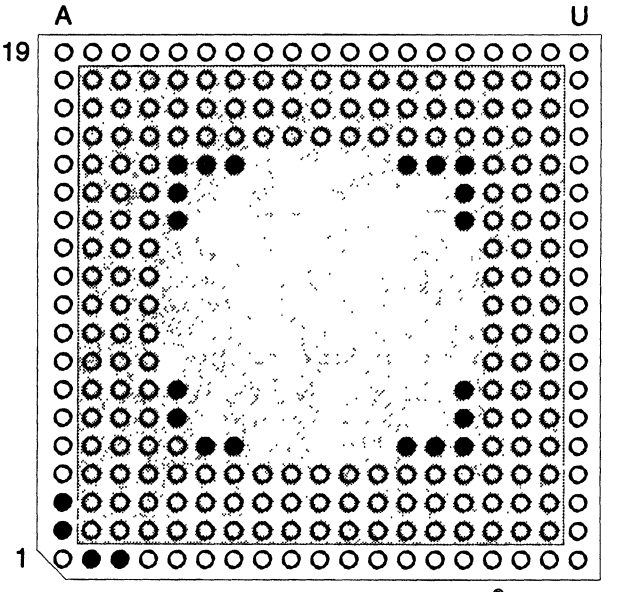


Figure 14-8. 237-Pin, PGA ZIF Socket 3

Table 15-1. Thermal Resistance, IntelDX2™ OverDrive® Processor with Attached Heat Sink

θ_{JS} = 2.5°C/W	Airflow (LFM)				
	0	200	400	600	800
θ_{JA}(°C/W)	14.0	10.0	7.5	6.2	5.7

Table 15-2. Thermal Resistance, IntelDX4™ OverDrive® Processor with Attached Heat Sink

θ_{JS} = 2.0°C/W	Airflow (LFM)			
	0	50	100	200
θ_{JA}(°C/W)	11.5	10.7	9.5	7.0

15.0 THERMAL MANAGEMENT

The heat generated by the Intel OverDrive processor requires that heat dissipation be managed carefully. All OverDrive processors are supplied with a heat sink attached with adhesive to the package. System designs must, therefore, provide sufficient clearance (a minimum of 0.25″ above the heat sink) for the processor and the attached heat sink.

Section 14 contains the physical dimensions for each of the heat sinks and packages used.

The standard product markings and logo for the Intel OverDrive processor with the attached heat sink will be included on a $1in^2$ plate located on the top, center of the heat sink.

The heat sink is omni-directional, allowing air to flow from any direction in order to achieve adequate cooling. The thermal resistance values for the OverDrive processors with an attached heat sink are shown in Table 15-1 through Table 15-3.

The Pentium OverDrive processor and system chassis have several unique design requirements due to the attached active heat sink. The following sections provide sample maximum system operating temperature calculations so that systems may be designed to comply with the thermal requirements of the Pentium OverDrive processor.

15.1 Thermal Calculations for a Hypothetical System

The following equation can be used to calculate the maximum operating temperature of a system.

$T_{A(IN)} = T_{SINK} - (Power * \theta_{SI})$

The parameters are defined as follows:

- $T_{A(IN)}$: The temperature of the air going **into** the heat sink fan unit.
- T_{SINK}: Temperature of heat sink base, as measured in the center.
- Power: Dissipation in Watts = $V_{CC} * I_{CC}$
- θ_{SI}: Heat Sink to Internal Temperature [$T_{A(IN)}$] Thermal Resistance
- $T_{A(OUT)}$: The temperature of the air outside the system.

Since the Pentium OverDrive processor uses an active heat sink, θ_{SI} is relatively constant, regardless of the airflow provided to the processor. The θ_{SI} is provided in Table 15-3. Table 15-4 details the maximum current requirements of the Pentium OverDrive processor. The maximum allowable $T_{A(IN)}$ is 55°C for both 25 MHz and 33 MHz with the heat sink attached.

Table 15-3. Thermal Resistance (°C/W) θ_{SI}

Processor Type	θ_{SI}—°C/W
Active Heat Sink	2.4

Table 15-4. Pentium® OverDrive® Processor Typical and Maximum I_{CC} Values

System Frequency (MHz)	Processor Typical I_{CC} (mA)	Processor Maximum I_{CC} (mA)
25	TBD	2200
33	TBD	2600

I_{CC} is dependent upon the V_{CC} level of the system, processor bus loading, software code sequences, and silicon process variations. For the Pentium OverDrive processor specifications, the maximum I_{CC} value is derived by testing a sample of components under the following worst case conditions: V_{CC} = 5.3V, full DC current loads on all output pins, and running a file with the predicted worst case software code sequences at the specified frequency. The typical I_{CC} value published is the I_{CC} corresponding to the worst observed I_{CC} value for an average component running under the above worst case conditions. No additional margin is added to this value. I_{CC} typical is not a guaranteed specification.

15.2 Airflow

Since the Pentium OverDrive processor employs an active heat sink, it is not as important that the processor heat sink receive direct airflow, rather that the system has sufficient capability to remove the warm air that the Pentium OverDrive processor will generate. This implies that enough airflow exists at the Pentium OverDrive processor socket site to keep localized heating from occurring. This can be accomplished by a standard power supply fan with a clear path to the processor. Figure 15-1 shows how system design can cause localized heating to occur by limiting the airflow in the area of the processor. The airflow supplied in the system should also be enough to insure that the OEM processor shipped with the system will meet the OEM processor thermal specifications before the system is upgraded with the Pentium OverDrive processor.

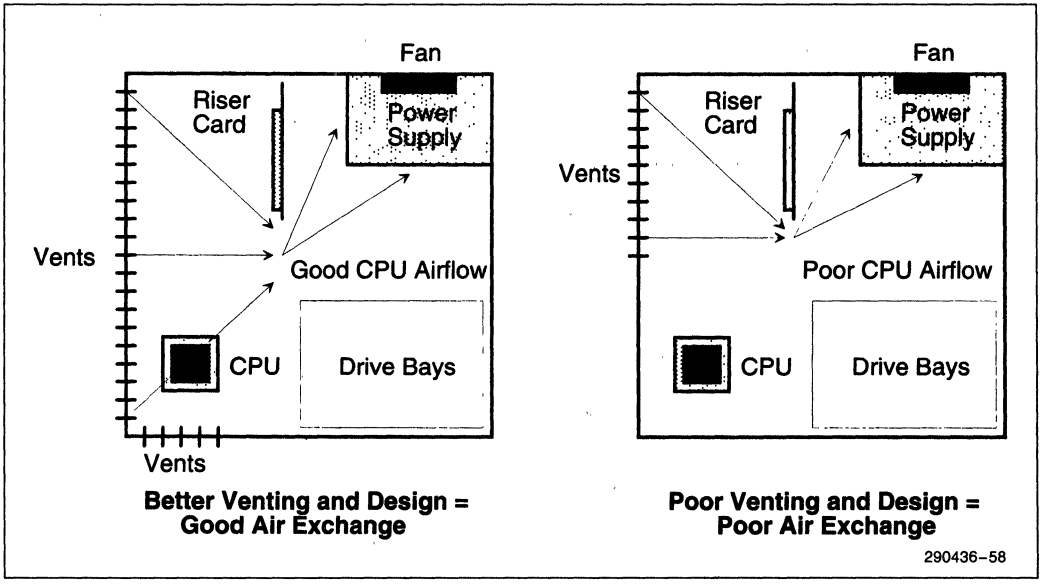


Figure 15-1. Pentium® OverDrive® Processor Airflow Design Examples

APPENDIX A
CACHE FUNCTIONALITY

Cache Introduction

Special hooks are provided to support the Pentium OverDrive processor on-chip write back cache and to maintain cache consistency. The external environment can dynamically change the caching policy of the Pentium OverDrive processor on a line by line basis.

The Pentium OverDrive processor has separate code and data caches. Each of the caches are 16 Kbytes in size and each is organized as a 4-way set associative cache. The data cache follows the MESI cache consistency protocol while the code cache follows a subset of that protocol. For a complete description of the cache see the *Intel486 Microprocessor Family Data Book*(2).

NOTE:
2. A generic discussion on the operation of cache memories can be found in the Intel Cache Tutorial available from your Intel sales representative or from Intel's Literature department, order #296543-002.

Cache Organization

The Pentium OverDrive processor includes separate code and data caches on chip to meet its performance goals. The code and data caches can be accessed simultaneously. The code cache can provide up to 16 bytes of raw opcodes and the data cache can provide data for two data references all in the same clock. Each of the caches are accessed with physical addresses and each cache has its own TLB (translation look aside buffer) to translate linear addresses to physical addresses. A cache consistency protocol called the **MESI** protocol is implemented in the data cache to ensure data consistency in a multiprocessor environment.

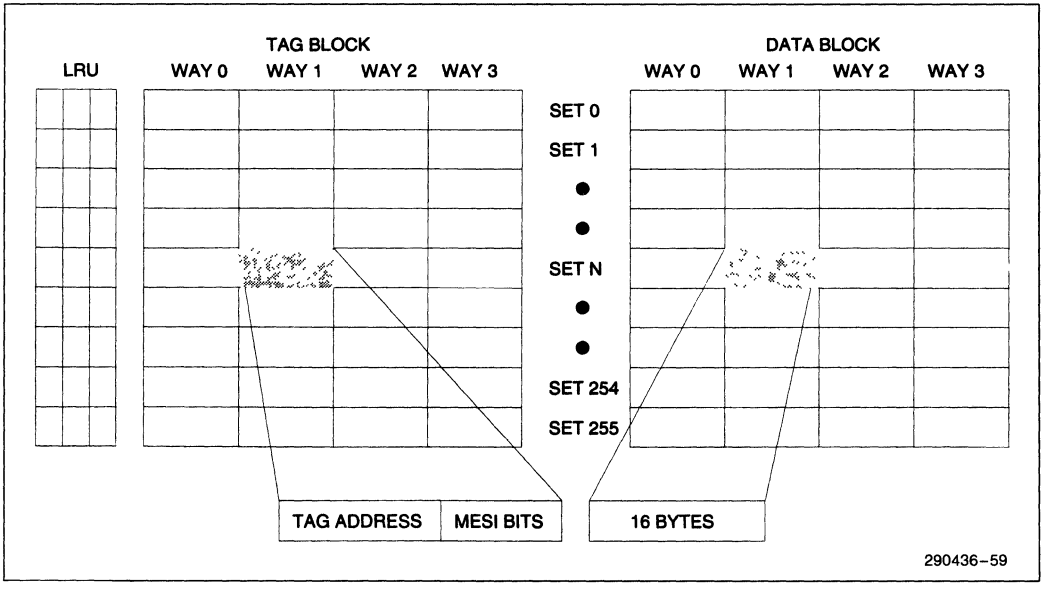


Figure A-1. Conceptual Organization of Data Cache

Each of the caches are 16 Kbytes in size and each is organized as a 4-way set associative cache. There are 256 sets in each cache, each set containing 4 lines. Each cache line is 16 bytes wide. Replacement in both the data and instruction caches is handled by a pseudo LRU mechanism which requires three bits per set in each of the caches. A conceptual diagram of the organization of the data cache is shown in Figure A-1.

The data cache can support two data references simultaneously in one clock, one from each of the two pipelines. It is a write back cache with full support for data consistency in a multimaster environment. This is implemented with two status bits associated with each cache line. The data cache can optionally be configured in write through mode on a line by line basis when in write back cache mode. The storage array in the data cache is single ported but interleaved on 4 byte boundaries to be able to provide data for two simultaneous accesses to the same cache line. The tags in the data cache are triple ported. One of the ports is dedicated to snooping while the other two are used to lookup two independent addresses corresponding to data references from each of the pipelines. The code cache tags are also triple ported. Again, one port is dedicated to support snooping and other two ports facilitate split line accesses (simultaneously accessing upper half of one line and lower half of the next line).

The data cache has a 4-way set associative, 64-entry TLB for 4 KB pages and a separate 4-way set associative, 8-entry TLB to support 4 MB pages. The code cache has one 4-way set associative, 32-entry TLB for 4 KB pages as well as 4 MB pages which are cached in 4 KB increments. The TLBs associated with the instruction cache are single ported whereas the data cache TLBs are fully dual ported to be able to translate two independent linear addresses for two data references simultaneously. Replacement in the TLBs is handled by a pseudo LRU mechanism (similar to the Intel486™ CPU) that requires 3 bits per set. The tag and data arrays of the TLBs are parity protected with a parity bit associated with each of the tag and data entries in the TLBs.

State Transition Tables

Lines cached in the Pentium OverDrive processor can change state because of Pentium OverDrive processor generated activity or as a result of activity on the Pentium OverDrive processor bus generated by other bus masters (snooping). As shown in the following tables, state transitions occur because of Pentium OverDrive processor generated transactions (memory reads/writes) and snooping by the external system. This protocol has minor differences from the MEI protocol of the Write-Back Enhanced IntelDX2 processor as detailed in Appendix B.

READ CYCLE

The state transitions for the data cache during reads are shown in Table A-1. For a cache line that is in the M (Modified), E (Exclusive) or S (Shared) states, the data is transferred from the cache to the core, with no bus cycle generated.

Three different cases can occur when a cache read occurs for an I-state (Invalid) line. An access to an invalid line indicates a miss in the cache, so a read cycle will be generated. If the **CACHE#** and **KEN#** pins are sampled low, and **WB/WT#** is high, then the line will be stored in the E-state in the cache. **WB/WT#** is sampled with the first **BRDY#** or **RDY#** of the transfer, while **KEN#** is sampled one clock before the first **RDY#** or **BRDY#.** If the **CACHE#** and **KEN#** are low, and **WB/WT#** is low, then the cache will be defined as write-through. If **PWT** is HIGH, cache line fills will always be stored as shared lines, even if **WB/WT#** is high. This will cause the line to be stored in a S state. If either **CACHE#** or **KEN#** is high, then the line is non-cacheable, so it will remain in the I state.

Table A-1. Data Cache State Transitions for Pentium® OverDrive® Processor Initiated Unlocked Read Cycles

Present State	Pin Activity	Next State	Description
M	n/a	M	Read hit; data is provided to the Pentium OverDrive processor core by cache. No bus cycle is generated.
E	n/a	E	Read hit; data is provided to the Pentium OverDrive processor core by cache. No bus cycle is generated.
S	n/a	S	Read hit; Data is provided to the Pentium OverDrive processor by the cache. No bus cycle is generated.
I	**CACHE#** low AND **KEN#** low AND **WB/WT#** high AND **PWT** low	E	Data item does not exist in cache (MISS). A bus cycle (read) will be generated by the Pentium OverDrive processor. This state transition will happen if **WB/WT#** is sampled high with first **BRDY#** or **RDY#**.
I	**CACHE#** low AND **KEN#** low AND (**WB/WT#** low OR **PWT** high)	S	Same as previous read miss case except that **WB/WT#** is sampled low with first **BRDY#** or **RDY#**. If **PWT** is high, **WB/WT#** is ignored and the resulting line state is always "S".
I	**CACHE#** high OR **KEN#** high	I	**KEN#** pin inactive; the line is not intended to be cached in the Pentium OverDrive processor.

NOTE:
The transition from I to E or S-states (based on **WB/WT#**) happens only if the line is cacheable. If **KEN#** is sampled high, the line is not cached and remains in the I-state.

WRITE CYCLE

The state transitions of data cache lines during Pentium OverDrive processor generated write cycles are illustrated in Table A-2. Writes to SHARED lines in the data cache are always sent out on the bus along with updating the cache with the write item. The status of the **PWT** and **WB/WT#** pins during these write cycles on the bus determines the state transitions in the data cache during writes to S-state lines.

A write to a SHARED line in the data cache will generate a write cycle on the Pentium OverDrive processor bus to update memory and/or invalidate the contents of other caches. If the **PWT** pin is driven high when the write cycle is run on the bus, the line will be updated, and will stay in the S- state regardless of the status of the **WB/WT#** pin that is sampled with the first **BRDY#** or **RDY#**. If **PWT** is driven low, the status of the **WB/WT#** pin sampled along with the first **BRDY#** or **RDY#** for the write cycle determines what state (E or S) the line transitions to.

The state transition from S to E is the only transition in which the data and the status bits are not updated at the same time. The data will be updated when the cycle is written to the Pentium OverDrive processor write buffers. The state transition does not occur until the write has completed on the bus (last **BRDY#** or **RDY#** has been returned). Writes to the line after the transition to the E-state will not generate bus cycles. However, it is possible that writes to the same line that were buffered before the transition to the E-state will generate bus cycles after the transition to E-state.

An inactive **EWBE#** input will stall subsequent writes to an E- or an M- state line until **EWBE#** is returned active.

Table A-2. Data Cache State Transitions for Pentium® OverDrive® Processor Initiated Write Cycles

Present State	Pin Activity	Next State	Description
M	n/a	M	Write hit; update data cache. No bus cycle generated to update memory.
E	n/a	M	Write hit; update cache only. No bus cycle generated; line is now MODIFIED.
S	**PWT** low AND **WB/WT#** high	E	Write hit; data cache updated with write data item. A write through cycle is generated on bus to update memory and/or invalidate contents of other caches. All subsequent writes to E- or M-state lines are held off until completion of write cycle is known and state transition happens.
S	**PWT** low AND **WB/WT#** low	S	Same as above case of write to S-state line except that **WB/WT#** is sampled low.
S	**PWT** high	S	Same as above cases of writes to S-state lines except that this is a write hit to a line in a write-through page; status of **WB/WT#** pin is ignored.
I	n/a	I	Write MISS; a write through cycle is generated on the bus to update external memory. No allocation is done.

NOTE:
Memory writes are buffered while I/O writes are not. There is no guarantee of synchronization between completion of memory writes on the bus and instruction execution after the write.

INQUIRE CYCLES (SNOOPING)

The purpose of inquire cycles is to check whether the address being presented is contained within the caches in the Pentium OverDrive processor. Inquire cycles may be initiated with or without an invalidation request (**INV** = 1 or 0). The processor samples the snoop address during the clock that **EADS#** is active. An inquire cycle is run through the data and code caches through a dedicated snoop port to determine if the address is contained in one of the Pentium OverDrive processor caches. If the address is in a Pentium OverDrive processor cache, the **HIT#** pin is asserted. If the address hits a modified line in the processor , the **HITM#** pin is also asserted and the modified line is then written back to external memory.

Table A-3 shows the state transitions for inquire cycles.

Processor Code Cache Consistency Protocol

The Pentium OverDrive processor code cache follows a subset of the MESI protocol. Access to lines in the code cache are either a Hit (Shared) or a Miss (Invalid). In the case of a read hit, the cycle is serviced internally to the Pentium OverDrive processor and no bus activity is generated. In the case of a read miss, the read is sent to the external bus and may be converted to a line fill.

Lines are never overwritten in the code cache. Writes generated by the Pentium OverDrive processor are snooped by the code cache. If there is a hit, the line is invalidated. If there is a miss, no action is taken by the code cache.

Warm Reset Cache Behavior

The **INIT** pin can be used to reset the Pentium OverDrive processor without invalidating the on-chip cache. The Pentium OverDrive processor state after **INIT** is the same as the state after **RESET** except that the internal caches, floating point registers, and SMM Base Register retain whatever values they had prior to recognition of **INIT**. The **INIT** signal can be used instead of **RESET** for warm resets when the cache contents need to be maintained. However, **INIT** cannot be used in lieu of **RESET** after power up. For more information on the INIT and the Pentium OverDrive processor, please see Section 10.3.

Table A-3. Cache State Transitions during Inquire Cycles

Present State	Next State INV = 1	Next State INV = 0	Description
M	I	S	Snoop hit to a MODIFIED line indicated by **HIT#** and **HITM#** pins low. Pentium OverDrive processor schedules the writing back of the modified line to memory.
E	I	S	Snoop hit indicated by **HIT#** pin low; no bus cycle generated.
S	I	S	Snoop hit indicated by **HIT#** pin low; no bus cycle generated.
I	I	I	Address not in cache; **HIT#** pin high.

APPENDIX B
DESIGNING FOR Write-Back Enhanced IntelDX2™/ Pentium® OverDrive® PROCESSOR COMPATIBILITY

End Users have made upgradability an expected feature in any personal computer purchase. The Pentium OverDrive Processor is the intended upgrade for systems based on the Write-Back Enhanced IntelDX2 processor, an IntelDX2 processor with an on-board write back cache. When installed in a system designed to support a write back processor cache, the Pentium OverDrive processor can reach its full performance potential.

To make the task of designing an upgradable system easier, the Write-Back Enhanced IntelDX2 and the Pentium OverDrive processor have been designed to be compatible with one another. However, since the Pentium OverDrive processor retains many of its Pentium processor features, certain system design considerations must be taken into account to ensure that a Write-Back Enhanced IntelDX2 processor system can be upgraded seamlessly.

Throughout this section, a "Single Socket Design" will refer to a motherboard design that has only one processor site. This means that the Write-Back Enhanced IntelDX2 processor must be removed from the socket before the Pentium OverDrive processor can be installed. A "Dual Socket Design" refers to a two socket motherboard, one in which the Write-Back Enhanced IntelDX2 resides in a fixed location, and the Pentium OverDrive processor is installed into an empty upgrade socket. This section will make reference to both types of designs, but concentrates on the single socket design strategy.

The following section provides a list of considerations that must be examined to allow a Pentium OverDrive processor to operate properly in a system designed to support the Write-Back Enhanced IntelDX2 processor and a write back processor cache. The considerations listed here are only intended to be relevant to the Write-Back Enhanced IntelDX2 processor Enhanced Bus Mode although some may still be valid for the Standard Bus Mode. These considerations are provided as guidelines only, and should be used in conjunction with the rest of this document to ensure proper Pentium OverDrive processor operation.

Pinout Differences

The Pentium OverDrive processor pinout is based on a 19x19 PGA package as opposed to the Write-Back Enhanced IntelDX2 processor 17x17 PGA package. Most of the signals that are common between the Write-Back Enhanced IntelDX2 processor and the Pentium OverDrive processor exist on the same pin on both parts (assuming that pin A1 on the Write-Back Enhanced IntelDX2 processor = Pin B2 on the Pentium OverDrive processor).

NOTE:
All references to Pentium OverDrive processor pins are with respect to a 19x19 grid, while references to Write-Back Enhanced IntelDX2 processor pins are on a 17x17 grid.

ADDITIONAL PINS ON THE Pentium® OverDrive® PROCESSOR

The Pentium OverDrive processor defines signal pins that provide functionality that the Write-Back Enhanced IntelDX2 processor does not have. These pins are defined in Table B-1.

Table B-1. Additional Pentium® OverDrive® Processor Pins

Signal Name	Pentium® OverDrive® Processor Pin/ Write-Back Enhanced IntelDX2 Processor Pin	Comment
BLEN# (Input)	A7/Not Present	**BLEN#** (Burst Length) controls write bursting on the Pentium OverDrive processor. It can not be toggled and should be tied high, low, or left unconnected. It must be driven LOW if the system is able to accept burst write backs. If **BLEN#** is driven HIGH to force write backs to be written out as four separate write cycles, HOLD will not be recognized until all four cycles have completed, even though each cycle will have its own **ADS#** and **BLAST#.**
EWBE# (Input)	P1/Not Present	**EWBE#** (External Write Buffer Empty) will allow writes to E or M state lines when asserted LOW. If it is sampled HIGH the processor will hold off writes to E or M state lines until asserted LOW again.
HIT# (Output)	U2/Not Present	**HIT#** provides an indication that an external snoop has hit a M,E or S state line in the internal cache.
UP# (Output)	C15/B14	**UP#** is driven LOW to indicate to the system that an upgrade processor is installed. In a single socket design, it shares a pin with the Write-Back Enhanced IntelDX2 processor signal, **TMS.** In a dual socket design, it should be connected to the Write-Back Enhanced IntelDX2 processor **UP#** (Input).

Due to its larger package and the ability to consume more power, the Pentium OverDrive processor also defines a number of extra V_{CC} and V_{SS} pins that the Write-Back Enhanced IntelDX2 processor does not support. The extra pins are listed in Table B-2.

Table B-2. Additional V_{CC} and V_{SS} Pins

V_{CC}	V_{SS}	V_{SS}	V_{CC}	V_{SS}	V_{SS}
	A5	U3	K19	G19	
A9	A8	U5	L1	H1	
A10	A12	U6	L19	H19	
A11	A13	U7	R1	M1	
A16	A14	U8	R19	M19	
D1	A15	U12	U4	N19	
D19	A17	U13	U9	Q1	
J1	C19	U14	U10	Q19	
J19	E1	U15	U11	S1	
K1	E19	U17	U16	S19	

PINS NOT SUPPORTED BY THE Pentium® OverDrive® PROCESSOR

The Pentium OverDrive processor supports all of the pins on the Write-Back Enhanced IntelDX2 processor except those required for JTAG boundary scan functionality and the **UP#** input pin. Single socket designs should ensure that if boundary scan features are implemented, they will not interfere with the operation of the Pentium OverDrive processor. Dual socket designs should not route the Pentium OverDrive processor into the boundary scan chain. Table B-3 lists the location of the pins that exist on the Write-Back Enhanced IntelDX2 processor, and the corresponding different signals on the Pentium OverDrive processor.

Table B-3. Unsupported Write-Back Enhanced IntelDX2 Processor Pins

Write-Back Enhanced IntelDX2 Processor Signal Name/Pin	Pentium® OverDrive® Processor Signal Name/Pin	Comment
TCK (Input) Pin A3	**INC** Pin B4	Pin B4 on the Pentium OverDrive processor is defined as an **INC** pin so that **TCK** can be used by the Write-Back Enhanced IntelDX2 processor in a single socket design without requiring a jumper.
TDI (Input) Pin A14	**INC** Pin B15	Pin B15 on the Pentium OverDrive processor is defined as an **INC** pin so that **TDI** can be used by the Write-Back Enhanced IntelDX2 processor in a single socket design without requiring a jumper.
TDO (Output) Pin B16	**INC** Pin C17	Pin C17 on the Pentium OverDrive processor is defined as an **INC** pin so that **TDO** can be used by the Write-Back Enhanced IntelDX2 processor in a single socket design without requiring a jumper.
TMS (Input) Pin B14	**UP#** (Output) Pin C15	If the Write-Back Enhanced IntelDX2 processor boundary scan features are used in a single socket design, the design should ensure that **TMS** will not conflict with the Pentium OverDrive processor **UP#** when the upgrade is installed.
UP# (Input) Pin C11	**INC** Pin D12	Pin D12 on the Pentium OverDrive Processor is defined as an **INC** pin so that there can be no conflict with **UP#** (Input).

SHARED SIGNALS LOCATED ON DIFFERENT PINS

There are several signals that the Pentium OverDrive Processor has in common with the Write-Back Enhanced IntelDX2 processor, but which are located on different pins. An example of this would be the **HITM#** signal. On the Pentium OverDrive processor, it is located in the outer row of pins, while on the Write-Back Enhanced IntelDX2 processor, it is located on one of the inner row processors. Single socket designs require that these signals be tied together so that the use of a jumper to reroute the signal is unnecessary. This is done through the use of the **INC** pin.

INC pins, by definition are "internally not connected" and may be used for routing of signals. Certain Pentium OverDrive processor **INC** pins should be connected to the signals that correspond to the Write-Back Enhanced IntelDX2 processor write back signals and the **SRESET** signal. Table B-4 details pins that should be routed together. Please note that the last two columns in Table B-4 are the pins on the Pentium OverDrive socket which must be routed together.

Table B-4. Single Socket Compatibility Signals

Signal	Write-Back Enhanced IntelDX2 Processor Signal Pin	Pentium® OverDrive® Processor Signal Pin	Pentium® OverDrive® Processor INC Pins
INV	A10	N1	B11
HITM#	A12	U1	B13
CACHE#	B12	G1	C13
WB/WT#	B13	T1	C14
INIT	C10	F19	D11
FERR#	C14	B14	D15

Figure B-1 shows an example of how the INC pins shown in Table B-4 should be connected together to allow single socket compatibility between the Write-Back Enhanced IntelDX2 processor and the Pentium OverDrive processor. The figure is provided as an example only and is not intended to be guide for how the signals should actually be routed on a motherboard.

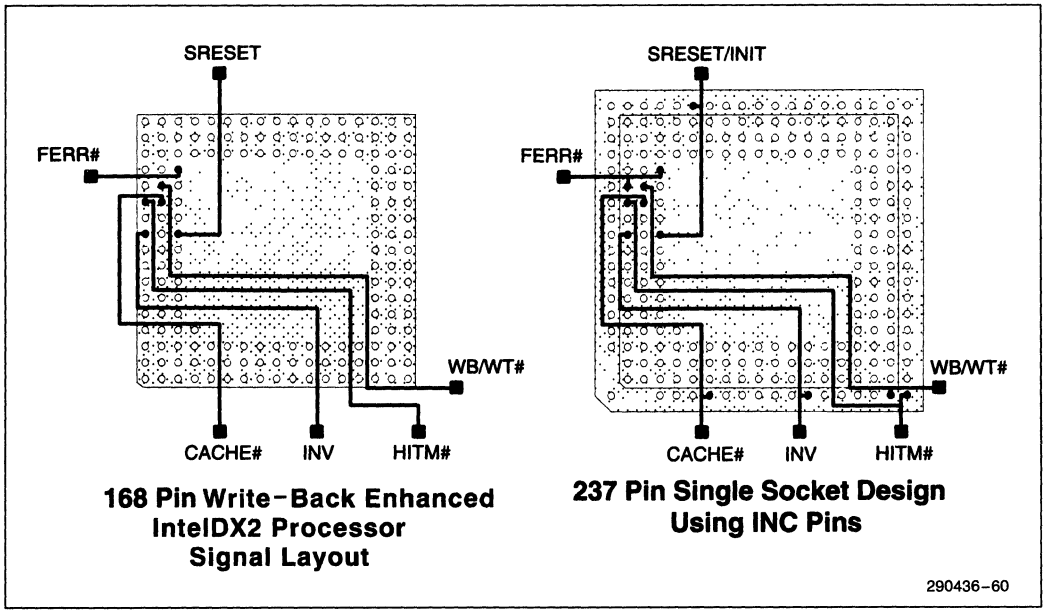


Figure B-1. Sample Routing of INC Pins

Functional Differences

The Pentium OverDrive processor and the Write-Back Enhanced IntelDX2 processor have been designed to be functionally alike, but because the Pentium OverDrive processor is based on the advanced Pentium processor core, there are some minor differences that should be accounted for to ensure that a Pentium OverDrive processor will operate properly in a Write-Back Enhanced IntelDX2 processor-based system.

WRITE BACK PROCESSOR CACHE CONSIDERATIONS

Both the Write-Back Enhanced IntelDX2 processor and the Pentium OverDrive processor support an internal write back processor cache. The Pentium OverDrive processor carries over the Pentium processor cache protocol which supports multiple processors in the same system. The Write-Back Enhanced IntelDX2 processor cache coherency protocol is designed for single processor systems and is a subset of the Pentium OverDrive processor MESI protocol. If implemented correctly, a system can be designed that will support both protocols without sacrificing functionality. The differences of the cache cycles between the two processors are described below in Table B-5.

For any cache read cycle (read hit or line fill), the Pentium OverDrive processor behaves in the same manner as the Write-Back Enhanced IntelDX2 processor.

As detailed in the Table B-5, the Pentium OverDrive processor allows cache transitions on write through cycles (write hits to 'S' state lines). If **PWT** is driven low, the processor will use the state of **WB/WT#** to determine the final state of the entire cache line, even if only part of the line was written out by the write through cycle. If **WB/WT#** is driven HIGH, the state of the line will be changed to the 'E' state. If driven LOW, the line will remain in the 'S' state. This differs from the Write-Back Enhanced IntelDX2 processor and its dedicated 'S' state which always enforces the write through properties of a line stored in the 'S' state. If a system allows regions of memory to be stored as write through only via the **WB/WT#** pin, and other memory regions are stored as write back lines, then **WB/WT#** must toggle properly to

Table B-5. Differences in Cache Cycles

Type of Cycle	Pentium® OverDrive® Processor Line State Transition	Write-Back Enhanced IntelDX2 Processor Line State Transition	Comments
Cache Write Hit to 'S' State Line	'S' to 'E' OR 'S' to 'S'	'S' to 'S'	The Pentium OverDrive processor samples **WB/WT#** on write through cycles if **PWT** is LOW to determine what kind of transition should occur.
External Snoop Hit **INV** = 0	'M' to 'S' 'E' to 'S' 'S' to 'S'	'M' to 'E' 'E' to 'E' 'S' to 'S'	The assumption that a Write-Back Enhanced IntelDX2 processor will operate in a single processor system allows for direct transitions to the 'E' state.

ensure that 'S' state lines are never changed to write back 'E' lines. If the cache is used in a write back mode only (linefills are never stored in the 'S' state), then there will be no cache coherency issues, but there can be a performance issue due to snoop hits with **INV** = LOW.

For external snoop hits into the cache, the Pentium OverDrive processor differs in behavior from the Write-Back Enhanced IntelDX2 processor only in external snoop hits that drive **INV** = LOW. As shown in Table 10-5, any time a snoop hit occurs with **INV** = LOW, the Pentium OverDrive processor will change the cache line state to 'S'. The Write-Back Enhanced IntelDX2 processor will allow lines defined as write back ('M' and 'E') to transition to the initial write back line state of 'E'. If the line started out in the 'S' state, the Write-Back Enhanced IntelDX2 processor will protect the write through status of the line by keeping it in the 'S' state. On the Pentium OverDrive processor, if a system design supports snoop cycles with INV = LOW, this could result in write back lines ('M' or 'E') being stored in a write through state ('S'). This will not cause memory coherency issues, but any time a write cycle to the line is generated, the cycle will be driven to the bus, rather than simply stored in the cache in the 'M' state. To avoid performance issues, systems that can drive **INV** = LOW should drive **WB/WT#** = HIGH when a write through cycle to such a line occurs on the bus. As shown in Table B-5, this will ensure that a write back line that has accidentally been converted to a write through line will not cause more than one unnecessary write cycle on the bus.

STANDARD/ENHANCED BUS DIFFERENCES

Both the Write-Back Enhanced IntelDX2 processor and the Pentium OverDrive processor use the **WB/WT#** pin as an initialization input on the falling edge of **RESET.** Several signals behave differently depending on if the processors are operating in enhanced bus mode or standard bus mode. Some of these behaviors are slightly different between the Pentium OverDrive processor and the Write-Back Enhanced IntelDX2 processor, and are listed below in Table B-6.

Table B-6. Differences in Bus Modes

Functionality	Pentium® OverDrive® Processor	Write-Back Enhanced IntelDX2 Processor
Standard Bus Mode CPUID	0153xh x = 0 to F	0043xh x = 0 to F
Enhanced Bus Mode CPUID	0153xh x = 0 to F	0047xh x = 0 to F
Standard Bus Mode **INIT**	**INIT** is treated as an Interrupt	SRESET is not an interrupt
Standard Bus Mode **FLUSH#**	**FLUSH#** will take about 15 Bus CLKs No Write Backs	**FLUSH#** will take 1 Bus CLK No Write Backs

One item to note in Table B-6 is that the **INIT** pin on the Pentium OverDrive processor is treated as an edge triggered interrupt in both the standard and enhanced bus modes. This means that **INIT** will not be recognized until instruction boundaries, and after **INIT** has been asserted and recognized, more ADS# cycles can be started, even if **INIT** has not been deasserted.

SOFTWARE DIFFERENCES

Due to the Pentium processor features found on the Pentium OverDrive processor, there are several software visible differences that should be accounted for in any system specific firmware or BIOS routines.

Cache Testing and Test Registers

The cache of the Pentium OverDrive processor is structured as separate code and data caches, each 16 KBytes in size. Because of this structure, the Pentium OverDrive processor supports Model Specific Registers (MSR's) for cache testing, rather than the Write-Back Enhanced IntelDX2 processor Test Registers. Any attempt to access the Write-Back Enhanced IntelDX2 processor test registers on the Pentium OverDrive processor will result in invalid opcode exceptions. For more information on testing the caches of the Pentium OverDrive processor, please contact Intel.

Timing Loops

Timing loops (i.e.: executing a tight loop that does nothing) are a common method of providing a software based delay for I/O recovery. Because of the Pentium processor core and an increased core speed, the Pentium OverDrive processor will be able to execute instructions must faster than previous generations of processors. It is suggested that timing loops be avoided, and that a hardware based delay scheme be developed, such as writing to a dummy I/O port that will delay the returning of **RDY#** for a fixed amount of time.

EXTERNAL SNOOPING REQUIREMENTS ON EADS#

The specification set out in the Hardware Design Considerations Section (Sections 10.1.1.1 and 10.1.1.2) must be observed to ensure that EADS# will be recognized properly by both the Pentium OverDrive processor and the Write-Back Enhanced IntelDX2 processor.

DIFFERENCES IN STOP GRANT STATE OPERATION

If the **STPCLK#** pin is asserted and the Pentium OverDrive processor issues the stop grant special cycle, the processor is in the stop grant state. While in this state, the following conditions apply:

1) If any of the following interrupts are asserted, they will be latched and serviced as soon as the **STPCLK#** pin is released and the processor exits the stop grant state:

   FLUSH#, SMI#, NMI, INIT

   If an interrupt is asserted and then released while the processor is in the stop grant state, it will be recognized once the processor exits the stop grant state even though the interrupt may be deasserted. This behavior is different than that of the Write-Back Enhanced IntelDX2 processor, which requires these interrupts to be held until the processor has exited the stop grant state.

   NOTE:
   INTR is a level triggered interrupt and will not be latched. It must be held until the interrupt acknowledge cycle to guarantee recognition.

2) Unlike the Write-Back Enhanced IntelDX2 processor, if **INIT** is asserted while in the stop grant state, the processor will not automatically exit the stop grant state and perform the **INIT.** As mentioned above, **INIT** is latched and will be recognized once the processor exits the stop grant state when **STPCLK#** is deasserted.

FUTURE PENTIUM® OVERDRIVE® PROCESSOR FOR PENTIUM PROCESSOR (510\60, 567\66)-BASED SYSTEMS SOCKET SPECIFICATIONS

1.0 INTRODUCTION

The Future Pentium® OverDrive® processor is an end user single-chip CPU upgrade product for Pentium processor (510\60, 567\66)-based systems. The Future Pentium OverDrive processor will speed up most software applications by 40% to 70%. It is binary compatible with the Pentium processor (510\60, 567\66).

An upgrade socket (Socket 4) has been defined along with the Pentium processor (510\60, 567\66) as part of the processor architecture. The Future Pentium OverDrive processor will be socket compatible with the Pentium processor (510\60, 567\66). The Future Pentium OverDrive processor is packaged in a 273-pin ceramic pin grid array package with an attached fan/heatsink present on the turbo upgrade processor component.

Execution tracing is not supported in the Future Pentium OverDrive processor, and performance monitoring is implemented differently than in the Pentium processor (510\60, 567\66).

1.1 Upgrade Objectives

Systems using the Pentium processor (510\60, 567\66) must use Socket 4 to also accept the Future Pentium OverDrive processor. Inclusion of upgrade Socket 4 in Pentium processor (510\60, 567\66) systems provides the end user with an easy and cost effective way to increase system performance. The process of simply installing an upgrade component into an easy to use Zero Insertion Force (ZIF) socket to achieve enhanced system performance is familiar to the millions of end users and dealers who have purchased Intel Math CoProcessor upgrades to boost system floating-point performance.

Inclusion of Socket 4 in Pentium processor (510\60, 567\66) systems provides the end-user with an easy and cost-effective way to increase system performance. The paradigm of simply installing an additional component into an easy to use Zero Insertion Force (ZIF) Socket to achieve enhanced system performance is familiar to the millions of end-users and dealers who have purchased Intel math coprocessor upgrades to boost system floating point performance.

The majority of upgrade installations which take advantage of Socket 4 will be performed by end users and resellers. Therefore, it is important that the design be "end user easy," and that the amount of training and technical expertise required to install the upgrade processors be minimized. Upgrade installation instructions should be clearly described in the system user's manual. In addition, by making installation simple and foolproof, PC manufacturers can reduce the risk of system damage, warranty claims and service calls.

November 1995
Order Number: 290484-004

Feedback from Intel's Math CoProcessor upgrade customers highlights three main characteristics of end user easy designs:

- accessible socket location
- clear indication of upgrade component orientation
- minimization of insertion force

The Future Pentium OverDrive processor will support the Intel 82430 PCIset. Unlike the Pentium processor (510\60, 567\66), the Future Pentium OverDrive processor will not support the 82496 Cache Controller and 82491 Cache SRAM chip set.

1.2 Intel Verification Program

The Intel Verification Program ensures that a Pentium processor (510\60, 567\66)-based personal computer meets a minimum set of design criteria for reliable and straightforward CPU upgradability with a Future Pentium OverDrive processor. Testing performed at the Intel Verification Labs confirms that future Pentium OverDrive processor specifications for mechanical, thermal, electrical, functional, and end-user installation attributes have been met. While system designs may exceed these minimum design criteria, the intent is to provide end-users with confidence that computer systems based on verified designs can be upgraded with Futute Pentium OverDrive processors.

The OEM submits production-ready designs to one of Intel's worldwide Verification Labs for testing. The OEM benefits from advance testing of the design prior to availability of the Future Pentium OverDrive processor. By identifying and resolving upgradability problems before a system is introduced, the OEM increases system quality and reduces future support costs associated with end-user calls and complications when the CPU upgrade is ultimately installed.

Contact your local Intel representative for more information on the Intel Verification Program for Pentium processor (510\60, 567\66)-based systems.

2.0 Future Pentium® OverDrive® Processor Socket

The following drawings in Figure 1 show the preliminary worst case socket footprints from two qualified Socket 4 vendors, AMP and Yamaichi. OEMs should work directly with socket vendors for the most current socket information.

To order Socket 4 from AMP and Yamaichi, the phone numbers and part numbers are:

AMP:	1-800-522-6752	Part#: 916510-1
Yamaichi:	1-800-769-0797	Part#: NP11J-273K13221

Figure 2 shows the Future Pentium OverDrive processor chip's orientation in the Socket 4.

For a complete list of Qualified Sockets and Vendor Order Numbers, call the Intel Faxback number for your geographical area and have document #7209 automatically faxed to you.

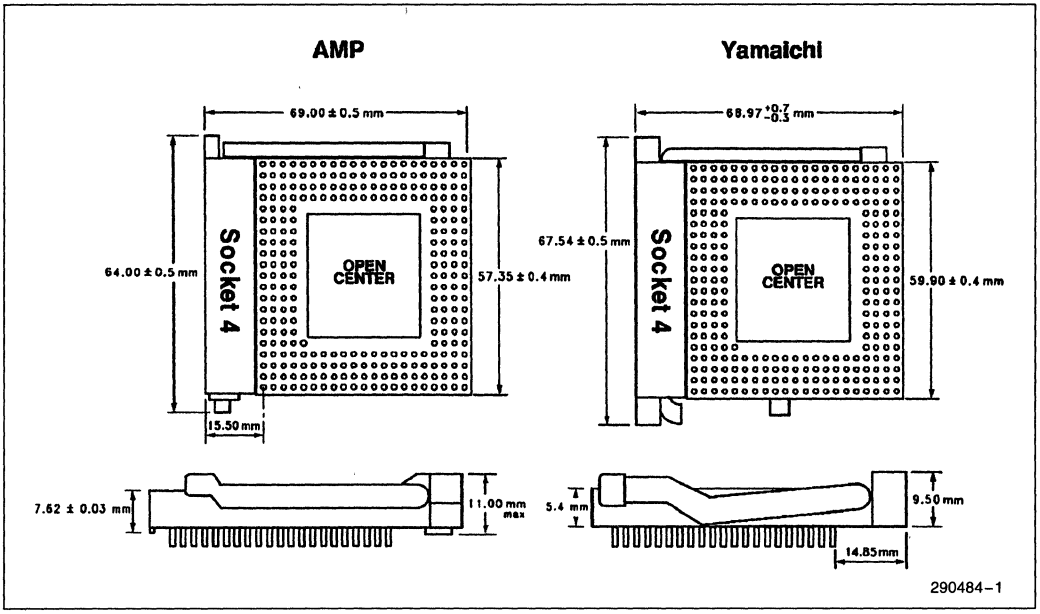


Figure 1. Socket 4 Footprint Dimensions (See socket manufacturer for the most current information.)

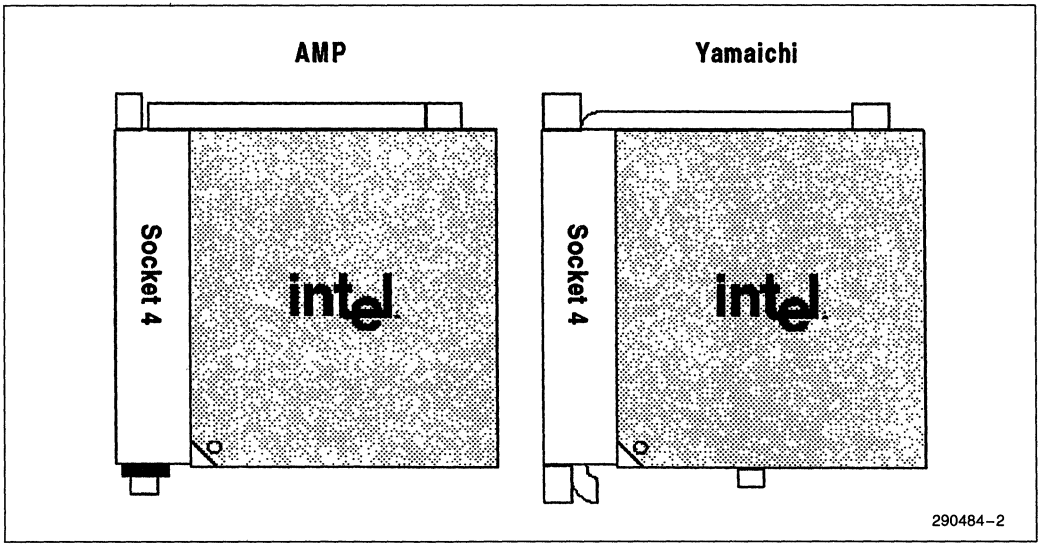


Figure 2. Chip Orientation in Socket 4

3.0 SOCKET 4 PINOUT

The Future Pentium OverDrive processor pinout is identical to that of the Pentium processor (510\60, 567\66). Note that all input pins must meet their AC/DC specifications to guarantee proper functional behavior. Figure 3 shows the Socket 4 pinout.

Locations E17, S17 and S5 should be plugged on Socket 4 in order to ensure that the Pentium processor (510\60, 567\66) or OverDrive processor chip is installed in the socket with the correct orientation.

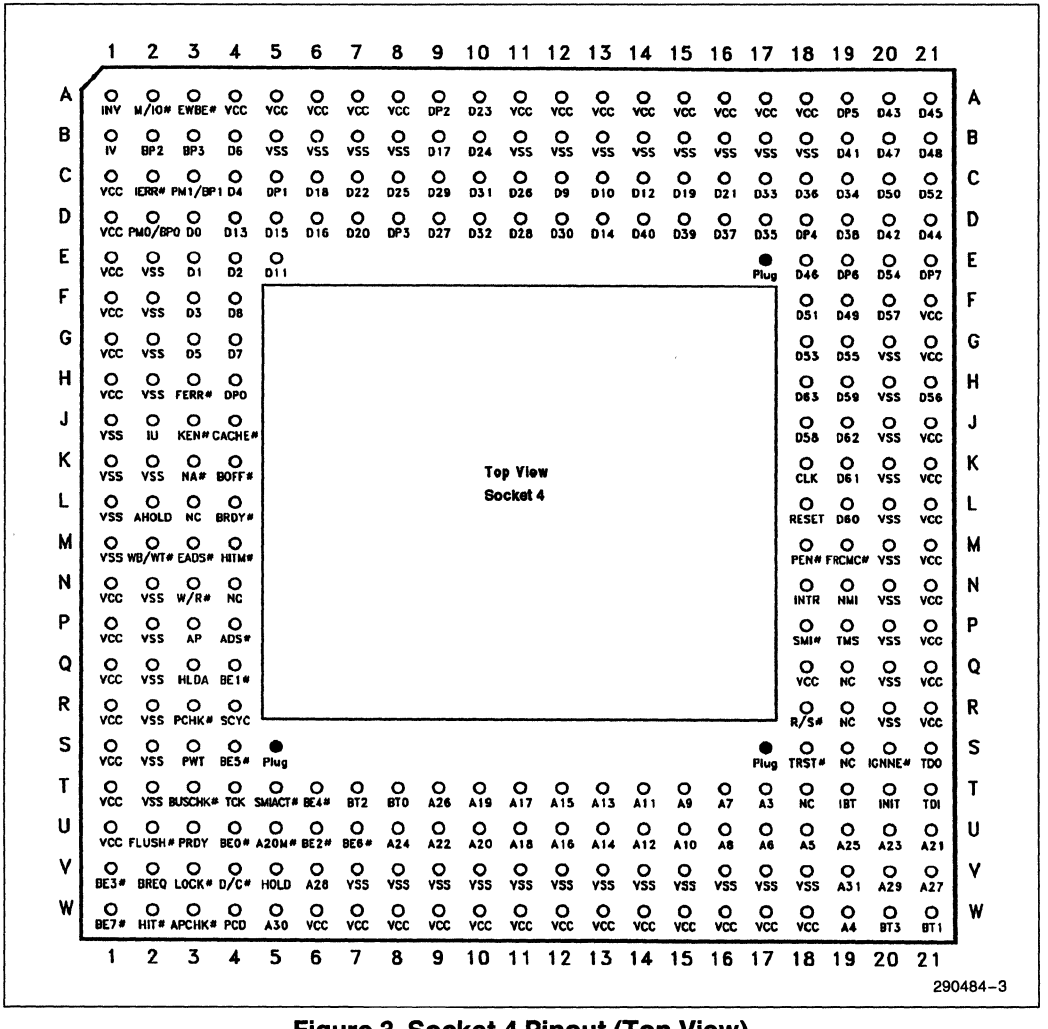


Figure 3. Socket 4 Pinout (Top View)

4.0 ELECTRICAL SPECIFICATIONS

The Future Pentium OverDrive processor will have the same power and ground specifications, decoupling recommendations, connection specifications and maximum ratings as the Pentium processor (510\60, 567\66).

4.1 Absolute Maximum Ratings For Upgrade

The on-chip voltage regulation and fan/heatsink devices included on the Future Pentium OverDrive processor require different stress ratings than the Pentium processor (510\60, 567\66). The voltage regulator is surface mounted on the Future Pentium OverDrive processor and is, therefore, an integral part of the assembly. The Future Pentium OverDrive processor storage temperature ratings area tightened as a result. The fan is a detachable unit, and the storage temperature is stated separately in the table below. Functional operation of the Future Pentium OverDrive processor remains 0°C to 70°C.

Table 1. Absolute Maximum Ratings

Future Pentium® OverDrive® Processor and Voltage Regulator Assembly:					
	Parameter	**Min**	**Max**	**Unit**	**Notes**
	Storage Temperature	−30	100	°C	
	Case Temperature Under Bias	−30	100	°C	
Fan:					
	Parameter	**Min**	**Max**	**Unit**	**Notes**
	Storage Temperature	−30	75	°C	
	Case Temperature Under Bias	−30	75	°C	

WARNING:
Stressing the devices beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

4.2 DC Specifications

The Future Pentium OverDrive processor will have the same DC specifications as the Pentium processor (510\60, 567\66), including I_{CC} (Power Supply Current) as shown in Table 2.

Table 2. OverDrive® Processor I_{CC} Specifications(2)

Symbol	**Parameter**	**Min**	**Max**	**Unit**	**Notes**
I_{CC}	Power Supply Current		3200 2910	mA mA	66 MHz(1) 60 MHz(1)

NOTES:
1. Worst case average I_{CC} for a mix of test patterns. (The mix of test patterns will be determined after silicon is examined.)
2. Refer to Pentium processor at iCOMP index 510\60 MHz, and Pentium processor at iCOMP index 567\66 MHz datasheet for remaining Pentium processor (510\60, 567\66) DC and V_{CC} specifications.

Refer to Pentium processor at iCOMP index 510\60 MHz, and Pentium processor at iCOMP index 567\66 MHz datasheets for a listing of the remaining DC specifications.

4.3 AC Specifications

The Future Pentium OverDrive processor will have the same AC specifications as the Pentium processor (510\60, 567\66). Refer to Pentium processor at iCOMP index 510\60 MHz, and Pentium processor at iCOMP index 567\66 MHz datasheets for a listing of the AC specifications. The functional parameters for the Future Pentium OverDrive processor's AC specifications are the same as those for Pentium processor (510\60, 567\66) except T_{SINK} as shown below:

$$T_{SINK} = 0°C \text{ to } + 70°C$$

5.0 MECHANICAL SPECIFICATIONS

The Future Pentium OverDrive processor for Pentium processor (510\60, 567\66)-based systems is packaged in a 273-pin ceramic pin grid array (PGA) with attached fan/heatsink. The pins are arranged in a 21 x 21 matrix and the package dimensions will be 2.16″ x 2.16″ (5.49 cm x 5.49 cm). See Table 3.

Table 3. OverDrive® Processor Package Information Summary

Package Type	Total Pins	Pin Array	Package Size
PGA	273	21 x 21	2.16″ x 2.16″ (5.49 cm x 5.49 cm)

NOTE:
See DC Specifications for more detailed power specifications.

As can be seen in the mechanical dimensions in Table 4 and Figure 4, the actual height required by the heatsink and fan is less than the total space allotted. Since the Future Pentium OverDrive processor for Pentium processor (510\60, 567\66)-based systems employs a fan/heatsink, a certain amount of space is required above the fan/heatsink unit to ensure that the airflow is not blocked. Figure 5 shows unacceptable blocking of the airflow for the Future Pentium OverDrive processor fan/heatsink. Figure 6 details the minimum space needed around the PGA package to ensure proper heatsink airflow.

As shown in Figure 6, it is acceptable to allow any device (i.e., add-in cards, surface mount device, chassis, etc.) to enter within the free space distance of 0.2″ from the PGA package if it is not taller than the level of the heatsink base. In other words, if a component is taller than height "B," it cannot be closer to the PGA package than distance "A." This applies to three of the four sides of the PGA package, although the back and handle sides of a ZIF socket will generally automatically meet this specification since they have widths larger than distance "A."

Table 4. OverDrive® Processor Mechanical Specifications

Family: Ceramic Pin Grid Array Package						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
A		33.98	Solid Lid		1.338	Solid Lid
A1	2.84	3.50	Solid Lid	0.112	0.138	Solid Lid
A2	0.33	0.43	Solid Lid	0.013	0.017	Solid Lid
A3	2.51	3.07		0.099	0.121	
A4		20.32			0.800	
A5	10.16			0.400		
B	0.43	0.51		0.017	0.020	
D	54.61	55.11		2.150	2.170	
D1	50.67	50.93		1.995	2.005	
E1	2.29	2.79		0.090	0.110	
L	3.05	3.30		0.120	0.130	
N	273			273		
S1	1.65	2.16		0.065	0.085	

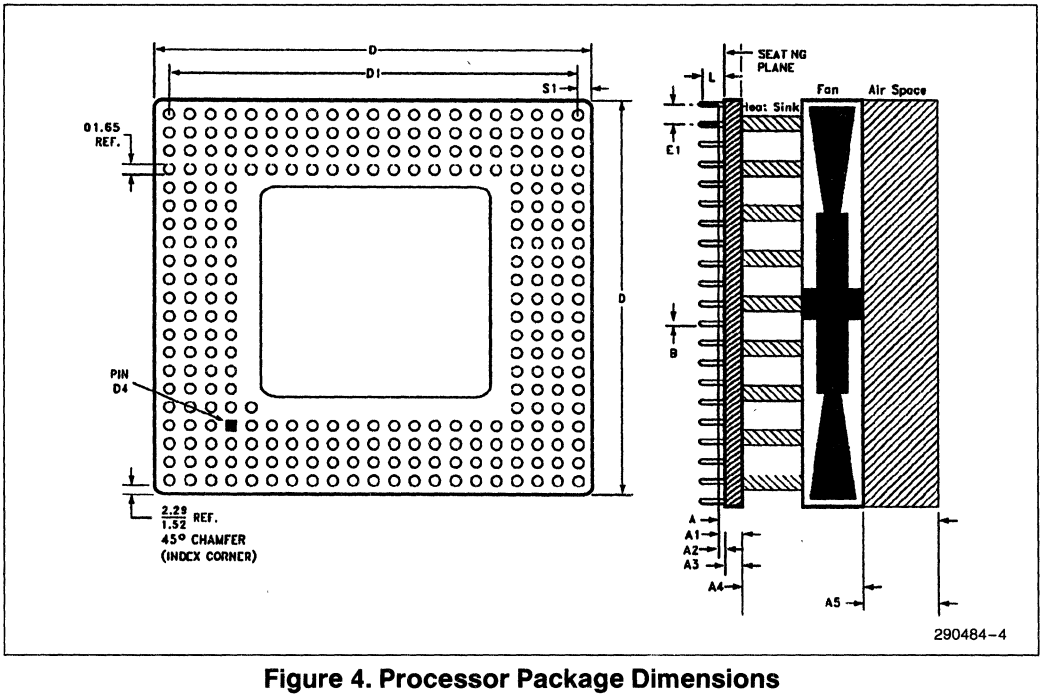


Figure 4. Processor Package Dimensions

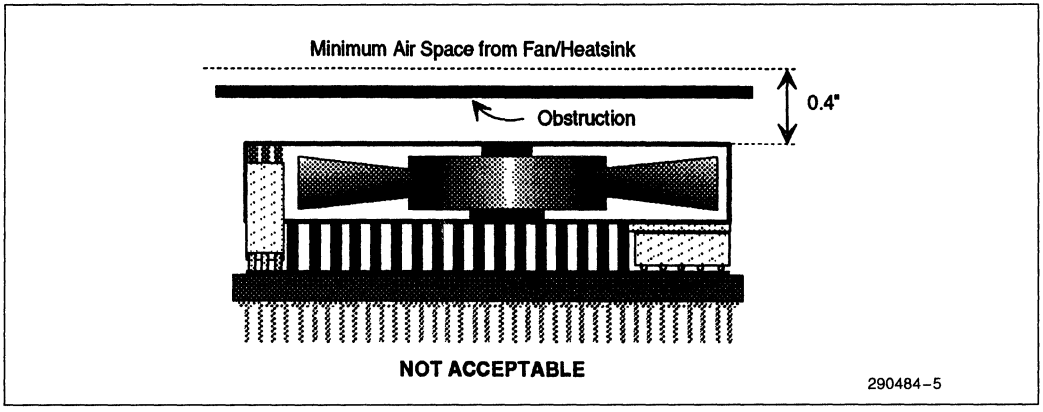


Figure 5. Fan/Heatsink Top Space Requirements

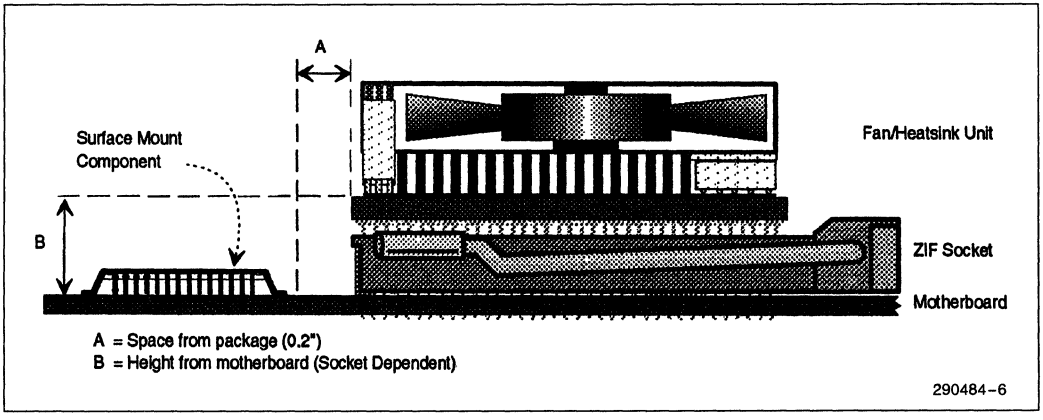


Figure 6. Required Free Space from Sides of PGA Package

6.0 THERMAL SPECIFICATIONS

The fan/heatsink cooling solution will properly cool the Future Pentium OverDrive processor as long as the maximum air temperature entering the fan/heatsink cooling solution (T_A(In)) does not exceed 45°C. It is left up to the OEM to ensure that T_A(In) meets this specification by providing sufficient airflow around the Future Pentium OverDrive processor heatsink unit.

Intel's fan/heatsink will dissipate approximately 1W and is powered by the chip such that no external wires or connections are required. The extra power needed for the fan/heatsink is taken into account in the I_{CC} numbers of the processor. Additionally, Intel is evaluating the feasibility of having the Future Pentium OverDrive processor monitor its temperature. No BIOS or hardware changes will be needed for this thermal protection mechanism. The shutdown temperature will be greater than the maximum temperature specification of the processor. The fan/heatsink unit will be designed to be removable so that if fan failure should occur, the unit may be easily replaced. Figure 7 gives a functional representation of the processor and fan/heatsink unit. The actual fan/heatsink unit may be different from the one shown in the figure.

Since the Future Pentium OverDrive processor for Pentium processor (510\60, 567\66)-based systems employs a fan/heatsink, it is not as important that the processor heatsink receive direct airflow, rather that the system has sufficient capability to remove the warm air that the Future Pentium OverDrive processor will generate. This implies that enough airflow exists at the Socket 4 to keep localized heating from occurring. This can be accomplished by a standard power supply fan with a clear path to the processor. Figure 8 shows how system design can cause localized heating to occur by limiting the airflow in the area of the processor. The airflow supplied in the system should also be enough to ensure that the OEM processor shipped with the system will meet the OEM processor thermal specifications before the system is upgraded with the Future Pentium OverDrive processor.

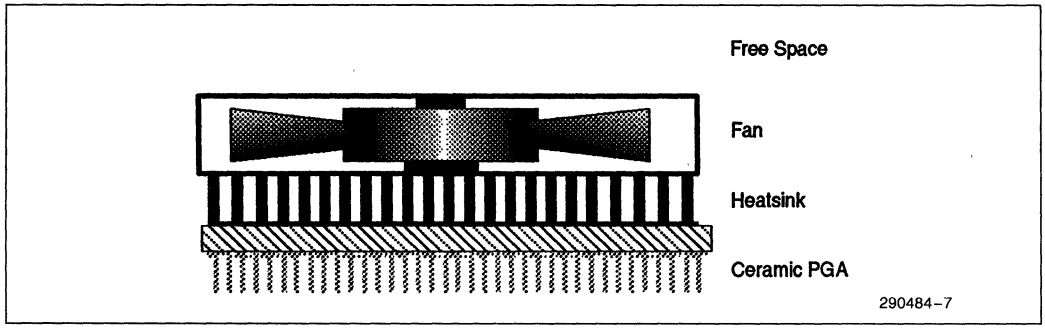


Figure 7. Fan/Heatsink Example

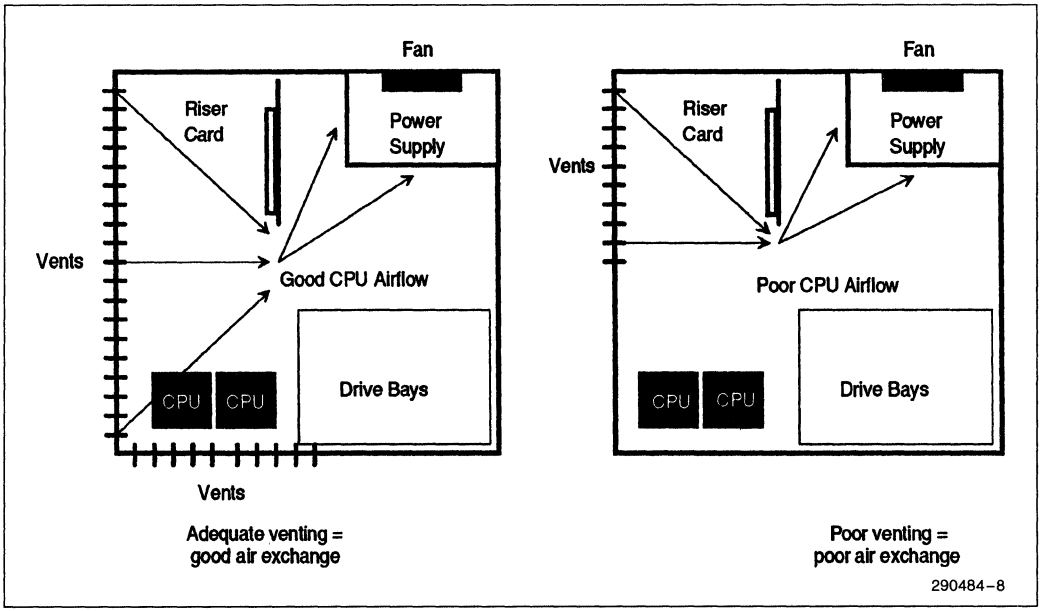


Figure 8. Airflow Design Examples

7.0 TESTABILITY

7.1 Boundary Scan

The Future Pentium OverDrive processor supports the IEEE Standard 1149.1 boundary scan using the Test Access Port (TAP) and TAP Controller. The boundary scan register for the Future Pentium OverDrive processor contains a cell for each pin. The turbo upgrade component will have a different bit order than the Pentium processor (510\60, 567\66). If the TAP port on your system will be used by an end user following installation of the Future Pentium OverDrive processor, please contact Intel for the bit order of the upgrade processor boundary scan register.

Pentium® PROCESSOR REFERENCE SHEET

For more information regarding the Pentium® Processor, you may obtain the *Pentium® Processor Family Developer's Manual* by calling our toll-free literature distribution center at 1-800-548-4725. The *Pentium® Processor Family Developer's Manual* is a three-volume set with details on Pentium Processor hardware and software design parameters, with specifications also for the 82496 Cache Controller/82491 Cache SRAM, and 82497 Cache Controller/82492 Cache SRAM chipsets.

Also listed in this chapter are brief technical profiles of other Pentium Processor peripheral products, including the 82489DX Advanced Interrupt Controller and the 82430 PCIset. For more information on these products, please consult the 1995 Peripherals Handbook.

241815-1

November 1995
Order Number: 241815-002

Pentium® PROCESSORS AT iCOMP® INDEX 735\90, 610\75 MHz WITH VOLTAGE REDUCTION TECHNOLOGY

SmartDie™ Product Specification

- **Compatible with Large Software Base**
  - **MS-DOS*, Windows*, OS/2*, UNIX***
- **32-Bit CPU with 64-Bit Data Bus**
- **Superscalar Architecture**
  - **Two Pipelined Integer Units Are Capable of Two Instructions Per Clock**
  - **Pipelined Floating Point Unit**
- **Separate Code and Data Caches**
  - **8 Kbyte Code, 8 Kbyte Writeback Data**
  - **MESI Cache Protocol**
- **Advanced Design Features**
  - **Branch Prediction**
  - **Virtual Mode Extensions**
- **Low Voltage BiCMOS Silicon Technology**
- **4 Mbyte Pages for Increased TLB Hit Rate**
- **IEEE 1149.1 Boundary Scan**
- **Internal Error Detection Features**
- **SL Enhanced Power Management Features**
  - **System Management Mode**
  - **Clock Control**
- **Voltage Reduction Technology**
  - **2.9V V_{CC} for Core Supply**
  - **3.3V V_{CC} for I/O Buffer Supply**
- **Fractional Bus Operation**
  - **75 MHz Core / 50 MHz Bus**
  - **90 MHz Core / 60 MHz Bus**
- **Intel SmartDie Product**
  - **Full AC/DC Testing at Die Level**
  - **0°C to 105°C (Junction) Temperature Range**

2

NOTICE: This document contains preliminary information on new products in production. It is valid for the devices indicated in the revision history. This specification is subject to change without notice. Verify with your local Intel sales office that you have the latest product specification before finalizing a design.

REFERENCE INFORMATION: The information in this document is provided as a supplement to the Standard Package Data Sheet on a specific product. Please reference the Standard Package Data Sheet/Book (Order No. 242557) for additional product information and specifications not found in this document.

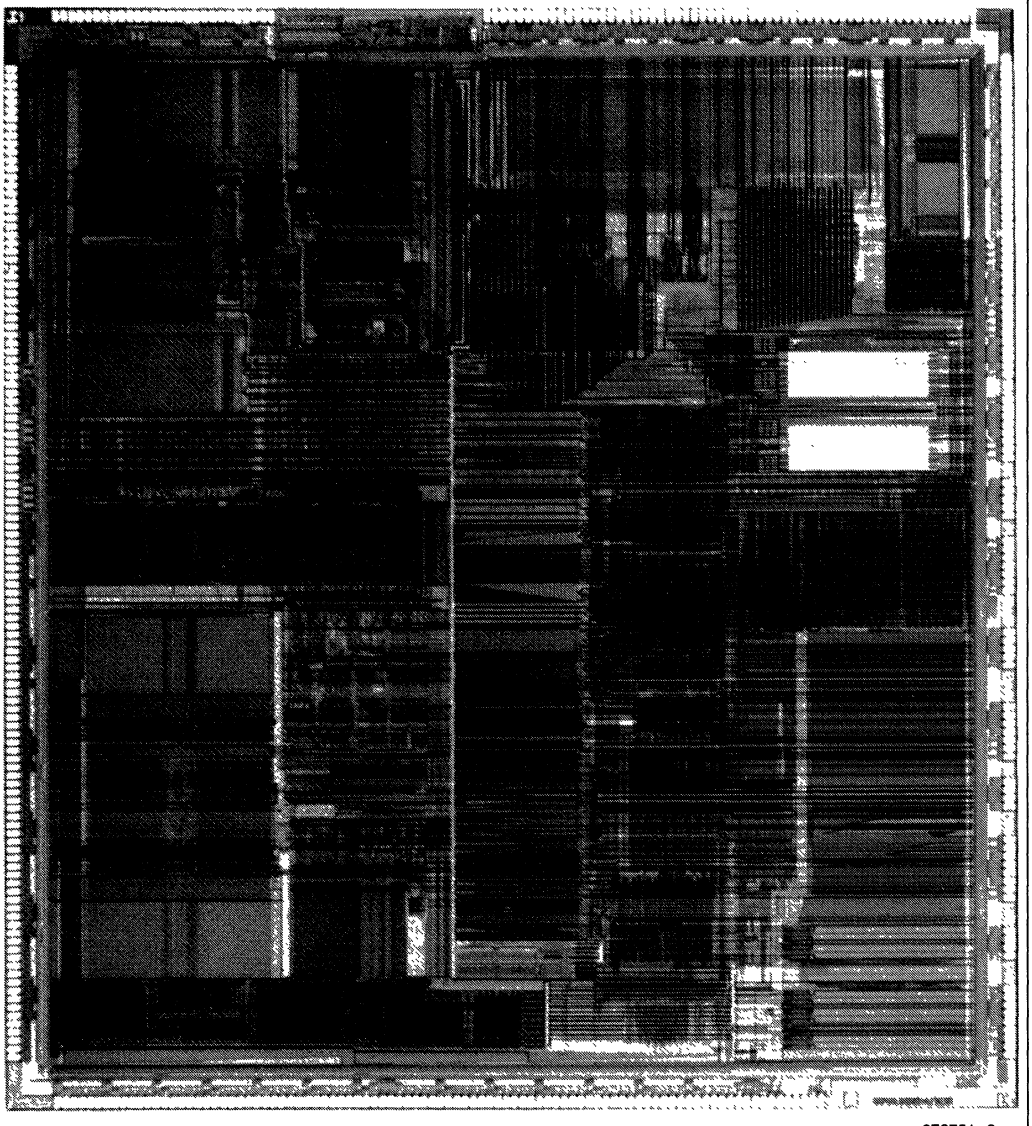
The Pentium processor is fully compatible with the entire installed base of applications for DOS, Windows*, OS/2*, and UNIX*, and all other software that runs on any earlier Intel 8086 family product. The Pentium processor's superscalar architecture can execute two instructions per clock cycle. Branch prediction and separate caches also increase performance. The pipelined floating-point unit delivers workstation level performance. Separate code and data caches reduce cache conflicts while remaining software transparent. The Pentium processor with voltage reduction has 3.3 million transistors. It is built on Intel's advanced low voltage BiCMOS silicon technology, and has full SL Enhanced power management features, including System Management Mode (SMM) and clock control. The additional SL Enhanced features, 2.9V core operation along with 3.3V I/O buffer operation, and the availability of a SmartDie product version, which are not available in the Pentium processor (510\60, 567\66), make the SmartDie product Pentium processor with voltage reduction technology ideal for enabling mobile Pentium processor designs.

December 1995
Order Number: 272751-002

Pentium® PROCESSORS AT iCOMP® INDEX 735/90, 610/ 75 MHz WITH VOLTAGE REDUCTION TECHNOLOGY

SmartDie™ Product Specification

CONTENTS



272751-3

Figure 1. Pentium® Processor Die Photo

1.0 DIE SPECIFICATIONS

The plot and the die photo on the opposite page indicate the orientation of the die in the GEL-PAK* (shipping container). Die are aligned as shown relative to a 45° notch which is in one corner of the GEL-PAK. An Intel internal manufacturing name 80P54LM appears on Pentium processor die.

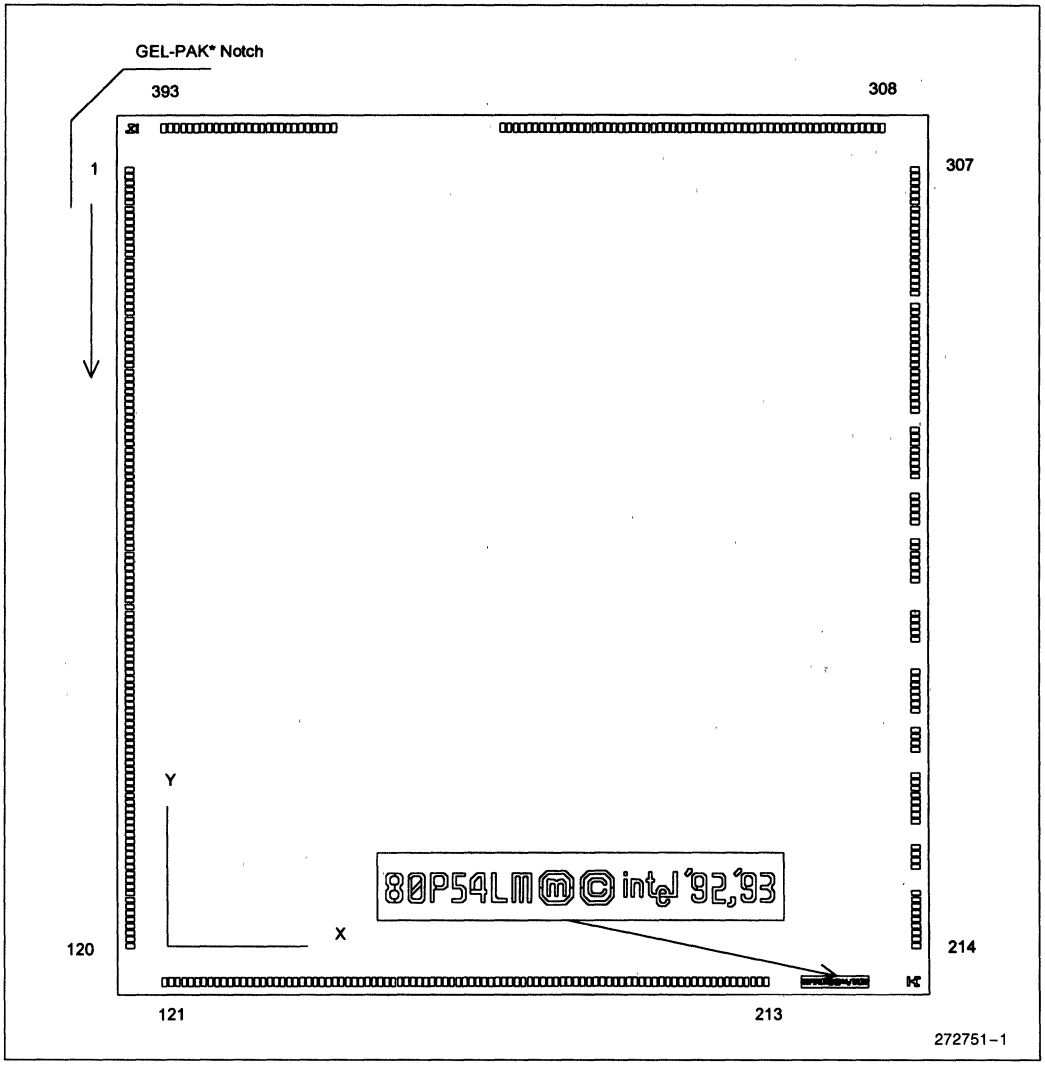


Figure 2. Pentium® Processor Die/Bond Pad Layout

1.1 Pad Description

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 1 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
001	D/C#	−224.8	221.0	−5710	5614
002	PWT	−224.8	217.3	−5710	5519
003	PCD	−224.8	213.6	−5710	5424
004	V_{CC}_3.3	−224.8	209.8	−5710	5329
005	V_{SS}	−224.8	206.1	−5710	5234
006	N.C.	−224.8	202.3	−5710	5139
007	LOCK#	−224.8	198.6	−5710	5044
008	V_{SS}	−224.8	194.9	−5710	4949
009	V_{CC}_2.9	−224.8	191.1	−5710	4854
010	V_{CC}_2.9	−224.8	187.4	−5710	4759
011	V_{SS}	−224.8	183.6	−5710	4664
012	V_{CC}_3.3	−224.8	179.9	−5710	4569
013	V_{SS}	−224.8	176.2	−5710	4474
014	AP	−224.8	172.4	−5710	4379
015	V_{SS}	−224.8	168.7	−5710	4284
016	V_{CC}_2.9	−224.8	164.9	−5710	4189
017	V_{CC}_2.9	−224.8	161.2	−5710	4094
018	V_{SS}	−224.8	157.5	−5710	3999
019	HLDA	−224.8	153.7	−5710	3904
020	BREQ	−224.8	150.0	−5710	3809
021	V_{CC}_3.3	−224.8	146.2	−5710	3714
022	V_{SS}	−224.8	142.5	−5710	3619
023	APCHK#	−224.8	138.8	−5710	3524
024	PCHK#	−224.8	135.0	−5710	3429
025	PRDY	−224.8	131.3	−5710	3334
026	SMIACT#	−224.8	127.5	−5710	3239
027	V_{SS}	−224.8	123.8	−5710	3144
028	V_{CC}_2.9	−224.8	120.1	−5710	3049
029	V_{CC}_2.9	−224.8	116.3	−5710	2954
030	V_{SS}	−224.8	112.6	−5710	2859
031	N.C.	−224.8	108.8	−5710	2764
032	V_{CC}_3.3	−224.8	105.1	−5710	2669
033	V_{SS}	−224.8	101.4	−5710	2574
034	N.C.	−224.8	97.6	−5710	2479
035	N.C.	−224.8	93.9	−5710	2384

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 2 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
036	N.C.	−224.8	90.1	−5710	2289
037	HOLD	−224.8	86.4	−5710	2194
038	WB/WT#	−224.8	82.6	−5710	2099
039	V_{SS}	−224.8	78.9	−5710	2004
040	V_{CC}_2.9	−224.8	75.2	−5710	1909
041	V_{CC}_2.9	−224.8	71.4	−5710	1814
042	V_{SS}	−224.8	67.7	−5710	1719
043	NA#	−224.8	63.9	−5710	1624
044	BOFF#	−224.8	60.2	−5710	1529
045	BRDY#	−224.8	56.5	−5710	1434
046	N.C.	−224.8	52.7	−5710	1339
047	V_{SS}	−224.8	49.0	−5710	1244
048	V_{CC}_2.9	−224.8	45.2	−5710	1149
049	V_{CC}_2.9	−224.8	41.5	−5710	1054
050	V_{SS}	−224.8	37.8	−5710	959
051	KEN#	−224.8	34.0	−5710	864
052	AHOLD	−224.8	30.3	−5710	769
053	INV	−224.8	26.5	−5710	674
054	EWBE#	−224.8	22.8	−5710	579
055	V_{SS}	−224.8	19.1	−5710	484
056	V_{CC}_2.9	−224.8	15.3	−5710	389
057	V_{CC}_2.9	−224.8	11.6	−5710	294
058	V_{SS}	−224.8	7.8	−5710	199
059	V_{CC}_3.3	−224.8	4.1	−5710	104
060	V_{SS}	−224.8	0.4	−5710	9
061	CACHE#	−224.8	−3.4	−5710	−86
062	M/IO#	−224.8	−7.1	−5710	−181
063	V_{CC}_3.3	−224.8	−10.9	−5710	−276
064	V_{SS}	−224.8	−14.6	−5710	−371
065	BP3	−224.8	−18.3	−5710	−466
066	BP2	−224.8	−22.1	−5710	−561
067	PM1/BP1	−224.8	−25.8	−5710	−656
068	PM0/BP0	−224.8	−29.6	−5710	−751
069	FERR#	−224.8	−33.3	−5710	−846
070	V_{SS}	−224.8	−37.0	−5710	−941
071	V_{CC}_2.9	−224.8	−40.8	−5710	−1036
072	V_{CC}_2.9	−224.8	−44.5	−5710	−1131

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 3 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
073	V_{SS}	−224.8	−48.3	−5710	−1226
074	IERR#	−224.8	−52.0	−5710	−1321
075	V_{CC}_3.3	−224.8	−55.7	−5710	−1416
076	V_{SS}	−224.8	−59.5	−5710	−1511
077	DP7	−224.8	−63.2	−5710	−1606
078	D63	−224.8	−67.0	−5710	−1701
079	D62	−224.8	−70.7	−5710	−1796
080	D61	−224.8	−74.4	−5710	−1891
081	V_{SS}	−224.8	−78.2	−5710	−1986
082	V_{CC}_2.9	−224.8	−81.9	−5710	−2081
083	V_{CC}_2.9	−224.8	−85.7	−5710	−2176
084	V_{SS}	−224.8	−89.4	−5710	−2271
085	V_{CC}_3.3	−224.8	−93.1	−5710	−2366
086	V_{SS}	−224.8	−96.9	−5710	−2461
087	D60	−224.8	−100.6	−5710	−2556
088	D59	−224.8	−104.4	−5710	−2651
089	D58	−224.8	−108.1	−5710	−2746
090	D57	−224.8	−111.8	−5710	−2841
091	V_{SS}	−224.8	−115.6	−5710	−2936
092	V_{CC}_2.9	−224.8	−119.3	−5710	−3031
093	V_{CC}_2.9	−224.8	−123.1	−5710	−3126
094	V_{SS}	−224.8	−126.8	−5710	−3221
095	V_{CC}_3.3	−224.8	−130.5	−5710	−3316
096	V_{SS}	−224.8	−134.3	−5710	−3411
097	D56	−224.8	−138.0	−5710	−3506
098	DP6	−224.8	−141.8	−5710	−3601
099	D55	−224.8	−145.5	−5710	−3696
100	D54	−224.8	−149.2	−5710	−3791
101	V_{SS}	−224.8	−153.0	−5710	−3886
102	V_{CC}_2.9	−224.8	−156.7	−5710	−3981
103	V_{CC}_2.9	−224.8	−160.5	−5710	−4076
104	V_{SS}	−224.8	−164.2	−5710	−4171
105	V_{CC}_3.3	−224.8	−167.9	−5710	−4266
106	V_{SS}	−224.8	−171.7	−5710	−4361
107	D53	−224.8	−175.4	−5710	−4456
108	D52	−224.8	−179.2	−5710	−4551
109	D51	−224.8	−182.9	−5710	−4646

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 4 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
110	D50	−224.8	−186.6	−5710	−4741
111	V_{SS}	−224.8	−190.4	−5710	−4836
112	V_{CC}_2.9	−224.8	−194.1	−5710	−4931
113	V_{CC}_2.9	−224.8	−197.9	−5710	−5026
114	V_{SS}	−224.8	−201.6	−5710	−5121
115	V_{CC}_3.3	−224.8	−205.3	−5710	−5216
116	V_{SS}	−224.8	−209.1	−5710	−5311
117	D49	−224.8	−212.8	−5710	−5406
118	D48	−224.8	−216.6	−5710	−5501
119	DP5	−224.8	−220.3	−5710	−5596
120	D47	−224.8	−224.0	−5710	−5691
121	V_{CC}_3.3	−205.3	−244.8	−5214	−6217
122	V_{SS}	−201.5	−244.8	−5119	−6217
123	D46	−197.8	−244.8	−5024	−6217
124	D45	−194.1	−244.8	−4929	−6217
125	D44	−190.3	−244.8	−4834	−6217
126	D43	−186.6	−244.8	−4739	−6217
127	V_{CC}_3.3	−182.8	−244.8	−4644	−6217
128	V_{SS}	−179.1	−244.8	−4549	−6217
129	D42	−175.4	−244.8	−4454	−6217
130	D41	−171.6	−244.8	−4359	−6217
131	D40	−167.9	−244.8	−4264	−6217
132	DP4	−164.1	−244.8	−4169	−6217
133	V_{CC}_3.3	−160.4	−244.8	−4074	−6217
134	V_{SS}	−156.7	−244.8	−3979	−6217
135	D39	−152.9	−244.8	−3884	−6217
136	D38	−149.2	−244.8	−3789	−6217
137	D37	−145.4	−244.8	−3694	−6217
138	D36	−141.7	−244.8	−3599	−6217
139	V_{CC}_3.3	−138.0	−244.8	−3504	−6217
140	V_{SS}	−134.2	−244.8	−3409	−6217
141	D35	−130.5	−244.8	−3314	−6217
142	D34	−126.7	−244.8	−3219	−6217
143	D33	−123.0	−244.8	−3124	−6217
144	D32	−119.3	−244.8	−3029	−6217
145	V_{CC}_3.3	−115.5	−244.8	−2934	−6217
146	V_{SS}	−111.8	−244.8	−2839	−6217

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 5 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
147	DP3	−108.0	−244.8	−2744	−6217
148	D31	−104.3	−244.8	−2649	−6217
149	D30	−100.6	−244.8	−2554	−6217
150	D29	−96.8	−244.8	−2459	−6217
151	V_{CC}_3.3	−93.1	−244.8	−2364	−6217
152	V_{SS}	−89.3	−244.8	−2269	−6217
153	D28	−85.6	−244.8	−2174	−6217
154	D27	−81.9	−244.8	−2079	−6217
155	D26	−78.1	−244.8	−1984	−6217
156	D25	−74.4	−244.8	−1889	−6217
157	V_{CC}_3.3	−70.6	−244.8	−1794	−6217
158	V_{SS}	−66.9	−244.8	−1699	−6217
159	V_{SS}	−63.2	−244.8	−1604	−6217
160	V_{CC}_2.9	−59.4	−244.8	−1509	−6217
161	V_{CC}_2.9	−55.7	−244.8	−1414	−6217
162	V_{SS}	−51.9	−244.8	−1319	−6217
163	D24	−48.2	−244.8	−1224	−6217
164	DP2	−44.5	−244.8	−1129	−6217
165	D23	−40.7	−244.8	−1034	−6217
166	D22	−37.0	−244.8	−939	−6217
167	V_{CC}_3.3	−33.2	−244.8	−844	−6217
168	V_{SS}	−29.5	−244.8	−749	−6217
169	D21	−25.8	−244.8	−654	−6217
170	D20	−22.0	−244.8	−559	−6217
171	D19	−18.3	−244.8	−464	−6217
172	D18	−14.5	−244.8	−369	−6217
173	V_{CC}_3.3	−10.8	−244.8	−274	−6217
174	V_{SS}	−7.1	−244.8	−179	−6217
175	D17	−3.3	−244.8	−84	−6217
176	D16	0.4	−244.8	11	−6217
177	DP1	4.2	−244.8	106	−6217
178	D15	7.9	−244.8	201	−6217
179	V_{CC}_3.3	11.6	−244.8	296	−6217
180	V_{SS}	15.4	−244.8	391	−6217
181	D14	19.1	−244.8	486	−6217
182	D13	22.9	−244.8	581	−6217
183	D12	26.6	−244.8	676	−6217

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 6 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (= .001 inch)		Microns	
		X	Y	X	Y
184	D11	30.3	−244.8	771	−6217
185	V_{CC}_3.3	34.1	−244.8	866	−6217
186	V_{SS}	37.8	−244.8	961	−6217
187	D10	41.6	−244.8	1056	−6217
188	D9	45.3	−244.8	1151	−6217
189	D8	49.0	−244.8	1246	−6217
190	DP0	52.8	−244.8	1341	−6217
191	V_{CC}_3.3	56.5	−244.8	1436	−6217
192	V_{SS}	60.3	−244.8	1531	−6217
193	D7	64.0	−244.8	1626	−6217
194	D6	67.7	−244.8	1721	−6217
195	D5	71.5	−244.8	1816	−6217
196	D4	75.2	−244.8	1911	−6217
197	V_{CC}_3.3	79.0	−244.8	2006	−6217
198	V_{SS}	82.7	−244.8	2101	−6217
199	D3	86.4	−244.8	2196	−6217
200	D2	90.2	−244.8	2291	−6217
201	D1	93.9	−244.8	2386	−6217
202	D0	97.7	−244.8	2481	−6217
203	V_{SS}	101.4	−244.8	2576	−6217
204	V_{CC}_2.9	105.2	−244.8	2671	−6217
205	V_{CC}_2.9	108.9	−244.8	2766	−6217
206	V_{SS}	112.6	−244.8	2861	−6217
207	N.C.	116.4	−244.8	2956	−6217
208	N.C.	120.1	−244.8	3051	−6217
209	N.C.	123.9	−244.8	3146	−6217
210	N.C.	127.6	−244.8	3241	−6217
211	V_{CC}_3.3	131.3	−244.8	3336	−6217
212	V_{SS}	135.1	−244.8	3431	−6217
213	V_{CC}_3.3	138.8	−244.8	3526	−6217
214	TCK	224.8	−224.0	5710	−5691
215	TDO	224.8	−220.3	5710	−5596
216	TDI	224.8	−216.6	5710	−5501
217	TMS	224.8	−212.8	5710	−5406
218	V_{SS}	224.8	−209.1	5710	−5311
219	V_{CC}_2.9	224.8	−205.3	5710	−5216
220	V_{CC}_2.9	224.8	−201.6	5710	−5121

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 7 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
221	V_{SS}	224.8	−197.9	5710	−5026
222	TRST#	224.8	−194.1	5710	−4931
223	V_{SS}	224.8	−179.2	5710	−4551
224	V_{CC}_2.9	224.8	−175.4	5710	−4456
225	V_{CC}_2.9	224.8	−171.7	5710	−4361
226	V_{SS}	224.8	−167.9	5710	−4266
227	N.C.	224.8	−153.0	5710	−3886
228	V_{CC}_3.3	224.8	−149.2	5710	−3791
229	V_{SS}	224.8	−145.5	5710	−3696
230	V_{CC}_2.9	224.8	−141.8	5710	−3601
231	V_{CC}_2.9	224.8	−138.0	5710	−3506
232	V_{SS}	224.8	−134.3	5710	−3411
233	N.C.	224.8	−130.5	5710	−3316
234	N.C.	224.8	−126.8	5710	−3221
235	V_{SS}	224.8	−111.8	5710	−2841
236	V_{CC}_2.9	224.8	−108.1	5710	−2746
237	V_{CC}_2.9	224.8	−104.4	5710	−2651
238	V_{SS}	224.8	−100.6	5710	−2556
239	V_{SS}	224.8	−89.4	5710	−2271
240	V_{SS}	224.8	−85.7	5710	−2176
241	V_{SS}	224.8	−81.9	5710	−2081
242	V_{SS}	224.8	−78.2	5710	−1986
243	V_{CC}_2.9	224.8	−74.4	5710	−1891
244	V_{CC}_2.9	224.8	−70.7	5710	−1796
245	V_{SS}	224.8	−67.0	5710	−1701
246	N.C.	224.8	−48.3	5710	−1226
247	V_{SS}	224.8	−44.5	5710	−1131
248	V_{CC}_2.9	224.8	−40.8	5710	−1036
249	V_{CC}_2.9	224.8	−37.0	5710	−941
250	V_{SS}	224.8	−33.3	5710	−846
251	V_{CC}_3.3	224.8	−14.6	5710	−371
252	V_{SS}	224.8	−10.9	5710	−276
253	V_{CC}_2.9	224.8	−7.1	5710	−181
254	V_{CC}_2.9	224.8	−3.4	5710	−86
255	V_{SS}	224.8	0.4	5710	9
256	N.C.	224.8	4.1	5710	104
257	STPCLK#	224.8	7.8	5710	199

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 8 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
258	N.C.	224.8	19.1	5710	484
259	V_{SS}	224.8	22.8	5710	579
260	V_{CC}_2.9	224.8	26.5	5710	674
261	V_{CC}_2.9	224.8	30.3	5710	769
262	V_{SS}	224.8	34.0	5710	864
263	N.C.	224.8	45.2	5710	1149
264	N.C.	224.8	49.0	5710	1244
265	BF	224.8	52.7	5710	1339
266	V_{SS}	224.8	56.5	5710	1434
267	V_{CC}_2.9	224.8	60.2	5710	1529
268	V_{CC}_2.9	224.8	63.9	5710	1624
269	V_{SS}	224.8	67.7	5710	1719
270	V_{SS}	224.8	71.4	5710	1814
271	N.C.	224.8	82.6	5710	2099
272	V_{SS}	224.8	86.4	5710	2194
273	V_{CC}_2.9	224.8	90.1	5710	2289
274	V_{CC}_2.9	224.8	93.9	5710	2384
275	V_{SS}	224.8	97.6	5710	2479
276	N.C.	224.8	101.4	5710	2574
277	PEN#	224.8	105.1	5710	2669
278	INIT	224.8	108.8	5710	2764
279	IGNNE#	224.8	112.6	5710	2859
280	V_{SS}	224.8	116.3	5710	2954
281	V_{CC}_2.9	224.8	120.1	5710	3049
282	V_{CC}_2.9	224.8	123.8	5710	3144
283	V_{SS}	224.8	127.5	5710	3239
284	SMI#	224.8	131.3	5710	3334
285	INTR	224.8	135.0	5710	3429
286	R/S#	224.8	138.8	5710	3524
287	NMI	224.8	142.5	5710	3619
288	N.C.	224.8	150.0	5710	3809
289	A21	224.8	153.7	5710	3904
290	A22	224.8	157.5	5710	3999
291	A23	224.8	161.2	5710	4094
292	V_{SS}	224.8	164.9	5710	4189
293	V_{CC}_3.3	224.8	168.7	5710	4284
294	A24	224.8	172.4	5710	4379

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 9 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (= .001 inch)		Microns	
		X	Y	X	Y
295	A25	224.8	176.2	5710	4474
296	A26	224.8	179.9	5710	4569
297	A27	224.8	183.6	5710	4664
298	V_{SS}	224.8	187.4	5710	4759
299	V_{CC}_3.3	224.8	191.1	5710	4854
300	A28	224.8	194.9	5710	4949
301	A29	224.8	198.6	5710	5044
302	A30	224.8	202.3	5710	5139
303	A31	224.8	206.1	5710	5234
304	V_{SS}	224.8	209.8	5710	5329
305	V_{CC}_3.3	224.8	213.6	5710	5424
306	V_{CC}_2.9	224.8	217.3	5710	5519
307	V_{SS}	224.8	221.0	5710	5614
308	A3	206.2	244.8	5238	6217
309	V_{SS}	202.5	244.8	5143	6217
310	V_{CC}_3.3	198.8	244.8	5048	6217
311	A4	195.0	244.8	4953	6217
312	A5	191.3	244.8	4858	6217
313	V_{SS}	187.5	244.8	4763	6217
314	V_{CC}_3.3	183.8	244.8	4668	6217
315	A6	180.1	244.8	4573	6217
316	A7	176.3	244.8	4478	6217
317	V_{SS}	172.6	244.8	4383	6217
318	V_{CC}_3.3	168.8	244.8	4288	6217
319	A8	165.1	244.8	4193	6217
320	V_{SS}	161.4	244.8	4098	6217
321	V_{CC}_2.9	157.6	244.8	4003	6217
322	V_{CC}_2.9	153.9	244.8	3908	6217
323	V_{SS}	150.1	244.8	3813	6217
324	A9	146.4	244.8	3718	6217
325	V_{SS}	142.6	244.8	3623	6217
326	V_{CC}_3.3	138.9	244.8	3528	6217
327	A10	135.2	244.8	3433	6217
328	A11	131.4	244.8	3338	6217
329	V_{SS}	127.7	244.8	3243	6217
330	V_{CC}_3.3	123.9	244.8	3148	6217
331	A12	120.2	244.8	3053	6217

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 10 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
332	V_{SS}	116.5	244.8	2958	6217
333	V_{CC}_2.9	112.7	244.8	2863	6217
334	V_{CC}_2.9	109.0	244.8	2768	6217
335	V_{SS}	105.2	244.8	2673	6217
336	A13	101.5	244.8	2578	6217
337	V_{SS}	97.8	244.8	2483	6217
338	V_{CC}_3.3	94.0	244.8	2388	6217
339	A14	90.3	244.8	2293	6217
340	V_{SS}	86.5	244.8	2198	6217
341	V_{CC}_2.9	82.8	244.8	2103	6217
342	V_{CC}_2.9	79.1	244.8	2008	6217
343	V_{SS}	75.3	244.8	1913	6217
344	A15	71.6	244.8	1818	6217
345	V_{SS}	67.8	244.8	1723	6217
346	V_{CC}_3.3	64.1	244.8	1628	6217
347	A16	60.4	244.8	1533	6217
348	A17	56.6	244.8	1438	6217
349	V_{SS}	52.9	244.8	1343	6217
350	V_{CC}_3.3	49.1	244.8	1248	6217
351	A18	45.4	244.8	1153	6217
352	V_{SS}	41.7	244.8	1058	6217
353	V_{CC}_2.9	37.9	244.8	963	6217
354	V_{CC}_2.9	34.2	244.8	868	6217
355	V_{SS}	30.4	244.8	773	6217
356	A19	26.7	244.8	678	6217
357	V_{SS}	23.0	244.8	583	6217
358	V_{CC}_3.3	19.2	244.8	488	6217
359	A20	15.5	244.8	393	6217
360	V_{SS}	11.7	244.8	298	6217
361	V_{CC}_2.9	8.0	244.8	203	6217
362	V_{CC}_2.9	4.3	244.8	108	6217
363	V_{SS}	0.5	244.8	13	6217
364	V_{CC}_2.9	−3.2	244.8	−82	6217
365	V_{SS}	−7.0	244.8	−177	6217
366	RESET	−10.7	244.8	−272	6217
367	N.C.	−108.0	244.8	−2744	6217
368	CLK	−111.8	244.8	−2839	6217

Table 1. Pentium® Processor Bond Pad Center Data (Sheet 11 of 11)

PAD#	SIGNAL	Pad Center			
		Mils (=.001 inch)		Microns	
		X	Y	X	Y
369	SCYC	−115.5	244.8	−2934	6217
370	V_{SS}	−119.3	244.8	−3029	6217
371	V_{CC}_3.3	−123.0	244.8	−3124	6217
372	BE7#	−126.7	244.8	−3219	6217
373	BE6#	−130.5	244.8	−3314	6217
374	BE5#	−134.2	244.8	−3409	6217
375	BE4#	−138.0	244.8	−3504	6217
376	V_{SS}	−141.7	244.8	−3599	6217
377	V_{CC}_3.3	−145.4	244.8	−3694	6217
378	BE3#	−149.2	244.8	−3789	6217
379	BE2#	−152.9	244.8	−3884	6217
380	BE1#	−156.7	244.8	−3979	6217
381	BE0#	−160.4	244.8	−4074	6217
382	A20M#	−164.1	244.8	−4169	6217
383	FLUSH#	−167.9	244.8	−4264	6217
384	BUSCHK#	−171.6	244.8	−4359	6217
385	W/R#	−175.4	244.8	−4454	6217
386	V_{SS}	−179.1	244.8	−4549	6217
387	V_{CC}_3.3	−182.8	244.8	−4644	6217
388	HIT#	−186.6	244.8	−4739	6217
389	HITM#	−190.3	244.8	−4834	6217
390	V_{SS}	−194.1	244.8	−4929	6217
391	V_{CC}_3.3	−197.8	244.8	−5024	6217
392	ADS#	−201.5	244.8	−5119	6217
393	EADS#	−205.3	244.8	−5214	6217

NOTES:
1. N.C. signifies No Connect. These pads must not be connected.
2. The symbol '#' is used at the end of the signal to denote an active low signal.
3. X-Y pad coordinates represent bond pad centers and are relative to the center of the die.
4. Boundary Scan (JTAG) is implemented through the following pads:
   214 (TCK). 215 (TDO), 216 (TDI), 217 (TMS), 222 (TRST).

2.0 INTEL DIE PRODUCTS PROCESSING

2.1 Test Procedure

Intel has instituted full-speed functional testing at the die level for all SmartDie products. This level of testing is ordinarily performed only after assembly into a package. Each die is tested to the same electrical limits as the equivalent packaged unit.

2.2 Wafer Probe

Wafer probing is performed on every wafer produced in Intel Fabs. The process consists of specific electrical tests and device-specific functionality tests.

At the wafer level, built-in test structures are probed to verify that device electrical characteristics are in control and meet specifications. Measurements are made of transistor threshold voltages and current characteristics; poly and contact resistance; gate oxide and junction integrity; and specific parameters critical to the particular technology and device type. Wafer-to-wafer, across-the-wafer run-to-run variation and conformance to spec limits are checked.

The actual devices on each wafer are then probed for both functionality and performance to specifications. Additional reliability tests are also included in the probe steps.

2.3 Wafer Saw

Probed wafers are transferred to Intel's assembly sites to be sawed. The saw cuts totally through the wafer.

2.4 Die Inspection

Upon completion of the wafer saw, the die are moved to pick and place equipment that removes reject die. The remaining die are submitted to the same visual inspection as standard packaged product. The compliant die are then transferred to GEL-PAKs for shipment.

2.5 Packing Procedure

Intel will ship all Intel die products in GEL-PAKs*. GEL-PAKs eliminate the die edge damage usually associated with die cavity plates or chip trays.

The backside of each die adheres to the gel membrane in the GEL-PAK, eliminating the risk of damage to the active die surface. A simple vacuum release mechanism allows for pick and place removal at the customer's site.

Only die from the same wafer lot are packaged together in a GEL-PAK, and all die are placed in the GEL-PAKs with a consistent orientation. The GEL-PAKs are then sealed and labeled with the following information:

- Intel SmartDie
- Intel Part Number
- Assembly Process Order / Spec
- ROM Code (if applicable)
- Customer Part Number (if applicable)
- Assembly Lot Traveler Number
- Finished Product Order Number
- Quantity
- Seal Date
- Country of Origin

NOTE:
GEL-PAKs require a Vacuum Release Station. Contact Vichem Corporation for more information.

2.6 Inspection Steps

Multiple inspection steps are performed during the die fabrication and packing flow. These steps are performed according to the same specifications and criteria established for Intel's standard packaged product. Specific inspection steps include a wafer saw visual as well as a final die visual just before die are sealed in moisture barrier bags.

2.7 Storage Requirements

Intel die products will be shipped in GEL-PAKs and sealed in a moisture-barrier anti-static bag with a desiccant. No special storage procedures are required while the bag is still unopened. Once opened, the GEL-PAK should be stored in a dry, inert atmosphere to prevent corrosion of the bond pads.

2.8 Electro-Static Discharge (ESD)

Components are ESD sensitive.

3.0 SPECIFICATIONS

Specifications within this document are specific to a particular die revision and are subject to change without notice. Verify with your local Intel Sales Office that you have the latest data before finalizing a design.

3.1 Physical Specifications

Table 2 defines Pentium processor physical specifications.

Table 2. Pentium® Processor Physical Specifications

Die Revision:	A-1
Post-Saw Die Dimensions:	Mils: X = 463 ± 0.5, Y = 503 ± 0.5 See associated Die/Bond Pad Layout for X, Y orientation.
Die Thickness:	17 ± 1 mils
Minimum Pad Pitch:	Pads may not be evenly pitched. Minimum pitch is: 95 microns (3.7 mils)
Pad Passivation Opening Size:	Mils: 2.8 x 5.0 (single pads) Microns: 71 x 128 (single pads)
Bond Pad Metallization: (outermost layer first)	14,000 Angstroms Al (0.5% Cu), 1000 Angstroms Ti, 365 Angstroms TiN
Pads per Die:	393
Die Backside Material: (outermost layer first)	1600 Angstroms Gold, 150 Angstroms Chrome
Passivation: (outermost layer first)	6.13 microns polyimide, 0.45 microns nitride
Intel Fabrication Process:	BiCMOS (min. feature size 0.6 microns)

3.2 DC Specifications

NOTICE: This data sheet contains preliminary information on new products in production. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

ABSOLUTE MAXIMUM RATINGS*

GEL-PAK Storage Temperature ..... 0°C to +70°C

Junction Temperature Under Bias ... −65°C to +110°C
3V Supply Voltage wrt. V_{SS} ..... −0.5V to +4.6V
2.9V Supply Voltage wrt. V_{SS} ..... −0.5V to +4.1V
3V Only Buffer DC Input Voltage. −0.5V to V_{CC3} +0.5V; not to exceed 4.6V(1)
5V Safe Buffer DC Input Voltage .... −0.5V to +6.5V(2)

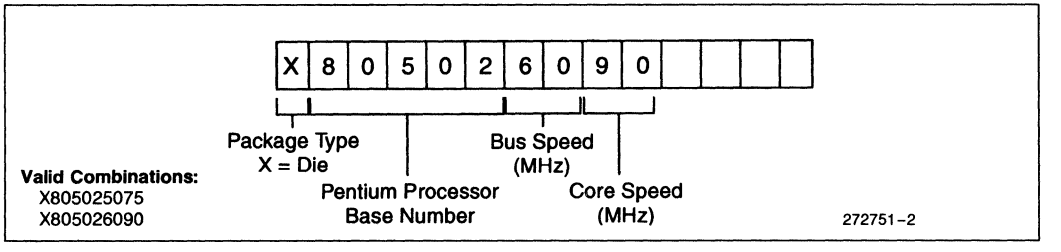
WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

OPERATING CONDITIONS*

V_{CC3} (I/O Supply Voltage) ..... 3.3V ± 165 mV
V_{CC2} (Core Supply Voltage) ..... 2.9V ± 165 mV
T_J (Junction Temperature Under Bias) .. 0°C to 105°C(3)
Core Operating Frequency ..... 75, 90 MHz (50, 60 MHz Bus)
Substrate Bias ..... DriveV_{SS}

NOTES:
1. Applies to all Pentium processors with Voltage Reduction Technology inputs except CLK.
2. Applies to CLK.
3. Average die surface temperature.

4.0 DEVICE NOMENCLATURE



5.0 REFERENCE INFORMATION

Document Title	Order #
Pentium® Processors at iComp® Index 735/90, 610/75 MHz with Voltage Reduction Technology	242557
Pentium® Processors and Related Products	241732
AP-479 Pentium® Processor Clock Design (Application Note)	241574
AP-480 Pentium® Processor Thermal Design Guidelines (Application Note)	241575

6.0 REVISION HISTORY

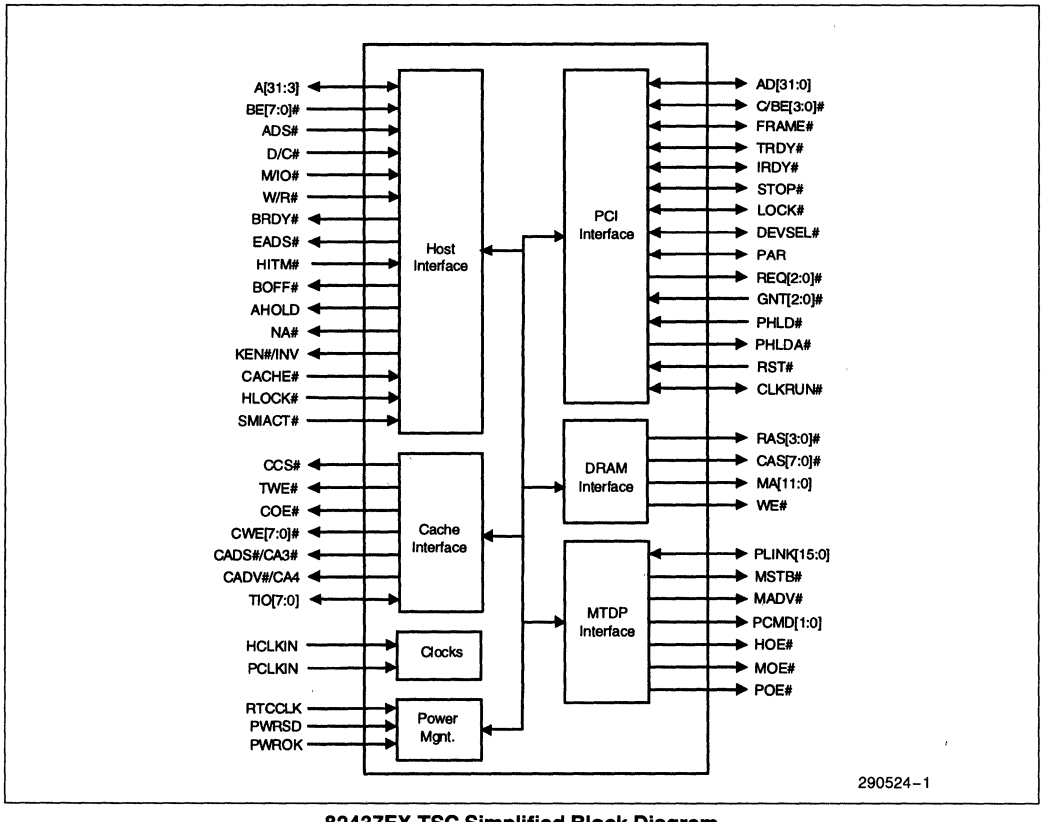
Revision	Date	Description
-001	7/95	Initial Release
-002	10/95	Pads 228, 239, 240, 241—Signal names modified.

82437MX SYSTEM CONTROLLER (MTSC) AND 82438MX DATA PATH UNIT (MTDP)

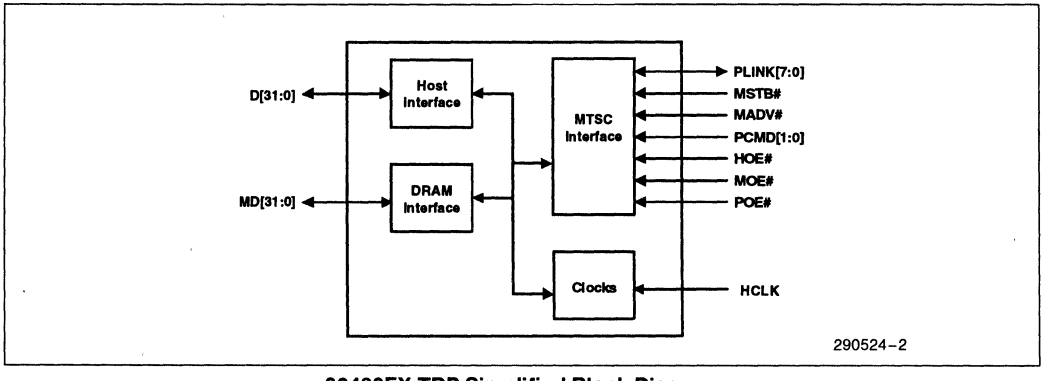
- **Supports the Pentium Processor at iCOMP™ Index 1000/120 MHz, iCOMP Index 735/90 MHz and the 610/75 MHz Pentium Processor**
- **Integrated Second Level Cache Controller**
  - **— Direct Mapped Organization**
  - **— Write-Back Cache Policy**
  - **— Cacheless, 256 Kbyte, and 512 Kbyte**
  - **— Standard, Burst and Pipelined Burst SRAMs**
  - **— Cache Hit Read/Write Cycle Timings at 3-1-1-1 with Burst or Pipelined Burst SRAMs**
  - **— Back-to-Back Read Cycles at 3-1-1-1-1-1-1-1 with Burst or Pipelined Burst SRAMs**
  - **— Integrated Tag/Valid Status Bits for Cost Savings and Performance**
  - **— Supports 3.3V SRAMs for Tag Address**
- **Integrated DRAM Controller**
  - **— 64-Bit Data Path to Memory**
  - **— 4 Mbytes to 128 Mbytes Main Memory**
  - **— EDO/Hyper Page Mode DRAM (x-2-2-2 Reads) Provides Superior Cacheless Designs**
  - **— Standard Page Mode DRAMs**
  - **— 4 RAS Lines**
  - **— 4 Qword Deep Buffer for 3-1-1-1 Posted Write Cycles**
  - **— Symmetrical and Asymmetrical DRAMs**
  - **— 3V or 5V DRAMs**
- **Power Management**
  - **— DRAM Refresh During Suspend**
  - **— Self Refresh and Extended Refresh**
- **Fully Synchronous 25/30/33 MHz PCI Bus Interface**
  - **— 100 MB/s Instant Access Enables native signal processing on Pentium Processors**
  - **— Synchronized CPU-to-PCI Interface for High Performance Graphics**
  - **— PCI Bus Arbiter: MPIIX and Three PCI Bus Masters Supported**
  - **— CPU-to-PCI Memory Write Posting with 4 Dword deep buffers**
  - **— Converts Back-to-Back Sequential CPU to PCI Memory Writes to PCI Burst Writes**
  - **— PCI-to-DRAM Posting of 12 Dwords**
  - **— PCI-to-DRAM up to 120 Mbytes/Sec Bandwidth Utilizing Snoop Ahead Feature**
- **NAND Tree for Board-Level ATE Testing**
- **208 Pin QFP for the 82437MX System Controller (MTSC); 100 Pin TQFP for Each 82438MX Data Path (MTDP)**

The 82430MX PCIset consists of the 82437MX System Controller (MTSC), two 82438MX Data Paths (MTDP), and the 82371MX PCI IO IDE Xcelerator (MPIIX). The MTSC/MTDP form a Host-to-PCI bridge and provide the second level cache control and a full function 64-bit data path to main memory. The MTSC integrates the cache and main memory DRAM control functions and provides bus control for transfers between the CPU, cache, main memory, and the PCI Bus. The second level (L2) cache controller supports a write-back cache policy for cache sizes of 256 KBytes and 512 KBytes. Cacheless designs are also supported. The cache memory can be implemented with either standard, burst, or pipelined burst SRAMs. An external Tag RAM is used for the address tag and an internal Tag RAM for the cache line status bits. For the MTSC's DRAM controller, four rows are supported for up to 128 MBytes of main memory. The MTSC's optimized PCI interface allows the CPU to sustain the highest possible bandwidth to the graphics frame buffer at all frequencies. Using the snoop ahead feature, the MTSC allows PCI masters to achieve full PCI bandwidth. The MTDPs provide the data paths between the CPU/cache, main memory, and PCI. For increased system performance, the MTDPs contain read prefetch and posted write buffers.

December 1995
Order Number: 290524-001



82437FX TSC Simplified Block Diagram



82438FX TDP Simplified Block Diagram

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

82371MX PCI I/O IDE XCELERATOR (MPIIX)

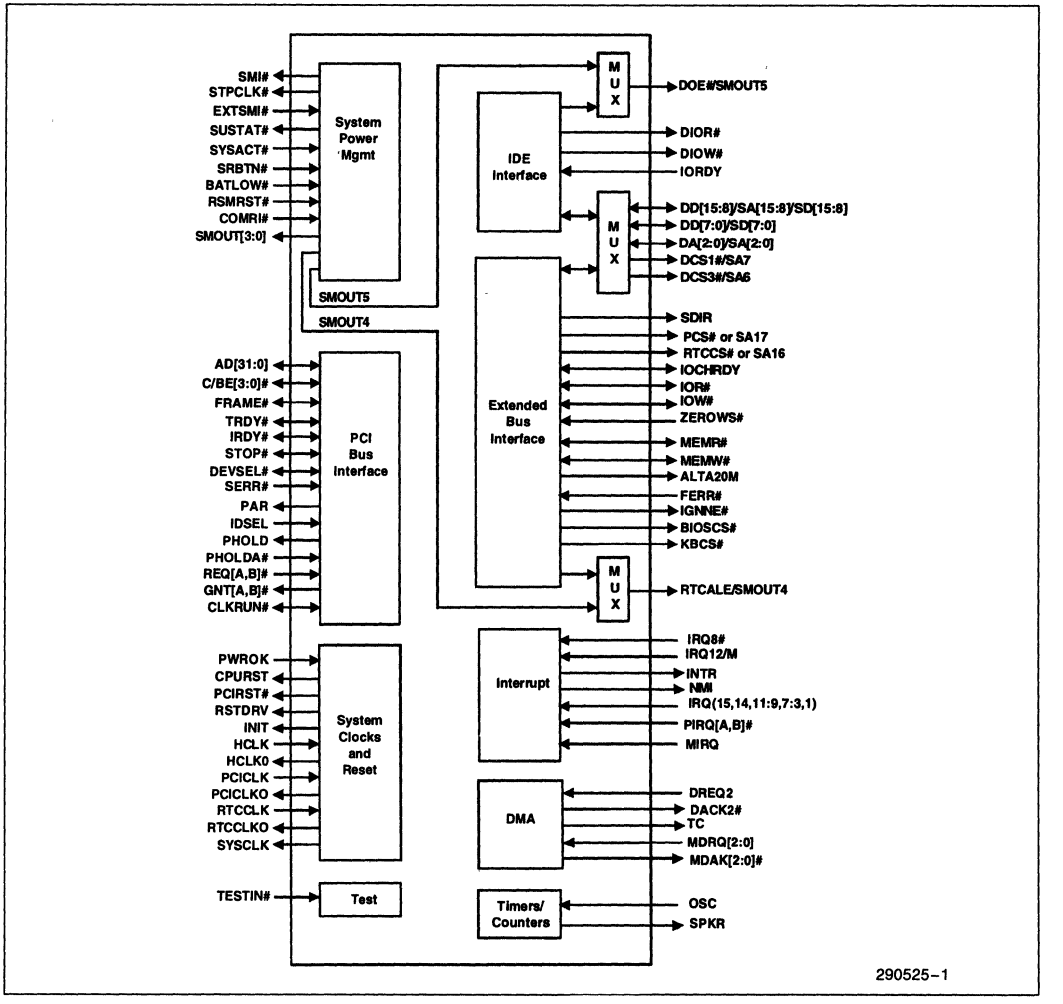
- **Provides a Bridge between the PCI Bus and Extended I/O Bus**
  - **PCI Bus; 25 MHz–33 MHz**
  - **Extended I/O Bus; 7.5 MHz–8.33 MHz**
- **System Power Management (Intel SMM Support)**
  - **Programmable System Management Interrupt (SMI)— Hardware/Software Events, EXTSMI#**
  - **Programmable CPU Clock Control (STPCLK#) With Auto Clock Throttle**
  - **Peripheral Device Power Management (Local Standby)**
  - **Suspend State Support (Suspend-to-DRAM and Suspend-to-Disk)**
- **Enhanced DMA Functions**
  - **Two 8237 DMA Controllers**
  - **Compatible DMA Transfers**
  - **PC/PCI DMA Expansion for Docking Support**
- **Fast IDE Interface**
  - **PIO Mode 4 Transfers**
  - **2x16 Bit Posted Write Buffer and 1x32 bit Read Prefetch Buffer**
- **Plug-n-Play Port for Motherboard Devices**
  - **3 Steerable DMA Channels**
  - **1 Steerable Interrupt Line (Plus 2 Steerable PCI Interrupts)**
  - **1 Programmable Chip Select**
- **Functionality of One 82C54 Timer**
  - **System Timer**
  - **Refresh Request**
  - **Speaker Tone Output**
- **Functionality of Two 82C59 Interrupt Controllers**
  - **14 Interrupts Supported**
  - **Independently Programmable for Edge/Level Sensitivity**
- **X-Bus Peripheral Support**
  - **Chip Select Decode**
  - **Controls Lower X-Bus Data Byte Transceiver**
- **Non-Maskable Interrupts (NMI)**
  - **PCI System Error Reporting**
- **NAND Tree for Board-Level ATE Testing**
- **176-Pin TQFP**

The 82371MX PCI I/O IDE Xcelerator (MPIIX) provides the bridge between the PCI bus and the ISA-like Extended I/O expansion bus. In addition, the 82371MX has an IDE interface that supports two IDE devices providing an interface for IDE hard disks and CD ROMs. The MPIIX integrates many common I/O functions found in ISA based PC systems—a seven channel DMA controller, two 82C59 interrupt controllers, an 8254 timer/counter, Intel SMM power management support, and control logic for NMI generation. Chip select decoding is provided for BIOS, real time clock, and keyboard controller. Edge/Level interrupts and interrupt steering are supported for PCI plug and play compatibility.

The MPIIX also provides the Extended I/O Bus for a direct connection to Super I/O devices providing a complete PC compatible I/O solution. MPIIX also provides support for the "Mobile PC/PCI" DMA Expansion protocol that enables the implementation of Docking Stations with full ISA and PCI capability without running the full ISA bus across the docking connector. For motherboard plug and play compatibility, the 82371MX also provides three steerable DMA channels, up to three steerable interrupt lines, and a programmable chip select. The interrupt lines can be routed to any of the available ISA interrupts.

The MPIIX's power management function supports SMI# interrupt sources, extensive clock control (including Auto Clock Throttling), peripheral power idle detection with access traps, system Suspend-to-DRAM and Suspend-to-Disk.

December 1995
Order Number: 290525-001



82371MX MPIIX Simplified Block Diagram

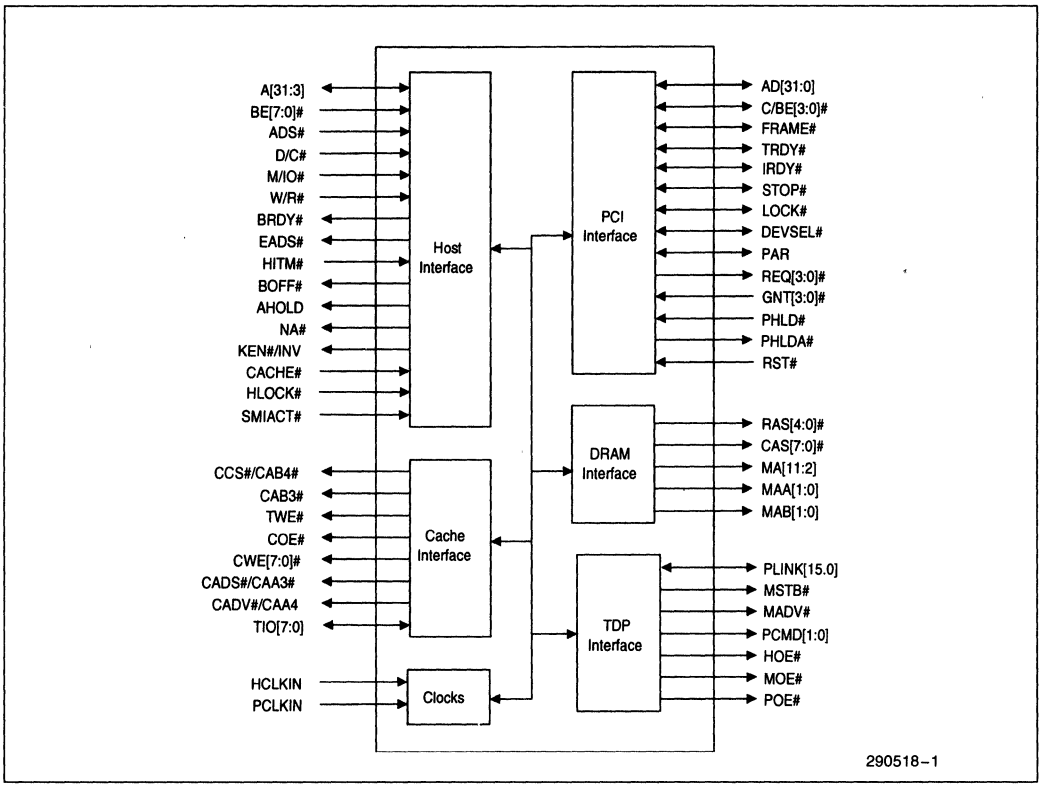
The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

82430FX PCIset DATASHEET 82437FX SYSTEM CONTROLLER (TSC) AND 82438FX DATA PATH UNIT (TDP)

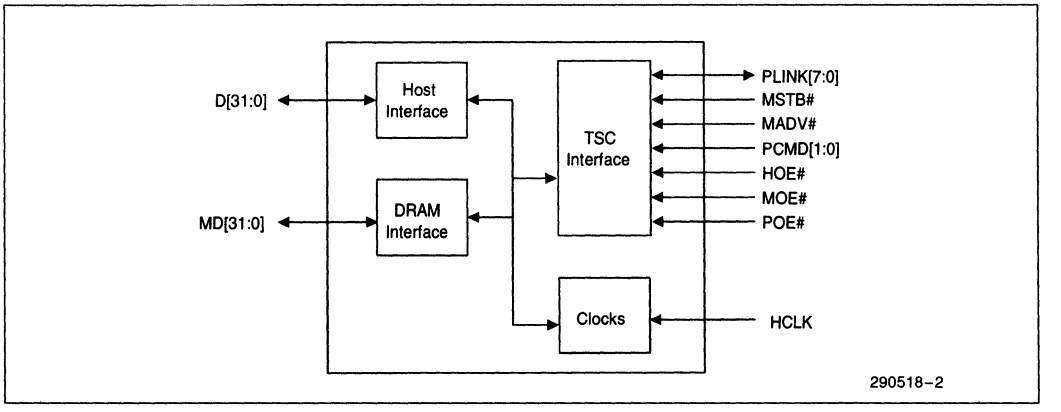
- Supports the Pentium® Processor at iCOMP® Index 1110\133 MHz, iCOMP Index 1000\120 MHz, iCOMP Index 815\100 MHz, iCOMP Index 735\90 MHz and the iCOMP Index 610\75 MHz
- Integrated Second Level Cache Controller
  - Direct Mapped Organization
  - Write-Back Cache Policy
  - Cacheless, 256-Kbyte, and 512-Kbyte
  - Standard Burst and Pipelined Burst SRAMs
  - Cache Hit Read/Write Cycle Timings at 3-1-1-1 with Burst or Pipelined Burst SRAMs
  - Back-to-Back Read Cycles at 3-1-1-1-1-1-1-1 with Burst or Pipelined Burst SRAMs
  - Integrated Tag/Valid Status Bits for Cost Savings and Performance
  - Supports 5V SRAMs for Tag Address
- Integrated DRAM Controller
  - 64-Bit Data Path to Memory
  - 4 Mbytes to 128 Mbytes Main Memory
  - EDO/Hyper Page Mode DRAM (x-2-2-2 Reads) or Standard Page Mode DRAMs
  - 5 RAS Lines
  - 4 Qword Deep Buffer for 3-1-1-1 Posted Write Cycles
  - Symmetrical and Asymmetrical DRAMs
  - 3V or 5V DRAMs
- EDO DRAM Support
  - Highest Performance with Burst or Pipelined Burst SRAMs
  - Superior Cacheless Designs
- Fully Synchronous 25/30/33 MHz PCI Bus Interface
  - 100 MB/s Instant Access Enables Native Signal Processing (NSP) on Pentium Processors
  - Synchronized CPU-to-PCI Interface for High Performance Graphics
  - PCI Bus Arbiter: PIIX and Four PCI Bus Masters Supported
  - CPU-to-PCI Memory Write Posting with 4 Dword Deep Buffers
  - Converts Back-to-Back Sequential CPU to PCI Memory Writes to PCI Burst Writes
  - PCI-to-DRAM Posting of 12 Dwords
  - PCI-to-DRAM up to 120 Mbytes/Sec Bandwidth Utilizing Snoop Ahead Feature
- NAND Tree for Board-Level ATE Testing
- 208 Pin QFP for the 82437FX System Controller (TSC); 100 Pin QFP for Each 82438FX Data Path (TDP)

The 82430FX PCIset consists of the 82437FX System Controller (TSC), two 82438FX Data Paths (TDP), and the 82371FB PCI ISA IDE Xcelerator (PIIX). The PCIset forms a Host-to-PCI bridge and provides the second level cache control and a full function 64-bit data path to main memory. The TSC integrates the cache and main memory DRAM control functions and provides bus control for transfers between the CPU, cache, main memory, and the PCI Bus. The second level (L2) cache controller supports a write-back cache policy for cache sizes of 256 Kbytes and 512 Kbytes. Cacheless designs are also supported. The cache memory can be implemented with either standard, burst, or pipelined burst SRAMs. An external Tag RAM is used for the address tag and an internal Tag RAM for the cache line status bits. For the TSC's DRAM controller, five rows are supported for up to 128 Mbytes of main memory. The TSC's optimized PCI interface allows the CPU to sustain the highest possible bandwidth to the graphics frame buffer at all frequencies. Using the snoop ahead feature, the TSC allows PCI masters to achieve full PCI bandwidth. The TDPs provide the data paths between the CPU/cache, main memory, and PCI. For increased system performance, the TDPs contain read prefetch and posted write buffers.

December 1995
Order Number: 290518-001



82437FX TSC Simplified Block Diagram



82438FX TDP Simplified Block Diagram

82430FX PCIset DATASHEET 82437FX SYSTEM CONTROLLER (TSC) AND 82438FX DATA PATH UNIT (TDP)

CONTENTS

CONTENTS

1.0 ARCHITECTURE OVERVIEW OF TSC/TDP

The 82430FX PCIset (Figure 1) consists of the 82437FX System Controller (TSC), two 82438FX Data Path (TDP) units, and the 82371FB PCI IDE ISA Xcelerator (PIIX). The TSC and two TDPs form a Host-to-PCI bridge. The PIIX is a multi-function PCI device providing a PCI-to-ISA bridge and a fast IDE interface. The PIIX also provides power management and has a plug and play port.

The two TDPs provide a 64-bit data path to the host and to main memory and provide a 16-bit data path (PLINK) between the TSC and TDP. PLINK provides the data path for CPU to PCI accesses and for PCI to main memory accesses. The TSC and TDP bus interfaces are designed for 3V and 5V busses. The TSC/TDP connect directly to the Pentium® processor 3V host bus; The TSC/TDP connect directly to 5V or 3V main memory DRAMs; and the TSC connects directly to the 5V PCI Bus.

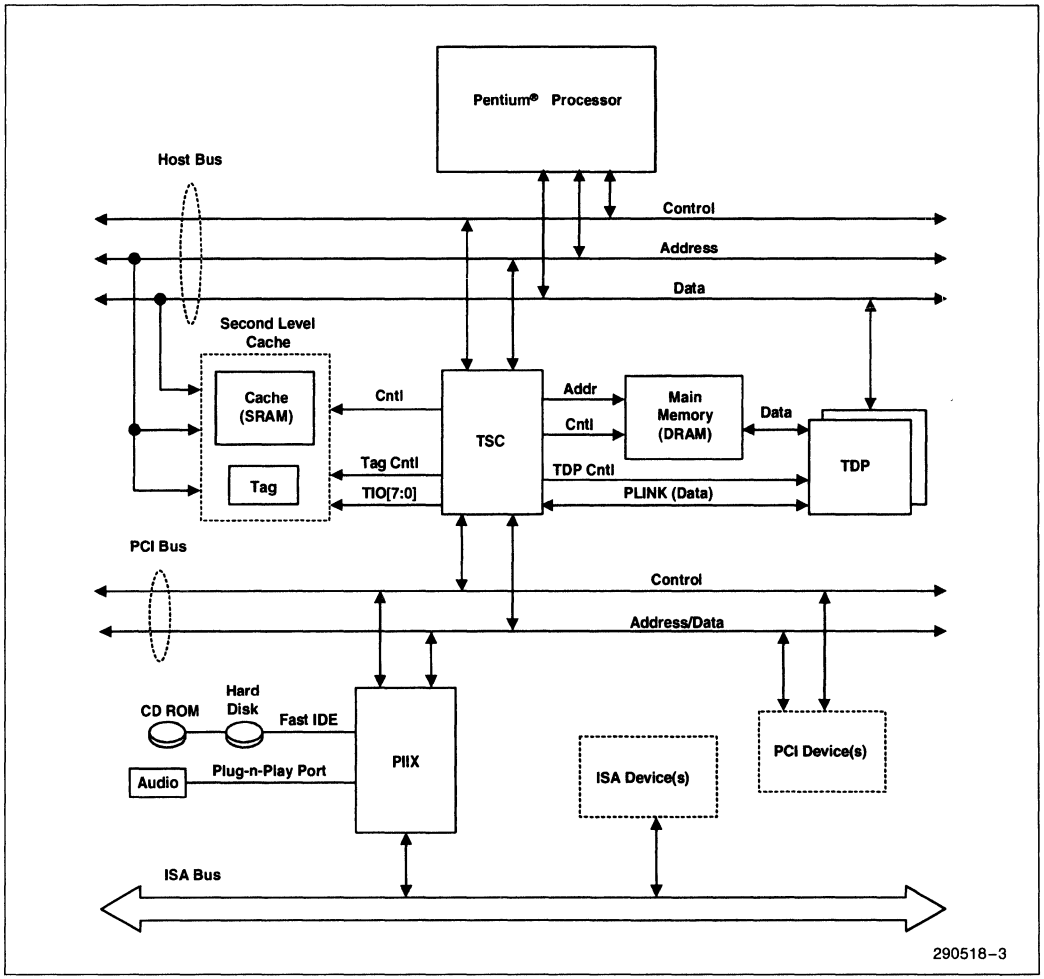


Figure 1. 82430FX PCIset System

DRAM Interface

The DRAM interface is a 64-bit data path that supports both standard page mode and Extended Data Out (EDO) (also known as Hyper Page Mode) memory. The DRAM interface supports 4 Mbytes to 128 Mbytes with five RAS lines available and also supports symmetrical and asymmetrical addressing for 512K, 1M, 2M, and 4M deep DRAMs.

Second Level Cache

The TSC supports a write-back cache policy providing all necessary snoop functions and inquire cycles. The second level cache is direct mapped and supports both a 256-Kbyte or 512-Kbyte SRAM configuration using either burst, pipelined burst, or standard SRAMs. The burst 256-Kbyte configuration performance is 3-1-1-1 for read/write cycles; pipelined back-to-back reads can maintain a 3-1-1-1-1-1-1-1 transfer rate.

TDP

Two TDPs create a 64-bit CPU and main memory data path. The TDP's also interface to the TSC's 16-bit PLINK inter-chip bus for PCI transactions. The combination of the 64-bit memory path and the 16-bit PLINK bus make the TDP's a cost effective solution, providing optimal CPU-to-main memory performance while maintaining a small package footprint (100 pins each).

PCI Interface

The PCI interface is 2.0 compliant and supports up to 4 PCI bus masters in addition to the PIIX bus master requests. While the TSC and TDP's together provide the interface between PCI and main memory, only the TSC connects to the PCI Bus.

Buffers

The TSC and TDP's together contain buffers for optimizing data flow. A four Qword deep buffer is provided for CPU-to-main memory writes, second level cache write back cycles, and PCI-to-main memory transfers. This buffer is used to achieve 3-1-1-1 posted writes to main memory. A four Dword buffer is used for CPU-to-PCI writes. In addition, a four Dword PCI Write Buffer is provided which is combined with the DRAM Write Buffer to supply a 12 Dword deep buffering for PCI to main memory writes.

System Clocking

The processor, second level cache, main memory subsystem, and PLINK bus all run synchronous to the host clock. The PCI clock runs synchronously at half the host clock frequency. The TSC and TDP's have a host clock input and the TSC has a PCI clock input. These clocks are derived from an external source and have a maximum clock skew requirement with respect to each other.

2.0 SIGNAL DESCRIPTION

This section provides a detailed description of each signal. The signals are arranged in functional groups according to their associated interface.

The "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

The following notations are used to describe the signal type.

I Input is a standard input-only signal.

O Totem pole output is a standard active driver.

o/d Open drain.

t/s Tri-State is a bi-directional, tri-state input/output pin.

s/t/s Sustained tri-state is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. An external pull-up is required to sustain the inactive state until another agent drives it and must be provided by the central resource.

2.1 TSC Signals

2.1.1 HOST INTERFACE (TSC)

Signal Name	Type	Description
A[31:3]	I/O 3V	**ADDRESS BUS:** A[31:3] connect to the address bus of the CPU. During CPU cycles A[31:3] are inputs. These signals are driven by the TSC during cache snoop operations. Note that A[31:28] provide poweron/reset strapping options for the second level cache.
BE[7:0]#	I 3V	**BYTE ENABLES:** The CPU byte enables indicate which byte lane the current CPU cycle is accessing. All eight byte lanes are provided to the CPU if the cycle is a cacheable read regardless of the state of BE[7:0]#.
ADS#	I 3V	**ADDRESS STATUS:** The CPU asserts ADS# to indicate that a new bus cycle is being driven.
BRDY#	O 3V	**BUS READY:** The TSC asserts BRDY# to indicate to the CPU that data is available on reads or has been received on writes.
NA#	O 3V	**NEXT ADDRESS:** When burst SRAMs are used in the second level cache or the second level cache is disabled, the TSC asserts NA# in T2 during CPU write cycles and with the first assertion of BRDY# during CPU linefills. NA# is never asserted if the second level cache is enabled with asynchronous SRAMs. NA# on the TSC must be connected to the CPU NA# pin for all configurations.
AHOLD	O 3V	**ADDRESS HOLD:** The TSC asserts AHOLD when a PCI master is accessing main memory. AHOLD is held for the duration of the PCI burst transfer. The TSC negates AHOLD when the PCI to main memory read/write cycles complete and during PCI peer transfers.
EADS#	O 3V	**EXTERNAL ADDRESS STROBE:** Asserted by the TSC to inquire the first level cache when servicing PCI master accesses to main memory.
BOFF#	O 3V	**BACK OFF:** Asserted by the TSC when required to terminate a CPU cycle that was in progress.
HITM#	I 3V	**HIT MODIFIED:** Asserted by the CPU to indicate that the address presented with the last assertion of EADS# is modified in the first level cache and needs to be written back.
M/IO#, D/C#, W/R#	I 3V	**MEMORY/IO; DATA/CONTROL; WRITE/READ:** Asserted by the CPU with ADS# to indicate the type of cycle on the host bus.
HLOCK#	I 3V	**HOST LOCK:** All CPU cycles sampled with the assertion of HLOCK# and ADS#, until the negation of HLOCK# must be atomic (i.e., no PCI activity to main memory is allowed).
CACHE#	I 3V	**CACHEABLE:** Asserted by the CPU during a read cycle to indicate the CPU can perform a burst line fill. Asserted by the CPU during a write cycle to indicate that the CPU will perform a burst write-back cycle. If CACHE# is asserted to indicate cacheability, the TSC asserts KEN# either with the first BRDY#, or with NA#, if NA# is asserted before the first BRDY#.

Signal Name	Type	Description
KEN#/INV	O 3V	**CACHE ENABLE/INVALIDATE:** KEN#/INV functions as both the KEN# signal during CPU read cycles and the INV signal during first level cache snoop cycles. During CPU cycles, KEN#/INV is normally low. The TSC drives KEN# high during the first BRDY# or NA# assertion of a non-cacheable (in first level cache) CPU read cycle. The TSC drives INV high during the EADS# assertion of a PCI master DRAM write snoop cycle and low during the EADS# assertion of a PCI master DRAM read snoop cycle.
SMIACT#	I 3V	**SYSTEM MANAGEMENT INTERRUPT ACTIVE:** The CPU asserts SMIACT# when it is in system management mode as a result of an SMI. After SMM space (located at A0000h) is loaded and locked by BIOS, this signal must be sampled active with ADS# for the processor to access the SMM space of DRAM.

2.1.2 DRAM INTERFACE (TSC)

Signal Name	Type	Description
RAS[4:0]#	O 3V	**ROW ADDRESS STROBE:** These pins select the DRAM row.
CAS[7:0]#	O 3V	**COLUMN ADDRESS STROBE:** These pins always select which bytes are affected by a DRAM cycle.
MA[11:2]	O 3V	**MEMORY ADDRESS:** This is the row and column address for DRAM.
MAA[1:0]	O 3V	**MEMORY ADDRESS COPY A:** One copy of the MAs that change during a burst read or write of DRAM.
MAB[1:0]	O 3V	**MEMORY ADDRESS COPY B:** A second copy of the MAs that change during a burst read or write of DRAM.

2.1.3 SECONDARY CACHE INTERFACE (TSC)

Signal Name	Type	Description
CADV#/ CAA4	O 3V	**CACHE ADVANCE/CACHE ADDRESS 4 (COPY A):** This pin has two modes of operation depending on the type of SRAMs selected via hardware strapping options or programming the CC Register. The CAA4 mode is used when the L2 cache consists of asynchronous SRAMs. CAA4 is used to sequence through the Qwords in a cache line during a burst operation. CADV# mode is used when the L2 cache consists of burst SRAMs. In this mode, assertion causes the burst SRAM in the L2 cache to advance to the next Qword in the cache line.
CADS#/CAA3	O 3V	**CACHE ADDRESS STROBE/CACHE ADDRESS 3 (COPY A):** This pin has two modes of operation depending on the type of SRAMs selected via hardware strapping options or programming the CC Register. The CAA3 mode is used when the L2 cache consists of asynchronous SRAMs. CAA3 is used to sequence through the Qwords in a cache line during a burst operation. CADS# mode is used when the L2 cache consists of burst SRAMs. In this mode assertion causes the burst SRAM in the L2 cache to load the BSRAM address register from the BSRAM address pins.
CAB3	O 3V	**CACHE ADDRESS 3 (COPY B):** CAB3 is used when the L2 cache consists of asynchronous SRAMs. CAB3 is used to sequence through the Qwords in a cache line during a burst operation
CCS#/CAB4	O 3V	**CACHE CHIP SELECT/CACHE ADDRESS (COPY B):** This pin has two modes of operation depending on the type of SRAMs selected via hardware strapping options or programming the CC Register. The CAB4 mode is used when the L2 cache consists of asynchronous SRAMs. CAB4 is used to sequence through the Qwords in a cache line during a burst operation. A L2 cache consisting of burst SRAMs will power up, if necessary, and perform an access if CCS# is asserted when CADS# is asserted. A L2 cache consisting of burst SRAMs will power down if CCS# is negated when CADS# is asserted. When CCS# is negated, a L2 cache consisting of burst SRAMs ignores ADS#. If CCS# is asserted when ADS# is asserted, a L2 cache consisting of burst SRAMs will power up, if necessary, and perform an access.
COE#	O 3V	**CACHE OUTPUT ENABLE**: The secondary cache data RAMs drive the CPU's data bus when COE# is asserted.
CWE[7:0]#	O 3V	**CACHE WRITE ENABLE:** Each CWE# corresponds to one byte lane. Assertion causes the byte lane to be written into the secondary cache data RAMs if they are powered up.
TIO[7:0]	I/O 5V	**TAG ADDRESS:** These are inputs during CPU accesses and outputs during L2 cache line fills and L2 cache line invalidates due to inquire cycles. TIO[7:0] contain the L2 tag address for 256-Kbyte L2 caches. TIO[6:0] contains the L2 tag address and TIO7 contains the L2 cache valid bit for 512-Kbyte caches.
TWE#	O 5V	**TAG WRITE ENABLE:** When asserted, new state and tag addresses are written into the external tag.

2.1.4 PCI INTERFACE (TSC)

Signal Name	Type	Description
AD[31:0]	I/O 5V	**ADDRESS DATA BUS:** The standard PCI address and data lines. The address is driven with FRAME# assertion and data is driven or received in following clocks.
C/BE[3:0]#	I/O 5V	**COMMAND, BYTE ENABLE:** The command is driven with FRAME# assertion. Byte enables corresponding to supplied or requested data are driven on following clocks.
FRAME#	I/O 5V	**FRAME:** Assertion indicates the address phase of a PCI transfer. Negation indicates that one more data transfer is desired by the cycle initiator.
DEVSEL#	I/O 5V	**DEVICE SELECT:** The TSC drvies DEVSEL# when a PCI initiator attempts to access main memory. DEVSEL# is asserted at medium decode time.
IRDY#	I/O 5V	**INITIATOR READY:** Asserted when the initiator is ready for a data transfer.
TRDY#	I/O 5V	**TARGET READY:** Asserted when the target is ready for a data transfer.
STOP#	I/O 5V	**STOP:** Asserted by the target to request the master to stop the current transaction.
LOCK#	I/O 5V	**LOCK:** Used to establish, maintain, and release resource locks on PCI.
REQ[3:0]#	I 5V	**REQUEST:** PCI master requests for PCI.
GNT[3:0]#	O 5V	**GRANT:** Permission is given to the master to use PCI.
PHLD#	I 5V	**PCI HOLD**: This signal comes from the PIIX. PHLD# is the PIIX request for the PCI Bus. The TSC flushes the DRAM Write Buffers and acquires the host bus before granting PIIX via PHLDA#. This ensures that the guaranteed access time is met for ISA masters.
PHLDA#	O 5V	**PCI HOLD ACKNOWLEDGE:** This signal is driven by the TSC to grant PCI to the PIIX.
PAR	I/O 5V	**PARITY:** A single parity bit is provided over AD[31:0] and C/BE[3:0].
RST#	I 5V	**RESET:** When asserted, RST# resets the TSC and sets all register bits to the default value.

2.1.5 TDP INTERFACE (TSC)

Signal Name	Type	Description
PLINK[15:0]	I/O 3V	**PCI LINK:** These signals are connected to the PLINK data bus on the TDP. This is the data path between the TSC and TDP. Each TDP connects to one byte of the 16-bit bus.
MSTB#	O 3V	**MEMORY STROBE:** Assertion causes data to be posted in the DRAM Write Buffer.
MADV#	O 3V	**MEMORY ADVANCE:** For memory write cycles, assertion causes a Qword to be drained from the DRAM Write Buffer and the next data to be made available to the MD pins of the TDPs. For memory read cycles, assertion causes a Qword to be latched in the DRAM Input Register.
PCMD[1:0]	O 3V	**PLINK COMMAND:** This field controls how data is loaded into the PLINK input and output registers.
HOE#	O 3V	**HOST OUTPUT ENABLE:** This signal is used as the output enable for the Host Data Bus.
MOE#	O 3V	**MEMORY OUTPUT ENABLE:** This signal is used as the output enable for the memory data bus. A buffered copy of MOE# also serves as a WE# select for the DRAM array.
POE#	O 3V	**PLINK OUTPUT ENABLE:** This signal is used as the output enable for the PLINK Data Bus.

2.1.6 CLOCKS (TSC)

Signal Name	Type	Description
HCLKIN	I 5V	**HOST CLOCK IN:** This pin receives a buffered host clock. This clock is used by all of the TSC logic that is in the Host clock domain. This should be the same clock net that is delivered to the CPU. The net should tee and have equal lengths from the tee to the CPU and the TSC.
PCLKIN	I 5V	**PCI CLOCK IN:** This pin receives a buffered divide-by-2 host clock. This clock is used by all of the TSC logic that is in the PCI clock domain.

2.2 TDP Signals

2.2.1 DATA INTERFACE SIGNALS (TDP)

Signal Name	Type	Description
HD[31:0]	I/O 3V	**HOST DATA**: These signals are connected to the CPU data bus. The CPU data bus is interleaved between the two TDPs for every byte, effectively creating an even and an odd TDP.
MD[31:0]	I/O 3V/5V	**MEMORY DATA:** These signals are connected to the DRAM data bus. The DRAM data bus is interleaved between the two TDPs for every byte, effectively creating an even and an odd TDP.
PLINK[7:0]	I/O 3V	**PCI LINK:** These signals are connected to the PLINK data bus on the TSC. This is the data path between the TSC and TDP. Each TDP connects to one byte of the 16-bit bus.

2.2.2 TSC INTERFACE SIGNALS (TDP)

Signal Name	Type	Description
MSTB#	I 3V	**MEMORY STROBE:** Assertion causes data to be posted in the DRAM Write Buffer.
MADV#	I 3V	**MEMORY ADVANCE:** For memory write cycles, assertion causes a Qword to be flushed from the DRAM Write Buffer and the next data to be made available to the MD pins of the TDPs. For memory read cycles, assertion causes a Qword to be latched in the DRAM Input register.
PCMD[1:0]	I 3V	**PLINK COMMAND:** This field controls how data is loaded into the PLINK input and output registers.
HOE#	I 3V	**HOST OUTPUT ENABLE:** This signal is used as the output enable for the Host Data Bus.
MOE#	I 3V	**MEMORY OUTPUT ENABLE:** This signal is used as the output enable for the Memory Data Bus.
POE#	I 3V	**PLINK OUTPUT ENABLE:** This signal is used as the output enable for the PLINK Data Bus.

2.2.3 CLOCK SIGNAL (TDP)

Signal Name	Type	Description
HCLK	I 5V	**HOST CLOCK**: Primary clock input used to drive the part.

2.3 Signal State During Reset

Table 1 shows the state of all TSC and TDP output and bi-directional signals during a hard reset (RST# asserted). The TSC samples the strapping options on the A[31:28] signal lines on the rising edge of RST#. When RST# is asserted, the TSC enables the TDP outputs via the HOE#, MOE#, and POE# TSC/TDP interface signals. When RST# is negated, the TSC resets the TDP logic by driving HOE#, MOE#, and POE# inactive for two HCLKs.

Table 1. Output and I/O Signal States During Hard Reset

Signal	State
TSC	
A[31:28]	Input
A[27:3]	Low
BRDY#	High
NA#	High
AHOLD	High
EADS#	High
BOFF#	High
KEN#/INV	Low
RAS[4:0]#	Low
CAS[7:0]#	Low
MA[11:2]	Low
MAA[1:0]	Low
MAB[1:0]	Low
CADV#/CAA4	High
CADS#/CAA3	High

Signal	State
CCS#/CAB4	High
CAB3	High
COE#	High
CWE[7:0]#	High
TIO[7:0]#	Tri-state
TWE#	High
AD[31:0]	Low
C/BE[3:0]#	Low
FRAME#	Tri-state
DEVSEL#	Tri-state
IRDY#	Tri-state
TRDY#	Tri-state
STOP#	Tri-state
LOCK#	Tri-state
GNT[3:0]#	Tri-state
PHLDA#	Tri-state

Signal	State
PAR	Low
PLINK[15:0]	Low
MSTB#	High
MADV#	High
PCMD[1:0]	High
HOE#	High
MOE#	Low
POE#	Low
TDP	
HD[31:0]	Tri-state Low
MD[31:1]	Tri-state Low
MD0	NAND Tree Output
PLINK[7:0]	Tri-state Low

2

3.0 REGISTER DESCRIPTION

The TSC contains two sets of software accessible registers (Control and Configuration registers), accessed via the Host CPU I/O address space. Control Registers control access to PCI configuration space. Configuration Registers reside in PCI configuration space and specify PCI configuration, DRAM configuration, cache configuration, operating parameters, and optional system features.

The TSC internal registers (both I/O Mapped and Configuration registers) are only accessible by the Host CPU and cannot be accessed by PCI masters. The registers can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities, with the exception of CONFADD which can only be accessed as a Dword. All multi-byte numeric fields use "little-endian" ordering (i.e., lower addresses contain the least significant parts of the field). The following nomenclature is used for access attributes.

RO **Read Only**. If a register is read only, writes to this register have no effect.

R/W **Read/Write**. A register with this attribute can be read and written.

R/WC **Read/Write Clear**. A register bit with this attribute can be read and written. However, a write of 1 clears (sets to 0) the corresponding bit and a write of 0 has no effect.

Some of the TSC registers described in this section contain reserved bits. Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back.

In addition to reserved bits within a register, the TSC contains address locations in the PCI configuration space that are marked "Reserved" (Table 2). The TSC responds to accesses to these address locations by completing the Host cycle. Software should not write to reserved TSC configuration locations in the device-specific region (above address offset 3Fh).

During a hard reset (RST# asserted), the TSC sets its internal configuration registers to predetermined **default** states. The default state represents the minimum functionality feature set required to successfully bring up the system. Hence, it does not represent the optimal system configuration. It is the responsibility of the system initialization software (usually BIOS) to properly determine the DRAM configurations, cache configuration, operating parameters and optional system features that are applicable, and to program the TSC registers accordingly.

3.1 Control Registers

The TSC contains two registers that reside in the CPU I/O address space—the Configuration Address (CONFADD) Register and the Configuration Data (CONFDATA) Register. These registers can not reside in PCI configuration space because of the special functions they perform. The Configuration Address Register enables/disables the configuration space and determines what portion of configuration space is visible through the Configuration Data window.

3.1.1 CONFADD—CONFIGURATION ADDRESS REGISTER

I/O Address: 0CF8h (Dword access only)
Default Value: 00000000h
Access: Read/Write

CONFADD is a 32-bit register accessed only when referenced as a Dword. A Byte or Word reference will "pass through" the Configuration Address Register to the PCI Bus. The CONFADD Register contains the Bus Number, Device Number, Function Number, and Register Number for which a subsequent configuration access is intended.

Bit	Descriptions
31	**Configuration Enable (CONE):** 1 = Enable; 0 = Disable.
30:24	**Reserved.**
23:16	**Bus Number (BUSNUM):** When BUSNUM is programmed to 00h, the target of the configuration cycle is either the TSC or the PCI Bus that is directly connected to the TSC, depending on the Device Number field. If the Bus Number is programmed to 00h and the TSC is not the target, a type 0 configuration cycle is generated on PCI. If the Bus Number is non-zero, a type 1 configuration cycle is generated on PCI with the Bus Number mapped to AD[23:16] during the address phase.
15:11	**Device Number (DEVNUM):** This field selects one agent on the PCI Bus selected by the Bus Number. During a Type 1 Configuration cycle, this field is mapped to AD[15:11]. During a Type 0 configuration cycle, this field is decoded and one of AD[31:11] is driven to 1. The TSC is always Device Number 0.
10:8	**Function Number (FUNCNUM):** This field is mapped to AD[10:8] during PCI configuration cycles. This allows the configuration registers of a particular function in a multi-function device to be accessed. The TSC responds to configuration cycles with a function number of 000b; all other function number values attempting access to the TSC (Device Number = 0, Bus Number = 0) generate a type 0 configuration cycle on the PCI Bus with no IDSEL asserted, which results in a master abort.
7:2	**Register Number (REGNUM):** This field selects one register within a particular bus, device, and function as specified by the other fields in the Configuration Address Register. This field is mapped to AD[7:2] during PCI configuration cycles.
1:0	**Reserved.**

3.1.2 CONFDATA—CONFIGURATION DATA REGISTER

I/O Address: CFCh
Default Value: 00000000h
Access: Read/Write

CONFDATA is a 32-bit read/write window into configuration space. The portion of configuration space that is referenced by CONFDATA is determined by the contents of CONFADD.

Bit	Descriptions
31:0	**Configuration Data Window (CDW):** If bit 31 of CONFADD is 1, any I/O reference in the CONFDATA I/O space is mapped to configuration space using the contents of CONFADD.

3.2 PCI Configuration Registers

The PCI Bus defines a slot based "configuration space" that allows each device to contain up to 256 8-bit configuration registers. The PCI specification defines two bus cycles to access the PCI configuration space—**Configuration Read** and **Configuration Write**. While memory and I/O spaces are supported by the Pentium microprocessor, configuration space is not supported. The PCI specification defines two mechanisms to access configuration space, Mechanism #1 and Mechanism #2. The TSC only supports Mechanism #1. Table 2 shows the TSC configuration space.

The configuration access mechanism makes use of the CONFADD Register and CONFDATA Register. To reference a configuration register, a Dword I/O write cycle is used to place a value into CONFADD that specifies the PCI Bus, the device on that bus, the function within the device, and a specific configuration register of the device function being accessed. CONFADD[31] must be 1 to enable a configuration cycle. Then, CONFDATA becomes a window onto four bytes of configuration space specified by the contents of CONFADD. Read/write accesses to CONFDATA generates a PCI configuration cycle to the address specified by CONFADD.

Type 0 Access

If the Bus Number field of CONFADD is 0, a type 0 configuration cycle is generated on PCI. CONFADD[10:2] is mapped directly to AD[10:2]. The Device Number field of CONFADD is decoded onto AD[31:11]. The TSC is Device #0 and does not pass its configuration cycles to PCI. Thus, AD11 is never asserted. (For accesses to device #1, AD12 is asserted, etc., to Device #20 which asserts AD31.) Only one AD line is asserted at a time. All device numbers higher than 20 cause a type 0 configuration access with no IDSEL asserted, which results in a master abort.

Type 1 Access

If the Bus Number field of CONFADD is non-zero, a type 1 configuration cycle is generated on PCI. CONFADD[23:2] are mapped directly to AD[23:2]. AD[1:0] are driven to 01 to indicate a Type 1 Configuration cycle. All other lines are driven to 0.

Table 2. TSC Configuration Space

Address Offset	Symbol	Register Name	Access
00–01h	VID	Vendor Identification	RO
02–03h	DID	Device Identification	RO
04–05h	PCICMD	Command Register	R/W
06–07h	PCISTS	Status Register	RO, R/WC
08h	RID	Revision Identification	RO
09h		Reserved	
0Ah	SUBC	Sub-Class Code	RO
0Bh	BCC	Base Class Code	RO
0Ch		Reserved	
0Dh	MLT	Master Latency Timer	R/W
0Eh		Reserved	
0Fh	BIST	BIST Register	R/W
10–49h		Reserved	
50h	PCON	PCI Control Register	R/W
51h		Reserved	
52h	CC	Cache Control	R/W
53–56h		Reserved	
57h	DRAMC	DRAM Control	R/W
58h	DRAMT	DRAM Timing	R/W
59–5Fh	PAM[6:0]	Programmable Attribute Map (7 registers)	R/W
60–64h	DRB[4:0]	DRAM Row Boundary (5 registers)	R/W
65–67h		Reserved	
68h	DRT	DRAM Row Type	R/W
69–71h		Reserved	
72h	SMRAM	System Management RAM Control	R/W
73–FFh		Reserved	

3.2.1 VID—VENDOR IDENTIFICATION REGISTER

Address Offset: 00–01h
Default Value: 8086h
Attribute: Read Only

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identify any PCI device. Writes to this register have no effect.

Bit	Description
15:0	**Vendor Identification Number**: This is a 16-bit value assigned to Intel.

3.2.2 DID—DEVICE IDENTIFICATION REGISTER

Address Offset: 02–03h
Default Value: 122h
Attribute: Read Only

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect.

Bit	Description
15:0	**Device Identification Number**. This is a 16-bit value assigned to the TSC.

3.2.3 PCICMD—PCI COMMAND REGISTER

Address Offset: 04–05h
Default: 06h
Access: Read/Write

This register controls the TSC's ability to respond to PCI cycles.

Bit	Descriptions
15:10	**Reserved.**
9	**Fast Back-to-Back: (Not Implemented)** This bit is hardwired to 0.
8	**SERR# Enable (SERRE): (Not Implemented)** This bit is hardwired to 0.
7	**Address/Data Stepping: (Not Implemented)** This bit is hardwired to 0.
6	**Parity Error Enable (PERRE): (Not Implemented)** This bit is hardwired to 0.
5:3	**Reserved:** These bits are hardwired to 0.
2	**Bus Master Enable (BME): (Not Implemented)** The TSC does not support disabling of its bus master capability on the PCI Bus. This bit is hardwired to 1.
1	**Memory Access Enable (MAE):** 1 = Enable PCI master access to main memory, if the PCI address selects enabled DRAM space; 0 = Disable (TSC does not respond to main memory accesses).
0	**I/O Access Enable (IOAE): (Not Implemented)** This bit is hardwired to 0. The TSC does not respond to PCI I/O cycles.

3.2.4 PCISTS—PCI STATUS REGISTER

Address Offset: 06–07h
Default Value: 0200h
Access: Read Only, Read/Write Clear

PCISTS reports the occurrence of a PCI master abort and PCI target abort. PCISTS also indicates the DEVSEL# timing that has been set by the TSC hardware.

Bit	Descriptions
15	**Detected Parity Error (DPE): (Not Implemented)** This bit is hardwired to 1.
14	**Signaled System Error (SSE)R/WC:** This bit is hardwired to 0.
13	**Received Master Abort Status (RMAS)—R/WC:** When the TSC terminates a Host-to-PCI transaction (TSC is a PCI master) with an unexpected master abort, this bit is set to 1. **NOTE:** Master abort is the normal and expected termination of PCI special cycles. Software sets this bit to 0 by writing 1 to it.
12	**Received Target Abort Status (RTAS)—R/WC:** When a TSC-initiated PCI transaction is terminated with a target abort, RTAS is set to 1. Software sets RTAS to 0 by writing 1 to it.
11	**Signaled Target Abort Status (STAS):** This bit is hardwired to 0. The TSC never terminates a PCI cycle with a target abort.
10:9	**DEVSEL# Timing (DEVT)—RO:** This 2-bit field indicates the timing of the DEVSEL# signal when the TSC responds as a target, and is hard-wired to the value 01b (medium) to indicate the slowest time that DEVSEL# is generated.
8	**Data Parity Detected (DPD)—R/WC:** This bit is hardwired to 0
7	**Fast Back-to-Back (FB2B): (Not Implemented)** This bit is hardwired to 0.
6:0	**Reserved.**

3.2.5 RID—REVISION IDENTIFICATION REGISTER

Address Offset: 08h
Default Value: See stepping information document
Access: Read Only

This register contains the revision number of the TSC.

Bit	Description
7:0	**Revision Identification Number:** This is an 8-bit value that indicates the revision identification number for the TSC.

3.2.6 SUBC—SUB-CLASS CODE REGISTER

Address Offset: 0Ah
Default Value: 00h
Access: Read Only

This register indicates the function sub-class in relation to the Base Class Code.

Bit	Description
7:0	**Sub-Class Code (SUBC):** 00h = Host bridge.

3.2.7 BCC—BASE CLASS CODE REGISTER

Address Offset: 0Bh
Default Value: 06h
Access: Read Only

This register contains the Base Class Code of the TSC.

Bit	Description
7:0	**Base Class Code (BASEC):** 06h = Bridge device.

3.2.8 MLT—MASTER LATENCY TIMER REGISTER

Address Offset: 0Dh
Default Value: 00h
Access: Read/Write

MLT is an 8-bit register that controls the amount of time the TSC, as a bus master, can burst data on the PCI Bus. The Count Value is an 8-bit quantity. However, MLT[2:0] are hardwired to 0. MLT is also used to guarantee the host CPU a minimum amount of the system resources as described in the PCI Bus Arbitration section.

Bit	Description
7:3	**Master Latency Timer Count Value:** The number of clocks programmed in the MLT represents the minimum guaranteed time slice (measured in PCI clocks) allotted to the TSC, after which it must surrender the bus as soon as other PCI masters are granted the bus. The default value of MLT is 00h or 0 PCI clocks. The MLT should always be programmed to a non-zero value.
2:0	**Reserved:** Hardwired to 0.

3.2.9 BIST—BIST REGISTER

Address Offset: 0Fh
Default: 00h
Access: Read/Write

The Built In Self Test (BIST) function is not supported by the TSC. Writes to this register have no affect.

Bit	Descriptions
7	**BIST Supported—RO:** 00h = Disable BIST function.
6	**Start BIST:** This function is not supported and writes have no affect.
5:4	**Reserved.**
3:0	**Completion Code—RO:** This field always returns 0 when read and writes have no affect.

3.2.10 PCON—PCI CONTROL REGISTER

Address Offset: 50h
Default: 00h
Access: Read/Write

The PCON Register enables and disables features related to the PCI unit operation not already covered in the PCI required configuration space.

Bit	Descriptions
7:5	**CPU Inactivity Timer Bits:** This field selects the value used in the CPU Inactivity Timer. This timer counts CPU inactivity in PCI clocks. The inactivity window is defined as the last BRDY# to the next ADS#. When active, the CPU is default owner of the PCI Bus. If the CPU is inactive, PHOLD and the REQx# lines are given priority. **Bits[7:5] PCI Clocks** 000 1 001 2 010 3* 011 4 100 5* 101 6 110 7 111 8 * Recommended settings
4	**Reserved.**
3	**Peer Concurrency Enable Bit (PCE):** 1 = Peer Concurrency enabled. 0 = Peer Concurrency disabled. This bit is normally programmed to 1.
2	**CPU-to-PCI Write Bursting Disable Bit:** 0 = CPU-to-PCI Write bursting enabled. 1 = CPU-to-PCI Write bursting disabled.
1	**PCI Streaming Bit:** 0 = PCI streaming enabled. 1 = PCI streaming disabled.
0	**Bus Concurrency Disable Bit:** 0 = Bus concurrency enabled. 1 = Bus concurrency disabled.

3.2.11 CC—CACHE CONTROL REGISTER

Address Offset: 52h
Default: SSSS0010 (S = Strapping option)
Access: Read/Write

The CC Register selects the secondary cache operations. This register enables/disables the L2 cache, adjusts cache size, defines the cache SRAM type, and controls tag initialization. After a hard reset, CC[7:4] reflect the inverted signal levels on the host address lines A[31:28].

Bit	Description
7:6	**Secondary Cache Size (SCS):** This field reflects the inverted signal level on the A[31:30] pins at the rising edge of the RESET signal (default). The default values can be overwritten with subsequent writes to the CC Register. The options for this field are: **Bits[7:6] — Secondary Cache Size** 0 0 — Cache not populated 0 1 — 256 Kbytes 1 0 — 512 Kbytes 1 1 — Reserved **NOTE:** 1. When SCS = 00, the L2 cache is disabled and the cache tag state is frozen. 2. To enable the L2 cache, SCS must be non-zero and the FLCE bit must be 1.
5:4	**SRAM Type (SRAMT):** This field reflects the inverted signal level on the A[29:28] pins at the rising edge of the RESET signal (default). The default values can be overwritten with subsequent writes to the CC Register. The options for this field are: **Bits[5:4] — SRAM Type** 0 0 — Pipelined Burst 0 1 — Burst 1 0 — Asynchronous 1 1 — Pipelined Burst for 512K/Dual-bank inplementations. Selects 3-1-1-1-2-1-1-1 instead of 3-1-1-1-1-1-1-1 back-to-back burst timings with NA# enabled. An extra clock is inserted for bank turnaround. SCS must be set to 10.
3	**NA# Disable Bit:** 0 = NA# will be asserted as appropriate by the TSC (default). 1 = TCS's NA# signal is never asserted. This bit should be configured as desired before either the L1 or L2 caches are enabled.
2	**Reserved.**
1	**Secondary Cache Force Miss or Invalidate (SCFMI):** When SCFMI = 1, the L2 hit/miss detection is disabled, and all tag lookups result in a miss. If the L2 is enabled, the cycle is processed as a miss. If the L2 is populated but disabled (FLCE = 0) and SCFMI = 1, any CPU read cycle invalidates the selected tag entry. When SCFMI = 0, normal L2 cache hit/miss detection and cycle processing occurs. Software can flush the cache (cause all modified lines to be written back to main memory) by setting SCFMI to 1 with the L2 cache enabled (SCS≠00 and FLCE = 1), and reading all L2 cache tag address locations.

Bit	Description
0	**First Level Cache Enable (FLCE):** FLCE enables/disables the first level cache. When FLCE = 1, the TSC responds to CPU cycles with KEN# asserted for cacheable memory cycles. When FLCE = 0, KEN# is always negated and line fills to either the first level or L2 cache are prevented. Note that, when FLCE = 1 and SCFMI = 1, writes to the cache are also forced as misses. Thus, it is possible to create incoherent data between main memory and the L2 cache. A summary of FLCE/SCFMI bit interactions is as follows: **FLCE SCFMI L2 Cache Result** 0 0 Disabled 0 1 Disabled; tag invalidate on reads 1 0 Normal L2 cache operation (dependent on SCS) 1 1 Enabled; miss forced on reads/writes

3.2.12 DRAMC—DRAM CONTROL REGISTER

Address Offset: 57h
Default Value: 01h
Access: Read/Write

This 8-bit register controls main memory DRAM operating modes and features.

Bit	Description
7:6	**Hole Enable (HEN):** This field enables a memory hole in main memory space. CPU cycles matching an enabled hole are passed on to PCI. PCI cycles matching an enabled hole are ignored by the TSC (no DEVSEL#). Note that a selected hole is not remapped. Note that this field should not be changed while the L2 cache is enabled. **Bits[7:6] Hole Enabled** 00 None 01 512 Kbytes –640 Kbytes 10 15 Mbytes –16 Mbytes 11 Reserved
5:4	**Reserved.**
3	**EDO Detect Mode Enable (EDME):** This bit, if set to 1, enables a special timing mode for BIOS to detect EDO DRAM type on a bank-by-bank basis. Once all DRAM row banks have been tested for EDO, the EDME bit should be set to 0. Otherwise, performance will be seriously impacted. An algorithm for using the EDME bit 3 follows the table.
2:0	**DRAM Refresh Rate (DRR):** The DRAM refresh rate is adjusted according to the frequency selected by this field. Note that refresh is also disabled via this field, and that disabling refresh results in the eventual loss of DRAM data. **Bits[2:0] Host Bus Frequency** 000 Refresh Disabled 001 50 MHz 010 60 MHz 011 66 MHz 1XX Reserved

DRAM Type Detection

The EDO Detect Mode Enable field (bit 3) provides a special timing mode that allows BIOS to determine the DRAM type in each of the banks of main memory DRAM. To exploit the performance improvements from EDO DRAMs, the BIOS should provide for dynamic detection of any EDO DRAMs in the DRAM rows.

3.2.13 DRAMT—DRAM TIMING REGISTER

Address Offset: 58h
Default Value: 00h
Access: Read/Write

This 8-bit register controls main memory DRAM timings. While most system designs will be able to use one of the faster burst mode timings, slower rates may be required in certain system designs to support layouts with longer trace lengths or slower DRAMs.

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>7</td><td>Reserved.</td></tr>
<tr><td>6:5</td><td>DRAM Read Burst Timing (DRBT): The DRAM read burst timings are controlled by the DRBT field. The timing used depends on the type of DRAM on a per-bank basis, as indicated by the DRT register.

<table>
<tr><th>DRBT</th><th>EDO Burst Rate</th><th>Standard Page Mode Rate</th></tr>
<tr><td>00</td><td>x444</td><td>x444</td></tr>
<tr><td>01</td><td>x333</td><td>x444</td></tr>
<tr><td>10</td><td>x222</td><td>x333</td></tr>
<tr><td>11</td><td>Reserved</td><td>Reserved</td></tr>
</table>
This field is typically set to "01" or "10" depending on the system configuration.</td></tr>
<tr><td>4:3</td><td>DRAM Write Burst Timing (DWBT): The DRAM write burst timings are controlled by the DWBT field. Slower rates may be required in certain system designs to support layouts with longer trace lengths or slower DRAMs. Most system designs will be able to use one of the faster burst mode timings.

<table>
<tr><th>DWBT</th><th>Standard Page Mode Rate</th></tr>
<tr><td>00</td><td>x444</td></tr>
<tr><td>01</td><td>x333</td></tr>
<tr><td>10</td><td>x222 (see notes 1 and 2)</td></tr>
<tr><td>11</td><td>Reserved</td></tr>
</table>
NOTES:
1. Minimum 3-Clock CAS# Cycle Time for Single Writes. The DWBT field controls the minimum CAS# cycle time for single and burst write cycles, except for the x222 programming case in which the minimum cycle time for single writes is limited to 3-clocks.
2. Two clock writes should not be programmed at 66 MHz.</td></tr>
<tr><td>2</td><td>RAS to CAS Delay (RCD): RCD controls the DRAM page miss and row miss leadoff timings. When RCD = 1, the RAS active to CAS active delay is 2 clocks. When RCD = 0, the timing is 3 clocks. Note that RCD timing adjustments are independent to DLT timing adjustments.

<table>
<tr><th>RCD</th><th>RAS to CAS Delay</th></tr>
<tr><td>0</td><td>3</td></tr>
<tr><td>1</td><td>2</td></tr>
</table></td></tr>
</table>

Bit	Description
1:0	**DRAM Leadoff Timing (DLT):** The DRAM leadoff timings for page/row miss cycles are controlled by the DLT bits. DLT controls the MA setup to the first CAS# assertion. The DLT bits do not effect page hit cycles. **DLT — Read Leadoff — Write Leadoff — RAS# Precharge** 00 — 8 — 6 — 3 01 — 7 — 5 — 3 10 — 8 — 6 — 4 11 — 7 — 5 — 4 Note that the DLT field and RCD bit have cumulative affects (i.e., setting DLT0 = 0 and RCD = 0 results in two additional clocks betwwen RAS# assertion and CAS# assertion).

3.2.14 PAM—PROGRAMMABLE ATTRIBUTE MAP REGISTERS (PAM[6:0])

Address Offset: PAM0 (59h)—PAM6 (5Fh)
Default Value: 00h
Attribute: Read/Write

The TSC allows programmable memory and cacheability attributes on 14 memory segments of various sizes in the 640-Kbyte to 1-Mbyte address range. Seven Programmable Attribute Map (PAM) Registers are used to support these features. Three bits are used to specify L1 cacheability and memory attributes for each memory segment. These attributes are:

RE **Read Enable**. When RE = 1, the CPU read accesses to the corresponding memory segment are directed to main memory. Conversely, when RE = 0, the CPU read accesses are directed to PCI.

WE **Write Enable**. When WE = 1, the CPU write accesses to the corresponding memory segment are directed to main memory. Conversely, when WE = 0, the CPU write accesses are directed to PCI.

CE **Cache Enable**. When CE = 1, the corresponding memory segment is L1 cacheable. CE must not be set to 1 when RE = 0 for any particular memory segment. When CE = 1 and WE = 0, the corresponding memory segment is cached in the first level cache only on CPU code read cycles.

The RE and WE attributes permit a memory segment to be Read Only, Write Only, Read/Write, or disabled (Table 3.). For example, if a memory segment has RE = 1 and WE = 0, the segment is Read Only.

Table 3. Attribute Definition

Read/Write Attribute	Definition
Read Only	**Read cycles:** CPU cycles are serviced by the main memory or second level cache in a normal manner. **Write cycles:** CPU initiated write cycles are ignored by the DRAM interface as well as the second level cache. Instead, the cycles are passed to PCI for termination. Areas marked as read only are L1 cacheable for code accesses only. These regions are not cached in the second level cache.
Write Only	**Read cycles:** All read cycles are ignored by main memory as well as the second level cache. CPU-initiated read cycles are passed onto PCI for termination. The write only state can be used while copying the contents of a ROM, accessible on PCI, to main memory for shadowing, as in the case of BIOS shadowing. **Write cycles**: CPU write cycles are serviced by main memory and L2 cache in a normal manner.
Read/Write	This is the normal operating mode of main memory. Both read and write cycles from the CPU and PCI are serviced by main memory and L2 cache interface.
Disabled	All read and write cycles to this area are ignored by the main memory and cache interface. These cycles are forwarded to PCI for termination.

Each PAM Register controls two regions, typically 16-Kbyte in size. Each of these regions has a 4-bit field. The four bits that control each region have the same encoding and are defined in Table 4.

PCI master access to main memory space is also controlled by the PAM Registers. If the PAM programming indicates a region is writeable, then PCI master writes are accepted (DEVSEL# generated). If the PAM programming indicates a region is readable, PCI master reads are accepted. If a PCI write to a non-writeable main memory region or a PCI read of a non-readable main memory region occurs, the TSC does not accept the cycle (DEVSEL# is not asserted). PCI master accesses to enabled memory hole regions are not accepted by the TSC.

Table 4. Attribute Bit Assignment

Bits [7,3] Reserved	Bits [6,2] Cache Enable	Bits [5,1] Write Enable	Bits [4,0] Read Enable	Description
x	x	0	0	Main memory disabled; accesses directed to PCI
x	0	0	1	Read only; main memory write protected; non-cacheable
x	1	0	1	Read only; main memory write protected; L1 cacheable for code accesses only
x	0	1	0	Write only
x	0	1	1	Read/write; non-cacheable
x	1	1	1	Read/write; cacheable

As an example, consider a BIOS that is implemented on the expansion bus. During the initialization process, BIOS can be shadowed in main memory to increase the system performance. To shadow BIOS, the attributes for that address range should be set to write only. BIOS is shadowed by first performing a read of that address. This read is forwarded to the expansion bus. The CPU then performs a write of the same address, which is directed to main memory. After BIOS is shadowed, the attributes for that memory area are set to read only so that all writes are forwarded to the expansion bus.

Table 5. PAM Registers and Associated Memory Segments

PAM Reg	Attribute Bits				Memory Segment	Comments	Offset
PAM0[3:0]	**Reserved**						59h
PAM0[7:4]	R	CE	WE	RE	0F0000–0FFFFFh	BIOS Area	59h
PAM1[3:0]	R	CE	WE	RE	0C0000–0C3FFFh	ISA Add-on BIOS	5Ah
PAM1[7:4]	R	CE	WE	RE	0C4000–0C7FFFh	ISA Add-on BIOS	5Ah
PAM2[3:0]	R	CE	WE	RE	0C8000–0CBFFFh	ISA Add-on BIOS	5Bh
PAM2[7:4]	R	CE	WE	RE	0CC000–0CFFFFh	ISA Add-on BIOS	5Bh
PAM3[3:0]	R	CE	WE	RE	0D0000–0D3FFFh	ISA Add-on BIOS	5Ch
PAM3[7:4]	R	CE	WE	RE	0D4000–0D7FFFh	ISA Add-on BIOS	5Ch
PAM4[3:0]	R	CE	WE	RE	0D8000–0DBFFFh	ISA Add-on BIOS	5Dh
PAM4[7:4]	R	CE	WE	RE	0DC000–0DFFFFh	ISA Add-on BIOS	5Dh
PAM5[3:0]	R	CE	WE	RE	0E0000–0E3FFFh	BIOS Extension	5Eh
PAM5[7:4]	R	CE	WE	RE	0E4000–0E7FFFh	BIOS Extension	5Eh
PAM6[3:0]	R	CE	WE	RE	0E8000–0EBFFFh	BIOS Extension	5Fh
PAM6[7:4]	R	CE	WE	RE	0EC000–0EFFFFh	BIOS Extension	5Fh

NOTE:
The CE bit should not be changed while the L2 cache is enabled.

DOS Application Area (00000–9FFFh)

Read, write, and cacheability attributes are always enabled and are not programmable for the 0–640-Kbyte DOS application region.

Video Buffer Area (A0000–BFFFFh)

This 128-Kbyte area is not controlled by attribute bits. CPU-initiated cycles in this region are always forwarded to PCI for termination. This area is not cacheable. See section 3.2.16 for details on the use of this range as SMRAM.

Expansion Area (C0000–DFFFFh)

This 128-Kbyte area is divided into eight 16-Kbyte segments. Each segment can be assigned one of four read/write states: read-only, write-only, read/write, or disabled memory that is disabled is not remapped. Cacheability status can also be specified for each segment.

Extended System BIOS Area (E0000–EFFFFh)

This 64-Kbyte area is divided into four 16-Kbyte segments. Each segment can be assigned independent cacheability, read, and write attributes. Memory segments that are disabled are not remapped elsewhere.

System BIOS Area (F0000–FFFFFh)

This area is a single 64-Kbyte segment. This segment can be assigned cacheability, read, and write attributes. When disabled, this segment is not remapped.

Extended Memory Area (100000–FFFFFFFFh)

The extended memory area can be split into several parts;

- Flash BIOS area from 4 Gbyte to 4 Gbyte - 512 Kbyte (aliased on ISA at 16 Mbyte - 15.5 Mbyte)
- Main Memory from 1 Mbyte to a maximum of 128 Mbytes
- PCI Memory space from the top of main memory to 4 Gbyte - 512 Kbyte

On power-up or reset the CPU vectors to the Flash BIOS area, mapped in the range of 4 Gbyte to 4 Gbyte - 512 Kbyte. This area is physically mapped on the expansion bus. Since these addresses are in the upper 4 Gbyte range, the request is directed to PCI.

The main memory space can occupy extended memory from a minimum of 1 Mbyte up to 128 Mbytes. This memory is cacheable.

PCI memory space from the top of main memory to 4Gbytes is always non-cacheable.

3.2.15 DRB—DRAM ROW BOUNDARY REGISTERS

Address Offset: 60h(DRB0)—64h(DRB4)
Default Value: 02h
Access: Read/Write

The TSC supports 5 rows of DRAM. Each row is 64 bits wide. The DRAM Row Boundary Registers define upper and lower addresses for each DRAM row. Contents of these 8-bit registers represent the boundary addresses in 4-Mbyte granularity. Note that bit 0 of each DRB must always be programmed to 0 for proper operation.

DRB0 = Total amount of memory in row 0 (in 4 Mbytes)
DRB1 = Total amount of memory in row 0 + row 1 (in 4 Mbytes)
DRB2 = Total amount of memory in row 0 + row 1 + row 2 (in 4 Mbytes)
DRB3 = Total amount of memory in row 0 + row 1 + row 2 + row 3 (in 4 Mbytes)
DRB4 = Total amount of memory in row 0 + row 1 + row 2 + row 3 + row4 (in 4 Mbytes)

The DRAM array can be configured with 512Kx32, 1Mx32, 2Mx32, and 4Mx32 SIMMs. Each register defines an address range that causes a particular RAS# line to be asserted (e.g., if the first DRAM row is 8 Mbytes in size then accesses within the 0 to 8-Mbyte range causes RAS0# to be asserted).

Bit	Description
7:6	**Reserved.**
5:0	**Row Boundary Address:** This 6-bit field is compared against address lines A[27:22] to determine the upper address limit of a particular row (i.e., DRB minus previous DRB = row size).

Row Boundary Address

These 6-bit values represent the upper address limits of the 5 rows (i.e., this row minus previous row = row size). Unpopulated rows have a value equal to the previous row (row size = 0). DRB4 reflects the maximum amount of DRAM in the system. The top of memory is determined by the value written into DRB4. If DRB4 is greater than 128 Mbytes, then 128 Mbytes of DRAM are available.

As an example of a general purpose configuration where 4 physical rows are configured for either single-sided or double-sided SIMMs, the memory array would be configured like the one shown Figure 2. In this configuration, the TSC drives two RAS# signals directly to the SIMM rows. If single-sided SIMMs are populated, the even RAS# signal is used and the odd RAS# is not connected. If double-sided SIMMs are used, both RAS# signals are used.

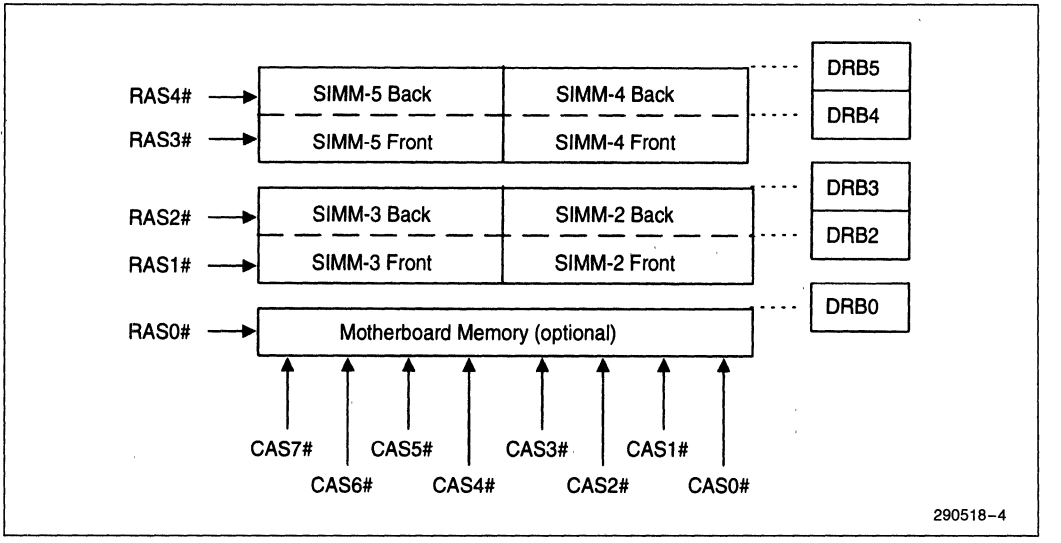


Figure 2. SIMMs and Corresponding DRB Registers

The following 2 examples describe how the DRB Registers are programmed for cases of single-sided and double-sided SIMMs on a motherboard having a total of four 8-byte or eight 4-byte SIMM sockets.

Example #1

The memory array is populated with four single-sided 1MB x 32 SIMMs, a total of 16 Mbytes of DRAM. Two SIMMs are required for each populated row making each populated row 8 Mbytes in size.

DRB0 = 02h	populated (2 SIMMs, 8 Mbyte this row)
DRB1 = 04h	populated (2 SIMMs, 8 Mbyte this row)
DRB2 = 04h	empty row
DRB3 = 04h	empty row
DRB4 = 04h	empty row

Example #2

The memory array is populated with two 2-Mbyte x 32 double-sided SIMMs (one row), and four 4-Mbyte x 32 single-sided SIMMs (two rows), yielding a total of 96 Mbytes of DRAM. The DRB Registers are programmed as follows:

DRB0 = 04h	populated with 16 Mbytes, ½ of double-sided SIMMs
DRB1 = 08h	the other 16 Mbytes of the double-sided SIMMs
DRB2 = 10h	populated with 32 Mbytes, one of the sided SIMMs
DRB3 = 18h	the other 32 Mbytes of single-sided SIMMs
DRB4 = 18h	empty row

2

3.2.16 DRT—DRAM ROW TYPE REGISTER

Address Offset: 68h

Default Value: 00h

Access: Read/Write

This 8-bit register identifies the type of DRAM (EDO or page mode) used in each row, and should be programmed by BIOS for optimum performance if EDO DRAMs are used. The TSC uses these bits to determine the correct cycle timing on DRAM cycles.

Bit	Description
7:5	**Reserved.**
4:0	**DRAM Row Type (DRT[4:0]):** Each bit in this field corresponds to the DRAM row identified by the corresponding DRB Register. Thus, DRT0 corresponds to row 0, DRT1 to row 1, etc. When DRTx = 0, page mode DRAM timings are used for that bank. When DRTx = 1, EDO DRAM timings are used for that bank.

3.2.17 SMRAM—SYSTEM MANAGEMENT RAM CONTROL REGISTER

Address Offset: 72h
Default Value: 02h
Access: Read/Write

The System Management RAM Control Register controls how accesses to this space are treated. The Open, Close, and Lock SMRAM Space bits function only when the SMRAM enable bit is set to 1. Also, the OPEN bit (DOPEN) should be set to 0 before the LOCK bit (DLCK) is set to 1.

Bit	Description
7	**Reserved.**
6	**SMM Space Open (DOPEN):** When DOPEN = 1 and DLCK = 0, SMM space DRAM is made visible, even when SMIACT# is negated. This is intended to help BIOS initialize SMM space. Software should ensure that DOPEN = 1 is mutually exclusive with DCLS = 1. When DLCK is set to 1, DOPEN is set to 0 and becomes read only.
5	**SMM Space Closed (DCLS):** When DCLS = 1, SMM space DRAM is not accessible to data references, even if SMIACT# is asserted. Code references may still access SMM space DRAM. This allows SMM software to reference "through" SMM space to update the display, even when SMM space is mapped over the VGA range. Software should ensure that DOPEN = 1 is mutually exclusive with DCLS = 1.
4	**SMM Space Locked (DLCK):** When DLCK is set to 1, the TSC sets DOPEN to 0 and both DLCK and DOPEN become read only. DLCK can be set to 1 via a normal configuration space write but can only be cleared by a power-on reset. The combination of DLCK and DOPEN provide convenience with security. The BIOS can use the DOPEN function to initialize SMM space and then use DLCK to "lock down" SMM space in the future so that no application software (or BIOS itself) can violate the integrity of SMM space, even if the program has knowledge of the DOPEN function.
3	**SMRAM Enable (SMRAME):** When SMRAME = 1, the SMRAM function is enabled, providing 128 Kbytes of DRAM accessible at the A0000h address while in SMM (ADS# with SMIACT#).
2:0	**SMM Space Base Segment (DBASESEG):** This field selects the location of SMM space. "SMM DRAM" is not remapped. It is simply "made visible" if the conditions are right to access SMM space. Otherwise, the access is forwarded to PCI. DBASESEG = 010 is the only allowable setting and selects the SMM space as A0000–BFFFFh. All other values are reserved. PCI masters are not allowed access to SMM space.

Table 6 summarizes the operation of SMRAM space cycles targeting SMI space addresses (A and B segments):

Table 6. SMRAM Space Cycles

SMRAME	DLCK	DCLS	DOPEN	SMIACT#	Code Fetch	Data Reference
0	X	X	X	X	PCI	PCI
1	0	0	0	0	DRAM	DRAM
1	0	X	0	1	PCI	PCI
1	0	0	1	X	DRAM	DRAM
1	0	1	0	0	DRAM	PCI
1	0	1	1	X	INVALID	INVALID
1	1	0	X	0	DRAM	DRAM
1	1	X	X	1	PCI	PCI
1	1	1	X	0	DRAM	PCI

4.0 FUNCTIONAL DESCRIPTION

This section provides a functional description of the TSC and TDP.

4.1 Host Interface

The Host Interface of the TSC is designed to support the Pentium processor. The host interface of the TSC supports 50 MHz, 60 MHz, and 66 MHz bus speeds. The 82430FX PCIset supports the Pentium processor with a full 64-bit data bus, 32-bit address bus, and associated internal write-back cache logic. Host bus addresses are decoded by the TSC for accesses to main memory, PCI memory, and PCI I/O. The TSC also supports the pipelined addressing capability of the Pentium processor.

4.2 PCI Interface

The 82437FX integrates a high performance interface to the PCI local bus taking full advantage of the high bandwidth and low latency of PCI. Five PCI masters are supported by the integrated arbiter including a PCI-to-ISA bridge and four general PCI masters. The TSC acts as a PCI master for CPU accesses to PCI. The PCI Bus is clocked at one half the frequency of the CPU clock. This divided synchronous interface minimizes latency for CPU-to-PCI cycles and PCI-to-main memory cycles.

The TSC/TDPs integrate posted write buffers for CPU memory writes to PCI. Back-to-back sequential memory writes to PCI are converted to burst writes on PCI. This feature allows the CPU to continue posting dword writes at the maximum bandwidth for the Pentium processor for the highest possible transfer rates to the graphics frame buffer.

Read prefetch and write posting buffers in the TSC/TDPs enable PCI masters to access main memory at up to 120 MB/second. The TSC incorporates a snoop-ahead feature which allows PCI masters to continue bursting on both reads and writes even as the bursts cross cache line boundaries.

4.3 Secondary Cache Interface

The TSC integrates a high performance second level cache controller using internal/external tags and provides a full first level and second level cache coherency mechanism. The second level cache is direct mapped, non-sectored, and supports a write-back cache policy. Cache lines are allocated on read misses (no write allocate).

The second level cache can be configured for either 256-Kbyte or 512-Kbyte cache sizes using either synchronous burst or pipelined burst SRAMs, or standard asynchronous SRAMs. For the 256-Kbyte configurations, an 8kx8 standard SRAM is used to store the tags. For the 512-Kbyte configurations, a 16kx8 standard SRAM is used to store the tags and the valid bits. A 5V SRAM is used for the Tag.

A second level cache line is 32 bytes wide. In the 256-Kbyte configurations, the second level cache contains 8K lines, while the 512-Kbyte configurations contain 16K lines. Valid and modified status bits are kept on a per-line basis. Cacheability of the entire memory space in the first level cache is supported. For the second level cache, only the lower 64 Mbytes of main memory are cacheable (only main memory controlled by the TSC DRAM interface is cached). PCI memory is not cached. Table 7 shows the different standard SRAM access time requirements for different host clock frequencies.

Table 7. SRAM Access Time Requirements

Host Clock Frequency (MHz)	Standard SRAM Access Time (ns)	Burst SRAM Clock-to-Output Access Time (ns)	Standard	Burst
			Tag RAM Access Time (ns)	Tag RAM Access Time (ns)
50	20 (17 ns Buffer)	13.5	30	20
60	17 (10 ns Buffer)	10	20	15
66	15 (7 ns Buffer)	8.5	15	15

4.3.1 CLOCK LATENCIES

Table 8 and Table 9 list the latencies for various processor transfers to and from the second level cache for standard and burst SRAM. The clock counts are identical for pipelined and non-pipelined burst SRAM.

Table 8. Second Level Cache Latencies with Standard SRAM

Cycle Type	HCLK Count
Burst Read	3-2-2-2
Burst Write (Write Back)	4-3-3-3
Single Read	3
Single Write	4

Table 9. Second Level Cache Latencies with Burst SRAM

Cycle Type	HCLK Count
Burst Read	3-1-1-1
Burst Write (Write Back)	3-1-1-1
Single Read	3
Single Write	3
Pipelined Back-to-Back Burst Reads	3-1-1-1, 1-1-1-1

4.3.2 SNOOP CYCLES

Snoop cycles are used to maintain coherency between the caches (first and second level) and main memory. The TSC generates a snoop (or inquire) cycle to probe the first level and second level caches when a PCI master attempts to access main memory. Snoop cycles are performed by driving the PCI master address onto the host address bus and asserting EADS#.

To maintain optimum PCI bandwidth to main memory, the TSC utilizes a "snoop ahead" algorithm. Once the snoop for the first cache line of a transfer has completed, the TSC automatically snoops the next sequential cache line. This algorithm enables the TSC to continue burst transfers across cache line boundaries.

Reads

If the snoop cycle generates a first level cache hit to a modified line, the line in the first level cache is written back to main memory (via the DRAM Posted Write Buffers). The line in the second level cache (if it exists) is invalidated. Note that the line in the first level cache is not invalidated if the INV pin on the CPU is tied to the KEN# signal from the TSC. The TSC drives KEN#/INV low with EADS# assertion during PCI master read cycles.

At the same time as the first level snoop cycle, the TSC performs a tag look-up to determine whether the addressed memory is in the second level cache. If the snoop cycle generates a second level cache hit to a modified line and there was not a hit in the first level cache (HITM# not asserted), the second level cache line is written back to main memory (via the DRAM Posted Write Buffers) and changed to the "clean" state. The PCI master read completes after the data has been written back to main memory.

Writes

PCI Master write cycles never result in a write directly into the second level cache. A snoop hit to a modified line in either the first or second caches results in a write-back of that line to main memory. If both the first and second level caches have modified lines, the line is written back from the first level cache. In all cases, lines in the first and second level caches are invalidated and the PCI write to main memory occurs after the writeback completes. A PCI master write snoop hit to an unmodified line in either the first or second level caches results in the line being invalidated. The TSC drives KEN#/INV with EADS# assertion during PCI master write cycles.

4.3.3 CACHE ORGANIZATION

Figure 3, Figure 4, Figure 5, Figure 6, and Figure 7 show the connections between the TSC and the external tag RAM and data SRAM. A 512K standard SRAM cache is implemented with 64Kx8 data SRAMs and a 16Kx8 tag RAM. The second ADS# pin from the CPU should be used to drive the ADSP# pin on Burst or Pipelined Burst SRAMs.

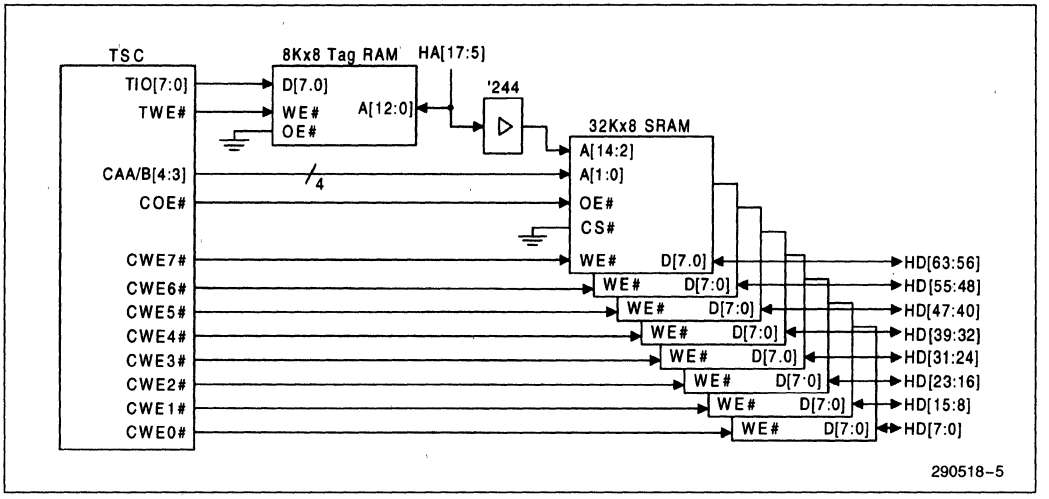


Figure 3. 256-Kbyte Second Level Cache (Standard SRAM)

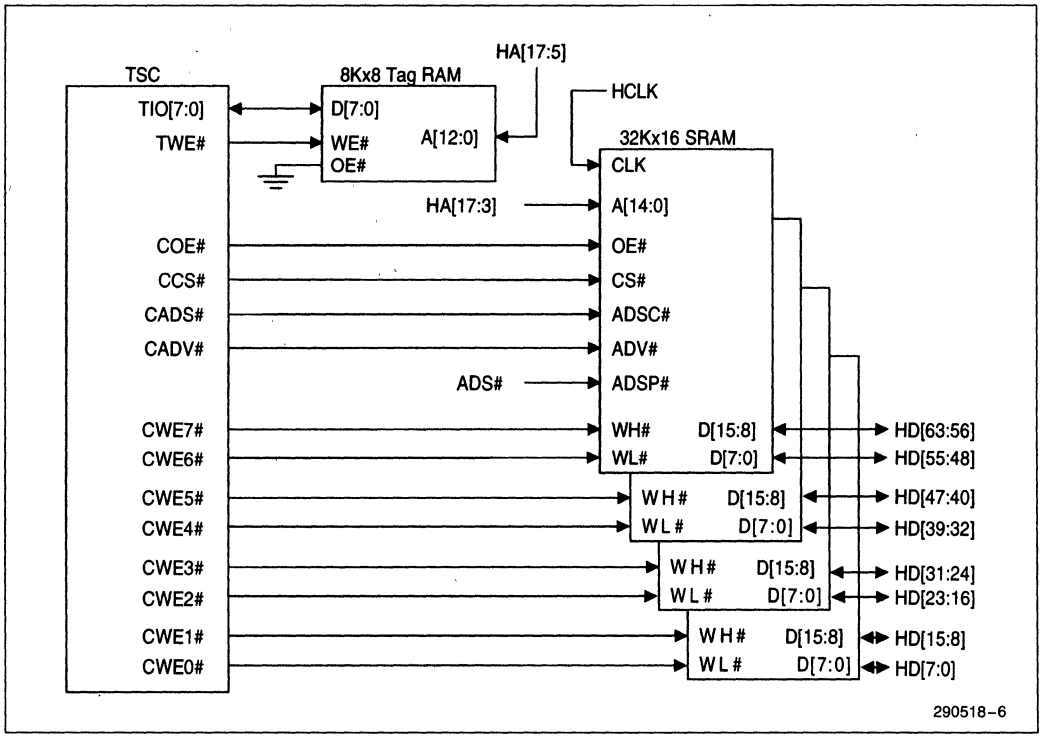


Figure 4. 256-Kbyte Second Level Cache (Burst SRAM)

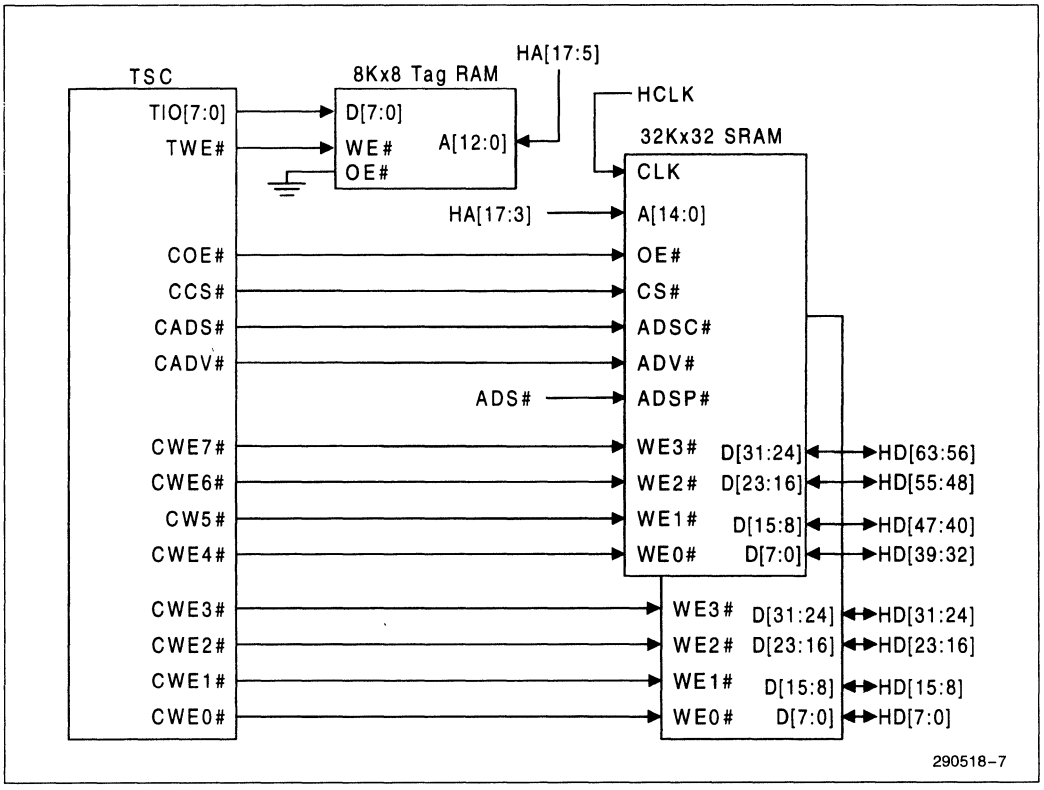


Figure 5. 256-Kbyte Second Level Cache (Burst SRAM)

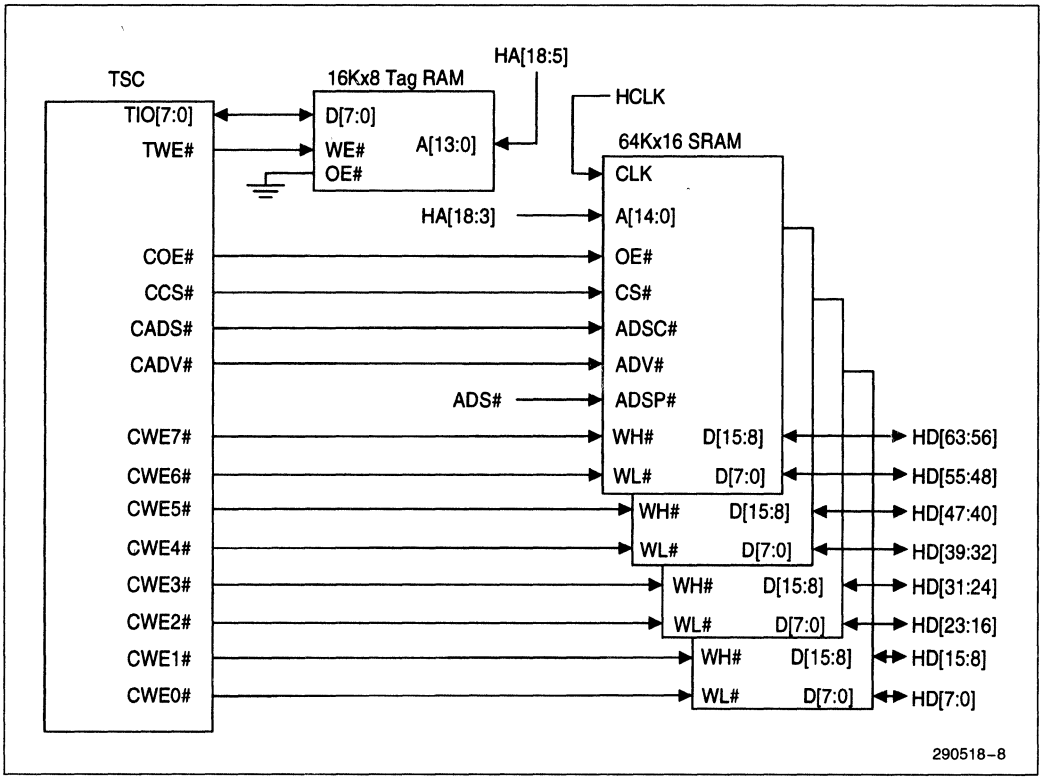


Figure 6. 512-Kbyte Second Level Cache (Burst SRAM)

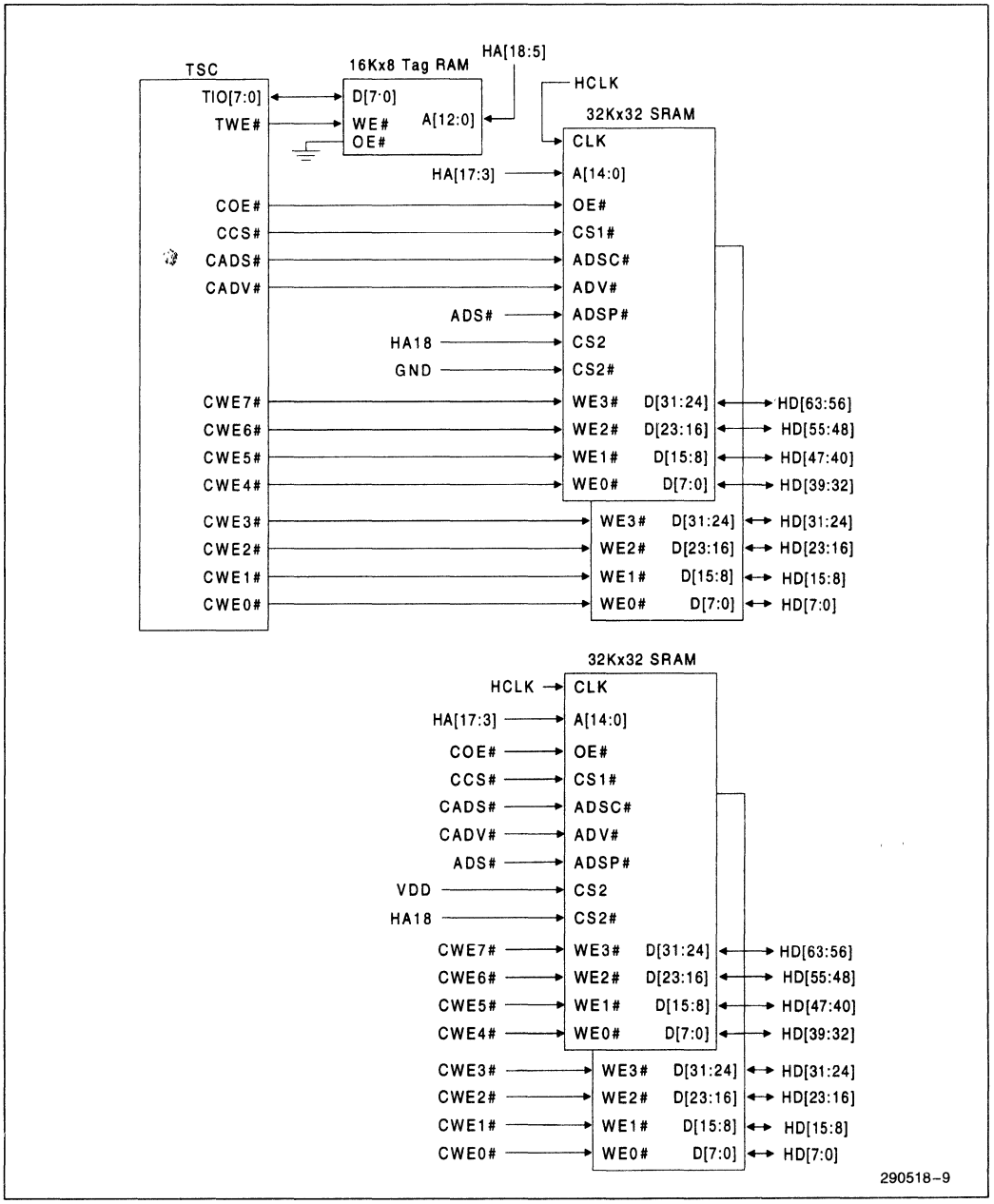


Figure 7. Two Bank 512-Kbyte Second Level Cache (Pipelined Burst SRAM)

4.4 DRAM Interface

The 82430FX PCIset's main memory DRAM interface supports a 64-bit wide memory array and main memory sizes from 4 to 128 Mbytes. The TSC generates the RAS#, CAS#, WE# (using MOE#) and multiplexed addresses for the DRAM array and controls the data flow through the 82438FX TDP's. For CPU-to-DRAM cycles the address flows through the TSC and data flows through the TDP's. For PCI or ISA cycles to memory the address flows through the TSC and data flows to the TDP's through the TSC and PLINK bus. The TSC and TDP DRAM interfaces are synchronous to the CPU clock.

The 82430FX PCIset supports industry standard 32-bit wide memory modules with fast page-mode DRAMs and EDO (Extended Data Out) DRAMs (also known as Hyper Page mode). With twelve multiplexed address lines (MA[11:0]), the TSC supports 512Kx32, 1M32, 2Mx32, and 4Mx32 SIMM's (both symmetrical and asymmetrical addressing). Five RAS# lines permit up to five rows of DRAM and eight CAS# lines provide byte write control over the array. The TSC supports 60 ns and 70 ns DRAMs (both single and double-sided SIMM's). The TSC also provides an automatic RAS# only refresh, at a rate of 1 refresh per 15.6 ms at 66 MHz, 60 MHz, and 50 MHz. A refresh priority queue and "smart refresh" algorithm are used to minimize the performance impact due to refresh.

The DRAM controller interface is fully configurable through a set of control registers (see Register Description section for programming details). The DRAM interface is configured by the DRAM Control Mode Register, the five DRAM Row Boundary (DRB) Registers, and the DRAM Row Type (DRT) Register. The DRAM Control Mode Registers configure the DRAM interface to select fast page-mode or EDO DRAMs, RAS timings, and CAS rates. The five DRB Registers define the size of each row in the memory array, enabling the TSC to assert the proper RAS# line for accesses to the array.

Seven Programmable Attribute Map (PAM) Registers are used to specify the cacheability, PCI enable, and read/write status of the memory space between 640 Kbytes and 1 Mbyte. Each PAM Register defines a specific address area enabling the system to selectively mark specific memory ranges as cacheable, read-only, write-only, read/write, or disabled. When a memory range is disabled, all CPU accesses to that range are forwarded to PCI.

The TSC also supports one of two memory holes; either from 512 Kbytes−640 Kbytes or from 15 Mbytes−16 Mbytes in main memory. Accesses to the memory holes are forwarded to PCI. The memory hole can be enabled/disabled through the DRAM Control Register. All other memory from 1 Mbyte to the top of main memory is read/write and cacheable.

The SMRAM memory space is controlled by the SMRAM Control Register. This register selects if the SMRAM space is enabled, opened, closed, or locked. SMRAM space is between 640 Kbytes and 768 Kbytes. See section 3.2.17 for more details on SMRAM.

4.4.1 DRAM ORGANIZATION

Figure 8 illustrates a 4-SIMM configuration that supports 4 double-sided SIMM's and motherboard DRAM. RAS0# is used for motherboard memory. This memory should not be implemented with SIMMs.

Except for motherboard memory, a row in the DRAM array is made up of two SIMM's that share a common RAS# line. Within any given row, the two SIMMs must be the same size. For different rows, SIMM densities can be mixed in any order. Each row is controlled by 8 CAS lines. EDO and Standard page mode DRAM's can be mixed between rows or within a row. When DRAM types are mixed (EDO and standard page mode), each row will run optimized for that particular type of DRAM. If DRAM types are mixed within a row, page mode timings must be selected.

SIMMs can be used for the sockets connected to RAS[2:1]# and RAS[4:3]#. The two RAS lines permit double-sided SIMMs to be used in these socket pairs. The following rules apply to the SIMM configuration.

1. SIMM sockets can be populated in any order.
2. SIMM socket pairs need to be populated with the same densities. For example, SIMM sockets RAS[2:1]# should be populated with identical densities. However, SIMM sockets using RAS[4:3]# can be populated with different densities than the SIMM socket pair using RAS[2:1]#.

3. The TSC only recognizes a maximum of 128 Mbytes of main memory, even if populated with more memory.
4. EDO's and standard page mode can both be used.

4.4.2 MAIN MEMORY ADDRESS MAP

The main memory organization (Figure 9) represents the maximum 128 Mbytes of address space. Accesses to memory space above the top of main memory, video buffer range, or the memory gaps (if enabled) are not cacheable and are forwarded to PCI. Below 1 Mbyte, there are several memory segments with selectable cacheability.

4.4.3 DRAM ADDRESS TRANSLATION

The multiplexed row/column address to the DRAM memory array is provided by MA[11:0] which are derived from the host address bus or PCI address as defined by Table 10. The TSC has a 4-Kbyte page size. The page offset address is driven on the MA[8:0] lines when driving the column address. The MA[11:0] lines are translated from the address lines A[24:3] for all memory accesses.

Table 10. DRAM Address Translation

Memory Address, MA[11:0]	11	10	9	8	7	6	5	4	3	2	1	0
Row Address	A24	A23	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12
Column Address	X	A24	A22	A11	A10	A9	A8	A7	A6	A5	A4	A3

The types of DRAMs depth configuration supported are:

Depth	Row Width	Column Width
512K	10	9
1M	10	10
2M	11	10
4M	11	11
4M	12	10

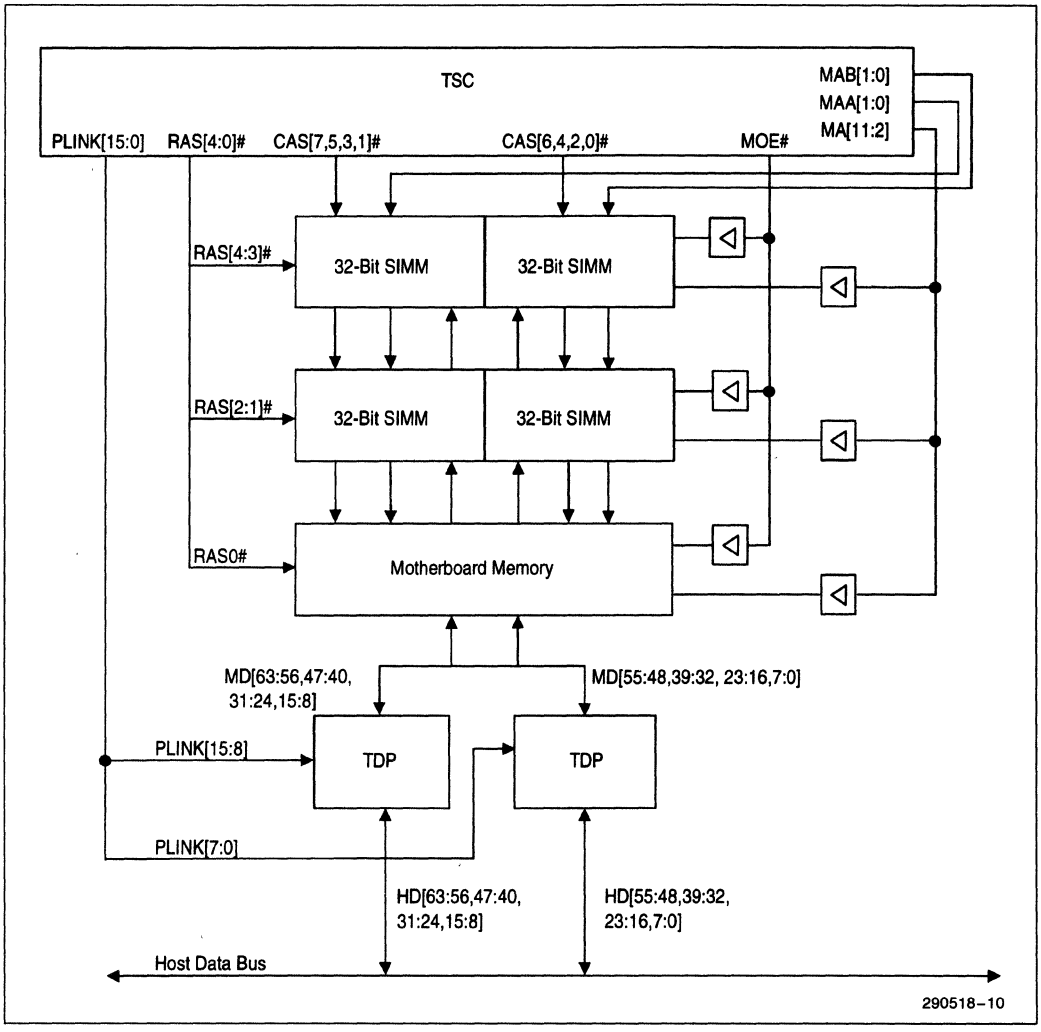


Figure 8. DRAM Array Connections

4.4.4 DRAM PAGE MODE

For any row containing standard page mode DRAM on read cycles, the TSC keeps CAS[7:0]# asserted until data is sampled by the TDPs.

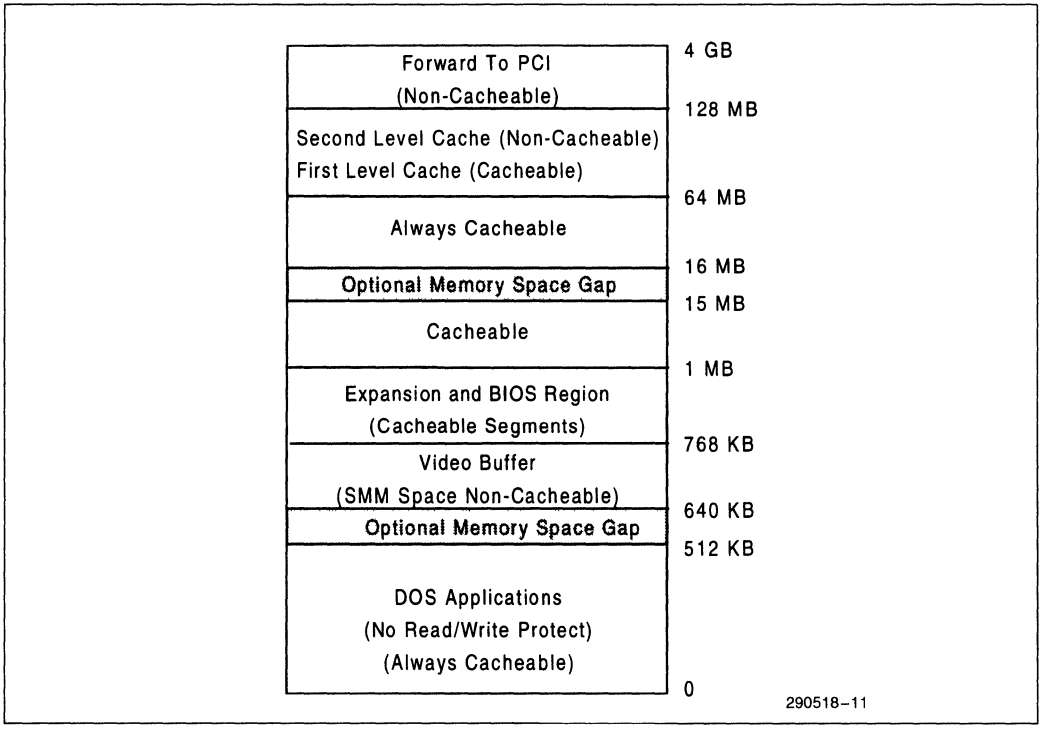


Figure 9. Memory Space Organization

4.4.5 EDO MODE

Extended Data Out (or Hyper Page Mode) DRAM is designed to improve the DRAM read performance. EDO DRAM holds the memory data valid until the next CAS# falling edge. Note that standard page mode DRAM tri-states the memory data when CAS# negates to precharge. With EDO, the CAS# precharge overlaps the memory data valid time. This allows CAS# to negate earlier while still satisfying the memory data valid window time.

The EDO Detect Mode Enable bit in the DRAM Control Register enables a special timing mode for BIOS to detect the DRAM type on a row by row basis.

4.4.6 DRAM PERFORMANCE

The DRAM performance is controlled by the DRAM Timing Register, processor pipelining, and by the type of DRAM used (EDO or standard page mode). Table 11 lists both EDO and standard page mode optimum timings.

Table 11. CPU to DRAM Performance Summary

Processor Cycle Type (Pipelined)	Clock Count (ADS# to BRDY#)	Comments
Burst read page hit	7-2-2-2	EDO
Read row miss	9-2-2-2 (note 1)	EDO
Read page miss	12-2-2-2	EDO
Back-to-back burst reads page hit	7-2-2-2-3-2-2-2	EDO
Burst read page hit	7-3-3-3	Standard page mode
Burst read row miss	9-3-3-3 (note 1)	Standard page mode
Burst read page miss	12-3-3-3	Standard page mode
Back-to-back burst read page hit	7-3-3-3-3-3-3-3	Standard page mode
Posted write	3-1-1-1	EDO/Standard page mode
Retire data for posted write	Every 3 clocks	EDO/Standard page mode

NOTES:
1. Due to the MA[11:2] to RAS# setup requirements, if a page is open, two clocks are added to the leadoff.
2. Read and write rates to DRAM are programmable via the DRAMT Register.

4.4.7 DRAM REFRESH

The TSC supports RAS# only refresh and generates refresh requests. The rate that requests are generated is determined by the DRAM Control Register. When a refresh request is generated, the request is placed in a four entry queue. The DRAM controller services a refresh request when the refresh queue is not empty and the controller has no other requests pending. When the refresh queue is full, refresh becomes the highest priority request and is serviced next by the DRAM controller, regardless of other pending requests. When the DRAM controller begins to service a refresh request, the request is removed from the refresh queue.

There is also a "smart refresh" algorithm implemented in the refresh controller. Except for Bank 0, refresh is only performed on banks that are populated. For bank 0, refresh is always performed. If only one bank is populated, using bank 0 will result in better performance.

4.4.8 SYSTEM MANAGEMENT RAM

The 82430FX PCIset support the use of main memory as System Management RAM (SMRAM), enabling the use of System Management Mode. When this function is disabled, the TSC memory map is defined by the DRB and PAM Registers. When SMRAM is enabled, the TSC reserves the A and B segments of main memory for use as SMRAM.

SMRAM is placed at A0000-BFFFFh via the SMRAM Space Register. Enhanced SMRAM features can also be enabled via this register. PCI masters can not access SMRAM when it is programmed to the A and B segments.

When the TSC detects a CPU stop grant special cycle, it generates a PCI Stop Grant Special cycle with 0002h in the message field (AD[15:0]) and 0012h in the message dependent data field (AD[31:16]) during the first data phase (IRDY# asserted).

4.5 82438FX Data Path (TDP)

The TDP's provide the data path for host-to-main memory, PCI-to-main memory, and host-to-PCI cycles. Two TDP's are required for the 82430FX PCIset system configuration. The TSC controls the data flow through the TDP's with the PCMD[1:0], HOE#, POE#, MOE#, MSTB#, and MADV# signals.

The TDP's have three data path interfaces; the host bus (HD[63:0]). the memory bus (MD[63:0]), and the PLINK[15:0] bus between the TDP and TSC. The data paths for the TDP's are interleaved on byte boundaries (Figure 10). Byte lanes 0, 2, 4, and 6 from the host CPU data bus connects to the even order TDP and byte lanes 1, 3, 5, and 7 connect to the odd order TDP. PLINK[7:0] connects to the even order TDP and PLINK[15:8] connect to the odd order TDP.

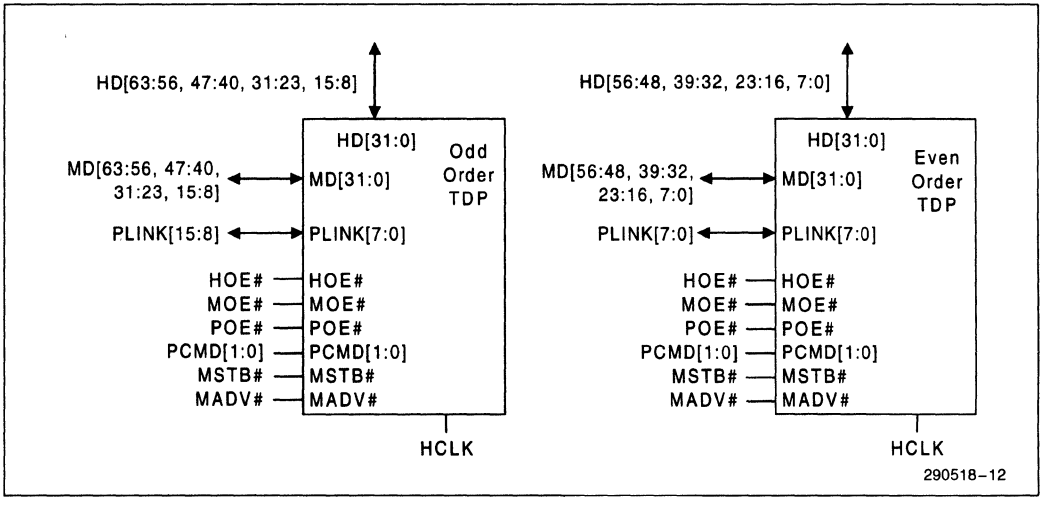


Figure 10. TDP 64-Bit Data Path Partitioning

4.6 PCI Bus Arbitration

The TSC's PCI Bus arbiter allows PCI peer-to-peer traffic concurrent with CPU main memory/second level cache cycles. The arbiter supports five PCI masters (Figure 11). REQ[3:0]#/GNT[3:0]# are used by PCI masters other than the PCI-to-ISA expansion bridge (PIIX). PHLD#/PHLDA# are the arbitration request/grant signals for the PIIX and provide guaranteed access time capability for ISA masters. PHLD#/PHLDA# also optimize system performance based on PIIX known policies.

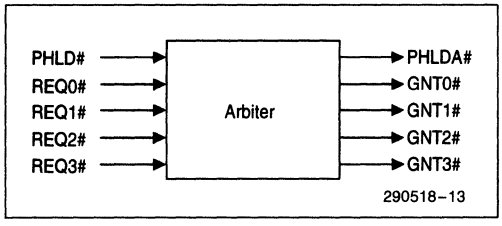


Figure 11. Arbiter

4.6.1 PRIORITY SCHEME AND BUS GRANT

The arbitration mechanism employs two interacting priority queues; one for the CPU and one for the PCI agents. The CPU queue guaranatees that the CPU is explicity granted the bus on every fourth arbitration event. The PCI priority queue determines which PCI agent is granted when PCI wins the arbitration event.

A rotating priority scheme is used to determine the highest priority requester in the case of simultaneous requests. If the highest priority input at arbitration time does not have an active request, the next priority active requester is granted the bus. Granting the bus to a lower priority requester does not change the rotation order, but it does advance the priority rotation. The rotation priority chain is fixed (Figure 12). If the highest priority agent does not request the bus, the next agent in the chain is the highest priority, and so forth down the chain.

When no PCI agents are requesting the bus, the CPU is the default owner of the bus. CPU cycles incur no additional delays in this state.

The grant signals (GNTx#) are normally negated after FRAME# assertion or 16 PCLKs from grant assertion, if no cycle has started. Once asserted, PHLDA# is only negated after PHLD# has been negated.

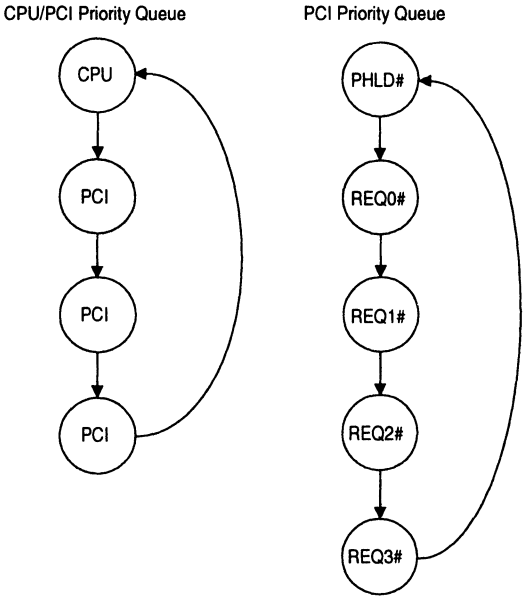
4.6.2 CPU POLICIES

The CPU never explicitly requests the bus. Instead, the arbiter grants the bus to the CPU when:

- the CPU is the highest priority
- PCI agents do not require main memory (peer-to-peer transfers or bus idle) and the PCI Bus is not currently locked by a PCI master

When the CPU is granted as highest priority, the MLT timer is used to guarantee a minimum amount of system resources to the CPU before another requesting PCI agent is granted.

An AHOLD mechanism controls granting the bus to the CPU.



NOTES:
1. For the CPU/PCI priority queue, the CPU is granted the PCI Bus after three consecutive PCI grants.
2. For the PCI priority queue, the last agent granted is always dropped to the bottom of the queue for the next arbitration cycle. However, the order of the chain is always preserved.

Figure 12. Arbitration Priority Rotation

4.7 Clock Generation and Distribution

The TSC and CPU should be clocked from one clock driver output to minimize skew between the CPU and TSC. The TDPs should share another clock driver output.

4.7.1 RESET SEQUENCING

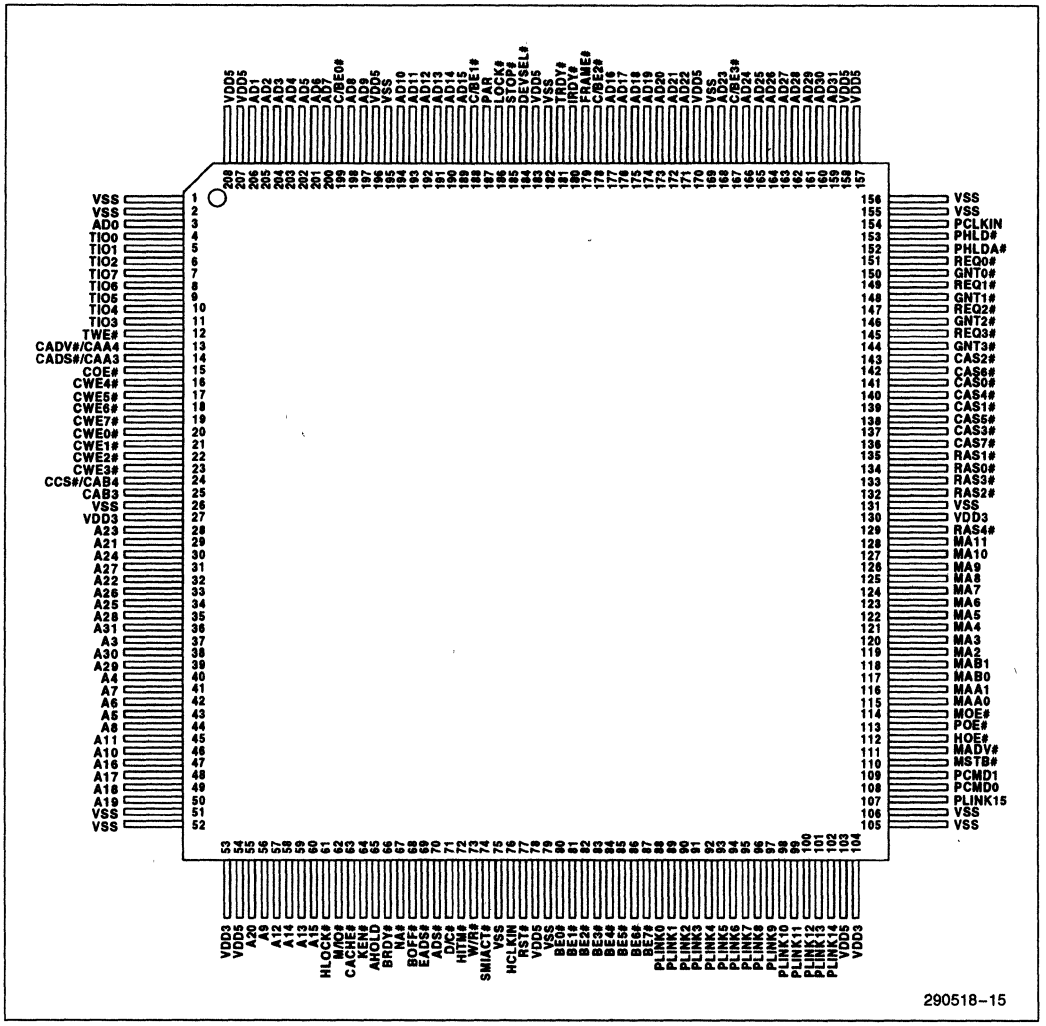
The TSC is asynchronously reset by the PCI reset (RST#). After RST# is negated, the TSC resets the TDP by driving HOE#, MOE#, and POE# to 1 for two HCLKs. The TSC changes HOE#, MOE#, and POE# to their default value after the TDP is reset.

Arbiter (Central Resource) Functions on Reset

The TSC arbiter includes support for PCI central resource functions. These functions include driving the AD[31:0], C/BE[3:0]#, and PAR signals when no agent is granted the PCI Bus and the bus is idle. The TSC drives 0's on these signals during reset and drives valid levels when no other agent is granted and the bus is idle.

5.0 PINOUT AND PACKAGE INFORMATION

5.1 82437FX Pinout



290518-15

Figure 13. 82437FX Pin Assignment

Table 12. 82437FX Alphabetical Pin Assignment

Name	Pin#	Type
A3	37	I/O
A4	40	I/O
A5	43	I/O
A6	42	I/O
A7	41	I/O
A8	44	I/O
A9	56	I/O
A10	46	I/O
A11	45	I/O
A12	57	I/O
A13	59	I/O
A14	58	I/O
A15	60	I/O
A16	47	I/O
A17	48	I/O
A18	49	I/O
A19	50	I/O
A20	55	I/O
A21	29	I/O
A22	32	I/O
A23	28	I/O
A24	30	I/O
A25	34	I/O
A26	33	I/O
A27	31	I/O
A28	35	I/O
A29	39	I/O
A30	38	I/O
A31	36	I/O

Name	Pin#	Type
AD0	3	I/O
AD1	206	I/O
AD2	205	I/O
AD3	204	I/O
AD4	203	I/O
AD5	202	I/O
AD6	201	I/O
AD7	200	I/O
AD8	198	I/O
AD9	197	I/O
AD10	194	I/O
AD11	193	I/O
AD12	192	I/O
AD13	191	I/O
AD14	190	I/O
AD15	189	I/O
AD16	177	I/O
AD17	176	I/O
AD18	175	I/O
AD19	174	I/O
AD20	173	I/O
AD21	172	I/O
AD22	171	I/O
AD23	168	I/O
AD24	166	I/O
AD25	165	I/O
AD26	164	I/O
AD27	163	I/O
AD28	162	I/O

Name	Pin#	Type
AD29	161	I/O
AD30	160	I/O
AD31	159	I/O
ADS#	70	I
AHOLD	65	O
BE0#	80	I
BE1#	81	I
BE2#	82	I
BE3#	83	I
BE4#	84	I
BE5#	85	I
BE6#	86	I
BE7#	87	I
BOFF#	68	O
BRDY#	66	O
C/BE0#	199	I/O
C/BE1#	188	I/O
C/BE2#	178	I/O
C/BE3#	167	I/O
CAB3	25	O
CACHE#	63	I
CADS# / CAA3	14	O
CADV# / CAA4	13	O
CAS0#	141	O
CAS1#	139	O
CAS2#	143	O
CAS3#	137	O
CAS4#	140	O

2

Table 12. 82437FX Alphabetical Pin Assignment (Continued)

Name	Pin#	Type
CAS5#	138	O
CAS6#	142	O
CAS7#	136	O
CCS#/ CAB4	24	O
COE#	15	O
CWE0#	20	O
CWE1#	21	O
CWE2#	22	O
CWE3#	23	O
CWE4#	16	O
CWE5#	17	O
CWE6#	18	O
CWE7#	19	O
D/C#	71	I
DEVSEL#	184	I/O
EADS#	69	O
FRAME#	179	I/O
GNT0#	150	O
GNT1#	148	O
GNT2#	146	O
GNT3#	144	O
HCLKIN	76	I
HITM#	72	I
HLOCK#	61	I
HOE#	112	O
IRDY#	180	I/O
KEN#	64	O
LOCK#	186	I/O
M/IO#	62	I
MA2	119	O

Name	Pin#	Type
MA3	120	O
MA4	121	O
MA5	122	O
MA6	123	O
MA7	124	O
MA8	125	O
MA9	126	O
MA10	127	O
MA11	128	O
MAA0	115	O
MAA1	116	O
MAB0	117	O
MAB1	118	O
MADV#	111	O
MOE#	114	O
MSTB#	110	O
NA#	67	O
PAR	187	I/O
PCLKIN	154	I
PCMD0	108	O
PCMD1	109	O
PHLD#	153	I
PHLDA#	152	O
PLINK0	88	I/O
PLINK1	89	I/O
PLINK2	90	I/O
PLINK3	91	I/O
PLINK4	92	I/O
PLINK5	93	I/O
PLINK6	94	I/O
PLINK7	95	I/O

Name	Pin#	Type
PLINK8	96	I/O
PLINK9	97	I/O
PLINK10	98	I/O
PLINK11	99	I/O
PLINK12	100	I/O
PLINK13	101	I/O
PLINK14	102	I/O
PLINK15	107	I/O
POE#	113	O
RAS0#	134	O
RAS1#	135	O
RAS2#	132	O
RAS3#	133	O
RAS4#	129	O
REQ0#	151	I
REQ1#	149	I
REQ2#	147	I
REQ3#	145	I
RST#	77	I
SMIACT#	74	I
STOP#	185	I/O
TIO0	4	I/O
TIO1	5	I/O
TIO2	6	I/O
TIO3	11	I/O
TIO4	10	I/O
TIO5	9	I/O
TIO6	8	I/O
TIO7	7	I/O
TRDY#	181	I/O
TWE#	12	O

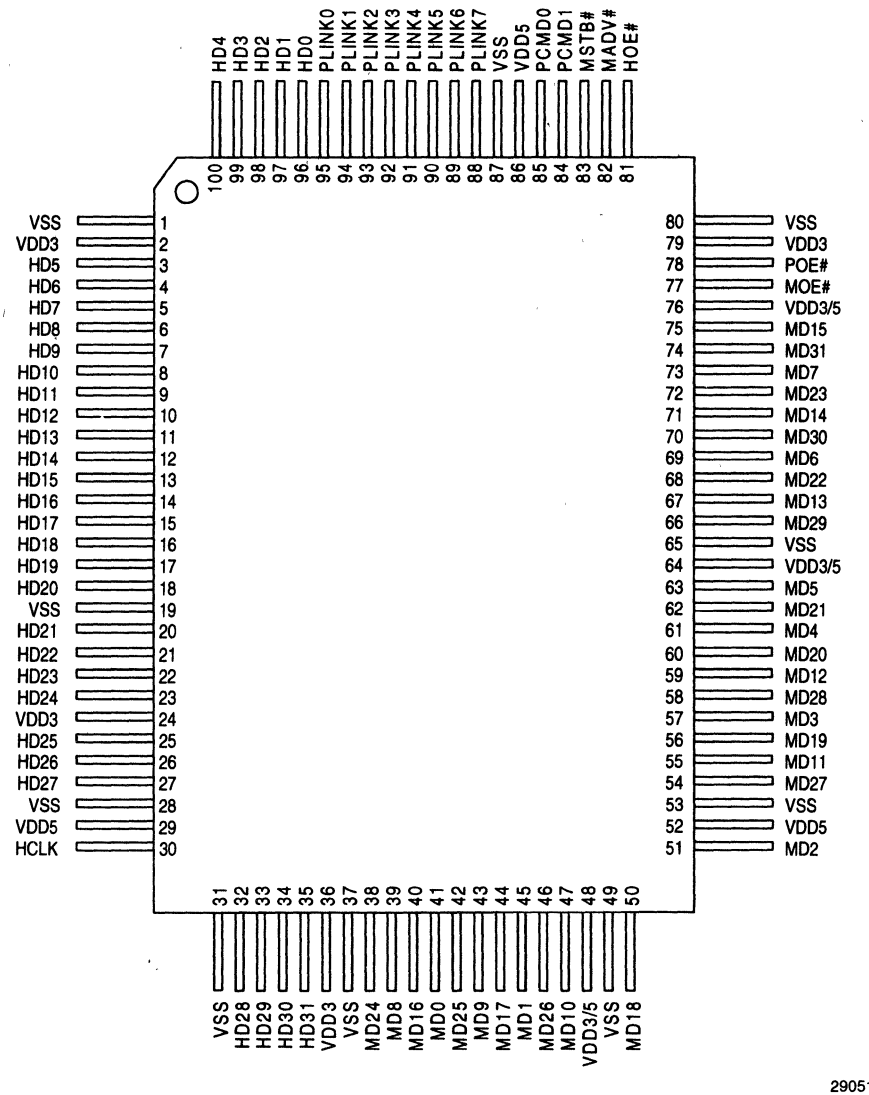
Table 12. 82437FX Alphabetical Pin Assignment (Continued)

Name	Pin#	Type
VDD3	27	V
VDD3	53	V
VDD3	104	V
VDD3	130	V
VDD5	54	V
VDD5	78	V
VDD5	103	V
VDD5	157	V
VDD5	158	V
VDD5	170	V

Name	Pin#	Type
VDD5	183	V
VDD5	196	V
VDD5	207	V
VDD5	208	V
VSS	1	V
VSS	2	V
VSS	26	V
VSS	51	V
VSS	52	V
VSS	75	V

Name	Pin#	Type
VSS	79	V
VSS	105	V
VSS	106	V
VSS	131	V
VSS	155	V
VSS	156	V
VSS	169	V
VSS	182	V
VSS	195	V
W/R#	73	I

5.2 82438FX Pinout



290518-16

NOTES:
VDD3 is connected to the 3.3V rail; VDD5 is connected to the 5V, VDD3/5 is connected to the 3.3V rail for 3.3V DRAMs and to the 5V rail for 5V DRAMs.

Figure 14. 82438FX Pin Assignment

Table 13. 82438FX Alphabetical Pin Assignment

Name	Pin#	Type
HCLK	30	I
HD0	96	I/O
HD1	97	I/O
HD2	98	I/O
HD3	99	I/O
HD4	100	I/O
HD5	3	I/O
HD6	4	I/O
HD7	5	I/O
HD8	6	I/O
HD9	7	I/O
HD10	8	I/O
HD11	9	I/O
HD12	10	I/O
HD13	11	I/O
HD14	12	I/O
HD15	13	I/O
HD16	14	I/O
HD17	15	I/O
HD18	16	I/O
HD19	17	I/O
HD20	18	I/O
HD21	20	I/O
HD22	21	I/O
HD23	22	I/O
HD24	23	I/O
HD25	25	I/O
HD26	26	I/O
HD27	27	I/O
HD28	32	I/O

Name	Pin#	Type
HD29	33	I/O
HD30	34	I/O
HD31	35	I/O
HOE#	81	I
MADV#	82	I
MD0	41	I/O
MD1	45	I/O
MD2	51	I/O
MD3	57	I/O
MD4	61	I/O
MD5	63	I/O
MD6	69	I/O
MD7	73	I/O
MD8	39	I/O
MD9	43	I/O
MD10	47	I/O
MD11	55	I/O
MD12	59	I/O
MD13	67	I/O
MD14	71	I/O
MD15	75	I/O
MD16	40	I/O
MD17	44	I/O
MD18	50	I/O
MD19	56	I/O
MD20	60	I/O
MD21	62	I/O
MD22	68	I/O
MD23	72	I/O
MD24	38	I/O

Name	Pin#	Type
MD25	42	I/O
MD26	46	I/O
MD27	54	I/O
MD28	58	I/O
MD29	66	I/O
MD30	70	I/O
MD31	74	I/O
MOE#	77	I
MSTB#	83	I
PCMD0	85	I
PCMD1	84	I
PLINK0	95	I/O
PLINK1	94	I/O
PLINK2	93	I/O
PLINK3	92	I/O
PLINK4	91	I/O
PLINK5	90	I/O
PLINK6	89	I/O
PLINK7	88	I/O
POE#	78	I
VDD3	2	V
VDD3	24	V
VDD5	29	V
VDD3	36	V
VDD3/5	48	V
VDD5	52	V
VDD3/5	64	V
VDD3/5	76	V
VDD3	79	V
VDD5	86	V

Table 13. 82438FX Alphabetical Pin Assignment (Continued)

Name	Pin #	Type
VSS	1	V
VSS	19	V
VSS	28	V
VSS	31	V

Name	Pin #	Type
VSS	37	V
VSS	49	V
VSS	53	V
VSS	65	V

Name	Pin #	Type
VSS	80	V
VSS	87	V

5.3 82437FX Package Dimensions

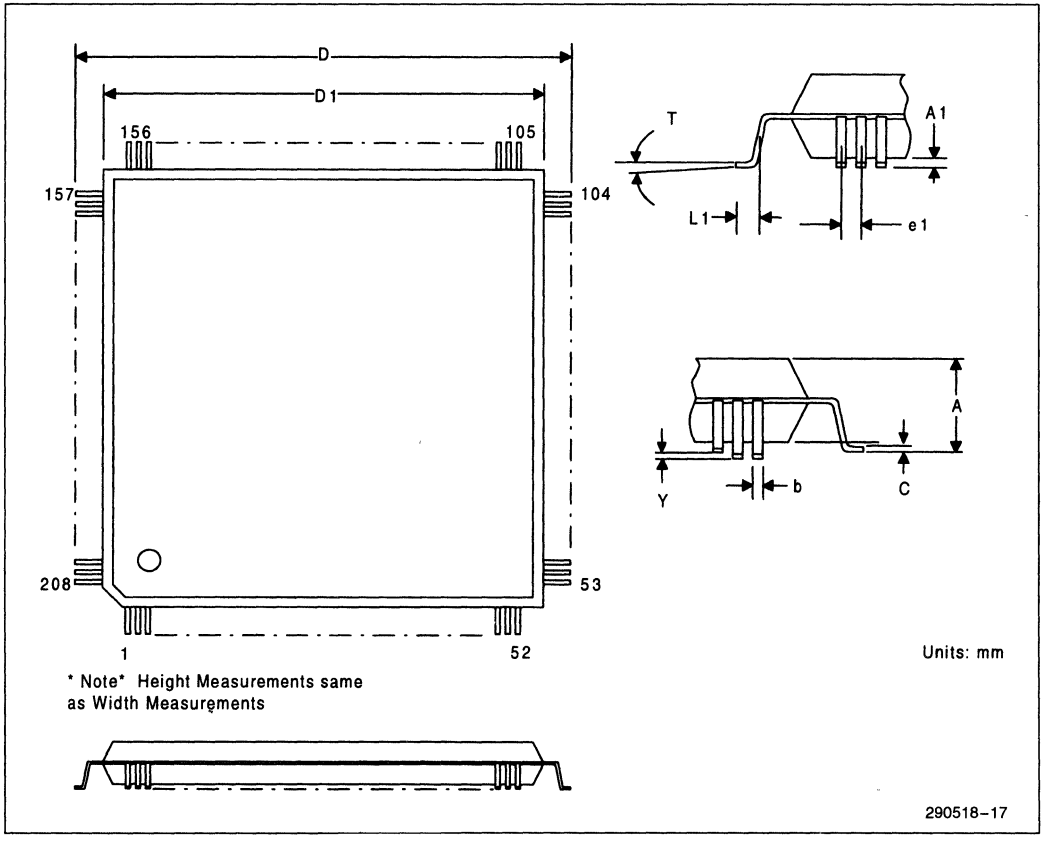


Figure 15. 208 Pin Quad Flat Pack (QFP) Dimensions

Table 14. 208 Pin Quad Flat Pack (QFP) Dimensions

Symbol	Description	Value (mm)
A	Seating Height	4.25 (max)
A1	Stand-off	0.05 (min); 0.40 (max)
b	Lead Width	0.2 ± 0.10
C	Lead Thickness	0.15 +0.1/-0.05
D	Package Length and Width, including pins	30.6 ± 0.4
D1	Package Length and Width, excluding pins	28 ± 0.2
e1	Linear Lead Pitch	0.5 ± 0.1
Y	Lead Coplanarity	0.08 (max)
L1	Foot Length	0.5 ± 0.2
T	Lead Angle	0° - 10°

5.4 82438FX Package Dimensions

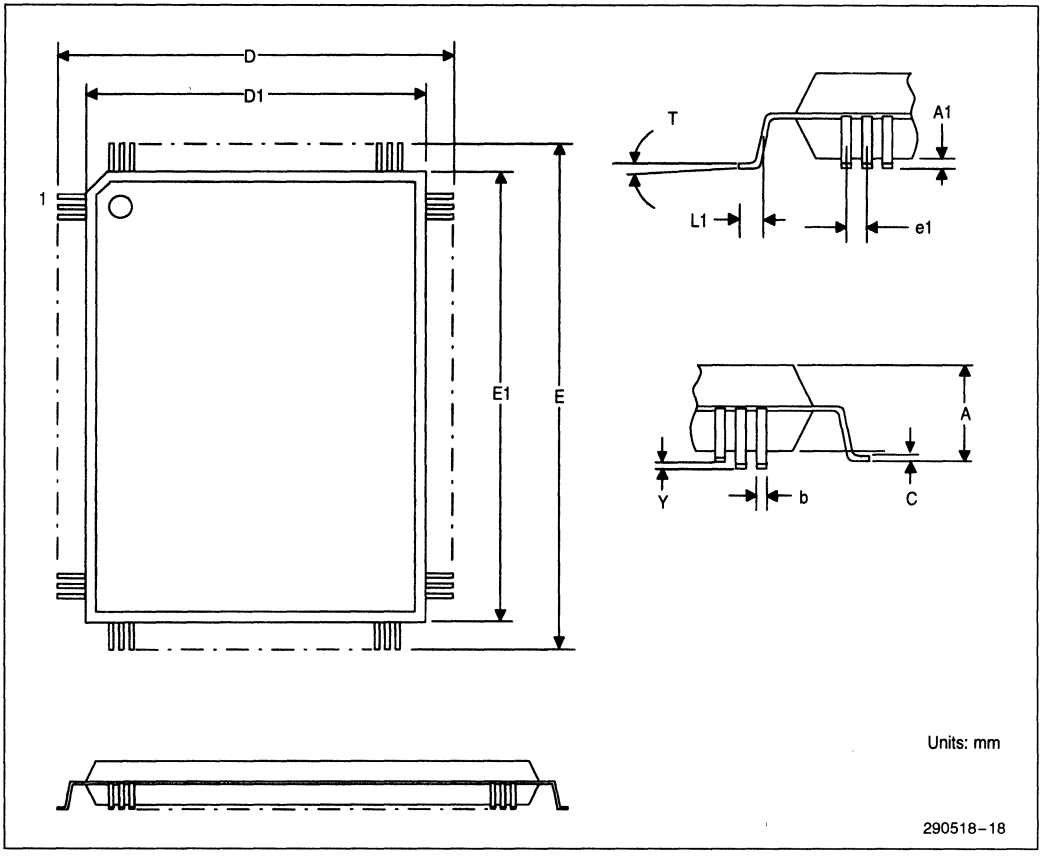


Figure 16. 100 Pin Plastic Quad Flat Pack (PQFP) Dimensions

Table 15. 100 Pin Quad Flat Pack (QFP) Dimensions

Symbol	Description	Value (mm)
A	Seating Height	3.3 (max)
A1	Stand-off	0.0 (min); 0.50 (max)
b	Lead Width	0.3 ± 0.10
C	Lead Thickness	0.15 +0.1/-0.05
D	Package Width, including pins	17.9 ± 0.4
D1	Package Width, excluding pins	14 0.2
E	Package Length, including pins	23.9 ± 0.4
E1	Package Length, excluding pins	20 ± 0.2
e1	Linear Lead Pitch	0.65 ± 0.12
Y	Lead Coplanarity	0.1 (max)
L1	Foot Length	0.8 0.2
T	Lead Angle	0° - 10°

6.0 82437FX TSC TESTABILITY

6.1 Test Mode Description

The test modes are decoded from the REQ#[3:0] and qualified with the RESET# pin. Test mode selection is asynchronous, these signals need to remain in their respective state for the duration of the test modes. The test modes are defined as follows.

Test Mode	RST#	REQ0#	REQ1#	REQ2#	REQ3#
NAND Tree	0	0	0	0	0

6.2 NAND Tree Mode

Tri-states all outputs and bi-directional buffers except for RST#, REQ#[3:0], GNT#[3:1]. The NAND tree follows the pins sequentially around the chip skipping only RESET#, REQ#[3:0], and GNT#[3:1]. The first input of the NAND chain GNT0#, and the NAND chain is routed counter-clockwise around the chip (e.g., GNT0#, PHLDA#, . . .). The only valid outputs during NAND tree mode are GNT1#, GNT2#, and GNT3#. GNT1# and GNT#3 are both final outputs of the NAND tree, and GNT2# is the halfway point of the NAND tree.

To perform a NAND tree test, all pins included in the NAND tree should be driven to 1 with the exception of PCLKIN which should be driven to 0.

Beginning with GNT0# and working counter-clockwise around the chip, each pin can be toggled with a resulting toggle observed on GNT3#, GNT2#, and GNT1#. The GNT2# output is provided so that the NAND tree test can be divided into two sections.

Table 16. NAND Tree Cell Order for the 82437FX

Pin #	Pin Name	Notes
77	RST#	Must be 0 to enter NAND tree mode. This signal must remain 0 during the entire NAND tree test.
145	REQ3#	Must be 0 to enter NAND tree mode. This signal must remain 0 during the entire NAND tree test.
147	REQ2#	Must be 0 to enter NAND tree mode. This signal must remain 0 during the entire NAND tree test.
149	REQ1#	Must be 0 to enter NAND tree mode. This signal must remain 0 during the entire NAND tree test.
151	REQ0#	Must be 0 to enter NAND tree mode. This signal must remain 0 during the entire NAND tree test.
150	GNT0#	Start of the NAND tree chain.
152	PHLDA#	
153	PHOLD#	
154	PCLKIN	PCLKIN needs to be 0 to pass the NAND-tree chain, a logic 1 will block the NAND-tree chain.
159	AD31	
160	AD30	
161	AD29	
162	AD28	
163	AD27	
164	AD26	
165	AD25	
166	AD24	
167	C/BE3#	
168	AD23	

Pin #	Pin Name	Notes
171	AD22	
172	AD21	
173	AD20	
174	AD19	
175	AD18	
176	AD17	
177	AD16	
178	C/BE2#	
179	FRAME#	
180	IRDY#	
181	TRDY#	
184	DEVSEL#	
185	STOP#	
186	LOCK#	
187	PAR	
188	C/BE1#	
189	AD15	
190	AD14	
191	AD13	
192	AD12	
193	AD11	
194	AD10	
197	AD9	
198	AD8	
199	C/BE0#	
200	AD7	
201	AD6	
202	AD5	
203	AD4	
204	AD3	
205	AD2	

Table 16. NAND Tree Cell Order for the 82437FX (Continued)

Pin #	Pin Name	Notes
206	AD1	
3	AD0	
4	TIO0	
5	TIO1	
6	TIO2	
7	TIO7	
8	TIO6	
9	TIO5	
10	TIO4	
11	TIO3	
12	TWE#	
13	CADV# / CAA4	
14	CADS# / CAA3	
15	COE#	
16	CWE4#	
17	CWE5#	
18	CWE6#	
19	CWE7#	
20	CWE0#	
21	CWE1#	
22	CWE2#	
23	CWE3#	
24	CCS# / CAB4	
25	CAB3	
28	A23	
29	A21	
30	A24	
31	A27	
32	A22	

Pin #	Pin Name	Notes
33	A26	
34	A25	
35	A28	
36	A31	
37	A3	
38	A30	
39	A29	
40	A4	
41	A7	
42	A6	
43	A5	
44	A8	
45	A11	
46	A10	
47	A16	
48	A17	
49	A18	
50	A19	
55	A20	
56	A9	
57	A12	
58	A14	
59	A13	
60	A15	
61	HLOCK#	
62	M/IO#	
63	CACHE#	
64	KEN#	
65	AHOLD	
66	BRDY#	

Table 16. NAND Tree Cell Order for the 82437FX (Continued)

Pin #	Pin Name	Notes
67	NA#	
68	BOFF#	
69	EADS#	
70	ADS#	
71	D/C#	
72	HITM#	
73	W/R#	
74	SMIACT#	
76	HCLKIN	
80	BE0#	
81	BE1#	
82	BE2#	
83	BE3#	
84	BE4#	
85	BE5#	
86	BE6#	
87	BE7#	
88	PLINK0	
89	PLINK1	
90	PLINK2	
91	PLINK3	
92	PLINK4	
93	PLINK5	
94	PLINK6	
95	PLINK7	
96	PLINK8	
97	PLINK9	
98	PLINK10	
99	PLINK11	
100	PLINK12	

Pin #	Pin Name	Notes
101	PLINK13	
102	PLINK14	
107	PLINK15	
108	PCMD0	
109	PCMD1	
110	MSTB#	
111	MADV#	
112	HOE#	
113	POE#	
114	MOE#	
115	MAA0	
116	MAA1	
117	MAB0	
118	MAB1	
119	MA2	
120	MA3	
121	MA4	
122	MA5	
123	MA6	
124	MA7	
125	MA8	
126	MA9	
127	MA10	
128	MA11	
129	RAS4#	
132	RAS2#	
133	RAS3#	
134	RAS0#	
135	RAS1#	
136	CAS7#	

Table 16. NAND Tree Cell Order for the 82437FX (Continued)

Pin #	Pin Name	Notes
137	CAS3#	
138	CAS5#	
139	CAS1#	
140	CAS4#	
141	CAS0#	
142	CAS6#	

Pin #	Pin Name	Notes
143	CAS2#	
144	GNT3#	Final output of the NAND tree chain.
146	GNT2#	Half way point of the NAND tree chain.
148	GNT1#	Final output of the NAND-tree chain.

Figure 17 is a schematic of the NAND tree circuitry.

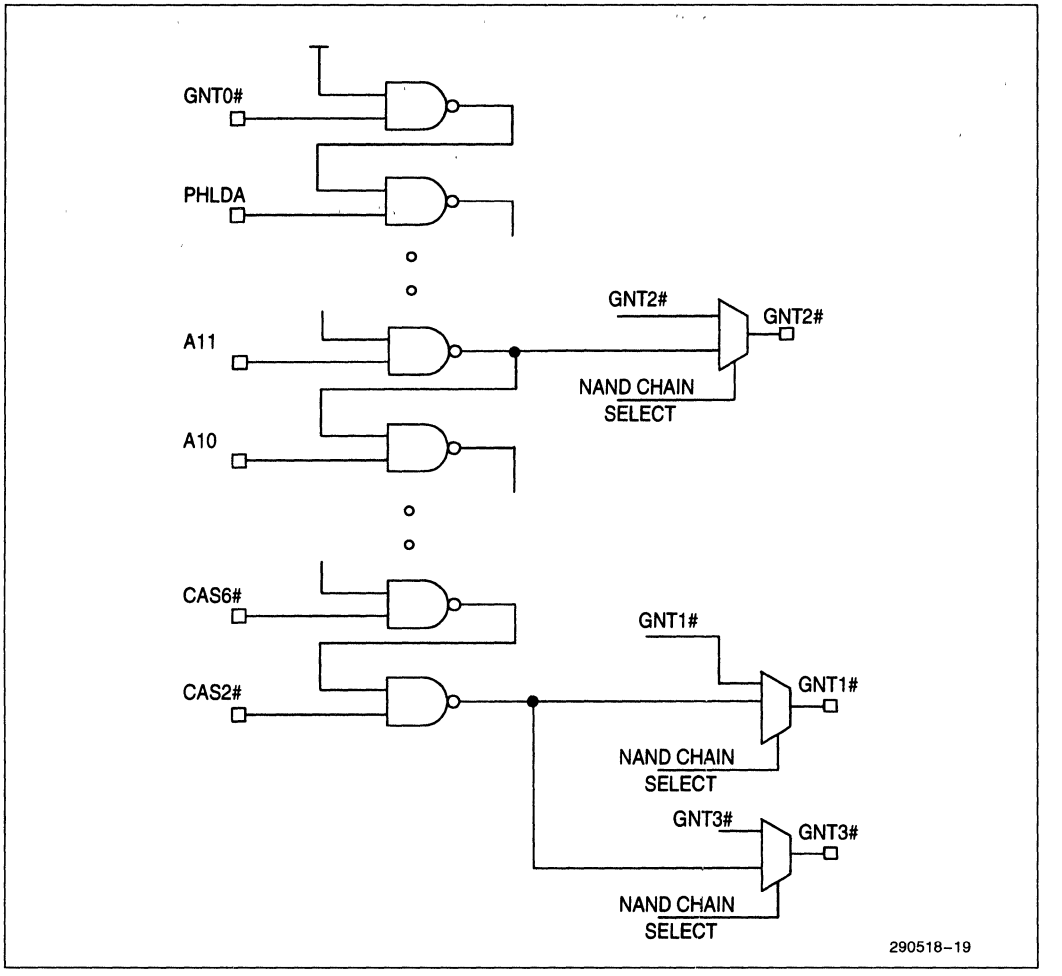


Figure 17. 82437FX NAND Tree Circuitry

NAND Tree Timing Requirements

Allow 800 ns for the input signals to propagate to the NAND tree outputs (input-to-output propagation delay specification).

7.0 82438FX TDP TESTABILITY

7.1 Test Mode Description

The test modes are decoded from HOE#, MOE#, POE#, and MSTB#. HCLK must be active for at least one clock to sample the above signals. Once these signals are sampled then HCLK must be asserted to 0 for the duration of the NAND tree test. The test modes are defined as follows.

Test Mode	HOE#	MOE#	POE#	MSTB#
NAND Tree	1	1	1	1

NAND tree mode is exited by starting HCLK with HOE#, MOE#, and POE# **not** equal to "111".

7.2 NAND Tree Mode

Tri-states all outputs and bidirectional buffers except for MD0 which is the output of the NAND tree. The NAND tree follows the pins sequentially around the chip skipping only HCLK and MD0. The first input of the NAND chain is HD5, and the NAND chain is routed counter-clockwise around the chip (e.g., HD5, HD6 . . .). The only valid output during NAND tree mode is MD0.

To perform a NAND tree test, all pins included in the NAND tree should be driven to 1, with the exception of HCLK and MDO. Beginning with HD5 and working counter-clockwise around the chip, each pin can be toggled with a resulting toggle observed on MD0. After changing an input pin to 0, keep it at 0 for the remainder of the NAND tree test.

Table 17. NAND Tree Cell Order for the 82438FX

Pin #	Pin Name	Notes
77	MOE#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
78	POE#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
81	HOE#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
83	MSTB#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
30	HCLK	HCLK must clock at least once to sample MOE#, POE#, HOE#, and MSTB# to select NAND tree mode. Then to enter NAND tree mode HCLK must remain 0. To exit NAND tree mode HCLK must be started.
41	MD0	Output of the NAND tree chain.
3	HD5	First signal in the NAND tree chain.
4	HD6	
5	HD7	
6	HD8	
7	HD9	
8	HD10	
9	HD11	
10	HD12	
11	HD13	
12	HD14	
13	HD15	
14	HD16	
15	HD17	
16	HD18	
17	HD19	
18	HD20	
20	HD21	
21	HD22	
22	HD23	
23	HD24	
25	HD25	
26	HD26	
27	HD27	
32	HD28	
33	HD29	
34	HD30	
35	HD31	
38	MD24	
39	MD8	
40	MD16	
42	MD25	
43	MD9	
44	MD17	
45	MD1	
46	MD26	
47	MD10	
50	MD18	
51	MD2	
54	MD27	
55	MD11	

Table 17. NAND Tree Cell Order for the 82438FX (Continued)

Pin #	Pin Name	Notes
56	MD19	
57	MD3	
58	MD28	
59	MD12	
60	MD20	
61	MD4	
62	MD21	
63	MD5	
66	MD29	
67	MD13	
68	MD22	
69	MD6	
70	MD30	
71	MD14	
72	MD23	
73	MD7	
74	MD31	
75	MD15	
77	MOE#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
78	POE#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
81	HOE#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
82	MADV#	
83	MSTB#	Must be 1 to enter NAND tree mode. This pin is included in the NAND tree mode input pin sequence.
84	PCMD1	
85	PCMD0	
88	PLINK7	
89	PLINK6	
90	PLINK5	
91	PLINK4	
92	PLINK3	
93	PLINK2	
94	PLINK1	
95	PLINK0	
96	HD0	
97	HD1	
98	HD2	
99	HD3	
100	HD4	Final signal in the NAND tree chain.

Figure 18 is a schematic of the NAND tree circuitry.

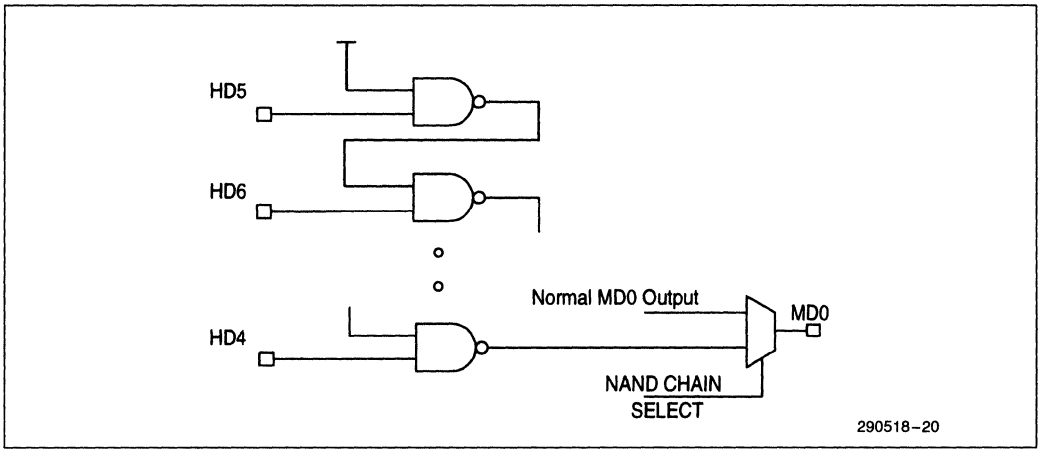


Figure 18. 82438FX NAND Tree Circuitry

NAND Tree Timing Requirements

Allow 500 ns for the input signals to propagate to the NAND tree outputs (input-to-output propagation delay specification).

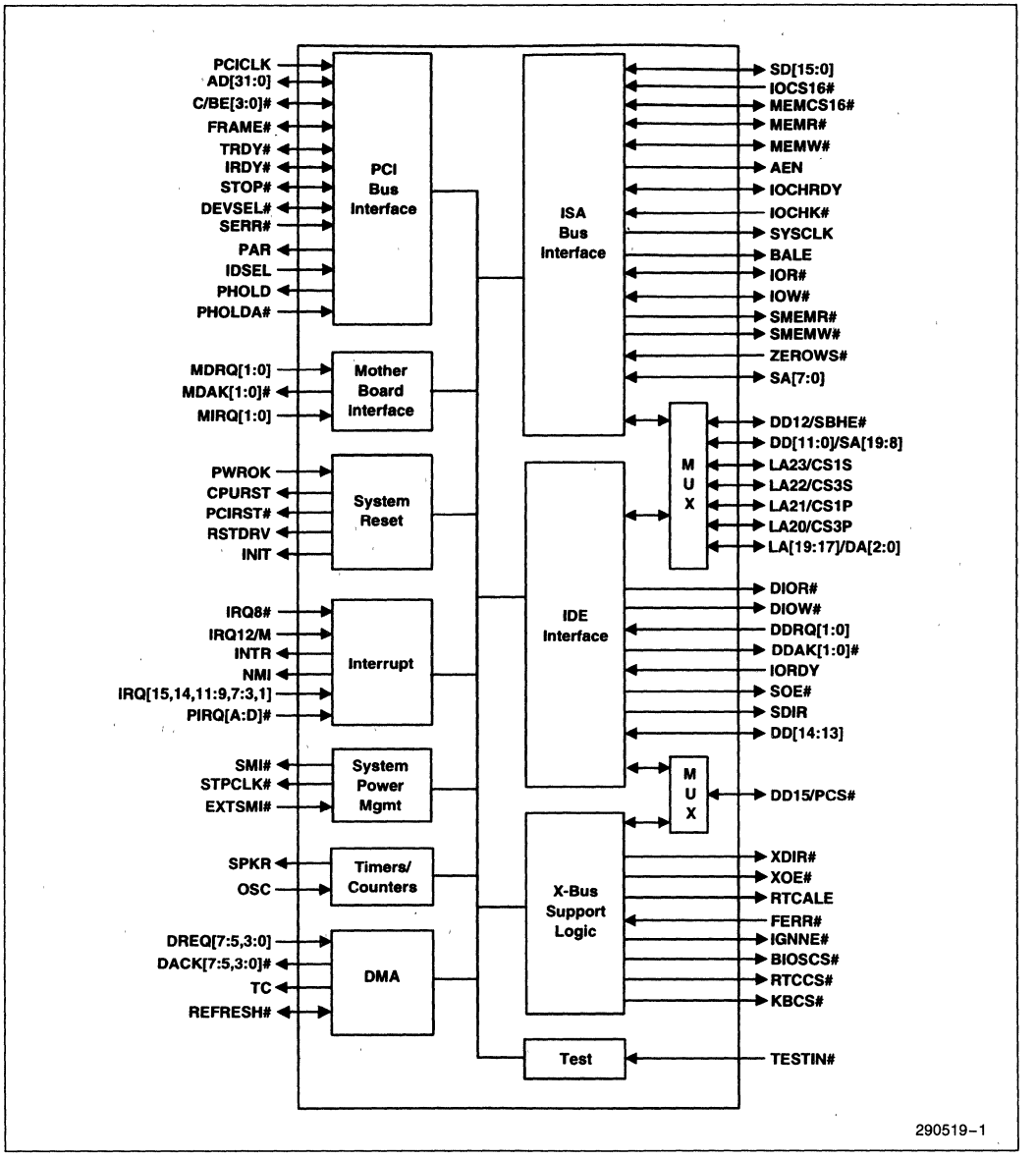
82371FB PCI ISA IDE XCELERATOR (PIIX)

- **Provides a Bridge Between the PCI Bus and ISA Bus**
- **PCI and ISA Master/Slave Interface**
  - **— PCI from 25 to 33 MHz**
  - **— ISA from 7.5 to 8.33 MHz**
  - **— Five ISA Slots**
- **Fast IDE Interface**
  - **— Supports PIO and Bus Master IDE**
  - **— Supports Up to Mode 4 Timings**
  - **— Transfer Rates to 22 Mbytes/Sec**
  - **— 8 x 32-Bit Buffer for Bus Master IDE PCI Burst Transfers**
- **Plug-and-Play Port for Motherboard Devices**
  - **— 2 Steerable DMA Channels**
  - **— Fast DMA with 4-Byte Buffer**
  - **— Up to 2 Steerable Interrupt Lines**
  - **— 1 Programmable Chip Select**
- **Steerable PCI Interrupts for PCI Device Plug-and-Play**
- **Functionality of One 82C54 Timer**
  - **— System Timer**
  - **— Refresh Request**
  - **— Speaker Tone Output**
- **Functionality of Two 82C59 Interrupt Controllers**
  - **— 14 Interrupts Supported**
  - **— Independently Programmable for Edge/Level Sensitivity**
- **Enhanced DMA Functions**
  - **— Two 8237 DMA Controllers**
  - **— Fast Type F DMA**
  - **— Compatible DMA Transfers**
  - **— Seven Independently Programmable Channels**
- **X-Bus Peripheral Support**
  - **— Chip Select Decode**
  - **— Controls Lower X-Bus Data Byte Transceiver**
- **System Power Management (Intel SMM Support)**
  - **— Programmable System Management Interrupt (SMI)Hardware Events, Software Events, EXTSMI#**
  - **— Programmable CPU Clock Control (STPCLK#)**
  - **— Fast-On/Off Mode**
- **Non-Maskable Interrupts (NMI)**
  - **— PCI System Error Reporting**
- **NAND Tree for Board-Level ATE Testing**
- **208-Pin QFP**

The 82371FB PCI ISA IDE Xcelerator (PIIX) is a multi-function PCI device implementing a PCI-to-ISA bridge. In addition, the PIIX has an IDE interface with both programmed I/O (PIO) and Bus Master functions. As a PCI-to-ISA bridge, the PIIX integrates many common I/O functions found in ISA-based PC systems—a seven channel DMA controller, two 82C59 interrupt controllers, an 8254 timer/counter, Intel SMM power management support, and control logic for NMI generation. In addition to compatible transfers, each DMA channel supports type F transfers. Chip select decoding is provided for BIOS, real time clock, and keyboard controller. Edge/level interrupts and interrupt steering are supported for PCI plug and play compatibility.

For motherboard plug-and-play compatibility, the PIIX also provides two steerable DMA channels (including type F transfers), up to two steerable interrupt lines, and a programmable chip select. The interrupt lines can be routed to any of the available ISA interrupts.

The PIIXs fast IDE interface supports two IDE connectors for up to four IDE devices providing an interface for IDE hard disks and CD ROMs.



82371FB PIIX Simplified Block Diagram

82371FB PCI ISA IDE XCELERATOR (PIIX)

CONTENTS

CONTENTS

CONTENTS

CONTENTS

1.0 SIGNAL DESCRIPTION

This section contains a detailed description of each signal. The signals are arranged in functional groups according to their interface.

The "#" symbol at the end of a signal name indicates that the active or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

Certain signal pins provide two separate functions. At the system level, these pins drive other signals with different functions through external buffers or transceivers. These pins have two different signal names depending on the function. These signal names have been noted in the signal description tables, with the signal whose function is being described in **bold** font. (For example, LA23/**CS1S** is in the section describing CS1S and **LA23**/CS1S is in the section describing LA23).

The following notations are used to describe the signal type:

I *Input* is a standard input-only signal.

O *Totem Pole Output* is a standard active driver.

I/O *Input/Output* is a bi-directional, tri-state signal.

od *Open Drain* allows multiple devices to share as a wire-OR.

t/s *Tri-state* is a bi-directional, tri-state input/output pin.

s/t/s *Sustained Tri-state* is an active low tri-state signal owned and driven by one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. An external pull-up is required to sustain the inactive state until another agent drives it and must be provided by the central resource.

2

1.1 PCI Interface Signals

Signal Name	Type	Description
PCICLK	I	**PCI CLOCK:** PCICLK provides timing for all transactions on the PCI Bus. All other PCI signals are sampled on the rising edge of PCICLK, and all timing parameters are defined with respect to this edge. PCI frequencies of 25–33 MHz are supported.
AD[31:0]	I/O	**PCI ADDRESS/DATA:** The standard PCI address and data lines. The address is driven with FRAME# assertion and data is driven or received in following clocks.
C/BE[3:0]#	I/O	**BUS COMMAND AND BYTE ENABLES:** The command is driven with FRAME# assertion. Byte enables corresponding to supplied or requested data is driven on following clocks.
FRAME#	I/O (s/t/s)	**FRAME:** Assertion indicates the address phase of a PCI transfer. Negation indicates that one or more data transfer is desired by the cycle initiator.
TRDY#	I/O (s/t/s)	**TARGET READY:** Asserted when the target is ready for a data transfer.
IRDY#	I/O (s/t/s)	**INITIATOR READY:** Asserted when the initiator is ready for a data transfer.
STOP#	I/O (s/t/s)	**STOP:** Asserted by the target to request the master to stop the current transaction.
IDSEL	I	**INITIALIZATION DEVICE SELECT:** IDSEL is used as a chip select during configuration read and write transactions.

Signal Name	Type	Description
DEVSEL#	I/O (s/t/s)	**DEVICE SELECT:** The PIIX asserts DEVSEL# to claim a PCI transaction through positive or subtractive decoding.
PAR	O	**CALCULATED PARITY SIGNAL:** PAR is "even" parity and is calculated on 36 bits—AD[31:0] plus C/BE[3:0]#.
SERR#	I	**SYSTEM ERROR:** SERR# can be pulsed active by any PCI device that detects a system error condition. Upon sampling SERR# active, the PIIX can be programmed to generate a non-maskable interrupt (NMI) to the CPU.
PHOLD#	O	**PCI HOLD:** The PIIX asserts this signal to request the PCI Bus.
PHLDA#	I	**PCI HOLD ACKNOWLEDGE:** The TSC asserts this signal to grant the PCI Bus to the PIIX.

1.2 Motherboard I/O Device Interface Signals

Signal Name	Type	Description
MDRQ[1:0]	I	**MOTHERBOARD DEVICE DMA REQUEST:** The signals can be connected internally to any of DREQ[3:0,7:5]. Each pair of request/acknowledge signals is controlled by a separate register. Each signal can be configured as steerable interrupts for motherboard devices.
MDAK[1:0]#	O	**MOTHERBOARD DEVICE DMA ACKNOWLEDGE:** These signals can be connected internally to any of DACK[3:0,7:5]. Each pair of request/acknowledge signals is controlled by a separate register. Each signal can be configured as steerable interrupts for motherboard devices.
MIRQ[1:0]	I	**MOTHERBOARD DEVICE INTERRUPT REQUEST:** The MIRQ signals can be internally connected to interrupts IRQ[15,14,12:9,7:3]. Each MIRQx line has a separate Route Control Register. If MIRQx and PIRQx are steered to the same ISA interrupt, the device connected to the MIRQx should produce active high, level-sensitive interrupts. If the Bus Master mode of the IDE interface is used and the secondary IDE channel is used, the interrupt for that channel must be connected to MIRQ0. If an MIRQ line is steered to a given IRQ input to the internal 8259, the corresponding ISA IRQ is masked, unless the Route Control Register is programmed to allow the interrupts to be shared. The should only be done if the device connected to the MIRQ line and the device connected to the ISA IRQ line both produce active high level interrupts.

1.3 IDE Interface Signals

Signal Name	Type	Description
DD[15:0]/ PCS#, SBHE#, SA[19:8]	**I/O** O I/O I/O	**DISK DATA:** These signals directly drive the corresponding signals on up to two IDE connectors (primary and secondary). In addition, these signals are buffered (using 2xALS245's on the motherboard) to produce the SA[19:8] and PCS# and SBHE# signals (see separate descriptions).
DIOR#	O	**DISK I/O READ:** This signal directly drives the corresponding signal on up to two IDE connectors (primary and secondary).
DIOW#	O	**DISK I/O WRITE:** This signal directly drives the corresponding signal on up to two IDE connectors (primary and secondary).
DDRQ[1:0]	I	**DISK DMA REQUEST:** These input signals are directly driven from the DRQ signals on the primary (DDRQ0) and secondary (DDRQ1) IDE connectors. They are used in conjunction with any ISA-Compatible DMA channel.
DDAK[1:0]#	O	**DISK DMA ACKNOWLEDGE:** These signals directly drive the DAK# signals on the primary (DDAK0#) and secondary (DDAK1#) IDE connectors. These signals are used in conjunction with the PCI Bus Master IDE function and are not associated with any ISA-Compatible DMA channel.
IORDY	I	**IO CHANNEL READY:** This input signal is directly driven by the corresponding signal on up to two IDE connectors (primary and secondary).
SOE#	O	**SYSTEM ADDRESS TRANSCEIVER OUTPUT ENABLE:** This signal controls the output enables of the 245 transceivers that interface the DD[15:0] signals to the SA[19:8], SBHE#, and PCS# signals.
SDIR	O	**SYSTEM ADDRESS TRANSCEIVER DIRECTION:** This signal controls the direction of the '245 transceivers that interface the DD[15:0] signals to the SA[19:8], SBHE#, and PCS# signals. Default condition is high (transmit). When an ISA Bus Master is granted use of the bus, the transceivers are turned around to drive the ISA address [19:8] on DD[15:3]. The address can then be latched by the PIIX. In this case, the SDIR signal is low (receive). The SOE# and SDIR signals taken together as a group can assume one of three states: **SOE# SDIR State** 0 1 PCI to ISA transaction 1 1 PCI to IDE 0 0 ISA Bus Master

2

Signals Buffered from LA[23:17]

These signals are buffered from the LA[23:17] lines by an ALS244 tri-state buffer. The output enable of this buffer is tied asserted. These signals are set up with respect to the IDE command strobes (DIOR# and IOW#) and are valid throughout I/O transactions targeting the ATA Register clock(s).

Signal Name	Type	Description
LA23/ **CS1S**	I/O	**CHIP SELECT:** CS1S is for the ATA command register block and corresponds to the inverted CS1FX# on the secondary IDE connector. CS1S is inverted externally (see PCI Local Bus IDE section).
LA22/ **CS3S**	I/O	**CHIP SELECT:** CS3S is for the ATA control register block and corresponds to the inverted CS3FX# on the secondary IDE connector. CS35 is inverted externally (see PCI Local Bus IDE section).
LA21/ **CS1P**	I/O	**CHIP SELECT:** CS1P is for the ATA command register block and corresponds to the inverted CS3FX# on the primary IDE connector. CS1P is inverted externally (see PCI Local Bus IDE section).
LA20/ **CS3P**	I/O	**CHIP SELECT:** CS3P is for the ATA control register block and corresponds to the inverted CS3FX# on the primary IDE connector. CS3P is inverted externally (see PCI Local Bus IDE section).
LA[19:17]/ **DA[2:0]**	I/O	**DISK ADDRESS:** DA[2:0] are used to indicate which byte in either the ATA command block or control block is being addressed.

1.4 ISA Interface Signals

Signal Name	Type	Description
BALE	O	**BUS ADDRESS LATCH ENABLE:** BALE is an active high signal asserted by the PIIX to indicate that the address (SA[19:0], LA[23:17]) and SBHE# signal lines are valid.
AEN	O	**ADDRESS ENABLE:** AEN is asserted during DMA cycles to prevent I/O slaves from misinterpreting DMA cycles as valid I/O cycles. This signal is also driven high during PIIX initiated refresh cycles. When TC is sampled low on the assertion of PWORK (external DMA mode), the PIIX tri-states this signal.
SYSCLK	O	**ISA SYSTEM CLOCK:** SYSCLK is the reference clock for the ISA Bus and drives the bus directly. SYSCLK is generated by dividing PCICLK by 3 or 4. The SYSCLK frequencies supported are 7.5 MHz and 8.33 MHz. SYSCLK is a divided down version of PCICLK. **Hardware Strapping Option** SYSCLK is tri-stated when PWROK is negated. The value of SYSCLK is sampled on the assertion of PWROK: If sampled high, the ISA clock divisor is 3 (for 25 MHz PCI). If sampled low, the divisor is 4 (for 30 and 33 MHz PCI).
IOCHRDY	I/O	**I/O CHANNEL READY:** Resources on the ISA Bus negate IOCHRDY to indicate that additional time (wait states) is required to complete the cycle. This signal is normally high on the ISA Bus. IOCHRDY is an input when the PIIX owns the ISA Bus and the CPU or a PCI agent is accessing an ISA slave or during DMA transfers. IOCHRDY is output when an external ISA Bus Master owns the ISA Bus and is accessing DRAM or a PIIX register.

Signal Name	Type	Description
IOCS16#	I	**16-BIT I/O CHIP SELECT:** This signal is driven by I/O devices on the ISA Bus to indicate that they support 16-bit I/O bus cycles.
IOCHK#	I	**I/O CHANNEL CHECK:** IOCHK# can be driven by any resource on the ISA Bus. When asserted, it indicates that a parity or an uncorrectable error has occurred for a device or memory on the ISA Bus. If enabled, a NMI is generated to the CPU.
IOR#	I/O	**I/O READ:** IOR# is the command to an ISA I/O slave device that the slave may drive data on to the ISA data bus (SD[15:0]).
IOW#	I/O	**I/O WRITE:** IOW# is the command to an ISA I/O slave device that the slave may latch data from the ISA data bus (SD[15:0]).
LA[23:17]/ CS1S# CS3S# CS1P# CS3P# DA[2:0]	**I/O/** O O O O O	**UNLATCHED ADDRESS:** The LA[23:17] address lines are bi-directional. These address lines allow accesses to physical memory on the ISA Bus up to 16 Mbytes. The LA[23:17] are also used to drive the IDE interface chip selects and address lines via an external ALS244 buffer. See the IDE Interface signal descriptions.
SA[7:0], **SA[19:8]/** DD[11:0]	I/O, **I/O,** I/O,	**SYSTEM ADDRESS BUS:** These bi-directional address lines define the selection with the granularity of one byte within the one Mbyte section of memory defined by the LA[23:17] address lines. The address lines SA[19:17] that are coincident with LA[19:17] are defined to have the same values as LA[19:17] for all memory cycles. For I/O accesses, only SA[15:0] are used.
SBHE#/ DD12	**I/O** I/O	**SYSTEM BYTE HIGH ENABLE:** SBHE# indicates, when asserted, that a byte is being transferred on the upper byte (SD[15:8]) of the data bus. SBHE# is negated during refresh cycles.
MEMCS16#	od	**MEMORY CHIP SELECT 16:** MEMCS16# is a decode of LA[23:17] without any qualification of the command signal lines. ISA slaves that are 16-bit memory devices drive this signal low. The PIIX drives this signal low during ISA master to DRAM Cycles.
MEMR#	I/O	**MEMORY READ:** MEMR# is the command to a memory slave that it may drive data onto the ISA data bus. This signal is also driven by the PIIX during refresh cycles.
MEMW#	I/O	**MEMORY WRITE:** MEMW# is the command to a memory slave that it may latch data from the ISA data bus.
SMEMR#	O	**STANDARD MEMORY READ:** The PIIX asserts SMEMR# to request an ISA memory slave to drive data onto the data lines. If the access is below 1 Mbyte (00000000–000FFFFFh) during DMA compatible, PIIX master, or ISA master cycles, the PIIX asserts SMEMR#. SMEMR# is a delayed version of MEMR#.
SMEMW#	O	**STANDARD MEMORY WRITE:** The PIIX asserts SMEMW# to request an ISA memory slave to accept data from the data lines. If the access is below 1 Mbyte (00000000–000FFFFFh) during DMA compatible, PIIX master, or ISA master cycles, the PIIX asserts SMEMW#. SMEMW# is a delayed version of MEMW#.

Signal Name	Type	Description
ZEROWS#	I	**ZERO WAIT-STATES:** An ISA slave asserts ZEROWS# after its address and command signals have been decoded to indicate that the current cycle can be shortened. A 16-bit ISA memory cycle can be reduced to two SYSCLKs. An 8-bit memory or I/O cycle can be reduced to three SYSCLKs. ZEROWS# has no effect during 16-bit I/O cycles.
SD[15:0]	I/O	**SYSTEM DATA:** SD[15:0] provide the 16-bit data path for devices residing on the ISA Bus. SD[15:8] correspond to the high order byte and SD[7:0] correspond to the low order byte. SD[15:0] are undefined during refresh.

1.5 DMA Signals

Signal Name	Type	Description
DREQ [7:5,3:0]	I	**DMA REQUEST:** The DREQ lines are used to request DMA service from the PIIXs DMA controller or for a 16-bit master to gain control of the ISA expansion bus. The active level (high or low) is programmed via the DMA Command Register. The request must remain active until the appropriate DACKx# signal is asserted.
DACK [7:5,3:0]#	O	**DMA ACKNOWLEDGE:** The DACK output lines indicate that a request for DMA service has been granted by the PIIX or that a 16-bit master has been granted the bus. The active level (high or low) is programmed via the DMA Command Register. These lines should be used to decode the DMA slave device with the IOR# or IOW# line to indicate selection. If used to signal acceptance of a Bus Master request, this signal indicates when it is legal to assert MASTER#. When TC is sampled low on the assertion of PWORK (external DMA mode), the PIIX tri-states these signals.
TC	O	**TERMINAL COUNT:** The PIIX asserts TC to DMA slaves as a terminal count indicator. When all the DMA channels are not in use, TC is negated (low). **Hardware Strapping Option** This strapping option selects between the internal ISA DMA mode and external DMA mode. When TC is sampled high on the assertion of PWROK (ISA DMA mode), the PIIX drives the AEN, TC, and DACK#[7:5,3:0] normally. When TC is sampled low on the assertion of PWROK (external DMA mode), the PIIX tri-states the AEN, TC, and DACK[7:5,3:0]# signals, and also forwards PCI masters I/O accesses to location 0000h to ISA. TC has an internal pull-up resistor. For normal operation, this pin is pulled high by the internal pull-up.
REFRESH#	I/O	**REFRESH:** As an output, REFRESH# indicates when a refresh cycle is in progress. It should be used to enable the SA[15:0] address to the row address inputs of all banks of dynamic memory on the ISA Bus. Thus, when MEMR# is asserted, the entire expansion bus dynamic memory is refreshed. Memory slaves must not drive any data onto the bus during refresh. As an output, this signal is driven directly onto the ISA Bus. This signal is an output only when the PIIX DMA refresh controller is a master on the bus responding to an internally generated request for refresh. As an input, REFRESH# is driven by 16-bit ISA Bus Masters to initiate refresh cycles.

1.6 Timer/Counter Signals

Signal Name	Type	Description
SPKR	O	**SPEAKER DRIVE:** The SPKR signal is the output of counter 2.
OSC	I	**OSCILLATOR:** OSC is the 14.31818 MHz ISA clock signal. It is used by the internal 8254 Timer.

1.7 Interrupt Controller Signals

Signal Name	Type	Description
IRQ[15,14, 11:9,7:3,1]	I	**INTERRUPT REQUEST:** The IRQ signals provide both system board components and ISA Bus I/O devices with a mechanism for asynchronously interrupting the CPU. The assertion mode of these inputs depends on the programming of the two ELCR Registers. The IRQ14 signal (pin 83) must be used by the Bus Master IDE interface function to signal interrupts on the primary IDE channel.
IRQ8#	I	**INTERRUPT REQUEST EIGHT SIGNAL:** IRQ8# is always an active low edge triggered interrupt input (i.e., this interrupt can not be modified by software). Upon PCIRST#, IRQ8# is placed in active low edge sensitive mode.
IRQ12/M	I	**INTERRUPT REQUEST/MOUSE INTERRUPT:** In addition to providing the standard interrupt function (see IRQ[15,14,11:9,7:3,1] signal description), this pin can be programmed (via X-Bus Chip Select Register) to provide a mouse interrupt function.
PIRQ[3:0]#	I	**PROGRAMMABLE INTERRUPT REQUEST:** The PIRQx# signals can be shared with interrupts IRQ[15,14,12:9,7:3] as described in the Interrupt Steering section. Each PIRQx# line has a separate Route Control Register. These signals require external pull-up resisters.
INTR	od	**CPU INTERRUPT:** INTR is driven by the PIIX to signal the CPU that an interrupt request is pending and needs to be serviced. The interrupt controller must be programmed following PCIRST# to ensure that INTR is at a known state.
NMI	od	**NON-MASKABLE INTERRUPT:** NMI is used to force a non-maskable interrupt to the CPU. The PIIX generates an NMI when either SERR# or IOCHK# is asserted, depending on how the NMI Status and Control Register is programmed.

1.8 System Power Management (SMM) Signals

Signal Name	Type	Description
SMI#	od	**SYSTEM MANAGEMENT INTERRUPT:** SMI# is an active low synchronous output that is asserted by the PIIX in response to one of many enabled hardware or software events.
STPCLK#	od	**STOP CLOCK:** STPCLK# is an active low synchronous output that is asserted by the PIIX in response to one of many hardware or software events. STPCLK# connects directly to the CPU and is synchronous to PCICLK.
EXTSMI#	I	**EXTERNAL SYSTEM MANAGEMENT INTERRUPT:** EXTSMI# is a falling edge triggered input to the PIIX indicating that an external device is requesting the system to enter SMM mode. An external pullup should be placed on this signal if it is not used or it is not guaranteed to be always driven.

1.9 X-Bus Signals

Signal Name	Type	Description
XDIR#	O	**X-BUS DIRECTION:** XDIR# is tied directly to the direction control of a 74F245 that buffers the X-Bus data (XD[7:0]). XDIR# is asserted for all I/O read cycles, regardless if the accesses are to a PIIX supported device. XDIR# is only asserted for memory cycles if BIOS space has been decoded. For PCI Master and ISA master-initiated read cycles, XDIR# is asserted from the falling edge of either IOR# or MEMR# (from MEMR# only if BIOS space has been decoded), depending on the cycle type. When the rising edge of IOR# or MEMR# occurs, the PIIX negates XDIR#. For DMA read cycles from the X-Bus, XDIR# is asserted from DACKx# falling and negated from DACKx# rising. At all other times, XDIR# is negated.
XOE#	O	**X-BUS OUTPUT ENABLE:** XOE# is tied directly to the output enable of a 74F245 that buffers the X-Bus data (XD[7:0]) from the system data bus (SD[7:0]). XOE# is asserted when a PIIX supported X-Bus device is decoded, and the devices decode is enabled in the X-Bus Chip Select Enable Register (XBCS Register). XOE# is asserted from the falling edge of the ISA commands (IOR#, IOW#, MEMR#, or MEMW#) for PCI Master and ISA master-initiated cycles. XOE# is negated from the rising edge of the ISA command signals for CPU and PCI Master-initiated cycles and the SA[16:0] and LA[23:17] address for ISA master-initiated cycles. XOE# is not generated during any access to an X-Bus peripheral in which it's decode space has been disabled.
DD15/ **PCS#**	O	**PROGRAMMABLE CHIP SELECT:** PCS# is asserted for ISA I/O cycles that are generated by PCI masters and subtractively decoded to ISA, if the access hits the address range programmed into the PCSC Register. The X-Bus buffer signals are enabled when the chip select is asserted (i.e., it is assumed that the peripheral that is selected via this pin resides on the X-Bus).
BIOSCS#	O	**BIOS CHIP SELECT:** BIOSCS# is asserted during read or write accesses to BIOS. BIOSCS# is driven combinatorially from the ISA addresses SA[16:0] and LA [23:17], except during DMA. During DMA cycles, BIOSCS# is not generated.
KBCS#	O	**KEYBOARD CONTROLLER CHIP SELECT:** KBCS# is asserted during I/O read or write accesses to KBC locations 60h and 64h. This signal is driven combinatorially from the ISA addresses SA[16:0] and LA [23:17]. For DMA cycles, KBCS# is never asserted.
RTCCS#	O	**REAL TIME CLOCK CHIP SELECT:** RTCCS# is asserted during read or write accesses to RTC location 71h. RTCCS# can be tied to a pair of external OR gates to generate the real time clock read and write command signals.
RTCALE	O	**REAL TIME CLOCK ADDRESS LATCH:** RTCALE is used to latch the appropriate memory address into the RTC. A write to port 70h with the appropriate RTC memory address that will be written to or read from, causes RTCALE to be asserted. RTCALE is asserted based on IOW# falling and remains asserted for two SYSCLKs.
FERR#	od	**NUMERIC COPROCESSOR ERROR:** This signal is tied to the coprocessor error signal on the CPU. IGNNE# is only used if the PIIX coprocessor error reporting function is enabled in the XBCSA Register. If FERR# is asserted, the PIIX generates an internal IRQ13 to it's interrupt controller unit. The PIIX then asserts the INTR output to the CPU. FERR# is also used to gate the IGNNE# signal to ensure that IGNNE# is not asserted to the CPU unless FERR# is active. FERR# has a weak internal pull-up used to ensure a high level when the coprocessor error function is disabled.

Signal Name	Type	Description
IGNNE#	od	**IGNORE ERROR:** This signal is connected to the ignore error pin on the CPU. IGNNE# is only used if the PIIX coprocessor error reporting function is enabled in the XBCSA Register. If FERR# is asserted, indicating a coprocessor error, a write to the Coprocessor Error Register (F0h) causes the IGNNE# to be asserted. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted when the Coprocessor Error Register is written, the IGNNE# signal is not asserted.

1.10 System Reset Signals

Signal Name	Type	Description
PWROK	I	**POWER OK:** When asserted, PWROK is an indication to the PIIX that power and PCICLK have been stable for at least 1 ms. PWROK can be driven asynchronously. When PWROK transitions from low to high, the PIIX asserts CPURST, PCIRST# and RSTDRV.
CPURST	od	**CPU RESET:** The PIIX asserts CPURST to reset the CPU. The PIIX asserts CPURST during power-up and when a hard reset sequence is initiated through the RC Register. CPURST is driven synchronously to the rising edge of PCICLK. If a hard reset is initiated through the RC Register, the PIIX resets it's internal registers to the default state.
PCIRST#	O	**PCI RESET:** The PIIX asserts PCIRST# to reset devices that reside on the PCI Bus. The PIIX asserts PCIRST# during power-up and when a hard reset sequence is initiated through the RC Register. PCIRST# is driven inactive a minimum of 1 ms after PWROK is driven active. PCIRST# is driven active for a minimum of 1 ms when initiated through the RC Register. PCIRST# is driven asynchronously relative to PCICLK.
INIT	O	**INITIALIZATION:** The PIIX asserts INIT if it detects a shut down special cycle on the PCI Bus or if a soft reset is initiated via the RC Register.
RSTDRV	O	**RESET DRIVE:** The PIIX asserts this signal during a hard reset and during power-up to reset ISA Bus devices. RSTDRV is also asserted for a minimum of 1 ms if a hard reset has been programmed in the RC Register.

1.11 Test Signals

Signal Name	Type	Description
TESTIN#	I	**TEST INPUT:** The Test signal is used to tri-state all of the PIIX outputs. This input contains an internal pull-up resistor.

1.12 Signal State During Reset

Table 1 shows the state of all PIIX output and bi-directional signals during a hard reset. A hard reset is initiated when PWROK is asserted or by programming a hard reset through the RC Register.

Table 1. Output and I/O Signal States During Hard Reset

Signal	State
AD[31:0]	Low
C/BE[3:0]#	Low
FRAME#	Tri-state
TRDY#	Tri-state
IRDY#	Tri-state
STOP#	Tri-state
DEVSEL#	Tri-state
PAR	Input
PHOLD#	High
MDAK[1:0]#	High
DD[15:0]/ PCS#,SBHE#, SA[19:8]	Tri-state
SA[7:0]	Undefined
DIOR#	High
DIOW#	High
DDAK[1:0]#	High*
SOE#	High
SDIR	High
LA23/CS1S	Undefined

Signal	State
LA22/CS3S	Undefined
LA21/CS1P	Undefined
LA20/CS3P	Undefined
LA[19:17]/ DA[2:0]	Undefined
BALE	Low
AEN	Depends on strapping option
SYSCLK	Strapping option
IOCHRDY	Tri-state
IOR#	High
IOW#	High
MEMCS16#	Open drain
MEMR#	Tri-state
MEMW#	Tri-state
SMEMR#	High
SMEMW#	High
SD[15:0]	Tri-state
DACK[7:5, 3:0]#	Depends on strapping option

Signal	State
TC	High*
REFRESH#	Tri-state
SPKR	Low
INTR	Open drain
NMI	Open drain
SMI#	Open drain
STPCLK#	Open drain
XDIR#	High
XOE#	High
BIOSCS#	High
KBCS#	High
RTCCS#	High
RTCALE	Low
FERR#	Open drain
IGNNE#	Open drain
CPURST	Open drain
PCIRST#	Low
INIT	Open drain
RSTDRV	High

* DDAK[0] and TC are pulled high with an internal pull-up.

2.0 REGISTER DESCRIPTION

The PIIX internal registers are organized into five groups—PCI Configuration Registers (function 0), PCI Configuration Registers (function 1), ISA-Compatible Registers, PCI Bus Master IDE Registers, and Power Management Registers. These registers are discussed in this section.

Some of the PIIX registers contain reserved bits. Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back.

During a hard reset, the PIIX sets its internal registers to predetermined **default** states. The default values are indicated in the individual register descriptions.

The following notation is used to describe register access attributes:

RO Read Only. If a register is read only, writes have no effect.

WO Write Only. If a register is write only, reads have no effect.

R/W Read/Write. A register with this attribute can be read and written. Note that individual bits in some read/write registers may be read only.

R/WC Read/Write Clear. A register bit with this attribute can be read and written. However, a write of 1 clears (sets to 0) the corresponding bit and a write of 0 has no effect.

2.1 Register Access

Table 2, Table 3, and Table 4, show the I/O assignments for the PCI Configuration Registers (function 0), ISA Compatible Registers, and PCI Configuration Registers (function 1). The CPU and PCI masters have access to all PIIX internal registers. In addition, ISA masters have access to some of the ISA-Compatible Registers (see Table 4). Table 5 show the I/O assignments for the Bus Master IDE Interface Registers.

PCI Configuration Registers (Functions 0 and 1)

The PIIX is a multi-function device on the PCI Bus implementing two functions—PCI-to-ISA Bridge (function 0) and IDE Interface (function 1). These functions can be independently configured with two sets of PCI Configuration Registers in compliance with the PCI Local Bus Specification, Revision 2.0. The two sets of configuration registers are accessed by the CPU through a mechanism defined for multi-functional PCI devices. The PIIX does not assert DEVSEL# for PCI Configuration cycles that target functions 2 through 7.

ISA Compatible Registers

The ISA-Compatible Registers (e.g., DMA Registers, timer/counter registers, X-Bus Registers, and NMI Registers) are accessed through normal I/O space. Except for the DMA Registers, the PIIX positively decodes accesses to the ISA-Compatible Registers. The PIIX subtractively decodes accesses to all I/O space registers contained within the ISA-Compatible DMA function. This permits another device in the system to implement the compatible DMA function.

PCI master accesses to the ISA-Compatible Registers can be 8, 16, 24, or 32 bits. However, the PIIX only responds to the least significant byte. On writes the other bytes are not loaded and on reads the other bytes have invalid data. The PIIX responds as an 8-bit ISA I/O slave when accessed by an ISA master. See the PCI Local Bus IDE section for accesses to the IDE Register blocks located in the IDE device.

In general, accesses from CPU or PCI masters to the internal PIIX registers are not broadcast to the ISA Bus. Exceptions to this are read/write accesses to 70h and F0h and write accesses to 80h, 84–86h, 88h, 8C–8Eh, 90h, 94–96h, 98h, and 9C–9Eh. These accesses are broadcast to the ISA Bus. Note that aliasing of the 90–9Fh to 80–8Fh can be enabled/disabled via the ISA Controller Recovery Timer Register.

Power Management Registers

There are two power management registers located in normal I/O space. These registers are accessed (by PCI Bus Masters) with 8-bit accesses. The other power management registers are located in PCI configuration space for function 0.

PCI Bus Master IDE Registers

The PCI Bus Master IDE function uses 16 bytes located in normal I/O space, allocated via the BMIBA Register (a PCI base address register). All Bus Master IDE I/O space registers can be accessed as 8, 16, or 32 bit quantities.

Table 2. PCI Configuration Registers—Function 0 (PCI to ISA Bridge)

Configuration Offset	Mnemonic	Register	Register Access
00–01h	VID	Vendor Identification	RO
02–03h	DID	Device Identification	RO
04–05h	PCICMD	PCI Command	R/W
06–07h	PCISTS	PCI Device Status	R/WC
08h	RID	Revision Identification	RO
09h	PI	Programming Interface	RO
0Ah	SUBC	Sub Class Code	RO
0Bh	BCC	Base Class Code	RO
0C–0Dh		Reserved	
0Eh	HEDT	Header Type	RO
0F–4Bh		Reserved	
4Ch	IORT	ISA I/O Controller Recovery Timer	R/W
4Dh		Reserved	
4Eh	XBCS	X-Bus Chip Select Enable	R/W
4F–5Fh		Reserved	
60–63h	PIRQRC[A:D]	PCI IRQ Route Control	R/W
64–68h		Reserved	
69h	TOM	Top of Memory	R/W
6A–6Bh	MSTAT	Miscellaneous Status	R/W
6C–6Fh		Reserved	
70–71h	MBIRQ[1:0]	Motherboard IRQ Route Control	R/W
72–75h		Reserved	
76–77h	MBDMA[1:0]	Motherboard Device DMA Control	R/W
78–79h	PCSC	Programmable Chip Select Control	R/W
7A–9Fh		Reserved	
A0h	SMICNTL	SMI Control	R/W
A1h		Reserved	
A2–A3h	SMIEN	SMI Enable	R/W
A4–A7h	SEE	System Event Enable	R/W

Table 2. PCI Configuration Registers—Function 0 (PCI to ISA Bridge) (Continued)

Configuration Offset	Mnemonic	Register	Register Access
A8h	FTMR	Fast-Off Timer	R/W
A9h		Reserved	
AA–ABh	SMIREQ	SMI Request	R/W
ACh	CTLTMR	Clock Scale STPCLK# Low Timer	R/W
ADh		Reserved	
AEh	CTHTMR	Clock Scale STPCLK# High Timer	R/W
AF–FFh		Reserved	

Table 3. PCI Configuration Registers—Function 1 (IDE Interface)

Configuration Offset	Mnemonic	Register	Register Access
00–01h	VID	Vendor Identification	RO
02–03h	DID	Device Identification	RO
04–05h	PCICMD	Command	R/W
06–07h	PCISTS	PCI Device Status	R/WC
08h	RID	Revision Identification	RO
09h	PI	Programming Interface	RO
0Ah	SUBC	Sub Class Code	RO
0Bh	BCC	Base Class Code	RO
0Ch		Reserved	
0Dh	MLT	Master Latency Timer	R/W
0Eh	HEDT	Header Type	RO
0F–1Fh		Reserved	
20–23h	BMIBA	Bus Master Interface Base Address	R/W
24–3Fh		Reserved	
40–43h	IDETIM	IDE Timing Modes	R/W
44–FFh		Reserved	

Table 4. ISA-Compatible and Power Management Registers

Address (hex)	Address (bits)				Type	Name	Access
	FEDC	BA98	7654	3210			
0000h[3]	0000	0000	000x	0000	r/w	DMA1 CH0 Base and Current Address	PCI
0001h[3]	0000	0000	000x	0001	r/w	DMA1 CH0 Base and Current Count	PCI
0002h[3]	0000	0000	000x	0010	r/w	DMA1 CH1 Base and Current Address	PCI
0003h[3]	0000	0000	000x	0011	r/w	DMA1 CH1 Base and Current Count	PCI
0004h[3]	0000	0000	000x	0100	r/w	DMA1 CH2 Base and Current Address	PCI
0005h[3]	0000	0000	000x	0101	r/w	DMA1 CH2 Base and Current Count	PCI
0006h[3]	0000	0000	000x	0110	r/w	DMA1 CH3 Base and Current Address	PCI
0007h[3]	0000	0000	000x	0111	r/w	DMA1 CH3 Base and Current Count	PCI
0008h[3]	0000	0000	000x	1000	r/w	DMA1 Status(r) Command(w) Register	PCI
0009h[3]	0000	0000	000x	1001	wo	DMA1 Write Request	PCI
000Ah[3]	0000	0000	000x	1010	wo	DMA1 Write Single Mask Bit	PCI
000Bh[3]	0000	0000	000x	1011	wo	DMA1 Write Mode	PCI
000Ch[3]	0000	0000	000x	1100	wo	DMA1 Clear Byte Pointer	PCI
000Dh[3]	0000	0000	000x	1101	wo	DMA1 Master Clear	PCI
000Eh[3]	0000	0000	000x	1110	wo	DMA1 Clear Mask	PCI
000Fh[3]	0000	0000	000x	1111	r/w	DMA1 Read/Write All Mask Register Bits	PCI
0020h	0000	0000	001x	xx00	r/w	INT 1 Control	PCI/ISA
0021h	0000	0000	001x	xx01	r/w	INT 1 Mask	PCI/ISA
0040h	0000	0000	010x	0000	r/w	Timer Counter 1 - Counter 0 Count	PCI/ISA
0041h	0000	0000	010x	0001	r/w	Timer Counter 1 - Counter 1 Count	PCI/ISA
0042h	0000	0000	010x	0010	r/w	Timer Counter 1 - Counter 2 Count	PCI/ISA
0043h	0000	0000	010x	0011	wo	Timer Counter 1 Command Mode	PCI/ISA
0060h[1]	0000	0000	0110	0000	r	Reset XBus IRQ12/M and IRQ1	PCI/ISA
0061h	0000	0000	0110	0001	r/w	NMI Status and Control	PCI/ISA
0070h[1]	0000	0000	0111	0xx0	wo	CMOS RAM Address and NMI Mask Reg	PCI/ISA
0080h[2,3]	0000	0000	100x	0000	r/w	DMA Page (Reserved)	PCI/ISA
0081h[3]	0000	0000	100x	0001	r/w	DMA Channel 2 Page	PCI/ISA
0082h[3]	0000	0000	1000	0010	r/w	DMA Channel 3 Page	PCI/ISA
0083h[3]	0000	0000	100x	0011	r/w	DMA Channel 1 Page	PCI/ISA
0084h[2,3]	0000	0000	100x	0100	r/w	DMA Page (Reserved)	PCI/ISA
0085h[2,3]	0000	0000	100x	0101	r/w	DMA Page (Reserved)	PCI/ISA

Table 4. ISA-Compatible and Power Management Registers (Continued)

Address (hex)	Address (bits)				Type	Name	Access
	FEDC	BA98	7654	3210			
0086h[2,3]	0000	0000	100x	0110	r/w	DMA Page (Reserved)	PCI/ISA
0087h[3]	0000	0000	100x	0111	r/w	DMA Channel 0 Page	PCI/ISA
0088h[2,3]	0000	0000	100x	0100	r/w	DMA Page (Reserved)	PCI/ISA
0089h[3]	0000	0000	100x	1001	r/w	DMA Channel 6 Page	PCI/ISA
008Ah[3]	0000	0000	100x	1010	r/w	DMA Channel 7 Page	PCI/ISA
008Bh[3]	0000	0000	100x	1011	r/w	DMA Channel 5 Page	PCI/ISA
008Ch[2,3]	0000	0000	100x	1100	r/w	DMA Page (Reserved)	PCI/ISA
008Dh[2,3]	0000	0000	100x	1101	r/w	DMA Page (Reserved)	PCI/ISA
008Eh[2,3]	0000	0000	100x	1110	r/w	DMA Page (Reserved)	PCI/ISA
008Fh[3]	0000	0000	100x	1111	r/w	DMA Low Page Register Refresh	PCI/ISA
00A0h	0000	0000	101x	xx00	r/w	INT 2 Control	PCI/ISA
00A1h	0000	0000	101x	xx01	r/w	INT 2 Mask	PCI/ISA
00B2h	0000	0000	1011	0010	r/w	Advanced Power Management Control	PCI
00B3h	0000	0000	1011	0011	r/w	Advanced Power Management Status	PCI
00C0h[3]	0000	0000	1100	000x	r/w	DMA2 CH0 Base and Current Address	PCI
00C2h[3]	0000	0000	1100	001x	r/w	DMA2 CH0 Base and Current Count	PCI
00C4h[3]	0000	0000	1100	010x	r/w	DMA2 CH1 Base and Current Address	PCI
00C6h[3]	0000	0000	1100	011x	r/w	DMA2 CH1 Base and Current Count	PCI
00C8h[3]	0000	0000	1100	100x	r/w	DMA2 CH2 Base and Current Address	PCI
00CAh[3]	0000	0000	1100	101x	r/w	DMA2 CH2 Base and Current Count	PCI
00CCh[3]	0000	0000	1100	110x	r/w	DMA2 CH3 Base and Current Address	PCI
00CEh[3]	0000	0000	1100	111x	r/w	DMA2 CH3 Base and Current Count	PCI
00D0h[3]	0000	0000	1101	000x	r/w	DMA2 Status(r) Command(w)	PCI
00D2h[3]	0000	0000	1101	001x	wo	DMA2 Write Request	PCI
00D4h[3]	0000	0000	1101	010x	wo	DMA2 Write Single Mask Bit	PCI
00D6h[3]	0000	0000	1101	011x	wo	DMA2 Write Mode	PCI
00D8h[3]	0000	0000	1101	100x	wo	DMA2 Clear Byte Pointer	PCI
00DAh[3]	0000	0000	1101	101x	wo	DMA2 Master Clear	PCI
00DCh[3]	0000	0000	1101	110x	wo	DMA2 Clear Mask	PCI
00DEh[3]	0000	0000	1101	111x	r/w	DMA2 Read/Write All Mask Register Bits	PCI
00F0h[1]	0000	0000	1111	0000	wo	Coprocessor Error	PCI/ISA
04D0h	0000	0100	1101	0000	r/w	INT-1 Edge/Level Control	PCI/ISA

2

Table 4. ISA-Compatible and Power Management Registers (Continued)

Address (hex)	Address (bits)				Type	Name	Access
	FEDC	BA98	7654	3210			
04D1h	0000	0100	1101	0001	r/w	INT-2 Edge/Level Control	PCI/ISA
0CF9h	0000	1100	1111	1001	r/w	Reset Control	PCI

NOTES:
1. Read and write accesses to these locations are always broadcast to the ISA Bus.
2. Write accesses to these locations are broadcast to the ISA Bus. Read Accesses are not. If programmed in the ISA Controller Recovery Timer Register, the PIIX will not alias the 90h–9Fh address range with the following addresses; 80h, 84h–86h, 88h, and 8C–8Eh. In this case, accesses to the 90h–9Fh address range for the previously specified addresses are forwarded to the ISA Bus for both reads and writes and are ignored during ISA Master cycles (i.e., they are no-longer considered PIIX registers). Note that port 92 is always a distinct ISA Register and is always forwarded to the ISA Bus.
3. ISA-Compatible DMA Register I/O space accesses are always subtractively decoded.

Table 5. PCI Bus Master IDE I/O Registers

Offset From Base Address	Mnemonic	Register	Register Access
00h	BMICP	Bus Master IDE Command (Primary)	R/W
01h		Reserved	
02h	BMISP	Bus Master IDE Status (Primary)	R/WC
03h		Reserved	
04–07h	BMIDTPP	Bus Master IDE Descriptor Table Pointer (Primary)	R/W
08h	BMICS	Bus Master IDE Command (Secondary)	R/W
09h		Reserved	
0Ah	BMISS	Bus Master IDE Status (Secondary)	R/WC
0Bh		Reserved	
0C–0Fh	BMIDTPS	Bus Master IDE Descriptor Table Pointer (Secondary)	R/W

NOTE:
The base address is programmable via the BMIBA Register (20-23h; function 1).

2.2 PCI Configuration Registers—Function 0 (PCI to ISA Bridge)

2.2.1 VID—VENDOR IDENTIFICATION REGISTER (FUNCTION 0)

Address Offset: 00–01h
Default Value: 8086h
Attribute: Read Only

The VID Register contains the vendor identification number. This register, along with the Device Identification Register, uniquely identifies any PCI device. Writes to this register have no effect.

Bit	Description
15:0	**Vendor Identification Number:** This is a 16-bit value assigned to Intel.

2.2.2 DID—DEVICE IDENTIFICATION REGISTER (FUNCTION 0)

Address Offset: 02–03h
Default Value: 122Eh
Attribute: Read Only

The DID Register contains the device identification number. This register, along with the VID Register, define the PIIX. Writes to this register have no effect.

Bit	Description
15:0	**Device Identification Number:** This is a 16-bit value assigned to the PIIX.

2.2.3 PCICMD—COMMAND REGISTER (FUNCTION 0)

Address Offset: 04–05h
Default Value: 0007h
Attribute: Read/Write

Bit	Description
15:10	**Reserved:** Read as 0.
9	**Fast Back-to-Back Enable:** (Not Implemented). This bit is hardwired to 0.
8:5	**Reserved:** Read as 0.
4	**Postable Memory Write Enable:** (Not Implemented). This bit is hardwired to 0.
3	**Special Cycle Enable (SCE):** 1 = Enable (the PIIX recognizes shutdown special cycle). 0 = Disable (the PIIX ignores all PCI special cycles).
2	**Bus Master Enable (BME):** (Not Implemented). The PIIX does not support disabling it's Bus Master capability. This bit is hardwired to 1.
1	**Memory Access Enable (MAE):** (Not Implemented). The PIIX does not support disabling access to main memory. This bit is hardwired to 1.
0	**I/O Space Access Enable (IOSE):** The PIIX does not support disabling it's response to PCI I/O cycles. This bit is hardwired to 1.

2.2.4 PCISTS—PCI DEVICE STATUS REGISTER (FUNCTION 0)

Address Offset: 06–07h
Default Value: 0200h
Attribute: Read/Write, Read Only

The PCISTS Register reports the occurrence of a PCI master-abort by the PIIX or a PCI target-abort when the PIIX is a master. The register also indicates the PIIX DEVSEL# signal timing.

Bit	Description
15	**Detected Parity Error (PERR):** (Not Implemented). Read as 0.
14	**Signaled SERR# Status (SERRS):** (Not Implemented). Read as 0.
13	**Master-Abort Status (MA)—R/W:** When the PIIX, as a master (for the ISA bridge function), generates a master-abort, MA is set to 1. Software sets MA to 0 by writing 1 to this bit location.
12	**Received Target-Abort Status (RTA)—R/W:** When the PIIX is a master on the PCI Bus (for the ISA bridge function) and receives a target-abort, this bit is set to 1. Software sets RTA to 0 by writing 1 to this bit location.
11	**Signaled Target-Abort Status (STA)—R/W:** This bit is set when the PIIX ISA bridge function is targeted with a transaction that the PIIX terminates with a target abort. Software sets STA to 0 by writing 1 to this bit location.
10:9	**DEVSEL# Timing Status (DEVT)—RO:** The PIIX always generates DEVSEL# with medium timing for ISA functions. Thus, DEVT = 01. This DEVSEL# timing does not include configuration cycles.
8	**PERR# Response:** (Not Implemented). Read as 0.
7	**Fast Back to Back—RO:** This bit indicates to the PCI Master that PIIX as a target is capable of accepting fast back-to-back transactions.
6:0	**Reserved:** Read as 0s.

2.2.5 RID—REVISION IDENTIFICATION REGISTER (FUNCTION 0)

Address Offset: 08h
Default Value: Refer to stepping information
Attribute: Read Only

This 8 bit register contains device stepping information. Writes to this register have no effect.

Bit	Description
7:0	**Revision ID Byte:** This register is hardwired to the default value.

2.2.6 PI—PROGRAMMING INTERFACE REGISTER (FUNCTION 0)

Address Offset: 09h
Default Value: 00h
Attribute: Read Only

This register contains the device programming interface information related to the Sub Class Code Register and Base Class Code Register definition for this function.

Bit	Description
7:0	**Programming Interface:** 00h-hardwired as a PCI-to-ISA bridge.

2.2.7 SUBC—SUB CLASS CODE REGISTER (FUNCTION 0)

Address Offset: 0Ah
Default Value: 01h
Attribute: Read Only

This register indicates the function sub class in relation to the Base Class Code Register.

Bit	Description
7:0	**Sub-Class Code (SCC):** 01h = PCI-to-ISA bridge.

2.2.8 BCC—BASE CLASS CODE REGISTER (FUNCTION 0)

Address Offset: 0Bh
Default Value: 06h
Attribute: Read Only

This register contains the Base Class Code of the PIIX.

Bit	Description
7:0	**Base Class Code (BASEC):** 06h = bridge device.

2.2.9 HEDT—HEADER TYPE REGISTER (FUNCTION 0)

Address Offset: 0Eh
Default Value: 80h
Attribute: Read Only

The HEDT Register identifies the PIIX as a multi-function device.

Bit	Description
7:0	**Device Type (DEVCET):** 80h = multi-function device.

2.2.10 IORT—ISA I/O RECOVERY TIMER REGISTER (FUNCTION 0)

Address Offset: 4Ch
Default Value: 4Dh
Attribute: Read/Write

The I/O recovery mechanism in the PIIX is used to add additional recovery delay between CPU or PCI master originated 8-bit and 16-bit I/O cycles to the ISA Bus. The PIIX automatically forces a minimum delay of 3.5 SYSCLKs between back-to-back 8- and 16-bit I/O cycles to the ISA Bus. This delay is measured from the rising edge of the I/O command (IOR# or IOW#) to the falling edge of the next I/O command. If a delay of greater than 3.5 SYSCLKs is required, the ISA I/O Recovery Time Register can be programmed to increase the delay in increments of SYSCLKs. No additional delay is inserted for back-to-back I/O "sub cycles" generated as a result of byte assembly or disassembly. This register defaults to 8 and 16-bit recovery enabled with one SYSCLK clock added to the standard I/O recovery.

Bit	Description
7	**DMA Reserved Page Register Aliasing Control (DMAAC):** When DMAAC = 0, the PIIX aliases I/O accesses in the 90–9Fh range to the 80–8Fh range. In this case, the PIIX only forwards write accesses to these locations to the ISA Bus. When DMAAC = 1, the PIIX disables aliasing for the following registers; 80h, 84-86h, 88h, and 8C-8Eh. When disabled, the PIIX forwards read and write accesses to those registers to the ISA Bus. Note that port 92h is always a distinct ISA Register in the 90–9Fh range and is always forwarded to the ISA Bus. When DMAAC = 1, ISA master accesses to the 90–9Fh range are ignored by the PIIX. Also, when DMAAC = 1, the PIIX does not re-load the power management Fast-Off-Timer with its original value for accesses to the 90–9Fh address range.
6	**8-Bit I/O Recovery Enable:** 1 = Enable the recovery time programmed in bits[5:3]. 0 = Disable recovery times in bits[5:3] and the recovery timing of 3.5 SYSCLKs is inserted.
5:3	**8-Bit I/O Recovery Times:** When bit 6 = 1, this 3-bit field defines the recovery time for 8-bit I/O. **Bit[5:3] SYSCLK Bit[5:3] SYSCLK** 001 1 101 5 010 2 110 6 011 3 111 7 100 4 000 8
2	**16-Bit I/O Recovery Enable:** 1 = Enable, the recovery times programmed in bits[1:0]. 0 = Disable, programmable recovery times in bits[1:0] and the recovery timing of 3.5 SYSCLKs is inserted.
1:0	**16-Bit I/O Recovery Times:** When bit 2 = 1, this 2-bit field defines the recovery time for 16-bit I/O. **Bit[1:0] SYSCLK** 01 1 10 2 11 3 00 4

2.2.11 XBCS—X-BUS CHIP SELECT REGISTER (FUNCTION 0)

Address Offset: 4Eh

Default Value: 03h

Attribute: Read/Write

This register enables/disables accesses to the RTC, keyboard controller and BIOS. Disabling any of these bits prevents the chip select and X-Bus output enable control signal (XOE#) for that device from being generated. This register also provides coprocessor error and mouse functions.

Bit	Description
7	**Extended BIOS Enable:** When bit 7 = 1 (enabled), PCI master accesses to locations FFF80000–FFFDFFFFh are forwarded to ISA and result in the generation of BIOSCS# and XOE#. When forwarding the additional 384-Kbyte region at the top of 4 Gbytes, the PIIX allows the PCI address A[23:20] to propagate to the ISA LA[23:20] lines as all 1's, aliasing this 384-Kbyte region to the top of the 16-Mbyte space. To avoid contention, ISA add-in memory must not be present in this region (00F80000–00FDFFFFh). When bit 7 = 0, the PIIX does not generate BIOSCS# or XOE#.
6	**Lower BIOS Enable:** When bit 6 = 1 (enabled), PCI master, or ISA master accesses to the lower 64-Kbyte BIOS block (E0000–EFFFFh) at the top of 1 Mbyte, or the aliases at the top of 4 Gbyte (FFFE0000–FFFEFFFFh) result in the generation of BIOSCS# and XOE#. When forwarding the region at the top of 4 Gbytes to the ISA Bus, the ISA LA[23:20] lines are all 1's, aliasing this region to the top of the 16 Mbyte space. To avoid contention, ISA addin memory must not be present in this region (00F80000–00FDFFFFh). When bit 6 = 0, the PIIX does not generate BIOSCS# or XOE# during these accesses and does not forward the accesses to ISA.
5	**Coprocessor Error Function Enable:** 1 = Enable, the FERR# input, when asserted, triggers IRQ13 (internal). FERR# is also used to gate the IGNNE# output.
4	**IRQ12/M Mouse Function Enable:** 1 = Mouse function; 0 = Standard IRQ12 interrupt function.
3	**Reserved.**
2	**BIOSCS# Write Protect Enable:** 1 = Enable (BIOSCS# is asserted for BIOS memory read and write cycles in decoded BIOS region); 0 = Disable (BIOSCS# is only asserted for BIOS read cycles).
1	**Keyboard Controller Address Location Enable:** 1 = Enable KBCS# and XOE# for address locations 60h and 64h. 0 = Disable KBCS#/XOE# for accesses to these locations.
0	**RTC Address Location Enable:** 1 = Enable RTCCS#/RTCALE and XOE# for accesses to address locations 70–77h. 0 = Disable RTCCS#/RTCALE and XOE# for these accesses.

2.2.12 PIRQRC[A:D]—PIRQx ROUTE CONTROL REGISTERS (FUNCTION 0)

Address Offset: 60h (PIRQRCA#)–63h (PIRQRCD#)

Default Value: 80h

Attribute: Read/Write

These registers control the routing of the PIRQ[A:D]# signals to the IRQ inputs of the interrupt controller. Each PIRQx# can be independently routed to any one of 11 interrupts. All four PIRQx# lines can be routed to the same IRQx input. Note that the IRQ that is selected through bits[3:0] must be set to level sensitive mode in the corresponding ELCR Register. When a PIRQ signal is routed to an interrupt controller IRQ, the PIIX masks the corresponding IRQ signal.

Bit	Description
7	**Interrupt Routing Enable:** 0 = Enable; 1 = Disable
6:4	**Reserved.** Read as 0s.
3:0	**Interrupt Routing:** When bit 7 = 0, this field selects the routing of the PIRQx to one of the interrupt controller interrupt inputs.

Bits[3:0]	IRQ Routing	Bits[3:0]	IRQ Routing	Bits[3:0]	IRQ Routing
0000	Reserved	0110	IRQ6	1011	IRQ11
0001	Reserved	0111	IRQ7	1100	IRQ12
0010	Reserved	1000	Reserved	1101	Reserved
0011	IRQ3	1001	IRQ9	1110	IRQ14
0100	IRQ4	1010	IRQ10	1111	IRQ15
0101	IRQ5				

2.2.13 TOM—TOP OF MEMORY REGISTER (FUNCTION 0)

Address Offset: 69h

Default Value: 02h

Attribute: Read/Write

This register enables the forwarding of ISA and DMA memory cycles to the PCI Bus and sets the top of main memory accessible by ISA or DMA devices. In addition, this register controls the forwarding of ISA or DMA accesses to the lower BIOS region (E0000-EFFFFh) and the 512–640-Kbyte main memory region (80000-A0000h). The Top of Memory Configuration Register must be set by the BIOS.

Bit	Description
7:4	**Top Of Memory:** The top of memory can be assigned in 1-Mbyte increments from 1–16 Mbytes. ISA or DMA accesses within this region, and not in the memory hole region, are forwarded to PCI.

Bits[7:4]	Top of Memory	Bits[7:4]	Top of Memory	Bits[7:4]	Top of Memory
0000	1 Mbyte	0110	7 Mbyte	1011	12 Mbyte
0001	2 Mbyte	0111	8 Mbyte	1100	13 Mbyte
0010	3 Mbyte	1000	9 Mbyte	1101	14 Mbyte
0011	4 Mbyte	1001	10 Mbyte	1110	15 Mbyte
0100	5 Mbyte	1010	11 Mbyte	1111	16 Mbyte
0101	6 Mbyte				

NOTE:

The PIIX only supports a main memory hole at the top of 16 Mbytes. Thus, if a 1-Mbyte memory hole is created for the TSC's DRAM controller between 15 and 16 Mbytes, the PIIX Top of Memory should be set at 15 Mbytes.

3	**ISA/DMA Lower BIOS Forwarding Enable:** 1 = Enable (forwarded to PCI, if XBCS Register bit 6 = 0); 0 = Disable (contained to ISA). Note that if the XBCS Register bit 6 = 1, ISA/DMA accesses in this region are always contained to ISA.
2	**Reserved.**
1	**ISA/DMA 512–640 Kbyte Region Forwarding Enable**: 1 = Enable (forwarded to PCI); 0 = Disable (contained to ISA).
0	**Reserved.**

2.2.14 MSTAT—MISCELLANEOUS STATUS REGISTER (FUNCTION 0)

Address Offset: 6B–6Ah
Default Value: Undefined
Attribute: Read Only (bit 2 read/write)

This register reports the hardware strapping options selected for internal ISA DMA or external DMA mode and the ISA clock divisor.

Bit	Description
15:3	**Reserved.**
2	**PCI Header Type Bit Enable:** This bit controls the "Header Type Bit" in PIIX register 0Eh which defines the PIIX as a multifunction device. This bit defaults to 1 (multifunction device), and should be left in the default state. This bit is read/write.
1	**Internal ISA DMA or External DMA Mode Status (IEDMAS):** This bit reports the strapping option selected on the TC signal. This bit is 0 for normal DMA operation. This bit indicates strapping at the TC pin during reset (pulled high at reset for a value of 0).
0	**ISA Clock Divisor Status:** This bit reports the strapping option on the SYSCLK signal. 1 = clock divisor of 3 (PCICLK = 25 MHz). 0 = Clock divisor of 4 (PCICLK = 33 MHz). Note that, for PCICLK = 30 MHz, a clock divisor of 4 must be selected and produces a SYSCLK of 7.5 Mhz.

2.2.15 MBIRQ[1:0]—MOTHERBOARD DEVICE IRQ ROUTE CONTROL REGISTERS (FUNCTION 0)

Address Offset: 70h—MBIRQ0; 71h—MBIRQ1
Default Value: 80h
Attribute: Read/Write

These registers control the routing of motherboard device interrupts (MIRQ[1:0]) to the internal IRQ inputs of the interrupt controller. Each MIRQx# can be independently routed to any one of the interrupts. If the Bus Master mode of the IDE interface is used and there is a secondary IDE channel, the interrupt for that channel must be connected to MIRQ0.

Note that when a MIRQ line and a PIRQ# line are steered to the same ISA interrupt, the device connected to the MIRQ line must be set for active high, level-sensitive interrupts. In this case, the ISA interrupt will be masked. Bit 6 of that Motherboard Device IRQ Route Control Register must be programmed to 0.

Bit	Description
7	**Interrupt Routing Enable:** 0 = Enable routing; 1 = Disable routing.
6	**MIRQx/IRQx Sharing Enable:** 0 = Disable sharing; 1 = Enable sharing. When sharing is disabled and bit 7 of this register is 0, the interrupt specified by bits[3:0] is masked. Interrupt sharing should only be enabled when the device connected to the MIRQ line and the device connected to the ISA IRQ line both produce active high, level-sensitive interrupts.
5:4	**Reserved:** Read as 0s.
3:0	**Interrupt Routing:** When bit 7 = 0, this field selects the routing of the MBIRQx to one of the interrupt controller interrupt inputs.

Bits[3:0]	IRQ Routing	Bits[3:0]	IRQ Routing	Bits[3:0]	IRQ Routing
0000	Reserved	0110	IRQ6	1011	IRQ11
0001	Reserved	0111	IRQ7	1100	IRQ12
0010	Reserved	1000	Reserved	1101	Reserved
0011	IRQ3	1001	IRQ9	1110	IRQ14
0100	IRQ4	1010	IRQ10	1111	IRQ15
0101	IRQ5				

2.2.16 MBDMA[1:0]—MOTHERBOARD DEVICE DMA CONTROL REGISTERS (FUNCTION 0)

Address Offset: 76h—MBDMA0#; 77h—MBDMA1#
Default Value: 04h
Attribute: Read/Write

These registers control the routing of motherboard device DMA signals (MDRQ[1:0] and MDAK[1:0]) to the DREQ and DACK# signals on the 8237 DMA controller unit. This register also enables a fast transfer mode (type F, 3 SYSCLK) for motherboard devices.

When a MDRQ/MDAK# pair is programmed for a given 8237 DMA channel and DMC=1, the MDRQ/MDAK# signals are masked. If both motherboard DMAs are used, the motherboard DMAs should be programmed to different compatible DMA channels. Programming both motherboard DMAs to the same compatible DMA channel results in unpredictable device operation.

When DMC=1, this register enables type F transfers and the 4-byte DMA buffer for an ISA peripheral on a given channel. When DMC=0, this register steers the corresponding MDRQ/MDAK# signals to a compatible ISA channel for a motherboard peripheral and also enable type F transfers and the 4-byte DMA buffer.

Bit	Description
7	**Type F and DMA Buffer Enable (FAST):** 1=Enable for the channel selected by bits[2:0]. 0=Disable for the channel selected by bits[2:0].
6:4	**Reserved.** Read as 0s.
3	**Disable Motherboard Channel (DMC):** When this bit 3=0, the MDRQ/MDAK# pair associated with this channel is routed to the compatible ISA channel determined by the CHNL field (bits[2:0]). When bit 3=1, the ISA DREQ/DACK# pair is used for that channel.
2:0	**DMA Channel Select (CHNL):** This field selects the DMA channel connected to the MDRQ/MDAK# pair.

Bits[2:0]	DMA Channel	Bits[2:0]	DMA Channel
000	0	100	default (disabled)
001	1	101	5
010	2	110	6
011	3	111	7

2.2.17 PCSC—PROGRAMMABLE CHIP SELECT CONTROL REGISTER (FUNCTION 0)

Address Offset: 78–79h
Default Value: 0002h
Attribute: Read/Write

This register controls the assertion of the PCS# programmable chip select signal. The PCS# signal is asserted for subtractively decoded I/O cycles generated by PCI masters that fall in the range specified by this register. The address is programmable to any 16 bit I/O space location and the range is programmable to be 4, 8 or 16 bytes. A split range is precluded. The upper 16 address bits (AD[31:16]) must be 0 for the address to be decoded and the PCS# signal asserted. The PCS# signal is never asserted for ISA Bus Masters access.

Bit	Description
15:2	**PCS Address (PCSADDR):** This field defines a 16 bit I/O space address (4 byte range) that causes the PCS# signal to assert. Address bits [3:2] may be masked (considered "don't care") by programming bits [1:0] of this register.
1:0	**PCS Address Mask:** When bit 1 = 1, PCSADDR3 is masked. When bit 0 = 1, PCSADDR2 is masked. **Bits[1:0] Range** 00 4 bytes (default) 01 8 bytes, contiguous 10 Disabled 11 16 bytes, contiguous

2.2.18 SMICNTL—SMI CONTROL REGISTER (FUNCTION 0)

Address Offset: A0h
Default Value: 08h
Attribute: Read/Write

The SMICNTL Register provides Fast-Off Timer control, STPCLK# enable/disable, and CPU clock scaling. This register also enables/disables the system management interrupt (SMI).

Bit	Description
7:5	**Reserved.**
4:3	**Fast-Off Timer Freeze (CTMRFRZ):** This field enables/disables the Fast-Off Timer and when enabled, selects the timer count granularity as shown below: **Bits[4:3]** — **Count Granularity (33 MHz PCICLK)** — **Count Granularity (30 MHz PCICLK)** — **Count Granularity (25 MHz PCICLK)** 00 — 1 Minute — 1.1 Minute — 1.32 Minute 01 — Disabled (default) — Disabled (default) — Disabled (default) 10 — 1 PCICLK — 1 PCICLK — 1 PCICLK 11 — 1 msec — 1.1 msec — 1.32 msec
2	**STPCLK# Scaling Enable (CSTPCLKSC):** 1 = Enable; 0 = Disable. When enabled (and bit 1 = 1), the high and low times for the STPCLK# signal are controlled by the Clock Scaling STPCLK# High Timer and Clock Scaling STPCLK# Low Timer Registers.
1	**STPCLK# Signal Enable (CSTPCLKE):** 1 = Enable; 0 = Disable. When enabled, an APMC Register read causes STPCLK# to be asserted. When disabled, the STPCLK# signal is disabled and is negated (high). Software can set this bit to 0 by writing 0 to it.
0	**SMI# Gate (CSMIGATE):** 1 = Enable; 0 = Disable. When enabled, a system management interrupt condition asserts the SMI# signal. When disabled, the SMI# signal is masked and negated. This bit only affects the SMI# signal and does not affect the detection/recording of SMI events (i.e., this bit does not affect the SMI status bits in the SMIREQ Register). Thus, if an SMI is pending when this bit is set to 1, the SMI# signal is asserted.

2.2.19 SMIEN—SMI ENABLE REGISTER (FUNCTION 0)

Address Offset: A2–A3h
Default Value: 0000h
Attribute: Read/Write

This register enables the generation of SMI (asserting the SMI# signal) for the associated hardware events (bits[5:0]), and software events (bit 7). When a hardware event is enabled, the occurrence of a corresponding event results in the assertion of SMI#, if enabled via the SMICNTL Register. The SMI# is asserted independent of the current power state (Power-On or Fast-Off). The default for all sources in this register is disabled.

Bit	Description
15:8	**Reserved.**
7	**APMC Write SMI Enable:** 1 = Enable; 0 = Disable.
6	**EXTSMI# SMI Enable:** 1 = Enable; 0 = Disable.
5	**Fast-Off Timer SMI Enable:** 1 = Enable; 0 = Disable. When enabled, the timer generates an SMI when it decrements to 0.
4	**IRQ12 SMI Enable (PS/2 Mouse Interrupt):** 1 = Enable; 0 = Disable.
3	**IRQ8 SMI Enable (RTC Alarm Interrupt):** 1 = Enable; 0 = Disable.
2	**IRQ4 SMI Enable (COM2/COM4 Interrupt or Mouse):** 1 = Enable; 0 = Disable.
1	**IRQ3 SMI Enable (COM1/COM3 Interrupt or Mouse):** 1 = Enable; 0 = Disable.
0	**IRQ1 SMI Enable (Keyboard Interrupt):** 1 = Enable; 0 = Disable.

2.2.20 SEE—SYSTEM EVENT ENABLE REGISTER (FUNCTION 0)

Address Offset: A4–A7h
Default Value: 00000000h
Attribute: Read/Write

This register enables hardware events as system events and break events for power management control. The default for each system/break event in this register is disabled. Bits[31,29,15:3,1:0] generate both system and break events and bit 30 generates break events only.

System Events: Activity by these events can keep the system from powering down. When a system event is enabled, the corresponding hardware event activity prevents a Fast-Off power-down condition by reloading the Fast-Off Timer with its initial count.

Break Events: These events can awaken a powered down system. When a break event is enabled, the corresponding hardware event activity powers up the system by negating STPCLK#.

Bit	Description
31	**Fast-Off SMI Enable (FSMIEN):** 1 = Enable; 0 = Disable.
30	**INTR Enable (FINTREN):** 1 = Enable; 0 = Disable. When enabled, INTR is used as a global break event. In this case, any IRQ that is generated causes the system to powerup via the negation of STPCLK#, regardless of the state of bits[15:3,1:0] in this register.
29	**Fast-Off NMI Enable (FNMIEN):** 1 = Enable; 0 = Disable.
28:16	**Reserved.**
15:3	**Fast-Off IRQ[15:3] Enable (FIRQ[15:3]EN):** 1 = Enable; 0 = Disable.
2	**Reserved.**
1:0	**Fast-Off IRQ[1:0] Enable (FIRQ[1:0]EN):** 1 = Enable; 0 = Disable.

2.2.21 FTMR—FAST-OFF TIMER REGISTER (FUNCTION 0)

Address Offset: A8h
Default Value: 0Fh
Attribute: Read/Write

The Fast-Off Timer indicates (through an SMI) that the system has been idle for a preprogrammed period of time. When the timer expires, an SMI special cycle is generated. The count time interval is programmable (via the SMICNTL Register). The granularity of the counter is programmable via the SMICNTL Register.

NOTE:

1. Before writing to the FTMR Register, the Fast-Off Timer must be stopped via bits[4:3] of the SMICNTL Register.

Bit	Description
7:0	**Fast-Off Timer Value:** Bits[7:0] contain one less than the actual count-down value. Thus, if X is programmed into this register, the countdown value is X + 1. The X + 1 value is loaded into the counter when an enabled system event occurs. When the Fast-Off Timer reaches 00h, an SMI is generated and the timer is re-loaded with the X + 1 value. When the Fast-Off Timer is enabled (via the SMICNTL Register), the timer counts down from this value. A read from the FTMR Register returns the value last written.

2.2.22 SMIREQ—SMI REQUEST REGISTER (FUNCTION 0)

Address Offset: AA–ABh
Default Value: 00h
Attribute: Read/Write

The SMIREQ Register contains status bits indicating which enabled event caused a SMI.

NOTES:

1. The SMIREQ bits are set, cleared, or read independently of each other and independently of the CSMIGATE bit in the SMICNTL Register.
2. If software attempts to set a status bit to 0 at the same time that the PIIX is setting it to 1, the bit is set to 1.
3. Each of the SMIREQ bits is set by the PIIX in response to the activation of the corresonding SMI event. If the SMI event is still active when the corresponding SMIREQ bit is set to 0, the PIIX does not set the status bit back to 1 (i.e., there is only one status indication per active SMI event).
4. When an IRQx signal is asserted, the corresponding IRQx status bit is set to 1. If the IRQx signal is still active when software sets the corresponding status bit to 0, the status bit is not set back to 1. The IRQx may be negated before software sets the status bit to 0. However, if the status bit is set to 0 at the same time a new IRQx is activated, the status bit remains at 1. This indicates to the SMI handler that a new SMI event has been detected.
5. If an IRQx is set in level mode and shared by two devices, the IRQ should not be enabled as an SMI# event. The PIIX's SMIREQ bits are essentially set with an edge. When the second IRQ occurs on a shared IRQ, there is not second edge and the SMI# will not be generated for the second IRQ.

Bit	Description
15:8	**Reserved.**
7	**APM SMI Status (RAPMC):** The PIIX sets this bit to 1 to indicate that a write to the APM Control Register caused an SMI. Software sets this bit to 0 by writing 0 to it.
6	**EXTSMI# SMI Status (REXT):** The PIIX sets this bit to 1 to indicate that EXTSMI# caused an SMI. Software sets this bit to 0 by writing 0 to it.
5	**Fast-Off Timer Expired Status (RFOT):** The PIIX sets this bit to 1 to indicate that the Fast-Off Timer expired and caused an SMI. Software sets this bit to 0 by writing 0 to it. Note that the timer re-starts counting 1 the next clock after it expires.
4	**IRQ12 Request SMI Status (RIRQ12):** The PIIX sets this bit to 1 to indicate that IRQ12 caused an SMI. Software sets this bit to 0 by writing 0 to it.
3	**IRQ8# Request SMI Status (RIRQ8):** The PIIX sets this bit to 1 to indicate that IRQ8# caused an SMI. Software sets this bit to 0 by writing 0 to it.
2	**IRQ4 Request SMI Status (RIRQ4):** The PIIX sets this bit to 1 to indicate that IRQ4 caused an SMI. Software sets this bit to 0 by writing 0 to it.
1	**IRQ3 Request SMI Status (RIRQ3):** The PIIX sets this bit to 1 to indicate that IRQ3 caused an SMI. Software sets this bit to 0 by writing 0 to it.
0	**IRQ1 Request SMI Status (RIRQ1):** The PIIX sets this bit to 1 to indicate that IRQ1 caused an SMI. Software sets this bit to 0 by writing 0 to it.

2.2.23 CTLTMR—CLOCK SCALE STPCLK# LOW TIMER (FUNCTION 0)

Address Offset: ACh
Default Value: 00h
Attribute: Read/Write

The value in this register defines the duration of the STPCLK# asserted period when bit 2 in the SMICNTL Register is set to 1. The value in this register is loaded into the STPCLK# Timer when STPCLK# is asserted. The STPCLK# timer is a divide of PCI clocks and is, therefore, frequency dependent.

The base count for a value of 0 is as follows:

50.0 MHz timebase is 42 microseconds
60.0 MHz timebase is 35 microseconds
66.6 MHz timebase is 32 microseconds

The numbers above are derived from the formula: # of PCI clocks STPCLK# asserted (or negated) = 1 + 1056* (programmed value + 1) where "programmed value" = the value programmed in the clock scale STPCLK# low or high timers, register offset 0ACh and 0AEh.

Bit	Description
7:0	**Clock Scaling STPCLK# Low Timer Value:** Bits[7:0] define the duration of the STPCLK# asserted period during clock throttling.

2.2.24 CTHTMR—CLOCK SCALE STPCLK# HIGH TIMER (FUNCTION 0)

Address Offset: AEh
Default Value: 00h
Attribute: Read/Write

The value in this register defines the duration of the STPCLK# negated period when bit 2 in the SMICNTL Register is set to 1. The value in this register is loaded into the STPCLK# timer when STPCLK# is negated. The STPCLK# timer is a divide of PCI clicks and is, therefore, frequency dependent. See the STPCLK# Low Timer description in Section 2.2.23..

Bit	Description
7:0	**Clock Scaling STPCLK# High Timer Value:** Bits[7:0] define the duration of the STPCLK# negated period during clock throttling.

2.3 PCI Configuration Registers—Function 1 (IDE Interface)

The PIIX is a multi-function device, as indicated by bit 7 of the Header Type Register. The PCI IDE interface function uses Function 1.

2.3.1 VID—VENDOR ID REGISTER (FUNCTION 1)

Address Offset: 00–01h
Default Value: 8086h
Attribute: Read Only

The VID Register contains the vendor identification number. This register, along with the Device Identification Register, uniquely identify any PCI device. Writes to this register have no effect.

Bit	Description
15:0	**Vendor Identification Number:** This is a 16-bit value assigned to Intel.

2.3.2 DID—DEVICE IDENTIFICATION REGISTER (FUNCTION 1)

Address Offset: 02–03h
Default Value: 1230h
Attribute: Read Only

The DID Register contains the device identification number. This register, along with the VID Register, define the PIIX. Writes to this register have no effect.

Bit	Description
15:0	**Device Identification Number:** This is a 16-bit value assigned to the PIIX.

2.3.3 PCICMD—COMMAND REGISTER (FUNCTION 1)

Address Offset: 04–05h
Default Value: 0000h
Attribute: Read/Write

The PCICMD Register controls access to the I/O Space Registers.

Bit	Description
15:10	**Reserved:** Read 0.
9	**Fast Back to Back Enable (FBE): (Not Implemented).** This bit is hardwired to 0.
8:5	**Reserved:** Read as 0.
4	**Memory Write and Invalidate Enable (MWI): (Not Implemented).** This bit is hardwired to 0.
3	**Special Cycle Enable (SCE): (Not Implemented).** This bit is hardwired to 0.
2	**Bus Master Enable (BME):** 1 = Enables the PIIX to be an IDE Bus Master. 0 = Disables the PIIX from generating PCI accesses for the IDE Bus Master function. This bit must be programmed to 1 by BIOS for Bus Master IDE operation.
1	**Memory Space Enable (MSE): (Not Implemented).** This bit is hardwired to 1.
0	**I/O Space Enable (IOSE):** This bit controls access to the I/O Space Registers. When IOSE = 1, access to the Legacy IDE ports (both primary and secondary) and the PCI Bus Master IDE I/O Registers is enabled. The Base Address Register for the PCI Bus Master IDE I/O Registers should be programmed before this bit is set to 1.

2.3.4 PCISTS—PCI DEVICE STATUS REGISTER (FUNCTION 1)

Address Offset: 06–07h
Default Value: 0280h
Attribute: Read/Write

PCISTS is a 16-bit status register for the IDE interface function. The register also indicates the PIIX's DEVSEL# signal timing.

Bit	Description
15	**Detected Parity Error (PERR): (Not Implemented).** Read as 0.
14	**SERR# Status (SERRS): (Not Implemented).** Read as 0.
13	**Master-Abort Status (MAS)—R/W:** When the Bus Master IDE interface function, as a master, generates a master abort, MA is set to 1. Software sets MA to 0 by writing 1 to this bit.
12	**Received Target-Abort Status (RTA)—R/W:** When the Bus Master IDE interface function is a master on the PCI Bus and receives a target abort, this bit is set to 1. Software sets RTA to 0 by writing 1 to this bit.
11	**Signaled Target Abort Status (STA)—R/W:** This bit is set when the PIIX IDE interface function is targeted with a transaction that the PIIX terminates with a target abort. Software resets STA to 0 by writing 1 to this bit.
10:9	**DEVSEL# Timing Status (DEVT)—RO:** For the PIIX, DEVT = 01 indicating medium timing for DEVSEL# assertion when performing a positive decode. DEVSEL# timing does not include configuration cycles.
8	**Data Parity Detected (DPD): (Not Implemented).** Read as 0.
7	**Fast Back-to-Back Capable (FBC)—RO:** Hardwired to 1. This bit indicates to the PCI Master that PIIX, as a target, is capable of accepting fast back-to-back transactions.
6:0	**Reserved:** Read as 0s.

2.3.5 RID—REVISION IDENTIFICATION REGISTER (FUNCTION 1)

Address Offset: 08h
Default Value: Refer to stepping information
Attribute: Read Only

This 8-bit register contains device stepping information. Writes to this register have no effect.

Bit	Description
7:0	**Revision ID Byte:** The register is hardwired to the default value during manufacturing.

2.3.6 PI—PROGRAMMING INTERFACE REGISTER (FUNCTION 1)

Address Offset: 09h
Default Value: 80h
Attribute: Read Only

This register contains the device programming interface information related to the Sub Class Code Register and Base Class Code Register definition for this function.

Bit	Description
7:0	**Programming Interface:** 80h = Capable of IDE Bus Master operation.

2.3.7 SUBC—SUB CLASS CODE REGISTER (FUNCTION 1)

Address Offset: 0Ah
Default Value: 01h
Attribute: Read Only

This register indicates the function sub-class in relation to the Base Class Code Register.

Bit	Description
7:0	**Sub Class Code (SUBC):** 01h = IDE controller.

2.3.8 BCC—BASE CLASS CODE REGISTER (FUNCTION 1)

Address Offset: 0Bh
Default Value: 01h
Attribute: Read Only

This register contains the Base Class Code of the IDE function on the PIIX.

Bit	Description
7:0	**Base Class Code (BASEC):** 01h = Mass storage device.

2.3.9 MLT—MASTER LATENCY TIMER REGISTER (FUNCTION 1)

Address Offset: 0Dh
Default Value: 00h
Attribute: Read/Write

MLT controls the amount of time PIIX, as a Bus Master, can burst data on the PCI Bus. The count value is an 8 bit quantity. However, MLT[3:0] are reserved and 0 when determining the count value. MLT is cleared and suspended when PIIX is not asserting FRAME#. When PIIX asserts FRAME#, the counter begins counting. If PIIX finishes its transaction before the count expires, the MLT count is ignored. If the count expires before the transaction completes (count = # of clocks programmed in MLT), PIIX initiates a transaction termination as soon as its PHLDA# is removed. The number of clocks programmed in the MLT represents the guaranteed time slice (measured in PCI clocks) allotted to PIIX. The default value of MLT is 00h, or 0 PCI clocks.

Bit	Description
7:4	**Master Latency Timer Count Value:** PIIX-initiated PCI burst cycles can last indefinitely, as long as PHLDA# remains active. However, if PHLDA# is negated after the burst cycle is initiated, PIIX limits the burst cycle to the number of PCI Bus clocks specified by this field.
3:0	**Reserved.**

2.3.10 HEDT—HEADER TYPE REGISTER (FUNCTION 1)

Address Offset: 0Eh
Default Value: 80h
Attribute: Read Only

The HEDT Register identifies the PIIX as a multi-function device.

Bit	Description
7:0	**Device Type (DEVICET):** 80h = Multi-function device.

2.3.11 BMIBA—BUS MASTER INTERFACE BASE ADDRESS REGISTER (FUNCTION 1)

Address Offset: 20–23h
Default Value: 00000001h
Attribute: Read/Write

This register selects the base address of a 16-byte I/O space to provide a software interface to the Bus Master functions. Only 12 bytes are actually used (6 bytes for primary and 6 bytes for secondary).

Bit	Description
31:16	**Reserved:** Hardwired to 0.
15:4	**Bus Master Interface Base Address:** These bits provide the base address for the Bus Master Interface Registers and correspond to AD[15:4].
3:2	**Reserved:** Hardwired to 0.
1	**Reserved.**
0	**Resource Type Indicator (RTE)—RO:** This bit is hardwired to 1 indicating that the base address field in this register maps to I/O space.

2.3.12 IDETM—IDE TIMING REGISTER (FUNCTION 1)

Address Offset: Primary Channel = 40–41h; Secondary Channel = 42–43h
Default Value: 0000h
Attribute: Read/Write Only

This register controls the PIIX's IDE interface and selects the timing characteristics of the PCI Local Bus IDE cycle.

Bit	Description
15	**IDE Decode Enable (IDE):** 1 = Enable; 0 = Disable. When enabled, I/O transactions on PCI targeting the IDE ATA Register blocks (command block and control block) are positively decoded on PCI and driven on the PIIX IDE interface. When disabled, PIIX subtractively decodes these accesses to ISA.
14	**Reserved.**
13:12	**IORDY Sample Point (ISP):** This field selects the number of clocks between DIOx# assertion and the first IORDY sample point. **Bits[13:12] Number Of Clocks** 00 5 01 4 10 3 11 2
11:10	**Reserved.**
9:8	**Recovery Time (RTC):** This field selects the minimum number of clocks between the last IORDY# sample point and the DIOx# strobe of the next cycle. **Bits[9:8] Number Of Clocks** 00 4 01 3 10 2 11 1
7	**DMA Timing Enable Only (DTE1):** When DTE1 = 1, fast timing mode is enabled for DMA data transfers for drive 1. Note that PIO transfers to the IDE data port still run in compatible timing.
6	**Prefetch and Posting Enable (PPE1):** When PPE1 = 1, prefetch and posting to the IDE data port is enabled for drive 1.
5	**IORDY Sample Point Enable Drive Select 1 (IE1):** When IE1 = 0, IORDY sampling is disabled for Drive 1. The internal IORDY signal is forced asserted guaranteeing that IORDY is sampled asserted at the first sample point as specified by the ISP field in this register. When IE1 = 1 and the currently selected drive (via a copy of bit 4 of 1x6h) is Drive 0, all accesses to the enabled I/O address range sample IORDY. The IORDY sample point is specified by the ISP field in this register.

Bit	Description
4	**Fast Timing Bank Drive Select 1 (TIME1):** When TIME1 = 0, accesses to the data port of the enabled I/O address range use the 16 bit compatible timing PCI local bus path. When TIME1 = 1 and the currently selected drive (via a copy of bit 4 of 1x6h) is Drive 1, accesses to the data port of the enabled I/O address range use the fast timing bank PCI local bus IDE path. Accesses to the data port use fast timing only if bit 7 of this register (DTE1) is zero. Accesses to all non-data ports of the enabled I/O address range use the 8 bit compatible timing PCI local bus path.
3	**DMA Timing Enable Only (DTE0):** When DTE0 = 1, fast timing mode is enabled for DMA data transfers for drive 0. Note that PIO transfers to the IDE data port still run in compatible timing.
2	**Prefetch and Posting Enable (PPE0):** 1 = Enable; 0 = Disable. When enabled, prefetch and posting to the IDE data port is enabled for drive 0.
1	**IORDY Sample Point Enable Drive Select 0 (IE0):** When IE0 = 0, IORDY sampling is disabled for Drive 0. The internal IORDY signal is forced asserted guaranteeing that IORDY is sampled asserted at the first sample point as specified by the ISP field in this register. When IE0 = 1 and the currently selected drive (via a copy of bit 4 of 1x6h) is Drive 0, all accesses to the enabled I/O address range sample IORDY. The IORDY sample point is specified by the ISP field in this register.
0	**Fast Timing Bank Drive Select 0 (TIME0):** When TIME0 = 0, accesses to the data port of the enabled I/O address range uses the 16 bit compatible timing PCI local bus path. When TIME0 = 1 and the currently selected drive (via a copy of bit 4 of 1x6h) is Drive 0, accesses to the data port of the enabled I/O address range use the fast timing bank PCI local bus IDE path. Accesses to the data port use fast timing only if bit 3 of this register (DTE0) is 0. Accesses to all non-data ports of the enabled I/O address range use the 8 bit compatible timing PCI local bus path.

2.4 ISA-Compatible Registers

The ISA-Compatible Registers contain the DMA, timer/counter, and interrupt registers. This group also contains the X-Bus, coprocessor, NMI, and reset registers.

2.4.1 DMA REGISTERS

The PIIX contains DMA circuitry that incorporates the functionality of two 82C37 DMA controllers (DMA1 and DMA2). The DMA Registers control the operation of the DMA controllers and are all accessible from the Host CPU via the PCI Bus interface. In addition, some of the registers are accessed from the ISA Bus via ISA I/O space. Unless otherwise stated, a CPURST sets each register to its default value.

2.4.1.1 DCOM—DMA Command Register

I/O Address: Channels 0–3—08h; Channels 4–7—0D0h
Default Value: 00h (CPURST or Master Clear)
Attribute: Write Only

This 8-bit register controls the configuration of the DMA. Note that disabling channels 4-7 also disables channels 0–3, since channels 0–3 are cascaded into channel 4.

Bit	Description
7	**DACK# ACTIVE Level (DACK# [3:0,(7:5)]):** 1 = Active high; 0 = Active low.
6	**DREQ Sense Assert Level (DREQ[3:0,(7:5)]):** 1 = Active low; 0 = Active high.
5	**Reserved:** Must be 0.
4	**DMA Group Arbitration Priority:** 1 = Rotating priority; 0 = Fixed priority
3	**Reserved**: Must be 0.
2	**DMA Channel Group Enable:** 1 = Disable; 0 = Enable.
1:0	**Reserved**: Must be 0.

2.4.1.2 DCM—DMA Channel Mode Register

I/O Address: Channels 0–3 = 0Bh; Channels 4–7 = 0D6h
Default Value: Bits[7:2] = 0, Bits[1:0] = undefined (CPURST or Master Clear)
Attribute: Write Only

Each channel has a 16-bit DMA Channel Mode Register. The Channel Mode Registers provide control over DMA transfer type, transfer mode, address increment/decrement, and autoinitialization.

Bit	Description
7:6	**DMA Transfer Mode:** Each DMA channel can be programmed in one of four different modes: **Bits[7:6] Transfer Mode** 00 Demand mode 01 Single mode 10 Block mode 11 Cascade mode
5	**Address Increment/Decrement Select:** 0 = Increment; 1 = Decrement.
4	**Autoinitialize Enable:** 1 = Enable; 0 = Disable.

Bit	Description
3:2	**DMA Transfer Type:** When Bits[7:6] = 11, the transfer type bits are irrelevant. **Bits[3:2] Transfer Type** 00 Verify transfer 01 Write transfer 10 Read transfer 11 Illegal
1:0	**DMA Channel Select:** Bits[1:0] select the DMA Channel Mode Register written to by bits[7:2]. **Bits[1:0] Channel** 00 Channel 0 (4) 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)

2.4.1.3 DR—DMA Request Register

I/O Address: Channels 0–3—09h; Channels 4–7—0D2h

Default Value: Bits[1:0] = undefined, Bits[7:2] = 0 (CPURST or Master Clear)

Attribute: Write Only

The Request Register is used by software to initiate a DMA request. The DMA responds to the software request as though DREQx is asserted. These requests are non-maskable and subject to prioritization by the priority encoder network. For a software request, the channel must be in Block Mode. The Request Register status for DMA1 and DMA is output on bits[7:4] of a Status Register read.

Bit	Description
7:3	**Reserved**: Must be 0.
2	**DMA Channel Service Request:** 0 = Resets the individual software DMA channel request bit. 1 = Sets the request bit. Generation of a TC also sets this bit to 0.
1:0	**DMA Channel Select:** Bits[1:0] select the DMA channel mode Register to program with bit 2. **Bits[1:0] Channel** 00 Channel 0 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)

2.4.1.4 Mask Register—Write Single Mask Bit

I/O Address: Channels 0–3—0Ah; Channels 4–7—0D4h
Default Value: Bits[1:0] = undefined; Bit 2 = 1; Bits[7:3] = 0 (CPURST or a Master Clear)
Attribute: Write Only

A channel's mask bit is automatically set when the Current Byte/Word Count Register reaches terminal count (unless the channel is programmed for autoinitialization). Setting the entire register disables all DMA requests until a clear mask register instruction allows them to occur. This instruction format is similar to the format used with the DMA Request Register. Masking DMA channel 4 (DMA controller 2, channel 0) also masks DMA channels [3:0].

Bit	Description
7:3	**Reserved**: Must be 0.
2	**Channel Mask Select:** 1 = Disable DREQ for the selected channel. 0 = Enable DREQ for the selected channel.
1:0	**DMA Channel Select:** Bits[1:0] select the DMA Channel Mode Register for bit 2. **Bits[1:0] Channel** 00 Channel 0 (4) 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)

2.4.1.5 Mask Register—Write All Mask Bits

I/O Address: Channels 0–3—0Fh; Channels 4–7—0DEh
Default Value: Bit[3:0] = 1; Bit[7:4] = 0 (CPURST or Master Clear)
Attribute: Read/Write

A channel's mask bit is automatically set to 1 when the Current Byte/Word Count Register reaches terminal count (unless the channel is programmed for autoinitialization). Setting bits[3:0] to 1 disables all DMA requests until a clear mask register instruction enables the requests. Note that, masking DMA channel 4 (DMA controller 2, channel 0), masks DMA channels [3:0]. Also note that masking DMA controller 2 with a write to port 0DEh also masks DREQ assertions from DMA controller 1.

Bit	Description
7:4	**Reserved**: Must be 0.
3:0	**Channel Mask Bits:** 1 = Disable the corresponding DREQ(s); 0 = Enable the corresponding DREQ(s). **Bit Channel** 0 0 (4) 1 1 (5) 2 2 (6) 3 3 (7)

2.4.1.6 DS—DMA Status Register

I/O Address: Channels 0–3—08h; Channels 4–7—0D0h
Default Value: 00h
Attribute: Read Only

Each DMA controller has a read-only DMA Status Register that indicates which channels have reached terminal count and which channels have a pending DMA request.

Bit	Description
7:4	**Channel Request Status:** When a valid DMA request is pending for a channel (on its DREQ signal line), the corresponding bit is set to 1. When a DMA request is not pending for a particular channel, the corresponding bit is set to 0. The source of the DREQ may be hardware or a software request. Note that channel 4 does not have DREQ or DACK lines, so the response for a read of DMA2 status for channel 4 is irrelevant. **Bit** **Channel** 4 0 5 1 (5) 6 2 (6) 7 3 (7)
3:0	**Channel Terminal Count Status:** 1 = TC is reached; 0 = TC is not reached. **Bit** **Channel** 0 0 1 1 (5) 2 2 (6) 3 3 (7)

2.4.1.7 DMA Base and Current Address Registers (8237 Compatible Segment)

I/O Address: DMA Channel 0000h, DMA Channel 40C0h
DMA Channel 1002h, DMA Channel 50C4h
DMA Channel 2004h, DMA Channel 60C8h
DMA Channel 3006h, DMA Channel 70CCh
Default Value: XXXXh (CPURST or Master Clear)
Attribute: Read/Write

This register works in conjunction with the Low Page Register. After an autoinitialization, this register retains the original programmed value. Autoinitialize takes place after a TC. The address register is automatically incremented or decremented after each transfer. This register is read/written in successive 8-bit bytes. The programmer must issue the "Clear Byte Pointer Flip-Flop" command to reset the internal byte pointer and correctly align the write prior to programming the Current Address Register. Autoinitialize takes place only after a TC.

Bit	Description
15:0	**Base and Current Address [15:0]:** These bits represent address bits[15:0] used when forming the 24-bit address for DMA transfers.

2.4.1.8 DMA Base and Current Byte/Word Count Registers (Compatible Segment)

I/O Address:	DMA Channel 0001h	DMA Channel 40C2h
	DMA Channel 1003h	DMA Channel 50C6h
	DMA Channel 2005h	DMA Channel 60CAh
	DMA Channel 3007h	DMA Channel 70CEh
Default Value:	XXXXh (CPURST or Master Clear)	
Attribute:	Read/Write	

This register determines the number of transfers to be performed. The actual number of transfers is one more than the number programmed in the Current Byte/Word Count Register When the value in the register is decremented from zero to FFFFh, a TC is generated. Autoinitialize can only occur when a TC occurs. If it is not autoinitialized, this register has a count of FFFFh after TC.

For transfers to/from an 8-bit I/O, the Byte/Word count indicates the number of bytes to be transferred. This applies to DMA channels 0–3. For transfers to/from a 16-bit I/O, with shifted address, the Byte/Word count indicates the number of 16-bit words to be transferred. This applies to DMA channels 5–7.

Bit	Description
15:0	**Base and Current Byte/ Word Count:** These bits represent the 16 byte/word count bits used when counting down a DMA transfer.

2.4.1.9 DMA Memory Low Page Registers

I/O Address:	DMA Channel 0087h	DMA Channel 508Bh
	DMA Channel 1083h	DMA Channel 6089h
	DMA Channel 2081h	DMA Channel 708Ah
	DMA Channel 3082h	
Default Value:	XXh (CPURST or Master Clear)	
Attribute:	Read/Write	

This register works in conjunction with the Current Address Register. After an autoinitialization, this register retains the original programmed value. Autoinitialize takes place after a TC.

Bit	Description
7:0	**DMA Low Page [23:16]:** These bits represent address bits[23:16] of the 24-bit DMA address.

2.4.1.10 DMA Clear Byte Pointer Register

I/O Address: Channels 0–3—00Ch; Channels 4–7—0D8h
Default Value: All bits undefined
Attribute: Write Only

Writing to this register executes the Clear Byte Pointer Command. This command is executed prior to reading/writing a new address or word count to the DMA. The command initializes the byte pointer flip-flop to a known state so that subsequent accesses to register contents address upper and lower bytes in the correct sequence. The Clear Byte Pointer Command (or CPURST or the Master Clear Command) clears the internal latch used to address the upper or lower byte of the 16-bit Address and Word Count Registers.

Bit	Description
7:0	**Clear Byte Pointer:** No specific pattern. Command enabled with a write to the I/O port address.

2.4.1.11 DMC—DMA Master Clear Register

I/O Address: Channel 0–3—00Dh; Channel 4–7—0DAh
Default Value: All bits undefined
Attribute: Write Only

This software instruction has the same effect as the hardware reset.

Bit	Description
7:0	**Master Clear:** No specific pattern. Command enabled with a write to the I/O port address.

2.4.1.12 DCLM—DMA Clear Mask Register

I/O Address: Channel 0–3—00Eh; Channel 4–7—0DCh
Default Value: All bits undefined
Attribute: Write Only

Bit	Description
7:0	**Clear Mask Register:** No specific pattern. Command enabled with a write to the I/O port address.

2.4.2 TIMER/COUNTER REGISTER DESCRIPTION

2.4.2.1 TCW—Timer Counter Control Word Register

I/O Address: 043h
Default Value: All bits undefined
Attribute: Write Only

The Timer Control Word Register specifies the counter selection, the operating mode, the counter byte programming order and size of the count value, and whether the counter counts down in a 16-bit or binary-coded decimal (BCD) format. After writing the control word, a new count can be written at any time. The new value takes effect according to the programmed mode.

Bit	Description
7:6	**Counter Select:** The Read Back Command is selected when bits[7:6] are both 1. **Bit[7:6]** — **Function** 00 — Counter 0 01 — Counter 1 10 — Counter 2 11 — Read Back Command
5:4	**Read/Write Select:** The Counter Latch Command is selected when bits[5:4] are both 0. **Bit[5:4]** — **Function** 00 — Counter Latch Command 01 — R/W Least Significant Byte 10 — R/W Most Significant Byte 11 — R/W LSB then MSB
3:1	**Counter Mode Selection:** Bits[3:1] select one of six possible counter modes. **Bit[3:1]** — **Mode** — **Function** 000 — 0 — Out signal on end of count (=0) 001 — 1 — Hardware retriggerable one-shot X10 — 2 — Rate generator (divide by n counter) X11 — 3 — Square wave output 100 — 4 — Software triggered strobe 101 — 5 — Hardware triggered strobe
0	**Binary/BCD Countdown Select:** 0 = Binary countdown. The largest possible binary count is 2^{16}. 1 = Binary coded decimal (BCD) count is used. The largest BCD count allowed is 10^4.

Read Back Command

The Read Back Command is used to determine the count value, programmed mode, and current states of the OUT pin and Null count flag of the selected counter or counters. The Read Back Command is written to the Timer Control Word Register which latches the current states of the above mentioned variables. The value of the counter and its status may then be read by I/O access to the counter address. Note that the Timer Counter Register bit definitions are different during the Read Back Command than for a normal Timer Counter Register write.

Bit	Description
7:6	**Read Back Command:** When bits[7:6] = 11, the Read Back Command is selected during a write to the Timer Control Word Register. Following the Read Back Command, I/O reads from the selected counter's I/O addresses produce the current latch status, the current latched count, or both if bits 4 and 5 are both 0.
5	**Latch Count of Selected Counters:** When bit 5 = 0, the current count value of the selected counters will be latched. When bit 5 = 1, the count will not be latched.
4	**Latch Status of Selected Counters:** When bit 4 = 0, the status of the selected counters will be latched. When bit 4 = 1, the status will not be latched. The status byte format is described in Section 4.3.3, Interval Timer Status Byte Format Register.
3	**Counter 2 Select:** When bit 3 = 1, Counter 2 is selected for the latch command selected with bits 4 and 5. When bit 3 = 0, status and/or count will not be latched.
2	**Counter 1 Select:** When bit 2 = 1, Counter 1 is selected for the latch command selected with bits 4 and 5. When bit 2 = 0, status and/or count will not be latched.
1	**Counter 0 Select:** When bit 1 = 1, Counter 0 is selected for the latch command selected with bits 4 and 5. When bit 1 = 0, status and/or count will not be latched.
0	**Reserved**: Must be 0.

Counter Latch Command

The Counter Latch Command latches the current count value at the time the command is received. If a Counter is latched once and then, some time later, latched again before the count is read, the second Counter Latch Command is ignored. The count read will be the count at the time the first Counter Latch Command was issued. If the counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read successively (read, write, or programming operations for other counters may be inserted between the reads). Note that the Timer Counter Register bit definitions are different during the Counter Latch Command than for a normal Timer Counter Register write. Note that if a counter is programmed to read/write two-byte counts, a program must not transfer control between reading the first and second byte to another routine that also reads from that same counter. Otherwise, an incorrect count will be read.

Bit	Description
7:6	**Counter Selection:** Bits 6 and 7 are used to select the counter for latching. **Bit[7:6] Function** 00 latch counter 0 01 latch counter 1 10 latch counter 2 11 Read Back Command
5:4	**Counter Latch Command:** When bits[5:4] = 00, the Counter Latch Command is selected during a write to the Timer Control Word Register. Following the Counter Latch Command, I/O reads from the selected counter's I/O addresses produce the current latched count.
3:0	**Reserved**: Must be 0.

2.4.2.2 Interval Timer Status Byte Format Register

I/O Address: Counter 0—040h; Counter 1—041h; Counter 2—042h

Default Value: Bits[6:0] = X; Bit 7 = 0

Attribute: Read Only

Each counter's status byte can be read following an Interval Timer Read Back Command. If latch status is chosen (bit 4 = 0, Read Back Command) as a read back option for a given counter, the next read from the counter's Counter Access Ports Register returns the status byte.

Bit	Description
7	**Counter OUT Pin State:** 1 = Pin is 1; 0 = Pin is 0.
6	**Count Register Status:** This bit indicates when the last count written to the Count Register (CR) has been loaded into the counting element (CE). 0 = Count has been transferred from CR to CE and is available for reading. 1 = Count has not been transferred from CR to CE and is not yet available for reading.
5:4	**Read/Write Selection Status:** Bits[5:4] reflect the read/write selection made through bits[5:4] of the Control Register. **Bit[5:4] Function** 00 Counter Latch Command 01 R/W Least Significant Byte (LSB) 10 R/W Most Significant Byte (MSB) 11 R/W LSB then MSB
3:1	**Mode Selection Status:** Bits[3:1] return the counter mode programming. **Bit[3:1] Mode Selected — Bit[3:1] Mode Selected** 000 0 — X11 3 001 1 — 100 4 X10 2 — 101 5
0	**Countdown Type Status:** 0 = Binary countdown; 1 = Binary coded decimal (BCD) countdown.

2.4.2.3 Counter Access Ports Register

I/O Address: Counter 0—040h; Counter 1—041h; Counter 2—042h

Default Value: All bits undefined

Attribute: Read/Write

Each of these I/O ports is used for writing count values to the Count Registers; reading the current count value from the counter by either an I/O read, after a counter-latch command, or after a Read Back Command; and reading the status byte following a Read Back Command.

Bit	Description
7:0	**Counter Port bit[x]**: Each counter I/O port address is used to program the 16-bit Count Register. The order of programming, either LSB only, MSB only, or LSB then MSB, is defined with the Interval Counter Control Register. The counter I/O port is also used to read the current count from the Count Register and return counter programming status following a Read Back Command.

2.4.3 INTERRUPT CONTROLLER REGISTERS

The PIIX contains an ISA-Compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The interrupt registers control the operation of the interrupt controller.

2.4.3.1 ICW1—Initialization Command Word 1 Register

I/O Address: INT CNTRL-1—020h; INT CNTRL-2—0A0h
Default Value: All bits undefined
Attribute: Write Only

A write to Initialization Command Word 1 starts the interrupt controller initialization sequence. Addresses 020h and 0A0h are referred to as the base addresses of CNTRL-1 and CNTRL-2, respectively. An I/O write to the CNTRL-1 or CNTRL-2 base address with bit 4 equal to 1 is interpreted as ICW1. For PIIX-based ISA systems, three I/O writes to "base address + 1" must follow the ICW1. The first write to "base address + 1" performs ICW2, the second write performs ICW3, and the third write performs ICW4.

ICW1 starts the initialization sequence during which the following automatically occur:

1. The Interrupt Mask Register is cleared.
2. IRQ7 input is assigned priority 7.
3. The slave mode address is set to 7.
4. Special Mask Mode is cleared and Status Read is set to IRR.
5. If IC4 was set to 0, then all functions selected by ICW4 are set to 0. However, ICW4 must be programmed in the PIIX implementation of this interrupt controller, and IC4 must be set to 1.

Bit	Description
7:5	**ICW/OCW Select:** These bits should be 000 when programming the PIIX.
4	**ICW/OCW Select:** Bit 4 must be 1 to select ICW1. After the fixed initialization sequence to ICW1, ICW2, ICW3, and ICW4, the controller base address is used to write to OCW2 and OCW3. Bit 4 is 0 on writes to these registers. A 1 on this bit at any time will force the interrupt controller to interpret the write as an ICW1. The controller will then expect to see ICW2, ICW3, and ICW4.
3	**Edge/Level Bank Select (LTIM):** This bit is disabled. Its function is replaced by the Edge/Level Triggered Control (ELCR) Registers.
2	**ADI:** Ignored for the PIIX.
1	**Single or Cascade (SNGL):** This bit must be programmed to 0.
0	**ICW4 Write Required (IC4):** This bit must be set to 1.

2.4.3.2 ICW2—Initialization Command Word 2 Register

I/O Address: INT CNTRL-1—021h; INT CNTRL-2—0A1h
Default Value: All bits undefined
Attribute: Write Only

ICW2 is used to initialize the interrupt controller with the five most significant bits of the interrupt vector address.

Bit	Description
7:3	**Interrupt Vector Base Address:** Bits[7:3] define the base address in the interrupt vector table for the interrupt routines associated with each interrupt request level input.
2:0	**Interrupt Request Level:** Must be programmed to all 0s.

2.4.3.3 ICW3—Initialization Command Word 3 Register (Master Controller)

I/O Address: INT CNTRL-1—021h
Default Value: All bits undefined
Attribute: Write Only

The meaning of ICW3 differs between CNTRL-1 and CNTRL-2. On CNTRL-1, the master controller, ICW3 indicates which CNTRL-1 IRQ line physically connects the INTR output of CNTRL-2 to CNTRL-1.

Bit	Description
7:3	**Reserved:** Must be programmed to all 0s.
2	**Cascaded Mode Enable:** This bit must be programmed to 1 selecting cascade mode.
1:0	**Reserved:** Must be programmed to all 0s.

2.4.3.4 ICW3—Initialization Command Word 3 Register (Slave Controller)

I/O Address: INT CNTRL-2—0A1h
Default Value: All bits undefined
Attribute: Write Only

On CNTRL-2, the slave controller, ICW3 is the slave identification code broadcast by CNTRL-1.

Bit	Description
7:3	**Reserved:** Must be programmed to all 0s.
2:0	**Slave Identification Code:** Must be programmed to 010b.

2.4.3.5 ICW4—Initialization Command Word 4 Register

I/O Address: INT CNTRL-1—021h; INT CNTRL-2—0A1h
Default Value: 01h
Attribute: Write Only

Both PIIX interrupt controllers must have ICW4 programmed as part of their initialization sequence.

Bit	Description
7:5	**Reserved:** Must be programmed to all 0s.
4	**Special Fully Nested Mode (SFNM):** Bit 4, SFNM, should normally be disabled by writing 0 to this bit. If SFNM = 1, the special fully nested mode is programmed.
3	**Buffered Mode (BUF):** Must be programmed to 0 selecting non-buffered mode.
2	**Master/Slave in Buffered Mode:** Should always be programmed to 0. Bit not used.
1	**AEOI (Automatic End of Interrupt):** This bit should normally be programmed to 0. This is the normal end of interrupt. If this bit is 1, the automatic end of interrupt mode is programmed.
0	**Microprocessor Mode:** Must be programmed to 1 indicating an Intel Architecture-based system.

2.4.3.6 OCW1—Operational Control Word 1 Register

I/O Address: INT CNTRL-1—021h; INT CNTRL-2—0A1h
Default Value: 00h
Attribute: Read/Write

OCW1 sets and clears the mask bits in the Interrupt Mask Register (IMR). Each interrupt request line may be selectively masked or unmasked any time after initialization. The IMR stores the interrupt line mask bits. The IMR operates on the IRR. Masking of a higher priority input does not affect the interrupt request lines of lower priority. Unlike status reads of the ISR and IRR, for reading the IMR, no OCW3 is needed. The output data bus contains the IMR when an I/O read is active and the I/O address is 021h or 0A1h (OCW1). All writes to OCW1 must occur following the ICW1-ICW4 initialization sequence, since the same I/O ports are used for OCW1, ICW2, ICW3 and ICW4.

Bit	Description
7:0	**Interrupt Request Mask (Mask [7:0]):** When 1 is written to any bit in this register, the corresponding IRQx line is masked. For example, if bit 4 is set to 1, then IRQ4 is masked. Interrupt requests on IRQ4 do not set channel 4's Interrupt Request Register (IRR) bit as long as the channel is masked. When 0 is written to any bit in this register, the corresponding IRQx is unmasked. Note that masking IRQ2 on CNTRL-1 also masks the interrupt requests from CNTRL-2, which is physically cascaded to IRQ2.

2.4.3.7 OCW2—Operational Control Word 2 Register

I/O Address: INT CNTRL-1—020h; INT CNTRL-2—0A0h

Default Value: Bits[4:0] = undefined; Bits[7:5] = 001

Attribute: Write Only

OCW2 controls both the Rotate Mode and the End of Interrupt Mode. Following a CPURST or ICW initialization, the controller enters the fully nested mode of operation. Both rotation mode and specific EOI mode are disabled following initialization.

Bit	Description
7:5	**Rotate and EOI Codes:** R, SL, EOI - These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations is listed above under the bit definition. **Bits[7:5] Function** — **Bits[7:5] Function** 001 Non-specific EOI Cmd — 000 Rotate in Auto EOI Mode (Clear) 011 Specific EOI Cmd — 111 *Rotate on Specific EOI Cmd 101 Rotate on Non-Specific EOI Cmd — 110 *Set Priority Cmd 100 Rotate in Auto EOI Mode (Set) — 010 No Operation * L0 - L2 Are Used
4:3	**OCW2 Select:** Must be programmed to 00 selecting OCW2.
2:0	**Interrupt Level Select (L2, L1, L0):** L2, L1, and L0 determine the interrupt level acted upon when the SL bit is active (bit 6). When the SL bit is inactive, bits[2:0] do not have a defined function; programming L2, L1 and L0 to 0 is sufficient in this case. **Bits[2:0] Interrupt Level** — **Bits[2:0] Interrupt Level** 000 IRQ 0(8) — 100 IRQ 4(12) 001 IRQ 1(9) — 101 IRQ 5(13) 010 IRQ 2(10) — 110 IRQ 6(14) 011 IRQ 3(11) — 111 IRQ 7(15)

2.4.3.8 OCW3—Operational Control Word 3 Register

I/O Address: INT CNTRL-1—020h; INT CNTRL-2—0A0h

Default Value: Bits[6,0] = 0; Bits[7,4:2] = Undefined; Bits[5,1] = 1

Attribute: Read/Write

OCW3 serves three important functionsEnable Special Mask Mode, Poll Mode control, and IRR/ISR Register read control.

Bit	Description
7	**Reserved:** Must be 0.
6	**Special Mask Mode (SMM):** If ESMM = 1 and SMM = 1, the interrupt controller enters Special Mask Mode. If ESMM = 1 and SMM = 0, the interrupt controller is in normal mask mode. When ESMM = 0, SMM has no effect.
5	**Enable Special Mask Mode (ESMM):** 1 = Enable SMM bit; 0 = Disable SMM bit.
4:3	**OCW3 Select:** Must be programmed to 01 selecting OCW3.

Bit	Description
2	**Poll Mode Command:** 0 = Disable Poll Mode Command. When bit 2 = 1, the next I/O read to the interrupt controller is treated as an interrupt acknowledge cycle indicating highest priority request.
1:0	**Register Read Command:** Bits[1:0] provide control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). When bit 1 = 0, bit 0 doesl not affect the register read selection. When bit 1 = 1, bit 0 selects the register status returned following an OCW3 read. If bit 0 = 0, the IRR will be read. If bit 0 = 1, the ISR will be read. Following ICW initialization, the default OCW3 port address read will be "read IRR". To retain the current selection (read ISR or read IRR), always write 0 to bit 1 when programming this register. The selected register can be read repeatedly without reprogramming OCW3. To select a new status register, OCW3 must be reprogrammed prior to attempting the read. **Bits[1:0] Function** 00 No Action 01 No Action 10 Read IRQ Register 11 Read IS Register

2.4.3.9 ELCR1—Edge/Level Triggered Register

I/O Address: INT CNTRL-1—4D0h
Default Value: 00h
Attribute: Read/Write

ELCR1 Register allows IRQ3–IRQ7 to be edge or level programmable on an interrupt-by-interrupt basis. IRQ0, IRQ1 and IRQ2 are not programmable and are always edge sensitive.

Bit	Description
7	**IRQ7 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
6	**IRQ6 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
5	**IRQ5 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
4	**IRQ4 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
3	**IRQ3 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
2:0	**Reserved:** Must be 0.

2.4.3.10 ELCR2—Edge/Level Triggered Register

I/O Address: INT CNTRL-2—4D1h
Default Value: 00h
Attribute: Read/Write

ELCR2 Register allows IRQ[15,14,12:9] to be edge or level programmable on an interrupt by interrupt basis. Note that IRQ[13,8#] are not programmable and are always edge sensitive.

Bit	Description
7	**IRQ15 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
6	**IRQ14 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
5	**Reserved:** Must be 0.
4	**IRQ12 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
3	**IRQ11 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
2	**IRQ10 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
1	**IRQ9 ECL:** 0 = edge triggered mode; 1 = level sensitive mode.
0	**Reserved:** Must be 0.

2.4.4 X-BUS, COPROCESSOR, AND RESET REGISTERS

2.4.4.1 Reset X-Bus IRQ12 and IRQ1 Register

I/O Address: 60h
Default Value: N/A
Attribute: Read Only

This register clears the mouse interrupt function and the keyboard interrupt (IRQ1). Reads to this address are monitored by the PIIX. When the mouse interrupt function is enabled (X-Bus Chip Select Register), the mouse interrupt function is provided on the IRQ12/M input signal. In this mode, a mouse interrupt generates an interrupt through IRQ12 to the Host CPU. A read of 60h releases IRQ12. Reads/writes flow through to the ISA Bus.

Bit	Description
7:0	**Reset IRQ12 and IRQ1:** No specific pattern. A read of address 60h executes the command.

2.4.4.2 Coprocessor Error Register

I/O Address: F0h
Default Value: N/A
Attribute: Write Only

Writing to this register causes the PIIX to assert IGNNE#. The PIIX also negates IRQ13 (internal to the PIIX). Note that IGNNE# is not asserted unless FERR# is active. Reads/writes flow through to the ISA Bus.

Bit	Description
7:0	No special pattern required: A write to address F0h executes the command.

2.4.4.3 RC—Reset Control Register

I/O Address: CF9h
Default Value: 00h
Attribute: Read/Write

Bits 1 and 2 in this register are used by the PIIX to generate a hard reset or a soft reset.

Bit	Description
7:4	**Reserved.**
3	**Reserved.**
2	**Reset CPU (RCPU):** This bit is used to initiate a hard or soft reset to the CPU. To perform a reset, this bit should be set to 0 and then set to 1. This "0" to "1" transition initiates the reset. During a hard reset, the PIIX asserts CPURST, PCIRST#, and RSTDRV. The PIIX initiates a hard reset when this register is programmed for a hard reset or PWROK transitions from low to high. This bit cannot be read as 1.
1	**System Reset (SRST):** This bit is used in conjunction with bit 2 in this register to initiate a hard reset. When SRST = 1, the PIIX initiates a hard reset to the CPU when bit 2 in this register transitions from 0 to 1. When SRST = 0, the PIIX initiates a soft reset when bit 2 in this register transitions from 0 to 1.
0	**Reserved.**

2.4.5 NMI REGISTERS

The NMI logic incorporates two different 8-bit registers. The CPU reads the NMISC Register to determine the NMI source (bits set to 1). After the NMI interrupt routine processes the interrupt, software clears the NMI status bits by setting the corresponding enable/disable bit to 1. The NMI Enable and Real-Time Clock Register can mask the NMI signal and disable/enable all NMI sources.

To ensure that all NMI requests are serviced, the NMI service routine software flow should be as follows:

1. NMI is detected by the processor on the rising edge of the NMI input.
2. The processor will read the status stored in port 061h to determine what sources caused the NMI. The processor may then set to 0 the register bits controlling the sources that it has determined to be active. Between the time the processor reads the NMI sources and sets them to 0, an NMI may have been generated by another source. The level of NMI will then remain active. This new NMI source will not be recognized by the processor because there was no edge on NMI.
3. The processor must then disable all NMIs by setting bit 7 of port 070H to 1 and then enable all NMIs by setting bit 7 of port 070H to 0. This will cause the NMI output to transition low then high if there are any pending NMI sources. The CPU's NMI input logic will then register a new NMI.

2.4.5.1 NMISC—NMI Status Control Register

I/O Address: 061h
Default Value: 00h
Attribute: Read/Write, Read Only

This register reports the status of different system components, control the output of the speaker counter (Counter 2), and gate the counter output that drives the SPKR signal.

Bit	Description
7	**SERR# NMI Source Status—RO:** Bit 7 is set if a system board agent (PCI devices or main memory) detects a system board error and pulses the PCI SERR# line. This interrupt source is enabled by setting bit 2 to 0. To reset the interrupt, set bit 2 to 0 and then set it to 1. When writing to port 061h, bit 7 must be 0.
6	**IOCHK# NMI Source Status—RO:** Bit 6 is set if an expansion board asserts IOCHK# on the ISA Bus. This interrupt source is enabled by setting bit 3 to 0. To reset the interrupt, set bit 3 to 0 and then set it to 1. When writing to port 061h, bit 6 must be 0.
5	**Timer Counter 2 OUT Status—RO:** The Counter 2 OUT signal state is reflected in bit 5. The value on this bit following a read is the current state of the Counter 2 OUT signal. Counter 2 must be programmed following a CPURST for this bit to have a determinate value. When writing to port 061h, bit 5 must be 0.
4	**Refresh Cycle Toggle—RO:** The Refresh Cycle Toggle signal toggles from either 0 to 1 or 1 to 0 following every refresh cycle. When writing to port 061h, bit 4 must be 0.
3	**IOCHK# NMI Enable—R/W:** 1 = Clear and disable; 0 = Enable IOCHK# NMIs.
2	**PCI SERR# Enable—R/W:** 1 = Clear and Disable; 0 = Enable.
1	**Speaker Data Enable—R/W:** 0 = SPKR output is 0; 1 = the SPKR output is the Counter 2 OUT signal value.
0	**Timer Counter 2 Enable—R/W:** 0 = Disable; 1 = Enable.

2.4.5.2 NMI Enable and Real-Time Clock Address Register

I/O Address: 070h
Default Value: Bit[6:0] = undefined; Bit 7 = 1
Attribute: Write Only

This port is shared with the real-time clock. Do not modify the contents of this register without considering the effects on the state of the other bits. Reads and writes to this register address flow through to the ISA Bus.

Bit	Description
7	**NMI Enable:** 1 = Disable; 0 = Enable.
6:0	**Real Time Clock Address:** Used by the Real Time Clock on the Base I/O component to address memory locations. Not used for NMI enabling/disabling.

2.5 System Power Management Registers

This section describes two power management registers—APMS and APMC Registers. These registers are located in normal I/O space and must be accessed (via the PCI Bus) with 8 bit accesses.

2.5.1 APMC—ADVANCED POWER MANAGEMENT CONTROL PORT

I/O Address: 0B2h
Default Value: 00h
Attribute: Read/Write

This register passes data (APM Commands) between the OS and the SMI handler. In addition, writes can generate an SMI and reads can cause STPCLK# to be asserted. The PIIX operation is not affected by the data in this register.

Bit	Description
7:0	**APM Control Port (APMC):** Writes to this register store data in the APMC Register and reads return the last data written. In addition, writes generate an SMI, if bit 7 of the SMIEN Register and bit 0 of the SMICNTL Register are both is set to 1. Reads cause the STPCLK# signal to be asserted, if bit 1 of the SMICNTL Register is set to 1. Reads do not generate an SMI.

2.5.2 APMS—ADVANCED POWER MANAGEMENT STATUS PORT

I/O Address: 0B3h
Default Value: 00h
Attribute: Read/Write

This register passes status information between the OS and the SMI handler. The PIIX operation is not affected by the data in this register.

Bit	Description
7:0	**APM Status Port (APMS):** Writes store data in this register and reads return the last data written.

2.6 PCI Bus Master IDE Registers

The PCI Bus Master IDE function uses 16 bytes of I/O space, allocated via the BMIBA Register (A PCI Base Address Register). All Bus Master IDE I/O Space Registers can be accessed as byte, word, or Dword quantities. The description of the 16 bytes of I/O registers follows:

2.6.1 BMICOM—BUS MASTER IDE COMMAND REGISTER

Address Offset: Primary Channel—Base + 00h; Secondary Channel—Base + 08h

Default Value: 00h

Attribute: Read / Write

Bit	Description
7:4	**Reserved**
3	**Bus Master Read/Write Control (RWCON):** 0 = Reads; 1 = Writes. This bit must NOT be changed when the Bus Master function is active.
2:1	**Reserved**
0	**Start/Stop Bus Master (SSBM):** 1 = Start; 0 = Stop. When this bit is set to 1, Bus Master operation starts. The controller transfers data between the IDE device and memory only when this bit is set. Master operation can be stopped by writing 0 to this bit. This results in all state information being lost (i.e., master mode operation cannot be stopped and then resumed). If this bit is set to 0 while Bus Master operation is still active (i.e., Bit 0 = 1 in the Bus Master IDE Status Register for that IDE channel) and the drive has not yet finished its data transfer (bit 2 = 0 in the channel's Bus Master IDE Status Register), the Bus Master command is aborted and data transferred from the drive may be discarded before being written to system memory. This bit is intended to be set to 0 after the data transfer is completed, as indicated by either bit 0 or bit 2 being set in the IDE Channel's Bus Master IDE Status Register.

2.6.2 BMISTA—BUS MASTER IDE STATUS REGISTER

Address Offset: Primary Channel—Base + 02h; Secondary Channel—Base + 0Ah

Default Value: 00h

Attribute: Read/Write Clear

This register provides status information about the IDE device and state of the IDE DMA transfer. Table 6 describes IDE Interrupt Status and Bus Master IDE Active bit states after a DMA transfer has been started.

Bit	Description
7	**Reserved:** This bit is hardwired to 0.
6	**Drive 1 DMA Capable (DMA1CAP)—R/W:** 1 = Drive 1 is capable of DMA transfers. This bit is a software controlled status bit that indicates IDE DMA device capability and does not affect hardware operation.
5	**Drive 0 DMA Capable (DMA0CAP)—R/W:** 1 = Drive 0 is capable of DMA transfers. This bit is a software controlled status bit that indicates IDE DMA device capability and does not affect hardware operation.
4:3	**Reserved.**

Bit	Description
2	**IDE Interrupt Status—R/WC:** This bit, when set to 1, indicates when an IDE device has asserted its interrupt line. When bit 2 = 1, all read data from the IDE device has been transferred to main memory and all write data has been transferred to the IDE device. Software sets this bit to 0 by writing 1 to it. **The IRQ14 signal (pin 83) must be used for the primary channel and MIRQ0 is used for the secondary channel.** If the interrupt status bit is set to 0 by writing 1 to this bit while the interrupt line (IRQ14 or MIRQ0) is still at the active level, this bit remains 0 until another assertion edge is detected on the interrupt line.
1	**IDE DMA Error—R/WC:** This bit is set to 1 when the PIIX encounters a target abort or master abort while transferring data on the PCI Bus. Software sets this bit to 0 by writing 1 to it.
0	**Bus Master IDE Active (BMIDEA)—RO:** The PIIX sets this bit to 1 when bit 0 in the BMICOM Register is set to 1. The PIIX sets this bit to 0 when the last transfer for a region is performed (where EOT for that region is set in the region descriptor). The PIIX also sets this bit to 0 when bit 0 of the BMICOM Register is set to 0. When this bit is read as 0, all data transferred from the drive during the previous Bus Master command is visible in system memory, unless the Bus Master command was aborted.

Table 6. Interrupt/Activity Status Combinations

Bit 2	Bit 0	Description
0	1	DMA transfer is in progress. No interrupt has been generated by the IDE device.
1	0	The IDE device generated an interrupt and the Physical Region Descriptors exhausted. This is normal completion where the size of the physical memory regions is equal to the IDE device transfer size.
1	1	The IDE device generated an interrupt. The controller has not reached the end of the physical memory regions. This is a valid completion case when the size of the physical memory regions is larger than the IDE device transfer size.
0	0	Error condition. If the IDE DMA Error bit is 1, there is a problem transferring data to/from memory. Specifics of the error have to be determined using bus-specific information. If the Error bit is 0, the PRD's specified a smaller size than the IDE transfer size.

2.6.3 BMIDTP—BUS MASTER IDE DESCRIPTOR TABLE POINTER REGISTER

Address Offset: Primary Channel—Base + 04h; Secondary Channel—Base + 0Ch
Default Value: 00000000h
Attribute: Read/Write

This register provides the base memory address of the descriptor table. The Descriptor Table must be Dword aligned and not cross a 4-Kbyte boundary in memory.

Bit	Description
31:2	**Descriptor Table Base Address:** Bits[31:2] correspond to A[31:2].
1:0	**Reserved.**

3.0 FUNCTIONAL DESCRIPTION

This section describes each of the major functions on the PIIX including the memory and I/O address map, DMA controller, interrupt controller, timer/counter, and power management. The PCI, ISA, X-Bus, and IDE interfaces.

3.1 Memory and I/O Address Map

The PIIX interfaces to two system buses—PCI and ISA Buses. The PIIX provides positive decode for certain I/O and memory space accesses on these buses as described in this section. ISA masters and DMA devices have access to PCI memory and some of the internal PIIX registers as described in the Register Description section. ISA masters and DMA devices do not have access to host or PCI I/O space.

3.1.1 I/O ACCESSES

The PIIX positively decodes accesses to the PCI Configuration Registers (PCI only), power management registers (PCI only), and Bus Master IDE interface registers (PCI only). The PIIX also positively decodes the ISA-Compatible Registers (PCI and ISA), except for the DMA Register I/O space which is subtractively decoded. For details concerning accessing these registers, see Register Description section.

The PIIX also provides positive decode for BIOS, X-Bus, and system event decode for SMM support. In addition, the PIIX positively decodes PCI Bus accesses to registers located on the IDE device, when enabled. For IDE port accesses, see PCI Local Bus IDE section.

3.1.2 MEMORY ADDRESS MAP

For PCI accesses to ISA memory, accesses below 16 Mbyte (including BIOS space) that are not claimed by a PCI device (subtractive decode) are forwarded to ISA. For write accesses that are not claimed by an ISA slave, the cycle completes normally (i.e. 8-bit, 6 SYSCLK cycle). For read accesses that are not claimed by an ISA slave, the PIIX returns data corresponding to the state of the ISA Bus and completes the cycle normally (i.e. 8-bit, 6 SYSCLK cycle).

For ISA/DMA accesses to main memory, all accesses to memory locations 0–512 Kbytes (512–640 Kbytes, if enabled), or accesses above 1 Mbyte and below the top of memory are forwarded to the PCI Bus (Table 7). The Top of Memory is equal to the value programmed in the Top of Memory Register (bits [7:3]). All remaining ISA originated accesses are confined to the ISA Bus.

Table 7. DMA and ISA Master Accesses to Main Memory

Memory Space	Response
Top of main memory to 128 Mbytes	Confine to ISA
1 Mbyte to top of main memory	Forward to main memory[1]
1 Mbyte minus 128 Kbytes to 1 Mbyte minus 64 Kbytes	Confine to ISA[2]
640 Kbytes to 1 Mbyte minus 128 Kbytes	Confine to ISA
512–640 Kbytes	Confine to ISA[3]
0–512 Kbytes	Forward to PCI

NOTES:
1. Except accesses to programmed memory hole.
2. Forward to main memory if bit 6 = 0 in the XBCS Register and bit 3 = 1 in the TOM Register.
3. Forward to main memory if bit 1 = 0 in the TOM Register.

3.1.3 BIOS MEMORY

The PIIX supports 512 Kbytes of BIOS space. This includes the normal 128-Kbyte space plus an additional 384-Kbyte BIOS space (known as the extended BIOS area). The XBCS Register provides BIOS space access control. Access to the lower 64-Kbyte block of the 128-Kbyte space and the extended BIOS space can be individually enabled/disabled. In addition, write protection can be programmed for the entire BIOS space.

PCI Access to BIOS Memory

The 128-Kbyte BIOS memory space is located at 000E0000–000FFFFFh (top of 1 Mbyte) and is aliased at FFFE0000h (top of 4 Gbytes). This 128-Kbyte block is split into two 64-Kbyte blocks. Accesses to the top 64 Kbytes (000F0000–000FFFFFh) are forwarded to the ISA Bus and BIOSCS# is always generated. Accesses to the bottom 64 Kbytes (000E0000–000EFFFFh) are forwarded to the ISA Bus and BIOSCS# is only generated when this BIOS region is enabled. If this BIOS region is enabled (bit 6 = 1 in the XBCS Register), accesses to the aliased region at the top of 4 Gbytes (FFFE0000h - FFFEFFFFh) are forwarded to ISA and BIOSCS# generated. If disabled, these accesses are not forwarded to ISA and BIOSCS# is not generated.

The additional 384-Kbyte region resides at FFF80000–FFFDFFFFh. If this BIOS region is enabled (bit 7 = 1 in the XBCS Register), these accesses (FFF80000h–FFFDFFFFh) are forwarded to ISA and BIOSCS# generated. If disabled, these accesses are not forwarded to ISA and BIOSCS# not generated.

ISA Access to BIOS Memory

The PIIX confines all ISA-initiated BIOS accesses to the top 64 Kbytes of the 128-Kbyte region (F0000–FFFFFh) to the ISA Bus, even if BIOS is shadowed in main memory. Accesses to the bottom 64 Kbytes of the 128-Kbyte BIOS region (E0000–EFFFFh) are confined to the ISA Bus, when this region is enabled. When the BIOS region is disabled, accesses are forwarded to main memory.

Accesses to the top 64-Kbyte BIOS region always generates BIOSCS#. Accesses to the bottom 64-Kbyte BIOS region generate BIOSCS#, when this region is enabled.

3.2 PCI Interface

The PIIX incorporates a fully PCI Bus-compatible master and slave interface. As a PCI master, the PIIX runs cycles on behalf of DMA, ISA masters, or a Bus Master IDE. As a PCI slave, the PIIX accepts cycles initiated by PCI masters targeted for the PIIX's internal register set or the ISA Bus. The PIIX directly supports the PCI interface running at either 25 MHz, 30 MHz, or 33 MHz.

3.2.1 PCI COMMAND SET

Bus commands indicate to the slave the type of transaction the master is requesting. Bus commands are encoded on the C/BE[3:0]# lines during the address phase of a PCI cycle.

Table 8. PCI Commands

C/BE[3:0]#	Command Type As Slave	Supported As Slave	Supported As Master
0000	Interrupt Acknowledge	Yes	No
0001	Special Cycle	Yes[4]	No
0010	I/O Read	Yes	No
0011	I/O Write	Yes	No
0100	Reserved	No[3]	No
0101	Reserved	No[3]	No
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved	No[3]	No
1001	Reserved	No[3]	No
1010	Configuration Read	Yes	No
1011	Configuration Write	Yes	No
1100	Memory Read Multiple	No[2]	No
1101	Reserved	No[3]	No
1110	Memory Read Line	No[2]	No
1111	Memory Write and Invalidate	No[1]	No

NOTES:
1. Treated as Memory Write.
2. Treated as Memory Read.
3. Reserved cycles are considered invalid by the PIIX and are completely ignored. All internal address decoding is ignored and DEVSEL# is never to be asserted.
4. The PIIX responds to 1 type of special cyclea Shut Down special cycle. All other special cycles are ignored by the PIIX.

3.2.2 TRANSACTION TERMINATION

The PIIX supports the standard PCI cycle terminations as described in the PCI Local Bus specification.

PIIX as a Master—Master-Initiated Termination

The PIIX supports three forms of master-initiated termination:

1. Normal termination of a completed transaction.
2. Normal termination of an incomplete transaction due to timeout (applies to line buffer operations-IDE Bus Master).
3. Abnormal termination due to the slave not responding to the transaction (Abort).

PIIX as a Master—Response to Target-Initiated Termination

As a master, the PIIX responds in one of three ways to a target-termination—Target-Abort, Retry, or Disconnect.

PIIX as a Target—Target-Initiated Termination

The PIIX supports three forms of target-initiated termination—Disconnect, Retry, Target Abort.

3.2.3 PARITY SUPPORT

As a master, the PIIX generates address parity for read/write cycles and data parity when the PIIX is providing the data. As a slave, the PIIX generates data parity for read cycles. The PIIX does not check parity and does not generate SERR#. However, the PIIX does generate an NMI when another PCI device asserts SERR# (if enabled).

PAR is the calculated parity signal. PAR is even parity and is calculated on 36 bitsAD[31:0] signals plus C/BE[3:0]#. PAR is always calculated on 36 bits, regardless of the valid byte enables. PAR is only guaranteed to be valid one PCI clock after the corresponding address or data phase.

3.2.4 PCI ARBITRATION

The PIIX requests the use of the PCI Bus on behalf of ISA devices (Bus Masters and DMA) and IDE DMA slave devices using the PHOLD# and PHLDA# signals. These signals connect to the TSC where the PCI arbiter is located.

ISA devices (Bus Master or DMA) assert DREQ to gain access to the ISA Bus. In response, the PIIX asserts PHOLD#. The PIIX keeps DACK negated until the PIIX has ownership of the PCI Bus and Memory. The PCI arbiter asserts PHLDA# to the PIIX when the above conditions are met. The PIIX gives ownership of the ISA Bus (PCI and Memory) to the ISA device after sampling PHLDA# asserted.

3.3 ISA Interface

The PIIX incorporates a fully ISA Bus compatible master and slave interface. The PIIX directly drives five ISA slots without external data buffers. External transceivers are used on the SA[19:8] and SBHE# signals to permit these signals to be used with the IDE interface (Figure 1). The ISA interface also provides byte swap logic, I/O recovery support, wait state generation, and SYSCLK generation.

The ISA interface supports the following types of cycles:

- PCI master-initiated I/O and memory cycles to the ISA Bus
- DMA compatible cycles between main memory and ISA I/O and between ISA I/O and ISA memory
- Enhanced DMA cycles between PCI memory and ISA I/O (for motherboard devices only)
- ISA refresh cycles initiated by either the PIIX or an external ISA master
- ISA master-initiated memory cycles to PCI and ISA master-initiated I/O cycles to the internal PIIX registers, as shown in ISA-Compatible Register table in the Register Description section.

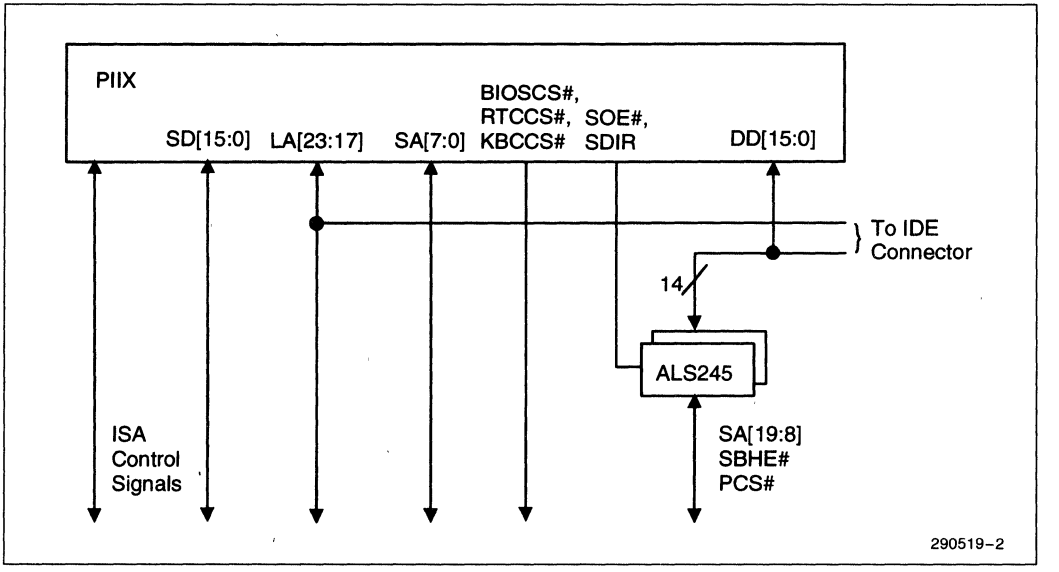


Figure 1. ISA Interface

3.4 DMA Controller

The DMA circuitry incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels (Channels 0–3 and Channels 5–7). DMA Channel 4 is used to cascade the two controllers and defaults to cascade mode in the DMA Channel Mode (DCM) Register. In addition to accepting requests from DMA slaves, the DMA controller also responds to requests that are initiated by software. Software may initiate a DMA service request by setting any bit in the DMA Channel Request Register to 1. The DMA controller for Channels 0–3 is referred to as "DMA-1" and the controller for Channels 4–7 is referred to as "DMA-2".

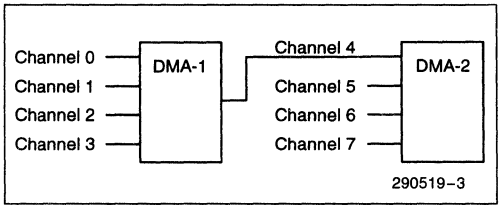


Figure 2. Internal DMA Controller

Each DMA channel is hardwired to the compatible settings for DMA device size; channels [3:0] are hardwired to 8-bit count-by-bytes transfers and channels [7:5] are hardwired to 16-bit count-by-words (address shifted) transfers. The PIIX provides the timing control and data size translation necessary for the DMA transfer between the memory (ISA or main memory) and the ISA Bus device. ISA-Compatible DMA timing is supported. Type F transfers are supported for motherboard devices and ISA add-in cards, when programmed via the MBDMAx Register.

The PIIX provides 24-bit addressing in compliance with the ISA-Compatible specification. Each channel includes a 16-bit ISA-Compatible Current Register that contains the 16 least-significant bits of the 24-bit address, an ISA Compatible Page Register that contains the eight next most significant bits of address. The DMA controller also features refresh address generation, and auto-initialization following a DMA termination.

The DMA controller is either in master or slave mode. In master mode, the DMA controller is either servicing a DMA slave's request for DMA cycles or allowing a 16-bit ISA master to use the bus (via a

cascaded DREQ signal). In slave mode, the PIIX monitors both the ISA Bus and PCI, decoding and responding to I/O read and write commands that address its registers.

Note that a DMA device (I/O device) is always on the ISA Bus, but the memory referenced is located on either an ISA Bus device or on PCI. When the PIIX is running a compatible DMA cycle, it drives the MEMR# or MEMW# strobes if the address is less than 16 Mbytes (000000–FFFFFFh). These memory strobes are generated regardless of whether the cycle is decoded for PCI or ISA memory. The SMEMR# and SMEMW# are generated if the address is less than 1 Mbytes (0000000–00FFFFFh). If the address is greater than 16 Mbytes (1000000–7FFFFFFh), the MEMR# or MEMW# strobe is not generated in order to avoid aliasing problems.

The PIIX drives the AEN signal asserted (high) during DMA cycles to prevent the I/O devices from misinterpreting the DMA cycle as a valid I/O cycle. The BALE signal is also driven high during DMA cycles. Also, during DMA memory read cycles to the PCI Bus, the PIIX return all 1s to the ISA Bus if the PCI cycle is either target aborted or master aborted.

NOTES:

1. For type F timing mode DMA transfers, the channel must be programmed with a memory range that will be forwarded to PCI. This means that if BIOS detects that ISA memory is used in the system (i.e., that the top of memory reported to the operating system is higher than the top of memory programmed in the PIIX Top of Memory Register), the BIOS should not enable type F for any channel.
2. All DMA channels support type F transfers. However, only two channels can be programmed for type F transfers at the same time.
3. In external DMA mode (selected via a strapping option on the TC signal) the PIIX tri-states the AEN, TC, and DACK[7:5, 3:0]# signals, and also forwards PCI master I/O accesses to location 0000h to ISA.

3.4.1 TYPE F TIMING

The type F DMA cycles can be used with motherboard devices and ISA add-in cards, through the use of the MDRQ[1:0] and MDAK[1:0]# signals. The type F cycles occur back to back at a minimum repetition rate of 3 SYSCLKs (360 ns minimum). The type F cycles are always performed using the 4-byte DMA buffer.

DMA Buffer for Type F Transfers

The PIIX has a 4-byte buffer that is used to reduce the PCI utilization resulting from DMA transfers by motherboard devices. The DMA buffer is always used in conjunction with the type F transfers. The type F transfers and the use of the DMA buffer are invoked in the MBDMAx Register. The 4-byte buffer and the type F timings may be used only when the DMA channel is programmed to increment mode (not decrement), and cannot be used when the channel is programmed to operate in block mode (single transfer mode and demand mode are legal). In addition, verify transfers are not supported with type F DMA.

3.4.2 ISA REFRESH CYCLES

Refresh cycle requests are generated by two sources—the refresh controller inside the PIIX component or ISA Bus Masters other than the PIIX. In both cases, the PIIX generates the ISA memory refresh. The PIIX enables address lines SA[7:0]. Thus, when MEMR# goes active, the entire ISA system memory is refreshed at one time. Memory slaves on the ISA Bus must not drive any data onto the data bus during the refresh cycle. The PIIX maintains a four deep buffer to record internally generated refresh requests that have not been serviced. Counter 1 in the timer register set should be programmed to provide a request for refresh about every 15 μs.

PIIX Initiated Refresh Cycle

The PIIX asserts REFRESH# to indicate a refresh cycle and then drives the address lines SA[7:0] onto the ISA Bus and generates MEMR# and SMEMR#. The PIIX drives AEN and BALE high for the entire refresh cycle. The memory device may extend this refresh cycle by pulling IOCHRDY low.

ISA Bus refresh cycles are completely decoupled from DRAM Refresh. Transactions driven by PCI masters that target ISA or IDE resources while refresh is active are held off with wait states until the refresh is complete.

ISA Master Initiated Refresh Cycle

If an ISA Bus Master holds the ISA Bus longer than 15 μs, the ISA master must initiate memory refresh cycles. If the ISA Bus Master initiates a refresh cycle before it relinquishes the bus, it floats the address lines and control signals and asserts the REFRESH# to the PIIX. The PIIX drives address lines SA[7:0] and MEMR# onto the ISA Bus. BALE is driven high and AEN is driven low for the entire refresh cycle.

If the ISA Bus Master holding the bus does not a generate a refresh request and the PIIX's internal refresh request is not serviced within the normal 15 μs, a refresh queue counter is incremented. The counter records up to four incomplete refresh cycles, which are all executed as soon as PIIX gets the ISA Bus.

3.4.3 ISA DATA RETURNED ON A MASTER ABORT CYCLE

The PIIX will return random data to an ISA device on a Master Abort Cycle (i.e., ISA DMA access to main memory).

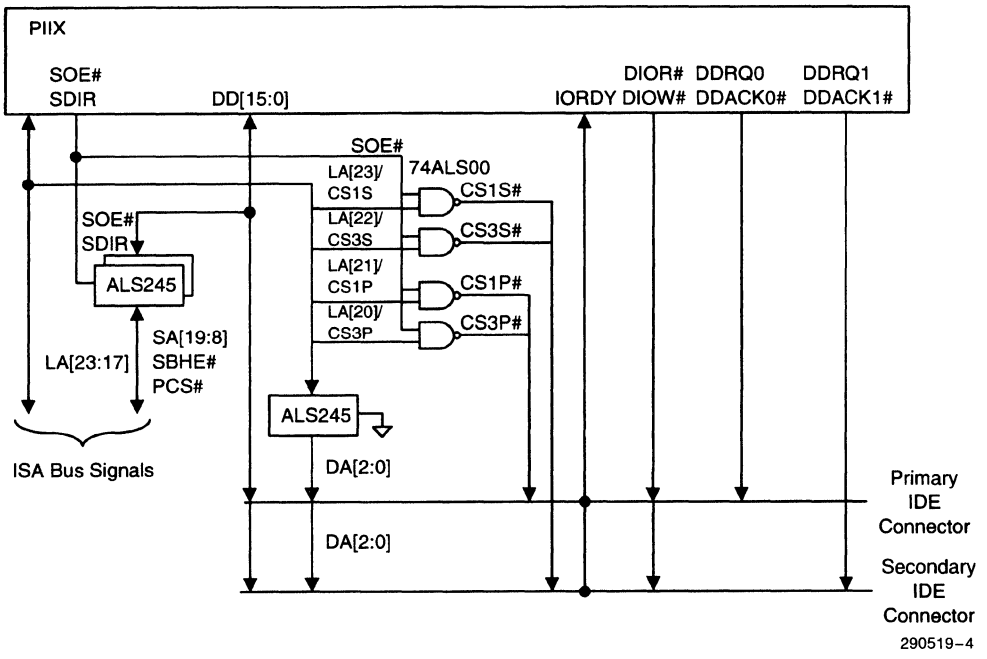
3.5 PCI Local Bus IDE

The PIIX integrates a high-performance interface from PCI to IDE. This interface is capable of accelerated PIO data transfers as well as acting as a PCI Bus Master on behalf of an IDE DMA slave device. The PIIX provides an interface for both primary and secondary IDE connectors (Figure 3).

The IDE data transfer command strobes, DMA request and grant signals, and IORDY signal interface directly to the PIIX. The IDE data lines interface directly to the PIIX, and are buffered to provide part of the ISA address bus as well as the X-Bus chip select signals. The IDE address and chip select signals are multiplexed onto the LA[23:17] lines. The IDE connector signals are driven from the LA[23:17] lines by an ALS244 buffer.

NOTE:
The IRQ14 signal (pin 83) must be used to signal interrupts for the primary channel in Bus Master mode, MIRQ[0] must be used for the secondary channel.

Only PCI masters have access to the IDE port. ISA Bus Masters cannot access the IDE I/O port addresses. Memory targeted by the IDE interface acting as a PCI Bus Master on behalf of IDE DMA slaves must reside on PCI, usually main memory implemented by the host-to-PCI bridge.



NOTES:
Support for Older Drives: There are cases where the PIIX asserts both IDE chip selects (CS1x and CS3x). Some older drives may not operate properly when both chip selects are asserted. Because the IDE chip selects are muxed with the ISA LA lines, the 74ALS00 in the figure is used to ensure proper operation of older drives by gating the LA signals with SOE#.

Figure 3. PIIX IDE Interface

3.5.1 ATA REGISTER BLOCK DECODE

The IDE ATA I/O ports are decoded by the PIIX when enabled in the PCICMD and IDETIM Registers for function 1 (ATA stands for "AT Attachment"—the specification for AT compatible drive interfaces). The actual ATA Registers are implemented in the drive itself. An access to the IDE Registers results in the assertion of the appropriate chip select for the register. The transaction is then run using compatible timing and using the IDE command strobes (DIOR#, DIOW#).

For each cable (primary and secondary), there are two I/O ranges; the command block that corresponds to the CS1x# chip select, and the control block that corresponds to the CS3x# chip select. The command block is an 8-byte range while the control block is a 4-byte range. The upper 16 bits of the I/O address are decoded as all 0s. Note that the IDE chip select signals on the PIIX must be inverted externally to provide active low signals.

Primary Command Block Offset:	01F0h
Primary Control Block Offset:	03F4h
Secondary Command Block Offset:	0170h
Secondary Control Block Offset:	0374h

Table 9 specifies the registers as they affect the PIIX hardware definition.

Table 9. IDE Legacy I/O Port Definition: COMMAND BLOCK (CS1x# Chip Select)

IO Offset	Register Function (Read/Write)	Access
00h	Data	R/W
01h	Error/Features	R/W
02h	Sector Count	R/W
03h	Sector Number	R/W
04h	Cylinder Low	R/W
05h	Cylinder High	R/W
06h	Drive/Head	R/W
07h	Status/Command	R/W

The Data Register is accessed as a 16-bit register for PIO transfers (except for ECC bytes). All other registers are accessed as 8-bit quantities.

Table 10. IDE Legacy I/O Port Definition: COMMAND BLOCK (CS3x# Chip Select)

IO Offset	Register Function (Read/Write)	Access
00h	Reserved	reserved
01h	Reserved	reserved
02h	Alt Status/Device Control	R/W
03h	Forward to ISA (Floppy)	R/W

The PIIX claims all accesses to these ranges. The byte enables do not have to be externally decoded to assert DEVSEL#. Accesses to byte 3 of the Control Block are forwarded to ISA where the floppy disk controller responds.

Each of the two drives (drive 0 or 1) on a cable implement separate ATA Register files. To determine the targeted drive, the PIIX shadows the value of bit 4 (drive bit) of byte 6 (drive/head register) of the ATA command block (CS1x#) for each of the two IDE connectors (primary and secondary).

3.5.2 ENHANCED TIMING MODES

The PIIX includes fast timing modes. The fast timing modes can be enabled only for the IDE data ports. All other transactions to the IDE Registers are run in single transaction mode with compatible timings.

Up to two IDE devices may be attached per IDE connector (drive 0 and drive 1). For each connector, only one fast timing mode may be specified (via the IDETIM Register). This mode can be applied to drive 0, drive 1, or both. Transactions targeting the other drive will use compatible timing.

3.5.2.1 Back-to-Back PIO IDE Transactions

IDE data port transaction latency consists of startup latency, cycle latency, and shutdown latency. Cycle latency consists of the I/O strobe assertion length and recovery time. Recovery time is provided so that transactions may occur back-to-back on the IDE interface (without incurring startup and shutdown latency) without violating minimum cycle periods for the IDE interface.

Startup latency is incurred when a PCI master cycle targeting the IDE data port is decoded and the DA[2:0] and CSxx# lines are not set up. Startup latency provides the setup time for the DA[2:0] and CSxx# lines with respect to the read and write strobes (DIOR# and DIOW#).

Shutdown latency is incurred after outstanding scheduled IDE data port transactions (either a non-empty write post buffer or an outstanding read prefetch cycles) have completed and before other transactions can proceed. It provides hold time on the DA[2:0] and CSxx# lines with respect to the read and write strobes (DIOR# and DIOW#).

Cycle latency is the latency incurred by each individual 16 bit IDE data port transfer, and consists of command strobe width and recovery time. The command strobe assertion width is selected by the IDETIM Register and may be set to 2, 3, 4, or 5 PCI clocks. The recovery time is selected by the IDETIM Register and may be set to 1, 2, 3, or 4 PCI clocks.

If IORDY is asserted when the initial sample point is reached, no wait states are added. If IORDY is negated when the initial sample point is reached, additional wait states are added. Since the rising edge of IORDY must be synchronized, at least two additional PCI clocks are added.

NOTE:
Bit 2 (16 bit I/O recovery enable) of the ISA Controller Recovery Timer Register does not add wait states to IDE data port read accesses when any of the fast timing modes are enabled.

3.5.2.2 IORDY Masking

The IORDY signal can be forced asserted on a drive-by-drive basis via the IDETIM Register.

3.5.2.3 PIO 32 Bit IDE Data Port Mode

If the 32-bit IDE data port mode is enabled (via bit 4 and 0 of the IDETIM Register), 32-bit accesses to the IDE data port address (default 01F0h primary, etc.) result in two back-to-back 16-bit transactions to IDE. The 32-bit data port feature is enabled for all timings, not just enhanced timing.

3.5.3 BUS MASTER FUNCTION

The PIIX can act as a PCI Bus Master on behalf of an IDE slave device. Two PCI Bus Master channels are providedone channel for each IDE connector (primary and secondary). By performing the IDE data transfer as a PCI Bus Master, the PIIX off-loads the CPU and improves system performance in multitasking environments.

NOTE:
The PCICMD Command Register (Function 1) bit 2 must be programmed to 1 for Bus Master operation.

Physical Region Descriptor Format

The physical memory region to be transferred is described by a Physical Region Descriptor (PRD). The PRDs are stored in a table in memory. The data transfer proceeds until all regions described by the PRDs in the table have been transferred. Note that the Bus Master IDE does not support memory for regions or PRDs on ISA.

Each PRD entry is 8 bytes in length. The first 4 bytes specify the byte address of a physical memory region. The next 2 bytes specify the count of the region in bytes (64-Kbyte limit per region). A value of 0 in these 2 bytes indicates 64 Kbytes. Bit 7 of the last byte indicates the end of the table (EOT). Bus Master operation terminates when the last descriptor has been completed.

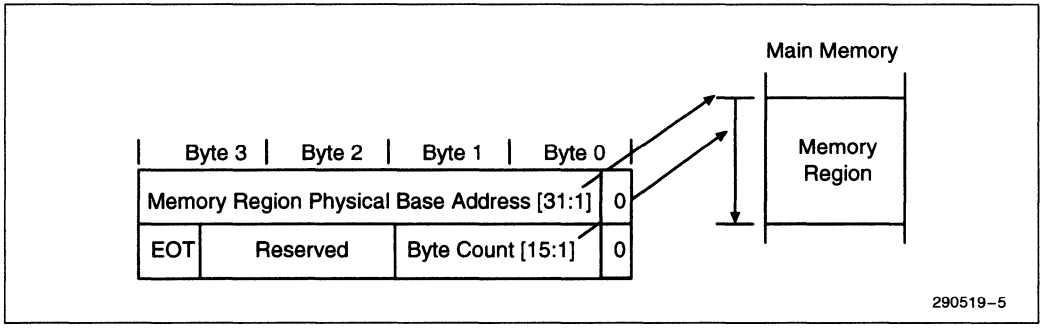


Figure 4. Physical Region Descriptor Table Entry

NOTE:
The memory region specified by the descriptor cannot straddle a 64-Kbyte boundary. This means that the byte count can be limited to 64 Kbytes and the incrementer for the current address register need only extend from bit 1 to bit 15. Also, the total sum of the descriptor byte counts must be equal to or greater than the size of the disk transfer request. If greater than the disk transfer request, the driver must terminate the Bus Master transaction (by setting bit 0 in the Bus Master IDE Command Register to 0) when the drive issues an interrupt to signal transfer completion.

Operation

To initiate a Bus Master transfer between memory and an IDE DMA slave device, the following steps are required:

1. Software prepares a PRD Table in main memory. Each PRD is 8 bytes long and consists of an address pointer to the starting address and the transfer count of the memory buffer to be transferred. In any given PRD Table, two consecutive PRDs are offset by 8-bytes and are aligned on a 4-byte boundary.
2. Software provides the starting address of the PRD Table by loading the PRD Table Pointer Register . The direction of the data transfer is specified by setting the Read/Write Control bit. Clear the Interrupt bit and Error bit in the Status Register.
3. Software issues the appropriate DMA transfer command to the disk device.
4. Engage the Bus Master function by writing 1 to the Start bit in the Bus Master IDE Command Register for the appropriate channel. The first entry in the PRD table is fetched by the PIIX. The channel remains masked until the first descriptor is loaded.
5. The controller transfers data to/from memory responding to DMA requests from the IDE device.
6. At the end of the transfer, the IDE device signals an interrupt.
7. In response to the interrupt, software resets the Start/Stop bit in the command register. It then reads the controller status and then the drive status to determine if the transfer completed successfully.

When the last data transfer for a region has been completed on the IDE interface, the next descriptor is fetched from the table. The descriptor contents are loaded into the Current Base and Current Count Registers.

The last PRD in a table has the End of List (EOL) bit set. The PCI Bus Master data transfers terminates when the physical region described by the last PRD in the table has been completely transferred. The active bit in the BMISx Register is set to 0 and the DDRQx signal is masked.

NOTE:
The IRQ14 signal (pin 83) must be used to signal interrupts for the primary channel in Bus Master mode, MIRQ[0] must be used for the secondary channel.

Line Buffer

A single line buffer exists for the PCI Bus Master IDE interface. This buffer is not shared with any other function. The buffer is maintained in either the read state or the write state. The size of the buffer is 32 bytes, and is aligned on the cache line boundary. The line buffer allows burst data transfers to proceed at peak transfer rates.

Arbitration

The two Bus Master IDE channels are fairly arbitrated (round robin between the two). The ISA DMA channels and the IDE Bus Master channels are arbitrated fairly as a group (fairness between the two groups). This arbitration is not programmable.

3.6 Interval Timer

The PIIX contains three counters that are equivalent to those found in the 82C54 programmable interval timer. The three counters are contained in one PIIX timer unit, referred to as Timer-1. Each counter output provides a key system function. Counter 0 is connected to interrupt controller IRQ0 and provides a system timer interrupt for a time-of-day, diskette time-out, or other system timing functions. Counter 1 generates a refresh request signal and Counter 2 generates the tone for the speaker. The 14.31818 MHz counters normally use OSC as a clock source.

Counter 0, System Timer

This counter functions as the system timer by controlling the state of IRQ0 and is typically programmed for mode 3 operation. The counter produces a square wave with a period equal to the product of the counter period (838 ns) and the initial count value. The counter loads the initial count value one counter period after software writes the count value to the counter I/O address. The counter initially asserts IRQ0 and decrements the count value by two each counter period. The counter negates IRQ0 when the count value reaches 0. It then reloads the initial count value and again decrements the initial count value by two each counter period. The counter then asserts IRQ0 when the count value reaches 0, reloads the initial count value, and repeats the cycle, alternately asserting and negating IRQ0.

Counter 1, Refresh Request Signal

This counter provides the refresh request signal and is typically programmed for mode 2 operation. The counter negates refresh request for one counter period (838 ns) during each count cycle. The initial count value is loaded one counter period after being written to the counter I/O address. The counter initially asserts refresh request, and negates it for 1 counter period when the count value reaches 1. The counter then asserts refresh request and continues counting from the initial count value.

Counter 2, Speaker Tone

This counter provides the speaker tone and is typically programmed for mode 3 operation. The counter provides a speaker frequency equal to the counter clock frequency (1.193 MHz) divided by the initial count value. The speaker must be enabled by a write to port 061h.

3.7 Interrupt Controller

The PIIX provides an ISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The two controllers are cascaded so that 13 external and three internal interrupts are possible. The master interrupt controller provides IRQ [7:0] and the slave interrupt controller provides IRQ[15:8] (Figure 5). The three internal interrupts are used for internal functions only and are not available to the user. IRQ2 is used to cascade the two controllers together. IRQ0 is used as a system timer interrupt and is tied to Interval Timer 1, Counter 0. IRQ13 is connected internally to FERR#. The remaining 13 interrupt lines (IRQ[15,14,12:3,1]) are available for external system interrupts. Edge or level sense selection is programmable on an individual channel by channel basis.

The Interrupt unit also supports interrupt steering. The PIIX can be programmed to allow the four PCI active low interrupts (PIRQ[3:0]#) to be internally routed to one of 11 interrupts (IRQ[15,14,12:9,7:3]). In addition, up to six interrupt signals dedicated to motherboard devices (MIRQ[5:0]) may be routed to any of the 11 interrupts.

The Interrupt Controller consists of two separate 82C59 cores. Interrupt Controller 1 (CNTRL-1) and Interrupt Controller 2 (CNTRL-2) are initialized separately and can be programmed to operate in different modes. The default settings are: 80x86 Mode, Edge Sensitive (IRQ[15:0]) Detection, Normal EOI, Non-Buffered Mode, Special Fully Nested Mode disabled, and Cascade Mode. CNTRL-1 is connected as the Master Interrupt Controller and CNTRL-2 is connected as the Slave Interrupt Controller.

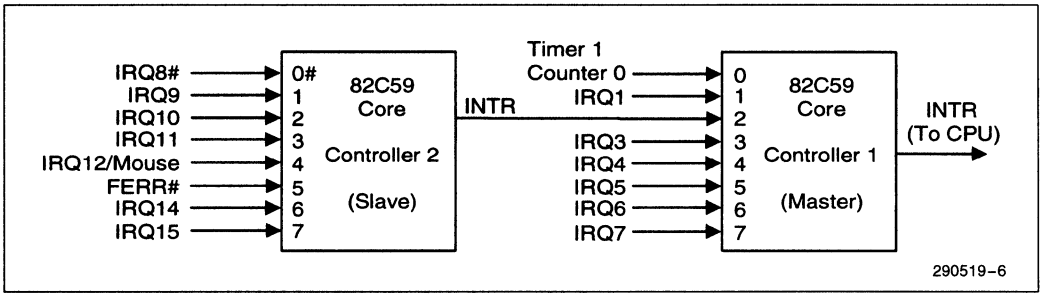


Figure 5. Block Diagram of the Interrupt Controller

Note that IRQ13 is generated internally (as part of the coprocessor error support) by the PIIX. IRQ12/M is generated internally (as part of the mouse support) when bit-4 in the XBCS Register is set to 1. When this bit is set to 0, the standard IRQ12 function is provided and IRQ12 appears externally.

3.7.1 PROGRAMMING THE ICWs/OCWs

The Interrupt Controller accepts two types of command words generated by the CPU or Bus Master:

1. **Initialization Command Words (ICWs)**: Before normal operation can begin, each Interrupt Controller in the system must be initialized. In the 82C59, this is a two to four byte sequence. However, for the PIIX, each controller must be initialized with a four byte sequence. This four byte sequence is required to configure the interrupt controller correctly for the PIIX implementation. This implementation is ISA-compatible. The four initialization command words are referred to by their acronyms: ICW1, ICW2, ICW3, and ICW4. The base address for each interrupt controller is a fixed location in the I/O memory space, at 0020h for CNTRL-1 and at 00A0h for CNTRL-2.

   An I/O write to the CNTRL-1 or CNTRL-2 base address with data bit 4 equal to 1 is interpreted as ICW1. For PIIX-based ISA systems, three I/O writes to "base address + 1" (021h for CNTRL-1 and 0A0h for CNTRL-2) must follow the ICW1. The first write to "base address + 1" (021h/0A0h) performs ICW2, the second write performs ICW3, and the third write performs ICW4.
2. **Operation Command Words (OCWs)**: These are the command words that dynamically reprogram the interrupt controller to operate in various interrupt modes. Any interrupt lines can be masked by writing an OCW1. A 1 written in any bit of this command word masks incoming interrupt requests on the corresponding IRQx line. OCW2 is used to control the rotation of interrupt priorities when operating in the rotating priority mode and to control the End of Interrupt (EOI) function of the controller. OCW3 set up reads of the ISR and IRR, enable/disables the Special Mask Mode (SMM), and sets up the interrupt controller in polled interrupt mode. The OCWs can be written to the Interrupt Controller any time after initialization.

3.7.2 EDGE AND LEVEL TRIGGERED MODE

In ISA systems this mode is programmed using bit 3 in ICW1. With PIIX this bit is disabled and a new register for edge- and level-triggered mode selection (per interrupt input) is included. This is the Edge/Level control Registers ELCR1 and ELCR2. The default programming is equivalent to programming the LTIM bit (ICW1 bit 3) to 0 (all interrupts selected for edge triggered mode). Note that IRQ0, 1, 2, 8#, and 13 can not be programmed for level-sensitive mode and can not be modified by software.

If an ELCR bit = 0, an interrupt request is recognized by a low-to-high transition on the corresponding IRQx input. The IRQ input can remain high without generating another interrupt.

If an ELCR bit = 1, an interrupt request is recognized by a low level on the corresponding IRQ input and there is no need for an edge detection. The interrupt request must be removed before the EOI command is issued to prevent a second interrupt from occurring.

In both the edge and level triggered modes, the IRQ inputs must remain active until after the falling edge of the first INTERRUPT ACKNOWLEDGE CYCLE. If the IRQ input goes inactive before this time, a default IRQ7 occurs when the CPU acknowledges the interrupt. This can be a useful safeguard for detecting interrupts caused by spurious noise glitches on the IRQ inputs. To implement this feature, the IRQ7 routine is used for "clean up" simply executing a return instruction, thus ignoring the interrupt. If IRQ7 is needed for other purposes, a default IRQ7 can still be detected by reading the ISR. A normal IRQ7 interrupt sets the corresponding ISR bit. A default IRQ7 does not set this bit. However, if a default IRQ7 routine occurs during a normal IRQ7 routine, the ISR remains set. In this case, it is necessary to keep track of whether or not the IRQ7 routine was previously entered. If another IRQ7 occurs, it is a default.

3.7.3 INTERRUPT STEERING

The PIIX can be programmed to allow four PCI programmable interrupts (PIRQ[3:0]#) to be internally routed to one of 11 interrupts (IRQ[15,14,12:9,7:3]) using the PIRQx Route Control Register. PCLK is used to synchronize the PIRQx# inputs. The assignment is programmable through the PIRQx Route Control Registers. One or more PIRQx# lines can be routed to the same IRQx input. If interrupt steering is not required, the Route Registers can be programmed to disable steering.

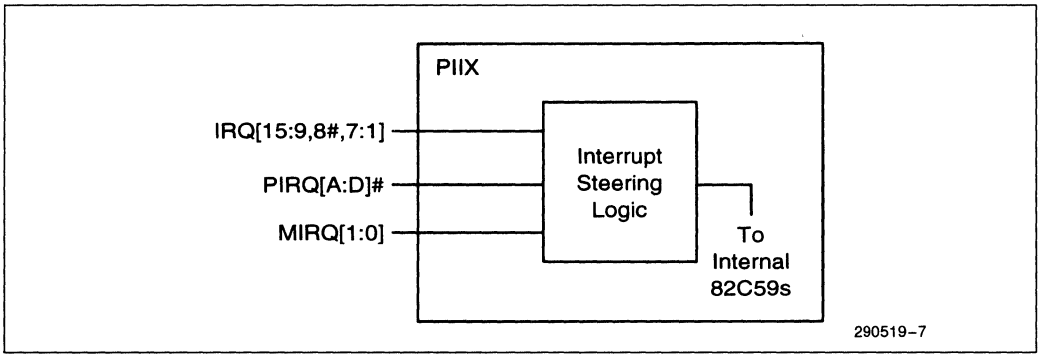


Figure 6. Interrupt Steering

The PIRQx# lines are defined as active low, level-sensitive to allow multiple interrupts on a PCI Board to share a single line across the connector. When a PIRQx# is routed to a specified IRQ line, the software must change the IRQ's corresponding ELCR bit to level-sensitive mode. Note that this means that the selected IRQ can no longer be used by an ISA device, unless that ISA device can respond as an active low level sensitive interrupt.

The PIIX also supports up to two programmable interrupts (MIRQ[1:0]; intended for use with motherboard devices) to be routed to one of the 11 interrupts (IRQ[15,14,12:9,7:3]) using the MBIRQx Route Control Register. The routing is accomplished in the same manner as for the PIRQx# inputs, except that the interrupts are active high. Two MIRQx lines may be routed to the same IRQx input. If interrupt steering is not required, the MBIRQx Registers can be programmed to disable routing.

When more than one MIRQ line is routed to an IRQ input, the software must change the IRQ's corresponding ELCR bit to level sensitive mode. Interrupt sharing for motherboard devices must be evaluated for the particular combination of devices under consideration. The IRQ selected bit MBIRQx[3:0] can no longer be used by an ISA device, unless that ISA device can respond as an active high level sensitive interrupt.

3.8 X-Bus Peripheral Support

The PIIX provides positive decode (chip selects) and X-Bus buffer control (XDIR# and XOE#) for a real time clock, keyboard controller and BIOS for PCI and ISA initiated cycles. The PIIX also generates RTCALE (address latch enable) for the RTC. The chip selects are generated combinatorially from the ISA SA(16:0) and LA (23:17) address lines (it is assumed that ISA masters drive SA(19:16) and LA(23:17) low when accessing I/O devices). The PIIX also provides PS/2 mouse support via the IRQ12/M signal and coprocessor functions (FEER# and IGNNE#). The chip selects and X-Bus buffer control lines can be enabled/disabled via the XBCS Register.

Coprocessor Error Function

This function provides coprocessor error support for the CPU and is enabled via the XBCS Register. FERR# is tied directly to the coprocessor error signal of the CPU. If FERR# is driven active to the PIIX, an internal IRQ13 is generated and the INTR output from the PIIX is driven active. When a write to I/O location F0h is detected, the PIIX negates IRQ13 (internal to the PIIX) and drives IGNNE# active. IGNNE# remains active until FERR# is driven inactive. IGNNE# is not driven active unless FERR# is active.

Mouse Function

When the mouse interrupt function is enabled (via the XBCS Register), the mouse interrupt function is provided on the IRQ12/M input signal. In this mode, a mouse interrupt generates an interrupt through IRQ12 to the Host CPU. The PIIX informs the CPU of this interrupt via a INTR. A read of 60h releases IRQ12. If the mouse interrupt function is disabled, a read of address 60h has no effect on IRQ12/M. Reads and writes to this register flow through to the ISA Bus. For additional information, see the IRQ12/M description in the Signal Description.

3.9 Power Management

The PIIX has extensive power management capability permitting a system to operate in a low power state without being powered down. In a typical desktop personal computer there are two statesPower-On and Power-Off. Leaving a system powered on when not in use wastes power. The PIIX provides a Fast-On/Off feature that creates a third state called Fast-Off (Figure 7). When in the Fast-Off state, the system consumes less power than the Power-On state.

The PIIX's power management architecture is based on three functions—System Management Mode (SMM), Clock Control, and Advanced Power Management (APM). Software (called SMM code) controls the transitions between the Power-On state and the Fast-Off state. The PIIX invokes this software by generating an SMI to the CPU (asserting the SMI# signal). A variety of programmable events are provided that can generate an SMI. The SMM code places the system in either the Power-On state or the Fast-Off state.

A Fast-On event is an event that instructs the computer (via an SMI to the CPU) to enter the Power-On state in anticipation of system activity by the user. Fast-On events are programmable and include moving the mouse, pressing a key on the keyboard, an external hardware event, an incoming call to a system FAX/Modem, a RTC alarm, or the operating system.

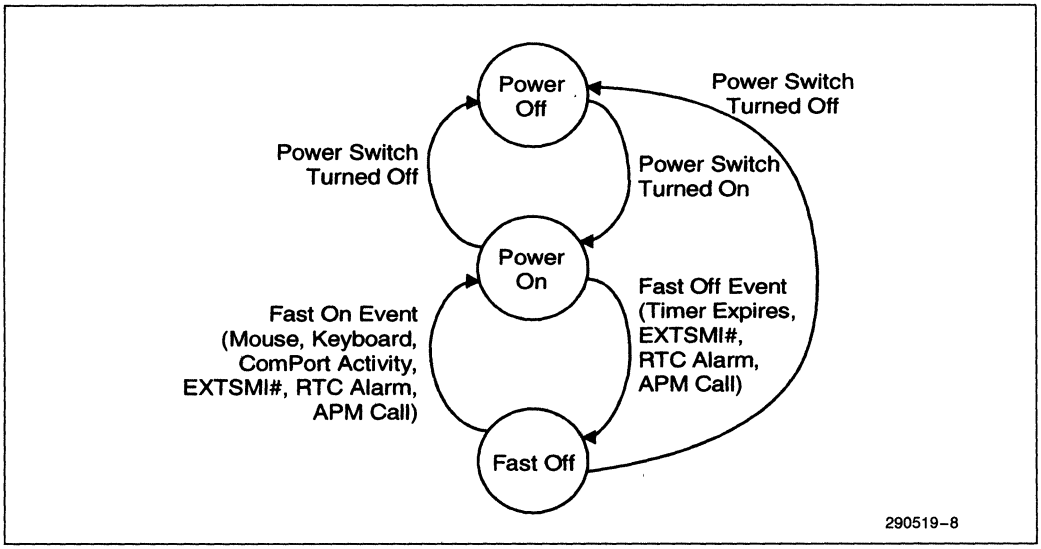


Figure 7. Fast-On/Off Flow

3.9.1 SMM MODE

SMM mode is invoked by asserting the SMI# signal to the CPU. The PIIX provides a variety of programmable events that can generate an SMI. When the CPU receives an SMI, it enters SMM mode and executes SMM code out of SMRAM. Depending on the current state, the SMM code places the system in either the Power-On state or the Fast-Off state. In the Power-On state, the computer system operates normally. In this state, one of the four programmable events listed below can trigger an SMI.

1. A global idle timer called the Fast-Off Timer expires (an indication that the end user has not used the computer for a programmed period of time).
2. The EXTSMI# pin is asserted.
3. The operating system issues an APM call.

3.9.2 SMI SOURCES

The SMI# signal can be asserted by hardware interrupt events, the Fast-Off Timer, an external SMI event (EXTSMI#), and software events (via the APMC and APMS Registers). Enable/disable bits (in the SMIEN Register) permit each event to be individually masked from generating an SMI. In addition, the SMI# signal can be globally enabled/disabled in the SMICNTL Register. Status of the individual events causing an SMI is provided in the SMIREQ Register. For detailed information on the SMI control/status registers, refer to the Register Description section.

Hardware Interrupt Events

Hardware events (IRQ[12,8#,4,3,1] and the Fast-Off Timer) are enabled/disabled from generating an SMI in the SMIEN Register. When enabled, the occurrence of the corresponding hardware event generates an SMI (asserts the SMI# signal), regardless of the current power state of the system.

Fast-Off Timer

The Fast-Off Timer is used to indicate (through an SMI) that the system has been idle for a programmed period of time. The timer counts down from a programmed start value and when the count reaches 00h, can generate an SMI. The timer decrement rate is programmable (via the SMICNTL Register) and is re-loaded each time a system event occurs. This counter should not be programmed to 00h. System and break events are described in the SEE Register.

EXTSMI#

The EXTSMI# input pin provides the system designer the capability to invoke SMM with external hardware. For example, the EXTSMI# input could be connected to a "green button" permitting the user to enter the Fast-Off state by depressing a button. The EXTSMI# generation of an SMI is enabled/disabled in the SMIEN Register.

Software Events

Software events (accessing the APMx Registers) indicate that the OS is passing power management information to the SMI handler. There are two Advanced Power Management (APM) Registers—APM Control (APMC) and APM Status (APMS) Registers. These registers permit software to generate an SMI; by writing to the APMC Register. For example, the APMC Register can be used to pass an APM command between APM OS and BIOS and the APMS Register could be used to pass data between the OS and the SMI handler.

The two APM Registers are located in normal I/O space. The PIIX subtractively decodes PCI accesses to these registers and forwards the accesses to the ISA Bus. The APM Registers are not accessible by ISA masters. Note that the remaining power management registers are located in PCI Configuration space.

3.9.3 CLOCK CONTROL

The CPU can be put in a low power state by asserting the STPCLK# signal. STPCLK# is an interrupt to the CPU. However, for this type of interrupt, the CPU does not generate an interrupt acknowledge cycle. Once the STPCLK# interrupt is executed, the CPU enters the stop-grant state. In this state, the CPU's internal clocks are disabled and instruction execution is stopped. The stop-grant state is exited when the STPCLK# signal is negated.

Software can assert STPCLK# (if enabled via the SMICNTL Register) by a read of the APMC Register. Note that STPCLK# can also be periodically asserted by using clock scaling as described below.

The PIIX automatically negates STPCLK# when a break event occurs (if enabled in the SEE Register). Software can negate STPCLK# by disabling STPCLK# in the SMICNTL Register or by a write to the APMC Register.

Clock Scaling (Emulating Clock Division)

Clock scaling permits the PIIX to periodically place the CPU in a low power state. This emulates clock division. When clock scaling is enabled, the CPU runs at full frequency for a pre-defined time period and then is stopped for a pre-defined time period. The run/stop time interval ratio emulates the clock division effect from a power/performance point of view. However, clock scaling is more effective than dividing the CPU frequency. For example, if the CPU is in the stop grant state and a break event occurs, the CPU clock returns to full frequency. In addition, there is no recovery time latency to start the clock.

Two programmable 8-bit clock scale timer control registers set the STPCLK# high (negate) and low (assert) times—the CTLTMRH and CTLTMRL Registers. The timer is clocked by a 32-μsec internal clock. This allows a programmable timer interval for both the STPCLK# high and low times of 0–8 msec.

3.10 Reset Support

The PIIX integrates the system reset logic for the system. The PIIX generates CPURST, PCIRST#, and RSTDRV during power up (PWROK) and when a hard reset is initiated through the RC Register. The following PIIX signals interface directly to the processor: CPURST, INTR, NMI, IGNNE#, SMI#, and STPCLK#.

These signals are open drain. Thus, external logic is not required for interfacing with the processors based on 3.3V technology which do no support 5V tolerant input buffers. During power-up these signals are driven low to prevent problems associated with 5V/3.3V power sequencing.

Some PCI devices may drive 3.3V friendly signals directly to 3.3V devices that are not 5V tolerant. If such signals are powered from the 5V supply, they must be driven low when PCIRST# is asserted. Some of these signals may need to be driven high before CPURST is negated. PCIRST# is negated 1 ms to 2 ms before CPURST to allow time for this to occur.

3.10.1 HARDWARE STRAPPING OPTIONS

There are two hardware strapping options on the PIIX. The SYSCLK signal is used during a hard reset to select the ISA clock divisor (sampled high for divisor of 3 MHz–25 MHz PCI operation; sampled low for a divisor of 4 MHz–33 MHz or 30 MHz PCI operation.

4.0 PINOUT AND PACKAGE INFORMATION

4.1 82371FB (PIIX) Pinout

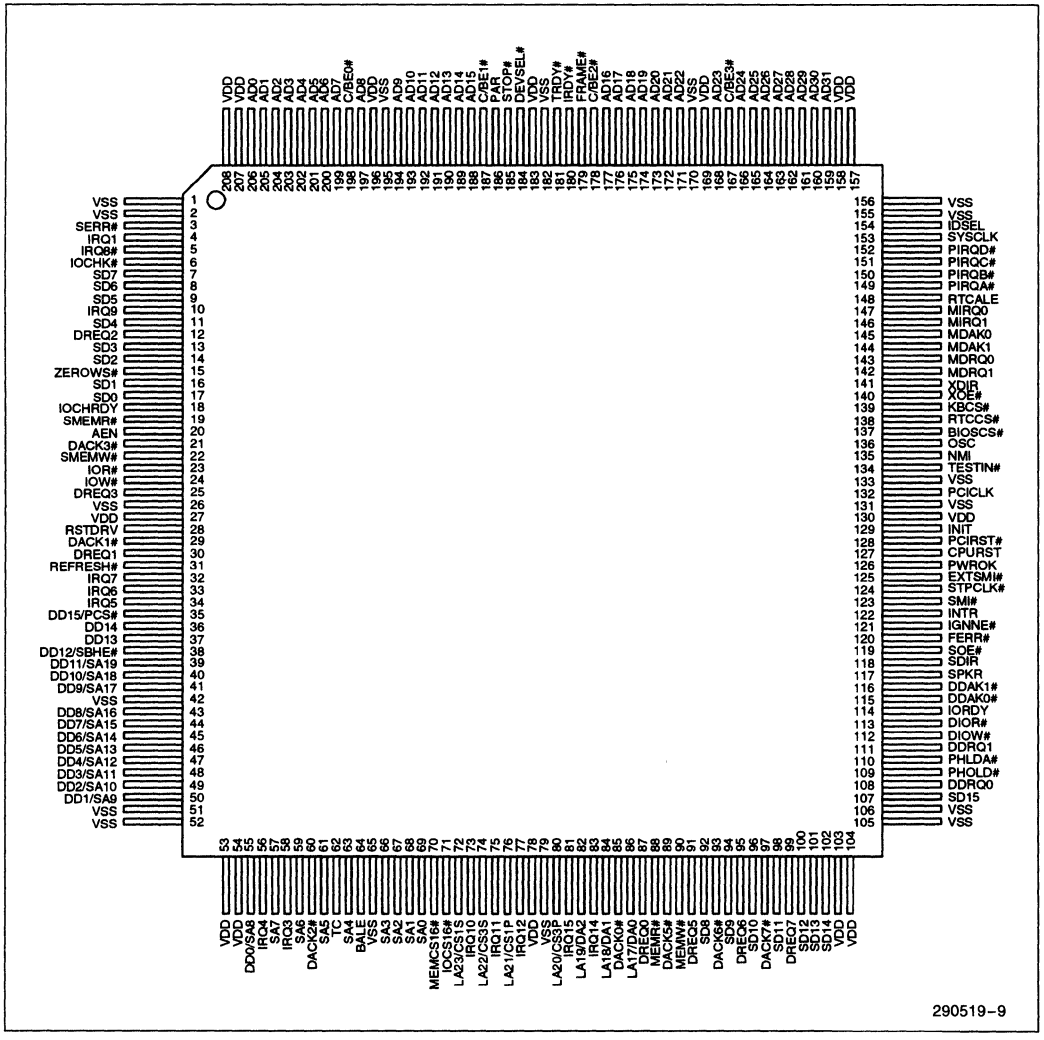


Figure 8. 82371FB (PIIX) Pinout

Table 11. Alphabetical Pin Assignment

Name	Pin#	Type
AD0	206	I/O
AD1	205	I/O
AD2	204	I/O
AD4	202	I/O
AD5	201	I/O
AD6	200	I/O
AD7	199	I/O
AD8	197	I/O
AD9	194	I/O
AD10	193	I/O
AD11	192	I/O
AD12	191	I/O
AD13	190	I/O
AD14	189	I/O
AD15	188	I/O
AD16	177	I/O
AD17	176	I/O
AD18	175	I/O
AD19	174	I/O
AD20	173	I/O
AD21	172	I/O
AD22	171	I/O
AD23	168	I/O
AD24	166	I/O
AD25	165	I/O
AD26	164	I/O
AD27	163	I/O
AD28	162	I/O
AD29	161	I/O
AD3	203	I/O

Name	Pin#	Type
AD30	160	I/O
AD31	159	I/O
AEN	20	O
BALE	64	O
BIOSCS#	137	O
C/BE0#	198	I/O
C/BE1#	187	I/O
C/BE2#	178	I/O
C/BE3#	167	I/O
CPURST	127	O
DACK0#	85	O
DACK1#	29	O
DACK2#	60	O
DACK3#	21	O
DACK5#	89	O
DACK6#	93	O
DACK7#	97	O
DD0/SA8	55	I/O
DD1/SA9	50	I/O
DD2/SA10	49	I/O
DD3/SA11	48	I/O
DD4/SA12	47	I/O
DD5/SA13	46	I/O
DD6/SA14	45	I/O
DD7/SA15	44	I/O
DD8/SA16	43	I/O
DD9/SA17	41	I/O
DD10/SA18	40	I/O
DD11/SA19	39	I/O
DD12/SBHE#	38	I/O

Name	Pin#	Type
DD13	37	I/O
DD14	36	I/O
DD15/PCS#	35	I/O
DDAK0#	115	O
DDAK1#	116	O
DDRQ0	108	I
DDRQ1	111	I
DEVSEL#	184	I/O
DIOR#	113	O
DIOW#	112	O
DREQ0	87	I
DREQ1	30	I
DREQ2	12	I
DREQ3	25	I
DREQ5	91	I
DREQ6	95	I
DREQ7	99	I
EXTSMI#	125	I
FERR#	120	I
FRAME#	179	I/O
IDSEL	154	I
IGNNE#	121	O
INIT	129	O
INTR	122	O
IOCHK#	6	I
IOCHRDY	18	I/O
IOCS16#	71	I
IOR#	23	I/O
IORDY	114	I
IOW#	24	I/O

Table 11. Alphabetical Pin Assignment (Continued)

Name	Pin#	Type
IRDY#	180	I/O
IRQ1	4	I
IRQ3	58	I
IRQ4	56	I
IRQ5	34	I
IRQ6	33	I
IRQ7	32	I
IRQ8#	5	I
IRQ9	10	I
IRQ10	73	I
IRQ11	75	I
IRQ12	77	I
IRQ14	83	I
IRQ15	81	I
KBCS#	139	O
LA17/DA0	86	I/O
LA18/DA1	84	I/O
LA19/DA2	82	I/O
LA20/CS3P	80	I/O
LA21/CS1P	76	I/O
LA22/CS3S	74	I/O
LA23/CS1S	72	I/O
MDAK1	144	O
MDAK0	145	O
MDRQ0	143	I
MDRQ1	142	I
MEMCS16#	70	I/O
MEMR#	88	I/O
MEMW#	90	I/O
MIRQ0	147	O
MIRQ1	146	I

Name	Pin#	Type
NMI	135	O
OSC	136	I
PAR	186	O
PCICLK	132	I
PCIRST#	128	O
PHLDA#	110	I
PHOLD#	109	O
PIRQA#	149	I
PIRQB#	150	I
PIRQC#	151	I
PIRQD#	152	I
PWROK	126	I
REFRESH#	31	I/O
RSTDRV	28	O
RTCALE	148	O
RTCCS#	138	O
SA0	69	I/O
SA1	68	I/O
SA2	67	I/O
SA3	66	I/O
SA4	63	I/O
SA5	61	I/O
SA6	59	I/O
SA7	57	I/O
SD0	17	I/O
SD1	16	I/O
SD2	14	I/O
SD3	13	I/O
SD4	11	I/O
SD5	9	I/O
SD6	8	I/O

Name	Pin#	Type
SD7	7	I/O
SD8	92	I/O
SD9	94	I/O
SD10	96	I/O
SD11	98	I/O
SD12	100	I/O
SD13	101	I/O
SD14	102	I/O
SD15	107	I/O
SDIR	118	O
SERR#	3	I
SMEMR#	19	O
SMEMW#	22	O
SMI#	123	O
SOE#	119	O
SPKR	117	O
STOP#	185	I/O
STPCLK#	124	O
SYSCLK	153	O
TC	62	O
TESTIN#	134	I
TRDY#	181	I/O
VDD	27	V
VDD	53	V
VDD	54	V
VDD	78	V
VDD	103	V
VDD	104	V
VDD	130	V
VDD	157	V
VDD	158	V

Table 11. Alphabetical Pin Assignment (Continued)

Name	Pin#	Type
VDD	169	V
VDD	183	V
VDD	196	V
VDD	207	V
VDD	208	V
VSS	1	V
VSS	2	V
VSS	26	V
VSS	42	V

Name	Pin#	Type
VSS	51	V
VSS	52	V
VSS	65	V
VSS	79	V
VSS	105	V
VSS	106	V
VSS	131	V
VSS	133	V
VSS	155	V

Name	Pin#	Type
VSS	156	V
VSS	170	V
VSS	182	V
VSS	195	V
XDIR	141	I
XOE#	140	O
ZEROWS#	15	I

5.0 PACKAGE DIMENSIONS

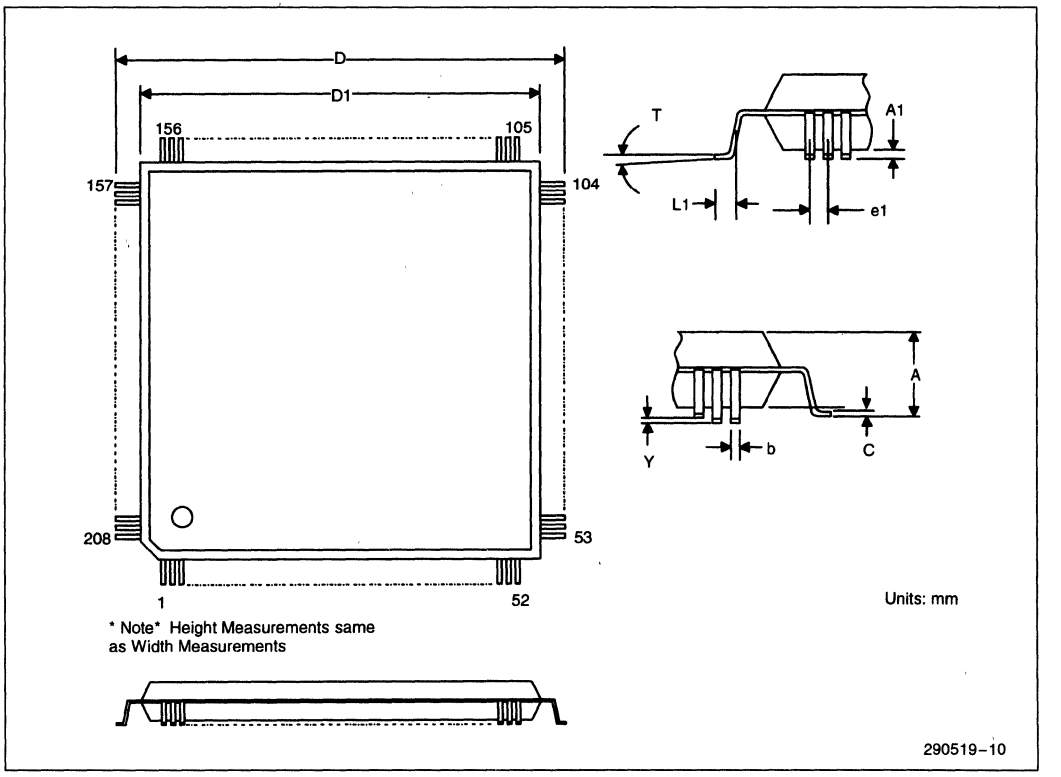


Figure 9. 208-Pin Quad Flat Pack (QFP) Dimensions

Table 12. 208-Pin Quad Flat Pack (QFP) Dimensions

Symbol	Description	Value (mm)
A	Seating Height	4.25 (max)
A1	Stand-off	0.05 (min); 0.40 (max)
b	Lead Width	0.2 ± 0.10
C	Lead Thickness	0.15 +0.1/-0.05
D	Package Length and Width, including pins	30.6 ± 0.4
D1	Package Length and Width, excluding pins	28 ± 0.2
e1	Linear Lead Pitch	0.5 ± 0.1
Y	Lead Coplanarity	0.08 (max)
L1	Foot Length	0.5 ± 0.2
T	Lead Angle	0° - 10°

6.0 82371FB TESTABILITY

6.1 Test Mode Description

The test modes are decoded from the IRQ inputs (IRQ 7, 6, 5) and qualified with the TESTIN# pin. Test mode selection is asynchronous. These signals need to remain in their respective state for the duration of the test modes. The test modes are defined as follows.

Test Mode	IRQ7	IRQ6	IRQ5	TESTIN#
NAND Tree	0	x	x	0
NAND Tree	x	x	0	0
Tri-state All Outputs	1	1	1	0

6.2 NAND Tree Mode

Tri-states all outputs and bi-directional buffers except for XDIR and DACK1#. Every output buffer except for XDIR and DACK1# is configured as an input in NAND tree mode and included in the NAND chain. The first input of the NAND chain is MDRQ1, and the NAND chain is routed counter-clockwise around the chip (e.g., MDRQ1, MDRQ0, MDAK1#, . . .). DACK1# is an intermediate output, and XDIR is the final output. PCICLK and TESTIN# are the only input pins not included in the NAND chain. Note in the table above there are two possible ways to select NAND tree test mode.

To perform a NAND tree test, all pins included in the NAND tree should be driven to 1 except for the following pins, which use inverting Schmitt trigger inputs and should be driven to 0:

Pin #	Pin Name
4	IRQ1
5	IRQ8#
6	IOCHK#
10	IRQ9
15	ZEROWS#
32	IRQ7
33	IRQ6
34	IRQ5
56	IRQ4
58	IRQ3
73	IRQ10
75	IRQ11
77	IRQ12
81	IRQ15
83	IRQ14
126	PWROK

Beginning with MDRQ and working counter-clockwise around the chip, each pin can be toggled and a resulting toggle observed on DACK1# or XDIR. The DACK1# output is provided so that the NAND tree test can be divided into two sections.

Table 13. NAND Tree

Pin #	Pin Name	Notes
134	TESTIN#	TESTIN# should be driven to 0 for the duration of the NAND tree test.
32	IRQ7	Test mode select signal.
33	IRQ6	Test mode select signal.
34	IRQ5	Test mode select signal.
142	MDRQ1	
143	MDRQ0	
144	MDK1	
145	MDK0	
146	MIRQ1	
147	MIRQ0	
148	RTCALE	
149	PIRQA#	
150	PIRQB#	
151	PIRQC#	
152	PIRQD#	
153	SYSCLK	
154	IDSEL	
159	AD31	
160	AD30	
161	AD29	
162	AD28	
163	AD27	
164	AD26	
165	AD25	
166	AD24	
167	C/BE3#	
168	AD23	
171	AD22	
172	AD21	
173	AD20	
174	AD19	
175	AD18	
176	AD17	
177	AD16	
178	C/BE2#	
179	FRAME#	
180	IRDY#	
181	TRDY#	
184	DEVSEL#	
185	STOP#	
186	PAR	
187	C/BE1#	
188	AD15	
189	AD14	
190	AD13	
191	AD12	
192	AD11	
193	AD10	
194	AD9	
197	AD8	
198	C/BE0#	
199	AD7	
200	AD6	
201	AD5	
202	AD4	
203	AD3	
204	AD2	
205	AD1	

Table 13. NAND Tree (Continued)

Pin #	Pin Name	Notes
206	AD0	
3	SERR#	
4	IRQ1	Inverted input signal
5	IRQ8#	Inverted input signal
6	IOCHK#	Inverted input signal
10	IRQ9	Inverted input signal
11	SD4	
12	DREQ2	
13	SD3	
14	SD2	
15	ZEROWS#	Inverted input signal
16	SD1	
17	SD0	
18	IOCHRDY	
19	SMEMR#	
20	AEN	
21	DACK3#	
22	SMEMW#	
23	IOR#	
24	IOW#	
25	DREQ3	
28	RSTDRV	
29	DACK1#	Intermediate NAND-tree output.
30	DREQ1	
31	REFRESH#	
32	IRQ7	Inverted input signal
33	IRQ6	Inverted input signal
34	IRQ5	Inverted input signal
35	DD15	
36	DD14	

Pin #	Pin Name	Notes
37	DD13	
38	DD12	
39	DD11	
40	DD10	
41	DD9	
43	DD8	
44	DD7	
45	DD6	
46	DD5	
47	DD4	
48	DD3	
49	DD2	
50	DD1	
55	DD0	
56	IRQ4	Inverted input signal
57	SA7	
58	IRQ3	Inverted input signal
59	SA6	
60	DACK2#	
61	SA5	
62	TC	
63	SA4	
64	BALE	
66	SA3	
67	SA2	
68	SA1	
69	SA0	
70	MEMCS16#	
71	IOCS16#	
72	LA23	

Table 13. NAND Tree (Continued)

Pin #	Pin Name	Notes
73	IRQ10	Inverted input signal
74	LA22	
75	IRQ11	Inverted input signal
76	LA21	
77	IRQ12	Inverted input signal
80	LA20	
81	IRQ15	Inverted input signal
82	LA19	
83	IRQ14	Inverted input signal
84	LA18	
85	DACK0#	
86	LA17	
87	DREQ0	
88	MEMR#	
89	DACK5#	
90	MEMW#	
91	DREQ5	
92	SD8	
93	DACK6#	
94	SD9	
95	DREQ6	
96	SD10	
97	DACK7#	
98	SD11	
99	DREQ7	
100	SD12	
101	SD13	
102	SD14	
107	SD15	
108	DDRQ0	

Pin #	Pin Name	Notes
109	PHOLD#	
110	PHLDA#	
111	DDRQ1	
112	DIOW#	
113	DIOR#	
114	IORDY	
115	DDAK0#	
116	DDAK1#	
117	SPKR	
118	SDIR	
119	SOE#	
120	FERR#	
121	IGNNE#	
122	INTR	
123	SMI#	
124	STPCLK#	
125	EXTSMI#	
126	PWROK	Inverted input signal
127	CPURST#	
128	PCIRST#	
129	INIT	
132	PCICLK	Input only, not included in the NAND tree test mode.
135	NMI	
136	OSC	
137	BIOSCS#	
138	RTCCS#	
139	KBCS#	
140	XOE#	
141	XDIR	Final NAND tree output

Figure 10 is a schematic of the NAND tree circuitry.

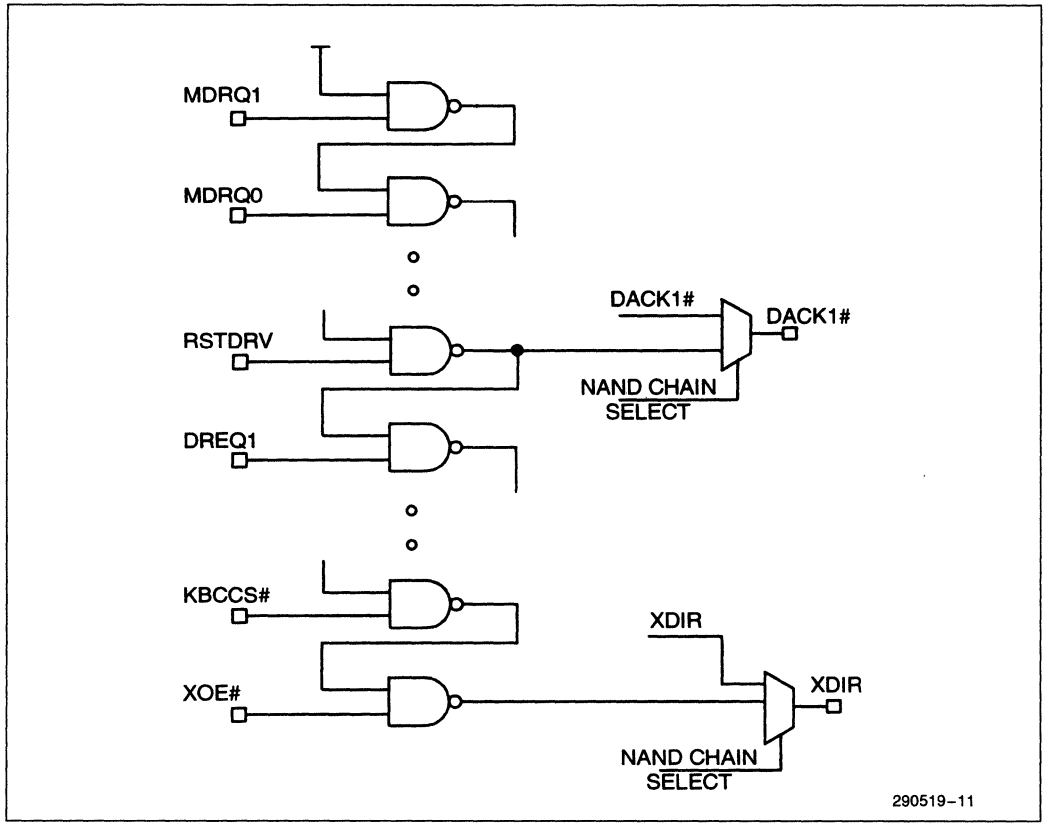


Figure 10. NAND Tree Circuitry

NAND Tree Timing Requirements

Allow 500 ns for the input signals to propagate to the NAND tree outputs (input-to-output propagation delay specification).

6.3 Tri-state Mode

The TESTIN# signal must be 0 and IRQ's 7, 6, and 5 must be 1 to enter the tri-state test mode. When in the tri-state test mode, all outputs and bi-directional pins are tri-stated, including the NAND tree outputs.

82430LX/82430NX PCIset

- Supports the Pentium® Processor at 60 and 66 MHz (82430LX)
- Supports the Pentium Processor at iCOMP™ Index 735\90 MHz, Pentium Processor at iCOMP Index 815\100 MHz, and Pentium Processor at iCOMP Index 610\75 MHz
- Supports Uni-Processor (UP) or Dual-Processor (DP) Configurations
- Interfaces the Host and Standard Buses to the PCI Local Bus
  - Up to 132 MBytes/Sec Transfer Rate
  - Full Concurrency Between CPU Host Bus and PCI Bus Transactions
- Integrated Cache Controller Provided for Optional Second Level Cache
  - 256 KByte or 512 KByte Cache
  - Write-Back or Write-Through Policy (82430LX)
  - Write-Back Policy (82430NX)
  - Standard or Burst SRAM
- Integrated Tag RAM for Cost Savings on Second Level Cache
- Supports the Pipelined Address Mode of the Pentium Processor for Higher Performance
- Provides a 64-Bit Interface to DRAM Memory
  - From 2 MBytes to 512 MBytes of Main Memory
  - 70 ns and 60 ns DRAMs Supported
- Optional ISA or EISA Standard Bus Interface
  - Single Component ISA Controller
  - Two Component EISA Bus Interface
  - Minimal External Logic Required
- Supports Burst Read and Writes of Memory from the CPU and PCI Buses
- Five Integrated Write Posting and Read Prefetch Buffers Increase CPU and PCI Performance
- Host CPU Writes to PCI Converted to Zero Wait-State PCI Bursts with Optional TRDY# Connection
- Integrated Low Skew Host Bus Clock Driver for Cost and Board Space Savings
- PCIset Operates Synchronous to the CPU and PCI Clocks
- Byte Parity Support for the Host and Main Memory Buses
  - Optional Parity on the Second Level Cache

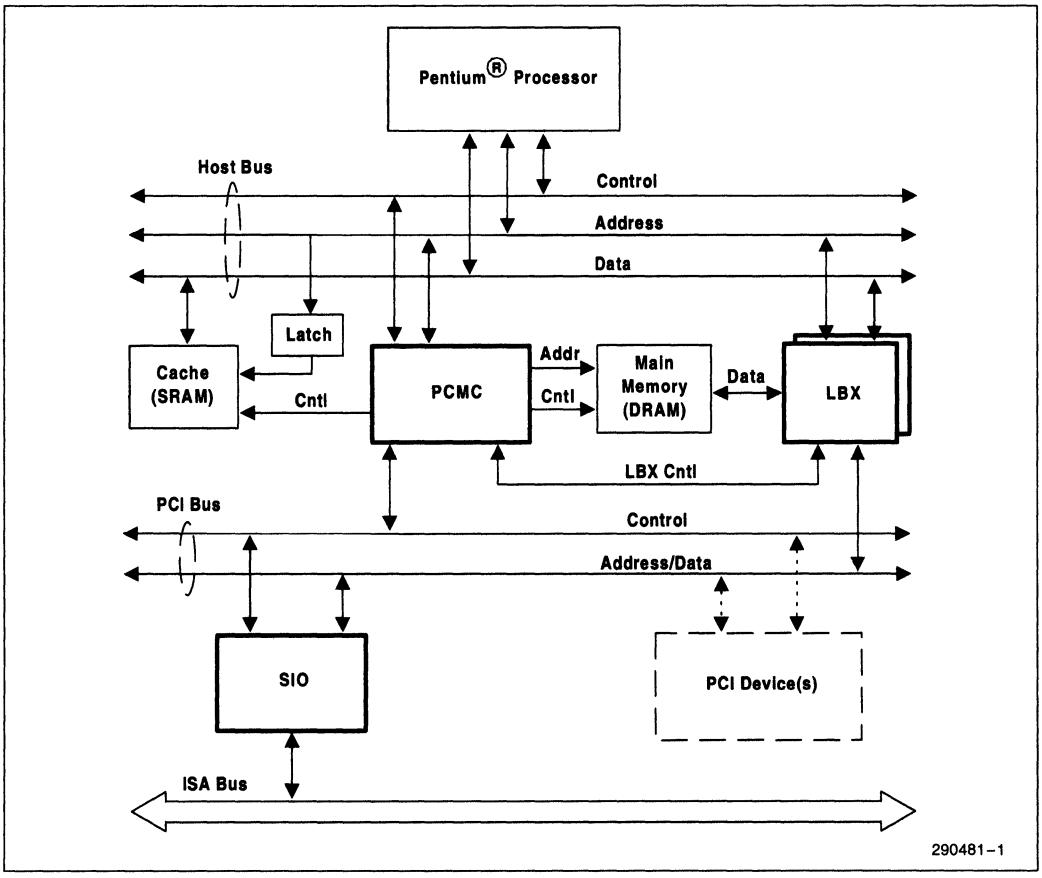
The 82430LX/82430NX PCIsets provide the Host/PCI bridge, cache/main memory controller, and an I/O subsystem core (either PCI/EISA or PCI/ISA bridge) for the next generation of high-performance personal computers based on the Pentium processor. System designers can take advantage of the power of the PCI Local bus for the local I/O while maintaining access to the large base of EISA and ISA expansion cards, and corresponding software applications. Extensive buffering and buffer management within the bridges ensures maximum efficiency in all three bus environments (Host CPU, PCI, and EISA/ISA Buses).

The 82430LX PCIset consists of the 82434LX PCI/Cache Memory Controller (PCMC) and the 82433LX Local Bus Accelerator (LBX) components, plus, either a PCI/ISA bridge or a PCI/EISA bridge. The PCMC and LBX provide the core cache and main memory architecture and serve as the Host/PCI bridge. For an ISA-based system, the 82430LX PCIset includes the 82378ZB System I/O (SIO) component as the PCI/ISA bridge. For an EISA-based system, the 82430LX PCIset includes the 82375EB/SB PCI/EISA Bridge (PCEB) and the 82374EB/SB EISA System Component (ESC). The PCEB and ESC work in tandem to form the complete PCI/EISA bridge. Both the ISA and EISA-based systems are shown on the following pages.

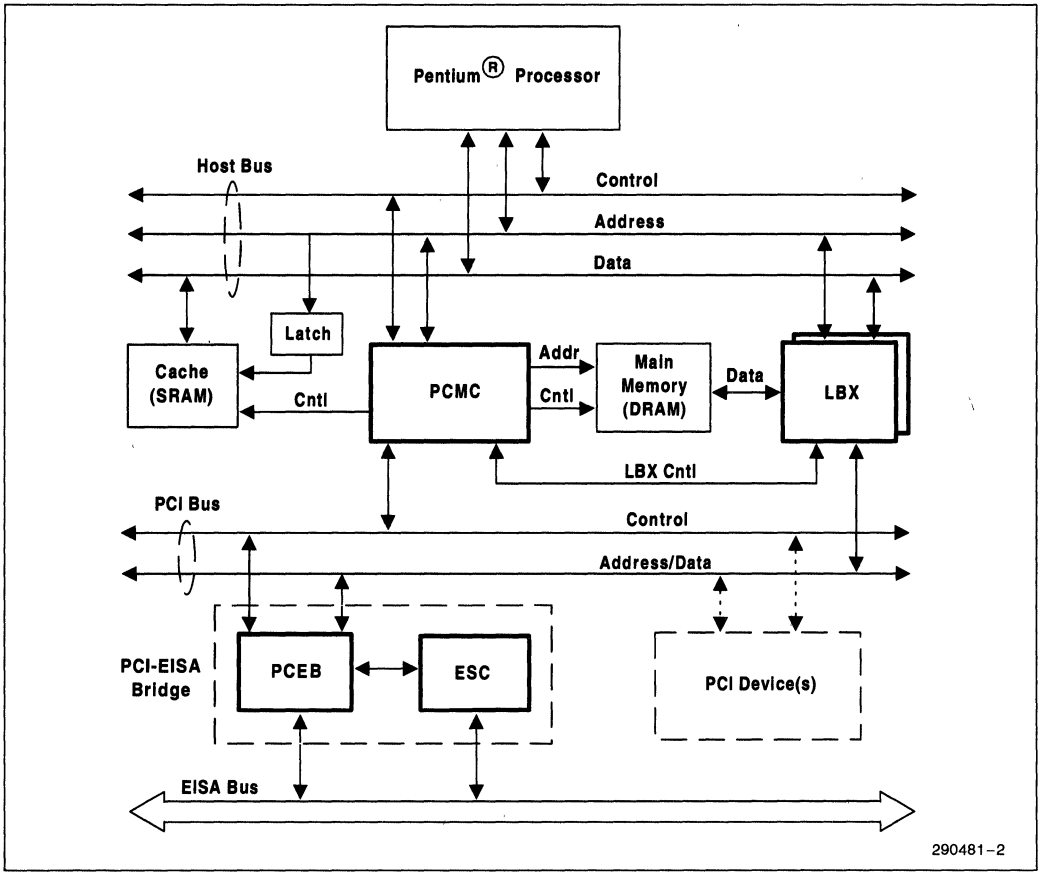
The 82430NX PCIset consists of the 82434NX PCI/Cache Memory Controller (PCMC) and the 82433NX Local Bus Accelerator (LBX) components, plus, either a PCI/ISA bridge or a PCI/EISA bridge. For an ISA-based system, the 82430NX PCIset includes the 82378ZB System I/O (SIO) component as the PCI/ISA bridge. For the DP ISA based system, the 82430NX PCIset includes the 82379AB. For UP or DP EISA-based systems, the 82430NX PCIset includes the 82375EB/SB PCI/EISA Bridge (PCEB) and the 82374EB/SB EISA System Component (ESC).

This document describes both the 82430LX and 82430NX. Unshaded areas describe the 82434LX. Shaded areas, like this one, describe 82430NX operations that differ from the 82434LX.

November 1995
Order Number: 290481-005



82430LX or 82430NX PCIset ISA Block Diagram



82430LX or Uni-Processor 82430NX PCIset EISA Block Diagram

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

82496 CACHE CONTROLLER AND 82491 CACHE SRAM FOR USE WITH THE Pentium® PROCESSOR

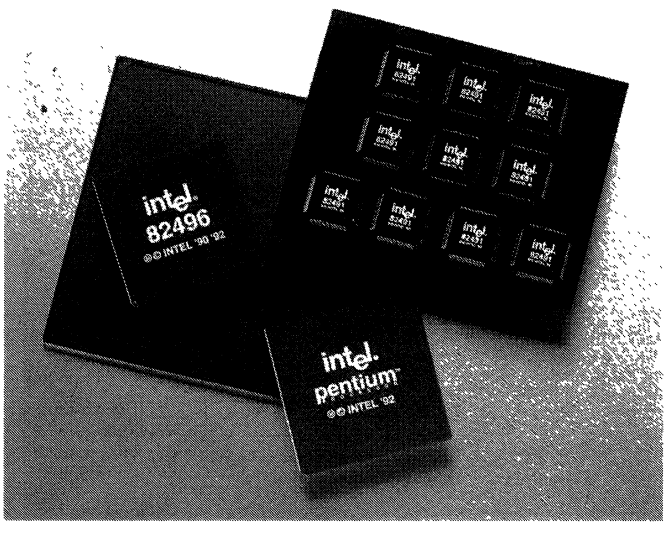
- **High Performance Second Level Cache**
  - **Zero Wait States at 66 MHz**
  - **Two-way Set Associative**
  - **Write-Back with MESI Protocol**
  - **Concurrent CPU Bus and Memory Bus Operation**
  - **Boundary Scan**
- **Pentium® Processor**
  - **Chip Set Version of Pentium Processor**
  - **Superscalar Architecture**
  - **Enhanced Floating Point**
  - **On-chip 8K Code and 8K Data Caches**
  - **See Pentium® Processor Family Developers Manual, Volume 2 for more Information**
- **Highly Flexible**
  - **256K to 512K with parity**
  - **32, 64, or 128-Bit Wide Memory Bus**
  - **Synchronous, Asynchronous, and Strobed Memory Bus Operation**
  - **Selectable Bus Widths, Line Sizes, Transfers, and Burst Orders**
- **Full Multiprocessing Support**
  - **Concurrent CPU, Memory Bus, and Snoop Operations**
  - **Complete MESI Protocol**
  - **Internal/External Parity Generation/ Checking**
  - **Supports Read-for Ownership, Write-Allocation, and Cache-to-Cache Transfers**

2

The 82496 Cache Controller and multiple 82491 Cache SRAMs combine with the Pentium processor to form a CPU Cache chip set designed for high performance servers and function-rich desktops. The high speed interconnect between the CPU and cache components has been optimized to provide zero-wait state operation. This CPU Cache chip set is fully compatible with existing software, and has new data integrity features for mission critical applications.

The 82496 cache controller implements the MESI write-back protocol for full multiprocessing support. Dual ported buffers and registers allow the 82496 to concurrently handle CPU bus, memory bus, and internal cache operation for maximum performance.

The 82491 is a customized high-performance SRAM that supports 32, 64, and 128-bit wide memory bus widths, 16, 32, and 64 byte line sizes, and optional sectoring. The data path between the CPU bus and memory bus is separated by the 82491, allowing the CPU bus to handshake synchronously, asynchronously, or with a strobed protocol, and allowing concurrent CPU bus and memory bus operations.



241814-1

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

November 1995
Order Number: 241814-001

82497 CACHE CONTROLLER AND 82492 CACHE SRAM

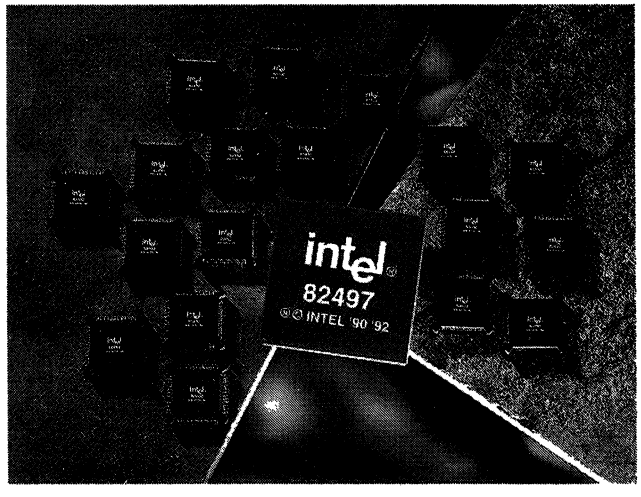
For Use with the Pentium® Processor (735\90, 815\100)

- **High Performance Second Level Cache**
  - **Zero Wait States at 66 MHz**
  - **Two-Way Set Associative**
  - **Writeback with MESI Protocol**
  - **Concurrent CPU Bus and Memory Bus Operation**
  - **Boundary Scan**
- **Pentium® Processor (735\90, 815\100)**
  - **Chip Set Version of Pentium® Processor (735\90, 815\100)**
  - **Superscalar Architecture**
  - **Enhanced Floating Point**
  - **On-Chip 8K Code and 8K Data Caches**
  - **See *Pentium® Processor Family Developer's Manual, Volume 2* for More Information**
- **Highly Flexible**
  - **256K to 512K with Parity**
  - **32-, 64-, or 128-Bit Wide Memory Bus**
  - **Synchronous, Asynchronous and Strobed Memory Bus Operation**
  - **Selectable Bus Widths, Line Sizes, Transfers and Burst Orders**
- **Full Multiprocessing Support**
  - **Concurrent CPU, Memory Bus and Snoop Operations**
  - **Complete MESI Protocol**
  - **Internal/External Parity Generation/ Checking**
  - **Supports Read For Ownership, Write-Allocation and Cache-to-Cache Transfers**

The 82497 Cache Controller and multiple 82492 Cache SRAMs combine with the Pentium® processor (735\90, 810\100) to form a CPU Cache chip set designed for high performance servers and function-rich desktops. The high-speed interconnect between the CPU and cache components has been optimized to provide zero-wait state operation. This CPU Cache chip set is fully compatible with existing software, and has new data integrity features for mission critical applications.

The 82497 cache controller implements the MESI write-back protocol for full multiprocessing support. Dual ported buffers and registers allow the 82497 to concurrently handle CPU bus, memory bus, and internal cache operation for maximum performance.

The 82492 is a customized high-performance SRAM that supports 32-, 64-, 128-bit wide memory bus widths, 16-, 32-, and 64-byte line sizes, and optional sectoring. The data path between the CPU bus and memory bus is separated by the 82492, allowing the CPU bus to handshake synchronously, asynchronously, or with a strobed protocol, and allowing concurrent CPU bus and memory bus operations.



242425-1

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

November 1995
Order Number: 242425-002

82498 CACHE CONTROLLER AND 82493 CACHE SRAM

For use with the Pentium® Processor (735/90, 815/100)

- **High Performance Second Level Cache**
  - **— Zero Wait States at 66 MHz**
  - **— Two-Way Set Associative**
  - **— Writeback with MESI Protocol**
  - **— Concurrent CPU Bus and Memory Bus Operation**
  - **— Boundary Scan**
- **Pentium® Processor (735/90, 815/100)**
  - **— Chip Set Version of Pentium® Processor (735/90, 815/100)**
  - **— Superscalar Architecture**
  - **— Enhanced Floating Point**
  - **— On-Chip 8K Code and 8K Data Caches**
  - **— See *Pentium® Processor Family Developer's Manual, Volume 2* for More Information**
- **Highly Flexible**
  - **— 1 Mbyte to 2 Mbyte**
  - **— 64-, or 128-Bit Wide Memory Bus**
  - **— Synchronous, Asynchronous and Strobed Memory Bus Operation**
  - **— Selectable Bus Widths, Line Sizes, Transfers and Burst Orders**
- **Full Multiprocessing Support**
  - **— Concurrent CPU, Memory Bus and Snoop Operations**
  - **— Complete MESI Protocol**
  - **— Internal/External Parity Generation/Checking**
  - **— Supports Read For Ownership, Write-Allocation and Cache-to-Cache Transfers**

2

The 82498 Cache Controller and multiple 82493 Cache SRAMs combine with the Pentium® processor (735/90, 815/100) and future Pentium Processors to form a CPU Cache chip set designed for high performance servers and function-rich desktops. The high-speed interconnect between the CPU and cache components has been optimized to provide zero-wait state operation. This CPU Cache chip set is fully compatible with existing software, and has new data integrity features for mission critical applications.

The 82498 Cache Controller implements the MESI write-back protocol for full multiprocessing support. Dual ported buffers and registers allow the 82498 to concurrently handle CPU bus, memory bus, and internal cache operation for maximum performance.

The 82493 is a customized high-performance SRAM that supports 64-, and 128-bit wide memory bus widths, 32-, and 64-byte line sizes, and optional sectoring. The data path between the CPU bus and memory bus is separated by the 82493, allowing the CPU bus to handshake synchronously, asynchronously, or with a strobed protocol, and allowing concurrent CPU bus and memory bus operations.

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

November 1995
Order Number: 242426-002

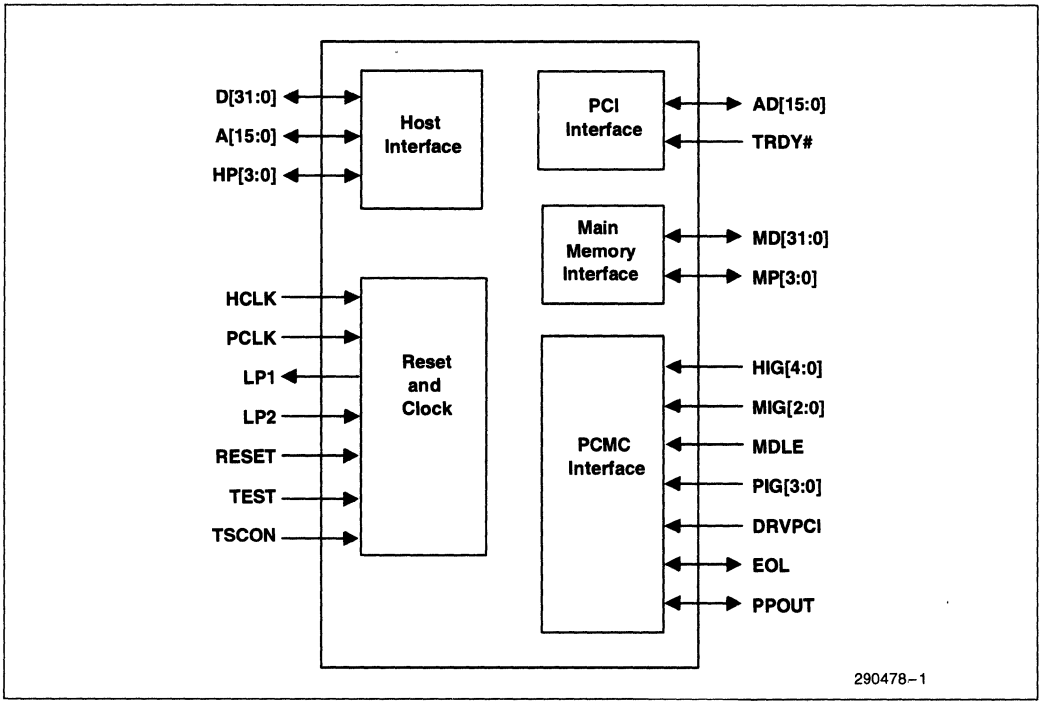
82433LX/82433NX
LOCAL BUS ACCELERATOR (LBX)

- **Supports the Full 64-bit Pentium® Processor Data Bus at Frequencies up to 66 MHz (82433LX and 82433NX)**
- **Drives 3.3V Signal Levels on the CPU Data and Address Buses (82433NX)**
- **Provides a 64-Bit Interface to DRAM and a 32-Bit Interface to PCI**
- **Five Integrated Write Posting and Read Prefetch Buffers Increase CPU and PCI Performance**
  - **— CPU-to-Memory Posted Write Buffer 4 Qwords Deep**
  - **— PCI-to-Memory Posted Write Buffer Two Buffers, 4 Dwords Each**
  - **— PCI-to-Memory Read Prefetch Buffer 4 Qwords Deep**
  - **— CPU-to-PCI Posted Write Buffer 4 Dwords Deep**
  - **— CPU-to-PCI Read Prefetch Buffer 4 Dwords Deep**
- **CPU-to-Memory and CPU-to-PCI Write Posting Buffers Accelerate Write Performance**
- **Dual-Port Architecture Allows Concurrent Operations on the Host and PCI Buses**
- **Operates Synchronously to the CPU and PCI Clocks**
- **Supports Burst Read and Writes of Memory from the Host and PCI Buses**
- **Sequential CPU Writes to PCI Converted to Zero Wait-State PCI Bursts with Optional TRDY# Connection**
- **Byte Parity Support for the Host and Memory Buses**
  - **— Optional Parity Generation for Host to Memory Transfers**
  - **— Optional Parity Checking for the Secondary Cache**
  - **— Parity Checking for Host and PCI Memory Reads**
  - **— Parity Generation for PCI to Memory Writes**
- **160-Pin QFP Package**

Two 82433LX or 82433NX Local Bus Accelerator (LBX) components provide a 64-bit data path between the host CPU/Cache and main memory, a 32-bit data path between the host CPU bus and PCI Local Bus, and a 32-bit data path between the PCI Local Bus and main memory. The dual-port architecture allows concurrent operations on the host and PCI Buses. The LBXs incorporate three write posting buffers and two read prefetch buffers to increase CPU and PCI performance. The LBX supports byte parity for the host and main memory buses. The 82433NX is intended to be used with the 82434NX PCI/Cache/Memory Controller (PCMC). The 82433LX is intended to be used with the 82434LX PCMC. During bus operations between the host, main memory and PCI, the PCMC commands the LBXs to perform functions such as latching address and data, merging data, and enabling output buffers. Together, these three components form a "Host Bridge" that provides a full function dual-port data path interface, linking the host CPU and PCI bus to main memory.

This document describes both the 82433LX and 82433NX. Shaded areas, like this one, describe the 82433NX operations that differ from the 82433LX.

December 1995
Order Number: 290478-004



LBX Simplified Block Diagram

2

82433LX/82433NX
LOCAL BUS ACCELERATOR (LBX)

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

The 82430 PCIset consists of the 82434LX PCMC and 82433LX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The 82430NX PCIset consists of the 82434NX PCMC and 82433NX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The PCMC and LBX provide the core cache and main memory architecture and serves as the Host/PCI bridge. An overview of the PCMC follows the system overview section.

The Local Bus Accelerator (LBX) provides a high performance data and address path for the 82430LX/82430NX PCIset. The LBX incorporates five integrated buffers to increase the performance of the Pentium processor and PCI master devices. Two LBXs in the system support the following areas:

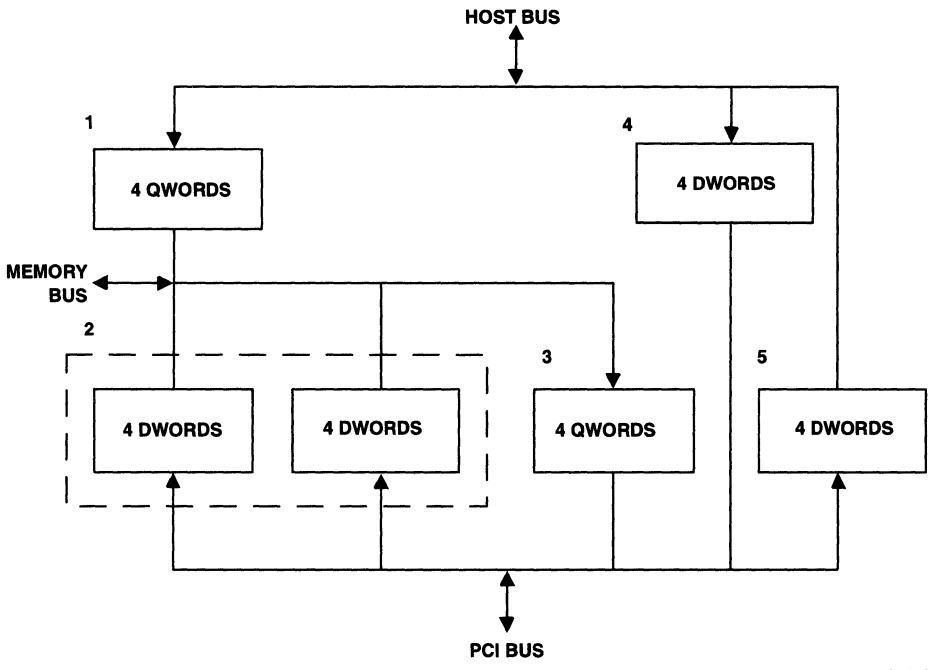
1. 64-bit data and 32-bit address bus of the Pentium processor.
2. 32-bit multiplexed address/data bus of PCI.
3. 64-bit data bus of the main memory.

In addition, the LBXs provide parity support for the three areas noted above (discussed further in Section 1.4).

1.1 Buffers in the LBX

The LBX components have five integrated buffers designed to increase the performance of the Host and PCI Interfaces of the 82430LX/82430NX PCIset.

With the exception of the PCI-to-Memory write buffer and the CPU-to-PCI write buffer, the buffers in the LBX store data only, addresses are stored in the PCMC component.



NOTES:
1. **CPU-to-Memory Posted Write Buffer:** This buffer is 4 Qwords deep, enabling the Pentium processor to write back a whole cache line in 4-1-1-1 timing, a total of 7 CPU clocks.
2. **PCI-to-Memory Posted Write Buffer:** A PCI master can post two consecutive sets of 4 Dwords (total of one cache line) or two single non-consecutive transactions.
3. **PCI-to-Memory Read Prefetch Buffer:** A PCI master to memory read transaction will cause this prefetch buffer to read up to 4 Qwords of data from memory, allowing up to 8 Dwords to be read onto PCI in a single burst transaction.
4. **CPU-to-PCI Posted Write Buffer:** The Pentium processor can post up to 4 Dwords into this buffer. The TRDY# connect option allows zero-wait state burst writes to PCI, making this buffer especially useful for graphic write operations.
5. **CPU-to-PCI Read Prefetch Buffer:** This prefetch buffer is 4 Dwords deep, enabling faster sequential Pentium processor reads when targeting PCI.

Figure 1. Simplified Block Diagram of the LBX Data Buffers

1.2 Control Interface Groups

The LBX is controlled by the PCMC via the control interface group signals. There are three interface groups: Host, Memory, and PCI. These control groups are signal lines that carry binary codes which the LBX internally decodes in order to implement specific functions such as latching data and steering data from PCI to memory. The control interfaces are described below.

1. **Host Interface Group:** These control signals are named HIG[4:0] and define a total of 29 (30 for the 82433NX) discrete commands. The PCMC sends HIG commands to direct the LBX to perform functions related to buffering and storing host data and/or address.
2. **Memory Interface Group:** These control signals are named MIG[2:0] and define a total of 7 discrete commands. The PCMC sends MIG commands to direct the LBX to perform functions related to buffering, storing, and retiring data to memory.
3. **PCI Interface Group:** These control signals are named PIG[3:0] and define a total of 15 discrete commands. The PCMC sends PIG commands to direct the LBX to perform functions related to buffering and storing PCI data and/or address.

1.3 System Bus Interconnect

The architecture of the 82430/82430NX PCIset splits the 64-bit memory and host data buses into logical halves in order to manufacture LBX devices with manageable pin counts. The two LBXs interface to the 32-bit PCI AD[31:0] bus with 16 bits each. Each LBX connects to 16 bits of the AD[31:0] bus and 32-bits of both the MD[0:63] bus and the D[0:63] bus. The lower order LBX (LBXL) connects to the low word of the AD[31:0] bus, while the high order LBX (LBXH) connects to the high word of the AD[31:0] bus.

Since the PCI connection for each LBX falls on 16-bit boundaries, each LBX does not simply connect to either the low Dword or high Dword of the Qword memory and host buses. Instead, the low order LBX buffers the first and third words of each 64-bit bus while the high order LBX buffers the second and fourth words of the memory and host buses.

As shown in Figure 2, LBXL connects to the first and third words of the 64-bit main memory and host data buses. The same device also drives the first 16 bits of the host address bus, A[15:0]. The LBXH device connects to the second and fourth words of the 64-bit main memory and host data buses. Correspondingly, LBXH drives the remaining 16 bits of the host address bus, A[31:16].

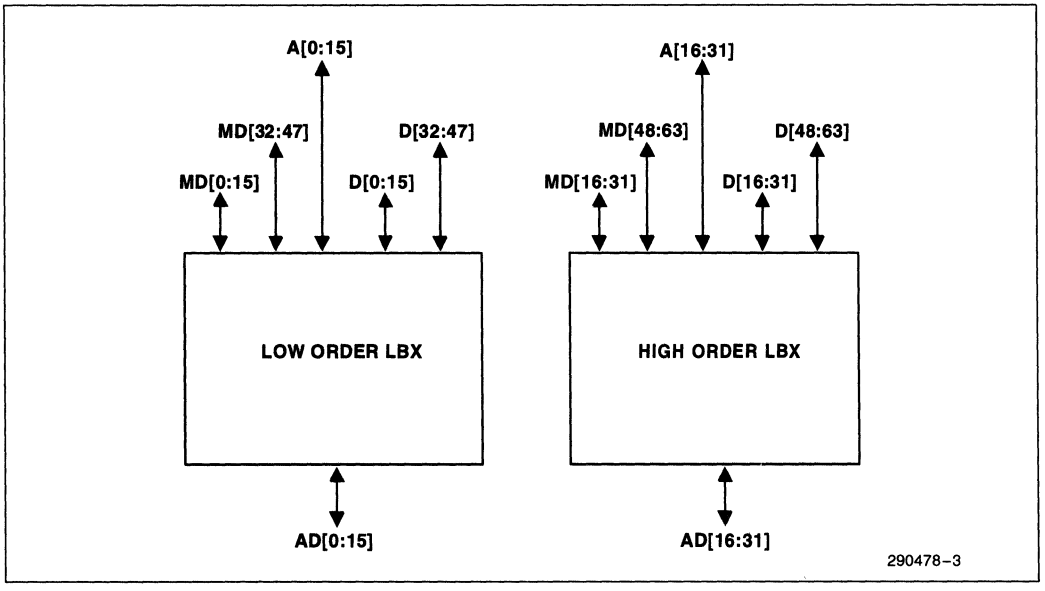


Figure 2. Simplified Interconnect Diagram of LBXs to System Buses

1.4 PCI TRDY# Interface

The PCI control signals do not interface to the LBXs, instead these signals connect to the 82434LX PCMC component. The main function of the LBXs PCI interface is to drive address and data onto PCI when the CPU targets PCI and to latch address and data when a PCI master targets main memory.

The TRDY# option provides the capability for zero-wait state performance on PCI when the Pentium processor performs sequential writes to PCI. This option requires that PCI TRDY# be connected to each LBX, for a total of two additional connections in the system. These two TRDY# connections are in addition to the single TRDY# connection that the PCMC requires.

1.5 Parity Support

The LBXs support byte parity on the host bus (CPU and second level cache) and main memory buses (local DRAM). The LBXs support parity during the address and data phases of PCI transactions to/from the host bridge.

2.0 SIGNAL DESCRIPTIONS

This section provides a detailed description of each signal. The signals (Figure 3) are arranged in functional groups according to their associated interface.

The '#' symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When '#' is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of 'active-low' and 'active-high' signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

2

The following notations are used to describe the signal type.

in Input is a standard input-only signal.

out Totem Pole output is a standard active driver.

t/s Tri-State is a bi-directional, tri-state input/output pin.

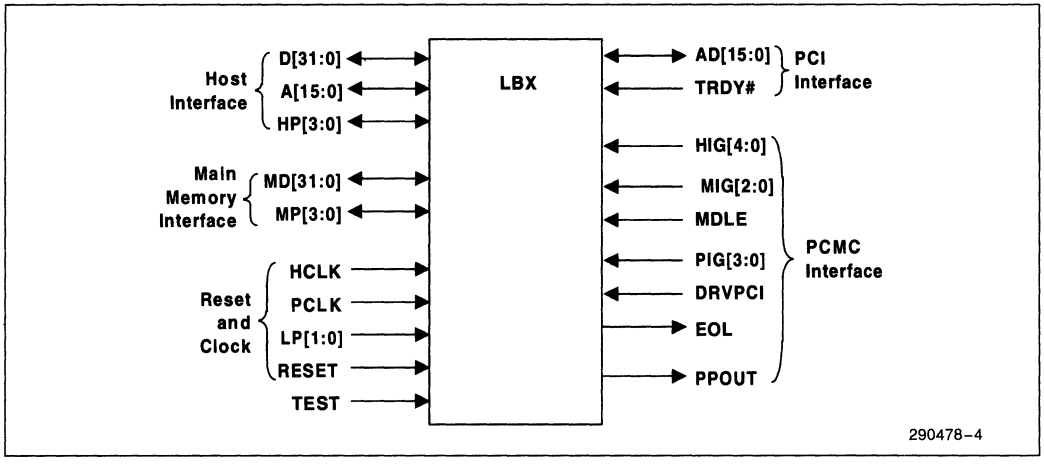


Figure 3. LBX Signals

2.1 Host Interface Signals

Signal	Type	Description
A[15:0]	t/s	**ADDRESS BUS:** The bi-directional A[15:0] lines are connected to the address lines of the host bus. The high order LBX (determined at reset time using the EOL signal) is connected to A[31:16], and the low order LBX is connected to A[15:0]. The host address bus is common with the Pentium processor, second level cache, PCMC and the two LBXs. During CPU cycles A[31:3] are driven by the CPU and A[2:0] are driven by the PCMC, all are inputs to the LBXs. During inquire cycles the LBX drives the PCI master address onto the host address lines A[31:0]. This snoop address is driven to the CPU and the PCMC by the LBXs to snoop L1 and the integrated second level tags, respectively. During PCI configuration cycles bound for the PCMC, the LBXs will send or receive the configuration data to/from the PCMC by copying the host data bus to/from the host address bus. The LBX drives both halves of the Qword host data bus with data from the 32-bit address during PCMC configuration read cycles. The LBX drives the 32-bit address with either the low Dword or the high Dword during PCMC configuration write cycles. In the 82433NX, these pins contain weak internal pull-down resistors. The high order 82433NX LBX samples A11 at the falling edge of reset to configure the LBX for PLL test mode. When A11 is sampled low, the LBX is in normal operating mode. When A11 is sampled high, the LBX drives the internal HCLK from the PLL on the EOL pin. Note that A11 on the high order LBX is connected to the A27 line on the CPU address bus. This same address line is used to put the PCMC into PLL test mode.
D[31:0]	t/s	**HOST DATA:** The bi-directional D[31:0] lines are connected to the data lines of the host data bus. The high order LBX (determined at reset time using the EOL signal) is connected to the host data bus D[63:48] and D[31:16] lines, and the low order LBX is connected to the host data bus D[47:32] and D[15:0] lines. In the 82433LX, these pins contain weak internal pull-up resistors. In the 82433NX, these pins contain weak internal pull-down resistors.
HP[3:0]	t/s	**HOST DATA PARITY:** HP[3:0] are the bi-directional byte parity signals for the host data bus. The low order parity bit HP[0] corresponds to D[7:0] while the high order parity bit HP[3] corresponds to D[31:24]. The HP[3:0] signals function as parity inputs during write cycles and as parity outputs during read cycles. Even parity is supported and the HP[3:0] signals follow the same timings as D[31:0]. In the 82433LX, these pins contain weak internal pull-up resistors. In the 82433NX, these pins contain weak internal pull-down resistors.

2.2 Main Memory (Dram) Interface Signals

Signal	Type	Description
MD[31:0]	t/s	**MEMORY DATA BUS:** MD[31:0] are the bi-directional data lines for the memory data bus. The high order LBX (determined at reset time using the EOL signal) is connected to the memory data bus MD[63:48] and MD[31:16] lines, and the low order LBX is connected to the memory data bus MD[47:32] and MD[15:0] lines. The MD[31:0] signals drive data destined for either the host data bus or the PCI bus. The MD[31:0] signals input data that originated from either the host data bus or the PCI bus. These pins contain weak internal pull-up resistors.
MP[3:0]	t/s	**MEMORY PARITY:** MP[3:0] are the bi-directional byte enable parity signals for the memory data bus. The low order parity bit MP[0] corresponds to MD[7:0] while the high order parity bit MP[3] corresponds to MD[31:24]. The MP[3:0] signals are parity outputs during write cycles to memory and parity inputs during read cycles from memory. Even parity is supported and the MP[3:0] signals follow the same timings as MD[31:0]. These pins contain weak internal pull-up resistors.

2

2.3 PCI Interface Signals

Signal	Type	Description
AD[15:0]	t/s	**ADDRESS AND DATA:** AD[15:0] are bi-directional data lines for the PCI bus. The AD[15:0] signals sample or drive the address and data on the PCI bus. The high order LBX (determined at reset time using the EOL signal) is connected to the PCI bus AD[31:16] lines, and the low order LBX is connected to the PCI AD[15:0] lines.
TRDY#	in	**TARGET READY:** TRDY# indicates the selected (targeted) device's ability to complete the current data phase of the bus operation. For normal operation, TRDY# is tied asserted low. When the TRDY# option is enabled in the PCMC (for zero wait-state PCI burst writes), TRDY# should be connected to the PCI bus.

2.4 PCMC Interface Signals

Signal	Type	Description
HIG[4:0]	in	**HOST INTERFACE GROUP:** These signals are driven from the PCMC and control the host interface of the LBX. The 82433LX decodes the binary pattern of these lines to perform 29 unique functions (30 for the 83433NX). These signals are synchronous to the rising edge of HCLK.
MIG[2:0]	in	**MEMORY INTERFACE GROUP:** These signals are driven from the PCMC and control the memory interface of the LBX. The LBX decodes the binary pattern of these lines to perform 7 unique functions. These signals are synchronous to the rising edge of HCLK.
PIG[3:0]	in	**PCI INTERFACE GROUP:** These signals are driven from the PCMC and control the PCI interface of the LBX. The LBX decodes the binary pattern of these lines to perform 15 unique functions. These signals are synchronous to the rising edge of HCLK.
MDLE	in	**MEMORY DATA LATCH ENABLE:** During CPU reads from DRAM, the LBX uses a clocked register to transfer data from the MD[31:0] and MP[3:0] lines to the D[31:0] and HP[3:0] lines. MDLE is the clock enable for this register. Data is clocked into this register when MDLE is asserted. The register retains its current value when MDLE is negated. During CPU reads from main memory, the LBX tri-states the D[31:0] and HP[3:0] lines on the rising edge of MDLE when HIG[4:0] = NOPC.
DRVPCI	in	**DRIVE PCI BUS:** This signals enables the LBX to drive either address or data information onto the PCI AD[15:0] lines.

2.4 PCMC Interface Signals (Continued)

Signal	Type	Description
EOL	t/s	**End Of Line:** This signal is asserted when a PCI master read or write transaction is about to overrun a cache line boundary. The low order LBX will have this pin connected to the PCMC (internally pulled up in the PCMC). The high order LBX connects this pin to a pull-down resistor. With one LBX EOL line being pulled down and the other LBX EOL pulled up, the LBX samples the value of this pin on the negation of the RESET signal to determine if it's the high or low order LBX.
PPOUT	t/s	**LBX PARITY:** This signal reflects the parity of the 16 AD lines driven from or latched into the LBX, depending on the command driven on PIG[3:0]. The PCMC uses PPOUT from both LBXs (called PPOUT[1:0]) to calculate the PCI parity signal (PAR) for CPU to PCI transactions during the address phase of the PCI cycle. The LBX uses PPOUT to check the PAR signal for PCI master transactions to memory during the address phase of the PCI cycle. When transmitting data to PCI the PCMC uses PPOUT to calculate the proper value for PAR. When receiving data from PCI the PCMC uses PPOUT to check the value received on PAR. If the L2 cache does not implement parity, the LBX will calculate parity so the PCMC can drive the correct value on PAR during L2 reads initiated by a PCI master. The LBX samples the PPOUT signal at the negation of reset and compares that state with the state of EOL to determine whether the L2 cache implements parity. The PCMC internally pulls down PPOUT[0] and internally pulls up PPOUT[1]. The L2 supports parity if PPOUT[0] is connected to the high order LBX and PPOUT[1] is connected to the low order LBX. The L2 is defined to not support parity if these connections are reversed, and for this case, the LBX will calculate parity. For normal operations either connection allows proper parity to be driven to the PCMC.

2.5 Reset and Clock Signals

Signal	Type	Description
HCLK	in	**HOST CLOCK:** HCLK is input to the LBX to synchronize command and data from the host and memory interfaces. This input is derived from a buffered copy of the PCMC HCLKx output.
PCLK	in	**PCI CLOCK:** All timing on the LBX PCI interface is referenced to the PCLK input. All output signals on the PCI interface are driven from PCLK rising edges and all input signals on the PCI interface are sampled on PCLK rising edges. This input is derived from a buffered copy of the PCMC PCLK output.
RESET	in	**RESET:** Assertion of this signal resets the LBX. After RESET has been negated the LBX configures itself by sampling the EOL and PPOUT pins. RESET is driven by the PCMC CPURST pin. The RESET signal is synchronous to HCLK and must be driven directly by the PCMC.
LP1	out	**LOOP 1:** Phase Lock Loop Filter pin. The filter components required for the LBX are connected to these pins.
LP2	in	**LOOP 2:** Phase Lock Loop Filter pin. The filter components required for the LBX are connected to these pins.
TEST	in	**TEST:** The TEST pin must be tied low for normal system operation.
TSCON	in	**TRI-STATE CONTROL:** This signal enables the output buffers on the LBX. This pin must be held high for normal operation. If TSCON is negated, all LBX outputs will tri-state.

3.0 FUNCTIONAL DESCRIPTION

3.1 LBX Post and Prefetch Buffers

This section describes the five write posting and read prefetching buffers implemented in the LBX. The discussion in this section refers to the operation of both LBXs in the system.

3.1.1 CPU-TO-MEMORY POSTED WRITE BUFFER

The write buffer is a queue 4 Qwords deep, it loads Qwords from the CPU and stores Qwords to memory. It is 4 Qwords deep to accommodate write-backs from the first or second level cache. It is organized as a simple FIFO. Commands driven on the HIG[4:0] lines store Qwords into the buffer, while commands on the MIG[2:0] lines retire Qwords from the buffer. While retiring Qwords to memory, the DRAM controller unit of the PCMC will assert the appropriate MA, CAS[7:0]#, and WE# signals. The PCMC keeps track of full/empty states, status of the data and address.

Byte parity for data to be written to memory is either propagated from the host bus or generated by the LBX. The LBX generates parity for data from the second level cache when the second level cache does not implement parity.

3.1.2 PCI-TO-MEMORY POSTED WRITE BUFFER

The buffer is organized as 2 buffers (4 Dwords each). There is an address storage register for each buffer. When an address is stored one of the two buffers is allocated and subsequent Dwords of data are stored beginning at the first location in that buffer. Buffers are retired to memory strictly in order, Qword at a time.

Commands driven on the PIG[3:0] lines post addresses and data into the buffer. Commands driven on HIG[4:0] result in addresses being driven on the host address bus. Commands driven on MIG[2:0] result in data being retired to DRAM.

For cases where the address targeted by the first Dword is odd, i.e. A[2] = 1, and the data is stored in an even location in the buffer, the LBX correctly aligns the Dword when retiring the data to DRAM. In other words the buffer is capable of retiring a Qword to memory where the data in the buffer is shifted by 1 Dword (Dword is position 0 shifted to 1, 1 shifted to 2 etc.). The DRAM controller of the PCMC asserts the correct CAS[7:0]# signals depending on the PCI C/BE[3:0]# signals stored in the PCMC for that Dword.

The End Of Line (EOL) signal is used to prevent PCI master writes from bursting past the cache line boundary. The device that provides "warning" to the PCMC is the low order LBX. This device contains the PCI master write low order address bits necessary to determine how many Dwords are left to the end of the line. Consequently, the LBX protocol uses the EOL signal from the low order LBX to provide this "end-of-line" warning to the PCMC, so that it may retry a PCI master write when it bursts past the cache line boundary. This protocol is described fully in Section 3.3.6.

The LBX calculates Dword parity on PCI write data, sending the proper value to the PCMC on PPOUT. The LBX generates byte parity on the MP signals for writing into DRAM.

3.1.3 PCI-TO-MEMORY READ PREFETCH BUFFER

This buffer is organized as a line buffer (4 Qwords) for burst transfers to PCI. The data is transferred into the buffer a Qword at a time and read out a Dword at a time. The LBX then effectively decouples the memory read rate from the PCI rate to increase concurrence.

Each new transaction begins by storing the first Dword in the first location in the buffer. The starting Dword for reading data out of the buffer onto PCI must be specified within a Qword boundary; that is the first requested Dword on PCI could be an even or odd Dword. If the snoop for a PCI master read results in a write-back from first or second level caches, this write back is sent directly to PCI and main memory. The following two paragraphs describe this process for cache line write-backs.

Since the write-back data from L1 is in linear order, writing into the buffer is straightforward. Only those Qwords to be transferred into PCI are latched into the PCI-to-memory read buffer. For example, if the address targeted by PCI is in the 3rd or 4th Qword in the line, the first 2 Qwords of write back data are discarded and not written into the read buffer. The primary cache write-back must always be written

completely to the CPU-to-Memory posted Write Buffer.

If the PCI master read data is read from the secondary cache, it is not written back to memory. Write-backs from the second level cache, when using burst SRAMs, are in Pentium processor burst order (the order depending on which Qword of the line is targeted by the PCI read). The buffer is directly addressed when latching second level cache write-back data to accommodate this burst order. For example, if the requested Qword is Qword 1, then the burst order is 1-0-3-2. Qword 1 is latched in buffer location 0, Qword 0 is discarded, Qword 3 is latched into buffer location 2 and Qword 2 is latched into buffer location 1.

Commands driven on MIG[2:0] and HIG[4:0] enter data into the buffer from the DRAM interface and the host interface (i.e. the caches), respectively. Commands driven on the PIG[3:0] lines drive data from the buffer onto the PCI AD[31:0] lines.

Parity driven on the PPOUT signal is calculated from the byte parity received on the host bus or the memory bus, whichever is the source. If the second level cache is the source of the data and does not implement parity, the parity driven on PPOUT is generated by the LBX from the second level cache data. If main memory is the source of the read data, PCI parity is calculated from the DRAM byte parity. Main memory must implement byte parity to guarantee correct PCI parity generation.

3.1.4 CPU-TO-PCI POSTED WRITE BUFFER

The CPU-to-PCI Posted Write Buffer is 4 Dwords deep. The buffer is constructed as a simple FIFO, with some performance enhancements. An address is stored in the LBX with each Dword of data. The structure of the buffer accommodates the packetization of writes to be burst on PCI. This is accomplished by effectively discarding addresses of data Dwords driven within a burst. Thus, while an address is stored for each Dword, an address is not necessarily driven on PCI for each Dword. The PCMC determines when a burst write may be performed based on consecutive addresses. The buffer also enables consecutive bytes to be merged within a single Dword, accommodating byte, word, and misaligned Dword string store and string move operations. Qword writes on the host bus are stored within the buffer as two individual Dword writes, with separate addresses.

The storing of an address with each Dword of data allows burst writes to be retried easily. In order to retry transactions, the FIFO is effectively "backed up" by one Dword. This is accomplished by making the FIFO physically one entry larger than it is logically. Thus, the buffer is physically 5 entries deep (an entry consists of an address and a Dword of data), while logically it is considered full when 4 entries have been posted. This design allows the FIFO to be backed up one entry when it is logically full.

Commands driven on HIG[4:0] post addresses and data into the buffer, and commands driven on PIG[3:0] retire addresses and data from the buffer and drive them onto the PCI AD[31:0] lines. As discussed previously, when bursting, not all addresses are driven onto PCI.

Data parity driven on the PPOUT signal is calculated from the byte parity received on the host bus. Address parity driven on PPOUT is calculated from the address received on the host bus.

3.1.5 CPU-TO-PCI READ PREFETCH BUFFER

This prefetch buffer is organized as a single buffer 4 Dwords deep. The buffer is organized as a simple FIFO. reads from the buffer are sequential; the buffer does not support random access of its contents. To support reads of less than a Dword the FIFO read pointer can function with or without a pre-increment. The pointer can also be reset to the first entry before a Dword is driven. When a Dword is read, it is driven onto both halves of the host data bus.

Commands driven on the HIG[4:0] lines enable read addresses to be sent onto PCI, the addresses are driven using PIG[3:0] commands. Read data is latched into the LBX by commands driven on the PIG[3:0] lines and the data is driven onto the host data bus using commands driven on the HIG[4:0] lines.

The LBX calculates Dword parity on PCI read data, sending the proper value to the PCMC on PPOUT. The LBX does not generate byte parity on the host data bus when the CPU reads PCI.

3.2 LBX Interface Command Descriptions

This section describes the functionality of the HIG, MIG and PIG commands driven by the PCMC to the LBXs.

3.2.1 HOST INTERFACE GROUP: HIG[4:0]

The Host Interface commands are shown in Table 1. These commands are issued by the host interface of the PCMC to the LBXs in order to perform the following functions:

- Reads from CPU-to-PCI read prefetch buffer when the CPU reads from PCI.
- Stores write-back data to PCI-to-memory read prefetch buffer when PCI read address results in a hit to a modified line in first or second level caches.
- Posts data to CPU-to-memory write buffer in the case of a CPU to memory write.
- Posts data to CPU-to-PCI write buffer in the case of a CPU to PCI write.
- Drives host address to Data lines and data to address lines for programming the PCMC configuration registers.

Table 1. HIG Commands

Command	Code	Description
NOPC	00000b	No Operation on CPU Bus
CMR	11100b	CPU Memory Read
CPRF	00100b	CPU Read First Dword from CPU-to-PCI Read Prefetch Buffer
CPRA	00101b	CPU Read Next Dword from CPU-to-PCI Read Prefetch Buffer, Toggle A
CPRB	00110b	CPU Read Next Dword from CPU-to-PCI Read Prefetch Buffer, Toggle B
CPRQ	00111b	CPU Read Qword from CPU-to-PCI Read Prefetch Buffer
SWB0	01000b	Store Write-Back Data Qword 0 to PCI-to-Memory Read Buffer
SWB1	01001b	Store Write-Back Data Qword 1 to PCI-to-Memory Read Buffer
SWB2	01010b	Store Write-Back Data Qword 2 to PCI-to-Memory Read Buffer
SWB3	01011b	Store Write-Back Data Qword 3 to PCI-to-Memory Read Buffer
PCMWQ	01100b	Post to CPU-to-Memory Write Buffer Qword
PCMWFQ	01101b	Post to CPU-to-Memory Write and PCI-to-Memory Read Buffer First Qword
PCMWNQ	01110b	Post to CPU-to-Memory Write and PCI-to-Memory Read Buffer Next Qword
PCPWL	10000b	Post to CPU-to-PCI Write Low Dword
MCP3L	10011b	Merge to CPU-to-PCI Write Low Dword 3 Bytes
MCP2L	10010b	Merge to CPU-to-PCI Write Low Dword 2 Bytes
MCP1L	10001b	Merge to CPU-to-PCI Write Low Dword 1 Byte
PCPWH	10100b	Post to CPU-to-PCI Write High Dword
MCP3H	10111b	Merge to CPU-to-PCI Write High Dword 3 Bytes
MCP2H	10110b	Merge to CPU-to-PCI Write High Dword 2 Bytes
MCP1H	10101b	Merge to CPU-to-PCI Write High Dword 1 Byte
LCPRAD	00001b	Latch CPU-to-PCI Read Address
DPRA	11000b	Drive Address from PCI A/D Latch to CPU Address Bus
DPWA	11001b	Drive Address from PCI-to-Memory Write Buffer to CPU Address Bus
ADCPY	11101b	Address to Data Copy in the LBX
DACPYH	11011b	Data to Address Copy in the LBX High Dword
DACPYL	11010b	Data to Address Copy in the LBX Low Dword
PSCD	01111b	Post Special Cycle Data
DRVFF	11110b	Drive FF..FF (All 1's) onto the Host Data Bus
PCPWHC	00011b	Post to CPU-to-PCI Write High Dword Configuration

NOTE:
All other patterns are reserved.

NOPC No Operation is performed on the host bus by the LBX hence it tri-states its host bus drivers.

CMR This command effectively drives DRAM data onto the host data bus. The LBX acts as a transparent latch in this mode, depending on MDLE for latch control. With the MDLE signal high the CMR command will cause the LBXs to buffer memory data onto the host bus. When MDLE is low. The LBX will drive onto the host bus whatever memory data that was latched when MDLE was negated.

CPRF This command reads the first Dword of the CPU-to-PCI read prefetch buffer. The read pointer of the FIFO is set to point to the first Dword. The Dword is driven onto the high and low halves of the host data bus.

CPRA This command increments the read pointer of the CPU-to-PCI read prefetch buffer FIFO and drives that Dword onto the host bus when it is driven after a CPRF or CPRB command. If driven after another CPRA command, the LBX drives the current Dword while the read pointer of the FIFO is not incremented. The Dword is driven onto the upper and lower halves of the host data bus.

CPRB This command increments the read pointer of the CPU-to-PCI read prefetch buffer FIFO and drives that Dword onto the host bus when it is driven after a CPRA command. If driven after another CPRB command, the LBX drives the current Dword while the read pointer of the FIFO is not incremented. The Dword is driven onto the upper and lower halves of the host data bus.

CPRQ This command drives the first Dword stored in the CPU-to-PCI read prefetch buffer onto the lower half of the host data bus, and drives the second Dword onto the upper half of the host data bus, regardless of the state of the read pointer. The read pointer is not affected by this command.

SWB0 This command stores a Qword from the host data lines into location 0 of the PCI-to-Memory Read Buffer. Parity is either generated for the data or propagated from the host bus based on the state of the PPOUT signals sampled at the negation of RESET when the LBXs were initialized.

SWB1 This command, (similar to SWB0), stores a Qword from the host data lines into location 1 of the PCI-to-Memory Read Buffer. Parity is either generated from the data or propagated from the host bus based on the state of the PPOUT signal sampled at the falling edge of RESET.

SWB2 This command, (similar to SWB0), stores a Qword written back from the first or second level cache into location 2 of the PCI-to-memory read buffer. Parity is either generated from the data or propagated from the host bus based on the state of the PPOUT signal sampled at the falling edge of RESET.

SWB3 This command stores a Qword from the host data lines into location 3 of the PCI-to-Memory Read Buffer. Parity is either generated for the data or propagated from the host bus based on the state of the PPOUT signal sampled at the falling edge of RESET.

PCMWQ This command posts one Qword of data from the host data lines to CPU-to-Memory Write Buffer in case of a CPU memory write or a write-back from the second level cache.

PCMWFQ If the PCI Memory read address leads to a hit on a modified line in the first level cache, then a write-back is scheduled and this data has to be written into the CPU-to-Memory Write Buffer and PCI-to-Memory Read Buffer at the same time. The write-back of the first Qword is done by this command to both the buffers.

PCMWNQ This command follows the previous command to store or post subsequent write-back Qwords.

PCPWL This command posts the low Dword of a CPU-to-PCI write. The CPU-to-PCI Write Buffer stores a Dword of PCI address for every Dword of data. Hence, this command also stores the address of the Low Dword in the address location for the data. Address bit 2 (A2) is not stored directly. This command assumes a value of 0 for A2 and this is what is stored.

MCP3L This command merges the 3 most significant bytes of the low Dword of the host data bus into the last Dword posted to the CPU-to-PCI write buffer. The address is not modified.

MCP2L This command merges the 2 most significant bytes of the low Dword of the host data bus into the last Dword posted to the CPU-to-PCI write buffer. The address is not modified.

MCP1L This command merges the most significant byte of the low Dword of the host data bus into the last Dword posted to the CPU-to-PCI write buffer. The address is not modified.

PCPWH This command posts the upper Dword of a CPU-to-PCI write, with its address, into the address location. Hence, to do a Qword write PCPWL has to be followed by a PCPWH. Address bit 2 (A2) is not stored directly. This command forces a value of 1 for A2 and this is what is stored.

MCP3H This command merges the 3 most significant bytes of the high Dword of the host data bus into the last Dword posted to the CPU-to-PCI Write Buffer. The address is not modified.

MCP2H This command merges the 2 most significant bytes of the high Dword of the host data bus into the last Dword posted to the CPU-to-PCI Write Buffer. The address is not modified.

MCP1H This command merges the most significant byte of the high Dword of the host data bus into the last Dword posted to the CPU-to-PCI Write Buffer. The address is not modified.

LCPRAD This command latches the host address to drive on PCI for a CPU-to-PCI read. It is necessary to latch the address in order to drive inquire addresses on the host address bus before the CPU address is driven onto PCI.

DPRA The PCI memory read address is latched in the PCI A/D latch by a PIG command LCPRAD, this address is driven onto the host address bus by DPRA. Used in PCI to memory read transaction.

DPWA The DPWA command drives the address of the current PCI Master Write Buffer onto the host address bus. This command is potentially driven for multiple cycles. When it is no longer driven, the read pointer will increment to point to the next buffer, and a subsequent DPWA command will read the address from that buffer.

ADCPY This command drives the host data bus with the host address. The address is copied on the high and low halves of the Qword data bus; i.e. A[31:0] is copied onto D[31:0] and D[63:32]. This command is used when the CPU writes to the PCMC configuration registers.

DACPYH This command drives the host address bus with the high Dword of host data. This command is used when the CPU writes to the PCMC configuration registers.

DACPYL This command drives the host address bus with the low Dword of host data. This command is used when the CPU writes to the PCMC configuration registers.

PSCD This command is used to post the value of the Special Cycle code into the CPU-to-PCI Posted Write Buffer. The value is driven onto the A[31:0] lines by the PCMC, after acquiring the address bus by asserting AHOLD. The value on the A[31:0] lines is posted into the DATA location in the CPU-to-PCI Posted Write Buffer.

DRVFF This command causes the LBX to drive all "1s" (i.e. FFFFFFFFh) onto the host data bus. It is used for CPU reads from PCI that terminate with master abort.

PCPWHC This command posts the high half of the CPU data bus. The LBXs post the high half of the data bus even if A2 from the PCMC is low. This command is used during configuration writes when using PCI configuration access mechanism #1.

3.2.2 MEMORY INTERFACE GROUP: MIG[2:0]

The Memory Interface commands are shown in Table 2. These commands are issued by the DRAM controller of the PCMC to perform the following functions:

- Retires data from CPU-to-Memory Write Buffer to DRAM.
- Stores data into PCI-to-Memory Read Buffer when the PCI read address is targeted to DRAM.
- Retires PCI-to-Memory Write Buffer to DRAM.

Table 2. MIG Commands

Command	Code	Description
NOPM	000b	No Operation on Memory Bus
PMRFQ	001b	Place into PCI-to-Memory Read Buffer First Qword
PMRNQ	010b	Place into PCI-to-Memory Read Buffer Next Qword
RCMWQ	100b	Retire CPU-to-Memory Write Buffer Qword
RPMWQ	101b	Retire PCI-to-Memory Write Buffer Qword
RPMWQS	110b	Retire PCI-to-Memory Write Buffer Qword Shifted
MEMDRV	111b	Drive Latched Data Onto Memory Bus for 1 Clock Cycle

NOTE:
All other patterns are reserved.

NOPMN Operation on the memory bus. The LBX tri-states its drivers driving the memory bus.

PMRFQ The PCI-to-Memory read address targets memory if there is a miss on first and second caches. This command stores the first Qword of data starting at the first location in the buffer. This buffer is 8 Dwords or 1 cache line deep.

PMRNQ This command stores subsequent Qwords from memory starting at the next available location in the PCI-to-Memory Read Buffer. It is always used after PMRFQ.

RCMWQ This command retires one Qword from the CPU-to-Memory Write Buffer to DRAM. The address is stored in the address queue for this buffer in the PCMC.

RPMWQ This command retires one Qword of data from one line of the PCI-to-Memory write buffer to DRAM. When all the valid data in one buffer is retired, the next RPMWQ (or RPMWQS) will read data from the next buffer.

RPMWQS This command retires one Qword of data from one line of PCI-to-Memory write buffer to DRAM. For this command the data in the buffer is shifted by one Dword (Dword in position 0 is shifted to 1, 1 to 2 etc.). This is because the address targeted by the first Dword of the write could be an odd Dword (i.e., address bit[2] is a 1). To retire a misaligned line this command has to be used for all the data in the buffer. When all the valid data in one buffer is retired, the next RPMWQ (or RPMWQS) will read data from the next buffer.

MEMDRV For a memory write operation the data on the memory bus is required for more than one clock cycle hence all DRAM retires are latched and driven to the memory bus in subsequent cycles by this command.

3.2.3 PCI INTERFACE GROUP: PIG[3:0]

The PCI Interface commands are shown in Table 3. These commands are issued by the PCI master/slave interface of the PCMC to perform the following functions:

- Slave posts address and data to PCI-to-Memory Write Buffer.
- Slave sends PCI-to-Memory read data on the AD bus.
- Slave latches PCI master memory address so that it can be gated to the host address bus.
- Master latches CPU-to-PCI read data from the AD bus.
- Master retires CPU-to-PCI write buffer.
- Master sends CPU-to-PCI address to the AD bus.

The PCI AD[31:0] lines are driven by asserting the signal DRVPCI. This signal is used for both master and slave transactions.

Parity is calculated on either the value being driven onto PCI or the value being received on PCI, depending on the command. In Table 3, the PAR column has been included to indicate the value that the PPOUT signals are based on. An "I" indicates that the PPOUT signals reflect the parity of the AD lines as inputs to the LBX. An "O" indicates that the PPOUT signals reflect the value being driven on the PCI AD lines. See Section 3.3.4 for the timing relationship between the PIG[3:0] command, the AD[31:0] lines, and the PPOUT signals.

Table 3. PIG Commands

Command	Code	PAR	Description
PPMWA	1000b	I	Post to PCI-to-Memory Write Buffer Address
PPMWD	1001b	I	Post to PCI-to-Memory Write Buffer Data
SPMRH	1101b	O	Send PCI Master Read Data High Dword
SPMRL	1100b	O	Send PCI Master Read Data Low Dword
SPMRN	1110b	O	Send PCI Master Read Data Next Dword
LCPRF	0000b	I	Latch CPU Read from PCI into Read Prefetch Buffer First Dword
LCPRA	0001b	I	Latch CPU Read from PCI into Prefetch Buffer Next Dword, A Toggle
LCPRB	0010b	I	Latch CPU Read from PCI into Prefetch Buffer Next Dword, B Toggle
DCPWA	0100b	O	Drive CPU-to-PCI Write Buffer Address
DCPWD	0101b	O	Drive CPU-to-PCI Write Buffer Data
DCPWL	0110b	O	Drive CPU-to-PCI Write Buffer Last Data
DCCPD	1011b	O	Discard Current CPU-to-PCI Write Buffer Data
BCPWR	1010b	O	Backup CPU-to-PCI Write Buffer for Retry
SCPA	0111b	O	Send CPU-to-PCI Address
LPMA	0011b	I	Latch PCI Master Address

NOTE:
All other patterns are reserved.

PPMWA This command selects a new buffer and places the PCI master address latch value into the address register for that buffer. The next PPMWD command posts write data in the first location of this newly selected buffer. This command also causes the EOL logic to decrement the count of Dwords remaining in the line.

PPMWD This command stores the value in the AD latch into the next data location in the currently selected buffer. This command also causes the EOL logic to decrement the count of Dwords remaining in the line.

SPMRH This command sends the high order Dword from the first Qword of the PCI-to-Memory Read Buffer onto PCI. This command also causes the EOL logic to decrement the count of Dwords remaining in the line.

SPMRL This command sends the low order Dword from the first Qword of the PCI-to-Memory Read Buffer onto PCI. This command also selects the Dword alignment for the transaction and causes the EOL logic to decrement the count of Dwords remaining in the line.

SPMRN This command sends the next Dword from the PCI-to-Memory Read Buffer onto PCI. This command also causes the EOL logic to decrement the count of Dwords remaining in the line. This command is used for the second and all subsequent Dwords of the current transaction.

LCPRF This command acquires the value of the AD[31:0] lines into the first location in the CPU-to-PCI Read Prefetch Buffer until a different command is driven.

LCPRA When driven after a LCPRF or LCPRB command, this command latches the value of the AD[31:0] lines into the next location into the CPU-to-PCI Read Prefetch Buffer. When driven after another LCPRA command, this command latches the value on AD[31:0] into the same location in the CPU-to-PCI Read Prefetch Buffer, overwriting the previous value.

LCPRB When driven after a LCPRA command, this command latches the value of the AD[31:0] lines into the next location into the CPU-to-PCI Read Prefetch Buffer. When driven after another LCPRB command, this command latches the value on AD[31:0] into the same location in the CPU-to-PCI Read Prefetch Buffer, overwriting the previous value.

DCPWA This command drives the next address in the CPU-to-PCI Write Buffer onto PCI. The read pointer of the FIFO is not incremented.

DCPWD This command drives the next data Dword in the CPU-to-PCI Write Buffer onto PCI. The read pointer of the FIFO is incremented on the next PCLK if TRDY# is asserted.

DCPWL This command drives the previous data Dword in the CPU-to-PCI Write Buffer onto PCI. This is the data which was driven by the last DCPWD command. The read pointer of the FIFO is not incremented.

DCCPD This command discards the current Dword in the CPU-to-PCI Write Buffer. This is used to clear write data when the write transaction terminates with master abort, where TRDY# is never asserted.

BCPWR For this command the CPU-to-PCI Write Buffer is "backed up" one entry such that the address/data pair last driven with the DCPWA and DCPWD commands will be driven again on the AD[31:0] lines when the commands are driven again. This command is used when the target has retried the write cycle.

SCPA This command drives the value on the host address bus onto PCI.

LPMA This command stores the previous AD[31:0] value into the PCI master address latch. If the EOL logic determines that the requested Dword is the last Dword of a line, then the EOL signal will be asserted; otherwise the EOL signal will be negated.

3.3 LBX Timing Diagrams

This section describes the timing relationship between the LBX control signals and the interface buses.

3.3.1 HIG[4:0] COMMAND TIMING

The commands driven on HIG[4:0] can cause the host address bus and/or the host data bus to be driven and latched. The following timing diagram illustrates the timing relationship between the driven command and the buses. The "host bus" in Figure 4 could be address and/or data.

Note that the Drive command takes two cycles to drive the host data bus, but only one to drive the address. When the NOPC command is sampled, the LBX takes only one cycle to release the host bus.

The Drive commands in Figure 4 are any of the following:

CMR	**CPRF**	**CPRA**	**CPRB**
CPRQ	**DPRA**	**DPWA**	**ADCPY**
DACPYH	**DACPYL**	**DRVFF**	

The Latch command in Figure 4 is any of the following:

SWB0	**SWB1**	**SWB2**	**SWB3**
PCMWQ	**PCMWFQ**	**PCMWNQ**	**PCPWL**
MCP3L	**MCP2L**	**MCP1L**	**PCPWH**
MCP3H	**MCP2H**	**LCPRAD**	**PSCD**

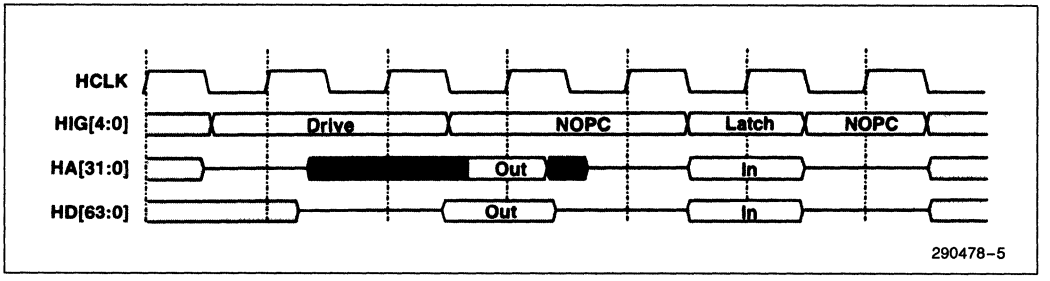


Figure 4. HIG[4:0] Command Timing

3.3.2 HIG[4:0] MEMORY READ TIMING

Figure 5 illustrates the timing relationship between the HIG[4:0], MIG[2:0], CAS[7:0]#, and MDLE signals for DRAM memory reads. The delays shown in the diagram do not represent the actual AC timings, but are intended only to show how the delay affects the sequencing of the signals.

When the CPU is reading from DRAM, the HIG[4:0] lines are driven with the CMR command that causes the LBX to drive memory data onto the HD bus. Until the MD bus is valid, the HD bus is driven with invalid data. When CAS[7:0]# assert, the MD bus becomes valid after the DRAM CAS[7:0]# access time. The MD and MP lines are directed through a synchronous register inside the LBX to the HD and HP lines. MDLE acts as a clock enable for this register. When MDLE is asserted, the LBX samples the MD and MP lines. When MDLE is negated, the MD and HD register retains its current value.

The LBX releases the HD bus based on sampling the NOPC command on the HIG[4:0] lines and MDLE being asserted. By delaying the release of the HD bus until MDLE is asserted, the LBX provides hold time for the data with respect to the write enable strobes (CWE[7:0]#) of the second level cache.

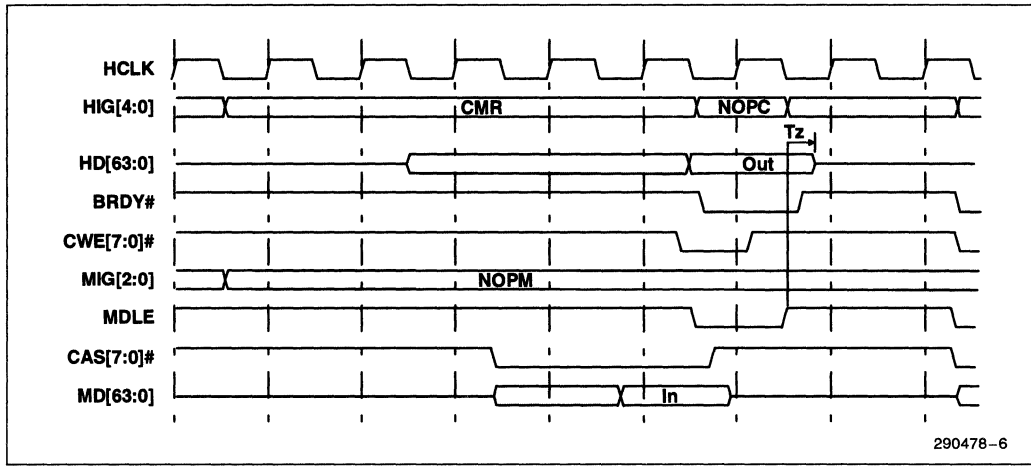


Figure 5. CPU Read from Memory

3.3.3 MIG[2:0] COMMAND

Figure 6 illustrates the timing of the MIG[2:0] commands with respect to the MD bus, CAS[7:0]#, and WE#. Figure 6 shows the MD bus transitioning from a read to a write cycle.

The Latch command in Figure 6 is any of the following:
PMRFQ PMRNQ

The Retire command in Figure 6 is any of the following:
RCMWQ RPMWQ RPMWQS

The data on the MD bus is sampled at the end of the first cycle into the LBX based on sampling the Latch command. The CAS[7:0]# signals can be negated in the next cycle. The WE# signal is asserted in the next cycle. The required delay between the assertion of WE# and the assertion of CAS[7:0]# means that the MD bus has 2 cycles to turn around; hence the NOPM command driven in the second clock. The LBX starts to drive the MD bus based on sampling the Retire command at the end of the third clock. After the Retire command is driven for 1 cycle, the data is held at the output by the MEMDRV command. The LBX releases the MD bus based on sampling the NOPM command at the end of the sixth clock.

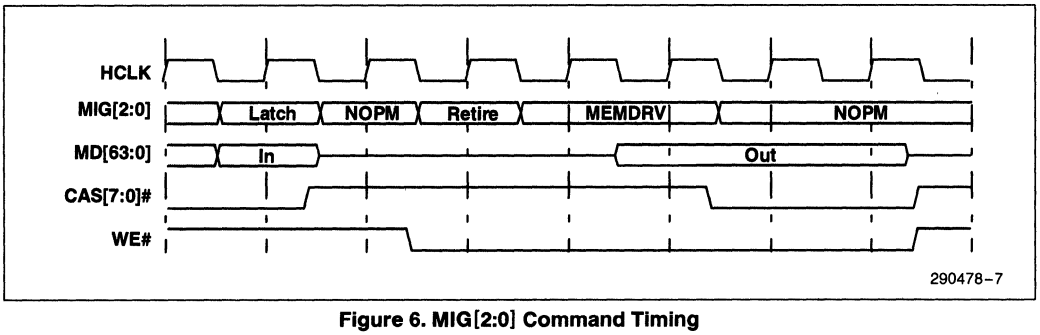


Figure 6. MIG[2:0] Command Timing

3.3.4 PIG[3:0] COMMAND, DRVPCI, AND PPOUT TIMING

Figure 7 illustrates the timing of the PIG[3:0] commands, the DRVPCI signal, and the PPOUT[1:0] signal relative to the PCI AD[31:0] lines.

The Drive commands in Figure 7 are any of the following:
SPMRH SPMRL SPMRN
DCPWA DCPWD DCPWL
SCPA

The Latch commands in Figure 7 are any of the following:
PPMWA PPMWD LPMA

The following commands do not fit in either category, although they function like Latch type commands with respect to the PPOUT[1:0] signals. They are described in Section 3.3.5.

LCPRF LCPRA LCPRB

The DRVPCI signal is driven synchronous to the PCI bus, enabling the LBXs to initiate driving the PCI AD[31:0] lines one clock after DRVPCI is asserted. As shown in Figure 7, if DRVPCI is asserted in cycle N, the PCI AD[31:0] lines are driven in cycle N+1. The negation of the DRVPCI signal causes the LBXs to asynchronously release the PCI bus, enabling the LBXs to cease driving the PCI AD[31:0] lines in the same clock that DRVPCI is negated. As shown in Figure 7, if DRVPCI is negated in cycle N, the PCI AD[31:0] lines are released in cycle N.

PCI address and data parity is available at the LBX interface on the PPOUT lines from the LBX. The parity for data flow from PCI to LBX is valid 1 clock cycle after data on the AD bus. The parity for data flow from LBX to PCI is valid in the same cycle as the data. When the AD[31:0] lines transition from input to output, there is no conflict on the parity lines due to the dead cycle for bus turnaround. This is illustrated in the sixth and seventh clock of Figure 7.

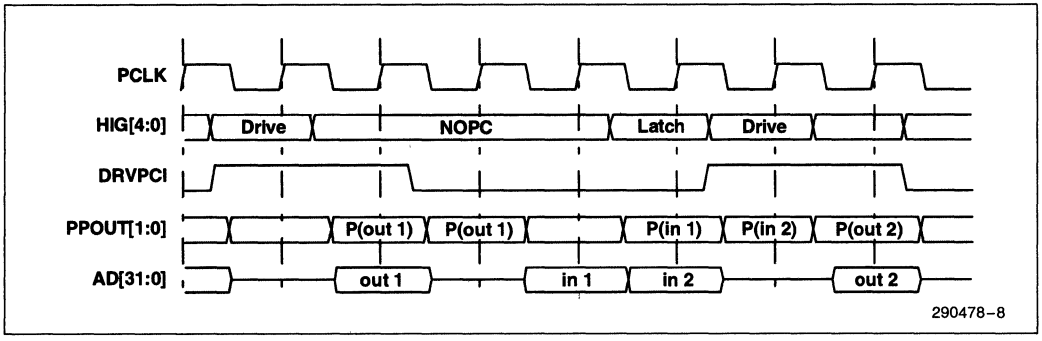


Figure 7. PIG[3:0] Command Timing

3.3.5 PIG[3:0]: READ PREFETCH BUFFER COMMAND TIMING

The structure of the CPU-to-PCI read prefetch buffer requires special considerations due to the partition of the PCMC and LBX. The PCMC interfaces only to the PCI control signals, while the LBXs interface only to the data. Therefore, it is not possible to latch a Dword of data into the prefetch buffer after it is qualified by TRDY#. Instead, the data is repetitively latched into the same location until TRDY# is sampled asserted. Only after TRDY# is sampled asserted is data valid in the buffer. A toggling mechanism is implemented to advance the write pointer to the next Dword after the current Dword has been qualified by TRDY#.

Other considerations of the partition are taken into account on the host side as well. When reading from the buffer, the command to drive the data onto the host bus is sent before it is known that the entry is valid. This method avoids the wait-state that would be introduced by waiting for an entry's TRDY# to be asserted before sending the command to drive the entry onto the host bus. The FIFO structure of the buffer also necessitates a toggling scheme to advance to the next buffer entry after the current entry has been successfully driven. Also, this method gives the LBX the ability to drive the same Dword twice, enabling reads of less than a Dword to be serviced by the buffer; reads of individual bytes of a Dword would read the same Dword 4 times.

The HIG[4:0] and PIG[3:0] lines are defined to enable the features described previously. The LCPRF PIG[3:0] command latches the first PCI read Dword into the first location in the CPU-to-PCI read prefetch buffer. This command is driven until TRDY# is sampled asserted. The valid Dword would then be in the first location of the buffer. The cycle after TRDY# is sampled asserted, the PCMC drives the LCPRA command on the PIG[3:0] lines. This action latches the value on the PCI AD[31:0] lines into the *next* Dword location in the buffer. Again, the LCPRA command is driven until TRDY# is sampled asserted. Each cycle the LCPRA command is driven, data is latched into the same location in the buffer. When TRDY# is sampled asserted, the PCMC drives the LCPRB command on the PIG[3:0] lines. This latches the value on the AD[31:0] lines into the next location in the buffer, the one *after* the location that the previous LCPRA command latched data into. After TRDY# has been sampled asserted again, the command switches back to LCPRA. In this way, the same location in the buffer can be filled repeatedly until valid, and when it is known that the location is valid, the next location can be filled.

The commands for the HIG[4:0], CPRF, CPRA, and CPRB, work exactly the same way. If the same command is driven, the same data is driven. Driving an appropriately different command results in the next data being driven. Figure 8 illustrates the usage of these commands.

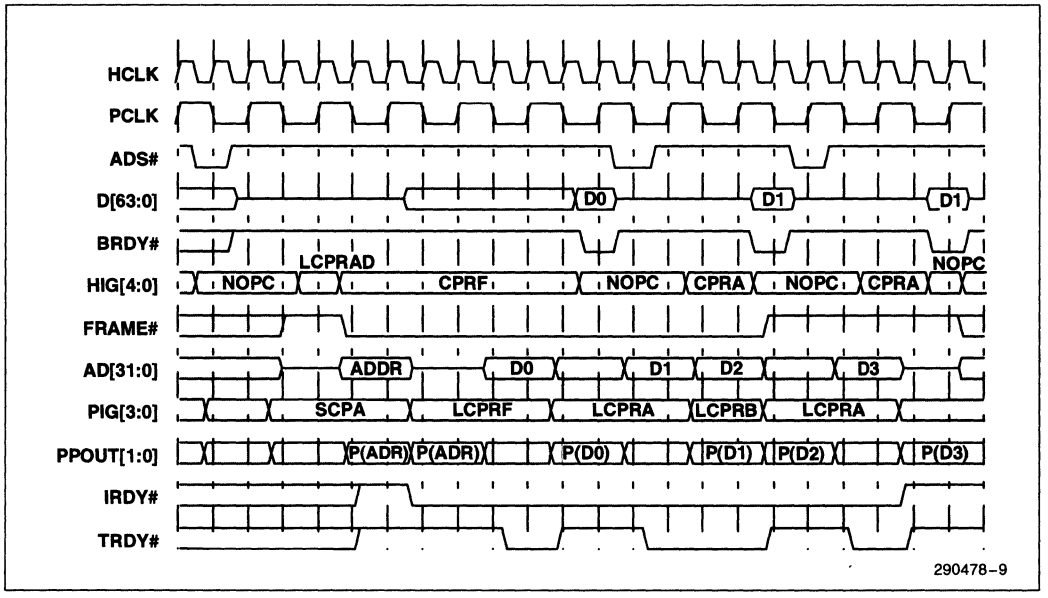


Figure 8. PIG[3:0] CPU-to-PCI Read Prefetch Buffer Commands

Figure 8 shows an example of how the PIG commands function on the PCI side. The LCPRF command is driven on the PIG[3:0] lines until TRDY# is sampled asserted at the end of the fifth PCI clock. The LCPRA command is then driven until TRDY# is again sampled asserted at the end of the seventh PCI clock. TRDY# is sampled asserted again so LCPRB is driven only once. Finally, LCPRA is driven again until the last TRDY# is asserted at the end of the tenth PCI clock. In this way, 4 Dwords are latched in the read CPU-to-PCI prefetch buffer.

Figure 8 also shows an example of how the HIG commands function on the host side of the LBX. Two clocks after sampling the CPRF command, the LBX drives the host data bus. The data takes two cycles to become stable. The first data driven in this case is invalid, since the data has not arrived on PCI. The data driven on the host bus changes in the seventh host clock, since the LCPRF command has been driven on the PIG[3:0] lines the previous cycle, latching a new value into the first location of the read prefetch buffer. At this point the data is not the correct value, since TRDY# has not yet been asserted on PCI. The LCPRF command is driven again in the fifth PCI clock while TRDY# is sampled asserted at the end of this clock. The requested data for the read is then latched into the first location of the read prefetch buffer and driven onto the host data bus, becoming valid at the end of CPU clock 12. The BRDY# signal can therefore be driven asserted in this clock. The following read transaction (issued in CPU clock 15) requests the next Dword, and so the CPRA command is driven on the HIG[4:0] lines, advancing to read the next location in the read prefetch buffer. As the correct data is already there, the command is driven only once for this transaction. The next read transaction requests data in the same Dword as the previous. Therefore, the CPRA command is driven again, the buffer is not advanced, and the same Dword is driven onto the host bus.

3.3.6 PIG[3:0]: END-OF-LINE WARNING SIGNALS: EOL

When posting PCI master writes, the PCMC must be informed when the line boundary is about to be overrun, as it has no way of determining this itself (recall that the PCMC does not receive any address bits from PCI). The low order LBX determines this, as it contains the low order bits of the PCI master write address and also tracks how many Dwords of write data have been posted. Therefore, the low order LBX component sends the "end-of-line" warning to the PCMC. This is accomplished with the EOL signal driven from the low order LBX to the PCMC. Figure 9 illustrates the timing of this signal.

1. The FRAME# signal is sampled asserted in the first cycle. The LPMA command is driven on the PIG[3:0] signals to hold the address while it is being decoded (e.g. in the MEMCS# decode circuit of the 82378 SIO). The first data (D0) remains on the bus until TRDY# is asserted in response to MEMCS# being sampled asserted in the third clock.
2. The PPMWA command is driven in response to sampling MEMCS# asserted. TRDY# is asserted in this cycle indicating that D0 has been latched at the end of the fourth clock. The action of the PPMWA command is to transfer the PCI address captured in the PCI AD latch at the end of the first clock to the posting buffer, and open the PCI AD latch in order to capture the data. This data will be posted to the write buffer in the following cycle by the PPMWD command.
3. The EOL signal is first negated when the LPMA command is driven on the PIG[3:0] signals. However, if the first data Dword accepted is also the last that should be accepted, the EOL signal will be asserted in the third clock. This is the "end-of-line" indication. In this case, the EOL signal is asserted as soon as the LPMA command has been latched. The action by the PCMC in response is to negate TRDY# and assert STOP# in the fifth clock. Note that the EOL signal is asserted even before the MEMCS# signal is sampled asserted in this case. The EOL signal will remain asserted until the next time the LPMA command is driven.
4. If the second Dword is the last that should be accepted, the EOL signal will be asserted in the fifth clock to negate TRDY# and assert STOP# on the following clock. The EOL signal is asserted in response to the PPMWA command being sampled, and relies on the knowledge that TRDY# for the first Dword of data will be sampled asserted by the master in the same cycle (at the end of the fourth clock). Therefore, to prevent a third assertion of TRDY# in the sixth clock, the EOL signal must be asserted in the fifth clock.

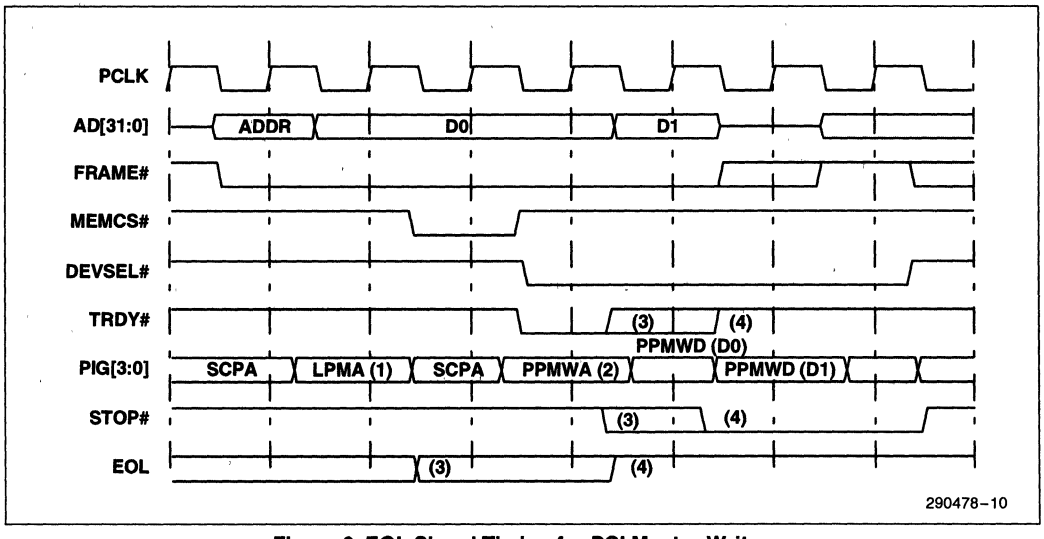


Figure 9. EOL Signal Timing for PCI Master Writes

A similar sequence is defined for PCI master reads. While it is possible to know when to stop driving read data due to the fact that the read address is latched into the PCMC before any read data is driven on PCI, the use of the EOL signal for PCI master reads simplifies the logic internal to the PCMC. Figure 10 illustrates the timing of EOL with respect to the PIG[3:0] commands to drive out PCI read data.

Note that unlike the PCI master write sequence, the STOP# signal is asserted with the last data transfer, not after.

1. The LPMA command sampled at the end of the second clock causes the EOL signal to assert if there is only one Dword left in the line, otherwise it will be negated. The first TRDY# will also be the last, and the STOP# signal will be asserted with TRDY#.
2. The SPMRH command causes the count of the number of Dwords left in the line to be decremented. If this count reaches one, the EOL signal is asserted. The next TRDY# will be the last, and STOP# is asserted with TRDY#.

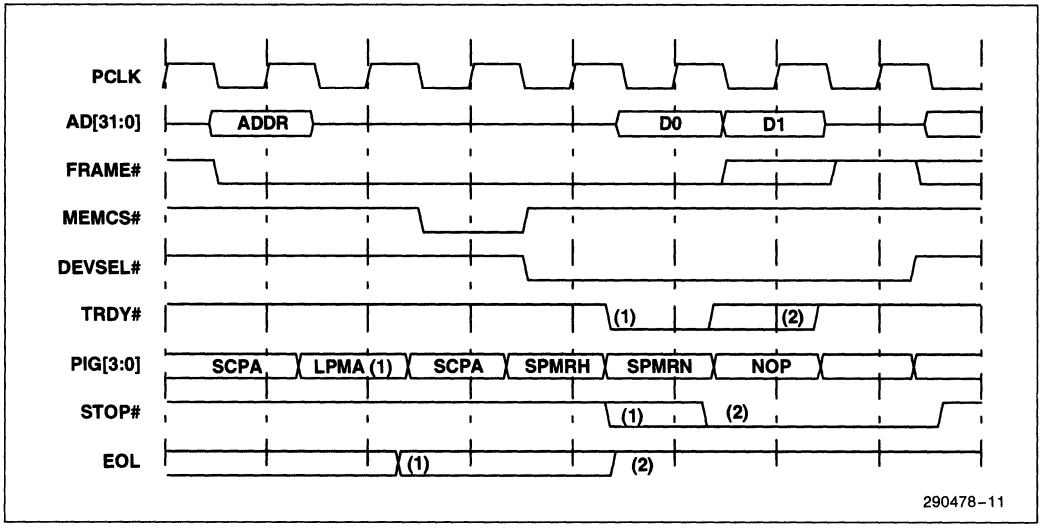


Figure 10. EOL Signal Timing for PCI Master Reads

3.4 PLL Loop Filter Components

As shown in Figure 11, loop filter components are required on the LBX components. A 4.7 KΩ 5% resistor is typically connected between pins LP1 and LP2. Pin LP2 has a path to the PLLAGND pin through a 100Ω 5% series resistor and a 0.01 μF 10% series capacitor. The ground side of capacitor C1 and the PLLVSS pin should connect to the ground plane at a common point. All PLL loop filter traces should be kept to minimal length and should be wider than signal traces. Inductor L1 is connected to the 5V power supply on both the 82433LX and 82433NX.

Some circuit boards may require filtering the power circuit to the LBX PLL. The circuit shown in Figure 11 will typically enable the LBX PLL to have higher noise immunity than without. Pin PLLVDD is connected to the 5V V_{CC} through a 10Ω 5% resistor. The PLLVDD and PLLVSS pins are bypassed with a 0.01 μF 10% series capacitor.

The high order 82433NX LBX samples A11 at the falling edge of reset to configure the LBX for PLL test mode. When A11 is sampled low, the LBX is in normal operating mode. When A11 is sampled high, the LBX drives the internal HCLK from the PLL on the EOL pin. Note that A11 on the high order LBX is connected to the A27 line on the CPU address bus. This same address line is used to put the PCMC into PLL test mode.

	Mercury 60 MHz	Mercury 66 MHz	Neptune
R1	4.7 KΩ	2.2 KΩ	4.7 KΩ
R2	100Ω	100Ω	100Ω
C2	0.01 μF	0.01 μF	0.01 μF
R3	10Ω	10Ω	10Ω
C1	0.47 μF	0.47 μF	0.47 μF
C1[1]	0.01 μF	0.01 μF	0.01 μF

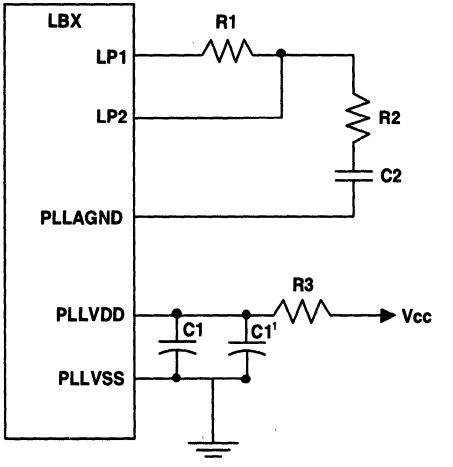


Figure 11. Loop Filter Circuit

3.5 PCI Clock Considerations

There is a 1.25 ns clock skew specification between the PCMC and the LBX that must be adhered to for proper operation of the PCMC/LBX timing. As shown in Figure 12, the PCMC drives PCLKOUT to an external clock driver which supplies copies of PCLK to PCI devices, the LBXs, and back to the PCMC. The skew specification is defined as the difference in timing between the signal that appears at the PCMC PCLKIN input pin and the signal that appears at the LBX PCLK input pin. For both the low order LBX and the high order LBX, the PCLK rising and falling edges must not be more than 1.25 ns apart from the rising and falling edge of the PCMC PCLKIN signal.

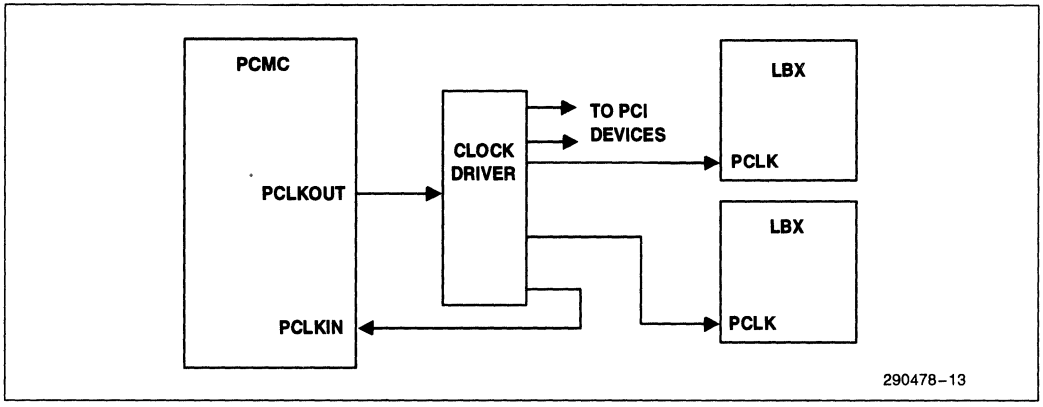


Figure 12. Clock Considerations

4.0 ELECTRICAL CHARACTERISTICS

4.1 Absolute Maximum Ratings

Table 4 lists stress ratings only. Functional operation at these maximums is not guaranteed. Functional operation conditions are given in Sections 4.2 and 4.3.

Extended exposure to the Absolute Maximum Ratings may affect device reliability.

Case Temperature under Bias .....0°C to +85°C

Storage Temperature .....−40°C to +125°C

Voltage on Any Pin
with Respect to Ground.....−0.3 to V_{CC} + 0.3V

Supply Voltage
with Respect to V_{SS} .....−0.3 to +7.0V

Maximum Power Dissipation: 1.4W (82433LX)

Maximum Total Power Dissipation .1.4W (82433NX)

Maximum Power Dissipation, V_{CC3} .....430 mW

The maximum total power dissipation in the 82433NX on the V_{CC} and V_{CC3} pins is 1.4W. The V_{CC3} pins may draw as much as 430 mW, however, total power will not exceed 1.4W.

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

**WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

4.2 Thermal Characteristics

The LBX is designed for operation at case temperatures between 0°C and 85°C. The thermal resistances of the package are given in the following tables.

Table 4. Thermal Resistance

Parameter	Air Flow Rate (Linear Feet per Minute)		
	0	400	600
θ_{JA} (°C/Watt)	51.9	37.1	34.8
θ_{JC} (°C/Watt)		10	

4.3 DC Characteristics

Host Interface Signals
A[15:0](t/s), D[31:0](t/s), HIG[4:0](in), HP[3:0](t/s)

Main Memory (DRAM) Interface Signals
MD[31:0](t/s), MP[3:0](t/s), MIG[2:0](in), MDLE(in)

PCI Interface Signals
AD[15:0](t/s), TRDY#(in), PIG[3:0](in), DRVPCI(in), EOL(t/s), PPOUT(t/s)

Reset and Clock Signals
HCLK(in), PCLK(in), RESET(in), LP1(out), LP2(in), TEST(in)

4.3.1 82433LX LBX DC CHARACTERISTICS

Functional Operating Range: V_{CC} = 4.75 V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Typical	Max	Unit	Notes
V_{IL1}	Input Low Voltage	−0.3		0.8	V	1
V_{IH1}	Input High Voltage	2.0		V_{CC} + 0.3	V	1
V_{IL2}	Input Low Voltage	−0.3		0.3 × V_{CC}	V	2
V_{IH2}	Input High Voltage	0.7 × V_{CC}		V_{CC} + 0.3	V	2
V_{OL1}	Output Low Voltage			0.4	V	3
V_{OH1}	Output High Voltage	2.4			V	3
V_{OL2}	Output Low Voltage			0.5	V	4
V_{OH2}	Output High Voltage	V_{CC} − 0.5			V	4
I_{OL1}	Output Low Current			1	mA	5
I_{OH1}	Output High Current	−1			mA	5
I_{OL2}	Output Low Current			3	mA	6
I_{OH2}	Output High Current	−2			mA	6

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C (Continued)

Symbol	Parameter	Min	Typical	Max	Unit	Notes
I_{OL3}	Output Low Current			3	mA	7
I_{OH3}	Output High Current	−1			mA	7
I_{IH}	Input Leakage Current			+10	μA	
I_{IL}	Input Leakage Current			−10	μA	
C_{IN}	Input Capacitance		4.6		pF	
C_{OUT}	Output Capacitance		4.3		pF	
$C_{I/O}$	I/O Capacitance		4.6		pF	

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: AD[15:0], A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0], TRDY#, RESET, HCLK, PCLK
2. V_{IL2} and V_{IH2} apply to the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI
3. V_{OL1} and V_{OH1} apply to the following signals: AD[15:0], A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]
4. V_{OL2} and V_{OH2} apply to the following signals: PPOUT, EOL
5. I_{OL1} and I_{OH1} apply to the following signals: PPOUT, EOL
6. I_{OL2} and I_{OH2} apply to the following signals: AD[15:0]
7. I_{OL3} and I_{OH3} apply to the following signals: A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]

4.3.2 82433NX LBX DC CHARACTERISTICS

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135 to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Typical	Max	Unit	Notes
V_{IL1}	Input Low Voltage	−0.3		0.8	V	1
V_{IH1}	Input High Voltage	2.0		V_{CC} + 0.3	V	1
V_{IL2}	Input Low Voltage	−0.3		0.3 x V_{CC}	V	2
V_{IH2}	Input High Voltage	0.7 x V_{CC}		V_{CC} + 0.3	V	2
V_{IL3}	Input Low Voltage	−0.3		0.8	V	3
V_{IH3}	Input High Voltage	2.0		V_{CC3} + 0.3	V	3
V_{OL1}	Output Low Voltage			0.4	V	4
V_{OH1}	Output High Voltage	2.4			V	4
V_{OL2}	Output Low Voltage			0.5	V	5
V_{OH2}	Output High Voltage	V_{CC} − 0.5			V	5
I_{OL1}	Output Low Current			1	mA	6
I_{OH1}	Output High Current	−1			mA	6
I_{OL2}	Output Low Current			3	mA	7
I_{OH2}	Output High Current	−2			mA	7

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C (Continued)

Symbol	Parameter	Min	Typical	Max	Unit	Notes
I_{OL3}	Output Low Current			3	mA	8
I_{OH3}	Output High Current	−1			mA	8
I_{IH}	Input Leakage Current			+10	μA	
I_{IL}	Input Leakage Current			−10	μA	
C_{IN}	Input Capacitance		4.6		pF	
C_{OUT}	Output Capacitance		4.3		pF	
$C_{I/O}$	I/O Capacitance		4.6		pF	

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: AD[15:0], MD[31:0], MP[3:0], TRDY#, RESET, HCLK, PCLK
2. V_{IL2} and V_{IH2} apply to the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI
3. V_{IL3} and V_{IH3} apply to the following signals: A[15:0], D[31:0], HP[3:0]
4. V_{OL1} and V_{OH1} apply to the following signals: AD[15:0], A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]
5. V_{OL2} and V_{OH2} apply to the following signals: PPOUT, EOL
6. I_{OL1} and I_{OH1} apply to the following signals: PPOUT, EOL
7. I_{OL2} and I_{OH2} apply to the following signals: AD[15:0]
8. I_{OL3} and I_{OH3} apply to the following signals: A[15:0], D[31:0], HP[3:0], MD[31:0], MP[3:0]
9. The output buffers for A[15:0], D[31:0] and HP[3:0] are powered with V_{CC3} and therefore drive 3.3V signal levels.

4.4 82433LX AC Characteristics

The AC specifications given in this section consist of propagation delays, valid delays, Input setup requirements, input hold requirements, output float delays, output enable delays, clock high and low times and clock period specifications. Figure 13 through Figure 21 define these specifications. Sections 4.3.1 through 4.3.3 list the AC Specifications.

In Figure 13 through Figure 21 VT = 1.5V for the following signals: MD[31:0], MP[3:0], D[31:0], HP[3:0], A[15:0], AD[15:0], TRDY#, HCLK, PCLK, RESET, TEST.

VT = 2.5V for the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI, PPOUT, EOL.

4.4.1 HOST AND PCI CLOCK TIMING, 66 MHZ (82433LX)

Functional Operating Range: V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLK Period	15	20	18	
t1b	HCLK High Time	5		18	
t1c	HCLK Low Time	5		18	
t1d	HCLK Rise Time		1.5	19	
t1e	HCLK Fall Time		1.5	19	
t1f	HCLK Period Stability		±100		ps[1]
t2a	PCLK Period	30		18	
t2b	PCLK High Time	12		18	
t2c	PCLK Low Time	12		18	
t2d	PCLK Rise Time		3	19	
t2e	PCLK Fall Time		3	19	
t3	HCLK to PCLK Skew	−7.2	5.8	21	

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5 Volts.

4.4.2 COMMAND TIMING, 66 MHZ (82433LX)

Functional Operating Range: V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C

Symbol	Parameter	Min	Max	Figure	Notes
t10a	HIG[4:0] Setup Time to HCLK Rising	5.4		15	
t10b	HIG[4:0] Hold Time from HCLK Rising	0		15	
t11a	MIG[2:0] Setup Time to HCLK Rising	5.4		15	
t11b	MIG[2:0] Hold Time from HCLK Rising	0		15	
t12a	PIG[3:0] Setup Time to PCLK Rising	15.6		15	
t12b	PIG[3:0] Hold Time from PCLK Rising	−1.0		15	
t13a	MDLE Setup Time to HCLK Rising	5.7		15	
t13b	MDLE Hold Time to HCLK Rising	−0.3		15	
t14a	DRVPCI Setup Time to PCLK Rising	6.5		15	
t14b	DRVPCI Hold Time from PCLK Rising	−0.5		15	
t15a	RESET Setup Time to HCLK Rising	3.1		15	
t15b	RESET Hold Time from HCLK Rising	0.3		15	

4.4.3 ADDRESS, DATA, TRDY#, EOL, TEST, TSCON AND PARITY TIMING, 66 MHz (82433LX)

Functional Operating Range: V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C

Symbol	Parameter	Min	Max	Figure	Notes
t20a	AD[15:0] Output Enable Delay from PCLK Rising	2		17	
t20b	AD[15:0] Valid Delay from PCLK Rising	2	11	14	1
t20c	AD[15:0] Setup Time to PCLK Rising	7		15	
t20d	AD[15:0] Hold Time from PCLK Rising	0		15	
t20e	AD[15:0] Float Delay from DRVPCI Falling	2	10	16	
t21a	TRDY# Setup Time to PCLK Rising	7		15	
t21b	TRDY# Hold Time from PCLK Rising	0		15	
t22a	D[31:0], HP[3:0] Output Enable Delay from HCLK Rising	0	7.7	17	2
t22b	D[31:0], HP[3:0] Float Delay from HCLK Rising	3.1	15.5	16	
t22c	D[31:0], HP[3:0] Float Delay from MDLE Rising	2	11.0	16	3
t22d	D[31:0], HP[3:0] Valid Delay from HCLK Rising	0	7.7	14	2
t22e	D[31:0], HP[3:0] Setup Time to HCLK Rising	3.0		15	
t22f	D[31:0], HP[3:0] Hold Time from HCLK Rising	0.3		15	
t23a	HA[15:0] Output Enable Delay from HCLK Rising	0	15.2	17	
t23b	HA[15:0] Float Delay from HCLK Rising	0	15.2	16	
t23c	HA[15:0] Valid Delay from HCLK Rising	0	16	14	7
t23cc	HA[15:0] Valid Delay from HCLK Rising	0	14.5		8
t23d	HA[15:0] Setup Time to HCLK Rising	15		15	4
t23e	HA[15:0] Setup Time to HCLK Rising	4.1		15	5
t23f	HA[15:0] Hold Time from HCLK Rising	0.3		15	
t24a	MD[31:0], MP[3:0] Valid Delay from HCLK Rising	0	12.0	14	6
t24b	MD[31:0], MP[3:0] Setup Time to HCLK Rising	4.0		15	
t24c	MD[31:0], MP[3:0] Hold Time from HCLK Rising	0.4		15	
t25	EOL, PPOUT Valid Delay from PCLK Rising	2.3	17.2	14	2
t26a	All Outputs Float Delay from TSCON Falling	0	30	16	
t26b	All Outputs Enable Delay from TSCON Rising	0	30	17	

NOTES:
1. Min: 0 pF, Max: 50 pF
2. 0 pF
3. When NOPC command sampled on previous rising HCLK on HIG[4:0]
4. CPU to PCI Transfers
5. When ADCPY command is sampled on HIG[4:0]
6. 50 pF
7. When DACPYL or DACPYH commands are sampled on HIG[4:0]
8. Inquire cycle

4.4.4 HOST AND PCI CLOCK TIMING, 60 MHz (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLK Period	16.6	20	18	
t1b	HCLK High Time	5.5		18	
t1c	HCLK Low Time	5.5		18	
t1d	HCLK Rise Time		1.5	19	
t1e	HCLK Fall Time		1.5	19	
t1f	HCLK Period Stability		±100		ps[1]
t2a	PCLK Period	33.33		18	
t2b	PCLK High Time	13		18	
t2c	PCLK Low Time	13		18	
t2d	PCLK Rise Time		3	19	
t2e	PCLK Fall Time		3	19	
t3	PCLK to PCMC PCLKIN: Input to Input Skew	−7.2	5.8	21	

NOTES:
1. Measured on rising edge of adjacent clocks at 1.5 Volts

4.4.5 COMMAND TIMING, 60 MHZ (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t10a	HIG[4:0] Setup Time to HCLK Rising	6.0		15	
t10b	HIG[4:0] Hold Time from HCLK Rising	0		15	
t11a	MIG[2:0] Setup Time to HCLK Rising	6.0		15	
t11b	MIG[2:0] Hold Time from HCLK Rising	0		15	
t12a	PIG[3:0] Setup Time to PCLK Rising	16.0		15	
t12b	PIG[3:0] Hold Time from PCLK Rising	0		15	
t13a	MDLE Setup Time to HCLK Rising	5.9		15	
t13b	MDLE Hold Time to HCLK Rising	−0.3		15	
t14a	DRVPCI Setup Time to PCLK Rising	7.0		15	
t14b	DRVPCI Hold Time from PCLK Rising	−0.5		15	
t15a	RESET Setup Time to HCLK Rising	3.4		15	
t15b	RESET Hold Time from HCLK Rising	0.4		15	

4.4.6 ADDRESS, DATA, TRDY#, EOL, TEST, TSCON AND PARITY TIMING, 60 MHz (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t20a	AD[15:0] Output Enable Delay from PCLK Rising	2		17	
t20b	AD[15:0] Valid Delay from PCLK Rising	2	11	14	1
t20c	AD[15:0] Setup Time to PCLK Rising	7		15	
t20d	AD[15:0] Hold Time from PCLK Rising	0		15	
t20e	AD[15:0] Float Delay from DRVPCI Falling	2	10	16	
t21a	TRDY# Setup Time to PCLK Rising	7		15	
t21b	TRDY# Hold Time from PCLK Rising	0		15	
t22a	D[31:0], HP[3:0] Output Enable Delay from HCLK Rising	0	7.9	17	2
t22b	D[31:0], HP[3:0] Float Delay from HCLK Rising	3.1	15.5	16	
t22c	D[31:0], HP[3:0] Float Delay from MDLE Rising	2	11.0	16	3
t22d	D[31:0], HP[3:0] Valid Delay from HCLK Rising	0	7.8	14	2
t22e	D[31:0],HP[3:0] Setup Time to HCLK Rising	3.4		15	
t22f	D[31:0], HP[3:0] Hold Time from HCLK Rising	0.3		15	
t23a	HA[15:0] Output Enable Delay from HCLK Rising	0	15.2	17	
t23b	HA[15:0] Float Delay from HCLK Rising	0	15.2	16	
t23c	HA[15:0] Valid Delay from HCLK Rising	0	18.5	14	7
t23cc	HA[15:0] Valid Delay from HCLK Rising	0	15.5		8
t23d	HA[15:0] Setup Time to HCLK Rising	15.0		15	4
t23e	HA[15:0] Setup Time to HCLK Rising	4.1		15	5
t23f	HA[15:0] Hold Time from HCLK Rising	0.3		15	
t24a	MD[31:0], MP[3:0] Valid Delay from HCLK Rising	0	12.0	14	6
t24b	MD[31:0], MP[3:0] Setup Time to HCLK Rising	4.4		15	
t24c	MD[31:0], MP[3:0] Hold Time from HCLK Rising	1.0		15	
t25	EOL, PPOUT Valid Delay from PCLK Rising	2.3	17.2	14	2
t26a	All Outputs Float Delay from TSCON Falling	0	30	16	
t26b	All Outputs Enable Delay from TSCON Rising	0	30	17	

NOTES:
1. Min: 0 pF, Max: 50 pF
2. 0 pF
3. When NOPC command sampled on previous rising HCLK on HIG[4:0]
4. CPU to PCI Transfers
5. When ADCPY command is sampled on HIG[4:0]
6. 50 pF
7. When DACPYL or DACPYH commands are sampled on HIG[4:0]
8. Inquire cycle

4.4.7 TEST TIMING (82433LX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t30	All Test Signals Setup Time to HCLK/PCLK Rising	10.0			In PLL Bypass Mode
t31	All Test Signals Hold Time to HCLK/PCLK Rising	12.0			In PLL Bypass Mode
t32	Test Setup Time to HCLK/PCLK Rising	15.0		15	
t33	Test Hold Time to HCLK/PCLK Rising	5.0		15	
t34	PPOUT Valid Delay from PCLK Rising	0.0	500	15	In PLL Bypass Mode

2

4.5 82433NX AC Characteristics

The AC specifications given in this section consist of propagation delays, valid delays, input setup requirements, input hold requirements, output float delays, output enable delays, clock high and low times and clock period specifications. Figure 13 through Figure 21 define these specifications. Section 4.5 lists the AC Specifications.

In Figure 13 through Figure 21 VT = 1.5V for the following signals: MD[31:0], MP[3:0], D[31:0], HP[3:0], A[15:0], AD[15:0], TRDY#, HCLK, PCLK, RESET, TEST.

VT = 2.5V for the following signals: HIG[4:0], PIG[3:0], MIG[2:0], MDLE, DRVPCI, PPOUT, EOL.

4.5.1 HOST AND PCI CLOCK TIMING, (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLK Period	15	20	18	
t1b	HCLK High Time	5		18	
t1c	HCLK Low Time	5		18	
t1d	HCLK Rise Time		1.5	19	
t1e	HCLK Fall Time		1.5	19	
t1f	HCLK Period Stability		±100		ps[1]
t2a	PCLK Period	30		18	
t2b	PCLK High Time	12		18	
t2c	PCLK Low Time	12		18	
t2d	PCLK Rise Time		3	19	
t2e	PCLK Fall Time		3	19	
t3	HCLK to PCLK Skew	−7.2	5.8	21	

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5 Volts.

4.5.2 COMMAND TIMING, (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t10a	HIG[4:0] Setup Time to HCLK Rising	5.5		15	
t10b	HIG[4:0] Hold Time from HCLK Rising	0		15	
t11a	MIG[2:0] Setup Time to HCLK Rising	5.5		15	
t11b	MIG[2:0] Hold Time from HCLK Rising	0		15	
t12a	PIG[3:0] Setup Time to PCLK Rising	14.5		15	
t12b	PIG[3:0] Hold Time from PCLK Rising	0.0		15	
t13a	MDLE Setup Time to HCLK Rising	5.5		15	
t13b	MDLE Hold Time to HCLK Rising	−0.3		15	
t14a	DRVPCI Setup Time to PCLK Rising	7.0		15	
t14b	DRVPCI Hold Time from PCLK Rising	−0.5		15	
t15a	RESET Setup Time to HCLK Rising	3.4		15	
t15b	RESET Hold Time from HCLK Rising	0.4		15	

4.5.3 ADDRESS, DATA, TRDY#, EOL, TEST, TSCON AND PARITY TIMING, (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t20a	AD[15:0] Output Enable Delay from PCLK Rising	2		17	
t20b	AD[15:0] Valid Delay from PCLK Rising	2	11	14	1
t20c	AD[15:0] Setup Time to PCLK Rising	7		15	
t20d	AD[15:0] Hold Time from PCLK Rising	0		15	
t20e	AD[15:0] Float Delay from DRVPCI Falling	2	10	16	
t21a	TRDY# Setup Time to PCLK Rising	7		15	
t21b	TRDY# Hold Time from PCLK Rising	0		15	
t22a	D[31:0], HP[3:0] Output Enable Delay from HCLK Rising	0	7.5	17	2
t22b	D[31:0], HP[3:0] Float Delay from HCLK Rising	3.1	15.5	16	
t22c	D[31:0], HP[3:0] Float Delay from MDLE Rising	2	9.5	16	3
t22d	D[31:0], HP[3:0] Valid Delay from HCLK Rising	0	7.5	14	2
t22e	D[31:0],HP[3:0] Setup Time to HCLK Rising	3.1		15	
t22f	D[31:0], HP[3:0] Hold Time from HCLK Rising	0.3		15	

Functional Operating Range: V_{CC} = 4.75V to 5V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C (Continued)

Symbol	Parameter	Min	Max	Figure	Notes
t23a	HA[15:0] Output Enable Delay from HCLK Rising	0	13.5	17	
t23b	HA[15:0] Float Delay from HCLK Rising	0	13.5	16	
t23c	HA[15:0] Valid Delay from HCLK Rising	0	17.5	14	7
t23cc	HA[15:0] Valid Delay from HCLK Rising	0	13.5		8
t23d	HA[15:0] Setup Time to HCLK Rising	15		15	4
t23e	HA[15:0] Setup Time to HCLK Rising	4.2		15	5
t23f	HA[15:0] Hold Time from HCLK Rising	0.3		15	
t24a	MD[31:0], MP[3:0] Valid Delay from HCLK Rising	0	12.0	14	6
t24b	MD[31:0], MP[3:0] Setup Time to HCLK Rising	4.4		15	
t24c	MD[31:0], MP[3:0] Hold Time from HCLK Rising	1.0		15	
t25	EOL, PPOUT Valid Delay from PCLK Rising	2.3	17.2	14	2
t26a	All Outputs Float Delay from TSCON Falling	0	30	16	
t26b	All Outputs Enable Delay from TSCON Rising	0	30	17	

NOTE:
1. Min: 0 pF, Max: 50 pF
2. 0 pF
3. When NOPC command sampled on previous rising HCLK on HIG[4:0]
4. CPU to PCI Transfers
5. When ADCPY command is sampled on HIG[4:0]
6. 50 pF
7. When DACPYL or DACPYH commands are sampled on HIG[4:0]
8. Inquire cycle

4.5.4 TEST TIMING (82433NX)

Functional Operating Range: V_{CC} = 4.75V to 5.25V; V_{CC3} = 3.135V to 3.465V, T_{CASE} = 0°C to +85°C

Symbol	Parameter	Min	Max	Figure	Notes
t30	All Test Signals Setup Time to HCLK/ PCLK Rising	10.0			In PLL Bypass Mode
t31	All Test Signals Hold Time to HCLK/ PCLK Rising	12.0			In PLL Bypass Mode
t32	Test Setup Time to HCLK/PCLK Rising	15.0		15	
t33	Test Hold Time to HCLK/PCLK Rising	5.0		15	
t34	PPOUT Valid Delay from PCLK Rising	0.0	500	15	In PLL Bypass Mode

4.5.5 TIMING DIAGRAMS

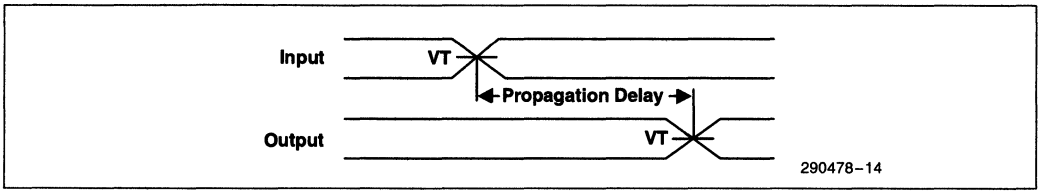


Figure 13. Propagation Delay

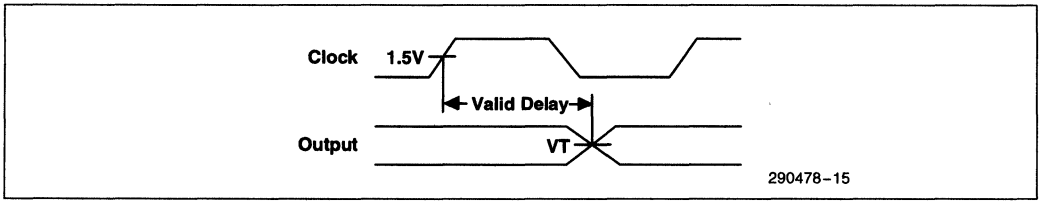


Figure 14. Valid Delay from Rising Clock Edge

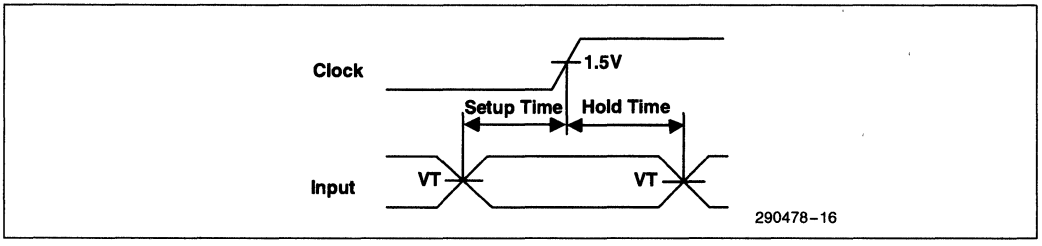


Figure 15. Setup and Hold Times

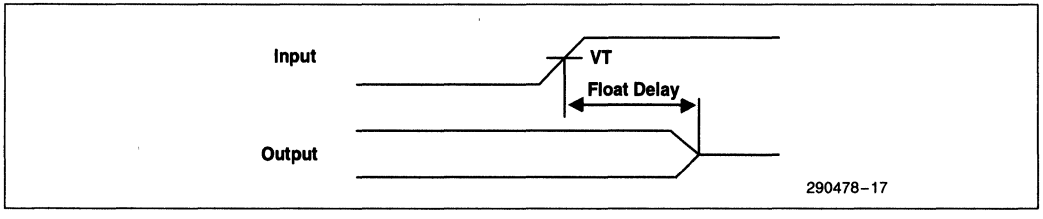


Figure 16. Float Delay

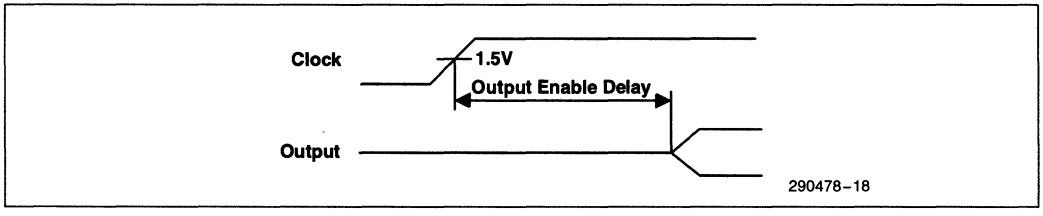


Figure 17. Output Enable Delay

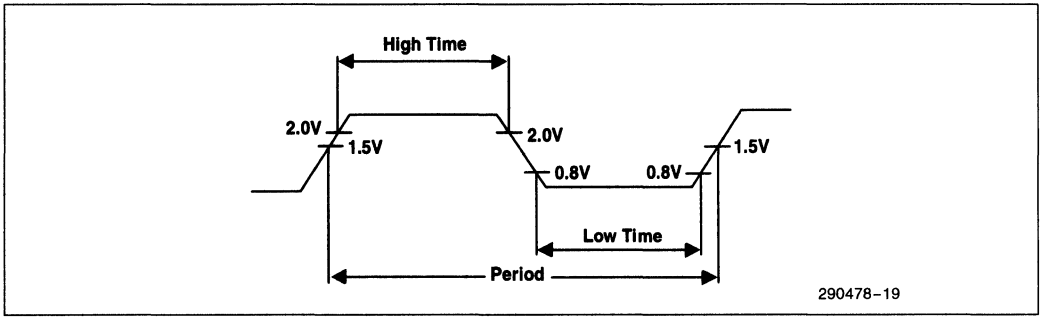


Figure 18. Clock High and Low Times and Period

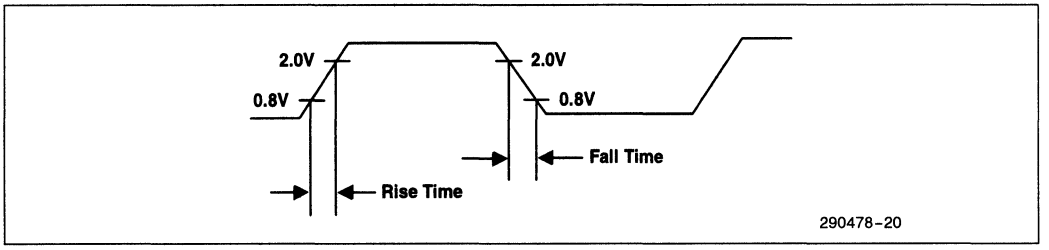


Figure 19. Clock Rise and Fall Times

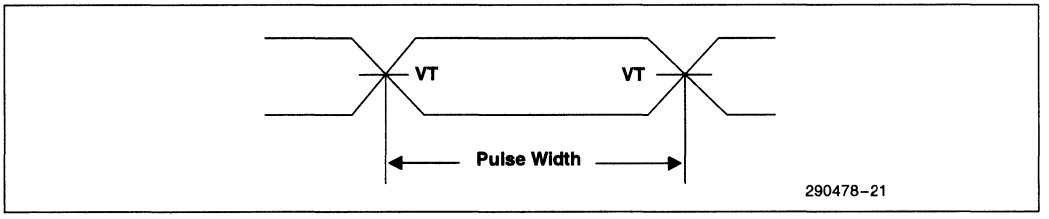


Figure 20. Pulse Width

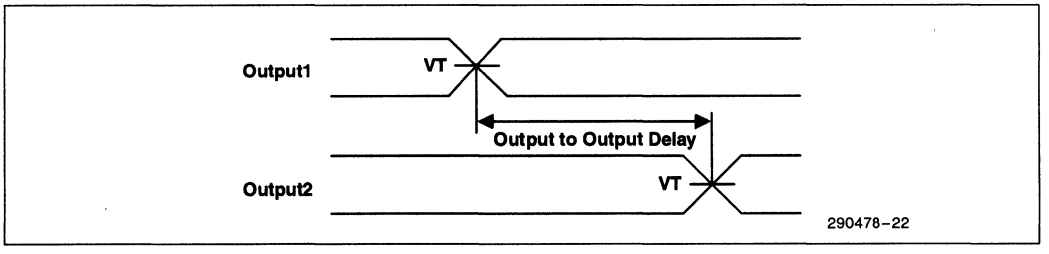


Figure 21. Output to Output Delay

5.0 PINOUT AND PACKAGE INFORMATION

5.1 Pin Assignment

Pins 1, 22, 41, 61, and 150 are VDD3 pins on the 82433NX. These pins must be connected to the 3.3V power supply. All other VDD pins on the 82433NX must be connected to the 5V power supply.

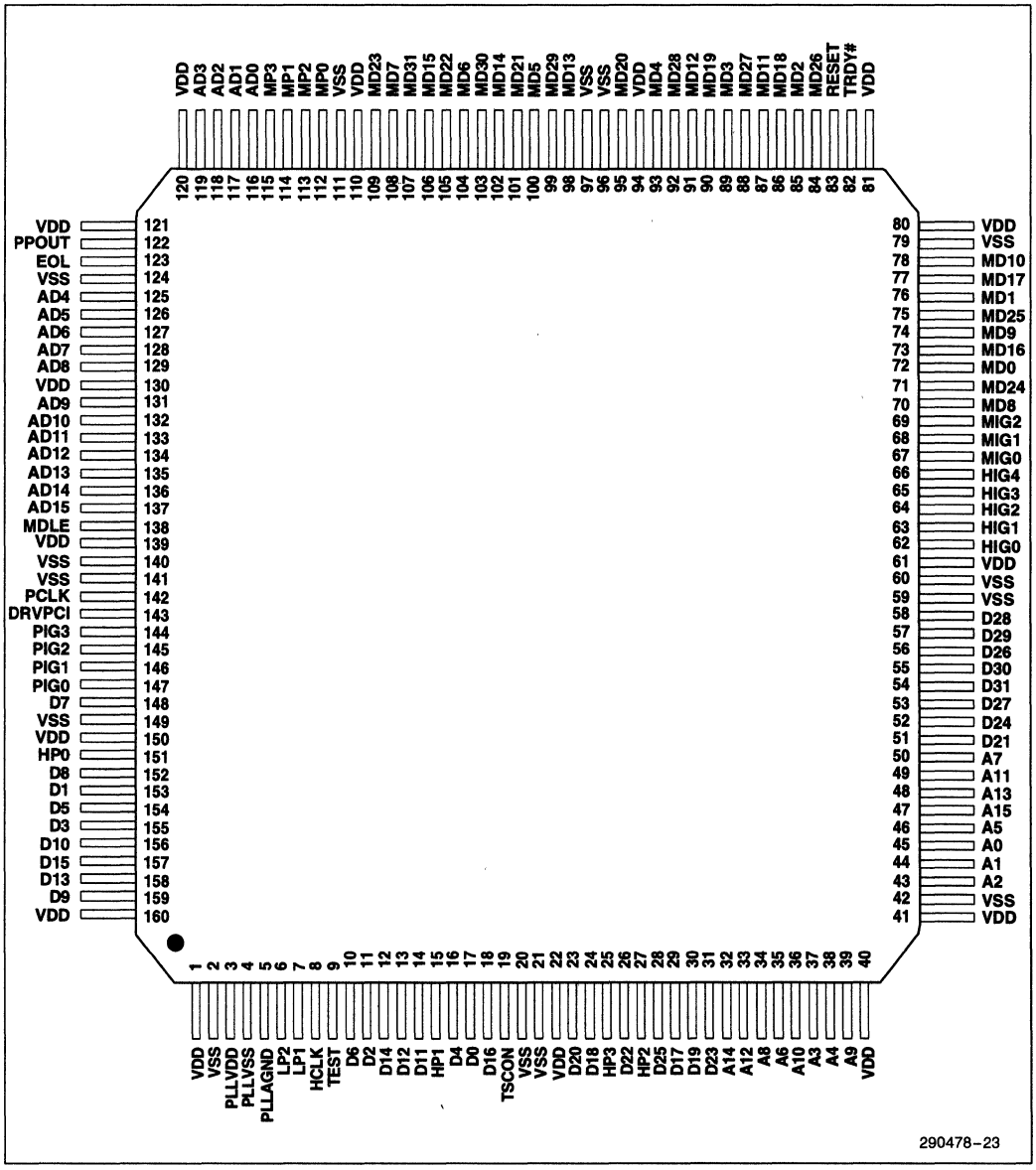


Figure 22. 82433LX and 82433NX Pin Assignment

Table 5. 82433LX and 82433NX Numerical Pin Assignment

Pin Name	Pin #	Type
V_{DD} (82433LX) V_{DD3} (82433NX)	1	V
V_{SS}	2	V
$PLLV_{DD}$	3	V
$PLLV_{SS}$	4	V
PLLAGND	5	V
LP2	6	in
LP1	7	out
HCLK	8	in
TEST	9	in
D6	10	t/s
D2	11	t/s
D14	12	t/s
D12	13	t/s
D11	14	t/s
HP1	15	t/s
D4	16	t/s
D0	17	t/s
D16	18	t/s
TSCON	19	in
V_{SS}	20	V
V_{SS}	21	V
V_{DD} (82433LX V_{DD3} (82433NX)	22	V
D20	23	t/s
D18	24	t/s
HP3	25	t/s

Pin Name	Pin #	Type
D22	26	t/s
HP2	27	t/s
D25	28	t/s
D17	29	t/s
D19	30	t/s
D23	31	t/s
A14	32	t/s
A12	33	t/s
A8	34	t/s
A6	35	t/s
A10	36	t/s
A3	37	t/s
A4	38	t/s
A9	39	t/s
V_{DD}	40	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	41	V
V_{SS}	42	V
A2	43	t/s
A1	44	t/s
A0	45	t/s
A5	46	t/s
A15	47	t/s
A13	48	t/s
A11	49	t/s
A7	50	t/s

Pin Name	Pin #	Type
D21	51	t/s
D24	52	t/s
D27	53	t/s
D31	54	t/s
D30	55	t/s
D26	56	t/s
D29	57	t/s
D28	58	t/s
V_{SS}	59	V
V_{SS}	60	V
V_{DD} (82433LX) V_{DD3} (82433NX)	61	V
HIG0	62	in
HIG1	63	in
HIG2	64	in
HIG3	65	in
HIG4	66	in
MIG0	67	in
MIG1	68	in
MIG2	69	in
MD8	70	t/s
MD24	71	t/s
MD0	72	t/s
MD16	73	t/s
MD9	74	t/s
MD25	75	t/s

Table 5. 82433LX and 82433NX Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
MD1	76	t/s
MD17	77	t/s
MD10	78	t/s
V_{SS}	79	V
V_{DD}	80	V
V_{DD}	81	V
TRDY#	82	in
RESET	83	in
MD26	84	t/s
MD2	85	t/s
MD18	86	t/s
MD11	87	t/s
MD27	88	t/s
MD3	89	t/s
MD19	90	t/s
MD12	91	t/s
MD28	92	t/s
MD4	93	t/s
V_{DD}	94	V
MD20	95	t/s
V_{SS}	96	V
V_{SS}	97	V
MD13	98	t/s
MD29	99	t/s
MD5	100	t/s
MD21	101	t/s
MD14	102	t/s
MD30	103	t/s
MD6	104	t/s

Pin Name	Pin #	Type
MD22	105	t/s
MD15	106	t/s
MD31	107	t/s
MD7	108	t/s
MD23	109	t/s
V_{DD}	110	V
V_{SS}	111	V
MP0	112	t/s
MP2	113	t/s
MP1	114	t/s
MP3	115	t/s
AD0	116	t/s
AD1	117	t/s
AD2	118	t/s
AD3	119	t/s
V_{DD}	120	V
V_{DD}	121	V
PPOUT	122	t/s
EOL	123	t/s
V_{SS}	124	V
AD4	125	t/s
AD5	126	t/s
AD6	127	t/s
AD7	128	t/s
AD8	129	t/s
V_{DD}	130	V
AD9	131	t/s
AD10	132	t/s

Pin Name	Pin #	Type
AD11	133	t/s
AD12	134	t/s
AD13	135	t/s
AD14	136	t/s
AD15	137	t/s
MDLE	138	in
V_{DD}	139	V
V_{SS}	140	V
V_{SS}	141	V
PCLK	142	in
DRVPCI	143	in
PIG3	144	in
PIG2	145	in
PIG1	146	in
PIG0	147	in
D7	148	t/s
V_{SS}	149	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	150	V
HP0	151	t/s
D8	152	t/s
D1	153	t/s
D5	154	t/s
D3	155	t/s
D10	156	t/s
D15	157	t/s
D13	158	t/s
D9	159	t/s
V_{DD}	160	V

Table 6. 82433LX and 82433NX Alphabetical Pin Assignment List

Pin Name	Pin #	Type
A0	45	t/s
A1	44	t/s
A2	43	t/s
A3	37	t/s
A4	38	t/s
A5	46	t/s
A6	35	t/s
A7	50	t/s
A8	34	t/s
A9	39	t/s
A10	36	t/s
A11	49	t/s
A12	33	t/s
A13	48	t/s
A14	32	t/s
A15	47	t/s
AD0	116	t/s
AD1	117	t/s
AD2	118	t/s
AD3	119	t/s
AD4	125	t/s
AD5	126	t/s
AD6	127	t/s
AD7	128	t/s
AD8	129	t/s
AD9	131	t/s
AD10	132	t/s
AD11	133	t/s
AD12	134	t/s

Pin Name	Pin #	Type
AD13	135	t/s
AD14	136	t/s
AD15	137	t/s
D0	17	t/s
D1	153	t/s
D2	11	t/s
D3	155	t/s
D4	16	t/s
D5	154	t/s
D6	10	t/s
D7	148	t/s
D8	152	t/s
D9	159	t/s
D10	156	t/s
D11	14	t/s
D12	13	t/s
D13	158	t/s
D14	12	t/s
D15	157	t/s
D16	18	t/s
D17	29	t/s
D18	24	t/s
D19	30	t/s
D20	23	t/s
D21	51	t/s
D22	26	t/s
D23	31	t/s
D24	52	t/s
D25	28	t/s

Pin Name	Pin #	Type
D26	56	t/s
D27	53	t/s
D28	58	t/s
D29	57	t/s
D30	55	t/s
D31	54	t/s
DRVPCI	143	in
EOL	123	t/s
HCLK	8	in
HIG0	62	in
HIG1	63	in
HIG2	64	in
HIG3	65	in
HIG4	66	in
HP0	151	t/s
HP1	15	t/s
HP2	27	t/s
HP3	25	t/s
LP1	7	out
LP2	6	in
MD0	72	t/s
MD1	76	t/s
MD2	85	t/s
MD3	89	t/s
MD4	93	t/s
MD5	100	t/s
MD6	104	t/s
MD7	108	t/s
MD8	70	t/s

Table 6. 82433LX and 82433NX Alphabetical Pin Assignment List (Continued)

Pin Name	Pin #	Type
MD9	74	t/s
MD10	78	t/s
MD11	87	t/s
MD12	91	t/s
MD13	98	t/s
MD14	102	t/s
MD15	106	t/s
MD16	73	t/s
MD17	77	t/s
MD18	86	t/s
MD19	90	t/s
MD20	95	t/s
MD21	101	t/s
MD22	105	t/s
MD23	109	t/s
MD24	71	t/s
MD25	75	t/s
MD26	84	t/s
MD27	88	t/s
MD28	92	t/s
MD29	99	t/s
MD30	103	t/s
MD31	107	t/s
MDLE	138	in
MIG0	67	in

Pin Name	Pin #	Type
MIG1	68	in
MIG2	69	in
MP0	112	t/s
MP1	114	t/s
MP2	113	t/s
MP3	115	t/s
PCLK	142	in
PIG0	147	in
PIG1	146	in
PIG2	145	in
PIG3	144	in
PLLAGND	5	V
$PLLV_{DD}$	3	V
$PLLV_{SS}$	4	V
PPOUT	122	t/s
RESET	83	in
TEST	9	in
TRDY	82	in
TSCON	19	in
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	1	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	22	V
V_{DD}	40	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	41	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	61	V

Pin Name	Pin #	Type
V_{DD}	80	V
V_{DD}	81	V
V_{DD}	94	V
V_{DD}	110	V
V_{DD}	120	V
V_{DD}	121	V
V_{DD}	130	V
V_{DD}	139	V
V_{DD} (82433LX) $V_{DD}3$ (82433NX)	150	V
V_{DD}	160	V
V_{SS}	2	V
V_{SS}	20	V
V_{SS}	21	V
V_{SS}	42	V
V_{SS}	59	V
V_{SS}	60	V
V_{SS}	79	V
V_{SS}	96	V
V_{SS}	97	V
V_{SS}	111	V
V_{SS}	124	V
V_{SS}	140	V
V_{SS}	141	V
V_{SS}	149	V

5.2 Package Information

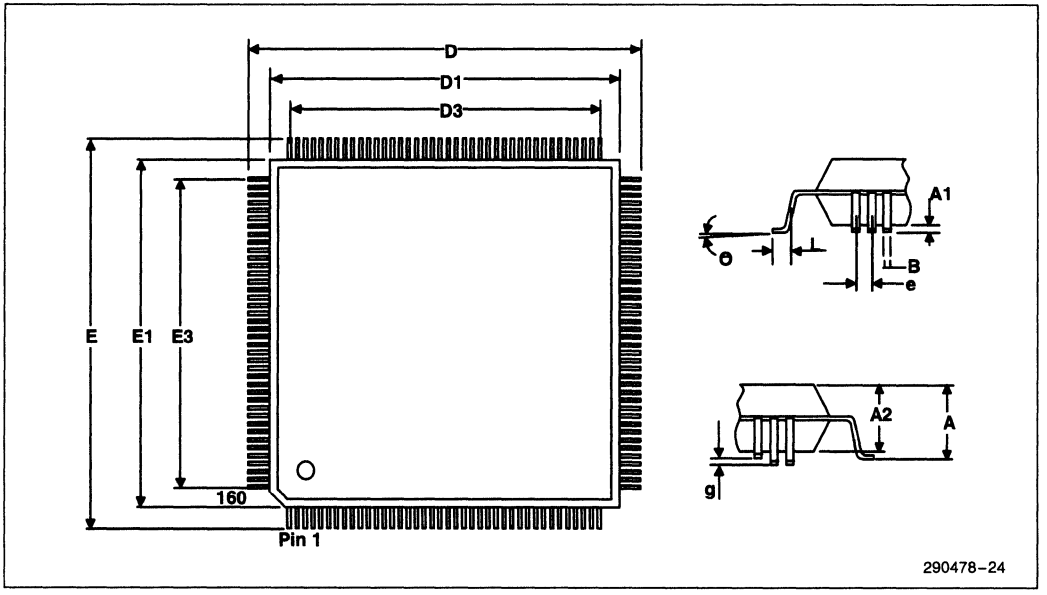


Figure 23. 82433LX and 82433NX 160-Pin QFP Package

Table 7. 160-Pin QFP Package Values

Symbol	Min Value (mm)	Max Value (mm)
A		4.45
A1	0.25	0.65
A2	3.30	3.80
B	0.20	0.40
D	31.00	32.40
D1	27.80	28.20
D3		25.55

Symbol	Min Value (mm)	Max Value (mm)
E	31.60	32.40
E1	27.80	28.20
E3		25.55
e		0.65
L	0.60	1.00
θ	0°	10°
g		0.1

6.0 TESTABILITY

The TSCON pin may be used to help test circuits surrounding the LBX. During normal operations, the TSCON pin must be tied to VCC or connected to VCC through a pull-up resistor. All LBX outputs are tri-stated when the TSCON pin is held low or grounded.

6.1 NAND Tree

A NAND tree is provided in the LBX for Automated Test Equipment (ATE) board level testing. The NAND tree allows the tester to set the connectivity of each of the LBX signal pins.

The following steps must be taken to put the LBX into PLL bypass mode and enable the NAND tree. First, to enable PLL bypass mode, drive RESET inactive, TEST active, and the DCPWA command (0100) on the PIG[3:0] lines. Then drive PCLK from low to high. DRVPCI must be held low on all rising edges of PCLK during testing in order to ensure that the LBX does not drive the AD[15:0] lines. The host and memory buses are tri-stated by driving NOPM (000) and NOPC (00000) on the MIG[2:0] and HIG[4:0] lines and driving two rising edges on HCLK. A rising edge on PCLK with RESET high will cause the LBXs to exit PLL bypass mode. TEST must remain high throughout the use of the NAND tree. The combination of TEST and DRVPCI high with a rising edge of PCLK must be avoided. TSCON must be driven high throughout testing since driving it low would tri-state the output of the NAND tree. A 10 ns hold time is required on all inputs sampled by PCLK or HCLK when in PLL bypass mode.

6.1.1 TEST VECTOR TABLE

The following test vectors can be applied to the 82433LX and 82433NX to put it into PLL bypass mode and to enable NAND tree testing.

6.1.2 NAND TREE TABLE

Table 9 shows the sequence of the NAND tree in the 82433LX and 82433NX. Non-inverting inputs are driven directly into the input of a NAND gate in the tree. Inverting inputs are driven into an inverter before going into the NAND tree. The output of the NAND tree is driven on the PPOUT pin.

Table 8. Test Vectors to put LBX Into PLL Bypass and Enable NAND Tree Testing

LBX Pin/Vector#	1	2	3	4	5	6	7	8	9	10	11
PCLK	0	1	0	0	1	1	1	1	1	1	1
PIG[3:0]	0h	0h	0h	4h	4h	4h	4h	4h	4h	4h	4h
RESET	1	1	1	0	0	0	1	1	1	1	1
HCLK	0	0	0	0	0	0	0	1	0	1	0
MIG[2:0]	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h
HIG[4:0]	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h	0h
TEST	1	1	1	1	1	1	1	1	1	1	1
DRVPCI	0	0	0	0	0	0	0	0	0	0	0

Table 9. NAND Tree Sequence

Order	Pin #	Signal	Non-Inverting
1	10	D6	Y
2	11	D2	Y
3	12	D14	Y
4	13	D12	Y
5	14	D11	Y
6	15	HP1	Y
7	16	D4	Y
8	17	D0	Y
9	18	D16	Y
10	23	D20	Y
11	24	D18	Y
12	25	HP3	Y
13	26	D22	Y
14	27	HP2	Y
15	28	D25	Y
16	29	D17	Y
17	30	D19	Y
18	31	D23	Y
19	32	A14	Y
20	33	A12	Y
21	34	A8	Y
22	35	A6	Y
23	36	A10	Y
24	37	A3	Y
25	38	A4	Y
26	39	A9	Y
27	43	A2	Y
28	44	A1	Y
29	45	A0	Y
30	46	A5	Y
31	47	A15	Y
32	48	A13	Y
33	49	A11	Y
34	50	A7	Y
35	51	D21	Y
36	52	D24	Y
37	53	D27	Y
38	54	D31	Y
39	55	D30	Y
40	56	D26	Y
41	57	D29	Y
42	58	D28	Y
43	62	HIG0	Y
44	63	HIG1	Y
45	64	HIG2	Y
46	65	HIG3	Y
47	66	HIG4	Y
48	67	MIG0	N
49	68	MIG1	N
50	69	MIG2	N
51	70	MD8	N
52	71	MD24	N
53	72	MD0	N
54	73	MD16	N
55	74	MD9	N
56	75	MD25	N
57	76	MD1	N
58	77	MD17	N
59	78	MD10	N
60	82	TRDY#	Y
61	83	RESET	N
62	84	MD26	N
63	85	MD2	N
64	86	MD18	N
65	87	MD11	N
66	88	MD27	N
67	89	MD3	N
68	90	MD19	N
69	91	MD12	N
70	92	MD28	N
71	93	MD4	N
72	95	MD20	N
73	98	MD13	N
74	99	MD29	N
75	100	MD5	N
76	101	MD21	N
77	102	MD14	N
78	103	MD30	N

Table 9. NAND Tree Sequence (Continued)

Order	Pin #	Signal	Non-Inverting
79	104	MD6	N
80	105	MD22	N
81	106	MD15	N
82	107	MD31	N
83	108	MD7	N
84	109	MD23	N
85	112	MP0	N
86	113	MP2	N
87	114	MP1	N
88	115	MP3	N
89	116	AD0	Y
90	117	AD1	Y
91	118	AD2	Y
82	119	AD3	Y
93	123	EOL	Y
94	125	AD4	Y
95	126	AD5	Y
96	127	AD6	Y
97	128	AD7	Y
98	129	AD8	Y
99	131	AD9	Y
100	132	AD10	Y
101	133	AD11	Y
102	134	AD12	Y
103	135	AD13	Y
104	136	AD14	Y
105	137	AD15	Y
106	138	MDLE	Y
107	143	DRVPCI	N
108	144	PIG3	N
109	145	PIG2	N
110	146	PIG1	N
111	147	PIG0	N
112	148	D7	Y
113	151	HP0	Y
114	152	D8	Y
115	153	D1	Y
116	154	D5	Y
117	155	D3	Y
118	156	D10	Y
119	157	D15	Y
120	158	D13	Y
121	159	D9	Y

6.2 PLL Test Mode

The high order 82433NX LBX samples A11 at the falling edge of reset to configure the LBX for PLL test mode. When A11 is sampled low, the LBX is in normal operating mode. When A11 is sampled high, the LBX drives the internal HCLK from the PLL on the EOL pin.

82434LX/82434NX PCI, CACHE AND MEMORY CONTROLLER (PCMC)

- Supports the Pentium™ Processor at iCOMP™ Index 510\60 MHz and iCOMP Index 567\66 MHz
- Supports the Pentium Processor at iCOMP Index 735\90 MHz, iCOMP Index 815\100 MHz, and iCOMP Index 610\75 MHz
- Supports Pipelined Addressing Capability of the Pentium Processor
- The 82430NX Drives 3.3V Signal Levels on the CPU and Cache Interfaces
- High Performance CPU/PCI/Memory Interfaces via Posted Write and Read Prefetch Buffers
- Fully Synchronous PCI Interface with Full Bus Master Capability
- Supports the Pentium Processor Internal Cache in Either Write-Through or Write-Back Mode
- Programmable Attribute Map of DOS and BIOS Regions for System Flexibility
- Integrated Low Skew Clock Driver for Distributing Host Clock
- Integrated Second Level Cache Controller
  - Integrated Cache Tag RAM
  - Write-Through and Write-Back Cache Modes for the 82434LX
  - Write-Back for the 82434NX
  - 82434NX Supports Low-Power Cache Standby
  - Direct Mapped Organization
  - Supports Standard and Burst SRAMs
  - 256-KByte and 512-KByte Sizes
  - Cache Hit Cycle of 3-1-1-1 on Reads and Writes Using Burst SRAMs
  - Cache Hit Cycle of 3-2-2-2 on Reads and 4-2-2-2 on Writes Using Standard SRAMs
- Integrated DRAM Controller
  - Supports 2 MBytes to 192 MBytes of Cacheable Main Memory for the 82434LX
  - Supports 2 MBytes to 512 MBytes of Cacheable Main Memory for the 82434NX
  - Supports DRAM Access Times of 70 ns and 60 ns
  - CPU Writes Posted to DRAM 4-1-1-1
  - Refresh Cycles Decoupled from ISA Refresh to Reduce the DRAM Access Latency
  - Six RAS# Lines (82434LX)
  - Eight RAS# Lines (82434NX)
  - Refresh by RAS#-Only, or CAS-Before-RAS#, in Single or Burst of Four
- Host/PCI Bridge
  - Translates CPU Cycles into PCI Bus Cycles
  - Translates Back-to-Back Sequential CPU Memory Writes into PCI Burst Cycles
  - Burst Mode Writes to PCI in Zero PCI Wait-States (i.e. Data Transfer Every Cycle)
  - Full Concurrency Between CPU-to-Main Memory and PCI-to-PCI Transactions
  - Full Concurrency Between CPU-to-Second Level Cache and PCI-to-Main Memory Transactions
  - Same Cache and Memory System Logic Design for ISA and EISA Systems
  - Cache Snoop Filter Ensures Data Consistency for PCI-to-Main Memory Transactions
- 208-Pin QFP Package

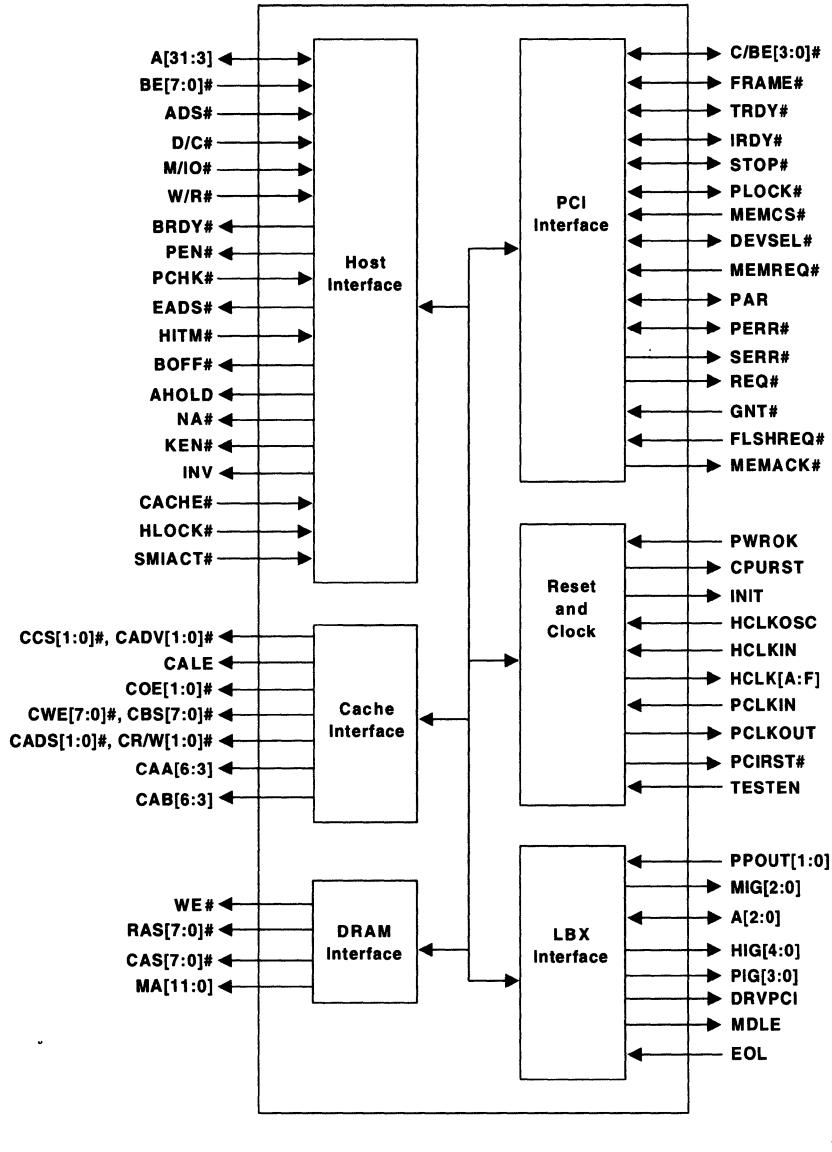
2

*Other brands and names are the property of their respective owners.

December 1994
Order Number: 290479-004

This document describes both the 82434LX and 82434NX. Unshaded areas describe the 82434LX. Shaded areas, like this one, describe 82434NX operations that differ from the 82434LX.

The 82434LX/82434NX PCI, Cache, Memory Controllers (PCMC) integrate the cache and main memory DRAM control functions and provide bus control for transfers between the CPU, cache, main memory, and the PCI Local Bus. The cache controller supports write-back (or write-through for 82434LX) cache policy and cache sizes of 256-KBytes and 512-KBytes. The cache memory can be implemented with either standard or burst SRAMs. The PCMC cache controller integrates a high-performance Tag RAM to reduce system cost.



NOTE:
RAS[7:6]# and MA11 are only on the 82434NX. CCS[1:0] functionality is only on the 82434NX.

Simplified Block Diagram of the PCMC

2

82434LX/82434NX PCI, CACHE AND MEMORY CONTROLLER (PCMC)

CONTENTS

CONTENTS

CONTENTS

PAGE

CONTENTS

CONTENTS

PAGE

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

This section provides an 82430LX/82430NX PCIset system overview that includes a description of the bus hierarchy and bridges between the buses. The 82430LX PCIset consists of the 82434LX PCMC and 82433LX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The 82430NX PCIset consists of the 82434NX PCMC and 82433NX LBX components plus either a PCI/ISA bridge or a PCI/EISA bridge. The PCMC and LBX provide the core cache and main memory architecture and serve as the Host/PCI bridge. An overview of the PCMC follows the system overview section.

1.1 System Overview

The 82430LX/82430NX PCIset provides the Host/PCI bridge, cache and main memory controller, and an I/O subsystem core (either PCI/EISA or PCI/ISA bridge) for the next generation of high-performance personal computers based on the Pentium processor. System designers can take advantage of the power of the PCI (Peripheral Component Interconnect) local bus while maintaining access to the large base of EISA and ISA expansion cards. Extensive buffering and buffer management within the bridges ensures maximum efficiency in all three buses (Host CPU, PCI, and EISA/ISA Buses).

For an ISA-based system, the PCIset includes the System I/O (82378IB SIO) component (Figure 1) as the PCI/ISA bridge. For an EISA-based system (Figure 2), the PCIset includes the PCI-EISA bridge (82375EB PCEB) and the EISA System Component (82374EB ESC). The PCEB and ESC work in tandem to form the complete PCI/EISA bridge.

1.1.1. BUS HIERARCHY—CONCURRENT OPERATIONS

Systems based on the 82430LX/82430NX PCIset contain three levels of buses structured in the following hierarchy:

- Host Bus as the execution bus
- PCI Bus as a primary I/O bus
- ISA or EISA Bus as a secondary I/O bus.

This bus hierarchy allows concurrency for simultaneous operations on all three buses. Data buffering permits concurrency for operations that crossover into another bus. For example, the Pentium processor could post data destined to the PCI in the LBX. This permits the Host transaction to complete in minimum time, freeing up the Host Bus for further transactions. The Pentium processor does not have to wait for the transfer to complete to its final destination. Meanwhile, any ongoing PCI Bus transactions are permitted to complete. The posted data is then transferred to the PCI Bus when the PCI Bus is available. The LBX implements extensive buffering for Host-to-PCI, Host-to-main memory, and PCI-to-main memory transactions. In addition, the PCEB/ESC chip set and the SIO implement extensive buffering for transfers between the PCI Bus and the EISA and ISA Buses, respectively.

Host Bus

Designed to meet the needs of high-performance computing, the Host Bus features:

- 64-bit data path
- 32-bit address bus with address pipelining
- Synchronous frequencies of 60 MHz and 66 MHz
- Synchronous frequency of 50 MHz (82430NX)
- Burst read and write transfers
- Support for first level and second level caches
- Capable of full concurrency with the PCI and memory subsystems
- Byte data parity
- Full support for Pentium processor machine check and DOS compatible parity reporting
- Support for Pentium processor System Management Mode (SMM).

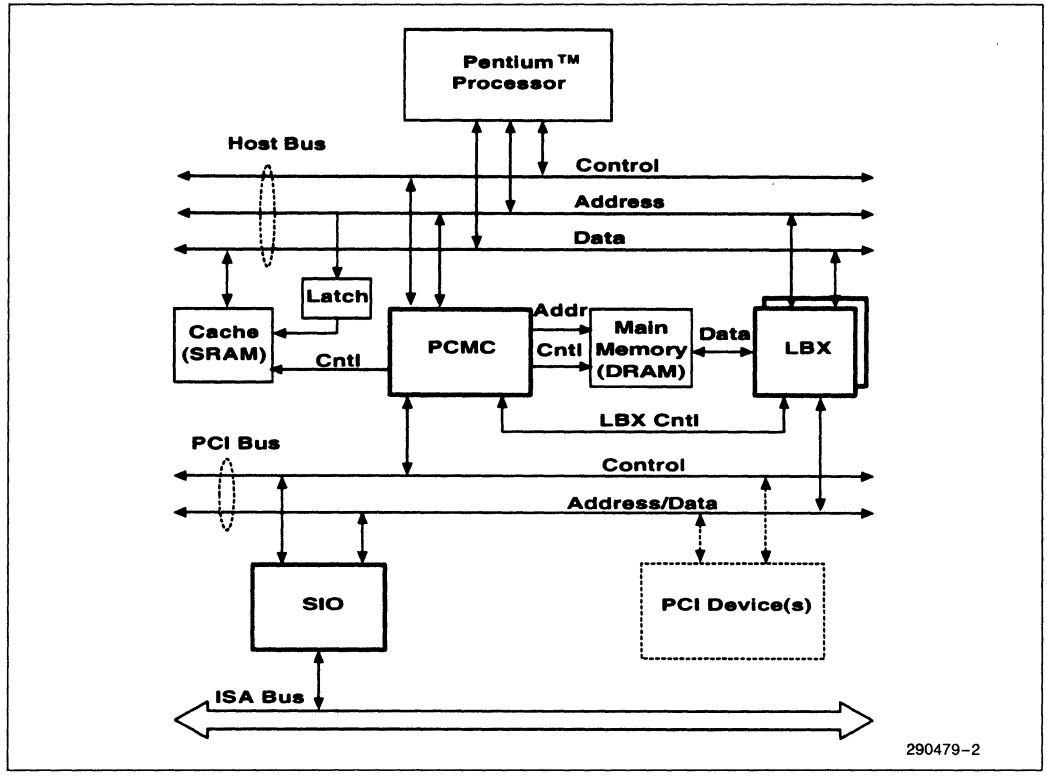


Figure 1. Block Diagram of a 82430LX/82430NX PCIset ISA System

2

PCI Bus

The PCI Bus is designed to address the growing industry needs for a standardized *local bus* that is not directly dependent on the speed and the size of the processor bus. New generations of personal computer system software such as Windows™ and Win-NT™ with sophisticated graphical interfaces, multi-tasking, and multi-threading bring new requirements that traditional PC I/O architectures cannot satisfy. In addition to the higher bandwidth, reliability and robustness of the I/O subsystem are becoming increasingly important. PCI addresses these needs and provides a future upgrade path. PCI features include:

- Processor independent
- Multiplexed, burst mode operation
- Synchronous at frequencies up to 33 MHz
- 120 MByte/sec usable throughput (132 MByte/sec peak) for a 32-bit data path

- Low latency random access (60 ns write access latency to slave registers from a master parked on the bus)
- Capable of full concurrency with the processor/memory subsystem
- Full multi-master capability allowing any PCI master peer-to-peer access to any PCI slave
- Hidden (overlapped) central arbitration
- Low pin count for cost effective component packaging (multiplexed address/data)
- Address and data parity
- Three physical address spaces: memory, I/O, and configuration
- Comprehensive support for autoconfiguration through a defined set of standard configuration functions.

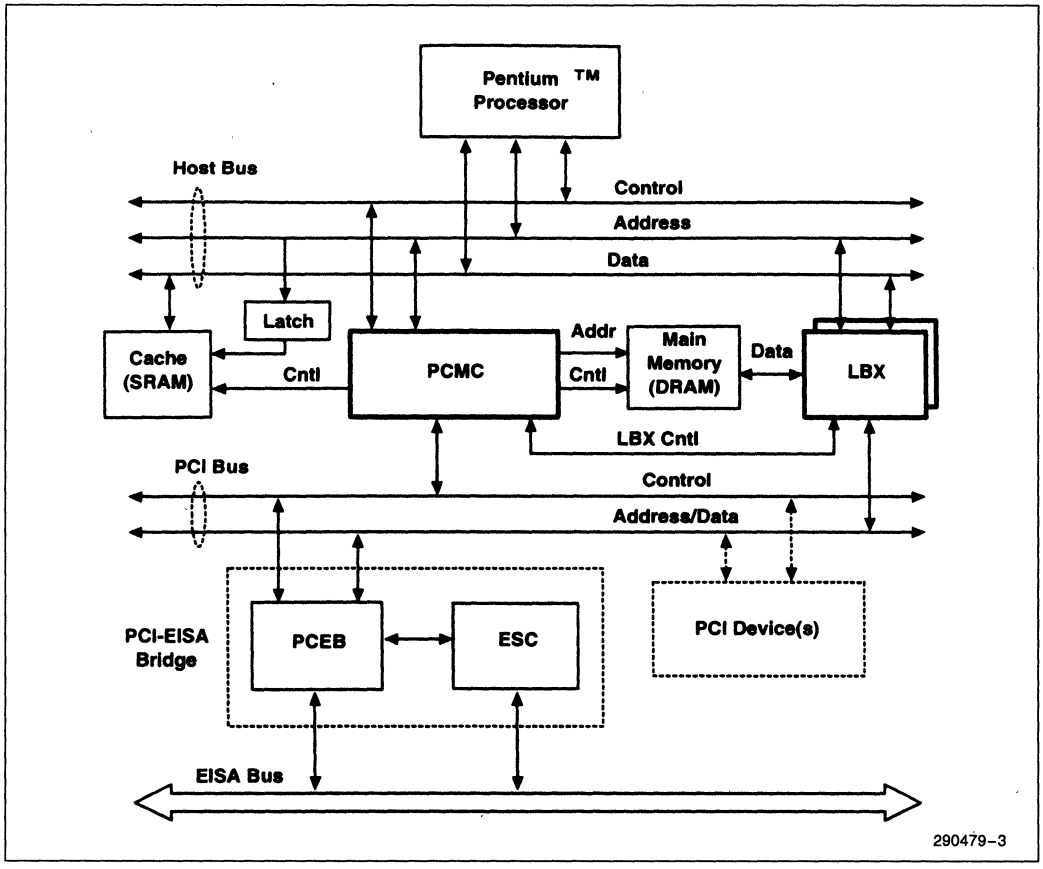


Figure 2. Block Diagram of the 82430LX/82430NX PCIset EISA System

ISA Bus

Figure 1 represents a system using the ISA Bus as the second level I/O bus. It allows personal computer platforms built around the PCI as a primary I/O bus to leverage the large ISA product base. The ISA Bus has 24-bit addressing and a 16-bit data path.

EISA Bus

Figure 2 represents a system using the EISA Bus as the second level I/O bus. It allows personal computer platforms built around the PCI as a primary I/O bus to leverage the large EISA/ISA product base. Combinations of PCI and EISA buses, both of which can be used to provide expansion functions, will satisfy even the most demanding applications.

Along with compatibility for 16-bit and 8-bit ISA hardware and software, the EISA bus provides the following key features:

- 32-bit addressing and 32-bit data path
- 33 MByte/sec bus bandwidth
- Multiple bus master support through efficient arbitration
- Support for autoconfiguration.

1.1.2 BUS BRIDGES

Host/PCI Bridge Chip Set (PCMC and LBX)

The PCMC and LBX enhance the system performance by allowing for concurrency between the Host CPU Bus and PCI Bus, giving each greater bus throughput and decreased bus latency. The LBX contains posted write buffers for Host-to-PCI, Host-to-main memory, and PCI-to-main memory transfers. The LBX also contains read prefetch buffers for Host reads of PCI, and PCI reads of main memory. There are two LBXs per system. The LBXs are controlled by commands from the PCMC. The PCMC/LBX Host/PCI bridge chip set is covered in more detail in Section 1.2, PCMC Overview.

PCI-EISA Bridge Chip Set (PCEB and ESC)

The PCEB provides the master/slave functions on both the PCI Bus and the EISA Bus. Functioning as a bridge between the PCI and EISA buses, the PCEB provides the address and data paths, bus controls, and bus protocol translation for PCI-to-EISA and EISA-to-PCI transfers. Extensive data buffering in both directions increase system performance by maximizing PCI and EISA Bus efficiency and allowing concurrency on the two buses. The PCEB's buffer management mechanism ensures data coherency. The PCEB integrates central bus control functions including a programmable bus arbiter for the PCI Bus and EISA data swap buffers for the EISA Bus. Integrated system functions include PCI parity generation, system error reporting, and programmable PCI and EISA memory and I/O address space mapping and decoding. The PCEB also contains a BIOS Timer that can be used to implement timing loops. The PCEB is intended to be used with the ESC to provide an EISA I/O subsystem interface.

The ESC integrates the common I/O functions found in today's EISA-based PCs. The ESC incorporates the logic for EISA Bus controller, enhanced seven channel DMA controller with scatter-gather support, EISA arbitration, 14 level interrupt controller, Advanced Programmable Interrupt Controller (APIC), five programmable timer/counters, non-maskable-interrupt (NMI) control, and power management. The ESC also integrates support logic to decode peripheral devices (e.g., the flash BIOS, real time clock, keyboard/mouse controller, floppy controller, two serial ports, one parallel port, and IDE hard disk drive).

PCI/ISA Bridge (SIO):

The SIO component provides the bridge between the PCI Bus and the ISA Bus. The SIO also integrates many of the common I/O functions found in today's ISA-based PCs. The SIO incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller that supports fast DMA transfers and scatter-gather, data buffers to isolate the PCI Bus from the ISA Bus and to enhance performance, PCI and ISA arbitration, 14 level interrupt controller, a 16-bit BIOS timer, three programmable timer/counters, and non-maskable-interrupt (NMI) control logic. The SIO also provides decode for peripheral devices (e.g., the flash BIOS, real time clock, keyboard/mouse controller, floppy controller, two serial ports, one parallel port, and IDE hard disk drive).

1.2 PCMC Overview

The PCMC (along with the LBX) provides three basic functions: a cache controller, a main memory DRAM controller, and a Host/PCI bridge. This section provides an overview of these functions. Note that, in this document, operational descriptions assume that the PCMC and LBX components are used together.

1.2.1 CACHE OPERATIONS

The PCMC provides the control for a second level cache memory array implemented with either standard asynchronous SRAMs or synchronous burst SRAMs. The data memory array is external to the PCMC and located on the Host address/data bus. Since the Pentium processor contains an internal cache, there can be two separate caches in a Host subsystem. The cache inside the Pentium processor is referred to as the first level cache (also called primary cache). A detailed description of the first level cache is beyond the scope of this document. The PCMC cache control circuitry and associated external memory array is referred to as the second level cache (also called secondary cache). The second level cache is unified, meaning that both CPU data and instructions are stored in the cache. The 82434LX PCMC supports both write-through and write-back caching policies and the 82434NX supports write-back.

The optional second level cache memory array can be either 256-KBytes or 512-KBytes in size. The cache is direct-mapped and is organized as either 8K or 16K cache lines of 32 bytes per line.

In addition to the cache data RAM, the second level cache contains a 4K set of cache tags that are internal to the PCMC. Each tag contains an address that is associated with the corresponding data sector (2 lines for a 256 KByte cache and 4 lines for a 512 KByte cache) and two status bits for each line in the sector.

During a main memory read or write operation, the PCMC first searches the cache. If the addressed code or data is in the cache, the cycle is serviced by the cache. If the addressed code or data is not in the cache, the cycle is forwarded to main memory.

For the write-through (82434LX only) and write-back (both 82434LX and 82434NX) policies, the cache operation is determined by the CPU read or write cycle as follows:

Write Cycle

If the caching policy is write-through and the write cycle hits in the cache, both the cache and main memory are updated. Upon a cache miss, only main memory is updated. The cache is not updated (no write-allocate).

If the caching policy is write-back and the write cycle hits in the cache, only the cache is updated; main memory is not affected. Upon a cache miss, only main memory is updated. The cache is not updated (no write-allocate).

Read Cycle

Upon a cache hit, the cache operation is the same for both write-through and write-back. In this case, data is transferred from the cache to the CPU. Main memory is not accessed.

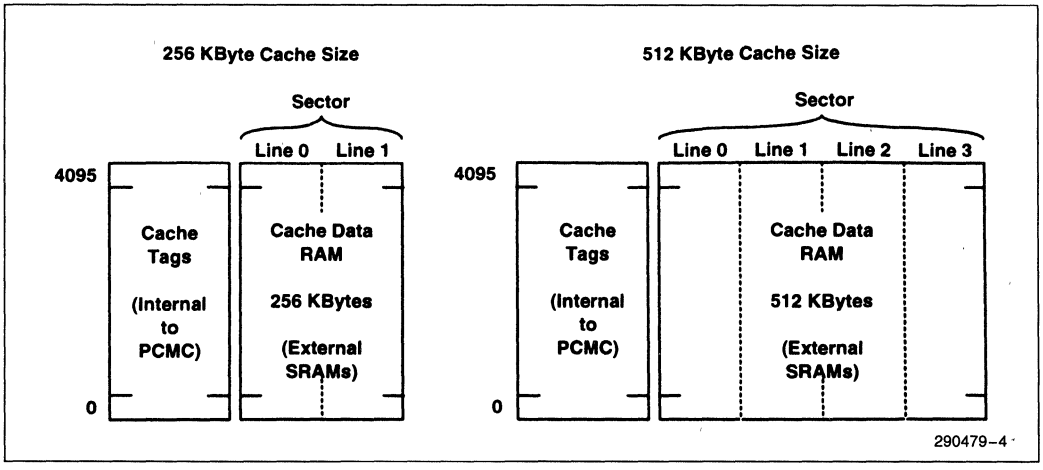


Figure 3. Second Level Cache Organization

If the read cycle causes a cache miss, the line containing the requested data is transferred from main memory to the cache and to the CPU. In the case of a write-back cache, if the cache line fill is to a sector containing one or more modified lines, the modified lines are written back to main memory and the new line is brought into the cache. For a modified line write-back operation, the PCMC transfers the modified cache lines to main memory via a write buffer in the LBX. Before writing the last modified line from the write buffer to main memory, the PCMC updates the first and second level caches with the new line, allowing the CPU access to the requested data with minimum latency.

1.2.1.1 Cache Consistency

The Snoop mechanism in the PCMC ensures data consistency between cache (both first level and second level) and main memory. The PCMC monitors PCI master accesses to main memory and when needed, initiates an inquire (snoop) cycle to the first and second level caches. The snoop mechanism guarantees that consistent data is always delivered to both the host CPU and PCI masters.

1.2.2 ADDRESS/DATA PATHS

Address paths between the CPU/cache and PCI and data paths between the CPU/cache, PCI, and main memory are supplied by two LBX components. The LBX is a companion component to the PCMC. Together, they form a Host/PCI bridge. The PCMC (via the PCMC/LBX interface signals), controls the address and data flow through the LBXs. Refer to the LBX data sheet for more details on the address and data paths.

Data is transferred to and from the PCMC internal registers via the PCMC address lines. When the Host CPU performs a write operation, the data is sent to the LBXs. When the PCMC decodes the cycle as an access to one of its internal registers, it asserts AHOLD to the CPU and instructs the LBXs to copy the data onto the Host address lines. When the PCMC decodes a Host read as an access to a PCMC internal register, it asserts AHOLD to the CPU. The PCMC then places the register data on its address lines and instructs the LBX to copy the data on the Host address bus to the Host data bus. When the register data is on the Host data bus, the PCMC negates AHOLD and completes the cycle.

1.2.2.1 Read/Write Buffers

The LBX provides an interface for the CPU address and data buses, PCI Address/Data bus, and the main memory DRAM data bus. There are three posted write buffers and one read-prefetch buffers implemented in the LBXs to increase performance and to maximize concurrency. The buffers are:

- CPU-to-Main Memory Posted Write Buffer (4 Qwords)
- CPU-to-PCI Posted Write Buffer (4 Dwords)
- PCI-to-Main Memory Posted Write Buffer (2 x 4 Dwords)
- PCI-to-Main Memory Read Prefetch Buffer (line buffer, 4 Qwords).

Refer to the LBX data sheet for details on the operation of these buffers.

2

1.2.3 HOST/PCI BRIDGE OPERATIONS

The PCMC permits the Host CPU to access devices on the PCI Bus. These accesses can be to PCI I/O space, PCI memory space, or PCI configuration space.

As a PCI device, the PCMC can be either a master initiating a PCI Bus operation or a target responding to a PCI Bus operation. The PCMC is a PCI Bus master for Host-to-PCI cycles and a target for PCI-to-main memory transfers. Note that the PCMC does not permit peripherals to be located on the Host Bus. CPU I/O cycles, other than to PCMC internal registers, are forwarded to the PCI Bus and PCI Bus accesses to the Host Bus are not supported.

When the CPU initiates a bus cycle to a PCI device, the PCMC becomes a PCI Bus master and translates the CPU cycle into the appropriate PCI Bus cycle. The Host/PCI Posted write buffer in the LBXs permits the CPU to complete CPU-to-PCI Dword memory writes in three CPU clocks (1 wait-state), even if the PCI Bus is currently busy. The posted data is written to the PCI device when the PCI Bus is available.

When a PCI Bus master initiates a main memory access, the PCMC (and LBXs) become the target of the PCI Bus cycle and responds to the read/write access. During PCI-to-main memory accesses, the PCMC automatically performs cache snoop operations on the Host Bus, when needed, to maintain data consistency.

As a PCI device, the PCMC contains all of the required PCI configuration registers. The Host CPU reads and writes these registers as described in Section 3.0, Register Description.

1.2.4 DRAM MEMORY OPERATIONS

The PCMC contains a DRAM controller that supports CPU and PCI master accesses to main memory. The PCMC DRAM interface supplies the control signals and address lines and the LBXs supply the data path. DRAM parity is generated for main memory writes and checked for memory reads.

For the 82434LX, the memory array is 64-bits wide and ranges in size from 2 MBytes–192 MBytes. The array can be implemented with either single-sided or double-sided SIMMs. DRAM SIMM sizes of 256K x 36, 1M x 36, and 4M x 36 are supported.

For the 82434NX, the memory array is 64-bits wide and ranges in size from 2 MBytes–512 MBytes. The array can be implemented with either single-sided or double-sided SIMMs. DRAM SIMM sizes of 256K x 36, 1M x 36, 4M x 36, and 16M x 36 are supported.

To provide optimum support for the various cache configurations, and the resultant mix of bus cycles, the system designer can select between 0-active RAS# and 1-active RAS# modes. These modes affect the behavior of the RAS# signal following either CPU-to-main memory cycles or PCI-to-main memory cycles.

The PCMC also provides programmable memory and cacheability attributes on 14 memory segments of various sizes in the ISA compatibility range (512 KByte–1 MByte address range). Access rights to these memory segments from the PCI Bus are controlled by the expansion bus bridge.

The PCMC permits a gap to be created in main memory within the 1 MByte–16 MBytes address range, accommodating ISA devices which are mapped into this range (e.g., ISA LAN card or an ISA frame buffer).

1.2.5 3.3V SIGNALS

The 82434NX PCMC drives 3.3V signal levels on the CPU and second level cache interfaces. Thus, no extra logic (i.e. 5V/3.3V translation) is required when interfacing to 3.3V processors and SRAMs. Six of the power pins on the 82434NX are VDD3 pins. These pins are connected to a 3.3V power supply. The VDD3 pins power the output buffers on the CPU and second level cache interfaces. The VDD3 pins also power the output buffers for the HCLK[A-F] outputs.

2.0 SIGNAL DESCRIPTIONS

This section provides a detailed description of each signal. The signals are arranged in functional groups according to their associated interface. The states of all of the signals during hard reset are provided in Section 8.0, System Clocking and Reset.

The "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

The following notations are used to describe the signal type.

in Input is a standard input-only signal

out Totem pole output is a standard active driver

o/d Open drain

t/s Tri-State is a bi-directional, tri-state input/output pin

s/t/s Sustained tri-state is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. An external pull-up is required to sustain the inactive state until another agent drives it and must be provided by the central resource.

2.1 Host Interface

Signal	Type	Description
A[31:0]	t/s	**ADDRESS BUS:** A[31:0] are the address lines of the Host Bus. A[31:3] are connected to the CPU A[31:3] lines and to the LBXs. A[2:0] are only connected to the LBXs. Along with the byte enable signals, the A[31:3] lines define the physical area of memory or I/O being accessed. During CPU cycles, the A[31:3] lines are inputs to the PCMC. They are used for address decoding and second level cache tag lookup sequences. Also during CPU cycles, A[2:0] are outputs and are generated from BE[7:0]#. A[27:24] provide hardware strapping options for test features. For more details on theses options, refer to Section 11.0 Testability. During inquire cycles, A[31:5] are inputs from the LBXs to the CPU and the PCMC to snoop the first and the second level cache tags, respectively. In response to a Flush or Flush Acknowledge Special Cycle, the PCMC asserts AHOLD and drives the addresses of the second level cache lines to be written back to main memory on A[18:7]. During CPU to PCI configuration cycles, the PCMC drives A[31:0] with the PCI configuration space address that is internally derived from the CPU physical I/O address. All PCMC internal configuration registers are accessed via A[31:0]. During CPU reads from PCMC internal configuration registers, the PCMC asserts AHOLD and drives the contents of the addressed register on A[31:0]. The PCMC then signals the LBXs to copy this value from the address lines onto the host data lines. During writes to PCMC internal configuration registers, the PCMC asserts AHOLD and signals the LBXs to copy the write data onto the A[31:0] lines. Finally, when in deturbo mode, the PCMC periodically asserts AHOLD and then drives A[31:0] to valid logic levels to keep these lines from floating for an extended period of time. A[31:28] provide hardware strapping options at powerup. For more details on strapping options, refer to Section 8.0, System Clocking and Reset. A[27:24] provide hardware strapping options for test features. For more details on these options, refer to Section 11.0 Testability.

Signal	Type	Description
BE[7:0]#	in	**BYTE ENABLES:** The byte enables indicate which byte lanes on the CPU data bus carry valid data during the current bus cycle. In the case of cacheable reads, all 8 bytes of data are driven to the Pentium processor, regardless of the state of the byte enables. The byte enable signals indicate the type of special cycle when M/IO# = D/C# = 0 and W/R# = 1. During special cycles, only one byte enable is asserted by the CPU. The following table depicts the special cycle types and their byte enable encodings: **Special Cycle Type** — **Asserted Byte Enable** Shutdown — BE0# Flush — BE1# Halt/Stop Grant — BE2# Write Back — BE3# Flush Acknowledge — BE4# Branch Trace Message — BE5# When the PCMC decodes a Shutdown Special Cycle, it asserts AHOLD, drives 000...000 (the PCI Shutdown Special Cycle Encoding) on the A[31:0] lines and signals the LBXs to latch the host address bus. The PCMC then drives a Special Cycle on PCI, signaling the LBXs to drive the latched address (00...00) on the AD[31:0] lines during the data phase. The PCMC then asserts INIT for 16 HCLKs. In response to Flush and Flush Acknowledge Special Cycles, the PCMC internally inspects the Valid and Modified bits for each of the Second Level Cache Sectors. If a line is both valid and modified, the PCMC drives the cache address of the line on the A[18:7] and CAA/CAB[6:3] lines and writes the line back to main memory. The valid and modified bits are both reset to 0. All valid and unmodified lines are simply marked invalid. In response to a write back special cycle, the PCMC simply returns BRDY# to the CPU. The second level cache will be written back to main memory in response to the following flush special cycle. If BE2# is asserted during a special cycle, the 82434NX uses A4 to determine if the cycle is a Halt or Stop Grant Special Cycle. If A4 = 0, the cycle is a Halt Special Cycle and if A4 = 1, the cycle is a Stop Grant Special cycle. In response to a halt special cycle, the PCMC asserts AHOLD, drives 000...001 (the PCI halt special cycle encoding) on the A[31:0] lines, and signals the LBXs to latch the host address bus. The PCMC then drives a special cycle on PCI, signaling the LBXs to drive the latched address (00...01) on the AD[31:0] lines during the data phase. When the 82434NX PCMC detects a CPU Stop Grant Special Cycle (M/IO# = 0, D/C# = 0, W/R# = 1, A4 = 1, BE[7:0]# = FBh), it generates a PCI Stop Grant Special cycle, with 0002h in the message field (AD[15:0]) and 0012h in the message dependent data field (AD[31:16]) during the first data phase (IRDY# asserted).
ADS#	in	**ADDRESS STROBE:** The Pentium processor asserts ADS# to indicate that a new bus cycle is beginning. ADS# is driven active in the same clock as the address, byte enable, and cycle definition signals. The PCMC ignores a floating low ADS# that may occur when BOFF# is asserted as the CPU is asserting ADS#.

Signal	Type	Description
BRDY#	out	**BURST READY:** BRDY# indicates that the system has responded in one of three ways: 1. valid data has been placed on the Pentium processor data pins in response to a read, 2. CPU write data has been accepted by the system, or 3. the system has responded to a special cycle.
NA#	out	**NEXT ADDRESS:** The PCMC asserts NA# for one clock when the memory system is ready to accept a new address from the CPU, even if all data transfers for the current cycle have not completed. The CPU may drive out a pending cycle two clocks after NA# is asserted and has the ability to support up to two outstanding bus cycles.
AHOLD	out	**ADDRESS HOLD:** The PCMC asserts AHOLD to force the Pentium processor to stop driving the address bus so that either the PCMC or LBXs can drive the bus. During PCI master cycles, AHOLD is asserted to allow the LBXs to drive a snoop address onto the address bus. If the PCI master locks main memory, AHOLD remains asserted until the PCI master locked sequence is complete and the PCI master negates PLOCK#. AHOLD is asserted during all accesses to PCMC internal configuration registers to allow configuration register accesses to occur over the A[31:0] lines. When in deturbo mode, the PCMC periodically asserts AHOLD to prevent the processor from initiating bus cycles in order to emulate a slower system. The duration of AHOLD assertion in deturbo mode is controlled by the Deturbo Frequency Control Register (offset 51h). When PWROK is negated, the PCMC asserts AHOLD to allow the strapping options on A[31:28] to be read. For more details on strapping options, see the System Clocking and Reset section.
EADS#	out	**EXTERNAL ADDRESS STROBE:** The PCMC asserts EADS# to indicate to the Pentium processor that a valid snoop address has been driven onto the CPU address lines to perform an inquire cycle. During PCI master cycles, the PCMC signals the LBXs to drive a snoop address onto the host address lines and then asserts EADS# to cause the CPU to sample the snoop address.
INV	out	**INVALIDATE:** The INV signal specifies the final state (invalid or shared) that a first level cache line transitions to in the event of a cache line hit during a snoop cycle. When snooping the caches during a PCI master write, the PCMC asserts INV with EADS#. When INV is asserted with EADS#, an inquire hit results in the line being invalidated. When snooping the caches during a PCI master read, the PCMC does not assert INV with EADS#. In this case, an inquire cycle hit results in a line transitioning to the shared state.
BOFF#	out	**BACKOFF:** The PCMC asserts BOFF# to force the Pentium processor to abort all outstanding bus cycles that have not been completed and float its bus in the next clock. The PCMC uses this signal to force the CPU to re-order a write-back due to a snoop cycle around a currently outstanding bus cycle. The PCMC also asserts BOFF# to obtain the CPU data bus for write-back cycles from the secondary cache due to a snoop hit. The CPU remains in bus hold until BOFF# is negated.
HITM#	in	**HIT MODIFIED:** The Pentium processor asserts HITM# to inform the PCMC that the current inquire cycle hit a modified line. HITM# is asserted by the Pentium processor two clocks after the assertion of EADS# if the inquire cycle hits a modified line in the primary cache.

Signal	Type	Description
M/IO# D/C# W/R#	in	**BUS CYCLE DEFINITION (MEMORY/INPUT-OUTPUT, DATA/CONTROL, WRITE/READ):** M/IO, D/C# and W/R# define Host Bus cycles as shown in the table below. **M/IO# D/C# W/R# Bus Cycle Type** Low Low Low Interrupt Acknowledge Low Low High Special Cycle Low High Low I/O Read Low High High I/O Write High Low Low Code Read High Low High Reserved High High Low Memory Read High High High Memory Write Interrupt acknowledge cycles are forwarded to the PCI Bus as PCI interrupt acknowledge cycles (i.e. C/BE[3:0]# = 0000 during the address phase). All I/O cycles and any memory cycles that are not directed to memory controlled by the PCMC DRAM controller are forwarded to PCI. The Pentium processor generates six different types of special cycles. The special cycle type is encoded on the BE[7:0]# lines.
HLOCK#	in	**HOST BUS LOCK:** The Pentium processor asserts HLOCK# to indicate the current bus cycle is locked. HLOCK# is asserted in the first clock of the first locked bus cycle and is negated after the BRDY# is returned for the last locked bus cycle. The Pentium processor guarantees HLOCK# to be negated for at least one clock between back-to-back locked operations. When a CPU locked cycle is directed to main memory, the PCMC guarantees that once the locked operation begins in main memory, the CPU has exclusive access to main memory (i.e., PCI master accesses to main memory will not be initiated until the CPU locked operation completes). When a CPU locked cycle is directed to PCI, the PCMC arbitrates for PLOCK# (PCI LOCK#) before initiating the cycle on PCI, except when the cycle is to the memory range defined by the Frame Buffer Range Register and the No Lock Requests bit in that register is set to 1.
CACHE#	in	**CACHEABILITY:** The Pentium processor asserts CACHE# to indicate the internal cacheability of a read cycle or that a write cycle is a burst write-back cycle. If the CPU drives CACHE# inactive during a read cycle, the returned data is not cached, regardless of the state of KEN#. The CPU asserts CACHE# for cacheable data reads, cacheable code fetches, and cache line write-backs. CACHE# is driven along with the cycle definition pins.
KEN#	out	**CACHE ENABLE:** The PCMC asserts KEN# to indicate to the CPU that the current cycle is cacheable. KEN# is asserted for all accesses to memory ranges 0–512-KBytes and 1024-KBytes to the top of main memory controlled by the PCMC when the Primary Cache Enable bit is set to 1, except in the following case: KEN# is not asserted for accesses to the top 64-KByte of main memory controlled by the PCMC when the SMRAM Enable bit in the DRAM Control Register (Offset 57h) is set to 1 and the area is not write protected. If the area is write protected and cacheable, KEN# is asserted for code read cycles, but is not asserted during data read cycle. KEN# is asserted for any CPU access within the range of 512-KBytes–1024-KBytes if the corresponding Cache Enable bit in the PAM[6:0] Registers (offsets 59h–5Fh) is set to 1. When the Pentium processor indicates that the current read cycle can be cached by asserting CACHE# and the PCMC responds with KEN#, the cycle is converted into a burst cache line fill. The CPU samples KEN# with the first of either BRDY# or NA#.

Signal	Type	Description
SMIACT#	in	**SYSTEM MANAGEMENT INTERRUPT ACTIVE:** The Pentium processor asserts SMIACT# to indicate that the processor is operating in System Management Mode (SMM). When the SMRAM Enable bit in the DRAM Control Register (offset 57h) is set to 1, the PCMC allows CPU accesses SMRAM as permitted by the SMRAM Space Register at configuration space offset 72h.
PEN#	out	**PARITY ENABLE:** The PEN# signal, along with the MCE bit in CR4 of the Pentium processor, determines whether a machine check exception will be taken by the CPU as a result of a parity error on a read cycle. The PCMC asserts PEN# during DRAM read cycles if the MCHK on DRAM/L2 Cache Data Parity Error Enable bit in the Error Command Register (offset 70h) is set to 1. The PCMC asserts PEN# during CPU second level cache read cycles if the MCHK on DRAM/L2 Cache Data Parity Error Enable and the L2 Cache Parity Enable bits in the Error Command Register (offset 70h) are both set to 1.
PCHK#	in	**DATA PARITY CHECK:** PCHK# is sampled by the PCMC to detect parity errors on CPU read cycles from main memory if the Parity Error Mask Enable bit in the DRAM Control Register (offset 57h) is reset to 0. PCHK# is sampled by the PCMC to detect parity errors on CPU read cycles from the second level cache if the L2 Cache Parity Enable bit in the Error Command Register (offset 70h) is set to 1. If incorrect parity was detected on a data read, the PCHK# signal is asserted by the Pentium processor two clocks after BRDY# is returned. PCHK# is asserted for one clock for each clock in which a parity error was detected.

2.2 DRAM Interface

Signal	Type	Description
RAS[5:0]#	out	**ROW ADDRESS STROBES:** The RAS[5:0]# signals are used to latch the row address on the MA[10:0] lines into the DRAMs. Each RAS[5:0]# signal corresponds to one DRAM row. The 82434LX PCMC supports up to 6 rows in the DRAM array. Each row is eight bytes wide. These signals drive the RAS# lines of the DRAM array directly, without external buffers.
RAS[7:6]#	out	**ROW ADDRESS STROBES:** The 82434NX supports up to eight rows of DRAM. RAS[7:6]# are used with RAS[5:0] to latch the row address on the MA[11:0] lines into the DRAMs. Each row is eight bytes wide. These signals drive the RAS# lines of the DRAM array directly, without external buffers.
CAS[7:0]#	out	**COLUMN ADDRESS STROBES:** The CAS[7:0]# signals are used to latch the column address on the MA[10:0] lines into the DRAMs. Each CAS[7:0]# signal corresponds to one byte of the eight byte-wide array. These signals drive the CAS# lines of the DRAM array directly, without external buffers. In a minimum configuration, each CAS[7:0]# line only has one SIMM load, while the maximum configuration has 6 SIMM loads.
WE#	out	**DRAM WRITE ENABLE:** WE# is asserted during both CPU and PCI master writes to main memory. During burst writes to main memory, WE# is asserted before the first assertion of CAS[7:0]# and is negated with the last CAS[7:0]#. The WE# signal is externally buffered to drive the WE# inputs on the DRAMs.
MA[10:0]	out	**DRAM MULTIPLEXED ADDRESS:** MA[10:0] provide the row and column address to the DRAM array. The 82434LX uses MA[10:0] for the complete DRAM address bus. The MA[10:0] lines are externally buffered to drive the multiplexed address lines of the DRAM array.
MA11	out	**DRAM MULTIPLEXED ADDRESS:** MA11 provides the extra addressability for the 16M x 36 SIMMs that are supported by the 82434NX. MA[11:0] provide the row and column address to the DRAM array. Like MA[10:0], MA11 is externally buffered to drive the multiplexed address lines of the DRAM array.

2.3 Cache Interface

Signal	Type	Description
CALE	out	**CACHE ADDRESS LATCH ENABLE:** CALE controls the external latch between the host address lines and the cache address lines. CALE is asserted to open the external latch, allowing the host address lines to propagate to the cache address lines. CALE is negated to latch the cache address lines.
CADS[1:0]#, CR/W[1:0]#	out	This signal pin has two functions, depending on the type of SRAMs used for the second level cache. **CACHE ADDRESS STROBE:** CADS[1:0]# are used with burst SRAMs. When asserted, CADS[1:0]# cause the burst SRAMs to latch the cache address on the rising edge of HCLK. CADS[1:0]# are glitch-free synchronous signals. CADS[1:0]# functionality is selected by the SRAM type bit in the Secondary Cache Control Register. Two copies of this signal are provided for timing reasons only. **CACHE READ/WRITE:** CR/W# provide read/write control to the second level cache when using asynchronous dual-byte select SRAMs. This functionality is selected by the SRAM Type and Cache Byte Control Bits in the Secondary Cache Control Register. The two copies of this signal are always driven to the same logic level.
CADV[1:0]#, CCS[1:0]#	out	This signal pin has two functions. The Cache Chip Select function is only enabled when the SRAM connectivity bit (bit 2) in the SCC Register is set to 1. **CACHE ADVANCE:** CADV[1:0]# are used with burst SRAMs to advance the internal two bit address counter inside the SRAMs to the next address of the burst sequence. Two copies of this signal are provided for timing reasons only. The two copies are always driven to the same logic level. **CACHE CHIP SELECT:** CCS[1:0]# are used with asynchronous SRAMs to de-select the SRAMs, placing them in a low power standby mode. When the CPU runs a halt or stop grant special cycle, the 82434NX negates CCS[1:0]#, placing the second level cache in a power saving mode. The PCMC then asserts CCS[1:0]# (activating the SRAMs) when the CPU asserts ADS#. When using burst SRAMs, only CCS1# implements the CCS# function. CADV0# retains the address advance function. CCS1# serve two purposes with burst SRAMs: 1) It is used (along with CADS[1:0]#) to place the SRAMs in a low power standby mode. When the CPU runs a halt or stop grant special cycle, the 82434NX negates CCS1# and asserts CADS[1:0]# for one clock, placing the SRAMs in a power saving mode. The PCMC then asserts CCS1# so that the next ADS# from the CPU places the SRAMs in an active mode. 2) CCS1# is used to block pipelined cycles from the SRAMs when the SRAMs are servicing a cycle. After NA# is asserted, the PCMC negates CCS1# preventing the SRAMs from sampling a new address. CCS1# is asserted again when the SRAMs have completed the current cycle.
CAA[6:3] CAB[6:3]	out	**CACHE ADDRESS [6:3]:** CAA[6:3] and CAB[6:3] are connected to address lines A[3:0] on the second level cache SRAMs. CAA[4:3] and CAB[4:3] are used with standard SRAMs to advance through the burst sequence. CAA[6:5] and CAB[6:5] are used during second level cache write-back cycles to address the modified lines within the addressed sector. Two copies of these signals are provided for timing reasons only. The two copies are always driven to the same logic level.

Signal	Type	Description
COE[1:0]#	out	**CACHE OUTPUT ENABLE:** COE[1:0]# are asserted when data is to be read from the second level cache and are negated at all other times. Two copies of this signal are provided for timing reasons only. The two copies are always driven to the same logic level.
CWE[7:0]#, CBS[7:0]#	out	This signal pin has two functions, depending on the type of SRAMs used for the second level cache. **CACHE WRITE ENABLES:** CWE[7:0]# are asserted to write data to the second level cache SRAMs on a byte-by-byte basis. CWE7# controls the most significant byte while CWE0# controls the least significant byte. These signals are cache write enables when using burst SRAMs (SRAM Type bit in SCC Register is 1) or when using asynchronous SRAMs (SRAM Type bit in SCC Register is 0) and the Cache Byte Control Bit is 1. **CACHE BYTE SELECTS:** The CBS[7:0]# lines provide byte control to the secondary cache when using dual-byte select asynchronous SRAMs. These signals are Cache Byte select lines when the SRAM Type and Cache Byte Control Bits in the SCC Register are both 0.

2.4 PCI Interface

Signal	Type	Description
C/BE[3:0]#	t/s	**PCI BUS COMMAND AND BYTE ENABLES:** C/BE[3:0]# are driven by the current bus master during the address phase of a PCI cycle to define the PCI command, and during the data phase as the PCI byte enables. The PCI commands indicate the current cycle type, and the PCI byte enables indicate which byte lanes carry meaningful data. C/BE[3:0]# are outputs of the PCMC during CPU cycles that are directed to PCI. C/BE[3:0]# are inputs when the PCMC acts as a slave. The command encodings and types are listed below. **C/BE[3:0]# — Command** 0000 — Interrupt Acknowledge 0001 — Special Cycle 0010 — I/O Read 0011 — I/O Write 0100 — Reserved 0101 — Reserved 0110 — Memory Read 0111 — Memory Write 1000 — Reserved 1001 — Reserved 1010 — Configuration Read 1011 — Configuration Write 1100 — Memory Read Multiple 1101 — Reserved 1110 — Memory Read Line 1111 — Memory Write and Invalidate

Signal	Type	Description
FRAME#	s/t/s	**CYCLE FRAME:** FRAME# is driven by the current bus master to indicate the beginning and duration of an access. FRAME# is asserted to indicate that a bus transaction is beginning. While FRAME# is asserted, data transfers continue. When FRAME# is negated, the transaction is in the final data phase. FRAME# is an output of the PCMC during CPU cycles which are directed to PCI. FRAME# is an input to the PCMC when the PCMC acts as a slave.
IRDY#	s/t/s	**INITIATOR READY:** The assertion of IRDY# indicates the current bus master's ability to complete the current data phase. IRDY# works in conjunction with TRDY# to indicate when data has been transferred. On PCI, data is transferred on each clock that both IRDY# and TRDY# are asserted. During read cycles, IRDY# is used to indicate that the master is prepared to accept data. During write cycles, IRDY# is used to indicate that the master has driven valid data on the AD[31:0] lines. Wait states are inserted until both IRDY# and TRDY# are asserted together. IRDY# is an output of the PCMC when the PCMC is the PCI master. IRDY# is an input to the PCMC when the PCMC acts as a slave.
TRDY#	s/t/s	**TARGET READY:** TRDY# indicates the target device's ability to complete the current data phase of the transaction. It is used in conjunction with IRDY#. A data phase is completed on each clock that TRDY# and IRDY# are both sampled asserted. During read cycles, TRDY# indicates that valid data is present on AD[31:0] lines. During write cycles, TRDY# indicates the target is prepared to accept data. Wait states are inserted on the bus until both IRDY# and TRDY# are asserted together. TRDY# is an output of the PCMC when the PCMC is the PCI slave. TRDY# is an input to the PCMC when the PCMC is a master.
DEVSEL#	s/t/s	**DEVICE SELECT:** When asserted, DEVSEL# indicates that the driving device has decoded its address as the target of the current access. DEVSEL# is an output of the PCMC when PCMC is a PCI slave and is derived from the MEMCS# input. MEMCS# is generated by the expansion bus bridge as a decode to the main memory address space. During CPU-to-PCI cycles, DEVSEL# is an input. It is used to determine if any device has responded to the current bus cycle, and to detect a target abort cycle. Master-Abort termination results if no subtractive decode agent exists in the system, and no one asserts DEVSEL# within a programmed number of clocks.
STOP#	s/t/s	**STOP:** STOP# indicates that the current target is requesting the master to stop the current transaction. This signal is used in conjunction with DEVSEL# to indicate disconnect, target-abort, and retry cycles. When PCMC is acting as a master on PCI, if STOP# is sampled active on a rising edge of PCLKIN, FRAME# is negated within a maximum of 3 clock cycles. STOP# may be asserted by the PCMC in three cases. If a PCI master attempts to access main memory when another PCI master has locked main memory, the PCMC asserts STOP# to signal retry. The PCMC detects this condition when sampling FRAME# and LOCK# both active during an address phase. When a PCI master is reading from main memory, the PCMC asserts STOP# when the burst cycle is about to cross a cache line boundary. When a PCI master is writing to main memory, the PCMC asserts STOP# upon filling either of the two PCI-to-main memory posted write buffers. Once asserted, STOP# remains asserted until FRAME# is negated.

Signal	Type	Description
PLOCK#	s/t/s	**PCI LOCK:** PLOCK# is used to indicate an atomic operation that may require multiple transactions to complete. PCI provides a mechanism referred to as "resource lock" in which only the target of the PCI transaction is locked. The assertion of GNT# on PCI does not guarantee control of the PLOCK# signal. Control of PLOCK# is obtained under its own protocol. When the PCMC is the PCI slave, PLOCK# is sampled as an input on the rising edge of PCLKIN when FRAME# is sampled active. If PLOCK# is sampled asserted, the PCMC enters into a locked state and remains in the locked state until PLOCK# is sampled negated on a following rising edge of PCLKIN, when FRAME# is sampled asserted.
REQ#	out	**REQUEST:** The PCMC asserts REQ# to indicate to the PCI bus arbiter that the PCMC is requesting use of the PCI Bus in response to a CPU cycle directed to PCI.
GNT#	in	**GRANT:** When asserted, GNT# indicates that access to the PCI Bus has been granted to the PCMC by the PCI Bus arbiter.
MEMCS#	in	**MAIN MEMORY CHIP SELECT:** When asserted, MEMCS# indicates to the PCMC that a PCI master cycle is targeting main memory. MEMCS# is generated by the expansion bus bridge. MEMCS# is sampled by the PCMC on the rising edge of PCLKIN on the first and second cycle after FRAME# has been asserted.
FLSHREQ#	in	**FLUSH REQUEST:** When asserted, FLSHREQ# instructs the PCMC to flush the CPU-to-PCI posted write buffer in the LBXs and to disable further posting to this buffer as long as FLSHREQ# remains active. The PCMC acknowledges completion of the CPU-to-PCI write buffer flush operation by asserting MEMACK#. MEMACK# remains asserted until FLSHREQ# is negated. FLSHREQ# is driven by the expansion bus bridge and is used to avoid deadlock conditions on the PCI Bus.
MEMREQ#	in	**MEMORY REQUEST:** When asserted, MEMREQ# instructs the PCMC to flush the CPU-to-PCI and CPU-to-main memory posted write buffers and to disable posting in these buffers as long as MEMREQ# is active. The PCMC acknowledges completion of the flush operations by asserting MEMACK#. MEMACK# remains asserted until MEMREQ# is negated. MEMREQ# is driven by the expansion bus bridge.
MEMACK#	out	**MEMORY ACKNOWLEDGE:** When asserted, MEMACK# indicates the completion of the operations requested by an active FLSHREQ# and/or MEMREQ#.
PAR	t/s	**PARITY:** PAR is an even parity bit across the AD[31:0] and C/BE[3:0]# lines. Parity is generated on all PCI transactions. As a master, the PCMC generates even parity on CPU writes to PCI, based on the PPOUT[1:0] inputs from the LBXs. During CPU read cycles from PCI, the PCMC checks parity by checking the value sampled on the PAR input with the PPOUT[1:0] inputs from the LBXs. As a slave, the PCMC generates even parity on PAR, based on the PPOUT[1:0] inputs during PCI master reads from main memory. During PCI master writes to main memory, the PCMC checks parity by checking the value sampled on PAR with the PPOUT[1:0] inputs.

Signal	Type	Description
PERR#	s/t/s	**PARITY ERROR:** PERR# may be pulsed by any agent that detects a parity error during an address phase, or by the master or the selected target during any data phase in which the AD lines are inputs. The PERR# signal is enabled when the PERR# on Receiving Data Parity Error bit in the Error Command Register (offset 70h) and the Parity Error Enable bit in the PCI Command Register (offset 04h) are both set to 1. When enabled, CPU-to-PCI write data is checked for parity errors by sampling the PERR# signal two PCI clocks after data is driven. Also, when enabled, PERR# is asserted by the PCMC when it detects a data parity error on CPU read data from PCI and PCI master write data to main memory. PERR# is neither sampled nor driven by the PCMC when either the PERR# on Receiving Data Parity Error bit in the Error Command Register or the Parity Error Enable bit in the PCI Command Register is reset to 0.
SERR#	o/d	**SYSTEM ERROR:** SERR# may be pulsed by any agent for reporting errors other than parity. SERR# is asserted by the PCMC whenever a serious system error (not necessarily a PCI error) occurs. The intent is to have the PCI central agent (for example, the expansion bus bridge) assert NMI to the processor. Control over the SERR# signal is provided via the Error Command Register (offset 70h) when the Parity Error Enable bit in the PCI Command Register (offset 04h) is set to 1. When the SERR# DRAM/L2 Cache Data Parity Error bit is set to 1, SERR# is asserted upon detecting a parity error on CPU read cycles from DRAM. If the L2 Cache Parity bit is also set to 1, SERR# will be asserted upon detecting a parity error on CPU read cycles from the second level cache. The Pentium processor indicates these parity errors to the PCMC via the PCHK# signal. When the SERR# on PCI Address Parity Error bit is set to 1, the PCMC asserts SERR# if a parity error is detected during the address phase of a PCI master cycle. When the SERR# on Received PCI Data Parity bit is set to 1, the PCMC asserts SERR# if a parity error is detected on PCI during a CPU read from PCI. During CPU to PCI write cycles, when the SERR# on Transmitted PCI Data Parity Error bit is set to 1, the PCMC asserts SERR# in response to sampling PERR# active. When the SERR# on Received Target Abort bit is set to 1, the PCMC asserts SERR# when the PCMC receives a target abort on a PCMC initiated PCI cycle. If the Parity Error Enable bit in the PCI Command Register is reset to 0, SERR# is disabled and is never asserted by the PCMC.

2.5 LBX Interface

Signal	Type	Description
HIG[4:0]	out	**HOST INTERFACE GROUP:** HIG[4:0] are outputs of the PCMC used to control the LBX HA (Host Address) and HD (Host Data) buses. Commands driven on HIG[4:0] cause the host data and/or address lines to be either driven or latched by the LBXs. See the 82433LX (LBX) Local Bus Accelerator Data Sheet for a listing of the HIG[4:0] commands.
MIG[2:0]	out	**MEMORY INTERFACE GROUP:** MIG[2:0] are outputs of the PCMC and control the LBX MD (Memory Data) bus. Commands driven on the MIG[2:0] lines cause the memory data lines to be either driven or latched by the LBXs. See the 82433LX (LBX) Local Bus Accelerator Data Sheet for a listing of the MIG[2:0] commands.
MDLE	out	**MEMORY DATA LATCH ENABLE:** During CPU reads from main memory, MDLE is used to control the latching of memory read data on the CPU data bus. MDLE is negated as CAS[7:0]# are negated to close the latch between the memory data bus and the host data bus. During CPU reads from main memory, the PCMC closes the memory data to host data latch in the LBXs as BRDY# is asserted and opens the latch after the CPU has sampled the data.
PIG[3:0]	out	**PCI INTERFACE GROUP:** PIG[3:0] are outputs of the PCMC used to control the LBX AD (PCI Address/Data) bus. Commands driven on the PIG[3:0] lines cause the AD lines to be either driven or latched. See the 82433LX (LBX) Local Bus Accelerator Data Sheet for a listing of the PIG[3:0] commands.
DRVPCI	out	**DRIVE PCI:** DRVPCI acts as an output enable for the LBX AD lines. When sampled asserted, the LBXs begin driving the PCI AD lines. When negated, the AD lines on the LBXs are tri-stated. The LBX AD lines are tri-stated asynchronously from the falling edge of DRVPCI.
EOL	in	**END OF LINE:** EOL is asserted by the low order LBX when a PCI master read or write transaction is about to overrun a cache line boundary. EOL has an internal pull-up resistor inside the PCMC. The low order LBX EOL signal connects to this PCMC input. The high order LBX EOL signal is connected to ground through an external pull-down resistor.
PPOUT[1:0]	in	**PCI PARITY OUT:** These signals reflect the parity of the 32 AD lines driven from or latched in the LBXs, depending on the command driven on PIG[3:0]. The PPOUT0 pin has a weak internal pull-down resistor. The PPOUT1 pin has a weak internal pull-up resistor.

2.6 Reset And Clock

Signal	Type	Description
HCLKOSC	in	**HOST CLOCK OSCILLATOR:** The HCLKOSC input is driven externally by a crystal oscillator. The PCMC generates six copies of HCLK from HCLKOSC (HCLKA–HCLKF). During power-up, HCLKOSC must stabilize for 1 ms before PWROK is asserted. If an external clock driver is used to clock the CPU, PCMC, LBXs and second level cache SRAMs instead of the HCLKA–HCLKF outputs, HCLKOSC must be tied either high or low.
HCLKA–HCLKF	out	**HOST CLOCK OUTPUTS:** HCLKA–HCLKF are six low skew copies of the host clock. These outputs eliminate the need for an external low skew clock driver.

Signal	Type	Description
HCLKIN	in	**HOST CLOCK INPUT:** All timing on the host, DRAM and second level cache interfaces is based on HCLKIN. If an external clock driver is used to clock the CPU, PCMC, LBXs and second level cache SRAMs, the externally generated clock must be connected to HCLKIN. During power-up HCLKIN must stabilize for 1 ms before PWROK is asserted.
CPURST	out	**CPU HARD RESET:** The CPURST pin is asserted in response to one of two conditions. **Powerup** 82434LX: During powerup the 82434LX asserts CPURST when PWROK is negated. When PWROK is asserted, the 82434LX first ensures that it has been initialized before negating CPURST. 82434NX: During powerup, the 82434NX PCMC negates CPURST while PWROK is negated. When PWROK is asserted, the 82434NX asserts CPURST for 2 ms. **Software** CPURST is also asserted when the System Hard Reset Enable bit in the Turbo-Reset Control Register (I/O address 0CF9h) is set to 1 and the Reset CPU bit toggles from 0 to 1 (82434LX and 82434NX). CPURST is driven synchronously to the rising edge of HCLKIN.
INIT	out	**INITIALIZATION:** INIT is asserted in response to any one of two conditions. When the System Hard Reset Enable bit in the Turbo-Reset Control Register is reset to 0 and the Reset CPU bit toggles from 0 to 1, the PCMC initiates a soft reset by asserting INIT. The PCMC also initiates a soft reset by asserting INIT in response to a shutdown special cycle. In both cases, INIT is asserted for a minimum of 2 Host clocks.
PWROK	in	**POWER OK:** When asserted, PWROK is an indication to the PCMC that power and HCLKIN have stabilized for at least 1 ms. PWROK can be driven asynchronously. 82434LX: When PWROK is negated, the 82434LX asserts both CPURST and PCIRST#. When PWROK is driven high, the 82434LX ensures that it is initialized before negating CPURST and PCIRST#. 82434NX: When PWROK is negated, the 82434NX negates CPURST and asserts PCIRST#. When PWROK is asserted, the 82434NX asserts CPURST for 2 ms. PCIRST# is negated 1 ms after PWROK is asserted.
PCLKOUT	out	**PCI CLOCK OUTPUT:** PCLKOUT is internally generated by a Phase Locked Loop (PLL) that divides the frequency of HCLKIN by 2. This output must be buffered externally to generate multiple copies of the PCI Clock. One of the copies must be connected to the PCLKIN pin.

Signal	Type	Description
PCLKIN	in	**PCI CLOCK INPUT:** An internal PLL locks PCLKIN in phase with HCLKIN. All timing on the PCMC PCI interface is referenced to the PCLKIN input. All output signals on the PCI interface are driven from PCLKIN rising edges and all input signals on the PCI interface are sampled on PCLKIN rising edges.
PCIRST#	out	**PCI RESET:** PCIRST# is asserted to initiate hard reset on PCI. PCIRST# is asserted in response to one of two conditions. **Power-up** During power-up the PCMC asserts PCIRST# when PWROK is negated. 82434LX: When PWROK is asserted the PCMC will first ensure that it has been initialized before negating PCIRST#. 82434NX: When PWROK is negated, the 82434NX asserts PCIRST#. The 82434NX then negates PCIRST# 1 ms after PWROK is asserted. **Software** PCIRST# is also asserted when the System Hard Reset Enable bit in the Turbo/Reset Control Register is set to 1 and the Reset CPU bit toggles from 0 to 1 (82434LX and 82434NX). PCIRST# is driven asynchronously.
TESTEN	in	**TEST ENABLE:** TESTEN must be tied low for normal system operation.

3.0 REGISTER DESCRIPTION

The 82434LX/82434NX PCMC contains two sets of software accessible registers. These registers are accessed via the Host CPU I/O address space. The PCMC also contains a set of configuration registers that reside in PCI configuration space and are used to specify PCI configuration, DRAM configuration, cache configuration, operating parameters and optional system features (see Section 3.2, PCI Configuration Space Mapped Registers). The PCMC internal registers (both I/O Mapped and Configuration registers) are only accessible by the Host CPU and cannot be accessed by PCI masters. The registers can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities. All multi-byte numeric fields use "little-endian" ordering (i.e., lower addresses contain the least significant parts of the field).

Some of the PCMC registers described in this section contain reserved bits. These bits are labeled "R". Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back.

In addition to reserved bits within a register, the PCMC contains address locations in the PCI configuration space that are marked "Reserved" (Table 1). The PCMC responds to accesses to these address locations by completing the Host cycle. When a reserved register location is read, 0000h is returned. Writes to reserved registers have no affect on the PCMC.

Upon receiving a hard reset via the PWROK signal, the PCMC sets its internal configuration registers to predetermined **default** states. The default state represents the minimum functionality feature set required to successfully bring up the system. Hence, it does not represent the optimal system configuration. It is the responsibility of the system initialization software (usually BIOS) to properly determine the DRAM configurations, cache configuration, operating parameters and optional system features that are applicable, and to program the PCMC registers accordingly.

The following nomenclature is used for access attributes.

RO **Read Only.** If a register is read only, writes to this register have no effect.

R/W **Read/Write.** A register with this attribute can be read and written.

R/WC **Read/Write Clear.** A register bit with this attribute can be read and written. However, a write of a 1 clears (sets to 0) the corresponding bit and a write of a 0 has no effect.

3.1 I/O Mapped Registers

The 82434LX PCMC contains three registers that reside in the CPU I/O address space—the Configuration Space Enable (CSE) Register, the Turbo-Reset Control (TRC) Register and the Forward (FORW) Register. These registers can not reside in PCI configuration space because of the special functions they perform. The CSE Register enables/disables the configuration space and, hence, can not reside in that space. The TRC Register enables/disables deturbo mode which effectively slows the processor to accommodate software programs that rely on the slow speed of PC/XT systems to time certain events. The FORW Register determines which of the possible hierarchical PCI Buses a cycle is directed. The 82434LX uses mechanism #2 for accessing PCI configuration space.

The 82434NX PCMC contains five registers that reside in the CPU I/O address spacethe Configuration Address (CONFADD) Register, the Configuration Space Enable (CSE) Register, the Turbo-Reset Control (TRC) Register, the Forward (FORW) Register, and the PCI Mechanism Control (PMC) Register. The CSE, TRC, and FORW Registers are the same for both the 82434LX and 82434NX PCMCs. The 82434NX can use either Configuration Access Mechanism #1 or #2 for accessing PCI configuration space. When Configuration Access Mechanism #1 is used (See Section 3.2, PCI Configuration Space Mapped Registers), The CONFADD Register enables/disables the configuration space and determines what portion of configuration space is visible through the Configuration Data (CONFDATA) window. The CSE and FORW Registers are used for Configuration Access Mechanism #2. The PCI Mechanism Control (PMC) Register selects whether Configuration Access Mechanism 1 or 2 is used (see the Rev 2.0 PCI Local Bus Specification).

3.1.1 CONFADD—CONFIGURATION ADDRESS REGISTER

I/O Address: 0CF8h Accessed as a Dword
Default Value: 00000000h
Access: Read/Write
Size: 32 bits

CONFADD is a 32-bit register used in Configuration Access Mechanism #1. It is accessed only when referenced as a Dword and PCAMS in the PMC Register is set to 1. Byte or Word references "pass through" the CONFADD Register to the I/O locations "behind" it. For example a byte access to 0CF8h will access the CSE Register, while a word access to CF8h will access both the CSE and TRC Registers. The CONFADD Register contains the Bus Number, Device Number, Function Number, and Register Number where the CONFDATA window is located.

Bit	Description
31	**CONFIGURATION ENABLE (CONE)—R/W:** When CONE = 1, accesses to PCI configuration space are enabled, if the PCAMS bit of the PMC register is also 1. When CONE = 0, accesses to PCI configuration space are disabled, if the PCAMS bit is 1. If the PCAMS bit is 0, this bit has no effect.
30:24	**RESERVED**
23:16	**BUS NUMBER (BUSNUM)—R/W:** When the BUSNUM is programmed to 00h, the target of the Configuration Cycle is either the PCMC or the PCI Local Bus that is directly connected to the PCMC. PCI Access Mechanism #1 can generate either type 0 or type 1 configuration cycles on PCI. A type 0 Configuration Cycle is generated on PCI if the Bus Number is programmed to 00h and the PCMC is not the target. If the Bus Number is non-zero a type 1 configuration cycle is generated on PCI with the Bus Number mapped to AD[23:16] during the address phase.
15:11	**DEVICE NUMBER (DEVNUM)—R/W:** This field selects one agent on the PCI Bus selected by the Bus Number. During a Type 1 Configuration cycle this field is mapped to AD[15:11]. During a Type 0 Configuration Cycle this field is decoded and one of AD[31:17] is driven to a 1. The PCMC is always Device Number 0.
10:8	**FUNCTION NUMBER (FUNCNUM)—R/W:** This field is mapped to AD[10:8] during PCI configuration cycles. This allows the configuration registers of a particular function in a multi-function device to be accessed.
7:2	**REGISTER NUMBER (REGNUM)—R/W:** This field selects one register within a particular Bus, Device, and Function as specified by the other fields in the Configuration Address Register. REGNUM is mapped to AD[7:2] during PCI configuration cycles.
1:0	**RESERVED**

3.1.2 CSE—CONFIGURATION SPACE ENABLE REGISTER

I/O Address: 0CF8h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The CSE Register enables/disables configuration space access and provides access to specific functions within a PCI agent. The register is located in the CPU I/O address space. The PCMC, as a Host/PCI Bridge, supports multi-function devices on the PCI Bus. The function number permits individual configuration spaces for up to eight functions within an agent. The register is located in the CPU I/O address space.

Bit	Description
7:4	**KEY FIELD (KEY)—R/W:** This field is used only when the PCI Mechanism Control Register (PMC) indicates Configuration Access Mechanism 2 is to be used. When the key field is programmed to 0h, the PCI configuration space is disabled. When the key field is programmed to a non-zero value, all CPU accesses to CnXXh (where n is a non zero value) are forwarded to PCI as configuration space accesses. Additionally, when the key field is programmed to a non-zero value, all CPU accesses to C0XXh are intercepted by the PCMC and directed to a PCMC internal register.
3:1	**FUNCTION NUMBER (FN)—R/W:** For multi-function devices, this field selects a particular function within a PCI device. During a configuration cycle, bits[3:1] become part of the PCI Bus address and correspond to AD[10:8].
0	**RESERVED**

3.1.3 TRC—TURBO-RESET CONTROL REGISTER

I/O Address: 0CF9h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The TRC Register is an 8-bit read/write register that selects turbo/deturbo mode of the CPU, initiates PCI Bus and CPU reset cycles, and initiates the CPU Built In Self Test (BIST). TRC is located in CPU I/O address space.

Bit	Description
7:3	**RESERVED**
2	**RESET CPU (RCPU)—R/W:** RCPU is used to initiate a hard reset or soft reset to the CPU. During a hard reset, the PCMC asserts CPURST and PCIRST#. The PCMC initiates a hard reset when this register is programmed for a hard reset or when the PWROK signal is asserted. During a soft reset, the PCMC asserts INIT. The PCMC initiates a soft reset when this register is programmed for a soft reset and in response to a shutdown special cycle. Note that a hard reset initializes the entire system and invalidates the CPU cache. A soft reset initializes only the CPU. The contents of the CPU cache are unaffected. This bit is used in conjunction with bit 1 of this register. Bit 1 must be set up prior to writing a 1 to this register. Thus, two write operations are required to initiate a reset using this bit. The first write operation programs bit 1 to the appropriate state while setting this bit to 0. The second write operation keeps bit 1 at the programmed state (1 or 0) while setting this bit to a 1. When RCPU transitions from a 0 to a 1, a hard reset is initiated if bit 1 = 1 and a soft reset is initiated if bit 1 = 0.
1	**SYSTEM HARD RESET ENABLE (SHRE)—R/W:** This bit is used in conjunction with bit 2 of this register to initiate either a hard or soft reset. When SHRE = 1, the PCMC initiates a hard reset to the CPU when bit 2 transitions from 0 to 1. When SHRE = 0, the PCMC initiates a soft reset when bit 2 transitions from 0 to 1.
0	**DETURBO MODE (DM)—R/W:** This bit enables and disables deturbo mode. When DM = 1, the PCMC is in the deturbo mode. In this mode, the PCMC periodically asserts the AHOLD signal to slow down the effective speed of the CPU. The AHOLD duty cycle is programmable through the Deturbo Frequency Control (DFC) Register. When DM = 0, the deturbo mode is disabled. Deturbo mode can be used to maintain backward compatibility with older software packages that rely on the operating speed of older processors. For accurate speed emulation, caching should be disabled. If caching is disabled during runtime, the following steps should be performed to make sure that modified lines have been flushed from the cache to main memory before entering deturbo mode. Disable the primary cache via the PCE bit in the HCS Register. This prevents the KEN# signal from being asserted, which prevents any further first and second level cache line fills. At this point, software executes the WBINVD instruction to flush the caches, and then sets DM to 1. When exiting the deturbo mode, the system software must first set DM to 0, then enable first and second level caching by writing to the HCS Register.

3.1.4 FORW—FORWARD REGISTER

I/O Address: 0CFAh
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

This 8-bit register specifies which PCI Bus configuration space is enabled in a multiple PCI Bus configuration. The default value for the FORW Register enables the configuration space of the PCI Bus connected to the PCMC.

Bit	Description
7:0	**FORWARD BUS NUMBER—R/W:** When this register value is 00h, the configuration space of the PCI Bus connected to the PCMC is enabled and the PCMC initiates a type 0 configuration cycle. If the value of this register is not 00h, the PCMC initiates a type 1 configuration cycle to forward the cycle (via one or more PCI/PCI Bridges) to the PCI Bus specified by the contents of this register. For non-zero values, bits[7:0] are mapped to AD[23:16], respectively.

3.1.5 PMC—PCI MECHANISM CONTROL REGISTER

I/O Address: 0CFBh
Default Value: 00h
Access: Read/Write
Size: 8 bits

The PMC Register selects whether PCI Configuration Access Mechanism 1 or 2 is to be used. The register is located in the CPU I/O address space.

Bit	Description
7:1	**RESERVED**
0	**PCI CONFIGURATION ACCESS MECHANISM SELECT (PCAMS)—R/W:** When PCAMS = 0, the PCMC uses to PCI Configuration Access Mechanism #2. When PCAMS = 1, the PCMC uses to PCI Configuration Access Mechanism #1. The CONFADD and CONFDATA Registers are only accessible when PCAMS = 1.

3.1.6 CONFDATA—CONFIGURATION DATA REGISTER

I/O Address: 0CFCh
Default Value: 00h
Access: Read/Write
Size: 32 bits

CONFDATA is a 32 bit read/write window into configuration space. The portion of configuration space that is referenced by CONFDATA is determined by the contents of CONFADD.

Bit	Description
31:0	**CONFIGURATION DATA WINDOW (CDW)—R/W:** When using Configuration Access Mechanism #1 if bit 31 of CONFADD is 1 any I/O reference that falls in the CONFDATA I/O space will be mapped to configuration space using the contents of CONFADD.

3.2 PCI Configuration Space Mapped Registers

The PCI Bus defines a slot based "configuration space" that allows each device to contain up to 256 8-bit configuration registers. The PCI specification defines two bus cycles to access the PCI configuration space—**Configuration Read** and **Configuration Write**. While memory and I/O spaces are supported by the Pentium processor, configuration space is not supported. For PCI configuration space access, the PCMC translates the Pentium processor I/O cycles into PCI configuration cycles. Table 1 shows the PCMC configuration space.

Table 1. PCMC Configuration Space

Address Offset	Register Symbol	Register Name	Access
00–01h	VID	Vendor Identification	RO
02–03h	DID	Device Identification	RO
04–05h	PCICMD	Command Register	R/W
06–07h	PCISTS	Status Register	RO, R/WC
08h	RID	Revision Identification	RO
09h	RLPI	Register-Level Programming Interface	RO
0Ah	SCCD	Sub-Class Code	RO
0Bh	BCCD	Base Class Code	RO
0Ch	—	Reserved	—
0Dh	MLT	Master Latency Timer	R/W
0Eh	—	Reserved	—
0Fh	BIST	BIST Register	RO
10–4Fh	—	Reserved	—
50h	HCS	Host CPU Selection	R/W
51h	DFC	Deturbo Frequency Control	R/W
52h	SCC	Secondary Cache Control	R/W
53h	HBC	Host Read/Write Buffer Control	R/W
54h	PBC	PCI Read/Write Buffer Control	R/W
55h	—	Reserved	—
56h	—	Reserved	—
57h	DRAMC	DRAM Control	R/W
58h	DRAMT	DRAM Timing	R/W
59–5Fh	PAM[6:0]	Programmable Attribute Map (7 Registers)	R/W
60–65h	DRB[5:0]	DRAM Row Boundary (6 Registers)	R/W
66–67h	DRB[7:6]	DRAM Row Boundary (2 Registers)	R/W
68–6Bh	DRBE	DRAM Row Boundary Extension	R/W
6C–6Fh	—	Reserved	—
70h	ERRCMD	Error Command	R/W

Table 1. PCMC Configuration Space (Continued)

Address Offset	Register Symbol	Register Name	Access
71h	ERRSTS	Error Status	R/WC
72h	SMRS	SMRAM Space Control	R/W
73–77h	—	Reserved	—
78–79h	MSG	Memory Space Gap	R/W
7A–7B	—	Reserved	—
7C–7Fh	FBR	Frame Buffer Range	R/W
80–FFh	—	Reserved	—

NOTE:
Shaded rows indicate register differences between the 82434LX and 82434NX devices. For non-shaded rows, the registers are the same for the two devices.

3.2.1 CONFIGURATION SPACE ACCESS MECHANISM

The 82434LX supports Configuration Space Access Mechanism #2 and the 82434NX supports both configuration space access mechanisms #1 and #2. The mechanism is selected via the PCAMS bit in the PMC Register. The bus cycles used to access PCMC internal configuration registers are described in Section 7.0, PCI Interface.

3.2.1.1 Access Mechanism #1:

For configuration access mechanism #1, the 82434NX PCMC uses the CONFADD and CONFDATA Registers. Note that while the CONFADD and PMC Register address spaces overlap, the CONFADD Register is referenced only by a Dword read or write to CF8h. This allows the PMC Register to be accessed by a byte write to CFBh, even when using configuration access mechanism #1.

To reference a configuration register with access mechanism #1, a Dword I/O write loads the CONFADD Register with a 32-bit value that specifies the PCI Bus, the device on that bus, the function within the device, and a specific configuration register of the device function being accessed (Figure 4). Bit 31 of the CONFADD Register must be 1 to enable a configuration cycle. CONFDATA then becomes a four byte window of configuration space specified by the contents of the CONFADD Register. A read or write to CONFDATA results in the PCMC translating CONFADD into a PCI configuration cycle.

Type 0 Access

If the BUSNUM field is 0, a Type 0 configuration cycle is performed on the PCI. Bus CONFADD[10:2] are mapped directly to AD[10:2]. The DEVNUM field is decoded onto AD[31:17] and AD[15:11] (for accesses to device 1, AD17 is asserted; for accesses to device #2, AD18 is asserted; etc.). The PCMC is Device #0 and does not pass its configuration cycles to the PCI Bus. Thus, AD16 is never asserted. For accesses to device 15, AD31 is asserted, etc. This mapping allows the same Device Number to activate the same AD line in either configuration access mechanism. All other AD lines are 0.

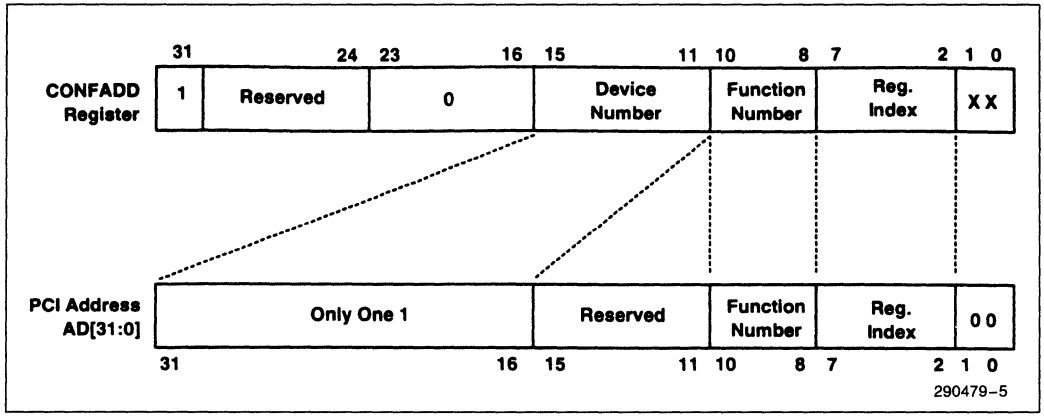


Figure 4. Mechanism #1 Type 0 Configuration Address to PCI Address Mapping

2

Type 1 Access

If the BUSNUM field of the CONFADD Register is non-zero, a Type 1 configuration cycle is performed on the PCI Bus. CONFADD[23:2] are mapped directly to AD[23:2] (Figure 5). AD[1:0] are driven to 01 to indicate a Type 1 Configuration cycle. All other lines are driven to 0.

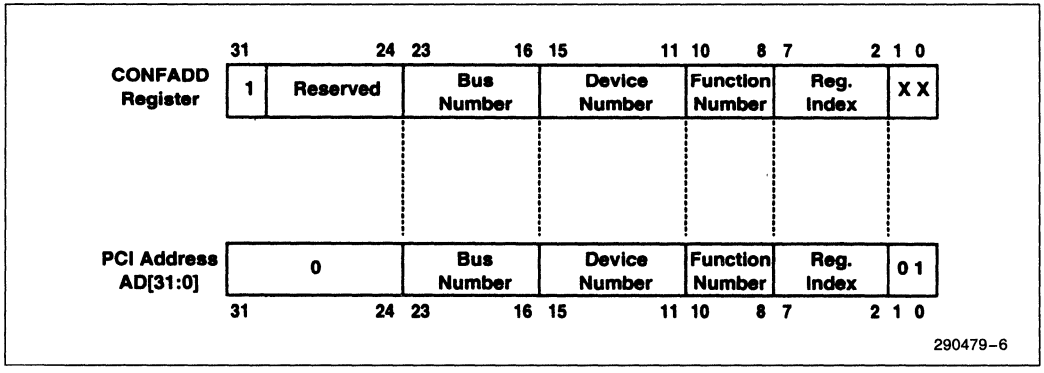


Figure 5. Mechanism #1 Type 1 Configuration Address to PCI Address Mapping

3.2.1.2 Access Mechanism #2

The 82434LX/82434NX PCMC uses the CSE and Forward Registers for configuration access mechanism #2. When PCI configuration space is enabled via the CSE Register, the PCMC maps PCI configuration space into 4-KBytes of CPU I/O space. Each PCI device has its own 256-Byte configuration space. When configuration space is enabled, CPU accesses to I/O locations CXXXh are translated into configuration space accesses. In this mode, the PCMC translates all I/O cycles in the C100h–CFFFh range into configuration cycles on the PCI Bus. I/O accesses within the C000h–C0FFh range are intercepted by the PCMC and are directed to the PCMC internal configuration registers. These cycles are not forwarded to the PCI Bus.

When configuration space access is disabled, CPU accesses to I/O locations CXXXh are forwarded to the PCI Bus I/O space. CPU cycles to I/O locations other than CXXXh are unaffected by whether the configuration mode is enabled or disabled. These cycles are always treated as ordinary I/O cycles by the PCMC.

Type 0 Access

If the Forward Register contains 00h a Type 0 configuration access is generated on the PCI Bus (Figure 6). For type 0 configuration cycles, AD[1:0] = 00. Host CPU address bits A[7:2] are not translated and become AD[7:2] on the PCI Bus. AD[7:2] select one of the 256 8-bit I/O locations in the PCI configuration space. The FUNCTION NUMBER field from the CSE Register (CSE[3:1]) is driven on AD[10:8]. Host CPU address bits A[11:8] are mapped to an IDSEL input for each of the 16 possible PCI devices. The IDSEL input for each PCI device must be hard-wired to one of the AD[31:16] signals on the PCI Bus. AD16 is reserved for the PCMC. When CPU address A[11:8] = Fh, PCI address bits A31 = 1 and A[30:16] = 00h. Other devices on the PCI Bus should not use AD16. Note that when A[11:8] = 0h, an access to the PCMC internal registers occurs and the cycle is not forwarded to the PCI Bus.

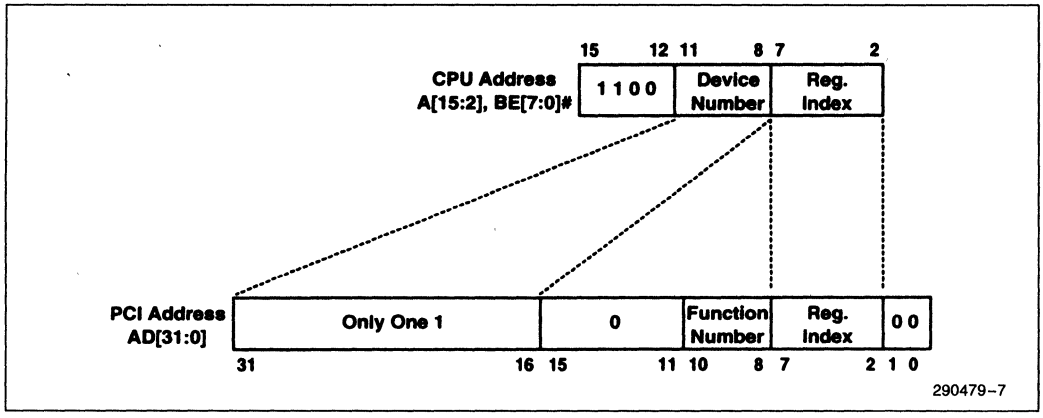


Figure 6. Mechanism #2 Type 0 Host-to-PCI Address Mapping

Type 1 Access

If the Forward Register is non-zero a Type 1 configuration access is generated on PCI. For type 1 configuration cycles, AD[1:0] = 01. AD[10:2] are generated the same as for the type 0 configuration cycle. Host CPU address bits A[11:8] contain the specific device number and are mapped to AD[14:11]. AD[23:16] contain the Bus Number of the PCI Bus that is to be accessed and corresponds to the Forward Address Register bits [7:0].

During a Type 1 configuration access AD[1:0] = 01 (Figure 7). The Register Index and Function Number are mapped to the AD lines the same way in Type 1 configuration access as in a Type 0 configuration access. CPU address bits A[11:8] are mapped directly to PCI lines AD[14:11] as the Device Number. The contents of the Forward Register are mapped to AD[23:16] to form the Bus Number.

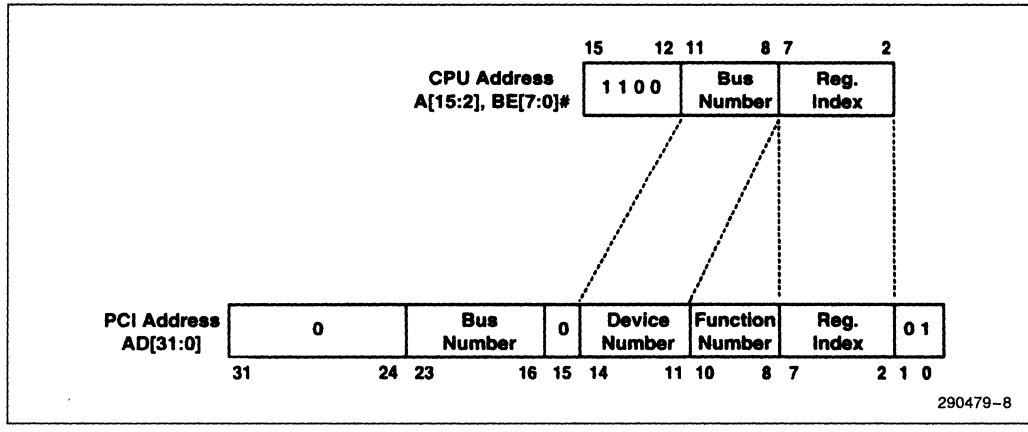


Figure 7. Mechanism #2 Type 1 Host-to-PCI Address Mapping

3.2.2 VID—VENDOR IDENTIFICATION REGISTER

Address Offset: 00–01h
Default Value: 8086h
Attribute: Read Only
Size: 16 bits

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identify any PCI device. Writes to this register have no effect.

Bits	Description
15:0	**VENDOR IDENTIFICATION NUMBER:** This is a 16-bit value assigned to Intel.

3.2.3 DID—DEVICE IDENTIFICATION REGISTER

Address Offset: 02–03h
Default Value: 04A3h
Attribute: Read Only
Size: 16 bits

This 16-bit register combined with the Vendor Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Bits	Description
15:0	**DEVICE IDENTIFICATION NUMBER:** This is a 16 bit value assigned to the PCMC.

3.2.4 PCICMD—PCI COMMAND REGISTER

Address Offset: 04–05h
Default: 06h
Attribute: Read/Write
Size: 16 bits

This 16-bit register provides basic control over the PCMC's ability to respond to PCI cycles. The PCICMD Register enables and disables the SERR# signal, the parity error signal (PERR#), PCMC response to PCI special cycles, and enables and disables PCI master accesses to main memory.

Bits	Description
15:9	**RESERVED**
8	**SERR# ENABLE (SERRE):** SERRE enables/disables the SERR# signal. When SERRE = 1 and PERRE = 1, SERR# is asserted if the PCMC detects a PCI Bus address/data parity error, or main memory (DRAM) or cache parity error, and the corresponding errors are enabled in the Error-Command Register. When SERRE = 1 and bit 7 in the Error Command Register is set to 1, the PCMC asserts SERR# when it detects a target abort on a PCMC-initiated PCI cycle. When SERRE = 0, SERR# is never asserted.
7	**RESERVED**
6	**PARITY ERROR ENABLE (PERRE):** PERRE controls the PCMC's response to PCI parity errors. This bit is a master enable for bit 3 of the ERRCMD Register. PERRE works in conjunction with the SERRE bit to enable SERR# assertion when the PCMC detects a PCI bus parity error, or a main memory or cache parity error.
5:3	**RESERVED**
2	**BUS MASTER ENABLE (BME):** The PCMC does not support disabling of its bus master capability on the PCI Bus. This bit is always set to 1, permitting the PCMC to function as a PCI Bus master. Writes to this bit position have no affect.
1	**MEMORY ACCESS ENABLE (MAE):** This bit enables/disables PCI master access to main memory (DRAM). When MAE = 1, the PCMC permits PCI masters to access main memory if the MEMCS# signal is asserted. When MAE = 0, the PCMC does not respond to PCI master main memory accesses (MEMCS# asserted).
0	**I/O ACCESS ENABLE (IOAE):** The PCMC does not respond to PCI I/O cycles, hence this command is not supported. PCI master access to I/O space on the Host Bus is always disabled.

3.2.5 PCISTS—PCI STATUS REGISTER

Address Offset: 06–07h
Default Value: 40h
Attribute: Read Only, Read/Write Clear
Size: 16 bits

PCISTS is a 16-bit status register that reports the occurrence of a PCI master abort, PCI target abort, and DRAM or cache parity error. PCISTS also indicates the DEVSEL# timing that has been set by the PCMC hardware. Bits[15:12] are read/write clear and bits[10:9] are read only.

Bits	Attribute	Description
15		**RESERVED**
14	R/WC	**SIGNALED SYSTEM ERROR (SSE):** When the PCMC asserts the SERR# signal, this bit is also set to 1. Software sets SSE to 0 by writing a 1 to this bit.
13	R/WC	**RECEIVED MASTER ABORT STATUS (RMAS):** When the PCMC terminates a Host-to-PCI transaction (PCMC is a PCI master), which is not a special cycle, with a master abort, this bit is set to 1. Software resets this bit to 0 by writing a 1 to it.
12	R/WC	**RECEIVED TARGET ABORT STATUS (RTAS):** When a PCMC-initiated PCI transaction is terminated with a target abort, RTAS is set to 1. The PCMC also asserts SERR# if the SERR# Target Abort bit in the ERRCMD Register is 1. Software resets RTAS to 0 by writing a 1 to it.
11		**RESERVED**
10:9	RO	**DEVSEL# TIMING (DEVT):** This 2-bit field indicates the timing of the DEVSEL# signal when the PCMC responds as a target. The PCI specification defines three allowable timings for assertion of DEVSEL#: 00 = fast, 01 = medium, and 10 = slow (DEVT = 11 is reserved). DEVT indicates the slowest time that a device asserts DEVSEL# for any bus command, except configuration read and write cycles. Note that these two bits determine the slowest time that the PCMC asserts DEVSEL#. However, the PCMC can also assert DEVSEL# in medium time. The PCMC asserts DEVSEL# in response to sampling MEMCS# asserted. The PCMC samples MEMCS# one and two clocks after FRAME# is asserted. If MEMCS# is asserted one PCI clock after FRAME# is asserted, then the PCMC responds with DEVSEL# in slow time.
8	R/WC	**DATA PARITY DETECTED (DPD):** This bit is set to 1 when all of the following conditions are met: 1). The PCMC asserted PERR# or sampled PERR# asserted. 2). The PCMC was the bus master for the operation in which the error occurred. 3). The PERRE bit in the Command Register is set to 1. Software resets DPD to 0 by writing a 1 to it.
7:0		**RESERVED**

3.2.6 RID—REVISION IDENTIFICATION REGISTER

Address Offset: 08h
Default Value: 03h for A-3 Stepping (82434LX)
01h for A-1 Stepping (82434LX)
10h for A-0 Stepping (82434NX)
11h for A-1 Stepping (82434NX)
Attribute: Read Only
Size: 8 bits

This register contains the revision number of the PCMC. These bits are read only and writes to this register have no effect. For the A-2 Stepping of the 82434LX, this value is 03h.

For the A-1 Stepping of the 82434NX, this value is 11h.

Bits	Description
7:0	**REVISION IDENTIFICATION NUMBER:** This is an 8-bit value that indicates the revision identification number for the PCMC.

3.2.7 RLPI—REGISTER-LEVEL PROGRAMMING INTERFACE REGISTER

Address Offset: 09h
Default Value: 00h
Attribute: Read Only
Size: 8 bits

This register defines the PCMC as having no defined register-level programming interface.

Bits	Description
7:0	**REGISTER-LEVEL PROGRAMMING INTERFACE (RLPI):** The value of 00h defines the PCMC as having no defined register-level programming interface.

3.2.8 SUBC—SUB-CLASS CODE REGISTER

Address Offset: 0Ah
Default Value: 00h
Attribute: Read Only
Size: 8 bits

This register defines the PCMC as a host bridge.

Bits	Description
7:0	**SUB-CLASS CODE (SCCD):** The value of this register is 00h defining the PCMC as host bridge.

3.2.9 BASEC—BASE CLASS CODE REGISTER

Address Offset: 0Bh
Default Value: 06h
Attribute: Read Only
Size: 8 bits

This register defines the PCMC as a bridge device.

Bits	Description
7:0	**BASE CLASS CODE (BCCD):** The value in this register is 06h defining the PCMC as bridge device.

3.2.10 MLT—MASTER LATENCY TIMER REGISTER

Address Offset: 0Dh
Default Value: 20h
Attribute: Read/Write
Size: 8 bits

MLT is an 8-bit register that controls the amount of time the PCMC, as a bus master, can burst data on the PCI Bus. MLT is used when the PCMC becomes the PCI Bus master and is cleared and suspended when the PCMC is not asserting FRAME#. When the PCMC asserts FRAME#, the counter is enabled and begins counting. If the PCMC finishes its transaction before the count expires, the MLT count is ignored. If the count expires before the transaction completes, the PCMC initiates a transaction termination as soon as its GNT# is removed. The number of clocks programmed in the MLT represents the guaranteed time slice (measured in PCI clocks) allotted to the PCMC, after which it must surrender the bus as soon as its GNT# is taken away. The number of clocks in the Master Latency Timer is the count value field multiplied by 16.

Bits	Description
7:4	**MASTER LATENCY TIMER COUNT VALUE:** If GNT# is negated after the burst cycle is initiated, the PCMC limits the duration of the burst cycle to the number of PCI Bus clocks specified by this field multiplied by 16.
3:0	**RESERVED**

3.2.11 BIST—BIST REGISTER

Address Offset: 0Fh
Default Value: 0h
Attribute: Read Only
Size: 8 bits

The BIST function is not supported by the PCMC. Writes to this register have no affect.

Bits	Attribute	Description
7	RO	**BIST SUPPORTED:** This read only bit is always set to 0, disabling the BIST function. Writes to this bit position have no affect.
6	RW	**START BIST:** This function is not supported and writes have no affect.
5:4		**RESERVED**
3:0	RO	**COMPLETION CODE:** This read only field always returns 0 when read and writes have no affect.

3.2.12 HCS—HOST CPU SELECTION REGISTER

Address Offset: 50h
Default Value: 82h (82434LX)
A2h (83434NX)
Access: Read/Write, Read Only
Size: 8 bits

The HCS Register is used to specify the Host CPU type and speed. This 8-bit register is also used to enable and disable the first level cache.

Bits	Access	Description
7:5	RO	**HOST CPU TYPE (HCT):** This field defines the Host CPU type. **82434LX** These bits are hardwired to 100 which selects the Pentium processor. All other combinations are reserved. **82434NX** In the 82434NX, these bits are reserved. Reads and writes to these bits have no effect.
4:3		**RESERVED**
2	R/W	**FIRST LEVEL CACHE ENABLE (FLCE):** FLCE enables and disables the first level cache. When FLCE = 1, the PCMC responds to CPU cycles with KEN# asserted for cacheable memory cycles. When FLCE = 0, KEN# is always negated. This prevents new cache line fills to either the first level or second level caches.
1:0	R/W	**HOST OPERATING FREQUENCY (HOF):** The DRAM refresh rate is adjusted according to the frequency selected by this field. For the 82434LX, only bit 0 is used and bit 1 is reserved. **82434LX** Bit 1 is reserved. If bit 0 is 1, the 82434LX supports a 66 MHz CPU. If bit 0 is 0, the 82434LX supports a 60 MHz CPU. **82434NX** These bits select the Host CPU frequency supported as follows: **Bits[1:0] Host CPU Frequency** 00 Reserved 01 50 MHz 10 60 MHz 11 66 MHz

3.2.13 DFC—DETURBO FREQUENCY CONTROL REGISTER

Address Offset: 51h
Default Value: 80h
Attribute: Read/Write
Size: 8 bits

Some software packages rely on the operating speed of the processor to time certain system events. To maintain backward compatibility with these software packages, the PCMC provides a mechanism to emulate a slower operating speed. This emulation is achieved with the PCMC's deturbo mode. The deturbo mode is enabled and disabled via the DM bit in the Turbo-Reset Control Register. When the deturbo mode is enabled, the PCMC periodically asserts AHOLD to slow down the effective speed of the CPU. The duty cycle of the AHOLD active period is controlled by the DFC Register.

Bits	Description
7:6	**DETURBO MODE FREQUENCY ADJUSTMENT VALUE:** This 8-bit value effectively defines the duty cycle of the AHOLD signal. DFC[7:6] are programmable and DFC[5:0] are 0. The value programmed into this register is compared against a free running 8-bit counter running at 1/8 the CPU clock. When the counter is greater than the value specified in this register, AHOLD is asserted. AHOLD is negated when the counter value is equal to or smaller than the contents of this register. AHOLD is negated when the counter rolls over to 00h. The deturbo emulation speed is directly proportional to the value in this register. Smaller values in this register yield slower deturbo emulation speed. The value of 00h is reserved.
5:0	**RESERVED**

3.2.14 SCC—SECONDARY CACHE CONTROL REGISTER

Address Offset: 52h
Default Value: SSS01R10 (82434LX)
SSS01010 (82434NX)
(S = Strapping option)
Attribute: Read/Write
Size: 8 bits

This 8-bit register defines the secondary cache operations. The SCC Register enables and disables the second level cache, adjusts cache size, selects the cache write policy, and defines the cache SRAM type. After hard reset, SCC[7:5] contain the opposite of the signal levels sampled on the Host address lines A[31:29].

Bits	Description
7:6	**SECONDARY CACHE SIZE (SCS):** This field defines the size of the second level cache. The values sampled on the A[31:30] lines at the rising edge of the PWROK signal are inverted and stored in this field. **Bits[7:6] — Secondary Cache Size** 00 — Cache not populated 01 — Reserved 10 — 256-KBytes 11 — 512-KBytes

Bits	Description
5	**SRAM TYPE (SRAMT):** This bit selects between standard SRAMs or burst SRAMS to implement the second level cache. When SRAMT = 0, standard SRAMs are selected. When SRAMT = 1, burst SRAMs are selected. This bit reflects the signal level on the A29 pin at the rising edge of the PWROK signal. This value can be overwritten with subsequent writes to the SCC Register.
4	**82434LX: SECONDARY CACHE ALLOCATION (SCA):** SCA controls when the PCMC performs line fills in the second level cache. When SCA is set to 0, only CPU reads of cacheable main memory with CACHE# asserted are cached in the second level cache. When SCA is set to 1, all CPU reads of cacheable main memory are cached in the second level cache.
3	**CACHE BYTE CONTROL (CBC):** When programmed for asynchronous SRAMs, this bit defines whether the cache uses individual write enables per byte or has a single write enable and byte select lines per byte. When CBC is set to 1, write enable control is used. When CBC is set to 0, byte select control is used.
2	**82434LX: RESERVED** **82434NX: SRAM CONNECTIVITY (SRAMC):** This bit enables different connectivities for the second level cache. When SRAMC is set to 0, the second level cache is in 82434LX compatible mode and all connections between the PCMC and second level cache SRAMs are the same as the 82434LX. When asynchronous SRAMs are used, setting this bit to 1 enables the CCS[1:0]# functionality. CCS[1:0]# are used with asynchronous SRAMs to de-select the SRAMs, placing them in a low power standby mode. When the CPU runs a halt or stop grant special cycle, the 82434NX negates CCS[1:0]#, placing the second level cache in a power saving mode. The PCMC then asserts CCS[1:0]# (activating the SRAMs) when the CPU asserts ADS#. When using burst SRAMs, setting this bit to 1 enables the CCS1# functionality and indicates to the PCMC that no external address latch is present.
1	**82434LX: SECONDARY CACHE WRITE POLICY (SCWP):** SCWP selects between write-back and write-through cache policies for the second level cache. When SCWP = 0 and the second level cache is enabled (bit 0 = 1), the second level cache is configured for write-through mode. When SCWP = 1 and the second level cache is enabled (bit 0 = 1), the second level cache is configured for write-back mode. **82434NX: RESERVED:** Secondary cache write-through mode is not supported. The secondary cache is always in write-back mode and this bit has no affect. SCWP can be set to 0, however, the 82434NX will still operate the secondary cache in write-back mode.
0	**SECONDARY CACHE ENABLE (SCE):** SCE enables and disables the secondary cache. When SCE = 1, the secondary cache is enabled. When SCE = 0, the secondary cache is disabled. When the secondary cache is disabled, the PCMC forwards all main memory cycles to the DRAM interface. Note that setting this bit to 0 does not affect existing valid cache lines. If a cache line contains modified data, the data is not written back to memory. Valid lines in the cache remain valid. When the secondary cache is disabled, the CWE[7:0]# lines remain negated. COE[1:0]# may still toggle. When system software disables secondary caching through this register during run-time, the software should first flush the second level cache. This process is accomplished by first disabling first level caching via the PCE bit in the HCS Register. This prevents the KEN# signal from being asserted, which disables any further line fills. At this point, software executes the WBINVD instruction to flush the caches. When the instruction completes, bit 0 of this register can be reset to 0, disabling the secondary cache. The first level cache can then be enabled by writing the PCE bit in the HCS Register.

3.2.15 HBC—HOST READ/WRITE BUFFER CONTROL

Address Offset: 53h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The HBC Register enables and disables Host-to-main memory and Host-to-PCI posting of write cycles. When posting is enabled, the write buffers in the LBX devices post the data that is destined for either main memory or PCI. This register also permits a CPU-to-main memory read cycle to be performed before any pending posted write data is written to memory.

Bits	Description
7:4	**RESERVED**
3	**READ-AROUND-WRITE ENABLE (RAWCM):** If enabled, the PCMC, during a CPU read cycle to memory where posted write cycles are pending, internally snoops the write buffers. If the address of the read differs from the posted write addresses, the PCMC initiates the memory read cycle ahead of the pending posted memory write. When RAWCM = 0, the pending posted write is written to memory before the memory read is performed. When RAWCM = 1, the PCMC initiates the memory read ahead of the pending posted memory writes.
2	**RESERVED**
1	**HOST-TO-PCI POSTING ENABLE (HPPE):** This bit enables/disables the posting of Host-to-PCI write data in the LBX posting buffers. When HPPE = 1, up to 4 Dwords of data can be posted to PCI. HPPE = 0 is reserved. Buffering is disabled and each CPU write does not complete until the PCI transaction completes (TRDY# is asserted).
0	**82434LX: HOST-TO-MEMORY POSTING ENABLE (HMPE):** This bit enables/disables the posting of Host-to-main memory write data in the LBX buffers. When HMPE = 1, the CPU can post a single write or a burst write (4 Qwords). The CPU burst write completes at 4-1-1-1 when the second level cache is in write-back mode and at 3-1-1-1 when the second level cache is either disabled or in write-through mode. When HMPE = 0, Host-to-main memory posting is disabled and the CPU write cycles do not complete until the data is written to memory. **82434NX: RESERVED:** For the 82434NX, posting is always enabled and this bit has no affect. The CPU can post a single write or burst write (4 Qwords). HMPE can be set to 0, however, the 82434NX will still allow posting of CPU-to-main memory writes.

3.2.16 PBC—PCI READ/WRITE BUFFER CONTROL REGISTER

Address Offset: 54h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The PBC Register enables and disables PCI-to-main memory write posting and permits single CPU-to-PCI writes to be assembled into PCI burst cycles.

Bits	Description
7:3	**RESERVED**
2	**LBXs CONNECTED TO TRDY#:** The TRDY# pin on the LBXs can be connected either to the PCI TRDY# signal or to ground. The cycle time for CPU-to-PCI writes is improved if TRDY# is connected to the LBXs. Since there are two LBXs used in a system, connecting this signal to the LBXs increases the electrical loading of TRDY# by two loads. When the LBXs are externally hard-wired to TRDY#, this bit should be set to 1. Note that this should be done prior to the first Host-to-PCI write or data corruption will occur. Setting this bit to 1 enables the capability of CPU-to-PCI writes at 2-1-1-1 . . . (PCI clocks). When this bit is 0, the LBXs are not connected to TRDY# and CPU-to-PCI writes are completed at 2-2-2-2 . . . timing.
1	**PCI BURST WRITE ENABLE (PBWE):** This bit enables and disables PCI Burst memory write cycles for back-to-back sequential CPU memory write cycles to PCI. When PBWE is set to 1, PCI burst writes are enabled. When PBWE is reset to 0, PCI burst writes are disabled and each single CPU write to PCI invokes a single PCI write cycle (each cycle has an associated FRAME# sequence).
0	**PCI-TO-MEMORY POSTING ENABLE (PMPE):** This bit enables and disables posting of PCI-to-memory write cycles. The posting occurs in a pair of four Dword-deep buffers in the LBXs. When PMPE is set to 1, these buffers are used to post PCI-to-main memory write data. When PMPE is reset to 0, PCI write transactions to main memory are limited to single transfers. The PCMC asserts STOP# with the first TRDY# to disconnect the PCI Master.

3.2.17 DRAMC—DRAM CONTROL REGISTER

Address Offset: 57h
Default Value: 31h
Attribute: Read/Write
Size: 8 bits

This 8-bit register controls main memory DRAM operating modes and features.

Bits	Description
7:6	**82434LX: RESERVED** **82434NX: DRAM BURST TIMING (DBT):** The DRAM interface can be configured for 3 different burst timings. The CAS# pulse width for X-3-3-3 timing is one clock shorter than the CAS# pulse width for X-4-4-4 timing. **Bits[7:6] Burst Timing** 0 0 X-4-4-4 Read/Write timing (default) 0 1 X-4-4-4 Read, X-3-3-3 Write timing 1 0 Reserved 1 1 X-3-3-3 Read/Write timing
5	**PARITY ERROR MASK (PERRM):** When PERRM = 1, parity errors generated during DRAM read cycles initiated by either the CPU request or a PCI Master are masked. This bit affects bits 0 and 1 of the Error Command Register and the ability of the PCMC to respond to PCHK# and assert SERR# when a DRAM parity error occurs. When PERRM is reset to 0, parity errors are not masked.
4	**0-ACTIVE RAS# MODE:** This bit determines if the DRAM page for a particular row remains open (i.e. RAS# remains asserted after a DRAM cycle) enabling the possibility that the next DRAM access may be either a page hit, a page miss, or a row miss. The DRAM interface is then in 1-active RAS# mode. If this bit is reset to 0, RAS# remains asserted after a DRAM cycle. If this bit is set to 1, RAS# is negated after every DRAM cycle, resulting in a row miss for every DRAM cycle. The DRAM interface is then in 0-active RAS# mode.
3	**SMRAM ENABLE (SMRE):** When SMRE = 1, CPU accesses to SMM space are qualified with the SMIACT# pin of the CPU. The location of this space is determined by the SBS field of the SMRAM Register. Read and write cycles to SMM space function normally if SMIACT# is asserted. If SMIACT# is negated when accessing this space, the cycle is forwarded to PCI. When SMRE = 0, accesses to SMM space are treated normally and SMIACT# has no effect. SMRE must be set to 1 to enable the use of the SMRAM Register at configuration space offset 72h.
2	**BURST OF FOUR REFRESH (BFR):** When BFR is set to 1, refreshes are performed in sets of four, at a frequency 1/4 of the normal refresh rate. The PCMC defers refreshes to idle times, if possible. When BFR is reset to 0, single refreshes occur at 15.6 μs refresh rate.
1	**82434LX: REFRESH TYPE (RT):** When RT = 1, the PCMC uses CAS#-before-RAS# timing to refresh the DRAM array. For this refresh type, the PCMC does not supply refresh addresses. When RT = 0, RAS# Only refresh is used and the PCMC drives refresh addresses on the MA[10:0] lines. RAS# only refresh can be used with any type of second level cache configuration (i.e., no second level cache is present, or either a burst SRAM or standard SRAM second level cache is implemented). CAS#-before-RAS# refresh should not be used when a standard SRAM second level cache is implemented. **82434NX: REFRESH TYPE (RT):** In addition to above, when RT = 0, RAS# only refresh is used and the PCMC drives refresh addresses on the MA[11:0] lines. Also, CAS#-before-RAS# refresh can be used with a standrad SRAM second level cache.
0	**REFRESH ENABLE (RE):** When RE is set to 1, the main memory array is refreshed as configured via bits 1 and 2 of this register. When RE is reset to 0, DRAM refresh is disabled. Note that disabling refresh results in the loss of DRAM data.

3.2.18 DRAMT—DRAM TIMING REGISTER

Address Offset: 58h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

For the 82434LX, this register controls the leadoff latency for CPU DRAM accesses.

For the 82434NX, this register provides additional control over DRAM timings. One additional wait-state can be independently added before the assertion of RAS#, the assertion of the first CAS#, or both. This is to allow more flexibility in the layout of the motherboard and in the selection of DRAM speed grades.

Bits	Description
7:2	**RESERVED**
1	**82434LX: RESERVED** **82434NX: RAS# WAIT-STATE (RWS):** When RWS = 1, one additional wait state will be inserted before RAS# is asserted for row misses or page misses in 1-Active RAS mode and all cycles in 0-Active RAS mode. This provides additional MA[11:0] setup time to RAS# assertion.
0	**CAS# WAIT-STATE (CWS):** When CWS = 1, one additional wait state will be inserted before the first assertion of CAS# within a burst cycle. There is no additional delay between CAS# assertions. This provides additional MA[11:0] setup time to CAS# assertion. The CWS bit is typically reset to 0 for 60 MHz operation and set to 1 for 66 MHz operation.

3.2.19 PAM—PROGRAMMABLE ATTRIBUTE MAP REGISTERS (PAM[6:0])

Address Offset: 59–5Fh
Default Value: PAM0 = 0Fh, PAM[1:6] = 00h
Attribute: Read/Write

The PCMC allows programmable memory and cacheability attributes on 14 memory segments of various sizes in the 512 KByte–1 MByte address range. Seven Programmable Attribute Map (PAM) Registers are used to support these features. Three bits are used to specify cacheability and memory attributes for each memory segment. These attributes are:

RE: **Read Enable.** When RE = 1, the CPU read accesses to the corresponding memory segment are directed to main memory. Conversely, when RE = 0, the CPU read accesses are directed to PCI.

WE: **Write Enable.** When WE = 1, the CPU write accesses to the corresponding memory segment are directed to main memory. Conversely, when WE = 0, the CPU write accesses are directed to PCI.

CE: **Cache Enable.** When CE = 1, the corresponding memory segment is cacheable. CE must not be set to 1 when RE is reset to 0 for any particular memory segment. When CE = 1 and WE = 0, the corresponding memory segment is cached in the first and second level caches only on CPU coded read cycles.

The RE and WE attributes permit a memory segment to be Read Only, Write Only, Read/Write, or disabled. For example, if a memory segment has RE = 1 and WE = 0, the segment is Read Only. The characteristics for memory segments with these read/write attributes are described in Table 2.

Table 2. Attribute Definition

Read/Write Attribute	Definition
Read Only	Read cycles: CPU cycles are serviced by the DRAM in a normal manner. Write cycles: CPU initiated write cycles are ignored by the DRAM interface as well as the cache. Instead, the cycles are passed to PCI for termination. Areas marked as Read Only are cacheable for Code accesses only. These regions may be cached in the second level cache, however as noted above, writes are forwarded to PCI, effectively write protecting the data.
Write Only	Read cycles: All read cycles are ignored by the DRAM interface as well as the second level cache. CPU-initiated read cycles are passed onto PCI for termination. The write only state can be used while copying the contents of a ROM, accessible on PCI, to main memory for shadowing, as in the case of BIOS shadowing. Write cycles: CPU write cycles are serviced by the DRAM and cache in a normal manner.
Read/Write	This is the normal operating mode of main memory. Both read and write cycles from the CPU and PCI are serviced by the DRAM and cache interface.
Disabled	All read and write cycles to this area are ignored by the DRAM and cache interface. These cycles are forwarded to PCI for termination.

Each PAM Register controls two regions, typically 16-KByte in size. Each of these regions have a 4-bit field. The four bits that control each region have the same encoding and are defined in Table 3.

Table 3. Attribute Bit Assignment

Bits[7,3] Reserved	Bits[6,2] Cache Enable	Bits[5,1] Write Enable	Bits[4,0] Read Enable	Description
x	x	0	0	DRAM Disabled, Accesses Directed to PCI
x	0	0	1	Read Only, DRAM Write Protected, Non-Cacheable
x	1	0	1	Read Only, DRAM Write Protected, Cacheable for Code Accesses Only
x	0	1	0	Write Only
x	0	1	1	Read/Write, Non-Cacheable
x	1	1	1	Read/Write, Cacheable

NOTE:
To enable PCI master access to the DRAM address space from C0000h to FFFFFh the MEMCS# configuration registers of the ISA or EISA bridge must be properly configured. These registers must correspond to the PAM Registers in the PCMC.

As an example, consider a BIOS that is implemented on the expansion bus. During the initialization process the BIOS can be shadowed in main memory to increase the system performance. When a BIOS is shadowed in main memory, it should be copied to the same address location. To shadow the BIOS, the attributes for that address range should be set to write only. The BIOS is shadowed by first doing a read of that address. This read is forwarded to the expansion bus. The CPU then does a write of the same address, which is directed to main memory. After the BIOS is shadowed, the attributes for that memory area are set to read only so that all writes are forwarded to the expansion bus.

Table 4. PAM Registers and Associated Memory Segments

PAM Reg	Attribute Bits		Memory Segment		Comments		Offset
PAM0[3:0]	R	CE	WE	RE	080000h–09FFFFh	512K–640K	59h
PAM0[7:4]	R	CE	WE	RE	0F0000h–0FFFFFh	BIOS Area	59h
PAM1[3:0]	R	CE	WE	RE	0C0000h–0C3FFFh	ISA Add-on BIOS	5Ah
PAM1[7:4]	R	CE	WE	RE	0C4000h–0C7FFFh	ISA Add-on BIOS	5Ah
PAM2[3:0]	R	CE	WE	RE	0C8000h–0CBFFFh	ISA Add-on BIOS	5Bh
PAM2[7:4]	R	CE	WE	RE	0CC000h–0CFFFFh	ISA Add-on BIOS	5Bh
PAM3[3:0]	R	CE	WE	RE	0D0000h–0D3FFFh	ISA Add-on BIOS	5Ch
PAM3[7:4]	R	CE	WE	RE	0D4000h–0D7FFFh	ISA Add-on BIOS	5Ch
PAM4[3:0]	R	CE	WE	RE	0D8000h–0DBFFFh	ISA Add-on BIOS	5Dh
PAM4[7:4]	R	CE	WE	RE	0DC000h–0DFFFFh	ISA Add-on BIOS	5Dh
PAM5[3:0]	R	CE	WE	RE	0E0000h–0E3FFFh	BIOS Extension	5Eh
PAM5[7:4]	R	CE	WE	RE	0E4000h–0E7FFFh	BIOS Extension	5Eh
PAM6[3:0]	R	CE	WE	RE	0E8000h–0EBFFFh	BIOS Extension	5Fh
PAM6[7:4]	R	CE	WE	RE	0EC000h–0EFFFFh	BIOS Extension	5Fh

DOS Application Area (00000h-9FFFh)

The 640-KByte DOS application area is split into two regions. The first region is 0–512-KByte and the second region is 512–640 KByte. Read, write, and cacheability attributes are always enabled and are not programmable for the 0–512 KByte region.

Video Buffer Area (A0000h-BFFFFh)

This 128-KByte area is not controlled by attribute bits. CPU-initiated cycles in this region are always forwarded to PCI for termination. This area is not cacheable.

Expansion Area (C0000h-DFFFFh)

This 128-KByte area is divided into eight 16-KByte segments. Each segment can be assigned one of four Read/Write states: read-only, write-only, read/write, or disabled Memory that is disabled is not remapped. Cacheability status can also be specified for each segment.

Extended System BIOS Area (E0000h-EFFFFh)

This 64-KByte area is divided into four 16-KByte segments. Each segment can be assigned independent cacheability, read, and write attributes. Memory segments that are disabled are not remapped elsewhere.

System BIOS Area (F0000h-FFFFFh)

This area is a single 64-KByte segment. This segment can be assigned cacheability, read, and write attributes. When disabled, this segment is not remapped.

Extended Memory Area (100000h-FFFFFFFFh)

The extended memory area can be split into several parts:

- Flash BIOS area from 4 GByte to 4 GByte–512-KByte (aliased on ISA at 16 MBytes–15.5 MBytes)
- DRAM Memory from 1 MByte to a maximum of 192 MBytes
- PCI Memory space from the top of DRAM to 4 GByte – 512-KByte
- Memory Space Gap between the range of 1 MByte up to 15.5 MBytes
- Frame Buffer Range mapped into PCI Memory Space or the Memory Space Gap.

On power-up or reset the CPU vectors to the Flash BIOS area, mapped in the range of 4 GByte to 4 GByte – 512-KByte. This area is physically mapped on the expansion bus. Since these addresses are in the upper 4 GByte range, the request is directed to PCI.

The DRAM memory space can occupy extended memory from a minimum of 2 MBytes up to 192 MBytes. This memory is cacheable.

The address space on PCI between the Flash BIOS (4 GByte to 4 GByte – 512 KByte) and the top of DRAM (including any remapped memory) may be occupied by PCI memory. This memory space is not cacheable.

3.2.20 DRB—DRAM ROW BOUNDARY REGISTERS

Address Offset:	60–65h (82434LX) 60–67h (82434NX)
Default Value:	02h
Attribute:	Read/Write
Size:	8 bits

Note the address offset for each DRB Register is DRB0=60h, DRB1=61h, DRB2=62h, DRB3=63h, DRB4=64h, DRB5=65h, DRB6=66h, and DRB7=67h.

3.2.20.1 82434LX Description

The PCMC supports 6 rows of DRAM. Each row is 64 bits wide. The DRAM Row Boundary Registers define upper and lower addresses for each DRAM row. Contents of these 8-bit registers represent the boundary addresses in MBytes.

DRB0 = Total amount of memory in row 0 (in MBytes)
DRB1 = Total amount of memory in row 0 + row 1 (in MBytes)
DRB2 = Total amount of memory in row 0 + row 1 + row 2 (in MBytes)
DRB3 = Total amount of memory in row 0 + row 1 + row 2 + row 3 (in MBytes)
DRB4 = Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 (in MBytes)
DRB5 = Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 (in MBytes)

The DRAM array can be configured with 256K x 36, 1M x 36 and 4M x 36 SIMMs. Each register defines an address range that will cause a particular RAS# line to be asserted (e.g. if the first DRAM row is 2 MBytes in size then accesses within the 0 MByte–2 MBytes range will cause RAS0# to be asserted). The DRAM Row

Boundary (DRB) Registers are programmed with an 8-bit upper address limit value. This upper address limit is compared to A[27:20] of the Host address bus, for each row, to determine if DRAM is being targeted. Since this value is 8 bits and the resolution is 1 MByte, the total bits compared span a 256 MByte space. However, only 192 MBytes of main memory is supported.

Bits	Description
7:0	**ROW BOUNDARY ADDRESS IN MBYTES:** This 8-bit value is compared against address lines A[27:20] to determine the upper address limit of a particular row, i.e. DRB − previous DRB = row size.

Row Boundary Address in MBytes

These 8-bit values represent the upper address limits of the six rows (i.e., this row - previous row = row size). Unpopulated rows have a value equal to the previous row (row size = 0). The value programmed into DRB5 reflects the maximum amount of DRAM in the system. Memory remapped at the top of DRAM, as a result of setting the Memory Space Gap Register, is not reflected in the DRB Registers. The top of memory is always determined by the value written into DRB5 added to the memory space gap size (if enabled).

As an example of a general purpose configuration where 3 physical rows are configured for either single-sided or double-sided SIMMs, the memory array would be configured like the one shown in Figure 8. In this configuration, the PCMC drives two RAS# signals directly to the SIMM rows. If single-sided SIMMs are populated, the even RAS# signal is used and the odd RAS# is not connected. If double-sided SIMMs are used, both RAS# signals are used.

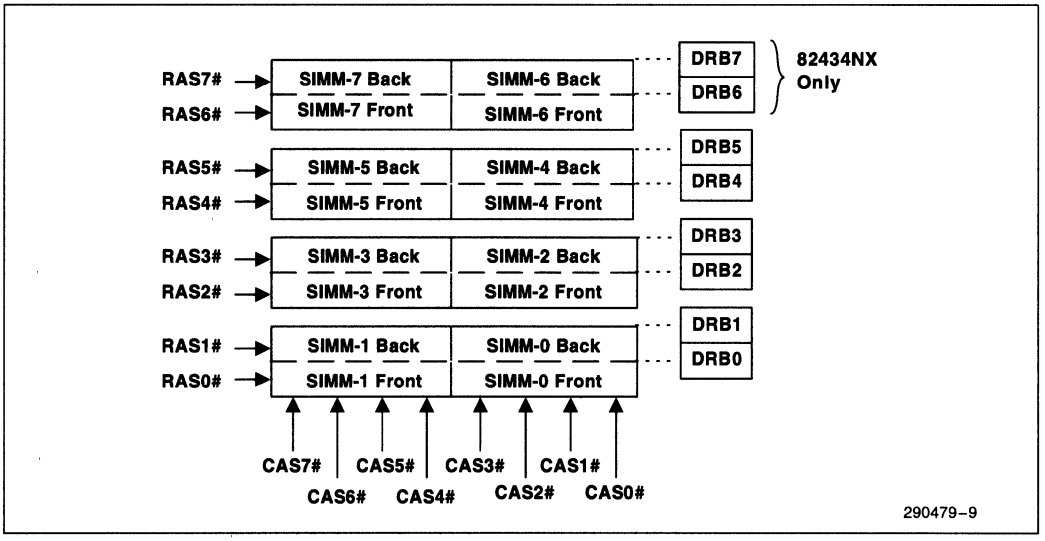


Figure 8. SIMMs and Corresponding DRB Registers

The following 2 examples describe how the DRB Registers are programmed for cases of single-sided and double-sided SIMMs on a motherboard having a total of 6 SIMM sockets.

Example #1

The memory array is populated with six single-sided 256-KByte x 36 SIMMs. Two SIMMs are required for each populated row making each populated row 2 MBytes in size. Filling the array yields 6 MBytes total DRAM. The DRB Registers are programmed as follows:

DRB0 = 02h populated
DRB1 = 02h empty row, not double-sided SIMMs
DRB2 = 04h populated
DRB3 = 04h empty row, not double-sided SIMMs
DRB4 = 06h populated
DRB5 = 06h empty row, not double-sided SIMMs, maximum memory = 6 MBytes.

Example #2

As an another example, if the first four SIMM sockets are populated with 2 MBytes x 36 double-sided SIMMs and the last two SIMM sockets are populated with 4 MBytes x 36 single-sided SIMMs then filling the array yields 64 MBytes total DRAM. The DRB Registers are programmed as follows:

DRB0 = 08h populated with 8 MBytes, ½ of the double-sided SIMMs
DRB1 = 10h the other 8 MBytes of the double-sided SIMMs
DRB2 = 18h populated with 8 MBytes, ½ of the double-sided SIMMs
DRB3 = 20h the other 8 MBytes of the double-sided SIMMs
DRB4 = 40h populated with 32 MBytes
DRB5 = 40h empty row, not double-sided SIMMs, maximum memory = 64 MBytes.

3.2.20.2 82434NX Description

The PCMC supports 8 rows of DRAM. Each row is 64 bits wide. The DRAM Row Boundary Registers define upper and lower addresses for each DRAM row. Contents of these 8-bit registers are concatenated with the associated nibble of the DRBE Register to form 12 bit quantities that represent the row boundary addresses in MBytes.

DRBE[3:0]	DRB0 =	Total amount of memory in row 0 (in MBytes)
DRBE[7:4]	DRB1 =	Total amount of memory in row 0 + row 1 (in MBytes)
DRBE[11:8]	DRB2 =	Total amount of memory in row 0 + row 1 + row 2 (in MBytes)
DRBE[15:12]	DRB3 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 (in MBytes)
DRBE[19:16]	DRB4 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 (in MBytes)
DRBE[23:20]	DRB5 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 (in Bytes)
DRBE[27:24]	DRB6 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 + row 6 (in MBytes)
DRBE[31:28]	DRB7 =	Total amount of memory in row 0 + row 1 + row 2 + row 3 + row 4 + row 5 + row 6 + row 7 (in MBytes)

The DRAM array can be configured with 256K x 36, 1M x 36, 4M x 36, and 16M x 36 SIMMs. Each register defines an address range that will cause a particular RAS# line to be asserted (e.g. if the first DRAM row is 2 MBytes in size then accesses within the 0 to 2 MBytes range will cause RAS0# to be asserted). The DRAM Row Boundary (DRB) Registers are programmed with an 8-bit upper address limit value. The DRBE Register extends the programming model of this mechanism to 12 bits, however only 10 bits are implemented at this time. This upper address limit is compared to A[29:20] of the Host address bus, for each row, to determine if DRAM is being targeted. Since this value is 10 bits and the resolution is 1 MByte, the total bits compared span a 1 GByte space. However, other resource limits in the PCMC cap the total usable DRAM space at 512 MBytes.

Bits	Description
7:0	**ROW BOUNDARY ADDRESS IN MBYTES:** This 8-bit value is concatenated with a nibble from the DRBE Register and then compared against address lines A[29:20] to determine the upper address limit of a particular row (i.e. DRB − previous DRB = row size).

Row Boundary Address in MBytes

These 10-bit values represent the upper address limits of the 8 rows (i.e., this row - previous row = row size). Unpopulated rows have a value equal to the previous row (row size = 0). The value programmed into DRBE[31:28] || DRB7 reflects the maximum amount of DRAM in the system. Memory remapped at the top of DRAM, as a result of setting the Memory Space Gap Register, is not reflected in the DRB Registers. The top of memory is determined by the value written into DRBE[31:28] || DRB7 added to the memory space gap size (if enabled). If DRBE[31:28] || DRB7 plus the memory space gap is greater than 512 MBytes then 512 MBytes of DRAM are available.

The following 2 examples describe how the DRB Registers are programmed for cases of single-sided and double-sided SIMMs on a motherboard having a total of 8 SIMM sockets.

2

Example #1

The memory array is populated with eight single-sided 256-KByte x 36 SIMMs. Two SIMMs are required for each populated row making each populated row 2 MBytes in size. Filling the array yields 8 MBytes total DRAM. The DRB Registers are programmed as follows:

DRBE[3:0] = 0h	DRB0 = 02h	populated
DRBE[7:4] = 0h	DRB1 = 02h	empty row, not double-sided SIMMs
DRBE[11:8] = 0h	DRB2 = 04h	populated
DRBE[15:12] = 0h	DRB3 = 04h	empty row, not double-sided SIMMs
DRBE[19:16] = 0h	DRB4 = 06h	populated
DRBE[23:20] = 0h	DRB5 = 06h	empty row, not double-sided SIMMs
DRBE[27:24] = 0h	DRB6 = 08h	populated
DRBE[31:28] = 0h	DRB7 = 08h	empty row, not double-sided SIMMs, max memory = 8 MBytes.

Example #2

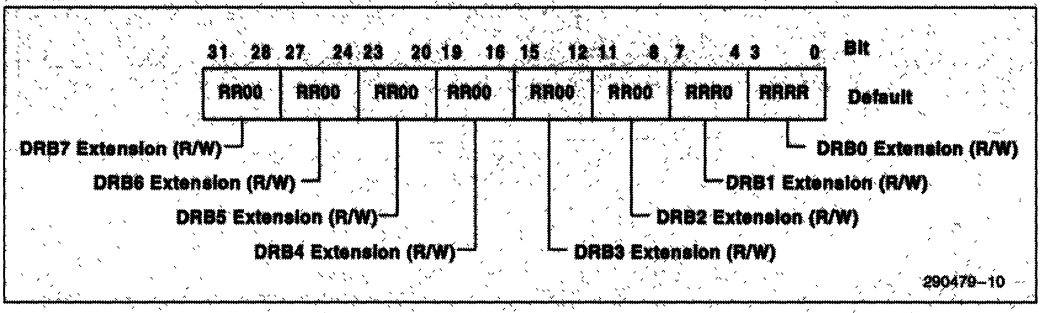
As an another example, if the first four SIMM sockets are populated with 2 MByte x 36 double-sided SIMMs and the last four SIMM sockets are populated with 16 MByte x 36 single-sided SIMMs then filling the array yields 288 MBytes total DRAM. The DRB Registers are programmed as follows:

DRBE[3:0] = 0h	DRB0 = 08h	populated with 8 MBytes, ½ of double-sided SIMMs
DRBE[7:4] = 0h	DRB1 = 10h	the other 8 MBytes of the double-sided SIMMs
DRBE[11:8] = 0h	DRB2 = 18h	populated with 8 MBytes, ½ of double-sided SIMMs
DRBE[15:12] = 0h	DRB3 = 20h	the other 8 MBytes of the double-sided SIMMs
DRBE[19:16] = 0h	DRB4 = A0h	populated with 128 MBytes
DRBE[23:20] = 0h	DRB5 = A0h	empty row, not double-sided SIMMs
DRBE[27:24] = 1h	DRB6 = 20h	populated with 128 MBytes
DRBE[31:28] = 1h	DRB7 = 20h	empty row, not double-sided SIMMs, max memory = 288 MBytes.

3.2.21 DRBE—DRAM ROW BOUNDARY EXTENSION REGISTER

Address Offset: 68-6Bh
Default Value: 0000h
Attribute: Read/Write
Size: 32 bits

The DRBE Register is not implemented in the 82434LX. This register contains an extension for each of the DRAM Row Boundary (DRB) Registers. Each nibble of the DRBE Register is concatenated with a DRB Register (see DRB Register section for details on the use of the DRB and DRBE Registers).



Bits	Description
31:0	**EXTENSIONS FOR DRB0 THROUGH DRB7:** Each nibble corresponds to a DRB. The nibble of the DRBE and its corresponding DRB are concatenated and used to indicate the boundaries between rows of DRAM.

3.2.22 ERRCMD—ERROR COMMAND REGISTER

Address Offset: 70h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The Error Command Register controls the PCMC responses to various system errors. Bit 6 of the PCICMD Register is the master enable for bit 3 of this register. Bit 6 of the PCICMD Register must be set to 1 to enable the error reporting function defined by bit 3 of this register. Bits 6 and 8 of the PCICMD Register are the master enables for bits 7, 6, 5, 4, and 1 of this register. Both bits 6 and 8 of the PCICMD Register must be set to 1 to enable the error reporting functions defined by bits 7, 6, 5, 4, and 1 of this register.

Bits	Description
7	**SERR# ON RECEIVED TARGET ABORT:** When this bit is set to 1 (and bit 8 of the PCICMD Register is 1), the PCMC asserts SERR# upon receiving a target abort. When this bit is set to 0, the PCMC is disabled from asserting SERR# upon receiving a target abort.
6	**SERR# ON TRANSMITTED PCI DATA PARITY ERROR:** When this bit is set to 1 (and bits 6 and 8 of the PCICMD Register are both 1), the PCMC asserts SERR# when it detects a data parity error as a result of a CPU-to-PCI write (PERR# detected asserted). When this bit is set to 0, the PCMC is disabled from asserting SERR# when data parity errors are detected via PERR#.
5	**82434LX: RESERVED** **82434NX: SERR# ON RECEIVED PCI DATA PARITY ERROR:** When this bit is set to 1 (and bits 6 and 8 of the PCICMD Register are both 1), the PCMC asserts SERR# when it detects a data parity error as a result of a CPU-to-PCI read (PAR incorrect with received data). In this case, the SERR# signal is asserted when parity errors are detected on PCI return data. When this bit is set to 0, the PCMC is disabled from asserting SERR# when data parity errors are detected during a CPU-to-PCI read.
4	**82434LX: RESERVED** **82434NX: SERR# ON PCI ADDRESS PARITY ERROR:** When this bit is set to 1 (and bits 6 and 8 of the PCICMD Register are both 1), the PCMC asserts SERR# when it detects an address parity error on PCI transactions. When this bit is set to 0, the PCMC is disabled from asserting SERR# when address parity errors are detected on PCI transactions.
3	**82434LX: RESERVED** **82434NX: PERR# ON RECEIVING A DATA PARITY ERROR:** This bit indicates whether the PERR# signal is implemented in the system. When this bit is set to 1 (and bit 6 of the PCICMD Register is 1), the PCMC asserts PERR# when it detects a data parity error (PAR incorrect with received data), either from a CPU-to-PCI read or a PCI master write to memory. When this bit is set to 0 (or bit 6 of the PCICMD Register is set to 0), the PERR# signal is not asserted by the PCMC.
2	**L2 CACHE PARITY ENABLE:** This bit indicates that the second level cache implements parity. When this bit is set to 1, bits 0 and 1 of this register control the checking of parity errors during CPU reads from the second level cache. If this bit is 0, parity is not checked when the CPU reads from the second level cache (PCHK# ignored) and neither bit 1 nor bit 0 apply.
1	**SERR# ON DRAM/L2 CACHE DATA PARITY ERROR ENABLE:** This bit enables/disables the SERR# signal for parity errors on reads from main memory or the second level cache. When this bit is set to 1 and bit 0 of this register is set to 1 (and bits 6 and 8 of the PCICMD Register are set to 1), SERR# is enabled upon a PCHK# assertion from the CPU when reading from main memory or the second level cache. The processor indicates that a parity error was received by asserting PCHK#. The PCMC then latches status information in the Error Status Register and asserts SERR#. When this bit is 0, SERR# is not asserted upon detecting a parity error. Bits[1:0] = 10 is a reserved combination. 0 = Disable assertion of SERR# upon detecting a DRAM/second level cache read parity error. 1 = Enable assertion of SERR# upon detecting a DRAM/second level cache read parity error.
0	**MCHK ON DRAM/L2 CACHE DATA PARITY ERROR ENABLE:** When this bit is set to 1, PEN# is asserted for data returned from main memory or the second level cache. The processor indicates that a parity error was received by asserting the PCHK# signal. In addition, the processor invokes a machine check exception, if enabled via the MCE bit in CR4 in the Pentium processor. The PCMC then latches status information in the Error Status register. When this bit is 0, PEN# is not asserted. Bits[1:0] = 10 is a reserved combination.

3.2.23 ERRSTS—ERROR STATUS REGISTER

Address Offset: 71h
Default Value: 00h
Attribute: Read/Write Clear
Size: 8 bits

The Error Status Register is an 8-bit register that reports the occurrence of PCI, second level cache, and DRAM parity errors. This register also reports the occurrence of a CPU shutdown cycle.

Bits	Description
7	**RESERVED**
6	**PCI TRANSMITTED DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a data parity error (PERR# asserted) as a result of a CPU-to-PCI write. Software resets this bit to 0 by writing a 1 to it.
5	**82434LX: RESERVED** **82434NX: PCI RECEIVED DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a data parity error (PAR incorrect with received data) as a result of a CPU-to-PCI read. Software resets this bit to 0 by writing a 1 to it.
4	**82434LX: RESERVED** **82434NX: PCI ADDRESS PARITY ERROR:** The PCMC sets this bit to a 1 when it detects an address parity error (PAR incorrect with received address and C/BE# lines) on a PCI master transaction. Software resets this bit to 0 by writing a 1 to it.
3	**MAIN MEMORY DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a parity error from the CPU PCHK# signal resulting from a CPU-to-main memory read. Software resets this bit to 0 by writing a 1 to it.
2	**L2 CACHE DATA PARITY ERROR:** The PCMC sets this bit to a 1 when it detects a parity error from the CPU PCHK# signal resulting from a CPU read access that hit in the second level cache. Software resets this bit to 0 by writing a 1 to it.
1	**RESERVED**
0	**SHUTDOWN CYCLE DETECTED:** The PCMC sets this bit to a 1 when it detects a shutdown special cycle on the Host Bus. Under this condition the PCMC drives a shutdown special cycle on PCI and asserts INIT. Software resets this bit to 0 by writing a 1 to it.

3.2.24 SMRS—SMRAM SPACE REGISTER

Address Offset: 72h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The PCMC supports a 64-KByte SMRAM space that can be selected to reside at the top of main memory, segment A0000–AFFFFh or segment B0000–BFFFFh. The SMM space defined by this register is not cacheable. This register defines a mechanism that allows the CPU to execute code out of the SMM space at either A0000h or B0000h while accessing the frame buffer on PCI. The SMRAM Enable bit in the DRAM Control Register must be 1 to enable the features defined by this register. Register bits[5:3] apply only when segment A0000-AFFFFh or B0000-BFFFFh are selected.

Bits	Description
7:6	**RESERVED**
5	**OPEN SMRAM SPACE (OSS):** When OSS = 1, the CPU can access SMM space without being in SMM mode. That is, accesses to SMM space are permitted even with SMIACT# negated. This bit is intended to be used during POST to allow the CPU to initialize SMRAM space before the first SMI# interrupt is issued.
4	**CLOSE SMRAM SPACE (CSS):** When CSS = 1 and SMRAM is enabled, CPU code accesses to the SMM memory range are directed to SMM space in main memory and data accesses are forwarded to PCI. This bit allows the CPU to read and write the frame buffer on PCI while executing SMM code. When CSS = 0 and SMRAM is enabled, all accesses to the SMRAM memory range, both code and data, are directed to SMRAM (main memory).
3	**LOCK SMRAM SPACE (LSS):** When LSS = 1, this bit prevents the SMM space from being manually opened, effectively disabling bit 5 of this register. Only a power-on reset can set this bit to 0.
2:0	**SMM BASE SEGMENT (SBS):** This field defines the 64 KByte base segment where SMM space is located. The memory that is defined by this field is non-cacheable.

Bits[2:0]	SMRAM Location	Bits[2:0]	SMRAM Location
000	Top of main memory	100	Reserved
001	Reserved	101	Reserved
010	A0000–AFFFFh	110	Reserved
011	B0000–BFFFFh	111	Reserved

3.2.25 MSG—MEMORY SPACE GAP REGISTER

Address Offset: 78-79h
Default Value: 00h
Attribute: Read/Write
Size: 16 bits

The Memory Space Gap Register defines the starting address and size of a gap in main memory. This register accommodates ISA devices that have their memory mapped into the 1 MByte–15.5 MByte range (e.g., an ISA LAN card or an ISA frame buffer). The Memory Space Gap Register defines a hole in main memory that transfers the cycles in this address space to the PCI Bus instead of main memory. This area is not cacheable.

The memory space gap starting address must be a multiple of the memory space gap size. For example, a 2 MByte gap must start at 2, 4, 6, 8, 10, 12, or 14 MBytes.

NOTE:
Memory that is disabled by the gap created by this register is remapped to the top of memory. This remapped memory is accessible, except in the case where this would cause the top of main memory to exceed 192 MBytes (or 512 MBytes for the 82434NX).

Bits	Description
15	**MEMORY SPACE GAP ENABLE (MSGE):** MSGE enables and disables the memory space gap. When MSGE is set to 1, the CPU accesses to the address range defined by this register are forwarded to PCI bus. The size of the gap created in main memory causes a corresponding amount of DRAM to be remapped at the top of main memory (top specified by DRB Registers). If the Frame Buffer Range is programmed below 16 MBytes and within main memory space, the MSG register must include the Frame Buffer Range. When MSGE is reset to 0, the memory space gap is disabled.
14:12	**MEMORY SPACE GAP SIZE (MSGS):** This 3 bit field defines the size of the memory space gap. If the Frame Buffer Range is programmed below 16 MBytes and within main memory space, this register must include the frame buffer range. The amount of main memory specified by these bits is remapped to the top of main memory. **Bit[14:12] — Memory Gap Size** 000 — 1 MByte 001 — 2 MBytes 011 — 4 MBytes 111 — 8 MBytes **NOTE:** All other combinations are reserved.
11:8	**RESERVED**
7:4	**MEMORY SPACE GAP STARTING ADDRESS (MSGSA):** These 4 bits define the starting address of the memory space gap in the space from 1 MByte–16 MBytes. These bits are compared against A[23:20]. The memory space gap starting address must be a multiple of the memory space gap size. For example, a 2 MBytes gap must start at 2, 4, 6, 8, 10, 12, or 14 MBytes.
3:0	**RESERVED**

3.2.26 FBR—FRAME BUFFER RANGE REGISTER

Address Offset: 7C-7Fh
Default Value: 0000h
Attribute: Read/Write
Size: 32 bits

This 32-bit register enables and disables a frame buffer area and provides attribute settings for the frame buffer area. The attributes defined in this register are intended to increase the performance of the frame buffer. The FBR Register can be used to accommodate PCI devices that have their memory mapped onto PCI from the top of main memory to 4 GByte–512-KByte range (e.g., a linear frame buffer). If the Frame Buffer Range is located within the 1 MByte–16 MBytes main memory region where DRAM is populated, the Memory Space Gap Register must be programmed to include the Frame Buffer Range.

Bits	Description
31:20	**BUFFER OFFSET (BO):** BO defines the starting address of the frame buffer address space in increments of 1 MByte. This 12-bit field is compared directly against A[31:20]. The frame buffer range can either be located at the top of memory, including remapped memory or within the memory space gap (i.e., frame buffer range programmed below 16 MBytes and within main memory space. When bits [31:20] = 0000h and bit 12 = 0, all features defined by this register are disabled.
19:14	**RESERVED**
13	**BYTE MERGING (BM):** Byte merging permits CPU-to-PCI byte writes to the LBX posted write buffer to be combined into a single transfer on the PCI Bus, when appropriate. When BM is set to 1, byte merging on CPU-to-PCI posted write cycles is enabled. When BM is reset to 0, byte merging is disabled.
12	**128K VGA RANGE ATTRIBUTE ENABLE (VRAE):** When VRAE = 1, the attributes defined in this register (bits [13, 10:7]) also apply to the VGA memory range of A0000h–BFFFFh regardless of the value programmed in the Buffer Offset field. When VRAE = 0, the attributes do not apply to the VGA memory range. Note that this bit only affects the mentioned attributes of the VGA memory range and does not enable or disable accesses to the VGA memory range.
11:10	**RESERVED**
9	**NO LOCK REQUESTS (NLR):** When NLR is set to 1, the PCMC never requests exclusive access to a PCI resource via the PCI LOCK# signal in the range defined by this register. When NLR is reset to 0, exclusive access via the PCI LOCK# signal in the range defined by this register is enabled.
8	**RESERVED**
7	**TRANSPARENT BUFFER WRITES (TBW):** When set to a 1, this bit indicates that writes to the Frame Buffer Range need not be flushed for deadlock or coherence reasons on synchronization events (i.e., PCI master reads, and the FLSHBUF#/MEMREQ# protocol). When reset to 0, this bit indicates that upon synchronization events, flushing is required for Frame Buffer writes posted in the CPU-to-PCI Write Buffer in the LBX
6:4	**RESERVED**
3:0	**BUFFER RANGE (BR):** These bits define the size of the frame buffer address space, allowing up to 16 MBytes of frame buffer. If the Frame Buffer Range is within the memory space gap, the buffer range is limited to 8 MBytes and must be included within the memory space gap. The bits listed below in the Reserved Buffer Offset (BO) Bits column are ignored by the PCMC for the corresponding buffer sizes. **Bits[3:0] / Buffer Size / Reserved Buffer Offset (BO) Bits** 0000 / 1 MByte / None 0001 / 2 MBytes / [20] 0011 / 4 MBytes / [21:20] 0111 / 8 MBytes / [22:20] 1111 / 16 MBytes / [23:20] **NOTE:** (all other combinations are reserved)

4.0 PCMC ADDRESS MAP

The Pentium processor has two distinct physical address spaces: Memory and I/O. The memory address space is 4 GBytes and the I/O address space is 64 KBytes. The PCMC maps accesses to these address spaces as described in this section.

4.1 CPU Memory Address Map

Figure 9 shows the address map for the 4 GByte Host CPU memory address space. Depending on the address range and whether a memory gap is enabled via the MSG Register, the PCMC forwards CPU memory accesses to either main memory or PCI memory. Accesses forwarded to main memory invoke operations on the DRAM interface and accesses forwarded to PCI memory invoke operations on PCI. Mapping to the PCI Bus permits PCI or EISA/ISA Bus-based memory.

The main memory size ranges from 2 MBytes–192 MBytes for the 82434LX and 2 MBytes–512 MBytes for the 82434NX. Memory accesses above 192 MBytes (512 MBytes for the 82434NX) are always forwarded to PCI. In addition, a memory gap can be created in the 1 MByte–16 MBytes region that provides a window to PCI-based memory. The location and size of the gap is programmable. Accesses to addresses in the gap are ignored by the DRAM controller and forwarded to PCI. Note that CPU memory accesses that are forwarded to PCI (including the Memory Space Gap) are not cacheable. Only main memory controlled by the PCMC DRAM interface is cacheable.

4.2 System Management RAM—SMRAM

The PCMC supports the use of main memory as System Management RAM (SMRAM) enabling the use of System Management Mode. This function is enabled and disabled via the DRAM Control Register. When this function is disabled, the PCMC memory map is defined by the DRB and PAM Registers. When SMRAM is enabled, the PCMC reserves the top 64-KBytes of main memory for use as SMRAM.

SMRAM can also be placed at A0000–AFFFFh or B0000–BFFFFh via the SMRAM Space Register. Enhanced SMRAM features can also be enabled via this register. PCI masters can not access SMRAM when it is programmed to the A or B segments.

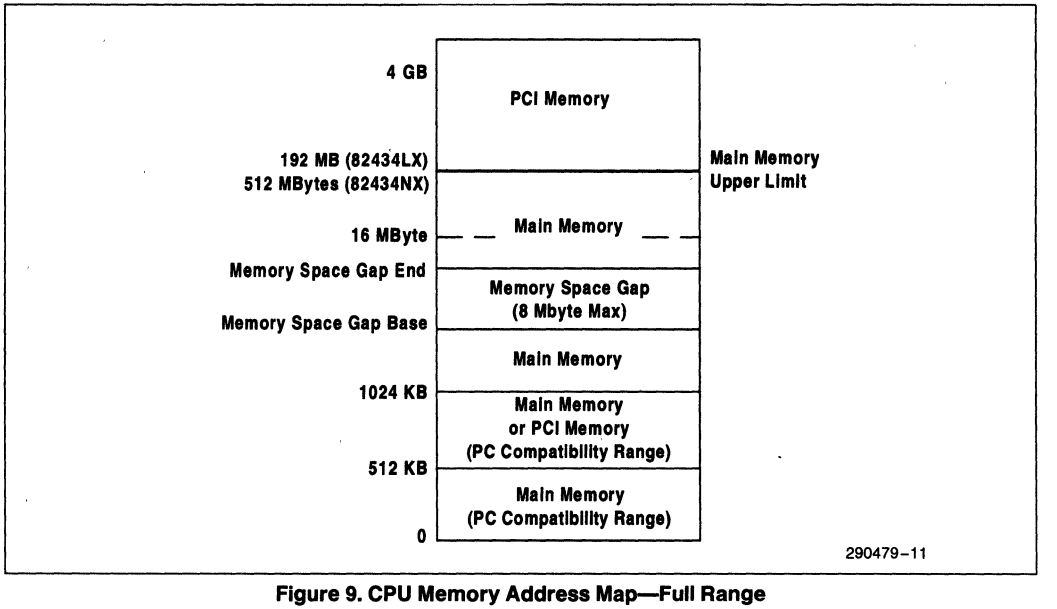


Figure 9. CPU Memory Address Map—Full Range

However, PCI masters can access SMRAM when the top of memory is selected.

When the 82434NX PCMC detects a CPU stop grant special cycle (M/IO# = 0, D/C# = 0, W/R# = 1, A4 = 1, BE[7:0]# = FBh), it generates a PCI Stop Grant Special cycle, with 0002h in the message field (AD[15:0]) and 0012h in the message dependent data field (AD[31:16]) during the first data phase (IRDY# asserted).

4.3 PC Compatibility Range

The PC Compatibility Range is the first MByte of the Memory Map. The 512 KByte–1 MByte range is subdivided into several regions as shown in Figure 10. Each region is provided with programmable attributes in the PAM Registers. The attributes are Read Enable (RE), Write Enable (WE) and Cache Enable (CE). The attributes determine readability, writeability and cacheability of the corresponding memory region. When the associated bit in the PAM Register is set to a 1, the attribute is enabled and when set to a 0 the attribute is disabled. The following rules apply for cacheability in the first level and second level caches:

1. If RE = 1, WE = 1, and CE = 1, the region is cacheable in the first level and second level caches.
2. If RE = 1, WE = 0, and CE = 1, the region is cacheable only on code reads (i.e., D/C# = 0). Data reads do not result in a line fill. Writes to the region are not serviced by the secondary cache, but are forwarded to PCI.

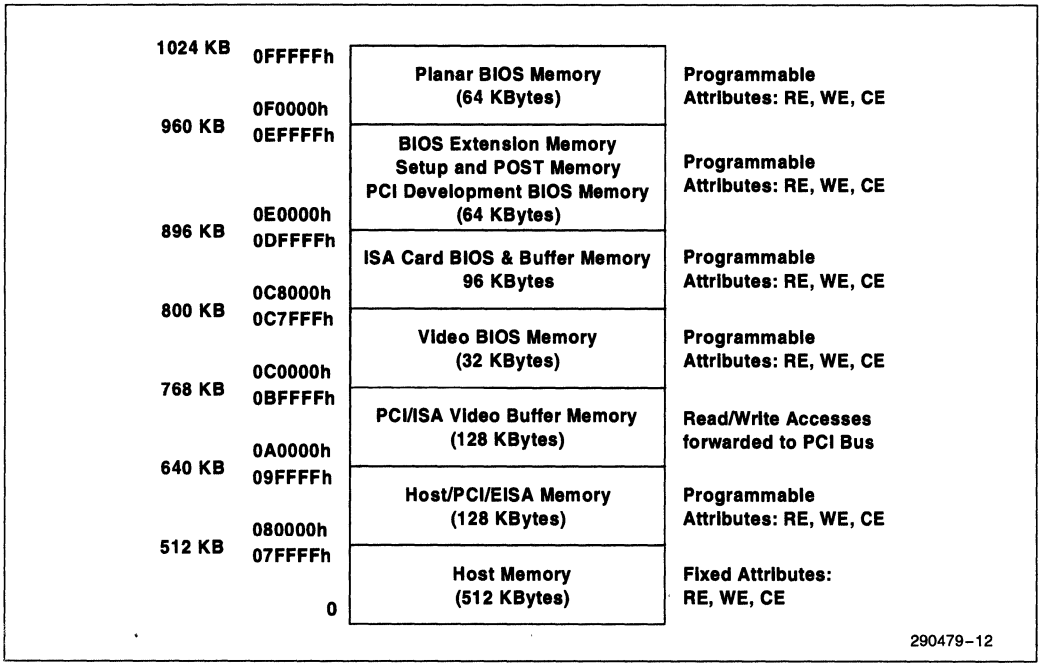
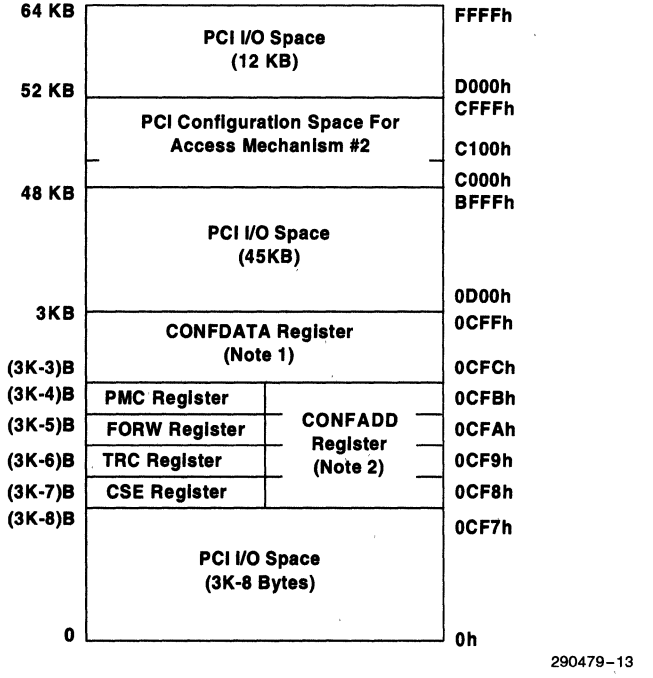


Figure 10. CPU Memory Address Map—PC Compatibility Range

The RE and WE bits for each region are used to shadow BIOS ROM in main memory for improved system performance. To shadow a BIOS area, RE is reset to 0 and WE is set to 1. RE is set to 1 and WE is reset to 0. Any writes to the BIOS area are forwarded to PCI.

4.4 I/O Address Map

I/O devices (other than the PCMC) are not supported on the Host Bus. The PCMC generates PCI Bus cycles for all CPU I/O accesses, except to the PCMC internal registers. Figure 11 shows the mapping for the CPU I/O address space. For the 82434LX, three PCMC registers are located in the CPU I/O address space—the Configuration Space Enable (CSE) Register, the Turbo-Reset Control (TRC) Register, and the Forward (FORW) Register.



NOTES:
1. This 82434NX register is only visible when configuration access mechanism #1 is enabled (via bit 31 of the CONFADD Register). Otherwise, this I/O range is in PCI I/O space.
2. This 82434NX register is accessed during Dword read/writes to 0CF8h. Byte or word cycles access the corresponding 8-bit registers, even if configuration access mechanism #1 is enabled.

Figure 11. CPU I/O Address Map

For the 82434NX, six PCMC registers are located in the CPU I/O address space—the Configuration Space Enable (CSE) Register, the Configuration Address Register (CONFADD), the Turbo-Reset Control (TRC) Register, the Forward (FORW) Register, the PCI Mechanism Control (PMC) Register, and the Configuration Data (CONFDATA) Register.

Except for the I/O locations of the above mentioned registers, all other CPU I/O accesses are mapped to either PCI I/O space or PCI configuration space. If the access is to PCI I/O space, the PCI address is the same as the CPU address. If the access is to PCI configuration space, the CPU address is mapped to a configuration space address as described in Section 3.0, Register Description.

If configuration space is enabled via the CSE Register (access mechanism #2), the PCMC maps accesses in the address range of C100h to CFFFh to PCI configuration space. Accesses to the PCMC configuration register range (C000h to C0FFh) are intercepted by the PCMC and not forwarded to PCI. If the configuration space is disabled in the CSE Register, CPU accesses to the configuration address range (C000h to CFFFh) are forwarded to PCI I/O space.

5.0 SECOND LEVEL CACHE INTERFACE

This section describes the second level cache interface for the 82434LX Cache (Section 5.1) and the 82434NX Cache (Section 5.2). The differences are in the following areas:

1. The 82434LX supports both write-through and write-back cache policies. The 82434NX only supports the write-back policy.
2. The 82434LX timings are for 60 and 66 MHz and the 82434NX timings are for 50, 60, and 66 MHz. Note that the cycle latencies for 60 and 66 MHz are the same for both devices.
3. When burst SRAMs are used to implement the secondary cache, address latches are not needed for the 82434NX type SRAM connectivity. However, a control bit has been added to the 82434NX that permits address latches for 82434LX type SRAM connectivity.
4. A low-power second level cache standby mode has been added to the 82434NX.
5. There are new or changed cache control bits as indicated by the shading in Section 3.0, Register Description. For example, the 82434NX supports zero wait-state cache at 50 MHz via the zero wait-state control bit.

NOTE:

- Second level cache sizes and organization are the same for the 82434LX and 82434NX.
- The general operation of the second level cache write-back policy is the same for the 82434LX and 82434NX. For example, the Valid and Modified bits operate the same for both devices. In addition, snoop operations are the same for both devices, as well as the handling of flush, flush acknowledge, and write-back special cycles.

5.1 82434LX Cache

The 82434LX PCMC integrates a high performance write-back/write-through second level cache controller providing integrated tags and a full first level and second level cache coherency mechanism. The second level cache controller can be configured to support either a 256-KByte cache or a 512 KByte cache using either synchronous burst SRAMs or standard asynchronous SRAMs. The cache is direct mapped and can be configured to support either a write-back or write-through write policy. Parity on the second level cache data SRAMs is optional.

The 82434LX contains 4096 address tags. Each tag represents a *sector* in the second level cache. If the second level cache is 256-KByte, each tag represents two cache lines. If the second level cache is 512-KByte, each tag represents four cache lines. Thus, in the 256-KByte configuration each sector contains two lines. In the 512-KByte configuration, each sector contains four lines. *Valid* and *modified* status bits are kept on a per line basis. Thus, in the case of a 256-KByte cache each tag has two valid bits and two modified bits associated with it. In the case of a 512-KByte cache each tag has four valid and four modified bits associated with it. Upon a CPU read cache miss, the PCMC inspects the valid and modified bits within the addressed sector and writes back to main memory only the lines marked both valid and modified. All of the lines in the sector are then invalidated. The line fill will then occur and the valid bit associated with the allocated line will be set. Only the requested line will be fetched from main memory and written into the cache. If no write-back is required, all of the lines in the sector are marked invalid. The line fill then occurs and the valid bit associated with the allocated line will be set. Lines are not allocated on write misses. When a CPU write hits a line in the second level cache, the modified bit for the line is set.

The second level cache is optional to allow the 82434LX PCMC to be used in a low cost configuration. A 256-KByte cache is implemented with a single bank of eight 32K x 9 SRAMs if parity is supported or 32K x 8 SRAMs if parity is not supported on the cache. A 512-KByte cache is implemented with four 64K x 18 SRAMs if parity is supported or 64K x 16 SRAMs if parity is not supported on the cache. Two 74AS373 latches complete the cache. Only main memory controlled by the PCMC DRAM interface is cached. Memory on PCI is not cached.

Figure 12 and Figure 13 depict the organization of the internal tags in the PCMC configured for a 256 KByte cache and a 512-KByte cache.

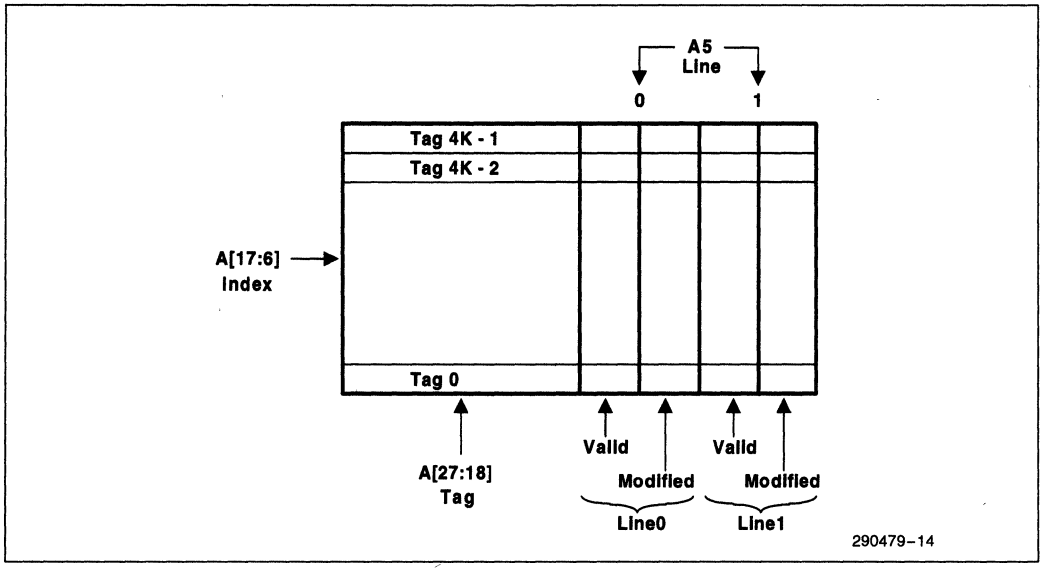


Figure 12. PCMC Internal Tags with 256-KByte Cache

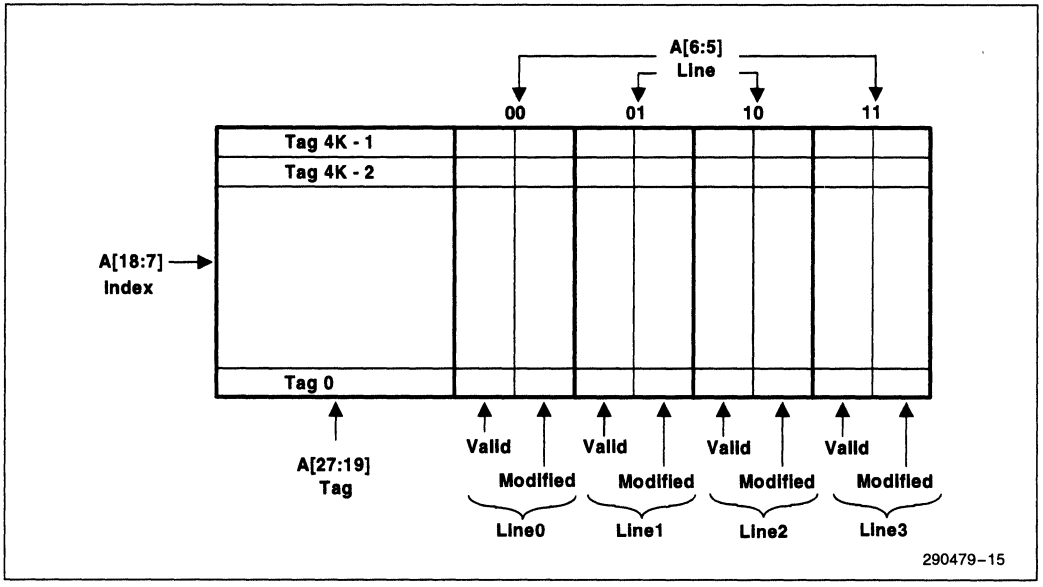


Figure 13. PCMC Internal Tags with 512-KByte Cache

In the 256-KByte cache configuration A[17:6] form the tag RAM index. The ten tag bits read from the tag RAM are compared against A[27:18] from the host address bus. Two valid bits and two modified bits are kept per tag in this configuration. Host address bit 5 is used to select between lines 0 and 1 within a sector. In the 512-KByte cache configuration A[18:7] form the tag RAM index. The nine bits read from the tag RAM are compared against A[27:19] from the host bus. Four valid bits and four modified bits are kept per tag. Host address bits 5 and 6 are used to select between lines 0, 1, 2 and 3 within a sector.

The Secondary Cache Controller Register at offset 52h in configuration space controls the secondary cache size, write and allocation policies, and SRAM type. The cache can also be enabled and disabled via this register.

Figure 14 through Figure 18 show the connections between the PCMC and the external cache data SRAMs and latches.

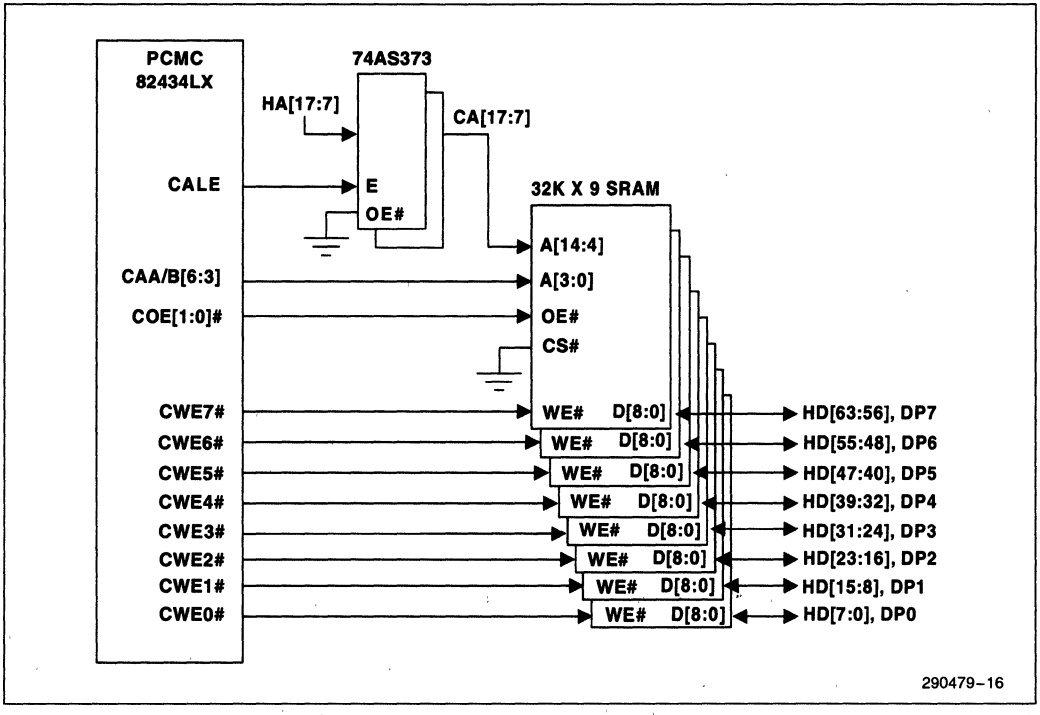


Figure 14. 82434LX Connections to 256-KByte Cache with Standard SRAM

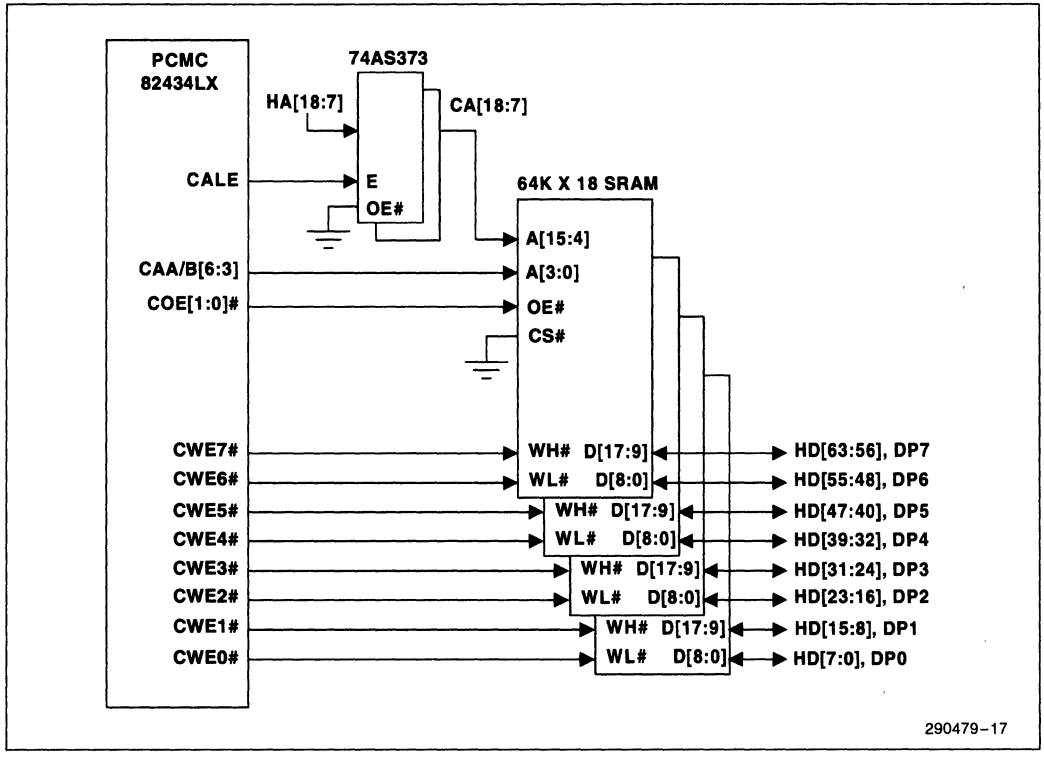


Figure 15. 82434LX Connections to 512-KByte Cache with Standard SRAM

2

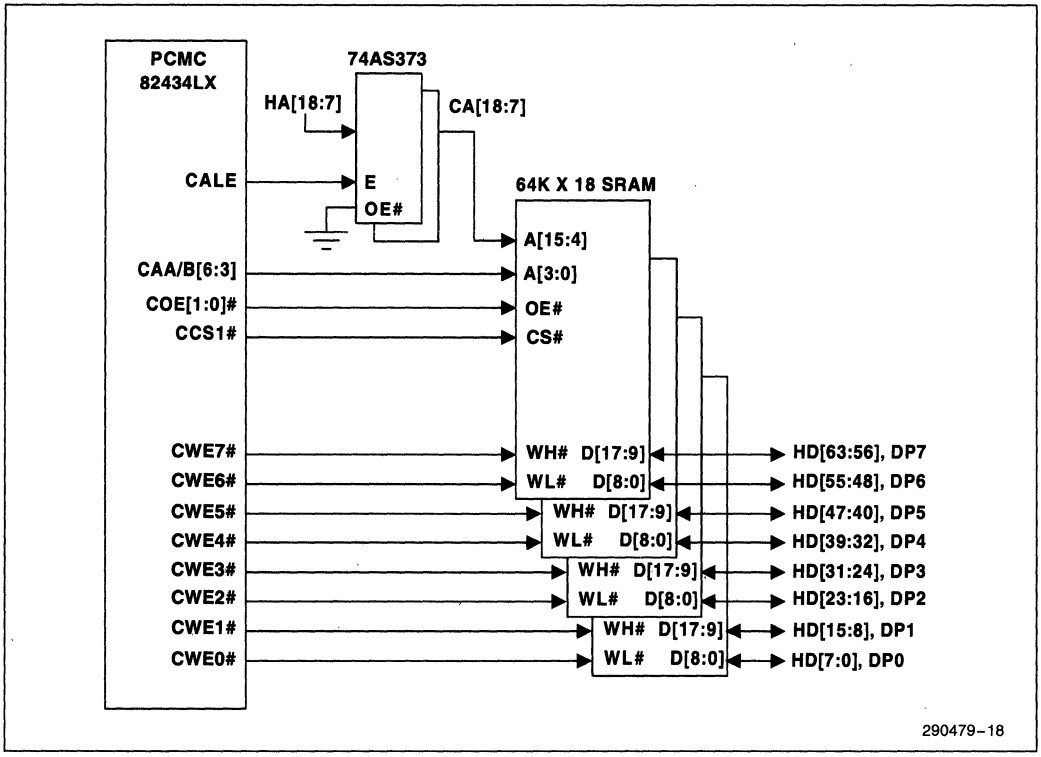


Figure 16. 82434LX Connections to 512-KByte Cache with Dual-Byte Select Standard SRAMs

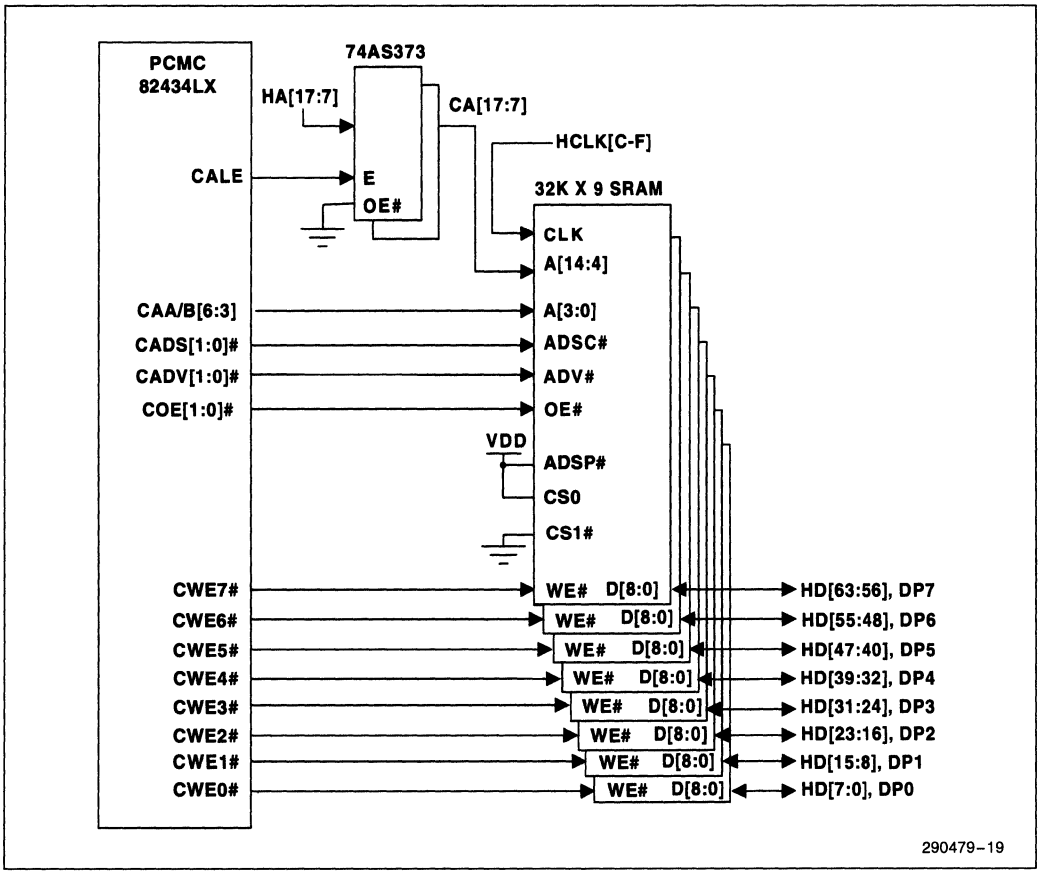


Figure 17. 82434LX Connections to 256-KByte Cache with Burst SRAM

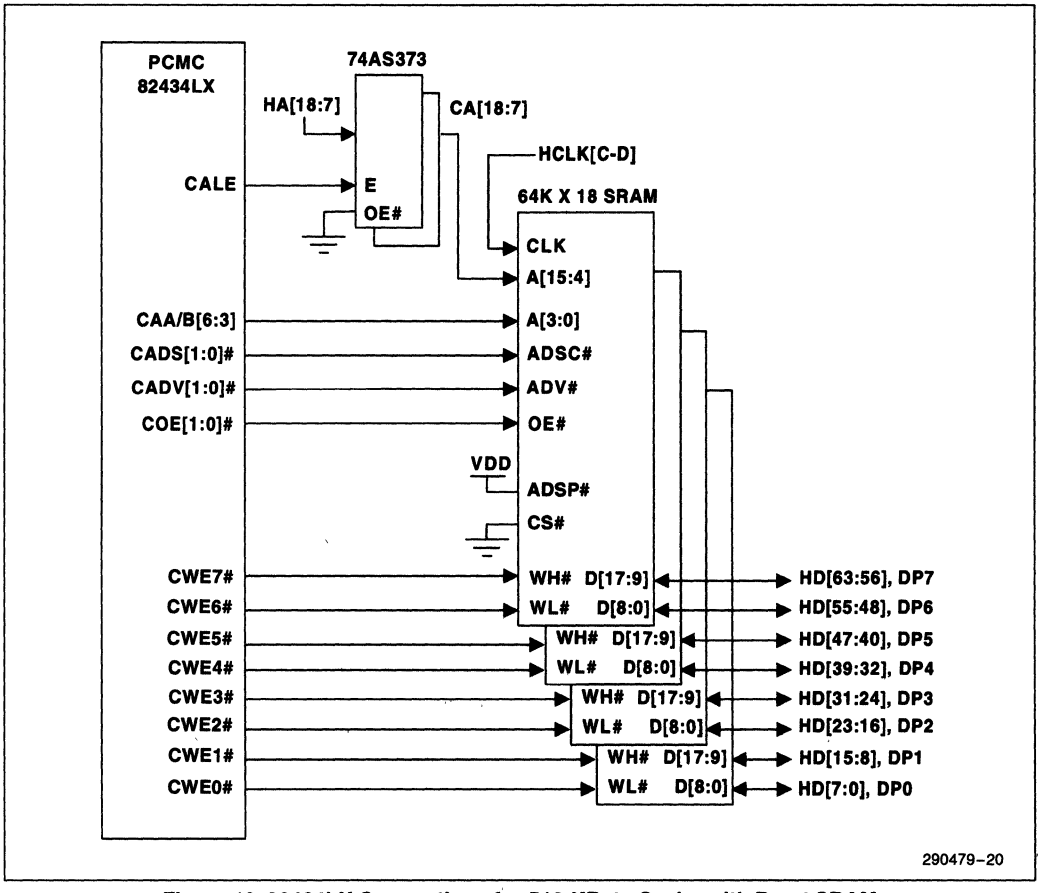


Figure 18. 82434LX Connections for 512-KByte Cache with Burst SRAM

When CALE is asserted, HA[18:7] flow through the address latch. When CALE is negated the address is captured in the latch allowing the processor to pipeline the next bus cycle onto the address bus. Two copies of CA[6:3], COE#, CADS# and CADV# are provided to reduce capacitive loading. Both copies should be used when the second level cache is implemented with eight 32K x 8 or 32K x 9 SRAMs. Either both copies or only one copy can be used with 64K x 18 or 64K x 16 SRAMs as determined by the system board layout and timing analysis. The two copies are always driven to the same logic level. CAA[4:3] and CAB[4:3] are used to count through the Pentium processor burst order when standard SRAMs are used to implement the cache.

With burst SRAMs, the address counting is provided inside the SRAMs. In this case, CAA[4:3] and CAB[4:3] are only used at the beginning of a cycle to load the initial low order address bits into the burst SRAMs. During CPU accesses, host address lines 6 and 5 are propagated to the CAA[6:5] and CAB[6:5] lines and are internally latched. When a CPU read cycle forces a line replacement in the second level cache, all modified lines within the addressed sector are written back to main memory. The PCMC uses CAA[6:5] and CAB[6:5] to select among the lines within the sector. The Cache Output Enables (COE[1:0]#) are asserted to enable the SRAMs to drive data onto the host data bus. The Cache Write Enables (CWE[7:0]#) allow byte control during CPU writes to the second level cache.

An asynchronous SRAM 512-KByte cache can be implemented with two different types of SRAM byte control. Figure 15 depicts the PCMC connections to a 512 KByte cache using 64K x 18 SRAMs or 64K x 16 SRAMs with two write enables per SRAM. Each SRAM has a high and low write enable. Figure 16 depicts the PCMC connections to a 512-KByte cache using 64K x 18 SRAMs or 64K x 16 SRAMs with two byte select lines per SRAM. Each SRAM has a high and low byte select.

The type of cache byte control (write enable or byte select) is programmed in the Cache Byte Control bit in the Secondary Cache Control Register at configuration space offset 52h. When this bit is set to 0, byte select control is used. In this mode, the CBS[7:0]# lines are multiplexed onto pins 90, 91, and 95-100 and CR/W[1:0]# pins are multiplexed onto pins 93 and 94. When this bit is set to 1, byte write enable control is used. In this mode, the CWE[7:0]# lines are multiplexed onto pins 90, 91, and 95-100. CADS[1:0]# and CADV[1:0]# are only used with burst SRAMs. The Cache Address Strobes (CADS[1:0]#) are asserted to cause the burst SRAMs to latch the cache address at the beginning of a second level cache access. CADS[1:0]# can be connected to either ADSP# or ADSC# on the SRAMs. The Cache Advance signals (CADV[1:0]#) are asserted to cause the burst SRAMs to advance to the next address of the burst sequence.

5.1.1 CLOCK LATENCIES (82434LX)

Table 5 and Table 6 list the latencies for various CPU transfers to or from the second level cache for standard SRAMs and burst SRAMs. Standard SRAM access times of 12 ns and 15 ns are recommended for 66 MHz and 60 MHz operation, respectively. Burst SRAM clock access times of 8 ns and 9 ns are recommended for 66 MHz and 60 MHz operation, respectively. Precise SRAM timing requirements should be determined by system board electrical simulation with SRAM I/O buffer models.

Table 5. Second Level Cache Latencies with Standard SRAM (82434LX)

Cycle Type	HCLK Count
Burst Read	3-2-2-2
Burst Write	4-2-2-2
Single Read	3
Single Write	4
Pipelined Back to Back Burst Reads	3-2-2-2/3-2-2-2
Burst Read followed by Pipelined Write	3-2-2-2/4

Table 6. Second Level Cache Latencies with Burst SRAM (82434LX)

Cycle Type	HCLK Count
Burst Read	3-1-1-1
Burst Write	3-1-1-1
Single Read	3
Single Write	3
Pipelined Back to Back Burst Reads	3-1-1-1/1-1-1-1
Read Followed by Pipelined Write	3-1-1-1/2

5.1.2 STANDARD SRAM CACHE CYCLES (82434LX)

The following sections describe the activity of the second level cache interface when standard asynchronous SRAMs are used to implement the cache.

5.1.2.1 Burst Read (82434LX)

Figure 19 depicts a burst read from the second level cache with standard SRAMs. The CPU initiates the read cycle by driving address and status onto the bus and asserting ADS#. Initially, the CA[6:3] are a propagation delay from the host address lines A[6:3]. Upon sampling W/R# active and M/IO# inactive, while ADS# is asserted, the PCMC asserts COE# to begin a read cycle from the SRAMs. CALE is negated, latching the address lines on the SRAM address inputs, allowing the CPU to pipeline a new address onto the bus. CA[4:3] cycle through the Pentium processor burst order, completing the cycle. PEN# is asserted with the first BRDY# and negated with the last BRDY# if parity is implemented on the second level cache data SRAMs and the MCHK DRAM/Second Level Cache Data Parity bit in the Error Command Register (offset 70h) is set.

Figure 20 depicts a burst read from the second level cache with standard 16- or 18-bit wide dual-byte select SRAMs. A single read cycle from the second level cache is very similar to the first transfer of a burst read cycle. CALE is not negated throughout the cycle. COE# is asserted as shown above, but is negated with BRDY#.

When the Secondary Cache Allocation (SCA) bit in the Secondary Cache Control Register is set to 1, the PCMC performs a line fill in the secondary cache, even if the CACHE# signal from the CPU is inactive. In this case, AHOLD is asserted to prevent the CPU from beginning a new cycle while the second level cache line fill is completing.

Back-to-back pipelined burst reads from the second level cache are shown in the Figure 21.

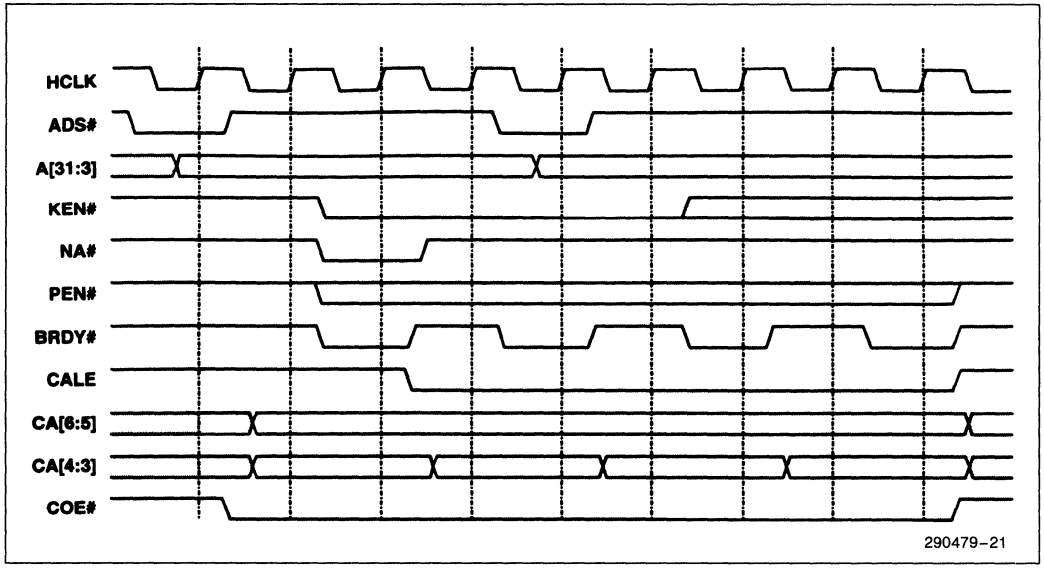


Figure 19. CPU Burst Read from Second Level Cache with Standard SRAM (82434LX)

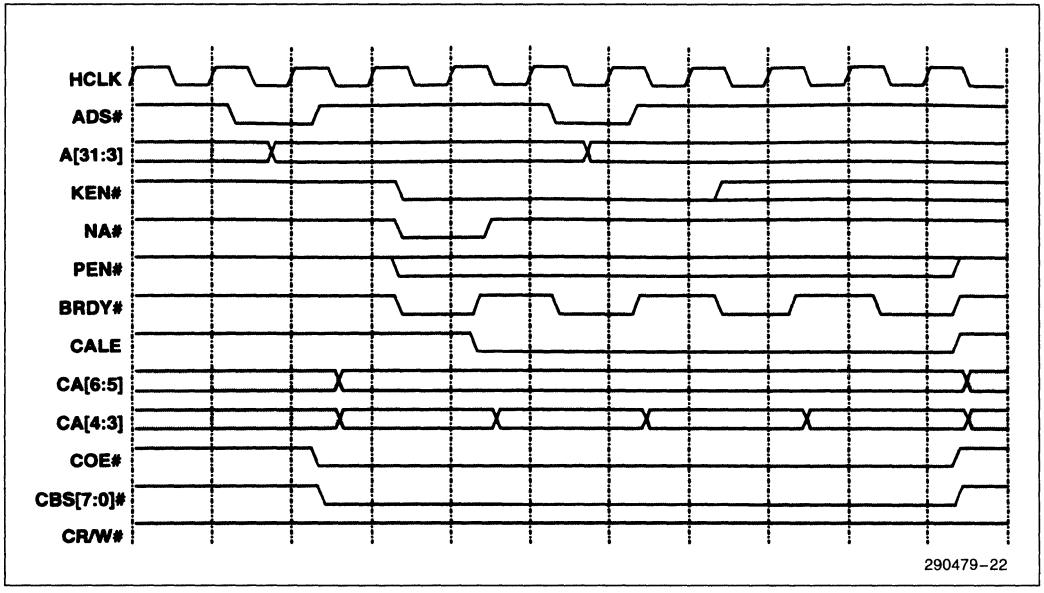


Figure 20. Burst Read from Second Level Cache with Dual-Byte Select SRAMs (82434LX)

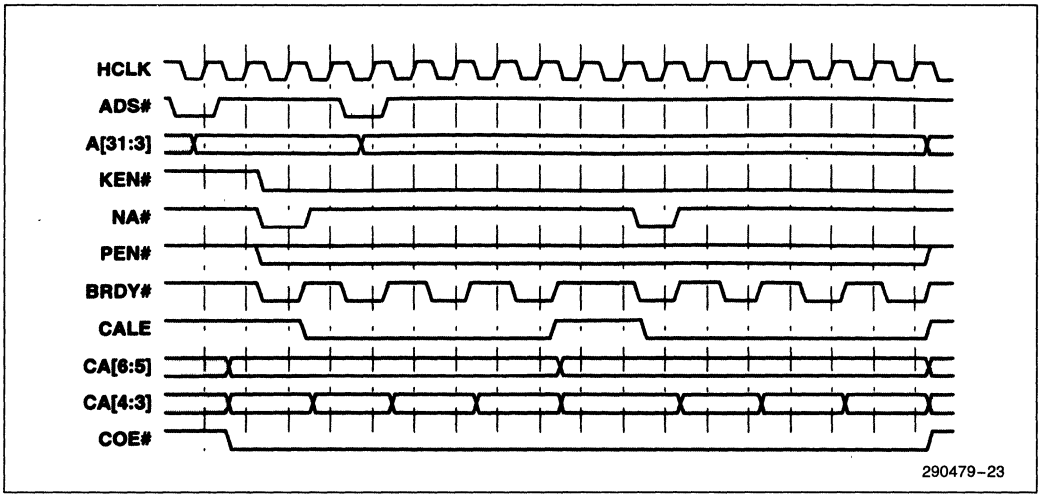


Figure 21. Pipelined Back-to-Back Burst Reads from Second Level Cache with Standard SRAM (82434LX)

Due to assertion of NA#, the CPU drives a new address onto the bus before the first cycle is complete. In this case, the second cycle is a hit in the second level cache. Immediately upon completion of the first read cycle, the PCMC begins the second cycle. When the first cycle completes, the PCMC drives the new address to the SRAMs on CA[6:3] and asserts CALE. The second cycle is very similar to the first, completing at a rate of 3-2-2-2. The cache address lines must be held at the SRAM address inputs until the first cycle completes. Only after the last BRDY# is returned, can CALE be asserted and CA[6:3] be changed. Thus, the pipelined cycle completes at the same rate as a non-pipelined cycle.

5.1.2.2 Burst Write (82434LX)

A burst write cycle is used to write back a cache line from the first level cache to either the second level cache or DRAM. Figure 22 depicts a burst write cycle to the second level cache with standard SRAMs. The CPU initiates the write cycle by driving address and status onto the bus and asserting ADS#. Initially, the CA[6:3] propagate from the host address lines A[6:3]. CALE is negated, latching the address lines on the SRAM address inputs, allowing the CPU to pipeline a new address onto the bus. Burst write cycles from the Pentium processor always begin with the low order Qword and advances to the high order Qword. CWE[7:0]# are generated from an internally delayed version of HCLK, providing address setup time to CWE[7:0]# falling and data setup time to CWE[7:0]# rising edges. HIG[4:0] are driven to PCMWQ (Post CPU to Memory Write Buffer Qword) only when the PCMC is programmed for a write-through write policy. When programmed for write-back mode, the modified bit associated with the line is set within the PCMC. The single write cycle is very similar to the first write of a burst write cycle. A burst read cycle followed by a pipelined write cycle with standard SRAMs is depicted in Figure 24.

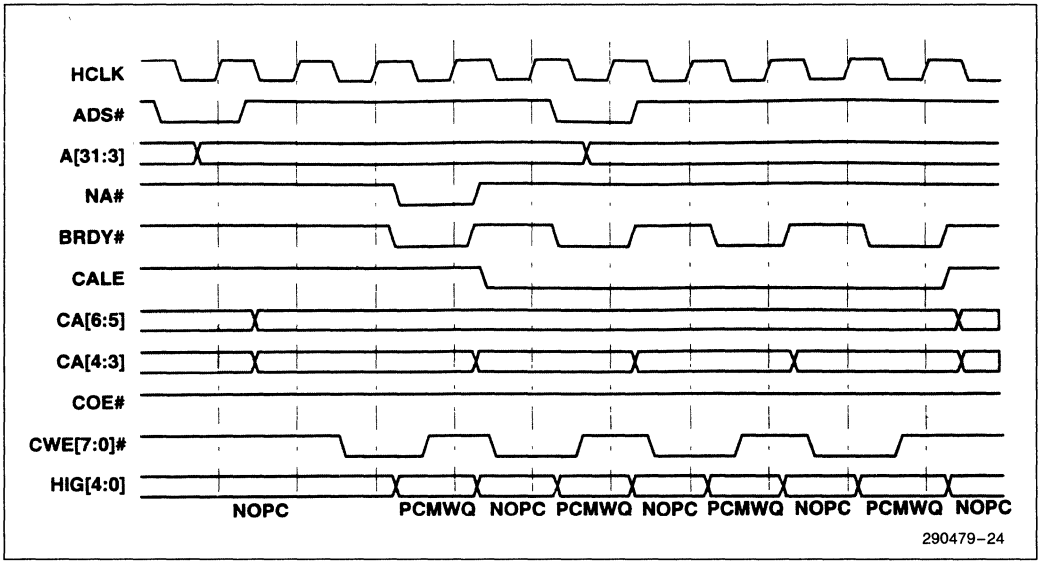


Figure 22. Burst Write to Second Level Cache with Standard SRAM (82434LX)

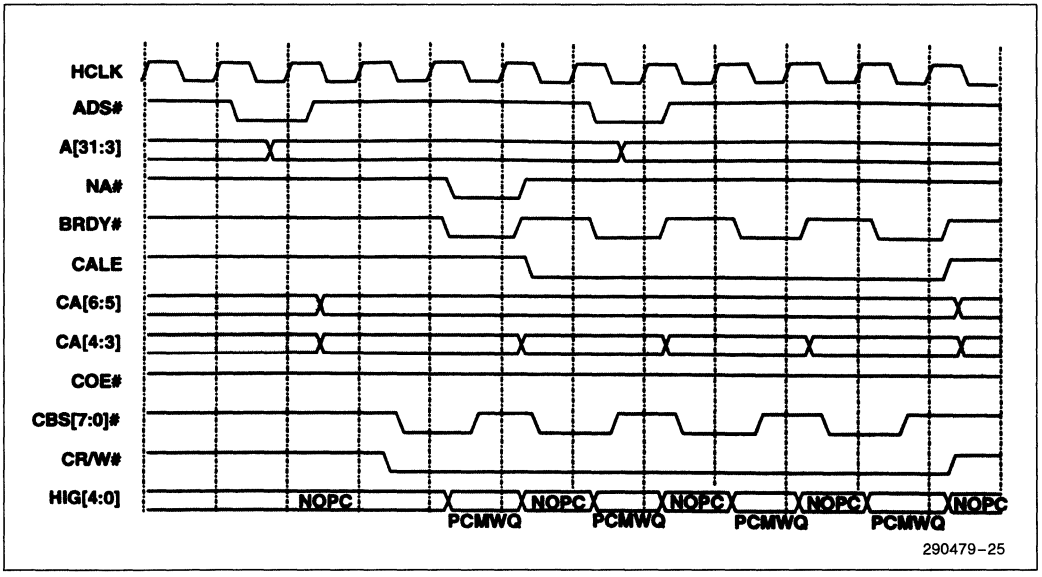


Figure 23. Burst Write to Second Level Cache with Dual-Byte Select Standard SRAMs (82434LX)

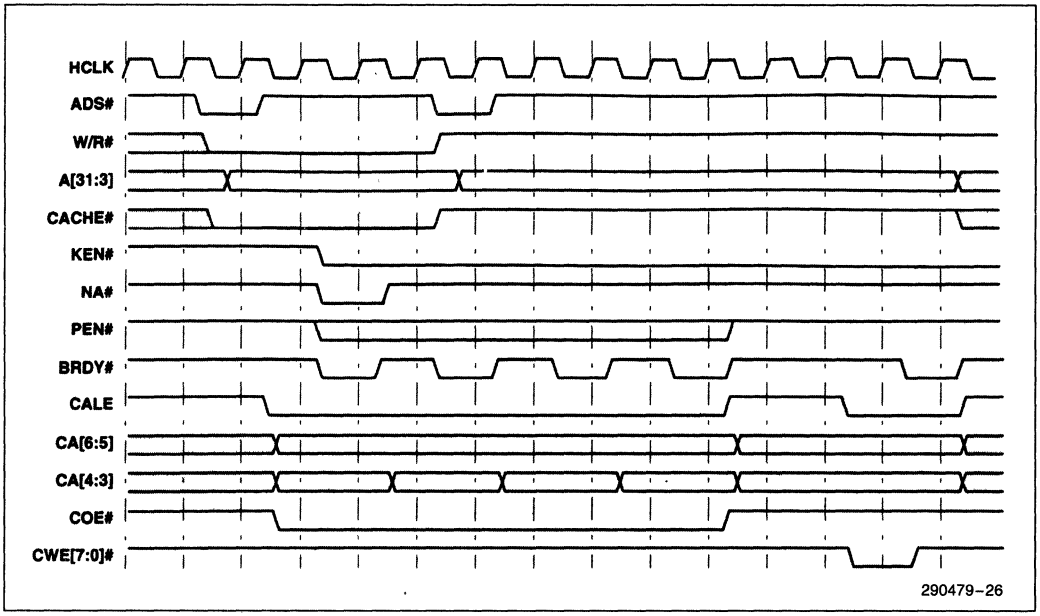


Figure 24. Burst Read Followed by Pipelined Write with Standard SRAM (82434LX)

5.1.2.3 Cache Line Fill (82434LX)

If the CPU issues a memory read cycle to cacheable memory that is not in the second level cache, a first and second level cache line fill occurs. Figure 25 depicts a CPU read cycle that results in a line fill into the first and second level caches.

Figure 27 depicts the host bus activity during a CPU read cycle that forces a write-back from the second level cache to the CPU-to-memory posted write buffer as the DRAM read cycle begins.

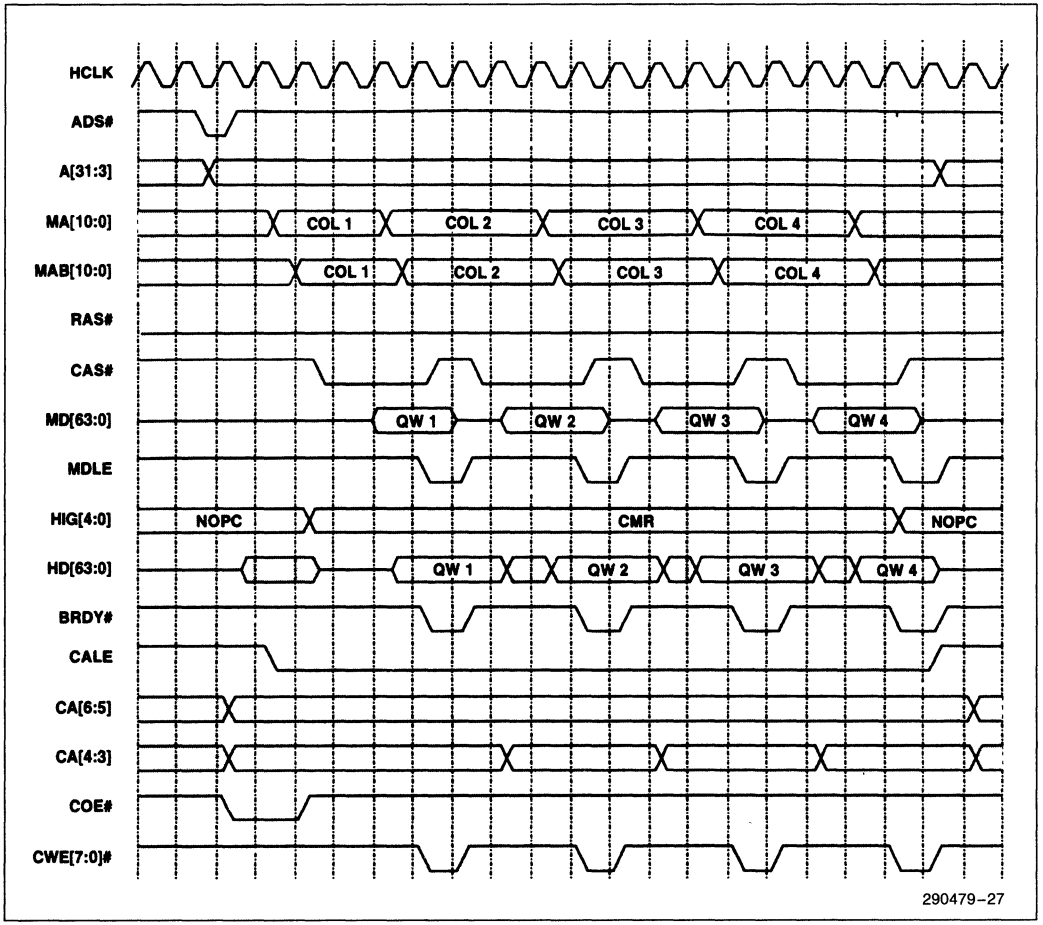


Figure 25. Cache Line Fill with Standard SRAM, DRAM Page Hit (82434LX)

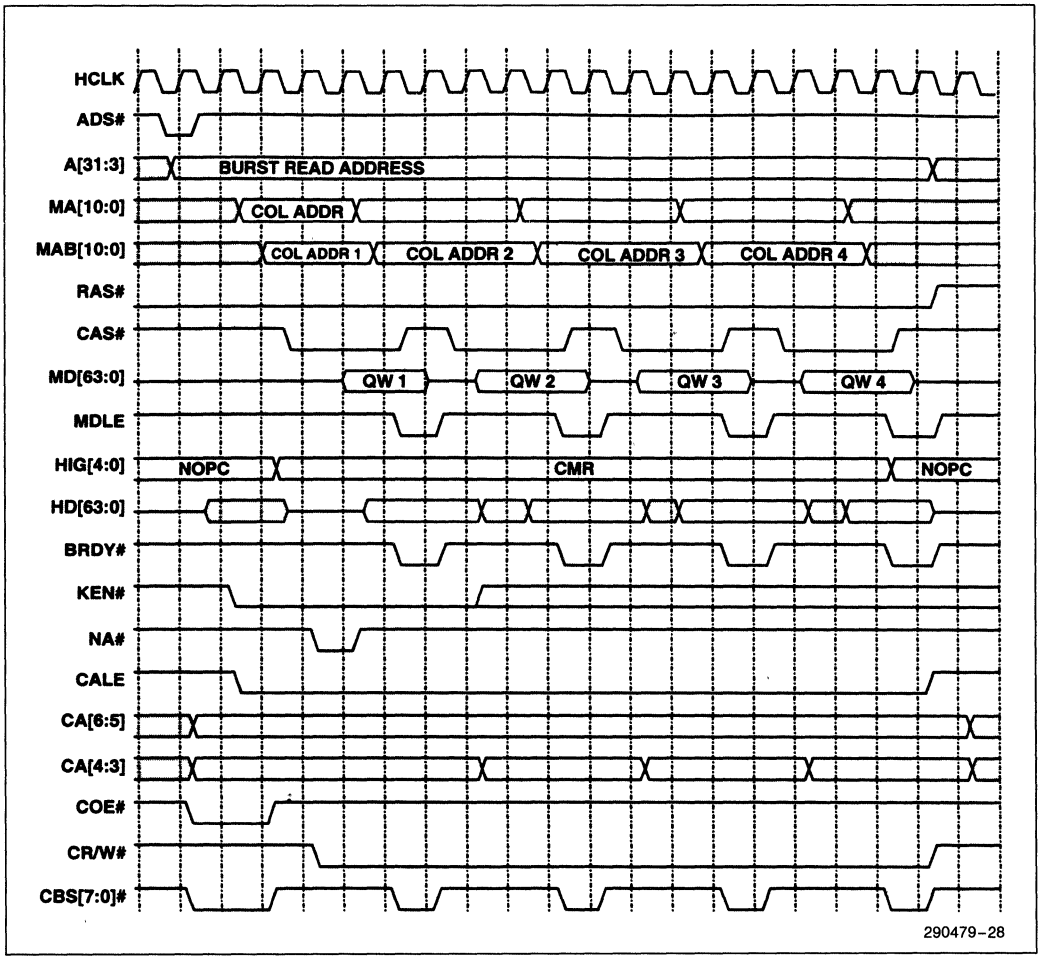


Figure 26. Cache Line Fill with Dual-Byte Select Standard SRAM, DRAM Page Hit (82434LX)

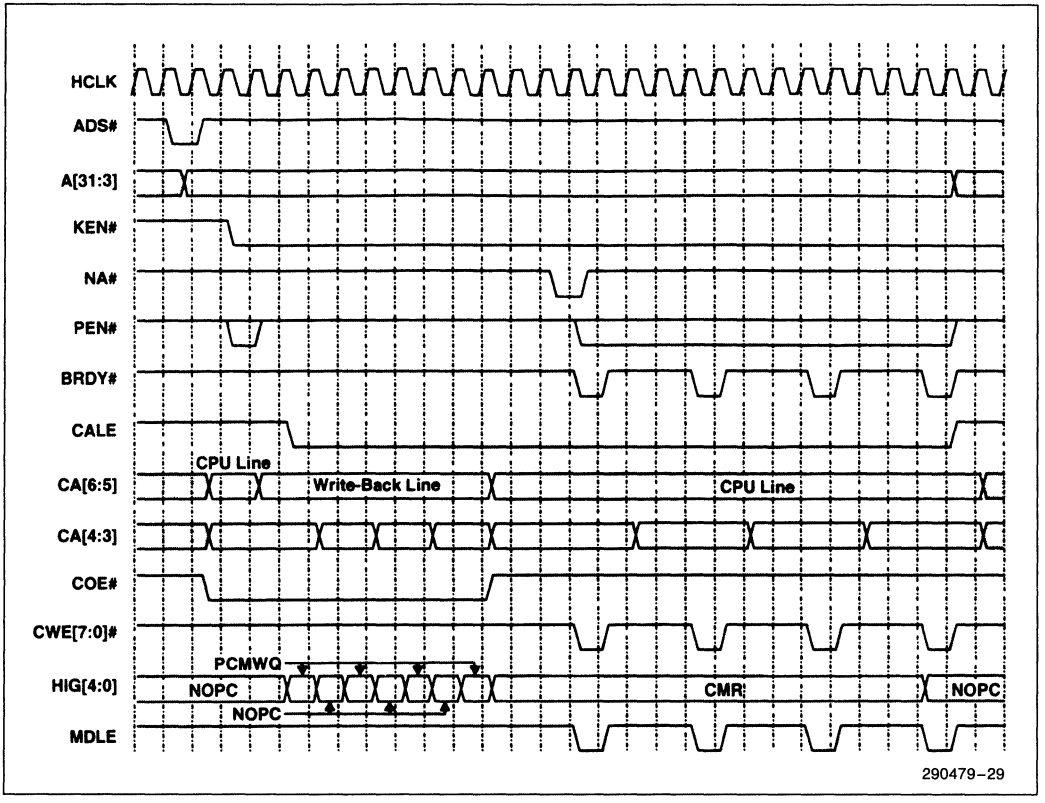


Figure 27. CPU Cache Read Miss, Write-Back, Line Fill with Standard SRAM (82434LX)

The CPU issues a memory read cycle that misses in the second level cache. In this instance, a modified line in the second level cache must be written back to main memory before the new line can be filled into the cache. The PCMC inspects the valid and modified bits for each of the lines within the addressed sector and writes back only the valid lines within the sector that are in the modified state. During the write-back cycle, CA[4:3] begin with the initial value driven by the Pentium processor and proceed in the Pentium processor burst order. CA[6:5] are used to count through the lines within the addressed sector. When two or more lines must be written back to main memory, CA[6:5] count in the direction from line 0 to line 3. CA[6:5] advance to the next line to be written back to main memory, skipping lines that are not modified. Figure 23 depicts the case of just one of the lines in a sector being written back to main memory. In this case, the entire line can be posted in the CPU-to-Main memory posted write buffer by driving the HIG[4:0] lines to the PCMWQ command as each Qword is read from the cache. At the same time, the required DRAM read cycle is beginning. As soon as the de-allocated line is written into the posted write buffer, the HIG[4:0] lines are driven to CMR (CPU Memory Read) to allow data to propagate from the DRAM data lines to the CPU data lines. The CWE[7:0]# lines are not generated from a delayed version of HCLK (as they are in the case of CPU to second level cache burst write), but from ordinary HCLK rising edges. CMR is driven on the HIG[4:0] lines

throughout the DRAM read portion of the cycle. With the fourth assertion of BRDY# the HIG[4:0] lines change to NOPC. The LBXs however, do not tri-state the host data lines until MDLE rises. CWE[7:0]# and MDLE track such that MDLE will not rise before CWE[7:0]#. Thus, the LBXs continue to drive the host data lines until CWE[7:0]# are negated. CA[6:3] remain at the valid values until the clock after the last BRDY#, providing address hold time to CWE[7:0]# rising.

PEN# is asserted as shown if the MCHK DRAM/L2 Cache Data Parity Error bit in the Error Command Register (offset 70h) is set. If the second level cache supports parity, PEN# is always asserted during CPU read cycles in the third clock in case the cycle hits in the cache.

If more than one line must be written back to main memory, the PCMC fills the CPU-to-Main Memory Posted Write Buffer and loads another Qword into the buffer as each Qword write completes into main memory. The writes into DRAM proceed as page hit write cycles from one line to the next, completing at a rate of X-4-4-4-5-4-4-4-5-4-4-4 for a three line write-back. All modified lines except for the last one to be written back are posted and written to memory before the DRAM read cycle begins. The last line to be written back is posted as the DRAM read cycle begins. Thus, the read data is returned to the CPU before the last line is retired to memory.

The line which was written into the second level cache is marked valid and unmodified by the PCMC. All the other lines in the sector are marked invalid. A subsequent CPU read cycle which hits in the same sector (but a different line) in the second level cache would then simply result in a line fill without any write-back.

5.1.3 BURST SRAM CACHE CYCLES (82434LX)

The following sections show the activity of the second level cache interface when burst SRAMs are used for the second level cache.

5.1.3.1 Burst Read (82434LX)

Figure 28 depicts a burst read from the second level cache with burst SRAMs.

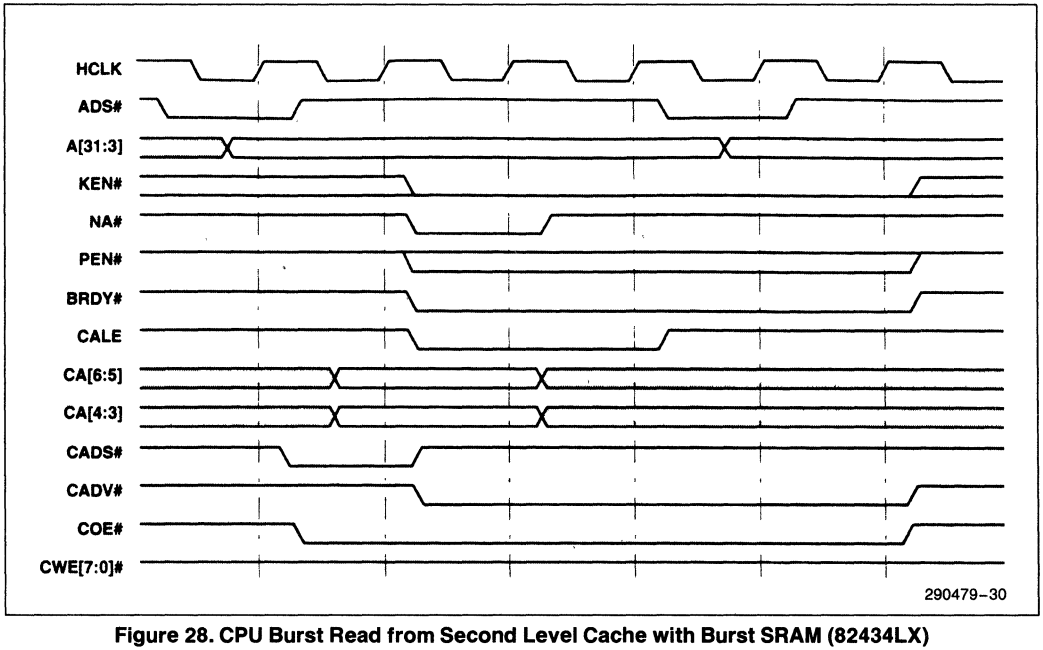


Figure 28. CPU Burst Read from Second Level Cache with Burst SRAM (82434LX)

The cycle begins with the CPU driving address and status onto Host Bus and asserting ADS#. The PCMC asserts CADS# and COE# in the second clock. After the address is latched by the burst SRAMs and the PCMC determines that no write-back cycles are required from the second level cache, CALE is negated. Back-to-back burst reads from the second level cache are shown in Figure 29.

When the Secondary Cache Allocation (SCA) bit in the Secondary Cache Control Register is set to 1, the PCMC performs a line fill in the secondary cache, even if the CACHE# signal from the CPU is negated. In this case, AHOLD is asserted to prevent the CPU from beginning a new cycle while the second level cache line fill is completing.

Back-to-back burst reads which hit in the second level cache complete at a rate of 3-1-1-1/1-1-1-1 with burst SRAMs. As the last BRDY# is being returned to the CPU, the PCMC asserts CADS# causing the SRAMs to latch the new address. This allows the data for the second cycle to be transferred to the CPU on the clock after the first cycle completes.

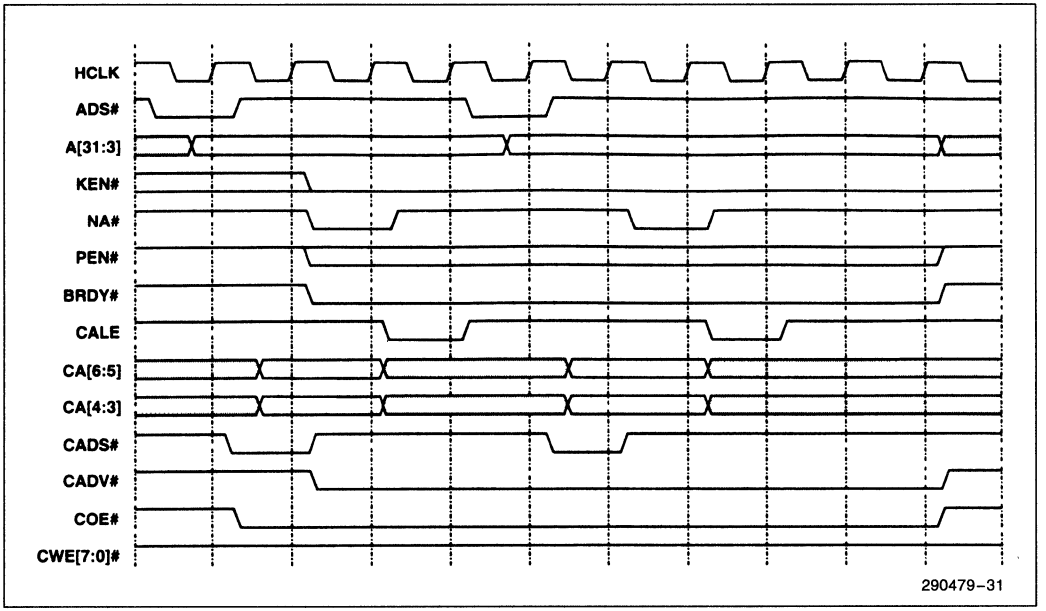


Figure 29. Pipelined Back-to-Back Burst Reads from Second Level Cache (82434LX)

5.1.3.2 Burst Write (82434LX)

A burst write cycle is used to write back a line from the first level cache to either the second level cache or DRAM. A burst write cycle from the first level cache to the second level cache is shown in Figure 30.

The Pentium processor always writes back lines starting with the low order Qword advancing to the high order Qword. CADS# is asserted in the second clock. CWE[7:0]# and BRDY# are asserted in the third clock. CADV# assertion is delayed by one clock relative to the burst read cycle. HIG[4:0] are driven to PCMWQ (Post CPU-to-Memory Write Buffer Qword) only when the PCMC is programmed for a write-through write policy. When programmed for write-back mode, the modified bit associated with the line is set within the PCMC. The single write is very similar to the first write in a burst write. CADS# is asserted in the second clock. BRDY# and CWE[7:0]# are asserted in the third clock. A burst read cycle followed by a pipelined single write cycle is depicted in Figure 31.

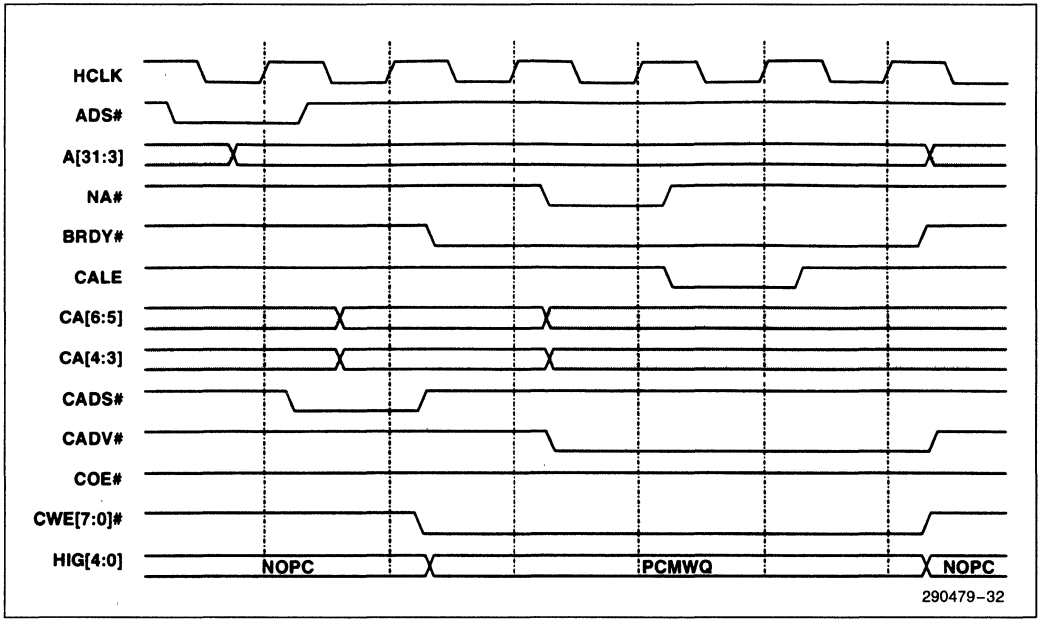


Figure 30. Burst Write to Second Level Cache with Burst SRAM (82434LX)

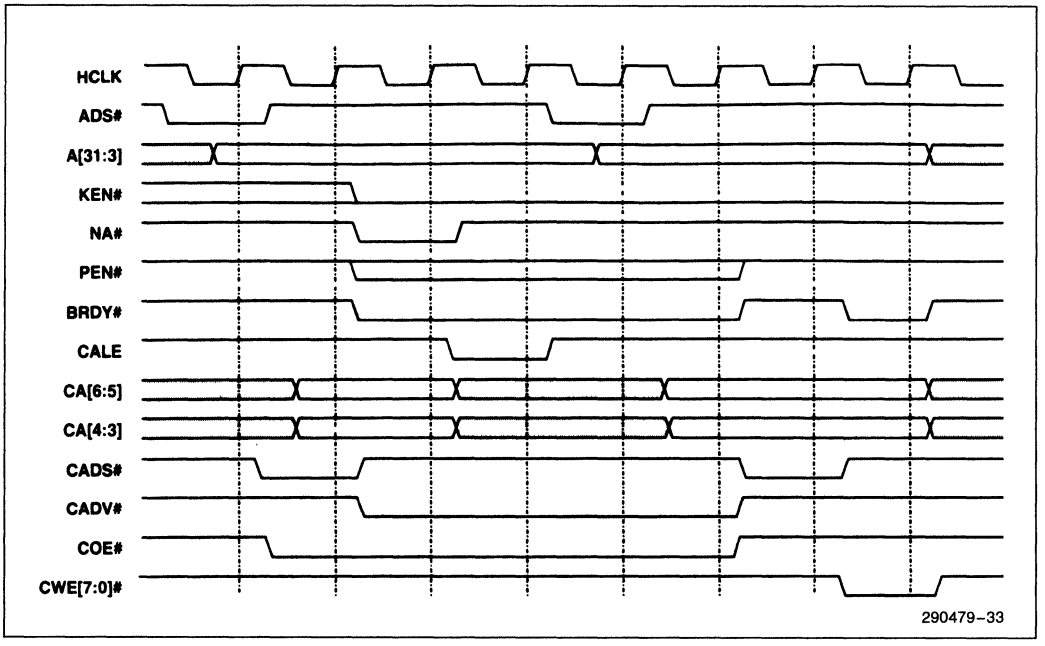


Figure 31. Burst Read Followed by Pipelined Single Write Cycle with Burst SRAM (82434LX)

5.1.3.3 Cache Line Fill (82434LX)

If the CPU issues a memory read cycle to cacheable memory which does not hit in the second level cache, a cache line fill occurs. Figure 32 depicts a first and second level cache line fill with burst SRAMs.

Figure 33 depicts a CPU read cycle which forces a write-back in the second level cache.

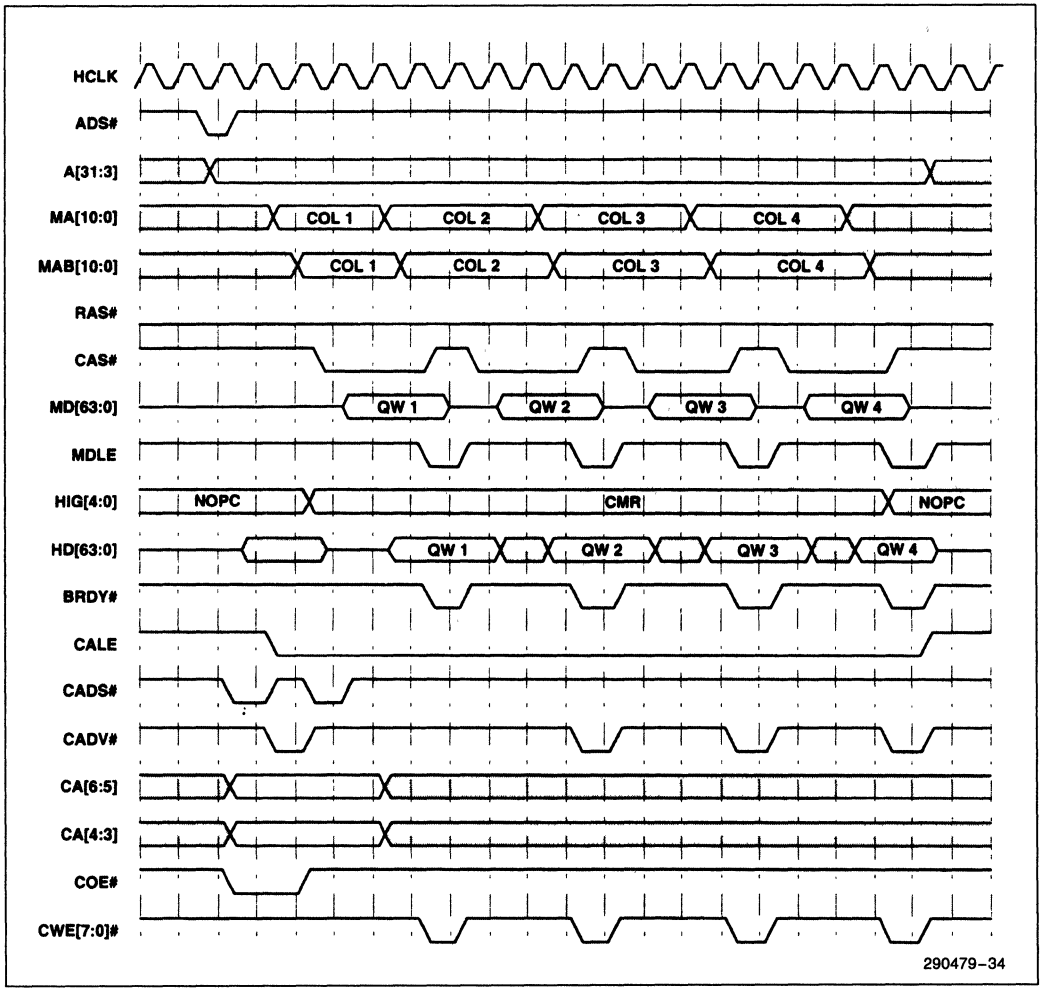


Figure 32. Cache Line Fill with Burst SRAM, DRAM Page Hit, 7-4-4-4 Timing (82434LX)

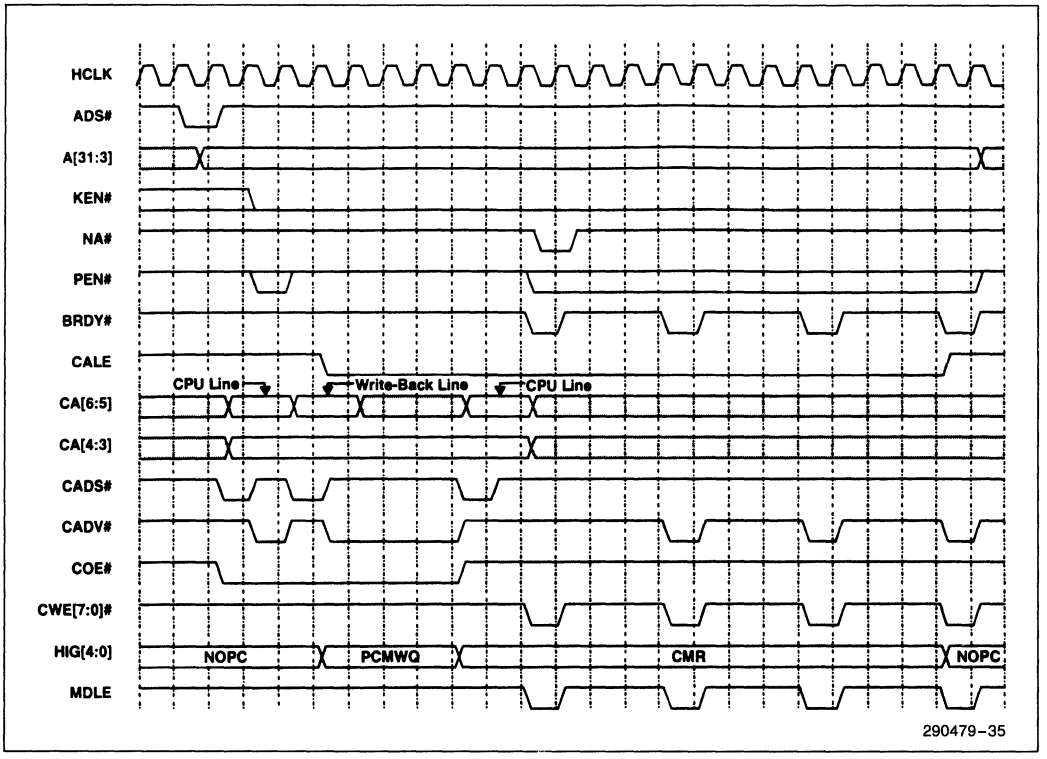


Figure 33. CPU Cache Read Miss, Write-Back, Line Fill with Burst SRAM (82434LX)

The CPU issues a memory read cycle which misses in the second level cache. In this instance, a modified line in the second level cache must be written back to main memory before the new line can be filled into the cache. The PCMC inspects the valid and modified bits for each of the lines within the addressed sector and writes back only the valid lines within the sector that are marked modified. CA[6:5] are used to count through the lines within the addressed sector. When two or more lines must be written back to main memory, CA[6:5] count in the direction from line 0 to line 3 after each line is written back. Figure 29 depicts the case of just one

of the lines in a sector being written back to main memory. In this case, the entire line can be posted in the CPU-to-Memory Posted Write Buffer by driving the HIG[4:0] lines to PCMWQ as each Qword is read from the cache. At the same time, the required DRAM read cycle is beginning. After the de-allocated line is written into the posted write buffer, the HIG[4:0] lines are driven to CMR (CPU Memory Read) to allow data to propagate from the DRAM data lines to the CPU data lines. Figure 29 assumes that the read from DRAM is a page hit and thus the first Qword is already read from the DRAMs when the transfer from cache to the CPU to Memory posting buffer is complete. The rest of the DRAM cycle completes at a -4-4-4 rate. CADV# is asserted with the last three BRDY# assertions. CMR is driven on the HIG[4:0] lines throughout the DRAM read portion of the cycle. Upon the fourth assertion of BRDY# the HIG[4:0] lines change to NOPC.

PEN# is asserted as shown if the MCHK DRAM/L2 Cache Data Parity Error bit in the Error Command Register (offset 70h) is set. If the second level cache supports parity, PEN# is always asserted during CPU read cycles in clock 3 in case the cycle hits in the cache.

If more than one line must be written back to main memory, the PCMC fills the CPU-to-Main Memory Posted Write Buffer and loads another Qword into the buffer as each Qword write completes into main memory. The writes into DRAM proceed as page hit write cycles from one line to the next, completing at a rate of X-4-4-4-5-4-4-4-5-4-4-4 for a three line write-back when programmed for X-4-4-4 DRAM write timing or X-3-3-3-4-3-3-3-4-3-3-3 when programmed for X-3-3-3 DRAM write timing. All modified lines except for the last one to be written back to memory are posted and retired to memory before the DRAM read cycle begins. The last line to be written back is posted as the DRAM read cycle begins. Thus, the read data is returned to the CPU before the last line is retired to memory.

The line which was written into the second level cache is marked valid and unmodified by the PCMC. All the other lines in the block are marked invalid. A subsequent CPU read cycle which hits the same sector (but a different line) in the second level cache results in a line fill without any write-back.

5.1.4 SNOOP CYCLES

Snoop cycles are the same for the 82434LX and 82434NX. The inquire cycle is used to probe the first level and second level caches when a PCI master attempts to access main memory. This is done to maintain coherency between the first and second level caches and main memory. When a PCI master first attempts to access main memory a snoop request is generated inside the PCMC. The PCMC supports up to two outstanding cycles on the CPU address bus at a time. Outstanding cycles include both CPU initiated cycles and snoop cycles. Thus, if the Pentium processor pipelines a second cycle onto the host address bus, the PCMC will not issue a snoop cycle until the first CPU cycle terminates. If the PCMC were to initiate a snoop cycle before the first CPU cycle were complete then for a brief period of time, three cycles would be outstanding. Thus, a snoop request is serviced with a snoop cycle only when either no cycle is outstanding on the CPU bus or one cycle is outstanding.

Snoop cycles are performed by driving the PCI master address onto the CPU address bus and asserting EADS#. The Pentium processor then performs a tag lookup to determine if the addressed memory is in the first level cache. At the same time the PCMC performs an internal tag lookup to determine if the addressed memory is in the second level cache. Table 7 describes how a PCI master read from main memory is serviced by the PCMC.

Table 7. Data Transfers for PCI Master Reads from Main Memory

Snoop Result		Action
First Level Cache	Second Level Cache	
Miss	Miss	Data is transferred from DRAM to PCI.
Miss	Hit Unmodified Line	Data is transferred directly from second level cache to PCI. The line remains valid and unmodified in the second level cache.
Miss	Hit Modified Line	Data is transferred directly from second level cache to PCI. Line remains valid and modified in the second level cache. The line is not written to DRAM.
Hit Unmodified Line	Miss	Data is transferred from DRAM to PCI.
Hit Unmodified Line	Hit Unmodified Line	Data is transferred directly from second level cache to PCI. The line remains valid and unmodified in the second level cache.
Hit Unmodified Line	Hit Modified Line	Data is transferred directly from second level cache to PCI. Line remains valid and modified in the second level cache. The line is not written to DRAM.
Hit Modified Line	Miss	A write-back from first level cache occurs. The data is sent to both PCI and the CPU-to-Memory Posted Write Buffer. The CPU-to-Memory Posted Write Buffer is then written to memory.
Hit Modified Line	Hit Unmodified Line	A write-back from first level cache occurs. The data is posted to PCI and written into the second level cache. When the second level cache is in write-back mode, the line is marked modified and is not written to DRAM. When the second level cache is in write-through mode, the line is posted and then written to DRAM.
Hit Modified Line	Hit Modified Line	A write-back from first level cache occurs. The data is posted to PCI and written into the second level cache. The line is not written to DRAM. This scenario can only occur when the second level cache is in write-back mode.

PCI master write cycles never result in a write directly into the second level cache. A snoop hit to a modified line in either the first level or second level cache results in a write-back of the line to main memory. The line is invalidated and the PCI write to main memory occurs after the write-back completes. The other lines in the sector are not written back to main memory or invalidated. A PCI master write snoop hit to an unmodified line in either the first level or second level cache results in the line being invalidated. Table 8 describes the actions taken by the PCMC when a PCI master writes to main memory.

Table 8. Data Transfers for PCI Master Writes to Main Memory

Snoop Result		Action
First Level Cache	Second Level Cache	
Miss	Miss	The PCI master write data is transferred from PCI to DRAM.
Miss	Hit Unmodified Line	The PCI master write data is transferred from PCI to DRAM. The line is invalidated in the second level cache.
Miss	Hit Modified Line	A write-back from second level cache to DRAM occurs. The PCI master write data is then written to DRAM. The line is invalidated in the second level cache.
Hit Unmodified Line	Miss	The first level cache line is invalidated. The PCI master write data is written to DRAM.
Hit Unmodified Line	Hit Unmodified Line	The line is invalidated in both the first level and second level caches. The PCI master write data is written to DRAM.
Hit Unmodified Line	Hit Modified Line	The first level cache line is invalidated. The second level cache line is written back to main memory and invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	Miss	The first level cache line is written back to DRAM and invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	Hit Unmodified Line	The first level cache line is written back to DRAM and invalidated. The second level cache line is invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	Hit Modified Line	The first level cache line is written back to DRAM and invalidated. The second level cache line is invalidated. The PCI master write data is then written to DRAM.

A snoop hit results in one of three transfers; a write-back from the first level cache posted to the LBXs, a write-back from the second level cache posted to the LBXs or a write-back from the first level cache posted to the LBXs and written to the second level cache. A snoop cycle that does not result in a write-back is depicted in Figure 34.

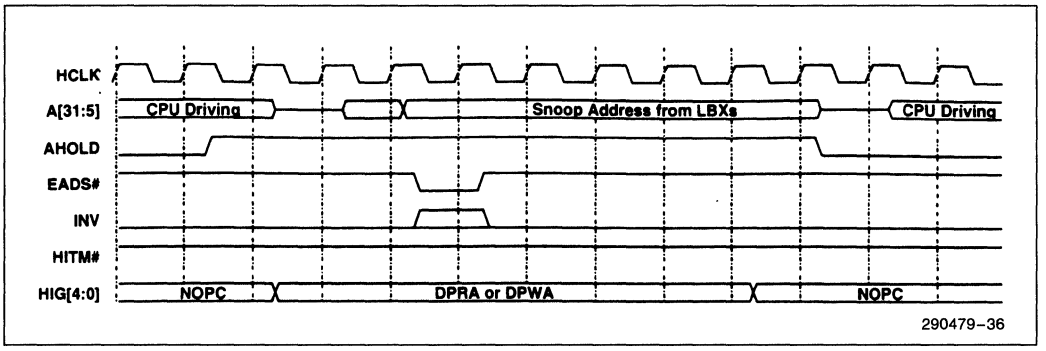


Figure 34. Snoop Hit to Unmodified Line in First Level Cache or Snoop Miss

The PCMC begins to service the snoop request by asserting AHOLD, causing the Pentium processor to tri-state the address bus in the clock after assertion. In the case of a PCI master read cycle, the PCMC drives the DPRA (Drive PCI Read Address) command onto the HIG[4:0] lines causing the LBXs to drive the PCI address onto the host address bus. For a write cycle, the PCMC drives the DPWA (Drive PCI Write Address to CPU Address Bus) command on the HIG[4:0] lines, also causing the LBXs to begin driving the host address bus. The PCMC then asserts EADS#, initiating the snoop cycle to the CPU. The INV signal is asserted by the PCMC only during snoops due to PCI master writes. INV remains negated during snoops due to PCI master reads. If the snoop results in a hit to a modified line in the first level cache, the Pentium processor asserts HITM#. The PCMC samples the HITM# signal two clocks after the CPU samples EADS# asserted to determine if the snoop hit in the first level cache. By this time the PCMC has completed an internal tag lookup to determine if the line is in the second level cache. Since this snoop does not result in a write-back, the NOPC command is driven on the HIG[4:0] lines, causing the LBXs to tri-state the address bus. The sequence ends with AHOLD negation.

If the Pentium processor asserts ADS# in the same clock as the PCMC asserts AHOLD, the PCMC will assert BOFF# in two cases. First, if the snoop cycle hits a modified line in the first level cache, the PCMC will assert BOFF# for 1 HCLK to re-order the write-back around the currently sending cycle. Second, if the snoop requires a write-back from the second level cache, the PCMC will assert BOFF# to enable the write-back from the secondary cache SRAMs.

Figure 35 depicts a snoop hit to a modified line in the first level cache due to a PCI master memory read cycle.

The snoop cycle begins when the PCMC asserts AHOLD causing the CPU to tri-state the address bus. The PCMC drives the DPRA (Drive PCI Read Address) command on to the HIG[4:0] lines causing the LBXs to drive the PCI address onto the host address bus. The PCMC then asserts EADS#, initiating the snoop to the first level cache. INV is not asserted since this is a PCI master read cycle. INV is only asserted with EADS# when the snoop cycle is in response to a PCI master write cycle. As the CPU is sampling EADS# asserted, the PCMC latches the address. Two clocks later, the PCMC completes the

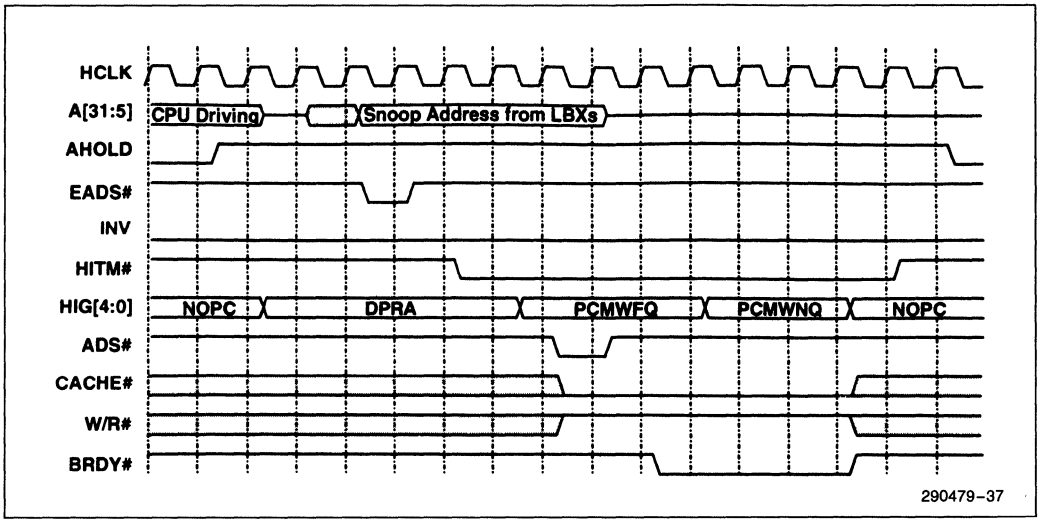


Figure 35. Snoop Hit to Modified Line in First Level Cache, Post Memory and PCI

internal tag lookup to determine if the line is in the second level cache. In this instance, the snoop hits a modified line in the first level cache and misses in the second level cache. Thus, the second level cache is not involved in the write-back cycle. The PCMC allows the LBXs to stop driving the address lines by driving NOPC command on the HIG[4:0] lines. The CPU then drives the write-back cycle onto the bus by asserting ADS# and driving the write-back data on the data lines even though AHOLD is still asserted. The write-back into the LBX buffers occurs at a rate of 3-1-1-1. The PCMC drives PCMWFQ on the HIG[4:0] lines for one clock causing the write data to be posted to both PCI and main memory. For the next three clocks, the HIG[4:0] lines are driven to PCMWNQ, posting the final three Qwords to both PCI and main memory.

A similar transfer from first level cache to the LBXs occurs when a snoop due to a PCI master write hits a modified line in the first level cache. In this case, the write-back is transferred to the CPU-to-Memory Posted Write Buffer. If the line is in the second level cache, it is invalidated. The cycle is similar to the snoop cycle shown above with two exceptions. The PCMC drives the DPWA command on the HIG[4:0] lines instead of the DPRA command. During the four clocks where the PCMC drives BRDY# active to the CPU, it also drives PCMWQ on the HIG[4:0] lines, causing the write to be posted to main memory.

In both of the above cases where a write-back from the first level cache is required, AHOLD is asserted until the write-back is complete. If the CPU has begun a read cycle directed to PCI and the snoop results in a hit to a modified line in the first level cache, BOFF# is asserted for one clock to abort the CPU read cycle and re-order the write-back cycle before the read cycle.

When a PCI master read or write cycle hits a modified line in the second level cache and either misses in the first level cache or hits an unmodified line in the first level cache, a write-back from the second level cache to the LBXs occurs. When a PCI master write snoop hits an unmodified line in the second level cache and either misses in the first level cache or hits an unmodified line in the first level cache, no data transfer from the second level cache occurs. The line is simply invalidated. In the case of a PCI master write cycle, the line is invalidated in both the first level and second level caches. In the case of a PCI master memory read cycle, neither cache is invalidated. A PCI master read from main memory which hits either a modified or unmodified line in the second level cache is shown in Figure 36.

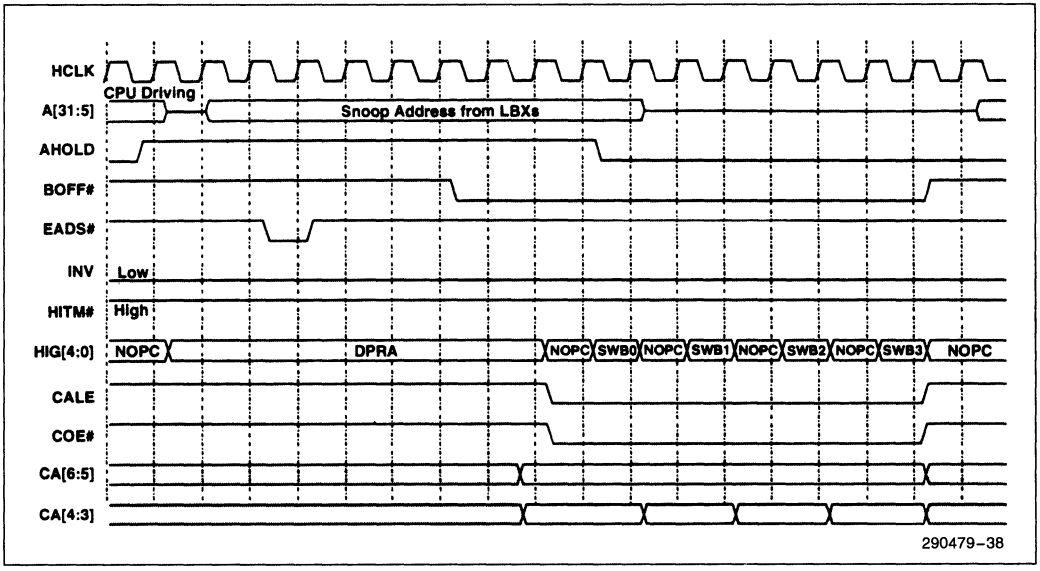


Figure 36. Snoop Hit to Modified Line in Second Level Cache, Store in PCI Read Prefetch Buffer

The snoop cycle begins with the PCMC asserting AHOLD, causing the CPU to tri-state the host address bus. The PCMC drives the DPRA command enabling the LBXs to drive the snoop address onto the host address bus. The PCMC asserts EADS#. INV is not asserted in this case since the snoop cycle is in response to a PCI master read cycle. If the snoop were in response to a PCI master write cycle then INV would be asserted with EADS#. Two clocks after the CPU samples EADS# active, the PCMC completes the internal tag lookup. In this case the snoop hit either an unmodified line or a modified line in the second level cache. Since HITM# is inactive, the snoop did not hit in the first level cache. The PCMC then schedules a read from the second level cache to be written to the LBXs. When the CPU burst cycle completes the PCMC negates the control signals to the second level cache and asserts CALE opening the cache address latch and allowing the snoop address to flow through to the SRAMs. The second level cache executes a read sequence which completes at 3-2-2-2 in the case of standard SRAMs and 3-1-1-1 in the case of burst SRAMs. During all snoop cycles where a write-back from the second level cache is required, BOFF# is asserted throughout the write-back cycle. This prevents the deadlock that would occur if the CPU is in the middle of a non-postable write and the data bus is required for the second level cache write-back.

When using burst SRAMs, the read from the SRAMs follows the Pentium processor burst order. However, the memory to PCI read prefetch buffer in the LBXs is organized as a FIFO and cannot accept data out of order. The SWB0, SWB1, SWB2 and SWB3 commands are used to write data into the buffer in ascending order. In the above example, the PCI master requests a data item which hits Qword 0 in the cache, thus CA[4:3] count through the following sequence: 0, 1, 2, 3 (00, 01, 10, 11). If the PCI mas-

ter requests a data item that hits Qword 1, the SWB0 command is sent via the HIG[4:0] lines to store Qword 1in the first buffer location. The next read from the cache is not in ascending order, thus a NOPC is sent on the HIG[4:0] lines. This Qword is not posted in the buffer. The next read from the cache is to Qword 3. SWB2 is sent on the HIG[4:0] lines. The final read from the cache is Qword 2. SWB1 is sent on the HIG[4:0] lines. Thus, Qword 1 is placed in entry 0 in the buffer, Qword 2 is placed in entry 1 in the buffer and Qword 3 is placed in entry 2 in the buffer. The ordering between the Qwords read from the cache and the HIG[4:0] commands when using burst SRAMs is summarized in Table 9.

Table 9. HIG[4:0] Command Sequence for Second Level Cache to PCI Master Read Prefetch Buffer Transfer

Burst Order from Cache	HIG[4:0] Command Sequence
0, 1, 2, 3	SWB0, SWB1, SWB2, SWB3
1, 0, 3, 2	SWB0, NOPC, SWB2, SWB1
2, 3, 0, 1	SWB0, SWB1, NOPC, NOPC
3, 2, 1, 0	SWB0, NOPC, NOPC, NOPC

When using standard asynchronous SRAMs, the read from the SRAMs occurs in a linear burst order. Thus, CAA[4:3] and CAB[4:3] count in a linear burst order and the Store Write Buffer commands are sent in linear order. The burst ends at the cache line boundary and does not wrap around and continue with the beginning of the cache line.

A PCI master write cycle which hits a modified line in the second level cache and either hits an unmodified line in the first level cache or misses in the first level cache will also cause a transfer from the second level cache to the LBXs. In this case, the read from the SRAMs is posted to main memory and the line is invalidated in the second level cache. The cycle would differ only slightly from the above cycle. INV would be asserted with EADS#. Instead of the DPRA command, the PCMC would use the DPWA command to drive the snoop address onto the host address bus. The write would be posted to the DRAM, thus the PCMC would drive the PCMWQ command on the HIG[4:0] lines to post the write to DRAM.

A snoop cycle can result in a write-back from the first level cache to both the second level and LBXs in the case of a PCI master read cycle which hits a modified line in the first level cache and hits either a modified or unmodified line in the second level cache. The line is written to both the second level cache and the memory to PCI read prefetch buffer. The cycle is shown in Figure 37.

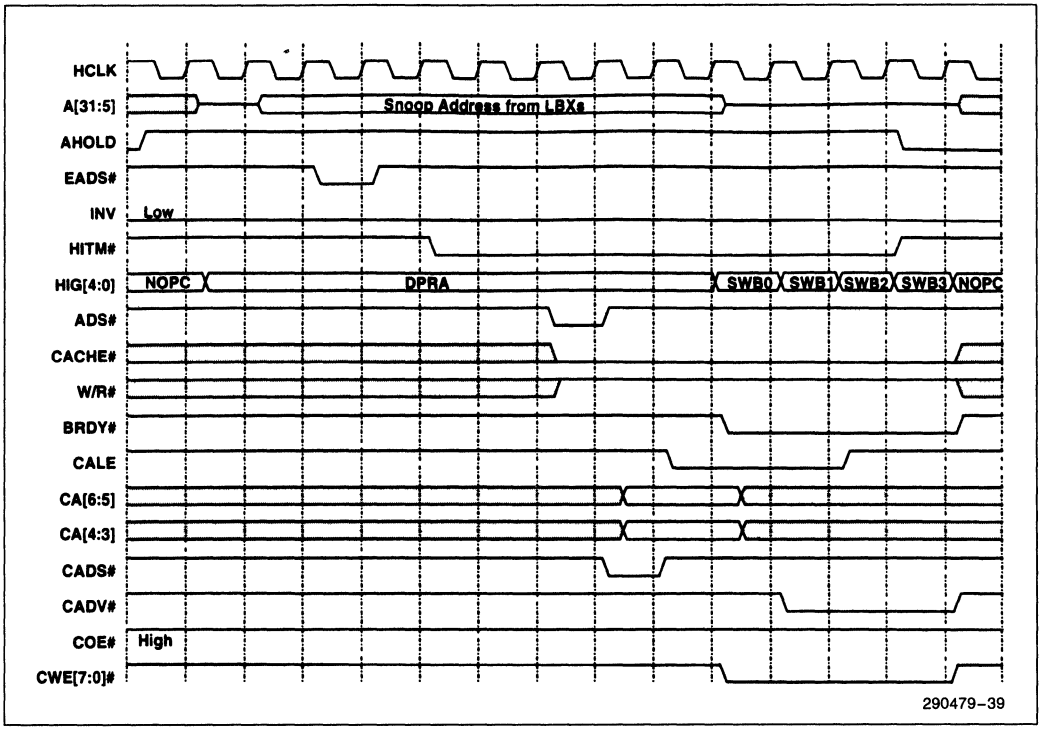


Figure 37. Snoop Hit to Modified Line in First Level Cache, Write-Back from First Level Cache to Second Level Cache and Send to PCI

This cycle is shown for the case of a second level cache with burst SRAMs. In this case, as it completes the second level cache tag lookup, the PCMC samples HITM# active. The write-back is written to the second level cache and simultaneously stored in the memory to PCI prefetch buffer. In the case shown in Figure 33, the PCI master requests a data item which is contained in Qword 0 of the cache line. Note that a write-back from the first level cache always starts with Qword 0 and finishes with Qword 3. Thus the HIG[4:0] lines are sequenced through the following order: SWB0, SWB1, SWB2, SWB3. If the PCI master requests a data item which is contained in Qword 1, the HIG[4:0] lines sequence through the following order: NOPC, SWB0, SWB1, SWB2. If the PCI master requests a data item which is contained in Qword 2, the HIG[4:0] lines sequence through the following order: NOPC, NOPC, SWB0, SWB1. If the PCI master requests a data item which is contained in Qword 3, the HIG[4:0] lines sequence through the following order: NOPC, NOPC, NOPC, SWB0. AHOLD is negated after the write-back cycle is complete.

If the CPU has begun a read cycle directed to PCI and the snoop results in a hit to a modified line in the first level cache, BOFF# is asserted for one clock to abort the CPU read cycle and re-order the write-back cycle before the pending read cycle.

5.1.5 FLUSH, FLUSH ACKNOWLEDGE AND WRITE-BACK SPECIAL CYCLES

There are three special cycles that affect the second level cache, flush, flush acknowledge, and write-back. If the processor executes an INVD instruction, it will invalidate all unmodified first level cache lines and issue a flush special cycle. If the processor executes a WBINVD instruction, it will write back all modified first level cache lines, invalidate the first level cache, and issue a write-back special cycle followed by a flush special cycle. If the Pentium processor FLUSH# pin is asserted, the CPU will write-back all modified first level cache lines, invalidate the first level cache, and issue a flush acknowledge special cycle.

The second level cache behaves the same way in response to the flush special cycle and flush acknowledge special cycle. Each tag is read and the valid and modified bits are examined. If the line is both valid and modified it is written back to main memory and the valid bit for that line is reset. All valid and unmodified lines are simply marked invalid. The PCMC advances to the next tag when all lines within the current sector have been examined. BRDY# is returned to the Pentium processor after all modified lines in the second level cache have been written back to main memory and all of the valid bits for the second level cache are reset. The sequence of write-back cycles will only be interrupted to service a PCI master cycle.

The write-back special cycle is ignored by the PCMC because all modified lines will be written back to main memory by the following flush special cycle. Upon decoding a write-back special cycle, the PCMC simply returns BRDY# to the Pentium processor.

5.2 82434NX Cache

The 82434NX PCMC integrates a high performance write-back second level cache controller, tag RAM and a full first and second level cache coherency mechanism. The cache is either 256 KBytes or 512 KBytes using either synchronous burst SRAMs or standard asynchronous SRAMs. Parity on the data SRAMs is optional. The cache uses a write-back write policy. Write-through mode is not supported.

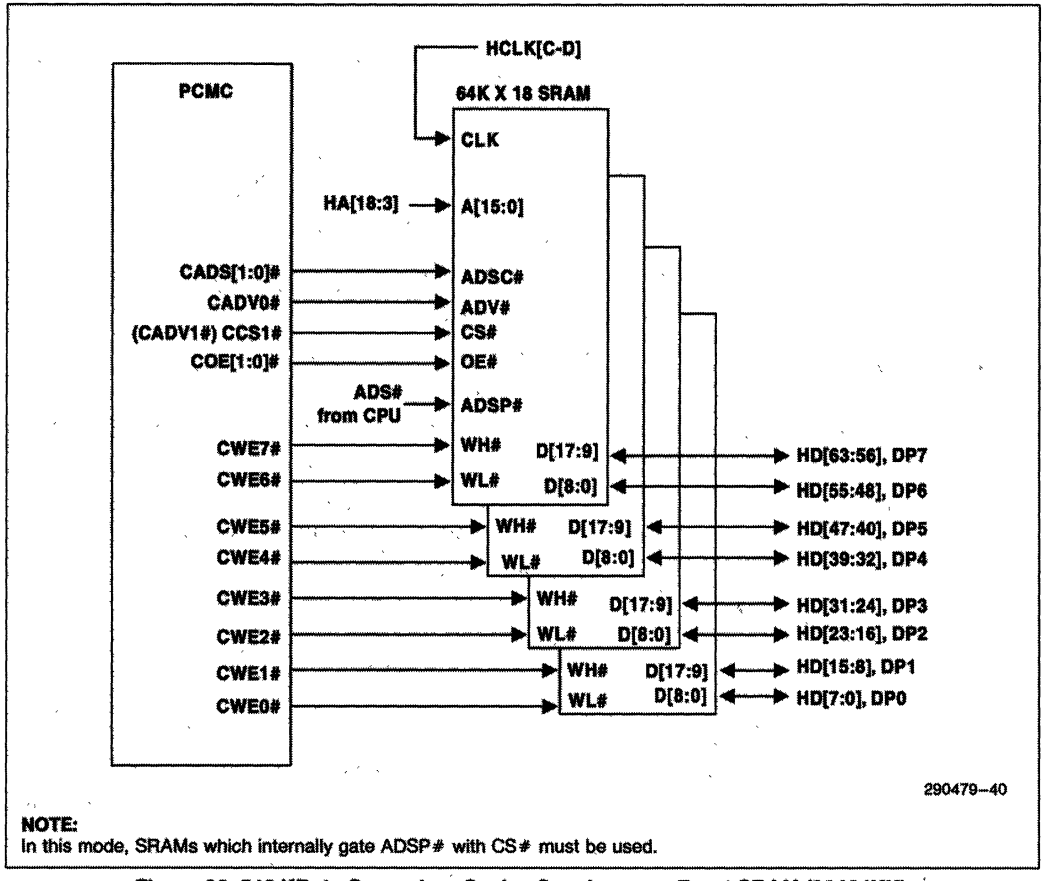
The 82434NX PCMC supports a direct mapped secondary cache. The PCMC contains 4096 tags. Each tag represents a sector in the cache. If the cache is 512 KB, each sector contains four cache lines. If the cache is 256 KB, each sector contains two cache lines. *Valid* and *Modified* bits are kept on a per line basis. The 82434NX Tag RAM is 1 bit wider than the 82434LX Tag RAM.

The PCMC can be configured to cache main memory on read cycles even when CACHE# is not asserted. When bit 4 in the Secondary Cache Control Register (offset 52h) is set to 1, all accesses to main memory, except those to SMM memory or any range marked non-cacheable via the PAM registers, are cached in the secondary cache. Accesses with CACHE# asserted result in a line fill in both the first and second level cache while accesses with CACHE# negated result in a line fill only in the second level cache. When bit 4 in the SCC Register is set to 0, only access with CACHE# asserted can generate a first and second level cache line fill.

When a Halt or Stop Grant Special Cycle is detected from the CPU, the 82434NX PCMC places the second level cache into the low power stand-by mode by deselecting the SRAMs and then generates the corresponding special cycle on PCI. (i.e., if the CPU cycle was a halt special cycle then the PCMC generates a halt special cycle on PCI and if the CPU cycle is a stop grant special cycle the PCMC generates a stop grant special cycle on PCI).

When a burst SRAM secondary cache is implemented, bit 2 of the Secondary Cache Control Register (offset 52h) is used to select between 82434LX SRAM connectivity and the new 82434NX SRAM connectivity. When set to 0, the secondary cache interface is in 82430-compatible mode. (i.e., the four low order address lines on the SRAMs are connected to CAA/B[6:3] on the PCMC. When set to 1, second level cache stand-by is enabled and no latch is used between the host CPU address lines and the SRAM address lines. All of the SRAM address lines are then connected directly to the CPU address lines. Write-back addresses are driven by the PCMC over the host address lines. When a standard SRAM secondary cache is implemented, bit 2 of the Secondary Cache Control Register (offset 52h) is used to enable second level cache stand-by. The default value of this bit is 0.

Figure 38 and Figure 41 show the connections between the PCMC and the external cache data SRAMs and latch for the case of an asynchronous SRAM cache.



NOTE:
In this mode, SRAMs which internally gate ADSP# with CS# must be used.

Figure 38. 512 KByte Secondary Cache, Synchronous Burst SRAM (82434NX)

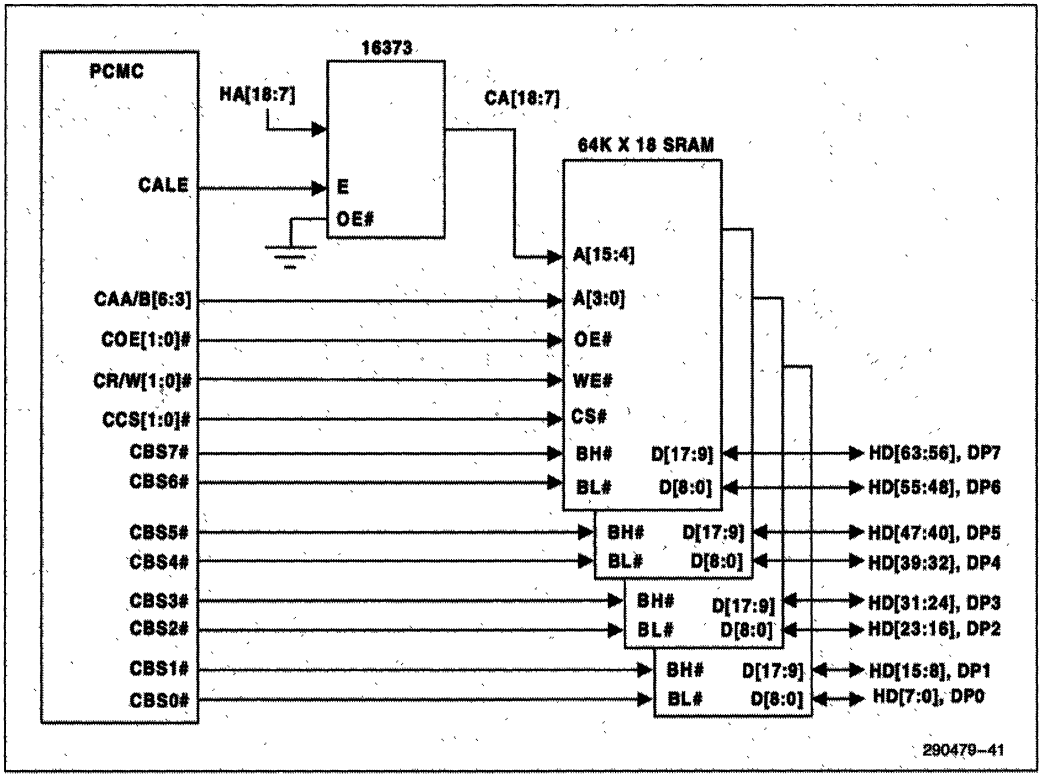


Figure 39. 512 KByte Secondary Cache, Standard Dual-Byte-Select (Asynch) SRAM, 50, 60 & 66 MHz

Figure 38 depicts the PCMC connections to a 512 KByte burst SRAM secondary cache when the PCMC is configured for 50, 60, or 66 MHz operation. Host address lines HA[18:3] are connected directly to the SRAM address lines, A[15:0]. ADS# from the CPU is connected to ADSP# on the SRAMs. CADV0# implements the address advance (ADV#) functionality. A new signal, CCS#, is multiplexed onto the CADV1# pin. When bit 2 in the SCC register is set to 1, SRAMs containing logic which gates ADSP# with CS# must be used. When negated, CCS# prevents the SRAMs from latching a new address due to a pipelined ADS# from the CPU during cache line fills. Note that, unlike the burst SRAM configuration with the 82430 PCIset, no external latch is used between the CPU address bus and the SRAM address lines. The SRAM Connectivity bit (bit 2) in the Secondary Cache Control register (offset 52h) must be set to 1 when using this cache configuration.

If the tag lookup results in a miss in the cache and the sector to be replaced contains one or more modified lines, the PCMC drives the write-back address from the A[18:3] lines on the host bus. Although not used in the write-back, A[31:19] (or A[31:18] in the case of a 256 KB cache) are driven to valid logic levels by the PCMC.

Figure 39 depicts the 82434NX PCMC connections to a 512 KByte standard asynchronous SRAM secondary cache. Figure 40 depicts the 82434NX connections to a 256 KByte asynchronous SRAM secondary cache. Host address lines HA[18:7] are driven through an external latch to form the upper SRAM address lines, CA[18:7]. CA[6:3] are driven from the PCMC. Figure 41 depicts the 82434NX PCMC connections to a 512 KByte standard SRAM secondary cache with dual-write-enable SRAMs.

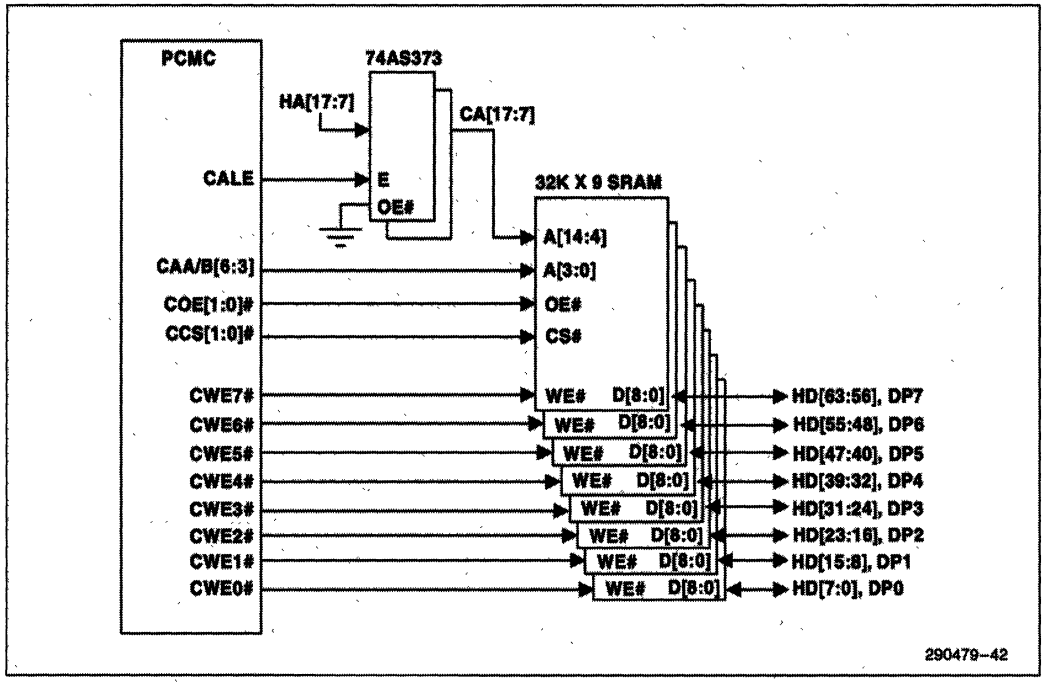


Figure 40. 82434NX Connections to 256 KByte Cache with Standard SRAM

2

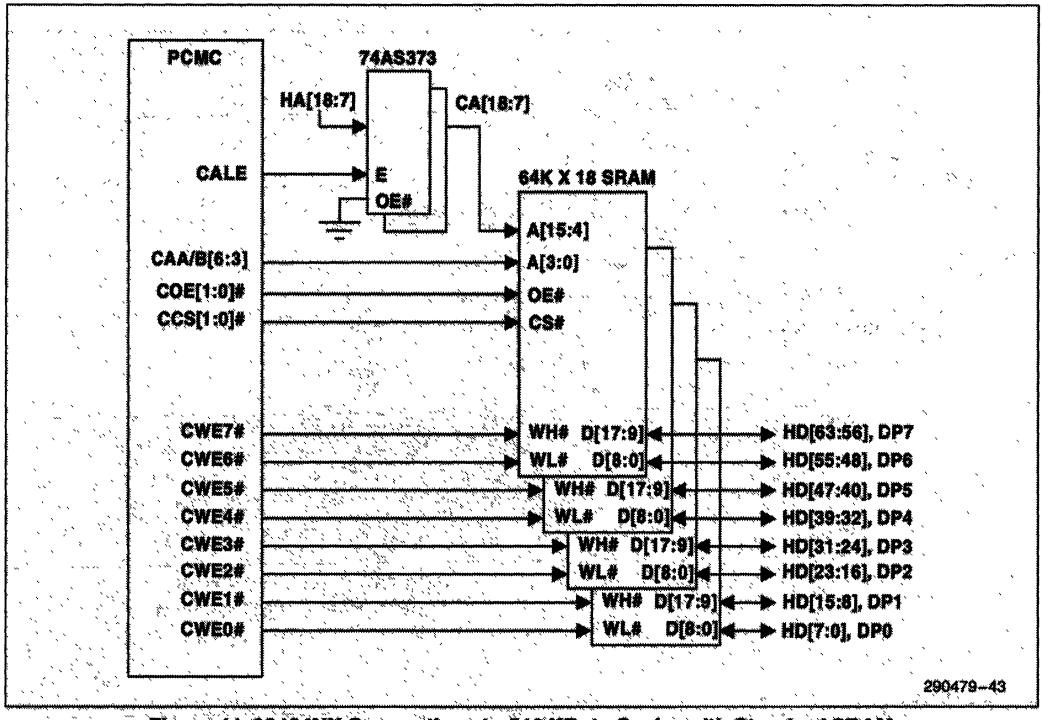


Figure 41. 82434NX Connections to 512 KByte Cache with Standard SRAM

5.2.1 CYCLE LATENCY SUMMARY (82434NX)

Table 10 and Table 11 summarize the clock latencies for CPU memory cycles which hit in the secondary cache.

Table 10. Secondary Cache Latencies with Synchronous Burst SRAM

Cycle Type	50, 60 and 66 MHz
Burst Read	3-1-1-1
Burst Write	3-1-1-1
Single Read	3
Single Write	3
Pipelined Back-to-Back Burst Reads	3-1-1-1-1-1-1-1
Burst Read Followed by Pipelined Write	3-1-1-1-2

Table 11. Secondary Cache Latencies with Standard Asynchronous SRAM (82434NX)

Cycle Type	50, 60 and 66 MHz
Burst Read	3-2-2-2
Burst Write	4-2-2-2
Single Read	3
Single Write	4
Pipelined Back-to-Back Burst Reads	3-2-2-2-3-2-2-2
Burst Read Followed by Pipelined Write	3-2-2-2-4

The 60 MHz and 66 MHz asynchronous SRAM latencies require 15 ns and 12 ns SRAMs, respectively. The 82434NX PCMC supports asynchronous SRAMs at 50 MHz. The 50 MHz (1 wait-state) timings require 20 ns SRAMs. The burst SRAMs speeds for 66 MHz, 60 MHz and 50 MHz operation are 8 ns, 9 ns, and 13 ns clock-to-output valid into a 0 pF test load. The SRAM access times listed in this paragraph are recommendations. Actual access time requirements are a function of system board layout and routing and should be validated with electrical simulation.

5.2.2 STANDARD SRAM CACHE CYCLES (82434NX)

At 50, 60 and 66 MHz, the timing of the second level cache interface with standard asynchronous SRAMs is identical to the timing in the 82430LX PCIset. Compared to the 82434LX second level cache, one additional connection can be made from the PCMC to the SRAMs. The CCS[1:0]# pins, in the case of asynchronous SRAMs, are multiplexed onto the CADV[1:0]# pins. These are then connected to the SRAM CS# pins. The two copies are functionally identical. The two copies are provided for timing reasons. These pins allow the PCMC to deselect the SRAMs, putting them into standby mode. When a halt special cycle or a stop grant special cycle is detected from the CPU, the PCMC negates CCS[1:0]#, placing the SRAMs into the low power standby mode. The PCMC then generates a halt or stop grant special cycle on PCI.

5.2.3 SECOND LEVEL CACHE STANDBY

When the PCMC detects a halt or stop grant special cycle from the CPU, it first places the second level cache into the low power stand-by mode by deselecting the SRAMs and then generates a halt or stop grant special cycle on PCI.

With a standard SRAM secondary cache, a halt or stop grant special cycle from the CPU causes the PCMC to negate CCS[1:0]#, deselecting the SRAMs and placing them in a low power standby mode. When the cache is in stand-by mode, the first bus cycle from the CPU brings the cache out of stand-by and into active mode, enabling the SRAMs to service the cycle in the case of a hit to the cache. The PCMC asserts CCS[1:0]# as a propagation delay from the falling edge of ADS#. CCS[1:0]# are then left asserted until the next halt or stop grant special cycle is occurs. When exiting the powerdown state, the PCMC ignores the Secondary Cache Leadoff wait-states bit and executes a 3-2-2-2 read or 4-2-2-2 write in order to allow the SRAMs time to power up. In the case of a read cycle, COE[1:0]# are asserted in clock two as in the case of ordinary read cycles.

When the SRAMs are powered down, the PCMC asserts CCS[1:0]# when performing a snoop cycle, regardless of whether the cycle hits in the second level cache. The PCMC then negates CCS# after the snoop cycle is complete.

With a burst SRAM secondary cache, a halt or stop grant special cycle from the CPU causes the PCMC to negate CCS# and assert CADS[1:0]#, deselecting the SRAMs, placing them in a low power standby mode. CCS# is then asserted and is left asserted by the PCMC. Thus, when the first cycle is driven from the CPU, the SRAMs sample ADSP# and CS# active, placing them in active mode and initiating the first access.

If the SRAMs are required to service a snoop, they are brought out of power-down when the PCMC asserts CADS[1:0]#. The PCMC always asserts CADS[1:0]# with CCS# negated after a snoop cycle is complete, regardless of whether the SRAMs were powered down prior to the snoop cycle.

5.2.4 SNOOP CYCLES

For snoop operations, refer to Section 5.1, 82434LX Cache.

5.2.5 FLUSH, FLUSH ACKNOWLEDGE, AND WRITE-BACK SPECIAL CYCLES

For flush, flush acknowledge, and write-back special cycles, refer to Section 5.1, 82434LX Cache.

6.0 DRAM INTERFACE

This section describes the DRAM interface for the 82434LX DRAM Interface (Section 6.1) and the 82434NX DRAM Interface (Section 6.2). The differences are in the following areas:

1. Increased maximum DRAM memory size to 512 MBytes. An extra address line (MA11) has been added to the 82434NX.
2. Two additional RAS# lines for a total of eight (RAS[0:7]#.
3. Addition of 50 MHz host-bus optimized DRAM timing sets. Thus, the 82434LX supports 60 and 66 MHz frequencies and the 82434NX supports 50, 60, and 66 MHz.

6.1 82434LX DRAM Interface

The 82434LX PCMC integrates a high performance DRAM controller supporting from 2–192 MBytes of main memory. The PCMC generates the RAS#, CAS#, WE# and multiplexed addresses for the DRAM array, while the data path to DRAM is provided by two 82433LX LBXs. The DRAM controller interface is fully configurable through a set of control registers. Complete descriptions of these registers are given in Section 3.0, Register Description. A brief overview of the registers which configure the DRAM interface is provided in this section.

The 82434LX controls a 64-bit memory array (72-bit including parity) ranging in size from 2 MBytes up to 192 MBytes using industry standard 36-bit wide memory modules with fast page-mode DRAMs. Both single- and double-sided SIMMs are supported. The eleven multiplexed address lines, MA[10:0] allow the PCMC to support 256K x 36, 1M x 36, and 4M x 36 SIMMs. The PCMC has six RAS# lines enabling the support of up to six rows of DRAM. Eight CAS# lines allow byte control over the array during read and write operations. The PCMC supports 70 and 60 ns DRAMs. The PCMC DRAM interface is synchronous to the CPU clock and supports page mode accesses to efficiently transfer data in bursts of four Qwords.

The DRAM interface of the PCMC is configured by the DRAM Control Mode Register (offset 57h) and the six DRAM Row Boundary (DRB) Registers (offsets 60h–65h). The DRAM Control Mode Register contains bits to configure the DRAM interface for RAS# modes and refresh options. In addition, DRAM Parity Error Reporting and System Management RAM space can be enabled and disabled. When System Management RAM is enabled, if SMIACT# from the Pentium processor is not asserted, all CPU read and write accesses to SMM memory are directed to PCI. The SMRAM Space Register at configuration space offset 72h provides additional control over the SMRAM space. The six DRB Registers define the size of each row in the memory array, enabling the PCMC to assert the proper RAS# line for accesses to the array.

CPU-to-Memory write posting and read-around-write operations are enabled and disabled via the Host Read/Write Buffer Control Register (offset 53h). PCI-to-Memory write posting is enabled and disabled via the PCI Read/Write Buffer Control Register (offset 54h). PCI master reads from main memory always result in the PCMC and LBXs reading the requested data and prefetching the next seven Dwords.

Seven Programmable Attribute Map (PAM) Registers (offsets 59h–5Fh) are used to specify the cacheability and read/write status of the memory space between 512 KBytes and 1 MByte. Each PAM Register defines a specific address area enabling the system to selectively mark specific memory ranges as cacheable, read-only, write-only, read/write or disabled. When a memory range is disabled, all CPU accesses to that range are directed to PCI.

Two other registers also affect the DRAM interface, the Memory Space Gap Register (offsets 78h–79h) and the Frame Buffer Range Register (offsets 7Ch–7Fh). The Memory Space Gap Register is used to place a logical hole in the memory space between 1 MByte to 16 MBytes to accommodate memory mapped ISA boards. The Frame Buffer Range Register, is used to map a linear frame buffer into the Memory Space Gap or above main memory. When enabled, accesses to these ranges are never directed to the DRAM interface, but are always directed to PCI.

6.1.1 DRAM CONFIGURATIONS

Figure 42 illustrates a 12-SIMM configuration which supports single-sided SIMMs. A row in the DRAM array is made up of two SIMMs which share a common RAS# line. SIMM0 and SIMM1 are connected to RAS0# and therefore, comprise row 0. SIMM10 and SIMM11 form row 5. Within any given row, the two SIMMs must be the same size. Among the six rows, SIMM densities can be mixed in any order. That is, there are no restrictions on the ordering of SIMM densities among the six rows.

The low order LBX (LBXL) is connected to byte lanes 5, 4, 1, and 0 of the host and memory data buses, and the lower two bytes of the PCI AD bus. The high order LBX (LBXH) is connected to byte lanes 7, 6, 3, and 2 of the host and memory data buses, and the upper two bytes of the PCI AD bus. Thus, SIMMs connected to LBXL are connected to CAS[5:4,1:0]# and SIMMs connected to LBXH are connected to CAS[7:6, 3:2]#.

The MA[10:0] and WE# lines are externally buffered to drive the large capacitance of the memory array. Three buffered copies of the MA[10:0] and WE# signals are required to drive the six row array.

Figure 43 illustrates a 6-SIMM configuration that supports either single- or double-sided SIMMs. In this configuration, single- and double-sided SIMMs can be mixed. For example, if single-sided SIMMs are installed into the sockets marked SIMM0 and SIMM1, then RAS0# is connected to the SIMMs and RAS1# is not connected. Row 0 is then populated and row 1 is empty. Two double-sided SIMMs could then be installed in the sockets marked SIMM2 and SIMM3, populating rows 2 and 3.

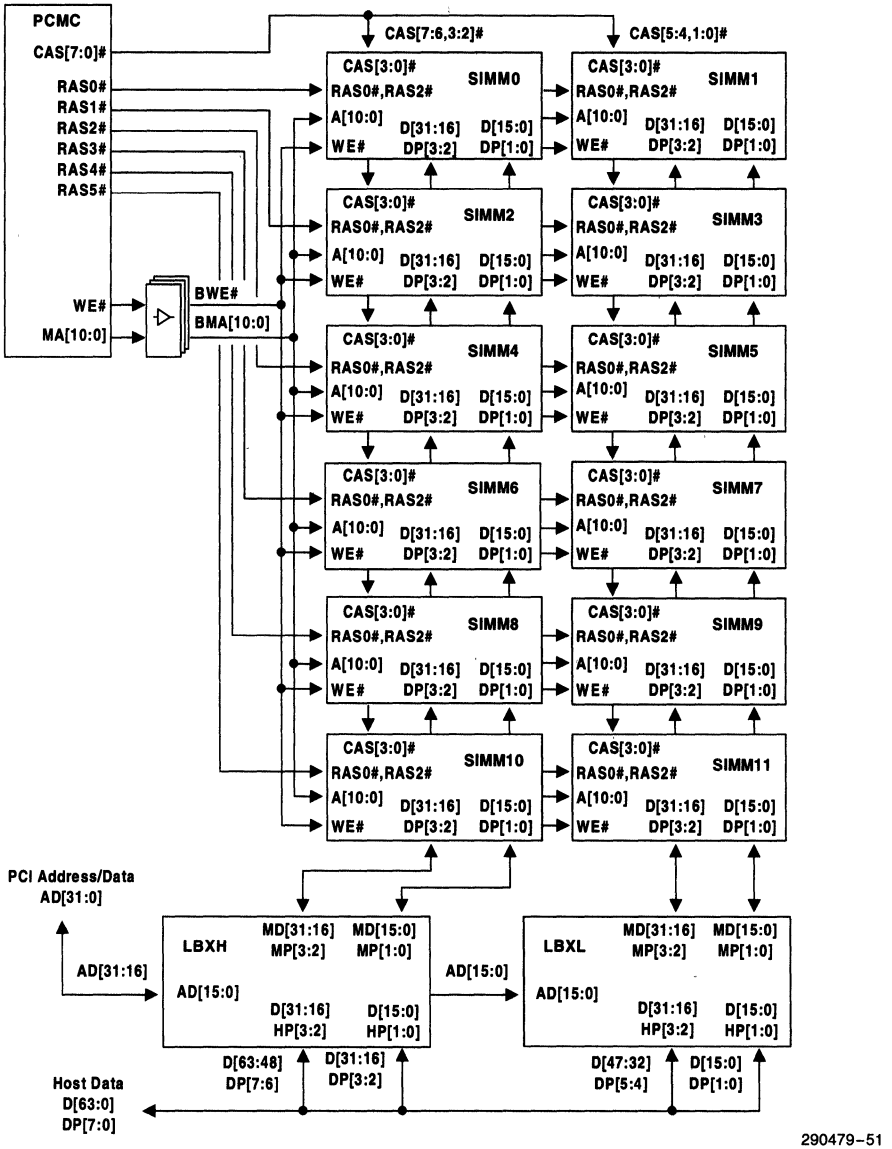
6.1.2 DRAM ADDRESS TRANSLATION

The 82434LX multiplexed row/column address to the DRAM memory array is provided by the MA[10:0] signals. The MA[10:0] bits are derived from the host address bus as defined by Table 12.

MA[10:0] are translated from the host address A[24:3] for all memory accesses, except those targeted to memory that has been remapped as a result of the creation of a memory space gap in the lower extended memory area. In the case of a cycle targeting remapped memory, the least significant bits come directly from the host address, while the more significant bits depend on the memory space gap start address, gap size, and the size of main memory.

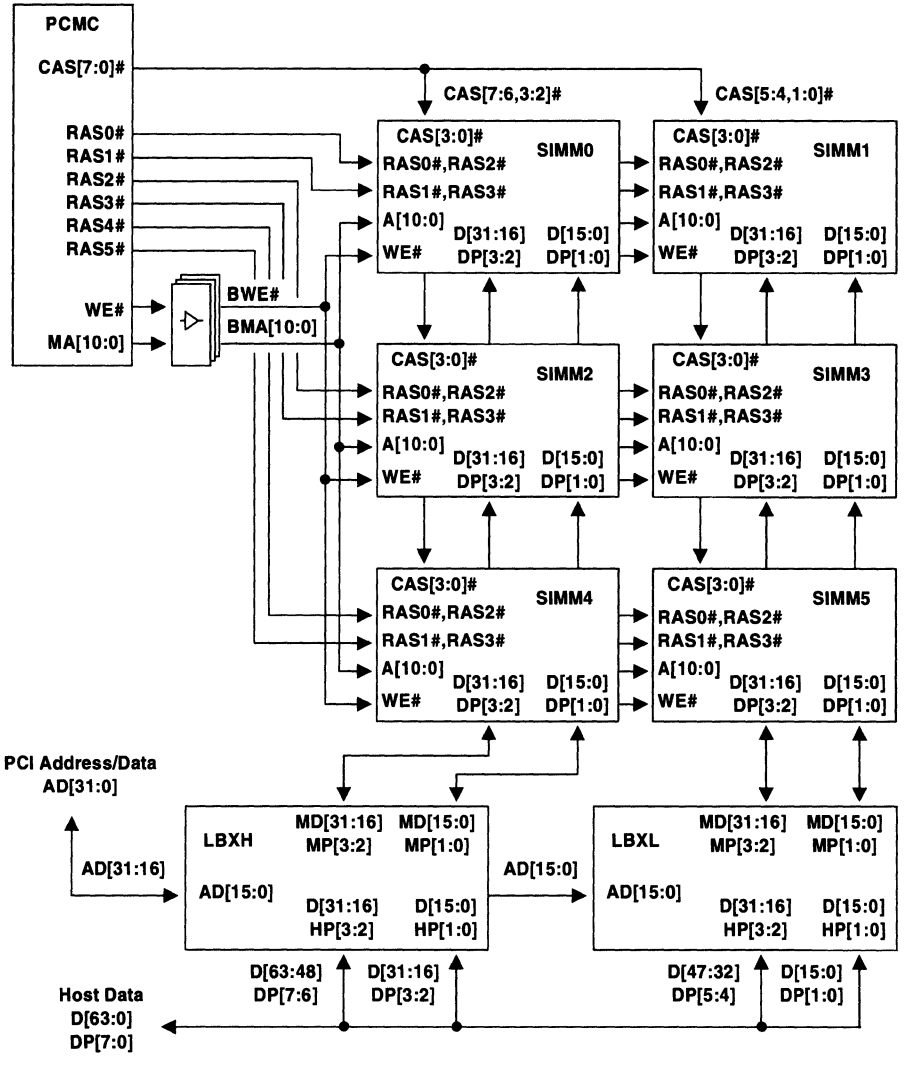
Table 12. DRAM Address Translation

Memory Address, MA[10:0]	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Row Address	A24	A22	A20	A19	A18	A17	A16	A15	A14	A13	A12
Column Address	A23	A21	A11	A10	A9	A8	A7	A6	A5	A4	A3



NOTE:
The figure shows the connections for the 82434LX. For the 82434NX, there are two additional RAS lines (RAS[7:6]#) and one additional address line (MA11).

Figure 42. 82434LX DRAM Configuration Supporting Single-Sided SIMMs



NOTE:
The figure shows the connections for the 82434LX. For the 82434NX, there are two additional RAS lines (RAS[7:6]#) and one additional address line (MA11).

Figure 43. 82434LX DRAM Configuration Supporting Single- or Double-Sided SIMMs

6.1.3 CYCLE TIMING SUMMARY

The 82434LX PCMC DRAM performance is summarized in Table 13 for all CPU read and write cycles.

Table 13. CPU to DRAM Performance Summary

Cycle Type	Burst, x-4-4-4 Timing	Single, x-4-4-4 Timing
Read Page Hit	7-4-4-4	7
Read Row Miss	11-4-4-4	11
Read Page Miss	14-4-4-4	14
Posted Write, WT L2	3-1-1-1	3
Posted Write, WB L2	4-1-1-1	4
Write Page Hit	12-4-4-4	12
Write Row Miss	13-4-4-4	13
Write Page Miss	16-4-4-4	16
0-Active RAS# Mode Read	10-4-4-4	10
0-Active RAS# Mode Write	12-4-4-4	12

CPU writes to the CPU-to-Memory Posted Write Buffer are completed at 3-1-1-1 when the second level cache is configured for write-through mode and 4-1-1-1 when the cache is configured for write-back mode. Table 14 shows the refresh performance in CPU clocks.

Table 14. Refresh Cycle Performance

Refresh Type	Hidden Refresh	RAS# only Refresh	CAS# before RAS#
Single	12	13	14
Burst of Four	48	52	56

6.1.4 CPU TO DRAM BUS CYCLES

This section describes the CPU-to-DRAM cycles for the 82434LX.

6.1.4.1 Read Page Hit

Figure 44 depicts a CPU burst read page hit from DRAM. The 82434LX PCMC decodes the CPU address as a page hit and drives the column address onto the MA[10:0] lines. CAS[7:0]# are then asserted to cause the DRAMs to latch the column address and begin the read cycle. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read page hit from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted.

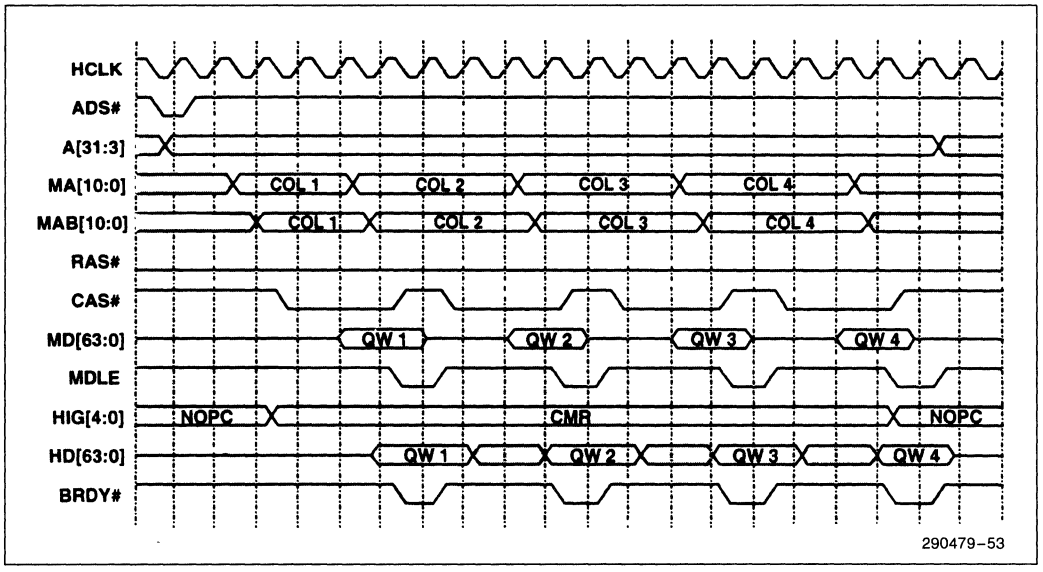


Figure 44. Burst DRAM Read Cycle-Page Hit

6.1.4.2 Read Page Miss

Figure 45 depicts a CPU burst read page miss from DRAM. The 82434LX decodes the CPU address as a page miss and switches from initially driving the column address to driving the row address on the MA[10:0] lines. RAS# is then negated to precharge the DRAMs and then asserted to cause the DRAMs to latch the new row address. The PCMC then switches the MA[10:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read page miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted.

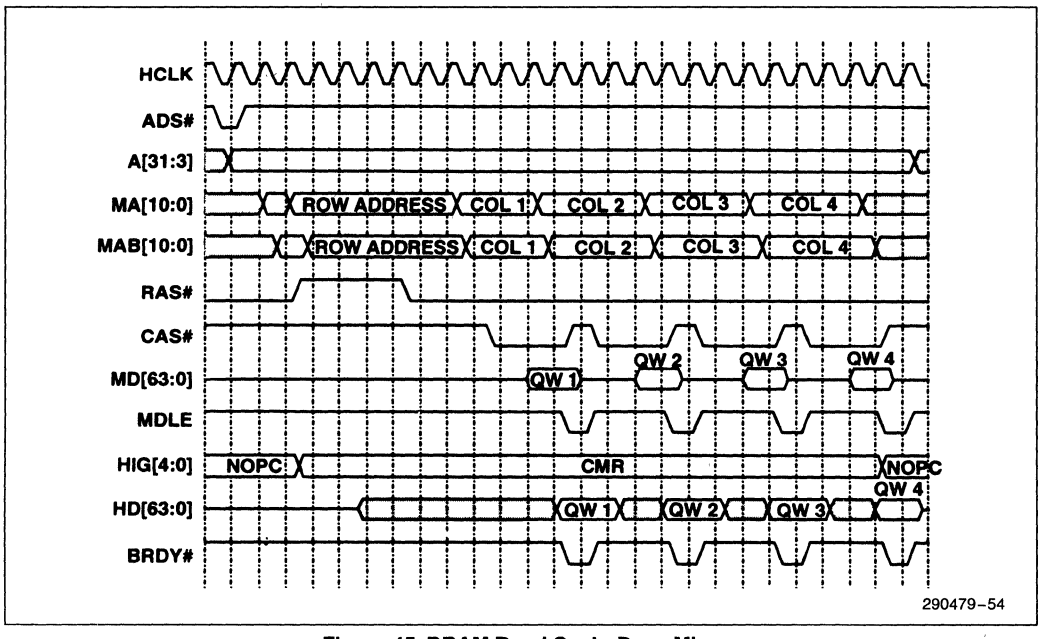


Figure 45. DRAM Read Cycle-Page Miss

6.1.4.3 Read Row Miss

Figure 46 depicts a CPU burst read row miss from DRAM. The 82434LX decodes the CPU address as a row miss and switches from initially driving the column address to driving the row address on the MA[10:0] lines. The RAS# signal that was asserted is negated and the RAS# for the currently accessed row is asserted. The PCMC then switches the MA[10:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read row miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted.

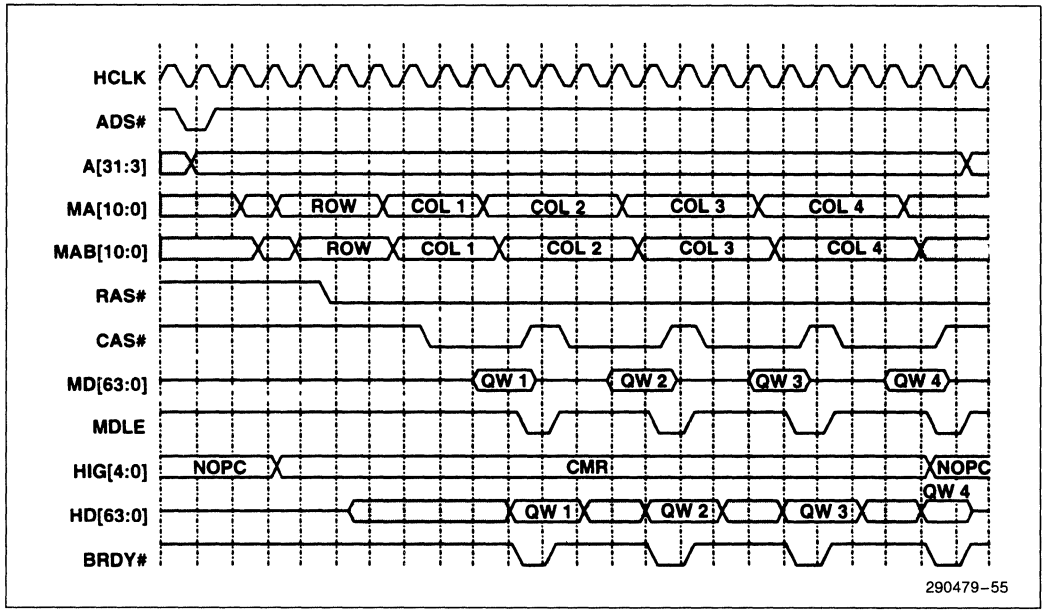


Figure 46. Burst DRAM Read Cycle-Row Miss

6.1.4.4 Write Page Hit

Figure 47 depicts a CPU burst write page hit from DRAM. The 82434LX decodes the CPU write cycle as a DRAM page hit. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When the cycle is decoded as a page hit, the PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword for three more clocks. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0]# are asserted.

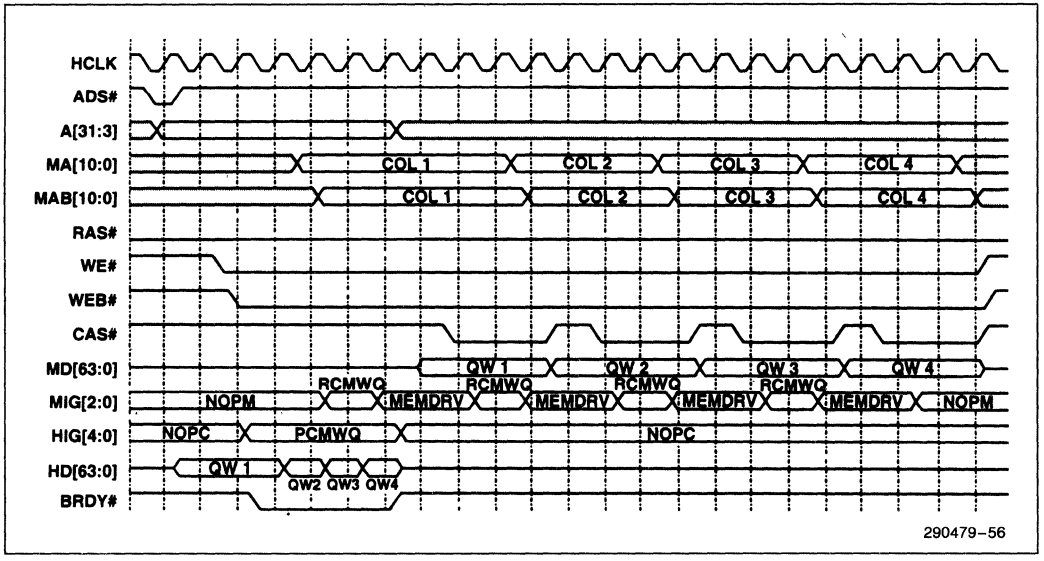


Figure 47. Burst DRAM Write Cycle-Page Hit

6.1.4.5 Write Page Miss

Figure 48 depicts a CPU burst write page miss to DRAM. The 82434LX decodes the CPU write cycle as a DRAM page miss. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When the cycle is decoded as a page miss, the PCMC switches the MA[10:0] lines from the column address to the row address and asserts WE#. The PCMC drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The RAS# signal for the currently decoded row is negated to precharge the DRAMs. RAS# is then asserted to cause the DRAMs to latch the row address. The PCMC then switches the MA[10:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0]# are asserted.

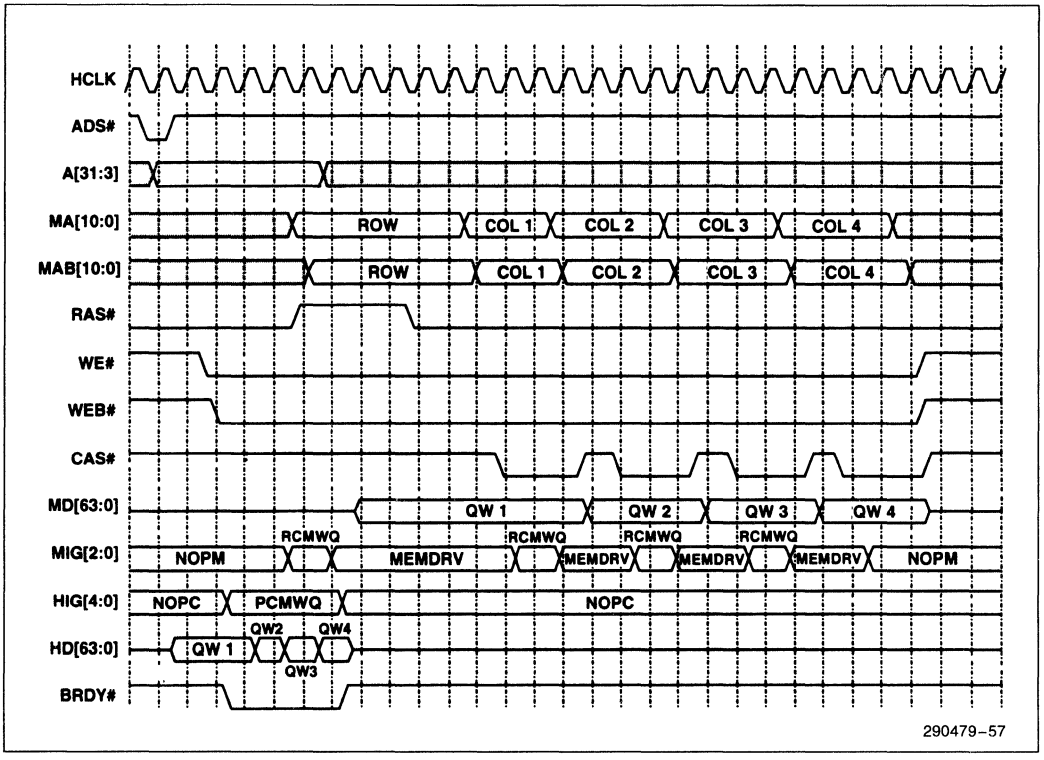


Figure 48. Burst DRAM Write Cycle-Page Miss

6.1.4.6 Write Row Miss

Figure 49 depicts a CPU burst write row miss to DRAM. The 82434LX decodes the CPU write cycle as a DRAM row miss. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When the cycle is decoded as a row miss, the PCMC negates the already active RAS# signal, switches the MA[10:0] lines from the column address to the row address and asserts the RAS# signal for the currently decoded row. The PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The PCMC then switches the MA[10:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0]# are asserted.

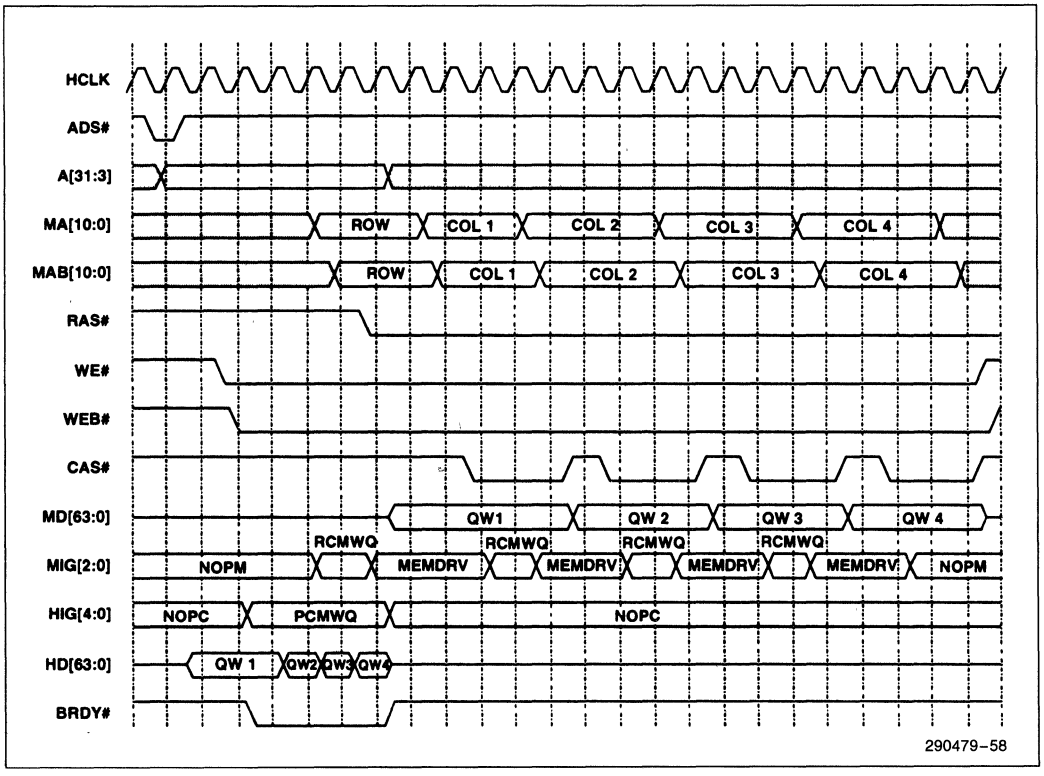


Figure 49. Burst DRAM Write Cycle-Row Miss

6.1.4.7 Read Cycle, 0-Active RAS# Mode

When in 0-active RAS# mode, every CPU cycle to DRAM results in a RAS# and CAS# sequence. RAS# is always negated after a cycle completes. Figure 50 depicts a CPU burst read cycle from DRAM where the 82434LX is configured for 0-active RAS# mode. When in 0-active RAS# mode, the PCMC defaults to driving the row address on the MA[10:0] lines. The PCMC asserts the RAS# signal for the currently decoded row causing the DRAMs to latch the row address. The PCMC then switches the MA[10:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the Pentium processor burst order, negating and asserting CAS[7:0]# to read each Qword. The host data is latched on the falling edge of MDLE, when CAS[7:0]# are negated. The latch is opened again when MDLE is sampled asserted by the LBXs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read row miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when BRDY# is asserted. RAS# is negated with CAS[7:0]#.

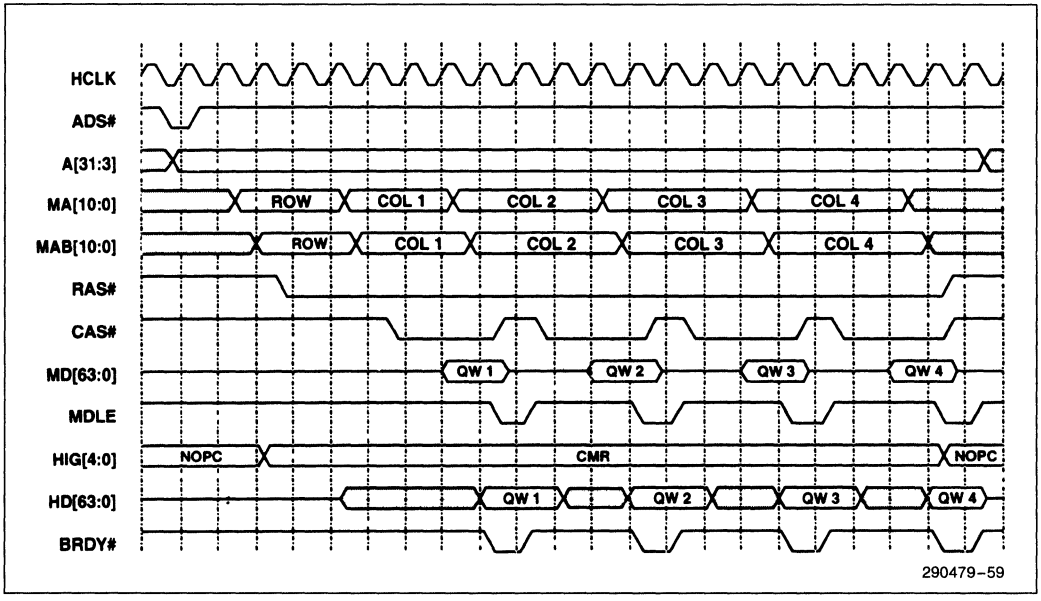


Figure 50. Burst DRAM Read Cycle, 0-Active RAS# Mode

6.1.4.8 Write Cycle, 0-Active RAS# Mode

When in 0-active RAS# mode, every CPU cycle to DRAM results in a RAS# and CAS# sequence. RAS# is always negated after a cycle completes. Figure 51 depicts a CPU Burst Write Cycle to DRAM where the 82434LX is configured for 0-active RAS# mode. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 4-1-1-1. The write is posted at 4-1-1-1 when the second level cache is configured for a write-back policy. The write is posted to DRAM at 3-1-1-1 when the second level cache is configured for a write-through policy. When in 0-active RAS# mode, the PCMC defaults to driving the row address on the MA[10:0] lines. The PCMC asserts the RAS# signal for the currently decoded row causing the DRAMs to latch the row address. The PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The PCMC then switches the MA[10:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. MIG[2:0] are driven to NOPM in the clock after CAS[7:0] are asserted.

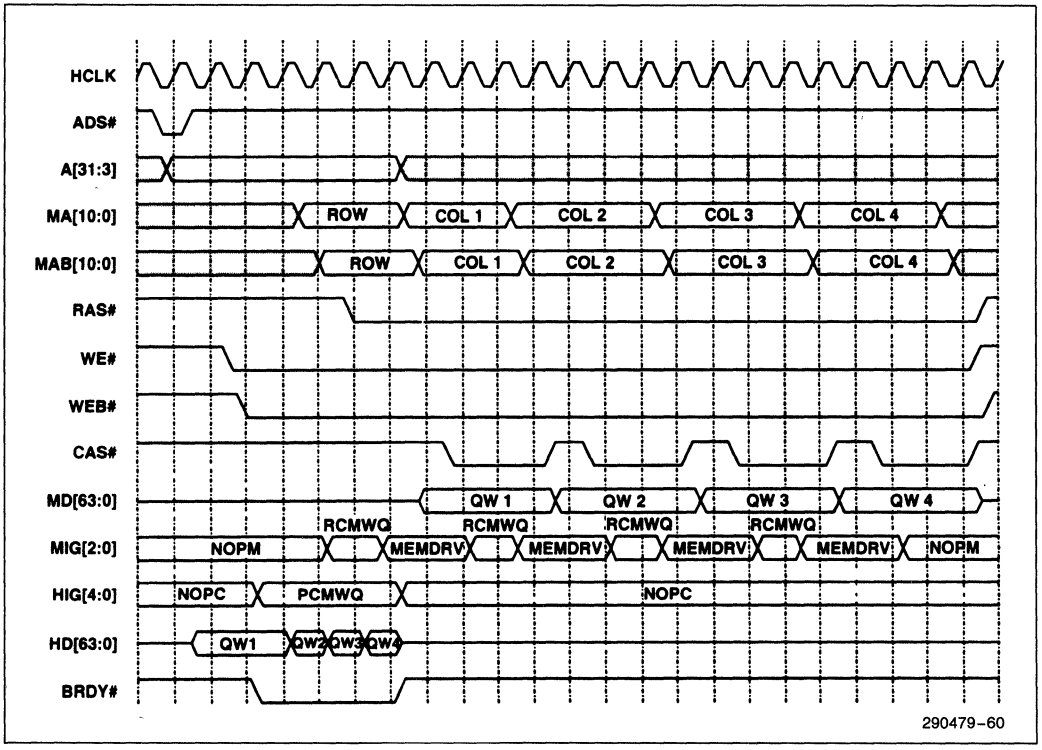


Figure 51. Burst DRAM Write Cycle, 0-Active RAS# Mode

6.1.5 REFRESH

The refresh of the DRAM array can be performed by either using RAS#-only or CAS#-before-RAS# refresh cycles. When programmed for CAS#-before-RAS# refresh, hidden refresh cycles are initiated when possible. RAS# only refresh can be used with any type of second level cache configuration (i.e., no second level cache is present, or either a burst SRAM or standard SRAM second level cache is implemented). CAS#-before-RAS# refresh can be enabled when either no second level cache is present or a burst SRAM second level cache is implemented. CAS#-before-RAS# refresh should not be used when a standard SRAM second level cache is implemented. The timing of internally generated refresh cycles is derived from HCLK and is independent of any expansion bus refresh cycles.

The DRAM controller contains an internal refresh timer which periodically requests the refresh control logic to perform either a single refresh or a burst of four refreshes. The single refresh interval is 15.6 μs. The interval for burst of four refreshes is four times the single refresh interval, or 62.4 μs. The PCMC is configured for either single or burst of four refresh and either RAS#-only or CAS#-before-RAS# refresh via the DRAM Control Register (offset 57h).

To minimize performance impact, refresh cycles are partially deferred until the DRAM interface is idle. The deferment of refresh cycles is limited by the DRAM maximum RAS# low time of 100 μs. Refresh cycles are initiated such that the RAS# maximum low time is never violated.

Hidden refresh cycles are run whenever all eight CAS# lines are active when the refresh cycle is internally requested. Normal CAS#-before-RAS# refresh cycles are run whenever the DRAM interface is idle when the refresh is requested, or when any subset of the CAS# lines is inactive as the refresh is internally requested.

To minimize the power surge associated with refreshing a large DRAM array the DRAM interface staggers the assertion of the RAS# signals during both CAS#-before-RAS# and RAS#-only refresh cycles. The order of RAS# edges is dependent on which RAS# was most recently asserted prior to the refresh sequence. The RAS# that was active will be the last to be activated during the refresh sequence. All RAS[5:0]# lines are negated at the end of refresh cycles, thus, the first DRAM cycle after a refresh sequence is a row miss.

2

6.1.5.1 RAS#-Only Refresh-Single

Figure 52 depicts a RAS#-only refresh cycle when the 82434LX is programmed for single refresh cycles. The diagram shows a CPU read cycle completing as the refresh timing inside the PCMC generates a refresh request. The refresh address is driven on the MA[10:0] lines. Since the CPU cycle was to row 0, RAS0# is negated. RAS1# is the first to be asserted. RAS2# through RAS5# are then asserted sequentially while RAS0# is driven high, precharging the DRAMs in row 0. RAS0# is then asserted after RAS5#. Each RAS# line is asserted for six host clocks.

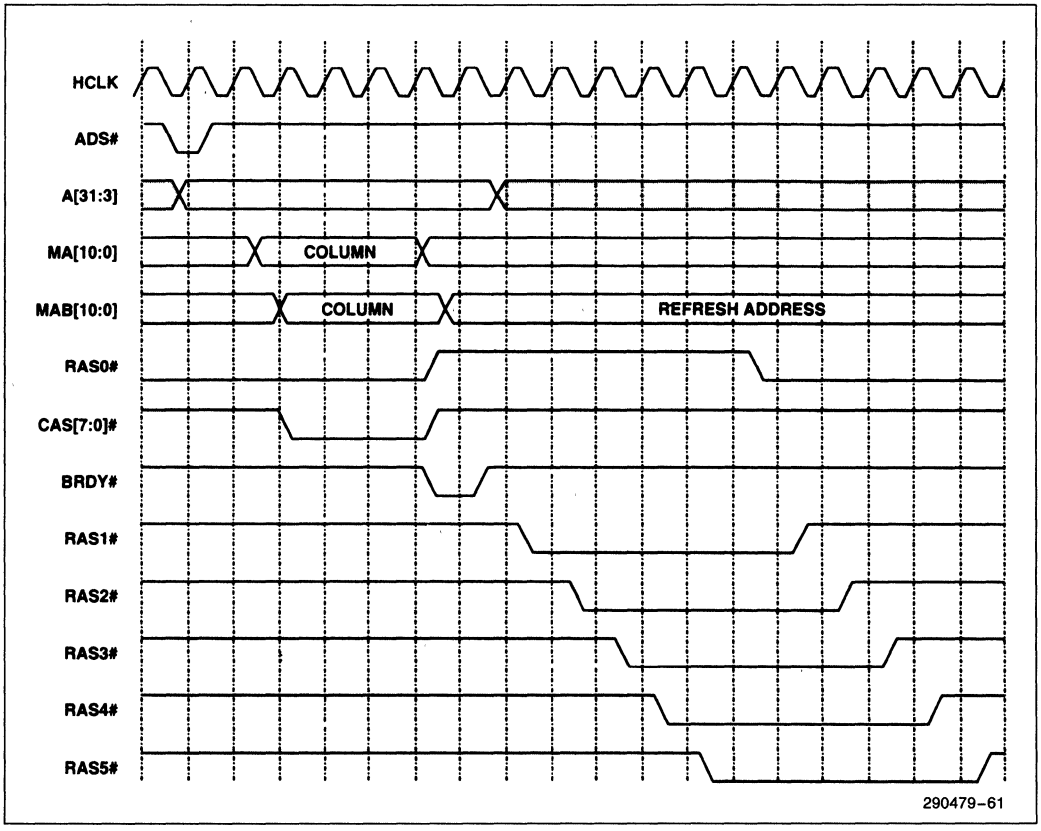


Figure 52. RAS# Only Refresh-Single

6.1.5.2 CAS#-before-RAS# Refresh-Single

Figure 53 depicts a CAS#-before-RAS# refresh cycle when the 82434LX is programmed for single refresh cycles. The diagram shows a CPU read cycle completing as the refresh timing inside the PCMC generates a refresh request. The CPU read cycle is less than a Qword, therefore a hidden refresh is not initiated. After the CPU read cycle completes, all of the RAS# and CAS# lines are negated. The PCMC then asserts CAS[7:0]# and then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last RAS# line asserted. Each RAS# line is asserted for six clocks.

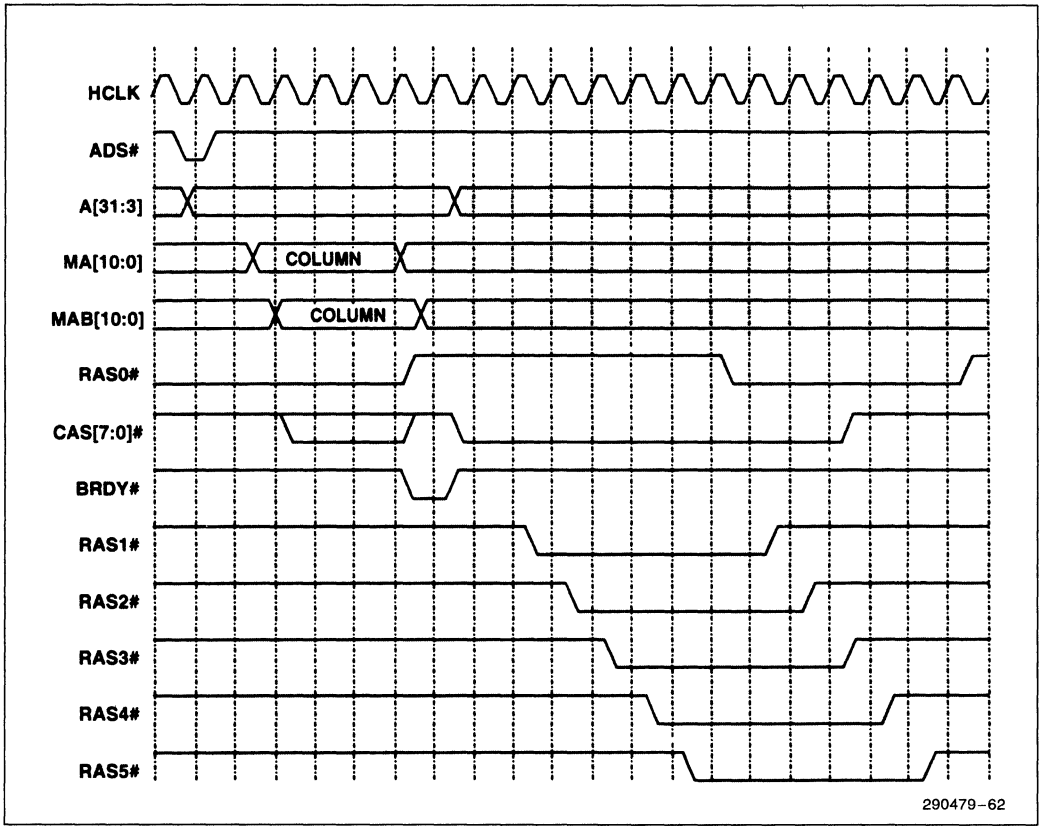


Figure 53. CAS#-before-RAS# Refresh-Single

6.1.5.3 Hidden Refresh-Single

Figure 54 depicts a hidden refresh cycle which takes place after a DRAM read page hit cycle. The diagram shows a CPU read cycle completing as the refresh timing inside the 82434LX generates a refresh request. The CPU read cycle is an entire Qword, therefore a hidden refresh is initiated. After the CPU read cycle completes, RAS# is negated, but all eight CAS# lines remain asserted. The PCMC then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last active RAS# line. Each RAS# line is asserted for six clocks.

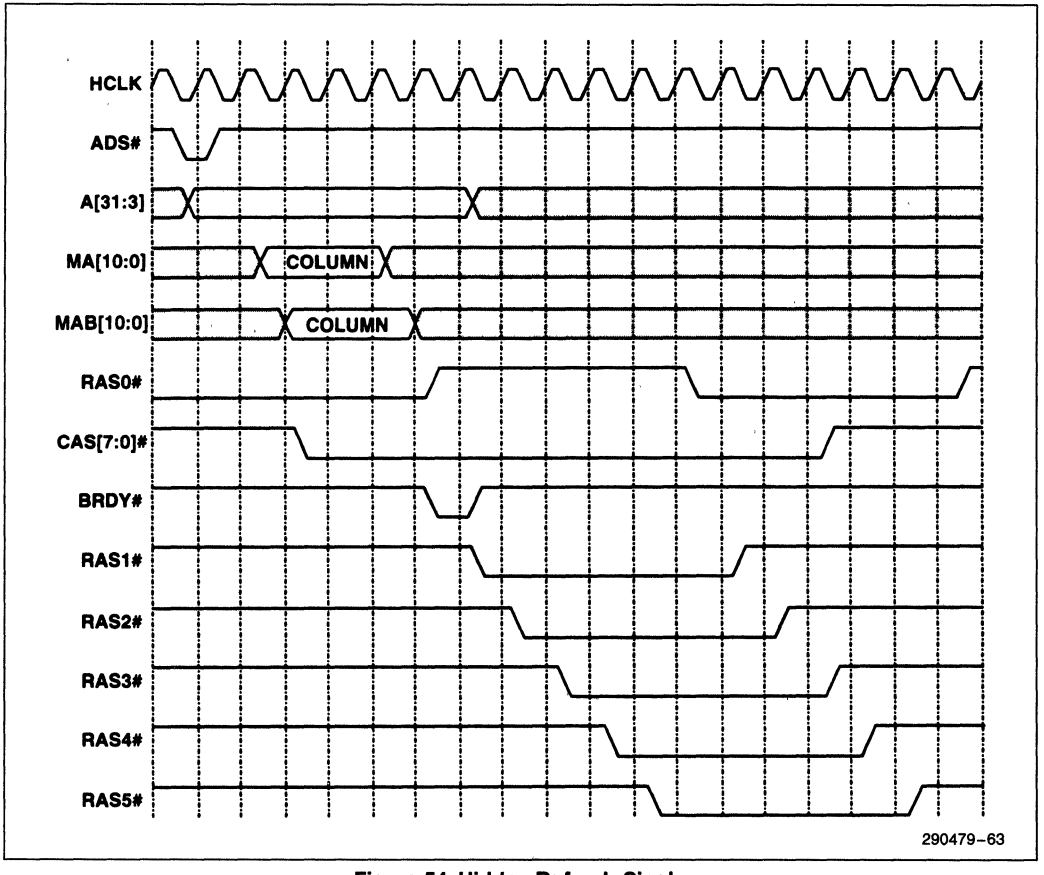


Figure 54. Hidden Refresh-Single

6.2 82434NX DRAM Interface

This section describes the 82434NX DRAM interface. Changes in the 82430NX PCIset from the 82430 PCIset include:

1. Increased maximum DRAM memory size to 512 MBytes. The 82430NX PCIset increases the maximum memory array size from 192 MBytes to 512 MBytes.
2. Two additional row address lines (RAS[7:6]#) for a total of eight (RAS[7:0]#).
3. Addition of 50 MHz host-bus optimized DRAM timing sets.
4. Three additional registers are added to support the increased memory sizeDRAM Row Boundary Registers 6 and 7 (DRB[7:6]) and the DRAM Row Boundary Extension (DRBE) Register.
5. Modified MA[11:0] timing to provide more MA[11:0] setup time to CAS[7:0]# assertion.

6.2.1 DRAM ADDRESS TRANSLATION

The MA[11:0] lines are translated from the host address lines A[26:3] for all memory accesses, except those targeted to memory that has been remapped as a result of the creation of a memory space gap in the lower extended memory area. In the case of a cycle targeting remapped memory, the least significant bits come directly from the host address, while the more significant bits depend on the memory space gap start address, gap size, and the size of main memory.

2

Table 15. DRAM Address Translation

Memory Address MA[11:0]	11	10	9	8	7	6	5	4	3	2	1	0
Column Address	A25	A23	A21	A11	A10	A9	A8	A7	A6	A5	A4	A3
Row Address	A26	A24	A22	A20	A19	A18	A17	A16	A15	A14	A13	A12

6.2.2 CYCLE TIMING SUMMARY

The 82434NX PCMC DRAM performance for 50 MHz Host bus clock is summarized in Table 13 for all CPU read and write cycles. The 60/66 MHz MA[11:0] timings when in X-4-4-4 mode have one difference from the 82434LX MA[11:0] timings. The MA lines switch to the next address in the burst sequence one clock sooner than in the 82434LX, providing more MA[11:0] setup time to CAS[7:0]# assertion. The 60/66 MHz DRAM timings for write cycles have been improved by 1 clock for all leadoffs. The 50 MHz timings shown below are selected by HOF=00, DBT=11, RWS=0 and CWS=0.

Table 16. CPU to DRAM Performance Summary for 50 MHz Host Bus Clock

Cycle Type	x-3-3-3 Timing(1)
Read (Page Hit/Row Miss/ Page Miss)	6/10/12-3-3-3
Posted Write	4-1-1-1
Write (Page Hit/Row Miss/ Page Miss)	10/11/13-3-3-3
0-Active RAS# Mode Reads	9-3-3-3
0-Active RAS# Mode Writes	9-3-3-3

NOTES:
1. Single cycle timings are identical to these leadoff timings.

Table 17. Refresh Cycle Performance (Independent of CPU frequency)

Refresh Type	Hidden Refresh	RAS# Only Refresh	CAS#-Before-RAS#
Single	16	17	18
Burst of Four	64	68	72

6.2.3 CPU TO DRAM BUS CYCLES

In this section, all timing diagrams are for 50 MHz DRAM timing, 1-Active RAS mode. The 60/66 MHz MA[11:0] timings when in X-4-4-4 mode have one difference from the 82434LX MA[11:0] timings. The MA lines switch to the next address in the burst sequence one clock sooner than in the 82434LX. The write cycle leadoffs are 1 clock earlier for 82430NX than 82430 (the MIGs and CAS timings improved by 1 clock). The 0-Active RAS# modes closely resemble the row miss cases. In 0-Active RAS# mode, RAS# is asserted one clock sooner than is shown in the row miss timing diagrams.

6.2.3.1 Burst DRAM Read Page Hit

Figure 55 depicts a CPU burst read page hit to DRAM. The 82434NX decodes the CPU address as a page hit and drives the column address onto the MA[11:0] lines. CAS[7:0]# are then asserted for two CLKs and negated for one CLK. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the processor burst order, negating and asserting CAS[7:0]# to read each Qword. The MD[63:0] data is sampled with HCLK in the LBXs when MDLE is asserted, and driven on the host bus the following cycle to meet the setup time of the CPU. BRDY# is then asserted. When MDLE is negated, the LBX continues to drive the latched HD[63:0] to ensure that the data hold time to CWE[7:0]# is met for standard SRAMs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read page hit from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when the last BRDY# is asserted.

The diagram also shows the typical control signal timing for a burst SRAM line fill operation. Note that CCS# inactive will mask any new ADS# (caused by the NA# assertion) to the burst SRAMs.

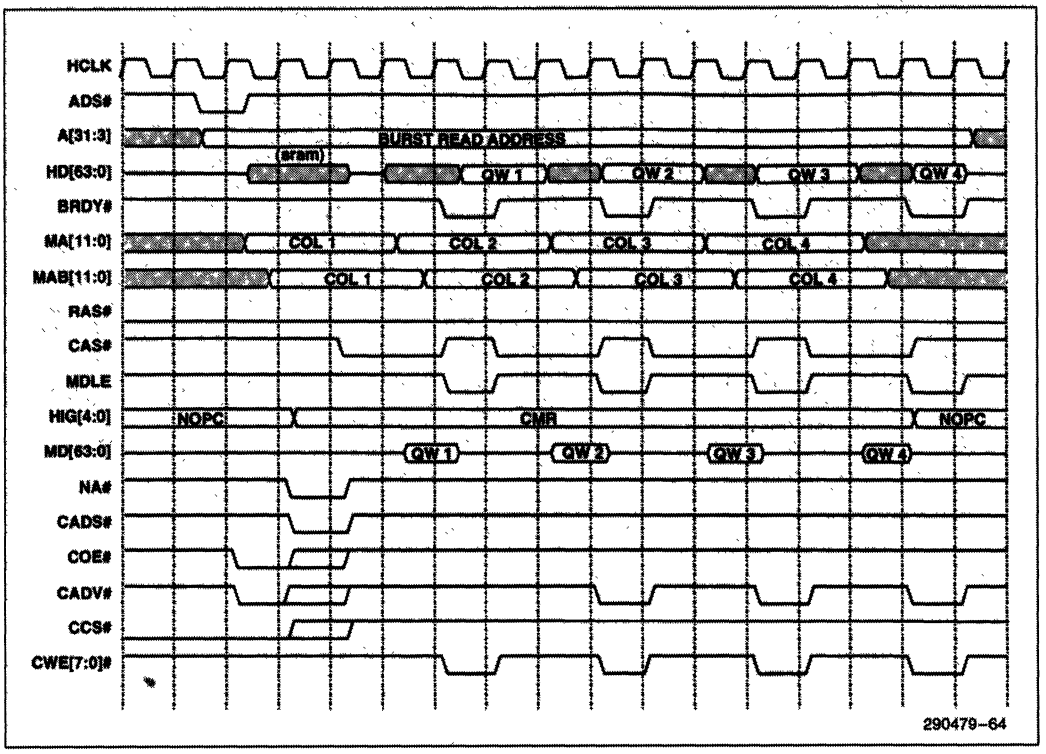


Figure 55. Burst DRAM Read Cycle-Page Hit

6.2.3.2 Burst DRAM Read Page Miss

Figure 56 depicts a CPU to DRAM burst read page miss cycle. The 82434NX decodes the CPU address as a page miss and switches from initially driving the column address to driving the row address on the MA[11:0] lines. RAS# is then negated to precharge the DRAMs and then asserted to latch the new DRAM row address. The PCMC then switches the MA[11:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the microprocessor burst order, negating and asserting CAS[7:0]# to read each Qword. The MD[63:0] data is sampled with HCLK in the LBXs when MDLE is asserted, and driven on the host bus the following cycle to meet the setup time of the CPU. BRDY# is then asserted. When MDLE is negated, the LBX continues to drive the latched HD[63:0] to ensure that the data hold time to CWE[7:0]# is met for standard SRAMs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. The HIG[4:0] lines are driven to NOPC when the last BRDY# is asserted.

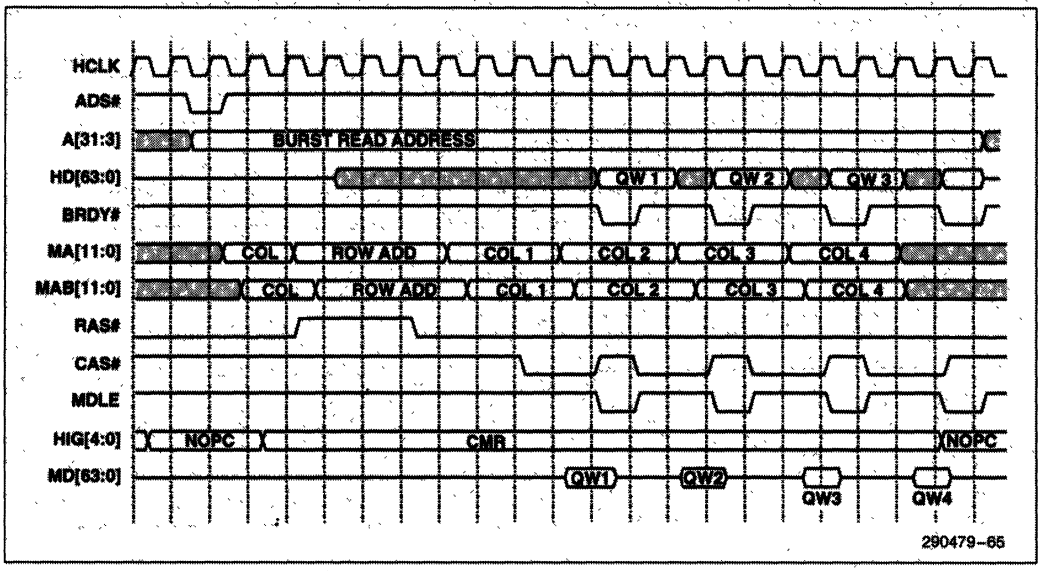


Figure 56. Burst DRAM Read Cycle-Page Miss

6.2.3.3 Burst DRAM Read Row Miss

Figure 57 depicts a CPU to DRAM burst read row miss cycle. The 82434NX decodes the CPU address as a row miss and switches from initially driving the column address to driving the row address on the MA[11:0] lines. The RAS# signal that was asserted is negated and the RAS# for the currently accessed row is asserted (RAS# is asserted 1 clock earlier in 0-Active RAS# Mode.) The PCMC then switches the MA[11:0] lines to drive the column address and asserts CAS[7:0]#. CMR (CPU Memory Read) is driven on the HIG[4:0] lines to enable the memory data to host data path through the LBXs. The PCMC advances the MA[1:0] lines through the microprocessor burst order, negating and asserting CAS[7:0]# to read each Qword. The MD[63:0] data is sampled with HCLK in the LBXs when MDLE is asserted, and driven on the host bus the following cycle to meet the setup time of the CPU. BRDY# is then asserted. When MDLE is negated, the LBX continues to drive the latched HD[63:0] to ensure that the data hold time to CWE[7:0]# is met for standard SRAMs. The LBXs tri-state the host data bus when HIG[4:0] change to NOPC and MDLE rises. A single read row miss from DRAM is similar to the first read of this sequence. The HIG[4:0] lines are driven to NOPC when the last BRDY# is asserted.

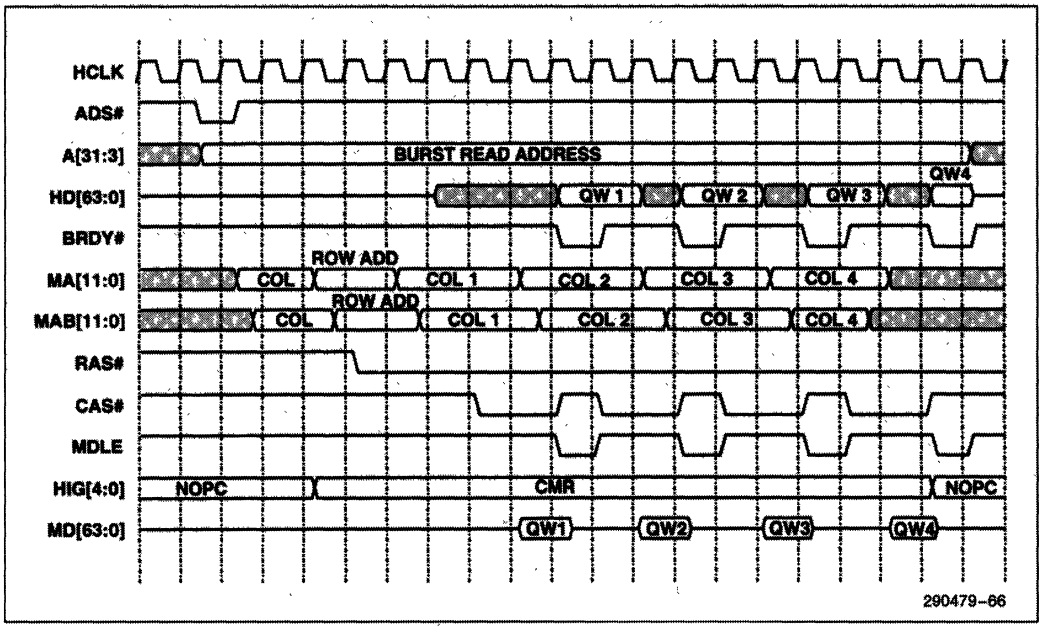


Figure 57. Burst DRAM Read Cycle-Row Miss

6.2.3.4 Burst DRAM Write Page Hit

Figure 58 depicts a CPU burst write page hit to DRAM. The 82434NX decodes the CPU write cycle as a DRAM page hit. The HIG[4:0] lines are driven to PCMWQ to post the write to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 3-1-1-1. When the cycle is decoded as a page hit, the PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword for two more clocks. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. The MIG[2:0] lines are driven to NOPM in the clock when the last CAS[7:0]# are asserted.

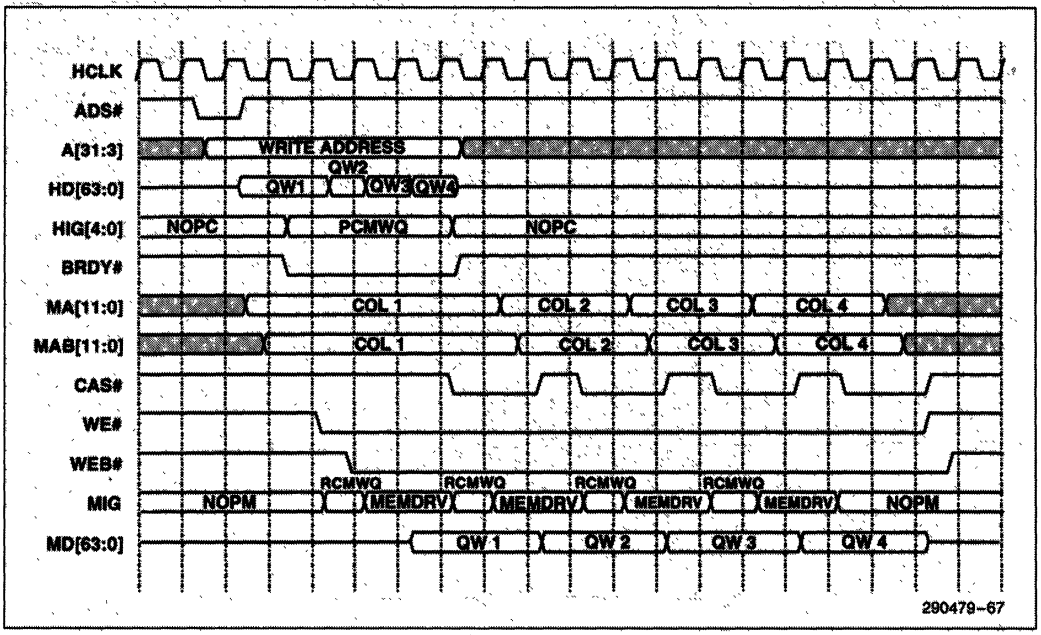


Figure 58. Burst DRAM Write Page Miss

6.2.3.5 Burst DRAM Write Page Miss

Figure 59 depicts a CPU burst write page miss to DRAM. The 82434NX decodes the CPU write cycle as a DRAM page miss and drives the PCMWQ command [HIG[4:0] lines] to post the write data to the LBXs. In the figure, the write cycle is posted to the CPU-to-Memory Posted Write Buffer at 3-1-1-1. When the cycle is decoded as a page miss, the PCMC switches the MA[11:0] lines from the column address to the row address and asserts WE# in clock 4. The PCMC drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The RAS# signal for the currently decoded row is negated to precharge the DRAMs. RAS# is then asserted to cause the DRAMs to latch the row address. The PCMC then switches the MA[11:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the Pentium processor burst order. A single write is similar to the first write of the burst sequence. The MIG[2:0] lines are driven to NOPM in the clock when the last CAS[7:0]# are asserted.

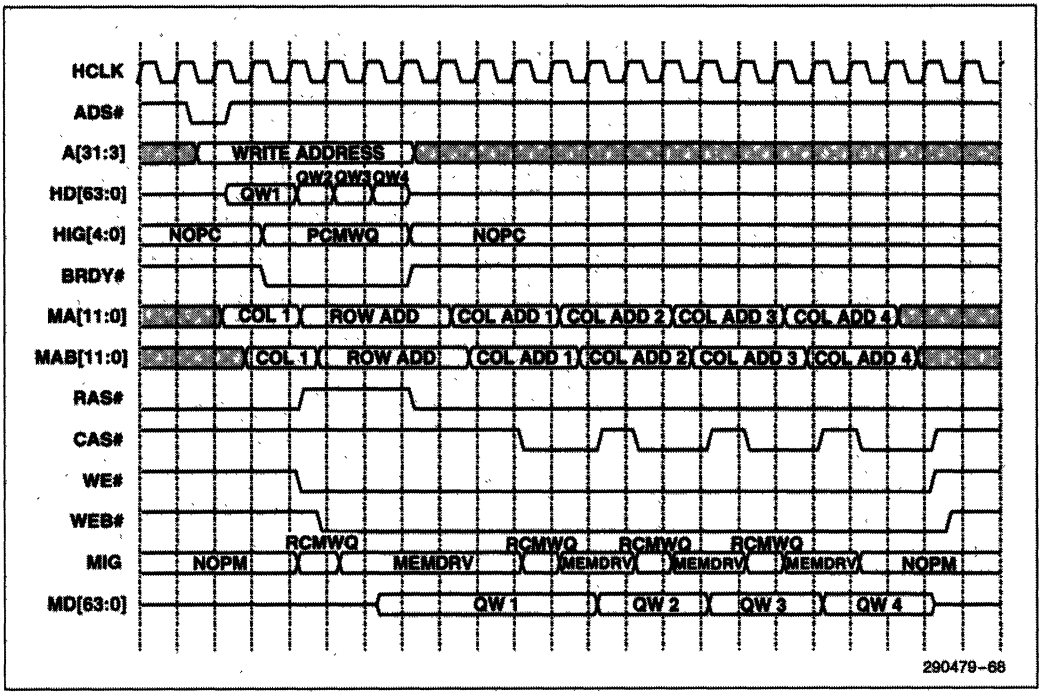


Figure 59. Burst DRAM Write Cycle-Page Miss

6.2.3.6 Burst DRAM Write Row Miss

Figure 60 depicts a CPU burst write row miss to DRAM. The 82434NX decodes the CPU write cycle as a DRAM row miss and the HIG[4:0] lines are driven to PCMWQ to post the write data into LBXs. When the cycle is decoded as a row miss, the PCMC negates the already active RAS# signal, switches the MA[11:0] lines from the column address to the row address and asserts the RAS# signal for the currently decoded row. The PCMC asserts WE# and drives the RCMWQ command on MIG[2:0] to enable the LBXs to drive the first Qword of the write onto the memory data lines. MEMDRV is then driven to cause the LBXs to continue to drive the first Qword. The PCMC then switches the MA[11:0] lines to the column address and asserts CAS[7:0]# to initiate the first write. CAS[7:0]# are then negated and asserted to perform the writes to the DRAMs as the MA[1:0] lines advance through the microprocessor burst order. A single write is similar to the first write of the burst sequence. The MIG[2:0] lines are driven to NOPM in the clock when the last CAS[7:0]# are asserted.

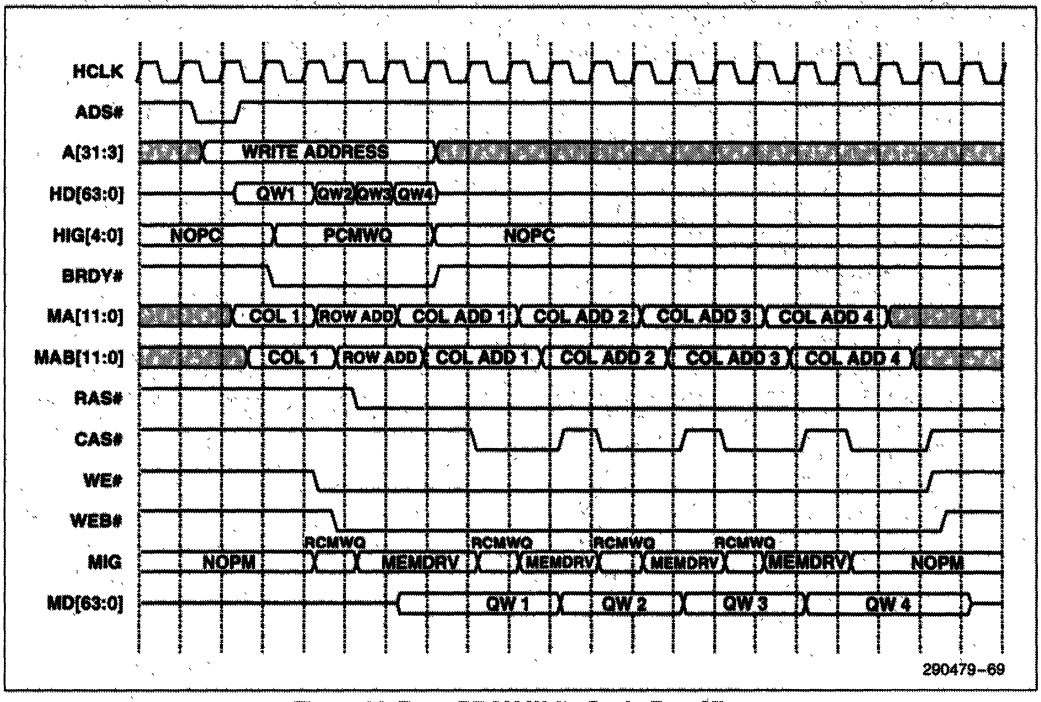


Figure 60. Burst DRAM Write Cycle-Row Miss

6.2.4 REFRESH

The refresh of the DRAM array can be performed by either using RAS#-only or CAS#-before-RAS# refresh cycles. When programmed for CAS#-before-RAS# refresh, hidden refresh cycles are initiated when possible. The timing of internally generated refresh cycles is derived from HCLK and is independent of any expansion bus refresh cycles.

The DRAM controller contains an internal refresh timer which periodically requests the refresh control logic to perform either a single refresh or a burst of four refreshes. The single refresh interval is 15.6 μs. The interval for burst of four refreshes is four times the single refresh interval, or 62.4 μs. The PCMC is configured for either single or burst of four refresh and either RAS#-only or CAS#-before RAS# refresh via the DRAM Control Register (offset 57h).

To minimize performance impact, refresh cycles are partially deferred until the DRAM interface is idle. Refresh cycles are initiated such that the RAS# maximum active time is never violated.

Hidden refresh cycles are run whenever all eight CAS# lines are active at the end of a read transaction when the refresh cycle is internally requested. Normal CAS#-before-RAS# refresh cycles are run whenever the DRAM interface is idle when the refresh is requested, or when any subset of the CAS# lines is inactive as the refresh is internally requested.

To minimize the power surge for refreshing a large DRAM array, the DRAM interface staggers the assertion and negation of the RAS# signals during both CAS#-before-RAS# and RAS#-only refresh cycles. The order of RAS# edges is dependent on which RAS# was most recently asserted prior to the refresh sequence. The RAS# that was active will be the last to be activated during the refresh sequence. All RAS[7:0]# lines are negated at the end of refresh cycles, making the first DRAM cycle after a refresh sequence a row miss.

6.2.4.1 RAS#-Only Refresh—Single

Figure 61 depicts a RAS#-only refresh cycle when the 82434NX is programmed for single refresh cycles. The diagram shows a cycle completing as the refresh timer inside the PCMC generates a refresh request. The refresh address is driven on the MA[11:0] lines. Since the cycle was to row 0, RAS0# is negated. RAS1# is the first to be asserted. RAS2# through RAS7# are then asserted sequentially while RAS0# is driven high, precharging the DRAMs in row 0. RAS0# is then asserted after RAS7#. Each RAS# line is asserted for eight host clocks.

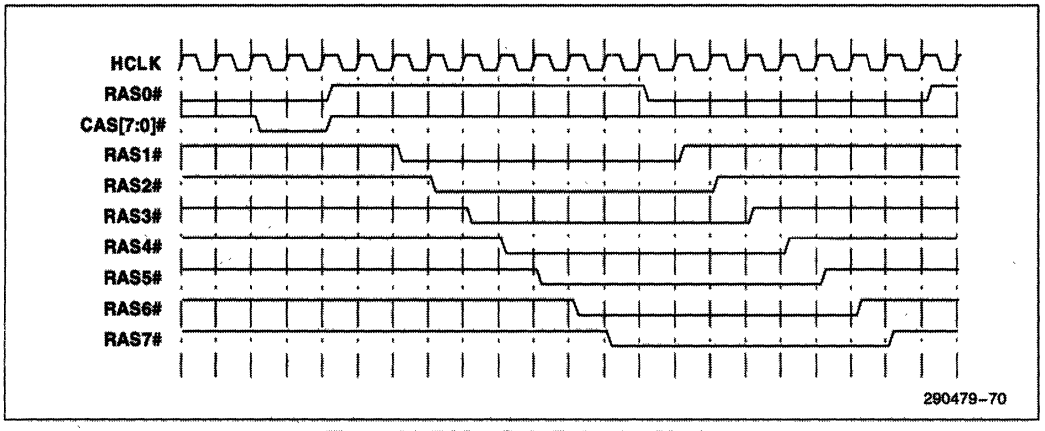


Figure 61. RAS#-Only Refresh—Single

6.2.4.2 CAS#-before-RAS# Refresh—Single

Figure 62 depicts a CAS#-before-RAS# refresh cycle when the 82434NX is programmed for single refresh cycles. The diagram shows a write cycle completing as the refresh timer inside the PCMC generates a refresh request. The cycle is less than a Qword, therefore a hidden refresh is not initiated. After the cycle completes, all of the RAS# and CAS# lines are negated. The PCMC then asserts CAS[7:0]# and then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last RAS# line asserted. Each RAS# line is asserted for eight clocks.

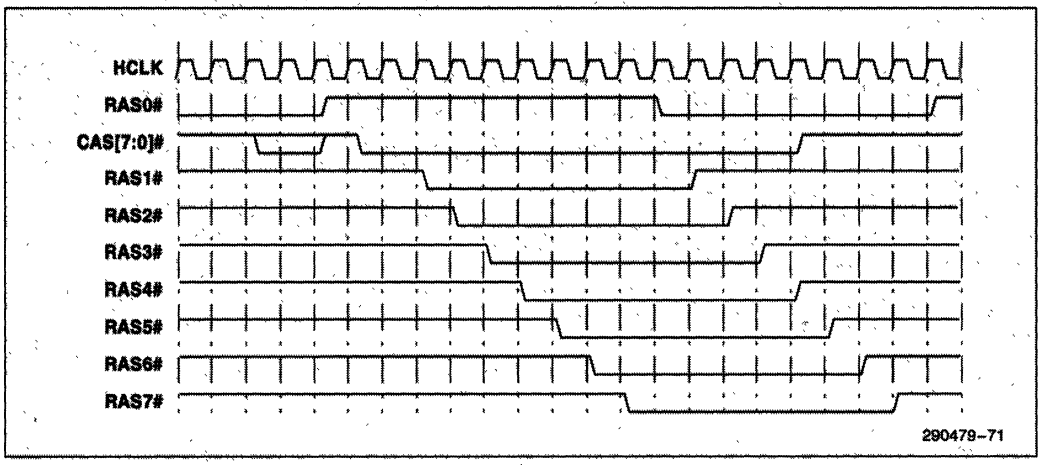


Figure 62. CAS#-Before-RAS# Refresh—Single

6.2.4.3 Hidden Refresh-Single

Figure 63 depicts a hidden refresh cycle which takes place after a DRAM read page hit cycle. The diagram shows a read cycle completing as the refresh timing inside the 82434NX PCMC generates a refresh request. The cycle is an entire Qword; therefore, a hidden refresh is initiated. After the cycle completes, RAS# is negated, but all eight CAS# lines remain asserted. The PCMC then sequentially asserts the RAS# lines, starting with RAS1# since RAS0# was the last active RAS# line. Each RAS# line is asserted for eight clocks.

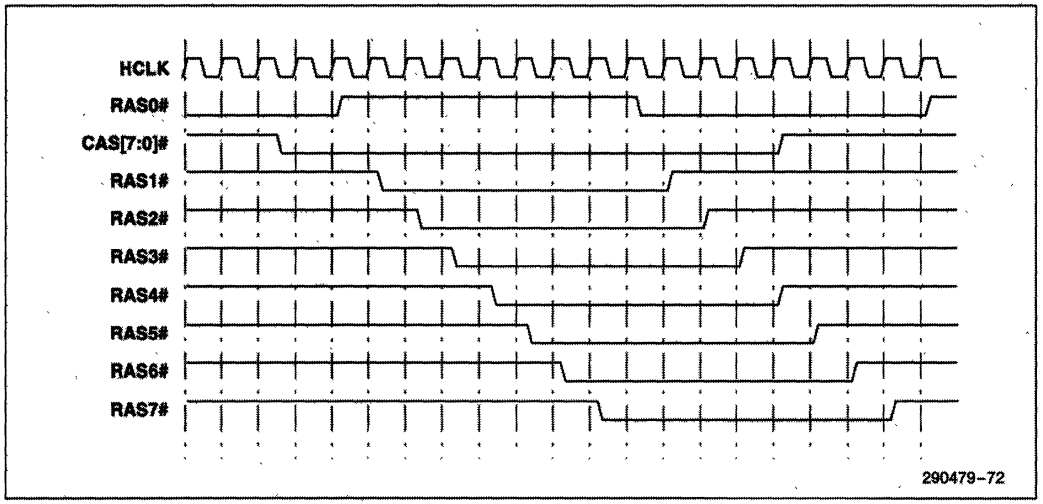


Figure 63. Hidden Refresh—Single

7.0 PCI INTERFACE

The description in this section applies to both the 82434LX and 82434NX.

7.1 PCI Interface Overview

The PCMC and LBXs form a high performance bridge from the Pentium processor to PCI and from PCI to main memory. During PCI-to-main memory cycles, the PCMC and LBXs act as a target on the PCI Bus, allowing PCI masters to read from and write to main memory. During CPU cycles, the PCMC acts as a PCI master. The CPU can then read and write I/O, memory and configuration spaces on PCI. When the CPU accesses I/O mapped and configuration space mapped PCMC registers, the PCMC intercepts the cycles and does not forward them to PCI. Although these CPU cycles do not result in a PCI bus cycle, they are described in this section since most of the PCMC internal registers are mapped into PCI configuration space.

7.2 CPU-to-PCI Cycles

7.2.1 CPU WRITE TO PCI

Figure 64 depicts a series of CPU memory writes which are posted to PCI. The CPU initiates the cycles by asserting ADS# and driving the memory address onto the host address lines. The PCMC asserts NA# in the clock after ADS# allowing the Pentium processor to drive another cycle onto the host bus two clocks later. The PCMC decodes the memory address and drives PCPWL on the HIG[4:0] lines, posting the host address bus and the low Dword of the data bus to the LBXs. The PCMC asserts BRDY#, terminating the CPU cycle with one wait state. Since NA# is asserted in the second clock of the first cycle, the Pentium processor does not insert an idle cycle after this cycle completes, but immediately drives the next cycle onto the bus. Thus, the Pentium processor maximum Dword write bandwidth of 89 MBytes/second is achieved during back-to-back Dword writes cycles. Each of the following write cycles is posted to the LBXs in three clocks.

In this example, the PCMC is parked on PCI and therefore, does not need to arbitrate for the bus. When parked, the PCMC drives the SCPA command on the PIG[3:0] lines and asserts DRVPCI, causing the host address lines to be driven on the PCI AD[31:0] lines. After the write is posted, the PCMC drives the DCPWA command on the PIG[3:0] lines to drive the previously posted address onto the AD[31:0] lines. The PCMC then drives DCPWD onto the PIG[3:0] lines, to drive the previously posted write data onto the AD[31:0] lines. As this is occurring on PCI, the second write cycle is being posted on the host bus. In this case, the second write is to a sequential and incrementing address. Thus, the PCMC leaves FRAME# asserted, converting the write cycle into a PCI burst cycle. The PCMC continues to drive the DCPWD command on the PIG[3:0] lines. The LBXs advance the posted write buffer pointer to point to the next posted Dword when DCPWD is sampled on PIG[3:0] and TRDY# is sampled asserted. Therefore, if the target inserts a wait-state by negating TRDY#, the LBXs continue to drive the data for the current transfer. The remaining writes are posted on the host bus, while the PCMC and LBXs complete the writes on PCI.

CPU I/O write cycles to PCI differ from the memory write cycle described here in that I/O writes are never posted. BRDY# is asserted to terminate the cycle only after TRDY# is sampled asserted, completing the cycle on PCI.

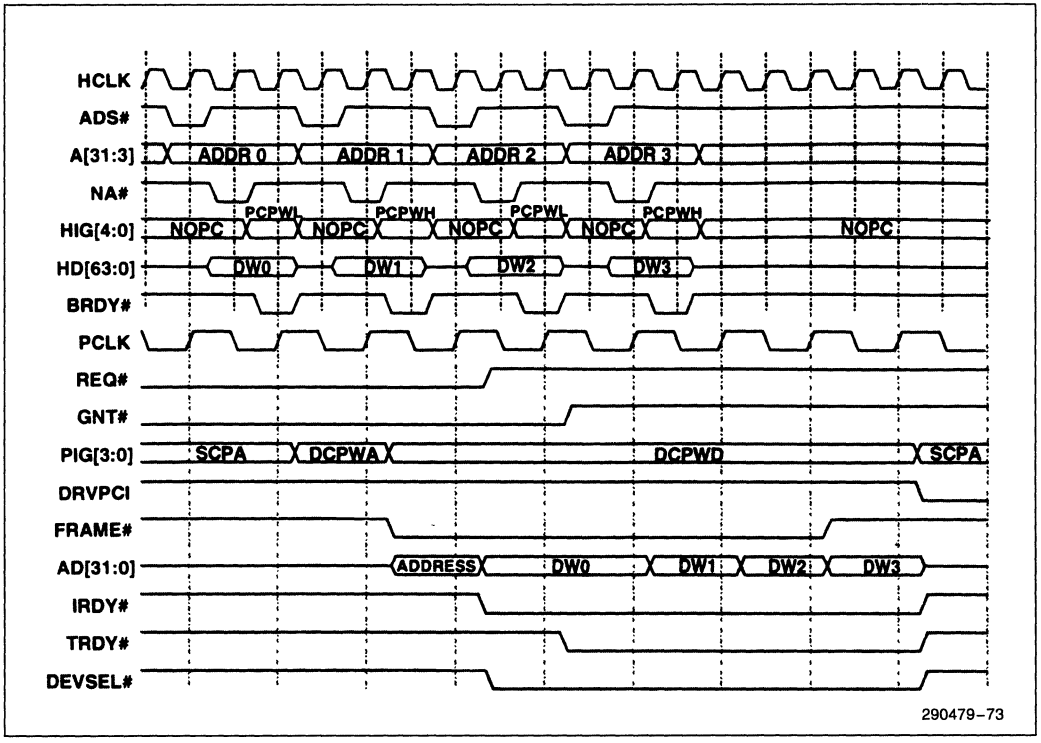


Figure 64. CPU Memory Writes to PCI

7.3 Register Access Cycles

The PCMC contains two registers which are mapped into I/O space, the Configuration Space Enable Register (I/O port CF8h) and the Turbo-Reset Control Register (I/O port CF9h). All other internal PCMC configuration registers are mapped into PCI configuration space. Configuration space must be enabled by writing a non-zero value to the Key field in the CSE Register before accesses to these registers can occur. These registers are mapped to locations C000h through C0FFh in PCI configuration space. If the Key field is programmed with 0h, CPU I/O cycles to locations C000h through CFFFh are forwarded to PCI as ordinary I/O cycles. Externally, accesses to the I/O mapped registers and the configuration space mapped registers use the same bus transfer protocol. Only the PCMC internal decode of the cycle differs. NA# is never asserted during PCMC configuration register or PCI configuration register access cycles. See Section 3.2, PCI Configuration Space Mapped Registers for details on the PCMC configuration space mapping mechanism.

7.3.1 CPU WRITE CYCLE TO PCMC INTERNAL REGISTER

A write to an internal PCMC register (either CSE Register, TRC Register or a configuration space-mapped register) is shown in Figure 65. The cycle begins with the address, byte enables and status signals (W/R#, D/C# and M/IO#) being driven to a valid state indicating an I/O write to either CF8h to access the CSE register, CF9h to access the TRC Register or C0XXh when configuration space is enabled to access a PCMC internal configuration register. The PCMC decodes the cycle and asserts AHOLD to tri-state the CPU address lines. The PCMC signals the LBXs to copy either the upper Dword or the lower Dword of the data bus onto the address lines. The PCMC makes the decision on which Dword to copy based on the BE[7:0]# lines. The HIG[4:0] lines are driven to DACPYH or DACPYL depending on whether the lower Dword of the data bus or the upper Dword of the data bus needs to be copied onto the address bus. The LBXs sample the HIG[4:0] command, and drive the data onto the address lines. The PCMC samples the A[31:0] lines on the second rising edge of HCLK after the LBXs begin driving the data. Finally, the PCMC negates AHOLD and asserts BRDY#, terminating the cycle.

If the write is to the CSE Register and the Key field is programmed to 0000b then configuration space is disabled. If the Key field is programmed to a non-zero value then configuration space is enabled.

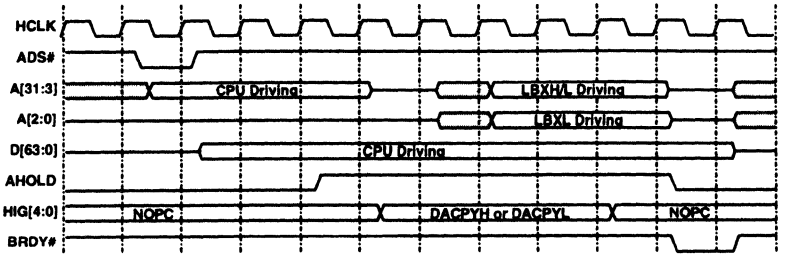


Figure 65. CPU Write to a PCMC Configuration Register

7.3.2 CPU READ FROM PCMC INTERNAL REGISTER

A read from an internal PCMC register (either CSE Register, TRC Register or a configuration space-mapped register) is shown in Figure 66. The I/O read cycle is from either CF8h to access the CSE register, CF9h to access the TRC Register or C0XXh when configuration space is enabled to access a configuration space-mapped register. The PCMC decodes the cycle and asserts AHOLD to tri-state the CPU address lines. The PCMC then drives the contents of the addressed register onto the A[31:0] lines. One byte is enabled on each rising HCLK edge for four consecutive clocks. The PCMC signals the LBXs that the current cycle is a read from an internal PCMC register by issuing the ADCPY command to the LBXs over the HIG[4:0] lines. The LBXs sample the HIG[4:0] command and copy the address lines onto the data lines. Finally, the PCMC negates AHOLD, and asserts BRDY# terminating the cycle.

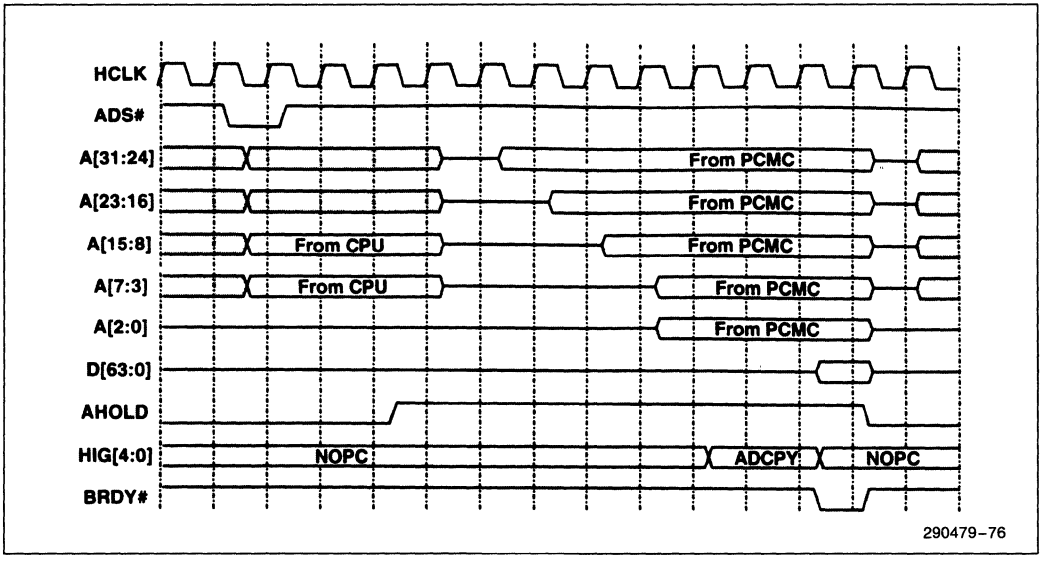


Figure 66. CPU Read from PCMC Configuration Register

7.3.3 CPU WRITE TO PCI DEVICE CONFIGURATION REGISTER

In order to write to or read from a PCI device configuration register the Key field in the CSE register must be programmed to a non-zero value, enabling configuration space. When configuration space is enabled, PCI device configuration registers are accessed by CPU I/O accesses within the range of CnXXh where each PCI device has a unique non-zero value of n. This allows a separate configuration space for each of 15 devices on PCI. Recall that when configuration space is enabled, the PCMC configuration registers are mapped into I/O ports C000h through C0FFh.

A write to a PCI device configuration register is shown in Figure 67. The PCMC internally latches the host address lines and byte enables. The PCMC asserts AHOLD to tri-state the CPU address bus and drives the address lines with the translated address for the PCI configuration cycle. The translation is described in Section 3.2, PCI Configuration Space Mapped Registers. On the HIG[4:0] lines, the PCMC signals the LBXs to latch either the upper Dword of the host data bus or the lower Dword of the host data bus to be driven onto PCI during the data phase of the PCI cycle. On the PIG[3:0] lines, the PCMC signals the LBXs to drive the latched host address lines on the PCI AD[31:0] lines. The upper two bytes of the address lines are used during configuration as IDSEL signals for the PCI devices. The IDSEL pin on each PCI device is connected to one of the AD[31:17] lines.

The PCMC drives the command for a configuration write (1011) onto the C/BE[3:0]# lines and asserts FRAME# for one PCI clock. The PCMC drives the PIG[3:0] lines signaling the LBXs to drive the contents of the PCI write buffer onto the PCI AD[31:0] lines. This command is driven for only one PCI clock before returning to the SCPA command on the PIG[3:0] lines. The LBXs continue to drive the AD[31:0] lines with the valid write data as long as DRVPCI is asserted. The PCMC then asserts IRDY# and waits until sampling the TRDY# signal active. When TRDY# is sampled asserted, the PCMC negates DRVPCI tri-stating the LBX AD[31:0] lines. BRDY# is asserted for one clock to terminate the CPU cycle.

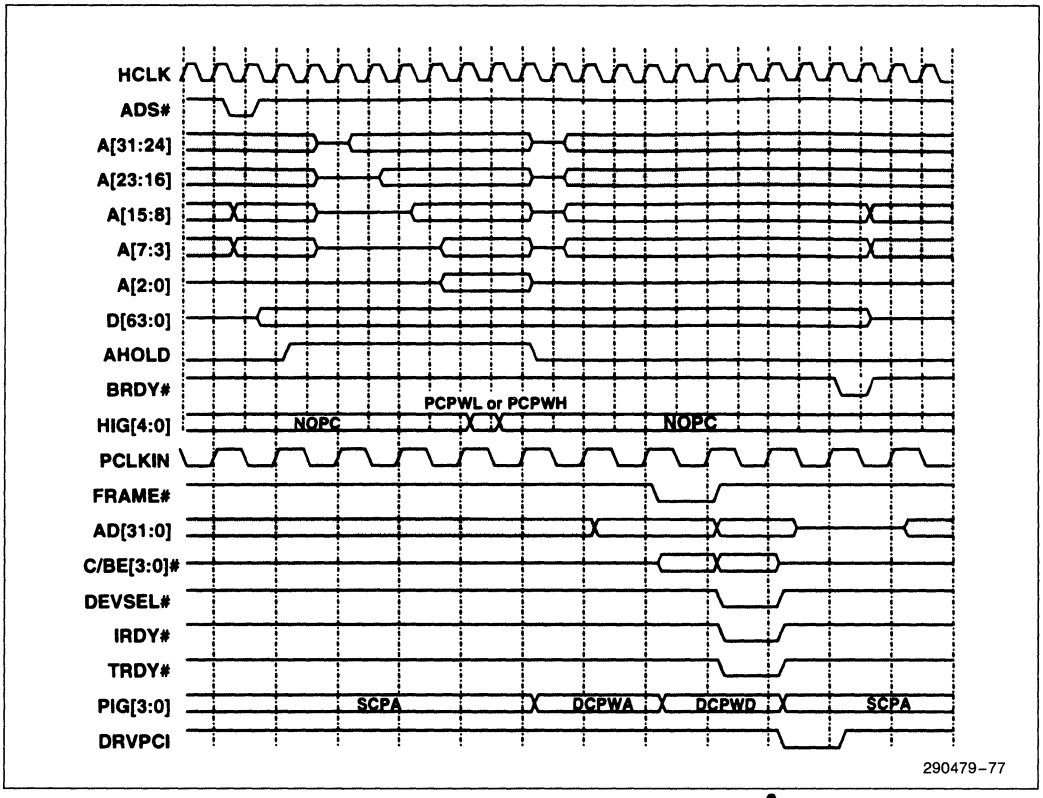


Figure 67. CPU Write to PCI Device Configuration Register

7.3.4 CPU READ FROM PCI DEVICE CONFIGURATION REGISTER

In order to write to or read from a PCI device configuration register the Key field in the CSE register must be programmed to a non-zero value, enabling configuration space. When configuration space is enabled, PCI device configuration registers are accessed by CPU I/O accesses within the range of CnXXh where each PCI device has a unique non-zero value of n. This allows a separate configuration space for each of 15 devices on PCI. Recall that when configuration space is enabled, the PCMC configuration registers occupy I/O addresses C0XXH.

A CPU read from a PCI device configuration register is shown in Figure 68. The PCMC internally latches the host address lines and byte enables. The PCMC asserts AHOLD to tri-state the CPU address bus. The PCMC drives the address lines with the translated address for the PCI configuration cycle. The translation is described in Section 3.2, PCI Configuration Space Mapped Registers. On the PIG[3:0] lines, the PCMC signals the LBXs to drive the latched host address lines on the PCI AD[31:0] lines. The upper two bytes of the address lines are used during configuration as IDSEL signals for the PCI devices. The IDSEL pin on each PCI device is connected to one of the AD[31:17] lines.

The PCMC drives the command for a configuration read (1010) onto the C/BE[3:0]# lines and asserts FRAME# for one PCI clock. The PCMC drives the PIG[3:0] lines signaling the LBXs to latch the data on the PCI AD[31:0] lines into the CPU-to-PCI first read prefetch buffer. The PCMC then drives the HIG[4:0] lines signaling the LBXs to drive the data from the buffer onto the host data lines. The PCMC asserts IRDY# and waits until sampling TRDY# active. After TRDY# is sampled active, BRDY# is asserted for one clock to terminate the CPU cycle.

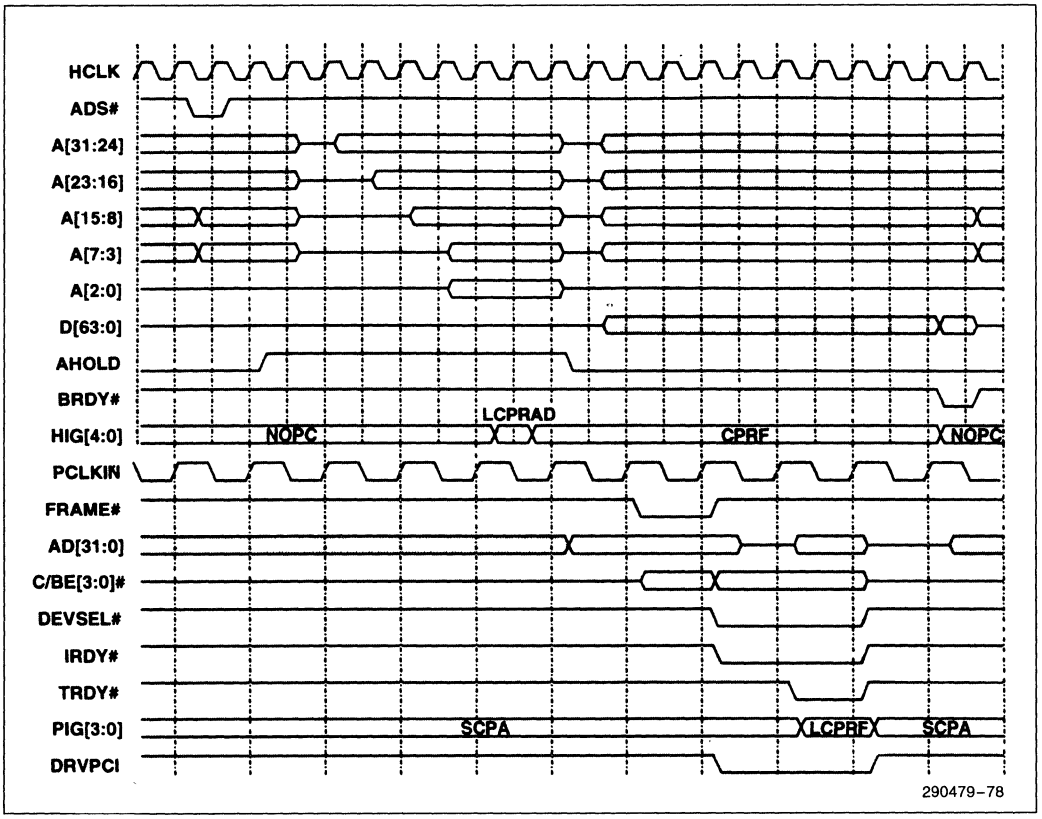


Figure 68. CPU Read from PCI Device Configuration Register

During system initialization, the CPU typically attempts to read from the configuration space of all 15 possible PCI devices to detect the presence of the devices. If no device is present, DEVSEL# is not be asserted and the cycle is terminated, returning FF ... FFh to the CPU. Figure 69 depicts an attempted read from a configuration register of a non-existent device. If no device responds then the PCMC aborts the cycle and sends the DRVFF command over the HIG[4:0] lines causing the LBXs to drive FF ... FFh onto the host data lines.

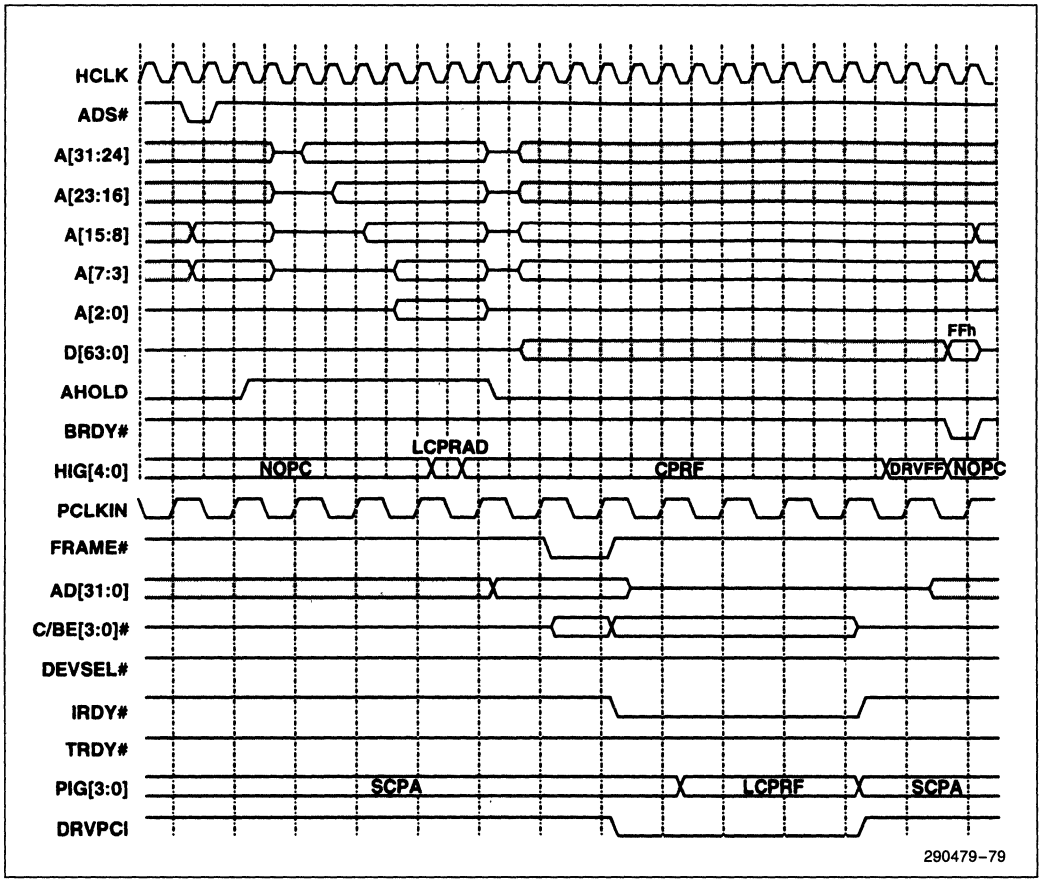


Figure 69. CPU Attempted Configuration Read from Non-Existent PCI Device

7.4 PCI-to-Main Memory Cycles

7.4.1 PCI MASTER WRITE TO MAIN MEMORY

Figure 70 depicts a PCI master burst write to main memory. The PCI master begins by driving the address on the AD[31:0] lines and asserting FRAME#. Upon sampling FRAME# active, the PCMC drives the LCPA command on the PIG[3:0] lines causing the LBXs to retain the address that was latched on the previous PCLK rising edge. The PCMC then samples MEMCS# active, indicating that the cycle is directed to main memory. The PCMC drives the PPMWA command on the PIG[3:0] lines to move the latched PCI address into the write buffer address register. The PCMC then drives the DPWA command on the HIG[4:0] lines enabling the LBXs to drive the PCI master write address onto the host address bus. The PCMC asserts EADS# to initiate a first level cache snoop cycle and simultaneously begins an internal second level cache snoop cycle. Since the snoop is a result of a PCI master write, INV is asserted with EADS#. HITM# remains negated and the snoop either hits an unmodified line or misses in the second level cache, thus no write-back cycles are required. If the snoop hit an unmodified line in either the first or second level cache, the line is invalidated. The cycle is immediately forwarded to the DRAM interface. The four posted Dwords are written to main memory as two Qwords with two CAS[7:0]# cycles. In this example, the DRAM interface is configured for X-3-3-3 write timing, thus each CAS[7:0]# low pulse is two HCLKs in length.

The PCMC disconnects the cycle by asserting STOP# when one of the two four-Dword-deep PCI-to-Memory Posted Write Buffers is full. If the master terminates the cycle before sampling STOP# asserted, then IRDY#, STOP# and DEVSEL# are negated when FRAME# is sampled negated. If the master intended to continue bursting, then the master negates FRAME# when it samples STOP# asserted. IRDY#, STOP# and DEVSEL# are then negated one clock later.

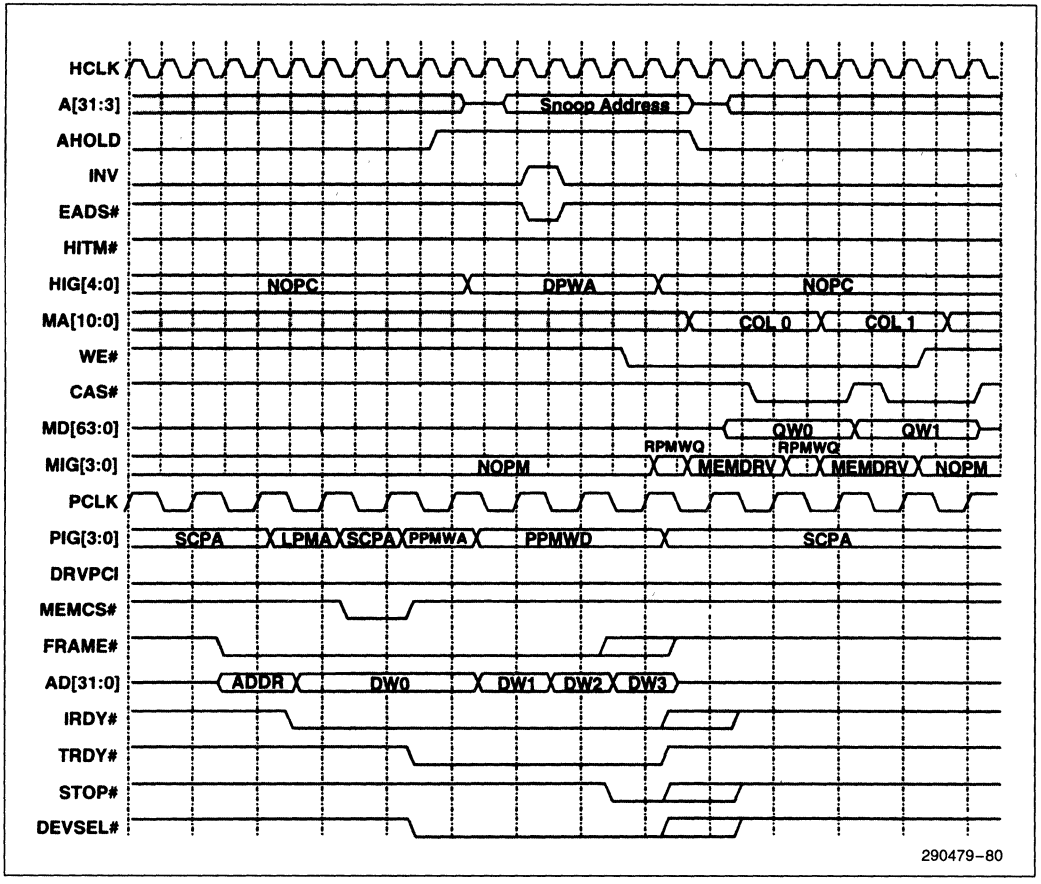


Figure 70. PCI Master Write to Main Memory-Page Hit

7.4.2 PCI MASTER READ FROM MAIN MEMORY

Figure 71 depicts a PCI master read from main memory. The PCI master initiates the cycle by driving the read address on the AD[31:0] lines and asserting FRAME#. The PCMC drives the LPMA command on the PIG[3:0] lines causing the LBXs to retain the address latched on the previous PCLK rising edge. The PCMC drives the DPRA command on the HIG[4:0] lines enabling the LBXs to drive the read address onto the host address lines. The snoop cycle misses in the second level cache and either hits an unmodified line or misses in the first level cache. The cycle is then forwarded to the DRAM interface. A read of four Qwords is performed. Each Qword is posted in the PCI-Memory Read Prefetch Buffer. The data is then driven onto PCI in an eight Dword burst cycle. If the master terminates the cycle before sampling STOP#, then IRDY#, STOP# and DEVSEL# are all negated after FRAME# is sampled inactive. If the master intended to continue bursting, then the master negates FRAME# when it samples STOP# asserted and IRDY#, STOP# and DEVSEL# are negated one clock later.

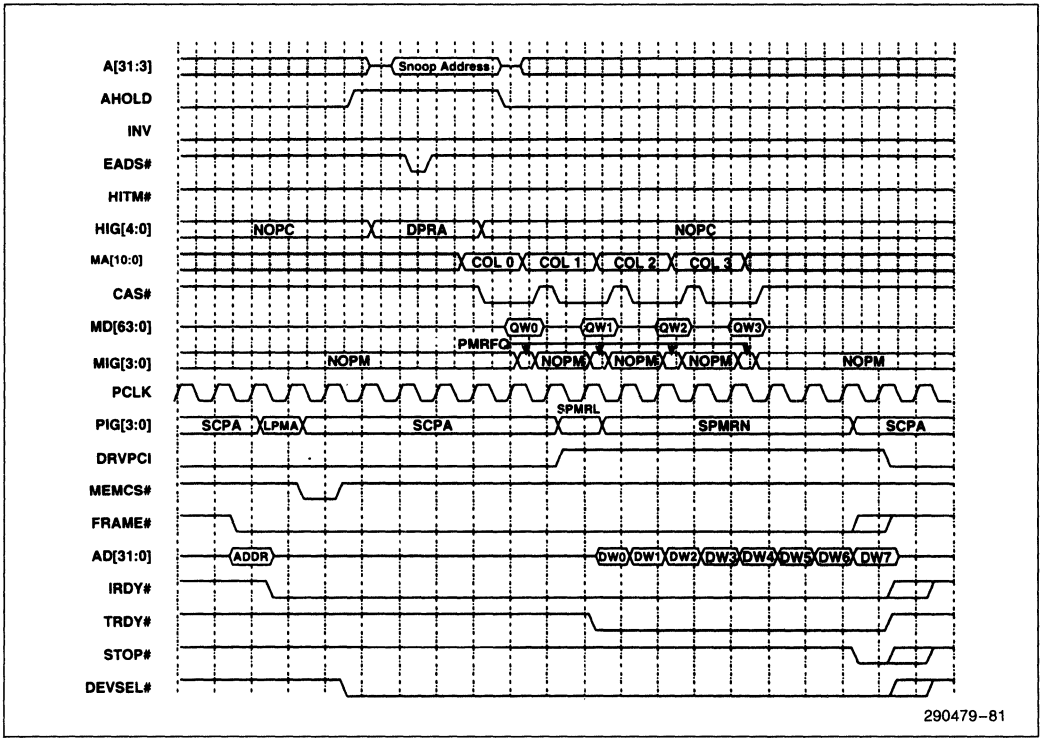


Figure 71. PCI Master Read from Main Memory-Page Hit

8.0 SYSTEM CLOCKING AND RESET

8.1 Clock Domains

The 82434LX and 82434NX PCMCs and 82433LX and 82433NX LBXs operate based on two clocks, HCLK and PCLK. The CPU, second level cache, and the DRAM interfaces operate based on HCLK. The PCI interface timing is based on PCLK.

8.2 Clock Generation and Distribution

Figure 72 shows an example of the 82434LX and 82434NX PCMC host clock distribution in the CPU, cache and memory subsystem. HCLK is distributed to the CPU, PCMC, LBXs and the second level cache SRAMs (in the case of a burst SRAM second level cache).

The host clock originates from an oscillator which is connected to the HCLKOSC input on the PCMC. The PCMC generates six low skew copies of HCLK, HCLKA–HCLKF. Figure 72 shows an example of a host clock distribution scheme for a uni-processor system. In this figure, clock loading is balanced with each HCLK output driving two loads in the system. Each clock output should drive a trace of length *k* with stubs at the end of the trace of length *l* connecting to the two loads. The *l* and *k* parameters should be matched for each of the six clock outputs to minimize overall system clock skew. One of the HCLK outputs is used to clock the PCMC and the Pentium processor. Because the clock driven to the PCMC HCLKIN input and the Pentium processor CLK input originates with the same HCLK output, clock skew between the PCMC and the CPU can be kept lower than between the PCMC and other system components. Another copy of HCLK is used to clock the LBXs. A 256 KByte burst SRAM second level cache can be implemented with eight 32 KByte x 9 synchronous SRAMs. The four remaining copies of HCLK are used to clock the SRAMs. Each HCLK output drives two SRAMs. A 512 KByte second level cache is implemented with four 64 KByte x 18 synchronous SRAMs. Two of the four extra copies are used to clock the SRAMs while the other two are unused. Any one of the HCLK outputs can be used to clock the PCMC and Pentium processor, the two LBXs or any pair of SRAMs. All six copies are identical in drive strength.

Figure 73 depicts the PCI clock distribution.

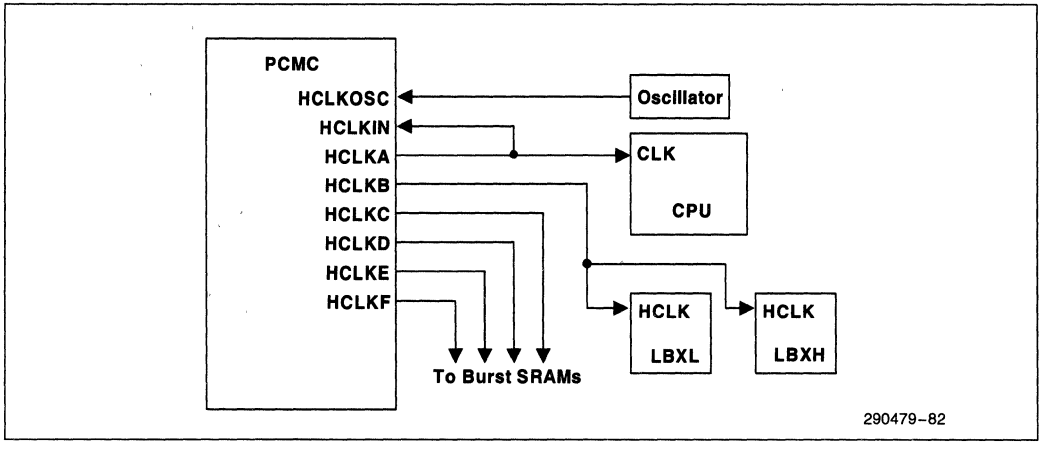


Figure 72. HCLK Distribution Example

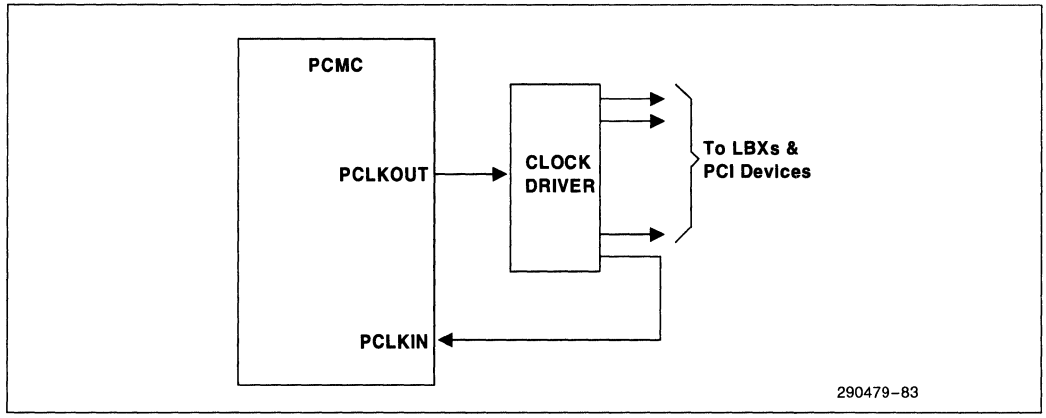


Figure 73. PCI Clock Distribution

The PCMC generates PCLKOUT with an internal Phase Locked Loop (PLL). The PCLKOUT signal is buffered using a single component to produce several low skew copies of PCLK to drive the LBXs and other devices on PCI. One of the outputs of the clock driver is directed back to the PCLKIN input on the PCMC. The PLL locks the rising edges of PCLKIN in phase with the rising edges of HCLKIN. The PLL effectively compensates for the delay of the external clock driver. The resulting PCI clock is one half the frequency of HCLK. Timing for all of the PCI interface signals is based on PCLKIN. All PCI interface inputs are sampled on PCLKIN rising edges and all outputs transition as valid delays from PCLKIN rising edges. Clock skew between the PCLKIN pin on the PCMC and the PCLK pins on the LBXs must be kept within 1.25 ns to guarantee proper operation of the LBXs.

8.3 Phase Locked Loop Circuitry

The 82434LX and 82434NX PCMCs each contain two internal Phase Locked Loops (PLLs). Loop filters and power supply decoupling circuitry must be provided externally. Figure 74 shows the PCMC connections to the external PLL circuitry.

One of the PCMC internal Phase Locked Loops (PLL) locks onto the HCLKIN input. The PLL is used by the PCMC in generating and sampling timing critical signals. An external loop filter is required. The PLLARC1 and PLLARC2 pins connect to the external HCLK loop filter. Two resistors and a capacitor form the loop filter. The loop filter circuitry should be placed as close as possible to the PCMC loop filter pins. The PLL also has dedicated power and ground pins, PLLAVDD, PLLAVSS and PLLAGND. These power pins require a low noise supply. PLLAVDD, PLLAVSS and PLLAGND must be connected to the RC network shown in Figure 74.

The second PCMC internal Phase Locked Loop (PLL) locks the PCLKIN input in phase with the HCLKIN input. The PLL is used by the PCMC to keep the PCI clock in phase with the host clock. An external loop filter is required. The PLLBRC1 and PLLBRC2 pins connect to the external PCLK loop filter. Two resistors and a capacitor form the loop filter. The loop filter circuitry should be placed as close as possible to the PCMC loop filter pins. The PLL also has dedicated power and ground pins, PLLBVDD, PLLBVSS and PLLBGND. These power pins require a low noise supply. PLLBVDD, PLLBVSS and PLLBGND must be connected to the RC network shown in Figure 74.

The resistance and capacitance values for the external PLL circuitry are listed below.

R1 = 10 KΩ ± 5%

R2 = 150Ω ± 5%

R3 = 33Ω ± 5%

C1 = 0.01 μF ± 10%

C2 = 0.47 μF ± 10%

An additional 0.01 μF capacitor in parallel with C2 will help to improve noise immunity.

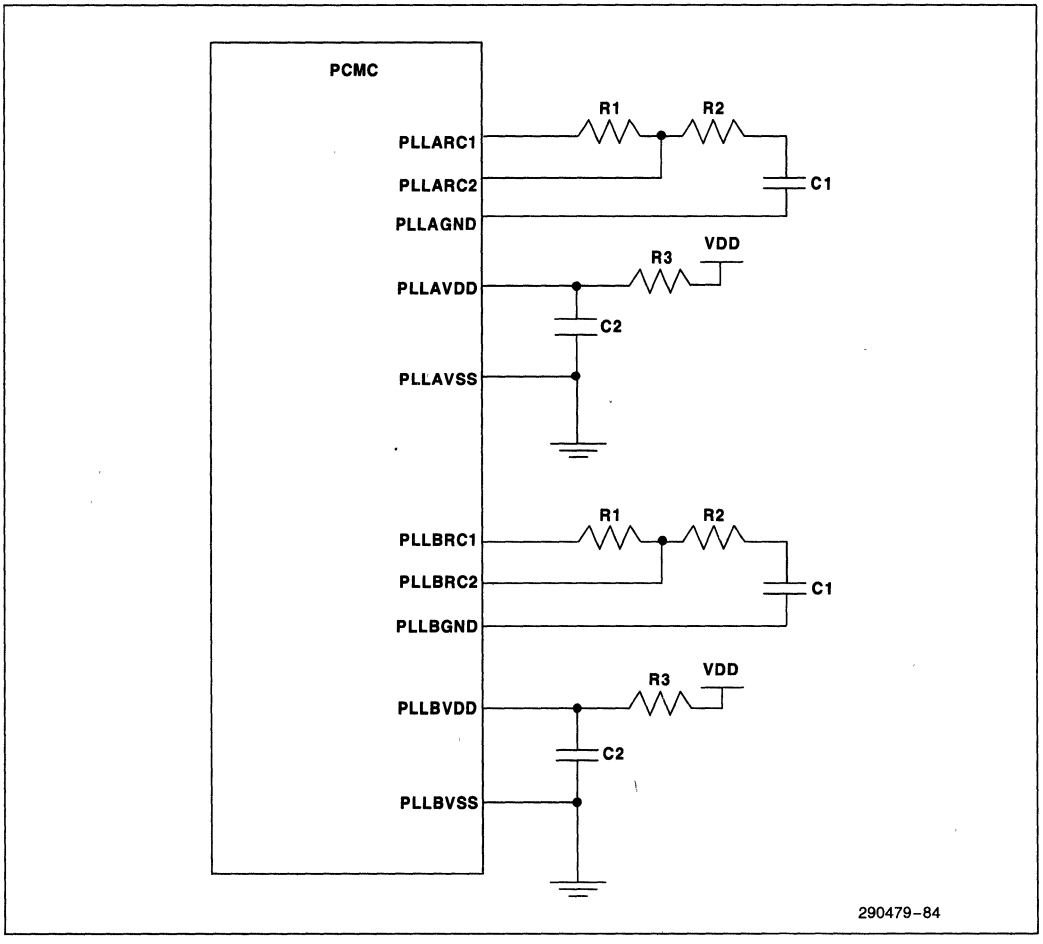


Figure 74. PCMC PLL Circuitry Connections

8.4 System Reset

Figure 75 shows the 82434LX and 82434NX PCMC system reset connections. The 82434LX and 82434NX PCMC reset logic monitors PWROK and generates CPURST, PCIRST# and INIT.

When asserted, PWROK is an indicator to the PCMC that VDD and HCLK have stabilized long enough for proper system operation. CPURST is asserted to initiate hard reset. INIT is asserted to initiate soft reset. PCIRST# is asserted to reset devices on PCI.

Hard reset is initiated by the PCMC in response to one of two conditions. First, hard reset is initiated when power is first applied to the system. PWROK must be driven inactive and must not be asserted until 1 ms after VDD and HCLK have stabilized at their AC and DC specifications. While PWROK is negated, the 82434LX asserts CPURST and PCIRST#. PWROK can be asserted asynchronously. When PWROK is asserted, the 82434LX first ensures that it has been completely initialized before negating CPURST and PCIRST#. CPURST is negated synchronously to the rising edge of HCLK. PCIRST# is negated asynchronously.

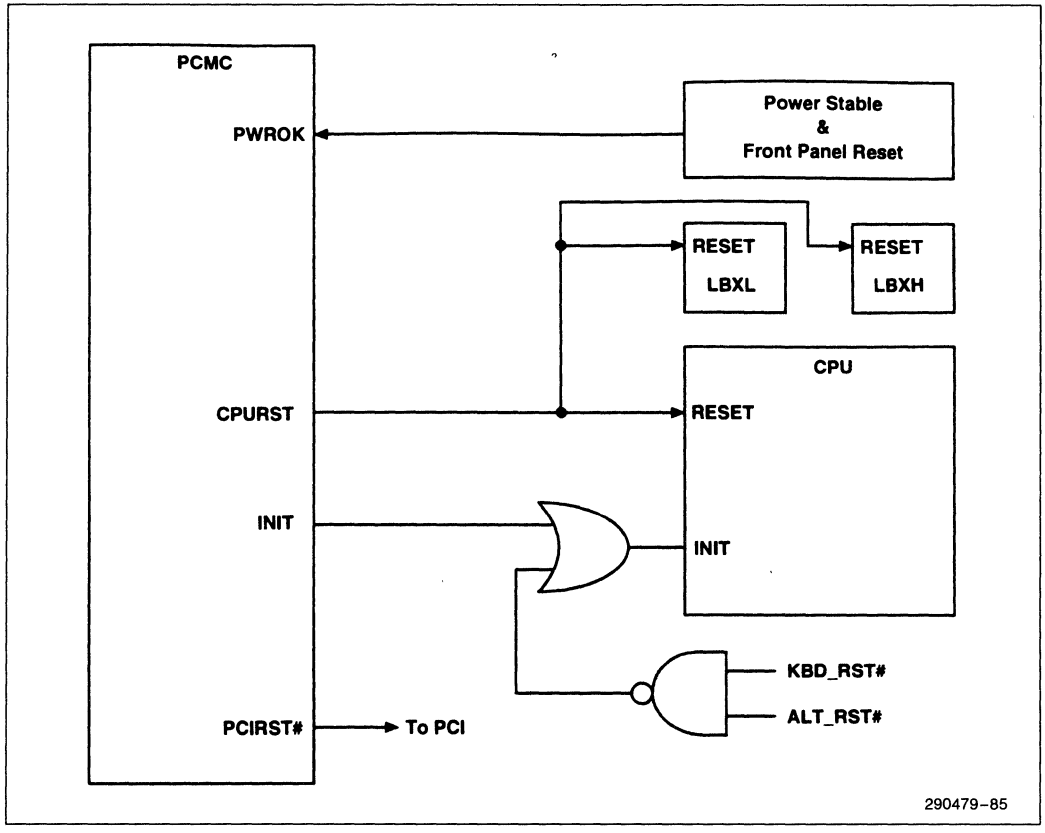


Figure 75. PCMC System Reset Logic

When PWROK is negated, the PCMC asserts AHOLD causing the CPU to tri-state the host address lines. Address lines A[31:29] are sampled by the PCMC 1 ms after the rising edge of PWROK. The values sampled on A[31:30] are inverted inside the PCMC and then stored in Configuration Register 52h bits 7 and 6. The A[31:30] strapping options are depicted in Table 18.

Table 18. A[31:30] Strapping Options

A[31:30]	Configuration Register 52h, Bits[7:6]	Secondary Cache Size
11	00	Not Populated
10	01	Reserved
01	10	256 KByte Cache
00	11	512 KByte Cache

The value sampled on A29 is inverted inside the PCMC and stored in the SRAM Type Bit (bit 5) in the SCC Register. A28 is required to be pulled high for compatibility with future versions of the PCMC.

The PCMC also initiates hard reset when the System Hard Reset Enable bit in the Turbo-Reset Control Register (I/O address CF9h) is set to 1 and the Reset CPU bit toggles from 0 to 1. The PCMC drives CPURST and PCIRST# active for a minimum of 1 ms.

Table 19 shows the state of all 82434LX PCMC output and bi-directional signals during hard reset. During hard reset both CPURST and PCIRST# are asserted. When the hard reset is due to PWROK negation, AHOLD is asserted. The PCMC samples the strapping options on the A[31:29] lines 1 ms after the rising edge of PWROK. When hard reset is initiated via a write to the Turbo-Reset Control Register (I/O port CF9h) AHOLD remains negated throughout the hard reset. Table 19 also applies to the 82434NX, with the exception of the signals listed in Section 8.5, 82434NX Reset Sequencing.

Table 19. 82434LX Output and I/O Signal States During Hard Reset

Signal	State	Signal	State
A[31:0]	Input	IRDY#	Input
AHOLD	High/Low	KEN#	Undefined
BOFF#	High	MA[10:0]	Undefined
BRDY#	High	MDLE	High
CAA[6:3]	Undefined	MEMACK#	High-Z
CAB[6:3]	Undefined	MIG[2:0]	Low
CADS[1:0]#	High	NA#	High
CADV[1:0]#	High	PAR	Input
CALE	High	PEN#	High
CAS[7:0]#	High	PERR#	Input
COE[1:0]#	High	PLOCK#	Input
CWE[7:0]#	High	PIG3	Low
C/BE[3:0]#	Input	PIG[2:0]	High
DEVSEL#	Input	RAS[5:0]#	High
DRVPCI	Low	REQ#	High-Z
EADS#	High	SERR#	Input
FRAME#	Input	STOP#	Input
HIG[4:0]	Low	TRDY#	Input
INIT	Low	WE#	High
INV	Low		

Soft reset is initiated by the PCMC in response to one of two conditions. First, when the System Hard Reset Enable bit in the TRC Register is reset to 0, and the Reset CPU bit toggles from 0 to 1, the PCMC initiates soft reset by asserting INIT for a minimum of 2 HCLKs. Second, the PCMC initiates a soft reset upon detecting a shutdown cycle from the CPU. In this case, the PCMC first broadcasts a shutdown special cycle on PCI and then asserts INIT for a minimum of 2 HCLKs.

8.5 82434NX Reset Sequencing

When PWROK is negated, the 82434NX PCMC drives the following signals low—BRDY#, NA#, AHOLD, EADS#, INV, BOFF#, KEN#, PEN#, CPURST, INIT, CALE, CADS[1:0]#, CADV[1:0]#, CAA[6:3], CAB[6:3], COE[1:0]#, CWE[7:0]#. HCLK[A:F] are driven as soon as the 3.3V supply is active. Note that CWE[7:0]# low prevents the second level cache data RAMs from driving the data bus, even though COE[1:0]# are also driven low. Also, note that BOFF# driven low causes the CPU to tri-state all outputs to the 82434NX PCMC and 82433NX LBX, except HITM#, SMIACT#, and PCHK#. This minimizes the number of signals that the CPU may drive to the PCMC when the 3.3V supply is active and the 5V supply is not active.

Figure 76 shows how the 82434NX sequences CPURST and PCIRST# in response to PWROK assertion.

Some PCI devices may drive 3.3V friendly signals directly to 3.3V devices that are not 5V tolerant. If such signals are powered from the 5V supply they must be driven low when PCIRST# is asserted. Some of these signals may need to be driven high before CPURST is negated. PCIRST# is negated 1 ms before CPURST to allow time for this to occur.

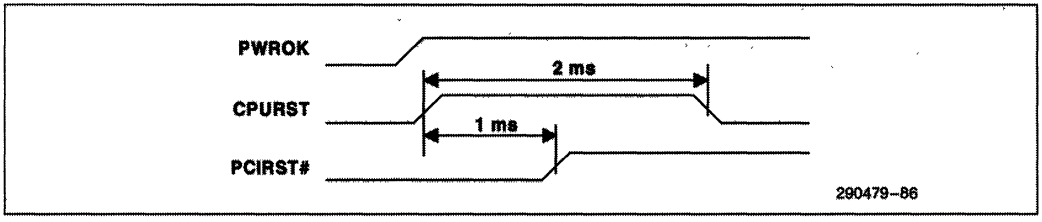


Figure 76. 82434NX Reset Sequencing at Power-Up

9.0 ELECTRICAL CHARACTERISTICS

9.1 Absolute Maximum Ratings

Case Temperature under Bias .....0°C to +85°C

Storage Temperature .....−55°C to +150°C

Voltage on Any Pin
with Respect to Ground.....−0.3 to V_{CC} + 0.3V

Supply Voltage
with Respect to V_{SS} .....−0.3 to +6.5V

Maximum Total Power Dissipation 2.0W

Maximum Power Dissipation, $V_{CC}3$ .....470 mW

The Maximum total power dissipation in the 82434NX on the V_{CC} and $V_{CC}3$ pins is 2.0W. The $V_{CC}3$ pins may draw as much as 470 mW, however, total power will not exceed 2.0W.

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

**WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

9.2 Thermal Characteristics

The 82434LX and 82434NX PCMCs are designed for operation at case temperatures between 0°C and 85°C. The thermal resistances of the package are given in Table 20.

Table 20. PCMC Package Thermal Resistance

Parameter	Air Flow Meters/Second (Linear Feet per Minute)				
	0 (0)	0.5 (98.4)	1.0 (196.9)	2.0 (393.7)	5.0 (984.3)
θ_{JA} (°C/Watt)	31	27	24.5	23	19
θ_{JC} (°C/Watt)	8.6				

9.3 82434LX DC Characteristics

Functional Operating Range (V_{CC} = 5V ±5%; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{IL1}	Input Low Voltage	−0.3	0.8	V	Note 1, V_{CC} = 4.75V
V_{IH1}	Input High Voltage	2.2	V_{CC} + 0.3	V	Note 1, V_{CC} = 5.25V
V_{IL2}	Input Low Voltage	−0.3	1.35	V	Note 2, V_{CC} = 4.75V
V_{IH2}	Input High Voltage	3.85	V_{CC} + 0.3	V	Note 2, V_{CC} = 5.25V
V_{T1}	Schmitt Trigger Threshold Voltage, Falling Edge	0.7	1.35	V	Note 3, V_{CC} = 5.0V
V_{T1+}	Schmitt Trigger Threshold Voltage, Falling Edge	1.4	2.2	V	Note 3, V_{CC} = 5.0V
V_{H1}	Hysteresis Voltage	0.3	1.2	V	Note 3, V_{CC} = 5.0V
V_{T2-}	Schmitt Trigger Threshold Voltage, Falling Edge	1.25	2.3	V	Note 3, V_{CC} = 5.0V
V_{T2+}	Schmitt Trigger Threshold Voltage, Rising Edge	2.3	3.7	V	Note 3, V_{CC} = 5.0V
V_{H2}	Hysterersis Voltage	0.3	1.2	V	Note 3, V_{CC} = 5.0V

Functional Operating Range (V_{CC} = 5V ±5%; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{OL1}	Output Low Voltage		0.5	V	Note 4
V_{OH1}	Output High Voltage	V_{CC} − 0.5		V	Note 4
V_{OL2}	Output Low Voltage		0.4	V	Note 5
V_{OH2}	Output High Voltage	2.4		V	Note 5
I_{OL1}	Output Low Current		1	mA	Note 6
I_{OH1}	Output High Current	−1		mA	Note 6
I_{OL2}	Output Low Current		3	mA	Note 7
I_{OH2}	Output High Current	−2		mA	Note 7
I_{OL3}	Output Low Current		6	mA	Note 8
I_{OH3}	Output High Current	−2		mA	Note 8
I_{OL4}	Output Low Current		3	mA	Note 9
I_{OH4}	Output High Current	−1		mA	Note 9
I_{IH}	Input Leakage Current		+10	uA	
I_{IL}	Input Leakage Current		−10	uA	
C_{IN}	Input Capacitance		12	pF	F_C = 1 MHz
C_{OUT}	Output Capacitance		12	pF	F_C = 1 MHz
$C_{I/O}$	I/O Capacitance		12	pF	F_C = 1 MHz

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: A[31:0], BE[7:0]#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, CACHE#, SMIACT#, PCLKIN, HCLKIN, HCLKOSC, FLSHBUF#, MEMCS#, SERR#, PERR#, MEMREQ#, GNT#, PLOCK#, STOP#, IRDY#, TRDY#, FRAME#, C/BE[3:0]#.
2. V_{IL2} and V_{IH2} apply to the following signals: PPOUT[1:0], EOL.
3. V_{T1-}, V_{T1+} and V_{H1} apply to PWROK. V_{T2-}, V_{T2+} and V_{H2} apply to TESTEN.
4. V_{OL1} and V_{OH1} apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, PCIRST#.
5. V_{OL2} and V_{OH2} apply to the following signals: REQ#, MEMACK#, FRAME#, C/BE[3:0]#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#, BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, A[31:0], PCLKOUT, HCLKA–HCLKF, CALE, COE[1:0]#, CWE[7:0]#, CADV[1:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], RAS[5:0]#, CAS[7:0]#, MA[10:0], WE#.
6. I_{OL1} and I_{OH1} apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, PCIRST#.
7. I_{OL2} and I_{OH2} apply to the following signals: C/BE[3:0]#, REQ#, MEMACK#, MA[10:0], WE#.
8. I_{OL3} and I_{OH3} apply to the following signals: FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#.
9. I_{OL4} and I_{OH4} apply to the following signals: BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, CPURST, INIT, A[31:0], PCLKOUT, CALE, COE[1:0]#, CADS[1:0]#, CADV[1:0]#, CWE[7:0]#, CAA[6:3], CAB[6:3], RAS[5:0]#, CAS[7:0]#.

9.4 82434NX DC Characteristics

Functional Operating Range (V_{CC} = 5V ±5%; $V_{CC}3$ = 3.135 to 3.465 V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{IL1}	Input Low Voltage	−0.3	0.8	V	Note 1, V_{CC} = 4.75V
V_{IH1}	Input High Voltage	2.2	V_{CC} + 0.3	V	Note 1, V_{CC} = 5.25V
V_{IL2}	Input Low Voltage	−0.3	1.35	V	Note 2, V_{CC} = 4.75V
V_{IH2}	Input High Voltage	3.85	V_{CC} + 0.3	V	Note 2, V_{CC} = 5.25V
V_{IL3}	Input Low Voltage	−0.3	0.8	V	Note 3, V_{ccq} = 3.135V
V_{IH3}	Input High Voltage	2.2	V_{CC} + 0.3	V	Note 3, V_{ccq} = 3.465V
V_{T1}	Schmitt Trigger Threshold Voltage, Falling Edge	0.7	1.35	V	Note 4, V_{CC} = 5.0V
V_{T1+}	Schmitt Trigger Threshold Voltage, Rising Edge	1.4	2.2	V	Note 4, V_{CC} = 5.0V
V_{H1}	Hysteresis Voltage	0.3	1.2	V	Note 4, V_{CC} = 5.0V
V_{T2-}	Schmitt Trigger Threshold Voltage, Falling Edge	1.25	2.3	V	Note 4, V_{CC} = 5.0V
V_{T2+}	Schmitt Trigger Threshold Voltage, Rising Edge	2.3	3.7	V	Note 4, V_{CC} = 5.0V
V_{H2}	Hysterersis Voltage	0.3	1.2	V	Note 4, V_{CC} = 5.0V
V_{OL1}	Output Low Voltage		0.5	V	Note 5
V_{OH1}	Output High Voltage	V_{CC} − 0.5		V	Note 5
V_{OL2}	Output Low Voltage		0.4	V	Note 6
V_{OH2}	Output High Voltage	2.4		V	Note 6
I_{OL1}	Output Low Current		1	mA	Note 7
I_{OH1}	Output High Current	−1		mA	Note 7
I_{OL2}	Output Low Current		3	mA	Note 8

Functional Operating Range (V_{CC} = 5V ±5%; $V_{CC}3$ = 3.135 to 3.465 V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Unit	Test Conditions
I_{OH2}	Output High Current	−2		mA	Note 8
I_{OL3}	Output Low Current		6	mA	Note 9
I_{OH3}	Output High Current	−2		mA	Note 9
I_{OL4}	Output Low Current		3	mA	Note 10
I_{OH4}	Output High Current	−1		mA	Note 10
I_{IH}	Input Leakage Current		+10	uA	
I_{IL}	Input Leakage Current		−10	uA	
C_{IN}	Input Capacitance		12	pF	F_C = 1 MHz
C_{OUT}	Output Capacitance		12	pF	F_C = 1 MHz
$C_{I/O}$	I/O Capacitance		12	pF	F_C = 1 MHz

2

NOTES:
1. V_{IL1} and V_{IH1} apply to the following signals: BE[7:0]#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, CACHE#, SMIACT#, PCLKIN, HCLKOSC, FLSHBUF#, MEMCS#, SERR#, PERR#, MEMREQ#, GNT#, PLOCK#, STOP#, IRDY#, TRDY#, FRAME#, C/BE[3:0]#.
2. V_{IL2} and V_{IH2} apply to the following signals: PPOUT[1:0], EOL.
3. VIL3 and VIH3 apply to the following signals: A[31:0], HCLKIN.
4. V_{T1-}, V_{T1+} and V_{H1} apply to PWROK. V_{T2-}, V_{T2+} and V_{H2} apply to TESTEN.
5. V_{OL1} and V_{OH1}apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, PCIRST#.
6. V_{OL2} and V_{OH2} apply to the following signals: REQ#, MEMACK#, FRAME#, C/BE[3:0]#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#, BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, A[31:0], PCLKOUT, HCLKA–HCLKF, CALE, COE[1:0]#, CWE[7:0]#, CADV[1:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], RAS[7:0]#, CAS[7:0]#, MA[11:0], WE#.
7. I_{OL1} and I_{OH1} apply to the following signals: HIG[4:0], MIG[2:0], PIG[3:0], DRVPCI, MDLE, A[31:8], A[2:0], PCIRST#.
8. I_{OL2} and I_{OH2} apply to the following signals: C/BE[3:0]#, REQ#, MEMACK#, MA[11:0], WE#.
9. I_{OL3} and I_{OH3} apply to the following signals: FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL#, PAR, PERR#, SERR#.
10. I_{OL4} and I_{OH4} apply to the following signals: BOFF#, AHOLD, BRDY#, NA#, EADS#, KEN#, INV, CPURST, INIT, A[7:3], PCLKOUT, CALE, COE[1:0]#, CADS[1:0]#, CADV[1:0]#, CWE[7:0]#, CAA[6:3], CAB[6:3], RAS[7:0]#, CAS[7:0]#.
11. The output buffers for BRDY#, NA#, AHOLD, EADS#, INV, BOFF#, KEN#, PEN#, CPURST, INIT, CALE, CADS[1:0], CADV[1:0]#, CAA[6:3], CAB[6:3], COE[1:0]#, CWE[7:0]#, A[31:3] AND HCLK[A:F] are powered with $V_{CC}3$ and therefore drive 3.3V signal levels.

9.5 82434LX AC Characteristics

The AC characteristics given in this section consist of propagation delays, valid delays, input setup requirements, input hold requirements, output float delays, output enable delays, output-to-output delays, pulse widths, clock high and low times and clock period specifications. Figure 77 through Figure 85 define these specifications. Section 9.5 lists the 82434LX AC Characteristics. Output test loads are listed in the right column.

In Figure 77 through Figure 85, VT = 1.5V for the following signals:

A[31:0], BE[7:0]#, PEN#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, EADS#, BRDY#, BOFF#, AHOLD, NA#, KEN#, INV, CACHE#, SMIACT#, INIT, CPURST, CALE, CADV[1:0]#, COE[1:0]#, CWE[7:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], WE#, RAS[5:0]#, CAS[7:0]#, MA[10:0], C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, GNT#, DEVSEL#, MEMREQ#, PAR, PERR#, SERR#, REQ#, MEMCS#, FLSHBUF#,MEMACK#, PWROK, HCLKIN, HCLKA–HCLKF, PCLKIN, PCLKOUT.

VT = 2.5V for the following signals:

PPOUT[1:0], EOL, HIG[4:0], PIG[3:0], MIG[2:0], DRVPCI, MDLE, PCIRST#.

9.5.1 HOST CLOCK TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Figure	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	15	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		85	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		85	0 pF

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.5.2 CPU INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Setup Time to HCLKIN Rising	4.6		79	
t10b	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Hold Time from HCLKIN Rising	0.8		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.5		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12aa	A[18:3] Falling Edge Setup Time to HCLKIN Rising	3.2		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ab	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.7			Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ac	A[18:3] Falling Edge Setup Time to HCLKIN Rising	4.1			Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5		79	Hold from HCLKIN rising two clocks after ADS# is sampled active by PCMC.
t12c	A[31:0] Setup Time to HCLKIN Rising	6.5		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	
t12h	A[2:0] Propagation Delay from BE[7:0]#	1	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	1.7	7.8	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	1.7	7.6	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	1.3	7.8	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	1.3	7.1	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	1.8	7.1	78	
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	1.3	7.4	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	0.9	7.5	78	
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	0.9	7.0	78	
t16d	KEN# Valid delay from HCLKIN Rising	1.3	7.6	78	
t17	INIT High Pulse Width	2 HCLKs		84	Soft reset via TRC register or CPU shutdown special cycle
t18	CPURST High Pulse Width	1 ms		84	Hard reset via TRC register, 0 pF

9.5.3 SECOND LEVEL CACHE STANDARD SRAM TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	14	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.4	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	1 HCLK		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	-1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	7.5	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	7.6	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF

9.5.4 SECOND LEVEL CACHE BURST SRAM TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.0	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	7.7	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	7.1	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.0	9.0	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.0	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	7.5	78	0 pF

9.5.5 DRAM INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[5:0]# Valid Delay from HCLKIN Rising	0	7.5	78	50 pF
t40b	RAS[5:0]# Pulse Width High	4 HCLKs − 5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	7.5	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLKIN − 5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:3]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.1	78	50 pF

9.5.6 PCI CLOCK TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.5.7 PCI INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, PLOCK#, PAR, PERR#, SERR#, Setup Time to PCLKIN Rising	7		79	
t60da	TRDY#, IRDY# Setup Time to PCLKIN Rising	8.1		77	
t60db	STOP#, DEVSEL# Setup Time to PCLKIN Rising	8.5		77	
t60e	C/BE[3:0]#, FRAME#, PLOCK#, PAR, PERR#, SERR# Hold Time from PCLKIN Rising	0		77	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register, 0 pF

9.5.8 LBX INTERFACE TIMING, 66 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.9V to 5.25V; T_{CASE} = 0°C to +70°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.5	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	0.7	10.9	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13.5	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	5.6	78	0 pF
t74b	MDLE Rising Edge Valid Delay from HCLKIN Rising	0.6	6.8	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

9.5.9 HOST CLOCK TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	16.66	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		82	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		82	0 pF

NOTE:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.5.10 CPU INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Setup Time to HCLKIN Rising	4.6		79	
t10b	ADS#, HITM#, W/R#, M/IO#, D/C#, HLOCK#, CACHE#, BE[7:0]#, SMIACT# Hold Time from HCLKIN Rising	1.1		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.5		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12aa	A[18:3] Falling Edge Setup Time to HCLKIN Rising	3.2		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ab	A[18:3] Rising Edge Setup Time to HCLKIN Rising	4.7		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12ac	A[18:3] Falling Edge Setup Time to HCLKIN Rising	4.1		79	Setup to HCLKIN Rising when ADS# is Sampled Active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5		79	Hold from HCLKIN rising two clocks after ADS# is sampled active by PCMC.
t12c	A[31:0] Setup Time to HCLKIN Rising	6.5		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12h	A[2:0] Propagation Delay from BE[7:0]#	1	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	1.4	8.4	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	2.0	7.6	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	2.0	7.6	78	
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	2.0	8.0	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	
t16d	KEN# Valid delay from HCLKIN Rising	1.7	8.2	78	
t17	INIT High Pulse Width	2 HCLKs		84	Soft reset via TRC register or CPU shutdown special cycle
t18	CPURST High Pulse Width	1 ms		84	Hard reset via TRC register, 0 pF

9.5.11 SECOND LEVEL CACHE STANDARD SRAM TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	15	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.4	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	1 HCLK		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	−1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	8	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF

9.5.12 SECOND LEVEL CACHE BURST SRAM TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	8.2	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.0	10.5	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.5	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	6.0	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	8.5	78	0 pF

9.5.13 DRAM INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[5:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t40b	RAS[5:0]# Pulse Width High	4 HCLKs−5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLK−5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:3]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.7	78	50 pF

9.5.14 PCI CLOCK TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.5.15 PCI INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Setup Time to PCLKIN Rising	9		79	
t60e	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Hold Time from PCLKIN Rising	0		79	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register, 0 pF

9.5.16 LBX INTERFACE TIMING, 60 MHz (82434LX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.7	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	1.5	12	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	6.8	78	0 pF
t74b	MDLE Rising Edge Valid Delay from HCLKIN Rising	0.6	6.8	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

9.6 82434NX AC Characteristics

The AC characteristics given in this section consist of propagation delays, valid delays, input setup requirements, input hold requirements, output float delays, output enable delays, output-to-output delays, pulse widths, clock high and low times and clock period specifications. Figure 77 through Figure 85 define these specifications. Output test loads are listed in the right column.

In Figure 77 through Figure 85, VT = 1.5V for the following signals:

A[31:0], BE[7:0]#, PEN#, D/C#, W/R#, M/IO#, HLOCK#, ADS#, PCHK#, HITM#, EADS#, BRDY#, BOFF#, AHOLD, NA#, KEN#, INV, CACHE#, SMIACT#, INIT, CPURST, CALE, CADV[1:0]#, COE[1:0]#, CWE[7:0]#, CADS[1:0]#, CAA[6:3], CAB[6:3], WE#, RAS[5:0]#, CAS[7:0]#, MA[10:0], C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, GNT#, DEVSEL#, MEMREQ#, PAR, PERR#, SERR#, REQ#, MEMCS#, FLSHBUF#, MEMACK#, PWROK, HCLKIN, HCLKA–HCLKF, PCLKIN, PCLKOUT.

VT = 2.5V for the following signals:

PPOUT[1:0], EOL, HIG[4:0], PIG[3:0], MIG[2:0], DRVPCI, MDLE, PCIRST#

2

9.6.1 HOST CLOCK TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	15	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		82	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		82	0 pF

NOTES:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.6.2 CPU INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, W/R#, Setup Time to HCLKIN Rising	4.6		79	
t10b	BE[7:0]# Setup Time to HCLKIN Rising	4.6		79	
t10c	HITM# Setup Time to HCLKIN Rising	6.4		79	
t10d	CACHE#, M/IO# Setup Time to HCLKIN Rising	4.6		79	
t10e	D/C# Setup Time to HCLKIN Rising	4.0		79	
t10f	HLOCK#, SMIACT#, Setup Time to HCLKIN Rising	4.0		79	
t10g	HITM#, M/IO#, D/C#, Hold Time from HCLKIN Rising	0.7		79	
t10h	W/R#, HLOCK#, Hold Time from HCLKIN Rising	0.8		79	
t10i	ADS#, BE[7:0]# Hold Time from HCLKIN Rising	1.1		79	
t10j	CACHE#, SMIACT# Hold Time from HCLKIN Rising	1.1		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[31:0] Setup Time to HCLKIN Rising	2.7		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5			HOLD from HCLKIN Rising two clocks after ADS# is sampled active by PCMC
t12c	A[31:0] Setup Time to HCLKIN Rising	6.0		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	
t12h	A[2:0] Propagation Delay from BE[7:0]#	1.0	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	1.6	7.5	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	1.6	7.5	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	.9	7.6	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	1.5	7.0	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	1.5	7.0	78	0 pF
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	1.5	7.5	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	1.2	7.0	78	0 pF
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	1.2	7.0	78	0 pF
t16d	KEN# Valid delay from HCLKIN Rising	1.5	7.5	78	0 pF
t17	INIT High Pulse Width	2 HCLKs		84	0 pF
t18	CPURST High Pulse Width	1 ms		84	0 pF; Hard reset via TRC register

9.6.3 SECOND LEVEL CACHE STANDARD SRAM TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	14	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.0	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	1 HCLK		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	−1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	8.0	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF
t26a	CCS[1:0]# Propagation Delay from ADS# Falling		7.0	77	0 pF; First access after powerdown
t26b	CCS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF; Entering powerdown

9.6.4 SECOND LEVEL CACHE BURST SRAM TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	8.2	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.0	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	8.0	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.5	9.0	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0.5	9.0	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0.5	6.0	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	8.0	78	0 pF

2

9.6.5 DRAM INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t40b	RAS[7:0]# Pulse Width High	4 HCLKs−5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLKIN−5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:3]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.7	78	50 pF
t43c	MA11 Propagation Delay from A[25:24]	0	28.0	77	50 pF
t43d	MA11 Valid Delay from HCLKIN Rising	0	12	78	50 pF

9.6.6 PCI CLOCK TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.6.7 PCI INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Setup Time to PCLKIN Rising	7		79	
t60e	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Hold Time from PCLKIN Rising	0		79	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C) (Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register,0 pF

9.6.8 LBX INTERFACE TIMING, 66 MHz (82434NX), PRELIMINARY

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.5	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	1.5	12	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	6.0	78	0 pF
t74b	MDLE Rising Edge Valid from HCLKIN Rising	0.6	6.0	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

9.6.9 HOST CLOCK TIMING, 50 and 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t1a	HCLKOSC High Time	6.0		82	
t1b	HCLKOSC Low Time	5.0		82	
t2a	HCLKIN Period	16.66	20	82	
t2b	HCLKIN Period Stability		±100		ps(1)
t2c	HCLKIN High Time	4		82	
t2d	HCLKIN Low Time	4		82	
t2e	HCLKIN Rise Time		1.5	83	
t2f	HCLKIN Fall Time		1.5	83	
t3a	HCLKA–HCLKF Output-to-Output Skew		0.5	85	0 pF
t3b	HCLKA–HCLKF High Time	5.0		82	0 pF
t3c	HCLKA–HCLKF Low Time	5.0		82	0 pF

NOTES:
1. Measured on rising edge of adjacent clocks at 1.5V.

9.6.10 CPU INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t10a	ADS#, W/R#, Setup Time to HCLKIN Rising	4.6		79	
t10b	BE[7:0]# Setup Time to HCLKIN Rising	4.6		79	
t10c	HITM# Setup Time to HCLKIN Rising	6.8		79	
t10d	CACHE#, M/IO# Setup Time to HCLKIN Rising	4.6		79	
t10e	D/C# Setup Time to HCLKIN Rising	4.6		79	
t10f	HLOCK#, SMIACT#, Setup Time to HCLKIN Rising	4.6		79	
t10g	HITM#, M/IO#, D/C#, Hold Time from HCLKIN Rising	0.7		79	
t10h	W/R#, HLOCK# Hold from HCLKIN Rising	0.8		79	
t10i	ADS#, BE[7:0]# Hold Time from HCLKIN Rising	0.9		79	
t10j	CACHE#, SMIACT# Hold Time from HCLKIN Rising	1.1		79	
t11a	PCHK# Setup Time to HCLKIN Rising	4.3		79	
t11b	PCHK# Hold Time from HCLKIN Rising	1.1		79	
t12a	A[31:0] Setup Time to HCLKIN Rising	3.0		79	Setup to HCLKIN rising when ADS# is sampled active by PCMC.
t12b	A[31:0] Hold Time from HCLKIN Rising	0.5		79	HOLD from HCLKIN Rising two clocks after ADS# is sampled active by PCMC
t12c	A[31:0] Setup Time to HCLKIN Rising	6.5		79	Setup to HCLKIN rising when EADS# is sampled active by the CPU.
t12d	A[31:0] Hold Time from HCLKIN Rising	1.5		79	Hold from HCLKIN rising when EADS# is sampled active by the CPU.

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)
(Continued)

Symbol	Parameter	Min	Max	Fig	Notes
t12e	A[31:0] Output Enable from HCLKIN Rising	0	13	81	
t12f	A[31:0] Valid Delay from HCLKIN Rising	1.3	13	78	0 pF
t12g	A[31:0] Float Delay from HCLKIN Rising	0	13	80	
t12h	A[2:0] Propagation Delay from BE[7:0]#	1.0	16	77	0 pF
t13a	BRDY# Rising Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t13b	BRDY# Falling Edge Valid Delay from HCLKIN Rising	2.1	7.9	78	0 pF
t14	NA# Valid Delay from HCLKIN Rising	1.4	8.4	78	0 pF
t15a	AHOLD Valid Delay from HCLKIN Rising	2.0	7.6	78	0 pF
t15b	BOFF# Valid Delay from HCLKIN Rising	2.0	7.6	78	0 pF
t16a	EADS#, INV, PEN# Valid Delay from HCLKIN Rising	2.0	8.0	78	0 pF
t16b	CPURST Rising Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	0 pF
t16c	CPURST Falling Edge Valid Delay from HCLKIN Rising	1.2	7.5	78	0 pF
t16d	KEN# Valid delay from HCLKIN Rising	1.7	8.2	78	0 pF
t17	INIT High Pulse Width	2 HCLKs		84	0 pF
t18	CPURST High Pulse Width	1 ms		84	0 pF; Hard reset via TRC register

9.6.11 SECOND LEVEL CACHE STANDARD SRAM TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t20a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t20b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	7.2	78	0 pF
t21a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.0	78	0 pF
t21b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	5.5	78	0 pF
t22a	CWE[7:0]#/CBS[7:0]# Falling Edge Valid Delay from HCLKIN Rising	2	14	78	CPU burst or single write to second level cache, 0 pF
t22b	CWE[7:0]#/CBS[7:0]# Rising Edge Valid Delay from HCLKIN Rising	3	15	78	CPU burst or single write to second level cache, 0 pF
t22c	CWE[7:0]#/CBS[7:0]# Valid Delay from HCLKIN Rising	1.4	7.7	78	Cache line Fill, 0 pF
t22d	CWE[7:0]#/CBS[7:0]# Low Pulse Width	14		84	0 pF
t22e	CWE[7:0]#/CBS[7:0]# Driven High before CALE Driven High	-1		85	Last write to second level cache during cache line fill, 0 pF
t22f	CAA[4:3]/CAB[4:3] Valid before CWE[7:0]# Falling	1.5		85	CPU burst write to second level cache, 0 pF
t23	CALE Valid Delay from HCLKIN Rising	0	8	78	0 pF
t24	CR/W[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t25	CBS[1:0]# Valid Delay from HCLKIN Rising; Reads from Cache SRAMs	1.0	12.0	78	0 pF
t26a	CCS[1:0]# Propagation Delay from ADS# Falling		7.0	77	0 pF; First access after powerdown
t26b	CCS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF; Entering powerdown

9.6.12 SECOND LEVEL CACHE BURST SRAM TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t30a	CAA[6:3]/CAB[6:3] Propagation Delay from A[6:3]	0	8.5	77	0 pF
t30b	CAA[6:3]/CAB[6:3] Valid Delay from HCLKIN Rising	0	8.2	78	0 pF
t31	CADS[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t32	CADV[1:0]# Valid Delay from HCLKIN Rising	1.5	8.2	78	0 pF
t33	CWE[7:0]# Valid Delay from HCLKIN Rising	1.0	10.5	78	0 pF
t34a	COE[1:0]# Falling Edge Valid Delay from HCLKIN Rising	0	9.5	78	0 pF
t34b	COE[1:0]# Rising Edge Valid Delay from HCLKIN Rising	0	6.0	78	0 pF
t35	CALE Valid Delay from HCLKIN Rising	0	8.5	78	0 pF

2

9.6.13 DRAM INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t40a	RAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t40b	RAS[7:0]# Pulse Width High	4 HCLKs–5		84	RAS# precharge at beginning of page miss cycle, 50 pF
t41a	CAS[7:0]# Valid Delay from HCLKIN Rising	0	8.0	78	50 pF
t41b	CAS[7:0]# Pulse Width High	1 HCLK–5		84	CAS# precharge during burst cycles, 50 pF
t42	WE# Valid Delay from HCLKIN Rising	0	21	78	50 pF
t43a	MA[10:0] Propagation Delay from A[23:3]	0	23	77	50 pF
t43b	MA[10:0] Valid Delay from HCLKIN Rising	0	10.7	78	50 pF
t43c	MA11 Propagation Delay from A[25:24]	0	24.3	77	50 pF
t43d	MA11 Valid Delay from HCLKIN Rising	0	12	78	50 pF

9.6.14 PCI CLOCK TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t50a	PCLKOUT High Time	13		82	20 pF
t50b	PCLKOUT Low Time	13		82	20 pF
t51a	PCLKIN High Time	12		82	
t51b	PCLKIN Low Time	12		82	
t51c	PCLKIN Rise Time		3	83	
t51d	PCLKIN Fall Time		3	83	

9.6.15 PCI INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t60a	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Valid Delay from PCLKIN Rising	2	11	78	Min: 0 pF Max: 50 pF
t60b	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Output Enable Delay from PCLKIN Rising	2		81	
t60c	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Float Delay from PCLKIN Rising	2	28	80	
t60d	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Setup Time to PCLKIN Rising	9		79	
t60e	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, PAR, PERR#, SERR#, DEVSEL# Hold Time from PCLKIN Rising	0		79	
t61a	REQ#, MEMACK# Valid Delay from PCLKIN Rising	2	12	78	Min: 0 pF Max: 50 pF
t61b	REQ#, MEMACK# Output Enable Delay from PCLKIN Rising	2		81	
t61c	REQ#, MEMACK# Float Delay from PCLKIN Rising	2	28	80	
t62a	FLSHREQ#, MEMREQ# Setup Time to PCLKIN Rising	12		79	
t62b	FLSHREQ#, MEMREQ# Hold Time from PCLKIN Rising	0		79	
t63a	GNT# Setup Time to PCLKIN Rising	10		79	
t63b	GNT# Hold Time from PCLKIN Rising	0		79	
t64a	MEMCS# Setup Time to PCLKIN Rising	7		79	
t64b	MEMCS# Hold Time from PCLKIN Rising	0		79	
t65	PCIRST# Low Pulse Width	1 ms		84	Hard Reset via TRC Register, 0 pF

9.6.16 LBX INTERFACE TIMING, 50 AND 60 MHz (82434NX)

Functional Operating Range (V_{CC} = 4.75V to 5.25V; $V_{CC}3$ = 3.135V to 3.465V; T_{CASE} = 0°C to +85°C)

Symbol	Parameter	Min	Max	Fig	Notes
t70	HIG[4:0] Valid Delay from HCLKIN Rising	0.8	6.7	78	0 pF
t71	MIG[2:0] Valid Delay from HCLKIN Rising	0.9	6.5	78	0 pF
t72	PIG[3:0] Valid Delay from PCLKIN Rising	1.5	12	78	0 pF
t73	PCIDRV Valid Delay from PCLKIN Rising	1	13	78	0 pF
t74a	MDLE Falling Edge Valid Delay from HCLKIN Rising	0.6	6.8	78	0 pF
t74b	MDLE Rising Edge Valid Delay from HCLKIN Rising	0.6	6.8	85	0 pF
t75a	EOL, PPOUT[1:0] Setup Time to PCLKIN Rising	7.7		79	
t75b	EOL, PPOUT[1:0] Hold Time from PCLKIN Rising	1.0		79	

2

9.6.17 TIMING DIAGRAMS

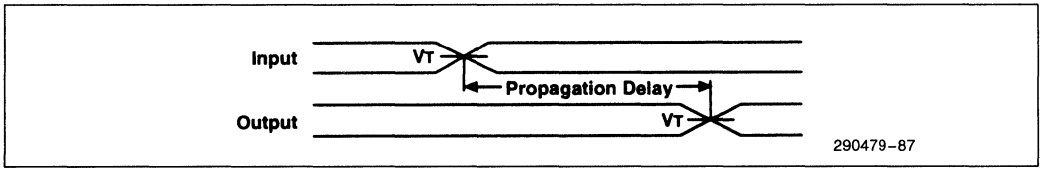


Figure 77. Propagation Delay

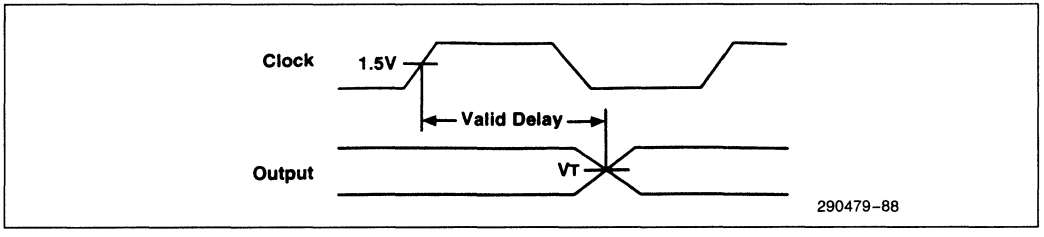


Figure 78. Valid Delay from Rising Clock Edge

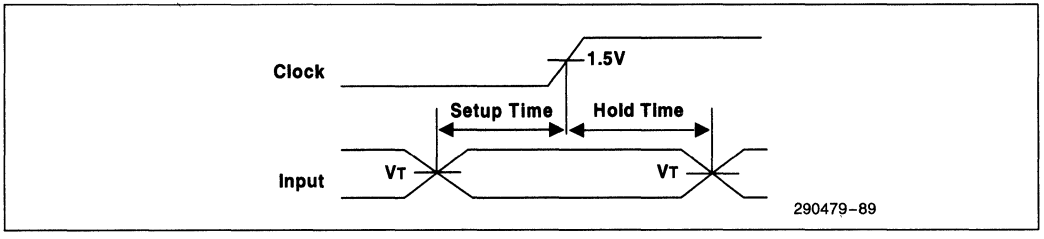


Figure 79. Setup and Hold Times

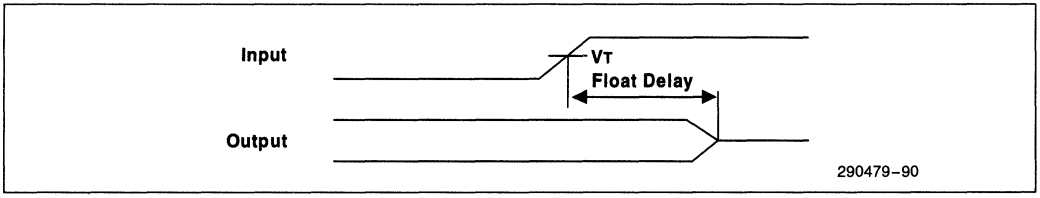


Figure 80. Float Delay

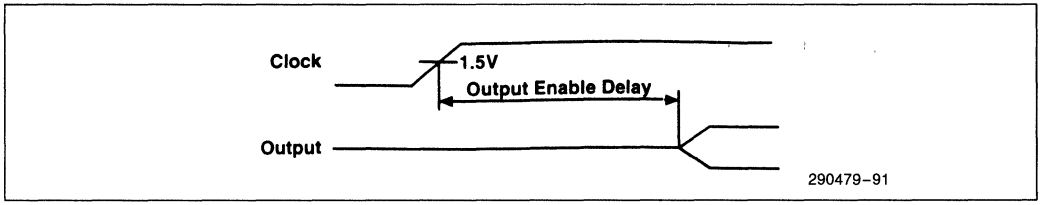


Figure 81. Output Enable Delay

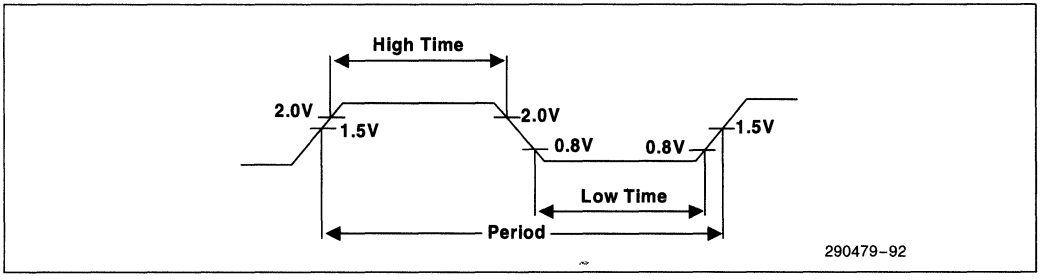


Figure 82. Clock High and Low Times and Period

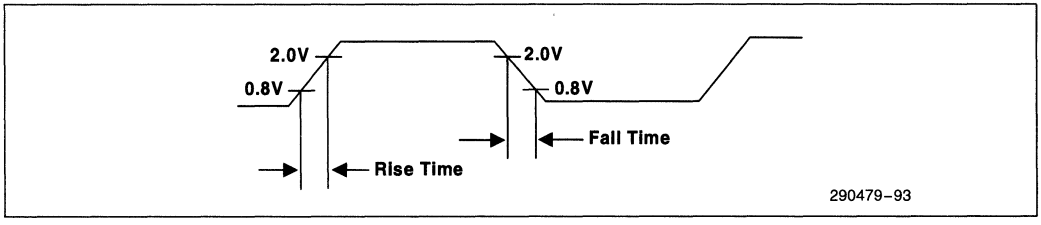


Figure 83. Clock Rise and Fall Times

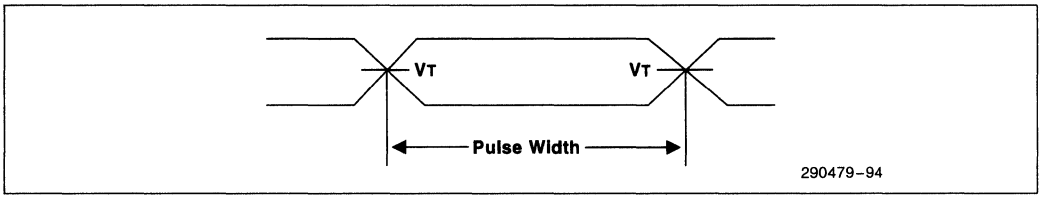


Figure 84. Pulse Width

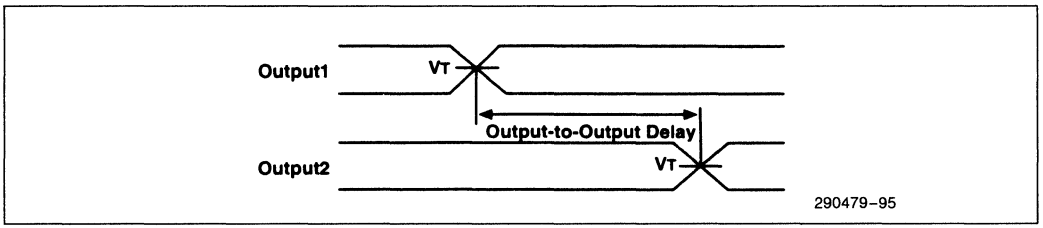
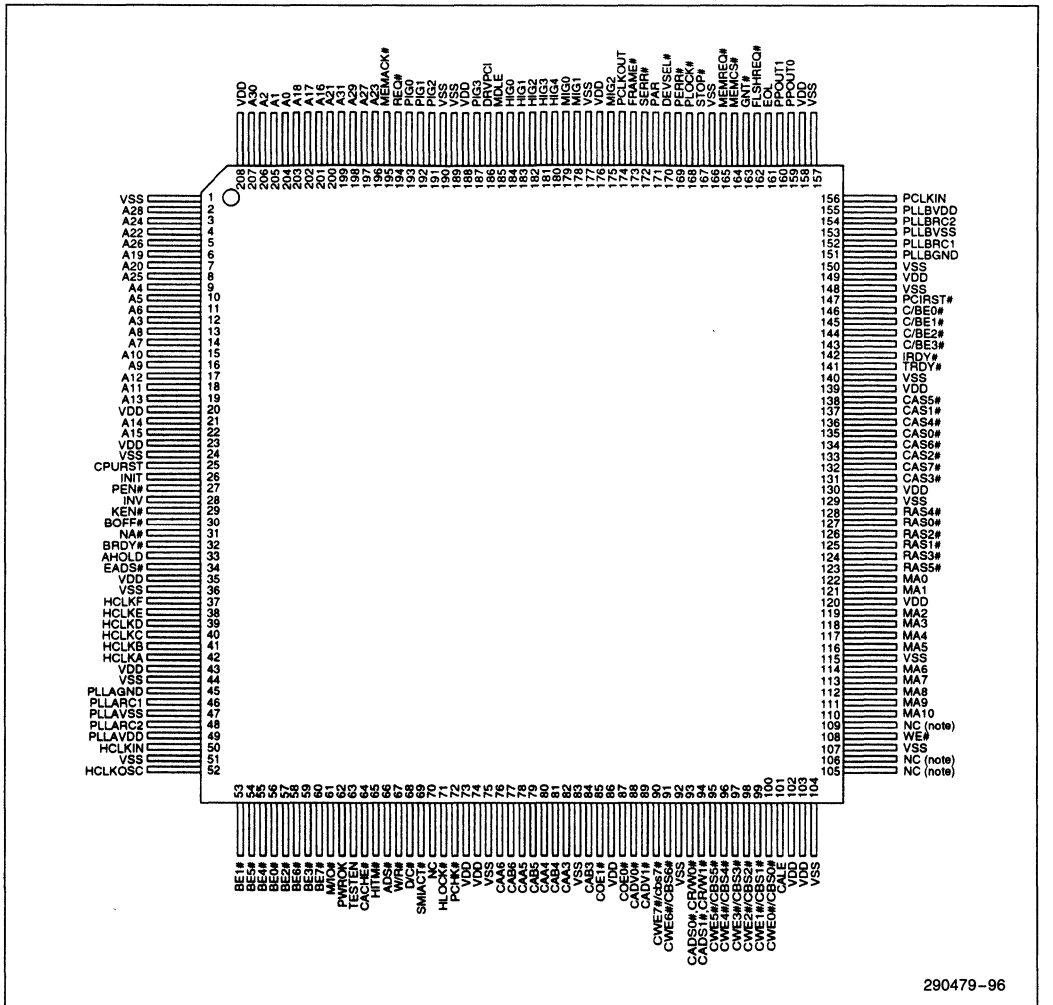


Figure 85. Output-to-Output Delay

10.0 PINOUT AND PACKAGE INFORMATION

10.1 Pin Assignment

Except for the pins listed in Figure 86 notes, the pin assignment for the 82434LX and 82434NX are the same.



290479-96

NOTES:
1. For the 82434NX, pin 105 = RAS6#, 106 = RAS7#, and 109 = MA11. These pins are no connects for the 82434LX and are signal connections for the 82434NX.
2. For the 82434NX, pins 23, 35, 43, 74, 86, and 102 are 3.3V VDD pins (i.e., VDD3). These pins are VDD pins for the 82434LX.

Figure 86. PCMC Pin Assignment

Table 21. 82434LX Alphabetical Pin Assignment

Pin Name	Pin #	Type
A0	204	t/s
A1	205	t/s
A2	206	t/s
A3	12	t/s
A4	9	t/s
A5	10	t/s
A6	11	t/s
A7	14	t/s
A8	13	t/s
A9	16	t/s
A10	15	t/s
A11	18	t/s
A12	17	t/s
A13	19	t/s
A14	21	t/s
A15	22	t/s
A16	201	t/s
A17	202	t/s
A18	203	t/s
A19	6	t/s
A20	7	t/s
A21	200	t/s
A22	4	t/s
A23	196	t/s
A24	3	t/s
A25	8	t/s
A26	5	t/s
A27	197	t/s
A28	2	t/s
A29	198	t/s
A30	207	t/s
A31	199	t/s
ADS#	66	in

Pin Name	Pin #	Type
AHOLD	33	out
BE0#	56	in
BE1#	53	in
BE2#	57	in
BE3#	59	in
BE4#	55	in
BE5#	54	in
BE6#	58	in
BE7#	60	in
BOFF#	30	out
BRDY#	32	out
CAA3	82	out
CAA4	80	out
CAA5	78	out
CAA6	76	out
CAB3	84	out
CAB4	81	out
CAB5	79	out
CAB6	77	out
CACHE#	64	in
CADS0#,CR/W0#	93	out
CADS1#,CR/W1#	94	out
CADV0# (82434LX) CADV0#/CCS0# (82434NX)	88	out
CADV1# (82434LX) CADV1#/CCS1# (82434NX)	89	out
CALE	101	out
CAS0#	135	out
CAS1#	137	out
CAS2#	133	out
CAS3#	131	out
CAS4#	136	out

Pin Name	Pin #	Type
CAS5#	138	out
CAS6#	134	out
CAS7#	132	out
CBE0#	146	t/s
CBE1#	145	t/s
CBE2#	144	t/s
CBE3#	143	t/s
COE0#	87	out
COE1#	85	out
CPURST	25	out
CWE0#/CBS0#	100	out
CWE1#/CBS1#	99	out
CWE2#/CBS2#	98	out
CWE3#/CBS3#	97	out
CWE4#/CBS4#	96	out
CWE5#/CBS5#	95	out
CWE6#/CBS6#	91	out
CWE7#/CBS7#	90	out
D/C#	68	in
DEVSEL#	170	s/t/s
DRVPCI	186	out
EADS#	34	out
EOL	161	in
FLSHREQ#	162	in
FRAME#	173	s/t/s
GNT#	163	in
HCLKA	42	out
HCLKB	41	out
HCLKC	40	out
HCLKD	39	out
HCLKE	38	out
HCLKF	37	out
HCLKIN	50	in

Table 21. 82434LX Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
HCLKOSC	52	in
HIG0	184	out
HIG1	183	out
HIG2	182	out
HIG3	181	out
HIG4	180	out
HITM#	65	in
HLOCK#	71	in
INIT	26	out
INV	28	out
IRDY#	142	s/t/s
KEN#	29	out
M/IO#	61	in
MA0	122	out
MA1	121	out
MA2	119	out
MA3	118	out
MA4	117	out
MA5	116	out
MA6	114	out
MA7	113	out
MA8	112	out
MA9	111	out
MA10	110	out

Pin Name	Pin #	Type
MA11 (82434NX only)	109	out
MDLE	185	out
MEMACK#	195	out
MEMCS#	164	in
MEMREQ#	165	in
MIG0	179	out
MIG1	178	out
MIG2	175	out
NA#	31	out
NC	70	NC
NC (82434LX only)	105	NC
NC (82434LX only)	106	NC
NC (82434LX only)	109	NC
PAR	171	t/s
PCHK#	72	in
PCIRST#	147	out
PCLKIN	156	in
PCLKOUT	174	out
PEN#	27	out
PERR#	169	s/o/d
PIG0	193	out
PIG1	192	out
PIG2	191	out
PIG3	187	out

Pin Name	Pin #	Type
PLLAGND	45	V
PLLARC1	46	in
PLLARC2	48	in
PLLAVDD	49	V
PLLAVSS	47	V
PLLBGND	151	V
PLLBRC1	152	in
PLLBRC2	154	in
PLLBVDD	155	V
PLLBVSS	153	V
PLOCK#	168	s/t/s
PPOUT0	159	in
PPOUT1	160	in
PWROK	62	in
RAS0#	127	out
RAS1#	125	out
RAS2#	126	out
RAS3#	124	out
RAS4#	128	out
RAS5#	123	out
RAS6# (82434NX only)	105	out
RAS7# (82434NX only)	106	out

Table 21. 82434LX Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
REQ#	194	out
SERR#	172	s/o/d
SMIACT#	69	in
STOP#	167	s/t/s
TESTEN	63	in
TRDY#	141	s/t/s
V_{DD}	20	V
V_{DD} (82434LX) V_{DD3} (82434NX)	23	V
V_{DD} (82434LX) V_{DD3} (82434NX)	35	V
V_{DD} (82434LX) V_{DD3} (82434NX)	43	V
V_{DD}	73	V
V_{DD} (82434LX) V_{DD3} (82434NX)	74	V
V_{DD} (82434LX) V_{DD3} (82434NX)	86	V
V_{DD} (82434LX) V_{DD3} (82434NX)	102	V

Pin Name	Pin #	Type
V_{DD}	103	V
V_{DD}	120	V
V_{DD}	130	V
V_{DD}	139	V
V_{DD}	149	V
V_{DD}	158	V
V_{DD}	176	V
V_{DD}	188	V
V_{DD}	208	V
V_{SS}	1	V
V_{SS}	24	V
V_{SS}	36	V
V_{SS}	44	V
V_{SS}	51	V
V_{SS}	75	V
V_{SS}	83	V

Pin Name	Pin #	Type
V_{SS}	92	V
V_{SS}	104	V
V_{SS}	107	V
V_{SS}	115	V
V_{SS}	129	V
V_{SS}	140	V
V_{SS}	148	V
V_{SS}	150	V
V_{SS}	157	V
V_{SS}	166	V
V_{SS}	177	V
V_{SS}	189	V
V_{SS}	190	V
W/R#	67	in
WE#	108	out

Table 22. Numerical Pin Assignment

Pin #	Pin Name	Type
1	V_{SS}	V
2	A28	t/s
3	A24	t/s
4	A22	t/s
5	A26	t/s
6	A19	t/s
7	A20	t/s
8	A25	t/s
9	A4	t/s
10	A5	t/s
11	A6	t/s
12	A3	t/s
13	A8	t/s
14	A7	t/s
15	A10	t/s
16	A9	t/s
17	A12	t/s
18	A11	t/s
19	A13	t/s
20	V_{DD}	V
21	A14	t/s
22	A15	t/s
23	V_{DD} (82434LX) V_{DD3} (82434NX)	V
24	V_{SS}	V
25	CPURST	out
26	INIT	out
27	PEN#	out
28	INV	out
29	KEN#	out
30	BOFF#	out
31	NA#	out
32	BRDY#	out
33	AHOLD	out
34	EADS#	out
35	V_{DD} (82434LX) V_{DD3} (82434NX)	V
36	V_{SS}	V
37	HCLKF	out
38	HCLKE	out
39	HCLKD	out
40	HCLKC	out
41	HCLKB	out
42	HCLKA	out
43	V_{DD} (82434LX) V_{DD3} (82434NX)	V
44	V_{SS}	V
45	PLLAGND	V
46	PLLARC1	in
47	PLLAVSS	V
48	PLLARC2	in
49	PLLAVDD	V
50	HCLKIN	in
51	V_{SS}	V
52	HCLKOSC	in
53	BE1#	in
54	BE5#	in
55	BE4#	in
56	BE0#	in
57	BE2#	in
58	BE6#	in
59	BE3#	in
60	BE7#	in
61	M/IO#	in
62	PWROK	in
63	TESTEN	in
64	CACHE#	in
65	HITM#	in
66	ADS#	in
67	W/R#	in
68	D/C#	in
69	SMIACT#	in
70	NC	NC
71	HLOCK#	in
72	PCHK#	in
73	V_{DD}	V
74	V_{DD} (82434LX) V_{DD3} (82434NX)	V
75	V_{SS}	V
76	CAA6	out
77	CAB6	out
78	CAA5	out
79	CAB5	out
80	CAA4	out
81	CAB4	out
82	CAA3	out
83	V_{SS}	V
84	CAB3	out
85	COE1#	out
86	V_{DD} (82434LX) V_{DD3} (82434NX)	V
87	COE0#	out
88	CADV0# (82434LX) CADV0#/CCS0# (82434NX)	out
89	CADV1# (82434LX) CADV1#/CCS1# (82434NX)	out

Table 22. Numerical Pin Assignment (Continued)

Pin #	Pin Name	Type
90	CWE7#/CBS7#	out
91	CWE6#/CBS6#	out
92	V_{SS}	V
93	CADS0#,CR/W0#	out
94	CADS1#,CR/W1#	out
95	CWE5#/CBS5#	out
96	CWE4#/CBS4#	out
97	CWE3#/CBS3#	out
98	CWE2#/CBS2#	out
99	CWE1#/CBS1#	out
100	CWE0#/CBS0#	out
101	CALE	out
102	V_{DD} (82434LX) V_{DD3} (82434NX)	V
103	V_{DD}	V
104	V_{SS}	V
105	NC (82434LX) RAS6# (82434NX)	NC out
106	NC (82434LX) RAS7# (82434NX)	NC out
107	V_{SS}	V
108	WE#	out
109	NC (82434LX) MA11 (82434NX)	NC out
110	MA10	out
111	MA9	out
112	MA8	out
113	MA7	out
114	MA6	out
115	V_{SS}	V
116	MA5	out
117	MA4	out
118	MA3	out

Pin #	Pin Name	Type
119	MA2	out
120	V_{DD}	V
121	MA1	out
122	MA0	out
123	RAS5#	out
124	RAS3#	out
125	RAS1#	out
126	RAS2#	out
127	RAS0#	out
128	RAS4#	out
129	V_{SS}	V
130	V_{DD}	V
131	CAS3#	out
132	CAS7#	out
133	CAS2#	out
134	CAS6#	out
135	CAS0#	out
136	CAS4#	out
137	CAS1#	out
138	CAS5#	out
139	V_{DD}	V
140	V_{SS}	V
141	TRDY#	s/t/s
142	IRDY#	s/t/s
143	CBE3#	t/s
144	CBE2#	t/s
145	CBE1#	t/s
146	CBE0#	t/s
147	PCIRST#	out
148	V_{SS}	V
149	V_{DD}	V
150	V_{SS}	V

Pin #	Pin Name	Type
151	PLLBGND	V
152	PLLBRC1	in
153	PLLBVSS	V
154	PLLBRC2	in
155	PLLBVDD	V
156	PCLKIN	in
157	V_{SS}	V
158	V_{DD}	V
159	PPOUT0	in
160	PPOUT1	in
161	EOL	in
162	FLSHREQ#	in
163	GNT#	in
164	MEMCS#	in
165	MEMREQ#	in
166	V_{SS}	V
167	STOP#	s/t/s
168	PLOCK#	s/t/s
169	PERR#	s/o/d
170	DEVSEL#	s/t/s
171	PAR	t/s
172	SERR#	s/o/d
173	FRAME#	s/t/s
174	PCLKOUT	out
175	MIG2	out
176	V_{DD}	V
177	V_{SS}	V
178	MIG1	out
179	MIG0	out
180	HIG4	out
181	HIG3	out
182	HIG2	out

Table 22. Numerical Pin Assignment (Continued)

Pin #	Pin Name	Type
183	HIG1	out
184	HIG0	out
185	MDLE	out
186	DRVPCI	out
187	PIG3	out
188	V_{DD}	V
189	V_{SS}	V
190	V_{SS}	V
191	PIG2	out

Pin #	Pin Name	Type
192	PIG1	out
193	PIG0	out
194	REQ#	out
195	MEMACK#	out
196	A23	t/s
197	A27	t/s
198	A29	t/s
199	A31	t/s
200	A21	t/s

Pin #	Pin Name	Type
201	A16	t/s
202	A17	t/s
203	A18	t/s
204	A0	t/s
205	A1	t/s
206	A2	t/s
207	A30	t/s
208	V_{DD}	V

10.2 Package Characteristics

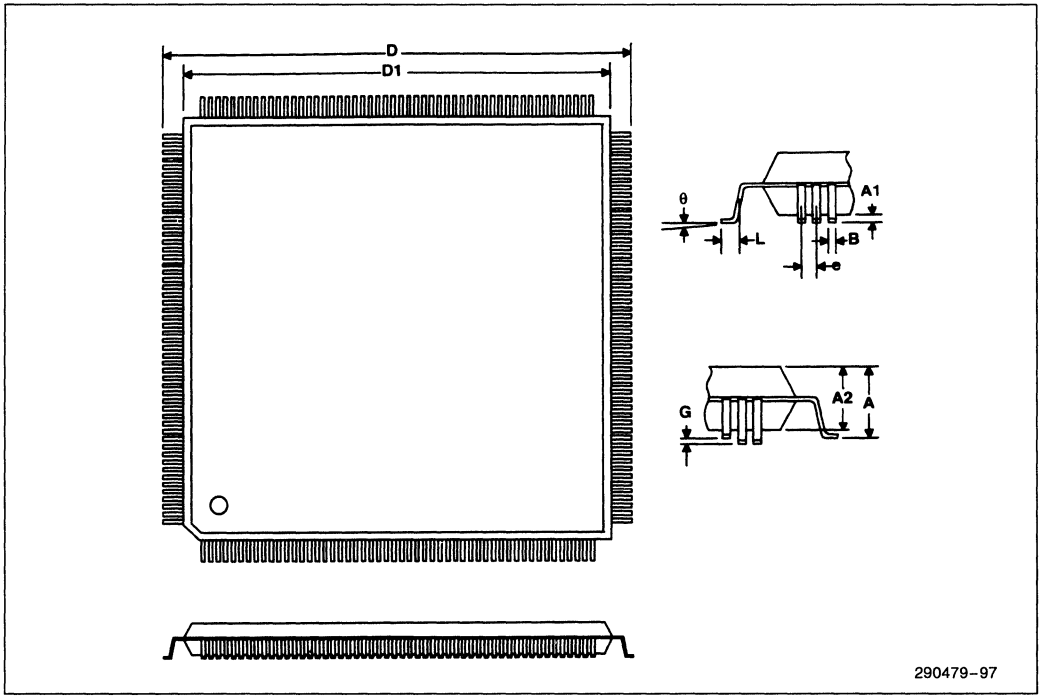


Figure 87. 208-Pin Quad Flatpack (QFP) Dimensions

Table 23. 82434LX Package Dimensions

Symbol	Description	Value (mm)
A	Seating Height	3.5 (max)
A1	Stand-Off Height	0.20–0.50
A2	Package Height	3.0 (nominal)
B	Lead Width	0.18 +0.1/−0.05
D	Package Length and Width, Including Pins	30.6 ± 0.3
D1	Package Length and Width, Excluding Pins	28 ± 0.1
e	Linear Lead Pitch	0.5 ± 0.1
G	Lead Coplanarity	0.1 (max)
L	Lead Length	0.5 ± 0.2
θ	Lead Angle	0°–10°

Table 24. 82434NX Package Dimensions

Symbol	Description	Value (mm)
A	Seating Height	3.7 (max)
A1	Stand-Off Height	0.05–0.50
A2	Package Height	3.45 (max)
B	Lead Width	0.13–0.27
D	Package Length and Width, Including Pins	30.6 ± 0.3
D1	Package Length and Width, Excluding Pins	28 ± 0.1
e	Linear Lead Pitch	0.5 (nominal)
G	Lead Coplanarity	0.1 (max)
L	Lead Length	0.5 ± 0.2
θ	Lead Angle	0°–10°

11.0 TESTABILITY

A NAND tree is provided in the 82434LX and 82434NX PCMCs for Automated Test Equipment (ATE) board level testing. The NAND tree allows the tester to test the connectivity of a subset of the PCMC signal pins.

For the 82434LX, the output of the NAND tree is driven on pin 109. The NAND tree is enabled when A24 = 1, A25 = 0, A26 = 1, and TESTEN = 1 at the rising edge of PWROK. PLL Bypass mode is enabled when A24 = 1, and TESTEN = 1 at the rising edge of PWROK. In PLL Bypass mode, the 82434LX and 82434NX PCMC AC specifications are affected as follows:

1. Output valid delays increase by 20 ns.
2. All hold times are 20 ns.
3. Setup times and propagation delays are unaffected.
4. Input clock high and low times are 100 ns.

In both the NAND tree test mode and PLL Bypass mode, TESTEN must remain asserted throughout the testing. A[28:24] should be set up at least 1 HCLK before the rising edge of PWROK and held at least 3 HCLKs after PWROK. Table 11 shows the order of the NAND tree inside the PCMC.

When not in NAND Tree test mode, the 82434LX drives the output of the host clock PLL onto pin 109.

82434NX Test Modes

The state of A[28:24], TESTEN, CPURST, and PWROK can place the 82434NX PCMC into two test modes. When PWROK is low, A[27:24] and TESTEN directly control the mode of operation of the PCMC. When PWROK is high, the state of A[27:24] and TESTEN are latched and the PCMC remains in the indicated mode until PWROK is again negated. The high order LBX samples the state of A27 on the falling edge of CPURST.

When PWROK is low and both TESTEN and A27 are low, the 82434NX drives MA11 onto pin 109. If both TESTEN and A27 are low when PWROK transitions from low to high, the PCMC continues to drive MA11 onto pin 109. If the high order LBX samples A27 low on the falling edge of CPURST, it will tri-state pin 123.

When PWROK is low, TESTEN is low, and A27 is high the PCMC drives the output of the host clock PLL onto pin 109. Observing pin 109 when in this mode indicates if the host clock PLL has locked onto the correct frequency. If TESTEN is low and A27 is high when PWROK transitions from low to high the PCMC continues to drive the output of the host clock PLL onto pin 109, regardless of the values of TESTEN and A27. If the high order LBX samples A27 high on the falling edge of CPURST, it drives the output of its host clock PLL onto pin 123. No phase delay information can be inferred from these outputs.

When PWROK is low, TESTEN is high, A26 is high, A25 is low, A28 is high and A24 is high, the PCMC will drive the output of the NAND tree onto pin 109. If TESTEN is high, A26 is high, and A25 is low when PWROK transitions from low to high, the PCMC continues to drive the output of the NAND tree onto pin 109.

A27 must be pulled low via a pulldown resistor to ground for normal operation.

Table 25. NAND Tree Order

Order	Pin #	Signal
1	141	TRDY#
2	142	IRDY#
3	143	CBE3#
4	144	CBE2#
5	145	CBE1#
6	146	CBE0#
7	159	PPOUT0
8	160	PPOUT1
9	161	EOL
10	162	FLSHBUF#
11	163	GNT#
12	164	MEMCS#
13	165	MEMREQ#
14	167	STOP#
15	168	PLOCK#
16	169	PERR#
17	170	DEVSEL#
18	171	PAR
19	172	SERR#
20	173	FRAME#
21	194	REQ#
22	196	A23
23	197	A27
24	198	A29
25	199	A31
26	200	A21
27	201	A16
28	202	A17
29	203	A18
30	204	A0
31	205	A1
32	206	A2
33	207	A30
34	2	A28
35	3	A24
36	4	A22
37	5	A26
38	6	A19
39	7	A20
40	8	A25
41	9	A4
42	10	A5
43	11	A6
44	12	A3
45	13	A8
46	14	A7
47	15	A10
48	16	A9
49	17	A12
50	18	A11
51	19	A13
52	21	A14
53	22	A15
54	53	BE1#
55	54	BE5#
56	55	BE4#
57	56	BE0#
58	57	BE2#
59	58	BE6#
60	59	BE3#
61	60	BE7#
62	61	M/IO#
63	64	CACHE#
64	65	HITM#
65	66	ADS#
66	67	W/R#
67	68	D/C#
68	69	SMIACT#
69	71	HLOCK#
70	72	PCHK#
71	63	TESTEN

ADDITIONAL TESTING NOTES:

1. HCLKOUT[6:1] can be toggled via HCLKIN.
2. CAx[6:3] are flow through outputs via A[6:3] after PWROK transitions high.
3. MA[10:0] are flow through outputs via A[13:3] after PWROK transitions high.
4. CAS[7:0]# outputs can be tested by performing a DRAM read cycle.
5. PCLKOUT can be tested in PLL bypass mode, frequency is HCLK/2.
6. PCIRST is the NAND Tree output of Tree Cell 6.
7. INIT is the NAND Tree output of Tree Cell 53.

82420/82430
PCIset BRIDGE COMPONENT

82378ZB (SIO), 82379AB (SIO.A) FOR ISA BUSES

- Provides the Bridge between the PCI Bus and ISA Bus
- 100% PCI and ISA Compatible
- Enhanced DMA Functions (82378ZB Only)
- Integrated Data Buffers to Improve Performance
- Integrated 16-bit BIOS Timer
- Arbitration for PCI Devices
- Arbitration for ISA Devices
- Integrates the Functionality of One 82C54 Timer
- Integrates the Functionality of Two 82C59 Interrupt Controllers
- Non-Maskable Interrupts (NMI)
- Four Dedicated PCI Interrupts
- Complete Support for SL Enhanced Intel486™ CPU's
- Integrated Power Management Support
  - — System Management Interrupts
  - — Fast Off Timer
  - — STPCLK# Signal to Throttle CPU Clock
  - — APM Port
- Provides I/O APIC for Dual-Processor (DP) Support

82374EB/SB (ESC), 82375EB/SB (PCEB) FOR EISA BUSES

- Provides the Bridge between the PCI Bus and EISA Bus
- 100% PCI and EISA Compatible
- Data Buffers Improve Performance
- Data Buffer Management Ensures Data Coherency
- Burst Transfers on both the PCI and EISA Buses
- 32-Bit Data Paths
- PCI and EISA Address Decoding and Mapping
- Programmable Main Memory Address Decoding
- Integrated EISA Compatible Bus Controller
- Supports Eight EISA Slots
- Provides Enhanced DMA Controller
- Provides High Performance Arbitration
- Integrates Support Logic for X-Bus Peripheral and more
- Integrates the Functionality of Two 82C59 Interrupt Controllers and Two 82C54 Timers
- Generates Non-Maskable Interrupts
- Provides BIOS Interface

The 82420/82430 PCIset Bridge components provide a bridge between the PCI to either EISA or ISA buses. The 82378 provides the bridge between PCI bus and the ISA bus while the 82374 and 83275 together provide the bridge between the PCI bus and the EISA bus.

The SIO integrates many of the common I/O functions found in todays ISA based PC systems. The SIO incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller and support for other decode logic. The 82379AB adds an APIC for dual-processing Pentium™ Processor systems.

November 1994
Order Number: 290509-002

The 82374 EISA System Component (ESC) and 83275 PCI-EISA Bridge (PCEB) together provide the EISA system compatible master/slave functions on both the PCI Local Bus and the EISA Bus and the common I/O functions found in today's EISA systems. The ESC incorporates the logic for an EISA (master and slave) interface, EISA bus controller, enhanced seven channel DMA controller with Scatter-Gather support, EISA arbitration, 14 channel interrupt controller, five programmable timer/counters and non-maskable control logic. The ESC also integrates support logic to decode peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive. The PCEB provides the address and data paths, bus controls, and bus protocol translation for PCI-to-EISA and EISA-to-PCI transfers. Extensive data buffering in both directions increases system performance by maximizing PCI and EISA Bus efficiency and allowing concurrency on the two buses. The PCEB integrates central bus control functions, PCI parity generation, system error reporting, and programmable PCI and EISA memory and I/O address space mapping and decoding.

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

82374EB/82374SB EISA SYSTEM COMPONENT (ESC)

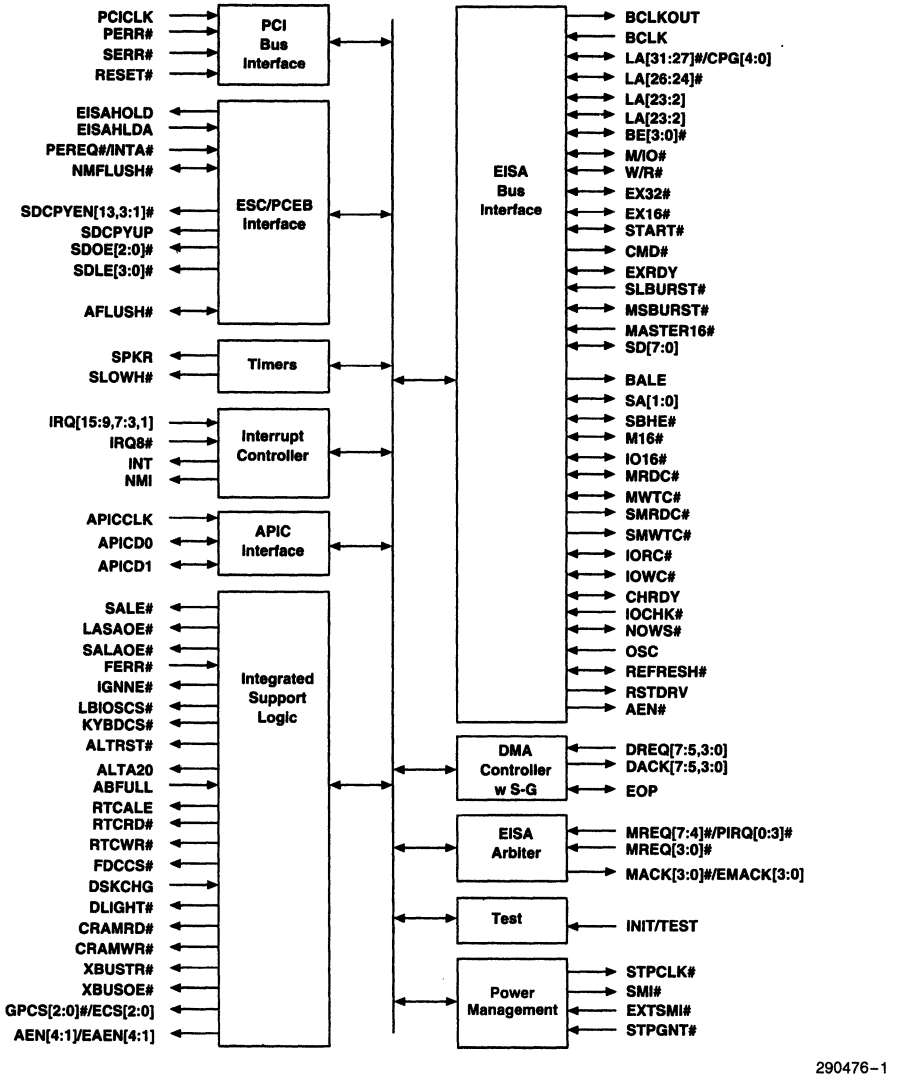
- Integrates EISA Compatible Bus Controller
  - — Translates Cycles Between EISA and ISA Bus
  - — Supports EISA Burst and Standard Cycles
  - — Supports ISA Zero Wait-State Cycles
  - — Supports Byte Assembly/ Disassembly for 8-, 16- and 32-Bit Transfers
  - — Supports EISA Bus Frequency of up to 8.33 MHz
- Supports Eight EISA Slots
  - — Directly Drives Address, Data and Control Signals for Eight Slots
  - — Decodes Address for Eight Slot Specific AENs
- Provides Enhanced DMA Controller
  - — Provides Scatter-Gather Function
  - — Supports Type A, Type B, Type C (Burst), and Compatible DMA Transfer
  - — Provides Seven Independently Programmable Channels
  - — Integrates Two 82C37A Compatible DMA Controllers
- Integrates the Functionality of two 82C59 Interrupt Controllers and two 82C54 Timers
  - — Provides 14 Programmable Channels for Edge or Level Interrupts
  - — Provides 4 PCI Interrupts Routible to any of 11 Interrupt Channels
  - — Supports Timer Function for Refresh Request, System Timer, Speaker Tone, Fail Safe Timer, and CPU Speed Control
- Advanced Programmable Interrupt Controller (APIC)
  - — Multiprocessor Interrupt Management
  - — Separate Bus For Interrupt Messages
- 5V CMOS Technology
- Provides High Performance Arbitration
  - — Supports Eight EISA Masters and PCEB
  - — Supports ISA Masters, DMA Channels, and Refresh
  - — Provides Programmable Arbitration Scheme for Fixed, Rotating, or Combination Priority
- Integrates Support Logic for X-Bus Peripherals
  - — Generates Chip Selects/Encoded Chip Selects for Floppy and Keyboard Controller, IDE, Parallel/ Serial Ports , and General Purpose Peripherals
  - — Provides Interface for Real Time Clock
  - — Generates Control Signals for X-Bus Data Transceiver
  - — Integrates Port 92, Mouse Interrupt, and Coprocessor Error Reporting
- Generates Non-Maskable Interrupts (NMI)
  - — PCI System Errors
  - — PCI Parity Errors
  - — EISA Bus Parity Errors
  - — Fail Safe Timer
  - — Bus Timeout
  - — Via Software Control
- Provides BIOS Interface
  - — Supports 512K Bytes of Flash or EPROM BIOS on the X-Bus
  - — Allows BIOS on PCI
  - — Supports Integrated VGA BIOS
- **82374SB System Power Management (Intel SMM Support)**
  - **— Fast On/Off Support via SMI GenerationHardware Events, Software Events, EXTSMI#, Fast Off Timer, System Events**
  - **— Programmable CPU Clock Control**
  - **— Enables Energy Efficient Desktop Systems**
- Only Available as Part of a Supported Kit
- 208-Pin QFP Package

December 1995
Order Number: 290476-004

This document describes both the 82374EB and 82374SB components. Unshaded areas describe the 82374EB. Shaded areas, like this one, describe the 82374SB operations that differ from the 82374EB.

The 82374EB/SB EISA System Component (ESC) provides all the EISA system compatible functions. The ESC with the PCEB provide all the functions to implement an EISA-to-PCI bridge and EISA I/O subsystem. The ESC integrates the common I/O functions found in today's EISA-based PC systems. The ESC incorporates the logic for an EISA (master and slave) interface, EISA bus controller, enhanced seven channel DMA controller with scatter-gather support, EISA arbitration, 14 channel interrupt controller, Advanced Programmable Interrupt Controller (APIC), five programmable timer/counters, and non-maskable-interrupt (NMI) control logic. The ESC also integrates support logic to decode peripheral devices such as the Flash BIOS, real time clock, keyboard/mouse controller, floppy controller, two serial ports, one parallel port, and IDE hard disk drive.

The 82374SB also contains support for SMM power management



NOTE:
SMM Interface and INIT are on the 82347SB only.

Simplified ESC Block Diagram

82374EB/82374SB EISA SYSTEM COMPONENT (ESC)

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

The PCI-EISA bridge chip set provides an I/O subsystem core for the next generation of high-performance personal computers (e.g., those based on the Intel486™ or Pentium® processors). System designers can take advantage of the power of the PCI (Peripheral Component Interconnect) for the local I/O bus while maintaining access to the large base of EISA and ISA expansion cards, and corresponding software applications. Extensive buffering and buffer management within the PCI-EISA bridge ensures maximum efficiency in both bus environments.

The chip set consists of two components—the 82375EB/SB PCI-EISA Bridge (PCEB) and the 82374EB/SB EISA System Component (ESC). These components work in tandem to provide an EISA I/O subsystem interface for personal computer platforms based on the PCI standard. This section provides an overview of the PCI and EISA Bus hierarchy followed by an overview of the PCEB and ESC components.

Bus Hierarchy—Concurrent Operations:

Figure 1 shows a block diagram of a typical system using the PCI-EISA Bridge chip set. The system contains three levels of buses structured in the following hierarchy:

- Host Bus as the execution bus
- PCI Bus as a primary I/O bus
- EISA Bus as a secondary I/O bus

This bus hierarchy allows concurrency for simultaneous operations on all three bus environments. Data buffering permits concurrency for operations that cross over into another bus environment. For example, a PCI device could post data into the PCEB, permitting the PCI Local Bus transaction to complete in a minimum time and freeing up the PCI Local Bus for further transactions. The PCI device does not have to wait for the transfer to complete to its final destination. Meanwhile, any ongoing EISA Bus transactions are permitted to complete. The posted data is then transferred to its EISA Bus destination when the EISA Bus is available. The PCI-EISA Bridge chip set implements extensive buffering for PCI-to-EISA and EISA-to-PCI bus transactions. In addition to concurrency for the operations that cross bus environments, data buffering allows the fastest operations within a particular bus environment (via PCI burst transfers and EISA burst transfers).

The PCI Local Bus with 132 MByte/sec and EISA with 33 MByte/sec peak data transfer rate represent bus environments with significantly different bandwidths. Without buffering, transfers that cross the single bus environment are performed at the speed of the slower bus. Data buffers provide a mechanism for data rate adoption so that the operation of the fast bus environment (PCI), i.e. usable bandwidth, is not significantly impacted by the slower bus environment (EISA).

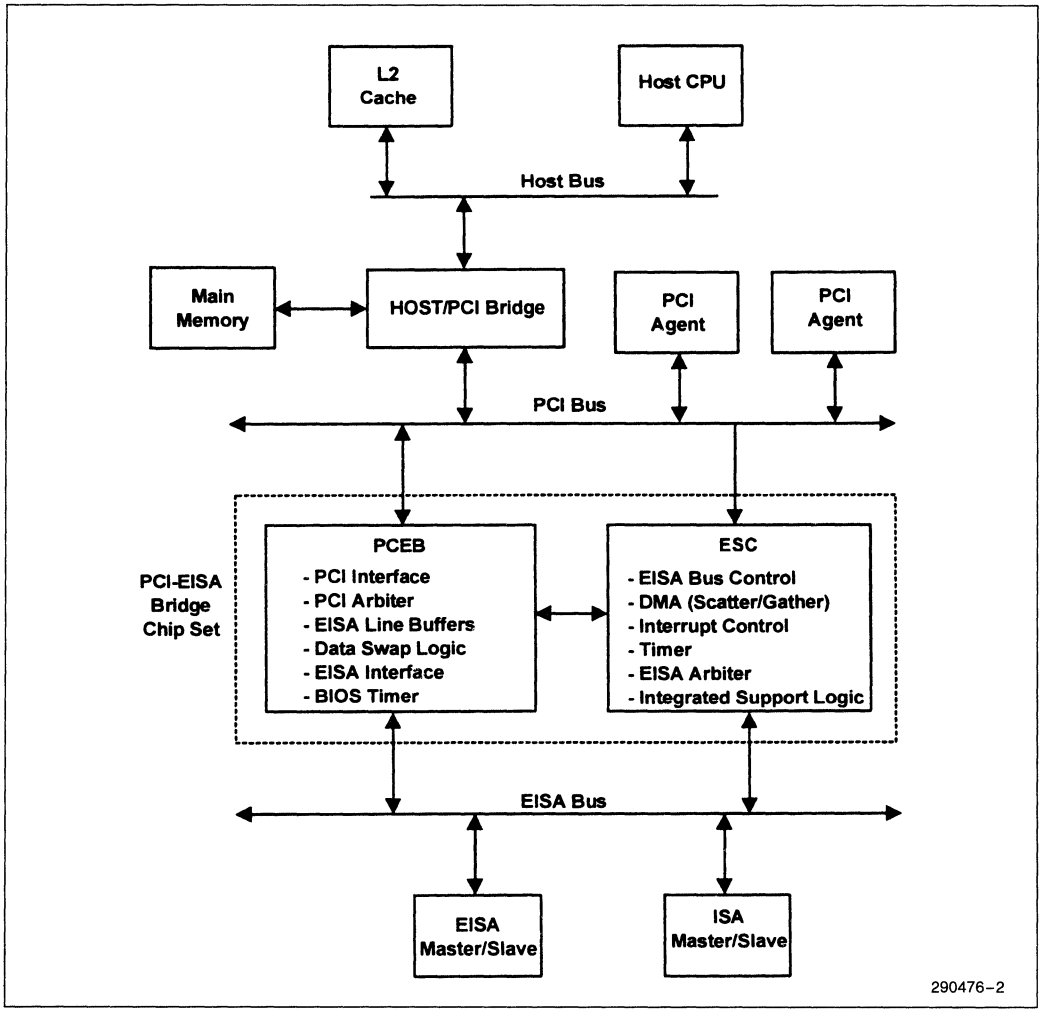


Figure 1. PCI-EISA Chip Set System Block Diagram

PCI Bus

The PCI Bus has been defined to address the growing industry needs for a standardized *local bus* that is not directly dependent on the speed and the size of the processor bus. New generations of personal computer system software such as Windows™ and Win-NT™ with sophisticated graphical interfaces, multi-tasking and multi-threading bring new requirements that traditional PC I/O architectures can not satisfy. In addition to the higher bandwidth, reliability and robustness of the I/O subsystem is becoming increasingly important. The PCI environment addresses these needs and provides an upgrade path for the future. PCI features include:

- Processor independent
- Multiplexed, burst mode operation
- Synchronous at frequencies from 20–33 MHz
- 120 MByte/sec usable throughput (132 MByte/sec peak) for 32 bit data path
- 240 MByte/sec usable throughput (264 MByte/sec peak) for 64 bit data path
- Optional 64 bit data path with operations that are transparent with the 32 bit data path
- Low latency random access (60 ns write access latency to slave registers from a master parked on the bus)
- Capable of full concurrency with processor/memory subsystem
- Full multi-master capability allowing any PCI master peer-to-peer access to any PCI slave
- Hidden (overlapped) central arbitration
- Low pin count for cost effective component packaging (address/data multiplexed)
- Address and data parity
- Three physical address spaces: memory, I/O, and configuration
- Comprehensive support for autoconfiguration through a defined set of standard configuration functions

System partitioning shown in Figure 1 illustrates how the PCI can be used as a common interface between different portions of a system platform that are typically supplied by the chip set vendor. These portions are the Host/PCI Bridge (including a main memory DRAM controller and an optional second level cache controller) and the PCI-EISA Bridge. Thus, the PCI allows a system I/O core design to be decoupled from the processor/memory treadmill, enabling the I/O core to provide maximum benefit over multiple generations of processor/memory technology. For this reason, the PCI-EISA Bridge can be used with different processors. Regardless of the new requirements imposed on the processor side of the Host/PCI Bridge (e.g. 64-bit data path, 3.3V interface, etc.) the PCI side remains unchanged which allows reusability not only of the rest of the platform chip set (i.e. PCI-EISA Bridge) but also of all other I/O functions interfaced at the PCI level. These functions typically include graphics, SCSI, and LAN.

EISA Bus

The EISA bus in the system shown in the Figure 1.0 represents a second level I/O bus. It allows personal computer platforms built around the PCI as a primary I/O bus to leverage the large EISA/ISA product base. Combinations of PCI and EISA buses, both of which can be used to provide expansion functions, will satisfy even the most demanding applications.

Along with compatibility with 16-bit and 8-bit ISA hardware and software, the EISA bus provides the following key features:

- 32-bit addressing and 32-bit data path
- 33 MByte/sec bus bandwidth
- Multiple bus master support through efficient arbitration
- Support for autoconfiguration

Integrated Bus Central Control Functions

The PCI-EISA Bridge chip set integrates central bus functions on both the PCI and EISA Buses. For the PCI Bus, the functions include PCI bus arbitration and default bus driver. For the EISA Bus, central functions include the EISA Bus controller and EISA arbiter are integrated in the ESC component and EISA Data Swap Logic is integrated in the PCEB.

Integrated System Functions

The PCI-EISA Bridge chip set integrates system functions including PCI parity and system errors reporting, buffer coherency management protocol, PCI and EISA memory and I/O address space mapping and decoding. For maximum flexibility all of these functions are programmable allowing for variety of optional features.

1.1 PCEB Overview

The PCEB provides the interface (bridge) between PCI and EISA buses by translating bus protocols in both directions. It uses extensive buffering on both the PCI and EISA interfaces to allow concurrent bus operations. The PCEB also implements the PCI central support functions (e.g., PCI arbitration, error signal support, and subtractive decoding). The major functions provided by the PCEB are described in this section.

PCI Bus Interface

The PCEB can be either a master or slave on the PCI Bus and supports bus frequencies from 25 MHz to 33 MHz. For PCI-initiated transfers, the PCEB can only be a slave. The PCEB becomes a slave when it positively decodes the cycle. The PCEB also becomes a slave for unclaimed cycles on the PCI Bus. These unclaimed cycles are either negatively or subtractively decoded by the PCEB and forwarded to the EISA Bus.

As a slave, the PCEB supports single cycle transfers for memory, I/O, and configuration operations and burst cycles for memory operations. Note that, burst transfers cannot be performed to the PCEB's internal registers. Burst memory write cycles to the EISA Bus can transfer up to four Dwords, depending on available space in the PCEB's Posted Write Buffers. When space is no longer available in the buffers, the PCEB terminates the transaction. This supports the Incremental Latency Mechanism as defined in the Peripheral Component Interconnect (PCI) Specification. Note that, if the Posted Write Buffers are disabled, PCI burst operations are not performed and all transfers are single cycle.

For EISA-initiated transfers to the PCI Bus, the PCEB is a PCI master. The PCEB permits EISA devices to access either PCI memory or I/O. While all PCI I/O transfers are single cycle, PCI memory cycles can be either single cycle or burst, depending on the status of the PCEB's Line Buffers. During EISA reads of PCI memory, The PCEB uses a burst read cycle of four Dwords to prefetch data into a Line Buffer. During EISA-to-PCI memory writes, the PCEB uses PCI burst cycles to flush the Line Buffers. The PCEB contains a programmable Master Latency Timer that provides the PCEB with a guaranteed time slice on the PCI Bus, after which it surrenders the bus.

As a master on the PCI Bus, the PCEB generates address and command signal (C/BE#) parity for read and write cycles, and data parity for write cycles. As a slave, the PCEB generates data parity for read cycles. Parity checking is not supported.

The PCEB, as a resource, can be locked by any PCI master. In the context of locked cycles, the entire PCEB subsystem (including the EISA Bus) is considered a single resource.

PCI Bus Arbitration

The PCI arbiter supports six PCI masters—The Host/PCI bridge, PCEB, and four other PCI masters. The arbiter can be programmed for twelve fixed priority schemes, a rotating scheme, or a combination of the fixed and rotating schemes. The arbiter can be programmed for bus parking that permits the Host/PCI Bridge default access to the PCI Bus when no other device is requesting service. The arbiter also contains an efficient PCI retry mechanism to minimize PCI Bus thrashing when the PCEB generates a retry. The arbiter can be disabled, if an external arbiter is used.

EISA Bus Interface

The PCEB contains a fully EISA-compatible master and slave interface. The PCEB directly drives eight EISA slots without external data or address buffering. The PCEB is only a master or slave on the EISA Bus for transfers between the EISA Bus and PCI Bus. For transfers contained to the EISA Bus, the PCEB is never a master or slave. However, the data swap logic contained in the PCEB is involved in these transfers, if data size translation is needed. The PCEB also provide support for I/O recovery.

EISA/ISA masters and DMA can access PCI memory or I/O. The PCEB only forwards EISA cycles to the PCI Bus if the address of the transfer matches one of the address ranges programmed into the PCEB for EISA-to-PCI positive decode. This includes the main memory segments used for generating MEMCS# from the EISA Bus, one of the four programmable memory regions, or one of the four programmable I/O regions. For EISA-initiated accesses to the PCI Bus, the PCEB is a slave on the EISA Bus. I/O accesses are always non-buffered and memory accesses can be either non-buffered or buffered via the Line Buffers. For buffered accesses, burst cycles are supported.

During PCI-initiated cycles to the EISA Bus, the PCEB is an EISA master. For memory write operations through the Posted Write Buffers, the PCEB uses EISA burst transfers, if supported by the slave, to flush the buffers. Otherwise, single cycle transfers are used. Single cycle transfers are used for all I/O cycles and memory reads.

PCI/EISA Address Decoding

The PCEB contains two address decoders—one to decode PCI-initiated cycles and the other to decode EISA-initiated cycles. The two decoders permit the PCI and EISA Buses to operate concurrently.

The PCEB can also be programmed to provide main memory address decoding on behalf of the Host/PCI bridge. When programmed, the PCEB monitors the PCI and EISA bus cycle addresses, and generates a memory chip select signal (MEMCS#) indicating that the current cycle is targeted to main memory residing behind the Host/PCI bridge. Programmable features include, read/write attributes for specific memory segments and the enabling/disabling of a memory hole. If MEMCS# is not used, this feature can be disabled.

In addition to the main memory address decoding, there are four programmable memory regions and four programmable I/O regions for EISA-initiated cycles. EISA/ISA master or DMA accesses to one of these regions are forwarded to the PCI Bus.

Data Buffering

To isolate the slower EISA Bus from the PCI Bus, the PCEB provides two types of data buffers. Buffer management control guarantees data coherency.

For EISA-initiated cycles to the PCI Bus, there are four 16-byte wide Line Buffers. These buffers permit prefetching of PCI memory read data and posting of PCI memory write data.

By using burst transactions to fill or flush these buffers, if appropriate, the PCEB maximizes bus efficiency. For example, an EISA device could fill a Line Buffer with byte, word, or Dword transfers and The PCEB would use a PCI burst cycle to flush the filled line to PCI memory.

BIOS Timer

The PCEB has a 16 bit BIOS Timer. The timer can be used by BIOS software to implement timing loops. The timer count rate is derived from the EISA clock (BCLK) and has an accuracy of ± 1 μs.

1.2 ESC Overview

The ESC implements system functions (e.g., timer/counter, DMA, and interrupt controller) and EISA subsystem control functions (e.g., EISA bus controller and EISA bus arbiter). The major functions provided by the ESC are described in this section.

EISA Controller

The ESC incorporates a 32-bit master and an 8-bit slave. The ESC directly drives eight EISA slots without external data or address buffering. EISA system clock (BCLK) generation is integrated by dividing the PCI clock (divide by 3 or divide by 4) and wait-state generation is provided. The AENx and MACKx signals provide a direct interface to four EISA slots and supports eight EISA slots with encoded AENx and MACKx signals.

The ESC contains an 8-bit data bus (lower 8 bits of the EISA data bus) that is used to program the ESC's internal registers. Note that for transfers between the PCI and EISA Buses, the PCEB provides the data path. Thus, the ESC does not require a full 32 bit data bus. A full 32-bit address bus is provided and is used during refresh cycles and for DMA operations.

The ESC performs cycle translation between the EISA Bus and ISA Bus. For mis-matched master/slave combinations, the ESC controls the data swap logic that is located in the PCEB. This control is provided through the PCEB/ESC interface.

DMA Controller

The ESC incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels. Each channel can be programmed for 8 or 16 bit DMA device size, and ISA-compatible, type "A", type "B", or type "C" timings. Full 32 bit addressing is provided. The DMA controller is also responsible for generating refresh cycles.

The DMA controller supports an enhanced feature called scatter/gather. This feature provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In scatter/gather mode, the DMA can read the memory address and word count from an array of buffer descriptors, located in main memory, called the scatter/gather descriptor (SGD) table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD table are handled.

Interrupt Controller

The ESC contains an EISA compatible interrupt controller that incorporates the functionality of two 82C59 Interrupt Controllers. The two interrupt controllers are cascaded providing 14 external and two internal interrupts.

Advanced Programmable Interrupt Controller (APIC)

In addition to the standard EISA compatible interrupt controller described above, the ESC incorporates the Advanced Programmable Interrupt Controller (APIC). While the standard interrupt controller is intended for use in a uni-processor system, APIC can be used in either a uni-processor or multi-processor system. APIC provides multi-processor interrupt management and incorporates both static and dynamic symmetric interrupt distribution across all processors. In systems with multiple I/O subsystems, each subsystem can have its own set of interrupts.

Timer/Counter

The ESC provides two 82C54 compatible timers (Timer 1 and Timer 2). The counters in Timer 1 support the system timer interrupt (IRQ0#), refresh request, and a speaker tone output (SPKR). The counters in Timer 2 support fail-safe timeout functions and the CPU speed control.

Integrated Support Logic

To minimize the chip count for board designs, the ESC incorporates a number of extended features. The ESC provides support for ALTA20 (Fast A20GATE) and ALTRST with I/O Port 92h. The ESC generates the control signals for SA address buffers and X-Bus buffer. The ESC also provides chip selects for BIOS, the keyboard controller, the floppy disk controller, and three general purpose devices. Support for generating chip selects with an external decoder is provided for IDE, a parallel port, and a serial port. The ESC provides support for a PC/AT compatible coprocessor interface and IRQ13 generation.

Power Management (82374SB)

Extensive power management capability permits a system to operate in a low power state without being powered down. Once in the low power state (called "Fast Off" state), the computer appears to be off. For example, the SMM code could turn off the CRT, line printer, hard disk drive's spindle motor, and fans. In addition, the CPU's clock can be governed. To the user, the machine appears to be in the off state. However, the system is actually in an extremely low power state that still permits the CPU to function and maintain communication connections normally associated with today's desktops (e.g., LAN, Modem, or FAX). Programmable options provide power management flexibility. For example, various system events can be programmed to place the system in the low power state or break events can be programmed to wake the system up.

2.0 SIGNAL DESCRIPTION

This section provides a detailed description of each signal. The signals are arranged in a functional group according to their associated interface.

The "#" symbol at the end of a signal indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not presented after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

The following notations are used to describe the signal type.

in Input is a standard input-only signal.

out Totem Pole Output is a standard active driver.

o/d Open Drain Input/Output.

t/s Tri-State is a bi-directional, tri-state input/output pin.

s/t/s Sustained Tri-State is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. A pull-up sustains the inactive state until another agent drives it and is provided by the central resource.

NOTE:

During a hard reset, INTR, NMI, IGNNE#, SMI# (on 82374SB), ALTRST#, STPCLK# (on 82374SB) and ALTA20 are driven low to prevent problems associated with 5V/3.3V power sequencing. Any outputs of the ESC that are directed to a 3.3V CPU must be driven through a 5V to 3.3V translator.

2.1 PCI Local Bus Interface Signals

Pin Name	Type	Description
PCICLK	in	**PCI CLOCK:** PCICLK provides timing for all transactions on the PCI bus. The ESC uses the PCI Clock (PCICLK) to generate EISA Bus Clock (BCLK). The PCICLK is divided by 3 or 4 to generate the BCLK. The EISA Bridge supports PCI Clock frequencies of 25 MHz through 33 MHz.
PERR#	in	**PARITY ERROR:** PERR# indicates a data parity error. PERR# may be pulsed active by any agent that detects an error condition. Upon sampling PERR# active, the ESC generates an NMI interrupt to the CPU.
SERR#	in	**SYSTEM ERROR:** SERR# may be pulsed active by any agent that detects an error condition. Upon sampling SERR# active, the ESC generates an NMI interrupt to the CPU.
RESET#	in	**SYSTEM RESET:** RESET# forces the entire ESC chip into a known state. All internal ESC state machines are reset and all registers are set to their default values. RESET# may be asynchronous to PCICLK when asserted or negated. Although asynchronous, negation must be a clean, bounce-free edge. The ESC uses RESET# to generate RSTDRV signal.

2.2 EISA Bus Interface Signals

Pin Name	Type	Description
BCLKOUT	out	**EISA BUS CLOCK OUTPUT**: BCLKOUT is typically buffered to create EISA Bus Clock (BCLK). The BCLK is the system clock used to synchronize events on the EISA/ISA bus. The BCLKOUT is generated by dividing the PCICLK. The ESC uses a divide by 3 or divide by 4 to generate the BCLKOUT.
BCLK	in	**EISA BUS CLOCK**: The ESC uses BCLK to synchronize events on the EISA bus. The ESC generates or samples all the EISA/ISA bus signals on either the rising or the falling edge of BCLK.

Pin Name	Type	Description
LA[31:27]#/ CPG[4:0]	t/s	**EISA ADDRESS BUS/CONFIGURATION RAM PAGE ADDRESS**: These are multiplexed signals. These signals behave as the EISA address bus under all conditions except during access cycle to the Configuration RAM. EISA Address Bus: LA[31:27]# are directly connected to the EISA address bus. The ESC uses the address bus in conjunction with the BE[3:0]# signals as inputs to decode accesses to its internal resources except in DMA and Refresh modes. During DMA and Refresh modes, these are outputs, and the ESC uses these signals in conjunction with BE[3:0]# to drive Memory address. Configuration Ram Page Address: CPG[4:0] are connected to Configuration SRAM address lines. During I/O access to 0800h-08FFh, the ESC drives these signals with the configuration page address (the value contained in register 0C00h). The Configuration RAM Page Address function can be disabled by setting Mode Select register bit 5 = 0.
LA[26:24]# and LA[23:2]	t/s	**EISA ADDRESS BUS**: These signals are directly connected to the EISA address bus. The ESC uses the address bus in conjunction with the BE[3:0]# signals as inputs to decode accesses to its internal resources except in DMA and Refresh modes. During DMA and Refresh modes, these are outputs, and the ESC uses these signals in conjunction with BE[3:0]# to drive Memory address.
BE[3:0]#	t/s	**BYTE ENABLES**: BE[3:0]# signals are directly connected to the EISA address bus. These signals indicate which byte on the 32-bit EISA data bus are involved in the current cycle. BE[3:0]# are inputs during EISA master cycles which do not require assembly/disassembly operation. For EISA master assembly/disassembly cycles, ISA master cycles, DMA, and Refresh cycles BE[3:0]# are outputs. BE0#: Corresponds to byte lane 0-SD[7:0] BE1#: Corresponds to byte lane 0-SD[15:8] BE2#: Corresponds to byte lane 0-SD[23:16] BE3#: Corresponds to byte lane 0-SD[31:24]
M/IO#	t/s	**MEMORY OR I/O CYCLE:** M/IO# signal is used to differentiate between memory cycles and I/O cycles on the EISA bus. A High value on this signal indicates a memory cycle, and a Low value indicates an I/O cycle. M/IO# is an input to the ESC during EISA master cycles, and M/IO# is an output during ISA, DMA, and ESC initiated Refresh cycles. M/IO# is floated during ISA master initiated Refresh cycles.
W/R#	t/s	**WRITE OR READ CYCLE:** W/R# signal is used to differentiate between write and read cycles on the EISA bus. A High value on this signal indicates a Write cycle , and a Low value indicates a Read cycle. W/R# is an input to the ESC during EISA master cycles, and W/R# is an output during ISA, DMA, and Refresh cycles.

Pin Name	Type	Description
EX32#	o/d	**EISA 32 BIT DEVICE DECODE**: EX32# signal is asserted by a 32-bit EISA slave device. EX32# assertion indicates that an EISA device has been selected as a slave, and the device has a 32-bit data bus size. The ESC uses this signal as an input as part of its slave decode to determine if data size translation and/or cycle translation is required. EX32# is an output of the ESC during the last portion of the mis-matched cycle. This is an indication to the backed-off EISA master that the data translation has been completed. The backed-off EISA master uses this signal to start driving the EISA bus again.
EX16#	o/d	**EISA 16-BIT DEVICE DECODE:** EX16# signal is asserted by a 16-bit EISA slave device. EX16# assertion indicates that an EISA device has been selected as a slave, and the device has a 16 bit data bus size. The ESC uses this signal as an input as part of its slave decode to determine if data size translation and/or cycle translation is required. EX16# is an output of the ESC during the last portion of the mis-matched cycle. This is an indication to the backed-off EISA master that the data translation has been completed. The backed-off EISA master uses this signal to start driving the EISA bus again.
START#	t/s	**START CYCLE:** START# signal provides timing control at the start of an EISA cycle. START# is asserted for one BCLK. START# is an input to the ESC during EISA master cycles except portions of the EISA master to mis-matched slave cycles where it becomes an output. During ISA, DMA, and Refresh cycles START# is an output.
CMD#	out	**COMMAND:** CMD# signal provides timing control within an EISA cycle. The ESC is a central resource of the CMD# signal, and the ESC generates CMD# during all EISA cycles. CMD# is asserted from the rising edge of BCLK simultaneously with the negation of START#, and remains asserted until the end of the cycle.
EXRDY	o/d	**EISA READY:** EXRDY signal is deasserted by EISA slave devices to add wait states to a cycle. EXRDY is an input to the ESC for EISA master cycles, ISA master cycles, and DMA cycles where an EISA slave has responded with EX32# or EX16# asserted. The ESC samples EXRDY on the falling edge of BCLK after CMD# is asserted (except during DMA compatible cycles). During DMA compatible cycles, EXRDY is sampled on the second falling edge of BCLK after CMD# is driven active. For all types of cycles if EXRDY is sampled inactive, the ESC keeps sampling it on every falling edge of BCLK#. EXRDY is an output for EISA master cycles decoded as accesses to the ESC internal registers. ESC forces EXRDY low for one BCLK at the start of a potential DMA burst write cycle to insure that the initial write data is held long enough to be sampled by the memory slave.
SLBURST#	in	**SLAVE BURST:** SLBURST# signal is asserted by an EISA slave to indicate that the device is capable of accepting EISA burst cycles. The ESC samples SLBURST# on the rising edge of BCLK at the end of START# for all EISA cycles. During DMA cycles, the ESC samples SLBURST# twice; once on the rising edge of BCLK at the beginning of START# and again on the rising edge of BCLK at the end of START#.

Pin Name	Type	Description
MSBURST#	t/s	**MASTER BURST:** MSBURST# signal is asserted by an EISA master to indicate EISA burst cycles. MSBURST# is asserted by an EISA master in response to an asserted SLBURST# signal. The ESC samples SLBURST# on the rising edge of BCLK that CMD# is asserted. If asserted, the ESC samples SLBURST# on all subsequent rising edges of BCLK until sampled negated. The ESC keeps CMD# asserted during Burst cycles. MSBURST# is an output during DMA burst cycles. The ESC drives MSBURST# active on the falling edge of BCLK, one half BCLK after SLBURST# is sampled active at the end of START#.
MASTER16#	in	**MASTER 16-BIT:** MASTER16# is asserted by a 16-bit EISA Bus master or an ISA Bus master device to indicate that it has control of the EISA Bus or ISA Bus. The ESC samples MASTER16# on the rising edge of BCLK that START# is asserted. If MASTER16# is sampled asserted, the ESC determines that a 16-bit EISA Bus master or an ISA Bus master owns the Bus. If MASTER16# is sampled negated at the first sampling point, the ESC will sample MASTER16# a second time on the rising edge of BCLK at the end of START#. If MASTER16# is sampled asserted here, the ESC determines that a 32-bit EISA Bus master has downshifted to a 16-bit Bus master, and thus, the ESC will disable the data size translation function.
SD[7:0]	t/s	**SYSTEM DATA:** SD[7:0] signals are directly connected to the System Data bus. The SD[7:0] pins are outputs during I/O reads when the ESC internal registers are being accessed and during interrupt acknowledge cycles. The SD[7:0] pins are input during I/O writes cycles when the ESC internal registers are being accessed.

2.3 ISA Bus Signals

Pin Name	Type	Description
BALE	out	**BUS ADDRESS LATCH ENABLE:** BALE signal is asserted by the ESC to indicate that a address (SA[19:0], LA[23:17]), AEN and SBHE# signal lines are valid. The LA[23:17] address lines are latched on the trailing edge of BALE. BALE remains active throughout DMA and ISA Master cycles and Refresh cycles.
SA[1:0]	t/s	**ISA ADDRESS BITS 0& 1:** SA[1:0] are the least significant bits of the ISA address bus. SA[1:0] are inputs to the ESC during ISA master cycles except during ISA master initiated Refresh cycles. The ESC uses the SA[1:0] in conjunction with SBHE# to generate BE[3:0]# on the EISA bus. The SA[1:0] are outputs of the ESC during EISA master cycles and DMA cycles. The ESC generates these from BE[3:0]#.
SBHE#	t/s	**ISA BYTE HIGH ENABLE:** SBHE# signal indicates that the high byte on the ISA data bus (SD[15:8]) is valid. SBHE# is an input to the ESC during ISA master cycles, except during ISA master initiated Refresh cycles. The ESC uses the SBHE# in conjunction with SA[1:0] to generate BE[3:0]# on the EISA bus. SBHE# is an output during EISA master and DMA cycles.

Pin Name	Type	Description
M16#	o/d	**MEMORY CHIP SELECT 16:** M16# is an input when the ESC component owns the ISA bus. M16# is an output when an external ISA bus Master owns the ISA bus. The ISA slave memory drives this signal Low if it is a 16-bit memory device. For ISA to EISA translation cycles, the ESC combinatorially asserts M16# if either EX32# or EX16# are asserted. This signal has an external pull-up resistor.
IO16#	o/d	**16 BIT I/O CHIP SELECT:** IO16# signal is used to indicate a 16-bit I/O bus cycle. This signal is asserted by the I/O devices to indicate that they support 16-bit I/O bus cycles. All I/O accesses to the ESC registers are run as 8-bit I/O bus cycles. This signal has an external pull-up resistor.
MRDC#	t/s	**MEMORY READ:** MRDC# signal indicates a read cycle to the ISA memory devices. MRDC# is the command to a memory slave that it may drive data onto the ISA data bus. MRDC# is an output when the ESC owns the ISA bus. MRDC# is an input when an external ISA Bus master owns the ISA Bus. This signal is driven by the ESC during refresh cycles.
MWTC#	t/s	**MEMORY WRITE:** MWTC# signal indicates a write cycle to the ISA memory devices. MWTC# is the command to a memory slave that it may latch data from the ISA data bus. MWTC# is an output when the ESC owns the ISA bus. MWTC# is an input when an ISA Bus master owns the ISA Bus.
SMRDC#	out	**SYSTEM MEMORY READ:** SMRDC# signal is asserted by the ESC to request a memory slave to drive data onto the data lines. SMRDC# indicates that the memory read cycle is for an address below the 1 MByte range on the ISA bus. This signal is also asserted during refresh cycles.
SMWTC#	out	**SYSTEM MEMORY WRITE:** SMWTC# signal is asserted by the ESC to request a memory slave to accept data from the data lines. SMWTC# indicates that the memory write cycle is for an address below the 1 MByte range.
IORC#	t/s	**I/O READ:** IORC# is the command to an ISA I/O slave device that it may drive data on to the data bus (SD[15:0]). The device must hold the data valid until after IORC# is negated. IORC# is an output when the ESC component owns the ISA bus. IORC# is an input when an ISA Bus master owns the ISA Bus.
IOWC#	t/s	**I/O WRITE:** IOWC# is the command to an ISA I/O slave device that it may latch data from the ISA data bus (SD[15:0]). IOWC# is an output when the ESC component owns the ISA Bus. IOWC# is an input when an ISA Bus master owns the ISA Bus.
CHRDY	o/d	**I/O CHANNEL READY:** CHRDY when asserted allows ISA Bus resources request additional time (wait-states) to complete the cycle. CHRDY is an input when the ESC owns the ISA Bus. CHRDY is an input to the ESC during compatible DMA cycles. CHRDY is an output during ISA Bus master cycles to PCI slave or ESC internal register. The ESC will ignores CHRDY for ISA-Bus master accessing an ISA-Bus slave.
IOCHK#	in	**I/O CHANNEL CHECK:** IOCHK# can be asserted by any resource on the ISA Bus. When asserted, it indicates that a parity or an uncorrectable error has occurred for a device or memory on the ISA Bus. A NMI will be generated to the CPU if enabled.

Pin Name	Type	Description
NOWS#	o/d	**ZERO WAIT STATES:** NOWS# indicates that an peripheral device wishes to execute a zero wait-state bus cycle (the normal default 16-bit ISA bus memory or I/O cycle is 3 BCLKS). When NOWS# is asserted, a 16-bit memory cycle will occur in two BCLKs and a 16-bit I/O cycle will occur in three BCLKs. When NOWS# is asserted by an 8-bit device the default 6 BCLKs cycle is shortened to 4 or 5 BCLKs. NOWS# is an input when the ESC performing bus translation cycles. NOWS# is an output when the ESC internal registers are accessed. If CHRDY and NOWS# are both asserted during the same clock then NOWS# will be ignored and wait-states will be added as a function of CHRDY (CHRDY has precedence over NOWS#).
OSC	in	**OSCILLATOR:** OSC is the 14.31818 MHz signal with 50% duty cycle. OSC is used by the ESC timers.
RSTDRV	out	**RESET DRIVE:** RSTDRV is asserted by the ESC. An asserted RSTDRV causes a hardware reset of the devices on the ISA Bus. RSTDRV is asserted whenever the RESET# input to the ESC is asserted.
REFRESH#	t/s	**REFRESH:** REFRESH# is used by the ESC as an output to indicate when a refresh cycle is in progress. It should be used to enable the SA[15:0] address to the row address inputs of all banks of dynamic memory on the ISA bus so that when MRDC# goes active, the entire expansion bus dynamic memory is refreshed. Memory slaves must not drive any data onto the bus during refresh and should not add wait states since this will affect the entire system throughput. As an output, this signal is driven directly onto the ISA bus. This signal is an output only when the ESC DMA Refresh is a master on the bus responding to an internally generated request for Refresh. Upon RESET this pin will tristate. Note that address lines [15:8] are driven during refresh, but the value is meaningless and is not used to refresh ISA bus memory. REFRESH# may asserted by an expansion bus adapter acting as a 16-bit ISA bus master.
AEN#	out	**ADDRESS ENABLE:** AEN# is driven high for Bus master cycles. AEN# is driven low for DMA cycles. and Refresh cycles. AEN# is used to disable I/O devices from responding to DMA and Refresh cycles. System designs which do not used the slots specific AENs (AEN[4:1]/EAEN[4:1]) provided by the ESC can use the AEN# signal to generate their own slot specific AENs.
AEN[4:1]/ EAEN[4:1]	out	**SLOT SPECIFIC ADDRESS ENABLE/ENCODED SLOT SPECIFIC ADDRESS ENABLE:** These pins have a slightly different function depending on the ESC configuration (Mode Select register bit 1 and bit 0). Slot Specific Address Enable: If the ESC is programmed to support 4 EISA slots, these signals function as Slot Specific Address Enables (AEN[4:1]). Encoded Slot Specific Address Enable: If the ESC has been programmed to support more than 4 EISA slots, then these signals behave as Encoded Address Enables (EAEN[4:1]). A discrete decoder is required to generate slot specific AENs. Refer to Section 5.8.1 AEN GENERATION for a detailed description of these signals.

2.4 DMA Signal Description

Pin Name	Type	Description
DREQ[7:5,3:0]	in	**DMA REQUEST:** DREQ signals are either used to request DMA service from the ESC or used to gain control of the ISA Bus by a ISA Bus master. The active level (high or low) is programmed in the Command registers. When the Command register bit 6 is programmed to 0, DREQ are asserted high, otherwise the DREQ are asserted low. All inactive to active edges of DREQ are assumed to be asynchronous. The request must remain asserted until the appropriate DACK is negated. At power-up and after RESET, these lines should be low (negated).
DACK# [7:5,3:0]	out	**DMA ACKNOWLEDGE:** DACK# indicate that a request for DMA service from the DMA subsystem has been recognized or that an ISA Bus master has been granted the bus. The level of the DACK lines when asserted may be programmed to be either high or low. This is accomplished by programming the DMA Command register. These lines should be used to decode the DMA slave device with the IORC# or IOWC# line to indicate selection. If used to signal acceptance of a bus master request, this signal indicates when it is legal to assert MASTER16#. If the DMA controller has been programmed for a timing mode other than compatible mode, and another device has requested the bus, and a 4 μs time has elapsed, DACK# will be negated and the transfer stopped before the transfer is complete. In this case, the transfer will be restarted at the next arbitration period in which the channel wins the bus. Upon reset these lines are negated.
EOP	t/s	**END OF PROCESS:** EOP pin acts in one of two modes, and it is directly connected to the TC line of the ISA Bus. In the first mode, EOP-In, the pin is an input and can be used by a DMA slave to stop a DMA transfer. In the second mode, TC-Out, it is used as a terminal count output by DMA slaves. An active pulse is generated when the byte counter reaches its last value. **EOP-In Mode:** During DMA, for all transfer types, the EOP pin is sampled by the ESC. If it is sampled asserted, the address bus is tristated and the transfer is terminated. **TC-Out Mode:** The EOP output will be asserted after a new address has been output if the byte count expires with that transfer. The EOP (TC) will stay asserted until AEN# is negated unless AEN is negated during an autoinitialization. EOP (TC) will be negated before AEN is negated during an autoinitialization. **Intout Mode:** In this mode the EOP signal has the same behavior as the Chaining Interrupt or the Scatter-Gather interrupt to the host processor (IRQ13). If a scatter-gather or chaining buffer is expired, EOP will go active on the falling edge of BCLK. Only the currently active channel's interrupt will be reflected on this pin. Other channel's with active interrupts pending will not affect the EOP pin. Whenever all the DMA channels are not in use, the EOP pin is kept in output mode and negated. After reset, the EOP pin is kept in output mode and negated.

2.5 EISA Arbitration Signals

<table>
<tr><th>Pin Name</th><th>Type</th><th>Description</th></tr>
<tr><td>MREQ[3:0]#</td><td>in</td><td>MASTER REQUEST: MREQ[3:0]# are slot specific signals used by EISA bus masters to request bus access. MREQ# once asserted, must remain asserted until the corresponding MACK# is asserted. The MREQ# is negated on the falling edge of BCLK slightly before the end of a master transfer. The LA[ ], BE[ ]#, M/IO#, and W/R# lines should be floated on or before the rising edge of BCLK after MREQ# is negated. The end of the last bus cycle is derived from CMD# in this case. The MREQ# signals are asserted on the falling edge of BCLK. MREQ# is always sampled on the rising edge of BCLK. MREQ# is synchronous with respect to BCLK. After asserting MREQ# , the corresponding master must not assert MREQ# until 1.5 BCLKs after CMD# is negated.</td></tr>
<tr><td>MREQ[7:4]#/
PIRQ[0:3]#</td><td>in</td><td>MASTER REQUEST/PCI INTERRUPT REQUEST: These pins behave in one of two modes depending on the state of the Mode Select Register bit 1 and bit 0.

Master Request: MREQ# lines are slot specific signals used by EISA bus masters to request bus access. This signal behave in the same manner as MREQ[3:0]# signals.

PCI Interrupt Request: PIRQ# are used to generate asynchronous interrupts to the CPU via the Programmable Interrupt Controller (82C59) integrated in the ESC. These signals are defined as level sensitive and are asserted low. The PIRQx# can be shared with PC compatible interrupts IRQ3:IRQ7, IRQ9:IRQ15. The PIRQx# Route Control Register determines which PCI interrupt is shared with which PC compatible interrupt.

<table>
<tr><th>Register Bit[1:0]</th><th colspan="4">Pins</th></tr>
<tr><th></th><th>MREQ7#/ PIRQ0#</th><th>MREQ6#/ PIRQ1#</th><th>MREQ5#/ PIRQ2#</th><th>MREQ4#/ PIRQ3#</th></tr>
<tr><td>00</td><td>PIRQ0#</td><td>PIRQ1#</td><td>PIRQ2#</td><td>PIRQ3#</td></tr>
<tr><td>01</td><td>PIRQ0#</td><td>PIRQ1#</td><td>MREQ5#</td><td>MREQ4#</td></tr>
<tr><td>10</td><td>PIRQ0#</td><td>MREQ6#</td><td>MREQ5#</td><td>MREQ4#</td></tr>
<tr><td>11</td><td>MREQ7#</td><td>MREQ6#</td><td>MREQ5#</td><td>MREQ4#</td></tr>
</table>
</td></tr>
</table>

Pin Name	Type	Description
MACK[3:0]# / EMACK[3:0]	out	**MASTER ACKNOWLEDGE:/ENCODED MASTER ACKNOWLEDGE:** These pins behave in one of two modes depending on the state of the Mode Select register bit 1 and bit 0. If the ESC is programmed to support 4 EISA slots, then these pins are used as MACK#. If the ESC is programmed to support more than 4 EISA slots, then these pins are used as EMACK# **Master Acknowledge:** The MACK[3:0]# signals are asserted from the rising edge of BCLK at which time the bus master may begin driving the LA[ ], BE[ ]#, M/IO#, and W/R# lines on the next falling edge of BCLK. MACK# will stay asserted until the rising edge of BCLK when MREQ# is sampled negated. MACK# is sampled by EISA Bus masters on the falling edge of BCLK. If another device has requested the bus, MACK# will be negated before MREQ# is negated. When MACK# is negated, the granted device has a maximum of 8 μs to negate MREQ# and begin a final bus cycle. The ESC may negate the MACK# signal a minimum of one BCLK after asserting it if another device (or refresh) is requesting the bus. Upon reset MACK# is negated. **Encoded Master Acknowledge:** EMACK# behaves like MACK#. The difference is that a discrete decoder is required to generate MACK# for the EISA Bus masters. Refer to Section 5.8.2 MACK Generation for details.

2

2.6 Timer Unit Signal

Pin Name	Type	Description
SPKR	out	**SPEAKER DRIVE:** SPKR is the output of Timer 1, Counter 2 and is "ANDed" with Port 061h bit 1 to provide Speaker Data Enable. This signal drives an external speaker driver device, which in turn drives the ISA system speaker. SPKR has a 24 mA drive capability. Upon reset, its output state is low.
SLOWH#	out	**SLOW DOWN CPU:** SLOWH# is the output of Timer 2, Counter 2. This counter is used to slow down the main CPU of its execution via the CPU's HOLD pin by pulse width modulation. The first read of I/O register in the 048h-04Bh range will enable SLOWH# signal to follow the output of the Timer 2, Counter 2. Upon reset, SLOWH# is negated. **Hardware Reset (Strapping Option)** During hardware reset this signal is an input and the level on the pin at the end of the reset sequence determines where BIOS resides. A high level indicates that BIOS resides on the X-Bus and a low level indicates that BIOS resides on the ISA Bus. The status is used by the ESC, to control the X-Bus transceivers during BIOS access. **NOTE:** For the 82374EB, this pin has an internal weak pull-up of approximately 8 KΩ. For proper configuration of the BIOS location during reset, a weak external pull-down resistor (approx. 500Ω) must be connected to this pin. An external pull-down resistor is not needed for the 82374SB.

2.7 Interrupt Controller Signals

Pin Name	Type	Description
IRQ[15:9], IRQ8#, IRQ[7:3,1]	in	**INTERRUPT REQUEST:** IRQ These signals provide both system board components and EISA bus I/O devices with a mechanism for asynchronously interrupting the CPU. The assertion mode of each interrupt can be programmed to be edge or level triggered. An asserted IRQ input must remain asserted until after the falling edge of INTA#. If the input is negated before this time, a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt. IRQ8# requires an external pull-up resistor (8 KΩ–10 KΩ).
INTR	out	**CPU INTERRUPT:** INTR is driven by the ESC to signal the CPU that an Interrupt request is pending and needs to be serviced. It is asynchronous with respect to BCLK or PCICLK and it is always an output. The interrupt controllers must be programmed following a reset to ensure that this pin takes on a known state. Upon reset the state of this pin is undefined.
NMI	out	**NON-MASKABLE INTERRUPT:** NMI is used to force a non-maskable interrupt to the CPU. The CPU registers an NMI when it detects a rising edge on NMI. NMI will remain active until a read from the CPU to the NMI register at port 061h is detected by the ESC. This signal is set to low upon reset.

2.8 APIC Bus Signals

Pin Name	Type	Description
APICCLK	in	**APIC BUS CLOCK:** APICCLK provides the timing reference for the APIC Bus. Changes on APICD[1:0]# are synchronous to the rising edge of APICCLK.
APICD[1:0]	od	**APIC DATA:** APICD1 and APICD0 are the APIC data bus signals. Interrupt messages are sent/received over this bus. APIC arbitration uses APICD1.

2.9 System Power Management Signals (82374SB Only)

Pin Name	Type	Description
STPCLK#	out	**STOP CLOCK:** STPCLK# is asserted by the ESC in response to one of many maskable hardware or software events. For 3.3V processors that are not 5V tolerant, STPCLK# is driven to the CPU STPCLK# pin through a 5V to 3.3V translator. When the CPU samples STPCLK# asserted it responds by stopping its internal clock. After a hard reset, this signal is negated.
SMI#	out	**SYSTEM MANAGEMENT INTERRUPT:** SMI# is asserted by the ESC in response to one of many maskable hardware or software events. For 3.3V processors that are not 5V tolerant, SMI# is driven to the CPU SMI# pin through a 5V to 3.3V translator The CPU recognizes the falling edge of SMI# as the highest priority interrupt in the system. The CPU responds by entering SMM (System Management Mode). SMI# is negated during and following reset. After a hard reset, this signal is negated.

Pin Name	Type	Description
EXTSMI#	in	**EXTERNAL SYSTEM MANAGEMENT INTERRUPT:** EXTSMI# is a falling edge triggered input to the ESC indicating that an external device is requesting the system to enter SMM mode. When enabled via the SMI Enable Register, a falling edge on EXTSMI# results in the assertion of the SMI# signal to the CPU. EXTSMI# is an asynchronous input to the ESC.
INIT/TEST	in	**INITIALIZE/TEST:** On the 82374SB, the function of this pin is selected by the value on the GPCS0# pin at reset. If GPCS0# is low, INIT is selected and if GPCS0# is high, TEST is selected. On the 82374EB, this pin only functions as the TEST pin. **INIT** INIT is connected to the INIT pin on the CPU and indicates to the ESC that a CPU soft reset is occuring. When asserted, the ESC ensures that STPCLK# is negated when the CPU comes out of the soft reset. The ESC also blocks SMI# generation when INIT is asserted. **TEST** For TEST signal description, see the TEST signal section.
STPGNT#	in	**STPCLK# GRANT:** When asserted, STPGNT# indicates to the ESC that a Stop grant PCI special cycle was recognized by the PCEB. The ESC may then negate the STPCLK# signal when the STPCLK# Timer expires.

2.10 ESC/PCEB Interface Signals

2.10.1 ARBITRATION AND INTERRUPT ACKNOWLEDGE CONTROL

Pin Name	Type	Description
EISAHOLD	out	**EISA HOLD:** EISAHOLD is used to request control of the EISA bus from its default owner, the PCEB. This signal is synchronous to PCICLK and is asserted when RESET# is asserted.
EISAHLDA	in	**EISA HOLD ACKNOWLEDGE:** EISAHLDA in used by the PCEB to inform the ESC that it has been granted ownership of EISA bus. This signal is synchronous to PCICLK.
PEREQ#/ INTA#	in	**PCI TO EISA REQUEST OR INTERRUPT ACKNOWLEDGE:** PEREQ#/INTA# is a dual function signal. The context of the signal pin is determined by the state of EISAHLDA signal. When EISAHLDA is deasserted this signal has the context of Interrupt Acknowledge i.e. if PEREQ#/INTA# is asserted it indicates to the ESC that current cycle on the EISA is an interrupt acknowledge. When EISAHLDA is asserted this signal has the context of PCI-to-EISA Request i.e. if PEREQ#/INTA# is asserted it indicates to the ESC that PCEB needs to obtain the ownership of the EISA bus on behalf of an PCI agent. This signal is synchronous to the PCICLK and it is driven inactive when RESET# is asserted.

2.10.2 PCEB BUFFER COHERENCY CONTROL

Pin Name	Type	Description
NMFLUSH#	t/s	**NEW MASTER FLUSH:** NMFLUSH# is a bi-directional signal which is used to provide handshake between PCEB and ESC to control flushing of system buffers on behalf of EISA masters. During an EISA bus ownership change, before ESC can grant the bus to the EISA master (or DMA) it must ensure that system buffers are flushed and buffers pointing (potentially) towards EISA subsystem are disabled. The ESC asserts NMFLUSH# signal for one PCI clock indicating the request for system buffer flushing. (After driving NMFLUSH# asserted for 1 PCI clock the ESC tri-states NMFLUSH# signal.) When PCEB samples NMFLUSH# asserted it starts immediately to drive NMFLUSH# asserted and initiates internal and external requests for buffer flushing. After all buffers have been flushed (indicated by the proper handshake signals), the PCEB negates NMFLUSH# for 1 PCI clock and stops driving it. When the ESC samples the signal deasserted that indicates that all system buffers are flushed, it grants EISA bus to an EISA master (or DMA). The ESC resumes responsibility of default NMFLUSH# driver and starts driving NMFLUSH# deasserted until the next time a new EISA master (or DMA) wins arbitration. This signal is synchronous with PCICLK and is negated by the ESC at reset.
AFLUSH#	t/s	**APIC FLUSH:** AFLUSH# is bi-directional signal between the PCEB and ESC that controls system buffer flushing on behalf of the APIC. After a reset the ESC negates AFLUSH# until the APIC is initialized and the first interrupt request is recognized.
SDCPYUP	out	**SYSTEM (DATA) COPY UP:** SDCPYUP is used to control the direction of the byte copy operation. A High on the signal indicates a COPY UP operation where the lower byte lower word of the SD data bus is copied on to the higher byte or higher word of the bus. A Low on the signal indicates a COPY DOWN operation where the higher byte(s) of the data bus are copied on to the lower byte(s) of the bus. The PCEB uses the signal to perform the actual data byte copy operation during mis-matched cycles.
SDOE[2:0]#	out	**SYSTEM DATA OUTPUT ENABLES:** SDOE# enable the SD data output of the PCEB Data Swap Buffers on to EISA bus. The ESC activates these signals only during mis-matched cycles. The PCEB uses these signal to enable the SD data buffers as follows: SDOE0#: Enables byte lane 0 SD[7:0] SDOE1#: Enables byte lane 1 SD[15:8] SDOE2#: Enables byte lane 2 SD[23:16] and byte lane 3 SD[31:24]
SDLE[3:0]#	out	**SYSTEM DATA LATCH ENABLES:** SDLE[3:0]# enable the latching of EISA data bus These signals are activated only during mis-matched cycles except PCEB initiated write cycle. The PCEB uses these signals to latch the SD data bus as follows: SDLE0#: Latch byte lane 0 SD[7:0] SDLE1#: Latch byte lane 0 SD[15:8] SDLE2#: Latch byte lane 0 SD[23:16] SDLE3#: Latch byte lane 0 SD[31:24]

2.11 Integrated Logic Signals

2.11.1 EISA ADDRESS BUFFER CONTROL

Pin Name	Type	Description
SALE#	out	**SA LATCH ENABLE:** SALE# is directly connected to F543s which buffer the LA addresses from the SA addresses. The rising edge of SALE# latches the LA address bit LA[19:2] to the SA address bit SA[19:2].
LASAOE#	out	**LA TO SA ADDRESS OUTPUT ENABLE:** LASAOE# is directly connected to the SA output buffer enables of the F543s. The ESC asserts LASAOE# during EISA master cycles. When LASAOE# is asserted, the LA to SA output buffers of the F543s are enabled.
SALAOE#	out	**SA TO LA ADDRESS OUTPUT ENABLE:** SALAOE# is connected to the LA output buffer enables of the F543s. This signal functionally is the exact opposite of LASAOE# signals. The ESC asserts SALAOE# during ISA master cycles. When LASAOE# is asserted, the SA to LA output buffers of the F543s are enabled.

2.11.2 COPROCESSOR INTERFACE

Pin Name	Type	Description
FERR#	in	**NUMERIC CO-PROCESSOR ERROR:** FERR# signal is tied to the Co-processor error signal of the CPU. If FERR# is asserted (Co-processor error detected by the CPU), an internal IRQ13 is be generated and the INTR from the ESC will be asserted.
IGNNE#	out	**IGNORE NUMERIC ERROR:** IGNNE# is tied to the ignore numeric error pin of the CPU. IGNNE# is asserted and internal IRQ13 is negated from the falling edge of IOWC# during an I/O write to location 00F0h. IGNNE# will remain asserted until FERR# is negated. During reset, this signal is driven low.

2.11.3 BIOS INTERFACE

Pin Name	Type	Description
LBIOSCS#	out	**LATCHED BIOS CHIP-SELECT:** LBIOSCS# indicates the that the current address is for the system BIOS. The ESC generates this signal by decoding the EISA LA addresses. The ESC uses a transparent latch to latch the decoded signal. The LBIOSCS# is latched on the falling edge of BALE and qualified with REFRESH#.

2.11.4 KEYBOARD CONTROLLER INTERFACE

Pin Name	Type	Description
KYBDCS#	out	**KEYBOARD CHIP SELECT:** KYBDCS# is connected to the chip select of the 82C42. KYBDCS# is active for I/O addresses 0060h and 0064h.
ALTRST#	out	**ALTERNATE RESET:** ALTRST# is used to reset the CPU under program control. This signal is AND'ed together externally with the reset signal (RSTAR#) from the keyboard controller to provide a software means of resetting the CPU. This provides a faster means of reset than is provided by the Keyboard controller. Writing a 1 to bit 0 in the Port 92 register will cause this signal to pulse active (low) for approximately 4 BCLK's. Before another ALTRST# pulse can be generated, bit 0 must be written back to a 0. During reset, this signal is driven low.
ALTA20	out	**ALTERNATE A20:** ALTA20 is used to force A20M# to the CPU low for support of real mode compatible software. This signal is externally OR'ed with the ALTA20 signal from the Keyboard controller and CPURST to control the A20M# input of the CPU. Writing a "0" to bit 1 of Port 92h Register will force ALTA20 inactive (low). This in turn will drive A20M# to the CPU low, if A20GATE from the keyboard controller is also low. Writing a "1" to bit 1 of the Port 92h Register will force ALTA20 active (high), which in turn will drive A20M# to the CPU high, regardless of the state of ALTA20 from the keyboard controller. Upon reset, this signal is driven low.
ABFULL	in	**AUXILIARY BUFFER FULL:** ABFULL is tied directly to the ABFULL signal on the keyboard controller on the system board. This signal indicates that the keyboard controller auxiliary buffer for the mouse interface is full. See the CLKDIV Register desciption for programming the ABFULL function. If this function is not used, ABFULL should be tied low through a 1K resistor.

2.11.5 REAL TIME CLOCK INTERFACE

Pin Name	Type	Description
RTCALE	out	**REAL TIME CLOCK ADDRESS LATCH ENABLE:** RTCALE is directly connected to the system Real Time Clock. The RTC uses this signal to latch the appropriate memory address. A write to port 070h with the appropriate Real Time Clock memory address that will be written to or read from will cause RTCALE to go active.
RTCRD#/ PIRQ3#	out	**REAL TIME CLOCK READ COMMAND/PCI INTERRUPT REQUEST 3:** This signal pin has two functions and the function is selected via the Mode Select Register. When functioning as RTCRD#, this signal is asserted for I/O reads from address 0071h. If the Power On Password protection is enabled (I/O Port 92h bit 3 = 1), then for accesses to RTC addresses 36h–3Fh (Port 70h), RTCRD# will not be asserted. For details on PIRQ3#, see the Mode Select Register description. For the PIRQ3# function, an external pull-up resistor (10-20 K) must be added to this signal.

Pin Name	Type	Description
RTCWR#/ PIRQ2#	out	**REAL TIME CLOCK WRITE COMMAND/PCI INTERRUPT REQUEST 2:** This signal pin has two functions, and the function is selected via the Mode Select Register. When functioning as RTCWR#, this signal is asserted for I/O writes to address 0071h. If the Power On Password protection is enabled (I/O Port 92h bit 3 = 1) then for accesses to RTC addresses 36h–3Fh (Port 70h) RTCWR# will not be generated. For details on PIRQ2#, see the Mode Select Register description. For the PIRQ2# function, an external pull-up resistor (10K–20K) must be added to this signal.

2.11.6 FLOPPY DISK CONTROLLER INTERFACE

Pin Name	Type	Description
FDCCS#/ PIRQ1#	out	**FLOPPY DISK CONTROLLER CHIP SELECT/PCI INTERRUPT REQUEST 1:** This signal has two functions and the function is selected via the Mode Select Register. As FDCCS# is asserted for I/O cycles to the floppy drive controller. When functioning as FDCCS#, this signal is also asserted when IDECS1# is decoded. See the Mode Select Register description for details on the PIRQ1# function of this signal. Note that for the PIRQ1# function, an external pull-up resistor (10 KΩ–20 KΩ) must be added to this signal.
DSKCHG	in	**DISK CHANGE:** DSKCHG signal is tied directly to the DSKCHG signal of the floppy controller. This signal is inverted and driven onto system data line 7 (SD7) during I/O read cycles to floppy address locations 3F7h (primary) or 377h (secondary) as indicated by the table below. Note that the primary and secondary locations are programmed in the X-Bus Address Decode Enable/Disable Register "A". **FDCCS# Decode** / **IDECSx# Decode** / **State of SD7 (output)** / **State of XBUSOE#** Enabled / Enabled / Tri-stated / Enabled Enabled / Disabled / Driven via DSKCHG / Disabled Disabled / Enabled / Tri-stated / Disabled (note) Disabled / Disabled / Tri-stated / Disabled **NOTE:** This mode is not supported because of potential contention between the X-Bus buffer and a floppy on the ISA bus driving the system bus at the same time during shared I/O accesses. This signal is also used to determine if the floppy controller is present on the X-Bus. It is sampled on the trailing edge of RESET, and If high, the Floppy is present. For systems that do not support a Floppy via the ESC, this pin should strapped low. If sampled low, the SD7 function, and XBUSOE# will not be enable for accesses to the floppy disk controller.

Pin Name	Type	Description
DLIGHT#/ PIRQ0#	out	**FIXED DISK ACTIVITY Light/PCI INTERRUPT REQUEST 0:** This signal has two functions, depending on the programming of the Mode Select Register. As DLIGHT#, this signal controls the fixed disk X light. When low, the light is on. When high, the light is off. If either bit 6 or bit 7 of the Port 92 register is set to a 1 (bit 6 and 7 are internally NOR'ed together), DLIGHT# is driven active (low). Setting both bits 6 and 7 low will cause DLIGHT# to be driven high. For the PIRQ0# function, see the Mode Select Register description. Note that for the PIRQ0# function, an external pull-up resistor (10 KΩ–20 KΩ) must be added to this signal.

2.11.7 CONFIGURATION RAM INTERFACE

Pin Name	Type	Description
CRAMRD#	out	**CONFIGURATION RAM READ COMMAND:** CRAMRD# is connected directly to the system Configuration RAM. The ESC asserts CRAMRD# for I/O reads from the address range programmed into the low and high bytes of the configuration RAM command registers.
CRAMWR#	out	**CONFIGURATION RAM WRITE COMMAND:** This is an active Low output. CRAMWR# is connected directly to the system Configuration RAM. The ESC activates CRAMWR# for I/O writes to the address range programmed into the low and high bytes of the configuration RAM command registers.

2.11.8 X-BUS CONTROL AND GENERAL PURPOSE DECODE

Pin Name	Type	Description
XBUST/R#	out	**X-BUS DATA TRANSMIT/RECEIVE:** XBUST/R# is tied directly to the direction control of a 74F245 that buffers the X-Bus data, XD(7:0), from the system data bus, SD(7:0). XBUST/R# is driven high (transmit) during I/O and memory reads for EISA and ISA masters. For DMA cycles (channel 2 only), XBUST/R# is driven high for the following cases: 1. Memory read, I/O write cycles where LBIOSCS# is asserted. 2. I/O read, memory write cycles where Digital Output Register bit 3 is set to 1. XBUST/R# is driven low (receive) under all other conditions.

Pin Name	Type	Description
XBUSOE#	out	**X-BUS DATA OUTPUT ENABLE:** XBUSOE# is tied directly to the output enable of a 74F245 that buffers the X-Bus data, XD(7:0), from the system data bus, SD(7:0). For EISA and ISA master memory read or write cycles, XBUSOE# is asserted when LBIOSCS# is asserted. Otherwise, XBUSOE# is not asserted. For EISA and ISA master I/O read or write cycles, SBUSOE# is asserted if an ISC supported X-Bus device has been decoded, and the decoding for that device has been enabled via the proper configuration registers. An exception to this is during an I/O read access to floppy location 3F7h (primary) or 377h (secondary) if the IDE decode space is disabled (i.e., IDE is not present on the X-Bus). In this case, XBUSOE# is not asserted. XBUSOE# is also not asserted during an I/O access to the floppy controller if DSKCHG is sampled low at reset. XBUSOE# is not asserted during DMA cycles, except for channel 2 DMA. For channel 2 DMA, XBUSOE# is asserted.
GPCS[2:0]# / ECS[2:0]	out	**GENERAL PURPOSE CHIP SELECT/ENCODED CHIP SELECT:** These are dual function signals. The function of these pins is selected through the Mode Select Register bit 4. **General Purpose Chip Select:** GPCS[2:0]# are chip selects for peripheral devices. The peripheral devices can be mapped in the I/O range by programming the General Purpose Chip Select Base Address registers and General Purpose Mask registers (offset 64h-6Eh). **Encoded Chip Select:** ECS[2:0] provide encoded chip select decoding for serial ports, parallel port, IDE and general purpose devices. The device chip selects for the peripheral devices are generated by using a F138 with ECS[2:0] as inputs. **Hardware Reset (Test Mode)** **82374SB:** During Reset, GPCS0/ECS0 is an input signal. The level of this signal is sampled at the end of the reset sequence to determine whether the TEST pin is used as the current TEST function (sampled "1") or as the INIT signal (sampled "0"). After reset, the existing GPCS/ECS functionality on this pin is maintained. Note that an internal pull-up of approximately 8 KΩ is included on this pin. If the INIT mode on the TEST pin is to be selected, an external pull-down of approximately 500Ω should be connected to the pin.

2.12 Test Signal

Pin Name	Type	Description
INIT/TEST	in	**TEST:** On the 82374EB, this pin only functions as a TEST pin. On the 82374SB, the function of this pin is selected by the value on the GPCS0# pin at reset. If GPCS0# is low, INIT is selected and if GPCS0# is high, TEST is selected. **INIT** For INIT signal description, see the Power Management Signal section. **TEST** TEST is used to tri-state all of the outputs. For normal operations, this signal should be tied to V_{CC}. For test mode, this pin should be tied to ground.

3.0 REGISTER DESCRIPTION

The ESC contains ESC configuration registers, DMA registers, Timer Unit registers, Interrupt Unit registers, and EISA configuration registers. All of the registers are accessible from the EISA bus. During a reset the ESC sets its internal registers to predetermined **default** states. The default values are indicated in the individual register descriptions.

The following notation is used to describe register access attributes:

RO **Read Only**. If a register is read only, writes have no effect.

WO **Write Only**. If a register is write only, reads have no effect.

R/W **Read/Write**. A register with this attribute can be read and written. Note that individual bits in some read/write registers may be read only.

3.1 Configuration Registers

The ESC's configuration registers are accessed through an indexing scheme. The index address register is located at I/O address 0022h, and the index data register is located at I/O address 0023h. The offset (data) written into the index address register selects the desired configuration register. Data for the selected configuration register can be read from or written to by performing a read or a write to the index data register. See the Address Decode section for a summary of configuration register index addresses.

Some of the ESC registers described in this section contain reserved bits. These bits are labeled "R". Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back.

In addition to reserved bits within a register, the ESC's configuration space contains address locations that are marked "Reserved" (See Address Decode Section). The ESC responds to accesses to these address locations by completing the Host cycle. When a reserved register location is read, 0000h is returned. Writes to reserved registers have no effect on the ESC.

3.1.1 ESCID—ESC ID REGISTER

Address Offset: 02h
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

Since the ESC configuration registers are accessed by the index addressing mechanism using I/O Ports 22h, and 23h, it is possible that another device in the system might use the same approach for configuration. In order to avoid contention with similar index register devices, the ID register must be written with 0Fh. The ESC will not respond to accesses to any other configuration register until the ID byte has been written in the ESC ID Register.

Bit	Description
7:0	**ESC ID Byte:** These bits must be written to a value of 0Fh before the ESC will respond to any other configuration register access. After a reset has occurred all of the configuration registers, except this register, are disabled.

3.1.2 RID—REVISION ID REGISTER

Address Offset: 08h
Default Value: 02h (82374EB: A-2 stepping)
03h (82374SB: B-0 stepping)
Attribute: Read only
Size: 8 Bits

This 8-bit register contains device stepping information. Writes to this register have no effect.

Bit	Description
7:0	**Revision ID Byte:** These bits contain the stepping information about the device. The register is hardwired to the default value during manufacturing. The register is read only. Writes have no effect on the register value.

3.1.3 MS—MODE SELECT REGISTER

Address Offset: 40h
Default Value: 20h
Attribute: Read/Write
Size: 8 Bits

This register selects the various functional modes of the ESC.

Bit	Description
7	**Reserved**
6	**MREQ[7:4]#/PIRQ[3:0]# Enable:** This bit enables the selected (MREQ[7:4]#/PIRQ[3:0]# functionality. 1 = Enabled; 0 = Disabled
5	**Configuration RAM Address:** This bit is used to enable or disable the configuration RAM Page Address (CPG[4:0]) generation. If this bit is set to 1, accesses to the configuration RAM space will generate the RAM page address on the LA[31:27]# pins. If this bit is set to 0, the CPG[4:0] signals will not be activated. The default for this bit is 1.

Pin Name	Description
4	**General Purpose Chip Selects:** This bit is used to select the functionality of the GPCS[2:0]#/ ECS[2:0] pins. If the bit is set to 0, the GPCS[2:0]# functionality is selected. If the bit is set to 1, the ESC[2:0] functionality is selected.
3	**System Error:** This bit is used to disable (0) or enable (1) the generation of NMI based on SERR# signal pulsing active. When this bit = 1 (and NMIs are enabled via the NMIERTC Register) and SERR# is asserted, the NMI signal is asserted. When this bit = 0, the NMI signal is negated and SERR# is disabled from generating an NMI. Note that other NMI sources are enabled/disabled via the NMISC Register.
2:0	**PIRQx Mux/Mapping Control:** These bits select muxing/mapping of PIRQ[3:0]# with MREQ[7:4] and group of X-Bus signals (DLIGHT#, RTCWR#, RTCRD#). Different bit combinations select the number of EISA slots or group of X-Bus signals which can be supported with the certain number of PIRQx# signals by determining the functionality of pins AEN[4:1]/EAEN[4:1], MACK[3:0]#/EMACK[3:0]#, MREQ[7:4]#/PIRQ[3:0]#, DLIGHT#/ PIRQ0#, FDCCS#/PIRQ1#, RTCWR#/PIRQ2#, RTCRD#/PIRQ3# as shown in Table 1.

Table 1. Mode Select Register

Bits [2:0]	Signal Function						
	AEN[4:1]/ EAEN[4:1]#	**MACK[3:0]#/ EMACK[3:0]#**	**MREQ[7:4#/ PIRQ[0:3]#**	**DLIGHT#/ PIRQ0#**	**FDDCS#/ PIRQ1#**	**RTCWR#/ PIRQ2#**	**RTCRD#/ PIRQ3#**
000	EAEN[4:1]#	EMACK[3:0]#	MREQ[7:4#	PIRQ0#	PIRQ1#	PIRQ2#	PIRQ3#
001	EAEN[4:1]#	EMACK[3:0]#	MREQ[7:4#	PIRQ0#	PIRQ1#	RTCWR#	RTCRD#
010	EAEN[4:1]#	EMACK[3:0]#	MREQ[7:4#	PIRQ0#	FDDCS#	RTCWR#	RTCRD#
011	EAEN[4:1]#	EMACK[3:0]#	MREQ[7:4#	DLIGHT#	FDDCS#	RTCWR#	RTCRD#
100	AEN[4:1]	MACK[3:0]#	PIRQ[0:3]#	DLIGHT#	FDDCS#	RTCWR#	RTCRD#
101	EAEN[4:1]#	EMACK[3:0]#	PIRQ0#, PIRQ1#, MREQ5#, MREQ4#	DLIGHT#	FDDCS#	RTCWR#	RTCRD#
110	EAEN[4:1]#	EMACK[3:0]#	PIRQ0#, MREQ6#, MREQ5#, MREQ4#	DLIGHT#	FDDCS#	RTCWR#	RTCRD#
111	EAEN[4:1]#	EMACK[3:0]#	MREQ[7:4#	DLIGHT#	FDDCS#	RTCWR#	RTCRD#

3.1.4 BIOSCSA—BIOS CHIP SELECT A REGISTER

Address Offset: 42h
Default Value: 10h
Attribute: Read/Write
Size: 8 Bits

The LBIOSCS# signal is used to decode access to the motherboard BIOS. The ESC decodes memory access to the following address ranges, and if the range has been enabled the LBIOSCS# signal is always asserted for memory reads in the enabled BIOS range. If the BIOS Write Enable bit is set in the configuration register BIOSCSB, the LBIOSCS# is also asserted for memory write cycles.

Bit	Description
7:6	**Reserved**
5	**Enlarged BIOS:** During Memory access to locations FFF80000h–FFFDFFFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.
4	**High BIOS:** During Memory access to locations 0F0000h–0FFFFFh, FF0000h–FFFFFFh, FFFF0000h–FFFFFFFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.
3	**Low BIOS 4:** During Memory access to locations 0EC000h–0EFFFFh, FFEEC000h–FFEEFFFFh, FFFEC000h–FFFEFFFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.
2	**Low BIOS 3:** During Memory access to locations 0E8000h–0EBFFFh, FFEE8000h–FFEEBFFFh, FFFE8000h–FFFEBFFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.
1	**Low BIOS 2:** During Memory access to locations 0E4000h–0E7FFFh, FFEE4000h–FFEE7FFFh, FFFE4000h–FFFE7FFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.
0	**Low BIOS 1:** During Memory access to locations 0E0000h–0E3FFFh, FFEE0000h–FFEE3FFFh, FFFE0000h–FFFE3FFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.

3.1.5 BIOSCSB—BIOS CHIP SELECT B REGISTER

Address Offset: 43h
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

The LBIOSCS# signal is used to decode access to the motherboard BIOS. The ESC decodes memory access to the following address ranges, and if the range has been enabled the LBIOSCS# signal is always asserted for memory reads in the enabled BIOS range. If the BIOS Write Enable bit is set in the configuration register BIOSCSB, the LBIOSCS# is also asserted for memory write cycles.

Bit	Description
7:4	**Reserved**
3	**BIOS Write Enable:** When enabled LBIOSCS# is asserted for memory read AND write cycles for addresses in the decoded and enabled BIOS range, otherwise LBIOSCS# is asserted for memory read cycles ONLY.
2	**16 Meg BIOS:** During Memory access to locations FF0000h–FFFFFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.
1	**High VGA BIOS:** During Memory access to locations 0C4000h–0C7FFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.
0	**Low VGA BIOS:** During Memory access to locations 0C0000h–0C3FFFh with this bit set, LBIOSCS# will be asserted for memory read cycles. If bit 3 of BIOSCSB is set, then LBIOSCS# will be asserted for write cycles as well.

3.1.6 CLKDIV—EISA CLOCK DIVISOR REGISTER

Address Offset: 4Dh
Default Value: xx001000b
Attribute: Read/Write
Size: 8 Bits

This register is used to select the integer value used to divide the PCI clock (PCICLK) to generate the EISA Bus Clock (BCLK) and enable/disable the co-processor error support. In addition, for the 82374SB, the register controls the ABFULL and KBFULL functions.

Bit	Description
7:6	**Reserved**
5	**Co-processor Error:** The state of this bit determines if the FERR# signal is connected to the ESC internal IRQ13 interrupt signal. If this bit is set to 1, the ESC will assert IRQ13 to the interrupt controller if FERR# signal is asserted. If this bit is set to 0, then the FERR# signal is ignored by the ESC (i.e. this signal is not connected to any logic in the ESC).
4	**82374EB: Reserved** **82374SB: ABFULL (With IRQ12):** When bit 4 = 0, the internal IRQ12 is directed to the interrupt controller and transitions on ABFULL have no affect on this interrupt signal. When bit 4 = 1, the assertion of ABFULL is latched and directed to the internal IRQ12 signal in the following manner: • If the interrupt controller is programmed for edge detect mode on IRQ12, a low-to-high transition is generated on the internal IRQ12 signal. Transitions on the IRQ12 input pin are not reflected on the internal IRQ12 signal. • If the interrupt controller is programmed for level-sensitive mode, a high-to-low transition is generated on the internal IRQ12 signal. Transitions on the IRQ12 input pin are also reflected on the internal IRQ12 signal. The latching of the ABFULL signal is cleared by an I/O read of address 60h (no aliasing) or by a hard reset.
3	**82374EB: Reserved** **82374SB: Keyboard Full (KBFULL):** This bit selects the edge-detect KBFULL function on the IRQ1 input signal. When bit 3 = 0, IRQ1 is directed to the interrupt controller. When bit 3 = 1 (default), IRQ1 is latched and directed to the interrupt controller. The latched IRQ1 is cleared by an I/O read of address 60h (no aliasing) or by a hard reset.
2:0	**Clock Divisor:** These bits are used to select the integer that is used to divide the PCICLK down to generate the BCLK. Upon reset, these bits are set to 000b (divisor of 4). **Bit[2:0] — Divisor — BCLK** 000 — 4 (33.33 MHz) — 8.33 MHz 001 — 3 (25 MHz) — 8.33 MHz 010 — Reserved 011 — Reserved 1xx — Reserved

3.1.7 PCSA—PERIPHERAL CHIP SELECT A REGISTER

Address Offset: 4Eh
Default Value: x0000111b
Attribute: Read/Write
Size: 8 Bits

This register is used to enable or disable accesses to the RTC, keyboard controller, Floppy Disk controller, and IDE. Disabling any of these bits will prevent the chip select and X-Bus transceiver control signal (XBUSOE#) for that device from being generated. This register is also used to select which address range (primary or secondary) will be decoded for the resident floppy controller and IDE. It also allows control of where the keyboard controller is physically located (X-Bus or elsewhere).This insures that there is no contention with the X-Bus transceiver driving the system data bus during read accesses to these devices.

Bit	Description
7	**Reserved**
6	**Keyboard Controller Mapping:** 0 = keyboard controller mapped to the X-Bus (default). 1 = keyboard controller not mapped to the X-Bus. When bit 6 = 0, the keyboard controller encoded chip select signal and the X-Bus transceiver enable (XBUSOE#) are generated for accesses to address locations 60h (82374EB/SB), 62h (82374EB only), 64h (82374EB/SB) and 66h (82374EB only). When bit 6 = 1, the keyboard controller chip select signals are generated for accesses to these address locations. However XBUSOE# is disabled. Bit 1 must be 1 for either value of this configuration bit to decode an access to locations 60h, 62h, 64h, or 66h.
5,3:2	**Floppy Disk and IDE, Floppy Disk Decodes:** Bits 2 and 3 are used to enable or disable the floppy locations as indicated. Bit 2 defaults to enabled (1) and bit 3 defaults to disabled (0) when a reset occurs. Bit 5 is used to select between the primary and secondary address range used by the Floppy Controller and the IDE. Only primary or only secondary can be programmed at any one time. This bit defaults to primary (0). The following table shows how these bits are used to select the floppy controller:

Address	Bit 2	Bit 3	Bit 5	DSKCHG	FDCCS#
X	X	X	X	0	1
3F0h,3F1h	X	1	0	1	0
3F2h–3F7h	1	X	0	1	0 (Note)
370h,371h	X	1	1	1	0
372h–37Fh	1	X	1	1	0 (Note)

NOTE:
If IDE decode is enabled, all accesses to locations 03F6h and 03F7h (primary) or 0376h and 0377h (secondary) will result in decode for IDECS1# (FDCCS# will not be generated). An external AND gate can be used to tie IDECS1# and FDCCS# together to insure that the floppy is enabled for these accesses.

Pin Name	Description
4	**IDE DECODE:** Bit 4 is used to enable or disable IDE locations 1F0h–1F7h (primary) or 170h–177h (secondary) and 3F6h,3F7h (primary) or 376h,377h (secondary). **82374EB:** When this bit is set to 0, the IDE encoded chip select signals and the X-Bus transceiver signal (XBUSOE#) are not generated for these addresses. **82374SB:** When this bit is set to 0, the IDE encoded chip select signals and the X-Bus transceiver signal (XBUSOE#) are not generated for addresses 1F0h–1F7h (primary) or 170h–177h (secondary) and 3F6h, or 376h. Note that read/write accesses to addresses 377h and 3F7h are not disabled and still generate XBUSOE#.
1	**KEYBOARD CONTROLLER DECODE:** Enables (1) or disables (0) the keyboard controller address locations 60h (82374EB/SB), 62h (82374EB only), 64h (82374EB/SB), and 66h (82374EB only). When this bit is set to 0, the keyboard controller encoded chip select signals and the X-Bus transceiver signal (XBUSOE#) are not generated for these locations. Note that the value of this bit affects control function (keyboard controlling mapping) provided by bit 6 of this register.
0	**Bit 0: REAL TIME CLOCK DECODE:** Enables (1) or disables (0) the RTC address locations 70h–77h. When this bit is set to 0, the RTC encoded chip select signals RTCALE, RTCRD, RTCWR#, and XBUSOE# signals are not generated for these addresses.

3.1.8 PCSB—PERIPHERAL CHIP SELECT B REGISTER

Address Offset: 4Fh
Default Value: CFh
Attribute: Read/Write
Size: 8 Bits

This register is used to enable or disable generation of the X-Bus transceiver signal (XBUSOE#) for accesses to the serial ports and parallel port locations. When disabled, the XBUSOE# signal for that device will not be generated.

Bit	Description
7	**CRAM Decode:** This bit is used to enable (1) or disable (0) I/O write accesses to location 0C00h and I/O read/write accesses to locations 0800h–08FFh. The configuration RAM read and write (CRAMRD#, CRAMWR#) strobes are valid for accesses to 0800h–08FFh.
6	**Port 92 Decode:** This bit is used to disable (0) access to Port 92. This bit defaults to enable (1) at PCIRST.
5:4	**Parallel Port Decode:** These bits are used to select which Parallel Port address range (LPT1, 2,or 3) is decoded. **Bits[5:4]** **Decode** 00 LPT1 (3BCh–3BFh) 01 LPT2 (378h–37Fh) 10 LPT3 (278h–27Fh) 11 Disabled

Pin Name	Description
3:2	**Serial Port B Address Decode:** If either COM1 or COM2 address ranges are selected, these bits default to disabled upon PCIRST. **Bits[3:2] Decode** 00 3F8h–3FFh (COM1) 01 2F8h–2FFh (COM2) 10 Reserved 11 Port A disabled
1:0	**Serial Port A Address Decode:** If either COM1 or COM2 address ranges are selected, these bits default to disabled upon PCIRST. **Bits[1:0] Decode** 00 3F8h–3FFh (COM1) 01 2F8h–2FFh (COM2) 10 Reserved 11 Port A disabled

3.1.9 EISAID[4:1]—EISA ID REGISTERS

Address Offset: 50h, 51h, 52h, 53h
Default Value: 00h, 00h, 00h, 00h
Attribute: Read/Write
Size: 8 Bits each

These 8 bit registers contain the EISA motherboard ID. The data in the register is reflected on the data bus for I/O cycles addressed to 0C80h–0C83h respectively.

Bit	Description
7:0	**EISA ID Byte:** These bits contain the EISA Motherboard ID information. On power up these bits default to 00h. These bit are written with the ID value during configuration. The value of these bits are reflected in I/O registers 0C80h–0C83h.

3.1.10 SGRBA—SCATTER/GATHER RELOCATE BASE ADDRESS REGISTER

Address Offset: 57h
Default Value: 04h
Attribute: Read/Write
Size: 8 Bits

The value programmed in this register determines the high order I/O address of the S-G registers. The default value is 04h.

Bit	Description
7:0	**S-G Relocate Byte:** These bits determine the I/O location of the Scatter Gather Registers. The Scatter-Gather register relocation range is xx10h–xx3Fh (default 0410h–043Fh). These bits determine the Byte 1 of the I/O address. Address signals LA[15:8] are compared against the contents of this register (bit[7:0]) to determine I/O accesses to the Scatter-Gather registers. The default on Power up is 04h.

3.1.11 APICBASE—APIC BASE ADDRESS RELOCATION

Address Offset: 59h
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

The APICBASE Register provides the modifier for the APIC base address.

82374EB:
APIC is mapped in the CPU memory space at the locations FEC0__x000h and FEC0__x010h (x = 0-Fh). The value of "x" is defined by bits[5:2]. Thus, the relocation register provides a 4 KByte address granularity. The default value of 00h provides APIC unit mapping at the addresses FEC00000h and FEC00010h.

82374SB:
APIC is mapped in the CPU memory space at the locations FEC0__xy00h and FEC0__xy10h (x = 0-Fh, y = 0,4,8,Ch). The value of "y" is defined by bits[1:0] and value of "x" is defined by bits[5:2]. Thus, the relocation register provides a 1 KByte address granularity (i.e. potentially up to 64 I/O APICs can be uniformly addresses in the memory space). The default value of 00h provides APIC unit mapping at the addresses FEC00000h and FEC00010h.

Bit	Description
7:6	**Reserved**
5:2	**X-Base Address—R/W:** Bits[5:2] are compared to host address bits A[15:12], respectively.
1:0	**82374EB: Reserved** **82374SB: Y-Base Address—R/W:** Bits[1:0] are compared to host address bits A[11:10], respectively.

3.1.12 PIRQ[0:3]#—PIRQ ROUTE CONTROL REGISTERS

Address Offset: 60h, 61h, 62h, 63h
Default Value: 80h
Attribute: Read/Write
Size: 8 Bits

These registers control the routing of PCI Interrupts (PIRQ[0:3]#) to the PC compatible interrupts. Each PCI interrupt can be independently routed to 1 of 11 compatible interrupts.

Interrupt Steering Programming Considerations
When using the PCI programmable interrupt steering feature, the following programming considerations apply:

1. Any interrupt steered to by a PIRQx# must be programmed to level sensitive mode.
2. For an interrupt used as a PIRQx#, that IRQ pin is also level sensitive. It is not permissible to use an interrupt on the EISA/ISA Bus as edge triggered as well as on the PCI Bus as level sensitive.
3. Registers that must be programmed when using a PIRQx# include the Mode Select Registers, Edge Level Registers, PIRQ[3:0]# Route Control Registers, and the Interrupt Mask Registers (listed in suggested programming order)

Bit	Description
7	**Routing of Interrupts:** When enabled (0) this bit routes the PCI Interrupt signal to the PC compatible interrupt signal specified in bits[6:0]. After a reset or a power-on this bit is disabled (set to 1).
6:0	**IRQx# Routing Bits:** These bits specify which IRQ signal to generate when the PCI Interrupt for this register has been triggered.

Bits[6:0]	IRQx#	Bits[6:0]	IRQx#
0000000	Reserved	0001001	IRQ9
0000001	Reserved	0001010	IRQ10
0000010	Reserved	0001011	IRQ11
0000011	IRQ3	0001100	IRQ12
0000100	IRQ4	0001101	Reserved
0000101	IRQ5	0001110	IRQ14
0000110	IRQ6	0001111	IRQ15
0000111	IRQ7	0010000 to	
0001000	Reserved	1111111	Reserved

3.1.13 GPCSLA[2:0]—GENERAL PURPOSE CHIP SELECT LOW ADDRESS REGISTER

Address Offset: 64h, 68h, 6Ch
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

This register contains the low byte of the General Purpose Peripheral mapping address. The contents of this register are compared with the LA[7:0] address lines. The contents of this register, the GPCSHA Register and the GPCSM Register control the generation the GPCS[2:0]# signal or the ESC[2:0] signal (101, 110 combination). If Mode Select Register (offset 40h) bit 4 = 1, offset register 6Ch is ignored.

Bit	Description
7:0	**GPCS Low Address Byte:** The contents of these bits are compared with the address lines LA[7:0] to generate the GPCS[2:0]# signal or the ECS[2:0] combination for this register. The mask register (GPCSM[2:0]) determines which bits to use during the comparison.

3.1.14 GPCSHA[2:0]—GENERAL PURPOSE CHIP SELECT HIGH ADDRESS REGISTER

Address Offset: 65h, 69h, 6Dh
Default Value: C0h
Attribute: Read/Write
Size: 8 Bits

This register contains the high byte of the General Purpose Peripheral mapping address. The contents of this register are compared with the LA[15:8] address lines. The contents of this register, the GPCSLA Register and the GPCSM Register control the generation the GPCS[2:0]# signal or the ESC[2:0] signal (101, 110 combination). If Mode Select Register (offset 40h) bit 4 = 1, offset register 6Dh is ignored.

Bit	Description
7:0	**GPCS High Address Byte:** The contents of these bits are compared with the address lines LA[15:8] to generate the GPCS[2:0]# signal or the ECS[2:0] combination for this register.

3.1.15 GPCSM[2:0]—GENERAL PURPOSE CHIP SELECT MASK REGISTER

Address Offset: 66h, 6Ah, 6Eh
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

This register contains the mask bits for determining the address range for which the GPCSx# signals are generated. If a register bit is set to a 1 then the corresponding bit in the GPCSL register is not compared with the address signal in the generation of the GPCSx# signals. If Mode Select Register (offset 40h) bit 4 = 1, offset register 6Eh is ignored.

Bit	Description
7:0	**GPCS Mask Register:** The contents of these bits are used to determine which bits to compare GPCSLA[2:0] with the address lines LA[7:0]. A 1 bit means the bit should not be compared.

3.1.16 GPXBC—GENERAL PURPOSE PERIPHERAL X-BUS CONTROL REGISTER

Address Offset: 6Fh
Default Value: xxxx x000b
Attribute: Read/Write
Size: 8 Bits

The register controls the generation of the X-BUS buffer output enable (XBUSOE#) signal for I/O accesses to the peripherals mapped in the General Purpose Chip Select address decode range. This register determines if the General Purpose Peripheral is placed on the XBUS or not. If the General Purpose Peripheral is on the X-Bus, then the corresponding bit is set to 1. Otherwise the bit is set to 0.

Bit	Description
7:3	**Reserved**
2	**XBUSOE# Generation for GPCS2#:** When this bit is enabled XBUSOE# will be generated when GPCS2# is generated; 1 = Enabled, 0 = Disabled.
1	**XBUSOE# Generation for GPCS1#:** When this bit is enabled XBUSOE# will be generated when GPCS1# is generated; 1 = Enabled, 0 = Disabled.
0	**XBUSOE# Generation for GPCS0#:** When this bit is enabled XBUSOE# will be generated when GPCS0# is generated;1 = Enabled, 0 = Disabled.

3.1.17 PAC—PCI/APIC CONTROL REGISTER

Address Offset: 70h
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

The PAC Register controls the operation of the INTR signal in APIC/PIC configuration and the routing of the System Management Interrupt (SMI).

Bit	Description
7:2	**Reserved**
1	**SMI Routing Control (SMIRC):** When SMIRC = 1, the SMI is routed via the APIC. When SMIRC = 0, the SMI is routed via the SMI# signal. Note that when SMRC = 1, INTR can not be routed through the APIC, since it is sharing the APIC interrupt input with SMI#.
0	**INTR Routing Control (INTRC):** When APIC is enabled (in mixed or pure APIC mode), this bit allows the ESC's external INTR signal to be masked (forces INTR to the inactive state but does not tri-states the signal). Thus, the CPU's INTR pin can be used (by providing a simple -gate) for the APIC Local Interrupt (LINTRx). However, INTR must not be masked via this bit when APIC is disabled and INTR is the only mechanism to signal the 8259 recognized interrupts to the CPU. When INTRC = 1, INTR is disabled (APIC must be enabled). When INTRC = 0, INTR is enabled.

3.1.18 TESTC—TEST CONTROL REGISTER

Address Offset: 88h
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

This register provides control for ESC manufacturing test modes. The functionality of this register is reserved.

3.1.19 SMICNTL—SMI CONTROL REGISTER

Address Offset: A0h
Default Value: 08h
Attribute: Read/Write
Size: 8 Bits

For the 82374SB, the SMICNTL Register provides Fast Off Timer control, STPCLK# enable/disable, and CPU clock scaling. This register also enables/disables the system management interrupt (SMI).

Bit	Description
7	**Reserved: Must be 0 when writing this register.**
6:4	**Reserved**
3	**Fast Off Timer Freeze (CTMRFRZ):** This field enables/disables the Fast Off Timer. When this bit is 1, the Fast Off timer stops counting. This prevents time-outs from occurring while executing SMM code. When this bit is 0, the Fast Off timer counts.

Pin Name	Description
2	**STPCLK# Scaling Enable (CSTPCLKSC):** This bit enables/disables control of the STPCLK# high/low times by the clock scaling timers. When bit 2 = 1, the STPCLK# signal scaling control is enabled. When enabled (and bit 1 = 1, enabling the STPCLK# signal), the high and low times for the STPCLK# signal are controlled by the Clock Scaling STPCLK# High Timer and Clock Scaling STPCLK# Low Timer Registers, respectively. When bit 2 = 0 (default), the scaling control of the STPCLK# signal is disabled.
1	**STPCLK# Signal Enable (CSTPCLKE):** This bit permits software to place the CPU into a low power state. When bit 1 = 1, the STPCLK# signal is enabled and a read from the APMC Register causes STPCLK# to be asserted. When bit 1 = 0 (default), the STPCLK# signal is disabled and is negated (high). Software can set this bit to 0 by writing a 0 to it or by any write to the APMC Register.
0	**SMI# Gate (CSMIGATE):** When bit 0 = 1, the SMI# signal is enabled and a system management interrupt condition causes the SMI# signal to be asserted. When bit 0 = 0 (default), the SMI# signal is masked and negated. This bit only affects the SMI# signal and does not effect the detection/recording of SMI events (i.e., This bit does not effect the SMI status bits in the SMIREQ Register). Thus, SMI conditions can be pending when this bit is set to 1. If an SMI is pending when this bit is set to 1, the SMI# signal is asserted.

3.1.20 SMIEN—SMI ENABLE REGISTER

Address Offset: A2-A3h
Default Value: 0000h
Attribute: Read/Write
Size: 16 Bits

For the 82374SB, his register enables the generation of SMI (asserting the SMI# signal) for the associated hardware events (bits[5:0]), and software events (bit 7). When a hardware event is enabled, the occurrence of a corresponding event results in the assertion of SMI#, if enabled via the SMICNTL Register. The SMI# is asserted independent of the current power state (Power-On or Fast Off). The default for all sources in this register is disabled.

Bit	Description
15:8	**Reserved**
7	**APMC Write SMI Enable:** This bit enables SMI for writes to the APMC Register. When bit 7 = 1, writes to the APMC Register generate an SMI. When bit 7 = 0, writes to the APMC Register do not generate an SMI.
6	**EXTSMI# SMI Enable:** When bit 6 = 1, asserting the EXTSMI# input signal generates an SMI. When bit 6 = 0, asserting EXTSMI# does not generate an SMI.
5	**Fast Off Timer SMI Enable:** This bit enables the Fast Off Timer to generate an SMI. When bit 5 = 1, the timer generates an SMI when it decrements to zero. When bit 5 = 0, the timer does not generate an SMI.
4	**IRQ12 SMI Enable (PS/2 Mouse Interrupt):** This bit enables the IRQ12 signal to generate an SMI. When bit 4 = 1, asserting the IRQ12 input signal generates an SMI. When bit 4 = 0, asserting IRQ12 does not generate an SMI.
3	**IRQ8 SMI Enable (RTC Alarm Interrupt):** This bit enables the IRQ8 signal to generate an SMI. When bit 3 = 1, asserting the IRQ8 input signal generates an SMI. When bit 3 = 0, asserting IRQ8 does not generate an SMI.

Bit	Description
2	**IRQ4 SMI Enable (COM2/COM4 Interrupt or Mouse):** Thsi bit enables the IRQ4 signal to generate an SMI. When bit 1 = 1, asserting the IRQ3 input signal generates an SMI. When bit 2 = 0, asserting IRQ4 does not generate an SMI.
1	**IRQ3 SMI Enable (COM1/COM3 Interrupt or Mouse):** This bit enables the IRQ3 signal to generate an SMI. When bit 1 = 1, asserting the IRQ3 input signal generates an SMI. When bit 1 = 0, asserting IRQ3 does not generate an SMI.
0	**IRQ1 SMI Enable (Keyboard Interrupt):** This bit enables the IRQ1 signal to generate an SMI. When bit 0 = 1, asserting the IRQ1 input signal generates an SMI. When bit 0 = 0, asserting IRQ1 does not generate an SMI.

3.1.21 SEE—SYSTEM EVENT ENABLE REGISTER

Address Offset: A4-A7h
Default Value: 00000000h
Attribute: Read/Write
Size: 32 Bits

For the 82374SB, this register enables hardware events as system events or break events for power management control. Note that all of the functional bits in the SEE Register provide system event control. In addition, all bits also provide break event control. The default for each system/break event in this register is disabled.

System events: Activity by these events can keep the system from powering down. When a system event is enabled, the corresponding hardware event activity prevents a Fast Off powerdown condition. Anytime the corresponding hardware event occurs (signal is asserted), the Fast Off Timer is re-loaded with its initial count.

Break events: These events can awaken a powered down system. When a break event is enabled, the corresponding hardware event activity powers up the system by negating STPCLK#. Note that STPCLK# is not negated until the stop grant special cycle has been generated by the CPU. Thus, from the time that STPCLK# is asserted until the stop grant cycle is returned, the occurrence of subsequent break events are latched in the ESC.

NOTE:
INIT is always enabled as a break event. However, INIT only causes a break event after a stop grant special cycle has been received. If INIT is asserted while STPCLK# is active and then negated before the stop grant cycle is received, INIT does not cause a break event.

Bit	Description
31	**Fast Off SMI Enable (FSMIEN):** When bit 31 = 1 (enabled), an SMI causes a system event that re-loads the Fast Off Timer and a break event that negates the STPCLK# signal. When bit 31 = 0 (disabled), an SMI does not re-load the Fast Off Timer or negate the STPCLK# signal.
30	**Reserved**
29	**Fast Off NMI Enable (FNMIEN):** When bit 29 = 1 (enabled), an NMI (e.g., parity error) causes a system event that re-loads the Fast Off Timer and a break event that negates the STPCLK# signal. When bit 29 = 0 (disabled), an SMI does not re-load the Fast Off Timer or negate the STPCLK# signal.

Bit	Description
28:16	**Reserved**
15:3	**Fast Off IRQ[15:3] Enable (FIRQ[15:3]EN):** These bits are used to prevent the system from entering Fast Off and break any current powerdown state when the selected hardware interrupt occurs. When a bit = 1 (enabled), the corresponding interrupt causes a system event that re-loads the Fast Off Timer and a break event that negates the STPCLK# signal. When a bit = 0 (disabled), the corresponding interrupt does not re-load the Fast Off Timer or negate the STPCLK# signal.
2	**Reserved**
1:0	**Fast Off IRQ[1:0] Enable (FIRQ[1:0]EN):** These bits are used to prevent the system from entering Fast Off and break any current powerdown state when the selected hardware interrupt occurs. When a bit = 1, the corresponding interrupt causes a system event that re-loads the Fast Off Timer and a break event that negates the STPCLK# signal. When a bit = 0 (disabled), the corresponding interrupt does not re-load the Fast Off Timer or negate the STPCLK# signal.

3.1.22 FTMR—FAST OFF TIMER REGISTER

Address Offset: A8h
Default Value: 0Fh
Attribute: Read/Write
Size: 8 Bits

For the 82374SB, the Fast Off Timer is used to indicate (through an SMI) that the system has been idle for a pre-programmed period of time. The Fast Off Timer consists of a count-down timer and the value programmed into this register is loaded into the Fast Off Timer when an enabled system event occurs. When the timer expires, an SMI special cycle is generated. When the Fast Off Timer is enabled (bit 3 = 0 in the SMICNTL Register), the timer counts down from the value loaded into this register. The count time interval is one minute. When the Fast Off Timer reaches 00h, an SMI is generated and the timer is re-load with the value programmed into this register. If an enabled system event occurs before the Fast Off Timer reaches 00h, the Fast Off Timer is re-loaded with the value in this register.

NOTE:
Before writing to the FTMR Register, the Fast Off Timer must be stopped via bit 3 of the SMICNTL Register. In addition, this register should NOT be programmed to 00h.

Bit	Description
7:0	**Fast Off Timer Value:** Bits[7:0] contain the starting count value. A read from the FTMR Register returns the value last written.

3.1.23 SMIREQ—SMI REQUEST REGISTER

Address Offset: AA-ABh
Default Value: 00h
Attribute: Read/Write
Size: 16 Bits

For the 82374SB, the SMIREQ Register contains status bits indicating the cause of an SMI. When an enabled event causes an SMI, the ESC automatically sets the corresponding event's status bit to 1. Software sets the status bits to 0 by writing a 0 to them. Only the ESC hardware can set status bits to a 1. Software

writing a 1 to any of the status bits has no effect. If software attempts to set a status bit to 0 at the same time that the ESC is setting it to 1, the bit is set to 1 (i.e., the ESC hardware dominates).

The SMI handler can query the status bits to see what caused the SMI and then branch to the appropriate routine. As the individual routines complete the handler resets the appropriate status bit by writing a 0 to the corresponding bit.

Each of the SMIREQ bits is set by the ESC in response to the activation of the corresponding SMI event. If the SMI event is still active when the corresponding SMIREQ bit is set to 0, the ESC does not set the status bit back to a 1 (i.e., there is only one status indication per active SMI event).

When an IRQx signal is asserted, the corresponding RIRQx bit is set to a 1. If the IRQx signal is still active when software sets the RIRQx bit to 0, RIRQx is not set back to a 1. The IRQx may be negated before software sets the RIRQx bit to 0. If the RIRQx bit is set to 0 at the same time a new IRQx is activated, RIRQx remains at 1. This indicates to the SMI handler that a new SMI event has been detected.

NOTE:

1. The SMIREQ bits are set, cleared, or read independently of each other and independently of the CSMIGATE bit in the SMICNTL Register.
2. If an IRQx is set in level mode and shared by two devices, the IRQ should not be enabled as an SMI# event. The ESC's SMIREQ bits are essentially set with an edge. When the second IRQ occurs on a shared IRQ, there is no second edge and the SMI# will not be generated for the second IRQ.

Bit	Description
15:8	**Reserved**
7	**APM SMI Status (RAPMC):** The ESC sets this bit to 1 to indicate that a write to the APM Control Register caused an SMI. Software sets this bit to a 0 by writing a 0 to it.
6	**EXTSMI# SMI Status (REXT):** The ESC sets this bit to 1 to indicate that EXTSMI# caused an SMI. Software sets this bit to a 0 by writing a 0 to it.
5	**Fast Off Timer Expired Status (RFOT):** The ESC sets this bit to 1 to indicate that the Fast Off Timer expired and caused an SMI. Software sets this bit to a 0 by writing a 0 to it. When the Fast Off Timer expires, the ESC sets this bit to a 1. Note that the timer re-starts counting one the next clock after it expires.
4	**IRQ12 Request SMI Status (RIRQ12):** The ESC sets this bit to 1 to indicate that IRQ12 caused an SMI. Software sets this bit to a 0 by writing a 0 to it.
3	**IRQ8# Request SMI Status:** The ESC sets this bit to 1 to indicate that IRQ8# caused an SMI. Software sets this bit to a 0 by writing a 0 to it.
2	**IRQ4 Request SMI Status:** The ESC sets this bit to 1 to indicate that IRQ4 caused an SMI. Software sets this bit to a 0 by writing a 0 to it.
1	**IRQ3 Request SMI Status:** The ESC sets this bit to 1 to indicate that IRQ3 caused an SMI. Software sets this bit to a 0 by writing a 0 to it.
0	**IRQ1 Request SMI Status:** The ESC sets this bit to 1 to indicate that IRQ1 caused an SMI. Software sets this bit to a 0 by writing a 0 to it.

3.1.24 CTLTMRCLOCK SCALE STPCLK# LOW TIMER

Address Offset: ACh
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

For the 82374SB, the value in this register defines the duration of the STPCLK# asserted period when bit 2 in the SMICNTL Register is set to 1. The value in this register is loaded into the STPCLK# Timer when STPCLK# is asserted. However, the timer does not start until the Stop Grant Bus Cycle is received. The STPCLK# timer counts using a 32 μs clock.

Bit	Description
7:0	**Clock Scaling STPCLK# Low Timer Value:** Bits[7:0] define the duration of the STPCLK# asserted period during clock throttling.

3.1.25 CTLTMRH—CLOCK SCALE STPCLK# HIGH TIMER

Address Offset: AEh
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

For the 82374SB, the value in this register defines the duration of the STPCLK# negated period when bit 2 in the SMICNTL Register is set to 1. The value in this register is loaded into the STPCLK# Timer when STPCLK# is negated. The STPCLK# timer counts using a 32 μs clock.

Bit	Description
7:0	**Clock Scaling STPCLK# High Timer Value:** Bits[7:0] define the duration of the STPCLK# negated period during clock throttling.

3.2 DMA Register Description

The ESC contains DMA circuitry that incorporates the functionality of two 82C37 DMA controllers (DMA1 and DMA2). The DMA registers control the operation of the DMA controllers and are all accessible from the EISA Bus. This section describes the DMA registers. Unless otherwise stated, a reset sets each register to its default value. The operation of the DMA is further described in Chapter 6.0, DMA Controller.

3.2.1 DCOM—COMMAND REGISTER

Register Location: 08h—Channels 0-3
0D0h—Channels 4-7
Default Value: 00h
Attribute: Write Only
Size: 8 Bits

This 8-bit register controls the configuration of the DMA. It is programmed by the microprocessor in the Program Condition and is cleared by reset or a Master Clear instruction. Note that disabling Channels 4-7 will also disable Channels 0-3, since Channels 0-3 are cascaded onto Channel 4. The DREQ and DACK# channel assertion sensitivity is assigned by channel group, not per individual Channel. For priority resolution the DMA

consists of two logical channel groups–Channels 0-3 (Controller 1-DMA1) and Channels 4-7 (Controller 2-DMA2). Both groups may be assigned fixed priority, one group can be assigned fixed priority and the second rotating priority, or both groups may be assigned rotating priority. A detailed description of the channel priority scheme is found in the DMA functional description, Section 6.5. Following a reset or DMA Master Clear, both DMA-1 and DMA-2 are enabled in fixed priority, the DREQ sense level is active high, and the DACK# assertion level is active low.

Bit	Description
7	**DACK# Assert Level:** Bit 7 controls the DMA channel request acknowledge (DACK#) assertion level. Following reset, the DACK# assertion level is active low. The low level indicates recognition and acknowledgment of the DMA request to the DMA slave requesting service. Writing a 0 to bit 7 assigns active low as the assertion level. When a 1 is written to this bit, a high level on the DACK# line indicates acknowledgment of the request for DMA service to the DMA slave.
6	**DREQ Sense Assert Level:** Bit 6 controls the DMA channel request (DREQ) assertion detect level. Following reset, the DREQ sense assert level is active high. In this condition, an active high level sampled on DREQ is decoded as an active DMA channel request. Writing a 0 to bit 6 assigns active high as the sense assert level. When a 1 is written to this bit, a low level on the DREQ line is decoded as an active DMA channel request.
5	**Reserved:** Must be 0.
4	**DMA Group Arbitration:** Each channel group is individually assigned either fixed or rotating arbitration priority. At reset, each group is initialized in fixed priority. Writing a 0 to bit 4 assigns fixed priority to the channel group, while writing a 1 assigns rotating priority to the group.
3	**Reserved:** Must be 0.
2	**DMA Group Enable:** Writing a 1 to this bit disables the DMA channel group, while writing a 0 to this bit enables the DMA channel group. Both channel groups are enabled following reset. Disabling Channel group 4-7 also disables Channel group 0-3, which is cascaded through Channel 4.
1:0	**Reserved:** Must be 0.

3.2.2 DCM—DMA CHANNEL MODE REGISTER

Register Location: 0Bh—Channels 0-3
0D6h—Channels 4-7
Default Value: 000000xxb
Attribute: Write Only
Size: 8 Bits

Each channel has a Mode Register associated with it. The Mode registers provide control over DMA Transfer type, transfer mode, address increment/decrement, and autoinitialization. When writing to the register, bits[1:0] determine which channel's Mode Register will be written and are not stored. Only bits[7:2] are stored in the mode register. This register is set to the default value upon reset and Master Clear. Its default value is Verify transfer, autoinitialize disable, Address increment, and Demand mode. Channel 4 defaults to cascade mode and cannot be programmed for any mode other than cascade mode.

Bit	Description
7:6	**DMA Transfer Mode:** Each DMA channel can be programmed in one of four different modes: single transfer, block transfer, demand transfer and cascade. **Bits[7:6] Transfer Mode** 00 Demand mode 01 Single mode 10 Block mode 11 Cascade mode
5	**Address Increment/Decrement Select:** Bit 5 controls address increment/decrement during multi-byte DMA transfers. When bit 5 = 0, address increment is selected. When bit 5 = 1, address decrement is selected. Address increment is the default after a PCIRST# cycle or Master Clear command.
4	**Autoinitialize Enable:** When bit 4 = 1, the DMA restores the Base Page, Address, and Word count information to their respective current registers following a terminal count (TC). When bit 4 = 0, the autoinitialize feature is disabled and the DMA does not restore the above mentioned registers. A PCIRST# or Master Clear disables autoinitialization (sets bit 4 to 0).
3:2	**DMA Transfer Type:** Verify, write and read transfer types are available. Verify transfer is the default transfer type upon PCIRST# or Master Clear. Write transfers move data from an I/O device to memory. Read transfers move data from memory to an I/O device. Verify transfers are pseudo transfers; addresses are generated as in a normal read or write transfer and the device responds to EOP etc. However, with Verify transfers, the ISA memory and I/O cycle lines are not driven. Bit combination 11 is illegal. When the channel is programmed for cascade ([7:6] = 11) the transfer type bits are irrelevant. **Bits[3:2] Transfer Type** 00 Verify transfer 01 Write transfer 10 Read Transfer 11 Illegal
1:0	**DMA Channel Select:** Bits[1:0] select the DMA Channel Mode Register that will be written by bits[7:2]. **Bits[1:0] Channel** 00 Channel 0 (4) 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)

3.2.3 DCEM—DMA CHANNEL EXTENDED MODE REGISTER

Register Location: 040Bh—Channels 0-3
04D6h—Channels 4-7
Default Value: 000000xxb
Attribute: Write Only
Size: 8 Bits

Each channel has an Extended Mode Register. The register is used to program the DMA device data size, timing mode, EOP input/output selection, and Stop register selection. When writing to the register, bits[1:0] determine which channel's Extended Mode Register will be written and are not stored. Only bits[7:2] are stored in the Extended Mode Register. Four timing modes are available: ISA-compatible, A, B, and Burst.

The default bit values for each DMA group are selected upon reset. A Master Clear or any other programming sequence will not set the default register settings. The default programmed values for DMA1 Channels 0-3 are 8-bit I/O Count by Bytes, Compatible timing, and EOP output. The default values for DMA2 Channels 4-7 are 16-bit I/O Count by Words with shifted address, Compatible timing, and EOP output. These default settings provide a rigorous ISA-compatible DMA implementation.

NOTE:
DMA1/DMA2 refer to the original PC-AT implementation which used two discrete 8237 DMA controllers. In this context, DMA1 refers to DMA Channels 0-3 and DMA2 refers to DMA Channels 4-7. The PC-AT used Channel 4 (Channel 0 of DMA2) as a cascade channel for DMA1. Consequently, Channel 4 is not used in compatible DMA controllers although the compatible DMA registers are kept to maintain compatibility with the original PC-AT. Because Channel 4 is not used, the DMA controller does not support extended registers for Channel 4.

Bit	Description
7	**Stop Register:** Bit 7 of this register selects whether or not the Stop registers associated with this channel are to be used. Normally the Stop Registers will not be used. This function was added to help support data communication or other devices that work from a ring buffer in memory. Upon reset, the bit 7 is set to 0-Stop register disabled. The detailed Stop register functional description discusses the use of the Stop registers.
6	**EOP Input/Output:** Bit 6 of the Extended Mode register selects whether the EOP signal is to be used as an output during DMA on this channel or an input. EOP will generally be used as an output, as was available on the PCAT. The input function was added to support Data Communication and other devices that would like to trigger an autoinitialize when a collision or some other event occurs. The direction of EOP is switched when DACK is changed (when a different channel wins the arbitration and is granted the bus). There may be some overlap of the ESC driving the EOP signal along with the DMA slave. However, during this overlap both devices will be driving the signal to a low level (negated). For example, assume Channel 2 is about to go inactive (DACK negated) and channel 1 is about to go active. If Channel 2 is programmed for "EOP OUT" and Channel 1 is programmed for "EOP IN", when Channel 2's DACK is negated and Channel 1's DACK is asserted, the ESC may be driving EOP to a low value on behalf of Channel 2 at the same time the device connected to Channel 1 is driving EOP in to the ESC, also at an inactive level. This overlap will only last until the ESC EOP output buffer is tristated, and will not effect the DMA operation. Upon reset, the value of bit 6 is 0 (EOP output selected).

Bit	Description
5:4	**DMA Cycle Timing Mode:** The ESC supports four DMA transfer timings: ISA-compatible, Type A, Type B, and Burst. Each timing and its corresponding code are described below. Upon reset, compatible timing is selected and the value of these bits is "00". The cycle timings noted below are for a BCLK (8.33 MHz maximum BCLK frequency). DMA cycles to ISA expansion bus memory will default to compatible timing if the channel is programmed in one of the performance timing modes (Type A, B, or Burst). 00 Compatible Timing DMA slaves on the ISA bus may run compatible DMA cycles. Bits[5:4] must be programmed to 00. Compatible timing is provided for DMA slave devices, which, due to some design limitation, cannot support one of the faster timings. Compatible timing runs at 9 BCLKs (1080 ns/single cycle) and 8 BCLKs (960 ns/cycle) during the repeated portion of a BLOCK or DEMAND mode transfers. 01 Type "A" Timing Type "A" timing is provided to allow shorter cycles to EISA memory. If ISA memory is decoded, the system automatically reverts to ISA DMA type compatible timing on a cycle-by-cycle basis. Type "A" timing runs at 7 BCLKs (840 ns/single cycle) and 6 BCLKs (720 ns/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed by system memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter, but it is expected that the DMA devices which provide the data access time or write data setup time should not require excess IOR# or IOW# command active time. Because of this, most ISA DMA devices should be able to use type "A" timing. 10 Type "B" Timing Type "B" timing is provided for 8-/16- bit ISA or EISA DMA devices which can accept faster I/O timing. Type "B" only works with EISA memory. Type "B" timing runs at 6 BCLKs (720 ns/single cycle) and 4 BCLKs (480 ns/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "B" timing requires faster DMA slave devices than compatible timing in that the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster. Some of the current ISA devices should be able to support type "B" timing, but these will probably be more recent designs using relatively fast technology. 11 Type "C" Timing (Burst) Burst timing is provided for high performance EISA DMA devices. The DMA slave device needs to monitor the EXRDY and IORC# or IOWC# signals to determine when to change the data (on writes) or sample the data (on reads). This timing will allow up to 33 MBytes per second transfer rate with a 32-bit DMA device and 32-bit memory. Note that 8- or 16-bit DMA devices are supported (through the programmable Address size) and that they use the "byte lanes" natural to their size for the data transfer. As with all bursts, the system will revert to two BCLK cycles if the memory does not support burst. When a DMA burst cycle accesses non-burst memory and the DMA cycle crosses a page boundary into burstable memory, the ESC will continue performing non-burst cycles. This will not cause a problem since the data is still transferred correctly.

Bit	Description
3:2	**Addressing Mode:** The ESC supports 8-, 16-, and 32-bit DMA device data sizes. The four data size options are programmable with bits[3:2]. Both the 8 bit I/O, "Count By Bytes" Mode and the 16-bit I/O, "Count By Words" (Address Shifted) Mode are ISA compatible. The 16-bit and 32-bit I/O, "Count By Bytes" Modes are EISA extensions. Byte assembly/disassembly is performed by the EISA Bus Controller. Each of the data transfer size modes is discussed below. 00 8-Bit I/O, "Count By Bytes" Mode In 8 bit I/O, "count by bytes" mode, the address counter can be programmed to any address. The count register is programmed with the "number of bytes minus 1" to transfer. 01 16-Bit I/O, "Count By Words" (Address Shifted) Mode In "count by words" mode (address shifted), the address counter can be programmed to any even address, but must be programmed with the address value shifted right by one bit. The Page registers are not shifted during DMA transfers. Thus, the least significant bit of the Low Page register is ignored when the address is driven out onto the bus. The Word Count register is programmed with the number of words minus 1 to be transferred. 10 32-Bit I/O, "Count By Bytes" Mode In 32-bit "count by bytes" mode, the address counter can be programmed to any byte address. For most DMA devices, however, it should only be programmed to a Dword aligned address. If the starting address is not Dword aligned then the DMA controller will do a partial Dword transfer during the first and last during the first and last transfers if necessary. The bus controller logic will do the byte/word assembly necessary to read or write any size memory device and both the DMA and bus controllers support burst for this mode. In this mode, the Address register is usually incremented or decremented by four and the byte count is usually decremented by four. The Count register should be programmed with the number of bytes to be transferred minus 1. 11 16-Bit I/O, "Count By Bytes" Mode In 16-bit "count by bytes" mode, the address counter can be programmed to any byte address. For most DMA devices, however, it should be programmed only to even addresses. If the address is programmed to an odd address, then the DMA controller will do a partial word transfer during the first and last transfer if necessary. The bus controller will do the byte/word assembly necessary to write any size memory device. In this mode, the Address register is incremented or decremented by two and the byte count is decremented by the number of bytes transferred during each bus cycle. The Word Count register is programmed with the "number of bytes minus 1" to be transferred. This mode is offered as an extension of the two ISA compatible modes discussed above. This mode should only be programmed for 16 bit ISA DMA slaves.
1:0	**DMA Channel Select:** Bits[1:0] select the particular channel that will have its DMA Channel Extend Mode Register programmed with bits[7:2]. **Bits[1:0]** — **Channel** 00 — Channel 0 (4) 01 — Channel 1 (5) 10 — Channel 2 (6) 11 — Channel 3 (7)

3.2.4 DR—DMA REQUEST REGISTER

Register Location: 09h—Channels 0-3
0D2h—Channels 4-7
Default Value: 000000xxb
Attribute: Write Only
Size: 8 Bits

Each channel has a Request bit associated with it in one of the two Request Registers. The Request register is used by software to initiate a DMA request. The DMA responds to the software request as though DREQ[x] is asserted. These requests are non-maskable and subject to prioritization by the Priority Encoder network (refer to the Channel Priority Functional Description). Each register bit is set or reset separately under software control or is cleared upon generation of a TC. The entire register is cleared upon reset or a Master Clear. It is not cleared upon a RSTDRV output. To set or reset a bit, the software loads the proper form of the data word. Bits[1:0] determine which channel Request register will be written. In order to make a software request, the channel must be in Block Mode. The Request register status for DMA1 and DMA2 is output on bits[7:4] of a Status register read to the appropriate port.

Bit	Description
7:3	**Reserved:** Must be 0.
2	**DMA Channel Service Request:** Writing a 0 to bit 2 resets the individual software DMA channel request bit. Writing a 1 to bit 2 will set the request bit. The request bit for each DMA channel is reset to 0 upon a reset or a Master Clear.
1:0	**DMA Channel Select:** Bits[1:0] select the DMA channel mode register to program with bit 2. **Bits[1:0] Channel** 00 Channel 0 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)

3.2.5 MASK REGISTER—WRITE SINGLE MASK BIT

Register Location: 0Ah—Channels 0-3
0D4h—Channels 4-7
Default Value: 000001xxb
Attribute: Write Only
Size: 1 Bit/Channel

Each DMA channel has a mask bit that can disable an incoming DMA channel service request DREQ[x] assertion. Two registers store the current mask status for DMA1 and DMA2. Setting the mask bit disables the incoming DREQ[x] for that channel. Clearing the mask bit enables the incoming DREQ[x]. A channel's mask bit is automatically set when the Current Word Count register reaches terminal count (unless the channel is programmed for autoinitialization). Each mask bit may also be set or cleared under software control. The entire register is also set by a reset or a Master Clear. Setting the entire register disables all DMA requests until a clear Mask register instruction allows them to occur. This instruction format is similar to the format used with the Request register.

Individually masking DMA Channel 4 (DMA controller 2, Channel 0) will automatically mask DMA Channels [3:0], as this Channel group is logically cascaded onto Channel 4. Setting this mask bit disables the incoming DREQ's for Channels [3:0].

Bit	Description
7:3	**Reserved:** Must be 0.
2	**DMA Channel Mask Set/Clear:** Writing a 1 to bit 2 sets the mask bit and disables the incoming DREQ for the selected channel. Writing a 0 to bit 2 clears the mask bit and enables the incoming DREQ for the elected channel.
1:0	**DMA Channel Select:** Bits[1:0] select the DMA Channel Mode Register to program with bit 2. **Bits[1:0] Channel** 00 Channel 0 (4) 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)

3.2.6 WAMB—WRITE ALL MASK BITS REGISTER

Register Location: 0Fh—Channels 0-3
0DEh—Channels 4-7
Default Value: 0Fh
Attribute: Read/Write
Size: 8 Bits

This command allows enabling and disabling of incoming DREQ assertions by writing the mask bits for each controller, DMA1 or DMA2, simultaneously rather than by individual channel as is done with the "Write Single Mask Bit" command. Two registers store the current mask status for DMA1 and DMA2. Setting the mask bit disables the incoming DREQ[x] for that channel. Clearing the mask bit enables the incoming DREQ[x]. Unlike the "Write Single Mask Bit" command, this command includes a status read to check the current mask status of the selected DMA channel group. When read, the mask register current status appears on bits[3:0]. A channel's mask bit is automatically set when the Current Word Count register reaches terminal count (unless the channel is programmed for autoinitialization). The entire register is also set by a reset or a Master Clear. Setting the entire register disables all DMA requests until a clear Mask register instruction allows them to occur.

Two important points should be taken into consideration when programming the mask registers. First, individually masking DMA Channel 4 (DMA controller 2, Channel 0) will automatically mask DMA Channels [3:0], as this channel group is logically cascaded onto Channel 4. Second, masking off DMA controller 2 with a write to port 0DEh will also mask off DREQ assertions from DMA controller 1 for the same reason: when DMA Channel 4 is masked, so are DMA Channels 0-3.

Bit	Description
7:4	**Reserved:** Must be 0.
3:0	**Channel Mask Bits:** Setting the bit(s) to a 1 disables the corresponding DREQ(s). Setting the bit(s) to a 0 enables the corresponding DREQ(s). Bits[3:0] are set to 1 upon PCIRST# or Master Clear. When read, bits[3:0] indicate the DMA channel [3:0] ([7:4]) mask status. **Bit Channel** 0 0(4) 1 1(5) 2 2(6) 3 3(7) **NOTE:** Disabling channel 4 also disables channels 0-3 due to the cascade of DMA1 through channel 4 of DMA2.

3.2.7 DS—DMA STATUS REGISTER

Register Location: 08h—Channels 0-3
0D0h—Channels 4-7
Default Value: 00h
Attribute: Read Only
Size: 8 Bits

Each DMA controller has a read-only Status register. A Status register read is used when determining which channels have reached terminal count and which channels have a pending DMA request. Bits[3:0] are set every time a TC is reached by that channel. These bits are cleared upon reset and on each Status Read. Bits[7:4] are set whenever their corresponding channel is requesting service.

Bit	Description
7:4	**Request Status:** When a valid DMA request is pending for a channel (on its DREQ signal line), the corresponding bit is set to 1. When a DMA request is not pending for a particular channel, the corresponding bit is set to 0. The source of the DREQ may be hardware, a timed-out block transfer, or a software request. Note that channel 4 does not have DREQ or DACK lines, so the response for a read of DMA2 status for channel 4 is irrelevant. **Bit Channel** 4 0 5 1(5) 6 2(6) 7 3(7)
3:0	**Terminal Count Status:** When a channel reaches terminal count (TC), its status bit is set to 1. If TC has not been reached, the status bit is set to 0. Note that channel 4 is programmed for cascade, and is not used for a DMA transfer. Therefore, the TC bit response for a status read on DMA2 for channel 4 is irrelevant. **Bit Channel** 0 0 1 1(5) 2 2(6) 3 3(7)

3.2.8 DB&CA—DMA BASE AND CURRENT ADDRESS REGISTER (8237 COMPATIBLE SEGMENT)

Register Location: 000h—DMA Channel 0
002h—DMA Channel 1
004h—DMA Channel 2
006h—DMA Channel 3
0C0h—DMA Channel 4
0C4h—DMA Channel 5
0C8h—DMA Channel 6
0CCh—DMA Channel 7
Default Value: 0000h
Attribute: Read/Write
Size: 16 Bits per channel

Each channel has a 16-bit Current Address register. This register holds the value of the 16 least significant bits of the full 32-bit address used during DMA transfers. The address is automatically incremented or decremented after each transfer and the intermediate values of the address are stored in the Current Address register during the transfer. This register is written to or read from by the microprocessor or bus master in successive 8-bit bytes. The programmer must issue the "Clear Byte Pointer Flip-Flop" command to reset the internal byte pointer and correctly align the write prior to programming the Current address register. After clearing the Byte Pointer Flip-flop, the first write to the Current Address port programs the low byte, bits[7:0], and the second write programs the high byte, bits[15:8]. This procedure applies for read cycles also. It may also be re-initialized by an autoinitialize back to its original value. autoinitialize takes place only after a TC or EOP.

Each channel has a Base Address register located at the same port address as the corresponding Current Address register. These registers store the original value of their associated Current registers. During autoinitialize these values are used to restore the Current registers to their original values. The Base registers are written simultaneously with their corresponding Current register in successive 8-bit bytes by the microprocessor. The Base registers cannot be read by any external agents.

In Scatter-Gather Mode these registers store the lowest 16-bits of the current memory address. During a Scatter-Gather transfer the DMA will load a reserve buffer into the base memory address register.

In Chaining Mode these register store the lowest 16-bits of the current memory address. The CPU will program the base register set with a reserve buffer.

Bit	Description
15:0	**Base and Current Address:** These bits represent the 16 least significant address bits used during DMA transfers. Together with the DMA Low Page register, they help form the ISA-compatible 24-bit DMA address. As an extension of the ISA compatible functionality, the DMA High Page register completes the 32-bit address needed when implementing ESC extensions such as DMA to the PCI bus slaves that can take advantage of full 32-bit addressability. Upon reset or Master Clear, the value of these bits is 0000h.

3.2.9 DB&CBW—DMA BASE AND CURRENT BYTE/WORD COUNT REGISTER (8237 COMPATIBLE SEGMENT)

Register Location: 001h—DMA Channel 0
003h—DMA Channel 1
005h—DMA Channel 2
007h—DMA Channel 3
0C2h—DMA Channel 4
0C6h—DMA Channel 5
0CAh—DMA Channel 6
0CEh—DMA Channel 7
Default Value: 0000h
Attribute: Read/Write
Size: 16 Bits per channel

Each channel has a 16-bit Current Byte/Word Count register. This register determines the lower 16 bits for the number of transfers to be performed. There is a total of 24 bits in the Byte/Word Count registers. The uppermost 8 bits are in the High Byte/Word Count register. The actual number of transfers will be one more than the number programmed in the Current Byte/Word Count register (i.e., programming a count of 100 will result in 101 transfers). The byte/word count is decremented after each transfer. The intermediate value of the byte/word count is stored in the register during the transfer. When the value in the register goes from zero to 0FFFFFFh, a TC will be generated.

Following the end of a DMA service it may also be re-initialized by an autoinitialization back to its original value. autoinitialize can occur only when a TC occurs. If it is not autoinitialized, this register will have a count of FFFFh after TC.

When the Extended Mode register is programmed for "count by word" transfers to/from a 16-bit I/O, with shifted address, the Byte/Word count will indicate the number of 16-bit words to be transferred.

When the Extended Mode register is programmed for "count by byte" transfers, the Byte/Word Count will indicate the number of bytes to be transferred. The number of bytes does not need to be a multiple of the transfer size in this case.

Each channel has a Base Byte/Word Count register located at the same port address as the corresponding Current Byte/Word Count register. These registers store the original value of their associated Current registers. During autoinitialize these values are used to restore the Current registers to their original values. The Base registers cannot be read by any external agents.

In Scatter-Gather mode these registers store the lowest 16-bits of the current Byte/Word Count. During a Scatter-Gather transfer the DMA will load a reserve buffer into the base Byte/Word Count register.

In Chaining Mode these register store the lowest 16-bits of the current Byte/Word Count. The CPU will then program the base register set with a reserve buffer.

Bit	Description
15:0	**Base and Current Byte/Word Count:** These bits represent the lower 16 byte/word count bits used when counting down a DMA transfer. Upon reset or Master Clear, the value of these bits is 0000h.

3.2.10 DMA BASE AND CURRENT HIGH BYTE/WORD COUNT REGISTER; DMA BASE HIGH BYTE/ WORD COUNT REGISTER

Register Location: 401h—DMA Channel 0
403h—DMA Channel 1
405h—DMA Channel 2
407h—DMA Channel 3
4C6h—DMA Channel 5
4CAh—DMA Channel 6
4CEh—DMA Channel 7
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits per channel

Each channel has a 8-bit Current High Byte/Word Count register. This register provides the uppermost 8 bits for the number of transfers to be performed. The byte/word count is decremented after each transfer. The intermediate value of the byte/word count is stored in the register during the transfer. When the value in the register goes from zero to FFFFh, a TC may be generated.

Following the end of a DMA service it may also be re-initialized by an autoinitialization back to its original value. autoinitialize can occur only when a TC occurs. If it is not autoinitialized, this register will have a count of FFFFh after TC.

The High Byte/Word Count register must be the last Byte/Word Count register programmed. Writing to the 8237 Compatible Byte/Word Count registers will clear the High Byte/Word Count register to 00h.

When the Extended Mode register is programmed for "count by word" transfers to/from a 16-bit I/O, with shifted address, the Byte/Word count will indicate the number of 16-bit words to be transferred.

When the Extended Mode register is programmed for "count by byte" transfers, the Byte/Word Count will indicate the number of bytes to be transferred. The number of bytes does not need to be a multiple of the transfer size in this case.

Each channel has a Base High Byte/Word Count register located at the same port address as the corresponding Current High Byte/Word Count register. These registers store the original value of their associated Current registers. During autoinitialize these values are used to restore the Current registers to their original values. Normally, the Base registers are written simultaneously with their corresponding Current register in successive 8 bit bytes by the microprocessor. However, in Chaining Mode only the Base register set is programmed and the Current register is not effected. The Base registers cannot be read by any external agents.

In Scatter-Gather mode these registers store the lowest 8 bits of the current High Byte/Word Count. During a Scatter-Gather transfer the DMA will load a reserve buffer into the base High Byte/Word Count register.

In Chaining Mode these register store the lowest 8 bits of the current High Byte/Word Count. The CPU will then program the base register set with a reserve buffer.

Bit	Description
7:0	**Base and Current High Byte/Word Count:** These bits represent the 8 high order byte/word count bits used when counting down a DMA transfer. Upon reset or Master Clear, the value of these bits is 00h.

3.2.11 DMA MEMORY LOW PAGE REGISTER; DMA MEMORY BASE LOW PAGE REGISTER

Register Location: 087h—DMA Channel 0
083h—DMA Channel 1
081h—DMA Channel 2
082h—DMA Channel 3
08Bh—DMA Channel 5
089h—DMA Channel 6
08Ah—DMA Channel 7
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits per channel

Each channel has an 8-bit Low Page register associated with it. The DMA memory Low Page register contains the eight second most-significant bits of the 32-bit address. It works in conjunction with the DMA controller's High Page register and Current Address register to define the complete (32-bit) address for the DMA channel. This 8-bit register is read or written directly by the processor or bus master. It may also be re-initialized by an autoinitialize back to its original value. autoinitialize takes place only after a TC or EOP.

Each channel has a Base Low Page Address register located at the same port address as the corresponding Current Low Page register. These registers store the original value of their associated Current Low Page registers. During autoinitialize these values are used to restore the Current Low Page registers to their original values. The 8-bit Base Low Page registers are written simultaneously with their corresponding Current Low Page register by the microprocessor. The Base Low Page registers cannot be read by any external agents.

During Scatter-Gather these registers store the 8 bits from the third byte of the current memory address. During a Scatter-Gather transfer the DMA will load a reserve buffer into the base memory address register.

In Chaining Mode these register store the 8 bits from the third byte of the current memory address. The CPU will program the base register set with a reserve buffer.

Bit	Description
7:0	**DMA LOW PAGE AND BASE LOW PAGE:** These bits represent the eight second most-significant address bits when forming the full 32-bit address for a DMA transfer. Upon reset or Master Clear, the value of these bits is 00h.

3.2.12 DMAP—DMA PAGE REGISTER

Register Location: 080h, 84h, 85h, 86h, 88h, 8Ch, 8Dh, 8Eh
Default Value: xxh
Attribute: Read/Write
Size: 8 Bits

These registers have no effect on the DMA operation. These registers provide extra storage space in the I/O space for DMA routines.

Bit	Description
7:0	**DMA PAGE:** These bit have no effect on the DMA operation. These bits only provide storage space in the I/O map.

3.2.13 DMALPR—DMA LOW PAGE REFRESH REGISTER

Register Location: 08Fh
Default Value: xxh
Attribute: Read/Write
Size: 8 Bits

The contents of this register are driven on the address byte 2 (LA[23:16]#) during Refresh cycles.

Bit	Description
7:0	**DMA LOW PAGE REFRESH:** The contents of the bits are driven on to the address bus(LA[23:16]) during refresh.

3.2.14 DMAMHPG—DMA MEMORY HIGH PAGE REGISTER; DMA MEMORY BASE HIGH PAGE REGISTER

Register Location: 0487h—DMA Channel 0
0483h—DMA Channel 1
0481h—DMA Channel 2
0482h—DMA Channel 3
048Bh—DMA Channel 5
0489h—DMA Channel 6
048Ah—DMA Channel 7
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits per channel

Each channel has an 8-bit High Page register. The DMA memory High Page register contains the eight most-significant bits of the 32-bit address. It works in conjunction with the DMA controller's Low Page register and Current Address register to define the complete (32-bit) address for the DMA channels and corresponds to the "Current Address" register for each channel. This 8-bit register is read or written directly by the processor or bus master. It may also be re-initialized by an autoinitialize back to its original value. Autoinitialize takes place only after a TC or EOP.

This register is reset to 00h during the programming of both the low page register and the Current Address register. Thus, if this register is not programmed after the other address and Low Page registers are programmed, then its value will be zero. In this case, the DMA channel will operate the same as an 82C37 (from an addressing standpoint). This is the address compatibility mode.

If the high 8 bits of the address are programmed after the other addresses, then the channel will modify its operation to increment (or decrement) the entire 32-bit address. This is unlike the 82C37 "Page" register in the original PCs which could only increment to a 64K boundary (for 8-bit channels) or 128K (for 16-bit channels). This is extended address mode. In this mode, the ISA bus controller will generate the signals MEMR# and MEMW# only for addresses below 16 MBytes.

Each channel has a Base High Page Address register located at the same port address as the corresponding Current High Page Address register. These registers store the original value of their associated Current registers. During autoinitialize these values are used to restore the Current registers to their original values. The 8 bit Base High Page registers are written simultaneously with their corresponding Current register by the microprocessor. The Base registers cannot be read by any external agents.

During Scatter-Gather these registers store the 8 bits from the highest byte of the current memory address. During a Scatter-Gather transfer the DMA will load a reserve buffer into the base memory address register.

In Chaining Mode these register store the 8 bits from the highest byte of the current memory address. The CPU will program the base register set with a reserve buffer.

Bit	Description
7:0	**DMA High Page and Base High Page:** These bits represent the eight most-significant address bits when forming the full 32-bit address for a DMA transfer. Upon reset or Master Clear, the value of these bits is 00h.

3.2.15 DMAHPGR—DMA HIGH PAGE REGISTER REFRESH

Register Location: 048Fh
Default Value: xxh
Attribute: Read/Write
Size: 8 Bits per channel

The contents of this register are driven on the address byte 3 (LA[31:24]#) during Refresh cycles.

Bit	Description
7:0	**DMA High Page Refresh:** The contents of the bits are driven on to the address bus (LA[31:24]) during refresh.

3.2.16 STOP REGISTERS

Register Location: 04E0h—CH0 Stop Reg Bits[7:2]
04E1h—CH0 Stop Reg Bits[15:8]
04E2h—CH0 Stop Reg Bits[23:16]
04E4h—CH1 Stop Reg Bits[7:2]
04E5h—CH1 Stop Reg Bits[15:8]
04E6h—CH1 Stop Reg Bits[23:16]
04E8h—CH2 Stop Reg Bits[7:2]
04E9h—CH2 Stop Reg Bits[15:8]
04EAh—CH2 Stop Reg Bits[23:16]
04ECh—CH3 Stop Reg Bits[7:2]
04EDh—CH3 Stop Reg Bits[15:8]
04EEh—CH3 Stop Reg Bits[23:16]
04F4h—CH5 Stop Reg Bits[7:2]
04F5h—CH5 Stop Reg Bits[15:8]
04F6h—CH5 Stop Reg Bits[23:16]
04F8h—CH6 Stop Reg Bits[7:2]
04F9h—CH6 Stop Reg Bits[15:8]
04FAh—CH6 Stop Reg Bits[23:16]
04FCh—CH7 Stop Reg Bits[7:2]
04FDh—CH7 Stop Reg Bits[15:8]
04FEh—CH7 Stop Reg Bits[23:16]
Default Value: See Below
Attribute: Read/Write
Size: See Below

The Stop registers are used to support a common data communication structure, the ring buffer. The ring buffer data structure and Stop Register operation are described in Section 6.7.4. The Stop registers, in conjunction with a channel's Base and Current address and byte count registers, are used to define a fixed portion of memory for use by the ring buffer data structure. Following a reset, these registers are not reset to 0.

Bit	Description
23:2	**Upper, Mid, Lower Stop Bits:** These 22 bits provide the Stop Address. If the Stop function is enabled then the channel will Stop whenever its Memory Address matches the Stop Address. Bits[23:16] are the upper stop bits. Bits[15:8] are the mid stop bits and bits[7:2] are the lower stop bits. Bits[1:0] are not used and are don't cares.

3.2.17 CHAIN—CHAINING MODE REGISTER

Register Location: 040Ah—Channels 0-3
04D4h—Channels 4-7
Default Value: 000000xxb
Attribute: Write Only
Size: 8 Bits

Each channel has a Chaining Mode register. The Chaining Mode register enables or disables DMA buffer chaining and indicates when the DMA Base registers are being programmed. When writing to the register, bits[1:0] determine which channel's Chaining Mode register to program. The chaining status and interrupt status for all channels can be determined by reading the Chaining Mode Status, Channel Interrupt Status, and Chain Buffer Expiration Control registers. The Chaining Mode register is reset to zero upon reset, access (read or write) of a channel's Mode register or Extended Mode register, or a Master Clear. The values upon reset are disable chaining mode and generate IRQ13.

Bit	Description
7:5	**Reserved:** Must be 0.
4	**Buffer Expired Signal:** After one of the two buffers in the DMA expires then the DMA will inform the CPU that the next buffer should be loaded into the base register set. This bit determines whether IRQ13 or EOP should be used to inform the CPU that the buffer is complete; 1 = generate TC, 0 = Generate IRQ13; 1 = Programming complete, 0 = Don't start chaining.
3	**Base Register Programming:** After the reserve buffer's address and word count are written to the base register set, this bit should be set to 1 to inform the DMA that the second buffer is ready for transfer.
2	**Buffer Chaining Mode:** Bit 2 enables the chaining mode logic. If the bit is set to 1 after the initial DMA address and word count are programmed, then the Base address and word count are available for programming the next buffer in the chain. 1 = Enable chaining, 0 = Disable chaining.
1:0	**DMA Channel Select:** Bits[1:0] select the DMA channel mode register to program with bits[4:2]. **Bits[1:0] Channel** 00 0 or 4 01 1 or 5 10 2 or 6 11 3 or 7

3.2.18 CHAINSTA—CHAINING MODE STATUS REGISTER

Register Location: 04D4h
Default Value: 00h
Attribute: Read Only
Size: 8 Bits

This register is read only and is used to determine if chaining mode for a particular channel is enabled or disabled. A 1 read in this register indicates that the channel's chaining mode is enabled. A 0 indicates that the chaining mode is disabled. All Chaining mode bits are disabled after a reset with reset. After the DMA is used in Chaining mode the CPU will need to clear the Chaining mode enable bit if non-Chaining mode is desired.

Bit	Description
7:5, 3:0	**Chaining Mode Status:** If this bit is set to 1 then this channel has chaining enabled by writing 1 to bit 2 of the Chaining Mode Register. This bit can be reset to 0 by either writing a 0 to bit 2 of the Chaining Mode Register or reset being asserted or by a Master Clear Command.
4	**Reserved**

3.2.19 CHINTST—CHANNEL INTERRUPT STATUS REGISTER

Register Location: 040Ah
Default Value: 00h
Attribute: Read Only
Size: 8 Bits

Channel Interrupt Status is a read only register and is used to indicate the source (channel) of a DMA chaining interrupt on IRQ13. The DMA controller asserts IRQ13 after reaching terminal count, with chaining mode enabled. It does not assert IRQ13 during the initial programming sequence that loads the Base registers. After a reset, a read of this register will produce 00h.

Bit	Description
7:5, 3:0	**Chaining Interrupt Status:** When a channel interrupt status read returns a 0, bits[7:5,3:0] indicates that channel did not assert IRQ13. When a channel interrupt status read returns a 1, then that channel asserted IRQ13 after reaching a Terminal Count.
4	**Reserved**

3.2.20 CHAINBEC—CHAIN BUFFER EXPIRATION CONTROL REGISTER

Register Location: 040Ch
Default Value: 00h
Attribute: Read Only
Size: 8 Bits

This register is read only and reflects the outcome of the expiration of a chain buffer. A Chain Buffer Expiration Control register bit with 0 indicates the DMA controller asserts IRQ13 when the DMA controller reaches terminal count. A 1 indicates the DMA controller asserts TC when the DMA controller reaches terminal count. This bit is programmed in bit 4 of the Chaining Mode register.

Bit	Description
7:5, 3:0	**Chaining Buffer Expired:** When a chain buffer expiration control read returns 0, bit[7:5,3:0] indicates that Channel [7:5,3:0] will assert IRQ13 when the DMA channel reaches terminal count. When a chain buffer expiration control read returns 1, bit[7:5,3:0] indicates that Channel [7:5,3:0] will assert TC when the DMA controller reaches terminal count. This bit will reset to 0 following a reset.
4	**Reserved**

3.2.21 SCATGA—SCATTER-GATHER COMMAND REGISTER

Register Location: 0410h—Channels 0
0411h—Channels 1
0412h—Channels 2
0413h—Channels 3
0415h—Channels 5
0416h—Channels 6
0417h—Channels 7
Default Value: 00xxxx00b
Attribute: Write Only, Relocateable
Size: 8 Bits

The Scatter-Gather command register controls operation of the Descriptor Table aspect of S-G transfers. The S-G command register is write only. The current S-G transfer status can be read in the S-G channel's corresponding S-G Status register. The S-G command register can initiate a S-G transfer, and stop a transfer.

Scatter-Gather commands are issued with command codes. Bits[1:0] are used to implement the code mechanism. The S-G codes are described in the table below. Bit 7 is used to control the IRQ13/EOP assertion that follows a terminal count. Bit 6 controls the effect of bit 7. Common Scatter-Gather command writes are listed in Table 2.

Table 2. Scatter Gather Command Bits

Command	Bits	
	7654	3210
No S-G operation (S-G NOOP)	0000	0000b
Start S-G	xx00	0001b
Stop S-G	xx00	0010b
Issue IRQ13 on Terminal Count	0100	00xxb
Issue EOP on Terminal Count	1100	00xxb

Note that the "x" don't care states in Table 2 do not preclude programming those bits during the command write. For instance, for any S-G command code on bits[1:0], an optional selection of IRQ13 or EOP can take place if bit 7 is set to 1 and the appropriate choice is made for bit 6. All 0's in the command byte indicate an S-G NOOP: no S-G command is issued, and EOP/IRQ13 modification is disabled. Note that an EOP/IRQ13 modification can be made while disabling the S-G command bits (bits[1:0] = 00b); conversely, an S-G command may be issued while EOP/IRQ13 modification is disabled (bit 6 = 0b). After a reset, or Master Clear, IRQ13 is disabled and EOP is enabled.

The Start command assumes the Base and Current registers are both empty and will request a prefetch automatically. It also sets the status register to S-G Active, Base Empty, Current Empty, not Terminated, and Next Null Indicator to 0. The EOP/IRQ13 bit will still reflect the last value programmed.

Bit	Description
7	**EOP/IRQ13 Selection:** Bit 7 is used to select whether EOP or IRQ will be asserted at termination caused by the last buffer expiring. The last buffer can be either the last buffer in the list or the last buffer loaded in the DMA while it is suspended. If this bit is set to 1 then EOP will be asserted whenever the last buffer is completed. If this bit is set to 0 then IRQ13 will be asserted whenever the last buffer is completed. EOP can be used to alert an expansion bus I/O device that a scatter-gather termination condition was reached; the I/O device in turn can assert its own interrupt request line, and invoke a dedicated interrupt handling routine. IRQ13 should be used whenever the CPU needs to be notified directly. Following reset, or Master Clear, the value stored for this bit is 0, and IRQ13 is selected. Bit 6 must be set to a 1 to enable this bit during an S-G Command register write. When bit 6 is a 0 during the write, bit 7 will not have any effect on the current EOP/IRQ13 selection.
6	**Enable IRQ13/EOP Programming:** Enabling IRQ13/EOP programming allows initialization or modification of the S-G termination handling bits. If bit 5 is reset to 0, bit 7 will not have any effect on the state of IRQ13 or EOP assertion. When bit 5 is set to a 1, bit 7 determines the termination handling following a terminal count.
5:2	**Reserved**
1:0	**S-G Command Code** **Bits[1:0] Function** 00 No S-G command operation is performed. Bits[7:5] may still be used to program EOP/IRQ13 selection. 01 The Start command initiates the scatter-gather process. Immediately after the Start command is issued a request is issued to fetch the initial buffer to fill the Base Register set in preparation for performing a transfer. The Buffer Prefetch request has the same priority with respect to other channels as the DREQ it is associated with. Within the channel, DREQ is higher in priority than a prefetch request. 10 The Stop command halts a Scatter-Gather transfer immediately. When a Stop command is given, the Terminate bit in the S-G Status register and the DMA channel mask bit are both set. 11 Reserved The S-G Status register contains information on the S-G transfer status. This register maintains dynamic status information on S-G Transfer Activity, the Current and Base Buffer state, S-G Transfer Termination, and the End of the List indicator.

3.2.22 SCAGAST—SCATTER-GATHER STATUS REGISTER

Register Location: Channels 0
0419h—Channels 1
041Ah—Channels 2
041Bh—Channels 3
041Dh—Channels 5
041Eh—Channels 6
041Fh—Channels 7
Default Value: 08h
Attribute: Read Only, Relocatable
Size: 8 Bits

The Scatter-Gather Status Register provides Scatter-Gather process status information to the CPU or Master. An active bit is set to 1 after the S-G Start command is issued. The active bit will be 0 before the initial start command, following a terminal count, and after an S-G Stop command is issued. The Current Buffer and Base Buffer State Bits indicate whether the corresponding register has a buffer loaded. It is possible for the Base Buffer State to be set while the Current Buffer State is cleared. When the Current Buffer transfer is complete, the Base Buffer will not be moved into the Current Buffer until the start of the next data transfer. Thus, the Current Buffer State is empty (cleared), while the Base Buffer State is full (set). The Terminate bit is set active after a Stop command, after TC for the last buffer in the list and both Base and Current buffers have expired. The EOP and IRQ13 Bits indicate which end of process indicator will be used to alert the system of an S-G process termination. The EOL status bit is set if DMA controller has loaded the last buffer of the Link List.

Bit	Description
7	**Next Link Null Indicator:** If the Next SGD fetched from memory during a fetch operation has the EOL value (1), the current value of the Next Link register is not overwritten. Instead, bit 7 of the channel's S-G Status register, the Next Link Null indicator, is set to a 1. If the fetch returns a EOL value not equal to (1), this bit is reset to 0. This status bit is written after every fetch operation. Following reset, or Master Clear, this bit is reset to 0. This bit is also cleared by an S-G Start Command Write.
6	**Reserved**
5	**IRQ13 or EOP on Last Buffer:** When the IRQ13/EOP status bit is 1, EOP was either defaulted to at reset or selected through the S-G Command register as the S-G process termination indicator. EOP will be issued to alert the system when a terminal count occurs or following the Stop Command. When this bit is returned as a 0, an IRQ13 will be issued to alert the CPU of this same status.
4	**Reserved**
3	**S-G Base Buffer State:** When the Base Buffer status bit contains a 0, the Base Buffer is empty. When the Base Buffer Status bit is set to 1, the Base buffer has a buffer link loaded. Note that the Base Buffer State may be set while the Current buffer state is cleared. This condition occurs when the Current Buffer expires following a transfer; the Base Buffer will not be moved into the Current Register until the start of the next DMA transfer.
2	**S-G Current Buffer State:** When the Current Buffer status bit contains a 0, the Current Buffer is empty. When the Current Buffer status bit is set to 1, the Current Buffer has a buffer link loaded and is considered full. Following reset, bit 2 is reset to 0.
1	**Reserved**

Bit	Description
0	**S-G Active:** The Scatter-Gather Active bit indicates the current S-G transfer status. Bit 0 will be a 1 after a S-G Start Command is issued. Bit 0 will be a 0 before the Start command is issued. Bit 0 will be a 0 after terminal count on the last buffer on the channel is reached. Bit 0 will also be a 0 after a S-G Stop command has been issued. Following reset, or Master Clear, this bit is reset to 0.

3.2.23 SCAGAD—SCATTER-GATHER DESCRIPTOR TABLE POINTER REGISTER

Register Location: 0420h–0423h—Channels 0
0424h–0424h—Channels 1
0428h–042Bh—Channels 2
042Ch–042Ch—Channels 3
0434h–0437h—Channels 5
0438h–043Bh—Channels 6
043Ch–043Fh—Channels 7
Default Value: See below
Attribute: Read/Write, Relocateable
Size: 32 Bits

The SGD Table Pointer register contains the 32 bit pointer to the first SGD entry in the SGD table in memory. Before the start of a S-G transfer, this register should have been programmed to point to the first SGD in the SGD table. Following a "Start" command, it initiates reading the first SGD entry by pointing to the first SGD entry to be fetched from the memory. Subsequently, at the end of the each buffer block transfer, the contents of the SGD table pointer registers are incremented by 8 until the end of the SGD table is reached.

When programmed by the CPU, the SGD Table Pointer Registers can be programmed with a single 32-bit PCI write. Note that the PCEB and EISA Bus Controller will split the 32-bit write into four 8-bit writes.

Following a prefetch to the address pointed to by the channel's SGD table pointer register, the new Memory Address is loaded into the Base Address register, the new Byte Count is loaded into the Base Byte Count register, and the newly fetched Next SGD replaces the current Next SGD value.

The end of the SGD table is indicated by a End of Table field having a MSB equal to 1. When this value is read during a SGD fetch, the current SGD value is not replaced. Instead, bit 7 of the channel's status register is set to a 1 when the EOL is read from memory.

Bit	Description
31:0	**SGD Table Pointer:** The SGD table pointer register contains a 32-bit pointer to the main memory location where the software maintains the Scatter Gather Descriptors for the linked-list buffers. These bits are translated into A[31:0] signals for accessing memory on the PCI.

3.2.24 CBPFF—CLEAR BYTE POINTER FLIP FLOP REGISTER

Register Location: 00Ch—Channels 0-3
0D8h—Channels 4-7
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

This command is executed prior to writing or reading new address or word count information to the DMA. This initializes the flip-flop to a known state so that subsequent accesses to register contents by the microprocessor will address upper and lower bytes in the correct sequence.

The Clear Byte Pointer command clears the internal latch used to address the upper or lower byte of the 16-bit address and Word Count registers. The latch is also cleared at power on by reset and by the Master Clear command. The Host CPU may read or write a 16-bit DMA controller register by performing two consecutive accesses to the I/O port. The Clear Byte Pointer command precedes the first access. The first I/O write to a register port loads the least significant byte, and the second access automatically accesses the most significant byte.

When the Host CPU is reading or writing DMA registers, two Byte Pointer Flip-Flops are used; one for Channels 0-3 and one for Channels 4-7. Both of these act independently. There are separate software commands for clearing each of them (0Ch for Channels 0-3, 0D8h for Channels 4-7).

Bit	Description
7:0	**Clear Byte Pointer FF:** No specific pattern. Command enabled with a write to the I/O port address.

3.2.25 DMC—DMA MASTER CLEAR REGISTER

Register Location: 00Dh—Channels 0-3
0DAh—Channels 4-7
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

This software instruction has the same effect as the hardware Reset. The Command, Status, Request, and Internal First/Last Flip-Flop registers are cleared and the Mask register is set. The DMA controller will enter the idle cycle.

There are two independent Master Clear Commands, 0Dh which acts on Channels 0-3, and 0DAh which acts on Channels 4-7.

Bit	Description
7:0	**Master Clear:** No specific pattern. Command enabled with a write to the I/O port address.

3.2.26 DCM—DMA CLEAR MASK REGISTER

Register Location: 00Eh—Channels 0-3
0DCh—Channels 4-7
Default Value: xxh
Attribute: Write Only
Size: n/a

Software Command This command clears the mask bits of all four channels, enabling them to accept DMA requests. I/O port 0Eh is used for Channels 0-3 and I/O port 0DCh is used for Channels 4-7.

Bit	Description
7:0	**Clear Mask:** No specific pattern. Command enabled with a write to the I/O port address.

3.3 Timer Unit Registers

The ESC contains five counters that are equivalent to those found in the 82C54 Programmable Interval Timer. The Timer registers control these counters and can be accessed from the EISA Bus via I/O space. This section describes the counter/timer registers on the ESC. The counter/timer operations are further described in Section 8.0, Interval Timer

3.3.1 TCW—TIMER CONTROL WORD REGISTER

Register Location: 043h—Timer 1
04Bh—Timer 2
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

The Timer Control Word specifies the counter selection, the operating mode, the counter byte programming order and size of the COUNT value, and whether it counts down in a 16-bit or binary-coded decimal (BCD) format. After writing the control word, a new count may be written at any time. The new value will take effect according to the programmed mode.

There are six programmable counting modes. Typically, the ESC Timer Unit Counters 0 and 2 are programmed for Mode 3, the Square Wave Mode, while Counter 1 is programmed in Mode 2, the Rate Generator Mode.

Two special commands are selected through the Control Word Register. The Counter Latch Command is selected when bits[5:4] are both 0. The Read-Back Command is selected when bits[7:6] are both 1. When either of these two commands are selected with the Control Word Register, the meaning of the other bits in the register changes. Both of these special commands, and the respective changes they make to the bit definitions in this register, are covered in detail under separate register descriptions later in this section.

Bits 4 and 5 are also used to select the count register programming mode. The programming process is simple:

1. Write a control word.
2. Write an initial count for each counter.
3. Load the LSB, MSB, or LSB then MSB.

The read/write selection chosen with the control word dictates the programming sequence that must follow when initializing the specified counter.

If a counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same counter. Otherwise, the counter will be loaded with an incorrect count. The count must always be completely loaded with both bytes.

Bits 6 and 7 are also used to select the counter for the control word you are writing.

Following reset, the control words for each register are undefined. You must program each timer to bring it into a known state. However, each counter OUT signal is reset to 0 following reset. The SPKR output, interrupt controller input IRQ0 (internal), bit 5 of port 061h, and the internally generated Refresh request are each reset to 0 following reset.

Bit	Description
7:6	**Counter Select:** The Counter Selection bits select the counter the control word acts upon as shown below. The Read Back Command is selected when bits[7:6] are both 1. **Bit[7:6] Function** 00 Counter 0 select 01 Counter 1 select 10 Counter 2 select 11 Read Back Command (see Section 3.3.2)
5:4	**Read/Write Select:** Bits[5:4] are the read/write control bits. The Counter Latch Command is selected when bits[5:4] are both 0. The read/write options include read/write least significant byte, read/write most significant byte, or read/write the LSB and then the MSB. The actual counter programming is done through the counter I/O port (040h, 041h, and 042h for counters 0, 1, and 2, respectively). **Bit[5:4] Function** 00 Counter Latch Command (see Section 3.3.3) 01 R/W Least Significant Byte (LSB) 10 R/W Most Significant Byte (MSB) 11 R/W LSB then MSB
3:1	**Counter Mode Selection:** Bits[3:1] select one of six possible modes of operation for the counter as shown below. Note that for the fail safe timer (timer 2, counter 0), modes 1, 2, 3, 4, and 5 are reserved. **Bit[3:1] Mode Function** 000 0 Out signal on end of count (=0) 001 1 Hardware retriggerable one-shot (Reserved for timer 2, counter 0.) X10 2 Rate generator (divide by n counter) (Reserved for timer 2, counter 0.) X11 3 Square wave output (Reserved for timer 2, counter 0.) 100 4 Software triggered strobe (Reserved for timer 2, counter 0.) 101 5 Hardware triggered strobe (Reserved for timer 2, counter 0.)
0	**Binary/BCD Countdown Select:** When bit 0 = 0, a binary countdown is used. The largest possible binary count is 2^{16}. When bit 0 = 1, a binary coded decimal (BCD) count is used. The largest BCD count allowed is 104.

3.3.2 TIMER READ BACK COMMAND REGISTER

Register Location: 043h—Timer 1
04Bh—Timer 2
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

The Read-Back command is used to determine the count value, programmed mode, and current states of the OUT pin and Null Count flag of the selected counter or counters. The Read-Back command is written to the Control Word register, which latches the current states of the above mentioned variables. The value of the counter and its status may then be read by I/O access to the counter address.

Status and/or count may be latched on one, two, or all three of the counters by selecting the counter during the write. The Count latched will stay latched until read, regardless of further latch commands. The count must be read before newer latch commands latch a new count. The Status latched by the read-back command will also remain latched until after a read to the counter's I/O port. To reiterate, the Status and Count are unlatched only after a counter read of the Status register, the Count register, or the Status and Count register in succession.

Both count and status of the selected counter(s) may be latched simultaneously by setting both the COUNT# and STATUS# bits [5:4] = 00b. This is functionally the same as issuing two consecutive, separate read-back commands. As stated above, if multiple count and/or status read-back commands are issued to the same counter(s) without any intervening reads, all but the first are ignored.

If both count and status of a counter are latched, the first read operation from that counter will return the latched status, regardless of which was latched first. The next one or two reads (depending on whether the counter is programmed for one or two byte counts) return the latched count. Subsequent reads return an unlatched count.

A register description of the Status Byte read follows later in this section. Note that bit definitions for a write to this port changed when the read-back command was selected, when compared to a normal control word write to this same port.

Bit	Description
7:6	**Read Back Command:** When bits[7:6] are both 1, the read-back command is selected during a write to the control word. The normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the read-back command is selected. Following the read-back command, I/O reads from the selected counter's I/O addresses produce the current latch status, the current latched count, or both if bits 4 and 5 are both 0.
5	**Latch Status of Selected Counters:** When bit 5 is a 1, the Current Count value of the selected counters will be latched. When bit 4 is a 0, the Status will not be latched.
4	**Latch Count of Selected Counters:** When bit 4 is a 1, the Status of the selected counters will be latched. When bit 4 is a 0, the Status will not be latched. The Status byte format is described in the next register description.
3	**Counter 2:** Counter 2 is selected for the latch command selected with bits 4 and 5 if bit 3 is a 1. If bit 3 is a 0, Status and/or Count will not be latched.
2	**Counter 1:** Counter 1 is selected for the latch command selected with bits 4 and 5 if bit 2 is a 1. If bit 2 is a 0, Status and/or Count will not be latched.
1	**Counter 0:** Counter 0 is selected for the latch command selected with bits 4 and 5 if bit 1 is a 1. If bit 1 is a 0, Status and/or Count will not be latched.
0	**Reserved:** Must be 0.

3.3.3 COUNTER LATCH COMMAND REGISTER

Register Location: 043h—Timer 1
04Bh—Timer 2
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

The Counter Latch command latches the current count value at the time the command is received. This command is used to insure that the count read from the counter is accurate (particularly when reading a two-byte count). The count value is then read from each counter's Count register. One, two or all three counters may be latched with one counter latch command.

If a Counter is latched once and then, some time later, latched again before the count is read, the second Counter Latch Command is ignored. The count read will be the count at the time the first Counter Latch Command was issued.

The count must be read according to the programmed format. Specifically, if the Counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other; read, write, or programming operations for other Counters may be inserted between them.

One precaution is worth noting. If a Counter is programmed to read/write two-byte counts, a program must not transfer control between reading the first and second byte to another routine which also reads from that same Counter. Otherwise, an incorrect count will be read. Finish reading the latched two-byte count before transferring control to another routine.

Note that bit definitions for a write to this port have changed when the read-back command was selected, when compared to a normal control word write to this same port.

Bit	Description
7:6	**Counter Selection:** Bits 6 and 7 are used to select the counter for latching. **Bit[7:6] Function** 00 Latch counter 0 select 01 Latch counter 1 select 10 Latch counter 2 select 11 Read Back Command select
5:4	**Specifies Counter Latch Command:** When bits[5:4] are both 0, the Counter Latch command is selected during a write to the control word. The normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the Counter Latch command is selected. Following the Counter Latch command, I/O reads from the selected counter's I/O addresses produce the current latched count.
3:0	**Reserved:** Must be 0.

3.3.4 TMSTAT—TIMER STATUS BYTE FORMAT REGISTER

Register Location: 040h—Timer 1, Counter 0
041h—Timer 1, Counter 1
042h—Timer 1, Counter 2
048h—Timer 2, Counter 0
04Ah—Timer 2, Counter 2
Default Value: 0xxxxxxxb
Attribute: Read Only
Size: 8 Bits per counter

Each Counter's Status Byte may be read following an Timer Read-Back Command. The Read-Back command is programmed through the counter control register. If "Latch Status" is chosen as a Read-Back option for a given counter, the next read from the counter's I/O port address returns the Status byte.

The Status byte returns the countdown type, either BCD or binary; the Counter Operational Mode; the Read/Write Selection status; the Null count, also referred to as the Count Register Status; and the current State of the counter OUT pin.

Bit	Description
7	**Counter OUT Pin State:** When this bit is a 1, the OUT pin of the counter is also a 1. When this bit is a 0, the OUT pin of the counter is also a 0.
6	**Count Register Status:** Also referred to as Null Count, indicates when the last count written to the Count Register (CR) has been loaded into the counting element (CE). The exact time this happens depends on the counter Mode and is described in the Mode definitions, but until the count is loaded into the counting element (CE), it can't be read from the counter. If the count is latched or read before the load time, the count value returned will not reflect the new count written to the register. When bit 6 is a 0, the count has been transferred from CR to CE and is available for reading. When bit 6 is a 1, the Null count condition exists. The count has not been transferred from CR to CE and is not yet available for reading.
5:4	**Read/Write Status:** Bits[5:4] reflect the read/write selection made through bits[5:4] of the control register. The binary codes returned during the status read match the codes used to program the counter read/write selection.
3:1	**Mode Selection Status:** Bits[3:1] return the counter mode programming. The binary code returned matches the code used to program the counter mode, as listed under the bit function above.
0	**Countdown Type Status:** Bit 0 reflects the current countdown type, either 0 for binary countdown or a 1 for binary coded decimal (BCD) countdown.

3.3.5 CAPS—COUNTER ACCESS PORTS

Register Location: 040h—Timer 1, Counter 0
041h—Timer 1, Counter 1
042h—Timer 1, Counter 2
048h—Timer 2, Counter 0
04Ah—Timer 2, Counter 2
Default Value: xxh
Attribute: Read/Write
Size: 8 Bits per counter

Each of these I/O ports is used for writing count values to the count registers; reading the current count value from the counter by either an I/O read, after a counter-latch command, or after a read-back command; and reading the Status byte following a read-back command.

Bit	Description
7:0	**Counter Access:** Each counter I/O port address is used to program the 16 bit count register. The order of programming, either LSB only, MSB only, or LSB then MSB, is defined with the Counter Control register at I/O port address 043h. The counter I/O port is also used to read the current count from the count register, and return the status of the counter programming following a read-back command.

3.4 Interrupt Controller Registers

The ESC contains an EISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The interrupt registers control the operation of the interrupt controller and can be accessed from the EISA Bus via I/O space. This section describes the Interrupt registers. The operation of the Interrupt Controller is described in Chapter 9.0.

3.4.1 ICW1—INITIALIZATION COMMAND WORD 1

Register Location: 020h—INT CNTRL-1
0A0h—INT CNTRL-2
Default Value: xxh
Attribute: Write Only
Size: 8 Bits per controller

A write to Initialization Command Word One starts the interrupt controller initialization sequence. Addresses 020h and 0A0h are referred to as the base addresses of CNTRL-1 and CNTRL-2 respectively.

An I/O write to the CNTRL-1 or CNTRL-2 base address with bit 4 equal to 1 is interpreted as ICW1. For ESC-based EISA systems, three I/O writes to "base address + 1" must follow the ICW1. The first write to "base address + 1" performs ICW2, the second write performs ICW3, and the third write performs ICW4.

ICW1 starts the initialization sequence during which the following automatically occur:

a. The edge sense circuit is reset, which means that following initialization, an interrupt request (IRQ) input must make a low-to-high transition to generate an interrupt.

b. The Interrupt Mask register is cleared.

c. IRQ7 input is assigned priority 7.

d. The slave mode address is set to 7.

e. Special Mask Mode is cleared and Status Read is set to IRR.

f. If IC4 was set to 0, then all functions selected by ICW4 are set to zero. However, ICW4 must be programmed in the ESC implementation of this interrupt controller, and IC4 must be set to a 1.

ICW1 has three significant functions within the ESC interrupt controller configuration. ICW4 is needed, so bit 0 must be programmed to a 1. There are two interrupt controllers in the system, so bit 1, SNGL, must be programmed to a 0 on both CNTRL-1 and CNTRL-2, to indicate a cascade configuration. Bit 4 must be a 1 when programming ICW1. OCW2 and OCW3 are also addressed at the same port as ICW1. This bit indicates that ICW1, and not OCW2 or OCW3, will be programmed during the write to this port.

Bit 2, ADI, and bits[7:5], A7-A5, are specific to an MSC-85 implementation. These bits are not used by the ESC interrupt controllers. Bits[7:5,2] should each be initialized to 0.

Bit	Description
7:5	**Reserved:** A7-A5 are MCS-85 implementation specific bits. They are not needed by the ESC. These bits should be 000b when programming the ESC.
4	**ICW/OCW Select:** Bit 4 must be a 1 to select ICW1. After the fixed initialization sequence to ICW1, ICW2, ICW3, and ICW4, the controller base address is used to write to OCW2 and OCW3. Bit 4 is a 0 on writes to these registers. A 1 on this bit at any time will force the interrupt controller to interpret the write as an ICW1. The controller will then expect to see ICW2, ICW3, and ICW4.
3	**Reserved:** This bit is not used in the ESC.
2	**Reserved:** ADI ignored for the ESC.
1	**SNGL:** This bit must be programmed to a 0 to indicate that two interrupt controllers are operating in cascade mode on the ESC.
0	**IC4:** This bit must be set to a 1. IC4 indicates that ICW4 needs to be programmed. The ESC requires that ICW4 be programmed to indicate that the controllers are operating in an 80x86 type system.

3.4.2 ICW2—INITIALIZATION COMMAND WORD 2

Register Location: 021h—INT CNTRL-1
0A1h—INT CNTRL-2I
Default Value: xxh
Attribute: Write Only
Size: 8 Bits per controller

ICW2 is used to initialize the interrupt controller with the five most significant bits of the interrupt vector address. The value programmed for bits[7:3] is used by the CPU to define the base address in the interrupt vector table for the interrupt routines associated with each IRQ on the controller. Typical ISA ICW2 values are 04h for CNTRL-1 and 70h for CNTRL-2. Section 9.8.1 of the Interrupt Unit Functional Description contains a table detailing the interrupt vectors for each interrupt request level, as they would appear when the vector is driven onto the data bus.

Bit	Description
7:3	**Interrupt Vector Base Address:** Bits[7:3] define the base address in the interrupt vector table for the interrupt routines associated with each interrupt request level input. For CNTRL-1, a typical value is 00001b, and for CNTRL-2, 10000b. The interrupt controller combines a binary code representing the interrupt level to receive service with this base address to form the interrupt vector that is driven out onto the bus. For example, the complete interrupt vector for IRQ[0] (CNTRL-1), would be 0000 1000b (CNTRL-1 [7:3] = 00001b and 000b representing IRQ[0]). This vector is used by the CPU to point to the address information that defines the start of the interrupt routine.
2:0	**Interrupt Request Level:** When writing ICW2, these bits should all be 0. During an interrupt acknowledge cycle, these bits will be programmed by the interrupt controller with the interrupt code representing the interrupt level to be serviced. This interrupt code is combined with bits[7:3] to form the complete interrupt vector driven onto the data bus during the second INTA# cycle. The table in Section 9.8.1 outlines each of these codes. The code is a simple three bit binary code: 000b represents IRQ0 (IRQ8), 001b IRQ1 (IRQ9), 010b IRQ2 (IRQ10), and so on until 111b IRQ7 (IRQ15).

3.4.3 ICW3—INITIALIZATION COMMAND WORD 3 (MASTER)

Register Location: 021h—INT CNTRL-1
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

The meaning of ICW3 differs between CNTRL-1 and CNTRL-2. On CNTRL-1, the master controller, ICW3 indicates which CNTRL-1 IRQ line physically connects the INT output of CNTRL-2 to CNTRL-1. ICW3 must be programmed to 04h, indicating the cascade of the CNTRL-2 INT output to the IRQ[2] input of CNTRL-1.

An interrupt request on IRQ2 causes CNTRL-1 to enable CNTRL-2 to present the interrupt vector address during the second interrupt acknowledge cycle.

Bit	Description
7:3, 1:0	**Cascade Interrupt Controller IRQs:** Bits[7:3] and bits[1:0] must be programmed to 0.
2	**Cascade Interrupt Controller IRQs:** Bit 2 must always be programmed to a 1. This bit indicates that CNTRL-2, the slave controller, is cascaded on interrupt request line two (IRQ[2]). When an interrupt request is asserted to CNTRL-2, the IRQ goes through the priority resolver. After the slave controller priority resolution is finished, the INT output of CNTRL-2 is asserted. However, this INT assertion does not go directly to the CPU. Instead, the INT assertion cascades into IRQ[2] on CNTRL-1. IRQ[2] must go through the priority resolution process on CNTRL-1. If it wins the priority resolution on CNTRL-1 and the CNTRL-1 INT signal is asserted to the CPU, the returning interrupt acknowledge cycle is really destined for CNTRL-2. The interrupt was originally requested at CNTRL-2, so the interrupt acknowledge is destined for CNTRL-2, and not a response for IRQ[2] on CNTRL-1. When an interrupt request from IRQ[2] wins the priority arbitration, in reality an interrupt from CNTRL-2 has won the arbitration. Because bit 2 of ICW3 on the master is set to 1, the master knows which identification code to broadcast on the internal cascade lines, alerting the slave controller that it is responsible for driving the interrupt vector during the second INTA# pulse.

3.4.4 ICW3—INITIALIZATION COMMAND WORD 3 (SLAVE)

Register Location: INT CNTRL-2 port address-0A1h
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

On CNTRL-2 (the slave controller), ICW3 is the slave identification code broadcast by CNTRL-1 from the trailing edge of the first INTA# pulse to the trailing edge of the second INTA# pulse. CNTRL-2 compares the value programmed in ICW3 with the incoming identification code. The code is broadcast over three ESC internal cascade lines. ICW3 must be programmed to 02h for CNTRL-2. When 010b is broadcast by CNTRL-1 during the INTA# sequence, CNTRL-2 assumes responsibility for broadcasting the interrupt vector during the second interrupt acknowledge cycle.

As an illustration, consider an interrupt request on IRQ[2] of CNTRL-1. By definition, a request on IRQ[2] must have been asserted by CNTRL-2. If IRQ[2] wins the priority resolution on CNTRL-1, the interrupt acknowledge cycle returned by the CPU following the interrupt is destined for CNTRL-2, not CNTRL-1. CNTRL-1 will see the INTA# signal, and knowing that the actual destination is CNTRL-2, will broadcast a slave identification code across the internal cascade lines. CNTRL-2 will compare this incoming value with the 010b stored in ICW3. Following a positive decode of the incoming message from CNTRL-1, CNTRL-2 will drive the appropriate interrupt vector onto the data bus during the second interrupt acknowledge cycle.

Bit	Description
7:3	**Reserved:** Must be 0.
2:0	**Slave Identification Code:** The Slave Identification code must be programmed to 010b during the initialization sequence. The code stored in ICW3 is compared to the incoming slave identification code broadcast by the master controller during interrupt acknowledge cycles.

3.4.5 ICW4—INITIALIZATION COMMAND WORD 4

Register Location: 021h—INT CNTRL-1
0A1h—INT CNTRL-2
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

Both ESC interrupt controllers must have ICW4 programmed as part of their initialization sequence. Minimally, the microprocessor mode bit, bit 0, must be set to a 1 to indicate to the controller that it is operating in an 80x86 based system. Failure to program this bit will result in improper controller operation during interrupt acknowledge cycles. Additionally, the Automatic End of Interrupt (AEOI) may be selected, as well as the Special Fully Nested Mode (SFNM) of operation.

The default programming for ICW4 is 01h, which selects 80x86 mode, normal EOI, buffered mode, and special fully nested mode disabled.

Bits 2 and 3 must be programmed to 0 for the ESC interrupt unit to function correctly.

Both bit 1, AEOI, and bit 4, SFNM, can be programmed if the system developer chooses to invoke either mode.

Bit	Description
7:5	**Reserved:** Must be 0.
4	**SFNM:** Bit 4, SFNM, should normally be disabled by writing a 0 to this bit. If SFNM = 1, the special fully nested mode is programmed.
3:2	**Master/Slave Buffer Mode (BUF):** Bit 3, BUF, must be programmed to 0 for the ESC. This is non-buffered mode. While different programming options are sometimes offered for bits 2 and 3, within the ESC interrupt unit, bits 2 and 3 must always be programmed to 00b.
1	**AEOI:** Bit 1, AEOI, should normally be programmed to 0. This is the normal end of interrupt. If AEOI = 1, the automatic end of interrupt mode is programmed.
0	**Microprocessor Mode:** The Microprocessor Mode bit must be programmed to 1 to indicate that the interrupt controller is operating in an 80x86 based system. Never program this bit to 0.

3.4.6 OCW1—OPERATION CONTROL WORD 1

Register Location: 021h—INT CNTRL-1
0A1h—INT CNTRL-2
Default Value: xxh
Attribute: Read/Write
Size: 8 Bits

OCW1 sets and clears the mask bits in the interrupt Mask register (IMR). Each interrupt request line may be selectively masked or unmasked any time after initialization. A single byte is written to this register. Each bit position in the byte represents the same-numbered channel: Bit 0 = IRQ[0], bit 1 = IRQ[1] and so on. Setting the bit to a 1 sets the mask, and clearing the bit to a 0 clears the mask. Note that masking IRQ[2] on CNTRL-1 will also mask all of controller 2's interrupt requests (IRQ8-IRQ15). Reading OCW1 returns the controller's mask register status.

The IMR stores the bits which mask the interrupt lines to be masked. The IMR operates on the IRR. Masking of a higher priority input will not effect the interrupt request lines of lower priority.

Unlike status reads of the ISR and IRR, for reading the IMR, no OCW3 is needed. The output data bus will contain the IMR whenever I/O read is active and the I/O port address is 021h or 0A1h (OCW1).

All writes to OCW1 must occur following the ICW1-ICW4 initialization sequence, since the same I/O ports are used for OCW1, ICW2, ICW3 and ICW4.

Bit	Description
7:0	**Interrupt Request Mask:** When a 1 is written to any bit in this register, the corresponding IRQ[x] line is masked. For example, if bit 4 is set to a 1, then IRQ[4] will be masked. Interrupt requests on IRQ[4] will not set Channel 4's interrupt request register (IRR) bit as long as the channel is masked. When a 0 is written to any bit in this register, the corresponding IRQ[x] mask bit is cleared, and interrupt requests will again be accepted by the controller. Note that masking IRQ[2] on CNTRL-1 will also mask the interrupt requests from CNTRL-2, which is physically cascaded to IRQ[2].

3.4.7 OCW2—OPERATION CONTROL WORD 2

Register Location: 020h—INT CNTRL-1
0A0h—INT CNTRL-2
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

OCW2 controls both the Rotate Mode and the End of Interrupt Mode, and combinations of the two. The three high order bits in an OCW2 write represent the encoded command. The three low order bits are used to select individual interrupt channels during three of the seven commands. The three low order bits (labeled L2, L1 and L0) are used when bit 6, the SL bit, is set to a 1 during the command.

Following a reset and ICW initialization, the controller enters the fully nested mode of operation. Non-specific EOI without rotation is the default. Both rotation mode and specific EOI mode are disabled following initialization.

Bit	Description
7:5	**Rotate and EOI Codes (R, SL, EOI):** These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations is listed above under the bit definition. **Bits[7:5] Function** 001 Non-specific EOI command 011 Specific EOI Command 101 Rotate on Non-Specific EOI Command 100 Rotate in Auto EOI Mode (Set) 000 Rotate in Auto EOI Mode (Clear) 111 *Rotate on Specific EOI Command 110 *Set Priority Command 010 No Operation * L0 - L2 Are Used
4:3	**OCW2 Select:** When selecting OCW2, bits 3 and 4 must both be 0. If bit 4 is a 1, the interrupt controller interprets the write to this port as an ICW1. Therefore, always ensure that these bits are both 0 when writing an OCW2.
2:0	**Interrupt Level Select (L2, L1, L0):** L2, L1, and L0 determine the interrupt level acted upon when the SL bit is active. A simple binary code, outlined above, selects the channel for the command to act upon. When the SL bit is inactive, these bits do not have a defined function; programming L2, L1 and L0 to 0 is sufficient in this case. **Bit[2:0] Interrupt Level** 000 IRQ 0(8) 001 IRQ 1(9) 010 IRQ 2(10) 011 IRQ 3(11) 100 IRQ 4(12) 101 IRQ 5(13) 110 IRQ 6(14) 111 IRQ 7(15)

3.4.8 OCW3—OPERATION CONTROL WORD 3

Register Location: 020h—INT CNTRL-1
0A0h—INT CNTRL-2
Default Value: x01xxx10b
Attribute: Read/Write
Size: 8 Bits

OCW3 serves three important functions; Enable Special Mask Mode, Poll Mode control, and IRR/ISR register read control.

First, OCW3 is used to set or reset the Special Mask Mode (SMM). The Special Mask Mode can be used by an interrupt service routine to dynamically alter the system priority structure while the routine is executing, through selective enabling/disabling of the other channel's mask bits.

Second, the Poll Mode is enabled when a write to OCW3 is issued with Bit 2 equal to 1. The next I/O read to the interrupt controller is treated like an interrupt acknowledge; a binary code representing the highest priority level interrupt request is released onto the bus.

Third, OCW3 provides control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). Either the ISR or IRR is selected for reading with a write to OCW3. Bits 0 and 1 carry the encoded command to select either register. The next I/O read to the OCW3 port address will return the register status specified during the previous write. The register specified for a status read is retained by the interrupt controller. Therefore, a write to OCW3 prior to every status read command is unnecessary, provided the status read desired is from the register selected with the last OCW3 write.

Bit	Description
7	**Reserved:** Must be 0.
6	**SMM:** If ESMM = 1 and SMM = 1 the Interrupt Controller will enter Special Mask Mode. If ESMM = 1 and SMM = 0 the Interrupt Controller will revert to normal mask mode. When ESMM = 0, SMM has no effect.
5	**Enable Special Mask Mode:** When this bit is set to 1 it enables the SMM bit to set or reset the Special Mask Mode. When ESMM = 0 the SMM bit becomes a "don't care".
4:3	**OCW3 Select:** When selecting OCW3, bit 3 must be a 1 and bit 4 must be 0. If bit 4 is a 1, the Interrupt Controller interprets the write to this port as an ICW1. Therefore, always ensure that bits[4:3] are "01b" when writing an OCW3.
2	**Poll Mode Command:** When bit 2 is a 0, the Poll command is not issued. When bit 2 is a 1, the next I/O read to the Interrupt Controller is treated as an Interrupt Acknowledge cycle. An encoded byte is driven onto the data bus, representing the highest priority level requesting service.
1:0	**Register Read Command:** Bits[1:0] provide control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). When bit 1 = 0, bit 0 will not effect the register read selection. When bit 1 = 1, bit 0 selects the register status returned following an OCW3 read. If bit 0 = 0, the IRR will be read. If bit 0 = 1, the ISR will be read. Following ICW initialization, the default OCW3 port address read will be "read IRR". To retain the current selection (read ISR or read IRR), always write a 0 to bit 1 when programming this register. The selected register can be read repeatedly without reprogramming OCW3. To select a new status register, OCW3 must be reprogrammed prior to attempting the read. **Bits[1:0] Function** 00 No Action 01 No Action 10 Read IRQ Register 11 Read IS Register

3.4.9 ELCR—EDGE/LEVEL CONTROL REGISTER

Register Location: 04D0h—INT CNTRL-1
04D1h—INT CNTRL-1
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

The Edge/Level Control Register is used to set the interrupts to be triggered by either the signal edge or the logic level. INT0, INT1, INT2, INT8, INT13 must be set to edge sensitive. After a reset all the INT signals are set to edge sensitive.

Programming Considerations:
If an interrupt is switched from level to edge sensitive, a false interrupt is generated on that interrupt line. If the IRQx line is high, then switching the level/edge bet from a 1 to a 0 causes the interrupt controller to detect an interrupt. Also note that even if this interrupt is masked when programming this register, the interrupt controller still latches the false interrupt. As soon as this interrupt is unmasked, the false interrupt is processed.

Thus, before switching the edge/level function, disable interrupts to the processor (either mask interrupts or CLI instruction). Then program the ELCR Register. Finally, re-initialize the interrupt controller to clear the false interrupt.

Bit	Description
7:0	**Edge/Level Select:** The bits select if the interrupts are triggered by either the signal edge or the logic level. A 0 bit represents an edge sensitive interrupt, and a 1 is for level sensitive. Bit[2:0] and bit 13 must be set to 0. After A reset or power-on these registers are set to 00h.

Bit	Port 04D0h	Port 04D1h
0	INT0	INT8
1	INT1	INT9
2	INT2	INT10
3	INT3	INT11
4	INT4	INT12
5	INT5	INT13
6	INT6	INT14
7	INT7	INT15

3.4.10 NMISC—NMI STATUS AND CONTROL REGISTER

Register Location: 061h
Default Value: X0X0 0000
Attribute: Read/Write, Read Only
Size: 8 Bits

This register is used to check the status of different system components, control the output of the Speaker Counter (Timer 1, Counter 2), and gate the counter output that drives the SPKR signal. This register also controls NMI generation and reports NMI source status. Note that NMI generation is globally enabled/disabled via the NMIERTC Register and NMI generation for SERR# is controlled via the MS Register. Bits[7:4] of this register are read-only and must be written as 0s when writing to this register. Bits[3:0] are read/write. Following reset, bit 7 returns the PCI System Board Parity Error status (PERR#) and bit 5 is undetermined until Counter 2 is properly programmed.

Bit	Description
7	**System Board Error—RO:** Bit 7 is set if the PERR# line is pulsed. This interrupt is enabled by setting bit 2 to 0. To reset the interrupt, set bit 2 to 0 and then set it to 1. Note that this bit does not reflect status of an NMI caused by SERR#, which is enabled and disabled/cleared via the MS Register.
6	**IOCHK# NMI Source—RO:** Bit 6 is set if an expansion board asserts IOCHK# on the ISA/EISA Bus. This interrupt is enabled by setting bit 3 to 0. To reset the interrupt, set bit 3 to 0 and then set it to 1.
5	**Timer 1, Counter 2—RO:** The Timer 1, Counter 2 OUT signal state is reflected in bit 5. The value on this bit following a read is the current state of the Counter 2 OUT signal. Counter 2 must be programmed following a reset for this bit to have a determinate value.
4	**Refresh Cycle Toggle—RO:** The Refresh Cycle Toggle signal toggles from either 0 to 1 or 1 to 0 following every refresh cycle.
3	**IOCHK# NMI Enable—R/W:** When bit 3 is a 1, IOCHK# NMI's are disabled and cleared, and when bit 3 is a 0 (default), IOCHK# NMI's are enabled.
2	**PCI System Board Error—R/W:** When bit 2 is a 1, the system board error is disabled and cleared. When bit 2 is a 0 (default), the system board parity error is enabled. Note that NMI generation for system board errors is enabled/disabled via bit 3 (System Error) of the Mode Select Register. Following reset, bit 2 is a 0, and system board errors are enabled.
1	**Speaker Data Enable—R/W:** Speaker Data Enable is ANDed with the Timer 1, Counter 2 OUT signal to drive the SPKR output signal. When bit 1 is a 0 (default), the result of the AND is always 0 and the SPKR output is always 0. When bit 1 is a 1, the SPKR output is equivalent to the Counter 2 OUT signal value.
0	**Timer 1, Counter 2 Gate Enable—R/W:** When bit 0 is a 0, Timer 1, Counter 2 counting is disabled. Counting is enabled when bit 0 is a 1. This bit controls the GATE input to Counter 2.

3.4.11 NMIERTC—NMI CONTROL AND REAL-TIME CLOCK ADDRESS

Register Location: 070h
Default Value: See below
Attribute: Write Only
Size: 8 Bits

The most-significant bit enables or disables all NMI sources including PERR#, SERR#, IOCHK#, Fail-Safe Timer, Bus Timeout, and the NMI Port. Write an 80h to The NMIERTC Register to mask the NMI signal. This register is shared with the real-time clock. The real-time-clock uses the lower six bits of this port to address memory locations. Writing to port 70h sets both the enable/disable bit and the memory address pointer. Do not modify the contents of this register without considering the effects on the state of the other bits.

Bit	Description
7	**NMI Enable:** Setting bit 7 to a 1 will disable all NMI sources. Setting the bit to a 0 enables the NMI interrupt.
6:0	**Real-Time Clock Address:** Used by the Real-Time Clock on the Base I/O component to address memory locations. Not used for NMI enabling/disabling.

3.4.12 NMIESC—NMI EXTENDED STATUS AND CONTROL REGISTER

Register Location: 0461h
Default Value: See below
Attribute: Read/Write, Read Only
Size: 8 Bits

This register is used to check the status of different system components, control the output of the Speaker Counter (Timer 1, Counter 2), and gate the counter output that drives the SPKR signal.

Bits 4, 5, 6, and 7 are read-only. Bits 0-3 are both read and write. When writing to this port, these bits must be written as 0's. Bit 7 returns the Fail-Safe Timer Status. This input comes from Timer 2, Counter 0. The current status of bit 2 enables or disables this Fail-Safe Timer NMI source. Bit 6 returns the Bus Timeout Status. Bit 6 is set if either a 64 BCLK or a 256 BCLK occurs. The current status of bit 3 enables or disables this Fail-Safe Timer NMI source. If NMI is caused by a Bus Timeout, bit 4 distinguished between the 8 μs (64 BCLK) and

32 μs (256 BCLK) timeout. Bit 5 is the current state of an I/O write to port 0462h. The current status of bit 1 enables or disables Software generated NMI. Bit 0 controls the state of the RSTDRV output signal. If bit 0 is set to 1, the RSTDRV signal is asserted and a system bus reset is performed. Bit 0 should be set long enough (> 8 BCLKs) for the system bus devices to be properly reset.

Bit	Description
7	**Fail-Safe Timer Status—RO:** This bit indicates the status of the Fail-safe Timer. When Timer 2, Counter 0 count expires, this bit is set to a 1 if bit 2 has previously been set to 1. A value of 0 indicates that the current NMI was not caused by the Fail-Safe Timer. A value of 1 indicates that the Fail-Safe timer has timed out.
6	**Bus Timeout Status—RO:** This bit indicates the status of Bus master timeout logic. If this bit is 0, the Bus Master timeout logic has not detected a bus timeout. If this bit is 1, the bus master timeout logic has detected a bus timeout.
5	**Software NMI Status—RO:** This bit indicates the status of the Software NMI port writes. A write to I/O port 0462 of any value will set this bit to 1 if bit 1 is set to 1. If this bit is 0, the current NMI was not caused by a write to the NMI Port. If this bit is 1, the current NMI was caused by a write to the NMI Port.
4	**Bus Timeout Status—RO:** This bit indicates the status of the 8 μs EISA Bus master timeout event. If the bit is 0, the current NMI was not caused by the 8 μs EISA bus master timeout. If this bit is 1, the current NMI was caused by this bus timeout.
3	**Bus Timeout Enable—R/W:** This bit enables/disables NMI EISA bus timeout. If this bit is 0, an NMI will not be generated for bus timeout. Also the NMI condition caused by the Bus timeout will be cleared. If this bit is 1 an NMI will be generated when Timer 2 Counter 0 count expires.
2	**Fail-Safe NMI Enable—R/W:** This bit enables/disables NMI when the Fail-Safe Timer timesout. If this bit is 0, an NMI will not be generated when the Timer 2 Counter 0 count expires. Also the NMI condition caused by the Fail-Safe Timer will be cleared. If this bit is 1 an NMI will be generated when Timer 2 Counter 0 count expires.
1	**Software NMI Enable—R/W:** This bit enables/disables software generated NMI. If this bit is 0, a write to I/O port 0462h will not generate an NMI. If this bit is 1 NMI will be generated for a write to I/O port 0462h.
0	**Bus Reset—R/W:** When bit 0 is a 0, RSTDRV signal function as a normal reset drive signal. When bit 0 is 1, the RSTDRV signal is asserted. Following reset, bit 0 is a 0 and the RSTDRV output is low.

3.4.13 SOFTNMI—SOFTWARE NMI GENERATION REGISTER

Register Location: 0462h
Default Value: xxh
Attribute: Write Only
Size: 8 Bits

A write to this port with any data will cause an NMI. This port provides a software mechanism to cause an NMI if interrupts are enabled.

Bit	Description
7:0	**Software NMI Port:** The bit pattern is not specific. A write to this port will generate a Software NMI if enabled.

3.5 EISA Configuration, Floppy Support, and Port 92h

3.5.1 CONFRAMP—CONFIGURATION RAM PAGE REGISTER

Register Location: 0C00h
Default Value: xxx00000b
Attribute: Read/Write
Size: 8 Bits

This register contains the Configuration RAM Page address. During accesses to the Configuration RAM (0800h–08FFh), the ESC drivers the CPG[4:0] signals with the value of bits[4:0] of this register. The CPG[4:0] signals are connected to address pins ADDR[12:8] of the Configuration RAM.

Bit	Description
7:5	**Reserved**
4:0	**CRAM Page Address:** The value of these bit selects a specific page from the Configuration RAM space. The SA[7:0] addresses select the location within this page during I/O accesses to the Configuration RAM. The value is driven onto CPG[4:0] during accesses to Configuration RAM.

3.5.2 DIGOUT—DIGITAL OUTPUT REGISTER

Register Location: 03F2h (Primary), 0372h (Secondary)
Default Value: xxxx0xxxb
Attribute: Write only
Size: 8 Bits

This register is used to prevent XBUSOE# from responding to DACK2# during a DMA read access to a floppy controller on the ISA bus. If a second floppy (residing on the ISA bus) is using DACK2# in conjunction with a floppy on the X-bus, this prevents the floppy on the X-Bus and the X-bus transceiver from responding to an access targeted for the floppy on the ISA bus. This register is also located in the floppy controller device.

Bit	Description
7:4	**Not Used:** These bits exist in the 82077 FDC. Refer to the 82077 data sheet for further details.
3	**DMA Enable:** When this bit is a 1, the assertion of DACK# will result in XBUSOE# being asserted. If this bit is 0, DACK2# has no effect on XBUSOE#. This port bit also exists on the 82077 FDC. This bit defaults to disable (0).
2:0	**Not Used:** These bits exist in the 82077 FDC. Refer to the 82077 data sheet for further details.

3.5.3 PORT 92 REGISTER

Register Location: 92h
Default Value: 00100100b
Attribute: Read/Write
Size: 8 Bits

This register is used to support the alternate reset (ALTRST#), alternate A20 (ALTA20), power-on password protection, and fixed disk light function (DLIGHT#). This register is only accessible if bit 6 in the Peripheral Chip Select Enable B Register is set to 1.

Bit	Description
7:6	**Fixed Disk Activity Light:** These bits are used to turn the Fixed Disk Activity Light on and off. When either of these bits are set to a 1, the light is turned on (DLIGHT# driven active). To turn the light off, both of these bits must be 0.
5	**Reserved:** This bit is reserved and will always return a 1 when read.
4	**Not Used:** This bit is not used and will always return a 0 when read.
3	**Power on Password Protection:** A 1 on this bit enables power-on password protection by inhibiting accesses to the RTC memory for RTC addresses (port 70h) from 36h to 3Fh. This is accomplished by not generating RTCRD# and RTCWR# signals for theses accesses.
2	**Reserved:** This bit is reserved and will always return a 1 when read.
1	**ALTA20 Signal:** Writing a 0 to this bit causes the ALTA20 signal to be driven low. Writing a 1 to this bit causes the ALTA20 signal to be driven high.
0	**ALTRST# Signal:** This read/write bit provides an alternate system reset function. This function provides an alternate means to reset the system CPU to effect a mode switch from Protected Virtual Address Mode to the Real Address Mode. This provides a faster means of reset than is provided by the Keyboard controller. This bit is set to a 0 by a system reset. Writing a 1 to this bit will cause the ALTRST# signal to pulse active (low) for approximately 4 SYSCLK's. Before another ALTRST# pulse can be generated, this bit must be written back to a 0.

3.5.4 LEISAMG—LAST EISA BUS MASTER GRANTED REGISTER

Register Location: 0464h
Default Value: xxh
Attribute: Read Only
Size: 8 Bits

This register contains information about which EISA bus master most recently had control of the EISA bus. A bit read of 0 indicates that the corresponding slot most recently was granted the bus.

Bit	Description
7:0	**Last EISA Bus Master:** A value of 1 is placed in the bit position of the most recently granted EISA Bus Master.

3.6 Power Management Registers

This section describes two power management registers that are in the 82374SBAPMS and APMC Registers. These registers are located in normal I/O space and must be accessed (via the CPU or PCI Bus) with 8-bit accesses. Note that the rest of the power management registers are part of the ESC configuration registers.

3.6.1 APMC—ADVANCED POWER MANAGEMENT CONTROL PORT

I/O Address: 0B2h
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

This register passes data (APM Commands) between the OS and the SMI handler. In addition, writes can generate an SMI and reads can cause STPCLK# to be asserted. The ESC operation is not effected by the data in this register.

Bit	Description
7:0	**APM Control Port (APMC):** Writes to this register store data in the APMC Register and reads return the last data written. In addition, writes generate an SMI, if bit 7 of the SMIEN Register and bit 0 of the SMICNTL Register are both is set to 1. Reads cause the STPCLK# signal to be asserted, if bit 1 of the SMICNTL Register is set to 1. Reads do not generate an SMI.

3.6.2 APMS—ADVANCED POWER MANAGEMENT STATUS PORT

I/O Address: 0B3h
Default Value: 00h
Attribute: Read/Write
Size: 8 Bits

This register passes status information between the OS and the SMI handler. The ESC operation is not effected by the data in this register.

Bit	Description
7:0	**APM Status Port (APMS):** Writes store data in this register and reads return the last data written.

3.7 APIC Registers

This section describes the registers used to program the Advanced Programmable Interrupt Controller. The I/O APIC registers are accessed by an indirect addressing scheme using two registers (IOREGSEL and IOWIN) that are located in the CPU's memory space (memory address specified by the APICBASE Register). To reference an I/O APIC register, a Dword memory write loads the IOREGSEL Register with a 32-bit value that specifies the APIC register. The IOWIN Register then becomes a four byte window pointing to the APIC register specified by bits[7:0] of the IOREGSEL Register. The register address table is shown in the Address Decode section.

All APIC registers are accessed using 32-bit loads and stores. This implies that to modify a field (e.g., bit, byte) in any register, the whole 32-bit register must be read, the field modified, and the 32-bits written back. In addition, registers that are described as 64-bits wide are accessed as multiple independent 32-bit registers.

3.7.1 IOREGSEL—I/O REGISTER SELECT REGISTER

Memory Address: FEC0 x000h (82374EB) (x = See APICBASE Register)

FEC0 xy00h (82374SB) (xy = See APICBASE Register)

Default Value: 00h
Attribute: Read/Write
Size: 32 Bits

This register selects an I/O APIC Unit register. The contents of the selected 32-bit register can be manipulated via the I/O Window Register.

Bit	Description
31:8	**Reserved**
7:0	**APIC Register Address:** Bits[7:0] specify the APIC register to be read/written via the IOWIN Register.

3.7.2 IOWIN—I/O WINDOW REGISTER

Memory Address: FEC0 x010h (82374EB) (x = See APICBASE Register)

FEC0 xy10h (82374SB) (xy = See APICBASE Register)

Default Value: 00h
Attribute: Read/Write
Size: 32 Bits

This register is mapped onto the I/O Unit's register selected by the IOREGSEL Register. Readability/writability by software is determined by the I/O APIC register that is currently selected.

Bit	Description
31:0	**APIC Register Data:** Memory references to this register are mapped to the APIC register specified by the contents of the IOREGSEL Register.

3.7.3 APICID—I/O APIC IDENTIFICATION REGISTER

Address Offset: 00h
Default Value: 00h
Attribute: Read/Write
Size: 32 Bits

This register contains the unit's 4-bit APIC ID. The ID serves as a physical name of the I/O APIC Unit. All APIC units using the APIC bus should have a unique APIC ID. The APIC bus arbitration ID for the I/O unit is also writtten during a write to the APICID Register (same data is loaded into both). This register must be programmed with the correct ID value before using the I/O APIC unit for message transmission.

Bit	Description
31:28	**Reserved**
27:24	**I/O APIC Identification:** This 4-bit field contains the I/O APIC identification.
23:0	**Reserved**

3.7.4 APICID—I/O APIC IDENTIFICATION REGISTER

Address Offset: 01h
Default Value: 00h
Attribute: Read Only
Size: 32 Bits

The I/O APIC Version Register identifies the APIC hardware version. Software can use this to provide compatibility between different APIC implementations and their versions. In addition, this register provides the maximum number of entries in the I/O Redirection Table.

Bit	Description
31:24	**Reserved**
23:16	**Maximum Redirection Entry:** This field contains the entry number (0 being the lowest entry) of the highest entry in the I/O Redirection Table. The value is equal to the number of interrupt input pins minus one of this I/O APIC. The range of values is 0 through 239. For the ESC, this value is 0Fh.
15:8	**Reserved**
7:0	**APIC VERSION:** This 8 bit field identifies the implementation version. The version numbers are assigned for 82489DX and APIC as follows: 0Xh = 82489DX 1Xh = APIC 20–FFh = Reserved For the ESC, the current value is 11h.

3.7.5 APICARB—I/O APIC ARBITRATION REGISTER

Address Offset: 02h
Default Value: 000F0011h
Attribute: Read Only
Size: 32 Bits

The APICARB Register contains the bus arbitration priority for the I/O APIC. This register is loaded whenever the I/O APIC ID Register is written.

Bit	Description
31:28	**Reserved**
27:24	**I/O APIC Identification:** This 4 bit field contains the I/O APIC identification.
23:0	**Reserved**

3.7.6 IOREDTBL[15:0]—I/O REDIRECTION TABLE REGISTERS

Address Offset: 10-11h (IOREDTBL0)
12-13h (IOREDTBL1)
14-15h (IOREDTBL2)
16-17h (IOREDTBL3)
18-19h (IOREDTBL4)
1A-1Bh (IOREDTBL5)
1C-1Dh (IOREDTBL6)
1E-1Fh (IOREDTBL7)
20-21h (IOREDTBL8)
22-23h (IOREDTBL9)
24-25h (IOREDTBL10)
26-27h (IOREDTBL11)
28-29h (IOREDTBL12)
2A-2Bh (IOREDTBL13)
2C-2Dh (IOREDTBL14)
2E-2Fh (IOREDTBL15)
Default Value: xx000000 00010xxxh
Attribute: Read/Write, Read Only
Size: 64 Bits each

There are 16 I/O Redirection Table entry registers. Each register is a dedicated entry for each interrupt input pin. Unlike IRQ pins of the 8259A, the notion of interrupt priority is completely unrelated to the position of the physical interrupt input pin on the APIC. Instead, software determines the vector (and therefore the priority) for each corresponding interrupt input pin. For each interrupt pin, the operating system can also specify the signal polarity (low active or high active), whether the interrupt is signaled as edges or levels, as well as the destination and delivery mode of the interrupt. The information in the redirection table is used to translate the corresponding interrupt pin information into an inter-APIC message.

For a signal on an edge-sensitive interrupt input pin to be recognized as a valid edge (and not a glitch), the input level on the pin must remain asserted until the I/O APIC Unit broadcasts the corresponding message over the APIC bus and the message has been accepted by the destination(s) specified in the destination field. Only then will the source APIC be able to recognize a new edge on that Interrupt Input pin. That new edge only results in a new invocation of the handler if its acceptance by the destination APIC causes the Interrupt Request Register bit to go from 0 to 1. (In other words, if the interrupt wasn't already pending at the destination.)

Bit	Description
63:56	**Destination Field:** If the Destination Mode of this entry is Physical Mode (bit 11 = 0), bits[59:56] contain an APIC ID. If Logical Mode is selected (bit 11 = 1), the Destination Field potentially defines a set of processors. Bits[63:56] of the Destination Field specify the logical destination address.
55:17	**Reserved**
16	**Interrupt Mask:** When this bit is 1, the interrupt signal is masked. Edge-sensitive interrupts signaled on a masked interrupt pin are ignored (i.e., not delivered or held pending). Level-asserts or negates occurring on a masked level-sensitive pin are also ignored and have no side effects. Changing the mask bit from unmasked to masked after the interrupt is accepted by a local APIC has no effect on that interrupt. This behavior is identical to the case where the device withdraws the interrupt before that interrupt is posted to the processor. It is software's responsibility to handle the case where the mask bit is set after the interrupt message has been accepted by a local APIC unit but before the interrupt is dispensed to the processor. When this bit is 0, the interrupt is not masked. An edge or level on an interrupt pin that is not masked results in the delivery of the interrupt to the destination.
15	**Trigger Mode—R/W:** The trigger mode field indicates the type of signal on the interrupt pin that triggers an interrupt. This bit is set to 1 for level sensitive and 0 for edge sensitive.
14	**Remote IRR—RO:** This bit is used for level triggered interrupts. Its meaning is undefined for edge triggered interrupts. For level triggered interrupts, this bit is set to 1 when local APIC(s) accept the level interrupt sent by the I/O APIC. The Remote IRR bit is set to 0 when an EOI message with a matching interrupt vector is received from a local APIC.
13	**Interrupt Input Pin Polarity (INTPOL)—R/W:** This bit specifies the polarity of the interrupt signal. A 0 selects high active and a 1 selects low active.
12	**Delivery Status (DELIVS)—RO:** The Delivery Status bit contains the current status of the delivery of this interrupt. Delivery Status is read-only and writes to this bit (as part of a 32 bit word) do not effect this bit. When bit 12 = 0 (IDLE), there is currently no activity for this interrupt. When bit 12 = 1 (Send Pending), the interrupt has been injected. However, its delivery is temporarily held up due to the APIC bus being busy or the inability of the receiving APIC unit to accept that interrupt at that time.
11	**Destination Mode (DESTMOD)—R/W:** This field determines the interpretation of the Destination field. When DESTMOD = 0 (physical mode), a destination APIC is identified by its ID. Bits 56 through 59 of the Destination field specify the 4 bit APIC ID. When DESTMOD = 1 (logical mode), destinations are identified by matching on the logical destination under the control of the Destination Format Register and Logical Destination Register in each Local APIC. Bits 56 through 63 (8 MSB) of the Destination field specify the 8 bit APIC ID.

Bit	Description
10:8	**Delivery Mode (DELMOD)—R/W:** The Delivery Mode is a 3 bit field that specifies how the APICs listed in the destination field should act upon reception of this signal. Note that certain Delivery Modes only operate as intended when used in conjunction with a specific trigger Mode. These restrictions are indicated in the following table for each Delivery Mode.

Bits [10:8]	Mode	Description
000	Fixed	Deliver the signal on the INTR signal of all processor cores listed in the destination. Trigger Mode for "fixed" Delivery Mode can be edge or level.
001	Lowest Priority	Deliver the signal on the INTR signal of the processor core that is executing at the lowest priority among all the processors listed in the specified destination. Trigger Mode for "lowest priority". Delivery Mode can be edge or level.
010	SMI	System Management Interrupt. A delivery mode equal to SMI requires an edge trigger mode. The vector information is ignored but must be programmed to all zeroes for future compatibility.
011	Reserved	
100	NMI	Deliver the signal on the NMI signal of all processor cores listed in the destination. Vector information is ignored. NMI must be programmed as edge-triggered for proper operation.
101	INIT	Deliver the signal to all processor cores listed in the destination by asserting the INIT signal. All addressed local APICs will assume their INIT state. INIT must be programmed as edge-triggered for proper operation
110	Reserved	
111	ExtINT	Deliver the signal to the INTR signal of all processor cores listed in the destination as an interrupt that originated in an externally connected (8259A-compatible) interrupt controller. The INTA cycle that corresponds to this ExtINT delivery is routed to the external controller that is expected to supply the vector. A Delivery Mode of "ExtINT" requires an edge trigger mode.

7:0	**Interrupt Vector (INTVEC)—R/W:** The vector field is an 8 bit field containing the interrupt vector for this interrupt. Vector values range from 10 to FEh.

4.0 ADDRESS DECODING

The ESC contains an address decoder to decode EISA/ISA master cycles. The ESC address decoder uses the address line LA[31:2], and byte enable BE[3:0]# to decode EISA master cycles. For ISA master cycles, the ESC uses address line LA[31:2], SA[1:0], and high byte enable SHBE# for address decode.

The ESC decodes the following set of addresses.

1. BIOS memory space.
2. I/O addresses contained within the ESC.
3. Configuration registers.
4. X-Bus Peripherals.
5. Memory addresses for accessing APIC.

4.1 BIOS Memory Space

The ESC supports a total of 512 KBytes of BIOS. The ESC will assert the LBIOSCS# signal for memory cycles decoded to be in the BIOS space. The 512 KBytes of BIOS includes the conventional 128 KBytes of BIOS and 384 KBytes of enlarged BIOS.

The 128 KBytes conventional BIOS memory space is mapped at 1 MByte boundary between memory address 000E0000h–000FFFFFh. The 128 KByte conventional BIOS memory space is split into one 64 KByte region, and four 16 KByte regions. These regions are Low BIOS region 1 (000E0000h–000E3FFFh), Low BIOS region 2 (000E4000h–000E7FFFh), Low BIOS region 3 (000E8000h–000EBFFFh), and Low BIOS region 4 (000EC000h–000EFFFFh) and High BIOS region (000F0000h–000FFFFFh). The ESC will assert the LBIOSCS# signal for memory cycles to these regions if the corresponding configuration bits in the BIOS Chip Select A register are set to enable (see Table 3).

The conventional BIOS is aliased at multiple memory regions. The aliased memory regions are at 16 MByte boundary (High BIOS only), 4 GByte minus 1 MByte boundary, and 4 GByte boundary. The ESC will assert LBIOSCS# for memory cycles to these aliased regions if the corresponding configuration bits in the BIOS Chip Select B register are also set to enable (see Table 3).

The ESC supported VGA BIOS on the motherboard by aliasing the VGA BIOS region to the conventional BIOS region. The VGA BIOS is accessed at memory region 0000C0000h–0000C7FFF. The VGA BIOS region is divided into a Low VGA region (000C0000h–000C3FFFh) and a High VGA region (000C4000h–000C7FFFh). If the BIOS Chip Select B register bit 0 (Low VGA BIOS Enable) and bit 1 (High VGA BIOS Enable) are set to enable, memory accesses to Low VGA BIOS region and High VGA BIOS region will be aliased to conventional Low BIOS region 1 and Low BIOS region 2 respectively and the ESC will assert LBIOSCS#

The ESC supports the 384 KBytes of enlarged BIOS as specified by the PCI specification. This 384 KByte region is mapped in memory space below the 4 GByte aliased conventional BIOS. The enlarged BIOS is accessed between FFF80000h–FFFDFFFFh memory space. If the enlarged BIOS is enabled in the BIOS Enable Chip Select 1 register bit 5 (Enlarged BIOS Enable), the ESC will assert LBIOSCS# signal for accesses to this region.

BIOS Location Auto-Detection

Some applications require that Flash-EPROM based BIOS be updated in the system from the data coming from the floppy disk. To support this, the X-bus signals must be properly controlled (i.e. the ESC's X-Bus control logic must be aware of physical BIOS location—X-bus or ISA Bus). This is supported transparently to the software by configuring ESC's X-Bus logic during RESET using the SLOWH# pin.

Logic level on SLOWH# pin is sampled at the end of reset sequence to determine whether BIOS resides on the X-Bus (1) or on the ISA bus (0). This information is used by the ESC to control the X-Bus transceivers during BIOS access.

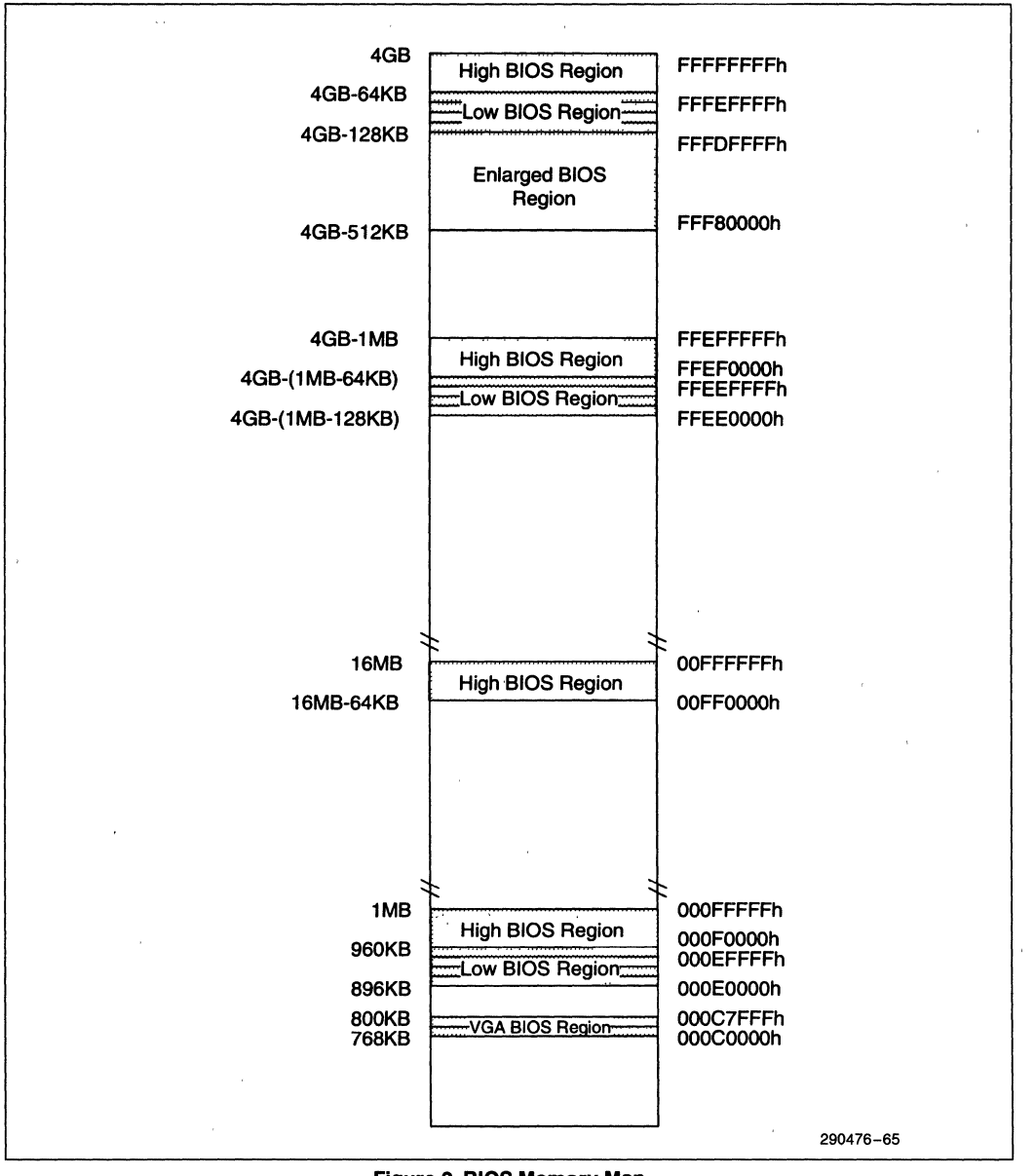


Figure 2. BIOS Memory Map

Table 3. BIOS Chip Select Enable Table

Memory Address Range	Low BIOS En				High BIOS En	ENL BIOS En	Low VGA BIOS En	High VGA BIOS En	16M BIOS En	LBIOSCS# Asserted
	1	2	3	4						
000C0000h to	x	x	x	x	x	x	**0**	x	x	No
000C3FFFh	x	x	x	x	x	x	**1**	x	x	Yes
000C4000h to	x	x	x	x	x	x	x	**0**	x	No
000C7FFFh	x	x	x	x	x	x	x	**1**	x	Yes
000E0000h to	**0**	x	x	x	x	x	x	x	x	No
000E3FFFh	**1**	x	x	x	x	x	x	x	x	Yes
000E4000h to	x	**0**	x	x	x	x	x	x	x	No
000E7FFFh	x	**1**	x	x	x	x	x	x	x	Yes
000E8000h to	x	x	**0**	x	x	x	x	x	x	No
000EBFFFh	x	x	**1**	x	x	x	x	x	x	Yes
000EC000h to	x	x	x	**0**	x	x	x	x	x	No
000EFFFFh	x	x	x	**1**	x	x	x	x	x	Yes
000F0000h to	x	x	x	x	**0**	x	x	x	x	No
000FFFFFh	x	x	x	x	**1**	x	x	x	x	Yes
(960KB to 1MB)										
00FF0000h to	x	x	x	x	**x**	x	x	x	**0**	No
00FFFFFFh	x	x	x	x	**0**	x	x	x	**0**	Yes
(16MB-64KB to	x	x	x	x	**1**	x	x		**1**	
16MB)										
FFEE0000h to	**0**	x	x	x	x	x	x	x	x	No
FFEE3FFFh	**1**	x	x	x	x	x	x	x	x	Yes
FFEE4000h to	x	**0**	x	x	x	x	x	x	x	No
FFEE7FFFh	x	**1**	x	x	x	x	x	x	x	Yes
FFEE8000h to	x	x	**0**	x	x	x	x	x	x	No
FFEEBFFFh	x	x	**1**	x	x	x	x	x	x	Yes
FFEEC000h to	x	x	x	**0**	x	x	x	x	x	No
FFEEFFFFh	x	x	x	**1**	x	x	x	x	x	Yes
FFEF0000h to	x	x	x	x	**0**	x	x	x	x	No
FFEFFFFFh	x	x	x	x	**1**	x	x	x	x	Yes
FFF80000h to	x	x	x	x	x	**0**	x	x	x	No
FFFDFFFFh	x	x	x	x	x	**1**	x	x	x	Yes
(4GB–512KB to										
4G–128KB										
FFFE0000h to	**0**	x	x	x	x	x	x	x	x	No
FFFE3FFFh	**1**	x	x	x	x	x	x	x	x	Yes
FFFE4000h to	x	**0**	x	x	x	x	x	x	x	No
FFFE7FFFh	x	**1**	x	x	x	x	x	x	x	Yes

2

Table 3. BIOS Chip Select Enable Table (Continued)

Memory Address Range	Low BIOS En				High BIOS En	ENL BIOS En	Low VGA BIOS En	High VGA BIOS En	16M BIOS En	LBIOSCS# Asserted
	1	2	3	4						
FFFE8000h to	x	x	**0**	x	x	x	x	x	x	No
FFFEBFFFh	x	x	**1**	x	x	x	x	x	x	Yes
FFFEC000h to	x	x	x	**0**	x	x	x	x	x	No
FFFEFFFFh	x	x	x	**1**	x	x	x	x	x	Yes
FFFF0000h to	x	x	x	x	**0**	x	x	x	x	No
FFFFFFFFh	x	x	x	x	**1**	x	x	x	x	Yes

NOTES:
1. "x" in the above table represents a don't care condition.
2. All the region control bits for the BIOS space are in the BIOS Chip Select A register and BIOS Chip Select 2 register at configuration offsets 42h and 43h respectively.

4.2 I/O Addresses Contained Within The ESC

The ESC integrates functions like DMA, Programmable Interrupt Controller, and Timers. All the compatibility registers associated with these functions are also integrated into the ESC. The ESC also integrates some additional registers like EISA System ID register in order to reduce the overall chip count in the system.

All the registers integrated in the ESC are located in the I/O range. These are 8-bit registers and are accessed through the ESC EISA interface. The ESC internal registers are at fixed I/O locations with the exception of DMA Scatter-Gather registers. The DMA Scatter-Gather registers default to the I/O addresses 0410h to 043Fh upon reset. These registers can be relocated by programming the Scatter-Gather Relocate Base Address register. The DMA Scatter-Gather registers can be relocated to I/O addresses range xx10h-xx3Fh.

Registers at I/O addresses 70h, 372h, and 3F2h are shared registers between ESC and external logic. Port 70h is duplicated in the Real Time Clock logic. Bit 3 of ports 372h and 3F2h reside in the ESC and the other bits reside in the Floppy Disk Controller. Table 4 documents the I/O address to the ESC internal registers.

Table 4. ESC I/O Register Address Map

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
0000H	0000	0000	000x	0000	R/W	DMA1 CH0 Base and Current Address	DMA
0001h	0000	0000	000x	0001	R/W	DMA1 CH0 Base and Current Count	DMA
0002h	0000	0000	000x	0010	R/W	DMA1 CH1 Base and Current Address	DMA
0003h	0000	0000	000x	0011	R/W	DMA1 CH1 Base and Current Count	DMA
0004h	0000	0000	000x	0100	R/W	DMA1 CH2 Base and Current Address	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
0005H	0000	0000	000x	0101	R/W	DMA1 CH2 Base and Current Count	DMA
0006h	0000	0000	000x	0110	R/W	DMA1 CH3 Base and Current Address	DMA
0007h	0000	0000	000x	0111	R/W	DMA1 CH3 Base and Current Count	DMA
0008h	0000	0000	000x	1000	R/W	DMA1 Status(r) Command(w) Register	DMA
0009h	0000	0000	000x	1001	WO	DMA1 Write Request	DMA
000Ah	0000	0000	000x	1010	WO	DMA1 Write Single Mask Bit	DMA
000Bh	0000	0000	000x	1011	WO	DMA1 Write Mode	DMA
000Ch	0000	0000	000x	1100	WO	DMA1 Clear Byte Pointer	DMA
000Dh	0000	0000	000x	1101	WO	DMA1 Master Clear	DMA
000Eh	0000	0000	000x	1110	WO	DMA1 Clear Mask	DMA
000Fh	0000	0000	000x	1111	R/W	DMA1 Read/Write All Mask Register Bits	DMA
0020h	0000	0000	001x	xx00	R/W	INT 1 Control	PIC
0021h	0000	0000	001x	xx01	R/W	INT 1 Mask	PIC
0022h	0000	0000	0010	0010	R/W	Configuration Address Index	CONF
0023h	0000	0000	0010	0011	R/W	Configuration Data Index	CONF
0040h	0000	0000	010x	0000	R/W	Timer 1 Counter 0—System Clock	TC
0041h	0000	0000	010x	0001	R/W	Timer1 Counter 1—Refresh Request	TC
0042h	0000	0000	010x	0010	R/W	Timer 1 Counter 2—Speaker Tone	TC
0043h	0000	0000	010x	0011	WO	Timer 1 Command Mode	TC
0048h	0000	0000	010x	1000	R/W	Timer 2 Counter 0—Fail-Safe Timer	TC
0049h	0000	0000	010x	1001	R/W	Timer 2 Counter 1—Reserved	TC
004Ah	0000	0000	010x	1010	R/W	Timer 2 Counter 2—CPU Speed Control	TC
004Bh	0000	0000	010x	1011	WO	Timer 2 Command Mode Register	TC
0061h	0000	0000	0110	00x1	R/W	NMI Status and Control	Control

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
0070H[1]	0000	0000	0111	0xx0	WO	NMI Mask	Control
0080h	0000	0000	100x	0000	R/W	DMA Page Register—Reserved	DMA
0081h	0000	0000	100x	0001	R/W	DMA Channel 2 Page	DMA
0082h	0000	0000	1000	0010	R/W	DMA Channel 3 Page	DMA
0083h	0000	0000	100x	0011	R/W	DMA Channel 1 Page	DMA
0084h	0000	0000	100x	0100	R/W	DMA Page Register—Reserved	DMA
0085h	0000	0000	100x	0101	R/W	DMA Page Register—Reserved	DMA
0086h	0000	0000	100x	0110	R/W	DMA Page Register—Reserved	DMA
0087h	0000	0000	100x	0111	R/W	DMA Channel 0 Page	DMA
0088h	0000	0000	100x	1000	R/W	DMA Page Register—Reserved	DMA
0089h	0000	0000	100x	1001	R/W	DMA Channel 6 Page	DMA
008Ah	0000	0000	100x	1010	R/W	DMA Channel 7 Page	DMA
008Bh	0000	0000	100x	1011	R/W	DMA Channel 5 Page	DMA
008Ch	0000	0000	100x	1100	R/W	DMA Page Register—Reserved	DMA
008Dh	0000	0000	100x	1101	R/W	DMA Page Register—Reserved	DMA
008Eh	0000	0000	100x	1110	R/W	DMA Page Register—Reserved	DMA
008Fh	0000	0000	100x	1111	R/W	DMA Refresh Page	DMA
0092h	0000	0000	1001	0010	R/W	System Control Port	Control
00A0h	0000	0000	101x	xx00	R/W	INT 2 Control	PIC
00A1h	0000	0000	101x	xx01	R/W	INT 2 Mask	PIC
00B2h	0000	0000	1011	0010	R/W	Advanced Power Management Control	APM
00B3h	0000	0000	1011	0011	R/W	Advanced Power Management Status	APM
00C0h	0000	0000	1100	000x	R/W	DMA2 CH0 Base and Current Address	DMA
00C2h	0000	0000	1100	001x	R/W	DMA2 CH0 Base and Current Count	DMA
00C4h	0000	0000	1100	010x	R/W	DMA2 CH1 Base and Current Address	DMA
00C6h	0000	0000	1100	011x	R/W	DMA2 CH1 Base and Current Count	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
00C8H	0000	0000	1100	100x	R/W	DMA2 CH2 Base and Current Address	DMA
00CAh	0000	0000	1100	101x	R/W	DMA2 CH2 Base and Current Count	DMA
00CCh	0000	0000	1100	110x	R/W	DMA2 CH3 Base and Current Address	DMA
00CEh	0000	0000	1100	111x	R/W	DMA2 CH3 Base and Current Count	DMA
00D0h	0000	0000	1101	000x	R/W	DMA2 Status(r) Command(w) Register	DMA
00D2h	0000	0000	1101	001x	WO	DMA2 Write Request	DMA
00D4h	0000	0000	1101	010x	WO	DMA2 Write Single Mask Bit	DMA
00D6h	0000	0000	1101	011x	WO	DMA2 Write Mode	DMA
00D8h	0000	0000	1101	100x	WO	DMA2 Clear Byte Pointer	DMA
00DAh	0000	0000	1101	101x	WO	DMA2 Master Clear	DMA
00DCh	0000	0000	1101	110x	WO	DMA2 Clear Mask	DMA
00DEh	0000	0000	1101	111x	R/W	DMA2 Read/Write All Mask Register Bits	DMA
00F0h	0000	0000	1111	0000	WO	Reset IRQ13	IRQ13
0372h[2]	0000	0011	0111	0010	WO	Secondary Floppy Disk Digital Output	FDCCS#
03F2h[2]	0000	0011	1111	0001	WO	Primary Floppy Disk Digital Output	FDCCS#
0400h	0000	0100	0000	0000	R/W	Reserved	DMA
0401h	0000	0100	0000	0001	R/W	DMA1 CH0 Base/Current Count	DMA
0402h	0000	0100	0000	0010	R/W	Reserved	DMA
0403h	0000	0100	0000	0011	R/W	DMA1 CH1 Base/Current Count	DMA
0404h	0000	0100	0000	0100	R/W	Reserved	DMA
0405h	0000	0100	0000	0101	R/W	DMA1 CH2 Base/Current Count	DMA
0406h	0000	0100	0000	0110	R/W	Reserved	DMA
0407h	0000	0100	0000	0111	R/W	DMA1 CH3 Base/Current Count	DMA
0408h	0000	0100	0000	1000	R/W	Reserved	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
0409H	0000	0100	0000	1001	R/W	Reserved	DMA
040Ah	0000	0100	0000	1010	R/W	DMA Chaining Mode Status/Interrupt Pending	DMA
040Bh	0000	0100	0000	1011	WO	DMA1 Extended Mode	DMA
040Ch	0000	0100	0000	1100	WO	Chaining Buffer Control	DMA
040Dh	0000	0100	0000	1101	R/W	Reserved	DMA
040Eh	0000	0100	0000	1110	R/W	Reserved	DMA
040Fh	0000	0100	0000	1111	R/W	Reserved	DMA
0410h	0000	0100	0010	0000	WO	DMA CH0 S-G Command	DMA
0411h	0000	0100	0010	0001	WO	DMA CH1 S-G Command	DMA
0412h	0000	0100	0010	0010	WO	DMA CH2 S-G Command	DMA
0413h	0000	0100	0010	0011	WO	DMA CH3 S-G Command	DMA
0415h	0000	0100	0010	0101	WO	DMA CH5 S-G Command	DMA
0416h	0000	0100	0010	0110	WO	DMA CH6 S-G Command	DMA
0417h	0000	0100	0010	0111	WO	DMA CH7 S-G Command	DMA
0418h	0000	0100	0010	1000	WO	DMA CH0 S-G Status	DMA
0419h	0000	0100	0010	1001	WO	DMA CH1 S-G Status	DMA
041Ah	0000	0100	0010	1010	WO	DMA CH2 S-G Status	DMA
041Bh	0000	0100	0010	1011	WO	DMA CH3 S-G Status	DMA
041Dh	0000	0100	0010	1101	WO	DMA CH5 S-G Status	DMA
041Eh	0000	0100	0010	1110	WO	DMA CH6 S-G Status	DMA
041Fh	0000	0100	0010	1111	WO	DMA CH7 S-G Status	DMA
0420h	0000	0100	0010	0000	RO	DMA CH0 S-G Descriptor Pointer	DMA
0421h	0000	0100	0010	0001	RO	DMA CH0 S-G Descriptor Pointer	DMA
0422h	0000	0100	0010	0010	RO	DMA CH0 S-G Descriptor Pointer	DMA
0423h	0000	0100	0010	0011	RO	DMA CH0 S-G Descriptor Pointer	DMA
0424h	0000	0100	0010	0100	RO	DMA CH1 S-G Descriptor Pointer	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
0425H	0000	0100	0010	0101	RO	DMA CH1 S-G Descriptor Pointer	DMA
0426h	0000	0100	0010	0110	RO	DMA CH1 S-G Descriptor Pointer	DMA
0427h	0000	0100	0010	0111	RO	DMA CH1 S-G Descriptor Pointer	DMA
0428h	0000	0100	0010	1000	RO	DMA CH2 S-G Descriptor Pointer	DMA
0429h	0000	0100	0010	1001	RO	DMA CH2 S-G Descriptor Pointer	DMA
042Ah	0000	0100	0010	1010	RO	DMA CH2 S-G Descriptor Pointer	DMA
042Bh	0000	0100	0010	1011	RO	DMA CH2 S-G Descriptor Pointer	DMA
042Ch	0000	0100	0010	1100	RO	DMA CH3 S-G Descriptor Pointer	DMA
042Dh	0000	0100	0010	1101	RO	DMA CH3 S-G Descriptor Pointer	DMA
042Eh	0000	0100	0010	1110	RO	DMA CH3 S-G Descriptor Pointer	DMA
042Fh	0000	0100	0010	1111	RO	DMA CH3 S-G Descriptor Pointer	DMA
0434h	0000	0100	0011	0100	RO	DMA CH5 S-G Descriptor Pointer	DMA
0435h	0000	0100	0011	0101	RO	DMA CH5 S-G Descriptor Pointer	DMA
0436h	0000	0100	0011	0110	RO	DMA CH5 S-G Descriptor Pointer	DMA
0437h	0000	0100	0011	0111	RO	DMA CH5 S-G Descriptor Pointer	DMA
0438h	0000	0100	0011	1000	RO	DMA CH6 S-G Descriptor Pointer	DMA
0439h	0000	0100	0011	1001	RO	DMA CH6 S-G Descriptor Pointer	DMA
043Ah	0000	0100	0011	1010	RO	DMA CH6 S-G Descriptor Pointer	DMA
043Bh	0000	0100	0011	1011	RO	DMA CH6 S-G Descriptor Pointer	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
043CH	0000	0100	0011	1100	RO	DMA CH7 S-G Descriptor Pointer	DMA
043Dh	0000	0100	0011	1101	RO	DMA CH7 S-G Descriptor Pointer	DMA
043Eh	0000	0100	0011	1110	RO	DMA CH7 S-G Descriptor Pointer	DMA
043Fh	0000	0100	0011	1111	RO	DMA CH7 S-G Descriptor Pointer	DMA
0461h	0000	0100	0110	0001	R/W	Extended NMI and Reset Control	Control
0462h	0000	0100	0110	0010	R/W	NMI I/O Interrupt Port	Control
0464h	0000	0100	0110	0100	RO	Last EISA Bus Master Granted (L)	Control
0480h	0000	0100	1000	0000	R/W	Reserved	DMA
0481h	0000	0100	1000	0001	R/W	DMA CH2 High Page	DMA
0482h	0000	0100	1000	0010	R/W	DMA CH3 High Page	DMA
0483h	0000	0100	1000	0011	R/W	DMA CH1 High Page	DMA
0484h	0000	0100	1000	0100	R/W	Reserved	DMA
0485h	0000	0100	1000	0101	R/W	Reserved	DMA
0486h	0000	0100	1000	0110	R/W	Reserved	DMA
0487h	0000	0100	1000	0111	R/W	DMA CH0 High Page	DMA
0488h	0000	0100	1000	1000	R/W	Reserved	DMA
0489h	0000	0100	1000	1001	R/W	DMA CH6 High Page	DMA
048Ah	0000	0100	1000	1010	R/W	DMA CH7 High Page	DMA
048Bh	0000	0100	1000	1011	R/W	DMA CH5 High Page	DMA
048Ch	0000	0100	1000	1110	R/W	Reserved	DMA
048Dh	0000	0100	1000	1101	R/W	Reserved	DMA
048Eh	0000	0100	1000	1110	R/W	Reserved	DMA
048Fh	0000	0100	100x	1111	R/W	DMA Refresh High Page	DMA
04C2h	0000	0100	1100	0010	R/W	Reserved	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
04C6H	0000	0100	1100	0110	R/W	DMA CH5 High Base & Current Count	DMA
04CAh	0000	0100	1100	1010	R/W	DMA CH6 High Base & Current Count	DMA
04CEh	0000	0100	1100	1110	R/W	DMA CH7 High Base & Current Count	DMA
04D0h	0000	0100	1101	0000	R/W	INT-1 Edge/Level Control	PIC
04D1h	0000	0100	1101	0001	R/W	INT-2 Edge/Level Control	PIC
04D2h	0000	0100	1101	0010	R/W	Reserved	DMA
04D3h	0000	0100	1101	0011	R/W	Reserved	DMA
04D4h	0000	0100	1101	0100	R/W	DMA2 Chaining Mode	DMA
04D5h	0000	0100	1101	1001	R/W	Reserved	DMA
04D6h	0000	0100	1101	0010	WO	DMA2 Extended Mode	DMA
04D7h	0000	0100	1101	0111	R/W	Reserved	DMA
04D8h	0000	0100	1101	1000	R/W	Reserved	DMA
04D9h	0000	0100	1101	1001	R/W	Reserved	DMA
04DAh	0000	0100	1101	1010	R/W	Reserved	DMA
04DBh	0000	0100	1101	1011	R/W	Reserved	DMA
04DCh	0000	0100	1101	1100	R/W	Reserved	DMA
04DDh	0000	0100	1101	1101	R/W	Reserved	DMA
04DEh	0000	0100	1101	1110	R/W	Reserved	DMA
04DFh	0000	0100	1101	1111	R/W	Reserved	DMA
04E0h	0000	0100	1110	0000	R/W	DMA CH0 Stop Register Bits[7:2]	DMA
04E1h	0000	0100	1110	0001	R/W	DMA CH0 Stop Register Bits[15:8]	DMA
04E2h	0000	0100	1110	0010	R/W	DMA CH0 Stop Register Bits[23:16]	DMA
04E3h	0000	0100	1110	0011	R/W	Reserved	DNA
04E4h	0000	0100	1110	0100	R/W	DMA CH1 Stop Register Bits[7:2]	DMA
04E5h	0000	0100	1110	0101	R/W	DMA CH1 Stop Register Bits[15:8]	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
04E6H	0000	0100	1110	0110	R/W	DMA CH1 Stop Register Bits[23:16]	DMA
04E7h	0000	0100	1110	0111	R/W	Reserved	DMA
04E8h	0000	0100	1110	1000	R/W	DMA CH2 Stop Register Bits[7:2]	DMA
04E9h	0000	0100	1110	1001	R/W	DMA CH2 Stop Register Bits[15:8]	DMA
04EAh	0000	0100	1110	1010	R/W	DMA CH2 Stop Register Bits[23:16]	DMA
04EBh	0000	0100	1110	1011	R/W	Reserved	DMA
04EC	0000	0100	1110	1100	R/W	DMA CH3 Stop Register Bits[7:2]	DMA
04EDh	0000	0100	1110	1101	R/W	DMA CH3 Stop Register Bits[15:8]	DMA
04EEh	0000	0100	1110	1110	R/W	DMA CH3 Stop Register Bits[23:16]	DMA
04EFh	0000	0100	1110	1111	R/W	Reserved	DMA
04F0h	0000	0100	1111	0000	R/W	Reserved	DMA
04F1h	0000	0100	1111	0001	R/W	Reserved	DMA
04F2h	0000	0100	1111	0010	R/W	Reserved	DMA
04F3h	0000	0100	1111	0011	R/W	Reserved	DMA
04F4h	0000	0100	1111	0100	R/W	DMA CH5 Stop Register Bits[7:2]	DMA
04F5h	0000	0100	1111	0101	R/W	DMA CH5 Stop Register Bits[15:8]	DMA
04F6h	0000	0100	1111	0110	R/W	DMA CH5 Stop Register Bits[23:16]	DMA
04F7h	0000	0100	1111	0111	R/W	Reserved	DMA
04F8h	0000	0100	1111	1000	R/W	DMA CH6 Stop Register Bits[7:2]	DMA
04F9h	0000	0100	1111	1001	R/W	DMA CH6 Stop Register Bits[15:8]	DMA
04FAh	0000	0100	1111	1010	R/W	DMA CH6 Stop Register Bits[23:16]	DMA

Table 4. ESC I/O Register Address Map (Continued)

Address (Hex)	Address (Bit)				Type	Name	Block
	FEDC	BA98	7654	3210			
04FBH	0000	0100	1111	1011	R/W	Reserved	DMA
04FC	0000	0100	1111	1100	R/W	DMA CH7 Stop Register Bits[7:2]	DMA
04FDh	0000	0100	1111	1101	R/W	DMA CH7 Stop Register Bits[15:8]	DMA
04FEh	0000	0100	1111	0111	R/W	DMA CH7 Stop Register Bits[23:16]	DMA
04FFh	0000	0100	1111	1111	R/W	Reserved	DMA
0C00h	0000	1100	0000	0000	R/W	Configuration RAM Page Register	Conf
0C80h	0000	1100	100	0000	RO	System Board ID Byte Lane 1 Bits[7:0]	Board ID
0C81h	0000	1100	100	0001	RO	System Board ID Byte Lane 2 Bits[15:8]	Board ID
0C82h	0000	1100	100	0010	RO	System Board ID Byte Lane 3 Bits[23:16]	Board ID
0C83h	0000	1100	1000	0011	RO	System Board ID Byte Lane 4 Bits[31:24]	Board ID

NOTES:
1. Port 70h resides in the ESC in addition the lower 7 bits of Port 70h reside in Real Time Clock also.
2. Bit 3 of ports 372h and 3F2h reside in the ESC while the other bits reside on the ISA bus.

4.3 Configuration Addresses

ESC configuration registers are accessed through I/O registers 22h and 23h. These I/O registers are used as index address register (22h) and index data register (23h). The index address register is used to write the configuration register address. The data (configuration register address) in register 22h is used to decode a configuration register. The selected configuration register can be read or written to by performing a read or a write operation to the index data register at I/O address 23h.

Table 5. Configuration Register Index Address

Configuration Offset	Abbreviation	Register Name
00–01h	—	Reserved
02h	ESCID	ESC ID
03–07h	—	Reserved
08h	RID	Revision ID
09–3Fh	—	Reserved

Table 5. Configuration Register Index Address (Continued)

Configuration Offset	Abbreviation	Register Name
40h	MS	Mode Select
41h	—	Reserved
42h	BIOSCSA	BIOS Chip Select A
43h	BIOSCSB	BIOS Chip Select B
44–4Ch	—	Reserved
4Dh	CLKDIV	BCLK Clock Divisor
4Eh	PCSA	Peripheral Chip Select A
4Fh	PCSB	Peripheral Chip Select B
50h	EISAID1	EISA ID Byte 1
51h	EISAID2	EISA ID Byte 2
52h	EISAID3	EISA ID Byte 3
53h	EISAID4	EISA ID Byte 4
54–56h	—	Reserved
57h	SGRBA	Scatter-Gather Relocate Base Address
58h	—	Reserved
59h	APICBA	APIC Base Address Relocation
60h	PIRQRC0	PIRQ0# Route Control
61h	PIRQRC1	PIRQ1# Route Control
62h	PIRQRC2	PIRQ2# Route Control
63h	PIRQRC3	PIRQ3# Route Control
64h	GPCSLA0	General Purpose Chip Select 0 Base Low Address
65h	GPCSHA0	General Purpose Chip Select 0 Base High Address
66h	GPCSM0	General Purpose Chip Select 0 Mask
67h	—	Reserved
68h	GPCSLA1	General Purpose Chip Select 1 Base Low Address
69h	GPCSHA1	General Purpose Chip Select 1 Base High Address
6Ah	GPCSM1	General Purpose Chip Select 1 Mask
6Bh	—	Reserved
6Ch	GPCSLA2	General Purpose Chip Select 2 Base Low Address
6Dh	GPCSHA2	General Purpose Chip Select 2 Base High Address

Table 5. Configuration Register Index Address (Continued)

Configuration Offset	Abbreviation	Register Name
6Eh	GPCSM2	General Purpose Chip Select 2 Mask
6Fh	GPXBC	General Purpose Peripheral X-Bus Control
70h	PACC	PIC/APIC Configuration Control
71-87h	—	Reserved
88h	TSTC	Test Control
89-9Fh	—	Reserved
A0h	SMICNTL	SMI Control
A2-A3h	SMIEN	SMI Enable
A4-A7h	SEE	System Event Enable
A8h	FTMR	Fast Off Timer
A9h	—	Reserved
AA-ABh	SMIREQ	SMI Request
ACh	CTLTMRL	Clock Scaling STPCLK# Low Timer
ADh	—	Reserved
AEh	CTLTMRH	Clock Scaling STPCLK# High Timer
AF-FFh	—	Reserved

4.4 X-Bus Peripherals

The ESC generates chip selects for certain functions that typically reside on the X-Bus. The ESC asserts the chip selects combinatorially from the LA addresses. The ESC generates chip select signals for the Keyboard Controller, Floppy Disk Controller, IDE, Parallel Port, Serial Port, and General Purpose peripherals. The ESC also generates read and write strobes for Real Time Clock and Configuration RAM. The read and write strobes are a function of LA addresses, the ISA read and write strobes (IORC# and IOWC#), and BCLK. All of the peripherals supported by the ESC are at fixed I/O addresses with the exception of the general purpose peripherals. The ESC support for these peripherals can be enabled or disabled through configuration registers Peripheral Chip Select A and Peripheral Chip Select B. The general purpose peripherals are mapped to I/O addresses by programming a set of configuration registers: General Purpose Chip Select x Base Low Address register, General Purpose Chip Select x Base High Address register, and General Purpose Chip Select x Mask register.

Table 6. X-Bus Chip Selects Decode

Address (Hex)	Address (Bit)				R/W	Name	Chip Select
	FEDC	BA98	7654	3210			
0060h	0000	0000	0110	00x0	R/W	Keyboard Controller	KYBDCS#
0064h	0000	0000	0110	01x0	R/W	Keyboard Controller	KYBDCS#
0070h	0000	0000	0111	0xx0	W	Real Time Clock	RTCALE
0071h	0000	0000	0111	0xx1	R/W	Real Time Clock	RTCWR#/ RTCRD#
0170h–0177h	0000	0001	0111	0xxx	R/W	IDE Controller 0-Secondary	ECS[2:0] = 011 (IDECS0#)
01F0h–01F7h	0000	0001	1111	0xxx	R/W	IDE Controller 0-Primary	ECS[2:0] = 011 (IDECS0#)
0278h–027Bh	0000	0010	0111	1000 to 1011	R/W	Parallel Port LPT3	ECS[2:0] = 010 (LPTCS#)
02F8h–02FFh	0000	0010	1111	xxxx	R/W	Serial Port COM2	ECS[2:0] = 00x (COMxCS#)
0370h–0375h	0000	0011	0111	0000 to 0101	R/W	Floppy Disk Controller-Secondary	FDCCS#
0376h	0000	0011	0111	0111	R/W	IDE Controller 1 Secondary	ECS[2:0] = 100 (IDECS1#)
0377h	0000	0011	0111	0110	R/W	IDE Controller 1 Secondary	ECS[2:0] = 100 (IDECS1#)
0377h	0000	0011	0111	0111	R/W	Floppy Disk Controller-Secondary	FDCCS#
0378h–037Bh	0000	0011	0111	1000 to 1011	R/W	Parallel Port LPT2	ECS[2:0] = 010 (LPTCS#)
03BCh–03BFh	0000	0011	1011	11xx	R/W	Parallel Port LPT 1	ECS[2:0] = 010 (LPTCS#)
03F0h–0375h	0000	0011	1111	0000 to 0101	R/W	Floppy Disk Controller-Primary	FDCCS#
03F6h	0000	0011	0111	0110	R/W	IDE Controller 1-Primary	ECS[2:0] = 100 (IDECS1#)

Table 6. X-Bus Chip Selects Decode (Continued)

Address (Hex)	Address (Bit)				R/W	Name	Chip Select
	FEDC	BA98	7654	3210			
03F7h	0000	0011	0111	0111	R/W	IDE Controller 1-Primary	ECS[2:0] = 100 (IDECS1#)
03F7h	0000	0011	0111	0111	R/W	Floppy Disk Controller-Primary	FDCCS#
03F8h–03FFh	0000	0011	1111	1000	R/W	Serial Port COM 1	ECS[2:0] = 00x (COMxCS#)
0800h–08FFh	0000	1000	xxxx	xxxx	W/R	Configuration RAM	CRAMWR#/ CRAMRD#

2

4.5 I/O APIC Registers

The APIC's registers are indirectly address through two 32 bit registers located in the CPU's memory space—the I/O Register Select (IOREGSEL) and I/O Window (IOWIN) Registers (Table 7). These registers can be relocated via the APIC Base Address Relocation Register and are aligned on 128 bit boundaries.

To access an I/O APIC register, the IOREGSEL Register is written with the address of the intended APIC register. Bits[7:0] of the IOREGSEL Register provide the address offset (Table 8). The IOWIN Register then becomes a 32-bit window pointing to the register selected by the IOREGSEL Register. Note that, for each redirection table register, there are two offset addresses (e.g., address offset 10h selects IOREDTBL0 bits[31:0] and 11h selects IOREDTBL0 bits[63:32].

Table 7. Memory Address For Accessing APIC Registers

Memory Address	Mnemonic	Register Name	Access
FEC0 x000h (82374EB) FEC0 xy00h (82374SB)	IOREGSEL	I/O Register Select	R/W
FEC0 x010h (82374EB) FEC0 xy10h (82374SB)	IOWIN	I/O Window	R/W

NOTE:
xy are determined by the x and y (82374SB only) fields in the APIC Base Address Relocation Register. Range for x = 0-Fh and the range for y = 0,4,8,Ch.

Table 8. I/O APIC Registers

Memory Address	Mnemonic	Register Name	Access
00h	IOAPICID	I/O APIC ID	R/W
01h	IOAPICVER	I/O APIC Version	RO
02h	IOAPICARB	I/O APIC Arbitration ID	RO
10-2Fh	IOREDTBL[0:15]	Redirection Table (Entries 0-15) (63 bits each)	R/W

NOTE:
Address Offset is determined by I/O Register Select Bits[7:0]

5.0 EISA CONTROLLER FUNCTIONAL DESCRIPTION

5.1 Overview

The EISA controller in the ESC provides Master/Slave EISA interface function for the ESC internal resources. In addition, the ESC acts as an EISA central resource for the system. As a system central resource, the EISA controller is responsible for generating the translation control signals necessary for bus-to-bus transfers. These translation includes transfer between devices on EISA Bus and ISA Bus and transfers between different size master device and slave device. The EISA controller generates the control signals for EISA Data Swap Buffers integrated in the PCEB. The ESC EISA interface generates cycles for DMA transfers, and refresh. The ESC internal registers are accessed through the EISA slave interface. The ESC is responsible for supporting the following:

Service EISA Master cycles to:

- EISA slaves devices.
- ISA slave devices.
- ESC internal registers.

Service ISA Master cycles to:

- EISA slave devices.
- ISA (mis-matched) slave devices.
- ESC internal registers.

Service DMA cycles :

- From/to DMA slave on the EISA bus to/from memory on the EISA/ISA bus.
- From/to DMA slave on the ISA bus to/from memory on the EISA/ISA bus.
- From/to DMA slave on the EISA/ISA bus to/from memory on the PCI bus.

Service Refresh Cycles

The EISA controller will service the refresh cycle by generating the appropriate address and command signals. These cycles are initiated by either the ESC internal refresh logic or by an external ISA-Bus Master.

Generates Data Swap Buffer Control

The EISA controller generates the control signals for the data bus swap control (assembly/disassembly) and swapping process to support data size mismatches of the devices on the EISA and ISA buses. The actual data steering and swapping is performed by the PCEB.

Generate Wait States

The wait state generator is responsible for generating the wait states based on the sampling of the EXRDY, CHRDY, NOWS# and the default wait states. The default wait state depends on the cycle type.

5.2 Clock Generation

The ESC is generates the EISA Bus clock. The ESC uses a divider circuit to generate the EISA Bus clock. The ESC supports PCI bus frequencies between 25 MHz and 33 MHz. The PCI clock is divided by 3 or 4 by the clock generation logic in the ESC. The EISA Clock Divisor register bits[2:0] select the divide value.

The ESC provides the EISA Bus clock as the BCLKOUT output. Although the ESC is capable of driving 240 pF load on the BCLKOUT pin, it is recommended that this signal be buffered to protect the EISA BCLK signal.

The ESC EISA control logic and EISA interface is synchronous to the BCLK input. A maximum delay of 15 ns is allowed between the BCLKOUT output and the BCLK input for proper device functionality.

Table 9. PCICLK and BCLK Frequency Relationship

PCICLK (MHz)	DIVISOR (Programmable)	BCLK (MHz)
25	3	8.33
30	4*	7.5
33.3	4*	8.33

NOTE:
The ESC wakes up after reset with a default divisor value of 4.

5.2.1 CLOCK STRETCHING

The ESC is capable of stretching EISA Bus clock (BCLKOUT) for PCEB generated EISA cycles. The ESC stretches the EISA Bus clock (BCLKOUT) in order to minimize the synchronization penalty between PCI clock and EISA clock for accesses to EISA Bus by PCI agents. The PCEB initiates an EISA cycle by asserting START# synchronous to PCICLK. The ESC ensures the START# minimum pulse width is met by stretching the EISA Bus clock low time.

The ESC samples START# on every PCICLK when the PCEB has the EISA Bus. After sampling START# asserted, the ESC delays the rising edge of BCLKOUT until the START# has met the 115 ns minimum pulse width specification.

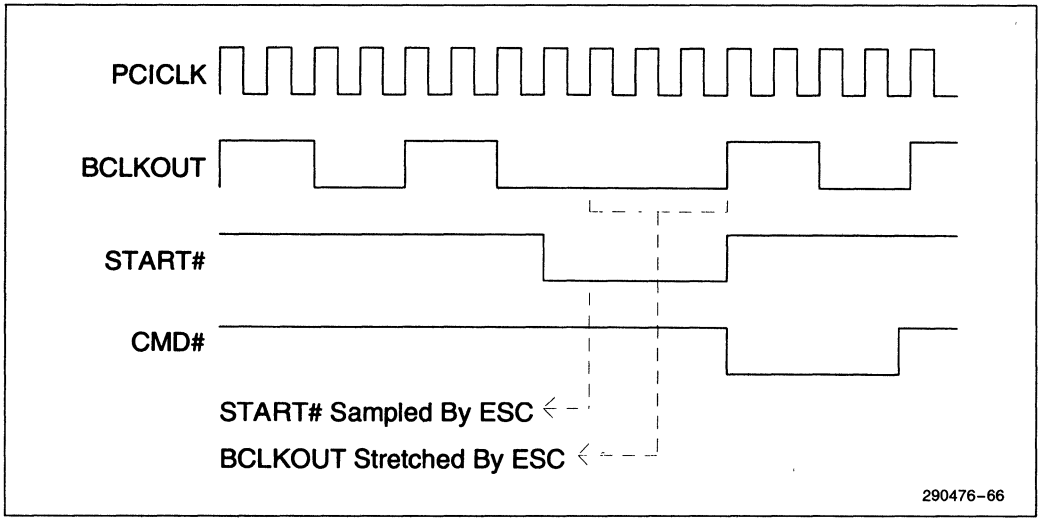


Figure 3. BCLK Stretching

5.3 EISA Master Cycles

EISA Master cycles are initiated on the EISA bus by an EISA Master (including PCEB for PCI agents). These cycles are accesses to the following resources:

- EISA slaves devices (including PCEB for PCI agents)
- ISA slave devices
- ESC internal registers (8-bit EISA Slave)

An EISA master gains control of the bus by asserting MREQx# (PEREQ# in case of PCEB) to the ESC. The ESC, after performing the necessary arbitration, asserts the corresponding MACKn# (negates EISAHOLD in case of the PCEB). Refer to Section 7.0 for arbitration protocol.

In response to receiving the acknowledge signal, the EISA Master starts the cycle by driving the bus with LA[31:02], BE[3:0], W/R, and M/IO. The EISA Master then asserts START# to indicate the beginning of the current cycle. A 16-bit EISA Master will also assert MASTER16# at this time. The ESC generates SBEH#, S1, and S0 signals from the BE[3:0]# signals.

5.3.1 EISA MASTER TO 32-BIT EISA SLAVE

An EISA slave after decoding it's address asserts EX32# or EX16#. The EISA master and the ESC use these signals to determine the EISA slave data size. The 32-bit or 16-bit EISA master continues with the cycles if EX32# or EX16# is asserted respectively. The ESC acts as a central resources for the EISA master and generates CMD# for the cycles. The ESC asserts CMD# on the same BCLK edge that START# is negated. The ESC monitors the EXRDY signal on the EISA bus to determine when to negate the CMD#. An EISA Slave can extend the cycle by negating EXRDY. EISA specification require that EXRDY not be held negated for more than 2.5 μs. A burstable EISA slave asserts SLBURST# signal the same time the slave decodes it's address. The EISA master will sample SLBURST# and assert MSBURST# if it is capable of bursting. The ESC keeps the CMD# asserted during a burst EISA transfer. The ESC deasserts CMD# to indicate the end of the burst transfer after the EISA master deasserts MSBURST#.

If EX16# is asserted, a 32-bit EISA master backs-off the bus by floating BE[3:0]# and START# (see Section 5.3.4). The ESC acts as a central resource for the EISA master in this case and takes over the mastership of the EISA bus by deriving START#, CMD#, and the appropriate byte enables. The ESC generates the necessary translation cycles for the EISA master and returns the bus ownership to the master by asserting EX32# and EX16#. The ESC monitors the EXRDY signal on the EISA bus to determine when to negate the CMD#. An EISA Slave can extend the cycle by negating EXRDY. EISA specification require that EXRDY not be held negated for more than 2.5 μs. A burstable EISA slave will assert the SLBURST# signal the same time when its address is decoded. The EISA master will sample SLBURST# and assert MSBURST# if it is capable of bursting. The ESC keeps the CMD# asserted during a burst EISA transfer. The ESC deasserts CMD# to indicate the end of the burst transfer after the EISA master deasserts MSBURST#.

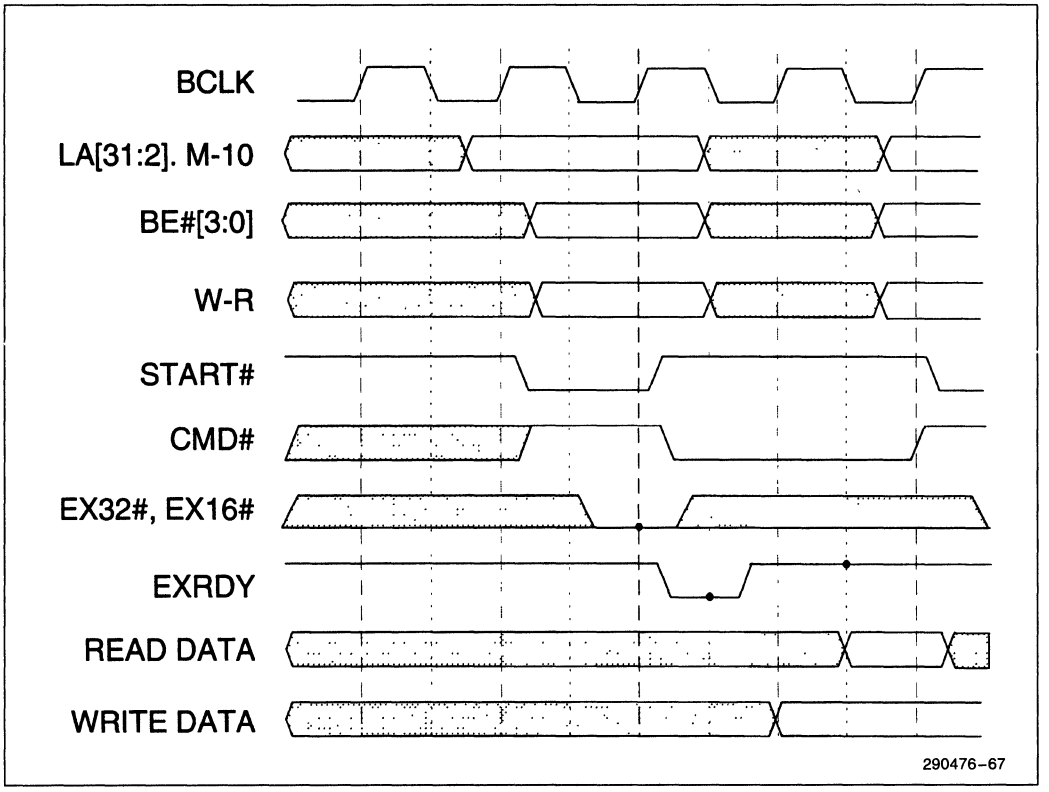


Figure 4. Standard EISA Master to EISA Slave Cycle

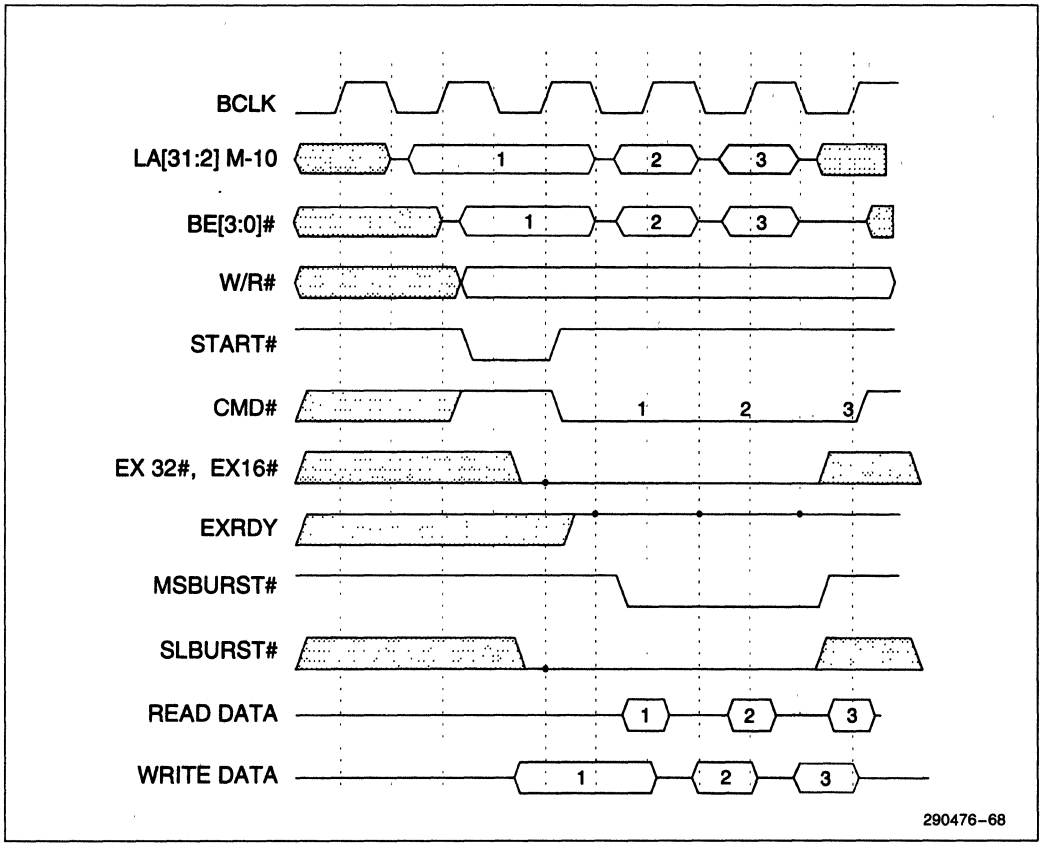


Figure 5. Burst EISA Master to EISA Slave Cycle

5.3.2 EISA MASTER TO 16-BIT ISA SLAVE

An ISA slave, after decoding it's address, asserts M16# or IO16#. The ESC monitors the EX32#, EX16#, M16#, and IO16# signals to determine the slave type. If EX32# and EX16# are negated and M16# or IO16# is asserted, the ESC performs ISA translation cycles for the EISA Bus master by generating BALE, MRDC#, MWRC#, IORC#, IOWC# signals as appropriate. The ISA slave can add wait states by negating CHRDY. The ESC samples CHRDY and translate it into EXRDY.

5.3.3 EISA MASTER TO 8-BIT EISA/ISA SLAVES

An 8-bit slave does not positively acknowledge it's selection by asserting any signal. The absence of an asserted EX32#, EX16#, M16#, and IO16# indicate to the ESC that an 8-bit device has been selected. The EISA master is backed-off the bus, and the ESC takes over mastership of the EISA/ISA bus. The ESC will run 8-bit translation cycles on the bus by deriving the EISA control signals and the ISA control signals. A slave can extend the cycles by negating EXRDY or CHRDY signals.

The ESC (Internal Registers) is accessed as an 8-bit slave.

5.3.4 EISA MASTER BACK-OFF

During EISA master transfer where the master and slave size is mis-matched, the EISA master is required to back-off the bus on the first falling edge of BCLK after START# is negated. The EISA master floats its START#, BE[3:0]#, and data lines at this time. This allows the ESC to performs translation cycle. The master must back-off the bus if a master/slave data size mis-match is determined, regardless if data size translation is performed.

At the end of the data size translation or transfer cycle control is transferred back to the bus master by the ESC by driving EX32# and EX16# active on the falling edge of BCLK, before the rising edge of BCLK that the last CMD# is negated. An additional BCLK is added at the end of the transfer to allow the exchanging of cycle control to occur.

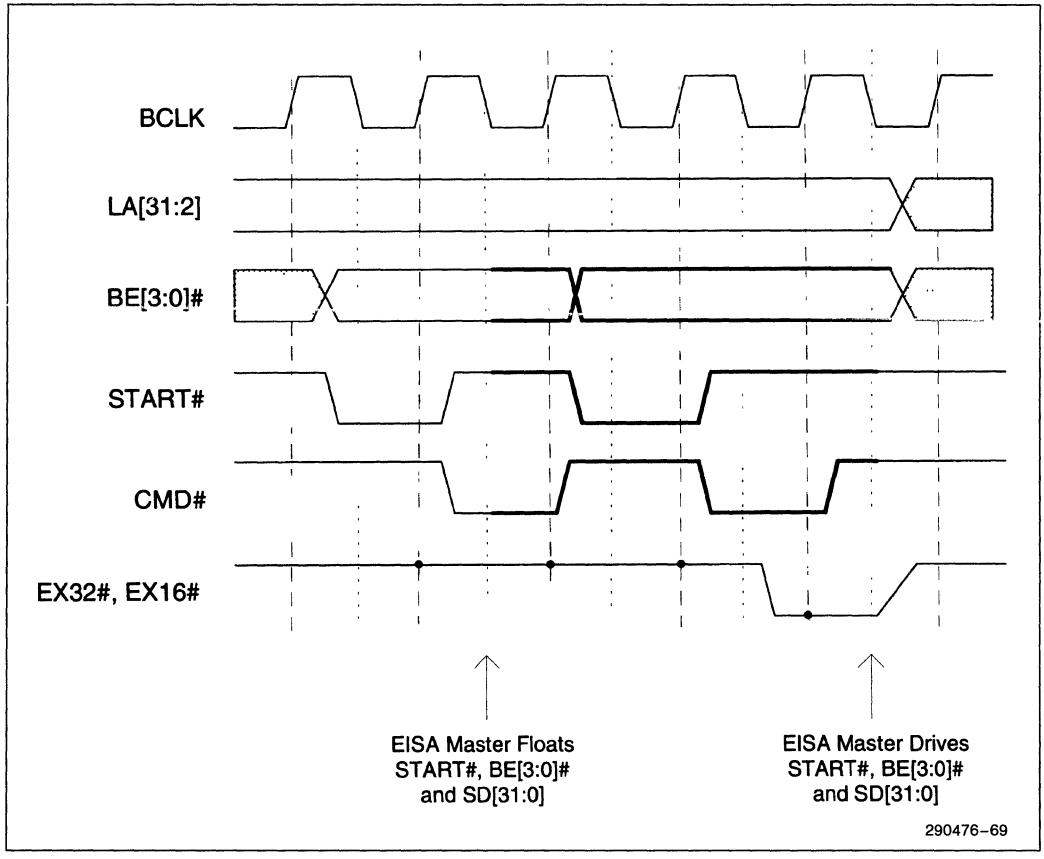


Figure 6. EISA Master Back Off Cycle

5.4 ISA Master Cycles

ISA cycles are initiated on the ISA bus by an ISA master. These cycles are accesses to the following system resources:

- EISA slaves devices (including PCEB for PCI agents).
- ISA slave devices.
- ESC internal registers (8-bit EISA Slave).

The ISA Master initiates such a cycle by asserting the DREQx# line to the ESC. The ESC, after performing the necessary arbitration, asserts the corresponding DACKx# line. Upon receiving an acknowledge from the ESC, the ISA master asserts the MASTER16# signal line to indicate that it has control of the ISA bus and a cycle on the ISA bus will take place. The ESC translates the ISA address signals SBHE#, SA1, and SA0 to EISA byte enables BE[3:0]#.

5.4.1 ISA MASTER TO 32-/16-BIT EISA SLAVE

An EISA slave will decode the address to determine if it has been selected. In response to a positive decode, the EISA slave will assert EX32# or EX16#. The ESC samples these signals to determine if an EISA Slave has been selected. If these signals are asserted, the ESC will perform ISA to EISA cycle translation by driving the EISA control signals.

The ISA Master asserts one of the ISA command signals MRDC#, MWTC#, IORC# or IOWC# depending on whether or not the access is to a memory, an I/O device or an I/O register. The ISA command signals will remain active until the end of the cycle. The ESC will generate the EISA translation by generating the EISA control signals; START#, CMD#, M/IO#, and W/R#.

The EISA slave can add wait states by negating EXRDY. The ESC samples EXRDY and translates it into CHRDY. The ESC will also generate the control signals to steer the data to the appropriate byte lanes for mismatched cycles.

5.4.2 ISA MASTER TO 16-BIT ISA SLAVE

An ISA Master initiates cycles to ISA slave devices. These cycles are either memory read/write or I/O read/write. The ISA bus Master is assumed to be 16-bit device, and it can access either 8- or 16-bit slave devices that reside on the ISA bus. A 16-bit ISA slave device will respond to a valid address by asserting M16# for memory cycles and IO16# for I/O cycles. The ESC is inactive during ISA Master cycles where either M16# or IO16# is sampled asserted.

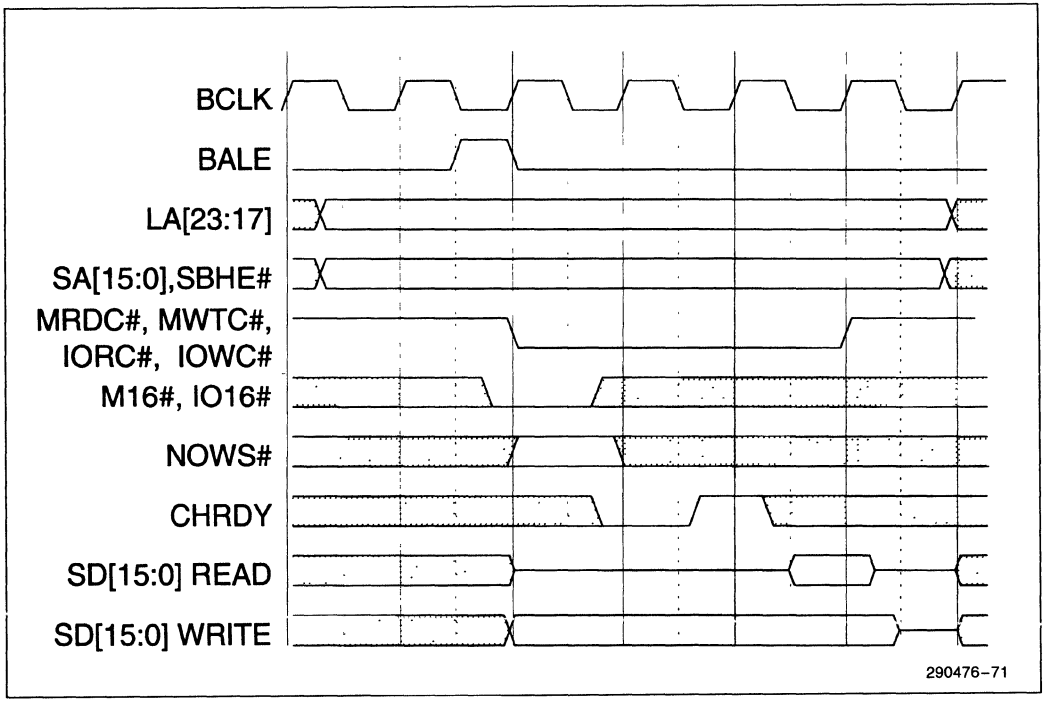


Figure 7. ISA Master to 16-Bit ISA Slave Cycles (3 BCLKs)

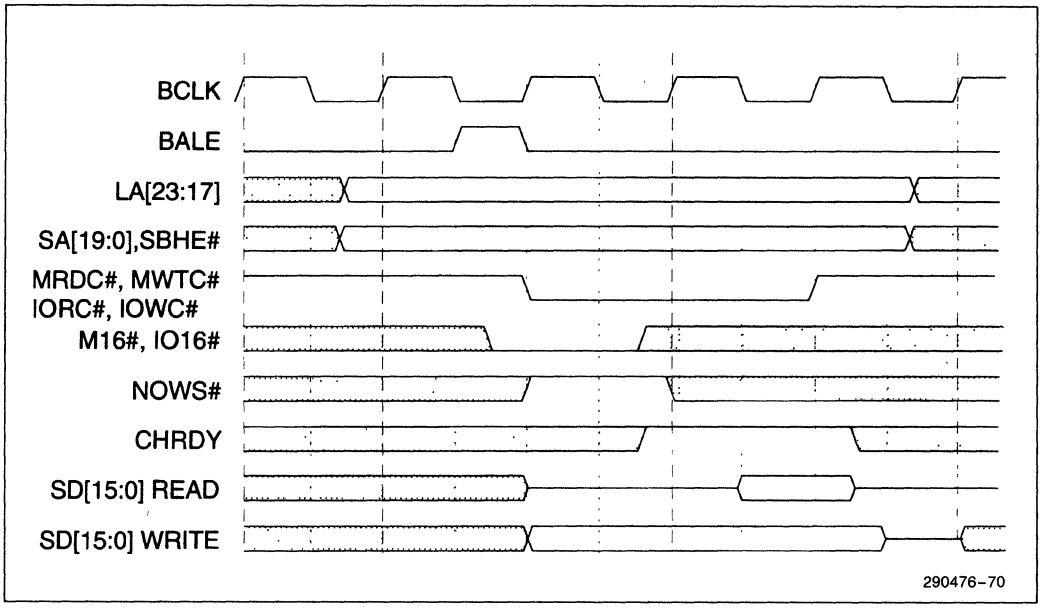


Figure 8. ISA Master to 16-Bit ISA Slave Extended Cycle (4 BCLKs)

5.4.3 ISA MASTER TO 8-BIT EISA/ISA SLAVE

An 8-bit slave does not positively acknowledge it's selection by asserting any signal. The absence of an asserted EX32#, EX16#, M16#, and IO16# indicate to the ESC that an 8-bit device has been selected. The EISA master is backed-off the bus, and the ESC takes over mastership of the EISA/ISA bus. The ESC will run 8-bit translation cycles on the bus by deriving the EISA control signals and the ISA control signals. A slave can extend the cycles by negating EXRDY or CHRDY signals. The ESC (Internal Registers) is accessed as an 8-bit slave.

5.4.4 ISA WAIT STATE GENERATION

There are three sources that can affect the generation of wait states for ISA cycles. The first is the default wait states, which determines the standard or default ISA bus cycle in the absence of any response from the slave. The second is cycle extension, which is indicated by the slave pulling the CHRDY signal line inactive (low). The CHRDY is high by default due to a pull-up resistor. Thus, the cycle will be extended until the CHRDY is returned to its active high value. The third way to change the number of wait states is when the slave asserts the NOWS# signal which makes the cycle shorter than the default or standard cycle.

ISA Memory slaves (8- and 16-bits) and ISA I/O slaves (only 8-bits) can shorten their default cycles by asserting the NOWS# signal lines. A 16-bit I/O slave cannot shorten its default cycles. When NOWS# is asserted at the same time the CHRDY is negated by the ISA slave device, NOWS# will be ignored and wait states will be added. (i.e.; CHRDY has precedence over NOWS#.)

DMA devices (I/O) cannot add wait states, but memory can. Table 10 shows the number of BCLKs for each cycle type (Memory, I/O, DMA), default, wait states added and with NOWS# asserted.

Table 10. Number of BCLKs for ISA Master Cycles

Cycle Type	Bus Size	No Wait State NOWS# = 0	Standard CHRDY = 1 NOWS# = 1	One Wait State CHRDY = 0
		Number of BCLKs		
Memory Read/Write	16	2	3	4
Memory Read/Write	8	4, 5	6	7
I/O Read/Write	16	3	3	4
I/O Read/Write	8	4, 5	6	7
DMA Compatible	8/16	8	8	10
DMA Type A(1)	8/16	NA	6	7
DMA Type B(1)	8/16	NA	4	5
DMA Type C(2)	8/16	NA	2	3

NOTES:
1. If ISA memory responds, the ESC will extend the cycle by 1 BCLK.
2. If ISA memory responds, the ESC will use DMA Type B read cycle timing.

5.5 Mis-Match Cycles

Data size translation is performed by the ESC for all mis-matched cycles. A mis-matched cycle is defined as a cycle in which the bus master and bus slave do not have equal data bus sizes (e.g., a 32-bit EISA master accessing a 16-bit ISA slave). The data size translation is performed in conjunction with the PCEB. The ESC generates the appropriate cycles and data steering control signals for mis-matched cycles. The PCEB uses the data steering control signals from the ESC to latch and redirect the data to the appropriate byte lanes. The ESC will perform one or more of the following operations depending on the master and slave type, transfer direction, and the number of byte enables active.

Table 11. Mis-Match Master Slave Combinations

Master Type	Cycle Type	Slave Type			
		32-Bit EISA	16-Bit EISA	16-Bit ISA	8-Bit EISA/ISA
32-bit EISA with 16-bit downshift	Standard Burst	match match	Mis-Match match	Mis-Match na	Mis-Match na
32-bit EISA	Standard Burst	match match	Mis-Match na	Mis-Match na	Mis-Match na
16-bit EISA	Standard Burst	Mis-Match Mis-Match	match match	Mis-Match na	Mis-Match na

NOTE:
na: Not Applicable. The cycle will never occur.

5.6 Data Swap Buffer Control Logic

For all mis-matched cycles, the ESC is responsible for performing data size translations. The ESC performs these data size translations by either becoming the master of the EISA/ISA Bus (see Section 5.3.4) or by directing the flow of data to the appropriate byte lanes. In both cases, the ESC generates Data Swap Buffer control signals to perform data size translation.

- SDCPYEN[13,3:1]
- SDCPYUP
- SDOE[2:0]#
- SDLE[3:0]#

The Data Swap Buffers are integrated in the PCEB (see PCEB data sheet Section 8.0 for Data Swap Buffer function description). The data size translation cycles consist of one or combinations of Assembly, Disassembly, Copy Up/Down, and Redrive.

ASSEMBLY

This occurs during reads when an EISA master data size is greater than the slave data size. ISA masters are required to perform assemble when accessing 8-bit slaves. Assembly consists of two, three, or four cycles depending on the master data size, slave data size, and number of active byte enables. During the assembly process, the data is latched in to the PCEB data latch/buffers. This data is driven or redriven on to the EISA bus during the last cycle. The master after initiating the cycle backs-off the bus (see the EISA master back-off section for details) when a mis-matched is detected. The ESC becomes the bus master and runs the appropriate number of cycles. At the end of the last cycle, the ESC transfer the control of bus back to the original master.

DISASSEMBLY

This occurs during writes when the EISA master data size is greater than the slave data size. ISA masters are required to perform disassemble when accessing 8-bit slaves. Disassembly consists of two, three, or four cycles depending on the master data size, slave data size, and number of active byte enables. During the disassembly process, the data is latched in the PCEB latch/buffers on the first cycle. This data is driven or redriven on to the EISA bus on subsequent cycles. The master after initiating the cycle backs-off the bus (see the EISA master back-off section for details) when a mis-matched is detected. The ESC becomes the bus master and runs the appropriate number of cycles. At the end of the last cycle, the ESC transfer the control of bus back to the original master.

COPY-UP

This occurs during reads when the master data size is greater than the slave data size and during writes when the master data size is smaller than the slave data size. The copy-up function is used for cycles with and without assembly/disassembly.

COPY-DOWN

This occurs during writes when the master data size is greater than the slave data size and during reads when the master data size is smaller than the slave data size. The copy-down function is used for cycles with and without assembly/disassembly.

RE-DRIVE

This occurs during reads and writes when both the master and slave are on the EISA/ISA bus and the PCEB is neither a master nor a slave. The re-drive function is always performed in conjunction with assembly/disassembly. During the assembly process, the last cycle is a re-drive cycle. During disassembly, all the cycles except the first cycle are re-drive cycles.

5.7 Servicing DMA Cycles

The ESC is responsible for performing DMA transfers. If the memory is determined (EX32# or EX16# asserted) to be on the EISA bus, the DMA cycle can be "A", "B", or "C" type. If the memory is determined to be on the ISA bus, then the DMA cycle will run as a compatible cycle. The DMA transfers are described in detail in Section 8.0.

5.8 Refresh Cycles

The ESC support refresh cycles on the EISA/ISA bus. The ESC asserts the REFRESH# signal to indicate when a refresh cycle is in progress. Refresh cycles are generated by two sources: the refresh unit inside the ESC or an external ISA bus masters. The EISA bus controller will enable the address lines LA[15:2] and the BE[3:0]#. The High and Low Page register contents will also be placed on the LA[31:16] bus during refresh. Memory slaves on the EISA/ISA bus must not drive any data onto the data bus during the refresh cycle. Slow memory slaves on the EISA/ISA may extend the refresh cycle by negating the EXRDY or CHRDY signal respectively. The refresh cycles are also described in Section 6.11.

5.9 EISA Slot Support

The ESC support up of 8 EISA slots. The ESC provides support for the 8 slots as follows:

- The ESC address and data output buffers directly drive 240 pF capacitive load on the Bus.
- The ESC generates slot specific AENx signals.
- The ESC supports EISA masters in all 8 slots.

The ESC generates encoded AENs and encoded Master Acknowledge signals for 8 slots and 8 masters. These signals must to decoded on the system board to generate the slot specific AENx signals and MACKx# signals. The ESC can be programmed through Mode Select register bit[1:0] to directly generate these signals for 4 slots and 4 masters.

5.9.1 AEN GENERATION

The ESC directly generates the slot specific AEN signals if the ESC is configured to support 4 AENx (Table 12). If the ESC is programmed to support more than 4 EISA AENx, the ESC will generate Encoded AEN signals. Discrete logic like a F138 is required to generate the slot specific AENs.

Table 12. AEN Generation

Cycle	A[15:12]	A[11:8]	A[7:4]	A[3:0]	AEN4	AEN3	AEN2	AEN1
DMA	xxxx	xxxx	xxxx	xxxx	1	1	1	1
IO	0000	xx00	xxxx	xxxx	1	1	1	1
IO	0001	xx00	xxxx	xxxx	1	1	1	0
IO	0010	xx00	xxxx	xxxx	1	1	0	1
IO	0011	xx00	xxxx	xxxx	1	0	1	1
IO	0100	xx00	xxxx	xxxx	0	1	1	1
IO	0101-1111	xx00	xxxx	xxxx	1	1	1	1
IO	xxxx	xx01	xxxx	xxxx	0	0	0	0
IO	xxxx	xx10	xxxx	xxxx	0	0	0	0
IO	xxxx	xx11	xxxx	xxxx	0	0	0	0
MEM	xxxx	xxxx	xxxx	xxxx	0	0	0	0

Table 13. Encoded AEN (AEN) Generation

Cycle	A[15:12]	A[11:8]	A[7:4]	A[3:0]	EAEN4	EAEN3	EAEN2	EAEN1
DMA	xxxx	xxxx	xxxx	xxxx	1	1	1	1
IO	0000	xx00	xxxx	xxxx	1	1	1	1
IO	0001	xx00	xxxx	xxxx	0	0	0	1
IO	0010	xx00	xxxx	xxxx	0	0	1	0
IO	0011	xx00	xxxx	xxxx	0	0	1	1
IO	0100	xx00	xxxx	xxxx	0	1	0	0
IO	0101	xx00	xxxx	xxxx	0	1	0	1
IO	0110	xx00	xxxx	xxxx	0	1	1	0
IO	0111	xx00	xxxx	xxxx	0	1	1	1
IO	1000	xx00	xxxx	xxxx	1	0	0	0
IO	1001-1111	xx00	xxxx	xxxx	1	1	1	1
IO	xxxx	xx01	xxxx	xxxx	0	0	0	0
IO	xxxx	xx10	xxxx	xxxx	0	0	0	0
IO	xxxx	xx11	xxxx	xxxx	0	0	0	0
MEM	xxxx	xxxx	xxxx	xxxx	0	0	0	0

NOTE:
EAEN[4:1] combinations not specified in the table are Reserved.

5.9.2 MACKX# GENERATION

The ESC generates the EISA Master Acknowledge signals if the ESC is configured for to directly support 4 masters through Mode Select register bit[1:0]. In this case the ESC generates MACKx#s for Master 0-3. If the ESC is programmed to support more than 4 EISA slots, the ESC will generate Encoded (E)MACKx#s. Discrete logic like a F138 is required to generate the MACKx#s for the Masters.

Table 14. Encoded MACK# (EMACKx#) Generation

EMACK [4:1]	MACK7#	MACK6#	MACK5#	MACK4#	MACK3#	MACK2#	MACK1#	MACK0#
0000	1	1	1	1	1	1	1	0
0001	1	1	1	1	1	1	0	1
0010	1	1	1	1	1	0	1	1
0011	1	1	1	1	0	1	1	1
0100	1	1	1	0	1	1	1	1
0101	1	1	0	1	1	1	1	1
0110	1	0	1	1	1	1	1	1
0111	0	1	1	1	1	1	1	1
1111	1	1	1	1	1	1	1	1

NOTE:
EMACK[4:1] combinations 1000–1110 are Reserved.

2

6.0 DMA CONTROLLER

6.1 DMA Controller Overview

The DMA circuitry incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels, (Channels 0-3 and Channels 5-7). DMA Channel 4 is used to cascade the two controllers together and will default to cascade mode in the Mode register. In addition to accepting requests from DMA slaves, the DMA also responds to requests that are initiated by software. Software may initiate a DMA service request by setting any DMA Channel Request register bit to a 1. The DMA controller for Channels 0-3 is referred to as "DMA-1" and the controller for Channels 4-7 is "DMA-2".

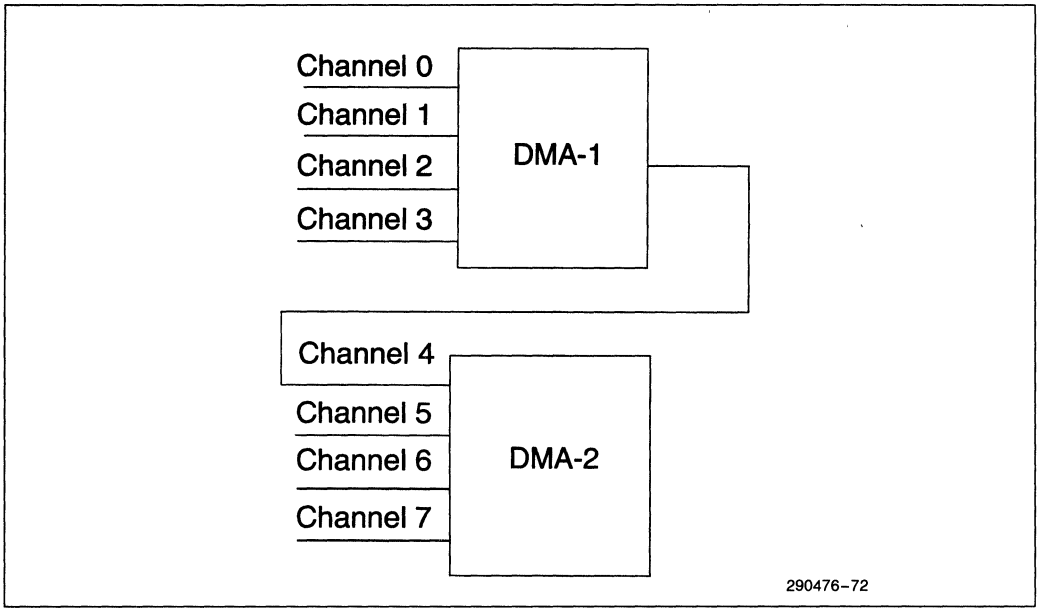


Figure 9. Internal DMA Controller

Each DMA channel can be programmed for 8- or 16-bit DMA device size. Each channel can also be programmed for compatibility, Type "A", Type "B", or Type "C"(burst transfer) timings. Each DMA channel defaults to the PC-AT compatible settings for DMA device size: channels [3:0] default to 8-bit, count-by-bytes transfers, while channels [7:5] default to 16-bit, count-by-words (address shifted) transfers. The ESC provides the timing control and data size translation necessary for DMA transfers between EISA/ISA agents of mismatched bus sizes.

The DMA Controller supports full 32-bit addressing. Each channel includes a 16-bit ISA compatible Current register which holds the 16 least-significant bits of the 32-bit address, and an ISA compatible Low Page register which contains the eight second most significant bits. An additional High Page register contains the eight most significant bits of the 32-bit address. The address counter can be programmed as either 16-bit compatible address counter or a full 32-bit address counter.

The channels can also be programmed for any of four transfer modes. The transfer modes include single, block, demand, or cascade. Each of the three active transfer modes (single, block, and demand), can perform three different types of transfers (read, write, or verify).

The DMA Controller also features refresh address generation, and auto-initialization following a DMA termination. EISA compatible buffer chaining is included as well as Stop registers to support ring buffer structures.

Scatter-Gather reduces CPU overhead by eliminating reprogramming of the DMA and I/O between buffers as well as reducing the number of interrupts.

The DMA Controller includes the EISA Bus arbiter which works with the PCEB's PCI bus arbiter. The arbiter determines which requester from among the requesting DMA slaves, EISA bus masters, the PCI bus, or Refresh should have the bus.

The DMA Controller is at any time either in master mode or slave mode. In master mode, the DMA controller is either servicing a DMA slave's request for DMA cycles, allowing an ISA master to use the bus via a cascaded DREQ signal, or granting the bus to an EISA master via MREQ#/MACK#. In slave mode, the ESC monitors both the EISA bus decoding and responding to I/O read and write commands that address its registers.

When the DMA is in master mode and servicing a DMA slave, it works in conjunction with the ESC EISA bus controller to create bus cycles on the EISA bus. The DMA places addresses onto the internal address bus and the bus controller informs the DMA when to place a new address on the internal bus.

6.2 DMA Transfer Modes

The channels can be programmed for any of four transfer modes. The transfer modes include single, block, demand, or cascade. Each of the three active transfer modes (single, block, and demand), can perform three different types of transfers (read, write, or verify). The ESC does not support memory to memory transfers.

6.2.1 SINGLE TRANSFER MODE

In Single Transfer mode the DMA is programmed to make one transfer only. The byte/word count will be decremented and the address decremented or incremented following each transfer. When the byte/word count "rolls over" from zero to FFFFFFh, or an external EOP is encountered, a Terminal Count (TC) will load a new buffer via Scatter-Gather, buffer chaining or autoinitialize if it is programmed to do so.

DREQ must be held active until DACK becomes active in order to be recognized. If DREQ is held active throughout the single transfer, the bus will be released to the CPU after a single transfer. With the DREQ asserted high, the DMA I/O device will rearbitrate for the bus. Upon winning the bus, another single transfer will be performed. This allows other bus masters a chance to arbitrate for, win, and execute cycles on the EISA Bus.

6.2.2 BLOCK TRANSFER MODE

In Block Transfer mode the DMA is activated by DREQ to continue making transfers during the service until a TC, caused by either a byte/word count going to FFFFFFh or an external EOP, is encountered. DREQ need only be held active until DACK becomes active. If the channel has been programmed for it, a new buffer will be loaded by buffer chaining or auto-initialization at the end of the service. In this mode, it is possible to lock out other devices for a period of time (including refresh) if the transfer count is programmed to a large number and Compatible timing is selected. Block mode can effectively be used with Type "A", Type "B", or Burst timing since the channel can be interrupted through the 4 μs timeout mechanism, and other devices (or Refresh) can arbitrate for and win the bus. See Section 7.0 on the EISA Bus Arbitration for a detailed description of the 4 μs timeout mechanism. Note that scatter-gather block mode is not supported.

6.2.3 DEMAND TRANSFER MODE

In Demand Transfer mode the DMA channel is programmed to continue making transfers until a TC (Terminal Count) is encountered or an external EOP is encountered, or until the DMA I/O device pulls DREQ inactive. Thus, transfers may continue until the I/O device has exhausted its data capacity. After the I/O device catches up, the DMA service is re-established when the DMA I/O device reasserts the channel's DREQ. During the time between services when the system is allowed to operate, the intermediate values of address and byte/word count are stored in the DMA controller Current Address and Current Byte/Word Count registers. A TC can cause a new buffer to be loaded via Scatter-Gather, buffer chaining or autoinitialize at the end of the service if the channel has been programmed for it.

6.2.4 CASCADE MODE

This mode is used to cascade more than one DMA controller together for simple system DMA requests for the additional device propagate through the priority network circuitry of the preceding device. The priority chain is preserved and the new device must wait for its turn to acknowledge requests. Within the ESC architecture, Channel 0 of DMA Controller two (DMA-2, Ch 4) is used to cascade DMA Controller one (DMA-1) to provide a total of seven DMA channels. Channel 0 on DMA-2 (labeled Ch 4 overall) connects the second half of the DMA system. This channel is not available for any other purpose.

In Cascade Mode, the DMA Controller will respond to DREQ with DACK, but the ESC will not drive the bus.

Cascade mode is also used to allow direct access of the system by 16-bit bus masters. These devices use the DREQ and DACK signals to arbitrate for the system bus and then they drive the address and command lines to control the bus. The ISA master asserts its ISA master request line (DREQx) to the DMA internal arbiter. If the ISA master wins the arbitration, the ESC responds with an ISA Master Acknowledge (DACKx) signal active. Upon sampling the DACKx line active, the ISA Master asserts MASTER16# signal and takes control of the EISA bus. The ISA Master has control of the EISA Bus, and the ISA Master may run cycles until it negates the MASTER16# signal.

6.3 DMA Transfer Types

Each of the three active transfer modes (Single, Block, or Demand) can perform three different types of transfers. These transfers are Read, Write and Verify.

Write Transfer

Write transfers move data from an EISA/ISA I/O device to memory located on EISA/ISA Bus or PCI Bus. The DMA indicates the transfer type to the EISA bus controller. The bus controller will activate IORC# and the appropriate EISA control signals (M/IO# and W/R#) to indicate a memory write.

Read Transfer

Read transfers move data from EISA/ISA or PCI memory to an EISA/ISA I/O device. The DMA indicates the transfer type to the EISA bus controller. The bus controller will activate IOWC# and the appropriate EISA control signals (M/IO# and W/R#) to indicate a memory read.

Verify Transfer

Verify transfers are pseudo transfers. The DMA controller operates as in Read or Write transfers, generating addresses and producing TC, etc. However, the ESC does not assert the memory and I/O control signals. Only the DACK signals are asserted. Internally the DMA controller will count BCLKs so that the DACK signals have a defined pulse width. This pulse width is nine BCLKs long. If Verify transfers are repeated during Block or Demand DMA requests, each additional pseudo transfer will add eight BCLKs. The DACK signals will not be toggled for repeated transfers.

6.4 DMA Timing

The ESC DMA provides four transfer timings. In addition to the compatible timings, the ESC DMA provides Type "A", Type "B", and Type "C" (burst) timings for I/O slave devices capable of running at faster speeds.

6.4.1 COMPATIBLE TIMINGS

Compatible timing is provided for DMA slave devices. Compatible timing runs at 9 BCLKs (1080 ns/single cycle) and 8 BCLKs (960 ns/cycle) during the repeated portion of a Block or Demand mode transfers.

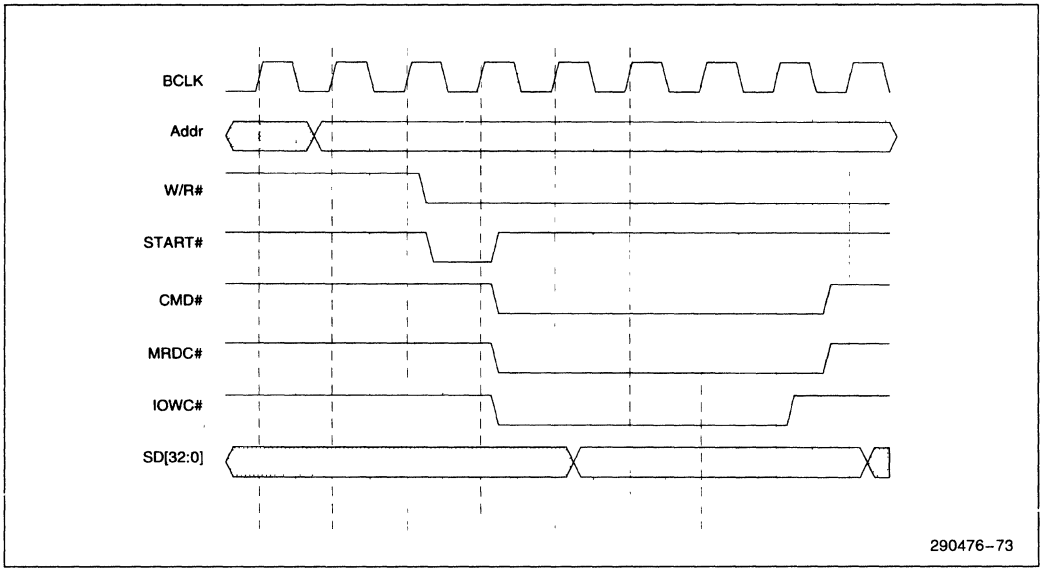


Figure 10. Compatible DMA Read Transfer (8 BCLKs)

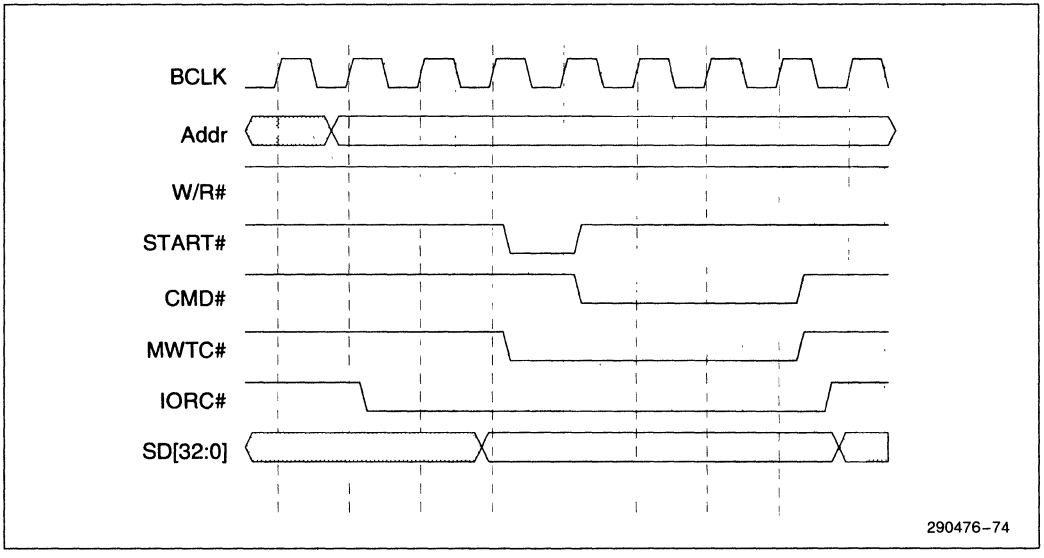


Figure 11. Compatible DMA Write Transfer (8 BCLKS)

6.4.2 TYPE "A" TIMING

Type "A" timing is provided to allow shorter cycles to EISA memory. (Note: Main memory behaves like EISA memory because the PCEB has an EISA slave interface.) Type "A" timing runs at 7 BCLKs (840 ns/single cycle) and 6 BCLKs (720 ns/cycle) during the repeated portion of a Block or Demand mode transfer. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed by system memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter, but it is expected that the DMA devices which provide the data access time or write data setup time should not require excess IORC# or IOWC# command active time. Because of this, most DMA devices should be able to use type "A" timing.

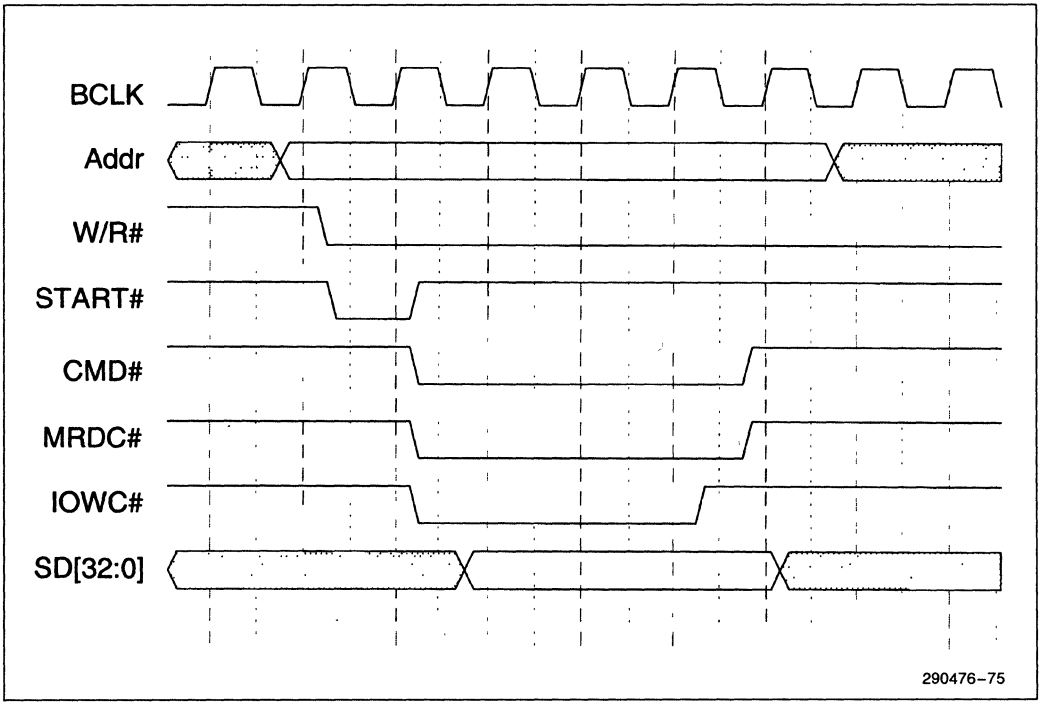


Figure 12. Type "A" DMA Read Transfers (6 BCLKS)

6.4.3 TYPE "B" TIMING

Type "B" timing is provided for 8-/16-bit DMA devices which can accept faster I/O timing. Type "B" only works with fast system memory. Type "B" timing runs at 6 BCLKs (720 ns/single cycle) and 4 BCLKs (480 ns/cycle) during the repeated portion of a block or demand mode transfer. Type "B" timing requires faster DMA slave devices than compatible timing in that the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster. Some of the current ISA devices should be able to support type "B" timing, but these will probably be more recent designs using relatively fast technology.

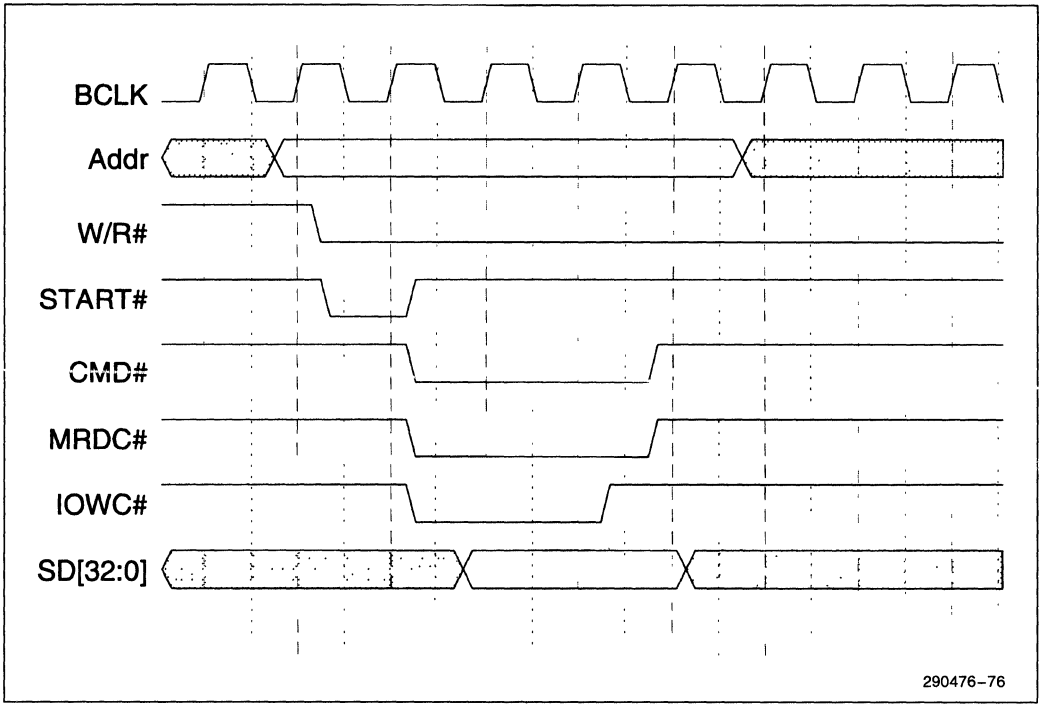


Figure 13. Type "B" DMA Read Transfer (4 BCLKS)

6.4.4 TYPE "C" (BURST) TIMING

Type "C" (burst) timing is provided for EISA DMA devices. The DMA slave device needs to monitor EXRDY and IORC# or IOWC# signals to determine when to change the data (on writes) or sample the data (on reads). This timing will allow up to 33 MBytes per second transfer rate with a 32-bit DMA device and 32-bit memory. Note that 8- or 16-bit DMA devices are supported (through the programmable DMA address increment) and that they use the "byte lanes" natural to their size for the data transfer. As with all bursts, the system will revert to two BCLK cycles if the memory does not support burst. When a DMA burst cycle accesses non-burst memory and the DMA cycle crosses a page boundary into burstable memory, the ESC will continue performing standard (non-burst) cycles. This will not cause a problem since the data is transferred correctly.

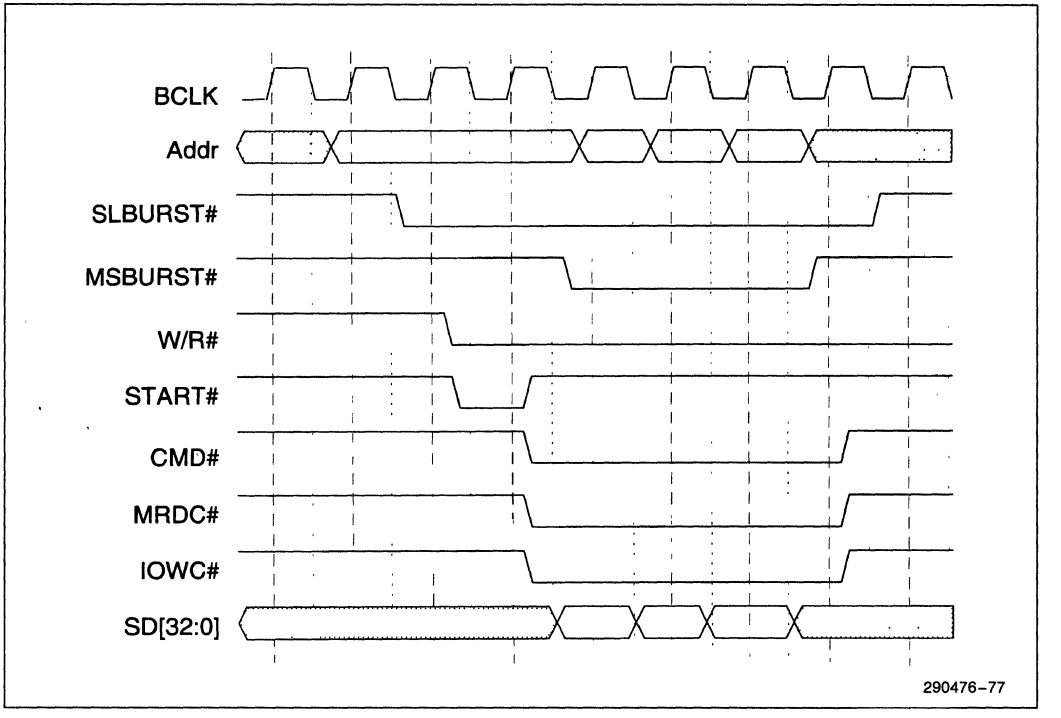


Figure 14. Type "C" (Burst) DMA Read Transfers (1 BCLK)

6.5 Channel Priority

For priority resolution the DMA consists of two logical channel groups-channels 0-3 and channels 4-6. Each group may be in either Fixed or Rotate mode, as determined by the Command register.

For arbitration purposes, the source of the DMA request is transparent. DMA I/O slaves normally assert their DREQ line to arbitrate for DMA service. However, a software request for DMA service can be presented through each channel's DMA Request register. A software request is subject to the same prioritization as any hardware request. Please see the detailed register description in Section 3.0 for Request Register programming information.

Fixed Priority

The initial fixed priority structure is as follows:

High priority	Low priority
(0, 1, 2, 3)	5, 6, 7

The fixed priority ordering is 0, 1, 2, 3, 5, 6, and 7. In this scheme, Channel 0 has the highest priority, and Channel 7 has the lowest priority. Channels [3:0] of DMA-1 assume the priority position of Channel 4 in DMA-2, thus taking priority over Channels 5, 6, and 7.

Rotating Priority

Rotation allows for "fairness" in priority resolution. The priority chain rotates so that the last channel serviced is assigned the lowest priority in the Channel group (0-3, 5-7). Channels 0-3 rotate as a group of 4. They are always placed between Channel 5 and Channel 7 in the priority list. Channel 5-7 rotate as part of a group of 4. That is, Channels (5-7) form the first three partners in the rotation, while Channel group (0-3) comprises the fourth position in the arbitration. Table 15 demonstrates rotation priority:

Table 15. Rotating Priority Example

Programmed Mode	Action	Priority High .....Low
Group (0-3) is in rotation mode	1) Initial Setting	(0, 1, 2, 3), 5, 6, 7
Group (4-7) is in fixed mode.	2) After servicing channel 2	(3, 0, 1, 2), 5, 6, 7
	3) After servicing channel 3	(0, 1, 2, 3), 5, 6, 7
Group (0-3) in rotation mode	1) Initial Setting	(0, 1, 2 3), 5, 6, 7
Group (4-7) is in rotation mode	2) After servicing channel 0	5, 6, 7, (1, 2, 3, 0)
	3) After servicing channel 5	6, 7, (1, 2, 3, 0), 5
(note that the first servicing of	4) After servicing channel 6	7, (1, 2, 3, 0), 5, 6
channel 0 caused double rotation).	5) After servicing channel 7	(1, 2, 3, 0), 5, 6, 7

6.6 Scatter-Gather Functional Description

Scatter-Gather provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In Scatter-Gather, the DMA can read the memory address and word count from an array of buffer descriptors called the Scatter-Gather Descriptor (SGD) Table. This allows the DMA to sustain DMA transfers until all buffers in the Scatter-Gather Descriptor Table are transferred.

The Scatter-Gather Command register and Scatter-Gather Status register are used to control the operational aspect of Scatter-Gather transfers (see Section 3.2 for details of these registers). The Scatter-Gather Descriptor Next Link register holds the address of the next buffer descriptor in the Scatter-Gather Descriptor Table.

The next buffer descriptor is fetched from the Scatter-Gather Descriptor Table by a DMA read transfer. DACK# will not be asserted for this transfer because the I/O device is the DMA itself and the DACK is internal to the ESC. The ESC will assert IOWC# for these bus cycles like any other DMA transfer. The ESC will behave as an 8-bit I/O slave and will run type "B" timings for a Scatter-Gather buffer descriptor transfer. EOP will be asserted at the end of the transfer.

To initiate a typical Scatter-Gather transfer between memory and an I/O device the following steps are involved:

1. Software prepares a Scatter-Gather Descriptor (SGD) table in system memory. Each Scatter-Gather descriptor is 8 bytes long and consists of an address pointer to the starting address and the transfer count of the memory buffer to be transferred. In any given SGD table, two consecutive SGDs are offset by 8 bytes and are aligned on a 4-byte boundary.
2. Each Scatter-Gather Descriptor for the linked list must contain the following information:
   a. Memory Address (buffer start) 4 bytes
   b. Byte Count (buffer size) 3 bytes
   c. End of Link List 1 bit (MSB)

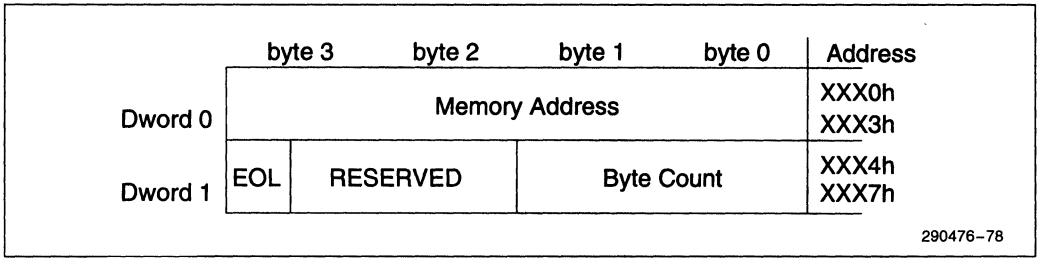


Figure 15. Scatter-Gather Descriptor Format

3. Initialize DMA Mode and Extended Mode registers with transfer specific information like 8-/16-bit I/O device, Transfer Mode, Transfer Type, etc.
4. Software provides the starting address of the Scatter-Gather Descriptor Table by loading the Scatter-Gather Descriptor Table Pointer register.
5. Engage the Scatter-Gather machine by writing a Start command to the Scatter-Gather Command register.
6. The Mask register should be cleared as last the last step of programming the DMA register set. This is to prevent DMA from starting a transfer with a partially loaded command description.
7. Once the register set is loaded and the channel is unmasked, the DMA will generate an internal request to fetch the first buffer from the Scatter Gather Descriptor Table.
8. The DMA will then respond to DREQ or software requests. The first transfer from the first buffer will move the memory address and word count from the Base register set to the Current register set. As long as Scatter-Gather is active and the Base register set is not loaded and the last buffer has not been fetched, the channel will generate a request to fetch a reserve buffer into the Base register set. The reserve buffer is loaded to minimize latency problems going from one buffer to another. Fetching a reserve buffer has a lower priority than completing DMA for the channel.
9. The DMA controller will terminate a Scatter-Gather cycle by detecting an End of List (EOL) bit in the SGD. After the EOL bit is detected, the channel will transfer the buffers in the Base and Current register sets if they are loaded. At Terminal Count the channel will assert EOP or IRQ13 depending on its programming and set the Terminate bit in the Scatter-Gather Status register. The Active bit in the Scatter-Gather Status register will be reset and the channel's Mask bit will be set.

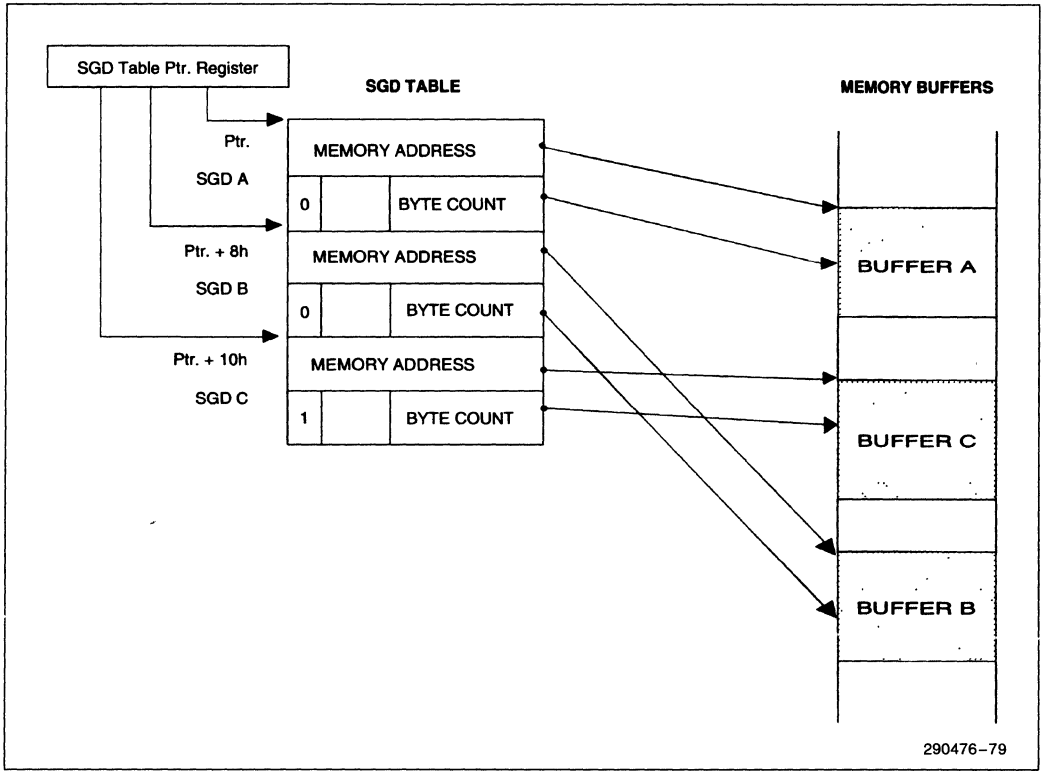


Figure 16. Link List Example

6.7 Register Functionality

See Section 3.2 for detailed information on register programming, bit definitions, and default values/functions after a reset.

DMA Channel 4 is used to cascade the two DMA controllers together and should not be programmed for any mode other than cascade. The Mode register for Channel 4 will default to cascade mode. Special attention should also be take when programming the Command and Mask registers as related to Channel 4 (refer to the Command and Mask register descriptions, Section 3.2).

6.7.1 ADDRESS COMPATIBILITY MODE

Whenever the DMA is operating in Address Compatibility mode, the addresses do not increment or decrement through the High and Low Page registers, and the high page register is set to 00h. This is compatible with the 82C37 and Low Page register implementation used in the PC AT. This mode is set when any of the lower three address bytes of a channel are programmed. If the upper byte of a channel's address is programmed last, the channel will go into Extended Address Mode. In this mode, the high byte may be any value and the address will increment or decrement through the entire 32-bit address.

After reset is negated all channels will be set to Address Compatibility Mode. The DMA Master Clear command will also reset the proper channels to Address Compatibility Mode. The Address Compatibility Mode bits are stored on a per channel basis.

6.7.2 SUMMARY OF THE DMA TRANSFER SIZES

Table 16 lists each of the DMA device transfer sizes. The column labeled "Word Count Register" indicates that the register contents represent either the number of bytes to transfer or the number of 16-bit words to transfer. The column labeled "Current Address Register Increment/Decrement" indicates the number added to or taken from the Current Address register after each DMA transfer cycle. The Mode Register determines if the Current Address register will be incremented or decremented.

Table 16. DMA Transfer Size

DMA Device Date Size And Word Count	Word Count Register	Current Address Increment/Decrement
8-bit I/O, Count By Bytes	Bytes	1
16-bit I/O, Count By Words (Address Shifted)	Words	1
16-bit I/O, Count By Bytes	Bytes	2
32-bit I/O, Count By Bytes	Bytes	4

6.7.3 ADDRESS SHIFTING WHEN PROGRAMMED FOR 16-BIT I/O COUNT BY WORDS

To maintain compatibility with the implementation of the DMA in the PC/AT which used the 82C37, the DMA will shift the addresses when the Extended Mode register is programmed for, or defaulted to, transfers to/from a 16-bit device count6.7.3-by-words. Note that the least significant bit of the Low Page register is dropped in 16-bit shifted mode. When programming the Current Address register while the DMA channel is in this mode, the Current Address must be programmed to an even address with the address value shifted right by one bit. The address shifting is as shown in Table 17.

Table 17. Address Shifting in 16-Bit I/O DMA Transfers

Output Address	8-Bit I/O Programmed Address	16-Bit I/O Programmed Address (Shifted)	16-Bit I/O Programmed Address (No Shift)	32-Bit I/O Programmed Address (No Shift)
A0 [16:1] A[31:17]	A0 A[16:1] A[31:17]	"0" A[15:0] A[31:17]	A0 A[16:01] A[31:17]	A0 A[16:01] A[31:17]

NOTE:
The least significant bit of the Low Register is dropped in 16-bit shifted mode.

6.7.4 STOP REGISTERS (RING BUFFER DATA STRUCTURE)

To support a common data communication data structure, (the ring buffer), a set of DMA registers have been provided. These registers are called Stop registers. Each channel has 22-bits of register location associated with it. The 22-bits are distributed between three different registers (one 8-bit and two 8-bit). The Stop registers can be enabled or disabled by writing to the channel's corresponding Extended Mode register.

The ring buffer data structure reserves a fixed portion of memory, on Dword boundaries, to be used for a DMA channel. Consecutively received frames or other data structures are stored sequentially within the boundaries of the ring buffer memory.

The beginning and end of the ring buffer area is defined in the Base Address register and the Base Address register + the Base Byte/Transfer Count. The incoming frames (data) are deposited in sequential locations of the ring buffer. When the DMA reaches the end of the ring buffer, indicating the byte count has expired, the DMA controller (if so programmed) will autoinitialize. Upon autoinitialization, the Current Address register will be restored from the Base Address register, taking the process back to the start of the ring buffer. The DMA will then be available to begin depositing the incoming bytes in the ring buffers sequential locations, providing that the CPU has read the data that was previously placed in those locations. The DMA determines that the CPU has read certain data by the value that the CPU writes into the Stop register.

Once the data of a frame is read by the CPU, the memory location it occupies becomes available for other incoming frames. The Stop register prevents the DMA from over writing data that has not yet been read by the CPU. After the CPU has read a frame from memory it will update the Stop register to point to the location that was last read. The DMA will not deposit data into any location beyond that pointed to by the Stop register. The last address transferred before the channel is masked is the first address that matches the Stop register.

For example, if the stop register = 00001Ch, the last three transfers are shown in Table 18.

Table 18. Stop Register Functionality Example

	By Bytes	By Words	By Words
Increment	XX00001Ah	XX000018h	XX000018h
	XX00001Bh	XX00001Ah	XX00001Ah
Decrement	XX00001Ch XX000021h	XX00001Ch XX000023h	XX00001Ch XX000023h
	XX000020h	XX000021h	XX000021h
	XX00001Fh	XX00001Fh	XX00001Fh

NOTE:
The Stop registers store values to compare against LA[23:2] only, so the size of the ring buffer is limited to 16 MBytes.

6.7.5 BUFFER CHAINING MODE AND STATUS REGISTERS

The Chaining Mode registers are used to implement the buffer chaining mode of a channel. The buffer chaining mode is useful when transferring data from a peripheral to several different areas of memory with one continuous transfer operation. Four registers are used to implement this function: the Chaining Mode register, the Chaining Mode Status Register, the Channel Interrupt Status register, and the Chain Buffer Expiration Control register.

The Chaining Mode register controls the buffer chaining initialization. Buffer chaining mode can be enabled or disabled. A Chaining Mode bit is used to indicate if Base register programming is complete and chaining can begin, or to hold off chaining because the Base registers still need programming. Another bit dictates the buffer expiration response by indicating whether an IRQ13 or EOP should be issued when the buffer needs reprogramming. The Chaining Mode Status Register indicates whether each channel's chaining mode is enabled or disabled.

The Channel Interrupt Status Register indicates the channel source of a DMA chaining interrupt on IRQ13. The CPU can read this register to determine which channel asserted IRQ13 following a buffer expiration. The Chain Buffer Expiration Control Register is a read only register that reflects the outcome after the expiration of a chain buffer. If a channel bit is set to 0, IRQ13 will be activated following the buffer expiration. If a channel bit is set to 1, EOP will be asserted following the buffer expiration.

6.7.6 AUTOINITIALIZE

By programming a bit in the Mode register, a channel may be set up as an autoinitialize channel. During Autoinitialization, the original values of the Current page, Current address and Current Byte/Word Count registers are automatically restored from the Base Address, and Word count registers of that channel following TC. The Base registers are loaded simultaneously with the Current registers by the microprocessor and remain unchanged throughout the DMA service. The mask bit is not set when the channel is in autoinitialize. Following autoinitialize the channel is ready to perform another DMA service, without CPU intervention, as soon as a valid DREQ is detected. (Note: Autoinitialize will not function if the channel is also programmed for Scatter-Gather or buffer chaining. Only one of these features should be enabled at a time.)

6.8 Software Commands

These are additional special software commands which can be executed in the Program Condition. They do not depend on any specific bit pattern on the data bus. The three software commands are:

1. Clear Byte Pointer Flip-Flop.
2. Master Clear.
3. Clear Mask Register.

6.8.1 CLEAR BYTE POINTER FLIP-FLOP

This command is executed prior to writing or reading new address or word count information to the DMA. This initializes the flip-flop to a known state so that subsequent accesses to register contents by the microprocessor will address upper and lower bytes in the correct sequence.

When the CPU is reading or writing DMA registers, two Byte Pointer Flip-Flops are used; one for Channels 0-3 and one for Channels 4-6. Both of these act independently. There are separate software commands for clearing each of them (0Ch for Channels 0-3, 0D8h for Channels 4-7).

An additional Byte Pointer Flip-Flop has been added for use when EISA masters are reading and writing DMA registers. (The arbiter state will be used to determine the current master of the bus.) This Flip-Flop is cleared when an EISA Master performs a write to either 00Ch or 0D8h. there is one Byte Pointer Flip Flop per eight DMA channels. This Byte Pointer was added to eliminate the problem of the CPU's byte pointer getting out of synchronization if an EISA Master takes the bus during the CPU's DMA programming.

6.8.2 DMA MASTER CLEAR

This software instruction has the same effect as the hardware Reset. The Command, Status, Request, and Internal First/Last Flip-Flop registers are cleared and the Mask register is set. The DMA Controller will enter the idle cycle.

There are two independent Master Clear Commands, 0Dh which acts on Channels 0-3, and 0DAh which acts on Channels 4-6.

6.8.3 CLEAR MASK REGISTER

This command clears the mask bits of all four channels, enabling them to accept DMA requests. I/O port 00Eh is used for Channels 0-3 and I/O port 0DCh is used for Channels 4-6.

6.9 Terminal Count/EOP Summary

Table 19 is a summary of the events that happen as a result of a terminal count or external EOP when running DMA in various modes.

Table 19. Terminal Count/EOP Summary Table

Conditions					
	AUTOINIT	No		Yes	
Event					
	Word Counter Expired	Yes	X	Yes	X
	EOP Input	X	Asserted	X	Asserted
Result					
	Status TC	set	set	set	set
	Mask	set	set	—	—
	SW Request	clr	clr	clr	clr
	Current Register	—	—	load	load

NOTES:
1. load = Load Current From Base
2. "-" = No Change
3. X = Don't Care
4. clr = Clear

6.10 Buffer Chaining

The buffer chaining mode of a channel is useful for transferring data from a peripheral to several different areas of memory within one transfer operation (from the DMA device's viewpoint). This is accomplished by causing the DMA to interrupt the CPU for more programming information while the previously programmed transfer is still in progress. Upon completion of the previous transfer, the DMA controller will then load the new transfer information automatically. In this way, the entire transfer can be completed without interrupting the operation of the DMA device. This mode is most useful for DMA single-cycle or demand modes where the transfer process allows time for the CPU to execute the interrupt routine.

The buffer chaining mode of a channel may be entered by programming the address and count of a transfer as usual. After the initial address and count is programmed, the Base registers are selected via the Chaining Mode register Chaining Mode Enabled bit. The address and count for the second transfer and both the Chaining Mode Enabled and the Program Complete bit of the Chaining Mode register should be programmed at this point, before starting the DMA process. When, during the DMA process, the Current Buffer is expired, the Base address, Page, and Count registers will be transferred to the Current registers and a signal that the buffer has been expired is sent to the programming master.

This signal will be an IRQ13 if the master is the CPU, or a TC if the programming master is an EISA Master device. The type of programming master is indicated in the DMA's Chaining Mode Register, bit 4. If the CPU is the programming master for the Channel, TC will be generated only if the Current buffer expires and there is no Next Buffer stored in the Base registers.

Upon the expiration of a Current Buffer, the new Base register contents should be programmed and both the Chaining Mode Enabled and Program complete bits of the Chaining Mode register should be set. This resets the interrupt, if the CPU was the programming master, and allows for the next Base register to Current register transfer. If the Program Complete bit is not set before the current transfer reaches TC, then the DMA controller will set the Mask bit and the TC bit in the Status register and stop transferring data. In this case, an over-run is likely to occur. To determine if this has, a read of either Status register or the Mask register can be done (the Mask register has been made readable). If the channel is masked or has registered a TC, the DMA channel has been stopped and the full address, count, and chaining mode must be programmed to return to normal operation.

Note that if the CPU is the programming master, an interrupt will only be generated if a Current Buffer expires and chaining mode is enabled. It will not occur during initial programming. The Channel Interrupt Status register will indicate pending interrupts only. That is, it will indicate an empty Base register with Chaining Mode enabled. When Chaining mode is enabled, only the Base registers are written by the processor, and only the Current registers can be read. The Current registers are only updated on a TC.

6.11 Refresh Unit

The ESC provides an EISA Bus compatible refresh unit that provides 14 bits of refresh address for EISA/ISA bus DRAMs that do not have their own local refresh units. The refresh system uses the combined functions of the Interval Timers, the DMA Arbiter, DMA address counter, and EISA Bus Controller. Functionally, the Refresh unit is a sub-section of the ESC DMA unit. The DMA Address Counter is used to increment the Refresh Address register following each refresh cycle. Interval Counter 1, Timer 1 generates an internal refresh request. The DMA Arbiter detects a Refresh signal from either the Counter/Timer or the REFRESH# input and determines when the refresh will be done. The DMA drives the refresh address out onto the LA address bus. The cycle is decoded and driven onto the EISA address bus by the EISA Bus Controller. The ESC EISA Bus Controller is responsible for generating the EISA cycle control signals. Timer 1 Counter 1 should be programmed to provide a refresh request about every 15 μs.

Requests for refresh cycles are generated by two sources: the ESC (Timer 1 Counter 1), and 16-bit masters that activate REFRESH# when they own the EISA bus.

If a 16-bit ISA bus master holds the bus longer than 15 μs, it must initiate memory refresh cycles. If the ISA Master initiates a Refresh cycle while it owns the bus, it floats the address lines and cycle control signals and asserts REFRESH# to the ESC. The ESC EISA Bus Controller generates the cycle control signals and the ESC DMA Refresh unit supplies the refresh address. The ISA Master must then wait one BCLK after MRDC# is negated before floating REFRESH# and driving the address lines and control signals.

Typically, the refresh cycle length is five BCLK's. The I/O slave can insert one wait state to extend the cycle to six BCLK's by asserting CHRDY. The ESC EISA Bus Controller, upon seeing REFRESH#, knows to run refresh cycles instead of DMA cycles.

7.0 EISA BUS ARBITRATION

The ESC receives requests for EISA Bus ownership from several different sources; from DMA devices, from the Refresh counter, from EISA masters and from PCI agents. PCI agents requesting the EISA Bus request the EISA Bus through the PCEB. Additionally, 16-bit ISA Masters may request the bus through a cascaded DMA channel (see the Cascade mode description in Section 6.2.4).

7.1 Arbitration Priority

At the top level of the arbiter, the ESC uses a three way rotating priority arbitration method. On a fully loaded bus, the order in which the devices are granted bus access is independent of the order in which they assert a bus request, since devices are serviced based on their position in the rotation. The arbitration scheme assures that DMA channels and EISA masters are able to access the bus with minimal latency.

The PCEB and EISA Masters share one of the slots in the three way rotating priority scheme. This sharing is a two way rotation between the CPU and EISA Masters as a group. In this arbitration scheme, the PCEB acts on behalf of the CPU and all other PCI masters.

EISA Masters have a rotating priority structure which can handle up to eight master requests.

The next position in the top level arbiter is occupied by the DMA. The DMA's DREQ lines can be placed in either fixed or rotating priority. The default mode is fixed and by programming the DMA Command registers, the priority can be modified to rotating priority mode.

7.2 Preemption

An EISA compatible arbiter ensures that minimum latencies are observed for both EISA DMA devices, and EISA Masters.

7.2.1 PCEB EISA BUS ACQUISITION AND PCEB PREEMPTION

EISA bus arbitration is intended to be optimized for CPU access the EISA bus. Since the CPU accesses to the EISA Bus through the PCEB, the PCEB is assumed to be the default owner of the EISA bus. The arbitration interface between the PCEB and the ESC is implemented as a HOLD/HLDA (EISAHOLD/ EISAHLDA) pair.

If a PCI cycle requires access to the EISA Bus while EISAHLDA signal is asserted (EISA Bus busy) the PCI cycle is retried, and the PCEB requests the EISA bus by asserting PEREQ#. The ESC, after sampling PEREQ# asserted, preempts the current owner of the EISA Bus. The ESC grants the EISA Bus by negating EISAHOLD signal.

The ESC asserts EISAHOLD to the PCEB when the ESC needs to acquire the ownership of the EISA bus. While EISAHOLD is asserted, the arbitration process is dynamic and may change (i.e. the ESC is still accepting EISA Bus requests). When the PCEB returns EISAHLDA, the arbiter freezes the arbitration process and determine the winner. If the new winner is an EISA Master or DMA channel, the ESC will assert NMFLUSH#. The ESC tri-states the NMFLUSH# output driver on the following clock. The PCEB holds NMFLUSH# asserted until all buffers are flushed. After all buffers are flushed, the PCEB negates NMFLUSH# and then tri-state the output buffer. After sampling NMFLUSH# negated, the ESC resumes driving NMFLUSH# on the next PCI clock. This way the ESC does not assert MACK# or DACK# until the PCEB acknowledges that all line buffers have been flushed.

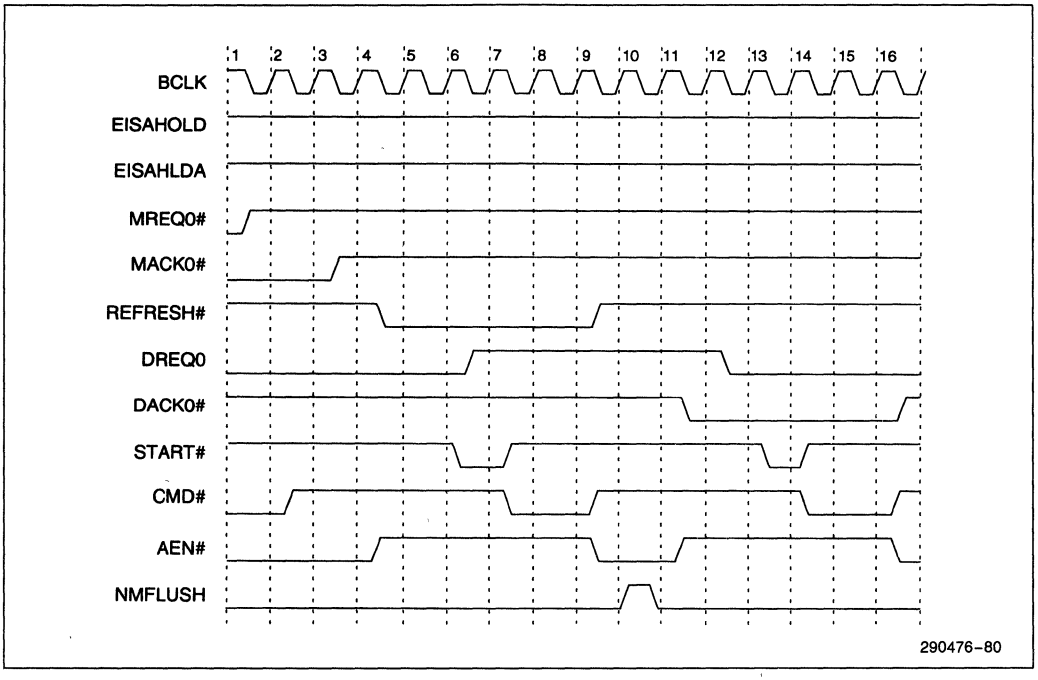


Figure 17. EISA Arbitration

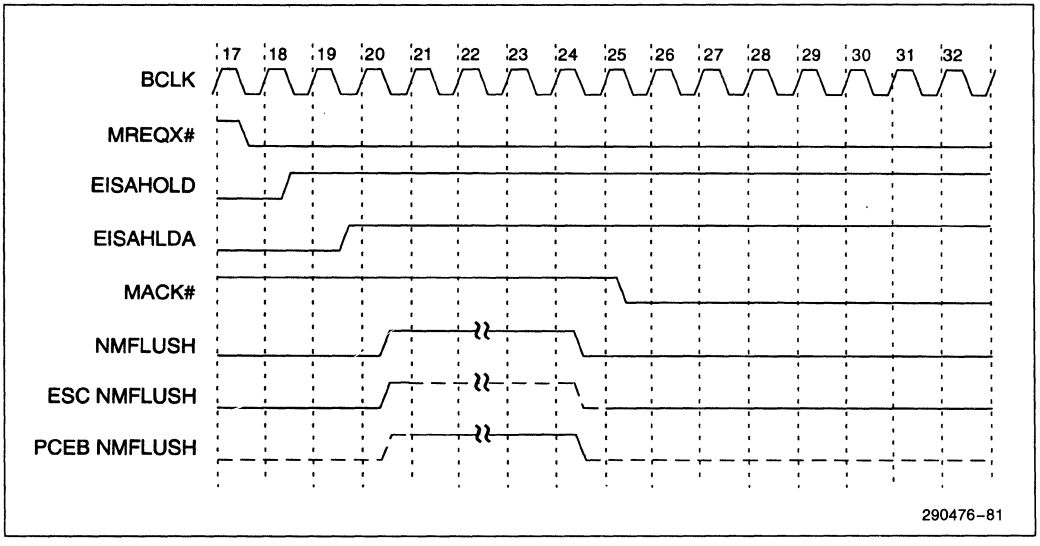


Figure 18. PCEB Preemption

7.2.2 EISA MASTER PREEMPTION

EISA specification requires that EISA Masters must release the bus within 64 BCLKs (8 μs) after the ESC negates MACKx#. If the bus master attempts to start a new bus cycle after this timeout period, a bus timeout (NMI) is generated and the RSTDRV is asserted to reset the offending bus master.

7.2.3 DMA PREEMPTION

A DMA slave device that is not programmed for compatible timing is preempted from the EISA Bus by another device that requests use of the bus. This will occur regardless of the priority of the pending request. For DMA devices not using compatible timing mode, the DMA controller stops the DMA transfer and releases the bus within 32 BCLK (4 μs) of a preemption request. Upon the expiration of the 4 μs timer, the DACK is negated after the current DMA cycle has completed. The EISA Bus then arbitrated for and granted to the highest priority requester. This feature allows flexibility in programming the DMA for long transfer sequences in a performance timing mode while guaranteeing that vital system services such as Refresh are allowed access to the expansion bus.

The 4 μs timer is not used in compatible timing mode. It is only used for DMA channels programmed for Type "A", Type "B", or Type "C" (Burst) timing. The 4 μs timer is also not used for 16-bit ISA masters cascaded through the DMA DREQ lines.

If the DMA channel that was preempted by the 4 μs timer is operating in Block mode, an internal bit will be set so that the channel will be arbitrated for again, independent of the state of DREQ.

7.3 Slave Timeouts

A slave which does not release EXRDY or CHRDY can cause the CMD# active time to exceed 256 BCLKs (32 μs). The ESC does not monitor EXRDY or CHRDY for this timeout. Typically this function is provided in a system through a third party add-in card. The add-in cards which monitor EXRDY or CHRDY assert IOCHK signal when the 256 BCLK count expires. The ESC in response asserts NMI.

The only way that a 16-bit ISA Master can be preempted from the EISA bus is if it exceeds the 256 BCLK

(32 μs) limit on CMD# active.

7.4 Arbitration During Non-Maskable Interrupts

If a non-maskable interrupt (NMI) is pending at the PCEB, and the PCEB is requesting the bus, the DMA and EISA Masters will be bypassed each time they come up for rotation. This gives the PCEB the EISA Bus bandwidth on behalf of the CPU to process the interrupt as fast as possible.

8.0 INTERVAL TIMERS

The ESC contains five counter/timers that are equivalent to those found in the 82C54 programmable interval timer. The five counters are contained in two separate ESC timer units, referred to as Timer-1 and Timer-2. The ESC uses the Timers to implement key EISA system functions. Timer-1 contains three counters, and Timer-2 contains two counters. EISA systems do not use the middle counter on Timer-2.

Interval Timer 1, Counter 0 is connected to the interrupt controller IRQ0 and provides a system timer interrupt for a time-of-day, diskette time-out, or other system timing functions. Counter 1 generates a refresh-request signal and Counter 2 generates the tone for the speaker.

Interval Timer 2, Counter 0 implements a fail safe timer. Counter 0 generates NMI at regular intervals, thus preventing the system from locking up. Counter 1 is not used. Counter 2 is used to slow down the CPU by means of pulse-width modulation. The output of Timer 2 Counter 2 is tied to the SLOWH# signal.

Table 20. Interval Timer Functions

Function	Counter 0 System Timer	Counter 0 Fail-Safe Timer
Gate	Always On	Always On
Clock In	1.193 MHz(OSC/12)	0.298 MHz(OSC/48)
Out	INT-1 IRQ0	NMI Interrupt
	Counter 1 Refresh Request	
Gate	Always On	
Clock In	1.193 MHz(0SC/12)	
Out	Refresh Request	
	Counter 2	**Counter 2**
Gate	Programmable	Refresh Request
Clock In	Port 61h	
Out	1.193 MHz(OSC/12)	8 MHz (BCLK)
	Speaker	CPU Speed Control (SLOWH#)

8.1 Interval Timer Address Map

Table 21 shows the I/O address map of the interval timer counters:

Table 21. Interval Timer I/O Address Map

I/O Port Address	Register Description
040h	Timer 1, System Timer (Counter 0)
041h	Timer 1, Refresh Request (Counter 1)
042h	Timer 1, Speaker Tone (Counter 2)
043h	Timer 1, Control Word Register
048h	Timer 2, Fail-Safe Timer (Counter 0)
049h	Timer 2, Reserved
04Ah	Timer 2, CPU Speed Control (Counter 2)
04Bh	Timer 2, Control Word Register

Timer 1—Counter 0, System Timer

This counter functions as the system timer by controlling the state of IRQ[0] and is typically programmed for Mode 3 operation. The counter produces a square wave with a period equal to the product of the counter period (838 ns) and the initial count value. The counter loads the initial count value one counter period after software writes the count value to the counter I/O address. The counter initially asserts IRQ[0] and decrements the count value by two each counter period. The counter negates IRQ[0] when the count value reaches 0. It then reloads the initial count value and again decrements the initial count value by two each counter period. The counter then asserts IRQ[0] when the count value reaches "0", reloads the initial count value, and repeats the cycle, alternately asserting and negating IRQ[0].

Timer 1—Counter 1, Refresh Request Signal

This counter provides the Refresh Request signal and is typically programmed for Mode 2 operation. The counter negates Refresh Request for one counter period (833 ns) during each count cycle. The initial count value is loaded one counter period after being written to the counter I/O address. The counter initially asserts Refresh Request, and negates it for 1 counter period when the count value reaches 1. The counter then asserts Refresh Request and continues counting from the initial count value.

Timer 1—Counter 2, Speaker Tone

This counter provides the speaker tone and is typically programmed for Mode 3 operation. The counter provides a speaker frequency equal to the counter clock frequency (1.193 MHz) divided by the initial count value. The speaker must be enabled by a write to port 061h (see Section 3.7 on the NMI Status and Control Ports).

Timer 2—Counter 0, Fail-Safe Timer

This counter functions as a fail-save timer by preventing the system from locking up. This counter generates an interrupt on the NMI line as the count expires by setting bit 7 on Port 0461h. Software routines can avoid the Fail-Safe NMI by resetting the counter before the timer count expires.

Timer 2—Counter 2, CPU Speed Control

This counter generates the SLOWH# to the CPU and is typically programmed for Mode 1 operation. The counter is triggered by the refresh request signal generated by Timer 1-Counter 1 only. If the counter is programmed, the counters SLOWH# output will stop the CPU for the programmed period of the one-shot every time a refresh request occurs. This counter is not configured or programmed until a speed reduction in the system is required.

8.2 Programming The Interval Timer

The counter/timers are programmed by I/O accesses and are addressed as though they are contained in two separate 82C54 interval timers. Timer 1 contains three counters and Timer 2 contains two counters. Each Timer is controlled by a separate Control Word register. Table 22 lists the six operating modes for the interval counters. Note that for the fail safe timer (timer 2, counter 0), only mode 0 is supported.

The interval timer is an I/O-mapped device. Several commands are available:

1. The Control Word Command specifies:
   — which counter to read or write
   — the operating mode
   — the count format (binary or BCD)
2. The Counter Latch Command latches the current count so that it can be read by the system. The count-down process continues.
3. The Read Back Command reads the count value, programmed mode, the current state of the OUT pins, and the state of the Null Count Flag of the selected counter.

The Read/Write Logic selects the Control Word register during an I/O write when address lines A1, A0 = 11. This condition occurs during an I/O write to port addresses 043h and 04Bh, the addresses for the Control Word Register on Timer 1 and Control Word Register on Timer 2 respectively. If the CPU writes to port 043h or port 04Bh, the data is stored in the respective Control Word Register and is interpreted as a Control Word used to define the operation of the Counters.

The Control Word Register is write-only. Counter Status information is available with the Read-Back Command.

Table 22. Counter Operating Modes

Mode	Function
0	Out signal on end of count (= 0)
1	Hardware retriggerable one-shot
2	Rate generator (divide by n counter)
3	Square wave output
4	Software triggered strobe
5	Hardware triggered strobe

Because the timer counters come up in an unknown state after power up, multiple refresh requests may be queued up. To avoid possible multiple refresh cycles after power up, program the timer counter immediately after power up.

Write Operations

Programming the interval timer is a simple process:

1. Write a control word.
2. Write an initial count for each counter.
3. Load the least and/or most significant bytes (as required by Control Word Bits 5, 4) of the 16-bit counter.

The programming procedure for the ESC timer units is very flexible. Only two conventions need to be observed. First, for each Counter, the Control Word must be written before the initial count is written. Second, the initial count must follow the count format specified in the Control Word (least significant byte only, most significant byte only, or least significant byte and then most significant byte).

Since the Control Word Register and the three Counters have separate addresses (selected by the A1, A0 inputs), and each Control Word specifies the Counter it applies to (SC0, SC1 bits), no special instruction sequence is required. Any programming sequence that follows the conventions above is acceptable.

A new initial count may be written to a Counter at any time without affecting the Counter's programmed Mode in any way. Counting will be effected as described in the Mode definitions. The new count must follow the programmed count format.

If a Counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same Counter. Otherwise, the Counter will be loaded with an incorrect count.

Interval Timer Control Word Format

The Control Word specifies the counter, the operating mode, the order and size of the COUNT value, and whether it counts down in a 16-bit or binary-coded decimal (BCD) format. After writing the control word, a new count may be written at any time. The new value will take effect according to the programmed mode.

If a counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same counter. Otherwise, the counter will be loaded with an incorrect count. The count must always be completely loaded with both bytes.

Read Operations

It is often desirable to read the value of a Counter without disturbing the count in progress. This is easily done in the ESC timer units.

There are three possible methods for reading the counters: a simple read operation, the Counter Latch Command, and the Read-Back Command.

Counter I/O Port Read

The first method is to perform a simple read operation. To read the Counter the CLK input of the selected Counter must be inhibited by using either the GATE input or external logic. Otherwise, the count may be in the process of changing when it is read, giving an undefined result. When reading the count value directly, follow the format programmed in the control register: read LSB, read MSB, or read LSB then MSB. Within the ESC timer unit, the GATE input on Timer 1 Counter 0, Counter 1 and Timer 2 Counter 0 are tied high. Therefore, the direct register read should not be used on these two counters. The GATE input of Timer 1 Counter 2 is controlled through I/O port 061h. If the GATE is disabled through this register, direct I/O reads of port 042h will return the current count value.

Counter Latch Command

The Counter Latch command latches the count at the time the command is received. This command is used to insure that the count read from the counter is accurate (particularly when reading a two-byte count). The count value is then read from each counter's Count register as was programmed by the Control register.

The selected Counter's output latch (OL) latches the count at the time the Counter Latch Command is received. This count is held in the latch until it is read by the CPU (or until the Counter is reprogrammed). The count is then unlatched automatically and the OL returns to "following" the counting element (CE). This allows reading the contents of the Counters "on the fly" without effecting counting in progress. Multiple Counter Latch Commands may be used to latch more than one Counter. Each latched Counter's OL holds its count until it is read. Counter Latch Commands do not effect the programmed Mode of the Counter in any way. The Counter Latch Command can be used for each counter in the ESC timer unit.

If a Counter is latched and then, some time later, latched again before the count is read, the second Counter Latch Command is ignored. The count read will be the count at the time the first Counter Latch Command was issued.

With either method, the count must be read according to the programmed format; specifically, if the Counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other; read, write, or programming operations for other Counters may be inserted between them.

Another feature of the ESC timer unit is that reads and writes of the same Counter may be interleaved. For example, if the Counter is programmed for two byte counts, the following sequence is valid:

1. Read least significant byte.
2. Write new least significant byte.
3. Read most significant byte.
4. Write new most significant byte.

One precaution is worth noting. If a Counter is programmed to read/write two-byte counts, a program must not transfer control between reading the first and second byte to another routine which also reads from that same Counter. Otherwise, an incorrect count will be read.

Read Back Command

The third method uses the Read-Back command. The Read-Back command is used to determine the count value, programmed mode, and current states of the OUT pin and Null Count flag of the selected counter or counters. The Read-Back command is written to the Control Word register, which causes the current states of the above mentioned variables to be latched. The value of the counter and its status may then be read by I/O access to the counter address.

The read-back command may be used to latch multiple counter output latches (OL) by setting the COUNT# bit D5 = 0 and selecting the desired counter(s). This single command is functionally equivalent to several counter latch commands, one for each counter latched. Each counter's latched count is held until it is read (or the counter is reprogrammed). Once read, a counter is automatically unlatched. The other counters remain latched until they are read. If multiple count read-back commands are issued to the same counter without reading the count, all but the first are ignored; i.e. the count which will be read is the count at the time the first read-back command was issued.

The read-back command may also be used to latch status information of selected counter(s) by setting STATUS# bit D4 = 0. Status must be latched to be read. The status of a counter is accessed by a read from that counter's I/O port address.

If multiple counter status latch operations are performed without reading the status, all but the first are ignored. The status returned from the read is the counter status at the time the first status read-back command was issued.

Both count and status of the selected counter(s) may be latched simultaneously by setting both the COUNT# and STATUS# bits[5:4] = 00b. This is functionally the same as issuing two consecutive, separate read-back commands. The above discussions apply here also. Specifically, if multiple count and/or status read-back commands are issued to the same counter(s) without any intervening reads, all but the first are ignored.

If both count and status of a counter are latched, the first read operation from that counter will return the latched status, regardless of which was latched first. The next one or two reads (depending on whether the counter is programmed for one or two type counts) return the latched count. Subsequent reads return unlatched count.

9.0 INTERRUPT CONTROLLER

The ESC provides an EISA compatible interrupt controller which incorporates the functionality of two 82C59 interrupt controllers. The two controllers are cascaded so that 14 external and two internal interrupts are possible. The master interrupt controller provides IRQ[7:0] and the slave interrupt controller provides IRQ[15:8] (Figure 19). The two internal interrupts are used for internal functions only and are not available at the chip periphery. IRQ2 is used to cascade the two controllers together and IRQ0 is used as a system timer interrupt and is tied to Interval Timer 1, Counter 0. The remaining 14 interrupt lines (IRQ1, IRQ3-IRQ15) are available for external system interrupts. Edge or level sense selection is programmable on a by-controller basis.

The Interrupt Controller consists of two separate 82C59 cores. Interrupt Controller 1 (CNTRL-1) and Interrupt Controller 2 (CNTRL-2) are initialized separately, and can be programmed to operate in different modes. The default settings are: 80x86 Mode, Edge Sensitive (IRQ0-15) Detection, Normal EOI, Non-Buffered Mode, Special Fully Nested Mode disabled, and Cascade Mode. CNTRL-1 is connected as the Master Interrupt Controller and CNTRL-2 is connected as the Slave Interrupt Controller.

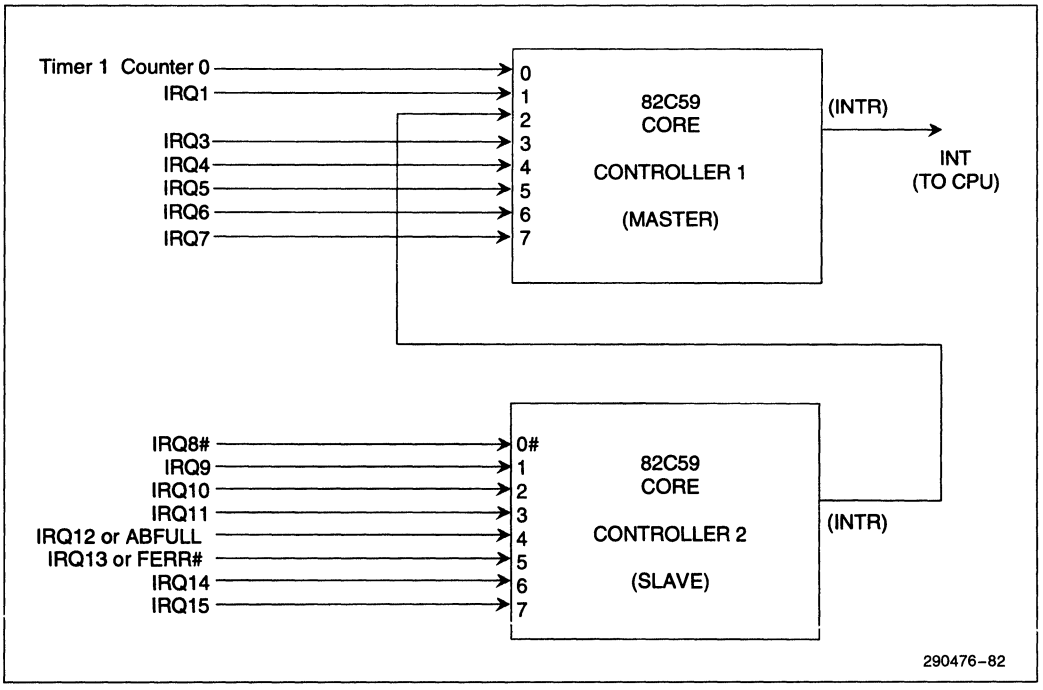


Figure 19. Block Diagram of the Interrupt Controller

Table 23 lists the I/O port address map for the interrupt registers.

Table 23. I/O Address Map

Interrupts	I/O Address	# of bits	Register
IRQ[7:0]	0020h	8	CNTRL-1 Control Register
IRQ[7:0]	0021h	8	CNTRL-1 Mask Register
IRQ[7:0]	04D0h	8	CNTRL-1 Edge/Level Control Register
IRQ[15:8]	00A0h	8	CNTRL-2 Control Register
IRQ[15:8]	00A1h	8	CNTRL-2 Mask Register
IRQ[15:8]	04D1h	8	CNTRL-2 Edge/Level Control Register

IRQ0, and IRQ2 are connected to the interrupt controllers internally. The other interrupts are always generated externally. IRQ12 and IRQ13 may be generated internally through the ABFULL and FERR# signals, respectively.

Table 24. Typical Interrupt Functions

Priority	Label	Controller	Typical Interrupt Source
1	IRQ0	1	Interval Timer 1, Counter 0 OUT
2	IRQ1	1	Keyboard
3-10	IRQ2	1	Interrupt from Controller 2
3	IRQ8#	2	Real Time Clock
4	IRQ9	2	Expansion Bus Pin B04
5	IRQ10	2	Expansion Bus Pin D03
6	IRQ11	2	Expansion Bus Pin D04
7	IRQ12	2	Expansion Bus Pin D05
8	IRQ13	2	Coprocessor Error, Chaining
9	IRQ14	2	Fixed Disk Drive Controller Expansion Bus Pin D07
10	IRQ15	2	Expansion Bus Pin D06
11	IRQ3	1	Serial Port 2, Expansion Bus B25
12	IRQ4	1	Serial Port 1, Expansion Bus B24
13	IRQ5	1	Parallel Port 2, Expansion Bus B23
14	IRQ6	1	Diskette Controller, Expansion Bus B22
15	IRQ7	1	Parallel Port 1, Expansion Bus B21

9.1 Interrupt Controller Internal Registers

Several registers are contained internally within each 82C59. The interrupts at the IRQ input lines are handled by two registers in cascade, the Interrupt Request Register (IRR) and the In-Service Register (ISR). The IRR is used to store all the interrupt levels which are requesting service and the ISR is used to store all the interrupt levels which are being serviced.

Internal circuitry determines the priorities of the bits set in the IRR. The highest priority is selected and strobed into the corresponding bit of the ISR during Interrupt Acknowledge Cycles.

The Interrupt Mask Register (IMR) stores the bits which mask the incoming interrupt lines. The IMR operates on the IRR. Masking of a higher priority input will not effect the interrupt request lines of lower priority inputs.

9.2 Interrupt Sequence

The powerful features of the Interrupt Controller in a microcomputer system are its programmability and the interrupt routine addressing capability. The latter allows direct or indirect jumping to the specific interrupt routine requested without any polling of the interrupting devices. The following shows the interrupt sequence for an x86 type system (the 8080 mode of the interrupt controller must never be selected when programming the ESC).

Note that externally, the interrupt acknowledge cycle sequence appears different than in a traditional discrete 82C59 implementation. However, the traditional interrupt acknowledge sequence is generated within the ESC and it is an EISA compatible implementation.

1. One or more of the Interrupt Request (IRQ[x]) lines are raised high, setting the corresponding IRR bit(s).
2. The Interrupt Controller evaluates these requests, and sends an INTR to the CPU, if appropriate.
3. The CPU acknowledges the INTR and responds with an interrupt acknowledge cycle. This cycle is translated into a PCI bus command. This PCI command is broadcast over the PCI bus as a single cycle as opposed to the two cycle method typically used.
4. Upon receiving an interrupt acknowledge cycle from the CPU over the PCI, the PCEB converts the single cycle into an INTA# pulse to the ESC. The ESC uses the INTA# pulse to generate the two cycles that the internal 8259 pair can respond to with the expected interrupt vector. The cycle conversion is performed by a functional block in the ESC Interrupt Controller Unit. The internally generated interrupt acknowledge cycle is completed as soon as possible as the PCI bus is held in wait states until the interrupt vector data is returned. Each cycle appears as an interrupt acknowledge pulse on the INTA# pin of the cascaded interrupt controllers. These two pulses are not observable at the ESC periphery.
5. Upon receiving the first internally generated interrupt acknowledge, the highest priority ISR bit is set and the corresponding IRR bit is reset. The Interrupt Controller does not drive the Data Bus during this cycle. On the trailing edge of the first cycle pulse, a slave identification code is broadcast by the master to the slave on a private, internal three bit wide bus. The slave controller uses these bits to determine if it must respond with an interrupt vector during the second INTA# cycle.
6. Upon receiving the second internally generated interrupt acknowledge, the Interrupt Controller releases an 8-bit pointer (the interrupt vector) onto the Data Bus where it is read by the CPU.
7. This completes the interrupt cycle. In the AEOI mode the ISR bit is reset at the end of the second interrupt acknowledge cycle pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt subroutine.

If no interrupt request is present at step four of either sequence (i.e., the request was too short in duration) the Interrupt Controller will issue an interrupt level 7.

9.3 80x86 Mode

When initializing the control registers of the 82C59, an option exists in Initialization Control Word Four (ICW4) to select either an 80x86 or an MSC-85 microprocessor based system. The interrupt acknowledge cycle is different in an MSC-85 based system than in the 80x86 based system: the interrupt acknowledge takes three INTA# pulses with the MSC-85, rather than the two pulses with the 80x86. The ESC is used only in an 80x86 based system. You must program each interrupt controller's ICW4 bit 0 to a "1" to indicate that the interrupt controller is operating in an 80x86 based system. This setting ensures proper operation during an interrupt acknowledge.

9.3.1 ESC INTERRUPT ACKNOWLEDGE CYCLE

As discussed, the CPU generates an interrupt acknowledge cycle that is translated into a single PCI command and broadcast across the PCI bus to the PCEB. The PCEB pulses the INTA# signal to the ESC. The ESC Interrupt Unit translates the INTA# signal into the two INTA# pulses expected by the interrupt controller subsystem. The Interrupt Controller uses the first interrupt acknowledge cycle to internally freeze the state of the interrupts for priority resolution. The first controller (CNTRL-1), as a master, issues a three bit interrupt code on the cascade lines to CNTRL-2 (internal to the ESC) at the end of the INTA# pulse. On this first cycle the interrupt controller block does not issue any data to the processor and leaves its data bus buffers disabled. CNTRL-2 decodes the information on the cascade lines, compares the code to the byte stored in Initialization

Command Word Three (ICW3), and determines if it will have to broadcast the interrupt vector during the second interrupt acknowledge cycle. On the second interrupt acknowledge cycle, the master (CNTRL-1) or slave (CNTRL-2), will send a byte of data to the processor with the acknowledged interrupt code composed as follows:

Table 25. Content of Interrupt Vector Byte for 80x86 System Mode

Interrupt	D7	D6	D5	D4	D3	D2	D1	D0
IRQ7,15	T7	T6	T5	T4	T3	1	1	1
IRQ6,14	T7	T6	T5	T4	T3	1	1	0
IRQ5,13	T7	T6	T5	T4	T3	1	0	1
IRQ4,12	T7	T6	T5	T4	T3	1	0	0
IRQ3,11	T7	T6	T5	T4	T3	0	1	1
IRQ2,10	T7	T6	T5	T4	T3	0	1	0
IRQ1,9	T7	T6	T5	T4	T3	0	0	1
IRQ0,8	T7	T6	T5	T4	T3	0	0	0

NOTE:
T7–T3 represent the interrupt vector address (refer Register Description section).

The byte of data released by the interrupt unit onto the data bus is referred to as the "interrupt vector". The format for this data is illustrated on a per-interrupt basis in Table 25.

9.4 Programming The Interrupt Controller

The Interrupt Controller accepts two types of command words generated by the CPU or bus master:

1. **Initialization Command Words (ICWs):** Before normal operation can begin, each Interrupt Controller in the system must be initialized. In the 82C59, this is a two to four byte sequence. However, for the ESC, each controller must be initialized with a four byte sequence. This four byte sequence is required to configure the interrupt controller correctly for the ESC implementation. This implementation is EISA-compatible.

   The four initialization command words are referred to by their acronyms: ICW1, ICW2, ICW3, and ICW4.

   The base address for each interrupt controller is a fixed location in the I/O memory space, at 0020h for CNTRL-1 and at 00A0h for CNTRL-2.

   An I/O write to the CNTRL-1 or CNTRL-2 base address with data bit 4 equal to 1 is interpreted as ICW1. For ESC-based EISA systems, three I/O writes to "base address + 1" (021h for CNTRL-1 and 0A0h for CNTRL-2) must follow the ICW1. The first write to "base address + 1" (021h/0A0h) performs ICW2, the second write performs ICW3, and the third write performs ICW4.

ICW1 starts the initialization sequence during which the following automatically occur:

a. Following initialization, an interrupt request (IRQ) input must make a low-to-high transition to generate an interrupt.

b. The Interrupt Mask Register is cleared.

c. IRQ7 input is assigned priority 7.

d. The slave mode address is set to 7.

e. Special Mask Mode is cleared and Status Read is set to IRR.

ICW2 is programmed to provide bits[7:3] of the interrupt vector that will be released onto the data bus by the interrupt controller during an interrupt acknowledge. A different base [7:3] is selected for each interrupt controller. Suggested values for a typical EISA system are listed in Table 26.

ICW3 is programmed differently for CNTRL-1 and CNTRL-2, and has a different meaning for each controller.

For CNTRL-1, the master controller, ICW3 is used to indicate which IRQx input line is used to cascade CNTRL-2, the slave controller. Within the ESC interrupt unit, IRQ2 on CNTRL-1 is used to cascade the INTR output of CNTRL-2. Consequently, bit 2 of ICW3 on CNTRL-1 is set to a 1, and the other bits are set to 0's.

For CNTRL-2, ICW3 is the slave identification code used during an interrupt acknowledge cycle. CNTRL-1 broadcasts a code to CNTRL-2 over three internal cascade lines if an IRQ[x] line from CNTRL-2 won the priority arbitration on the master controller and was granted an interrupt acknowledge by the CPU. CNTRL-2 compares this identification code to the value stored in ICW3, and if the code is equal to bits[2:0] of ICW3, CNTRL-2 assumes responsibility for broadcasting the interrupt vector during the second interrupt acknowledge cycle pulse.

ICW4 must be programmed on both controllers. At the very least, bit 0 must be set to a 1 to indicate that the controllers are operating in an 80x86 system.

2. **Operation Command Words (OCWs):** These are the command words which dynamically reprogram the Interrupt Controller to operate in various interrupt modes.

   Any interrupt lines can be masked by writing an OCW1. A 1 written in any bit of this command word will mask incoming interrupt requests on the corresponding IRQx line.

   OCW2 is used to control the rotation of interrupt priorities when operating in the rotating priority mode and to control the End of Interrupt (EOI) function of the controller.

   OCW3 is used to set up reads of the ISR and IRR, to enable or disable the Special Mask Mode (SMM), and to set up the interrupt controller in polled interrupt mode.

   The OCWs can be written into the Interrupt Controller any time after initialization. Table 26 shows an example of typical values programmed by the BIOS at power-up for the ESC interrupt controller.

Table 26. Suggested Default Values For Interrupt Controller Registers

Port	Value	Description of Contents
020h	11h	CNTLR-1, ICW1
021h	08h	CNTLR-1, ICW2 Vector Address for 000020h
021h	04h	CNTLR-1, ICW3 Indicates Slave Connection
021h	01h	CNTLR-1, ICW4 8086 Mode
021h	B8h	CNTLR-1, Interrupt Mask (may vary)
4D0h	00h	CNTLR-1, Edge/Level Control Register
0A0h	11h	CNTLR-2, ICW1
0A1h	70h	CNTLR-2, ICW2 Vector Address for 0001C0h
0A1h	02h	CNTLR-2, ICW3 Indicates Slave ID
0A1h	01h	CNTLR-2, ICW4 8086 Mode
4D1h	00h	CNTLR-2, Edge/Level Control Register
0A1h	BDh	CNTLR-2, Interrupt Mask (may vary)

Figure 20 illustrates the sequence software must follow to load the interrupt controller Initialization Command Words (ICWs). The sequence must be executed for CNTRL-1 and CNTRL-2. After writing ICW1, ICW2, ICW3, and ICW4 must be written in order. Any divergence from this sequence, such as an attempt to program an OCW, will result in improper initialization of the interrupt controller and unexpected, erratic system behavior. It is suggested that CNTRL-2 be initialized first, followed by CNTRL-1.

In the ESC, it is required that all four Initialization Command Words (ICWs) be initialized. As shown in Figure 20, all ICWs must be programmed prior to programming the OCWs.

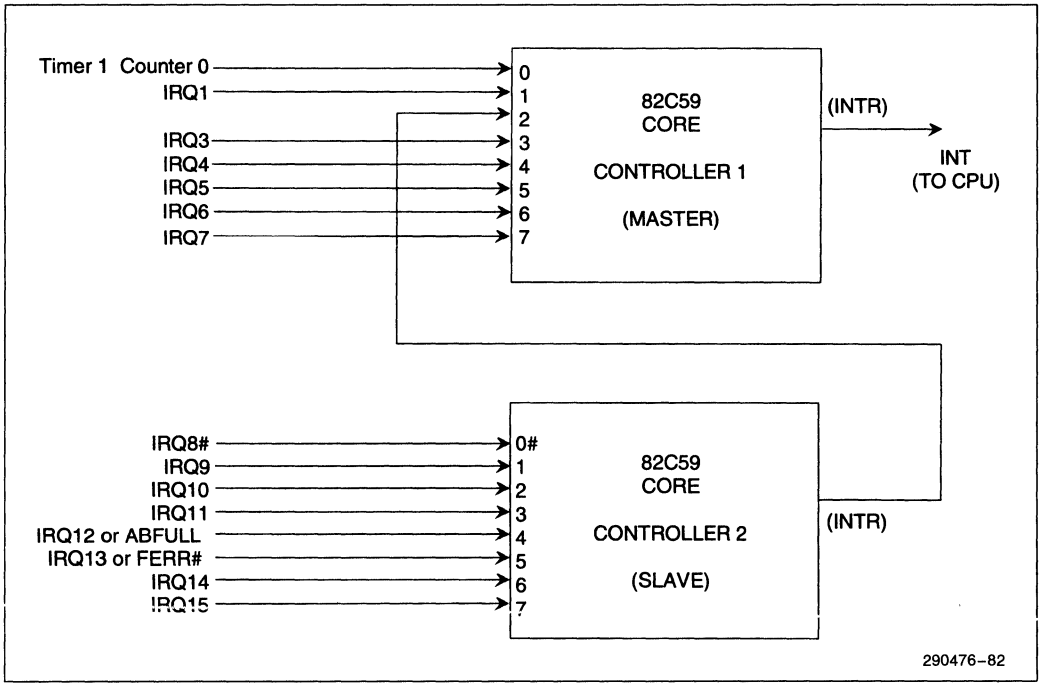


Figure 20. Initialization Sequence for ESC Initialization Command Words (ICWs)

9.5 End-Of-Interrupt Operation

9.5.1 END OF INTERRUPT (EOI)

The In Service (IS) bit can be reset either automatically following the trailing edge of the second internal INTA# pulse (when AEOI bit in ICW1 is set) or by a command word that must be issued to the Interrupt Controller before returning from a service routine (EOI command). An EOI command must be issued twice with this cascaded interrupt controller configuration, once for master and once for the slave.

There are two forms of EOI commands: Specific and Non-Specific. When the Interrupt Controller is operated in modes which preserve the fully nested structure, it can determine which IS bit to reset on EOI. When a Non-Specific EOI command is issued, the Interrupt Controller will automatically reset the highest IS bit of those that are set, since in the fully nested mode the highest IS level was necessarily the last level acknowledged and serviced. A non-specific EOI can be issued with OCW2 (EOI = 1, SL = 0, R = 0).

When a mode is used which may disturb the fully nested structure, the Interrupt Controller may no longer be able to determine the last level acknowledged. In this case a Specific End of Interrupt must be issued which includes as part of the command the IS level to be reset. A specific EOI can be issued with OCW2 (EOI = 1, SL = 1, R = 0, and L0-L2 is the binary level of the IS bit to be reset).

It should be noted that an IS bit that is masked by an IMR bit will not be cleared by a non-specific EOI if the Interrupt Controller is in the Special Mask Mode.

9.5.2 AUTOMATIC END OF INTERRUPT (AEOI) MODE

If AEOI = 1 in ICW4, then the Interrupt Controller will operate in AEOI mode continuously until reprogrammed by ICW4. Note that reprogramming ICW4 implies that ICW1, ICW2, and ICW3 must be reprogrammed first, in sequence. In this mode the Interrupt Controller will automatically perform a non-specific EOI operation at the trailing edge of the last interrupt acknowledge pulse. Note that from a system standpoint, this mode should be used only when a nested multilevel interrupt structure is not required within a single Interrupt Controller. The AEOI mode can only be used in a master Interrupt Controller and not a slave (on CNTRL-1 but not CNTRL-2).

9.6 Modes Of Operation

9.6.1 FULLY NESTED MODE

This mode is entered after initialization unless another mode is programmed. The interrupt requests are ordered in priority from 0 through 7 (0 being the highest). When an interrupt is acknowledged the highest priority request is determined and its vector placed on the bus. Additionally, a bit of the Interrupt Service register (IS[0:7]) is set. This IS bit remains set until the microprocessor issues an End of Interrupt (EOI) command immediately before returning from the service routine. Or, if the AEOI (Automatic End of Interrupt) bit is set, this IS bit remains set until the trailing edge of the second internal INTA#. While the IS bit is set, all further interrupts of the same or lower priority are inhibited, while higher levels will generate an interrupt (which will be acknowledged only if the microprocessor internal Interrupt enable flip-flop has been re-enabled through software).

After the initialization sequence, IRQ0 has the highest priority and IRQ7 the lowest. Priorities can be changed, as will be explained, in the rotating priority mode.

9.6.2 THE SPECIAL FULLY NESTED MODE

This mode will be used in the case of a big system where cascading is used, and the priority has to be conserved within each slave. In this case the special fully nested mode will be programmed to the master (using ICW4). This mode is similar to the normal nested mode with the following exceptions:

1. When an interrupt request from a certain slave is in service, this slave is not locked out from the master's priority logic and further interrupt requests from higher priority IRQ's within the slave will be recognized by the master and will initiate interrupts to the processor. (In the normal nested mode a slave is masked out when its request is in service and no higher requests from the same slave can be serviced.)
2. When exiting the Interrupt Service routine the software has to check whether the interrupt serviced was the only one from that slave. This is done by sending a non-specific End of Interrupt (EOI) command to the slave and then reading its In-Service register and checking for zero. If it is empty, a non-specific EOI can be sent to the master too. If not, no EOI should be sent.

9.6.3 AUTOMATIC ROTATION (EQUAL PRIORITY DEVICES)

In some applications there are a number of interrupting devices of equal priority. Automatic rotation mode provides for a sequential 8-way rotation. In this mode a device receives the lowest priority after being serviced. In the worst case, a device requesting an interrupt will have to wait until each of seven other devices are serviced at most once. Figure 21 shows an example of automatic rotation.

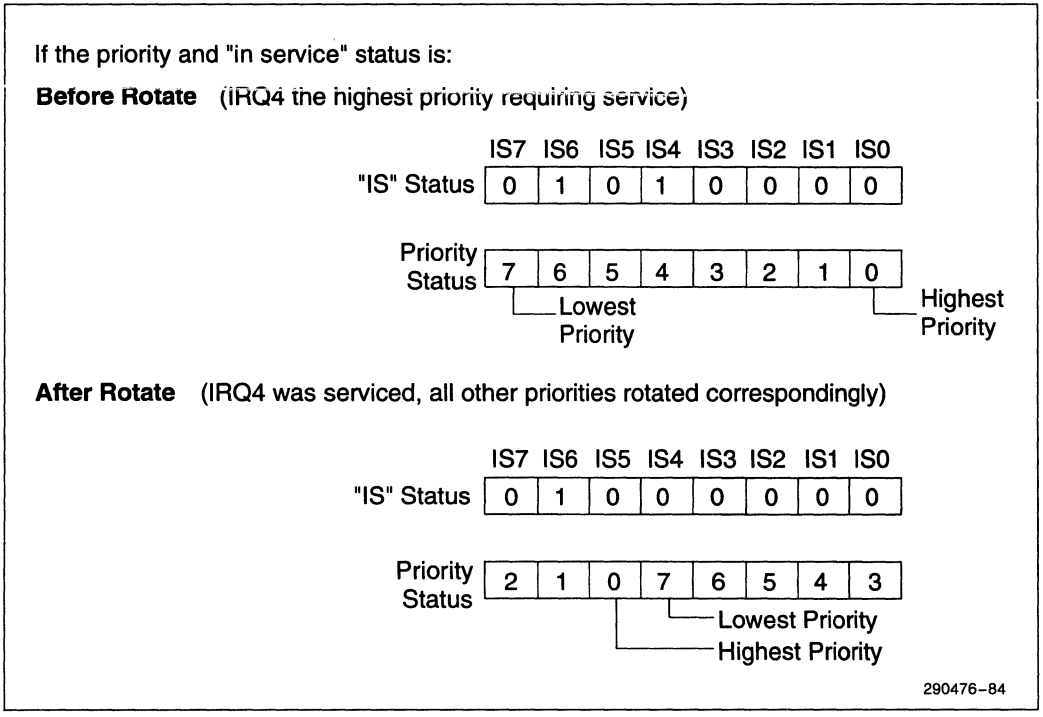


Figure 21. Automatic Rotation Mode Example

There are two ways to accomplish Automatic Rotation using OCW2, the Rotation on Non-Specific EOI Command (R=1, SL=0, EOI=1) and the Rotate in Automatic EOI Mode which is set by (R=1, SL=0, EOI=0) and cleared by (R=0, SL=0, EOI=0).

9.6.4 SPECIFIC ROTATION (SPECIFIC PRIORITY)

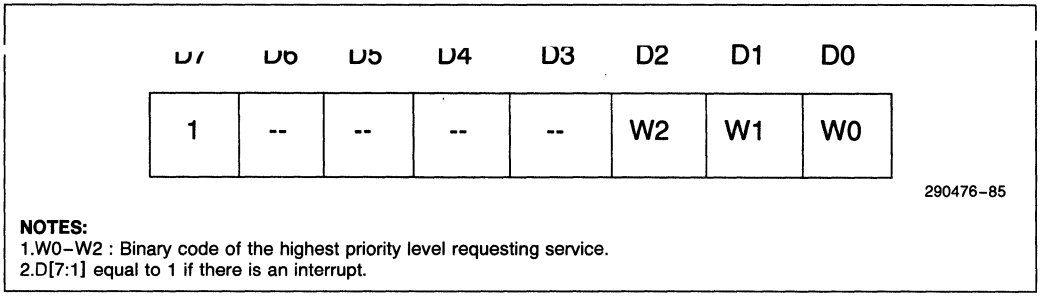
The programmer can change priorities by programming the bottom priority and thus fixing all other priorities. For example, if IRQ5 is programmed as the bottom priority device, then IRQ6 will be the highest priority device. The Set Priority command is issued in OCW2 where: R=1, SL=1; L0–L2 is the binary priority level code of the bottom priority device. See the register description for the bit definitions.

Note that in this mode internal status is updated by software control during OCW2. However, it is independent of the End of Interrupt (EOI) command (also executed by OCW2). Priority changes can be executed during an EOI command by using the Rotate on Specific EOI command in OCW2 (R=1, SL=1, EOI=1 and L0–L2=IRQ level to receive bottom priority).

9.6.5 POLL COMMAND

The Polled Mode can be used to conserve space in the interrupt vector table. Multiple interrupts that can be serviced by one interrupt service routine do not need separate vectors if the service routine uses the poll command. The Polled Mode can also be used to expand the number of interrupts. The polling interrupt service routine can call the appropriate service routine, instead of providing the interrupt vectors in the vector Table. In this mode the INTR output is not used and the microprocessor internal Interrupt Enable flip-flop is reset, disabling its interrupt input. Service to devices is achieved by software using a Poll command.

The Poll command is issued by setting P=1 in OCW3. The Interrupt Controller treats the next I/O read pulse to the Interrupt Controller as an interrupt acknowledge, sets the appropriate IS bit if there is a request, and reads the priority level. Interrupts are frozen from the I/O write to the I/O read. The word enabled onto the data bus during I/O read is shown in Figure 22.



NOTES:
1.W0–W2 : Binary code of the highest priority level requesting service.
2.D[7:1] equal to 1 if there is an interrupt.

Figure 22. Polled Mode

This mode is useful if there is a routine command common to several levels so that the INTA# sequence is not needed (saves ROM space).

9.6.6 CASCADE MODE

The Interrupt Controllers in the ESC system are interconnected in a cascade configuration with one master and one slave. This configuration can handle up to 15 separate priority levels.

The master controls the slaves through a three line internal cascade bus. When the master drives 010b on the cascade bus, this bus acts like a chip select to the slave controller.

In a cascade configuration, the slave interrupt outputs are connected to the master interrupt request inputs. When a slave request line is activated and afterwards acknowledged, the master will enable the corresponding slave to release the interrupt vector address during the second INTA# cycle of the interrupt acknowledge sequence.

Each Interrupt Controller in the cascaded system must follow a separate initialization sequence and can be programmed to work in a different mode. An EOI command must be issued twice: once for the master and once for the slave.

9.6.7 EDGE AND LEVEL TRIGGERED MODES

There are two ELCR registers, one for each 82C59 bank. They are located at I/O ports 04D0h (for the Master Bank, IRQ[0:1,3:7]) and 04D1h (for the Slave Bank, IRQ[8#:15]). They allow the edge and level sense selection to be made on an interrupt by interrupt basis instead of on a complete bank. Only the interrupts that connect to the EISA bus may be programmed for level sensitivity. That is IRQ (0,1,2,8#,13) must be programmed for edge sensitive operation. The LTIM bit is disabled in the ESC. The default programming is equivalent to programming the LTIM bit (ICW1 bit 3) to a 0.

If an ELCR bit is equal to "0", an interrupt request will be recognized by a low to high transition on the corresponding IRQ input. The IRQ input can remain high without generating another interrupt.

If an ELCR bit is equal to "1", an interrupt request will be recognized by a "low" level on the corresponding IRQ input, and there is no need for an edge detection. For level triggered interrupt mode, the interrupt request signal must be removed before the EOI command is issued or the CPU interrupt must be disabled. This is necessary to prevent a second interrupt from occurring.

In both the edge and level triggered modes the IRQ inputs must remain active until after the falling edge of the first INTA#. If the IRQ input goes inactive before this time a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt. This can be a useful safeguard for detecting interrupts caused by spurious noise glitches on the IRQ inputs. To implement this feature the IRQ7 routine is used for "clean up" simply executing a return instruction, thus ignoring the interrupt. If IRQ7 is needed for other purposes a default IRQ7 can still be detected by reading the ISR. A normal IRQ7 interrupt will set the corresponding ISR bit, a default IRQ7 won't. If a default IRQ7 routine occurs during a normal IRQ7 routine, however, the ISR will remain set. In this case it is necessary to keep track of whether or not the IRQ7 routine was previously entered. If another IRQ7 occurs it is a default.

IRQ13 still appears externally to be an edge sensitive interrupt even though it is shared internally with the Chaining interrupt. The Chaining interrupt is ORed after the edge sense logic.

9.7 Register Functionality

For a detailed description of the Interrupt Controller register set, please see Section 3.4, Interrupt Controller Register.

9.7.1 INITIALIZATION COMMAND WORDS

Four initialization command words (ICWs) are used to initialize each interrupt controller. Each controller is initialized separately. Following this initialization sequence, the interrupt controller is ready to accept interrupts.

9.7.2 OPERATION CONTROL WORDS (OCWS)

After the Initialization Command Words (ICWs) are programmed into the Interrupt Controller, the chip is ready to accept interrupt requests at its input lines. However, Interrupt Controller operation can be dynamically modified to fit specific software/hardware expectations. Different modes of operation are dynamically selected following initialization through the use of Operation Command Words (OCWs).

9.8 Interrupt Masks

9.8.1 MASKING ON AN INDIVIDUAL INTERRUPT REQUEST BASIS

Each Interrupt Request input can be masked individually by the Interrupt Mask register (IMR). This register is programmed through OCW1. Each bit in the IMR masks one interrupt channel if it is set to a "1". Bit 0 masks IRQ0, bit 1 masks IRQ1 and so forth. Masking an IRQ channel does not effect the other channel's operation, with one notable exception. Masking IRQ[2] on CNTRL-1 will mask off all requests for service from CNTRL-2. The CNTRL-2 INTR output is physically connected to the CNTRL-1 IRQ[2] input.

9.8.2 SPECIAL MASK MODE

Some applications may require an interrupt service routine to dynamically alter the system priority structure during its execution under software control. For example, the routine may wish to inhibit lower priority requests for a portion of its execution but enable some of them for another portion.

The difficulty here is that if an Interrupt Request is acknowledged and an End of Interrupt command did not reset its IS bit (i.e., while executing a service routine), the Interrupt Controller would have inhibited all lower priority requests with no easy way for the routine to enable them.

The Special Mask Mode enables all interrupts not masked by a bit set in the Mask Register. Interrupt service routines that require dynamic alteration of interrupt priorities can take advantage of the Special Mask Mode. For example, a service routine can inhibit lower priority requests during a part of the interrupt service, then enable some of them during another part.

In the Special Mask Mode, when a mask bit is set in OCW1, it inhibits further interrupts at that level and enables interrupts from all other levels (lower as well as higher) that are not masked.

Thus, any interrupts may be selectively enabled by loading the Mask register with the appropriate pattern.

Without Special Mask Mode, if an interrupt service routine acknowledges an interrupt without issuing an EOI to clear the IS bit, the interrupt controller inhibits all lower priority requests. The Special Mask Mode provides an easy way for the interrupt service routine to selectively enable only the interrupts needed by loading the Mask register.

The special Mask Mode is set by OCW3 where: SSMM = 1, SMM = 1, and cleared where SSMM = 1, SMM = 0.

9.9 Reading The Interrupt Controller Status

The input status of several internal registers can be read to update the user information on the system. The Interrupt Request Register (IRR) and In-Service Register (ISR) can be read via OCW3, as discussed in Section 3.7. The Interrupt Mask Register (IMR) is read via a read of OCW1, as discussed in Section 3.7. Here are brief descriptions of the ISR, the IRR, and the IMR.

Interrupt Request Register (IRR): 8-bit register which contains the status of each interrupt request line. Bits that are clear indicate interrupts that have not requested service. The Interrupt Controller clears the IRR's highest priority bit during an interrupt acknowledge cycle. (Not effected by IMR).

In-Service Register (ISR): 8-bit register indicating the priority levels currently receiving service. Bits that are set indicate interrupts that have been acknowledged and their interrupt service routine started. Bits that are cleared indicate interrupt requests that have not been acknowledged, or interrupt request lines that have not been asserted. Only the highest priority interrupt service routine executes at any time. The lower priority interrupt services are suspended while higher priority interrupts are serviced. The ISR is updated when an End of Interrupt Command is issued.

Interrupt Mask Register (IMR): 8-bit register indicating which interrupt request lines are masked.

The IRR can be read when, prior to the I/O read cycle, a Read Register Command is issued with OCW3 (RR = 1, RIS = 0).

The ISR can be read when, prior to the I/O read cycle, a Read Register Command is issued with OCW3 (RR = 1, RIS = 1).

The interrupt controller retains the ISR/IRR status read selection following each write to OCW3. Therefore, there is no need to write an OCW3 before every status read operation, as long as the current status read corresponds to the previously selected register. For example, if the ISR is selected for status read by an OCW3 write, the ISR can be read over and over again without writing to OCW3 again. However, to read the IRR, OCW3 will have to be reprogrammed for this status read prior to the OCW3 read to check the IRR. This is not true when poll mode is used. Polling Mode overrides status read when P = 1, RR = 1 in OCW3.

After initialization, the Interrupt Controller is set to read the IRR.

As stated, OCW1 is used for reading the IMR. The output data bus will contain the IMR status whenever I/O read is active. The address is 021h or 061h (OCW1).

9.10 Non-Maskable Interrupt (NMI)

An NMI is an interrupt requiring immediate attention and has priority over the normal interrupt lines (IRQx). The ESC indicates error conditions by generating a non-maskable interrupt.

The ESC generates NMI interrupts based on the following Hardware and Software events.

Hardware Events:

1. **Motherboard Parity Errors:** Memory parity errors for the motherboard memory. These errors are reported to the ESC through the PERR# signal line.
2. **System Errors:** System error on the motherboard. The system board uses the SERR# signal to indicate system errors to the ESC.
3. **Add-In Board Parity Errors:** Parity errors on the add-in memory boards on the EISA expansion bus. IOCHK# signal on the EISA bus is driven low by the add-in board logic when this error occurs.
4. **Fail-Safe Timer Timeout:** Fail-Safe Timer (Timer 2, Counter 0) count expires. If this counter has been set and enabled, and the count expires before a software routine can reset the counter.
5. **Bus Timeout:** An EISA bus Master or Slave exceeds the allocated time on the bus. A bus timeout occurs if an EISA Master does not relinquish the bus (MREQ# negated) within 64 BCLKS after it has been preempted (MACK# negated). A bus timeout also occurs if a memory slave extends the cycle (CHRDY negated) long enough to keep CMD# asserted for more than 256 BCLKS. The DMA controller does not cause a bus timeout. The ESC asserts RESDRV when a bus timeout occurs.

Software Events:

1. **Software Generated NMI: If an I/O Write access to Port 0462h occurs. The data value for this write is a don't care.**

The NMI logic incorporates four different 8-bit registers. These registers are used by the CPU to determine the source of the interrupt and to enable or disable/clear the interrupts. See Section 3.4, Interrupt Controller Registers, for the register details.

Table 27. NMI Register I/O Address Map

I/O Port Address	Register Description
0061h	NMI Status Register
0070h	NMI Enable Register
0461h	Extended NMI Register
0462h	Software NMI Register

Table 28. NMI Source Enable/Disable And Status Port Bits

NMI Source	IO Port Bit for Status Reads	IO Port Bit for Enable/Disable
PERR#	Port 0061h, Bit 7	Port 0061h, Bit 2
IOCHK#	Port 0061h, Bit 6	Port 0061h, Bit 3
Fail-Safe	Port 0461h, Bit 7	Port 0461h, Bit 2
Bus Timeout	Port 0461h, Bit 6	Port 0461h, Bit 3
Write to Port 0462h	Port 0461h, Bit 5	Port 0461h, Bit 1

The individual enable/disable bits clear the NMI detect flip-flops when disabled.

All NMI sources can be enabled or disabled by setting Port 070h bit[7]. This disable function does not clear the NMI detect Flip-Flops. This means, if NMI is disabled then enabled via Port 070h, then an NMI will occur when Port 070h is re-enabled if one of the NMI detect Flip-Flops had been previously set.

To ensure that all NMI requests are serviced, the NMI service routine software needs to incorporate a few very specific requirements. These requirements are due to the edge detect circuitry of the host microprocessor, 80386 or 80486. The software flow would need to be the following:

1. NMI is detected by the processor on the rising edge of the NMI input.
2. The processor will read the status stored in port 061h and 0461h to determine what sources caused the NMI. The processor may then reset the register bits controlling the sources that it has determined to be active. Between the time the processor reads the NMI sources and resets them, an NMI may have been generated by another source. The level of NMI will then remain active. This new NMI source will not be recognized by the processor because there was no edge on NMI.
3. The processor must then disable all NMI's by writing bit[7] of port 070H high and then enable all NMI's by writing bit[7] of port 070H low. This will cause the NMI output to transition low then high if there are any pending NMI sources. The CPU's NMI input logic will then register a new NMI.

10.0 ADVANCED PROGRAMMABLE INTERRUPT CONTROLLER (APIC)

In addition to the standard EISA compatible interrupt controller described in the previous section, the ESC incorporates the Advanced Programmable Interrupt Controller (APIC). While the standard interrupt controller is intended for use in a uni-processor system, APIC can be used in either a uni-processor or multi-processor system. APIC provides multi-processor interrupt management and incorporates both static and dynamic symmetric interrupt distribution across all processors. In systems with multiple I/O subsystems, each subsystem can have its own set of interrupts.

In a uni-processor system, APIC's dedicated interrupt bus can reduce interrupt latency over the standard interrupt controller (i.e., the latency associated with the propagation of the interrupt acknowledge cycle across multiple busses using the standard interrupt controller approach). Interrupts can be controlled by the standard EISA compatible interrupt controller unit, the I/O APIC unit, or mixed mode where both the standard and I/O APIC are used. The selection of which controller responds to an interrupt is determined by how the interrupt controllers are programmed. Note that it is the programmer's responsibility to make sure that the same interrupt input signal is not handled by both interrupt controllers.

At the system level, APIC consists of two parts (Figure 23)—one residing in the I/O subsystem (called the I/O APIC) and the other in the CPU (called the Local APIC). The ESC contains an I/O APIC unit. The local APIC and the I/O APIC communicate over a dedicated APIC bus. The ESC's I/O APIC bus interface consists of two bi-directional data signals (APICD[1:0]) and a clock input (APICCLK).

The CPU's Local APIC Unit contains the necessary intelligence to determine whether or not its processor should accept interrupts broadcast on the APIC bus. The Local Unit also provides local pending of interrupts, nesting and masking of interrupts, and handles all interactions with its local processor (e.g., the INTR/INTA/EOI protocol). The Local Unit further provides inter-processor interrupts and a timer, to its local processor. The register level interface of a processor to its local APIC is identical for every processor.

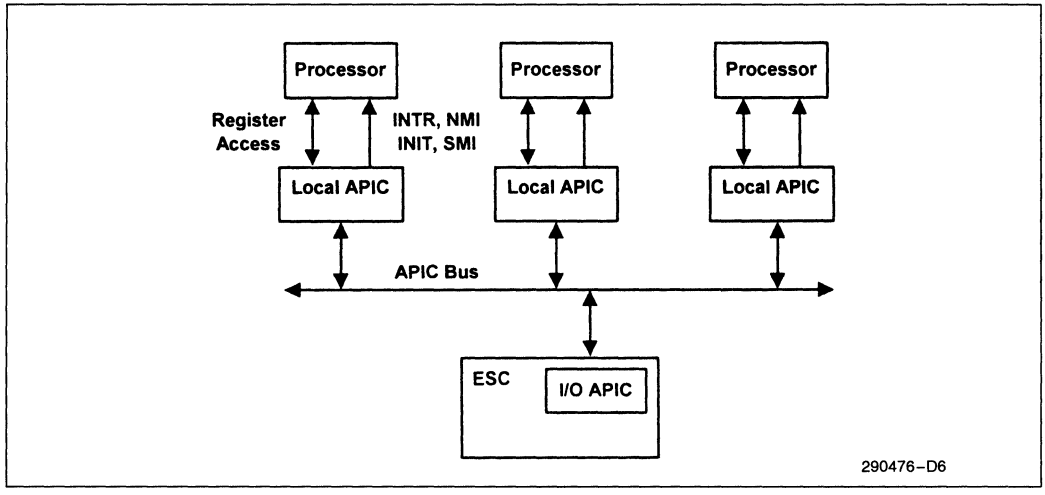


Figure 23. APIC System Structure

The ESC's I/O APIC Unit consists of a set of interrupt input signals, a 16-entry Interrupt Redirection Table, programmable registers, and a message unit for sending and receiving APIC messages over the APIC bus (Figure 24). I/O devices inject interrupts into the system by asserting one of the interrupt lines to the I/O APIC (Figure 25). The I/O APIC selects the corresponding entry in the Redirection Table and uses the information in that entry to format an interrupt request message. Each entry in the Redirection Table can be individually programmed to indicate edge/level sensitive interrupt signals, the interrupt vector and priority, the destination processor, and how the processor is selected (statically or dynamically). The information in the table is used to transmit a message to other APIC units (via the APIC bus).

The ESC's I/O APIC contains a set of programmable registers. Two of the registers (I/O Register Select and I/O Window Registers) are located in the CPU's memory space and are used to indirectly access the other APIC registers as described in Section 3.0, Register Description. The Version Register provides the implementation version of the I/O APIC. The I/O APIC ID Register is programmed with an ID value that serves as a physical name of the I/O APIC. This ID is loaded into the ARB ID Register when the I/O APIC ID Register is written and is used during bus arbitration.

NOTE:

1. When the ESC's I/O APIC receives an interrupt request, the ESC instructs the PCEB to flush its buffers and to request all system buffers pointing to PCI to be flushed (via the AFLUSH# signal). The APIC does not send the interrupt message over the APIC bus until the ESC receives confirmation from the PCEB (via the AFLUSH# signal) that all buffers have been flushed and temporarily disabled.
2. The interrupt number or the vector does not imply a particular priority for being sent. The I/O APIC continually polls the 16 interrupts in a rotating fashion, one at a time. The pending interrupt polled first is the one sent.

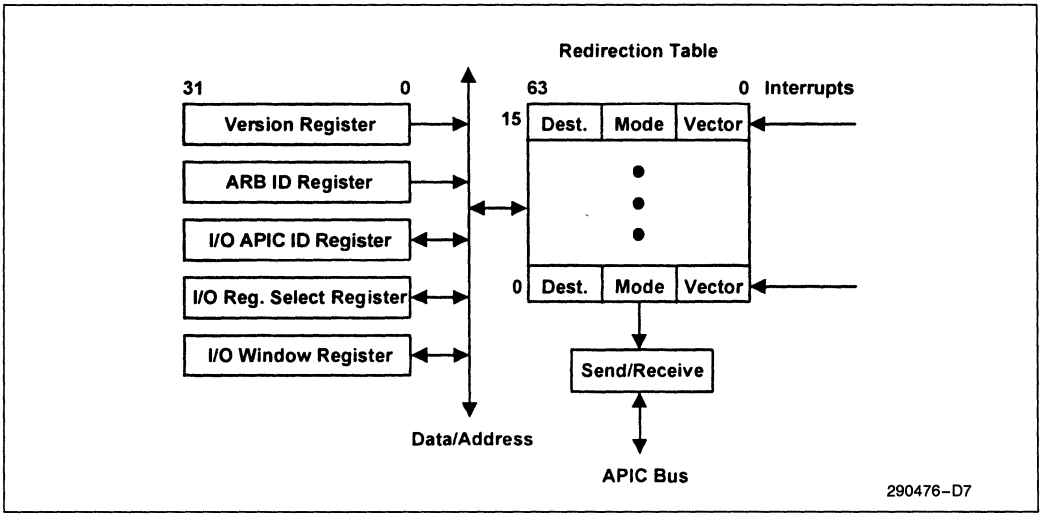


Figure 24. APIC Register Block Diagram

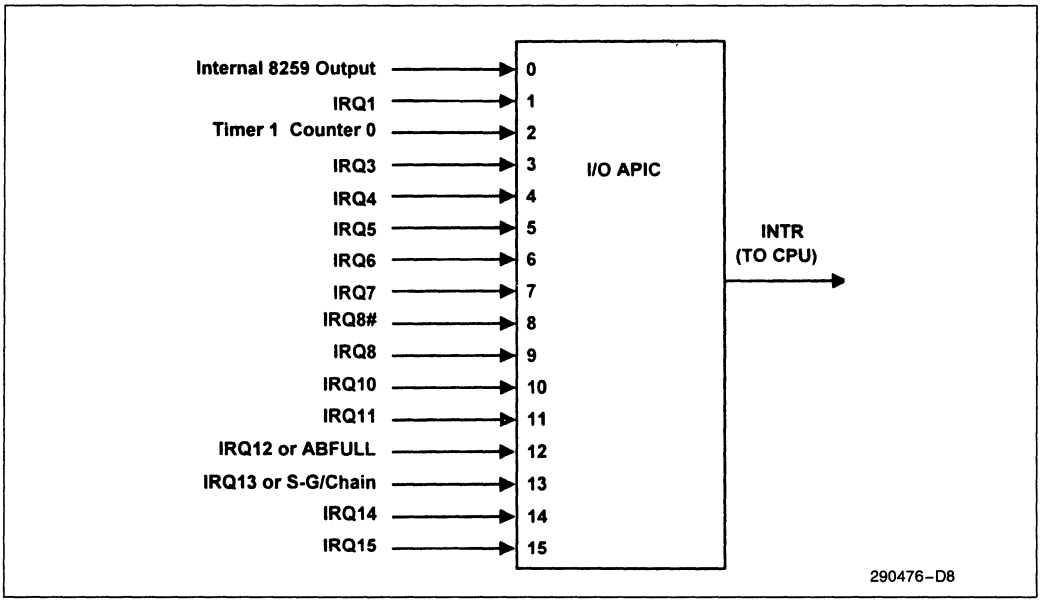


Figure 25. I/O APIC Interrupt Mapping

10.1 Physical Characteristics Of APIC Bus

The APIC bus is a 3-wire synchronous bus connecting all APICs (all I/O units and all local units). Two of these wires are used for data transmission, and one wire is a clock. For bus arbitration, the APIC uses only one of the data wires. The bus is logically a wire-OR and electrically an open-drain connection providing for both message transmission and arbitration for lowest priority. All the values mentioned in the protocol description are logical values (i.e. "Bus Driven" is logical 1 and "Bus Not Driven" is logical 0). The electrical values are 0 for logical one and 1 for logical zero.

10.2 Arbitration For APIC Bus

The APIC uses one wire arbitration to win the bus ownership. A rotating priority scheme is used for arbitration. The winner of the arbitration becomes the lowest priority agent and assumes an arbitration ID of 0. All other agents, except the agent whose arbitration ID is 15, increment their arbitration IDs by one. The agent whose ID was 15 takes the winner's arbitration ID and increments it by one. Arbitration IDs are changed (incremented or assumed) only for messages that are transmitted successfully. For lowest priority messages, the arbitration ID is updated before the final status cycle, which ultimately decides if the message is successful. A message is transmitted successfully if no CS error or acceptance error is reported for that message.

An APIC agent can acquire the bus using two different priority schemes; normal, or EOI (End Of Interrupt). EOI has the highest priority. EOI priority is used to send EOI messages for level interrupts from local APIC to the I/O APIC. When an agent requests the bus with EOI priority, all others requesting the bus with normal priorities back off.

A bus arbitration cycle starts by the agent driving a start cycle (bit 1 = EOI, bit 0 = 1) on the APIC bus (Table 29). Bit 1 = 1 indicates "EOI" priority and bit 1 = 0 indicates normal priority. Bit 0 should be 1.

In cycles 2 through 5, the agent drives the arbitration ID on bit 1 of the bus. High-order ID bits are driven first with successive cycles proceeding to the low bits of the ID. All arbitration losers in a given cycle drop off the bus, using every subsequent cycle as a tie breaker for the previous cycle. When all arbitration cycles are completed, there will be only one agent left driving the bus.

Table 29. Bus Arbitration Cycles

Cycle	Bit 1	Bit 0	Comments
1	EOI	1	0 1 = normal, 1 1 = EOI
2	ArbID3	0	Arbitration ID bits 3 through 0
3	ArbID2	0	
4	ArbID1	0	
5	ArbID0	0	

10.3 Bus Message Formats

After bus arbitration the winner is granted exclusive use of the bus and drives its message on the bus. APIC messages come in four formats—14 cycle EOI Message, 21 cycles Short Message, 33 cycles Lowest Priority Message, and 39 cycles Remote Read Message. All APICs on the APIC bus know the length of an interrupt message by checking the appropriate fields in the message.

EOI Message For Level Triggered Interrupts

The EOI Message is used to send an EOI cycle occurring for a level triggered interrupt from local APIC to the I/O APIC. This cycle contains the priority vector (V[7:0]) of the interrupt. When this message is received, the I/O APIC resets the Remote IRR bit for that interrupt. If the interrupt signal is still active after the RIRR bit is reset, the I/O APIC will treat it as a new interrupt.

Table 30. EOI Message

Cycle	Bit 1	Bit 0	Comments
1	EOI	1	0 1 = normal, 1 1 = EOI
2	ArbID3	0	Arbitration ID bits 3 through 0
3	ArbID2	0	
4	ArbID1	0	
5	ArbID0	0	
6	V7	V6	Interrupt vector V7–V0
7	V5	V4	
8	V3	V2	
9	V1	V0	

Table 30. EOI Message (Continued)

Cycle	Bit 1	Bit 0	Comments
10	C	C	Check Sum
11	0	0	Postamble
12	A	A	Status Cycle0
13	A1	A1	Status Cycle1
14	0	0	Idle

Short Message

Short Messages are used for the delivery of Fixed, NMI, SMI (82374SB only), Reset, ExtINT and Lowest Priority with Focus processor interrupts. The delivery mode bits (M[2:0]) specify the message. All short messages take 21 cycles, including the idle cycle.

Cycle 1 is the start cycle (Table 31). Cycles 2 through 5 are for bus arbitration as described earlier. APIC ID bits are sent on the bus one bit at a time (Only one data bus bit is used). The other bit should be zero. Cycles 6 and 7 provide destination mode and delivery mode bits. Cycle 8 provides level and trigger mode information. Cycles 10 through 13 are the 8-bit interrupt vector. The vector is only defined for delivery modes fixed, and lowest-priority. For delivery mode of "Remote Read", the vector field contains the address of the register to be read remotely.

If Destination Mode (DM) is 0 (physical mode), then cycles 15 and 16 are the APIC ID and cycles 13 and 14 are zero. If DM is 1 (logical mode), then cycles 13 through 16 are the 8-bit destination field. The interpretation of the logical mode 8-bit destination field is performed by the local units using the Destination Format Register. Shorthands of "all-incl-self" and "all-excl-self" both use physical destination mode and a destination field containing APIC ID value of all ones. The sending APIC knows whether it should (incl) or should not (excl) respond to its own message.

Cycle 17 is a checksum for the data in cycles 6 through 16. The (single) APIC driving the message provides this checksum in cycle 17.

Cycle 18 is a postamble cycle driven as 00 by all APICs to perform various internal computations based on the information contained in the received message. One of the computations takes the computed checksum of the data received in cycles 6 through 16 and compares it against the value in cycle 17. If any APIC computes a different checksum than the one passed in cycle 17, then that APIC signals an error on the APIC bus in cycle 19 by driving it as 11. If this happens, all APICs assume the message was never sent and the sender must try sending the message again, which includes re-arbitrating for the APIC bus. In lowest priority delivery when the interrupt has a focus processor, the focus processor signals this by driving 10 during cycle 19. This tells all the other APICs that the interrupt has been accepted, the LP arbitration is preempted, and short message format is used. Cycle 19 and 20 indicates the status of the message (i.e. accepted, check sum error, retry or error). Table 32 shows the status signals combinations and their meanings for all delivery modes.

The checksum is calculated iteratively on each cycle by adding the following terms:

1. The two least significant bits from the last cycle's checksum.
2. The current two data bits.
3. The carry bit from the last cycle's checksum shifted to the least significant bit.

Note that, at the beginning of the calculation, the three bits composing the previous cycle's checksum (two lower bits and carry) are zero.

Table 31. Short Message

Cycle	Bit 1	Bit 0	Comments
1	0	1	0 1 = Normal, 1 1 = EOI
2	ArbID3	0	Arbitration ID bits 3 through 0
3	ArbID2	0	
4	ArbID1	0	
5	ArbID0	0	
6	DM	M2	DM = Destination Mode
7	M1	M0	M2–M0 = Delivery Mode
8	L	TM	L = Level, TM = Trigger Mode
9	V7	V6	V7–V0 = Interrupt Vector
10	V5	V4	
11	V3	V2	
12	V1	V0	
13	D7	D6	Destination
14	D5	D4	
15	D3	D2	
16	D1	D0	
17	C	C	Checksum for Cycles 6–16
18	0	0	Postamble
19	A	A	Status Cycle 0
20	A1	A1	Status Cycle 1
21	0	0	Idle

Table 32. APIC Bus Message Status Information

Delivery Mode	Focus Processor	Status: A A	Comments	Status: A1 A1	Comments
Fixed, EOI	N/A	00	CS is OK	10	Accepted
Fixed, EOI	N/A	00	CS is OK	11	Retry
Fixed, EOI	N/A	00	CS is OK	0X	Error
Fixed, EOI	N/A	11	CS Error	XX	
Fixed, EOI	N/A	10	Error	XX	
Fixed, EOI	N/A	01	Error	XX	
NMI, SMM, Reset, ExtINT	N/A	00	CS is OK	10	Accepted
NMI, SMM, Reset, ExtINT	N/A	00	CS is OK	11	Error
NMI, SMM, Reset, ExtINT	N/A	00	CS is OK	0X	Error
NMI, SMM, Reset, ExtINT	N/A	11	CS Error	XX	
NMI, SMM, Reset, ExtINT	N/A	10	CS Error	XX	
NMI, SMM, Reset, ExtINT	N/A	01	CS Error	XX	
Lowest Priority	No	00	CS is OK. No Focus Processor	11	Go for LP Arb.
Lowest Priority	No	00	CS is OK. No Focus Processor	10	End and Retry
Lowest Priority	No	00	CS is OK. No Focus Processor	0X	Error
Lowest Priority	Yes	10	CS is OK. Focus Processor	XX	
Lowest Priority	Yes	11	CS Error	XX	
Lowest Priority	Yes	01	Error	XX	
Remote Read	N/A	00	CS is OK	XX	
Remote Read	N/A	11	CS Error	XX	
Remote Read	N/A	01	Error	XX	
Remote Read	N/A	10	Error	XX	

NOTE:
CS = Check Sum

Lowest Priority (LP) without Focus Processor (FP) Message

This message format is used to deliver an interrupt in the lowest priority mode in which it does not have a Focus Processor. Cycles 1 through 20 for this message are the same as for the Short Message discussed above. Status cycle 19 identifies if there is not a Focus processor (00) and a status value of 11 in cycle 20 indicates the need for lowest priority arbitration.

Cycle 21 through 28 are used to arbitrate for the lowest priority processor. The processors that take part in the arbitration drive their processor priority on the bus. Only the local APICs that have "free interrupt slots" participate in the lowest priority arbitration.

Cycle 29 through 32 are used to break a tie in case two or more processors have lowest priority. The bus arbitration IDs are used to break the tie.

Cycle 33 is an additional status cycle driven by the accepting local APIC. By receiving 10 during this cycle, the sending I/O APIC knows that there was a local APIC left after LP arbitration. Otherwise, the message is retried. Cycle 34 is an idle cycle.

Table 33. Lowest Priority without Focus Processor Message

Cycle	Bit 1	Bit 0	
1	0	1	0 1 = normal, 1 1 = EOI
2	ArbID3	0	Arbitration ID bits 3 through 0
3	ArbID2	0	
4	ArbID1	0	
5	ArbID0	0	
6	DM	M2	DM = Destination mode
7	M1	M0	M2–M0 = Delivery mode
8	L	TM	L = Level, TM = Trigger Mode
9	V7	V6	V7–V0 = Interrupt Vector
10	V5	V4	
11	V3	V2	
12	V1	V0	
13	D7	D6	Destination
14	D5	D4	
15	D3	D2	
16	D1	D0	
17	C	C	Checksum for cycles 6-16
18	0	0	Postamble
19	A	A	Status cycle 0
20	A1	A1	Status cycle 1

Table 33. Lowest Priority without Focus Processor Message (Continued)

Cycle	Bit 1	Bit 0	
21	P7	0	Inverted Processor Priority P7–P0
22	P6	0	
23	P5	0	
24	P4	0	
25	P3	0	
26	P2	0	
27	P1	0	
28	P0	0	
29	ArbID3	0	Arbitration ID 3 -0
30	ArbID2	0	
31	ArbID1	0	
32	ArbID0	0	
33	S	S	Status (10 means status OK; all other values indicate an error)
34	0	0	Idle

Remote Read Message

The Remote Read Message (Table 34) is used for a local APIC to read the register in another local APIC. The message format is the same as the Short Message for the first 20 cycles.

Cycles 21 through 36 contain the Remote Register data. The status information in cycle 37 specifies if the data is good or not. The Remote Read Message is always successful in that it is never retried (although the data may be valid or invalid). The reason is that the Remote Read Message is a debug feature, and a "hung" remote APIC that is unable to respond should not cause the debugger to hang up.

Table 34. Remote Read Message

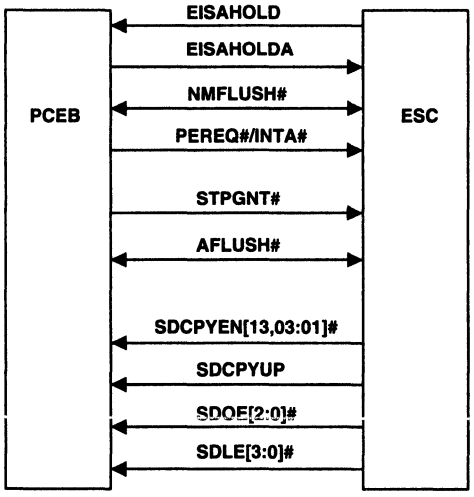
Cycle	Bit 1	Bit 0	
1	0	1	0 1 = normal, 1 1 = EOI
2	ArbID3	0	Arbitration ID bits 3 through 0
3	ArbID2	0	
4	ArbID1	0	
5	ArbID0	0	
6	DM	M2	DM = Destination mode
7	M1	M0	M2–M0 = Delivery mode
8	L	TM	L = Level, TM = Trigger Mode

Table 34. Remote Read Message (Continued)

Cycle	Bit 1	Bit 0	
9	V7	V6	V7–V0 = Interrupt Vector
10	V5	V4	
11	V3	V2	
12	V1	V0	
13	D7	D6	Destination
14	D5	D4	
15	D3	D2	
16	D1	D0	
17	C	C	Checksum for cycles 6-16
18	0	0	Postamble
19	A	A	Status cycle 0
20	A1	A1	Status cycle 1
21	d31	d30	Remote register data 31-0
22	d29	d28	
23	d27	d26	
24	d25	d24	
25	d23	d22	
26	d21	d20	
27	d19	d18	
28	d17	d16	
29	d15	d14	
30	d13	d12	
31	d11	d10	
32	d09	d08	
33	d07	d06	
34	d05	d04	
35	d03	d02	
36	d01	d00	
37	S	S	Data Status: 11 = valid, 00 = invalid
38	C	C	Check Sum for data d[31:00]
39	0	0	Idle

11.0 PCEB/ESC INTERFACE

The PCEB/ESC interface (Figure 26) provides the inter-chip communications between the PCEB and ESC. The interface provides control information between the two components for PCI/EISA arbitration, data size translations (controlling the PCEB's EISA data swap logic), and interrupt acknowledge cycles.



NOTE:
STPGNT# is only on the 82374SB (and companion 82375SB).

Figure 26. PCEB/ESC Interface Signals

11.1 Arbitration Control Signals

The PCEB contains the arbitration circuitry for the PCI Bus and the ESC contains the arbitration circuitry for the EISA Bus. The PCEB/ESC Interface contains a set of arbitration control signals (EISAHOLD, EISAHOLDA, NMFLUSH#, and PEREQ#/INTA#) that synchronize bus arbitration and ownership changes between the two bus environments. The signals also force PCI device data buffer flushing, if needed, to maintain data coherency during EISA Bus ownership changes.

The PCEB is the default owner of the EISA Bus. If another EISA/ISA master or DMA wants to use the bus, the ESC asserts EISAHOLD to instruct the PCEB to relinquish EISA Bus ownership. The PCEB completes any current EISA Bus transaction, tri-states its EISA Bus signals, and asserts EISAHOLDA to inform the ESC that the PCEB is off the bus.

For ownership changes, other than for a refresh cycle, the ESC asserts the NMFLUSH# signal to the PCEB (for one PCICLK) to instruct the PCEB to flush its Line Buffers pointing to the PCI Bus. The assertion of NMFLUSH# also instructs the PCEB to initiate flushing and to temporarily disable system buffers on the PCI Bus (via MEMREQ#, MEMACK, and FLSHREQ#). The buffer flushing maintains data coherency, in the event that the new EISA Bus master wants to access the PCI Bus. Buffer flushing also prevents dead-lock conditions between the PCI Bus and EISA Bus. Since the ESC/PCEB does not know ahead of time, whether the new master is going to access the PCI Bus or a device on the EISA Bus, buffers pointing to the PCI Bus are always flushed when there is a change of EISA Bus ownership, except for refresh cycles. For refresh cycles, the ESC

controls the cycle and, thus, knows that the cycle is not an access to the PCI Bus and does not initiate a flush request to the PCEB. After a refresh cycle, the ESC always surrenders control of the EISA Bus back to the PCEB.

NMFLUSH# is a bi-directional signal that is negated by the ESC when buffer flushing is not being requested. The ESC asserts NMFLUSH# to request buffer flushing. When the PCEB samples NMFLUSH# asserted, it starts driving the signal in the asserted state and begins the buffer flushing process. (The ESC tristates NMFLUSH# after asserting if for the initial 1 PCICLK period.) The PCEB keeps NMFLUSH# asserted until all buffers are flushed and then it negates the signal for 1 PCICLK. When the ESC samples NMFLUSH# negated, it starts driving the signal in the negated state, completing the handshake. When the ESC samples NMFLUSH# negated, it grants ownership to the winner of the EISA Bus arbitration (at the time NMFLUSH# was negated). Note that for a refresh cycle, NMFLUSH# is not asserted by the ESC.

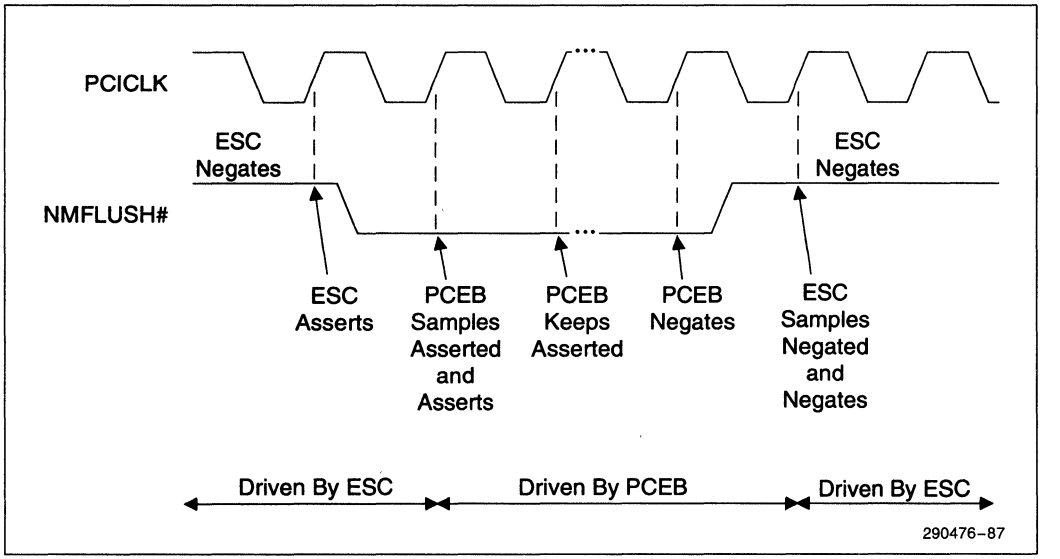


Figure 27. NMFLUSH# Protocol

When the EISA master completes its transfer and gets off the bus (i.e., removes its request to the ESC), the ESC negates EISAHOLD and the PCEB, in turn, negates EISAHOLDA. At this point, the PCEB resumes its default ownership of the EISA Bus.

If a PCI master requests access to the EISA Bus while the bus is owned by a master other than the PCEB, the PCEB retries the PCI cycle and requests ownership of the EISA Bus by asserting PEREQ#/INTA# to the ESC. PEREQ#/INTA# is a dual function signal that is a PCEB request for the EISA Bus (PEREQ# function) when EISAHOLDA is asserted. In response to the PCEB request for EISA Bus ownership, the ESC removes the grant to the EISA master. When the EISA master completes its current transactions and relinquishes the bus (removes its bus request), the ESC negates EISAHOLD and the PCEB, in turn, negates EISAHOLDA. At this point, a grant can be given to the PCI device for a transfer to the EISA Bus. Note that the INTA# function of the PEREQ#/INTA# signal is described in Section 11.5, Interrupt Acknowledge Control.

11.2 System Buffer Coherency Control—APIC

During an interrupt sequence, the system buffers must be flushed before the ESC's I/O APIC can send an interrupt message to the local APIC (CPU's APIC). The ESC and PCEB maintain buffer coherency when the ESC receives an interrupt request for its I/O APIC using the AFLUSH# signal.

11.3 Power Management

In response to the ESC's STPCLK# assertion, the CPU sends out a stop grant bus cycle to indicate that it has entered the stop grant state. The PCEB informs the ESC of the stop grant cycle using the STPGNT# signal.

11.4 EISA Data Swap Buffer Control Signals

The cycles in the EISA environment may require data size translations before the data can be transferred to its intermediate or final destination. As an example, a 32-bit EISA master write cycle to a 16-bit EISA slave requires a disassembly of a 32-bit Dword into 16 bit Words. Similarly, a 32-bit EISA master read cycle to a 16-bit slave requires an assembly of two 16-bit Words into a 32-bit Dword. The PCEB contains EISA data swap buffers to support data size translations on the EISA Bus. The operation of the data swap logic is described in the PCEB data sheet. The ESC controls the operation of the PCEB's data swap logic with the following PCEB/ESC interface signals. These signals are outputs from the ESC and inputs to the PCEB.

- SDCPYEN[13,03:01]#
- SDCPYUP
- SDOE[2:0]#
- SDLE[3:0]#

Copy Enable Outputs (SDCPYEN[13,3:1]#)

These signals enable the byte copy operations between data byte lanes 0, 1, 2 and 3 as shown in the Table 35. ISA master cycles do not perform assembly/disassembly operations. Thus, these cycles use SDCPYEN[13,03:01]# to perform the byte routing and byte copying between lanes. EISA master cycles however, can have assembly/disassembly operations. These cycles use SDCPYEN[13,03:01]# in conjunction with SDCPYUP and SDLE[3:0]#.

Table 35. Byte Copy Operations

Signal	Copy Between Byte Lanes
SDCPYEN01#	Byte 0 (bits[7:0]) and Byte 1 (bits[15:8])
SDCPYEN02#	Byte 0 (bits[7:0]) and Byte 2 (bits[23:16])
SDCPYEN03#	Byte 0 (bits[7:0]) and Byte 3 (bits[31:24])
SDCPYEN13#	Byte 1 (bits[15:8]) and Byte 3 (bits[31:24])

System Data Copy Up (SDCPYUP)

SDCPYUP controls the direction of the byte copy operations. When SDCPYUP is asserted (high), active lower bytes are copied onto the higher bytes. The direction is reversed when SDCPYUP is negated (low).

System Data Output Enable (SDOE[2:0]#)

These signals enable the output of the data swap buffers onto the EISA Bus (Table 36). SDOE[2:0] are re-drive signals in case of mis-matched cycles between EISA to EISA, EISA to ISA, ISA to ISA and the DMA cycles between the devices on EISA.

Table 36. Output Enable Operations

Signal	Byte Lane
SDOE0#	Applies to Byte 0 (bits[7:0])
SDOE1#	Applies to Byte 1 (bits[15:8])
SDOE2#	Applies to Byte 2 and Byte 3 (bits[31:16])

System Data to Internal (PCEB) Data Latch Enables (SDLE[3:0]#)

These signals latch the data from the EISA Bus into the data swap latches. The data is then either sent to the PCI Bus via the PCEB or re-driven onto the EISA Bus. SDLE[3:0]# latch the data from the corresponding EISA Bus byte lanes during PCI reads from EISA, EISA writes to PCI, DMA cycles between an EISA device and the PCEB. These signals also latch data during mismatched cycles between EISA to EISA, EISA to ISA, ISA to ISA, the DMA cycles between the devices on EISA, and any cycles that require copying of bytes, as opposed to copying and assembly/disassembly.

11.5 Interrupt Acknowledge Control

PEREQ#/INTA# (PCI to EISA Request or Interrupt Acknowledge) is a dual function signal and the selected function depends on the status of EISAHLDA. When EISAHLDA is negated, this signal is an interrupt acknowledge (INTA#) and supports interrupt processing. If interrupt acknowledge is enabled via the PCEB's PCICON Register and EISAHOLDA is negated, the PCEB asserts PEREQ#/INTA# when a PCI interrupt acknowledge cycle is being serviced. This informs the ESC that the forwarded EISA I/O read from location 04h is an interrupt acknowledge cycle. Thus, the ESC uses this signal to distinguish between a request for the interrupt vector and a read of the ESC's DMA register located at 04h. The ESC responds to the read request by placing the interrupt vector on SD[7:0].

12.0 INTEGRATED SUPPORT LOGIC

The ESC integrates support logic for assorted functions for a typical EISA system board. The following functions are directly supported by the ESC.

- EISA Address Buffer Control
- Coprocessor Interface
- BIOS Interface
- Keyboard Controller Interface
- Real Time Clock Interface
- Floppy Disk Controller Interface
- Configuration RAM Interface
- X-Bus and IDE Decode

NOTE:
The ESC directly supports X-Bus Floppy Disk Controller and IDE. If the IDE resides on another bus (e.g., ISA or PCI Bus), additional hardware (to modify the X-Bus signals) is required to support the X-Bus Floppy Disk Controller.

12.1 EISA Address Buffer Control

The EISA Bus consists of unlatched addresses (LA[31:2]) and latched addresses (SA[19:2]). EISA devices generate or monitor LA addresses, and ISA devices generate or monitor SA addresses. Three Discrete F543s are used to generate the SA address from LA and LA addresses from SA addresses (Figure 28). The ESC generates the control signals SALE#, LASAOE#, and SALAOE# for the F543s. These signals control the direction of the address flow. For EISA master, DMA, and Refresh cycles the, the LA addresses are generated by the master device, and the SA addresses are driven by the F543s. For ISA master devices, the SA addresses are generated by the master device, and the LA addresses by driven by the F543s.

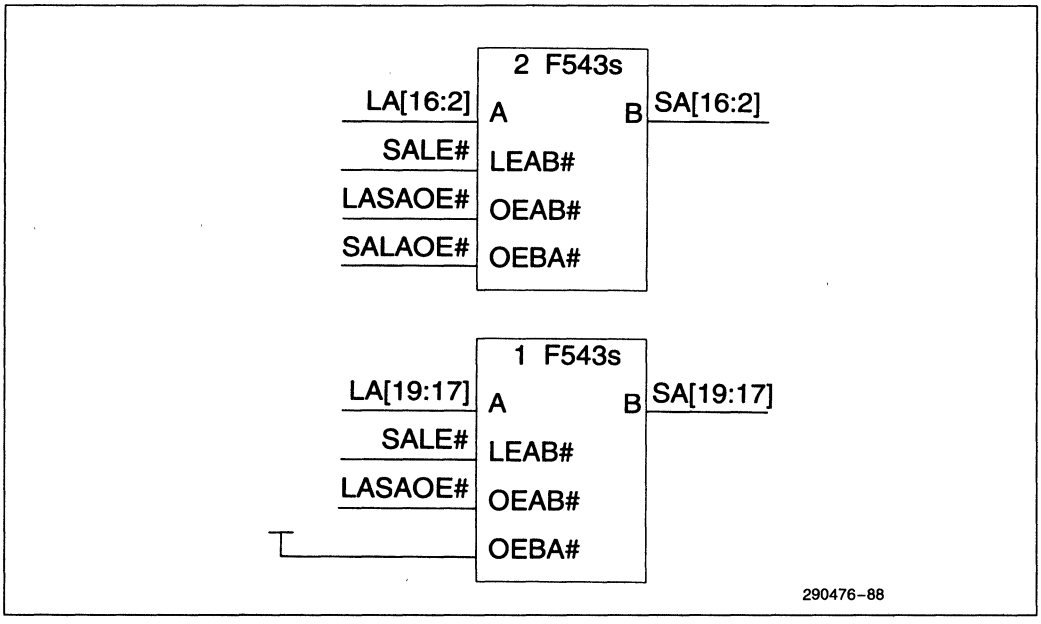


Figure 28. EISA Address Buffers

Table 37. EISA Address Buffer Control Function

Signal		Cycle	Type	
	EISA Master	**ISA Master**	**DMA**	**Refresh**
SALE#	Pulses	Low	Pulses	Pulses
LASAOE#	Low	High	Low	Low
SALAOE#	High	Low	High	High

12.2 Coprocessor Interface

The numeric coprocessor interface is designed to support PC/AT compatible numeric coprocessor exception handling. The EISA Clock Divider configuration register bit 5 needs to be set to a 1 in order to enable the coprocessor error support in the ESC. The coprocessor interface consists of FERR# signal and IGNNE# signal. The FERR# signal and IGNNE# signals are connected directly to the Floating Point Error pin and Ignore Floating Point Error pin of the CPU respectively.

Whenever an error during computation is detected, the CPU asserts the FERR# signal to the ESC. The ESC internally generates an interrupt on the IRQ13 line of the integrated Interrupt Controller. The result is a asserted INTR signal to the CPU.

When the ESC detects an I/O write to the internal port 00F0h, the ESC deasserts the internal IRQ13 line to the integrated Interrupt Controller. At the same time the ESC asserts the IGNNE# signal. The ESC keeps the IGNNE# signal asserted until the FERR# signal is negated by the CPU.

If the coprocessor error support is enabled in the EISA Clock Divider configuration register then the ESC IRQ13 pins cannot be used, and this pin should be tied to ground.

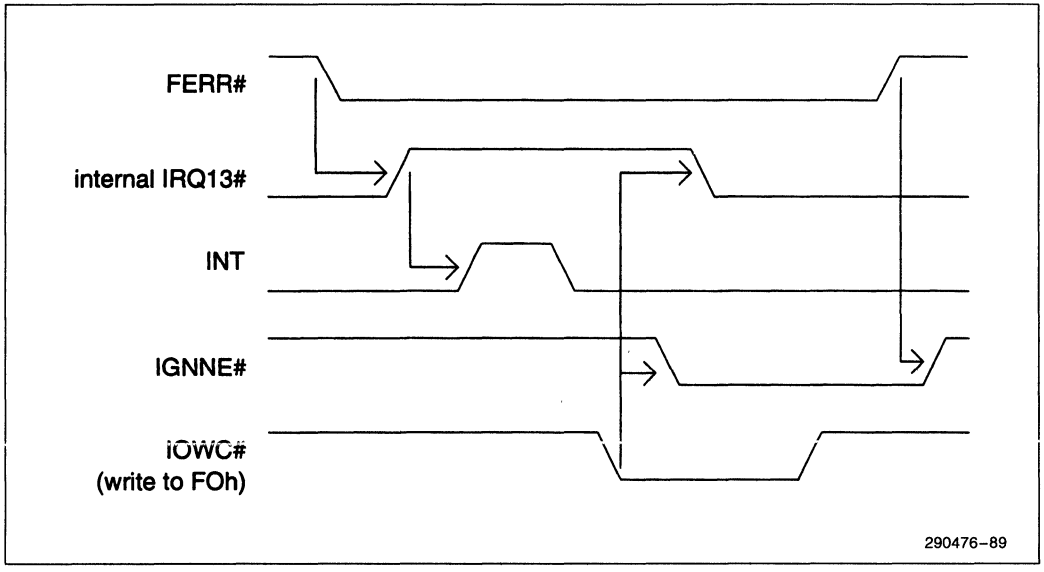


Figure 29. Coprocessor Interface Waveform

12.3 BIOS Interface

The ESC supports a total of 512 KBytes of BIOS memory. The ESC asserts the LBIOSCS# signal for EISA or ISA memory cycles decoded to be in the BIOS space. The 512 KBytes of BIOS includes the conventional 128 KBytes of BIOS and 384 KBytes of enlarged BIOS. The 128 KBytes of conventional BIOS is divided into multiple regions. Each region can be independently enabled or disabled by setting the appropriate bits in the BIOS Chip Select A register and BIOS Chip Select B register. The 128 KBytes of conventional BIOS is also aliased at different locations within the memory space. Refer to Section 4.1, BIOS Memory Space, for details.

The ESC generates the LBIOSCS# signal by internally latching the output of the BIOS address decode with BALE signal. The ESC asserts the LBIOSCS# for all read cycles in the enabled BIOS memory space. The ESC will assert LBIOSCS# signal for write cycles in the enabled BIOS memory space only if the BIOS Chip Select B register bit 3 is set to 1 (BIOS write enable).

12.4 Keyboard Controller Interface

The ESC provides a complete interface to a glueless interface to a 8x42 Keyboard Controller. The ESC Keyboard Controller interface consists of Keyboard Controller Chip Select (KYBDCS#) signal, Mouse interrupt (ABFULL) signal. The ESC also supports the fast Keyboard commands for CPU reset (ALTRST#) and address A20 enable (ALTA20) by integrating Port 92h.

The ESC asserts the KYBDCS# signal for I/O cycles to addresses 60h (82374EB/SB), 62h (82374EB only), 64h (82374EB/SB), and 66h (82374EB only) if the Peripheral Chip Select A register bit 1 is set to 1. The ESC uses the ABFULL signal to internally generate an interrupt request to the integrated Interrupt Controller on the IRQ12 line if EISA Clock Divisor register bit 4 is set to 1 (Mouse Interrupt Enable). A low to high transition on the ABFULL signal is internally latched by the ESC. The high level on this latch remains until a write to I/O port 60h is detected or the ESC is reset.

The ALTRST# is used to reset the CPU under software control. The ESC ALTRST# signal needs to be AND'ed externally with the reset signal from the keyboard controller. A write to the System Control Register (092h) bit 0 to set the bit to a 1 from a 0 causes the ESC to pulse the ALTRST# signal. ALTRST# is asserted for approximately 4 BCLKs. The ESC will not pulse the ALTRST# signal if bit 0 has previously been set to a 1.

12.5 Real Time Clock

The ESC provides a glueless interface for the Real Time Clock in the system. The ESC provides a Real Time Clock Address Latch Enable signal (RTCALE), a Real Time Clock read Strobe(RTCRD#), and a Real Time Clock Write strobe (RTCWR#). The ESC pulses the RTCALE signal asserted for one and a half BCLKs when an I/O write to address 70h is detected. The ESC asserts RTCRD# signal and RTCWR# signal for I/O read and write accesses to address 71h respectively.

The ESC also supports the power on password protection through the Real Time Clock. The power on password protection is enabled by setting the System Control register 092h bit 3 to a 1. The ESC does not assert RTCRD# signal or RTCWR# signal for I/O cycles to 71h if the access are addressed to Real Time Clock addresses (write to 70h) 36h to 3Fh if the power on password protection is enabled.

12.6 Floppy Disk Control Interface

The ESC supports interface to the 82077(SL) floppy disk controller chip. The ESC provides a Floppy Disk Controller Chip Select signal (FDCCS#). The ESC also provides a buffered Drive Interface (DSKCHG#) signal. In addition, the ESC generates the control for the disk light.

The ESC supports both the primary address range (03F0h–03F7h) and secondary address range (0370h–0377h) of the Floppy Disk Controller. The state of Peripheral Chip Select A register bit 5 determines which address range is decoded by the ESC as access to Floppy Disk Controller. If bit 5 is set to 0, the ESC will decode the primary Floppy Disk Controller address range. If bit 5 is set to 1, the ESC will decode the secondary Floppy Disk Controller address range.

The ESC supports the Drive Interface signal. During I/O accesses to address 03F7h (primary) or 0377h (secondary), the ESC drives the inverted state of the DSKCHG# signal on to the SD7 data line. The ESC uses the DSKCHG# signal to determine if the Floppy Disk Controller is present on the X-Bus. If the DSKCHG# signal is samples low during reset, the ESC will disable Floppy Disk Controller support.

The ESC also supports the Disk Light function by generating the DLIGHT# signal. If System Control 092h register bit 6 or bit 7 is set to a 1, the ESC will assert the DLIGHT# signal.

12.7 Configuration RAM Interface

The ESC provides the control signals for 8 Kbytes of external configuration RAM. The configuration RAM is used for storing EISA configuration system parameters. The configuration RAM is I/O mapped between location 0800h–08FFh. Due to the I/O address constraint (256 byte addresses for 8 Kbyte of RAM) , the configuration RAM is organized in 32 pages of 256 bytes each. The I/O port 0C00h is used to store the configuration RAM page address. The ESC integrates this port as Configuration RAM Page register. During a read or a write to the configuration RAM address space 0800h–08FFh, the ESC drives the configuration RAM page address by placing the content of the Configuration RAM Page Address register bits[4:0] on the EISA Address line LA[31:27]#. The ESC will also assert the CRAMRD# signal or the CRAMWR# signal for I/O read and write accesses to I/O address 0800h–08FFh. The ESC will only generate the configuration RAM page address and assert the CRAMRD# signal and CRAMWR# signal if the Peripheral Chip Select B register bit 7 is set to 1.

12.8 General Purpose Peripherals, IDE, Parallel Port, and Serial Port Interface

The ESC provides three dual function pins (GPCS[2:0]#.ECS[2:0]). The functionality of these pins is selected through the configuration Mode Select register bit 4. If Mode Select register bit 4 is set to 0 the general purpose chip select functionality is selected. If Mode Select register bit 4 is set to 1, the encoded chip select functionality is selected.

In general purpose chip select mode, the ESC generates three general purpose chip selects (GPCS[2:0]#). The decode for each general purpose chip selects is programmed through a set of three configuration registers; General Purpose Chip Select x Base Low Address register, General Purpose Chip Select x Base High Address register, and General Purpose Chip Select x Mask register. Each General Purpose Peripheral can be mapped anywhere in the 64 Kbytes of I/O address. The general purpose peripheral address range is programmable from 1 byte to 256 bytes with 2^n granularity.

In encoded chip select mode (ESC[2:0]), in addition to decoding the general purpose chip select 0 address and general purpose chip select 1 address, the ESC also decodes IDE, Parallel Ports, and Serial Ports addresses. The encoded chip select mode requires an external decoder like a F138 to generate the device chip selects from the ESC[2:0] signals.

The ESC generates encoded chip selects for two Serial Ports, COMACS# (ECS[2:0] = 000) and COMBCS# (ESC[2:0] = 001). The ESC supports Serial Port COM1 and Serial Port COM2. Accesses to Serial Port COM1 or Serial Port COM2 are individually programmed through Peripheral Chip Select B register bits[0:3] to generate a encoded chip select for COMACS# or COMBCS#.

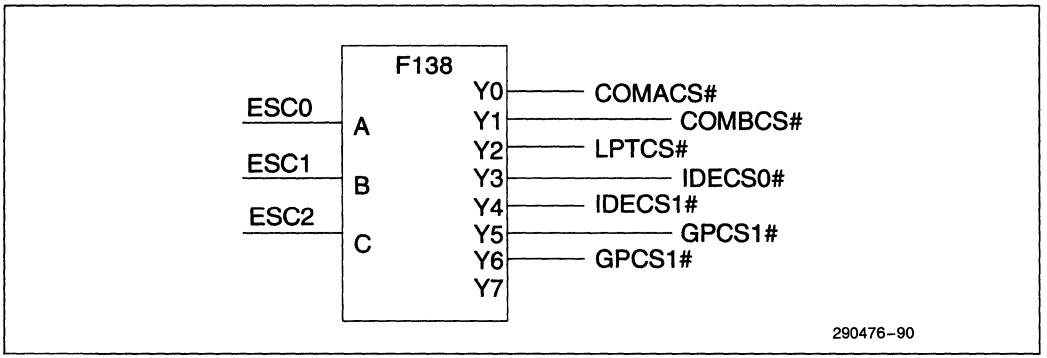


Figure 30. Encoded Chip Select Decoder Logic

Table 38. Encoded Chip Select Decode

ESC2	ESC1	ESC0	PERIPHERAL CS
0	0	0	COMACS#
0	0	1	COMBCS#
0	1	0	LPTCS#
0	1	1	IDECS0#
1	0	0	IDECS1#
1	0	1	GPCS0#
1	1	0	GPCS1#
1	1	1	idle state

NOTE:
Refer to Section 4.5 for the address decode of the peripheral chip selects.

12.9 X-Bus Control And General Purpose Decode

The X-Bus is a secondary data bus buffered from the EISA Bus. The X-Bus is used to interface with peripheral devices that do not require a high speed interface. Typically a discrete buffer device like a F245 is used to buffer the EISA Bus from the X-Bus. The ESC provides two control signals, XBUST/R# and XBUSOE#, for the discrete F245 buffer.

NOTE:
The ESC directly supports X-Bus floppy disk controller and X-Bus IDE hard disk controller. If IDE resides elsewhere (e.g., ISA Bus or PCI Bus), additional hardware is required to support the X-Bus floppy. The additional hardware is used to modify the X-Bus control signals. Figure 31 shows the logic needed to support an X-Bus floppy disk with IDE on the ISA or PCI bus.

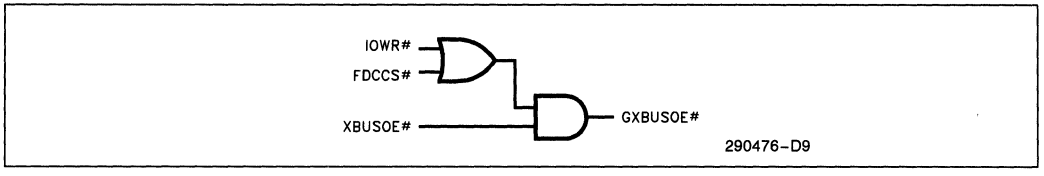


Figure 31

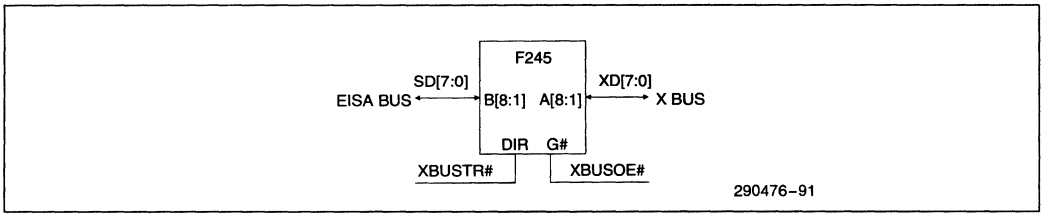


Figure 32. X-Bus Data Buffer

The XBUST/R# signal controls the direction of the data flow of the F245. When the XBUST/R# signal is high, the data direction of the F245 buffer is from the XD[7:0] bus to the SD[7:0] bus. The ESC drives the XBUST/R# signal high during EISA master I/O read cycles, ISA master I/O read cycles, DMA write cycles (write to memory), and memory read cycles decoded to be in the X-Bus BIOS address space. The ESC also drives the XBUST/R# signal high for DMA reads (reads from memory/writes to I/O) from the X-Bus BIOS address space. The X-Bus BIOS address space is defined as the enabled regions and enabled aliases of the BIOS memory space. See Section 4.1, BIOS Memory Space, for detailed description of the BIOS memory map and the configuration bits.

The XBUSOE# signal controls outputs of the F245. When the XBUSOE# signal is asserted, the F245 drives its A buffers or B buffers depending on the state of the XBUST/R# signal. The ESC asserts the XBUSOE# signal for I/O cycles decoded to be in the address range of the peripherals supported by the ESC if these peripherals are enabled in the Peripheral Chip Select A register and Peripheral Chip Select B register.

13.0 POWER MANAGEMENT (82374SB)

The ESC has extensive power management capability permitting a system to operate in a low power state without being powered down. In a typical desktop personal computer there are two states—Power On and Power-Off. Leaving a system powered on when not in use wastes power. The ESC provides a Fast On/Off feature that creates a third state called Fast Off (Figure 33). When in the Fast Off state, the system consumes less power than the Power-On state.

The ESC's power management architecture is based on three functionsSystem Management Mode (SMM), Clock Control, and Advanced Power Management (APM). Software (called SMM code) controls the transitions between the Power On state and the Fast Off state. The ESC invokes this software by generating an SMI to the CPU (asserting the SMI# signal). A variety of programmable events are provided that can generate an SMI. The SMM code places the system in either the Power On state or the Fast Off state.

A Fast On event is an event that instructs the computer (via an SMI to the CPU) to enter the Power-On state in anticipation of system activity by the user. Fast On events are programmable and include moving the mouse, pressing a key on the keyboard, an external hardware event, an incoming call to a system FAX/Modem, a RTC alarm, or the operating system.

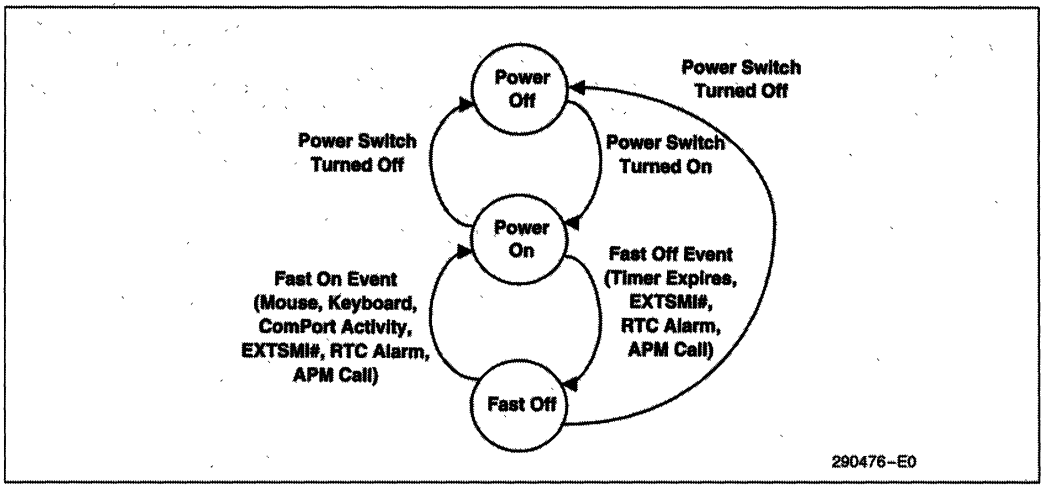


Figure 33. Fast On/Off Flow

13.1 SMM Mode

SMM mode is invoked by asserting the SMI# signal to the CPU. The ESC provides a variety of programmable events that can generate an SMI. When the CPU receives an SMI, it enters SMM mode and executes SMM code out of SMRAM. The SMM code places the system in either the Power On state or the Fast Off state. In the Power On state, the computer system operates normally. In this state one of the four programmable events listed below can trigger an SMI.

1. A global idle timer called the Fast Off timer expires (an indication that the end user has not used the computer for a programmed period of time).
2. The EXTSMI# pin is asserted.
3. An RTC alarm interrupt is detected.
4. The operating system issues an APM call.

13.2 SMI Sources

The SMI# signal can be asserted by hardware interrupt events, the Fast Off Timer, an external SMI event (EXTSMI#), and software events (via the APMC and APMS Registers). Enable/disable bits (in the SMIEN Register) permit each event to be individually masked from generating an SMI. In addition, the SMI# signal can be globally enabled/disabled in the SMICNTL Register. Status of the individual events causing an SMI is provided in the SMIREQ Register. For detailed information on the SMI control/status registers, refer to Section 3.0, Register Description.

Hardware Interrupt Events

Hardware events (IRQ[12,8#,4,3,1] and the Fast Off Timer) are enabled/disabled from generating an SMI in the SMIEN Register. When enabled, the occurrence of the corresponding hardware event generates an SMI (asserts the SMI# signal), regardless of the current power state of the system.

Fast Off Timer

The Fast Off Timer is used to indicate (through an SMI) that the system has been idle for a programmed period of time. The timer counts down from a programmed start value and when the count reaches 00h, can generate an SMI. The timer decrement rate is 1 count every minute and is re-loaded each time a System Event occurs. This counter should NOT be programmed to 00h.

System events are programmable events that can keep the system in the Power On state when there is system activity (Figure 34). These events are indicated by the assertion of IRQ[15:9,8#,7:3,1:0], NMI, or SMI signals. The system event prevents the system from entering the Fast Off state by re-loading the Fast Off Timer.

In addition to system events, *break events cause the system to transition from a Fast Off state to the Power On state. System events (and break events) are enabled/disabled in the SEE Register. When enabled and the associated hardware event occurs (signal is asserted), the Fast Off Timer is re-loaded with its initial count.*

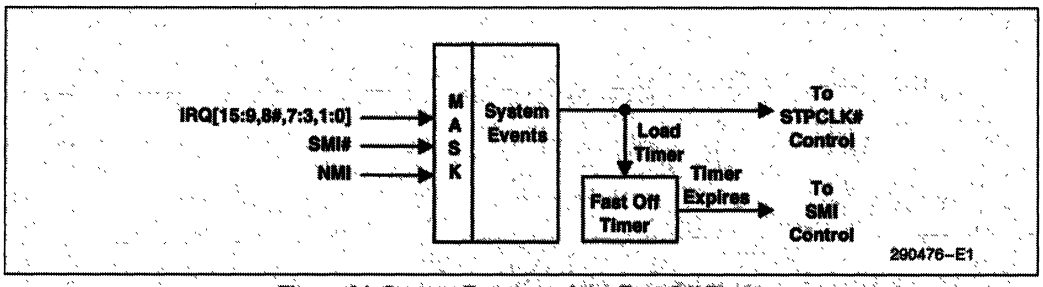


Figure 34. System Events and the Fast Off Timer

EXTSMI#

The EXTSMI# input pin provides the system designer the capability to invoke SMM with external hardware. For example, the EXTSMI# input could be connected to a "green button" permitting the user to enter the Fast Off state by depressing a button. The EXTSMI# generation of an SMI is enabled/disabled in the SMIEN Register.

Software Events

Software events (accessing the APMx Registers) indicate that the OS is passing power management information to the SMI handler. There are two Advanced Power Management (APM) registers—APM Control (APMC) and APM Status (APMS) Registers. These registers permit software to generate an SMI; by writing to the APMC Register. For example, the APMC can be used to pass an APM command between APM OS and BIOS and the APMS Register could be used to pass data between the OS and the SMI handler. For detailed descriptions of these registers, see Section 3.0, Register Description.

The two APM Registers are located in normal I/O space. The PCEB subtractively decodes PCI accesses to these registers and forwards the accesses to the EISA Bus. The APM Registers are not accessible by EISA masters. Note that the remaining power management registers are located in PCI configuration space.

13.3 SMI# And INIT Interaction

The SMI# input to the CPU is an edge sensitive signal. When an S-series processor is reset (INIT asserted), the processor resets the SMI# edge detect logic. After INIT is negated, it takes two clocks before the edge detect circuit can catch an edge. The ESC only asserts SMI# when INIT is negated. If the ESC asserts SMI# and then the INIT signal is sampled asserted, the ESC negates SMI#.

13.3.1 CLOCK CONTROL

The CPU can be put in a low power state by asserting the STPCLK# signal. STPCLK# is an interrupt to the CPU. However, for this type of interrupt, the CPU does not generate an interrupt acknowledge cycle. Once the STPCLK# interrupt is executed, the CPU enters the stop grant state. In this state, the CPU's internal clocks are disabled and instruction execution is stopped. The stop grant state is exited when the STPCLK# signal is negated.

Software can assert STPCLK#, if enabled via the SMICNTL Register, by a read of the APMC Register. Note that STPCLK# can also be periodically asserted by using clock scaling as described below.

The ESC automatically negates STPCLK# when a break event occurs (if enabled in the SEE Register) and the CPU stop grant special cycle has been received. Software can negate STPCLK# by disabling STPCLK# in the SMICNTL Register or by a write to the APMC Register.

NOTE:

1. INIT is always enabled as a break event. Otherwise, INIT acts exactly as other break events:
   — If STPCLK# is negated when INIT is asserted, the STPCLK high timer is reloaded.
   — If INIT is asserted when STPCLK# is asserted but before the stop grant bus cycle, STPCLK# negation waits until after the stop grant bus cycle. This happens after the CPU is reset when it samples STPCLK# still asserted.
   — If INIT is asserted when STPCLK# is asserted and after the stop grant bus cycle, STPCLK# is negated immediately. This guarantees that STPCLK# will be negated after the CPU is reset.
2. While the STPCLK# signal is asserted, the external interrupts (NMI, SMI# and INT) may be asserted to the CPU. If INTR is asserted, it will remain asserted until the CPU INTA cycle is detected. If SMI# (or NMI) is asserted, it remains asserted until the SMI (or NMI) handler clears the ESC's CSMIGATE (or sets the ESC's NMIMASK bit). Thus, SMI#, NMI and INTR can be applied to the CPU independent of the STPCLK# signal state. Note that when SMI#, NMI, and IRQx are enabled as break events, the occurrence of the break event negates STPCLK#.

Clock Scaling (Emulating Clock Division)

Clock scaling permits the ESC to periodically place the CPU in a low power state. This emulates clock division. When clock scaling is enabled, the CPU runs at full frequency for a pre-defined time period and then is stopped for a pre-defined time period. The run/stop time interval ratio emulates the clock division effect from a power/performance point of view. However, clock scaling is more effective than dividing the CPU frequency. For example, if the CPU is in the stop grant state and a break event occurs, the CPU clock returns to full frequency. In addition, there is no recovery time latency to start the clock.

Two programmable 8-bit clock scale timer control registers set the STPCLK# high (negate) and low (assert) times the CTLTMRH and CTLTMRL Registers. The timer is clocked by a 32 μs internal clock. This allows a programmable timer interval for both the STPCLK# high and low times of 0-8 ms. When enabled via the SMICNTL Register, the STPCLK# Timer operates as follows:

- When STPCLK# is negated, the timer is loaded with the value in the CTLTMRH Register and starts counting down. When the timer reaches 00h, STPCLK# is asserted. Since the timer is re-loaded with the contents of the CTLTMRH Register every time STPCLK# is negated (for break events or clock throttling), the STPCLK# minimum inactive time is guaranteed.
- When STPCLK# is asserted, the timer is loaded with the value in the CTLTMRL Register. The timer does not begin to count until the Stop Grant Special Cycle is received. When the timer reaches 00h, STPCLK# is negated. Note that a break event also negates STPCLK#.

NOTE:

If STPCLK# is negated and a break event occurs, the STPCLK# Timer is loaded with the value in the CTLTMRH Register.

13.4 Stop Grant Special Cycle

The Host-Bridge (e.g. PCMC) translates the CPU's stop grant cycle into a PCI special cycle. The PCEB recognizes the stop grant PCI special cycle and asserts STPGNT# low to ESC for one PCICLK. The ESC does not start the STPCLK low timer until STPGNT# is asserted.

During Halt or Autohalt state, the P54C does not respond to STPCLK# assertion with a stop grant cycle. However, during this state an INTR, SMI# or NMI# assertion causes the CPU to exit the halted state, and eventually recognize the STPCLK# assertion with a stop grant cycle. The system design must guarantee that INTR, SMI# and NMI# assertion is not blocked outside of the chipset while STPCLK# is asserted. Otherwise, a potential deadlock situation will exist.

13.5 Dual-Processor Power Management Support

Figure 35 depicts the power management support for dual-processor (DP) or P54CT upgrade processor configuration. The input signals of SMI#, STPCLK#, and NMI of both OEM and upgrade sockets are tied together.

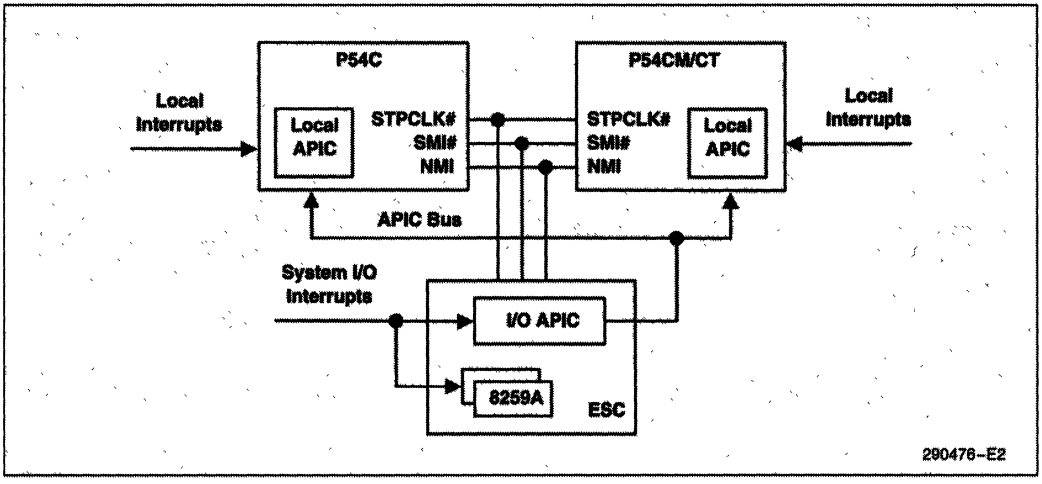


Figure 35. Dual Processor System Configuration

13.5.1 SMI# DELIVERY MECHANISM

For Uni or CT upgrade processor system configuration, SMI# can either be delivered through the ESC SMI# signal or I/O APIC. For the P54C/CM Dual-processor configuration, SMI# should be delivered through I/O APIC only. Ideally, the OS will put the CM processor in Autohalt after the CM processor received a Fast-Off SMI#. The CM processor will wake up if any non-masked system events occur.

13.5.2 STPCLK# TIED TO BOTH SOCKETS

To support a glueless upgrade socket, it is necessary to tie STPCLK# to both sockets. For P54C/CT processor configuration, the P54CT processor will disable P54C and the toggling of STPCLK# has no effect to P54C. For a P54C/CM DP configuration, the toggling of STPCLK# effects both processors (unless the processor is in Autohalt state). Both processors respond with a STPGNT special bus cycle after recognizing STPCLK# low. Both of the STPGNT special bus cycles are passed onto PCI by the PCMC as PCI STPGNT special cycles. The PCI STPGNT PCI cycle causes the PCEB component to assert the STPGNT# signal, depending on how the SCE bit in PCEB is programmed. The ESC recognizes the first STPGNT# assertion, and negates STPCLK# upon the Stop Clock timer expiration or a stop break event.

13.5.3 SMI#/INTR (APIC MODE)

When the APIC is used for interrupt delivery, additional considerations exist regarding ordering. If local interrupts (LINT0/1) are used in APIC mode, then the system can not guarantee an ordering between the local interrupts and any related SMI# events.

In DP mode, interrupts can generally be directed to a specific processor, which may not be the same processor that the SMI# is directed. The IRQ blocking logic in the ESC still operates with APIC delivery mode. Thus, if an IRQ is enabled to cause an SMI# event, it will be blocked until the CSMIGATE is cleared, regardless of where the IRQ or SMI is to be directed by the APIC.

2

14.0 ELECTRICAL CHARACTERISTICS

14.1 Maximum Ratings

Case Temperature Under Bias ..... −65°C to 110°C
Storage Temperature ..... −65°C to 150°C
Supply Voltages with Respect to Ground ..... −0.5V to V_{CC} + 0.5V
Voltage On Any Pin ..... −0.5V to V_{CC} + 0.5V
Power Dissipation .....0.70W fully loaded
.....0.55W with four slots

WARNING:

Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operating beyond the "Operating Conditions" is not recommended and extended exposure beyond "Operating Conditions" may affect reliability.

14.2 NAND Tree

A NAND Tree is provided primarily for VIL/VIH testing. The NAND Tree is also useful for Automated Test Equipment (ATE) at board-level testing. The NAND Tree allows the tester to test the solder connections for each individual signal pin.

The TEST pin, along with IRQ5 and IRQ3, activates the NAND Tree. Asserting TEST causes the output pulse train to appear on the EISAHOLD pin. IRQ5 must be driven high in order to enable the NAND Tree. The assertion of IRQ3 causes the ESC to disable its buffers.

The sequence of the ATE test is as follows:

1. Drive TEST low, IRQ3 high, and IRQ5 high.
2. Drive each pin that is a part of the NAND Tree high. Please note that not every pin is included in the tree. See table below for details.
3. Starting at pin 165 (DLIGHT#) and continuing with pins 167, 168, etc., individually drive each pin low. Expect EISAHOLD to toggle after each corresponding input pin is toggled. The final pin in the tree is pin 100 (EISAHOLD). Not every pin is toggled in sequential order. Please refer to the table for tree ordering. When IRQ3 is driven low, the test mode is exited, and the ESC's buffers will be enabled.
4. Before enabling the ESC's buffers (via IRQ3), turn off tester drivers.
5. Reset the ESC prior to proceeding with further testing.

Table 39. NAND Tree Cell Order (82374EB)

Pin #	Name
165	DLIGHT#[1]
167	FDCCS#
168	RTCWR#
169	RTCRD#
102	AFLUSH#
106	REFRESH#
107	APICCLK
108	APICD1
109	APICD0
110	IOCHK#
111	RSTDRV
112	IRQ9
113	DREQ2
114	NOWS#
115	CHRDY#
116	SMWTC#
121	SMRDC#
122	IOWC#
123	IORC#
125	DREQ3
127	DREQ1
135	IRQ7
136	IRQ6
141	BALE
142	OSC
143	SA1
144	SA0
145	M16#
146	SBHE#
147	IO16#
148	IRQ10

Pin #	Name
149	IRQ11
150	IRQ12
151	IRQ15
152	IRQ14
164	CRAMRD#
166	DSKCHG
171	ABFULL
179	CMD#
180	START#
186	EXRDY
187	EX32#
188	EX16#
189	SLBURST#
190	EOP
191	SPKR
193	IRQ8#
194	IRQ13
195	IRQ1
197	DREQ0
198	MRDC#
200	MWTC#
201	DREQ5
203	DREQ6
205	DREQ7
206	MASTER16#
3	LA31#/CPG4
4	LA30#/CPG3
5	LA29#/CPG2
6	LA28#/CPG1
7	LA27#/CPG0
8	LA26#
10	LA25#

Table 39. NAND Tree Cell Order (82374EB) (Continued)

Pin #	Name
11	LA24#
12	LA16
13	LA15
15	LA14
16	LA13
17	LA12
19	LA11
20	LA10
21	LA9
22	LA8
23	LA7
24	LA6
28	LA5
29	LA4
30	LA23
31	LA3
32	LA22
33	LA2
34	LA21
36	LA20
40	LA19
42	MREQ7#/PIRQ0#
43	LA18
44	MREQ6#/PIRQ1#
45	LA17
46	MREQ5#/PIRQ2#
47	MREQ4#/PIRQ3#
48	MREQ3#

Pin #	Name
49	MREQ2#
50	MREQ1#
51	MREQ0#
55	BE0#
56	BE1#
57	BE2#
58	BE3#
59	SD0
60	SD1
61	SD2
63	SD3
64	SD4
65	SD5
66	SD6
67	SD7
70	W/R#
71	M/IO#
72	MSBURST#
91	FERR#
95	RESET#
96	PERR#
97	SERR#
98	NMFLUSH#
99	PEREQ#/INTA#
138	IRQ4
139	IRQ3(2)
100	EISAHOLD(3)

Table 40. NAND Tree Cell Order (82374SB)

Pin #	Name
165	DLIGHT#(1)
167	FDCCS#
168	RTCWR#
169	RTCRD#
170	RTCALE
171	ABFULL
172	KYBDCS#
173	LBIOSCS#
174	SALAOE#
175	LASAOE#
176	SALE#
102	AFLUSH#
106	REFRESH#
107	APICCLK
108	APICD1
109	APICD0
110	IOCHK#
111	RSTDRV
112	IRQ9
113	DREQ2
114	NOWS#
115	CHRDY#
116	SMWTC#
117	AEN4
118	AEN3
119	AEN2
120	AEN1
121	SMRDC#
122	IOWC#
123	IORC#
124	DACK3#

Pin #	Name
125	DREQ3
126	DACK1#
127	DREQ1
133	STPGNT#
134	AEN#
135	IRQ7
136	IRQ6
140	DACK2#
141	BALE
142	OSC
143	SA1
144	SA0
145	M16#
146	SBHE#
147	IO16#
148	IRQ10
149	IRQ11
150	IRQ12
151	IRQ15
152	IRQ14
155	GPCS0#
159	GPCS1#
160	GPCS2#
161	XBUSOE#
162	XBUST/R#
163	CRAMWR#
164	CRAMRD#
166	DSKCHG
179	CMD#
180	START#
185	EXTSMI#
186	EXRDY

Table 40. NAND Tree Cell Order (82374SB) (Continued)

Pin #	Name
187	EX32#
188	EX16#
189	SLBURST#
190	EOP
191	SPKR
192	SLOWH#
193	IRQ8#
194	IRQ13
195	IRQ1
196	DACK0#
197	DREQ0
198	MRDC#
199	DACK5#
200	MWTC#
201	DREQ5
202	DACK6#
203	DREQ6
204	DACK7#
205	DREQ7
206	MASTER16#
207	MACK3#
3	LA31#/CPG4
4	LA30#/CPG3
5	LA29#/CPG2
6	LA28#/CPG1
7	LA27#/CPG0
8	LA26#
10	LA25#
11	LA24#
12	LA16
13	LA15
15	LA14

Pin #	Name
16	LA13
17	LA12
19	LA11
20	LA10
21	LA9
22	LA8
23	LA7
24	LA6
26	LA5
29	LA4
30	LA23
31	LA3
32	LA22
33	LA2
34	LA21
36	LA20
37	MACK2#
38	MACK1#
40	LA19
41	MACK0#
42	MREQ7#/PIRQ0#
43	LA18
44	MREQ6#/PIRQ1#
45	LA17
46	MREQ5#/PIRQ2#
47	MREQ4#/PIRQ3#
48	MREQ3#
49	MREQ2#
50	MREQ1#
51	MREQ0#
55	BE0#
56	BE1#

Table 40. NAND Tree Cell Order (82374SB) (Continued)

Pin #	Name
57	BE2#
58	BE3#
59	SD0
60	SD1
61	SD2
63	SD3
64	SD4
65	SD5
66	SD6
67	SD7
70	W/R#
71	M/IO#
72	MSBURST#
73	SDOE2#
74	SDOE1#
75	SDOE0#
76	SDCPYUP
80	SDCPYEN13#
81	SDCPYEN03#

Pin #	Name
82	SDCPYEN02#
83	SDCPYEN01#
84	SDLE0#
85	SDLE1#
86	SDLE2#
87	SDLE3#
91	FERR#
95	RESET#
96	PERR#
97	SERR#
98	NMFLUSH#
99	PEREQ#/INTA#
138	IRQ4
139	IRQ3(2)
100	EISAHOLD(3)

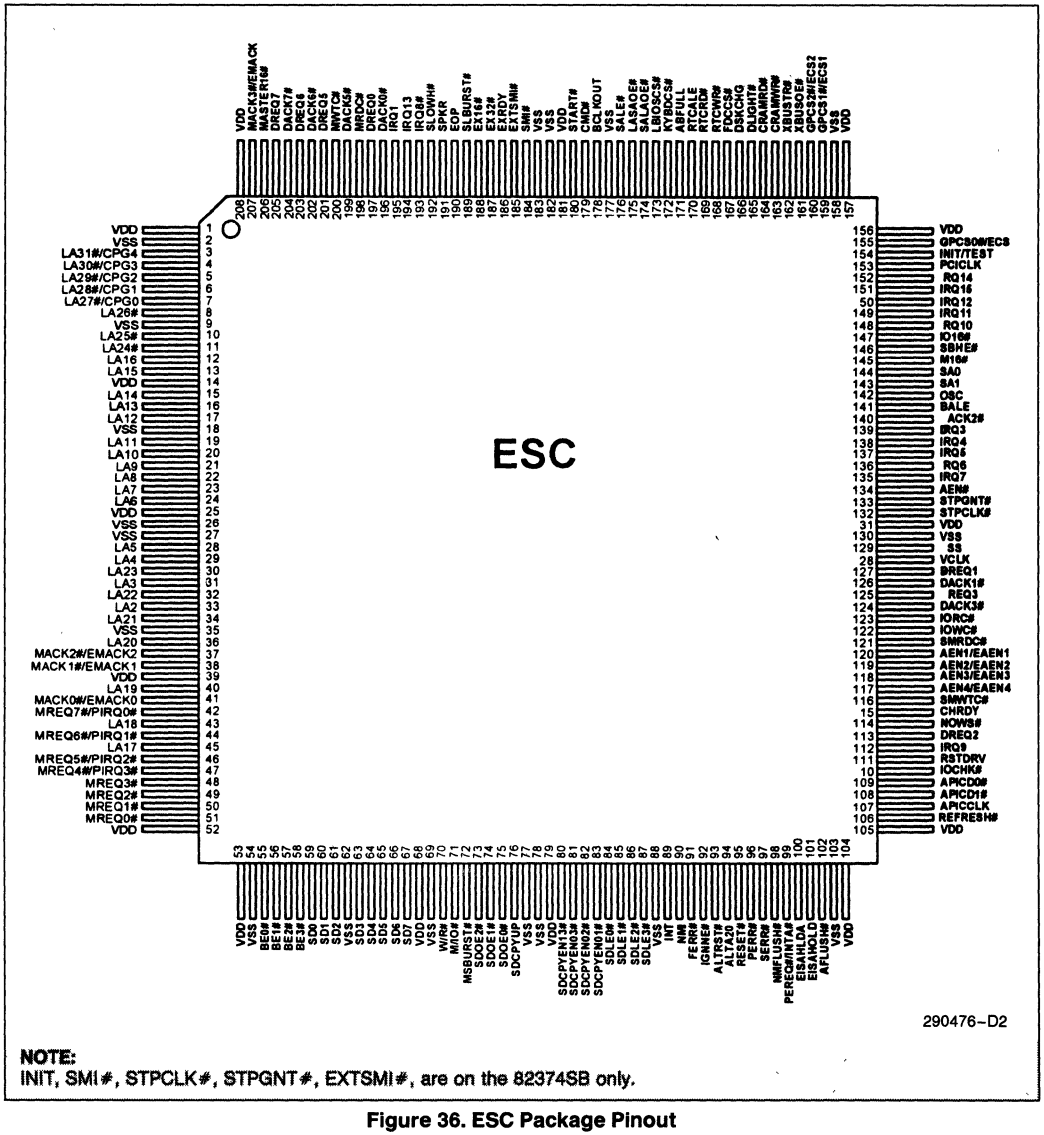
NOTES:
1. First Pin in NAND Tree.
2. Enables ESC's Buffers when 0.
3. Last Pin in NAND Tree.

2

15.0 PINOUT AND PACKAGE INFORMATION

The ESC package is a 208-pin Plastic Quad Flat Pack (PQFP). The package signals are shown in Figure 36 and listed in Table 41 and Table 42. Note that NC pins require individual pull-up resistors of 8 KΩ–10 KΩ.

15.1 Pinout And Pin Assignment



NOTE:
INIT, SMI#, STPCLK#, STPGNT#, EXTSMI#, are on the 82374SB only.

Figure 36. ESC Package Pinout

Table 41. ESC Alphabetical Pin Assignment

Name	Pin #	Type
ABFULL	171	in
AEN#	134	out
AEN1/EAEN1	120	out
AEN2/EAEN2	119	out
AEN3/EAEN3	118	out
AEN4/EAEN4	117	out
AFLUSH#	102	t/s
ALTA20	94	out
ALTRST#	93	out
APICCLK	107	in
APICD0#	109	od
APICD1#	108	od
BALE	141	out
BCLK	128	in
BCLKOUT	176	out
BE0#	55	t/s
BE1#	56	t/s
BE2#	57	t/s
BE3#	58	t/s
CHRDY	115	o/d
CMD#	179	out
CRAMRD#	164	out
CRAMWR#	163	out
DACK0#	196	out
DACK1#	126	out
DACK2#	140	out
DACK3#	124	out
DACK5#	199	out
DACK6#	202	out
DACK7#	204	out
DLIGHT#	165	out

Name	Pin #	Type
DREQ0	197	in
DREQ1	127	in
DREQ2	113	in
DREQ3	125	in
DREQ5	201	in
DREQ6	203	in
DREQ7	205	in
DSKCHG	166	in
EISAHLDA	100	in
EISAHOLD	101	out
EOP	190	t/s
EX16#	188	o/d
EX32#	187	o/d
EXRDY	186	o/d
EXTSMI# (82374SB)	185	in
FDCCS#	167	out
FERR#	91	in
GPCS0#/ECS0	155	out
GPCS1#/ECS1	159	out
GPCS2#/ECS2	160	out
IGNNE#	92	out
INIT/TEST (82374SB)	154	in
INTR	89	out
IO16#	147	o/d
IOCHK#	110	in
IORC#	123	t/s
IOWC#	122	t/s
IRQ1	195	in
IRQ3	139	in
IRQ4	138	in
IRQ5	137	in

Table 41. ESC Alphabetical Pin Assignment (Continued)

Name	Pin #	Type
IRQ6	136	in
IRQ7	135	in
IRQ8#	193	in
IRQ9	112	in
IRQ10	148	in
IRQ11	149	in
IRQ12	150	in
IRQ13	194	in
IRQ14	152	in
IRQ15	151	in
KYBDCS#	172	out
LA2	33	t/s
LA3	31	t/s
LA4	29	t/s
LA5	28	t/s
LA6	24	t/s
LA7	23	t/s
LA8	22	t/s
LA9	21	t/s
LA10	20	t/s
LA11	19	t/s
LA12	17	t/s
LA13	16	t/s
LA14	15	t/s
LA15	13	t/s
LA16	12	t/s
LA17	45	t/s
LA18	43	t/s
LA19	40	t/s
LA20	36	t/s
LA21	34	t/s
LA22	32	t/s

Name	Pin #	Type
LA23	30	t/s
LA24#	11	t/s
LA25#	10	t/s
LA26#	8	t/s
LA27#/CPG0	7	t/s
LA28#/CPG1	6	t/s
LA29#/CPG2	5	t/s
LA30#/CPG3	4	t/s
LA31#/CPG4	3	t/s
LASAOE#	175	out
LBIOSCS#	173	out
M/IO#	71	t/s
M16#	145	o/d
MACK0#/ EMACK0	41	out
MACK1#/ EMACK1	38	out
MACK2#/ EMACK2	37	out
MACK3#/ EMACK3	207	out
MASTER16#	206	in
MRDC#	198	t/s
MREQ0#	51	in
MREQ1#	50	in
MREQ2#	49	in
MREQ3#	48	in
MREQ4#PIRQ3#	47	in
MREQ5#/PIRQ2#	46	in
MREQ6#/PIRQ1#	44	in
MREQ7#/PIRQ0#	42	in
MSBURST#	72	t/s

Table 41. ESC Alphabetical Pin Assignment (Continued)

Name	Pin #	Type
MWTC#	200	t/s
NC (82374EB)	132	—
NC (82374EB)	133	—
NC (82374EB)	184	—
NC (82374EB)	185	—
NMFLUSH#	98	t/s
NMI	90	out
NOWS#	114	o/d
OSC	142	in
PCICLK	153	in
PEREQ#/INTA#	99	in
PERR#	96	in
REFRESH#	106	t/s
RESET#	95	in
RSTDRV	111	out
RTCALE	170	out
RTCRD#	169	out
RTCWR#	168	out
SA0	144	t/s
SA1	143	t/s
SALAOE#	174	out
SALE#	176	out
SBHE#	146	t/s
SD0	59	t/s
SD1	60	t/s
SD2	61	t/s
SD3	63	t/s
SD4	64	t/s
SD5	65	t/s
SD6	66	t/s
SD7	67	t/s
SDCPYEN01#	83	out

Name	Pin #	Type
SDCPYEN02#	82	out
SDCPYEN03#	81	out
SDCPYEN13#	80	out
SDCPYUP	76	out
SDLE0#	84	out
SDLE1#	85	out
SDLE2#	86	out
SDLE3#	87	out
SDOE0#	75	out
SDOE1#	74	out
SDOE2#	73	out
SERR#	97	in
SLBURST#	189	in
SLOWH#	192	out
SMI# (82374SB)	184	out
SMRDC#	121	out
SMWTC#	116	out
SPKR	191	out
START#	180	t/s
STPCLK# (82374SB)	132	out
STPGNT# (82374SB)	133	in
TEST (82374EB)	154	in
V_{DD}	1	V
V_{DD}	14	V
V_{DD}	25	V
V_{DD}	39	V
V_{DD}	52	V
V_{DD}	53	V
V_{DD}	68	V
V_{DD}	79	V
V_{DD}	104	V

Table 41. ESC Alphabetical Pin Assignment (Continued)

Name	Pin #	Type
V_{DD}	105	V
V_{DD}	131	V
V_{DD}	156	V
V_{DD}	157	V
V_{DD}	181	V
V_{DD}	208	V
V_{SS}	2	V
V_{SS}	9	V
V_{SS}	18	V
V_{SS}	26	V
V_{SS}	27	V
V_{SS}	35	V
V_{SS}	54	V
V_{SS}	62	V

Name	Pin #	Type
V_{SS}	69	V
V_{SS}	77	V
V_{SS}	78	V
V_{SS}	88	V
V_{SS}	103	V
V_{SS}	129	V
V_{SS}	130	V
V_{SS}	158	V
V_{SS}	177	V
V_{SS}	182	V
V_{SS}	183	V
W/R#	70	t/s
XBUSOE#	161	out
XBUST/R#	162	out

Table 42. ESC Numerical Pin Assignment

Name	Pin #	Type
1	V_{DD}	V
2	V_{SS}	V
3	LA31#/CPG4	t/s
4	LA30#/CPG3	t/s
5	LA29#/CPG2	t/s
6	LA28#/CPG1	t/s
7	LA27#/CPG0	t/s
8	LA26#	t/s
9	V_{SS}	V
10	LA25#	t/s
11	LA24#	t/s
12	LA16	t/s
13	LA15	t/s
14	V_{DD}	V
15	LA14	t/s
16	LA13	t/s
17	LA12	t/s
18	V_{SS}	V
19	LA11	t/s
20	LA10	t/s
21	LA9	t/s
22	LA8	t/s
23	LA7	t/s
24	LA6	t/s
25	V_{DD}	V
26	V_{SS}	V
27	V_{SS}	V
28	LA5	t/s

Name	Pin #	Type
29	LA4	t/s
30	LA23	t/s
31	LA3	t/s
32	LA22	t/s
33	LA2	t/s
34	LA21	t/s
35	V_{SS}	V
36	LA20	t/s
37	MACK2#/EMACK2	out
38	MACK1#/EMACK1	out
39	V_{DD}	V
40	LA19	t/s
41	MACK0#/EMACK0	out
42	MREQ7#/PIRQ0#	in
43	LA18	t/s
44	MREQ6#/PIRQ1#	in
45	LA17	t/s
46	MREQ5#/PIRQ2#	in
47	MREQ4#/PIRQ3#	in
48	MREQ3#	in
49	MREQ2#	in
50	MREQ1#	in
51	MREQ0#	in
52	V_{DD}	V
53	V_{DD}	V
54	V_{SS}	V
55	BE0#	t/s

Table 42. ESC Numerical Pin Assignment (Continued)

Name	Pin #	Type
56	BE1#	t/s
57	BE2#	t/s
58	BE3#	t/s
59	SD0	t/s
60	SD1	t/s
61	SD2	t/s
62	V_{SS}	V
63	SD3	t/s
64	SD2	t/s
65	SD5	t/s
66	SD6	t/s
67	SD7	t/s
68	V_{DD}	V
69	V_{SS}	V
70	W/R#	t/s
71	M/IO#	t/s
72	MSBURST#	t/s
73	SDOE2#	out
74	SDOE1#	out
75	SDOE0#	out
76	SDCPYUP	out
77	V_{SS}	V
78	V_{SS}	V
79	V_{DD}	V
80	SDCPYEN13#	out
81	SDCPYEN03#	out
82	SDCPYEN02#	out
83	SDCPYEN01#	out
84	SDLE0#	out
85	SDLE1#	out
86	SDLE2#	out
87	SDLE3#	out

Name	Pin #	Type
88	V_{SS}	V
89	INTR	out
90	NMI	out
91	FERR#	in
92	IGNNE#	out
93	ALTRST#	out
94	ALTA20	out
95	RESET#	in
96	PERR#	in
97	SERR#	in
98	NMFLUSH#	t/s
99	PEREQ# / INTA#	in
100	EISAHLDA	in
101	EISAHOLD	out
102	AFLUSH#	t/s
103	V_{SS}	V
104	V_{DD}	V
105	V_{DD}	V
106	REFRESH#	t/s
107	APICCLK	in
108	APICD1#	od
109	APICD0#	od
110	IOCHK#	in
111	RSTDRV	out
112	IRQ9	in
113	DREQ2	in
114	NOWS#	o/d
115	CHRDY	o/d
116	SMWTC#	out
117	AEN4/EAEN4	out
118	AEN3/EAEN3	out
119	AEN2/EAEN2	out

Table 42. ESC Numerical Pin Assignment (Continued)

Name	Pin #	Type
120	AEN1/EAEN1	out
121	SMRDC#	out
122	IOWC#	t/s
123	IORC#	t/s
124	DACK3#	out
125	DREQ3	in
126	DACK1#	out
127	DREQ1	in
128	BCLK	in
129	V_{SS}	V
130	V_{SS}	V
131	V_{DD}	V
132	NC (82374EB) STPCLK# (82374SB)	— out
133	NC (82374EB) STPGNT# (82374SB)	— in
134	AEN#	out
135	IRQ7	in
136	IRQ6	in
137	IRQ5	in
138	IRQ4	in
139	IRQ3	in
140	DACK2#	out
141	BALE	out
142	OSC	in
143	SA1	t/s
144	SA0	t/s
145	M16#	o/d
146	SBHE#	t/s
147	IO16#	o/d
148	IRQ10	in

Name	Pin #	Type
149	IRQ11	in
150	IRQ12	in
151	IRQ15	in
152	IRQ14	in
153	PCICLK	in
154	TEST (82374EB) INIT/TEST (82374SB)	in in
155	GPCS0#/ECS0	out
156	V_{DD}	V
157	V_{DD}	V
158	V_{SS}	V
159	GPCS1#/ECS1	out
160	GPCS2#/ECS2	out
161	XBUSOE#	out
162	XBUST/R#	out
163	CRAMWR#	out
164	CRAMRD#	out
165	DLIGHT#	out
166	DSKCHG	in
167	FDCCS#	out
168	RTCWR#	out
169	RTCRD#	out
170	RTCALE	out
171	ABFULL	in
172	KYBDCS#	out
173	LBIOSCS#	out
174	SALAOE#	out
175	LASAOE#	out
176	SALE#	out
177	V_{SS}	V
178	BCLKOUT	out
179	CMD#	out

Table 42. ESC Numerical Pin Assignment (Continued)

Name	Pin #	Type
180	START#	t/s
181	V_{DD}	V
182	V_{SS}	V
183	V_{SS}	V
184	NC (82374EB) SMI# (82374SB)	— out
185	NC (82374EB) EXTSMI# (82374SB)	— in
186	EXRDY	o/d
187	EX32#	o/d
188	EX16#	o/d
189	SLBURST#	in
190	EOP	t/s
191	SPKR	out
192	SLOWH#	out

Name	Pin #	Type
193	IRQ8#	in
194	IRQ13	in
195	IRQ1	in
196	DACK0#	out
197	DREQ0	in
198	MRDC#	t/s
199	DACK5#	out
200	MWTC#	t/s
201	DREQ5	in
202	DACK6#	out
203	DREQ6	in
204	DACK7#	out
205	DREQ7	in
206	MASTER16#	in
207	MACK3#/EMACK3	out
208	V_{DD}	V

15.2 Package Characteristics

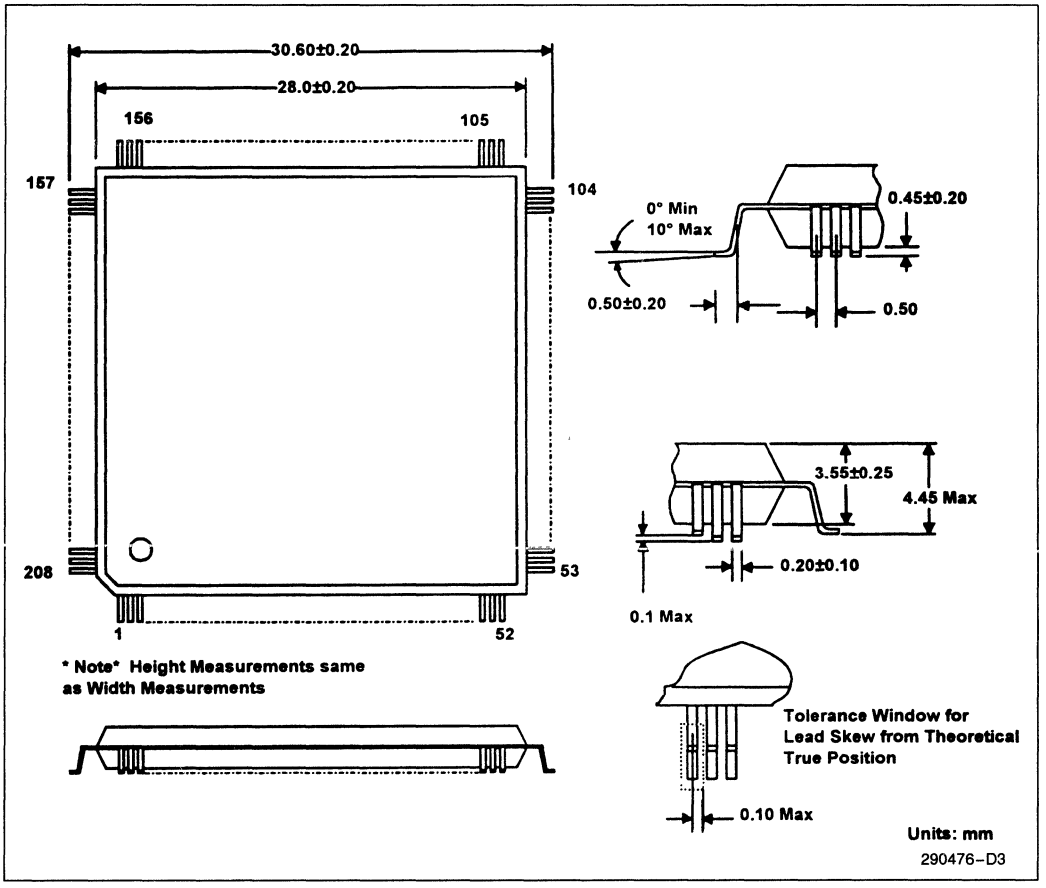


Figure 37. Packaging Dimension Information

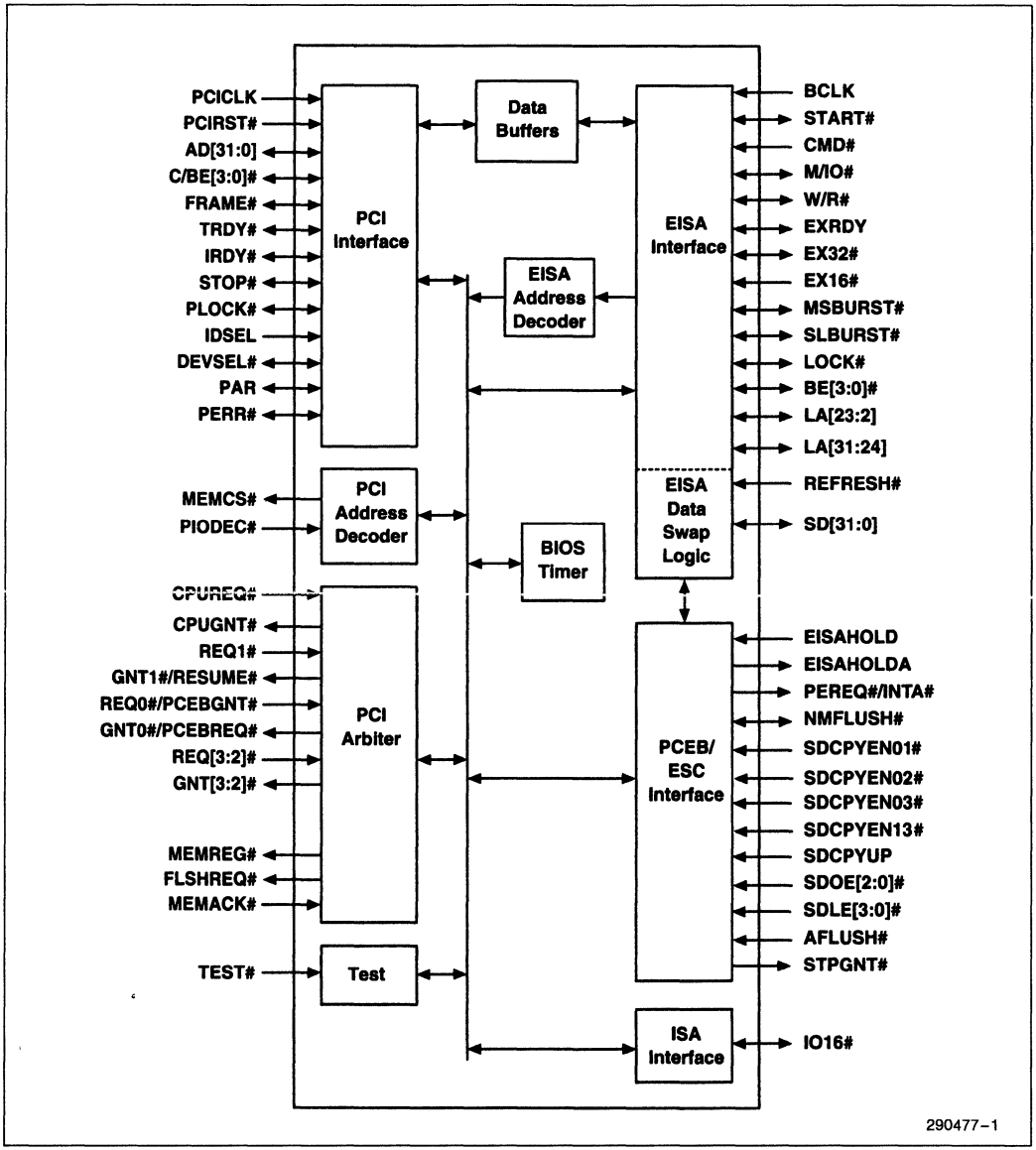
82375EB/82375SB PCI-EISA BRIDGE (PCEB)

- Provides the Bridge Between the PCI Local Bus and EISA Bus
- 100% PCI and EISA Compatible
  - PCI and EISA Master/Slave Interface
  - Directly Drives 10 PCI Loads and 8 EISA Slots
  - Supports PCI from 25 to 33 MHz
- Data Buffers Improve Performance
  - Four 32-bit PCI-to-EISA Posted Write Buffers
  - Four 16-byte EISA-to-PCI Read/Write Line Buffers
  - EISA-to-PCI Read Prefetch
  - EISA-to-PCI and PCI-to-EISA Write Posting
- Data Buffer Management Ensures Data Coherency
  - Flush Posted Write Buffers
  - Flush or Invalidate Line Buffers
  - System-Wide Data Buffer Coherency Control
- Burst Transfers on both the PCI and EISA Buses
- 32-Bit Data Paths
- Integrated EISA Data Swap Buffers
- Arbitration for PCI Devices
  - Supports Six PCI Masters
  - Fixed, Rotating, or a Combination of the Two
  - Supports External PCI Arbiter and Arbiter Cascading
- PCI and EISA Address Decoding and Mapping
  - Positive Decode of Main Memory Areas (MEMCS# Generation)
  - Four Programmable PCI Memory Space Regions
  - Four Programmable PCI I/O Space Regions
- Programmable Main Memory Address Decoding
  - Main Memory Sizes up to 512 MBytes
  - Access Attributes for 15 Memory Segments in First 1 MByte of Main Memory
  - Programmable Main Memory Hole
- Integrated 16-bit BIOS Timer
- Only Available as Part of a Supported Kit

The 82375EB/SB PCI-EISA Bridge (PCEB) provides the master/slave functions on both the PCI Local Bus and the EISA Bus. Functioning as a bridge between the PCI and EISA buses, the PCEB provides the address and data paths, bus controls, and bus protocol translation for PCI-to-EISA and EISA-to-PCI transfers. Extensive data buffering in both directions increases system performance by maximizing PCI and EISA Bus efficiency and allowing concurrency on the two buses. The PCEB's buffer management mechanism ensures data coherency. The PCEB integrates central bus control functions including a programmable bus arbiter for the PCI Bus and EISA data swap buffers for the EISA Bus. Integrated system functions include PCI parity generation, system error reporting, and programmable PCI and EISA memory and I/O address space mapping and decoding. The PCEB also contains a BIOS Timer that can be used to implement timing loops. The PCEB is intended to be used with the EISA System Component (ESC) to provide an EISA I/O subsystem interface.

This document describes both the 82375EB and 82375SB components. Unshaded areas describe the 82375EB. Shaded areas, like this one, describe the 82375SB operations that differ from the 82375EB.

December 1995
Order Number: 290477-004



PCEB Simplified Block Diagram

82375EB/82375SB PCI-EISA BRIDGE (PCEB)

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

The PCI-EISA bridge chip set provides an I/O subsystem core for the next generation of high-performance personal computers (e.g., those based on the Intel486™ or Pentium® processors). System designers can take advantage of the power of the PCI local bus while maintaining access to the large base of EISA and ISA expansion cards, and corresponding software applications. Extensive buffering and buffer management within the PCI-EISA bridge ensures maximum efficiency in both bus environments.

The chip set consists of two components—the 82375EB PCI-EISA Bridge (PCEB) and the 82374EB EISA System Component (ESC). These components work in tandem to provide an EISA I/O subsystem interface for personal computer platforms based on the PCI standard. This section provides an overview of the PCI and EISA Bus hierarchy followed by an overview of the PCEB and ESC components.

Bus Hierarchy—Concurrent Operations

Figure 1 shows a block diagram of a typical system using the PCI-EISA Bridge chip set. The system contains three levels of buses structured in the following hierarchy:

- Host Bus as the execution bus
- PCI Bus as a primary I/O bus
- EISA Bus as a secondary I/O bus

PCI Bus

The PCI Bus has been defined to address the growing industry needs for a standardized *local bus* that is not directly dependent on the speed and the size of the processor bus. New generations of personal computer system software such as Windows™ and Win-NT™ with sophisticated graphical interfaces, multi-tasking and multi-threading bring new requirements that traditional PC I/O architectures can not satisfy. In addition to the higher bandwidth, reliability and robustness of the I/O subsystem are becoming increasingly important. The PCI environment addresses these needs and provides an upgrade path for the future. PCI features include:

- Processor independent
- Multiplexed, burst mode operation
- Synchronous up to 33 MHz
- 120 MByte/sec usable throughput (132 MByte/sec peak) for 32-bit data path
- 240 MByte/sec usable throughput (264 MByte/sec peak) for 64-bit data path
- Optional 64-bit data path with operations that are transparent with the 32-bit data path
- Low latency random access (60 ns write access latency to slaves from a master parked on the bus)
- Capable of full concurrency with processor/memory subsystem
- Full multi-master capability allowing any PCI master peer-to-peer access to any PCI slave
- Hidden (overlapped) central arbitration
- Low pin count for cost effective component packaging (multiplexed address/data)
- Address and data parity
- Three physical address spaces: memory, I/O, and configuration
- Comprehensive support for autoconfiguration through a defined set of standard configuration functions

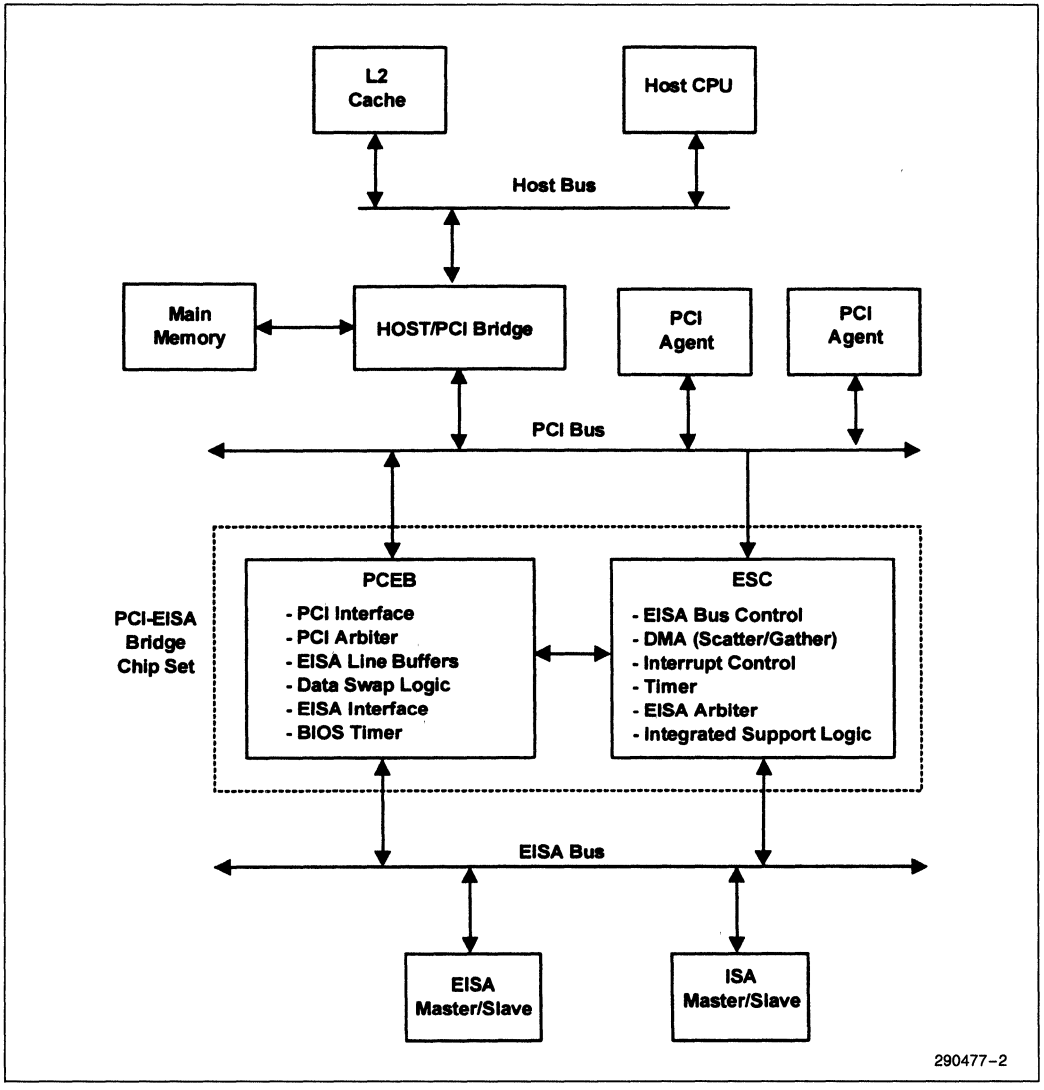


Figure 1. PCI-EISA System Diagram

System partitioning shown in Figure 1 illustrates how the PCI can be used as a common interface between different portions of a system platform that are typically supplied by the chip set vendor. These portions are the Host/PCI Bridge (including a main memory DRAM controller and an optional secondary cache controller) and the PCI-EISA Bridge. Thus, the PCI allows a system I/O core design to be decoupled from the processor/

memory treadmill, enabling the I/O core to provide maximum benefit over multiple generations of processor/memory technology. For this reason, the PCI-EISA Bridge can be used with different processors (i.e. derivatives of the Intel486 CPU or the new generation processors, such as the Pentium processor.) Regardless of the new requirements imposed on the processor side of the Host/PCI Bridge (e.g. 64-bit data path, 3.3V interface, etc.) the PCI side remains unchanged. This standard PCI environment allows reusability, not only of the rest of the platform chip set (i.e. PCI-EISA Bridge), but also of all other I/O functions interfaced at the PCI level. These functions typically include graphics, SCSI, and LAN.

EISA Bus

The EISA bus in the system shown in the Figure 1 represents a second level I/O bus. It allows personal computer platforms built around the PCI as a primary I/O bus to leverage the large EISA/ISA product base. Combinations of PCI and EISA buses, both of which can be used to provide expansion functions, will satisfy even the most demanding applications.

Along with compatibility for 16-bit and 8-bit ISA hardware and software, the EISA bus provides the following key features:

- 32-bit addressing and 32-bit data path
- 33 MByte/sec bus bandwidth
- Multiple bus master support through efficient arbitration
- Support for autoconfiguration

Integrated Bus Central Control Functions

The PCI-EISA Bridge chip set integrates central bus functions on both the PCI and EISA Buses. For PCI, the functions include PCI bus arbitration and the default bus driver. For the EISA Bus, central functions include the EISA Bus controller and EISA arbiter that are integrated in the ESC and EISA data swap buffers that are integrated in the PCEB.

Integrated System Functions

The PCI-EISA Bridge chip set integrates system functions including PCI parity and system error reporting, buffer management, PCI and EISA memory and I/O address space mapping and decoding. For maximum flexibility, all of these functions are programmable allowing for variety of optional features.

1.1 PCEB Overview

The PCEB and ESC form a PCI-EISA Bridge chip set. The PCEB/ESC interface provides the inter-chip communications between these two devices. The major functions provided by the PCEB are described in this section.

PCI Bus Interface

The PCEB can be either a master or slave on the PCI Bus and supports bus frequencies from 25-to-33 MHz. The PCEB becomes a slave when it positively decodes a PCI cycle. The PCEB also becomes a slave for unclaimed cycles on the PCI Bus. These unclaimed cycles are subtractively decoded by the PCEB and forwarded to the EISA Bus. As a slave, the PCEB supports single cycle transfers for memory, I/O, and configuration operations.

For EISA-initiated transfers to the PCI Bus, the PCEB is a PCI master. The PCEB permits EISA devices to access either PCI memory or I/O. While all PCI I/O transfers are single cycle, PCI memory cycles can be either single cycle or burst, depending on the status of the PCEB's Line Buffers. During EISA reads of PCI memory, the PCEB uses a burst read cycle of four Dwords to prefetch data into a Line Buffer. During EISA-to-PCI memory writes, the PCEB uses PCI burst cycles to flush the Line Buffers. The PCEB contains a programmable Master Latency Timer that provides the PCEB with a guaranteed time slice on the PCI Bus, after which it surrenders the bus.

As a master on the PCI Bus, the PCEB generates address and command signals (C/BE[3:0]#), address parity for read and write cycles, and data parity for write cycles. As a slave, the PCEB generates data parity for read cycles. Parity checking is not supported.

The PCEB, as a resource, can be locked by any PCI master. In the context of locked cycles, the entire PCEB subsystem (including the EISA Bus) is considered a single resource.

PCI Bus Arbitration

The PCI arbiter supports six PCI masters—the Host/PCI bridge, PCEB, and four other PCI masters. The arbiter can be programmed for twelve fixed priority schemes, a rotating scheme, or a combination of the fixed and rotating schemes. The arbiter can be programmed for bus parking that permits the Host/PCI Bridge default access to the PCI Bus when no other device is requesting service. The arbiter also contains an efficient PCI retry mechanism to minimize PCI Bus thrashing when the PCEB generates a retry.

EISA Bus Interface

The PCEB contains a fully EISA-compatible master and slave interface. The PCEB directly drives eight EISA slots without external data or address buffering. The PCEB is only a master or slave on the EISA Bus for transfers between the EISA Bus and PCI Bus. For transfers contained to the EISA Bus, the PCEB is never a master or slave. However, the data swap buffers contained in the PCEB are involved in these transfers, if data size translation is needed. The PCEB also provides support for I/O recovery.

EISA/ISA masters and DMA can access PCI memory or I/O. The PCEB only forwards EISA cycles to the PCI Bus if the address of the transfer matches one of the address ranges programmed into the PCEB for EISA-to-PCI positive decode. This includes the main memory segments used for generating MEMCS# from the EISA Bus, one of the four programmable memory regions, or one of the four programmable I/O regions. For EISA-initiated accesses to the PCI Bus, the PCEB is a slave on the EISA Bus. I/O accesses are always non-buffered and memory accesses can be either non-buffered or buffered via the Line Buffers. For buffered accesses, burst cycles are supported.

During PCI-initiated cycles to the EISA Bus, the PCEB is an EISA master. Single cycle transfers are used for I/O and memory read/write cycles from PCI to EISA.

PCI/EISA Address Decoding

The PCEB contains two address decoders—one to decode PCI-initiated cycles and the other to decode EISA-initiated cycles. The two decoders permit the PCI and EISA Buses to operate concurrently.

The PCEB can also be programmed to provide main memory address decoding on behalf of the Host/PCI bridge. When programmed, the PCEB monitors the PCI and EISA bus cycle addresses, and generates a memory chip select signal (MEMCS#) indicating that the current cycle is targeted to main memory residing behind the Host/PCI bridge. Programmable features include, read/write attributes for specific memory segments and the enabling/disabling of a memory hole. If not used, the MEMCS# feature can be disabled.

In addition to the main memory address decoding, there are four programmable memory regions and four programmable I/O regions for EISA-initiated cycles. EISA/ISA master or DMA accesses to one of these regions are forwarded to the PCI Bus.

Data Buffering

The PCEB contains four 16-byte wide Line Buffers for EISA-initiated cycles to the PCI Bus. The Line Buffers permit prefetching of read data from PCI memory and posting of data being written to PCI memory.

By using burst transactions to fill or flush these buffers, when appropriate, the PCEB maximizes bus efficiency. For example, an EISA device could fill a Line Buffer with byte, word, or Dword transfers and the PCEB would use a PCI burst cycle to flush the filled line to PCI memory.

BIOS Timer

The PCEB has a 16-bit BIOS Timer. The timer can be used by BIOS software to implement timing loops. The timer count rate is derived from the EISA clock (BCLK) and has an accuracy of ± 1 μs.

1.2 ESC Overview

The PCEB and ESC form a PCI-EISA bridge. The PCEB/ESC interface provides the inter-chip communications between these two devices. The major functions provided by the ESC are described in this section.

EISA Controller

The ESC incorporates a 32-bit master and an 8-bit slave. The ESC directly drives eight EISA slots without external data or address buffering. EISA system clock (BCLK) generation is integrated by dividing the PCI clock (divide by 3 or divide by 4) and wait state generation is provided. The AENx and MACKx signals provide a direct interface to four EISA slots and supports eight EISA slots with encoded AENx and MACKx signals.

The ESC contains an 8-bit data bus (lower 8 bits of the EISA data bus) that is used to program the ESC's internal registers. Note that for transfers between the PCI and EISA Buses, the PCEB provides the data path. Thus, the ESC does not require a full 32-bit data bus. A full 32-bit address bus is provided and is used during refresh cycles and for DMA operations.

The ESC performs cycle translation between the EISA Bus and ISA Bus. For mis-matched master/slave combinations, the ESC controls the data swap buffers that are located in the PCEB. This control is provided through the PCEB/ESC interface.

DMA Controller

The ESC incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels. Each channel can be programmed for 8- or 16-bit DMA device size, and ISA-compatible, type "A", type "B", or type "C" timings. Full 32-bit addressing is provided. The DMA controller also generates refresh cycles.

The DMA controller supports an enhanced feature called scatter/gather. This feature provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In scatter/gather mode, the DMA can read the memory address and word count from an array of buffer descriptors, located in main memory, called the scatter/gather descriptor (SGD) table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD table are handled.

Interrupt Controller

The ESC contains an EISA compatible interrupt controller that incorporates the functionality of two 82C59 Interrupt Controllers. The two interrupt controllers are cascaded providing 14 external and two internal interrupts.

Advanced Programmable Interrupt Controller (APIC)

In addition to the standard EISA compatible interrupt controller described above, the ESC incorporates the Advanced Programmable Interrupt Controller (APIC). While the standard interrupt controller is intended for use in a uni-processor system, APIC can be used in either a uni-processor or multi-processor system. APIC provides multi-processor interrupt management and incorporates both static and dynamic symmetric interrupt distribution across all processors. In systems with multiple I/O subsystems, each subsystem can have its own set of interrupts.

Timer/Counter

The ESC provides two 82C54 compatible timers (Timer 1 and Timer 2). The counters in Timer 1 support the system timer interrupt (IRQ0#), refresh request, and a speaker tone output (SPKR). The counters in Timer 2 support fail-safe time-out functions and the CPU speed control.

Integrated Support Logic

To minimize the chip count for board designs, the ESC incorporates a number of extended features. The ESC provides support for ALTA20 (Fast A20GATE) and ALTRST with I/O Port 92h. The ESC generates the control signals for SA address buffers and X-Bus buffer. The ESC also provides chip selects for BIOS, the keyboard controller, the floppy disk controller, and three general purpose devices. Support for generating chip selects with an external decoder is provided for IDE, a parallel port, and a serial port. The ESC provides support for a PC/AT compatible coprocessor interface and IRQ13 generation.

Power Management

Extensive power management capability permits a system to operate in a low power state without being powered down. Once in the low power state (called "Fast Off" state), the computer appears to be off. For example, the SMM code could turn off the CRT, line printer, hard disk drive's spindle motor, and fans. In addition, the CPU's clock can be governed. To the user, the machine appears to be in the off state. However, the system is actually in an extremely low power state that still permits the CPU to function and maintain communication connections normally associated with today's desktops (e.g., LAN, Modem, or FAX). Programmable options provide power management flexibility. For example, various system events can be programmed to place the system in the low power state or break events can be programmed to wake the system up.

2.0 SIGNAL DESCRIPTION

This section provides a detailed description of each signal. The signals are arranged in functional groups according to their associated interface.

The "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

The following notations are used to describe the signal type.

in Input is a standard input-only signal

out Totem Pole output is a standard active driver

o/d Open Drain input/output

t/s Tri-State is a bi-directional, tri-state input/output pin

s/t/s Sustained Tri-State is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. An external pull-up is required to sustain the inactive state until another agent drives it and must be provided by the central resource.

2.1 PCI Bus Interface Signals

Pin Name	Type	Description
PCICLK	in	**PCI CLOCK:** PCICLK provides timing for all transactions on the PCI Bus. All other PCI signals are sampled on the rising edge of PCICLK and all timing parameters are defined with respect to this edge. Frequencies supported by the PCEB range from 25 to 33 MHz.
PCIRST#	in	**PCI RESET:** PCIRST# forces the PCEB into a known state. All t/s and s/t/s signals are forced to a high impedance state, and the s/o/d signals are allowed to float high. The PCEB negates all GNT# lines to the PCI Bus and the PCEB negates its internal request. The PCEB drives AD[31:0], C/BE[3:0]#, and PAR during reset to keep these signals from floating (depending on the state of CPUREQ# and REQ1#—as described in the following paragraph). As long as PCIRST# is asserted, the PCEB drives the AD[31:0] signals to keep them from floating. Note that CPUREQ# must be sampled high when PCIRST# is asserted. All PCEB registers are set to their default values. PCIRST# may be asynchronous to PCICLK when asserted or negated. Although asynchronous, the negation of PCIRST# must be a clean, bounce-free edge. PCIRST# must be asserted for a minimum 1 μs, and PCICLK must be active during the last 100 μs of the PCIRST# pulse.
AD[31:0]	t/s	**ADDRESS AND DATA:** AD[31:0] is a multiplexed address and data bus. During the first clock of a transaction, AD[31:0] contain a physical address. During subsequent clocks, AD[31:0] contain data. A PCEB bus transaction consists of an address phase followed by one or more data phases. Little-endian byte ordering is used. AD[7:0] define the least significant byte (LSB) and AD[31:24] the most significant byte (MSB). The information contained in the two low order address bits varies by address space. In the I/O address space, AD[1:0] are used to provide full byte address. In the memory and configuration address space, AD[1:0] are driven "00" during the address phase. The other three encodings are reserved. See Section 5.0, PCI Interface for more details. When the PCEB is a target, AD[31:0] are inputs during the address phase of a transaction. During the following data phase(s), the PCEB may be asked to supply data on AD[31:0] as for a PCI read, or accept data as for a PCI write. As an Initiator, the PCEB drives a valid address on AD[31:0] (with exceptions related to AD[1:0]) during the address phase, and drives write or latches read data on AD[31:0] during the data phase. When PCIRST# is asserted, the PCEB drives the AD[31:0] signals to keep them from floating. In addition, the PCEB acts as the central resource responsible for driving the AD[31:0] signals when no device owns the PCI Bus and the bus is idle.

Pin Name	Type	Description
C/BE[3:0]#	t/s	**BUS COMMAND AND BYTE ENABLES:** The command and byte enable signals are multiplexed on the same PCI pins. During the address phase of a transaction, C/BE[3:0]# define the bus command for bus command definitions. During the data phase, C/BE[3:0]# are used as Byte Enables. The Byte Enables determine which byte lanes carry meaningful data. C/BE[0]# applies to byte 0 and C/BE[3]# to byte 3. C/BE[3:0]# are not used for address decoding. The PCEB drives C/BE[3:0]# as an initiator of a PCI Bus cycle and monitors C/BE[3:0]# as a target. When PCIRST# is asserted, the PCEB drives C/BE[3:0]# to keep them from floating. In addition, the PCEB acts as the central resource responsible for driving the C/BE[3:0]# signals when no device owns the PCI Bus and the bus is idle
FRAME#	s/t/s	**FRAME:** FRAME# is driven by the current initiator to indicate the beginning and duration of an access. FRAME# is asserted to indicate that a bus transaction is beginning. During a transaction, data transfers continue while FRAME# is asserted. When FRAME# is negated, the transaction is in the final data phase. FRAME# is an input when the PCEB is the target. FRAME# is an output when the PCEB is the initiator. During reset, this signal is tri-stated.
TRDY#	s/t/s	**TARGET READY:** TRDY#, as an output, indicates the target's ability to complete the current data phase of the transaction. TRDY# is used in conjunction with IRDY#. A data phase is completed on any clock that both TRDY# and IRDY# are sampled asserted. When PCEB is the target during a read cycle, TRDY# indicates that the PCEB has valid data present on AD[31:0]. During a write, it indicates that the PCEB, as a target, is prepared to latch data. TRDY # is an input to the PCEB when the PCEB is the initiator. During reset, this signal is tri-stated.
IRDY#	s/t/s	**INITIATOR READY:** IRDY#, as an output, indicates the initiator's ability to complete the current data phase of the transaction. IRDY# is used in conjunction with TRDY#. A data phase is completed on any clock that both IRDY# and TRDY# are sampled asserted. When PCEB is the initiator of a write cycle, IRDY# indicates that the PCEB has valid data present on AD[31:0]. During a read, it indicates the PCEB is prepared to latch data. IRDY# is an input to the PCEB when the PCEB is the target. During reset, this signal is tri-stated.
STOP#	s/t/s	**STOP:** As a target, the PCEB asserts STOP# to request that the master stop the current transaction. When the PCEB is an initiator, STOP# is an input. As an initiator, the PCEB stops the current transaction when STOP# is asserted. Different semantics of the STOP# signal are defined in the context of other handshake signals (TRDY# and DEVSEL#). During reset, this signal is tri-stated.
PLOCK#	s/t/s	**PCI LOCK:** PLOCK# indicates an atomic operation that may require multiple transactions to complete. PLOCK# is an input when PCEB is the target and output when PCEB is the initiator. When PLOCK# is sampled negated during the address phase of a transaction, a PCI agent acting as a target will consider itself a locked resource until it samples PLOCK# and FRAME# negated. When other masters attempt accesses to the PCEB (practically to the EISA subsystem) while the PCEB is locked, the PCEB responds with a retry termination. During reset, this signal is tri-stated.

Pin Name	Type	Description
IDSEL	in	**INITIALIZATION DEVICE SELECT:** IDSEL is used as a chip select during configuration read and write transactions. The PCEB samples IDSEL during the address phase of a transaction. If the PCEB samples IDSEL asserted during a configuration read or write, the PCEB responds by asserting DEVSEL# on the next cycle.
DEVSEL#	s/t/s	**DEVICE SELECT:** The PCEB asserts DEVSEL# to claim a PCI transaction as a result of positive or subtractive decode. As an output, the PCEB asserts DEVSEL# when it samples IDSEL asserted during configuration cycles to PCEB configuration registers. As an input, DEVSEL# indicates the response to a PCEB-initiated transaction. The PCEB, when not a master, samples this signal for all PCI transactions to decide whether to subtractively decode the cycle (except for configuration and special cycles). During reset, this signal is tri-stated.
PAR	t/s	**PARITY:** PAR is even parity across AD[31:0] and C/BE[3:0]#. When acting as a master, the PCEB drives PAR during the address and write data phases. As a target, the PCEB drives PAR during read data phases. When PCIRST# is asserted, the PCEB drives the PAR signal to keep it from floating. The PCEB acts as the central resource responsible for driving the PAR signal when no other device is granted the PCI Bus and the bus is idle. Note that the driving and tri-stating of the PAR signal is always one clock delayed from the corresponding driving and tri-stating of the AD[31:0] and C/BE[3:0]# signals.
PERR#	s/t/s	**PARITY ERROR:** PERR# reports data parity errors on all transactions, except special cycles. This signal can only be asserted (by the agent receiving data) two clocks following the data (which is one clock following the PAR signal that covered the data). The duration of PERR# is one clock for each data phase that a data parity error is detected. (If multiple data errors occur during a single transaction the PERR# signal is asserted for more than a single clock.) PERR# must be driven high for one clock before being tri-stated. During reset, this signal is tri-stated.

2.2 PCI Arbiter Signals

Pin Name	Type	Description
CPUREQ#	in	**CPU REQUEST:** CPUREQ# asserted indicates that the Host CPU requests use of the PCI Bus. During PCIRST#, this signal must be sampled high by the PCEB. When PCIRST# is asserted (and CPUREQ# is sampled high), the PCEB drives the AD, C/BE#, and PAR signals to keep them from floating.
REQ[3:0]#	in	**REQUEST:** A bus master asserts the corresponding request signal to request the PCI Bus.
CPUGNT#	out	**CPU GRANT:** The PCEB asserts CPUGNT# to indicate that the CPU master (Host Bridge) has been granted the PCI Bus. During PCI reset, CPUGNT# is tri-stated.
GNT[3:0]#	out	**GRANT:** The PCEB asserts one of the GNT[3:0] signals to indicate that the corresponding PCI master has been granted the PCI Bus. During PCI reset, these signals are tri-stated.
MEMREQ#	out	**MEMORY REQUEST:** If the PCEB is configured in Guaranteed Access Time (GAT) Mode, MEMREQ# is asserted when an EISA device or DMA requests the EISA Bus. The PCEB asserts this signal (along with FLSHREQ#) to indicate that the PCEB requires ownership of main memory. The PCEB asserts FLSHREQ# concurrently with asserting MEMREQ#. This signal is synchronous to the PCI clock. During reset, this signal is driven high. **FLSHREQ# / MEMREQ# / Meaning** 1 / 1 / Idle 0 / 1 / Flush buffers pointing towards PCI to avoid ISA deadlock 1 / 0 / GAT enabled or disabled: For buffer coherency in APIC systems, the buffers pointing to main memory must be flushed and disabled for the duration of assertion. 0 / 0 / GAT mode: Guarantee PCI Bus immediate access to main memory (this may or may not require the PCI-to-main memory buffers to be flushed first, depending on the number of buffers).
FLSHREQ#	out	**FLUSH REQUEST:** FLSHREQ# is asserted by the PCEB to command all of the system's posted write buffers pointing towards PCI to be flushed. This is required before granting the EISA Bus to an EISA master or the DMA. Note that, for APIC related buffer flush requests, this signal is negated. This signal is synchronous to the PCI clock. During reset, this signal is driven high.

Pin Name	Type	Description
MEMACK#	in	**MEMORY ACKNOWLEDGE:** MEMACK# is the response handshake that indicates to the PCEB that the function requested over the MEMREQ# and/or FLSHREQ# signals has been completed. If the PCEB is configured for Guaranteed Access Time Mode through the Arbiter Control Register, and both MEMREQ# and FLSHREQ# are asserted, the assertion of MEMACK# indicates to the PCEB that ownership of main memory has been granted and that all system buffers have been flushed and temporarily disabled. If MEMACK# is asserted in response to assertion of MEMREQ# (GAT either enabled or disabled), it indicates that the system's buffers pointing towards the main memory are flushed and temporarily disabled so that APIC can proceed with the interrupt message sequence. If FLSHREQ# is asserted and MEMREQ# is not asserted (with GAT mode being either enabled or disabled), the assertion of MEMACK# indicates that the system's posted write buffers pointing towards PCI are flushed and temporarily disabled, and the EISA Bus can be granted to an EISA master or DMA. This signal is synchronous to the PCI clock.

2.3 Address Decoder Signals

Pin Name	Type	Description
MEMCS#	out	**MEMORY CHIP SELECT:** MEMCS# is a programmable address decode signal provided to a Host CPU bridge. A Host bridge can use MEMCS# to forward a PCI cycle to the main memory behind the bridge. MEMCS# is asserted one PCI clock after FRAME# is sampled asserted (address phase) and is valid for one clock cycle before being negated. MEMCS# is driven high during reset.
PIODEC#	in	**PCI I/O SPACE DECODER:** PIODEC# can be used to provide arbitrarily complex EISA-to-PCI I/O address space mapping. This signal can be connected to the decode select output of an external I/O address decoder. When PIODEC# is asserted during an EISA I/O cycle, that cycle is forwarded to the PCI Bus. Note that an external pull-up resistor is required if this input signal is not used (i.e., not driven by the external logic).

2.4 EISA Interface Signals

Pin Name	Type	Description
BCLK	in	**BUS CLOCK:** BCLK is the system clock used to synchronize events on the EISA Bus. The ESC device generates BCLK (BCLKOUT), which is a divided down clock from a PCICLK. BCLK runs at a frequency that is dependent on PCICLK and a selected division factor (within the ESC). For example, a 25 MHz PCICLK and a division factor of 3 results in an 8.33 MHz BCLK.
START#	t/s	**START:** START# provides timing control at the start of the cycle and remains asserted for one BCLK period. When the PCEB is an EISA master, START# is an output signal. START# is asserted after LA[31:24]#, LA[23:2] and M/IO# become valid. START# is negated on the rising edge of the BCLK, one BCLK after it was asserted. The trailing edge of START# is always delayed from the rising edge of BCLK. When the PCEB is an EISA master, for cycles to a mismatched slave (see note at the end of this section), START# becomes an input signal at the end of the first START# phase and remains an input until the negation of the last CMD#. The ESC gains the control of the transfer and generates START#. When the PCEB is an EISA slave, START# is an input signal. It is sampled on the rising edge of BCLK. Upon PCIRST#, this signal is tri-stated and placed in output mode.
CMD#	in	**COMMAND:** CMD# provides timing control within the cycle. In all cases, CMD# is an input to the PCEB from the ESC. CMD# is asserted from the rising edge of BCLK, simultaneously with the negation of START#, and remains asserted until the end of the cycle.
M/IO#	t/s	**MEMORY OR I/O:** M/IO# identifies the current cycle as a memory or an I/O cycle. M/IO# is pipelined from one cycle to the next and must be latched by the slave. M/IO# = 1 indicates a memory cycle and M/IO# = 0 indicates an I/O cycle. When the PCEB is an EISA master, the M/IO# is an output signal. When the PCEB is an EISA slave, M/IO# is an input signal. The PCEB responds as an EISA slave for both memory and I/O cycles. Upon PCIRST#, this signal is tri-stated and is placed in output mode.
W/R#	t/s	**WRITE OR READ:** W/R# identifies the cycle as a write or a read cycle. The W/R# signal is pipelined from one cycle to the next and must be latched by the slave. W/R# = 1 indicates a write cycle and W/R# = 0 indicates a read cycle. When the PCEB is an EISA master, W/R# is an output signal. When the PCEB is an EISA slave, W/R# is an input signal. Upon PCIRST#, this signal is tri-stated and placed in output mode.

Pin Name	Type	Description
EXRDY	od	**EISA READY:** EXRDY is used by EISA I/O and memory slaves to request wait states during a cycle. Each wait state is a BCLK period. The PCEB, as an EISA master or slave, samples EXRDY. As an input, the EXRDY is sampled on the falling edge of BCLK after the CMD# has been asserted, and if inactive, each falling edge thereafter. When PCEB is an EISA slave, it may drive EXRDY low to introduce wait states. During reset, this signal is not driven.
EX32#	od	**EISA 32 BIT:** EX32# is used by the EISA slaves to indicate support of 32 bit transfers. When the PCEB is an EISA master, it samples EX32# on the same rising edge of BCLK that START# is negated. During mismatched cycles (see note at the end of this section), EX32# (and EX16#) is used to transfer the control back to the PCEB. EX32# (along with EX16#) is asserted by the ESC on the falling edge of BCLK before the rising edge of the BCLK when the last CMD# is negated. This indicates that the cycle control is transferred back to the PCEB. As an EISA slave, the PCEB always drives EX32# to indicate 32 bit support for EISA cycles. During reset, this signal is not driven.
EX16#	in	**EISA 16 BIT:** EX16# is used by the EISA slaves to indicate their support of 16 bit transfers. As an EISA master, the PCEB samples EX16# on the same rising edge of BCLK that START# is negated. During mismatched cycles (see note at the end of this section), EX16# (and EX32#) is used to transfer the control back to the PCEB. EX16# (along with EX32#) is asserted by the ESC on the falling edge of the BCLK before the rising edge of the BCLK when the last CMD# is negated. This indicates that the cycle control is transferred back to the PCEB. As an EISA slave, the PCEB never asserts EX16#.
MSBURST#	t/s	**MASTER BURST:** MSBURST# is an output when the PCEB is an EISA master and an input when the PCEB is a slave. As a master, the PCEB asserts MSBURST# to indicate to the slave that the next cycle is a burst cycle. If the PCEB samples SLBURST# asserted on the rising edge of BCLK after START# is asserted, the PCEB asserts MSBURST# on the next BCLK edge and proceeds with the burst cycle. As a slave, the PCEB monitors this signal in response to the PCEB asserting SLBURST#. The EISA master asserts MSBURST# to the PCEB to indicate that the next cycle is a burst cycle. As a slave, the PCEB samples MSBURST# on the rising edge of BCLK after the rising edge of BCLK that CMD# is asserted by the ESC. MSBURST# is sampled on all subsequent rising edges of BCLK until the signal is sampled negated. The burst cycle is terminated on the rising edge of BCLK when MSBURST# is sampled negated, unless EXRDY is sampled negated on the previous falling edge of BCLK. During reset, this signal is tri-stated.

Pin Name	Type	Description
SLBURST#	t/s	**SLAVE BURST:** SLBURST# is an input when the PCEB is an EISA master and an output when the PCEB is a slave. When the PCEB is a master, the slave indicates that it supports burst cycles by asserting SLBURST# to the PCEB. The PCEB samples SLBURST# on the rising edge of BCLK at the end of START# for EISA master cycles. When the PCEB is an EISA slave, this signal is an output. As a slave, the PCEB asserts this signal to the master indicating that the PCEB supports EISA burst cycles. During reset, this signal is tri-stated.
LOCK#	t/s	**LOCK:** When asserted, LOCK# guarantees exclusive memory access. This signal is asserted by the PCEB when the PCI master is running locked cycles to EISA slaves. When asserted, this signal locks the EISA subsystem. LOCK# can also be activated by a device on the EISA Bus. This condition is propagated to the PCI Bus via the PLOCK# signal. During reset, this signal is tri-stated.
BE[3:0]#	t/s	**BYTE ENABLES:** BE[3:0]# identify the specific bytes that are valid during the current EISA Bus cycles. When the PCEB is an EISA master and the cycles are directed to a matched slave (slave supports 32-bit transfers), the BE[3:0]# are outputs from the PCEB. When the cycles are directed to a mis-matched slave (slave does not support 32-bit transfers - see note), the BE[3:0]# are floated one and half BCLKs after START# is asserted. These signals become inputs (driven by the ESC) for the rest of the cycle. BE[3:0]# are pipelined signals and must be latched by the addressed slave. When the PCEB is an EISA/ISA/DMA slave, BE[3:0]# are inputs to the PCEB. Upon PCIRST#, these signals are tri-stated and placed in output mode.
LA[31:24]#, LA[23:2]	t/s	**LATCHABLE ADDRESS:** LA[31:24]# and LA[23:2] are the EISA address signals. When the PCEB is an EISA master, these signals are outputs from the PCEB. These addresses are pipelined and must be latched by the EISA slave. LA[31:24]# and LA[23:2] are valid on the falling edge of START#. Note that the upper address bits are inverted before being driven on LA[31:24]#. The timing for LA[31:24] and LA[23:2] are the same. When the PCEB is an EISA slave, these signals are inputs and are latched by the PCEB. For I/O cycles, the PCEB, as an EISA master, floats LA[31:24]# to allow for ESC's address multiplexing (during I/O cycle to configuration RAM). LA[23:2] are actively driven by the PCEB. For memory cycles, the PCEB as an EISA master, drives the LA address lines. During reset, these signals are tri-stated.

Pin Name	Type	Description
SD[31:0]	t/s	**SYSTEM DATA:** SD[31:0] are bi-directional data lines that transfer data between the PCEB and other EISA devices. Data transfer between EISA and PCI devices use these signals. The data swapping logic in the PCEB ensures that the data is available on the correct byte lanes for any given transfer. During reset, these signals are tri-stated.
REFRESH#	in	**REFRESH:** When asserted, REFRESH# indicates to the PCEB that the current cycle on the EISA Bus is a refresh cycle. It is used by the PCEB decoder to distinguish between EISA memory read cycles and refresh cycles.

NOTE:
Mis-matched Cycles. When the PCEB is an EISA master, cycles to the slaves, other than 32 bits transfers, are considered a mis-matched cycle. For mis-matched cycles, the PCEB backs off the EISA Bus one and half BCLKs after it asserted START# by releasing (floating) START#, BE[3:0]# and the SD[31:0] lines. The ESC device then takes control of the transfer. The ESC controls the transfer until the last transfer. At the end of the last transfer, the control is transferred back to the PCEB. The ESC transfers control back to the PCEB by asserting EX32# and EX16# on the falling edge of BCLK before the rising edge of BCLK when the last CMD# is negated.

2.5 ISA Interface Signals

An ISA interface signal is included to improve the PCEB's handling of I/O cycles on the EISA side of the bridge. This signal permits ISA masters to address PCI I/O slaves using the full 16-bit bus size. The signal also allows the PCEB to identify 8-bit I/O slaves for purposes of generating the correct amount of I/O recovery.

Pin Name	Type	Description
IO16#	o/d	**16-BIT I/O CHIP SELECT:** As an EISA slave, the PCEB asserts IO16# when PIODEC# is asserted or an I/O cycle to PCI is detected. As an EISA master, the PCEB uses IO16# as an input to determine the correct amount of I/O recovery time from the I/O Recovery Time (IORT) Register. This register contains bit-fields that are used to program recovery times for 8-bit and 16-bit I/O. When IO16# is asserted, the recovery time programmed into the 16-bit I/O field (bits [1:0]) , if enabled, is used. When IO16# is negated, the recovery time programmed into the 8-bit I/O field (bits [5:3]), if enabled, is used. This signal must have an external pull-up resistor. During reset, this signal is not driven.

2.6 PCEB/ESC Interface Signals

Pin Name	Type	Description
ARBITRATION AND INTERRUPT ACKNOWLEDGE CONTROL		
EISAHOLD	in	**EISA HOLD:** EISAHOLD is used by the ESC to request control of the EISA Bus from the PCEB. This signal is synchronous to PCICLK and is driven inactive when PCIRST# is asserted.
EISAHLDA	out	**EISA HOLD ACKNOWLEDGE:** The PCEB asserts EISAHLDA to inform the ESC that it has been granted ownership of the EISA Bus. This signal is synchronous to the PCICLK.
PEREQ#/ INTA#	out	**PCI-TO-EISA REQUEST OR INTERRUPT ACKNOWLEDGE:** PEREQ#/INTA# is a dual-function signal. The signal function is determined by the state of EISAHLDA signal. When EISAHLDA is negated, this signal is an interrupt acknowledge (i.e., PEREQ#/ INTA# asserted indicates to the ESC that the current cycle on the EISA is an interrupt acknowledge). When EISAHLDA is asserted, this signal is a PCI-to-EISA request (i.e. PEREQ#/ INTA# asserted indicates to the ESC that the PCEB needs to obtain the ownership of the EISA Bus on behalf of a PCI agent). This signal is synchronous to the PCICLK and it is driven inactive when PCIRST# is asserted.
STPGNT#	out	**STOP GRANT ACKNOWLEDGE:** STPGNT# is asserted when the PCEB receives a STOP GRANT PCI special cycle for one PCICLK period. This signal is only asserted when the PCI AD[31:0] signals equal 00120002h during the first data phase of the PCI special cycle. Data of 00120002h on AD[31:0] in subsequent data phases during a PCI special cycle does not result in the assertion of STPGNT#.

PIN NAME	Type	Description
PCEB BUFFER COHERENCY CONTROL		
NMFLUSH#	t/s	**NEW MASTER FLUSH:** The bi-directional NMFLUSH# signal provides handshake between the PCEB and ESC to control flushing of PCI system buffers on behalf of EISA masters. During an EISA Bus ownership change, before the ESC can grant the bus to the EISA master (or DMA), the ESC must ensure that system buffers are flushed and the buffers pointing towards the EISA subsystem are disabled. The ESC asserts NMFLUSH# for one PCI clock to request system buffer flushing. (After asserting NMFLUSH# for 1 PCI clock, the ESC tri-states NMFLUSH#.) When the PCEB samples NMFLUSH# asserted, it starts immediately to assert NMFLUSH# and begins flushing its internal buffers, if necessary. The PCEB also requests PCI system buffer flushing via the MEMREQ#, FLSHREQ#, and MEMACK# signals. When the PCEB completes its internal buffer flushing and MEMACK# is asserted (indicating that the PCI system buffer flushing is complete), the PCEB negates NMFLUSH# for 1 PCI clock and stops driving it. When the ESC samples NMFLUSH# negated, it grants the EISA Bus to an EISA master (or DMA). The ESC resumes responsibility of the default NMFLUSH# driver and starts driving NMFLUSH# negated until the next time a new EISA master (or DMA) wins arbitration. This signal is synchronous with PCICLK and is negated by the ESC at reset.
AFLUSH#	t/s	**APIC FLUSH:** AFLUSH# is bi-directional signal between the PCEB and ESC that controls system buffer flushing on behalf of the APIC. After a reset the ESC negates AFLUSH# until the APIC is initialized and the first interrupt request is recognized.

Pin Name	Type	Description
DATA SWAP BUFFER CONTROL		
SDCPYEN01# SDCPYEN02# SDCPYEN03# SDCPYEN13#	in	**COPY ENABLE:** These active Low signals perform byte copy operation on the EISA data bus (SD[31:0]). The Copy Enable signals are asserted during mis-matched cycles and are used by the PCEB to enable byte copy operations between the SD data byte lanes 0, 1, 2, and 3 as follows: SDCPYEN01#: Copy between Byte Lane 0 (SD[7:0]) and Byte Lane 1 (SD[15:8]) SDCPYEN02#: Copy between Byte Lane 0 (SD[7:0]) and Byte Lane 2 (SD[23:16]) SDCPYEN03#: Copy between Byte Lane 0 (SD[7:0]) and Byte Lane 3 (SD[31:24]) SDCPYEN13#: Copy between Byte Lane 1 (SD[15:8]) and Byte Lane 3 (SD[31:24]) Note that the direction of the copy is controlled by SDCPYUP.
SDCPYUP	in	**SYSTEM DATA COPY UP:** SDCPYUP controls the direction of the byte copy operation. A high on SDCPYUP indicates a COPY UP operation where the lower byte(s) of the SD data bus are copied onto the higher byte(s) of the bus. A low on the signal indicates a COPY DOWN operation where the higher byte(s) of the data bus are copied on to the lower byte(s) of the bus. The PCEB uses this signal to perform the actual data byte copy operation during mis-matched cycles.
SDOE[2:0]#	in	**SYSTEM DATA OUTPUT ENABLE:** These active Low signals enable the SD data output onto the EISA Bus. The ESC only activates these signals during mis-matched cycles. The PCEB uses these signal to enable the SD data buffers as follows: SDOE0# Enables byte lane 0 SD[7:0] SDOE1# Enables byte lane 1 SD[15:8] SDOE2# Enables byte lane 3 SD[31:24] and byte lane 2 SD[23:16]
SDLE[3:0]#	in	**SYSTEM DATA LATCH ENABLE:** SDLE[3:0]# enable the latching of data on the EISA Bus. These signals are activated only during mis-matched cycles, except PCEB-initiated write cycles. The PCEB uses these signals to latch the SD data bus as follows: SDLE0# Latch byte lane 0 SD[7:0] SDLE1# Latch byte lane 1 SD[15:8] SDLE2# Latch byte lane 2 SD[23:16] SDLE3# Latch byte lane 3 SD[31:24]

2.7 Test Signal

Pin Name	Type	Description
TEST#	in	**TEST:** This pin is used to tri-state all PCEB outputs. During normal operations, this pin must be tied high.

3.0 REGISTER DESCRIPTION

The PCEB contains both PCI configuration registers and I/O registers. The configuration registers (Table 1) are located in PCI configuration space and are only accessible from the PCI Bus. The addresses shown in the table for each register are offset values that appear on AD[7:2] and C/BE[3:0]#. The configuration registers can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities. All multi-byte numeric fields use "little-endian" ordering (i.e., lower addresses contain the least significant parts of the fields).

The BIOS Timer is the only non-configuration register (Section 3.2, I/O Registers). This register, like the configuration registers, is only accessible from the PCI Bus. The BIOS Timer Register can be accessed as byte, word, or Dword quantities.

Some of the PCEB registers contain reserved bits. These bits are labeled "Reserved". Software must take care to deal correctly with bit-encoded fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bits are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and the data then written back.

In addition to reserved bits within a register, the PCEB contains address locations in the PCI configuration space that are marked "Reserved" (Table 1). The PCEB responds to accesses to these address locations by completing the PCI cycle. When a reserved register location is read, 0000h is returned. Writes have no affect on the PCEB.

During a hard reset (PCIRST# asserted), the PCEB registers are set to pre-determined **default** states. The default values are indicated in the individual register descriptions.

During the address phase of a configuration cycle, Bits [10:8] encode one of eight possible functions on a device. The PCEB only supports one function; that of a bridge between the PCI and EISA/ISA Busses. This function has the code of 000. Thus, for accessing PCEB configuration registers, Bits[10:8] = 000 of the address. If the PCEB IDSEL is asserted and any of the above three bits is 1, the PCEB returns all zeros for a read and does not respond to a write.

3.1 Configuration Registers

Table 1 summarizes the PCEB configuration space registers. Following the table, is a detailed description of each register and register bit. The register descriptions are arranged in the order that they appear in Table 1. The following nomenclature is used for access attributes.

RO **Read Only**. If a register is read only, writes to this register have no effect.

R/W **Read/Write**. A register with this attribute can be read and written.

R/WC **Read/Write Clear**. A register bit with this attribute can be read and written. However, a write of a 1 clears (sets to 0) the corresponding bit and a write of a 0 has no effect.

NOTE:

Some register fields are used to program address ranges for various PCEB functions. The register contents represent the address bit value and not the signal level on the bus. For example, the upper address lines on the EISA Bus have inverted signals (LA[31:24]#). However, this inversion is automatically handled by the PCEB hardware and is transparent to the programmer.

Table 1. Configuration Registers

Address Offset	Abbreviation	Register Name	Access
00–01h	VID	Vendor Identification	RO
02–03h	DID	Device Identification	RO
04–05h	PCICMD	Command Register	R/W
06–07h	PCISTS	Status Register	RO, R/WC
08h	RID	Revision Identification	RO
09–0Ch	—	Reserved	—
0Dh	MLTIM	Master Latency Timer	R/W
0E–3Fh	—	Reserved	—
40h	PCICON	PCI Control	R/W
41h	ARBCON	PCI Arbiter Control	R/W
42h	ARBPRI	PCI Arbiter Priority Control	R/W
43h	ARBPRIX	PCI Arbiter Priority Control Extension	R/W
44h	MCSCON	MEMCS# Control	R/W
45h	MCSBOH	MEMCS# Bottom of Hole	R/W
46h	MCSTOH	MEMCS# Top of Hole	R/W
47h	MCSTOM	MEMCS# Top of Memory	R/W
48–49h	EADC1	EISA Address Decode Control 1	R/W
4A–4Bh	—	Reserved	—
4Ch	IORTC	ISA I/O Recovery Time Control	R/W
4Dh–53h	—	Reserved	—
54h	MAR1	MEMCS# Attribute Register #1	R/W
55h	MAR2	MEMCS# Attribute Register #2	R/W
56h	MAR3	MEMCS# Attribute Register #3	R/W
57h	—	Reserved	—
58h	PDCON	PCI Decode Control	R/W
59h	—	Reserved	—
5Ah	EADC2	EISA Address Decode Control 2	R/W
5Bh	—	Reserved	—

Table 1. Configuration Registers (Continued)

Address Offset	Abbreviation	Register Name	Access
5Ch	EPMRA	EISA-to-PCI Memory Region Attributes	R/W
5D–5Fh	—	Reserved	—
60–6Fh	MEMREGN[4:1]	EISA-to-PCI Memory Region Address (4 Registers)	R/W
70–7Fh	IOREGN[4:1]	EISA-to-PCI I/O Region Address (4 Registers)	R/W
80–81h	BTMR	BIOS Timer Base Address	R/W
84h	ELTCR	EISA Latency Timer Control Register	R/W
85–87h	—	Reserved	—
88–8Bh	PTCR	PCEB Test Control Register—**DO NOT WRITE**	—
8C–FFh	—	Reserved	—

3.1.1 VID—VENDOR IDENTIFICATION REGISTER

Address Offset: 00–01h
Default Value: 8086h
Attribute: Read Only
Size: 16 bits

The VID Register contains the vendor identification number. This register, along with the Device Identification Register, uniquely identify any PCI device. Writes to this register have no effect.

Bit	Description
15:0	**Vendor Identification Number:** This is a 16-bit value assigned to Intel.

3.1.2 DID—DEVICE IDENTIFICATION REGISTER

Address Offset: 02–03h
Default Value: 0482h
Attribute: Read Only
Size: 16 bits

The DID Register contains the device identification number. This register, along with the VID Register, define the PCEB. Writes to this register have no effect.

Bit	Description
15:0	**Device Identification Number:** This is a 16-bit value assigned to the PCEB.

3.1.3 PCICMD—PCI COMMAND REGISTER

Address Offset: 04–05h
Default Value: 0007h
Attribute: Read/Write, Read Only
Size: 16 bits

This 16-bit register contains PCI interface control information. This register enables/disables PCI parity error checking, enables/disables PCEB bus master capability, and enables/disables the PCEB to respond to PCI-originated memory and I/O cycles. Note that, for certain PCI functions that are not implemented within the PCEB, the control bits are still shown (labeled "not supported").

Bit	Description
15:9	**Reserved**
8	**SERR# Enable (SERRE)—Not Supported—RO:** Function of this bit is to control the SERR# signal. Since the PCEB does not implement the SERR# signal, this bit always reads as 0 (disabled).
7	**Wait State Control (WSC)—Not Supported—RO:** This bit controls insertion of wait-states for devices that do not meet the 33-10 PCI specification. Since PCEB meets the 33-10 specification, this control function is not implemented. WSC is always read as 0.
6	**Parity Error Enable (PERRE)—R/W:** PERRE controls the PCEB's response to PCI parity errors. When PERRE = 1, the PCEB asserts the PERR# signal when a parity error is detected. When PERRE = 0, the PCEB ignores any parity errors that it detects. After PCIRST#, PERRE = 0 (parity checking disabled).
5	**VGA Palette Snoop (VGPS)—Not Supported—RO:** This bit is intended only for specific control of PCI-based VGA devices and it is not applicable to the PCEB. This bit is not implemented and always reads as 0.
4	**Memory Write and Invalidate Enable (MWIE)—Not Supported—RO:** This is an enable bit for using the Memory Write and Invalidate command. The PCEB doesn't support this command as a master. As a slave the PCEB aliases this command to a memory write. This bit always reads as 0 (disabled).
3	**Special Cycle Enable (SCE)—Not Supported—RO:** Since this capability is not implemented, the PCEB does not respond to any type of special cycle. This bit always reads as 0.
2	**Bus Master Enable (BME)—R/W:** ME enables/disables the PCEB's PCI Bus master capability. When BME = 0, the PCEB bus master capability is disabled. This prevents the PCEB from requesting the PCI Bus on behalf of EISA/ISA masters, the DMA, or the Line Buffers. When BME = 1, the bus master capability is enabled. This bit is set to 1 after PCIRST#.
1	**Memory Space Enable (MSE)—R/W:** This bit enables the PCEB to accept PCI-originated memory cycles. When MSE = 1, the PCEB responds to PCI-originated memory cycles to the EISA Bus. When MSE = 0, the PCEB does not respond to PCI-originated memory cycles to the EISA Bus (DEVSEL# is inhibited). This bit is set to 1 (enabled for BIOS access) after PCIRST#.
0	**I/O Space Enable (IOSE)—R/W:** This bit enables the PCEB to accept PCI-originated I/O cycles. When IOSE = 1, the PCEB responds to PCI-originated I/O cycles. When IOSE = 0, the PCEB does not respond to a PCI I/O cycle (DEVSEL# is inhibited), including I/O cycles bound for the EISA Bus. This bit is set to 1 (I/O space enabled) after PCIRST#.

3.1.4 PCISTS—PCI STATUS REGISTER

Address Offset: 06–07h
Default Value: 0200h
Attribute: Read Only, Read/Write Clear
Size: 16 bits

This 16-bit register provides status information for PCI Bus-related events. Some bits are read/write clear. These bits are set to 0 whenever the register is written, and the data in the corresponding bit location is 1 (R/WC). For example, to clear bit 12 and not affect any other bits, write the value 0001_0000_0000_0000b to this register. Note that for certain PCI functions that are not implemented in the PCEB, the control bits are still shown (labeled "not supported").

Bit	Description
15	**Parity Error Status (PERRS)—R/WC:** This bit is set to 1 whenever the PCEB detects a parity error, even if parity error handling is disabled (as controlled by bit 6 in the PCI Command Register). Software sets PERRS to 0 by writing a 1 to this bit location.
14	**SERR# Status (SERRS)—Not Supported:** This bit is used to indicate that a PCI device asserted the SERR# signal. The PCEB does not implement this signal. SERRS is always read as 0.
13	**Master Abort Status (MA)—R/WC:** When the PCEB, as a master, generates a master abort, this bit is set to 1. Software sets MA to 0 by writing a 1 to this bit location.
12	**Received Target Abort Status (RTAS)—R/WC:** When the PCEB, as a master, receives a target abort condition, this bit is set to 1. Software sets RTAS to 0 by writing a 1 to this bit location.
11	**Signaled Target Abort Status (STAS)—Not Supported:** This bit is set to 1 by a PCI target device when they generate a Target Abort. Since the PCEB never generates a target abort, this bit is not implemented and will always be read as a 0.
10:9	**DEVSEL Timing Status (DEVT)—RO:** This read only field indicates the timing of the DEVSEL# signal when PCEB responds as a target. The PCI Specification defines three allowable timings for assertion of DEVSEL#: 00b = fast, 01b = medium, and 10b = slow (11b is reserved). DEVT indicates the slowest time that a device asserts DEVSEL# for any bus command, except configuration read and configuration write cycles. The PCEB implements medium speed DEVSEL# timing and, therefore, DEVT[10:9] = 01 when read.
8:0	**Reserved**

3.1.5 RID—REVISION IDENTIFICATION REGISTER

Address Offset: 08h
Default Value: 03h (82375EB, A-2 stepping)
04h (82375SB, B-0 stepping)
Attribute: Read Only
Size: 8 bits

This 8-bit register contains the device revision number of the PCEB. Writes to this register have no effect.

Bit	Description
7:0	**Revision Identification Number:** This 8-bit value is the revision number of the PCEB.

3.1.6 MLT—MASTER LATENCY TIMER REGISTER

Address Offset: 0Dh
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This 8-bit register contains the programmable value of the Master Latency Timer for use when the PCEB is a master on the PCI Bus. The granularity of the timer is 8 PCI clocks. Thus, bits[2:0] are not used and always read as 0s.

Bit	Description
7:3	**Count Value:** This 5-bit field contains the count value of the Master Latency Timer, with a granularity of 8 PCI clocks. For example, value 00101b provides a time-out period of 5x8 = 40 PCI clocks. Maximum count value is 11111b, which corresponds to 248 PCI clocks.
2:0	**Reserved**

3.1.7 PCICON—PCI CONTROL REGISTER

Address Offset: 40h
Default Value: 20h
Attribute: Read/Write
Size: 8 bits

This 8-bit register enables/disables the PCEB's data buffers, defines the subtractive decoding sample point, and enables/disables response to the PCI interrupt acknowledge cycle.

NOTE:
The Line Buffers are typically enabled or disabled during system initialization. These buffers should not be dynamically enabled/disabled during runtime. Otherwise, data coherency can be affected, if a buffer containing valid write data is disabled and then, later, re-enabled.

Bit	Description
7	**Reserved**
6	**EISA-To-PCI Line Buffer Enable (ELBE):** When ELBE = 0, the EISA-to-PCI Line Buffers are disabled and when ELBE = 1, the EISA-to-PCI Line Buffers are enabled. After PCIRST#, the Line Buffers are disabled (ELBE = 0). Note that when ELBE is set to 1, the line buffers are utilized for transfers to or from the regions defined by the REG[4:1] bits in the EPMRA register (offset 5Ch).
5	**Interrupt Acknowledge Enable (IAE):** When IAE = 0, the PCEB decodes PCI interrupt acknowledge cycles in a semi-subtractive manner. When there is data posted in the Line Buffers, the PCEB intervenes in the PCI interrupt acknowledge cycle by generating a retry. The PCEB also initiates a buffer flush operation and will keep generating retries until the buffers are flushed. The PCEB then subtractively decodes the PCI interrupt acknowledge cycle in order to allow an external PCI-based interrupt controller to respond with the vector. If no external PCI-based interrupt controller has responded to the PCI Interrupt Acknowledge cycle at the DEVSEL# sampling point, the cycle is handled by the PCEB in a subtractive decode manner. When IAE = 1, the PCEB positively decodes the interrupt acknowledge cycles and responds to the cycles in the normal fashion (i.e., uses the PEREQ#/INTA# signal to fetch the vector from the ESC, after the internal buffers are flushed).

Bit	Description
4:3	**Subtractive Decoding Sample Point (SDSP):** The SDSP field determines the DEVSEL# sample point, after which an inactive DEVSEL# results in the PCEB forwarding the unclaimed PCI cycle to the EISA Bus (subtractive decoding). This setting should match the slowest device in the system. When the MEMCS# function is enabled, MEMCS# is sampled as well as an early indication of an eventual DEVSEL#. **Bits[4:3] — Operation** 00 — Slow sample point (default value) 01 — Typical sample point 10 — Reserved 11 — Reserved
2	**Reserved.** This bit must be 0 when programming this register.
1:0	**Reserved**

3.1.8 ARBCON—PCI ARBITER CONTROL REGISTER

Address Offset: 41h
Default Value: 80h
Attribute: Read/Write
Size: 8 bits

This register controls the operation of the PCEB's internal PCI arbiter. The register enables/disables auto-PEREQ#, controls the master retry timer, enables/disables CPU bus parking, controls bus lock, and enables/disables the guaranteed access time (GAT) mode for EISA/ISA accesses.

NOTE:

1. For proper system operation, the master retry timer (bits[4:3]) must not be disabled. This field defaults to 00 (disabled) and must be program to either 01, 10, or 11.
2. The PCMC Host bridge device requires that bit 7 be set to 1 (default). However, other chip sets might need to have this function disabled to provide more optimum performance for EISA subsystems. This functionality is built-in to prevent starvation of PCI agents (in particular, the host bridge, i.e., CPU) when EISA masters are performing transactions in the GAT mode. If this function is disabled, the host bridge must be capable of generating the PCI Bus request, even when the Host Bus is not controlled by the CPU (CPU tri-stated all Host Bus signals, or even only address bus, in response to HOLD/AHOLD). The CPU pin that provides an indication of a request for the external bus (e.g. after cache miss) can be used by the host bridge to generate the request for the PCI Bus during GAT mode operations, even when no address lines are driven by the CPU.

Bit	Description
7	**Auto-PEREQ# Control (APC):** APC Enables/Disables control of the auto-PEREQ# function when GAT mode is enabled via bit 0 (GAT = 1). When APC = 1 (and GAT = 1), the PEREQ# signal is asserted whenever the EISAHLDA signal is asserted. When APC = 0, the PEREQ# signal is not automatically asserted but it will be activated upon PCI Bus request from any PCI agent. After PCIRST#, APC = 1 (enabled). See note.
6:5	**Reserved**
4:3	**Master Retry Timer (MRT):** This 2-bit field determines the number of PCICLKs after the first retry that a PCI initiator's bus request will be masked. Note that for proper system operation, this register must be programmed with either 01, 10, 11. **Bits[4:3] Operation** 00 Timer disabled, Retries never masked. (Default) 01 Retries unmasked after 16 PCICLK's. 10 Retries unmasked after 32 PCICLK's. 11 Retries unmasked after 64 PCICLK's.
2	**Bus Park (BP):** When BP = 1, the PCEB will park CPUREQ# on the PCI Bus when it detects the PCI Bus idle. If BP = 0, the PCEB takes responsibility for driving AD, C/BE# and PAR signals upon detection of bus idle state. After PCIRST#, BP = 0 (disabled).
1	**Bus Lock (BL):** When BL = 1, Bus Lock is enabled. The arbiter considers the entire PCI Bus locked upon initiation of any LOCKed transaction. When BL = 0, Resource Lock is enabled. A LOCKed agent is considered a LOCKed resource and other agents may continue normal PCI transactions. After PCIRST#, BL = 0 (disabled).
0	**Guaranteed Access Time (GAT):** When GAT = 1, the PCEB is configured for Guaranteed Access Time mode. This mode guarantees the 2.1 μs CHRDY time-out specification for the EISA/ISA Bus. When the PCEB is a PCI initiator on behalf of an EISA/ISA master, the PCI and main memory bus (host) are arbitrated for in serial and must be owned before the EISA/ISA master is given ownership of the EISA Bus. If the PCEB is not programmed for Guaranteed Access Time (GAT = 0), the EISA/ISA master is first granted the EISA Bus, before the PCI Bus is arbitrated. After a PCIRST#, GAT = 0 (disabled).

3.1.9 ARBPRI—PCI ARBITER PRIORITY CONTROL REGISTER

Address Offset: 42h
Default Value: 04h
Attribute: Read/Write
Size: 8 bits

This register controls the operating modes of the PCEB's internal PCI arbiter. The arbiter consists of four arbitration banks that support up to six masters and three arbitration priority modesfixed priority, rotating priority and mixed priority modes. See Section 5.4, PCI Bus Arbitration for details on programming and using different arbitration modes.

Bit	Description
7	**Bank 3 Rotate Control:** 1 = Enable; 0 = Disable
6	**Bank 2 Rotate Control:** 1 = Enable; 0 = Disable
5	**Bank 1 Rotate Control:** 1 = Enable; 0 = Disable
4	**Bank 0 Rotate Control:** 1 = Enable; 0 = Disable
3:2	**Bank 2 Fixed Priority Mode Select—b,a:** ba 00 = Bank0 > Bank3 > Bank1 10 = Bank3 > Bank1 > Bank0 01 = Bank1 > Bank0 > Bank3 11 = Reserved
1	**Bank 1 Fixed Priority Mode Select:** 1 = REQ3# > CPUREQ#; 0 = CPUREQ# > REQ3
0	**Bank 0 Fixed Priority Mode Select:** 1 = REQ0# > PCEBREQ#; 0 = PCEBREQ# > REQ0#. Note that PCEBREQ# is a PCEB internal signal.

3.1.10 ARBPRIX—PCI ARBITER PRIORITY CONTROL EXTENSION REGISTER

Address Offset: 43h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register controls the fixed priority mode for bank 3 of the PCEB's internal arbiter. The ARBPRIX Register is used in conjunction with the PCI Arbiter Priority Control (ARBPRI) Register.

Bit	Description
7:1	**Reserved**
0	**Bank 3 Fixed Priority Mode Select:** 1 = REQ2# > REQ1#; 0 = REQ1# > REQ2#.

3.1.11 MCSCON—MEMCS# CONTROL REGISTER

Address Offset: 44h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

The MCSCON Register provides the master enable for generating MEMCS#. This register also provides read enable (RE) and write enable (WE) attributes for two main memory regions (the 512 KByte - 640 KByte region and an upper BIOS region). PCI accesses within the enabled regions result in the generation of MEMCS#. Note that the 0-512 KByte region does not have RE and WE attribute bits. The 0-512 KByte region can only be disabled with the MEMCS# Master Enable bit (bit 4). Note also, that when the RE and WE bits are both 0 for a particular region, the PCI master can not access the corresponding region in main memory (MEMCS# is not generated for either reads or writes).

Bit	Description
7:5	**Reserved**
4	**MEMCS# Master Enable:** When bit 4 = 1, the PCEB asserts MEMCS# for all accesses to the defined MEMCS# region (as defined by the MCSTOM Register and excluding the memory hole defined by the MCSBOH and MCSTOH Registers), if the accessed location is in a region enabled by bits [3:0] of this register or in the regions defined by the MAR1, MAR2, and MAR3 registers. When bit 4 = 0, the entire MEMCS# function is disabled and MEMCS# is never asserted.
3	**Write Enable For 0F0000–0FFFFFh (Upper 64 KByte BIOS):** When bit 3 = 1, the PCEB generates MEMCS# for PCI master memory write accesses to the address range 0F0000–0FFFFFh. When bit 3 = 0, the PCEB does not generate MEMCS# for PCI master memory write accesses to the address range 0F0000–0FFFFFh.
2	**Read Enable For 0F0000–0FFFFFh (Upper 64 KByte BIOS):** When bit 2 = 1, the PCEB generates MEMCS# for PCI master memory read accesses to the address range 0F0000–0FFFFFh. When bit 2 = 0, the PCEB does not generate MEMCS# for PCI master memory read accesses to the address range 0F0000–0FFFFFh.
1	**Write Enable For 080000–09FFFFh (512–640 KByte):** When bit 1 = 1, the PCEB generates MEMCS# for PCI master memory write accesses to the address range 080000–09FFFFh. When bit 1 = 0, the PCEB does not generate MEMCS# for PCI master memory write accesses to the address range 080000–09FFFFh.
0	**Read Enable For 080000–09FFFFh (512–640 KByte):** When bit 0 = 1, the PCEB generates MEMCS# for PCI master memory read accesses to the address range 080000–09FFFFh. When bit 0 = 0, the PCEB does not generate MEMCS# for PCI master memory read accesses to the address range 080000–09FFFFh.

3.1.12 MCSBOH—MEMCS# BOTTOM OF HOLE REGISTER

Address Offset: 45h
Default value: 10h
Attribute: Read/Write
Size: 8 bits

This register defines the bottom of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSTOH Register. The hole is defined by the following equation:

TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by the MCSTOH Register and BOH is the bottom of the MEMCS# hole defined by this register.

For example, to program the BOH at 1 MByte, the value of 10h should be written to this register. To program the BOH at 2 MByte + 64 KByte this register should be programmed to 21h. To program the BOH at 8 MByte this register should be programmed to 80h.

When the TOH < BOH the hole is disabled. If TOH = BOH, the hole size is 64 KBytes. It is the responsibility of the programmer to guarantee that the BOH is at or above 1 MB. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH disables the hole.

Bit	Description
7:0	**Bottom of MEMCS# Hole:** Bits[7:0] correspond to address lines AD[23:16], respectively.

3.1.13 MCSTOH—MEMCS# TOP OF HOLE REGISTER

Address Offset: 46h
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

This register defines the top of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSBOH Register. The hole is defined by the following equation:

TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by this register and BOH is the bottom of the MEMCS# hole defined by the MCSBOH Register.

For example, to program the TOH at 1 MByte + 64 KByte, this register should be programmed to 10h. To program the TOH at 2 MByte + 128 KByte this register should be programmed to 21h. To program the TOH at 12 MByte this register should be programmed to BFh.

When the TOH < BOH the hole is disabled. If TOH = BOH, the hole size is 64 KBytes. It is the responsibility of the programmer to guarantee that the TOH is above 1 MByte. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH disables the hole.

Bit	Description
7:0	**Top of MEMCS# Hole:** Bits[7:0] correspond to address lines AD[23:16], respectively.

3.1.14 MCSTOM—MEMCS# TOP OF MEMORY REGISTER

Address Offset: 47h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

This register determines MEMCS# top of memory boundary. The top of memory boundary ranges from 2 MBytes-1 to 512 MBytes-1, in 2 MByte increments. This register is typically set to the top of main memory. Accesses ≥ 1 MByte and ≤ top of memory boundary results in the assertion of the MEMCS# signal (unless the address resides in the hole programmed via the MCSBOH and MCSTOH Registers). A value of 00h sets top of memory at 2 MBytes-1 (including the 2 MByte-1 address). A value of FFh sets the top of memory at 512 MByte-1 (including the 512 MByte-1 address).

Bit	Description
7:0	**Top of MEMCS# Memory Boundary:** Bits[7:0] correspond to address lines AD[28:21], respectively.

3.1.15 EADC1—EISA ADDRESS DECODE CONTROL 1 REGISTER

Address Offset 48–49h
Default value: 0001h
Attribute: Read/Write
Size: 16 bits

This 16-bit register specifies EISA-to-PCI mapping of the 0-1 MByte memory address range. For each bit position, the memory block is enabled if the corresponding bit = 1 and is disabled if the bit = 0. EISA or DMA memory cycles to the enabled blocks result in the EISA cycle being forwarded to the PCI Bus. For disabled memory blocks, the EISA memory cycle is not forwarded to the PCI Bus.

Bit	Description
15	**880–896 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
14	**864–880 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
13	**848–864 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
12	**832–848 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
11	**816–832 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
10	**800–816 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
9	**784–800 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
8	**768–784 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
7:3	**Reserved**
2	**640–768 KBytes VGA Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
1	**512–640 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.
0	**0–512 KBytes Memory Enable:** EISA-to-PCI mapping for this memory space is enabled when this bit is 1 and disabled when this bit is 0.

3.1.16 IORT—ISA I/O RECOVERY TIMER REGISTER

Address Offset: 4Ch
Default Value: 56
Attribute: Read/Write
Size: 8 bits

The I/O recovery logic is used to guarantee a minimum amount of time between back-to-back 8-bit and 16-bit PCI-to-ISA I/O slave accesses. These minimum times are programmable.

The I/O recovery mechanism in the PCEB is used to add recovery delay between PCI-originated 8-bit and 16-bit I/O cycles to ISA devices. The delay is measured from the rising edge of the EISA command signal (CMD#) to the falling edge of the next EISA command. The delay is equal to the number of EISA Bus clocks (BCLKs) that correspond to the value contained in bits [1:0] for 16-bit I/O devices and in bits[5:3] for 8-bit I/O devices. Note that no additional delay is inserted for back-to-back I/O "sub-cycles" generated as a result of byte assembly or disassembly. This register defaults to 8- and 16-bit recovery enabled with two clocks of I/O recovery.

Bit	Description
7	**Reserved**
6	**Bit I/O Recovery Enable:** This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits 5-3 are enabled. When this bit is set to 0, recovery times are disabled.
5:3	**8-Bit I/O Recovery times:** This 3-bit field defines the recovery times for 8-bit I/O. Programmable delays between back-to-back 8-bit PCI cycles to ISA I/O slaves is shown in terms of EISA clock cycles (BCLK). The selected delay programmed into this field is enabled/disabled via bit 6 of this register. **Bits [5:3] BCLK** 001 1 010 2 011 3 100 4 101 5 110 6 111 7 000 8
2	**16-Bit I/O Recovery Enable:** This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits 0 and 1 are enabled. When this bit is set to 0, recovery times are disabled.
1:0	**16-Bit I/O Recovery Times:** This 2-bit field defines the Recovery time for 16-bit I/O. Programmable delays between back-to-back 16-bit PCI cycles to ISA I/O slaves is shown in terms of EISA clock cycles (BCLK). The selected delay programmed into this field is enabled/disabled via bit 2 of this register. **Bits [1:0]B CLK** 01 1 10 2 11 3 00 4

3.1.17 MAR1—MEMCS# ATTRIBUTE REGISTER #1

Address Offset: 54h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the PCEB generates MEMCS# for PCI master, DMA, or EISA master memory read accesses to the corresponding segment in main memory. When the RE bit is set to a 0, the PCEB does not generate MEMCS# for PCI master, DMA, or EISA master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0-disabled), the PCI master, DMA, or EISA master can not access the corresponding segment in main memory.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the PCEB generates MEMCS# for PCI master, DMA, or EISA master memory write accesses to the corresponding segment in main memory. When this bit is set to a 0, the PCEB does not generate MEMCS# for PCI master, DMA, or EISA master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0-disabled), the PCI master, DMA, or EISA master can not access the corresponding segment in main memory.

Bit	Description
7	**0CC000–0CFFFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
6	**0CC000–0CFFFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable
5	**0C8000–0CBFFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
4	**0C8000–0CBFFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable
3	**0C4000–0C7FFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
2	**0C4000–0C7FFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable
1	**0C0000–0C3FFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
0	**0C0000–0C3FFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable

3.1.18 MAR2—MEMCS# ATTRIBUTE REGISTER #2

Address Offset: 55h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the PCEB generates MEMCS# for PCI master, DMA, or EISA master memory read accesses to the corresponding segment in main memory. When this bit is set to a 0, the PCEB does not generate MEMCS# for PCI master, DMA, or EISA master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0-disabled), the PCI master, DMA, or EISA master can not access the corresponding segment in main memory.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the PCEB generates MEMCS# for PCI master, DMA, or EISA master memory write accesses to the corresponding segment in main memory. When this bit is set to a 0, the PCEB does not generate MEMCS# for PCI master, DMA, or EISA master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0-disabled), the PCI master, DMA, or EISA master can not access the corresponding segment in main memory.

Bit	Description
7	**0DC000–0DFFFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
6	**0DC000–0DFFFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable
5	**0D8000–0DBFFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
4	**0D8000–0DBFFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable
3	**0D4000–0D7FFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
2	**0D4000–0D7FFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable
1	**0D0000–0D3FFFh Add-on BIOS: WE:** 1 = Enable; 0 = Disable
0	**0D0000–0D3FFFh Add-on BIOS: RE:** 1 = Enable; 0 = Disable

3.1.19 MAR3—MEMCS# ATTRIBUTE REGISTER #3

Address Offset: 56h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the PCEB generates MEMCS# for PCI master, DMA, EISA master memory read accesses to the corresponding segment in main memory. When this bit is set to a 0, the PCEB does not generate MEMCS# for PCI master, DMA, or EISA master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0-disabled), the PCI master can not access the corresponding segment in main memory.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the PCEB generates MEMCS# for PCI master , DMA, EISA master memory write accesses to the corresponding segment in main memory. When this bit is set to a 0, the PCEB does not generate MEMCS# for PCI master, DMA, or EISA master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0-disabled), the PCI master can not access the corresponding segment in main memory.

Bit	Description
7	**0EC000–0EFFFFh BIOS Extension: WE:** 1 = Enable; 0 = Disable
6	**0EC000–0EFFFFh BIOS Extension: RE:** 1 = Enable; 0 = Disable
5	**0E8000–0EBFFFh BIOS Extension: WE:** 1 = Enable; 0 = Disable
4	**0E8000–0EBFFFh BIOS Extension: RE:** 1 = Enable; 0 = Disable
3	**0E4000–0E7FFFh BIOS Extension: WE:** 1 = Enable; 0 = Disable
2	**0E4000–0E7FFFh BIOS Extension: RE:** 1 = Enable; 0 = Disable
1	**0E0000–0E3FFFh BIOS Extension: WE:** 1 = Enable; 0 = Disable
0	**0E0000–0E3FFFh BIOS Extension: RE:** 1 = Enable; 0 = Disable

3.1.20 PDCON—PCI DECODE CONTROL REGISTER

Address Offset: 58h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register enables/disables positive decode of PCI accesses to the IDE and 8259 locations residing in the expansion bus subsystem. For the 82374SB, this register controls the mode of address decode (subtractive or negative) for memory cycles on the PCI Bus.

Subtractive decoding:

PCI memory cycles that are not claimed on the PCI Bus (i.e., DEVSEL# inactive) are forwarded to the EISA Bus. This is the default on power up.

Negative decoding (82374SB Only):

PCI memory cycles that are not mapped to one of the regions defined by A, B, or C below, are immediately forwarded to the EISA Bus (i.e. without waiting for DEVSEL# time-out). PCI memory cycles that are decoded to one of the four programmable PCI memory regions, but are not claimed (DEVSEL# negated), are forwarded to the EISA Bus by subtractive decode.

A. Main memory locations defined by the MEMCS# mapping (MCSCON, MCSBOH, MCSTOH, MCSTOM, MAR1, MAR2, and MAR3 Registers).

B. The enabled Video Frame Buffer region, 0A0000–0BFFFFh (as indicated by bit 2 of the EADC1 Register).

C. The four programmable PCI memory regions (defined by the MEMREGN[4:1] registers).

NOTE:
If there are devices on the PCI that are not mapped into any of the regions defined by A, B, or C, then negative decoding can not be used.

Bit	Description
7:6	**Reserved**
5	**8259 Decode Control (8259DC):** This bit enables/disables positive decode of 8259 locations 0020h, 0021h, 00A0h and 00A1h. When this bit is 1, positive decode for these locations are enabled. When this bit is 0, positive decode for these locations is disabled. After reset, this bit is 0. Note that if positive decode is disabled, these 8259 locations can still be accessed via subtractive decode.
4	**IDE Decode Control (IDEDC):** This bit enables/disables positive decode of IDE locations 1F0–1F7h (primary) or 170–177h (secondary) and 3F6h,3F7h (primary) or 376h,377h (secondary). When IDEDC = 0, positive decode is disabled. When IDECD = 1, positive decode is enabled. After reset, this bit is 0. Note that if positive decode is disabled, these IDE locations can still be accessed via subtractive decode.
3:1	**Reserved**
0	**82375EB: Reserved.** Must be 0 when programming this register. **82375SB: PCI Memory Address Decoding Mode (PMAD):** This bit selects between subtractive and negative decoding. When PMAD = 1, negative decoding is selected. When PMAD = 0, subtractive decoding is selected. After reset, this bit is 0.

3.1.21 EADC2—EISA ADDRESS DECODE CONTROL EXTENSION REGISTER

Address Offset: 5Ah
Default value: 00h
Attribute: Read/Write
Size: 8 bits

This register specifies EISA-to-PCI mapping for the 896 KByte to 1 MByte memory address range (BIOS). If this memory block is enabled, EISA memory accesses in this range will result in the EISA cycles being forwarded to the PCI Bus. (Note that enabling this block is necessary if BIOS resides within the PCI and not within the EISA subsystem.)

This register also defines mapping for the 16 MByte - 64 KByte to 16 MByte memory address range. This mapping is important if the BIOS is aliased at the top 64 KBytes of 16 MBytes. If the region is enabled and this address range is within the hole defined by the MCSBOH and MCSTOH Registers or above the top of main memory defined by the MCSTOM Register, the EISA cycle is forwarded to the PCI.

Bit	Description
7:6	**Reserved**
5	**Top 64 KByte of 16 MByte Memory Space Enable (FF0000–FFFFFFh):** This memory block is enabled when this bit is 1 and disabled when this bit is 0.
4	**960 KBytes—1 MByte Memory Space Enable (0F0000–0FFFFFh):** This memory block is enabled when this bit is 1 and disabled when this bit is 0.
3	**944–960 KByte Memory Space Enable (0EC000–0EFFFFh):** This memory block is enabled when this bit is 1 and disabled when this bit is 0.
2	**928–944 KByte Memory Space Enable (0E8000–0EBFFFh):** This memory block is enabled when this bit is 1 and disabled when this bit is 0.
1	**912–928 KByte Memory Space Enable (0E4000–0E7FFFh):** This memory block is enabled when this bit is 1 and disabled when this bit is 0.
0	**896–912 KByte Memory Space Enable (0E0000–0E3FFFh):** This memory block is enabled when this bit is 1 and disabled when this bit is 0.

3.1.22 EPMRA—EISA-TO-PCI MEMORY REGION ATTRIBUTES REGISTER

Address Offset: 5Ch
Default value: 00h
Attribute: Read/Write
Size: 8 bits

This register defines buffering attributes for EISA accesses to PCI memory regions specified by MEMREGN[4:1] Registers. When an EPMRA bit is 1 (and the Line Buffers are enabled via the PCICON Register), EISA accesses to the corresponding PCI memory region are performed in buffered mode. In buffered mode, read prefetching and write posting/assembly are enabled. When an EPMRA bit is 0, EISA accesses to the corresponding PCI memory region are performed in non-buffered mode. In non-buffered mode, a buffer bypass path is used to complete the transaction.

NOTE:

1. Using buffered mode for EISA accesses to PCI memory regions that contain memory-mapped I/O devices can cause unintended side effects. In buffered mode, strong ordering is not preserved within a Dword. If the order of the writes to an I/O device is important, non-buffered mode should be used. Also, read-prefetch can cause unintended changes of status registers in the memory-mapped I/O device.
2. The Line Buffers are typically enabled or disabled during system initialization. These buffers should not be dynamically enabled/disabled during runtime. Otherwise, data coherency can be affected, if a buffer containing valid write data is disabled and then, later, re-enabled.

Bit	Description
7:4	**Reserved**
3	**Region 4 Attribute (REG-4):** EISA accesses to this PCI memory region are buffered when this bit is 1 and non-buffered when this bit is 0. If the Line Buffers are disabled via the PCICON Register (bit 6), buffering is disabled, regardless of the value of this bit.
2	**Region 3 Attribute (REG-3):** EISA accesses to this PCI memory region are buffered when this bit is 1 and non-buffered when this bit is 0. If the Line Buffers are disabled via the PCICON Register (bit 6), buffering is disabled, regardless of the value of this bit.
1	**Region 2 Attribute (REG-2):** EISA accesses to this PCI memory region are buffered when this bit is 1 and non-buffered when this bit is 0. If the Line Buffers are disabled via the PCICON Register (bit 6), buffering is disabled, regardless of the value of this bit.
0	**Region 1 Attribute (REG-1):** EISA accesses to this PCI memory region are buffered when this bit is 1 and non-buffered when this bit is 0. If the Line Buffers are disabled via the PCICON Register (bit 6), buffering is disabled, regardless of the value of this bit.

3.1.23 MEMREGN[4:1]—EISA-TO-PCI MEMORY REGION ADDRESS REGISTERS

Address Offset: 60-63h (Memory Region 1)
64-67h (Memory Region 2)
68-6Bh (Memory Region 3)
6C-6Fh (Memory Region 4)
Default Value: 0000FFFFh
Attribute: Read/Write
Size: 32 bits

These 32-bit registers provide four windows for EISA-to-PCI memory accesses. Each window defines a positively decoded programmable address region for mapping EISA memory space to the corresponding PCI memory space. This base and limit address fields define the size and location of the region within the 4 GByte PCI memory space. The base and limit addresses can be aligned on any 64 KByte boundary and each region can be sized in 64 KByte increments, up to the theoretical maximum size of 4 GByte. The default values of this register ensure that the regions are initially disabled.

A region is selected based on the following formula: Base Address ≤ address ≤ Limit Address.

Bit	Description
31:16	**Memory Region Limit Address:** For EISA-to-PCI accesses, bits[31:16] correspond to address lines LA[31:16] on the EISA Bus and AD[31:16] on the PCI Bus. This field determines the limit address of the memory region within the 4 GByte PCI memory space.
15:0	**Memory Region Base Address:** For EISA-to-PCI accesses, bits[15:0] correspond to address lines LA[31:16] on the EISA Bus and AD[31:16] on the PCI Bus. This field determines the starting address of the memory region within the 4 GByte PCI memory space.

3.1.24 IOREGN[4:1]—EISA-TO-PCI I/O REGION ADDRESS REGISTERS

Address Offset: 70-73h (I/O Region 1)
74-77h (I/O Region 2)
78-7Bh (I/O Region 3)
7C-7Fh (I/O Region 4)
Default value: 0000FFFCh
Attribute: Read/Write
Size: 32 bits

These 32-bit registers provide four windows for EISA-to-PCI I/O accesses. The windows define positively decoded programmable address regions for mapping EISA I/O space to the corresponding PCI I/O space. Each register determines the starting and limit addresses of the particular region within the 64 KByte PCI I/O space. The base and limit addresses can be aligned on any Dword boundary and each region can be sized in Dword increments (32-bits) up to the theoretical maximum size of 64 KByte. Default values for the base and limit fields ensure that the regions are initially disabled.

The I/O regions are selected based on the following formula: Base Address ≤ address ≤ Limit Address.

Bit	Description
31:18	**I/O Region Limit Address:** For EISA-to-PCI I/O accesses, bits[31:18] correspond to address lines LA[15:2] on the EISA Bus and AD[15:2] on the PCI Bus. This field determines the limit address of the region within the 64 KByte PCI I/O space.
17:16	**Reserved**
15:2	**I/O Region Base Address:** For EISA-to-PCI I/O accesses, bits[15:2] correspond to address lines LA[15:2] on the EISA Bus and AD[15:2] on the PCI Bus. This field determines the starting address of the region within the 65 KByte PCI I/O space.
1:0	**Reserved**

3.1.25 BTMR—BIOS TIMER BASE ADDRESS REGISTER

Address Offset: 80–81h
Default value: 0078h
Attribute: Read/Write
Size: 16 bits

This 16-bit register determines the base address for the BIOS Timer Register located in PCI I/O space. The BIOS Timer resides in the PCEB and is the only internal resource mapped to PCI I/O space. The base address can be set at Dword boundaries anywhere in the 64 KByte PCI I/O space. This register also provides the BIOS Timer access enable/disable control bit.

Bit	Description
15:2	**BIOS Timer Base Address:** Bits[15:2] correspond to PCI address lines AD[15:2].
1	**Reserved**
0	**BIOS Timer Enable (BTE):** When BTE = 1, the BIOS Timer is enabled. When BTE = 0, the BIOS Timer is disabled. The default is 0 (disabled).

3.1.26 ELTCR—EISA LATENCY TIMER CONTROL REGISTER

Address Offset: 84h
Default value: 7Fh
Attribute: Read/Write
Size: 8 bits

This register provides the control for the EISA Latency Timer (ELT). The register holds the initial count value used by the ELT. The ELT uses the PCI clock for counting . The ELT time-out period is equal to:

$$ELT_{timeout} = Value\{ELTCR(7:0)\} \times T_{pciclck}\ [ns]$$

where:

Tpciclk = 30 ns at 33 MHz (40 ns at 25 MHz).

Therefore, a maximum ELT time-out period at 33 MHz is 256 x 30 ns = 7.68 μs. The value written into this register is system dependent. It should be based on PCI latency characteristics controlled by the PCI Master Latency Timer mechanism and on EISA Bus arbitration/latency parametrics. A typical value corresponds to the ELT time-out period of 1–3 μs. When the value in the ELTCR Register is 0, the ELT mechanism is disabled. The ELTCR Register must be initialized before EISA masters or DMA are enabled.

Bit	Description
7:0	**EISA Latency Timer Count Value:** Bits[7:0] contain the initial count value for the EISA Latency Timer. When this field contains 00h, the EISA Latency Timer is disabled.

3.2 I/O Registers

The only PCEB internal resource mapped to the PCI I/O space is the BIOS Timer Register.

3.2.1 BIOSTM—BIOS TIMER REGISTER

Register Location: Programmable I/O address location (Dword aligned)
Default Value: 00 00 xx xxh
Attribute: Read/Write
Size: 32 bits

This 32-bit register is mapped to the PCI I/O space location determined by the value in the BTMR Register. Bit 0 of BTMR must be 1 to enable access to the BIOS Timer. The BIOS timer clock is derived from the EISA Bus clock (BCLK); either 8.25 or 8.33 MHz depending on the PCI clock. BCLK is divided by 8 to obtain the timer clock of 1.03 or 1.04 MHz. If a frequency other than 33 MHz or 25 MHz is used for PCI clock, the BIOS Timer clock will be affected. (It will always keep the same relation to the BCLK, i.e. 1:4 or 1:3, depending on the clock divisor.) The BIOS Timer is only accessible from the PCI Bus and is not accessible from the EISA Bus.

After data is written into BIOS Timer Register (BE1# and/or BE0# must be asserted), the BIOS timer starts decrementing until it reaches zero. It "freezes" at zero until the new count value is written.

Bit	Description
31:16	**Reserved**
15:0	**BIOS Timer Count Value:** The initial count value is written to bits[15:0] to start the timer. The value read is the current value of the BIOS Timer.

4.0 ADDRESS DECODING

Conceptually, the PCEB contains two programmable address decoders: one to decode PCI Bus cycles that need to be forwarded to the EISA Bus or serviced internally and the other to decode EISA Bus cycles that need to be forwarded to the PCI Bus. Two decoders permit the PCI and EISA Buses to operate concurrently (Figure 2). The PCEB can be programmed to respond to certain PCI memory or I/O region accesses as well as configuration space accesses to the PCEB's internal configuration registers. PCEB address decoding is discussed in Section 4.1.

The EISA address decoder decodes EISA Bus cycles generated by the bus master (DMA controller, ISA compatible master, or EISA compatible master) that need to be forwarded to the PCI Bus. The EISA decode logic can be programmed to respond to certain memory or I/O region accesses.

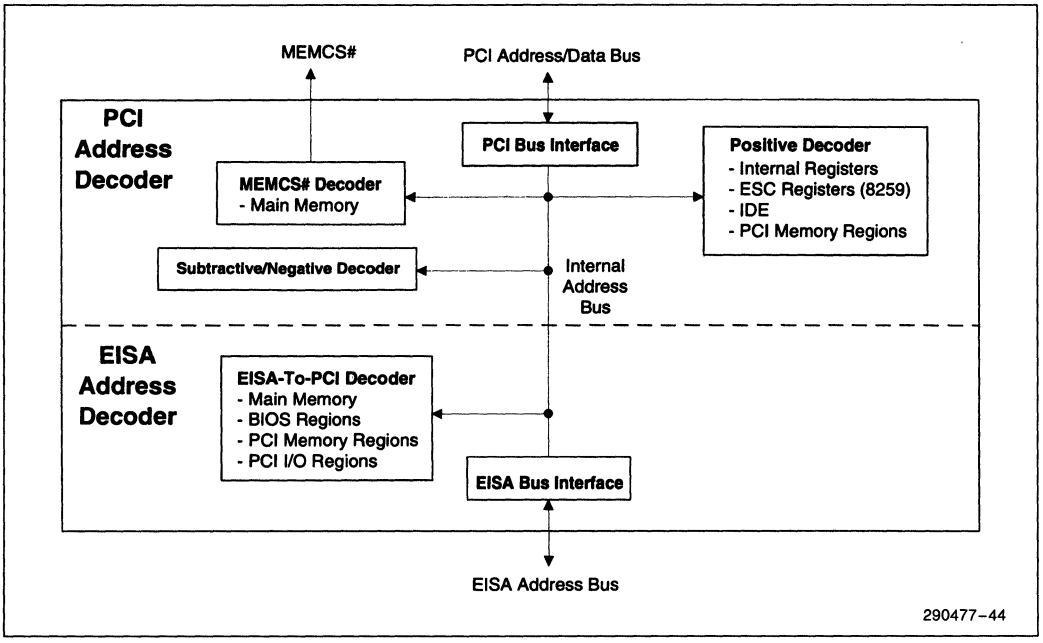


Figure 2. Block Diagram of Address Decoder

The PCEB provides three methods for decoding the current PCI Bus cycle. The PCEB can use positive, subtractive, or negative (82374SB only) decoding for these cycles, depending on the type of cycle, actions on the PCI Bus, and programming of the PCEB registers. For EISA Bus cycles, only positive decoding is used.

1. **Positive decoding**. With positive decoding, the PCI/EISA Bus cycle address is compared to the corresponding address ranges set up in the PCEB for positive decode. A match causes the PCEB decode logic to immediately service the cycle. The PCEB can be programmed (via the configuration registers) to positively decode selected memory or I/O accesses on both the PCI Bus and EISA Bus. Depending on the programming of the internal registers, the PCEB provides positive decoding for PCI accesses to selected address ranges in memory and I/O spaces and for EISA accesses to selected address ranges in memory and I/O spaces. Note that the decoding method for PCI accesses to the PCEB internal registers (configuration and I/O space registers) is not programmable and these accesses are always positively decoded.

2. **Subtractive decoding.** For PCI memory or I/O cycles, the PCEB uses subtractive decoding (or negative decoding, described in #3 of this list for the 82375SB) to respond to addresses that are not positively decoded. With subtractive decoding, if a memory or I/O cycle is not claimed on the PCI Bus (via DEVSEL#), the PCEB forwards the cycle to the EISA Bus. The PCEB waits a programmable number of PCICLKs (1 to 3 PCICLKs, as selected via the PCICON Register) for a PCI agent to claim the cycle. If the cycle is not claimed within the programmed number of PCICLKs (DEVSEL# time-out), the PCEB claims the cycle (asserts DEVSEL#) and forwards it to the EISA Bus. Note that the number of PCICLKs for a DEVSEL# time-out should be programmed to accommodate the slowest PCI Bus device.
3. **Negative decoding.** For the 82375SB, negative decoding is a programmable option (via the PDCON Register) that is only used for PCI memory cycles. With negative decoding, a PCI memory cycle that is not positively decoded by the PCEB as a main memory area (one of the MEMCS# generation areas) and is not in one of the four programmable EISA-to-PCI memory regions (defined by MEMREGN[4:1]) is immediately forwarded to the EISA Bus. This occurs without waiting for a DEVSEL# time-out to see if the cycle is going to be claimed on the PCI Bus. Thus, negative decoding can reduce the latency incurred by waiting for a DEVSEL# time-out that is associated with subtractive decoding. This increases throughput to the EISA Bus for unclaimed PCI memory cycles. If the DEVSEL# time-out is set to a 2 PCICLKtime-out, the latency is reduced by 1 PCICLK and for a 3 PCICLK time-out, the latency is reduced by 2 PCICLKs. For more information on negative (and subtractive) decoding, see Section 4.1.1.3, Subtractively and Negatively Decoded Cycles to EISA.

   Note that negative decoding imposes a restriction on the PCI system memory address map. PCI memory-mapped devices are restricted to one of the four programmable EISA-to-PCI regions (MEMREGN[4:1]). These regions always use subtractive decoding to forward an unclaimed cycle to the EISA Bus, even if negative decoding is enabled. Locating devices in these regions ensures that the PCI device has the allotted number of programmed PCICLKs (DEVSEL# time-out) to respond with DEVSEL#. Further, since the PCEB does not negatively decode I/O space addresses, enabling this feature does not impose restrictions on devices that are mapped to PCI I/O space.

4.1 PCI Cycle Address Decoding

The PCEB decodes addresses presented on the multiplexed PCI address/data bus during the address bus phase. AD[31:0] and the byte enables (C/BE[3:0]# during the data phase) are used for address decoding. C/BE[3:0]# are used during the data phase to indicate which byte lanes contain valid data. For memory cycles, the PCI address decoding is always a function of AD[31:2]. In the case of I/O cycles, all 32 address bits (AD[31:0]) are used to provide addressing with byte granularity. For configuration cycles, only a subset of the address lines carry address information.

The PCEB decodes the following PCI cycle addresses based on the contents of the relevant programmable registers:

1. Positively decodes PCEB configuration registers.
2. Positively decodes I/O addresses contained within the PCEB (BIOS Timer).
3. Positively decodes the following compatibility I/O registers to improve performance: a) Interrupt controller (8259) I/O registers contained within the ESC to optimize interrupt processing, if enabled through the PDCON Register, b) IDE registers, if enabled through the PDCON Register.
4. Positively decodes four programmable memory address regions contained within the PCI memory space.

5. Positively decodes memory addresses for selected regions of main memory (located behind the Host/PCI Bridge). When a main memory address is positively decoded, the PCEB asserts the MEMCS# signal to the Host/PCI Bridge. The PCEB does not assert DEVSEL#.
6. Subtractively or negatively (82375SB only) decodes cycles to the EISA Bus (see Section 4.1.1, Memory Space Address Decoding) .

NOTE:

A PCI requirement is that, upon power-up, PCI agents do not respond to any address. Typically, the only access to a PCI agent is through the IDSEL configuration mechanism until the agent is enabled during initialization. The PCEB/ESC subsystem is an exception to this since it controls access to the BIOS boot code. The PCEB subtractively decodes BIOS accesses and passes the accesses to the EISA Bus where the ESC generates BIOS chip select. This allows BIOS memory to be located in the PCI memory space.

4.1.1 MEMORY SPACE ADDRESS DECODING

The MCSCON, MCSTOP, MCSBOH, MCSTOM, and PDCON Registers are used to program the decoding for PCI Bus memory cycles.

4.1.1.1 Main Memory Decoding (MEMCS#)

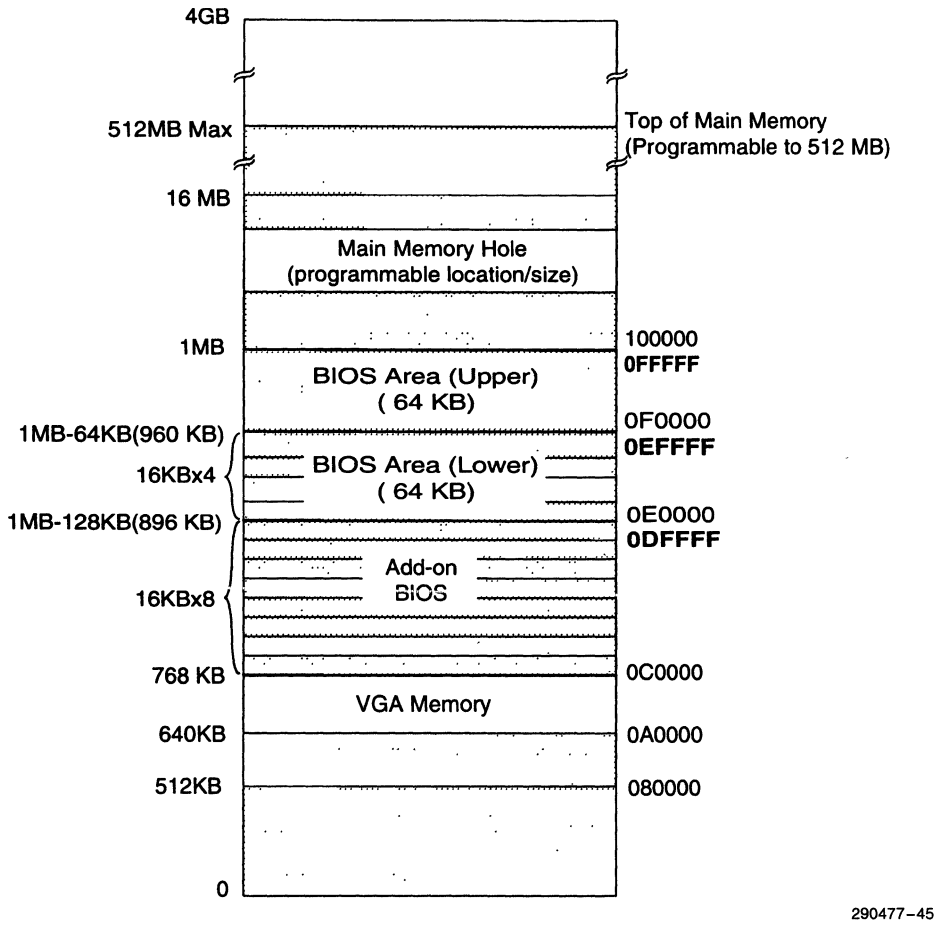
The PCEB supports positive decode of main memory areas by generating a memory chip select signal (MEMCS#) to the Host/PCI Bridge that contains the main memory interface control. The PCEB supports memory sizes up to 512 MBytes (i.e., the PCEB can be programmed to generate MEMCS# for this memory range). For PCI memory accesses above 512 MByte (512 MBytes to 4 GBytes), the PCEB does not generate MEMCS# and unclaimed cycles are forwarded to the EISA Bus using either subtractive or negative (82374SB only) decoding.

If a memory region is enabled, accesses to that region are positively decoded and result in the PCEB asserting MEMCS#. If a memory region is disabled, accesses do not generate MEMCS# and the cycle is either subtractively or negatively (82374SB only) decoded and forwarded to the EISA Bus.

Within the 512 MByte main memory range, the PCEB supports the enabling/disabling of sixteen individual memory ranges (Figure 3). Fourteen of the ranges are within the 640 KByte - 1 MByte area and have Read Enable (RE) and Write Enable (WE) attributes. These attributes permit positive address decoding for reads and writes to be independently enabled/disabled. This permits, for example, an address range to be positively decoded for a memory read and subtractively or negatively (82374SB only) decoded to the EISA Bus for a memory write.

The fifteenth range (0–512 KByte) and sixteenth range (programmable limit address from 2 MByte up to 512 MByte on 2 MByte increments) can be enabled or disabled but do not have RE/WE attributes. A seventeenth range is available that identifies a memory hole. Addresses within this hole will not generate a MEMCS#. These memory address ranges are:

- 0–512 KByte
- 512–640 KByte
- 640–768 KBytes (VGA memory page)
- 960 KByte to 1 MByte (BIOS Area)
- 768–896 KByte in 16 KByte segments (total of 8 segments)
- 896–960 KByte in 16 KByte segments (total of 4 segments)
- 960 KByte to 1 MByte (Upper BIOS area)
- 1–512 MByte in 2 MByte increments.
- Programmable memory hole in 64 KByte increments between 1 MByte and 16 MByte.



NOTE:
Shaded areas represent main memory decode regions for MEMCS#.

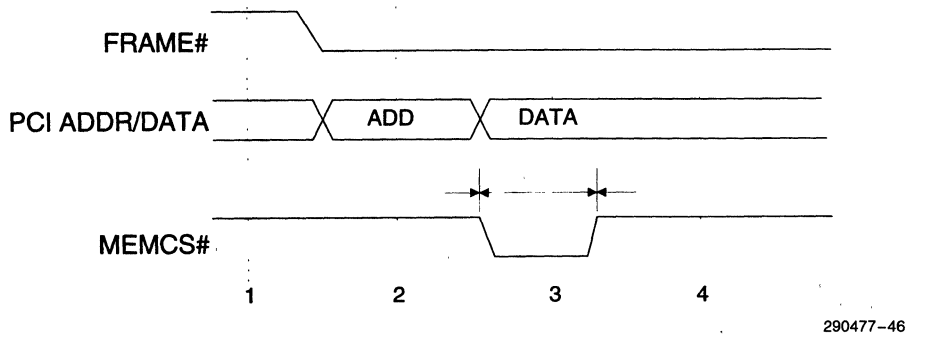
Figure 3. MEMCS# Decode Areas

Table 2 summarizes the attribute registers used in MEMCS# decoding. The MCSCON, MAR1, MAR2, and MAR3 Registers are used to assign RE/WE attributes to a particular memory range. The MEMCS# hole is programmed using the MCSTOH and MCSBOH Registers. The region above 1 MByte is programmed using the MCSTOM Register. The region from 0–512 KByte is enabled/disabled using bit 4 of the MCSCON Register. MCSCON bit 4 is also used to enable and disable the entire MEMCS# function.

Table 2. Read Enable/Write Enable Attributes For MEMCS# Decoding

Memory Attribute Registers (Register Bits are Shown in Brackets)	Attribute	Memory Segments	Comments
MCSCON[1:0]	WE RE	080000-09FFFFh	512K to 640K
MCSCON[3:2]	WE RE	0F0000-0FFFFFh	BIOS Area
MAR1[1:0]	WE RE	0C0000-0C3FFFh	Add-on BIOS
MAR1[3:2]	WE RE	0C4000-0C7FFFh	Add-on BIOS
MAR1[5:4]	WE RE	0C8000-0CBFFFh	Add-on BIOS
MAR1[7:6]	WE RE	0CC000-0CFFFFh	Add-on BIOS
MAR2[1:0]	WE RE	0D0000-0D3FFFh	Add-on BIOS
MAR2[3:2]	WE RE	0D4000-0D7FFFh	Add-on BIOS
MAR2[5:4]	WE RE	0D8000-0DBFFFh	Add-on BIOS
MAR2[7:6]	WE RE	0DC000-0DFFFFh	Add-on BIOS
MAR3[1:0]	WE RE	0E0000-0E3FFFh	BIOS Extension
MAR3[3:2]	WE RE	0E4000-0E7FFFh	BIOS Extension
MAR3[5:4]	WE RE	0E8000-0EBFFFh	BIOS Extension
MAR3[7:6]	WE RE	0EC000-0EFFFFh	BIOS Extension

The PCEB generates MEMCS# from the decode of the PCI address. MEMCS# is asserted during the first data phase as indicated in the Figure 4. MEMCS# is only asserted for one PCI clock period. The PCEB does not take any other action as a result of this decode, except to generate MEMCS#. It is the responsibility of the device using the MEMCS# signal to generate DEVSEL#, TRDY# and any other cycle response. The device using the MEMCS# will always generate DEVSEL# on the next clock. This fact can be used to avoid an extra clock delay in the subtractive decoder described in the next section.



NOTE:
Since MEMCS# is point-to-point (a sideband signal), the signal meets the guaranteed setup time to clock edge 3 (and clock edge 4). This fast generation of MEMCS# prevents the penalty caused by the decoding delay.

Figure 4. MEMCS# Generation

4.1.1.2 BIOS Memory Space

The BIOS memory space is subtractively decoded. BIOS is typically "shadowed" after configuration and initialization is complete. Thus, negative decoding is not implemented for accesses to the BIOS EPROM residing on the expansion bus.

The ESC decoder supports BIOS space up to 512 KBytes. The standard 128 KByte BIOS memory space is 000E 0000h to 000F FFFFh (top of 1 MByte), and aliased at FFFE 0000h to FFFF FFFFh (top of 4 GByte) and FFEE 0000h to FFEF FFFFh (top of 4 GByte -1 MByte). These aliased regions account for the CPU reset vector and the uncertainty of the state of the A20Gate when a software reset occurs.

Note that the ESC component contains the BIOS space decoder that provides address aliasing for BIOS at 4 GByte or 4 GByte - 1 MByte by ignoring the LA20 address line.

The additional 384 KByte BIOS memory space at FFF8 0000h to FFFD FFFFh is known as the enlarged BIOS memory space. Note that EISA memory (other than BIOS) must not reside within the address range from 4 GByte -1.5 MByte to 4 GByte - 1 MByte and from 4 GByte - 512 KByte to 4 GByte to avoid conflict with BIOS space.

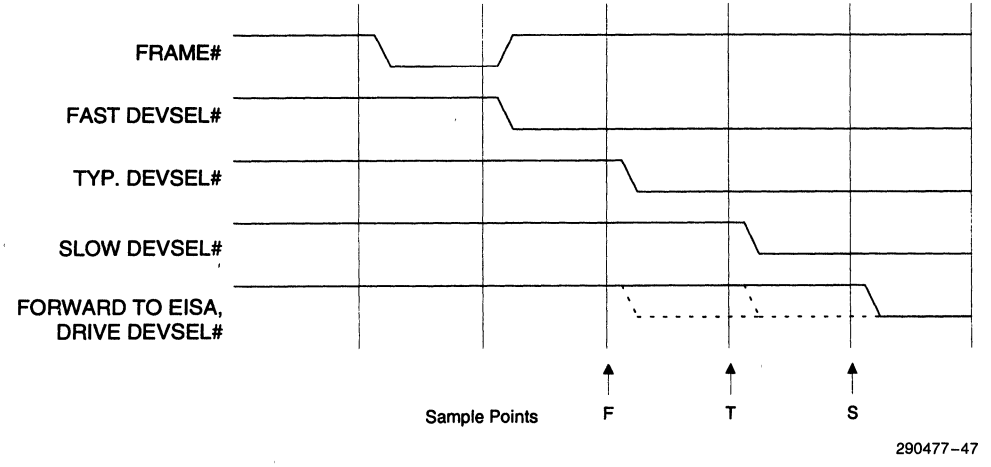
Since the BIOS device is 8 or 16 bits wide and typically has very long access times, PCI burst reads from BIOS space invoke a disconnect target termination (using the STOP# signal) after the first data transaction in order to meet the PCI incremental latency guidelines.

4.1.1.3 Subtractively And Negatively Decoded Cycles To EISA

The PCEB uses subtractive and negative (82375SB only) decoding to forward PCI Bus cycles to the EISA Bus. These modes are defined at the beginning of section 4.0. Bit 0 of the PDCON Register selects between negative and subtractive decoding.

For subtractive decoding on the 82375EB, the DEVSEL# sample point (Figure 5) can be configured to two different settings by programming the PCICON Register. If the "typical" point is selected, DEVSEL# is sampled at T, and, if inactive, the cycle is forwarded to EISA. If the "slow" point is selected, DEVSEL# is sampled at F, T, and S. The sample point should be configured to match the slowest PCI device in the system. This programmable capability (T or S) permits systems to optimize the DEVSEL# time-out latency to the response capabilities of the PCI devices in the system. The sample point selected must accommodate the slowest device on the PCI Bus. Note that when these unclaimed cycles are forwarded to the EISA Bus, the PCEB drives the DEVSEL# active. An active MEMCS# always results in an active DEVSEL# on the "Typical" sample point.

For subtractive decoding on the 82375SB, the DEVSEL# sample point can be configured to two different settings by programming the PCICON Register (slow and typical).



NOTE:
Fast DEVSEL# is not supported on the 82378EB or 82375SB.

Figure 5. DEVSEL# Sample Points

Only unclaimed PCI cycles within the memory address range from 0 to 4 GByte and I/O address range from 0 to 64 KByte are forwarded to EISA. Unclaimed PCI I/O cycles to address locations above 64 KBytes are not forwarded to the EISA Bus and the PCEB does not respond with DEVSEL#. In this case, these unclaimed cycles cause the master to terminate the PCI cycle with a master abort.

For the 82374SB, if negative decoding is used, the PCEB begins the PCI-to-EISA cycle forwarding process at the "fast" sample point. Compared to the system that uses subtractive decode at the "slow" sample point, negative decoding reduces the decoding overhead by 2 PCI clock cycles. In the case of subtractive decode at the "typical" sampling point, negative decoding reduces the overhead by 1 PCI clock.

The PCEB contains programmable configuration registers that define address ranges for PCI resident devices. There is a set of registers associated with MEMCS# decoding of main memory areas and set of registers for defining address mapping of up to four memory regions that are mapped to PCI for EISA Bus initiated cycles. Note that on the 82375EB, there is no equivalent mechanism for mapping the PCI memory regions to EISA and, therefore, all PCI memory cycles that need to be forwarded to the EISA Bus use subtractive decoding.

For the 82374SB, when negative decoding is selected, memory cycles with addresses other than those specified by the MEMCS# mapping for positive decode (via the MCSCON, MCSBOH, MCSTOH, MCSTOM, MAR1, MAR2, and MAR3 Registers) or the four programmable EISA-to-PCI memory regions (via MEMREGN[4:1]) are immediately forwarded to the EISA Bus without waiting for a DEVSEL# time-out.

Negative decoding has the following properties.

- All addresses above the top of main memory or within the MEMCS# hole (as defined by the MEMCS# map) are negatively decoded to EISA, except for the four programmable EISA-to-PCI memory regions. These regions (MEMREGN[4:1]) can overlap with active main memory ranges, the main memory hole, or with the memory space above the top of main memory. PCI accesses to MEMREGN[4:1] are always subtractively decoded to EISA.
- All addresses within MEMCS# defined ranges 640 KByte to 1 MByte can be either mapped to PCI or EISA using positive decoding. Some of these regions allow more detailed mapping based on programmable access attributes (read enable and write enable). This permits a region to be positively decoded for the enabled attribute and negatively decoded, if enabled, to the EISA Bus for the disabled attribute. For example, if a region is enabled for reads and disabled for writes, accesses to the region are positively decoded to the PCI for reads and negatively decoded, if enabled, to EISA for writes. If negative decoding is disabled (i.e., subtractive decoding enabled), the write is subtractively decoded to EISA.
- When negative decoding is enabled, MEMREGN[4:1] can still be set up for subtractive decoding. A PCI device that requires subtractive decoding must reside within Region [4:1]. As a result, the subtractive decoding penalty is only associated with some address ranges (i.e. some devices) and not with all non-PCI ranges. This feature can be used with PCI devices that dynamically change response on PCI cycles based on cycle type or an internal device state (e.g. intervention cycle).

  If a PCI device can not be located in one of the regions (Region [4:1]), then negative decoding can not be used. This could occur for systems with very specific address mapping requirements or systems where the device addresses that reside on the PCI Bus are highly fragmented and could not be accommodated with four regions.

  Note that the four regions do not limit mapping to only four devices. More than one device can be mapped into the same programmable region. These devices will reside within their own sub-regions, which are not necessarily contiguous.

4.1.2 PCEB CONFIGURATION REGISTERS

PCI accesses to the PCEB configuration registers are positively decoded. For a detailed address map of the PCEB configuration registers, see Section 3.1, Configuration Registers.

4.1.3 PCEB I/O REGISTERS

The only I/O-mapped register in the PCEB is the BIOS Timer Register. Section 3.2 provides details on the address mapping of this register. Note that the internal decode of the BIOS Timer Register is disabled after reset and all I/O accesses that are not contained within the PCI are subtractively decoded and passed to EISA Bus. To enable I/O access to the PCEB's BIOS Timer Register, The BTMR Register must be programmed.

4.1.4 POSITIVELY DECODED COMPATIBILITY I/O REGISTERS

The 8259 interrupt controller and IDE register locations are positively decoded. Access to the corresponding I/O address ranges must first be enabled through the PDCON Register.

PCI accesses to these registers are broadcast to the EISA Bus. These PCI accesses require the ownership of the EISA Bus, and will be retried if the EISA Bus is owned by an EISA/ISA master or the DMA.

4.1.4.1 ESC Resident PIC Registers

Access to the 8259 registers are positively decoded, if enabled through PDCON Register, to minimize access time to the system interrupt controller during interrupt processing (in particular during the EOI command sequence). Table 3 shows the 8259 I/O address map. After PCIRST#, positively decoded access to these address ranges is disabled.

Table 3. ESC Resident Programmable Interrupt Controller (PIC) Registers

Address (hex)	Address Bits FEDC	Address Bits BA98	Address Bits 7654	Address Bits 3210	Access Type	Register Name
0020h	0000	0000	001x	xx00	R/W	INT 1 Control
0021h	0000	0000	001x	xx01	R/W	INT 1 Mask
00A0h	0000	0000	101x	xx00	R/W	INT 2 Control
00A1h	0000	0000	101x	xx01	R/W	INT 2 Mask

4.1.4.2 EISA Resident IDE Registers

The PCI address decoder positively decodes IDE I/O addresses (Primary and Secondary IDE) that exist within the EISA subsystem (typically on the X-bus or as an ISA slave). This feature is implemented to minimize the decoding penalty for the systems that use IDE as a mass-storage controller. Table 4 shows IDE's I/O address map. Note that the PDCON Register controls the enable/disable function for IDE decoding. After PCIRST#, positive decode of the IDE address range is disabled.

Table 4. EISA Resident IDE Registers

Address (hex)	Addess (Bits)				Access Type	Register Name
	FEDC	BA98	7654	3210		
0170h	0000	0001	0111	0000	R/W	Secondary Data Register
0171h	0000	0001	0111	0001	R/W	Secondary Error Register
0172h	0000	0001	0111	0010	R/W	Secondary Sector Count Register
0173h	0000	0001	0111	0011	R/W	Secondary Sector Number Register
0174h	0000	0001	0111	0100	R/W	Secondary Cylinder Low Register
0175h	0000	0001	0111	0101	R/W	Secondary Cylinder High Register
0176h	0000	0001	0111	0110	R/W	Secondary Drive/head Register
0177h	0000	0001	0111	0111	R/W	Secondary Status Register
01F0h	0000	0001	1111	0000	R/W	Primary IDE Data Register
01F1h	0000	0001	1111	0001	R/W	Primary Error Register
01F2h	0000	0001	1111	0010	R/W	Primary Sector Count Register
01F3h	0000	0001	1111	0011	R/W	Primary Sector Number Register

Table 4. EISA Resident IDE Registers (Continued)

Address (hex)	Addess (Bits)				Access Type	Register Name
	FEDC	BA98	7654	3210		
01F4h	0000	0001	1111	0100	R/W	Primary Cylinder Low Register
01F5h	0000	0001	1111	0101	R/W	Primary Cylinder High Register
01F6h	0000	0001	1111	0110	R/W	Primary Drive/head Register
01F7h	0000	0001	1111	0111	R/W	Primary Status Register
0376h	0000	0011	0111	0110	R/W	Secondary Alternate Status Register
0377h	0000	0011	0111	0111	R	Secondary Drive Address Register
03F6h	0000	0011	1111	0110	R/W	Primary Alternate Status Register
03F7h	0000	0011	1111	0111	R	Primary Drive Address Register

4.2 EISA Cycle Address Decoding

For EISA Bus cycles, the PCEB address decoder determines the destination of EISA/ISA master and DMA cycles. This decoder provides the following functions:

- Positively decodes memory and I/O addresses that have been programmed into the PCEB for forwarding to the PCI Bus. This includes accesses to devices that reside directly on the PCI (memory Regions [4:1] and I/O Regions [4:1]) and segments of main memory that resides behind the Host/PCI Bridge.
- Provides access attributes for memory Regions [4:1]. These attributes are used to select the most optimum access mode (buffered or non-buffered).
- All cycles that are not positively decoded to be forwarded to PCI are contained within EISA.

NOTE:
The registers that reside in the PCEB (configuration registers and BIOS Timer) are not accessible from the EISA Bus.

4.2.1 POSITIVELY DECODED MEMORY CYCLES TO MAIN MEMORY

The EISA/ISA master or DMA addresses that are positively decoded by the PCEB are forwarded to the PCI Bus. If the address is not positively decoded by the PCEB, the cycle is not forwarded to the PCI Bus. Subtractive and negative (82374SB only) decoding are not used on the EISA Bus.

The PCEB permits several EISA memory address ranges (items a-i) to be positively decoded. EISA Bus cycles to these regions are forwarded to the PCI Bus. Regions described by a-f and h are fixed and can be enabled or disabled independently. These regions are controlled by the EADC1 and EADC2 Registers.

The region described by g defines a space starting at 1 MByte with a programmable upper boundary of 4 GByte - 2 MByte. Within this region a hole can be opened. Its size and location are programmable to allow a hole to be opened in memory space (for a frame buffer on the EISA Bus, for example). The size of this region and the hole are controlled by the MCSTOM, MCSBOH and MCSTOH Registers. If a hole in main memory is defined, then accesses to that address range are contained within EISA, unless defined by the EISA-to-PCI memory regions as a PCI destined access. (See next section.)

a. 0–512 KByte

b. 512–640 KByte

c. 640–768 KByte (VGA memory)

d. 768–896 KByte in eight 16 KByte sections (Expansion ROM)

e. 896–960 KByte in four 16 KByte sections (lower BIOS area)

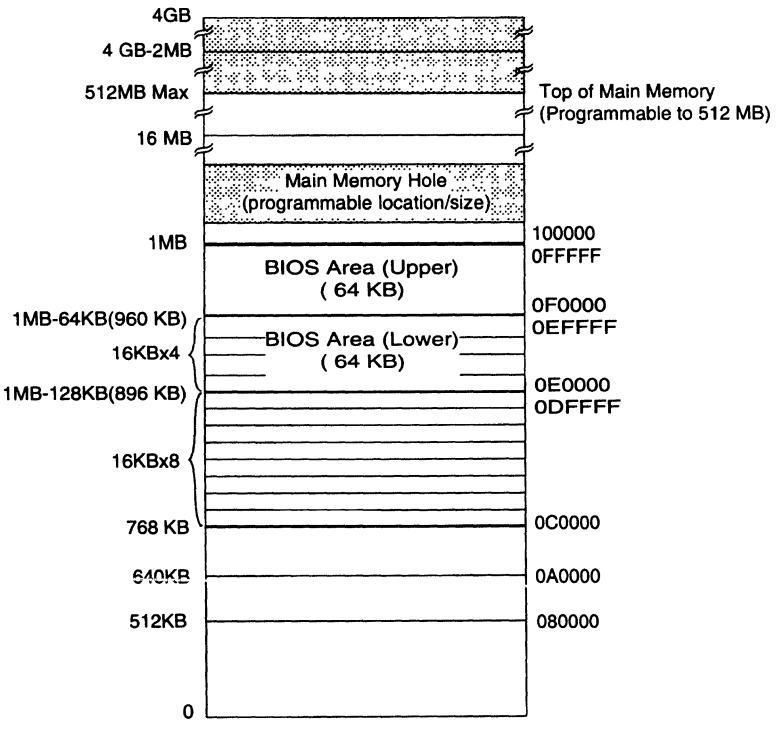
f. 960 KByte to 1 MByte (upper BIOS area)

g. 1 MByte to the top of memory (up to 4 GByte–2 MByte) within which a hole can be opened. Accesses to the hole are not forwarded to PCI. The top of the region can be programmed on 2 MByte boundaries up to 4 GByte–2 MByte. The hole can be between 64 KByte and 4 GByte–2 MByte in 64 KByte increments and located on any 64 KByte boundary.

h. 16 MByte–64 KByte to 16 MByte (FF0000–FFFFFFh). EISA memory cycles in this range are always forwarded to the PCI Bus, if this range exists in main memory as defined by the MEMCS# registers. In this case, the enable/disable control bit in EADC2 Register is a don't care. If this range is not defined in main memory (i.e., above the top of memory or defined as a hole in the main memory), EISA cycles to this address range are forwarded to the PCI Bus, based on the enable/disable bit in the EADC2 Register. (This capability is used to support access of BIOS at 16 MBytes.)

i. 4 GByte–2 MByte to 4 GByte. The address map must be programmed in a such way that this address range is always contained within EISA. This is to avoid conflict with local BIOS memory response in this address range. If this region must be mapped to PCI, then programming of the BIOS decoder Registers contained within the ESC must ensure that there is no conflict. To map this region to PCI, one of the four programmable EISA-to-PCI memory regions must be used. Mapping of this region to the PCI might be required in the case when BIOS resides on the PCI and the PCI/EISA system must have consistent address maps for both PCI and EISA.

For detailed information on the PCEB registers used to control these address regions, refer to Section 3.1, PCEB Configuration Registers.



NOTES:
1. The four programmable EISA-to-PCI regions (Regions[4:1]) are not shown. These regions can be located anywhere within the 4 GByte memory space. Accesses to these regions are forwarded to the PCI Bus.
2. EISA cycles that access shaded regions are contained to the EISA Bus, unless the access hits one of the four programmable EISA-to-PCI regions described in note 1.
3. Memory accesses to non-shaded areas are forwarded to the PCI Bus, if the regions is enabled. If the region is disabled, the access is contained to the EISA Bus, unless the access hits one of the four programmable EISA-to-PCI regions described in note 1.

Figure 6. EISA Address Decoder Map

EISA memory cycles positively decoded for forwarding to PCI are allowed to be handled by the PCEB's Line Buffer management logic, if the line buffering is enabled through the PCICON Register.

For EISA-to-PCI transactions there are 2 modes of operation of the PCEB's Line Buffers:

- Buffered: Read-prefetch, write posting with data assembly.
- Non-buffered: Bypass path used.

Accesses within the main memory address range are normally performed in buffered mode. If there are programmable memory regions defined within the main memory hole or above the top of the main memory MEMREGN[4:1], then the mode of access depends on configuration bits of the EPMRA Register. Access attribute bits associated with these regions override the default buffered mode for a particular address range in the case of programmable regions overlapping with active main memory regions.

Access to the 64 KByte area at the top of 16 MBytes (FF0000–FFFFFFh) on the PCI, if this region is within main memory or within the main memory hole and enabled via the EADC2 Register, are always forwarded in a non-buffered mode, unless overlapped with a programmable region that defines buffered access mode.

4.2.2 PROGRAMMABLE EISA-TO-PCI MEMORY ADDRESS REGIONS

The PCEB supports four programmable memory regions for EISA-to-PCI transfers. The PCEB positively decodes EISA memory accesses to these regions and forwards the cycle to the PCI Bus. This feature permits EISA master accesses to PCI devices that reside within these address ranges.

Regions can be enabled or disabled. After reset, all regions are disabled. Each region has an associated Base and Limit Address fields MEMREGN[4:1] that determine the size and location of each region. These registers are programmed with the starting address of the region (Base) and ending address of the region (Limit). The address range for a particular region is defined by the following equation:

$$Base\_Address \leq Address \leq Limit\_Address$$

These regions can be defined anywhere in the 4 GByte address space at 64 KByte boundaries and with 64 KByte granularity. In practical applications, the regions will be mapped within the main memory hole or above the top of the memory defined by the MEMCS# map.

Access to the memory locations within a region can be performed in one of two modes:

- **Non-Buffered Mode**: PCEB's EISA-to-PCI Line Buffers can be disabled for all EISA-to-PCI memory read/write accesses through the PCICON Register or for selected accesses through EPMRA Register.
- **Buffered Mode**: Line Buffers enabled. Read-prefetch and write-assembly/posting allowed (without strong ordering).

Since buffered mode provides maximum performance (and concurrency in non-GAT mode), it should be selected, unless the particular region is used for memory-mapped I/O devices. I/O devices can not be accessed in read-prefetch or write-assembly/posted fashion because of potential side-effects (see Section 6.0, Data Buffering).

4.2.3 PROGRAMMABLE EISA-TO-PCI I/O ADDRESS REGIONS

The PCEB provides four programmable I/O address regions. These regions are defined by Base and Limit addresses fields contained in the associated IOREGN[4:1] Registers. These regions can be defined anywhere within the 64 KByte I/O space on Dword boundaries (and with Dword granularity). See Section 4.1, PCEB Configuration Registers.

4.2.4 EXTERNAL EISA-TO-PCI I/O ADDRESS DECODER

Since the I/O address map may be highly fragmented, it is impractical to provide enough programmable regions to completely define mapping of registers for I/O devices on the PCI. The PCEB's input signal pin PIODEC# can be used, if a more complex I/O decode scheme is needed. PIODEC# complements the functions of the four PCEB programmable I/O regions with external decode logic. If PIODEC# is asserted during an EISA I/O cycle, the cycle is forwarded to the PCI Bus.

If the PIODEC# signal is not used, a pull-up resistor is required to provide an inactive signal level.

4.3 Palette DAC Snoop Mechanism

Some advanced graphics EISA/ISA expansion boards use the pre-DAC VGA pixel data from the VGA Special Feature Connector and merge it with advanced graphics data (multi-media for example). The merged data is then run through a replicated palette DAC on the advanced graphics expansion board to create the video monitor signal. The replicated palette DAC is kept coherent by snooping VGA palette DAC writes. Snooping becomes an issue in a system where the VGA controller is placed on the PCI Bus and the snooping graphics board is on the EISA expansion bus. Normally, the PCI VGA controller will respond to the palette DAC writes with DEVSEL#, so the PCEB will not propagate the cycle to the EISA Bus using subtractive decoding.

The burden for solving this problem is placed on the VGA subsystem residing on the PCI. The VGA subsystem on PCI must have an enable/disable bit associated with palette DAC accesses. When this bit is enabled the PCI VGA device responds in handshake fashion (generates DEVSEL#, TRDY#, etc.) to I/O reads and writes to the palette DAC space.

When this bit is disabled, the PCI VGA device responds in handshake fashion only to I/O reads to palette DAC space. I/O writes to the palette DAC space will be snooped (data latched) by the PCI VGA device, but the PCI VGA subsystem will not generate a DEVSEL#. In this case, the I/O write will be forwarded to the EISA Bus by the PCEB as a result of subtractive decode. The PCI VGA device must be able to snoop these cycles in the minimum EISA cycle time.

The state of palette-DAC snooping control bit does not affect I/O reads from the palette DAC space. Regardless of whether this bit is enabled or disabled, the PCI VGA device will service the I/O reads from the palette DAC space.

5.0 PCI INTERFACE

The PCEB provides the PCI Interface for the PCI-EISA Bridge. The PCEB can be an initiator (master) or target (slave) on the PCI Bus and supports the basic PCI Bus commands as described in Section 5.1.1, PCI Command Set. For EISA-to-PCI transfers, the PCEB is a master on the PCI Bus on behalf of the requesting EISA device. An EISA device can read and write either PCI memory or I/O space.

The PCEB forwards unclaimed PCI Bus cycles to EISA. For PCI Bus cycles that are not claimed, the PCEB becomes a slave on the PCI Bus (claiming the cycle via subtractive or negative decoding) and forwards the cycle to the EISA Bus. Note that negative decoding is only used on the 82374SB.

This section describes the PCI Bus transactions supported by the PCEB. The section also covers the PCI Bus latency mechanisms in the PCEB that limit a master's time on the bus and the PCEB support of parity. In addition, the PCEB contains PCI Bus arbitration circuitry that supports up to six masters. PCI Bus arbitration is described in Section 5.4.

NOTE:

1. All signals are sampled on the rising edge of the PCI clock. Each signal has a setup and hold window with respect to the rising clock edge, in which transitions are not allowed. Outside of this range, signal values or transitions have no significance.
2. The terms initiator and master are synonymous. Likewise, the terms target and slave are synonymous.
3. Readers should be familiar with the PCI Bus specification.

5.1 PCI Bus Transactions

This section presents the PCI Bus transactions supported by the PCEB.

5.1.1 PCI COMMAND SET

PCI Bus commands indicate to the target the type of transaction requested by the master. These commands are encoded on the C/BE[3:0] # lines during the address phase of a transfer. Table 5 summarizes the PCEB's support of the PCI Bus commands.

Table 5. PCEB-Supported PCI Bus Commands

C/BE[3:0] #	Command Type	Supported As Target	Supported As Initiator
0000	Interrupt Acknowledge	Yes	No
0001	Special Cycle	No	No
0010	I/O Read	Yes	Yes
0011	I/O Write	Yes	Yes
0100	Reserved	N/A(3)	N/A(3)
0101	Reserved	N/A(3)	N/A(3)
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved	N/A(3)	N/A(3)
1001	Reserved	N/A(3)	N/A(3)
1010	Configuration Read	Yes	No
1011	Configuration Write	Yes	No
1100	Memory Read Multiple	No(2)	No
1101	Reserved	N/A(3)	N/A(3)
1110	Memory Read Line	No(2)	No
1111	Memory Write and Invalidate	No(1)	No

NOTES:

1. As a target, the PCEB treats this command as a memory write command.
2. As a target, the PCEB treats this command as a a memory read command.
3. The PCEB considers a reserved command invalid and, as a target, completely ignores the transaction. All internal address decoding is ignored and the PCEB never asserts DEVSEL#. As a PCI master, the PCEB never generates a bus cycle with a reserved command type.

5.1.2 PCI CYCLE DESCRIPTIONS

Each PCI Command is listed below with the following format of information:

Command Type
- PCEB target support
  - -Decode method
  - -Data path
  - -PCEB response
  - -Result of no response on EISA
- PCEB initiator support
  - -Data path
  - -Conditions for generating command
  - -Result of no response on PCI

5.1.2.1 Interrupt Acknowledge

Target support:
Decode: Positive
Data Path: Flow through
Response:
The interrupt acknowledge cycle is subject to retry. If the PCEB is locked, or if the interrupt acknowledge cycle triggers buffer management activity, or if the EISA Bus is occupied by an EISA/ISA master or the DMA, the interrupt acknowledge cycle is retried.

The interrupt acknowledge command is a single byte read that is implicitly addressed to the interrupt controller in the ESC component. The address bits are logical "don't cares" during the address phase and the byte enables indicate to the PCEB that an 8-bit interrupt vector is to be returned on byte 0. After performing the necessary buffer management operations and obtaining ownership of the EISA Bus, the PCEB generates a single pulse on the PEREQ#/INTA# inter-chip signal and performs an I/O read cycle (on the EISA Bus) to the ESC internal registers residing at I/O address 04h. The ESC decode logic uses the PEREQ#/INTA# signal to distinguish between standard accesses to I/O address 04h (DMA controller) and special accesses that result in a vector being read by the PCEB. The PCEB holds the PCI Bus, in wait states, until the interrupt vector is returned. PEREQ#/INTA# remains asserted until the end of the read cycle.

Result of no response on EISA:
The PCEB runs a standard length EISA I/O read cycle and terminates normally. The value of the data returned as an interrupt vector is meaningless.

Initiator support: None.

NOTE:
The PCEB only responds to PCI interrupt acknowledge cycles if this operation is enabled via bit 5 of the PCICON Register).

5.1.2.2 Special Cycle

Target support: None.

Initiator support: None.

5.1.2.3 I/O Read

Target support:
Decode: Positive (PCEB and some ESC registers) & Subtractive
Data Path: Flow through
Response:
The PCEB claims I/O read cycles via positive or subtractive decoding and generates DEVSEL#. The internal PCEB registers (BIOS Timer) and the IDE and the 8259 registers are positively decoded. Any unclaimed cycle below 64 KByte is subtractively decoded and forwarded to the EISA Bus. The I/O read cycle is subject to retry. If the PCEB is locked, if the cycle triggers buffer management activity, or if the EISA Bus is occupied by an EISA/ISA master or the DMA, the I/O read cycle is retried. If the cycle gets retried due to an occupied EISA Bus, the EISA Bus is requested.

Once an I/O read cycle is accepted (not retried) by the PCEB, the PCI Bus is held in wait states using TRDY# until the cycle is completed internally or on the EISA Bus.

Burst I/O reads to the EISA Bus or to the PCEB are not supported. Therefore, any burst I/O read cycles decoded by the PCEB are target terminated after the first data transaction using the disconnect semantics of the STOP# signal (Disconnect A).

Result of no response on EISA: The PCEB runs a standard length EISA I/O cycle and terminates normally.

Initiator support:
The PCEB generates PCI Bus I/O read cycles on behalf of an EISA master. EISA cycles are forwarded to the PCI Bus if the I/O address is within one of four programmable I/O address regions as defined in Section 4.0 Address Decoding.

Result of no response on PCI: Master abort due to DEVSEL# time-out. PCEB returns data value FFFFFFFFh.

5.1.2.4 I/O Write

Target support:
Decode: Positive (PCEB/ESC registers) & Subtractive
Data Path: Flow through
Response:
I/O write cycles can be claimed by the PCEB via positive or subtractive decoding. In either case, the PCEB generates DEVSEL#. The internal PCEB registers (BIOS Timer), IDE registers and 8259 registers are positively decoded, if enabled. Any unclaimed cycle below 64 KByte is subtractively decoded and forwarded to the EISA Bus. The I/O write cycle is subject to retry. If the PCEB is locked, if the cycle triggers buffer management activity, or if the EISA Bus is occupied by an EISA/ISA master or the DMA, the I/O write cycle is retried. If the cycle is retried due to an occupied EISA Bus, the EISA Bus is requested.

Once an I/O write cycle is accepted (not retried) by the PCEB, the PCI Bus is held in wait states using TRDY# until the cycle is completed within the PCEB or on the EISA Bus.

Burst I/O writes to the EISA Bus or to the PCEB are not supported. Therefore, any burst I/O write cycles decoded by the PCEB are target terminated after the first data transaction using the disconnect semantics of the STOP# signal (Figure 5-12, Disconnect A).

Result of no response on EISA:
The PCEB runs a standard length EISA I/O cycle and terminates normally.

Initiator support:
The PCEB generates PCI I/O write cycles on behalf of an EISA master. EISA cycles are forwarded to the PCI Bus if the I/O address is within one of the four programmable I/O address regions defined in Section 4.0, Address Decoding.

Result of no response on PCI:
Master abort due to DEVSEL# time-out.

5.1.2.5 Memory Read

Target support:
Decode: Negative (82374SB only) and Subtractive
Data Path: Flow through
PCEB Response:
Memory read cycles may be claimed by the PCEB via negative or subtractive decoding. The PCEB claims the cycle by asserting DEVSEL#. Unclaimed PCI cycles (DEVSEL# time-out) are claimed by the PCEB via subtractively decoding and forwarded to the EISA Bus. The memory read cycle is subject to retry. If the PCEB is locked, if the cycle triggers buffer management activity, or if the EISA Bus is occupied by an EISA/ISA master or the DMA, the memory read cycle is retried. If the cycle is retried due to an occupied EISA Bus, the EISA Bus is requested.

Once a memory read cycle is accepted (not retried) by the PCEB, the PCI Bus is held in wait states, using TRDY#, until the cycle is completed to the EISA Bus.

Incremental burst memory reads destined for the EISA Bus take longer than the allowed 8 PCICLKs. Therefore, any burst memory read cycle decoded by the PCEB causes the PCEB to target terminate the cycle after the first data transaction using the disconnect semantics of the STOP# signal (Figure 5-8, Disconnect A).

Result of no response on EISA:
The PCEB runs a standard length EISA memory read cycle and terminates normally.

Initiator support:
Data Path: Line Buffer when enabled. Flow through when Line Buffer is disabled or it is a bypass cycle.

Cycle Generation Conditions:
As an initiator, the PCEB generates a PCI memory read cycle when it decodes an EISA memory read cycle destined to the PCI that can not be serviced by the Line Buffer. This condition occurs for EISA/ISA master and DMA cycles that can not be serviced by the Line Buffer because the Line Buffer is empty, there is a Line Buffer miss, or Line Buffering is disabled.

As an initiator, the PCEB only generates linear incrementing burst ordering that is signaled by AD[1:0] = 00 during the address phase. Other types of burst transfers (i.e. cache line toggle mode) are never initiated by the PCEB.

The PCEB generates a burst memory read when it is fetching 16 bytes into one of the four Line Buffers.

Result of no response on PCI:
Master abort due to DEVSEL# time-out. PCEB returns data value FFFFFFFFh.

5.1.2.6 Memory Write

Target support:
Decode: Negative (82374SB only) and Subtractive
Data Path: Flow through
PCEB Response:
Memory write cycles may be claimed by the PCEB via negative or subtractive decoding. The PCEB asserts DEVSEL# to claim the cycle. Unclaimed PCI cycles (DEVSEL# time-out) within the 4 GByte memory space are claimed by the PCEB via subtractively decoding and forwarded to the EISA Bus. The memory write cycle is subject to retry. If the PCEB is locked, the cycle triggers buffer management activity. If the EISA Bus is occupied by an EISA/ISA master or the DMA, the memory write cycle is retried. If the cycle is retried due to a disabled buffer because the EISA Bus is occupied, the EISA Bus is requested.

Once a memory write cycle is accepted (not retried) by the PCEB, the PCEB holds the PCI Bus in wait states (using TRDY#) until the cycle is completed on the EISA Bus.

Result of no response on EISA: The PCEB initiates a standard length EISA memory write cycle and terminates normally.

Initiator support:
Data Path: Line Buffer when enabled, flow through when Line Buffer is disabled.
Cycle Generation Conditions:
As an initiator, the PCEB generates a PCI memory write cycle when it decodes an EISA memory write cycle destined to PCI, that can not be serviced by the Line Buffer because it is disabled. This occurs for EISA/ISA masters and DMA cycles when the Line Buffer is disabled. The PCEB also generates a memory write cycle when the Line Buffer needs to be flushed. The Line Buffer is flushed under several conditions, including when the 16 byte line is full, when there is a "miss" to the current 16 byte line, or when it is required by the buffer management logic. (See Section 6.0, Data Buffering).

As an initiator, the PCEB generates only linear incrementing burst ordering that is signaled by AD[1:0] = "00" during address phase. Other types of burst transfers (i.e. cache line toggle mode) are never initiated by the PCEB.

Result of no response on PCI: Master abort due to DEVSEL# time-out.

5.1.2.7 Configuration Read, Configuration Write

Target support:
Decode: via IDSEL pin
Data Path: Flow through
Response:
The PCEB responds to configuration cycles by generating DEVSEL# when its IDSEL signal is asserted, regardless of the address. During configuration cycles, AD[7:2] are used to address the PCEB's configuration space. AD[31:8] are not used and are logical "don't cares". AD[1:0] must be zero.

Result of no response on EISA: N/A

Initiator support:
Configuration cycles are never generated by the PCEB.

5.1.2.8 Memory Read Multiple

Target support:
The PCEB aliases this command to a normal memory read cycle. See the Memory Read command description.

Initiator support:
Memory read multiple cycles are never generated by the PCEB.

5.1.2.9 Memory Read Line

Target support:
The PCEB aliases this command to a normal memory read. See the Memory Read command description.

Initiator support:
Memory read line cycles are never generated by the PCEB.

5.1.2.10 Memory Write And Invalidate

Target support:
Response:
The PCEB treats this command like a memory write. See the Memory Write command description.

Initiator support:
Cycle Generation Conditions:
The PCEB does not generate this command cycle.

5.1.3 PCI TRANSFER BASICS

The basic bus transfer mechanism on the PCI Bus is a burst. A burst is comprised of an address phase and one or more data phases. The PCI protocol specifies the following types of burst ordering (signaled via A[1:0] during the address phase):

A[1:0]	Burst Order
0 0	Linear Incrementing
0 1	Cache line toggle mode
1 X	Reserved

The PCEB only supports linear incrementing burst ordering as an initiator. Data transfers for ordering other than linear incrementing are disconnected by the PCEB (burst split into multiple single data transfers).

The fundamentals of all PCI data transfers are controlled with the following three signals:

- FRAME# is driven by the PCI master to indicate the beginning and end of a transaction.
- IRDY# is driven by the PCI master, allowing it to force wait states.
- TRDY# is driven by the PCI target, allowing it to force wait states.

The PCI Bus is idle when both FRAME# and IRDY# are negated. The first clock edge that FRAME# is sampled asserted is the address phase, and the address and bus command code are transferred on that clock edge. The next clock edge begins the first of one or more data phases. During the data phases, data is transferred between master and slave on each clock edge that both IRDY# and TRDY# are sampled asserted. Wait states may be inserted by either the master (by negating IRDY#) or the target (by negating TRDY#). When a PCI master has one more data transfer to complete the cycle (which could be immediately after the address phase), it negates FRAME#. IRDY# must be asserted at this time, indicating that the master is ready for the final data transfer. After the target indicates the final data transfer (TRDY# asserted), the master negates IRDY#, causing the target's PCI interface to return to the idle state (FRAME# and IRDY# negated), on the next clock edge.

For I/O cycles, PCI addressing is on byte boundaries and all 32 AD lines are decoded to provide the byte address. For memory cycles, AD[1:0] are used to define the type of burst ordering. For configuration cycles, DEVSEL# is strictly a function of IDSEL#. Configuration registers are selected as Dwords using AD[7:2]. The AD[1:0] must be 00 for the target to directly respond to the configuration cycle. The byte enables determine which byte lanes contain valid data.

Each PCI agent is responsible for its own positive address decode. Only one agent (the PCEB) on the PCI Bus may use subtractive decoding. The little endian addressing model is used.

The byte enables are used to determine which bytes carry meaningful data. These signals are permitted to change between data phases. The byte enables must be driven valid from the edge of the clock that starts each data phase and must stay valid for the entire data phase. Figure 7, the data phases begin on clocks 3 and 4. (Changing byte enables during a read burst transaction is generally not useful, but is supported on the bus.) The master is permitted to change the byte enables on each new data phase, although the read diagram does not show this. The timing for changing byte enables is the same for read and write transactions. If byte enables are important for the target on a read transaction, the target must wait for the byte enables to be driven on each data phase before completing the transfer.

5.1.3.1 Turn-Around-Cycle Definition

A turn-around-cycle is required on all signals that may be driven by more than one agent. The turn-around-cycle is required to avoid contention when one agent stops driving a signal and another agent begins, and must last at least one clock. The symbol that represents a turn-around-cycle in the timing relationship figures is a circular set of two lines, each with an arrow that points to the other's tail. This turn-around-cycle occurs at different times for different signals. For example, the turn-around-cycle for IRDY#, TRDY# and DEVSEL# occurs during the address phase and for FRAME#, C/BE# and AD, it occurs during the idle cycle.

5.1.3.2 Idle Cycle Definition

The cycle between clocks 7 and 8 in Figure 8 is called an idle cycle. Idle cycles appear on the PCI Bus between the end of one transaction and the beginning of the next. An idle cycle occurs when both FRAME# and IRDY# are negated.

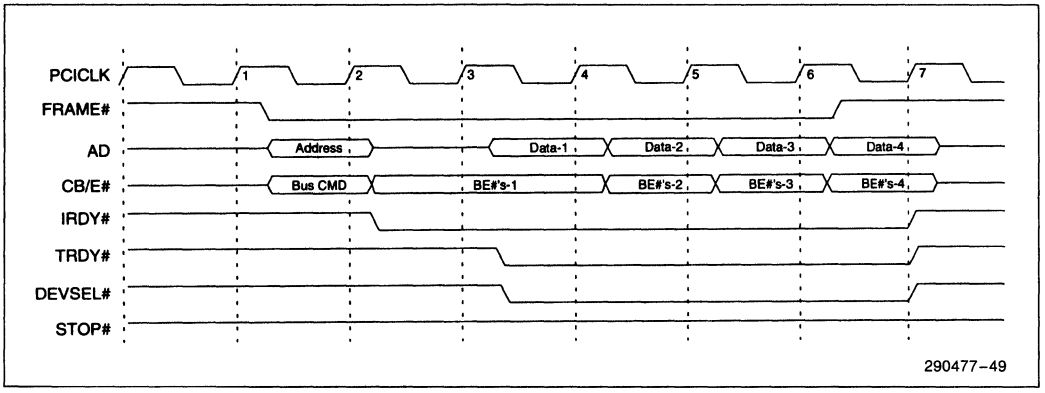


Figure 7. PCEB Burst Read from PCI Memory

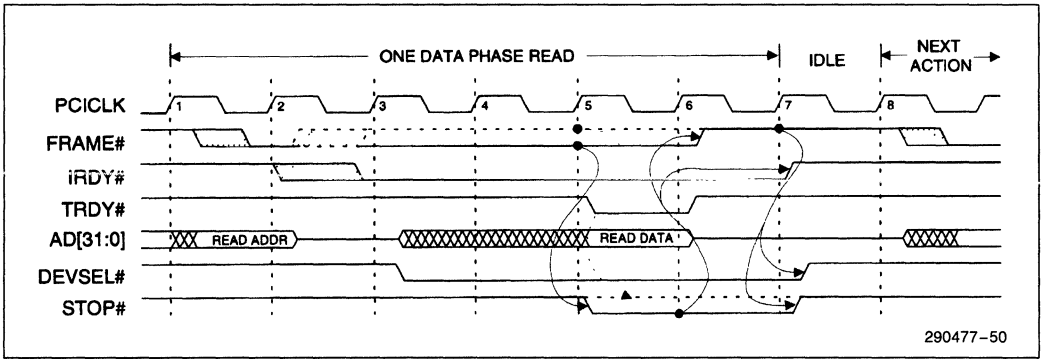


Figure 8. PCI Master Read from the PCEB (Burst with Target Termination)

5.1.4 BASIC READ

As a PCI master, the PCEB performs memory and I/O read transfers. Figure 7 shows a PCEB zero wait state burst read from PCI memory (PCEB is a master). If buffering of memory accesses is enabled, read transfers use prefetching. When reading data from PCI memory, the PCEB requests a minimum of 16 bytes (one data line of the Line Buffer), via a four data phase burst read cycle, to fill one of its internal Line Buffers. The PCEB does not buffer PCI I/O reads and only required data is transferred during these cycles. Read Cycles to PCI are generated on behalf of EISA/ISA masters and DMA devices.

The PCEB asserts FRAME# on clock 1 and places the address on AD[31:2]. CB/E[3:0]# contain a valid bus command. AD[1:0] contain the byte address for I/O cycles, burst order indication for memory cycles, and are 00 for configuration cycles.

The clock following the address phase is the beginning of the data phase. During the data phase, C/BE[3:0]# indicate which byte lanes are involved in the transaction. If the byte lanes involved in the transaction are different for data 1 and data 2, the PCEB drives new C/BE[3:0] values on clock 4. C/BE[3:0]# remain active until the end of the burst transfer.

The first data phase of a read transaction requires a turn-around-cycle, which is enforced by the target preventing the assertion of TRDY# until at least clock 3. The PCEB stops driving the address at clock 2. The target can not drive the AD bus until clock 3. This allows enough time for the PCEB to float its AD outputs. The target is required to drive the AD lines as soon as possible after clock 3, even though valid data may not be ready and the target may want to stretch the initial data phase by delaying TRDY#. This insures that the AD lines are not left floating for long intervals. The target must continue to drive these lines until the end of the burst transaction.

A single data phase is completed when the initiator of the cycle samples TRDY# asserted on the same clock that IRDY# is asserted. To add wait states, the target must negate TRDY# for one or more clock cycles. As a master, the PCEB does not add wait states. In Figure 7, data is transferred on clocks 4 and 5. The PCEB knows, at clock 6, that the next data phase is the last and negates FRAME#. As noted before, the PCEB can burst a maximum of four data cycles when reading from PCI memory.

As a PCI target, the PCEB responds to both I/O and memory read transfers. Figure 8 shows the PCEB, as a target, responding to a PCI master read cycle. For multiple read transactions, the PCEB always target terminates after the first data read transaction by asserting STOP# and TRDY#. These signals are asserted at the end of the first data phase. For single read transactions, the PCEB completes the cycle in a normal fashion (by asserting TRDY# without asserting STOP#). Figure 8 shows the fastest PCEB response to an access of an internal configuration register. During EISA Bus read accesses, the PCEB always adds wait states by negating TRDY# until the transfer on the EISA Bus is completed.

When the PCEB, as a target, samples FRAME# active during a read cycle and positively decodes the cycle, it asserts DEVSEL# on the following clock (clock 3 in Figure 8). Note that, if the PCEB subtractively or negatively (82374SB only) decodes the cycle, DEVSEL# is not asserted for two to three PCICLK's after FRAME# is sampled active. (see Section 5.1.9, Device Selection). When the PCEB asserts DEVSEL#, it also drives AD[31:0], even though valid data is not available. TRDY# is also driven from the same clock edge but it is not asserted until the PCEB is ready to drive valid data. TRDY# is asserted on the same clock edge that the PCEB drives valid data on AD[31:0]. If the PCEB presents valid read data during the first data phase and FRAME# remains active (multiple transaction indicated), the PCEB asserts TRDY# and STOP# to indicate target termination of the transfer.

5.1.5 BASIC WRITE

Figure 9 shows the PCEB, as a master, writing to PCI memory in zero wait states. Figure 10 shows the fastest response of the PCEB, as a target, to a memory or I/O write transaction generated by a PCI master.

As a PCI master, the PCEB performs memory write and I/O transfers. If buffering of memory accesses is enabled, write transfers are posted. When writing data to PCI memory, the PCEB writes a maximum of 16 bytes (one line of the Line Buffer) using a burst write cycle. I/O writes are always non-buffered transactions.

The PCEB generates PCI write cycles on behalf of EISA masters and DMA devices, and when the PCEB flushes its internal Line Buffer.

As a PCI target, the PCEB responds to both I/O and memory write transfers. If the EISA Bus is occupied, the PCI write is retried by the PCEB. When the PCEB owns the EISA Bus, the transaction proceeds. For burst I/O writes, the PCEB always target terminates after the first data transaction by asserting STOP# and TRDY# at the end of the first data phase. During a burst memory write, the PCEB always target terminates after the first data phase.

Figure 10 shows the fastest PCEB response to a write cycle targeted to an internal PCI configuration register. During I/O or memory write accesses to the EISA Bus, the PCEB always adds wait states. The PCEB adds wait states by holding TRDY# high until the transfer on the EISA Bus is completed.

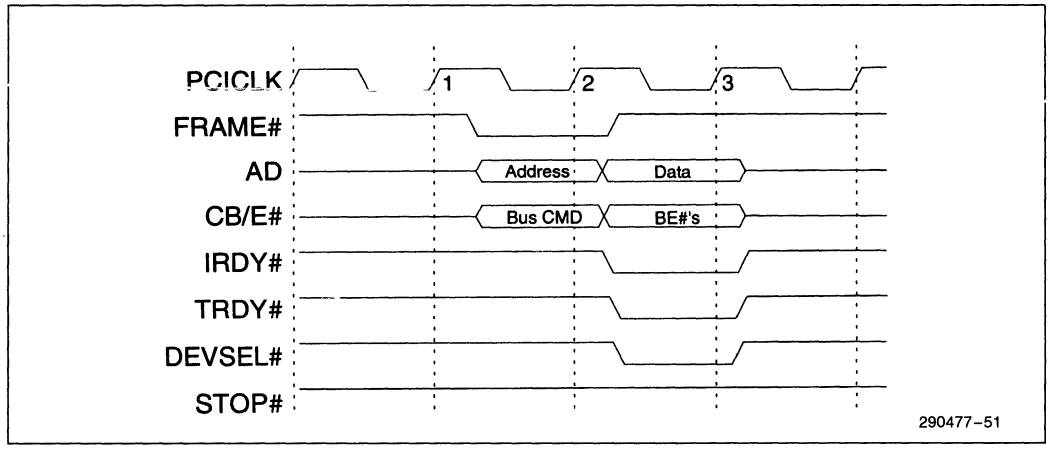


Figure 9. PCEB Write to PCI Memory

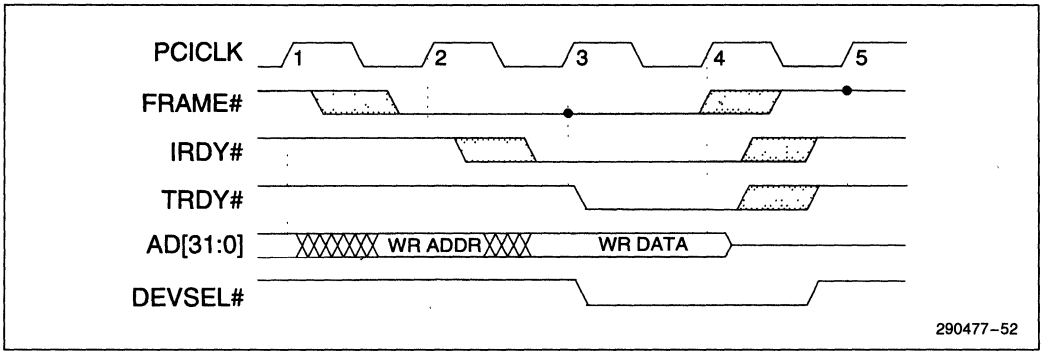


Figure 10. Fastest PCI Write to PCEB

5.1.6 CONFIGURATION CYCLES

One of the requirements of the PCI specification is that upon power up, PCI agents do not respond to any address. The only access allowed is through the IDSEL configuration mechanism. The PCEB is an exception to this since it controls access to the BIOS boot code. All PCEB/ESC subsystem addresses that are enabled after reset are accessible immediately after power up.

The configuration read or write command is used to configure the PCEB. During the address phase of the configuration read or write cycle, the PCEB samples its IDSEL (ID select) signal (not the address lines) to generate DEVSEL#. In this way, IDSEL acts as a chip select. During the address phase, AD[7:2] are used to select a particular configuration register and BE[3:0] to select a particular byte(s). The PCEB only responds to configuration cycles if AD[1:0] = 00. Reference Figure 11 for configuration reads and writes. Note that IDSEL is normally a "don't care", except during the address phase of a transaction. Upon decode of a configuration cycle and sampling IDSEL active, the PCEB responds by asserting DEVSEL# and TRDY#. An unclaimed configuration cycle is never forwarded to the EISA Bus.

Configuration cycles are not normally run in burst mode. If this happens, the PCEB splits the transfer into single cycles using the slave termination mechanism.

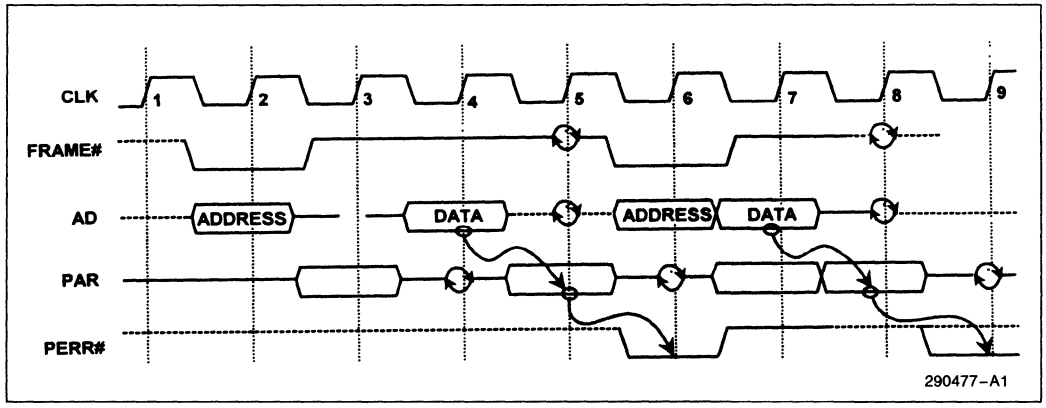


Figure 11. Configuration Cycle

2

5.1.7 INTERRUPT ACKNOWLEDGE CYCLE

The PCEB responds to an interrupt acknowledge cycle as decoded from the command during a valid address cycle (FRAME# asserted). The AD bus itself is a "don't care" to the PCEB during the address phase and, therefore, status of the internal PCI address decoder is not used for forwarding the cycle to the EISA Bus where the system interrupt controller resides.

The PCEB converts the PCI interrupt acknowledge cycle into an EISA I/O read access to the address 04h, with special semantics indicated to the ESC via the inter-chip signaling. Before the PCI interrupt acknowledge cycle can be converted into an EISA I/O read cycle, the EISA Bus must be owned. If the EISA Bus is not owned by the PCEB (EISAHLDA asserted), the PEREQ#/INTA# signal is asserted with PEREQ# semantics (PCI-to-EISA request). After the EISA Bus is acquired by the PCEB, the interrupt acknowledge sequence can proceed. The PCEB starts an I/O read cycle to address 04h and asserts PEREQ#/INTA# with INTA# semantics. The PEREQ#/INTA# remains asserted for the duration of the EISA I/O read cycle. Therefore, only a single pulse is generated on the PEREQ#/INTA# signal. Conversion of the single PCI interrupt acknowledge cycle into two interrupt acknowledge pulses (that is required for 8259 compatibility) occurs inside the ESC where the 8259-based interrupt controller resides. The ESC's EISA decoder uses the PEREQ#/INTA# signal (with INTA# semantics) to distinguish between normal I/O reads to the register located at address 04h (DMA1 Ch2 Base and Current Address) and the interrupt acknowledge sequence. The ESC holds the EISA Bus in wait states until the interrupt vector is returned to the PCEB (via SD[7:0]). The PCEB passes the vector to the PCI via AD[7:0] and then terminates the cycles both on EISA and PCI. Note that for compatibility reasons, only the ESC (containing the DMA controller) can respond to the EISA I/O read from 04h.

Figure 12 shows the PCI portion of the interrupt acknowledge sequence. The EISA portion of the sequence matches normal EISA I/O read timing, except that the PEREQ#/INTA# inter-chip signal is asserted during the bus cycle with INTA# semantics and, during the PCEB/ESC EISA Bus ownership exchange handshake, with PEREQ# semantics. Note that the PCEB responses to a PCI interrupt acknowledge cycle can be disabled by setting bit 5 in the PCI Control Register (PCICON) to 0.

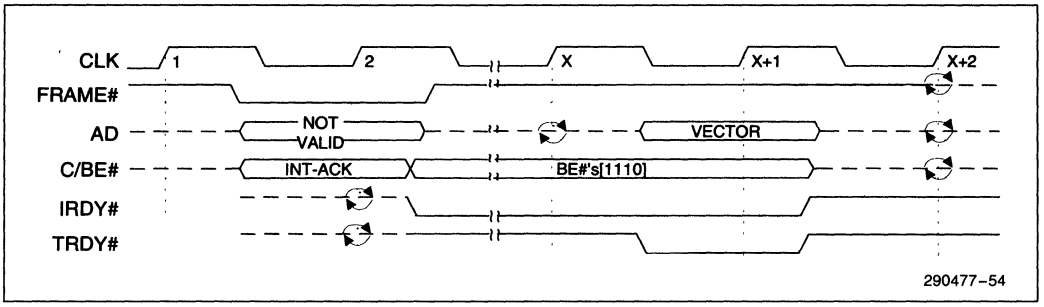


Figure 12. PCI Interrupt Acknowledge Cycle

5.1.8 EXCLUSIVE ACCESS

Refer to Figure 13, Figure 14, and Figure 15 for exclusive access timing relationships.

Target support:
PCI provides an exclusive access mechanism that allows non-exclusive accesses to proceed in the face of exclusive accesses. This is referred to as a Resource Lock. (Note that the exclusive access mechanism that locks the entire bus is Bus Lock.) The PCEB, as a resource, can be locked by any PCI initiator. In the context of locked cycles, the PCEB and entire EISA subsystem are considered a single resource. (EISA subsystem is indirectly locked during an exclusive access to the PCEB.) A locked access to any address contained within the EISA subsystem locks the entire subsystem from the PCI side. The PLOCK# signal is propagated to the EISA LOCK# signal. Note that write posting (PCI-to-EISA) is disabled for PCI locked cycles propagated to the EISA subsystem. The EISA Bus is not released to ESC until the locked sequence is complete. A subsequent PCI initiator access to the EISA subsystem, while it is locked, results in a retry. The PCEB becomes locked when it is the target of the access and PLOCK# is sampled negated during the address phase. The PCEB remains locked until FRAME# and PLOCK# are both sampled negated. When in a locked state, the PCEB only accepts requests when PLOCK# is sampled negated during the address phase. If PLOCK# is asserted during the address phase, the PCEB responds by asserting STOP# with TRDY# negated (RETRY).

As an unlocked target, the PCEB ignores PLOCK# when deciding if it should respond to a PCI address decoder hit. Also, if PLOCK# is sampled asserted during an address phase, the PCEB does not go into a locked state.

As a locked target, the PCEB responds to an initiator when it samples PLOCK# negated during the address phase of the cycle in which the PCEB is the target of the access. The locking master may negate PLOCK# at the end of the last data phase. When FRAME# and PLOCK# are both sampled negated, the PCEB goes to the unlocked state.

Note that the PCEB does not release the EISA Bus when it is in the locked state.

Initiator support:
When an EISA locked access to the PCI is encountered (EISA LOCK# asserted), the cycle is propagated to the PCI Bus as a PCI locked cycle. Line Buffers in the PCEB are bypassed. The PLOCK# signal must be negated (released) before an EISA agent can be granted the EISA Bus. Thus, when the PCEB acquires the PCI Bus on behalf of the EISA agent, a PCI LOCKED cycle can be performed, if needed.

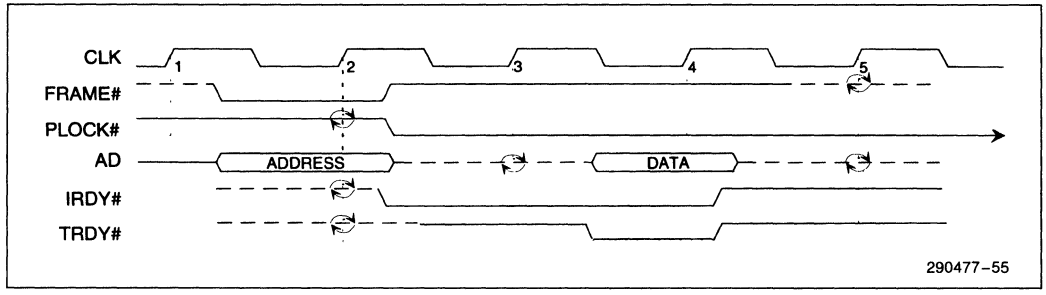


Figure 13. Beginning a Locked Cycle

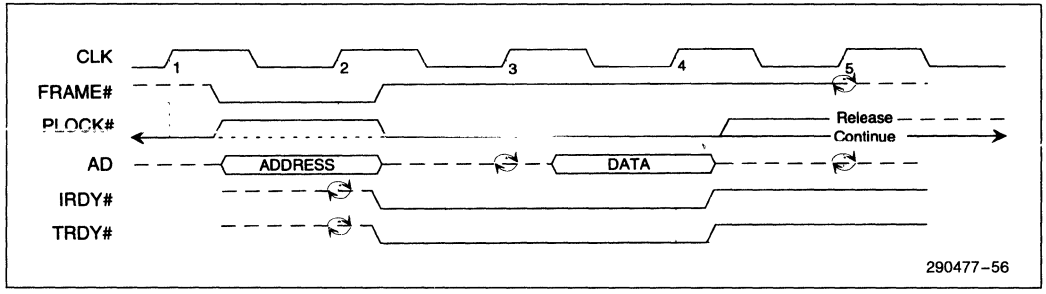


Figure 14. Continuing Locked Cycle

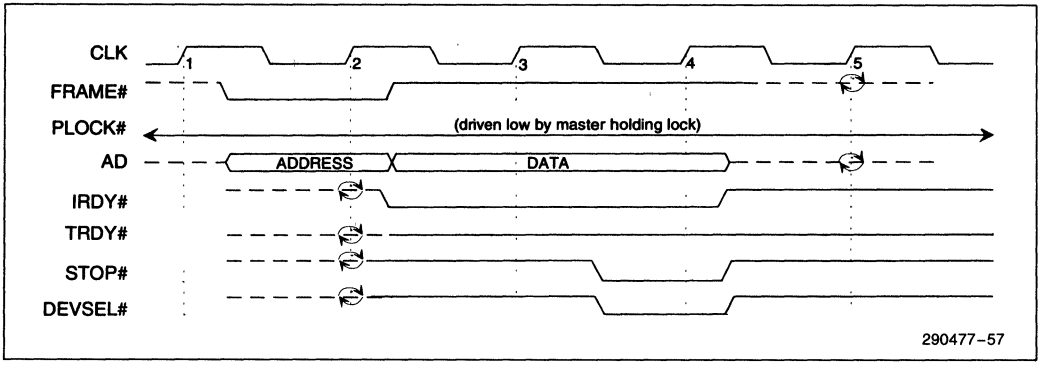


Figure 15. Access to Locked Target with PLOCK# Asserted During Address Phase

5.1.9 DEVICE SELECTION

The PCEB asserts DEVSEL# to indicate that it is the target of the PCI transaction. DEVSEL# is asserted when the PCEB, as a target, positively, subtractively, or negatively (82374SB only) decodes the PCI transaction. In all cases except one, once the PCEB asserts DEVSEL#, the signal remains asserted until FRAME# is negated (IRDY# is asserted) and either STOP# or TRDY# is asserted. The exception is a target abort, described in Section 5.1.10, Transaction Termination.

For most systems, PCI target devices are able to complete a decode and assert DEVSEL# within 2 or 3 clocks of FRAME# (medium and slow in the Figure 16). Accordingly, since the PCEB subtractively or negatively (82374SB only) decodes all unclaimed PCI cycles (except configuration cycles), it provides a configuration option to reduce by 1 clock the edge at which it samples DEVSEL#, allowing faster access to the expansion bus. Use of this option is limited by the slowest positive decode agent on the bus. This is described in more detail in Section 4.0, Address Decoding.

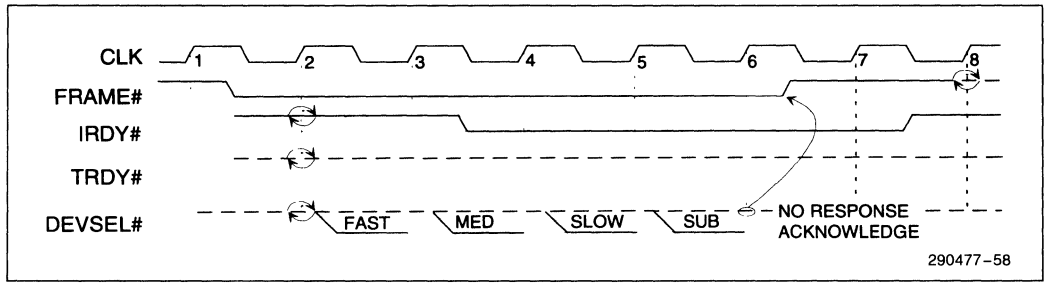


Figure 16. Device Selection (DEVSEL#)

5.1.10 TRANSACTION TERMINATION

Termination of a PCI cycle can be initiated by either a master or a target. The PCEB supports both master and target initiated termination. All transactions are concluded when FRAME# and IRDY# are both sampled negated, indicating that the PCI Bus is idle.

2

5.1.10.1 Master Initiated Termination

The PCEB supports three types of master initiated termination:

- **Completion:** Refers to the termination when the PCEB finishes the transaction normally. This is the most common type of termination.
- **Time-out:** Refers to termination when the PCEB's GNT# line is negated and its internal Master Latency Timer has expired. The intended transaction is not necessarily concluded. The timer may have expired because of a target-induced access latency, or because the intended operation was very long.
- **Abort:** Refers to termination when there is no target response (no DEVSEL# asserted) to a transaction within the programmed DEVSEL# response time.

Completions and time-outs are common while the abort is an abnormal termination. A normal termination of this type can be seen in Section 5.1.4 and 5.1.5 in the descriptions of the basic PCI read and write transaction.

The PCEB sends out a master abort (Figure 17) when the target does not respond to the PCEB-initiated transaction by asserting DEVSEL#. The PCEB checks DEVSEL# based on the programmed DEVSEL# sample point. If DEVSEL# is not asserted by the programmed sample point, the PCEB aborts the transaction by negating FRAME#, and then, one clock later, negating IRDY#. The master abort condition is abnormal and it indicates an error condition. The PCEB does not retry the cycle.

If the transaction is an EISA-to-PCI memory or I/O write, the PCEB terminates the EISA cycle with EXRDY. If the transaction is an EISA-to-PCI memory or I/O read, the PCEB returns FFFFFFFFh on the EISA Bus. This is identical to the way an unclaimed cycle is handled on the "normally ready" EISA Bus. If the Line Buffer is the requester of the PCI transaction, the master abort mechanism ends the PCI cycle, but no data is transferred into or out of the Line Buffer. The Line Buffer does not retry the cycle. The Received Master Abort Status bit in the PCI Status Register is set to 1 indicating that the PCEB issued a master abort.

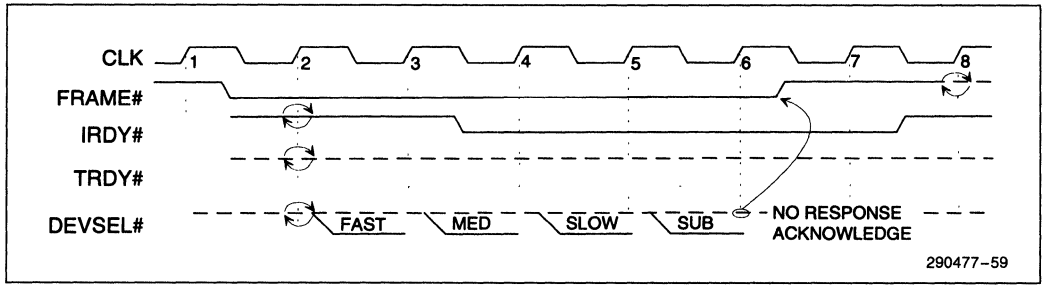


Figure 17. Master Initiated Termination (Master Abort)

5.1.10.2 Target Initiated Termination

The PCEB supports two forms of target-initiated termination:

- **Disconnect:** A disconnect termination occurs when the target is unable to respond within the latency guidelines of the PCI specifications. Note that this is not usually done on the first data phase.
- **Retry:** Retry refers to a termination requested because the PCEB is currently in a state that makes it unable to process the transaction.

Figure 18 and Figure 19 show four types of target-initiated terminations. The PCEB initiates a disconnect for PCI cycles destined to EISA after the first data phase due to incremental latency requirements. The difference between disconnect and retry is that the PCEB does not assert TRDY# for the retry case. This instructs the initiator to retry the transfer at a later time. No data is transferred in a retry termination since TRDY# and IRDY# are never both asserted. The PCEB retries a PCI initiator when:

- the PCEB buffers require management activity.
- the PCEB is locked and another PCI device attempts to select the PCEB without negating PLOCK# during the address phase.
- the EISA Bus is occupied by an EISA/ISA master or DMA.

Target abort is another form of target-initiated termination. Target abort resembles a retry, though the target must also negate DEVSEL#, along with assertion of STOP#. As a target, the PCEB never generates a target abort.

As a master, If the PCEB receives a target abort, it relinquishes the PCI Bus and sets the Received Target Abort Status bit in the PCI Status Register to a 1.

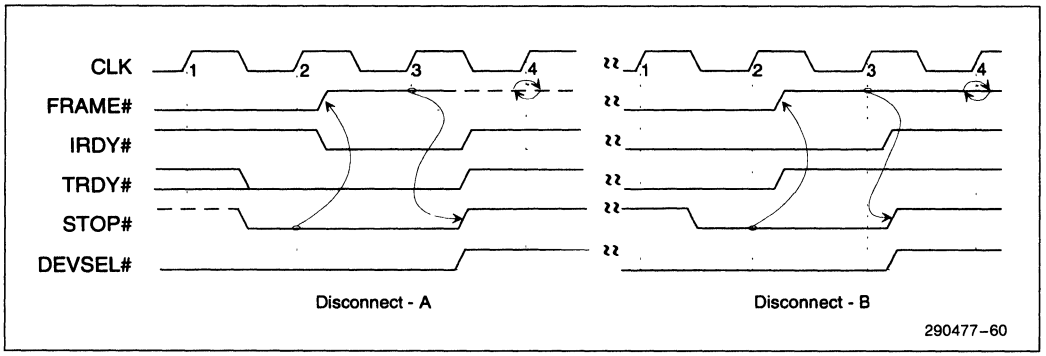


Figure 18. Target Initiated Termination

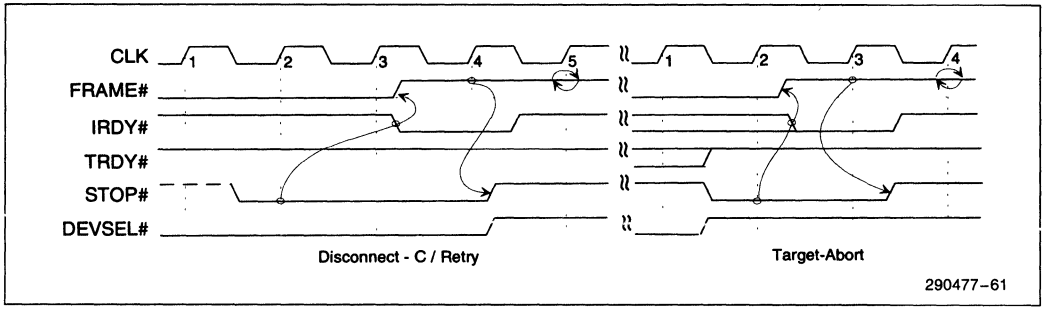


Figure 19. Target Initiated Termination

5.1.10.3 PCEB Target Termination Conditions

As a target, the PCEB terminates a transaction due to the following conditions:

Disconnect

- When a target, the PCEB always responds with a disconnect to a multiple data phase transaction (see the Incremental Latency Timer section),

Retry

- When the pending PCI cycle initiates buffer management activity.
- When the PCEB is locked, as a resource, and a PCI master tries to access the PCEB without negating the PLOCK# signal in the address phase.
- When the EISA Bus is occupied by an EISA/ISA master or DMA.

Target Abort

- The PCEB never generates a target abort.

5.1.10.4 PCEB Master Termination Conditions

As an initiator, the PCEB terminates a transaction due to the following conditions:

- Completion termination is always used by the PCEB signaling to the target that the PCEB is ready to complete the final data phase of the transaction.
- Master abort termination is issued if the PCEB does not receive a DEVSEL# from a target within five PCICLK's after FRAME# assertion. The PCEB sets the Received Master Abort Status bit in the PCI Status Register to a 1.
- Master initiated termination (disconnect) due to Master Latency Timer expiration when the PCEB's PCI Bus grant is removed (internal PCEBGNT# negated).

5.1.10.5 PCEB Responses/Results Of Termination

PCEB's response, as a target, to a master termination:

- Completion termination is the normal way of terminating a transaction.
- If a PCI initiator times out due to LT time-out and ends the current transaction, the PCEB cannot detect a difference between normal completion termination and time-out forced termination.

PCEB's response as a master to target termination:

- If the PCEB receives a target abort, it means that the target device is not capable of handling the transaction. The PCEB does not try the cycle again. If an EISA/ISA master or the DMA is waiting for the PCI cycle to terminate (EXRDY negated), the target abort condition causes the PCEB to assert EXRDY to terminate the EISA cycle. Note that write data is lost and the read data is meaningless. This is identical to the way an unclaimed cycle is handled on the "normally ready" EISA Bus. If the Line Buffer is the requester of the PCI transaction, the target abort mechanism ends the PCI cycle, but no valid data transfers are performed into or out of the Line Buffer. The Line Buffer does not try the cycle again. The Received Target Abort Status bit in the PCI Status Register is set to 1 indicating that the PCEB experienced a target abort condition.
- If the PCEB is retried as an initiator on the PCI Bus, it will remove it's request for 2 PCI clocks before asserting it again to retry the cycle.

- If the PCEB is disconnected as an initiator on the PCI Bus, it will respond very much as if it had been retried. The difference between retry and disconnect is that the PCEB did not see any data phase for the retry. Disconnect may be generated by a PCI slave when the PCEB is running a burst memory cycle to empty or to fill one line (16-byte) of the Line Buffers. In this case, the PCEB may need to finish a multi-data phase transfer and recycles through arbitration as required for a retry. An example is when an EISA agent (EISA/ISA master or DMA) issues a read request that the PCEB translates into a 16 byte prefetch (one line) and the PCEB is disconnected before the Line Buffer is completely filled.

5.1.11 PCI DATA TRANSFERS WITH SPECIFIC BYTE ENABLE COMBINATIONS

Non-Contiguous Combination of Byte Enables
As a master, the PCEB might generate non-contiguous combinations of data byte enables because of the nature of assembly operations in the Line Buffers.

As a target, the PCEB might need to respond to a non-contiguous combination of data byte enables. These cycles can not be passed directly to the EISA Bus; the EISA Bus specification does not allow non-contiguous combinations of byte enables. If this situation occurs, the PCEB splits the 32-bit transactions into two 16-bit transactions by first performing the lower word transfer (indicated by BE1# and BE0#) and then the upper word transfer (indicated by BE3# and BE2#).

BE[3:0]# = 1111
As a master, the PCEB might generate this combination of data byte enables during Line Buffer flush operations (burst write) to optimize the usage of the PCI Bus. Correct parity is driven during this transaction on the PCI Bus.

As a target, the PCEB might need to respond to this combination of data byte enables. If BE[3:0]# = 1111, the PCEB completes the transfer by asserting TRDY# and providing parity for read cycles. The PCEB does not forward the cycle to the EISA Bus.

5.2 PCI Bus Latency

The PCI specification provides two mechanisms that limit a master's time on the bus. They ensure predictable bus acquisitions when other masters are requesting bus access. These mechanisms are master-initiated termination supported by a Master Latency Timer (MLT) and a target-initiated termination (specifically, disconnect) supported by a target's incremental latency mechanism.

5.2.1 MASTER LATENCY TIMER (MLT)

The PCEB has a programmable Master Latency Timer (MLT). The MLT is cleared and suspended when the PCEB is not asserting FRAME#. The MLT is controlled via the MLT Register (see Section 4.1, PCEB Configuration Registers). When the PCEB, as a master, asserts FRAME#, it enables its MLT to count. If the PCEB completes its transaction (negates FRAME#) before the count expires, the MLT is ignored. If the count expires before the transaction completes (count = number clocks programmed into the MLT Register), the PCEB initiates a transaction termination as soon as its GNT# is removed. The number of clocks programmed into the MLT Register represents the guaranteed time slice (measured in PCICLKs) allotted to the PCEB; after which it surrenders the bus as soon as its GNT# is removed. (Actual termination does not occur until the target is ready.) Each master on PCI contains a master latency timer. The relative values programmed in each master timer determines how much of the PCI bandwidth is available to that master. Generally, if the EISA bus is heavily loaded with masters, the PCEB MLT register would be programmed with a relatively large value to give the PCEB a larger share of the PCI bus bandwidth.

5.2.2 INCREMENTAL LATENCY MECHANISM

As a target, the PCEB supports the Incremental Latency Mechanism for PCI-to-EISA cycles. The PCI specification states that for multi-data phase PCI cycles, if the incremental latency from current data phase (N) to the next data phase (N+1) is greater than eight PCICLKs, the target must manipulate TRDY# and STOP# to stop the transaction after the current data phase (N). All PCI-to-EISA cycles (memory read/write and I/O read/write) are automatically terminated (during a burst) after the first data phase because they require more than eight PCICLKs to complete on the EISA Bus.

Therefore, the PCEB does not need to specifically implement an 8 PCICLK timer and the PCEB handles a disconnect in a pre-determined fashion, based on the type of current transaction.

5.3 PCI Bus Parity Support And Error Reporting

PCI provides for parity and asynchronous system errors to be detected and reported separately. The PCEB/ESC chip set implements both mechanisms. The PCEB implements only parity generation and checking and it does not interface to the SERR# signal. Reporting of both PERR# and SERR# indicated errors is implemented in the ESC.

5.3.1 PARITY GENERATION AND CHECKING

The PCEB supports parity generation and checking on the PCI Bus. During the address and data phases, parity covers AD[31:0] and the C/BE[3:0]# lines, regardless of whether or not all lines carry meaningful information. Byte lanes that are not actually transferring data are still required to be driven with stable (albeit meaningless) data and are included in the parity calculation. Parity is calculated such that the number of 1s on AD[31:0], C/BE[3:0]#, and the PAR signals is an even number.

The role of the PCEB in parity generation/checking depends on the phase of the cycle (address or data), the type of bus cycle (read or write), and whether the PCEB is a master or target. The following paragraphs and Figure 20 summarize the behavior of the PCEB during the address and data phase of a PCI Bus cycle.

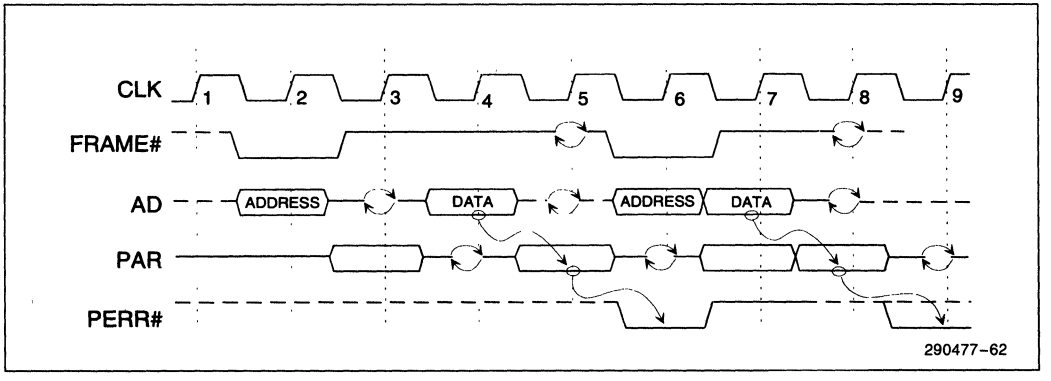


Figure 20. Parity Operation

5.3.1.1 Address Phase

As a master, the PCEB drives AD[31:0] and C/BE[3:0]# and calculates the corresponding parity value and drives it on the PAR signal, 1 clock later. As a target, the PCEB does not check parity during the address phase of a bus cycle.

5.3.1.2 Data Phase

As a master during a write cycle, the PCEB drives AD[31:0] and C/BE[3:0]# and calculates the corresponding parity value and drives it on the PAR signal, 1 clock later.

As a master during a read cycle, the PCEB only drives C/BE[3:0]#. The responding target drives AD[31:0] lines (data) and calculates parity based on the received C/BE[3:0]# and outgoing AD[31:0] signals. The target drives PAR during the following clock. The PCEB calculates parity based on the outgoing C/BE[3:0]# and the incoming AD[31:0] signals at the end of the data phase. It compares it with the incoming value of the PAR signal and asserts PERR# if there is no match.

As a target during a write cycle, the PCEB calculates parity on the incoming AD[31:0] and C/BE[3:0]# signals, and compares the result on the next clock with the incoming value on the PAR signal. If the value does not match, the PCEB asserts PERR#.

As a target during a read cycle, the PCEB calculates parity on the incoming C/BE[3:0]# and outgoing AD[31:0] signals. The PCEB drives the calculated parity value during the next clock. The master of the transaction receives the data, calculates parity on its outgoing C/BE[3:0]# and incoming AD[31:0] signals and compares its calculated value, on the next clock, with the parity value on the PAR signal (supplied by the PCEB). If the values do not match, the master asserts PERR#.

5.3.2 PARITY ERROR—PERR# SIGNAL

When the PCEB is involved in a bus transaction (master or target), it asserts the PERR# signal, if enabled via the PCICMD Register, to indicate a parity error for the bus cycle. PERR# is a sustained tri-state (s/t/s) type of signal (see Section 2.0, Signal Description). Note that PCI parity errors signaled by PERR#, are reported to the host processor via the ESC's system interrupt control logic. When the PCEB detects a parity error during one of its bus transactions, it sets the parity error status bit in the PCI Status Register, regardless of whether the PERR# signal is enabled via the PCICMD Register.

5.3.3 SYSTEM ERRORS

The PCEB does not generate system errors (SERR#). Thus, the PCEB does not have the capability of indicating parity errors during the address phase in which it is a potential target (i.e. not a master). Note that system errors are reported via the ESC (companion chip).

5.4 PCI Bus Arbitration

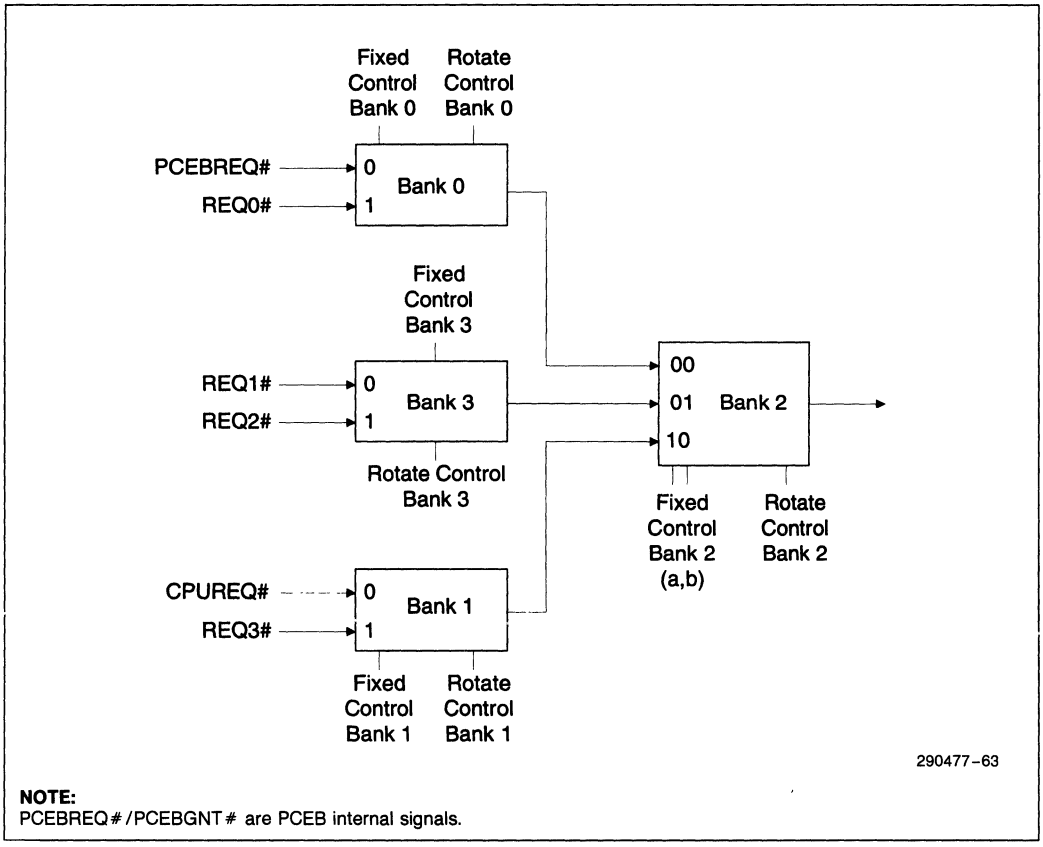
The PCEB contains a PCI Bus arbiter that supports six PCI Bus mastersThe Host/PCI Bridge, PCEB, and four other masters. The PCEB's REQ#/GNT# signals are internal. An external arbiter is not supported. Note that, for proper arbiter operation, CPUREQ# must be sampled high by the PCEB when PCIRST# makes a low-to-high transition. The internal arbiter contains several features that contribute to system efficiency:

- Use of the internal RESUME# signal to re-enable a backed-off initiator in order to minimize PCI Bus thrashing when the PCEB generates a retry.
- A programmable timer to re-enable retried initiators after a number of PCICLK's.
- A programmable PCI Bus lock or PCI resource lock function.
- The CPU Host/PCI can be optionally parked on the PCI Bus.

In addition, the PCEB has three PCI sideband signals (FLUSHREQ#, MEMREQ#, and MEMACK#) that are used to control system buffer coherency and control operations for the Guaranteed Access Time (GAT) mode.

5.4.1 PCI ARBITER CONFIGURATION

The PCI arbitration priority scheme is programmable through the configuration registers. The arbiter consists of four banks that can be configured so that the six masters to be arranged in a purely rotating priority scheme, one of 24 fixed priority schemes, or a hybrid combination (Figure 21).



NOTE:
PCEBREQ#/PCEBGNT# are PCEB internal signals.

Figure 21. Arbiter Conceptual Block Diagram

The PCEB implements PCI arbiter priority configuration registers ARBPRI and ARBPRIX mapped in the PCI's configuration space. Definition of the registers are as follows:

ARBPRI Register

Bit	Description
7	Bank 3 Rotate Control
6	Bank 2 Rotate Control
5	Bank 1 Rotate Control
4	Bank 0 Rotate Control
3	Bank 2 Fixed Priority Mode Select B
2	Bank 2 Fixed Priority Mode Select A
1	Bank 1 Fixed Priority Mode Select
0	Bank 0 Fixed Priority Mode Select

This register defaults to 04h at reset. This selects fixed mode #4 with the CPU the highest priority device guaranteeing that BIOS accesses can take place.

ARBPRIX Register

Bit	Description
7	Bank 3 Fixed Priority Mode select
6:0	Reserved

This register defaults to 00h at reset. The default value selects REQ1# as a higher priority request than REQ2# when Bank 3 operates in the fixed priority mode.

5.4.1.1 Fixed Priority Mode

The twelve selectable fixed priority schemes are listed in Table 6.

Table 6. Fixed Priority Mode Bank Control Bits

Mode	Bank					Priority				
	3	2b	2a	1	0	Highest				Lowest
0	0	0	0	0	0	PCEBREQ#	REQ0#	REQ1#/ REQ2#	CPUREQ#	REQ3#
1	0	0	0	0	1	REQ0#	PCEBREQ#	REQ1#/ REQ2#	CPUREQ#	REQ3#
2	0	0	0	1	0	PCEBREQ#	REQ0#	REQ1#/ REQ2#	REQ3#	CPUREQ#
3	0	0	0	1	1	REQ0#	PCEBREQ#	REQ1#/ REQ2#	REQ3#	CPUREQ#
4	0	0	1	0	0	CPUREQ#	REQ3#	PCEBREQ#	REQ0#	REQ1#/ REQ2#
5	0	0	1	0	1	CPUREQ#	REQ3#	REQ0#	PCEBREQ#	REQ1#/ REQ2#
6	0	0	1	1	0	REQ3#	CPUREQ#	PCEBREQ#	REQ0#	REQ1#/ REQ2#
7	0	0	1	1	1	REQ3#	CPUREQ#	REQ0#	PCEBREQ#	REQ1#/ REQ2#
8	0	1	0	0	0	REQ1#/ REQ2#	CPUREQ#	REQ3#	PCEBREQ#	REQ0#
9	0	1	0	0	1	REQ1#/ REQ2#	CPUREQ#	REQ3#	REQ0#	PCEBREQ#
A	0	1	0	1	0	REQ1#/ REQ2#	REQ3#	CPUREQ#	PCEBREQ#	REQ0#
B	0	1	0	1	1	REQ1#/ REQ2#	REQ3#	CPUREQ#	REQ0#	PCEBREQ#

Table 6. Fixed Priority Mode Bank Control Bits (Continued)

Mode	Bank					Priority				
	3	2b	2a	1	0	Highest				Lowest
	x	1	1	x	x	Reserved				
10	1	0	0	0	0	PCEBREQ#	REQ0#	REQ2#/ REQ1#	CPUREQ#	REQ3#
11	1	0	0	0	1	REQ0#	PCEBREQ#	REQ2#/ REQ1#	CPUREQ#	REQ3#
12	1	0	0	1	0	PCEBREQ#	REQ0#	REQ2#/ REQ1#	REQ3#	CPUREQ#
13	1	0	0	1	1	REQ0#	PCEBREQ#	REQ2#/ REQ1#	REQ3#	CPUREQ#
14	1	0	1	0	0	CPUREQ#	REQ3#	PCEBREQ#	REQ0#	REQ2#/ REQ1#
15	1	0	1	0	1	CPUREQ#	REQ3#	REQ0#	PCEBREQ#	REQ1#/ REQ2#
16	1	0	1	1	0	REQ3#	CPUREQ#	PCEBREQ#	REQ0#	REQ2#/ REQ1#
17	1	0	1	1	1	REQ3#	CPUREQ#	REQ0#	PCEBREQ#	REQ2#/ REQ1#
18	1	1	0	0	0	REQ2#/ REQ1#	CPUREQ#	REQ3#	PCEBREQ#	REQ0#
19	1	1	0	0	1	REQ2#/ REQ1#	CPUREQ#	REQ3#	REQ0#	PCEBREQ#
1A	1	1	0	1	0	REQ2#/ REQ1#	REQ3#	CPUREQ#	PCEBREQ#	REQ0#
1B	1	1	0	1	1	REQ2#/ REQ1#	REQ3#	CPUREQ#	REQ0#	PCEBREQ#
	x	1	1	x	x	Reserved				

Note that these two tables are permutations of the same table with different value of the Bank 3 fixed priority control bit. The fixed bank control bit(s) selects which requester is the highest priority device within that particular bank. Bits[7:4] must be programmed to all 0's (rotate mode disabled) to get these combinations. The selectable fixed priority schemes provide 24 of the 128 possible fixed mode permutations possible for the six masters.

5.4.1.2 Rotating Priority Mode

When any bank rotate control bit is set to a one, that particular bank rotates between the requesting inputs. Any or all banks can be set in rotate mode. If all four banks are set in rotate mode, the six supported masters are all rotated and the arbiter is in a pure rotating priority mode. If, within a rotating bank, the highest priority device (a) does not have an active request, the lower priority device (b or c) will be granted the bus. However, this does not change the rotation scheme. When the bank toggles, device b is the highest priority. Because of this, the maximum latency a device can encounter is two complete rotations.

5.4.1.3 Mixed Priority Mode

Any combination of fixed priority and rotate priority modes can be used in different arbitration banks to achieve a specific arbitration scheme.

5.4.1.4 Locking Masters

When a master acquires the PLOCK# signal, the arbiter gives that master highest priority until PLOCK# is negated and FRAME# is negated. This insures that a master that locked a resource will eventually be able to unlock that same resource.

5.4.2 ARBITRATION SIGNALING PROTOCOL

An agent requests the PCI Bus by asserting its REQ#. When the arbiter determines that an agent may use the PCI Bus, it asserts the agent's GNT#. Figure 22 shows an example of the basic arbitration cycle. Two agents (A and B) are used to illustrate how the arbiter alternates bus accesses. Note in Figure 22 that the current owner of the bus may keep its REQ# (REQ#-A) asserted when it requires additional transactions.

REQ#-A is asserted prior to or at clock 1 to request use of the PCI Bus. Agent A is granted access to the bus (GNT#-A is asserted) at clock 2. Agent A may start a transaction at clock 2 because FRAME# and IRDY# are negated and GNT#-A is asserted. Agent A's transaction starts when FRAME# is asserted (clock 3). Agent A requests another transaction by keeping REQ#-A asserted.

When FRAME# is asserted on clock 3, the arbiter determines that agent B has priority and asserts GNT#-B and negates GNT#-A on clock 4. When agent A completes its transaction on clock 4, it relinquishes the bus. All PCI agents can determine the end of the current transaction when both FRAME# and IRDY# are negated. Agent B becomes the PCI Bus owner on clock 5 (FRAME# and IRDY# are negated) and completes its transaction on clock 7. Note that REQ#-B is negated and FRAME# is asserted on clock 6, indicating that agent B requires only a single transaction. The arbiter grants the next transaction to agent A because its REQ# is still asserted.

5.4.2.1 REQ# And GNT# Rules

Figure 22 illustrates basic arbitration. Once asserted, GNT# may be negated according to the following rules:

1. If GNT# is negated at the same time that FRAME# is asserted, the bus transaction is valid and will continue.
2. One GNT# can be negated coincident with another being asserted, if the bus is not in the idle state. Otherwise, a one clock delay is incurred between the negation of the current master's GNT# and assertion of the next master's GNT#, to comply with the PCI specification.
3. While FRAME# is negated, GNT# may be negated, at any time, in order to service a higher priority master, or in response to the associated REQ# being negated.
4. If the MEMREQ# and MEMACK# are asserted, once the PCEB is granted the PCI Bus, the arbiter will not remove the internal grant until the PCEB removes it's request.

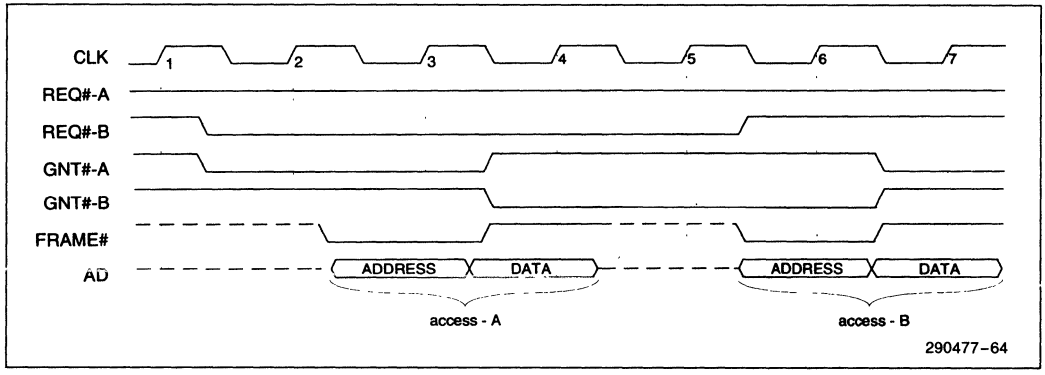


Figure 22. Basic Arbitration

5.4.2.2 Back-to-Back Transactions

Figure 23 illustrates arbitration for a back-to-back access. There are two types of back-to-back transactions by the same initiator; those that do not require a turn-around-cycle (see Section 5.1.3.1, Turn-Around-Cycle Definition) and those that do. A turn-around-cycle is not required when the initiator's second transaction is to the same target as the first transaction (to insure no TRDY# contention), and the first transaction is a write. This is a fast back-to-back. Under all other conditions, the initiator must insert a minimum of one turn-around-cycle.

During a fast back-to-back transaction, the initiator starts the next transaction immediately, without a turn-around-cycle. The last data phase completes when FRAME# is negated, and IRDY# and TRDY# are asserted. The current initiator starts another transaction on the same PCICLK that the last data is transferred for the previous transaction.

As a master, the PCEB does not know if it is accessing the same target, and, thus, does not generate fast back-to-back accesses. As a slave, the PCEB is capable of decoding fast back-to-back cycles.

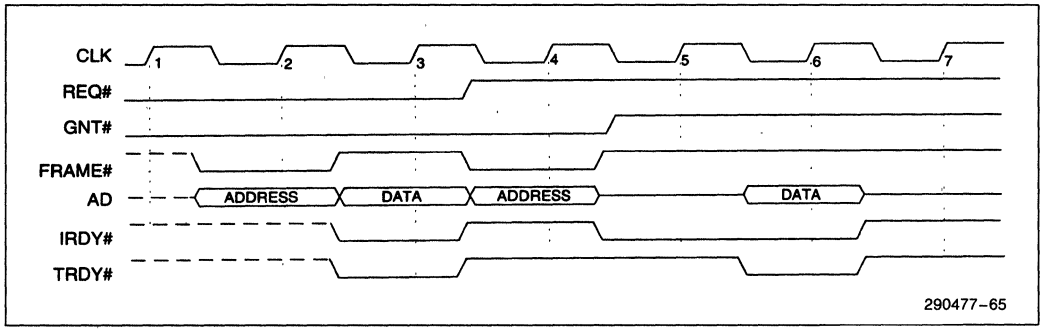


Figure 23. Arbitration for Back-to-Back Access

5.4.3 RETRY THRASHING RESOLVE

When a PCI initiator's access is retried, the initiator releases the PCI Bus for a minimum of two PCI clocks and then normally requests the PCI Bus again. To avoid thrashing of the bus with retry after retry, the PCI arbiter's state tracer provides REQ# masking. Tracking retried masters requires latching GNT# during FRAME# so that the correct retried master can be masked. The state tracer masks a REQ# after that particular agent is retried on the PCI Bus. The state tracer differentiates between two retry events. The two events include PCEB target retries and all other retries

For initiators that were retried by the PCEB as a target, the masked REQ# is flagged to be cleared upon RESUME# active. All other retries trigger the Master Retry Timer (described in Section 5.4.4.2, Master Retry Timer). When this timer expires, the mask is cleared.

5.4.3.1 Resume Function

The PCEB forces a retry to a PCI master (resulting in masking the REQ# of that master) for the following:

1. Buffer management activities (See Section 6.0, Data Buffering).
2. The EISA Bus is occupied by an EISA/ISA master or DMA.
4. The PCEB is locked as a resource and PLOCK# is asserted during the address phase.

5.4.3.2 Master Retry Timer

For any other retried PCI cycle, the arbiter masks the REQ# and flags it to be cleared by the expiration of a programmable timer. The first retry in this category triggers the programmable timer. Subsequent retries in this category are masked but do not reset the timer. Expiration of this programmable timer unmasks all REQ#s that are masked for this reason. The Retry Timer is programmable to 0 (disabled), 16, 32, or 64 PCICLKs.

If no other PCI masters are requesting the PCI Bus, all of the REQ#s masked for the timer are cleared and the timer is set to 0. Note that when there is a pending request that is internally masked, the PCEB does not park the CPU on the PCI Bus (i.e. PCI agent that uses CPUREQ#/CPUGNT# signal pair).

5.4.4 BUS LOCK MODE

As an option, the PCEB arbiter can be configured to run in Bus Lock Mode or Resource Lock Mode (default). The Bus Lock Mode is used to lock the entire PCI Bus. This may improve performance in some systems that frequently run quick read-modify-write cycles (i.e. access to the VGA frame buffer using the XCHG x86 instruction that automatically asserts the CPU LOCK# signal). Bus Lock Mode emulates the LOCK environment found in today's PC by restricting bus ownership when the PCI Bus is locked. While Bus Lock Mode improves performance in some systems, it may cause performance problems in other systems. With Bus Lock enabled, the arbiter recognizes a LOCK# being driven by an initiator and does not allow any other PCI initiator to be granted the PCI Bus until LOCK# and FRAME# are both negated, indicating the master released lock. When Bus Lock is disabled, the default resource lock mechanism is implemented (normal resource lock) and a higher priority PCI initiator could intervene between the cycles that are part of the locked sequence and run non-exclusive accesses to any unlocked resource.

CAUTION:

Bus Lock mode should not be used with non-GAT mode. If the system is initialized for both Bus Lock mode and non-GAT mode a deadlock situation might occur in the case where the first access to the locked device is a write instead of a read and the locked device has data in its internal posted write buffer. In GAT mode and/or Resource Lock mode this condition can not happen. If it is absolutely necessary to operate the system in the above mentioned combination of modes, then the posted write buffers of the device that might be involved in locked operations (typically semaphore in main memory) must be disabled.

5.4.5 MEMREQ#, FLSHREQ#, AND MEMACK# PROTOCOL

Before an EISA master or DMA can be granted the PCI Bus, it is necessary that all PCI system posted write buffers be flushed. Also, since the EISA-originated cycle could access memory on the Host/PCI Bridge, it is possible that the EISA master or DMA could be held in wait states (via EXRDY) waiting for the Host/PCI Bridge arbitration for longer than the 2.1 μs EISA/ISA specification. The PCEB has an optional mode called Guaranteed Access Time mode (GAT) that ensures that this timing specification is not violated. This is accomplished by delaying the EISA grant signal to the requesting master or DMA until the EISA Bus, PCI Bus, and the system memory bus are arbitrated for and owned.

The three sideband signals, MEMREQ#, FLSHREQ#, and MEMACK# are used to support the system Posted Write Buffer flushing and Guaranteed Access Time mechanism. The MEMACK# signal is the common acknowledge signal for both mechanisms. Note that, when MEMREQ# is asserted, FLSHREQ# is also asserted. Table 7 shows the relationship between MEMREQ# and FLSHREQ#.

Table 7. FLSHREQ# and MEMREQ#

FLSHREQ#	MEMREQ#	Meaning
1	1	Idle
0	1	Flush buffers pointing towards the PCI Bus to avoid EISA deadlock
1	0	Flush buffers pointing towards main memory for buffer coherency in APIC systems
0	0	GAT mode. Guarantees PCI Bus immediate access to main memory

5.4.5.1 Flushing System Posted Write Buffers

Once an EISA Bus owner (EISA/ISA master or the DMA) begins a cycle on the EISA Bus, the cycle can not be backed-off. It can only be held in wait states via EXRDY. In order to know the destination of EISA master cycles, the cycle needs to begin. After the cycle is started, no other device can intervene and gain ownership of the EISA Bus until the cycle is completed and arbitration is performed. A potential deadlock condition exists when an EISA-originated cycle to the PCI Bus forces a mandatory transaction to EISA, or when the PCI target is inaccessible due to an interacting event that also requires the EISA Bus. To avoid this potential deadlock, all PCI posted write buffers in the system must be disabled and flushed, before an EISA/ISA master or DMA can be granted the EISA Bus. The buffers must remain disabled while the EISA Bus is occupied. The following steps indicate the PCEB (and ESC) handshake for flushing the system posted write buffers.

1. When an EISA/ISA master, DMA or refresh logic requests the EISA Bus, the ESC component asserts EISAHOLD to the PCEB.
2. The PCEB completes the present cycle (and does not accept any new cycle) and gives the EISA Bus to the ESC by floating its EISA interface and asserting EISAHLDA. Before giving the bus to the ESC, the PCEB checks to see if it itself is locked as a PCI resource. It can not grant the EISA Bus as long as the PCEB is locked.

   At this point the PCEB's EISA-to-PCI Line Buffers and other system buffers (Host/PCI Bridge buffers) that are pointing to PCI are not yet flushed. The reason for this is that the ESC might request the bus in order to run a refresh cycle that does not require buffer flushing. That is not known until the EISA arbitration is frozen (after EISAHLDA is asserted).

   a. If the ESC needs to perform a refresh cycle, then it negates NMFLUSH# (an ESC-to-PCEB flush control signal). ESC drives the EISA Bus until it completes the refresh cycle and then gives the bus to the PCEB by negating EISAHOLD.

   b. If the ESC requested the EISA Bus on behalf of the EISA master, DMA or ISA master, then it asserts NMFLUSH# and tri-states the EISA Bus. The PCEB asserts the FLSHREQ# signal to the Host/PCI Bridge (and other bridges) to disable and flush posted write buffers.
3. When the Host/PCI Bridge completes its buffer disabling and flushing, it asserts MEMACK# to the PCEB. Other bridges in the system may also need to disable and flush their posted write buffers pointing towards PCI. This means that other devices may also generate MEMACK#. All of the MEMACK#s need to be "wire-OR'd". When the PCEB receives MEMACK# indicating that all posted write buffers have been flushed, it asserts NMFLUSH# to the ESC and the ESC gives the bus grant to the EISA device.
4. The PCEB continues to assert FLSHREQ# while the EISA/ISA master or DMA owns the EISA Bus. While FLSHREQ# is asserted the Host/PCI Bridge must keep its posted write buffers flushed.
5. MEMACK# should be driven inactive as soon as possible by the Host/PCI Bridge and other bridges after FLSHREQ# is negated. The PCEB waits until it detects MEMACK# negated before it can generate another FLSHREQ#.

5.4.5.2 Guaranteed Access Time Mode

When the PCEB's Guaranteed Access Time Mode is enabled (via the ARBCON Register), MEMREQ# and MEMACK# are used to guarantee that the ISA 2.1μs CHRDY specification is not violated. Note that EISA's 2.5 μs maximum negation time of the EXRDY signal is a subset of the ISA requirement. Thus, 2.1 μs satisfies both bus requirements.

When an **EISA/ISA master or DMA slave** requests the EISA Bus (MREQ# or DREQ# active), the EISA Bus, the PCI Bus, and the memory bus must be arbitrated for and all three must be owned before the EISA/ISA master or DMA is granted the EISA Bus. The following lists the sequence of events:

1. An EISA/ISA master, DMA, or refresh logic requests the EISA Bus. The ESC asserts EISAHOLD signal to the PCEB.
2. The PCEB completes the present cycle (i.e. does not accept any new cycle) and gives the bus to the ESC by floating its EISA interface and asserting EISAHLDA. Before giving the bus to the ESC, the PCEB checks to see if it is locked as a PCI resource. It can not grant the EISA Bus as long as the PCEB is locked.

At this point, the PCEB's EISA-to-PCI Line Buffers and other system buffers (e.g., Host/PCI Bridge buffers) that are pointing to the PCI Bus are not flushed. The reason is that the ESC might request the bus to run a refresh cycle that does not require buffer flushing. This is not known until the EISA arbitration is frozen (after EISAHLDA is asserted).

3. Depending on whether the pending cycle is a refresh, the ESC initiates one of the following two actions:
   a. If the ESC needs to perform a refresh cycle, then it asserts NMFLUSH# (an ESC-to-PCEB flush control signal). The ESC drives the EISA Bus until it completes the refresh cycle and then gives the bus to the PCEB by negating EISAHOLD.
   b. If the ESC requested the EISA Bus on behalf of the EISA master, DMA or ISA master, then it asserts NMFLUSH# and tri-states the EISA Bus. If the PCEB is programmed in GAT (Guaranteed Access Time mode), the MEMREQ# and FLSHREQ# signals are asserted simultaneously to indicate request for direct access to main memory and a request to flush the system's posted write buffers pointing towards the PCI (including the PCEB's internal buffers). These requirements are necessary to insure that once the PCI and EISA Buses are dedicated to the PCEB, the cycle generated by the PCEB will not require the PCI or EISA Buses, thus creating a deadlock. MEMREQ# and FLSHREQ# are asserted as long as the EISA/ISA master or DMA owns the EISA Bus.
4. Once the Host/PCI Bridge has disabled and flushed its posted write buffers, and the memory bus is dedicated to the PCI interface, it asserts MEMACK#. Other bridges in the system may also need to disable and flush their posted write buffers pointing towards PCI due to the FLSHREQ# signal. This means that other devices may also generate a MEMACK#. All of the MEMACK#s need to be "wire-OR'd". When the PCEB receives MEMACK#, it assumes that all of the critical posted write buffers in the system have been flushed and that the PCEB has direct access to main memory, located behind the Host/PCI Bridge.
5. When MEMACK# is asserted by the PCEB, it will request the PCI Bus (internal PCEBREQ# signal). Before requesting the PCI Bus, the PCEB checks to see that the PCI Bus does not have an active lock. The PCI Bus is granted to the PCEB when it wins the bus through the normal arbitration mechanism. Once the PCEB is granted the PCI Bus (internalPCEBGNT#), the PCEB checks to see if PLOCK# is negated before it grants the EISA Bus. If the PCI Bus is locked when the PCEB is granted the PCI Bus, the PCEB releases the REQ# signal and waits until the PLOCK# is negated before asserting REQ# again. Once the PCEB owns the PCI Bus (internal PCEBGNT#), and the MEMACK# and MEMREQ# signals are asserted, the PCI arbiter will not grant the PCI Bus to any other PCI master except the PCEB until the PCEB releases its PCI REQ# line.
6. When the PCEB is granted the PCI Bus (internal PCEBGNT#) and LOCK# is inactive, it asserts NMFLUSH# to the ESC and the ESC gives the bus grant to the EISA device.

7. When the EISA Bus is no longer owned by an EISA master or DMA, the PCEB negates MEMREQ# and FLSHREQ# and the PCI request signal (internal PCEBREQ#. The negation of MEMREQ# and FLSHREQ# indicates that direct access to the resource behind the bridge is no longer needed and that the posted write buffers may be enabled. Note that MEMACK# should be driven inactive as soon as possible by the Host/PCI Bridge and other bridges after MEMREQ# is negated. The PCEB waits until it detects MEMACK# negated before it can generate another MEMREQ# or FLSHREQ#.

The use of MEMREQ#, FLSHREQ#, and MEMACK# does not guarantee GAT mode functionality with ISA masters that don't acknowledge CHRDY. These signals just guarantee the CHRDY inactive specification.

5.4.5.3 Interrupt Synchronization-Buffer Flushing

The ESC contains the system interrupt controller consisting of an 8259 compatible interrupt controller and an I/O APIC. For the 8259 compatible interrupt controller, the PCEB/ESC chip set is the default destination of the PCI interrupt acknowledge cycles. Interrupts in the system are commonly used as a synchronization mechanism. If interrupts are used by the EISA agents to notify the Host CPU that data is written to main memory, then posted data buffers must be flushed before the vector is returned during the interrupt acknowledge sequence. The PCEB handles this transparently to the rest of the system hardware/software. It retries the PCI interrupt acknowledge cycles and flushes the PCEB Line Buffers, if necessary.

The Advanced Programmable Interrupt Controller (APIC) uses a private message passing bus to send interrupt information to the companion APIC(s) residing at the host CPU(s). To support interrupts as a synchronization mechanism, system buffer coherency must be guaranteed before interrupts can be processed. With ESC's interrupt controller operating in APIC mode the PCEB and ESC use the PCEB/ESC interchip signal AFLUSH# to maintain system buffer coherency.

5.4.6 BUS PARKING

PCI Bus parking can be enabled/disabled via the ARBCON Register. Parking is only allowed for the device that is connected to CPUREQ# (i.e. the Host/PCI Bridge). REQ[3:0]#, and the internal PCEBREQ# are not allowed to park on the PCI Bus. When bus parking is enabled, CPUGNT# is asserted when no other agent is currently using or requesting the bus. This achieves the minimum PCI arbitration latency possible.

Arbitration Latency

Parked: 0 PCICLKs for parked agent, 2 PCICLKs for all other.

Not Parked: 1 PCICLK for all agents.

Upon assertion of CPUGNT# due to bus parking enabled and the PCI Bus idle, the CPU (i.e., parked agent) must ensure AD[31:0], C/BE[3:0]#, and (one PCICLK later) PAR are driven. If bus parking is disabled, then the PCEB drives these signals when the bus is idle.

5.4.7 PCI ARBITRATION AND PCEB/ESC EISA OWNERSHIP EXCHANGE

There are two aspects of PCEB/ESC EISA Bus ownership exchange that are explained in this section. They are related to GAT mode and RESUME/RETRY operations.

The PCEB is the default owner of the EISA Bus. When control of the EISA Bus is given to the ESC, all PCI operations targeted to the EISA subsystem (including the PCEB) are retried. Retry causes assertion of the PEREQ#/INTA# signal with PEREQ# semantics. In this way, the PCEB indicates to the ESC that it needs to obtain ownership of the EISA Bus.

5.4.7.1 GAT Mode And PEREQ# Signaling

In GAT mode, the PCEB owns the PCI Bus on behalf of the EISA master and other PCI agents (e.g., the Host/PCI Bridge) can not generate PCI cycles. Therefore, the PCEB never generates a back-off (i.e. retry), as long as the EISA Bus is controlled by the ESC. This might cause starvation of the PCI agents (including the Host/PCI Bridge i.e., CPU) even in the case of a moderately loaded EISA subsystem. The solution is that PEREQ#, in the GAT mode, is generated when any of the PCI Bus request signals are asserted. For particular Host/PCI Bridge designs (e.g. PCMC) this will be not be an adequate solution since their PCI request can be activated only based on the CPU generated cycle directed to PCI. This will not be possible since the Host Bus (CPU bus) in the GAT mode is controlled by the Host/PCI Bridge and not by the CPU. The solution to this type of design is to generate PEREQ# immediately after entering the GAT mode. This feature is controlled via ARBCON Register (bit 7).

5.4.7.2 PCI Retry And EISA Latency Timer (ELT) Mechanism

When a PCI cycle is retried by the PCEB (in non-GAT mode) because the EISA Bus is controlled by the ESC (EISAHLDA asserted), an internal flag is set for the corresponding PCI master. This flag masks the request of a particular master until the PCEB acquires the ownership of EISA and the RESUME condition clears the flag. If the PCI master, which is now unmasked, does not acquire the ownership of the PCI Bus within the time period before ESC asserts EISAHOLD again, the EISA Bus can be surrendered to the ESC. Unmasked masters will eventually gain the access to the PCI Bus but the EISA Bus will not be available and the master will be retried again. This scenario can be repeated multiple times with one or more PCI masters and starvation will occur.

To solve this situation, the PCEB arbitration logic incorporates an EISA Latency Timer mechanism. This mechanism is based on the programmable timer that is started each time that the ESC requires the bus (EISAHOLD asserted) and there is a PCI agent that has been previously retried because of activity on the EISA Bus. As soon as the ELT timer expires, any PCI cycle which is currently in progress is retried and the EISA Bus is given back to the ESC after the current PCI-to-EISA transaction completes. If all the PCI requesters, masked because of EISAHLDA, are serviced before the ELT timer expires, the EISA Bus is immediately surrendered to the ESC. The ELT provides a minimum time slice for PCI masters to access the EISA bus even if EISA masters, ISA masters or DMA devices are attempting to acquire the EISA bus.

Generally, the ELT is set to a larger value if latency sensitive PCI masters which typically access EISA are present in the system. Larger ELT values, however, do increase the worst case latency for EISA devices which typically access devices on PCI (e.g. main memory).

The EISA Latency Timer (ELT) is controlled by the ELTCR Register. The value written into ELTCR is system dependent. It is typically between 1 and 3 μs.

6.0 DATA BUFFERING

The PCEB contains data buffers (Figure 24) to isolate the PCI Bus from the EISA Bus and to provide concurrent EISA and PCI Bus operations and APIC operations. The Line Buffers are used for EISA-to-PCI memory reads and writes. A control bit in the EPMRA Registers permis the Line Buffers to be enabled (accesses are buffered) or disabled (accesses are non-buffered). Non-buffered accesses use the bypass path. Note that PCI and EISA I/O read/write cycles and PCI configuration cycles are always non-buffered and use the bypass path.

When data is temporarily stored in the buffers between the EISA Bus and PCI Bus, there are potential data coherency issues. The PCEB guarantees data coherency by intervening when data coherency could be lost and either flushing or invalidating the buffered data, as appropriate.

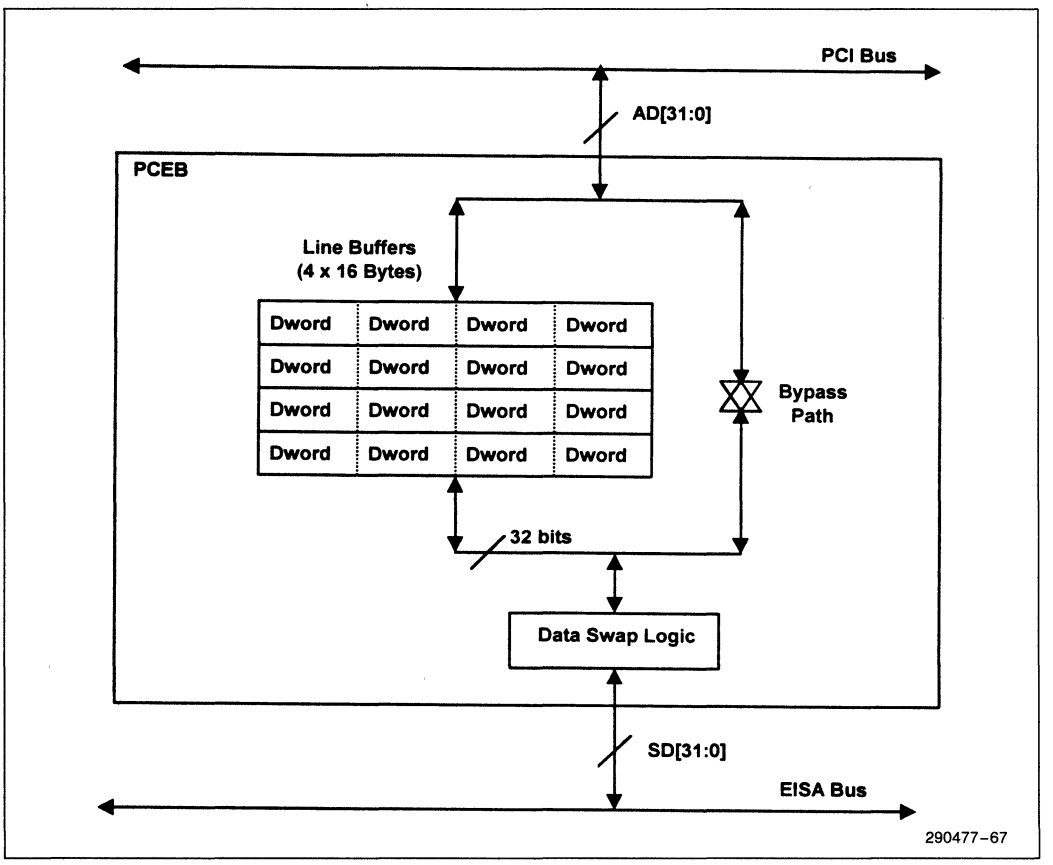


Figure 24. PCEB Data Buffers

6.1 Line Buffers

The PCEB contains four Line Buffers that are each four Dwords wide (16 bytes). The Line Buffers are bi-directional and are used by the EISA/ISA master and DMA to assemble/disassemble data. The data in each Line Buffer is aligned on 16 byte boundaries. When data is placed in one of the Line Buffers, the PCEB maintains the corresponding 16-byte boundary address until the data in the line is transferred to its destination or invalidated.

The Line Buffers can be enabled/disabled by writing to the PCICON Register. In addition, when the Line Buffers are enabled via the PCICON Register, buffering for accesses to the four programmable EISA-to-PCI memory regions (Region [4:1]) can be selectively disabled via the EPMRA Register.

During buffer operations, the four Line Buffers, collectively, are either in a write state or in a read state. These states are described in the following sections.

6.1.1 WRITE STATE

If a Line Buffer contains valid write data, it is in a *write state*. In the write state, data from the EISA/ISA master or DMA is posted in the Line Buffers. Posting means that the write operation on the EISA Bus completes when the data is latched in the buffer. The EISA master does not have to wait for the write to complete to its destination (memory on the PCI Bus). Posting permits the EISA Bus cycle to complete in a minimum time and permits concurrent EISA and PCI Bus operations. During posting, data accumulates in the Line Buffer until it is flushed (written to PCI memory) over the PCI Bus. A Line Buffer is scheduled for flushing by the PCEB when:

- the line becomes full.
- a subsequent write is a line miss (not within the current line boundary address range).
- the write is to an address of a lower Dword than the previous write. Note that writes to lower addresses within the same Dword do not cause a flush. Note also, that if two (or more) consecutive EISA Bus cycles are writes to the same Dword (i.e., the same byte or word locations within the Dword, or the same Dword for Dword writes) , the accessed buffer data is overwritten. However, if any of the flush conditions described in this list occur between the writes, the line is flushed before the next write and data is not overwritten.
- the last address location in the Line Buffer is accessed.
- a subsequent cycle is a read.
- the EISA Bus changes ownership.
- an interrupt acknowledge cycle is encountered.
- the ESC performs an EISA refresh cycle.
- the ESC's I/O APIC receives an interrupt request.

When a line is scheduled for flushing, the PCEB begins arbitration for the PCI Bus. If more than one line is scheduled to be flushed, the Line Buffers are flushed in a "first scheduled, first to be flushed" order. If the line to be flushed contains valid data in only one Dword, the PCEB uses a single data transfer cycle on the PCI Bus. Otherwise, flushing operations use burst transfers.

During flushing, write data within a Line Buffer is packetized into Dword quantities, when possible, for a burst transfer over the 32-bit PCI Bus. Packetizing occurs at two levels - Dwords within a line and bytes/words within a Dword. When a Line Buffer is flushed, all of the valid Dwords within the line are packetized into a single PCI burst write cycle. In addition, all valid data bytes within a Dword boundary are packetized into a single data phase of the burst cycle. Packetizing reduces the PCI arbitration latency and increases the effective PCI Bus bandwidth. When multiple Line Buffers are schedule for flushing, each Line Buffer is packetized separately. Packetizing across Line Buffer boundaries is not permitted.

During flushing, strong ordering is preserved at the Dword level (i.e., the Dwords are flushed to PCI memory in the same order that they were written into the Line Buffer). Note, however, that strong ordering is not preserved at the byte or word levels (i.e., even if byte or word transfers were used by the EISA/ISA master or DMA to sequentially write to a Dword within a Line Buffer, all of the bytes in the resulting Dword boundary are simultaneously flushed to PCI memory).

Because strong ordering is not preserved within a Dword boundary, care should be used when accessing memory-mapped I/O devices. If the order of byte or word writes to a memory-mapped I/O device needs to be preserved, buffered accesses should not be used. By locating memory-mapped I/O devices in the four programmable EISA-to-PCI memory regions, buffering to these devices can be selectively disabled.

6.1.2 READ STATE

If a Line Buffer contains valid read data, it is in a *read state*. Read data is placed in the Line Buffer by two PCEB mechanisms - fetching and prefetching. Data is placed in the Line Buffer on demand (fetching) when the data is requested by a read operation from the EISA/ISA master or DMA. The PCEB also prefetches data that has not been explicitly requested but is anticipated to be requested. Once in the Line Buffer, data is either read by the EISA/ISA master or DMA (and then invalidated) or invalidated without being read. Read data is invalidated when:

- data in the Line Buffer is read (transferred to the EISA/ISA master or DMA). This prevents reading of the same data more than once.
- a subsequent read is a line miss (not to the previously accessed Line Buffer). Valid data in the current Line Buffer is invalidated. If a new line had been prefetched during access to the current line, data in the prefetched line is not invalidated, unless the access also misses this line. In this case, the data in the prefetched line is invalidated.
- a subsequent cycle is a write. Data in all Line Buffers are invalidated.

If the requested data is in the Line Buffer, a line hit occurs and the PCEB transfers the data to the EISA/ISA master or DMA (and invalidates the hit data in the buffer). If EISA Bus reads hit two consecutive line addresses, the PCEB prefetches the next sequential line of data from PCI memory (using a PCI Bus burst transfer). This prefetch occurs concurrently with EISA Bus reads of data in the already fetched Line Buffer. If consecutive addresses are not accessed, the PCEB does not prefetch the next line.

A line miss occurs if the requested data is not in the Line Buffer. If a line miss occurs, the PCEB invalidates data in the missed Line Buffer. If the requested data is in a prefetched line, the read is serviced. If a line was not prefetched or the read missed the prefetched line, the PCEB invalidates any prefetched data and fetches the Dword containing the requested data. During this fetch, the PCEB holds off the EISA/ISA master or DMA with wait states (by negating EXRDY). When the requested data is in the Line Buffer, it is transferred to the EISA Bus. Simultaneously with the EISA Bus transfer, the PCEB prefetches the rest of the line data (Dwords whose addresses are within the line and above the Dword address of the requested data). The Dword containing the requested data and the rest of the Dwords in the line (located at higher addresses) are fetched from PCI memory using a burst transfer, unless the requested data is in the last Dword of a line. In this case, a single cycle read occurs on the PCI Bus.

For purposes of data read operations, all four 4-Dword buffers are used to form two 8-Dword lines (32 bytes each). There are only tow address pointers, one for each line. Fetching fractions of a line is accomplished as described above (i.e., starting from the first requested Dword).

The MSBURST# input signal is used to supplement control of the prefetch sequence. The MSBURST# signal is activated only when an EISA master desires to do burst transfers to access sequential data (although this is not an absolute EISA rule, i.e., theoretically the data can be non-sequential after an EISA slave indicates its ability via SLBURST#). This will occur during the first data transfer.

The Line Buffer control logic dynamically switches between two prefetch modes—Half Line Prefetch (16 bytes fetch) and Full Line Prefetch (32 bytes fetch)

The prefetch control logic has implemented a Sequential Access Flag which is cleared before the initial prefetch. Initial prefetch (first data fetch) starts in the Half Line Prefetch mode and is extended to Full Line Prefetch mode immediately after MSBURST# is sampled asserted at which time the Sequential Access Flag is automatically set (this is done on-the-fly during the first line fetch). If after the initial prefetch the Sequential Access Flag has not been set (MSBURST# remained not asserted) and the control logic recognizes two consecutive hits (in incrementally sequential Dwords including the first one which is originally requested), the Sequential Access Flag is set and the prefetch control logic switches to Full Line Prefetch mode. An additional 32-byte line (or fraction depending on alignment) will be fetched.

When the Sequential Access Flag is set, prefetching is accomplished using the Full Line Prefetch mode. Each time a line buffer (32 bytes) is available, an additional line will be fetched as long as the Sequential Access Flag remains set.

When out-of-order access is recognized within the prefetched data or a miss occurs when there is valid fetched data, the Sequential Access Flag is cleared and the prefetch mode changes to Half Line Prefetch. Also, the Sequential Access Flag is cleared when MSBURST# transitions from active to inactive.

When the Sequential Access Flag is not asserted, the prefetch control logic operates in Half Line Prefetch mode during which only 16 bytes of data is fetched at a time. The same test for sequential access is repeated, and if access is recognized, the Sequential Access Flag is set and the control switches to Full Line Prefetch mode.

6.2 Buffer Management Summary

Table 8 shows Line Buffer for different cycles. Note that the first three columns together define the cycles that may trigger buffer activity.

Table 8. Buffer Management Summary

Master (Origin)	Cycle Type	Slave (Destination)	Line Buffer Data in Write State	Line Buffer Data in Read State
PCI	Memory Read	EISA	Flush	No Action
PCI	Memory Write	EISA	No Action	Invalidate
PCI	I/O Read	EISA	Flush	No Action
PCI	I/O Write	EISA	No Action	Invalidate
PCI	Interrupt Acknowledge	PCEB/ESC	Flush	No Action
PCI	Configuration Cycle	PCEB Registers	No Action	No Action
PCI	Memory Read/Write	PCI	No Action	No Action
PCI	I/O Read/Write	PCI	No Action	No Action
EISA	Bus Ownership Change	—	Flush	No Action
EISA	Memory Read/Write	EISA	No Action	No Action

Table 8. Buffer Management Summary (Continued)

Master (Origin)	Cycle Type	Slave (Destination)	Line Buffer Data in Write State	Line Buffer Data in Read State
EISA	Memory Read/Write	PCI	(Note 1)	(Note 1)
EISA	I/O Read/Write	EISA	No Action	No Action
EISA	I/O Read/Write	PCI	Flush	Invalidate
ESC's I/O APIC	APIC Bus Message Transfer	Local APIC	Flush	No Action

NOTES:
1. Change from write to read operation or from read to write causes the Line Buffers to be flush or invalidate, respectively.
2. LOCKed cycles (both from PCI and EISA) are not buffered within the PCEB. They are processed using the bypass path.

7.0 EISA INTERFACE

The PCEB provides a fully EISA Bus compatible master and slave interface. This interface provides address and data signal drive capability for eight EISA slots and supports the following types of cycles:

- PCI-initiated memory and I/O read/write accesses to an EISA/ISA device.
- EISA/ISA/DMA-initiated memory and I/O read/write accesses to a PCI device (i.e. via the Line Buffers, if necessary).
- Accesses contained within the EISA Bus (only data swap buffers involved).

For transfers between the EISA Bus and PCI Bus, the PCEB translates the bus protocols. For PCI master-initiated cycles to the EISA Bus, the PCEB is a slave on the PCI Bus and a master on the EISA Bus. For EISA master-initiated cycles to the PCI Bus, the PCEB is a slave on the EISA Bus and a master on the PCI Bus.

NOTE:

1. The PCEB is not involved in refresh cycles on the EISA Bus. When the REFRESH# signal is asserted, the PCEB disables EISA Bus address decoding.
2. Wait state generation on the EISA Bus is performed by the ESC. ISA memory slaves (8 or 16 bits) and ISA I/O slaves can shorten their default or standard cycles by asserting the NOWS# signal line. It is the responsibility of the ESC to shorten these cycles when NOWS# is asserted. Note that ISA I/O 16-bit devices can shorten their cycles by asserting NOWS#. If CHRDY and NOWS# are driven low during the same cycle, NOWS# will not be used and wait states are added as a function of CHRDY. For more details on the wait state generation and the NOWS# signal, refer to the ESC data sheet.
3. All locked PCI cycles (PLOCK# asserted) destined to the EISA Bus are converted to EISA locked cycles using the LOCK# signal protocol. The PCEB is a locked resource during these cycles and maintains control of the EISA Bus until the locked PCI sequence is complete.
4. All locked EISA cycles (LOCK# asserted) destined to PCI are converted to PCI locked cycles using the PLOCK# signal protocol. The PLOCK# signal remains active as long as the EISA LOCK# signal is asserted.

5. The PCEB contains EISA data swap buffers for data size translations between mismatched PCI Bus and EISA Bus transfers and between mismatched devices contained on the EISA Bus. Thus, if data size translation is needed, the PCEB is involved in cycles contained to the EISA Bus, even if the PCEB is neither the master or slave. For data size translation operations, see Section 8.0, EISA Data Swap Buffers.
6. For ISA master cycles to PCI memory or I/O, the ESC translates the ISA signals to EISA signals. The PCEB, as an EISA slave, forwards the cycle to the PCI Bus.
7. For ISA master cycles to ISA/EISA slaves, the PCEB is not involved, except when the cycle requires data size translations. See the ESC data sheet for cycles that are contained within the EISA Bus (i.e., EISA-to-EISA, EISA-to-ISA, ISA-to-ISA, and ISA-to-EISA device cycles).
8. In this section, LA[31:24]# and LA[23:2] are collectively referred to as LA[31:2].

7.1 PCEB As An EISA Master

The PCEB is an EISA master for PCI-initiated cycles targeted to the EISA Bus. When the PCEB decodes the PCI cycle as a cycle destined to the EISA Bus (via subtractive or negative (82374SB only) decoding, as described in Section 4.0, Address Decoding), the PCEB becomes a slave on the PCI Bus. If the PCEB owns the EISA Bus, the cycle is forwarded to the EISA/ISA device. If the PCEB does not own the EISA Bus (EISAHOLDA is asserted to the ESC), the PCI master is retried and the PCEB issues an EISA Bus request to the ESC. For PCI-to-EISA I/O and memory read/write accesses, the PCEB runs standard EISA Bus cycles.

When cycles are forwarded to a matched EISA/ISA slave, the PCEB is the EISA master and controls the transfer until the cycle is terminated. For mismatched cycles to an EISA/ISA slave, the PCEB backs off the EISA Bus as described in Section 7.1.3, Back-Off Cycle.

7.1.1 STANDARD EISA MEMORY AND I/O READ/WRITE CYCLES

The standard EISA cycle completes one transfer each two BCLK periods (zero wait states). The standard EISA memory or I/O cycle begins when the PCEB presents a valid address on LA[31:2] and drives M/IO# high for a memory cycle and low for an I/O cycle. The address can become valid at the end of the previous cycle to allow address pipelining. The EISA slave decodes the address and asserts the appropriate signals to indicate the type of slave and whether it can perform any special timings. The slave asserts EX32# or EX16# to indicate support of EISA cycles.

For extended cycles, the EISA slave introduces wait states using the EXRDY signal. Wait states allow a slower slave to get ready to complete the transfer. The slave negates EXRDY after it decodes a valid address and samples START# asserted. The slave may hold EXRDY negated for a maximum of 2.5 μs to complete a transfer, and must release EXRDY synchronous to the falling edge of BCLK to allow a cycle to complete. Note that the PCEB, as an EISA master, never introduces wait states.

Figure 25 shows three data transfer cycles between an EISA master and an EISA slave. The first transfer is an extended transfer (EXRDY negated), followed by two standard cycles. For PCI cycles that are forwarded to the EISA Bus, the PCEB is the EISA master. The PCEB asserts START# to indicate the start of a cycle. The PCEB also drives W/R# to indicate a read or write cycle and BE[3:0]# to indicate the active bytes. The LA[31:2] and the BE[3:0] remain valid until after the negation of START#. A slave that needs to latch the address does so on the trailing edge of START#.

The ESC asserts CMD# simultaneously with the negation of START# to control data transfer to or from the slave. If a read cycle is being performed, the slave presents the requested data when CMD# is asserted and holds it valid until CMD# is negated by the ESC. For a write cycle, the PCEB presents the data prior to the assertion of CMD# and the slave latches it on or before the trailing edge of CMD#.

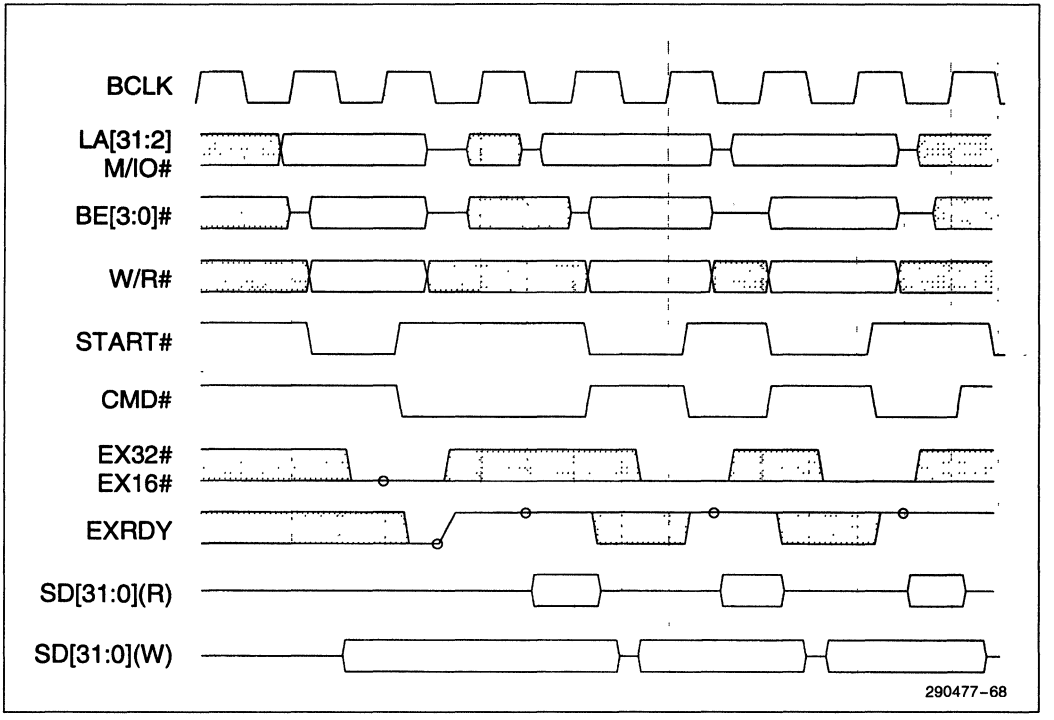


Figure 25. EISA Memory and I/O Read/Write Cycle (One Extended and Two Standard Cycles)

7.1.2 EISA BACK-OFF CYCLE

For mismatched cycles to an EISA/ISA slave, the PCEB, as a master, backs off the EISA Bus by floating the START#, BE[3:0]# and SD[31:0] signals one and half BCLKs after START# has been asserted. The ESC controls the EISA Bus for the duration of the cycle. This allows the ESC to perform data translation, if necessary. At the end of the cycle, the ESC transfers control back to the PCEB by asserting EX16# and EX32# on the falling edge of BCLK, before the rising edge of BCLK that the last CMD# is negated. Refer to the ESC data sheet for further details on master back-off and the cycle transfer control operations.

Figure 26 shows an example of a back-off sequence during a 32-bit EISA master to 16-bit EISA slave Dword read and write operation. The thick lines indicate the change of control between the master and the ESC.

PCEB Reading From a 16-bit EISA Slave

As a 32-bit EISA master, the PCEB begins by placing the address on LA[31:2] and driving M/IO#. The 16-bit EISA slave decodes the address and asserts EX16#. The PCEB asserts START#, W/R#, and BE[3:0]#. The ESC samples EX32# and EX16# on the rising edge of BCLK following the assertion of START# and asserts CMD#. At the same time, the PCEB negates START# and samples EX32#. When EX32# is sampled negated, the PCEB floats START# and BE[3:0]#. Note that, the PCEB continues to drive a valid address on LA[31:0].

The ESC negates CMD# after one BCLK period unless the slave adds wait states (negates EXRDY). The ESC latches SD[15:0] into the PCEB's data swap buffer on the trailing edge of CMD#. The ESC controls the PCEB data swap buffers via the PCEB/ESC Interface. The ESC then asserts START# and presents BE[3:0] (upper word enabled). The ESC negates START# and asserts CMD#. The slave latches the address on the trailing edge of START# and presents data on SD[15:0]. The ESC negates CMD# after one BCLK, unless the slave negates EXRDY. The ESC latches SD[15:0] into the PCEB data swap buffers on the trailing edge of CMD# and instructs the PCEB data swap buffer to copy D[15:0] to D[31:0] and asserts EX32#. Note that, since the transfer is intended for the PCEB, the data is not re-driven back out onto the EISA Bus. The ESC floats the START# and BE[3:0]#. The PCEB regains control of the EISA Bus after sampling EX32# and EX16# asserted.

PCEB Writing To a 16-bit EISA Slave

As a 32-bit EISA master, the PCEB begins by placing the address on LA[31:2] and driving M/IO#. The 16-bit EISA slave decodes the address and asserts EX16#. The PCEB asserts START#, W/R#, BE[3:0]#, and SD[31:0]. The ESC samples EX32# and EX16# on the rising edge of BCLK following the assertion of START# and asserts CMD#. At the same time, the PCEB negates START# and samples EX32#. When EX32# is sampled negated, the PCEB floats START#, SD[31:0], and BE[3:0]#. The data is latched in the PCEB's data swap buffers. Note that the PCEB continues to drive a valid address on LA[31:2].

The ESC instructs the PCEB to drive the data out on SD[31:0] and asserts CMD# after sampling EX32# negated. The slave may sample SD[15:0] while CMD# is asserted. The ESC negates CMD# after one BCLK, unless the slave adds wait states (negates EXRDY). The ESC then presents BE[3:0] (upper word enabled) and asserts START#. The ESC instructs the PCEB to copy SD[31:0] to SD[15:0], negates START# and asserts CMD#. The ESC negates CMD# after one BCLK, unless the slave negates EXRDY. The slave latches the address on the trailing edge of START# and samples SD[15:0] on the trailing edge of CMD#. The ESC returns control of the EISA Bus to the PCEB by floating BE[3:0]# and START#, then asserting EX32#. The PCEB samples EX32# and EX16# asserted on the rising edge of BCLK.

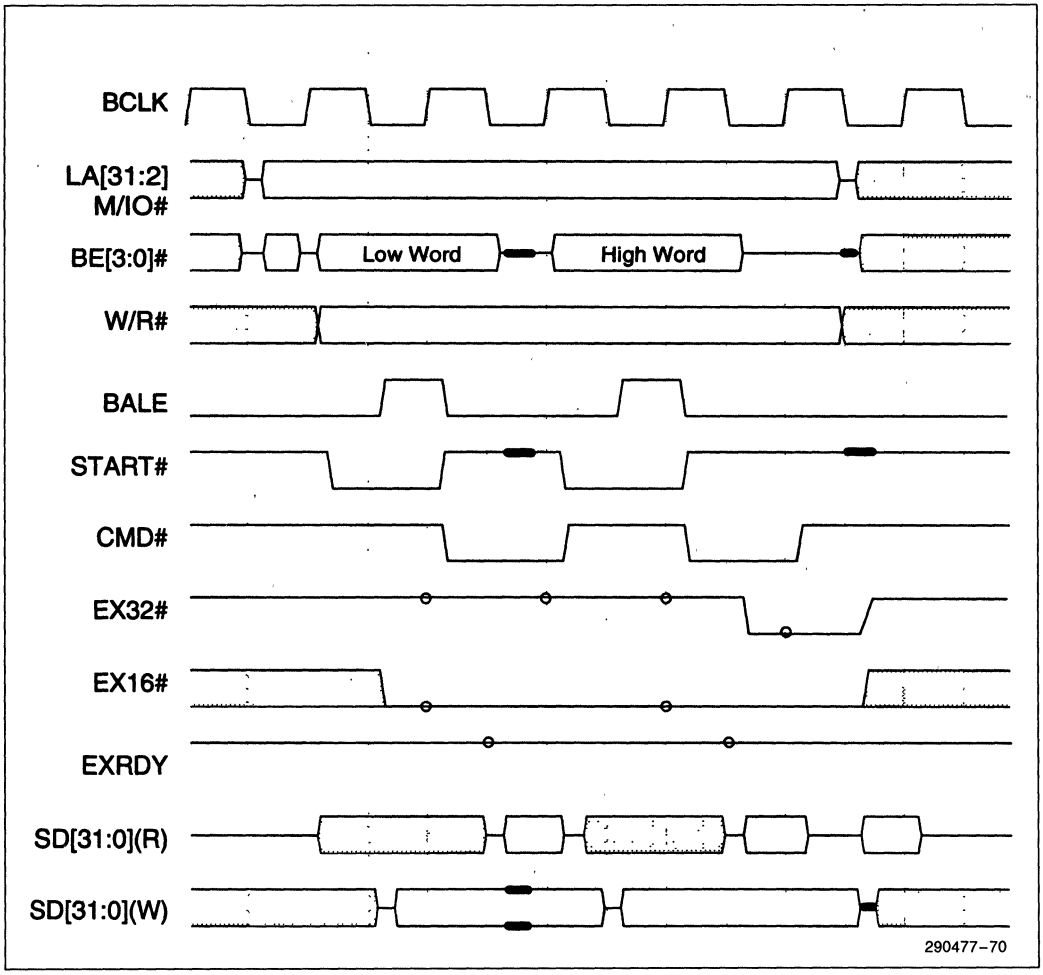


Figure 26. EISA Back-Off Cycle

7.2 PCEB As An EISA Slave

The PCEB is an EISA slave for EISA/ISA/DMA-initiated cycles targeted to the PCI Bus. If the PCEB positively decodes the address (access to one of the EISA programmed main memory segments or access to one of the programmable EISA-to-PCI memory or I/O regions), the PCEB becomes an EISA slave and the cycle is forwarded to the PCI Bus. If the PCEB does not positively decode the address, the cycle is contained to the EISA Bus. For cycles contained to the EISA Bus (i.e., EISA-to-EISA, EISA-to-ISA, ISA-to-ISA, and ISA-to-EISA device cycles), the PCEB is only involved when data size translation is needed.

The PCEB responds as a 32-bit EISA slave. If the EISA master size is not 32 bits, the cycle is a mismatch and invokes data size translation. For details on data size translation, refer to Section 8.0, EISA Data Swap Buffers.

All EISA master memory read cycles to PCI memory start as extended cycles, unless the cycle triggers a read hit to one of the four Line Buffers. If the data is available in the Line Buffers, the PCEB supplies the data to the EISA master without adding wait states. Otherwise, the cycle is extended (wait states added via EXRDY) until the data is available. Note that for non-buffered accesses, the EISA cycle is always extended until data is available from the PCI Bus.

If the Line Buffers are enabled, write cycles to PCI memory are posted in the Line Buffers. If the write can be immediately posted, wait states are not generated on the EISA Bus. Otherwise, the cycle is extended (via wait states) until the data can be posted. Note that writes can be posted to available Line Buffers concurrently with other Line Buffers being flushed to the PCI Bus.

All EISA master I/O read/write accesses to PCI I/O space are non-buffered and always start as extended cycles. Data transfer on the EISA Bus occurs when the requested data is available from the PCI Bus.

For mismatched cycles to the PCEB, the EISA/ISA master backs off the EISA Bus as described in Section 7.1.3, Back-Off Cycle.

7.2.1 EISA MEMORY AND I/O READ/WRITE CYCLES

The standard EISA cycle completes one transfer each two BCLK periods (zero wait states). The standard EISA memory or I/O cycle begins with the EISA master presenting a valid address on LA[31:2] and driving M/IO# high for a memory cycle and low for an I/O cycle. The address can become valid at the end of the previous cycle to allow address pipelining. When the PCEB positively decodes the address, it asserts EX32# to indicate 32-bit support. For memory cycles, the PCEB also asserts SLBURST# to indicate support for burst transfers.

For extended cycles, the PCEB introduces wait states using the EXRDY signal. The PCEB may hold EXRDY negated for a maximum of 2.5 μs to complete a transfer, and releases EXRDY synchronous to the falling edge of BCLK to allow a cycle to complete.

Figure 27 shows three data transfers between an EISA master and an EISA slave. The first transfer is an extended transfer (EXRDY negated), followed by two standard cycles. For EISA cycles that are forwarded to the PCI Bus, the PCEB is an EISA slave. The EISA master asserts START# to indicate the start of a cycle. The EISA master also drives W/R# to indicate a read or write cycle and BE[3:0]# to indicate the active bytes. The LA[31:2] and the BE[3:0] remain valid until after the negation of START#. The PCEB latches the address on the trailing edge of START#.

The ESC asserts CMD# simultaneously with the negation of START# to control data transfer to or from the PCEB. If a read cycle is being performed, the PCEB presents the requested data when CMD# is asserted and holds it valid until CMD# is negated by the ESC. For a write cycle, the EISA master must present the data prior to the assertion of CMD# and the PCEB latches it on the trailing edge of CMD#.

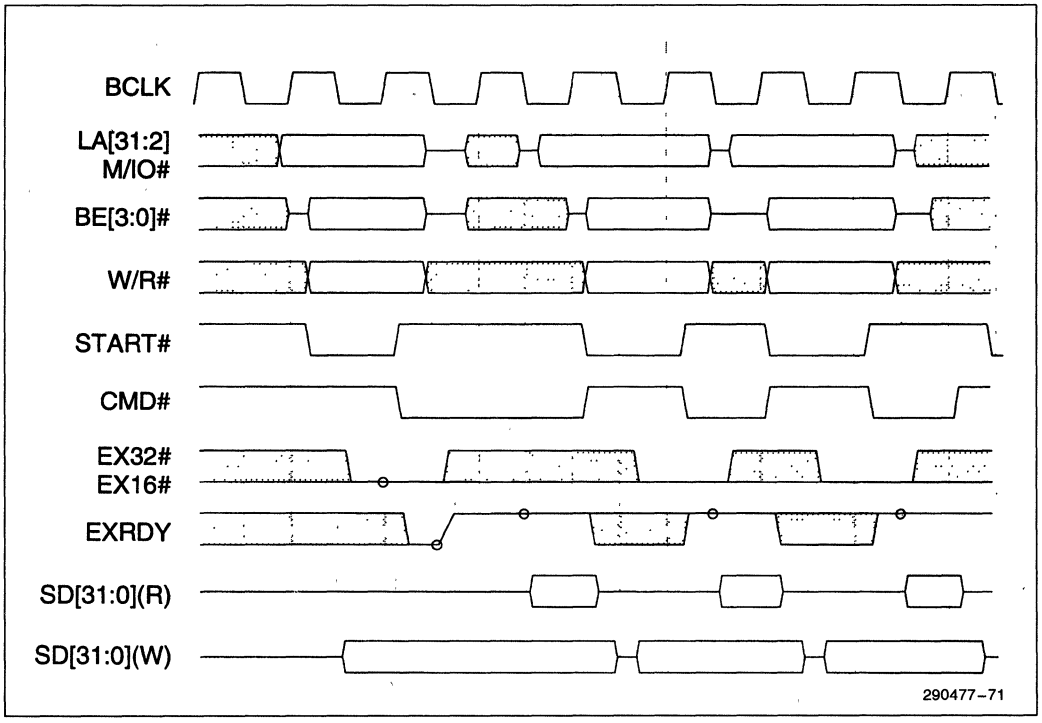


Figure 27. EISA Memory and I/O Read/Write Cycles (One Extended and Two Standard Cycles)

7.2.2 EISA MEMORY BURST CYCLES

The EISA burst cycles permit a continuous sequence of read or write cycles in zero wait-states (1 BCLK per transfer). A burst transfer is either all reads or all writes. Mixed cycles are not allowed. As an EISA slave, the PCEB supports burst memory reads and burst memory writes from/to its Line Buffers. Figure 28 shows an example of a burst sequence for both memory reads and writes on the EISA Bus. During the particular burst sequence, five data transfers occur with a wait state added on the third data transfer.

The first transfer in a burst transfer begins like the standard cycle described above. The EISA master presents a valid address on LA[31:2]. The PCEB, after decoding the address and M/IO#, responds by asserting SLBURST#. The EISA master must sample SLBURST# on the rising edge of BCLK at the trailing edge of START#. The EISA master asserts MSBURST# on the falling edge of BCLK and presents a second address to the PCEB. The ESC holds CMD# asserted while the burst is being performed. If MSBURST# is not asserted by the master, the cycle is run as a standard cycle.

If the cycle is a burst read, the EISA master presents burst addresses on the falling edge of every BCLK. The PCEB presents the data for that address, which is sampled one and half BCLKs later. If the cycle is a burst write, the EISA master presents the data on the rising edge of BCLK, a half cycle after presenting the address. The PCEB samples memory write data on the rising BCLK edge when CMD# is asserted (regardless of the state of MSBURST#). The EISA master terminates the burst cycles by negating MSBURST# and completing the last transfer.

To add wait states during a burst sequence, the PCEB negates EXRDY before the falling edge of BCLK (with CMD# asserted). The EISA master samples EXRDY on the falling edge of BCLK and extends the cycle until EXRDY is asserted. The EISA master can still change the next address even though EXRDY is negated.

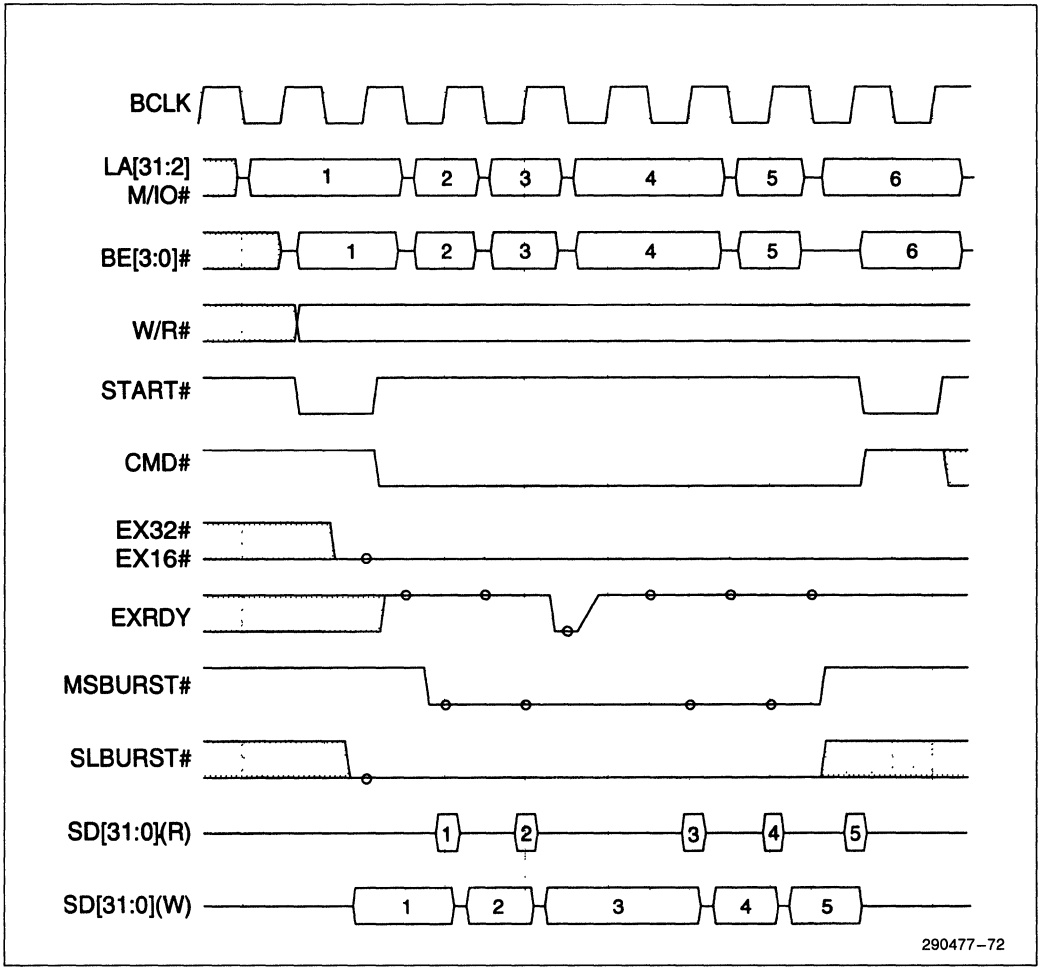


Figure 28. EISA Burst Cycle

7.3 I/O Recovery

The I/O recovery mechanism in the PCEB guarantees a minimum amount of time between back-to-back 8-bit and 16-bit PCI cycles to ISA I/O slaves. Delay times (in BCLKs) for 8-bit and 16-bit cycles are individually programmed via the IORT Register. Accesses to an 8-bit device followed by an access to a 16-bit device use the 8-bit recovery time. Similarly, accesses to a 16-bit device followed by an access to an 8-bit device use the 16-bit recovery time. The PCEB cycles to EISA I/O, DMA cycles, and EISA/ISA bus masters to I/O slaves do not require any delay between back-to-back I/O accesses.

Note that I/O recovery is only required for ISA I/O devices. However, since the PCEB does not distinguish between 8-bit ISA and 8-bit EISA, the delay is also applied to 8-bit EISA I/O accesses (i.e. the ESC).

8.0 EISA DATA SWAP BUFFERS

The PCEB contains a set of buffers/latches that perform data swapping and data size translations on the EISA Bus when the master and slave data bus sizes do not match (e.g., 32-bit EISA master accessing a 16-bit EISA slave). During a data size translation, the PCEB performs one or more of the following operations, depending on the master/slave type (PCI/EISA/ISA), transfer direction (read/write), and the number of byte enables active (BE[3:0]#):

- Data assembly or disassembly
- Data copying (up or down)
- Data re-drive

These operations are described in this section. An example is provided in Section 8.3, The Re-Drive Operation, that shows a cycle where all three functions are used.

The PCEB performs data size translations on the EISA Bus using the data swap buffer control signals generated by the ESC. These signals are described in Section 10.0, PCEB/ESC Interface.

8.1 Data Assembly And Disassembly

The data assembly/disassembly process occurs during PCI, EISA/ISA, and DMA cycles when the master data size is greater than the slave data size. For example, if a 32-bit PCI master is performing a 32-bit read cycle to an 8-bit ISA slave, the ESC intervenes and performs four 8-bit reads. The data is assembled in the PCEB (Figure 29). Once assembled, the PCEB transfers the data as a single Dword to the 32-bit PCI master during the fourth cycle. For a 32-bit write cycle, the PCEB disassembles the Dword by performing four write cycles to the slave. The actual number of cycles required to perform an assembly/disassembly process and make a transfer is a function of the number of bytes (BE[3:0]#) requested and the master/slave size combination.

During EISA master assembly/disassembly transfers, cycle control is transferred from the master to the ESC. The master relinquishes control by backing off the bus (i.e., by floating its START#, BE[3:0], and SD[31:0] signals on the first falling edge of BCLK after START# is negated). The ESC controls the assembly/disassembly process in the PCEB via the data swap buffer control signals on the PCEB/ESC interface. At the end of the assembly/disassembly process, cycle control is transferred back to the bus master (by the ESC asserting EX16# and EX32#). An additional BCLK is added at the end of the transfer to allow the exchanging of cycle control to occur. During DMA transfers, cycle control is maintained by the ESC for the entire cycle.

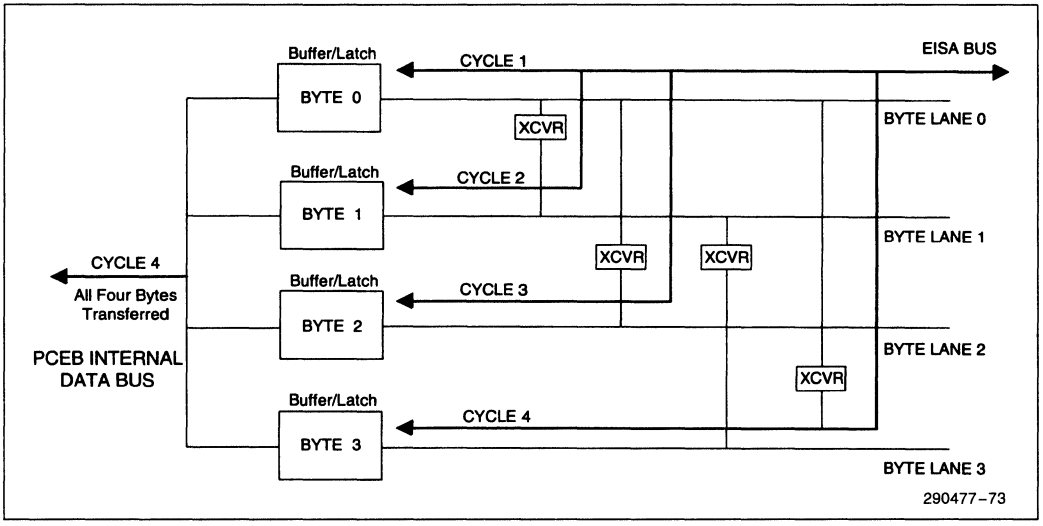


Figure 29. Assembly Function: PCI 32-bit Read from an 8-bit EISA or ISA SlaveBE[3:0] # = 0000

8.2 The Copy Operation (Up Or Down)

The copy operation (Figure 30) is invoked during data transfers between byte lanes. This operation allows the assembly/ disassembly of the data pieces during the cycles between mismatched master/slave combinations. For example, Section 8.1, Data Assembly and Disassembly, describes a 32-bit master read from an 8-bit slave where the data is copied up during the assembly process. Copy-up is used for data assembly and copy-down is used for data disassembly.

The copy-up and copy-down operations are also used during transfers where assembly or disassembly are not required. These transfers are:

- When the master size is smaller than the slave size (e.g. 16-bit EISA master cycle to a 32-bit EISA slave).
- Between a mis-matched master/slave combination when only a byte or a word needs to be transferred (e.g. 32-bit EISA master cycle to an 8-bit ISA slave and only a byte needs to be transferred).

The number of bytes copied up or down is a function of the number of bytes requested (BE[3:0]#) and the master/slave size combinations. During EISA master cycles where the data copying is performed, cycle control is transferred from the bus master to the ESC, except during transfers where the master's data size is smaller than the slave's data size. During DMA transfers, bus control is maintained by the ESC throughout the transfer.

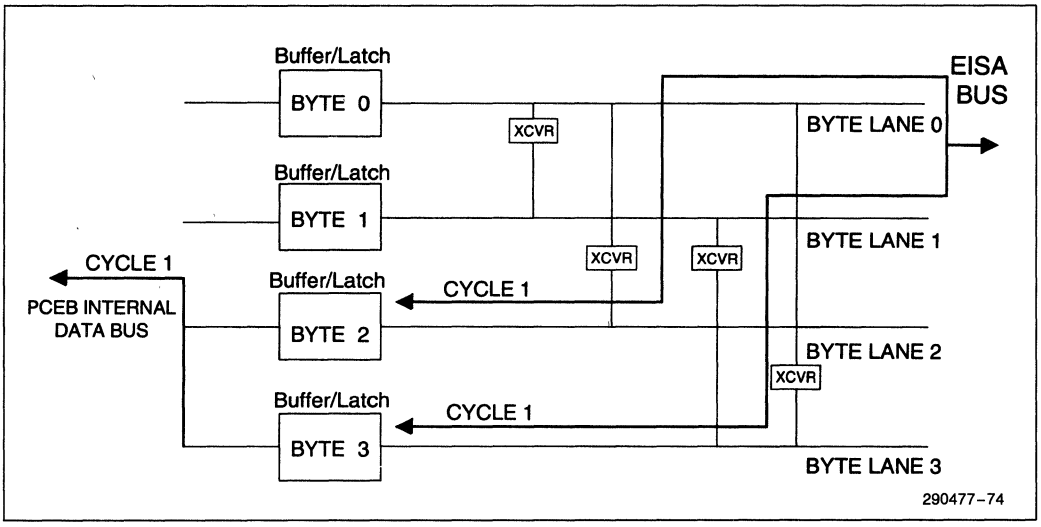


Figure 30. Copy Function: PCI 16-bit Read from a 16-bit EISA or ISA Slave—BE[3:0]# = 0011

8.3 The Re-Drive Operation

The re-drive operation (Figure 31) is used when both the master and the slave, other than PCEB, are on the EISA Bus and the master/slave size combination is mis-matched. Specifically, re-drive occurs:

- during EISA master and DMA cycles (excluding DMA compatible cycles) where the master's data size is greater than the slave's data size.
- during EISA master cycles to ISA slaves where the master/slave match in the size.
- during DMA burst write cycles to a non-burst memory slave.

During a re-drive cycle, the data is latched from the EISA Bus, and then driven back onto the appropriate EISA byte lanes. During a read cycle, the re-drive occurs after the necessary sub-cycles have been completed and the read data has been assembled. For example, when a 32 bit EISA master (other than PCEB) performs 32 bit read from an 8 bit EISA slave, the following sequence of events occurs:

1. The 32-bit EISA master initiates the read cycle. Since the master/slave combination is a mis-match, the master backs off the bus. The EISA master floats its START#, BE[3:0]# and SD[31: 0] lines. The cycle control is then transferred to the ESC.
2. The ESC brings in the first 8 bit data (byte 0) in the first cycle. The ESC asserts SDLE0# to the PCEB.
3. When SDLE0# is asserted, the PCEB latches byte 0 into the least significant byte lane.
4. In the second cycle, the ESC reads the next 8 bit data (byte 1). The PCEB uses SDLE1#, SDCPYUP and SDCPYEN0-1# to latch byte 1 and copy it to the second least significant byte lane (copy-up). This process continues for byte 2 and byte 3. On the fourth cycle, the Dword assembly is complete. During each of the 4 cycles, the ESC generates BE[3:0]# combinations.

5. The ESC instructs the PCEB to re-drive the assembled word to the master by asserting SDOE[2:0]#. In this case, all three SDOE[2:0]# signals are asserted.
6. When SDOE[2:0]# are asserted, the PCEB drives the 32 bit assembled data on SD[31:0] to be latched by the master. The ESC generates the byte enables (BE[3:0]#).
7. The ESC completes the transfer.
8. At the end of the cycle, The ESC transfers control of the EISA Bus back to the EISA master.

During a write cycle, the re-drive occurs after the write data from the master has been latched, and before the data has been disassembled. For example, during a 32-bit write by a 32-bit EISA master to an 8-bit EISA slave, in the first cycle of transfer, the data swap buffers latch the write data (Dword) from the master and drives the first byte back onto the lower byte lane of the EISA Bus. The EISA slave uses the byte enable (BE[3:0]#) combination put out by the EISA master during the first cycle to latch the least significant byte. For the subsequent cycles, the BE[3:0]# combination is generated by the ESC. The PCEB re-drives the second, third and the fourth byte on the second, third and the fourth cycles of the transfer. The number of cycles run is a function of the number of bytes requested (BE[3:0]#), and the master/slave size combinations.

During EISA master and DMA write cycles between master and slave combinations on the EISA/ISA Bus, where only copying is required and no assembly/disassembly is required, the data swap buffer treats this as a re-drive cycle. For example, during a write transfer between a 32-bit EISA master and a 16-bit EISA or ISA slave, where the master is driving data on the upper two byte lanes (BE[3:0]# = 0011), the data swap buffers latch the data on the byte lanes 2 and 3 (Figure 32). The data swap buffers will then re-drive the data onto byte lanes 2 and 3 while copying the data down to byte lanes 0 and 1, for latching by the slave device.

When the PCEB is involved as a master or slave, the re-drive function is disabled. When the PCEB reads 32-bit data from an 8-bit slave the following sequence of events occurs:

1. Same steps as steps 1-4 in the previous example.
2. Once the assembly is complete, the PCEB internally latches the data.
3. The control is transferred back to the PCEB.

NOTE:

During EISA master cycles that require re-driving, the control is transferred from the EISA master to the ESC before the data is re-driven on the data bus. However; during the DMA cycles, the cycle control is maintained by the ESC throughout the entire cycle.

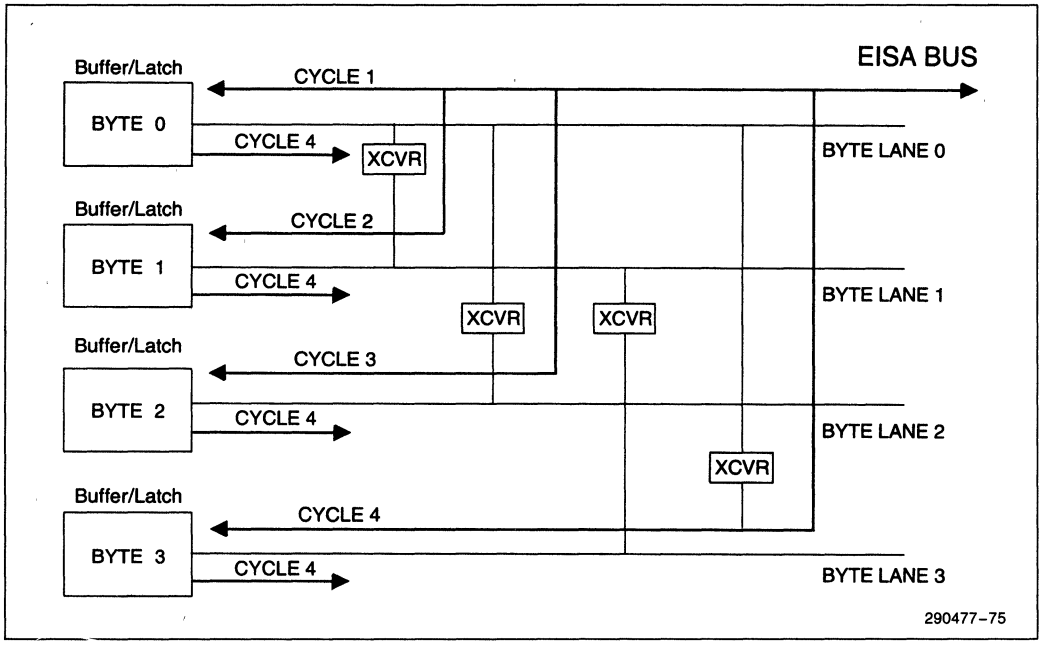


Figure 31. Re-Drive Function: 32-bit EISA Master Accessing an 8-bit EISA or ISA Slave 32-bit Read—BE[3:0] # = 0000

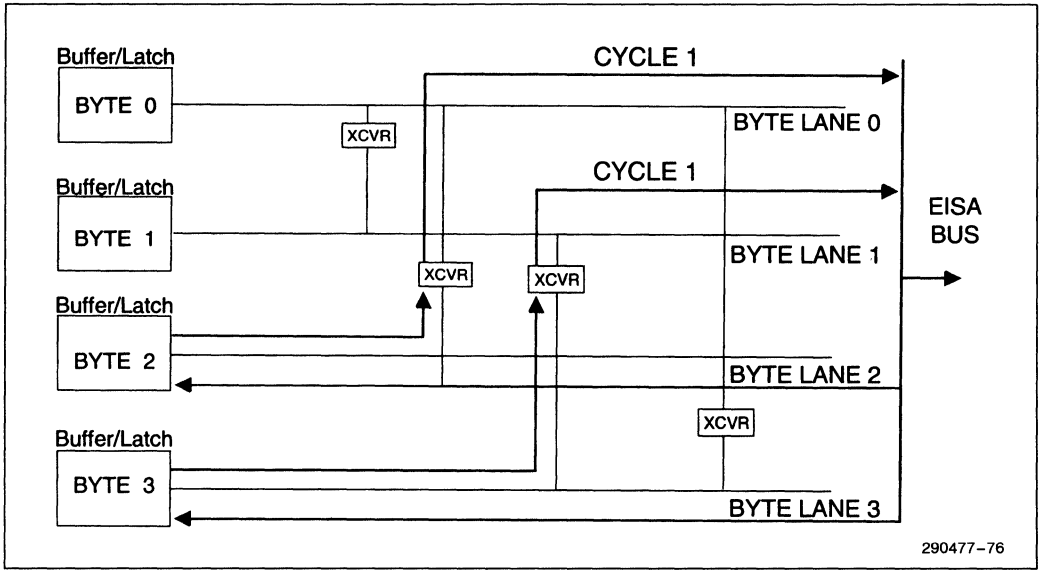


Figure 32. Copy witn Re-Drive: 32-bit EISA Master Accessing a 16-bit EISA or ISA Slave-One Word Write—BE[3:0] # = 0011

9.0 BIOS TIMER

The PCEB provides a system BIOS Timer that decrements at each edge of its 1.03/1.04 MHz clock (derived from the 8.25/8.33 MHz BCLK). Since the state of the counter is undefined at power-up, the BIOS Timer Register must be programmed before it can be used. The timer can be enabled/disabled by writing to the BIOS Timer Address Register.

The BIOS Timer Register can be accessed as a single 16-bit quantity or as 32-bit quantity. For 32-bit accesses, the upper 16 bits are don't care (reserved). The BIOS Timer I/O address location is software programmable. The address is determined by the value programmed into the BTMR Register and can be located on Dword boundaries anywhere in the 64 KByte PCI I/O space.

The BIOS Timer clock has a frequency of 1.03 or 1.04 MHz, depending on the value of BCLK (derived either from 25 MHz or 33 MHz PCICLK). This allows time intervals to be counted from 0 to approximately 65 ms. The accuracy of the counter is ± 1 μs.

9.1 BIOS Timer Operations

A write operation (either 16-bit or 32-bit) to the BIOS Timer Register initiates the counting sequence. After initialization, the BIOS timer starts decrementing until it reaches zero. When the value in the timer reaches zero, the timer stops decrementing and register value remains at zero until the timer is re-initialized.

After the timer is initialized, the current value can be read at any time. The timer can be re-programmed (new initial value written to the BIOS Timer Register) before the register value reaches zero. All write and read operations to the BIOS Timer Register should include all 16 counter bits. Separate accesses to the individual bytes of the counter must be avoided since this can cause unexpected results (incorrect count intervals).

10.0 PCEB/ESC INTERFACE

The PCEB/ESC interface (Figure 33) provides the inter-chip communications between the PCEB and ESC. The interface provides control information between the two components for PCI/EISA arbitration, data size translations (controlling the PCEB's EISA data swap buffers), and interrupt acknowledge cycles.

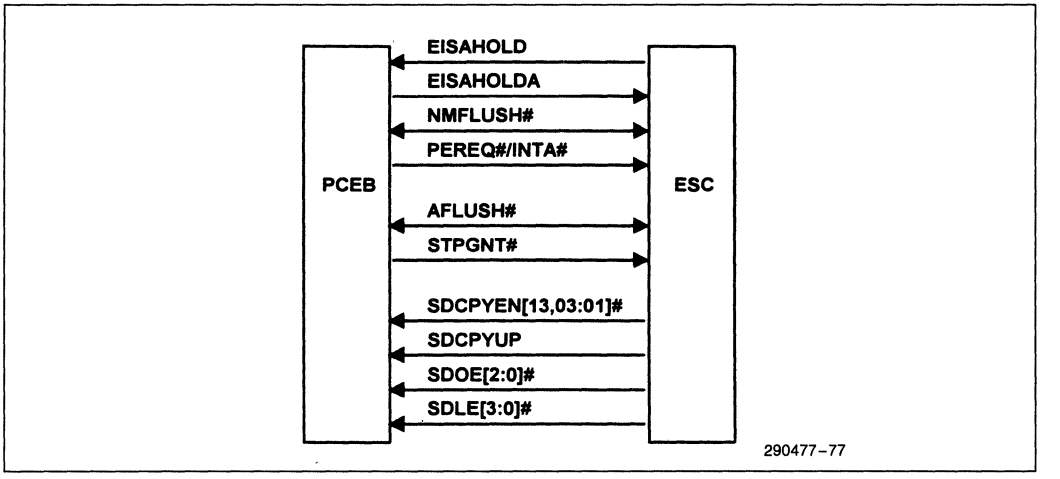


Figure 33. PCEB/ESC Interface Signals

10.1 Arbitration Control Signals

The PCEB contains the arbitration circuitry for the PCI Bus and the ESC contains the arbitration circuitry for the EISA Bus. The PCEB/ESC Interface contains a set of arbitration control signals (EISAHOLD, EISAHOLDA, NMFLUSH#, and PEREQ#/INTA#) that synchronize bus arbitration and ownership changes between the two bus environments. The signals also force PCI device data buffer flushing, if needed, to maintain data coherency during EISA Bus ownership changes.

The PCEB is the default owner of the EISA Bus. If another EISA/ISA master or DMA wants to use the bus, the ESC asserts EISAHOLD to instruct the PCEB to relinquish EISA Bus ownership. The PCEB completes any current EISA Bus transaction, tri-states its EISA Bus signals, and asserts EISAHOLDA to inform the ESC that the PCEB is off the bus.

For ownership changes, other than for a refresh cycle, the ESC asserts the NMFLUSH# signal to the PCEB (for one PCICLK) to instruct the PCEB to flush its Line Buffers pointing to the PCI Bus (Figure 34). The assertion of NMFLUSH# also instructs the PCEB to initiate flushing and to temporarily disable system buffers on the PCI Bus (via MEMREQ#, MEMACK#, and FLSHREQ#). The buffer flushing maintains data coherency, in the event that the new EISA Bus master wants to access the PCI Bus. Buffer flushing also prevents dead-lock conditions between the PCI Bus and EISA Bus. Since the ESC/PCEB do not know ahead of time, whether the new master is going to access the PCI Bus or a device on the EISA Bus, buffers pointing to the PCI Bus are always flushed when there is a change of EISA Bus ownership, except for refresh cycles. For refresh cycles, the ESC controls the cycle and, thus, knows that the cycle is not an access to the PCI Bus and does not initiate a flush request to the PCEB. After a refresh cycle, the ESC always surrenders control of the EISA Bus back to the PCEB.

NMFLUSH# is a bi-directional signal that is negated by the ESC when buffer flushing is not being requested. The ESC asserts NMFLUSH# to request buffer flushing. When the PCEB samples NMFLUSH# asserted, it starts driving the signal in the asserted state and begins the buffer flushing process. (The ESC tri-states NMFLUSH# after asserting it for the initial 1 PCICLK period.) The PCEB keeps NMFLUSH# asserted until all buffers are flushed and then it negates the signal for 1 PCICLK. When the ESC samples NMFLUSH# negated, it starts driving the signal in the negated state, completing the handshake. When the ESC samples NMFLUSH# negated, it grants ownership to the winner of the EISA Bus arbitration (at the time NMFLUSH# was negated). Note that for a refresh cycle, NMFLUSH# is not asserted by the ESC.

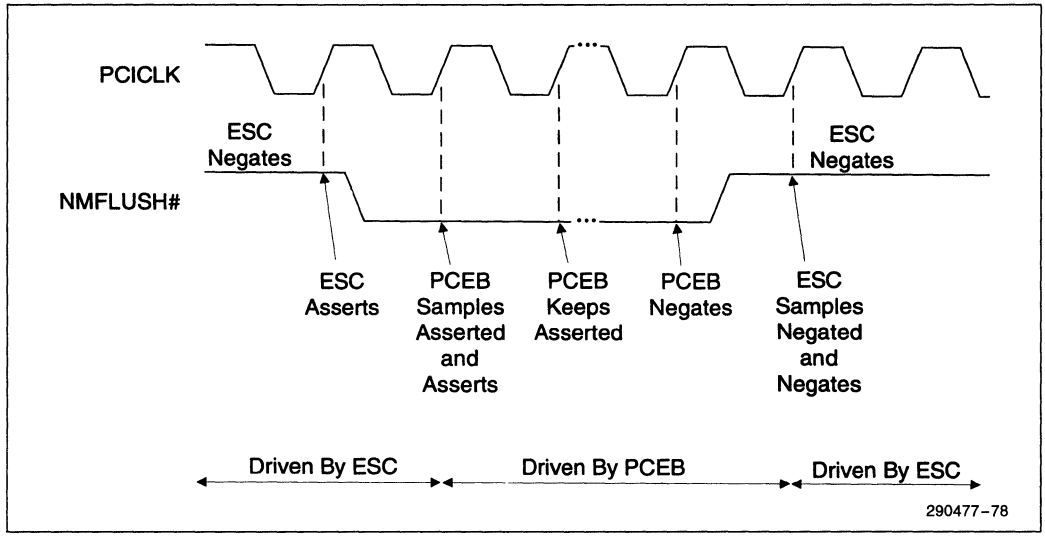


Figure 34. NMFLUSH# Protocol

When the EISA master completes its transfer and gets off the bus (i.e., removes its request to the ESC), the ESC negates EISAHOLD and the PCEB, in turn, negates EISAHOLDA. At this point, the PCEB resumes its default ownership of the EISA Bus.

If a PCI master requests access to the EISA Bus while the bus is owned by a master other than the PCEB, the PCEB retries the PCI cycle and requests ownership of the EISA Bus by asserting PEREQ#/INTA# to the ESC. PEREQ#/INTA# is a dual function signal that is a PCEB request for the EISA Bus (PEREQ# function) when EISAHOLDA is asserted. In response to the PCEB request for EISA Bus ownership, the ESC removes the grant to the EISA master. When the EISA master completes its current transactions and relinquishes the bus (removes its bus request), the ESC negates EISAHOLD and the PCEB, in turn, negates EISAHOLDA. At this point, a grant can be given to the PCI device for a transfer to the EISA Bus. Note that the INTA# function of the PEREQ#/INTA# signal is described in Section 10.3, Interrupt Acknowledge Control.

10.2 System Buffer Coherency Control-APIC

During an interrupt sequence, the system buffers must be flushed before the ESC's I/O APIC can send an interrupt message to the local APIC (CPU's APIC). The ESC and PCEB maintain buffer coherency when the ESC receives an interrupt request for its I/O APIC using the AFLUSH# signal.

10.3 Power Management (82375SB)

In response to the 82375SB ESC's STPCLK# assertion, the CPU sends out a stop grant bus cycle to indicate that it has entered the stop grant state. The PCEB uses the STPGNT# signal to inform the ESC of the stop grant cycle.

10.4 EISA Data Swap Buffer Control Signals

The cycles in the EISA environment may require data size translations before the data can be transferred to its intermediate or final destination. As an example, a 32-bit EISA master write cycle to a 16-bit EISA slave requires a disassembly of a 32-bit Dword into 16-bit words. Similarly, a 32-bit EISA master read cycle to a 16-bit slave requires an assembly of two 16 bit words into a 32-bit Dword. The PCEB contains EISA data swap buffers to support data size translations on the EISA Bus. The operation of the data swap buffers is described in Section 8.0, EISA Data Swap Buffers. The ESC controls the operation of the PCEB's data swap buffers with the following PCEB/ESC interface signals. These signals are outputs from the ESC and an inputs to the PCEB.

- SDCPYEN[13,03:01]#
- SDCPYUP
- SDOE[2:0]#
- SDLE[3:0]#

Copy Enable Outputs (SDCPYEN[13,03:01]#)

These signals enable the byte copy operations between data byte lanes 0, 1, 2 and 3 as shown in the Table 9. ISA master cycles do not perform assembly/disassembly operations. Thus, these cycles use SDCPYEN[13,03:01]# to perform the byte routing and byte copying between lanes. EISA master cycles however, can have assembly/ disassembly operations. These cycles use SDCPYEN[13,03:01]# in conjunction with SDCPYUP and SDLE[3:0]#.

Table 9. Byte Copy Operations

Signal	Copy between Byte Lanes
SDCPYEN01#	Byte 0 (bits[7:0]) and Byte 1 (bits[15:8])
SDCPYEN02#	Byte 0 (bits[7:0]) and Byte 2 (bits[23:16])
SDCPYEN03#	Byte 0 (bits[7:0]) and Byte 3 (bits[31:24])
SDCPYEN13#	Byte 1 (bits[15:8]) and Byte 3 (bits[31:24])

System Data Copy Up (SDCPYUP)

SDCPYUP controls the direction of the byte copy operations. When SDCPYUP is asserted (high), active lower bytes are copied onto the higher bytes. The direction is reversed when SDCPYUP is negated (low).

System Data Output Enable (SDOE[2:0]#)

These signals enable the output of the data swap buffers onto the EISA Bus (Table 10). SDOE[2:0] are re-drive signals in case of mis-matched cycles between EISA to EISA, EISA to ISA, ISA to ISA and the DMA cycles between the devices on EISA.

Table 10. Output Enable Operations

Signal	Byte Lane
SDOE0#	Applies to Byte 0 (bits[7:0])
SDOE1#	Applies to Byte 1 (bits[15:8])
SDOE2#	Applies to Byte 2 and Byte 3 (bits[31:16])

System Data to Internal (PCEB) Data Latch Enables (SDLE[3:0]#)

These signals latch the data from the EISA Bus into the data swap latches. The data is then either sent to the PCI Bus via the PCEB or re-driven onto the EISA Bus. SDLE[3:0]# latch the data from the corresponding EISA Bus byte lanes during PCI Reads from EISA, EISA writes to PCI, DMA cycles between an EISA device and the PCEB. These signals also latch data during mismatched cycles between EISA to EISA, EISA to ISA, ISA to ISA, the DMA cycles between the devices on EISA, and any cycles that require copying of bytes, as opposed to copying and assembly/disassembly.

10.5 Interrupt Acknowledge Control

PEREQ#/INTA# (PCI to EISA Request or Interrupt Acknowledge) is a dual function signal and the selected function depends on the status of EISAHLDA. When EISAHLDA is negated, this signal is an interrupt acknowledge (INTA#) and supports interrupt processing. If interrupt acknowledge is enabled via the PCEB's PCICON Register and EISAHOLDA is negated, the PCEB asserts PEREQ#/INTA# when a PCI interrupt acknowledge cycle is being serviced. This informs the ESC that the forwarded EISA I/O read from location 04h is an interrupt acknowledge cycle. Thus, the ESC uses this signal to distinguish between a request for the interrupt vector and a read of the ESC's DMA register located at 04h. The ESC responds to the read request by placing the interrupt vector on SD[7:0].

11.0 ELECTRICAL CHARACTERISTICS

11.1 Absolute Maximum Ratings

Case Temperature Under Bias ..... −65°C to 110°C

Storage Temperature ..... −65°C to 150°C

Supply Voltages with
Respect to Ground ..... −-0.5V to V_{CC} + 0.5V

Voltage On Any Pin ..... −-0.5V to V_{CC} + 0.5V

Power Dissipation (fully loaded) 0.95W

Power Dissipation (four slots) 0.75W

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

**WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

12.0 PINOUT AND PACKAGE INFORMATION

12.1 Pin Assignment

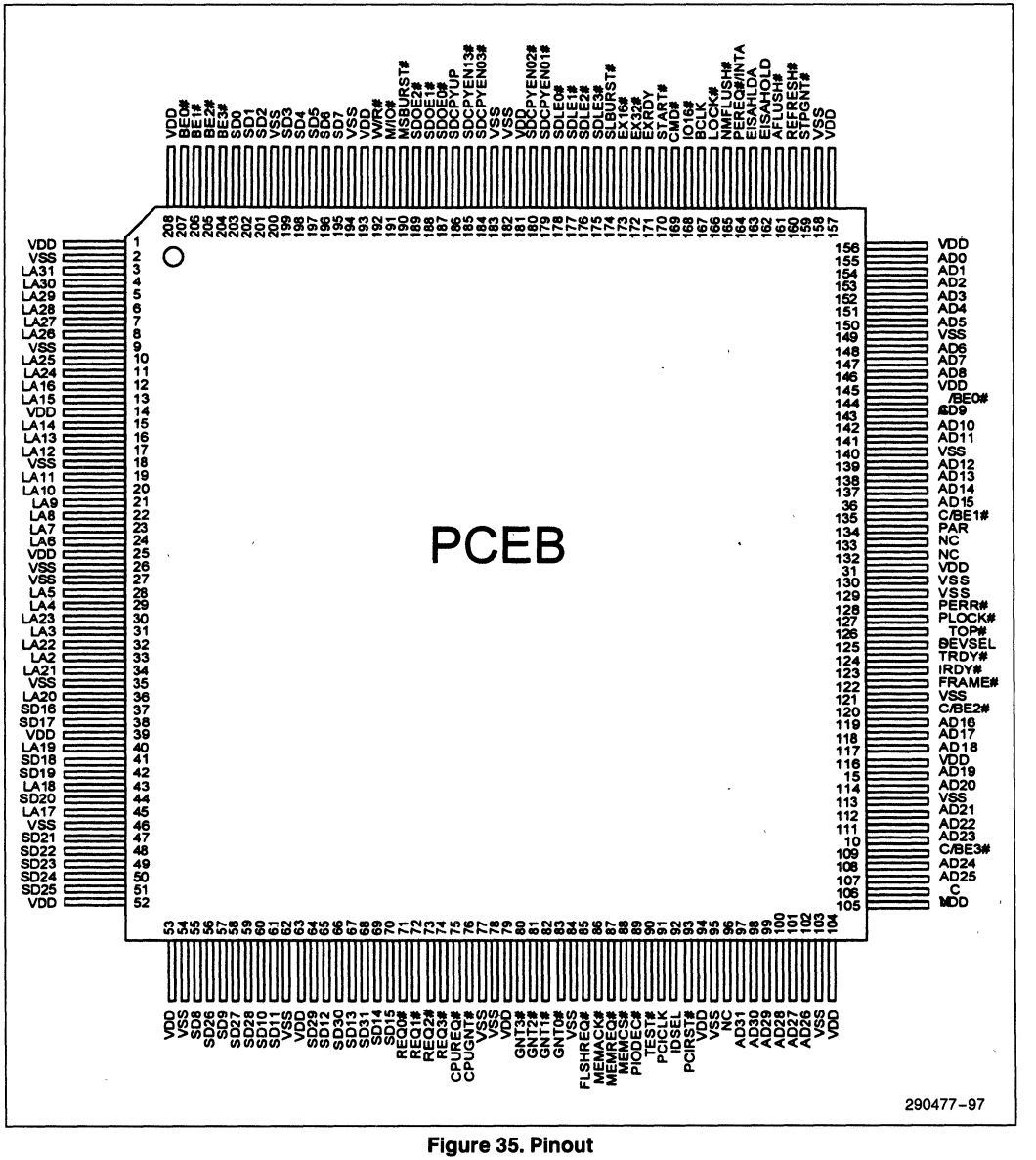


Figure 35. Pinout

Table 11. Alphabetical PCEB Pin Assignment

Name	Pin #	Type
AD0	155	t/s
AD1	154	t/s
AD2	153	t/s
AD3	152	t/s
AD4	151	t/s
AD5	150	t/s
AD6	148	t/s
AD7	147	t/s
AD8	146	t/s
AD9	143	t/s
AD10	142	t/s
AD11	141	t/s
AD12	139	t/s
AD13	138	t/s
AD14	137	t/s
AD15	136	t/s
AD16	119	t/s
AD17	118	t/s
AD18	117	t/s
AD19	115	t/s
AD20	114	t/s
AD21	112	t/s
AD22	111	t/s
AD23	110	t/s
AD24	108	t/s
AD25	107	t/s
AD26	102	t/s
AD27	101	t/s
AD28	100	t/s
AD29	99	t/s
AD30	98	t/s

Name	Pin #	Type
AD31	97	t/s
AFLUSH#	161	t/s
BCLK	167	in
BE0#	207	t/s
BE1#	206	t/s
BE2#	205	t/s
BE3#	204	t/s
C/BE0#	144	t/s
C/BE1#	135	t/s
C/BE2#	120	t/s
C/BE3#	109	t/s
CMD#	169	in
CPUGNT#	76	out
CPUREQ#	75	in
DEVSEL#	125	s/t/s
EISAHLDA	163	out
EISAHOLD	162	in
EX16#	173	in
EX32#	172	o/d
EXRDY	171	o/d
FLSHREQ#	85	out
FRAME#	122	s/t/s
GNT0#	83	out
GNT1#	82	out
GNT2#	81	out
GNT3#	80	out
IDSEL	92	in
IO16#	168	o/d
IRDY#	123	s/t/s
LA2	33	t/s
LA3	31	t/s
LA4	29	t/s

Table 11. Alphabetical PCEB Pin Assignment (Continued)

Name	Pin #	Type
LA5	28	t/s
LA6	24	t/s
LA7	23	t/s
LA8	22	t/s
LA9	21	t/s
LA10	20	t/s
LA11	19	t/s
LA12	17	t/s
LA13	16	t/s
LA14	15	t/s
LA15	13	t/s
LA16	12	t/s
LA17	45	t/s
LA18	43	t/s
LA19	40	t/s
LA20	36	t/s
LA21	34	t/s
LA22	32	t/s
LA23	30	t/s
LA24	11	t/s
LA25	10	t/s
LA26	8	t/s
LA27	7	t/s
LA28	6	t/s
LA29	5	t/s
LA30	4	t/s
LA31	3	t/s
LOCK#	166	t/s
M/IO#	191	t/s
MEMACK#	86	in
MEMCS#	88	out
MEMREQ#	87	out

Name	Pin #	Type
MSBURST#	190	t/s
NC	96	NC
NC	106	NC
NC	132	NC
NC	133	NC
NMFLUSH#	165	t/s
PAR	134	t/s
PCICLK	91	in
PCIRST#	93	in
PEREQ#/INTA#	164	out
PERR#	128	s/t/s
PIODEC#	89	in
PLOCK#	127	s/t/s
REFRESH#	160	in
REQ0#	71	in
REQ1#	72	in
REQ2#	73	in
REQ3#	74	in
SD0	203	t/s
SD1	202	t/s
SD2	201	t/s
SD3	199	t/s
SD4	198	t/s
SD5	197	t/s
SD6	196	t/s
SD7	195	t/s
SD8	55	t/s
SD9	57	t/s
SD10	60	t/s
SD11	61	t/s
SD12	65	t/s
SD13	67	t/s

Table 11. Alphabetical PCEB Pin Assignment (Continued)

Name	Pin #	Type
SD14	69	t/s
SD15	70	t/s
SD16	37	t/s
SD17	38	t/s
SD18	41	t/s
SD19	42	t/s
SD20	44	t/s
SD21	47	t/s
SD22	48	t/s
SD23	49	t/s
SD24	50	t/s
SD25	51	t/s
SD26	56	t/s
SD27	58	t/s
SD28	59	t/s
SD29	64	t/s
SD30	66	t/s
SD31	68	t/s
SDCPYEN01#	179	in
SDCPYEN02#	180	in
SDCPYEN03#	184	in
SDCPYEN13#	185	in
SDCPYUP	186	in
SDLE0#	178	in
SDLE1#	177	in
SDLE2#	176	in
SDLE3#	175	in
SDOE0#	187	in
SDOE1#	188	in
SDOE2#	189	in
SLBURST#	174	t/s
START#	170	t/s

Name	Pin #	Type
STOP#	126	s/t/s
STPGNT#	159	out
TEST#	90	in
TRDY#	124	s/t/s
V_{DD}	1	V
V_{DD}	14	V
V_{DD}	25	V
V_{DD}	39	V
V_{DD}	52	V
V_{DD}	53	V
V_{DD}	63	V
V_{DD}	79	V
V_{DD}	94	V
V_{DD}	104	V
V_{DD}	105	V
V_{DD}	116	V
V_{DD}	131	V
V_{DD}	145	V
V_{DD}	156	V
V_{DD}	157	V
V_{DD}	181	V
V_{DD}	193	V
V_{DD}	208	V
V_{SS}	2	V
V_{SS}	9	V
V_{SS}	18	V
V_{SS}	26	V
V_{SS}	27	V
V_{SS}	35	V
V_{SS}	46	V
V_{SS}	54	V
V_{SS}	62	V

Table 11. Alphabetical PCEB Pin Assignment (Continued)

Name	Pin #	Type
V_{SS}	77	V
V_{SS}	78	V
V_{SS}	84	V
V_{SS}	95	V
V_{SS}	103	V
V_{SS}	113	V
V_{SS}	121	V
V_{SS}	129	V
V_{SS}	130	V

Name	Pin #	Type
V_{SS}	140	V
V_{SS}	149	V
V_{SS}	158	V
V_{SS}	182	V
V_{SS}	183	V
V_{SS}	194	V
V_{SS}	200	V
W/R#	192	t/s

Table 12. Numerical PCEB Pin Assignment

Pin #	Name	Type
1	V_{DD}	V
2	V_{SS}	V
3	LA31	t/s
4	LA30	t/s
5	LA29	t/s
6	LA28	t/s
7	LA27	t/s
8	LA26	t/s
9	V_{SS}	V
10	LA25	t/s
11	LA24	t/s
12	LA16	t/s
13	LA15	t/s
14	V_{DD}	V
15	LA14	t/s
16	LA13	t/s
17	LA12	t/s
18	V_{SS}	V
19	LA11	t/s
20	LA10	t/s
21	LA9	t/s

Pin #	Name	Type
22	LA8	t/s
23	LA7	t/s
24	LA6	t/s
25	V_{DD}	V
26	V_{SS}	V
27	V_{SS}	V
28	LA5	t/s
29	LA4	t/s
30	LA23	t/s
31	LA3	t/s
32	LA22	t/s
33	LA2	t/s
34	LA21	t/s
35	V_{SS}	V
36	LA20	t/s
37	SD16	t/s
38	SD17	t/s
39	V_{DD}	V
40	LA19	t/s
41	SD18	t/s
42	SD19	t/s
43	LA18	t/s

Table 12. Numerical PCEB Pin Assignment (Continued)

Pin #	Name	Type
44	SD20	t/s
45	LA17	t/s
46	V_{SS}	V
47	SD21	t/s
48	SD22	t/s
49	SD23	t/s
50	SD24	t/s
51	SD25	t/s
52	V_{DD}	V
53	V_{DD}	V
54	V_{SS}	V
55	SD8	t/s
56	SD26	t/s
57	SD9	t/s
58	SD27	t/s
59	SD28	t/s
60	SD10	t/s
61	SD11	t/s
62	V_{SS}	V
63	V_{DD}	V
64	SD29	t/s
65	SD12	t/s
66	SD30	t/s
67	SD13	t/s
68	SD31	t/s
69	SD14	t/s
70	SD15	t/s
71	REQ0#	in
72	REQ1#	in
73	REQ2#	in
74	REQ3#	in
75	CPUREQ#	in

Pin #	Name	Type
76	CPUGNT#	out
77	V_{SS}	V
78	V_{SS}	V
79	V_{DD}	V
80	GNT3#	out
81	GNT2#	out
82	GNT1#	out
83	GNT0#	out
84	V_{SS}	V
85	FLSHREQ#	out
86	MEMACK#	in
87	MEMREQ#	out
88	MEMCS#	out
89	PIODEC#	in
90	TEST#	in
91	PCICLK	in
92	IDSEL	in
93	PCIRST#	in
94	V_{DD}	V
95	V_{SS}	V
96	NC	NC
97	AD31	t/s
98	AD30	t/s
99	AD29	t/s
100	AD28	t/s
101	AD27	t/s
102	AD26	t/s
103	V_{SS}	V
104	V_{DD}	V
105	V_{DD}	V
106	NC	NC
107	AD25	t/s

Table 12. Numerical PCEB Pin Assignment (Continued)

Pin #	Name	Type
108	AD24	t/s
109	C/BE3#	t/s
110	AD23	t/s
111	AD22	t/s
112	AD21	t/s
113	V_{SS}	V
114	AD20	t/s
115	AD19	t/s
116	V_{DD}	V
117	AD18	t/s
118	AD17	t/s
119	AD16	t/s
120	C/BE2#	t/s
121	V_{SS}	V
122	FRAME#	s/t/s
123	IRDY#	s/t/s
124	TRDY#	s/t/s
125	DEVSEL#	s/t/s
126	STOP#	s/t/s
127	PLOCK#	s/t/s
128	PERR#	s/t/s
129	V_{SS}	V
130	V_{SS}	V
131	V_{DD}	V
132	NC	NC
133	NC	NC
134	PAR	t/s
135	C/BE1#	t/s
136	AD15	t/s
137	AD14	t/s
138	AD13	t/s
139	AD12	t/s

Pin #	Name	Type
140	V_{SS}	Vv
141	AD11	t/s
142	AD10	t/s
143	AD9	t/s
144	C/BE0#	t/s
145	V_{DD}	V
146	AD8	t/s
147	AD7	t/s
148	AD6	t/s
149	V_{SS}	V
150	AD5	t/s
151	AD4	t/s
152	AD3	t/s
153	AD2	t/s
154	AD1	t/s
155	AD0	t/s
156	V_{DD}	V
157	V_{DD}	V
158	V_{SS}	V
159	STPGNT#	out
160	REFRESH#	in
161	AFLUSH#	t/s
162	EISAHOLD	in
163	EISAHLDA	out
164	PEREQ#/INTA#	out
165	NMFLUSH#	t/s
166	LOCK#	t/s
167	BCLK	in
168	IO16#	o/d
169	CMD#	in
170	START#	t/s
171	EXRDY	o/d

Table 12. Numerical PCEB Pin Assignment (Continued)

Pin #	Name	Type
172	EX32#	o/d
173	EX16#	in
174	SLBURST#	t/s
175	SDLE3#	in
176	SDLE2#	in
177	SDLE1#	in
178	SDLE0#	in
179	SDCPYEN01#	in
180	SDCPYEN02#	in
181	V_{DD}	V
182	V_{SS}	V
183	V_{SS}	V
184	SDCPYEN03#	in
185	SDCPYEN13#	in
186	SDCPYUP	in
187	SDOE0#	in
188	SDOE1#	in
189	SDOE2#	in
190	MSBURST#	t/s

Pin #	Name	Type
191	M/IO#	t/s
192	W/R#	t/s
193	V_{DD}	V
194	V_{SS}	V
195	SD7	t/s
196	SD6	t/s
197	SD5	t/s
198	SD4	t/s
199	SD3	t/s
200	V_{SS}	V
201	SD2	t/s
202	SD1	t/s
203	SD0	t/s
204	BE3#	t/s
205	BE2#	t/s
206	BE1#	t/s
207	BE0#	t/s
208	V_{DD}	V

12.2 Package Characteristics

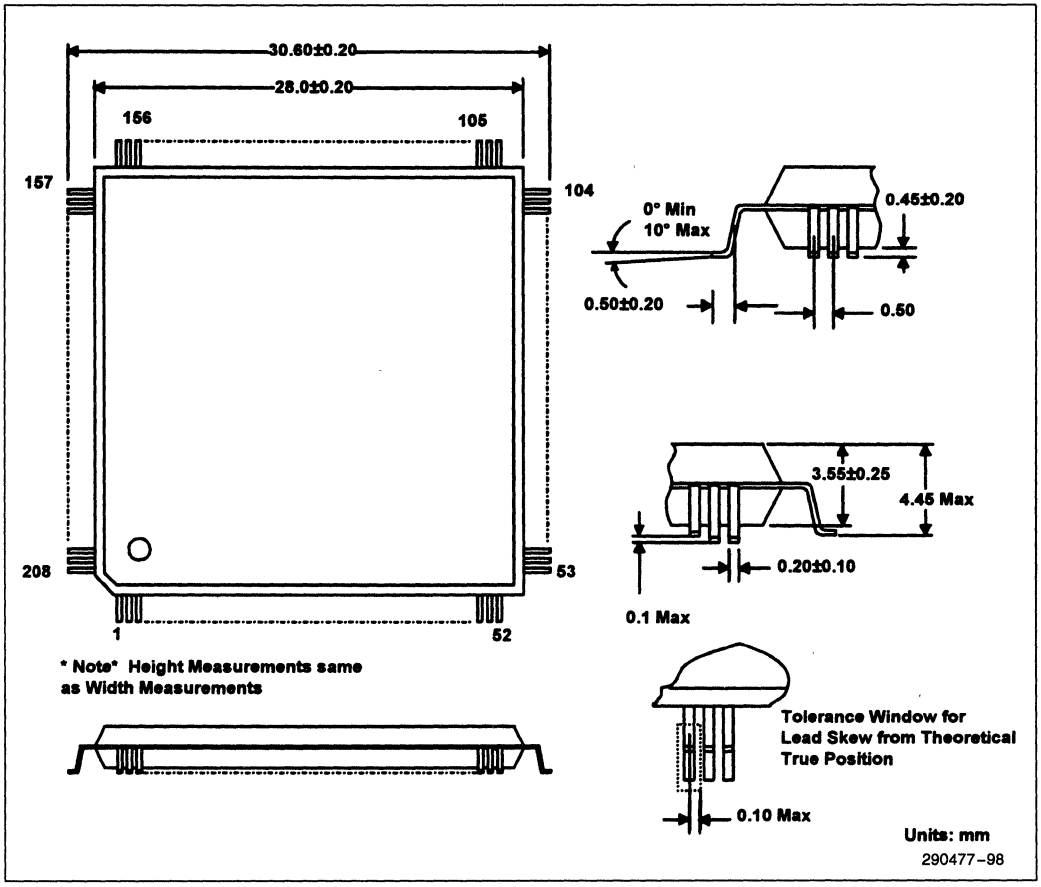


Figure 36. 208-Pin Quad Flat Pack (QFP) Dimensions

13.0 TESTABILITY

13.1 NAND Tree

A NAND Tree is provided primarily for VIL/VIH testing. The NAND Tree is also useful for Automated Test Equipment (ATE) at board level testing. The NAND Tree allows the tester to test the solder connections for each individual signal pin.

The TEST# pin, along with BCLK, PIODEC# and EX16#, activates the NAND Tree. The following combinations of PIODEC#, EX16#, and TEST# causes each buffer to be tri-stated.:

PIODEC# = 1 and EX16# = 0 and TEST# = 0
or
PIODEC# = 0 and EX16# = 1

Care must be taken as the test is in progress to ensure that one of the preceding combinations is valid. Otherwise, the test mode will be exited.

Asserting TEST# causes the output pulse train to appear on the EISAHLDA pin. BCLK must be driven low in order to enable the NAND Tree.

The sequence of the ATE test is as follows:

1. Drive TEST# low, EX16# high, PIODEC# low, and BCLK low.
2. Drive each pin high, except for the pins mentioned in the above discussion (TEST#, PIODEC#, and BCLK).
3. Starting at pin 168 (IO16#) and continuing with pins 169, 170, etc., individually drive each pin low, remembering to toggle PIODEC# from low to high when EX16# is toggled from high to low. Also, when PIODEC# is driven low, EX16# must be driven high. Expect EISAHLDA to toggle after each corresponding input pin is toggled. The final pin in the tree is pin 166 (LOCK#). BCLK is not part of the tree, and EISAHLDA is operated only as an output. Also, note that no-connect (NC), Vcc, and Vxx pins are not part of the NAND Tree.
4. Turn off tester drivers before enabling the PCEB's buffers (via PIODEC#, TEST#, and EX16#).
5. Reset the PCEB prior to proceeding with further testing.

Table 13. NAND Tree Cell Order

Pin #	Name
168	IO16#(1)
169	CMD#
170	START
171	EXRDY
172	EX32#
173	EX32#
174	SLBURST#
175	SDLE3#
176	SDLE2#
177	SDLE1#
178	SDLE0#
179	SDCPYEN01#
180	SDCPYEN02#
184	SDCPYEN03#
185	SDCPYEN13#
186	SDCPYUP
187	SDOE0#
188	SDOE1#
189	SDOE2#
190	MSBURST#
191	M/IO#
192	W/R#
195	SD7
196	SD6
197	SD5
198	SD4
199	SD3
201	SD2
202	SD1
203	SD0
204	BE3#

Pin #	Name
205	BE2#
206	BE1#
207	BE0#
3	LA31#
4	LA30#
5	LA29#
6	LA28#
7	LA27#
8	LA26#
10	LA25#
11	LA24#
12	LA16
13	LA15
15	LA14
16	LA13
17	LA12
18	V_{SS}
19	LA11
20	LA10
21	LA9
22	LA8
23	LA7
24	LA6
28	LA5
29	LA4
30	LA23
31	LA3
32	LA22
33	LA2
34	LA21
36	LA20
37	SD16

Pin #	Name
38	SD17
40	LA19
41	SD18
42	SD19
43	LA18
44	SD20
45	LA17
47	SD21
48	SD22
49	SD23
50	SD24
51	SD25
55	SD8
56	SD26
57	SD9
58	SD27
59	SD28
60	SD10
61	SD11
64	SD29
65	SD12
66	SD30
67	SD13
68	SD31
69	SD14
70	SD15
71	REQ0#
72	REQ1#
73	REQ2#
74	REQ3#
75	CPUREQ#
76	CPUGNT

Table 13. NAND Tree Cell Order

Pin #	Name
80	GNT3#
81	GNT2#
82	GNT1#
83	GNT0#
85	FLSHREQ#
86	MEMACK#
87	MEMREQ#
88	MEMCS#
89	PIODEC#(3)
91	PCICLK
92	IDSEL
93	PCIRST#
97	AD31
98	AD30
99	AD29
100	AD28
101	AD27
102	AD26
107	AD25
108	AD24
109	C/BE3#

Pin #	Name
110	AD23
111	AD22
112	AD21
114	AD20
115	AD19
117	AD18
118	AD17
119	AD16
120	C/BE2#
122	FRAME#
123	IRDY#
124	TRDY#
125	DEVSEL#
126	STOP#
127	PLOCK#
128	PERR#
134	PAR
135	C/BE1#
136	AD15
137	AD14
138	AD13

Pin #	Name
139	AD12
141	AD11
142	AD10
143	AD9
144	C/BE0#
146	AD6
147	AD7
148	AD6
150	AD5
151	AD4
152	AD3
153	AD2
154	AD1
155	AD0
160	REFRESH#
161	AFLUSH#
162	EISAHOLD
164	PEREQ#/INTA#
165	NMFLUSH#
166	LOCK#

NOTES:
1.Start of NAND Tree.
2.Must be 1 when PIODEC# is 0 and must be 0 when PIODEC# is 1.
3.Must be 0 when EX16# is 1 and must be 1 when EX16# is 0.

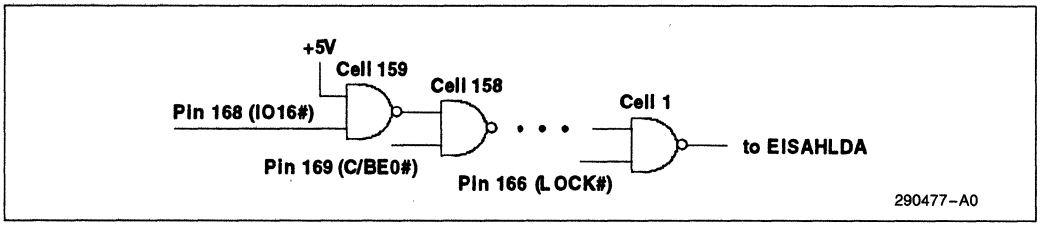


Figure 37. NAND Tree

82378ZB SYSTEM I/O (SIO) AND 82379AB SYSTEM I/O APIC (SIO.A)

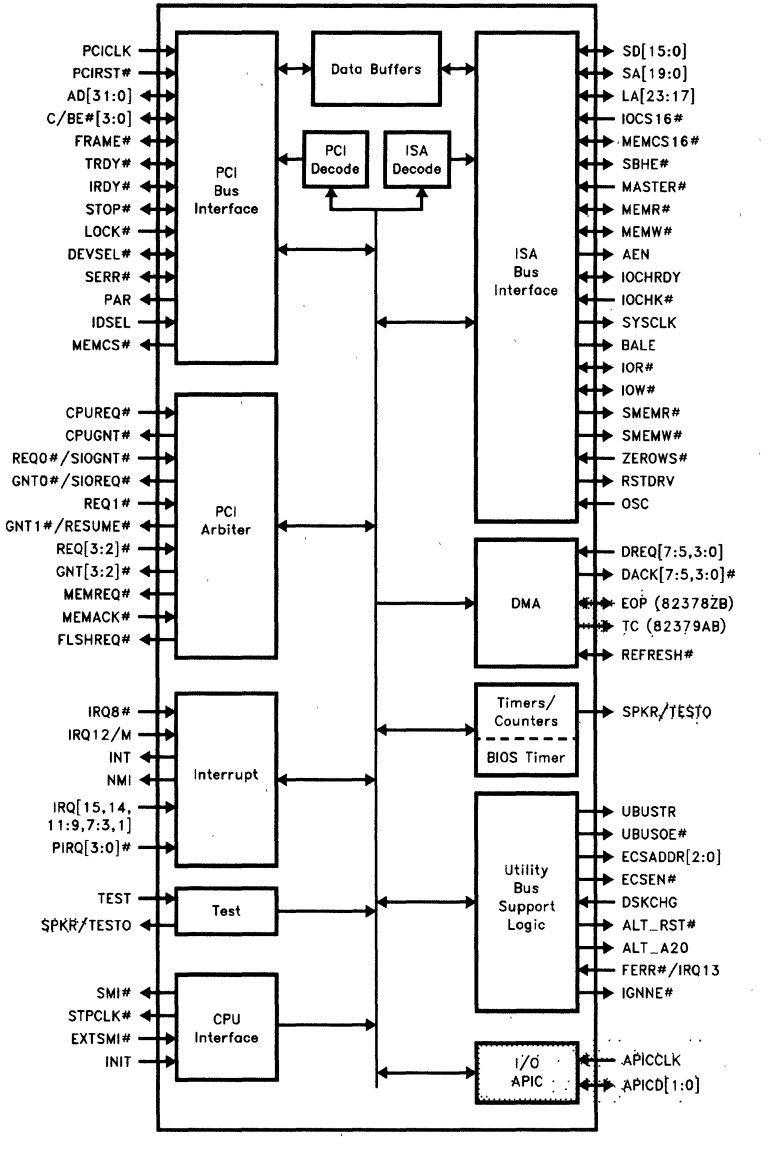
- Provides the Bridge between the PCI Bus and ISA Bus
- 100% PCI and ISA Compatible
  - PCI and ISA Master/Slave Interface
  - Directly Drives 10 PCI Loads and 6 ISA Slots
  - PCI at 25 MHz and 33 MHz
  - ISA from 6 MHz to 8.33 MHz
- Enhanced DMA Functions
  - Scatter/Gather (82378ZB)
  - Fast DMA Type A, B and F (82378ZB)
  - Compatible DMA Transfers
  - 32-bit Addressability(82378ZB)
  - 27-bit Addressability(82379AB)
  - Seven Independently Programmable Channels
  - Functionality of Two 82C37A DMA Controllers
- Data Buffers to Improve Performance
  - 8-Byte DMA/ISA Master Line Buffer
  - 32-bit Posted Memory Write Buffer to ISA
- Integrated 16-bit BIOS Timer
- Non-Maskable Interrupts (NMI)
  - PCI System Errors
  - ISA Parity Errors
- Arbitration for ISA Devices
  - ISA Masters
  - DMA and Refresh
- Four Dedicated PCI Interrupts
  - Level Sensitive
  - Mapped to Any Unused Interrupt
- Arbitration for PCI Devices
  - Six PCI Masters Supported
  - Fixed, Rotating, or a Combination
- Utility Bus (X-Bus) Peripheral Support
  - Provides Chip Select Decode
  - Controls Lower X-Bus Data Byte Transceiver
- Functionality of One 82C54 Timer
  - System Timer
  - Refresh Request
  - Speaker Tone Output
- Functionality of Two 82C59 Interrupt Controllers
  - 14 Interrupts Supported
  - Edge/Level Selectable Interrupts
- I/O APIC (Advanced Programmable Interrupt Controller (82379AB)
  - Support for Multi-Processor Systems
- System Power Management
  - Programmable System Management Interrupt (SMI)Hardware Events, Software Events, EXTSMI#
  - Programmable CPU Clock Control (STPCLK#)
  - Fast-On/Off Mode
- 208 Pin QFP Package

2

The 82378ZB System I/O (SIO) and 82379AB System I/O—APIC (SIO.A) components are PCI-to-ISA Bus Bridge devices. These devices integrate many of the common I/O functions found in today's ISA-based PC systems—a seven channel DMA controller, two 82C59 interrupt controllers, an 8254 timer/counter, a BIOS timer, Intel SMM power management support, and logic for NMI generation. In addition, the SIO and SIO.A each support a total of 6 PCI Masters, and 4 PCI Interrupts. Decode is provided for peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive. For both the SIO and SIO.A, each DMA channel supports compatibility transfers. The SIO also supports types A, B, and F transfers and scatter/gather. In addtion to the standard ISA-compatible interrupt controller that is in both the SIO and SIO.A, the SIO.A contains an Advance Programmable Interrupt Controller (IO APIC) for use in multi-processing systems.

This document describes both the 82378ZB (SIO) and 82379AB (SIO.A) components. Unshaded areas describe the 82378ZB. Shaded areas, like this one, describe differences between the 82379AB and 82378ZB.

December 1995
Order Number: 290571-001



NOTE:
The I/O APIC signals are only on the 82379AB. Also, SPKR/TESTO functions are multiplexed on the same signal pin on the 82379AB and are separate signals on the 82378ZB.

82378ZB and 82379AB Component Block Diagram

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

82378 SYSTEM I/O (SIO)

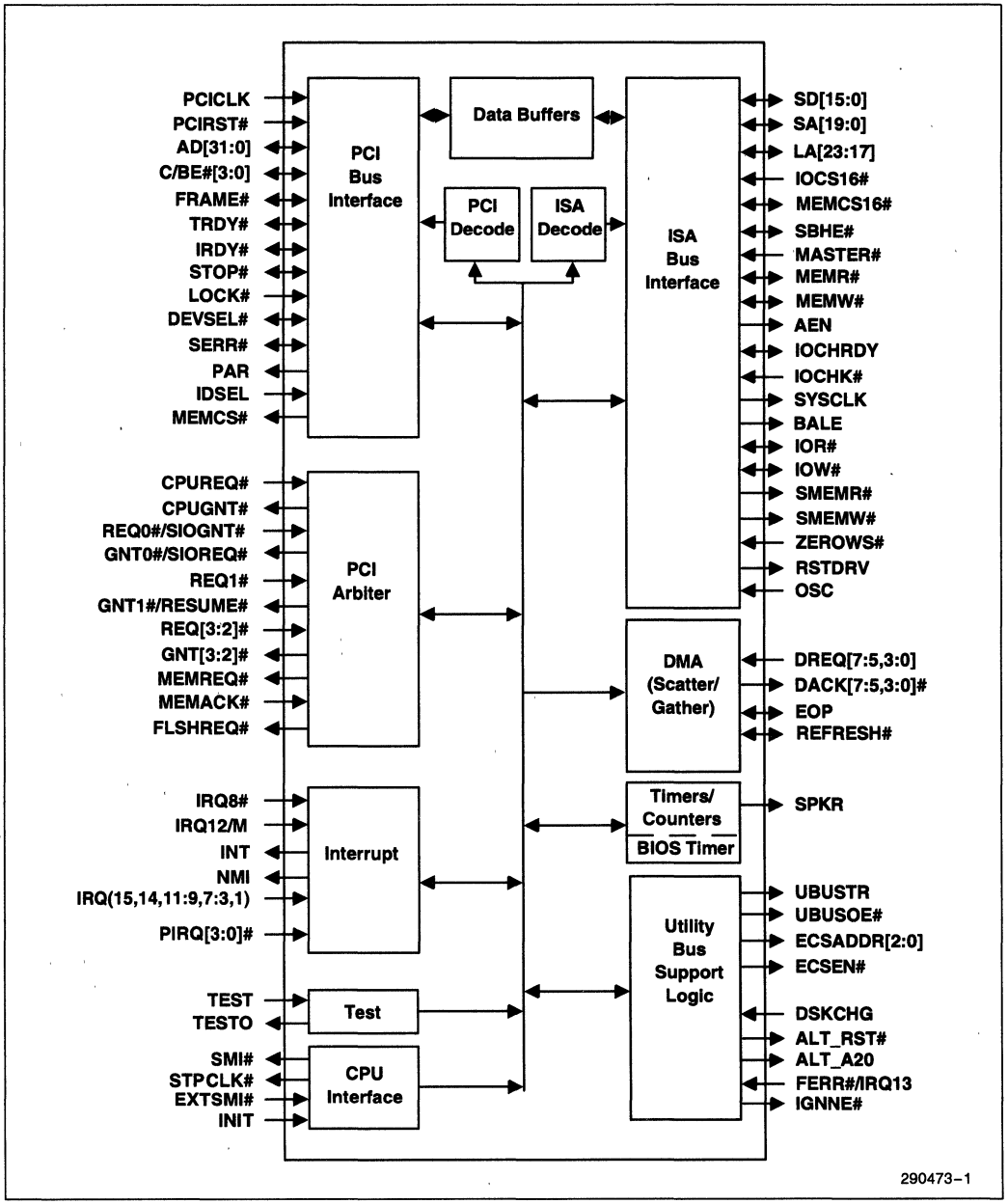
- **Provides the Bridge Between the PCI Bus and ISA Bus**
- **100% PCI and ISA Compatible**
  - **— PCI and ISA Master/Slave Interface**
  - **— Directly Drives 10 PCI Loads and 6 ISA Slots**
  - **— Supports PCI at 25 MHz and 33 MHz**
  - **— Supports ISA from 6 MHz to 8.33 MHz**
- **Enhanced DMA Functions**
  - **— Scatter/Gather**
  - **— Fast DMA Type A, B and F**
  - **— Compatible DMA Transfers**
  - **— 32-bit Addressability**
  - **— Seven Independently Programmable Channels**
  - **— Functionality of Two 82C37A DMA Controllers**
- **Integrated Data Buffers to Improve Performance**
  - **— 8-Byte DMA/ISA Master Line Buffer**
  - **— 32-bit Posted Memory Write Buffer to ISA**
- **Integrated 16-bit BIOS Timer**
- **Non-Maskable Interrupts (NMI)**
  - **— PCI System Errors**
  - **— ISA Parity Errors**
- **Arbitration for ISA Devices**
  - **— ISA Masters**
  - **— DMA and Refresh**
- **Four Dedicated PCI Interrupts**
  - **— Level Sensitive**
  - **— Can be Mapped to Any Unused Interrupt**
- **Arbitration for PCI Devices**
  - **— Six PCI Masters Supported**
  - **— Fixed, Rotating, or a Combination of the Two**
- **Utility Bus (X-Bus) Peripheral Support**
  - **— Provides Chip Select Decode**
  - **— Controls Lower X-Bus Data Byte Transceiver**
- **Integrates the Functionality of One 82C54 Timer**
  - **— System Timer**
  - **— Refresh Request**
  - **— Speaker Tone Output**
- **Integrates the Functionality of Two 82C59 Interrupt Controllers**
  - **— 14 Interrupts Supported**
  - **— Edge/Level Selectable Interrupts: Each Interrupt Individually Programmable**
- **Complete Support for SL Enhanced Intel486 CPU's**
  - **— SMI# Generation Based on System Hardware Events**
  - **— STPCLK# Generation to Power Down the CPU**

2

The 82378 System I/O (SIO) component provides the bridge between the PCI bus and the ISA expansion bus. The SIO also integrates many of the common I/O functions found in today's ISA based PC systems. The SIO incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller that supports fast DMA transfers and Scatter/Gather, data buffers to isolate the PCI bus from the ISA bus and to enhance performance, PCI and ISA arbitration, 14 level interrupt controller, a 16-bit BIOS timer, three programmable timer/counters, and Non-Maskable Interrupt (NMI) Control Logic. The SIO also provides decode for peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive.

The 82378 also supports several Advanced Power Management features such as SMI#, APM Register, Fast On and Fast Off Event Timers, Clock Throttling, and support for an external SMI# Interrupt. The 82378 also supports a total of 6 PCI Masters, and can support up to 4 PCI Interrupts.

December 1994
Order Number: 290473-004



SIO Component Block Diagram

82378 SYSTEM I/O (SIO)

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

The major functions of the SIO component are broken up into blocks as shown in the SIO Component Block Diagram. A description of each block is provided below.

PCI Bus Interface

The PCI Bus Interface provides the interface between the SIO and the PCI bus. The SIO provides both a master and slave interface to the PCI bus. As a PCI master, the SIO runs cycles on behalf of DMA, ISA masters, and the internal data buffer management logic when buffer flushing is required. The SIO will burst a maximum of two Dwords when reading from PCI memory, and one Dword when writing to PCI memory. The SIO does not generate PCI I/O cycles as a master. As a PCI slave, the SIO accepts cycles initiated by PCI masters targeted for the SIO's internal register set or the ISA bus. The SIO will accept a maximum of one data transaction before terminating the transaction. This supports the Incremental Latency Mechanism as defined in the Peripheral Component Interconnect (PCI) Specification.

As a master, the SIO generates address and command signal (C/BE#) parity for read and write cycles, and data parity for write cycles. As a slave, the SIO generates data parity for read cycles. Parity checking is not supported. The SIO also provides support for system error reporting by generating a Non-Maskable-Interrupt (NMI) when SERR# is driven active.

The SIO, as a resource, can be locked by any PCI master. In the context of locked cycles, the entire SIO subsystem (including the ISA bus) is considered a single resource.

The SIO directly supports the PCI Interface running at either 25 MHz or 33 MHz. If a frequency of less than 33 MHz is required (not including 25 MHz), a SYSCLK divisor value (as indicated in the ISA Clock Divisor Register) must be selected that guarantees that the ISA bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK range.

PCI Arbiter

The PCI arbiter provides support for six PCI masters; the Host Bridge, SIO, and four PCI masters. The arbiter can be programmed for a purely rotating scheme, fixed, or a combination of the two. The Arbiter can also be programmed to support bus parking. This gives the Host Bridge default access to the PCI bus when no other device is requesting service. The arbiter can be disabled if an external arbiter is used.

PCI Decode/ISA Decode

The SIO contains two address decoders; one to decode PCI initiated cycles and one to decode ISA master and DMA initiated cycles. Two decoders are used to allow the PCI and ISA buses to run concurrently.

The SIO is also programmable to provide address decode on behalf of the Host Bridge. When programmed, the SIO monitors the PCI and ISA address buses, and generates a memory chip select signal (MEMCS#) indicating that the current cycle is targeted for system memory residing behind the Host Bridge. This feature can be disabled through software.

Data Buffers

To isolate the slower ISA bus from the PCI bus, the SIO provides two types of data buffers. One Dword deep posted write buffer is provided for the posting of PCI initiated memory write cycles to the ISA bus. The second buffer is a bi-directional, 8-byte line buffer used for ISA master and DMA accesses to the PCI bus. All DMA and ISA master read and write cycles go through the 8-byte line buffer.

The data buffers also provide the data assembly or disassembly when needed for transactions between the PCI and ISA buses.

Buffering is programmable and can be enabled or disabled through software.

ISA Bus Interface

The SIO incorporates a fully ISA-bus compatible master and slave interface. The SIO directly drives six ISA slots without external data or address buffering. The ISA interface also provides byte swap logic, I/O recovery support, wait-state generation, and SYSCLK generation. The SIO supports ISA bus frequencies from 6 MHz to 8.33 MHz.

As an ISA master, the SIO generates cycles on behalf of DMA, Refresh, and PCI master initiated cycles. The SIO supports compressed cycles when accessing ISA slaves (i.e. ZEROWS# asserted). As an ISA slave, the SIO accepts ISA master accesses targeted for the SIO's internal register set or ISA master memory cycles targeted for the PCI bus. The SIO does not support ISA master initiated I/O cycles targeted for the PCI bus.

The SIO also monitors ISA master to ISA slave cycles to generate SMEMR# or SMEMW#, and to support data byte swapping, if necessary.

DMA

The DMA controller incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels. Each channel can be programmed for 8- or 16-bit DMA device size, and ISA-compatible or fast DMA type "A", type "B", or type "F" timings. Full 32-bit addressing is supported as an extension of the ISA-compatible specification. The DMA controller is also responsible for generating ISA refresh cycles.

The DMA controller supports an enhanced feature called Scatter/Gather. This feature provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In Scatter/Gather mode, the DMA can read the memory address and word count from an array of buffer descriptors, located in system memory, called the Scatter/Gather Descriptor (SGD) Table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD table are read.

Timer Block

The timer block contains three counters that are equivalent in function to those found in one 82C54 programmable interval timer. These three counters are combined to provide the System Timer function, Refresh Request, and speaker tone. The three counters use the 14.31818 MHz OSC input for a clock source.

In addition to the three counters, the SIO provides a programmable 16-bit BIOS timer. This timer can be used by BIOS software to implement timing loops. The timer uses the ISA system clock (SYSCLK) divided by 8 as a clock source. An 8:1 ratio between the SYSCLK and the BIOS timer clock is always maintained. The accuracy of the BIOS timer is ±1 ms.

Utility Bus (X-Bus) Logic

The SIO provides four encoded chip selects that are decoded externally to provide chip selects for Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and an IDE Hard Disk Drive. The SIO provides the control for the buffer that isolates the lower eight bits of the Utility Bus from the lower 8 bits of the ISA bus.

In addition to providing the encoded chip selects and Utility Bus buffer control, the SIO also provides Port 92 functions (Alternate Reset and Alternate A20), Coprocessor error reporting, the Floppy DSKCHG function, and a mouse interrupt input.

Interrupt Controller Block

The SIO provides an ISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The two interrupt controllers are cascaded so that 14 external and 2 internal interrupts are possible.

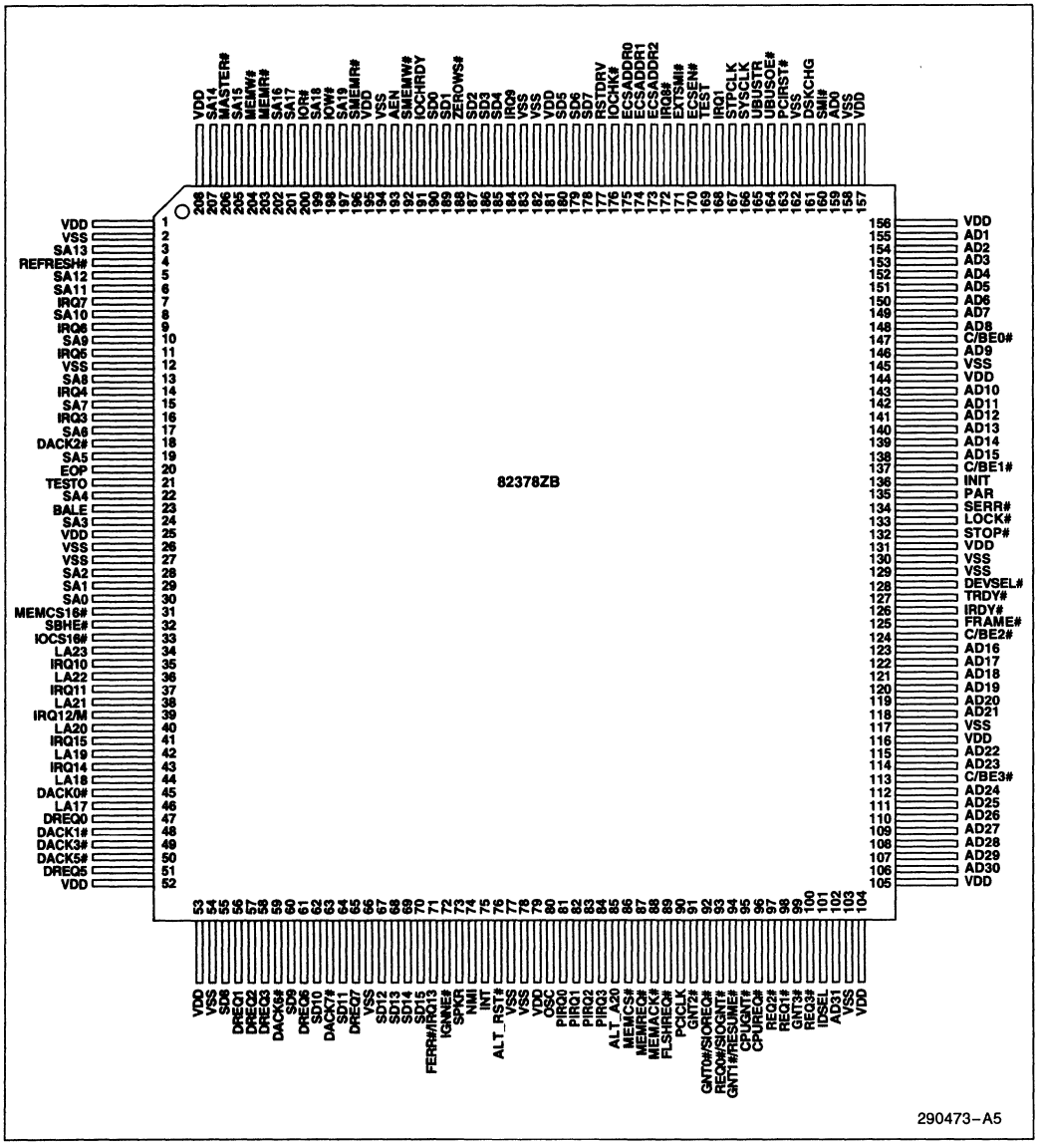
Test

The test block provides the interface to the test circuitry within the SIO. The test input can be used to tri-state all of the SIO outputs.

2.0 PIN ASSIGNMENT

The SIO package is a 208-pin Quad Flatpack (QFP). The package signals are listed in Table 1. The following notations are used to describe pin types.

Signal Type	Description
I	**Input** is a standard input-only signal.
O	**Totem Pole Output** is a standard active driver.
OD	**Open Drain** Input/Output
IO	**Input/Output** is a bidirectional, tri-state pin.
s/t/s	**Sustained Tri-State** is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. A pull-up sustains the inactive state until another agent drives it and is provided by the central resource.
t/s/o	**Tri-State Output**



290473-A5

Figure 1. SIO Package Pinout Diagram

Table 1. Alphabetical Pin Assignment

Pin Name	Pin #	Type
AD0	159	I/O
AD1	155	I/O
AD2	154	I/O
AD3	153	I/O
AD4	152	I/O
AD5	151	I/O
AD6	150	I/O
AD7	149	I/O
AD8	148	I/O
AD9	146	I/O
AD10	143	I/O
AD11	142	I/O
AD12	141	I/O
AD13	140	I/O
AD14	139	I/O
AD15	138	I/O
AD16	123	I/O
AD17	122	I/O
AD18	121	I/O
AD19	120	I/O
AD20	119	I/O
AD21	118	I/O
AD22	115	I/O
AD23	114	I/O
AD24	112	I/O
AD25	111	I/O
AD26	109	I/O
AD27	109	I/O
AD28	108	I/O
AD29	107	I/O
AD30	106	I/O
AD31	102	I/O
AEN	193	O
ALT_A20	85	O
ALT_RST#	76	O

Pin Name	Pin #	Type
BALE	23	O
C/BE0#	147	I/O
C/BE1#	137	I/O
C/BE2#	124	I/O
C/BE3#	113	I/O
CPUGNT#	95	t/s/o
CPUREQ#	96	I
DACK0#	45	O
DACK1#	48	O
DACK2#	18	O
DACK3#	49	O
DACK5#	50	O
DACK6#	59	O
DACK7#	63	O
DEVSEL#	128	I/O (s/t/s)
DREQ0	47	I
DREQ1	56	I
DREQ2	57	I
DREQ3	58	I
DREQ5	51	I
DREQ6	61	I
DREQ7	65	I
DSKCHG	161	I
ECSADDR0	175	O
ECSADDR1	174	O
ECSADDR2	173	O
ECSEN#	170	O
EOP	20	I/O
EXTSMI#	171	I
FERR#/IRQ13	71	I
FLSHREQ#	89	t/s/o
FRAME#	125	I/O (s/t/s)
GNT0#/SIOREQ#	92	t/s/o
GNT1#/RESUME#	94	t/s/o
GNT2#	91	t/s/o

Table 1. Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
GNT3#	99	t/s/o
IDSEL	101	I
IGNNE#	72	O
INIT	136	I
INT	75	O
IOCHK#	176	I
IOCHRDY	191	I/O
IOCS16#	33	I
IOR#	200	I/O
IOW#	198	I/O
IRDY#	126	I/O (s/t/s)
IRQ1	168	I
IRQ3	16	I
IRQ4	14	I
IRQ5	11	I
IRQ6	9	I
IRQ7	7	I
IRQ8#	172	I
IRQ9	184	I
IRQ10	35	I
IRQ11	37	I
IRQ12/M	39	I
IRQ14	43	I
IRQ15	41	I
LA17	46	I/O
LA18	44	I/O
LA19	42	I/O
LA20	40	I/O
LA21	38	I/O
LA22	36	I/O
LA23	34	I/O
LOCK#	133	I (s/t/s)
MASTER#	206	I
MEMACK#	88	I
MEMCS#	86	O

Pin Name	Pin #	Type
MEMCS16#	31	I/O (o/d)
MEMR#	203	I/O
MEMREQ#	87	t/s/o
MEMW#	204	I/O
NMI	74	O
OSC	80	I
PAR	135	O
PCICLK	90	I
PCIRST#	163	I
PIRQ0#	81	I
PIRQ1#	82	I
PIRQ2#	83	I
PIRQ3#	84	I
REFRESH#	4	I/O
REQ0#/SIOGNT#	93	I
REQ1#	98	I
REQ2#	97	I
REQ3#	100	I
RSTDRV	177	O
SA0	30	I/O
SA1	29	I/O
SA2	28	I/O
SA3	24	I/O
SA4	22	I/O
SA5	19	I/O
SA6	17	I/O
SA7	15	I/O
SA8	13	I/O
SA9	10	I/O
SA10	8	I/O
SA11	6	I/O
SA12	5	I/O
SA13	3	I/O
SA14	207	I/O
SA15	205	I/O

Table 1. Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
SA16	202	I/O
SA17	201	I/O
SA18	199	I/O
SA19	197	I/O
SBHE#	32	I/O
SD0	190	I/O
SD1	189	I/O
SD2	187	I/O
SD3	186	I/O
SD4	185	I/O
SD5	180	I/O
SD6	179	I/O
SD7	178	I/O
SD8	55	I/O
SD9	60	I/O
SD10	62	I/O
SD11	64	I/O
SD12	67	I/O
SD13	68	I/O
SD14	69	I/O
SD15	70	I/O
SERR#	134	I
SMEMR#	196	O
SMEMW#	192	O
SMI#	160	O
SPKR	73	O
STOP#	132	I/O (s/t/s)
STPCLK#	167	O
SYSCLK	166	O
TEST	169	I
TEST0	21	O
TRDY#	127	I/O (s/t/s)
UBUSOE#	164	O
UBUSTR	165	O

Pin Name	Pin #	Type
ZEROWS#	188	I
V_{DD}	1	V
V_{DD}	79	V
V_{DD}	104	V
V_{DD}	105	V
V_{DD}	116	V
V_{DD}	131	V
V_{DD}	144	V
V_{DD}	156	V
V_{DD}	157	V
V_{DD}	181	V
V_{DD}	25	V
V_{DD}	52	V
V_{DD}	53	V
V_{DD}	195	V
V_{DD}	208	V
V_{SS}	2	V
V_{SS}	12	V
V_{SS}	26	V
V_{SS}	27	V
V_{SS}	54	V
V_{SS}	66	V
V_{SS}	77	V
V_{SS}	78	V
V_{SS}	103	V
V_{SS}	117	V
V_{SS}	129	V
V_{SS}	130	V
V_{SS}	145	V
V_{SS}	158	V
V_{SS}	162	V
V_{SS}	182	V
V_{SS}	183	V
V_{SS}	194	V

Table 2. Numerical Pin Assignment

Pin Name	Pin #	Type
V_{DD}	1	V
V_{SS}	2	V
SA13	3	I/O
REFRESH#	4	I/O
SA12	5	I/O
SA11	6	I/O
IRQ7	7	I
SA10	8	I/O
IRQ6	9	I
SA9	10	I/O
IRQ5	11	I
V_{SS}	12	V
SA8	13	I/O
IRQ4	14	I
SA7	15	I/O
IRQ3	16	I
SA6	17	I/O
DACK2#	18	O
SA5	19	I/O
EOP	20	I/O
TEST0	21	O
SA4	22	I/O
BALE	23	O
SA3	24	I/O
V_{DD}	25	V
V_{SS}	26	V
V_{SS}	27	V
SA2	28	I/O
SA1	29	I/O
SA0	30	I/O
MEMCS16#	31	I/O (o/d)
SBHE#	32	I/O
IOCS16#	33	I
LA23	34	I/O
IRQ10	35	I

Pin Name	Pin #	Type
LA22	36	I/O
IRQ11	37	I
LA21	38	I/O
IRQ12/M	39	I
LA20	40	I/O
IRQ15	41	I
LA19	42	I/O
IRQ14	43	I
LA18	44	I/O
DACK0#	45	O
LA17	46	I/O
DREQ0	47	I
DACK1#	48	O
DACK3#	49	O
DACK5#	50	O
DREQ5	51	I
V_{DD}	52	V
V_{DD}	53	V
V_{SS}	54	V
SD8	55	I/O
DREQ1	56	I
DREQ2	57	I
DREQ3	58	I
DACK6#	59	O
SD9	60	I/O
DREQ6	61	I
SD10	62	I/O
DACK7#	63	O
SD11	64	I/O
DREQ7	65	I
V_{SS}	66	V
SD12	67	I/O
SD13	68	I/O
SD14	69	I/O
SD15	70	I/O

Table 2. Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
FERR#/IRQ13	71	I
IGNNE#	72	O
SPKR	73	O
NMI	74	O
INT	75	O
ALT_RST#	76	O
V_{SS}	77	V
V_{SS}	78	V
V_{DD}	79	V
OSC	80	I
PIRQ0#	81	I
PIRQ1#	82	I
PIRQ2#	83	I
PIRQ3#	84	I
ALT_A20	85	O
MEMCS#	86	O
MEMREQ#	87	t/s/o
MEMACK#	88	I
FLSHREQ#	89	t/s/o
PCICLK	90	I
GNT2#	91	t/s/o
GNT0#/SIOREQ#	92	t/s/o
REQ0#/SIOGNT#	93	I
GNT1#/RESUME#	94	t/s/o
CPUGNT#	95	t/s/o
CPUREQ#	96	I
REQ2#	97	I
REQ1#	98	I
GNT3#	99	I
REQ3#	100	I
IDSEL	101	I
AD31	102	I/O
V_{SS}	103	V
V_{DD}	104	V
V_{DD}	105	V

Pin Name	Pin #	Type
AD30	106	I/O
AD29	107	I/O
AD28	108	I/O
AD27	109	I/O
AD26	110	I/O
AD25	111	I/O
AD24	112	I/O
C/BE3#	113	I/O
AD23	114	I/O
AD22	115	I/O
V_{DD}	116	V
V_{SS}	117	V
AD21	118	I/O
AD20	119	I/O
AD19	120	I/O
AD18	121	I/O
AD17	122	I/O
AD16	123	I/O
C/BE2#	124	I/O
FRAME#	125	I/O (s/t/s)
IRDY#	126	I/O (s/t/s)
TRDY#	127	I/O (s/t/s)
DEVSEL#	128	I/O (s/t/s)
V_{SS}	129	V
V_{SS}	130	V
V_{DD}	131	V
STOP#	132	I/O (s/t/s)
LOCK#	133	I (s/t/s)
SERR#	134	I
PAR	135	O
INIT	136	I
C/BE1#	137	I/O
AD15	138	I/O
AD14	139	I/O
AD13	140	I/O

Table 2. Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
AD12	141	I/O
AD11	142	I/O
AD10	143	I/O
V_{DD}	144	V
V_{SS}	145	V
AD9	146	I/O
C/BE0#	147	I/O
AD8	148	I/O
AD7	149	I/O
AD6	150	I/O
AD5	151	I/O
AD4	152	I/O
AD3	153	I/O
AD2	154	I/O
AD1	155	I/O
V_{DD}	156	V
V_{DD}	157	V
V_{SS}	158	V
AD0	159	I/O
SMI#	160	O
DSKCHG	161	I
V_{SS}	162	V
PCIRST#	163	I
UBUSOE#	164	O
UBUSTR	165	O
SYSCLK	166	O
STPCLK#	167	O
IRQ1	168	I
TEST	169	I
ECSEN#	170	O
EXTSMI#	171	I
IRQ8#	172	I
ECSADDR2	173	O
ECSADDR1	174	O

Pin Name	Pin #	Type
ECSADDR0	175	O
IOCHK#	176	I
RSTDRV	177	O
SD7	178	I/O
SD6	179	I/O
SD5	180	I/O
V_{DD}	181	V
V_{SS}	182	V
V_{SS}	183	V
IRQ9	184	I
SD4	185	I/O
SD3	186	I/O
SD2	187	I/O
ZEROWS#	188	I
SD1	189	I/O
SD0	190	I/O
IOCHRDY	191	I/O
SMEMW#	192	O
AEN	193	O
V_{SS}	194	V
V_{DD}	195	V
SMEMR#	196	O
SA19	197	I/O
IOW#	198	I/O
SA18	199	I/O
IOR#	200	I/O
SA17	201	I/O
SA16	202	I/O
MEMR#	203	I/O
MEMW#	204	I/O
SA15	205	I/O
MASTER#	206	I
SA14	207	I/O
V_{DD}	208	V

3.0 SIGNAL DESCRIPTION

This section contains a detailed description of each signal. The signals are arranged in functional groups according to the interface.

Note that the "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of "active-low" and "active-high" signals. The term **assert**, or **assertion** indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term **negate**, or **negation** indicates that a signal is inactive.

3.1 PCI Bus Interface Signals

Signal Name	Type	Description
PCICLK	I	**PCI CLOCK:** PCICLK provides timing for all transactions on the PCI Bus. All other PCI signals are sampled on the rising edge of PCICLK, and all timing parameters are defined with respect to this edge. Frequencies supported by the SIO include 25 MHz and 33 MHz.
PCIRST#	I	**PCI RESET:** PCIRST# forces the SIO to a known state. AD[31:0], C/BE[3:0]#, and PAR are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states these signals from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO will drive these signals low again (synchronously 2–5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), these signals remain tri-stated until the SIO is required to drive them valid as a master or slave. FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated from the leading edge of PCIRST#. FRAME#, IRDY#, TRDY#, STOP#, and DEVSEL# remain tri-stated until driven by the SIO as either a master or a slave. MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated until driven by the SIO. After PCIRST#, MEMREQ# and FLSHREQ# are driven inactive asynchronously from PCIRST# inactive. CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are driven based on the arbitration scheme and the asserted REQx#'s. All registers are set to their default values. PCIRST# may be asynchronous to PCICLK when asserted or negated. Although asynchronous, negation must be a clean, bounce-free edge. Note that PCIRST# must be asserted for more than 1 μs.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
AD[31:0]	I/O	**PCI ADDRESS/DATA.** AD[31:0] is a multiplexed address and data bus. During the first clock of a transaction, AD[31:0] contain a physical byte address (32 bits). During subsequent clocks, AD[31:0] contain data. A SIO Bus transaction consists of an address phase followed by one or more data phases. Little-endian byte ordering is used. AD[7:0] define the least significant byte (LSB) and AD[31:24] the most significant byte (MSB). When the SIO is a target, AD[31:0] are inputs during the address phase of a transaction. During the following data phase(s), the SIO may be asked to supply data on AD[31:0] for a PCI read, or accept data for a PCI write. As a master, the SIO drives a valid address on AD[31:2] during the address phase, and drives write or latches read data on AD[31:0] during the data phase. The SIO always drives AD[1:0] low as a master. AD[31:0] are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states AD[31:0] from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives AD[31:0] low again (synchronously 2–5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), AD[31:0] remain tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving the AD[31:0] signals when no one is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive AD[31:0] as the central resource. The SIO is always responsible for driving AD[31:0] when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.
C/BE[3:0]#	I/O	**BUS COMMAND AND BYTE ENABLES:** The command and byte enable signals are multiplexed on the same PCI pins. During the address phase of a transaction, C/BE[3:0]# define the bus command. During the data phase C/BE[3:0]# are used as Byte Enables. The Byte Enables determine which byte lanes carry meaningful data. C/BE#[0] applies to byte 0, C/BE[1] to byte 1, C/BE[2] to byte 2, and C/BE#[3] to byte 3. The SIO drives C/BE[3:0]# as an initiator of a PCI Bus cycle and monitors C/BE[3:0]# as a Target. C/BE[3:0]# are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states C/BE[3:0]# from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives C/BE[3:0]# low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), C/BE[3:0]# remain tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving the C/BE[3:0]# signals when no one is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive C/BE[3:0]# as the central resource. The SIO is always responsible for driving C/BE[3:0]# when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
FRAME#	I/O (s/t/s)	**CYCLE FRAME:** FRAME# is driven by the current master to indicate the beginning and duration of an access. FRAME# is asserted to indicate a bus transaction is beginning. While FRAME# is asserted data transfers continue. When FRAME# is negated the transaction is in the final data phase. FRAME# is an input to the SIO when the SIO is the target. FRAME# is an output when the SIO is the initiator. FRAME# is tri-stated from the leading edge of PCIRST#. FRAME# remains tri-stated until driven by the SIO as either a master or a slave.
TRDY#	I/O (s/t/s)	**TARGET READY:** TRDY# indicates the SIO's ability to complete the current data phase of the transaction. TRDY# is used in conjunction with IRDY#. A data phase is completed when both TRDY# and IRDY# are sampled asserted. During a read, TRDY# indicates that the SIO, as a target, has placed valid data on AD[31:0]. During a write, it indicates the SIO, as a target is prepared to latch data. TRDY is an input to the SIO when the SIO is the initiator and an output when the SIO is a target. TRDY# is tri-stated from the leading edge of PCIRST#. TRDY# remains tri-stated until driven by the SIO as either a master or a slave.
IRDY#	I/O (s/t/s)	**INITIATOR READY:** IRDY# indicates the SIO's ability, as an initiator, to complete the current data phase of the transaction. It is used in conjunction with TRDY#. A data phase is completed on any clock that both IRDY# and TRDY# are sampled asserted. During a write, IRDY# indicates the SIO has valid data present on AD[31:0]. During a read, it indicates the SIO is prepared to latch data. IRDY is an input to the SIO when the SIO is the target and an output when the SIO is an initiator. IRDY# is tri-stated from the leading edge of PCIRST#. IRDY# remains tri-stated until driven by the SIO as either a master or a slave.
STOP#	I/O (s/t/s)	**STOP:** STOP# indicates that the SIO, as a target, is requesting a master to stop the current transaction. As a master, STOP# causes the SIO to stop the current transaction. STOP# is an output when the SIO is a target and an input when the SIO is an initiator. STOP# is tri-stated from the leading edge of PCIRST#. STOP# remains tri-stated until driven by the SIO as either a master or a slave.
LOCK#	I	**LOCK:** LOCK# indicates an atomic operation that may require multiple transactions to complete. LOCK# is always an input to the SIO. When the SIO is the target of a transaction and samples LOCK# negated during the address phase of a transaction, the SIO considers itself a locked resource until it samples LOCK# and FRAME# negated. When other masters attempt accesses while the SIO is locked, the SIO responds with a retry termination. LOCK# is tri-stated during reset.
IDSEL	I	**INITIALIZATION DEVICE SELECT:** IDSEL is used as a chip select during configuration read and write transactions. The SIO samples IDSEL during the address phase of a transaction. If IDSEL is sampled active, and the bus command is a configuration read or write, the SIO responds by asserting DEVSEL# on the next cycle.
DEVSEL#	I/O (s/t/s)	**DEVICE SELECT:** The SIO asserts DEVSEL# to claim a PCI transaction through positive or subtractive decoding. As an output, the SIO asserts DEVSEL# when it samples IDSEL active in configuration cycles to SIO configuration registers. The SIO also asserts DEVSEL# when an internal SIO address is decoded or when the SIO subtractively decodes a cycle. As an input, DEVSEL# indicates the response to a SIO master-initiated transaction. The SIO also samples this signal for all PCI transactions to decide to subtractively decode the cycle. DEVSEL# is tri-stated from the leading edge of PCIRST#. DEVSEL# remains tri-stated until driven by the SIO as either a master or a slave.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
PIRQ[3:0]#	I	**PCI INTERRUPT REQUEST:** PIRQ#s are used to generate asynchronous interrupts to the CPU via the Programmable Interrupt Controllers (82C59s) integrated in the SIO. These signals are defined as level sensitive and are asserted low. The PIRQx# interrupts can be steered into any unused IRQ interrupt. The PIRQx# Route Control Register determines which IRQ interrupt each PCI interrupt is steered into. These pins include a weak internal pull-up resistor.
PAR	O	**CALCULATED PARITY SIGNAL:** PAR is "even" parity and is calculated on 36 bits—AD[31:0] plus C/BE[3:0]#. "Even" parity means that the number of "1"s within the 36 bits plus PAR are counted and the sum is always even. PAR is always calculated on 36 bits regardless of the valid byte enables. PAR is generated for address and data phases and is only guaranteed to be valid one PCI clock after the corresponding address or data phase. PAR is driven and tri-stated identically to the AD[31:0] lines except that PAR is delayed by exactly one PCI clock. PAR is an output during the address phase (delayed one clock) for all SIO master transactions. It is also an output during the data phase (delayed one clock) when the SIO is the master of a PCI write transaction, and when it is the target of a read transaction. PAR is always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states PAR from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives PAR low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), PAR remains tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving PAR when no device is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive PAR as the central resource. The SIO is always responsible for driving PAR when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.
SERR#	I	**SYSTEM ERROR:** SERR# can be pulsed active by any PCI device that detects a system error condition. Upon sampling SERR# active, the SIO generates a non-maskable interrupt (NMI) to the CPU.

3.2 PCI Arbiter Signals

Signal Name	Type	Description
CPUREQ#	I	**CPU REQUEST:** This signal provides the following functions: 1. If CPUREQ# is sampled high on the trailing edge of PCIRST#, the internal arbiter is enabled. If CPUREQ# is sampled low on the trailing edge of PCIRST#, the internal arbiter is disabled. This requires that the host bridge drive CPUREQ# high during PCIRST#. 2. If the SIO's internal arbiter is enabled, this pin is configured as CPUREQ#. An active low assertion indicates that the CPU initiator desires the use of the PCI Bus. If the internal arbiter is disabled, this pin is meaningless after reset. This pin has a weak internal pull-up resistor.
REQ0#/ SIOGNT#	I	**REQUEST 0/SIO GRANT:** If the SIO's internal arbiter is enabled, this pin is configured as REQ0#. An active low assertion indicates that Initiator0 desires the use of the PCI Bus. If the internal arbiter is disabled, this pin is configured as SIOGNT#. When asserted, SIOGNT# indicates that the external PCI arbiter has granted use of the bus to the SIO. This pin has a weak internal pull-up resistor.
REQ1#	I	**REQUEST 1:** If the SIO's internal arbiter is enabled through the Arbiter Configuration Register, then this signal is configured as REQ1#. An active low assertion indicates that Initiator1 desires the use of the PCI Bus. If the internal arbiter is disabled, the SIO ignores REQ1# after reset. This pin has a weak internal pull-up resistor.
CPUGNT#	t/s/o	**CPU GRANT:** If the SIO's internal arbiter is enabled, this pin is configured as CPUGNT#. The SIO's internal arbiter asserts CPUGNT# to indicate that the CPU initiator has been granted the PCI Bus. If the internal arbiter is disabled, this signal is meaningless. CPUGNT# is tri-stated from the leading edge of PCIRST#. CPUGNT# is tri-stated until driven by the SIO. CPUGNT# is driven based on the arbitration scheme and the asserted REQx#'s.
GNT0#/ SIOREQ#	t/s/o	GRANT 0/SIO REQUEST: If the SIO's internal arbiter is enabled, this pin is configured as GNT0#. The SIO's internal arbiter asserts GNT0# to indicate that Initiator0 has been granted the PCI Bus. If the internal arbiter is disabled, this pin is configured as SIOREQ#. The SIO asserts SIOREQ# to request the PCI Bus. GNT0#/SIOREQ# is tri-stated from the leading edge of PCIRST#. GNT0#/SIOREQ# is tri-stated until driven by the SIO. GNT0#/SIOREQ# is driven based on the arbitration scheme and the asserted REQx#'s.
GNT1#/ RESUME#	t/s/o	**GRANT 1/RESUME:** If the SIO's internal arbiter is enabled, this pin is configured as GNT1#. The SIO's internal arbiter asserts GNT1# to indicate that Initiator1 has been granted the PCI Bus. If the internal arbiter is disabled, this pin is configured as RESUME#. The SIO asserts RESUME# to indicate that the conditions causing the SIO to retry the cycle has passed. GNT1#/RESUME# is tri-stated from the leading edge of PCIRST#. GNT1#/RESUME# is tri-stated until driven by the SIO. GNT1#/RESUME# is driven based on the arbitration scheme and the asserted REQx#'s.
REQ2#	I	**REQUEST 2:** This pin is an active low signal that indicates that Initiator2 desires the use of the PCI Bus. This signal has a weak internal pull-up resistor.

3.2 PCI Arbiter Signals (Continued)

Signal Name	Type	Description
REQ3#	I	**REQUEST 3:** This pin is an active low signal that indicates that Initiator3 desires the use of the PCI Bus. This signal has a weak internal pull-up resistor.
GNT2#	t/s/o	**GRANT 2:** This pin is configured as GNT2#. The SIO's internal arbiter asserts GNT2# to indicate that Initiator2 has been granted the PCI Bus. GNT2# is high upon reset.
GNT3#	t/s/o	**GRANT 3:** This pin is configured as GNT3#. The SIO's internal arbiter asserts GNT3# to indicate that Initiator3 has been granted the PCI Bus. GNT3# is high upon reset.
MEMREQ#	t/s/o	**MEMORY REQUEST:** If the SIO is configured in Guaranteed Access Time (GAT) Mode, MEMREQ# will be asserted when an ISA master or DMA is requesting the ISA Bus (along with FLSHREQ#) to indicate that the SIO requires ownership of the main memory. MEMREQ# is tri-stated from the leading edge of PCIRST#. MEMREQ# remains tri-stated until driven by the SIO. After PCIRST, MEMREQ# is driven inactive asynchronously from PCIRST# inactive. The SIO asserts FLSHREQ# concurrently with asserting MEMREQ#. **FLSHREQ# / MEMREQ# / Meaning** 1 / 1 / Idle 0 / 1 / Flush buffers pointing towards PCI to avoid ISA deadlock 1 / 0 / Reserved 0 / 0 / GAT mode. Guarantee PCI Bus immediate access to main memory (this may or may not require the PCI-to-main memory buffers to be flushed first depending on the number of buffers).
FLSHREQ#	t/s/o	**FLUSH REQUEST:** FLSHREQ# is generated by the SIO to command all of the system's posted write buffers pointing towards the PCI Bus to be flushed. This is required before granting the ISA Bus to an ISA master or the DMA. FLSHREQ# is tri-stated from the leading edge of PCIRST#. FLSHREQ# remains tri-stated until driven by the SIO. After PCIRST, FLSHREQ# is driven inactive asynchronously from PCIRST# inactive.
MEMACK#	I	**MEMORY ACKNOWLEDGE:** MEMACK# is the response handshake that indicates to the SIO that the function requested over the MEMREQ# and/or FLSHREQ# signals has been completed. In GAT mode (MEMREQ# and FLSHREQ# asserted), the main memory bus is dedicated to the PCI Bus and the system's posted write buffers pointing towards the PCI Bus have been flushed and are disabled. In non-GAT mode (FLSHREQ# asserted alone), this means the system's posted write buffers have been flushed and are disabled. In either case, the SIO can now grant the ISA Bus to the requester.

3.3 Address Decoder Signal

Signal Name	Type	Description
MEMCS#	O	**MEMORY CHIP SELECT:** MEMCS# is a programmable address decode signal provided to a Host CPU bridge. A CPU bridge can use MEMCS# to forward a PCI cycle to main memory behind the bridge. MEMCS# is driven one PCI clock after FRAME# is sampled active (address phase) and is valid for one clock cycle before going inactive. MEMCS# is high upon reset.

3.4 Power Management Signals

Signal Name	Type	Description
SMI#	O	**SYSTEM MANAGEMENT INTERRUPT:** SMI# is an active low output that is asserted by the SIO in response to one of many enableable hardware or software events. SMI# connects directly to the CPU. The SMI# signal is an asychronous input to the CPU. The CPU recognizes the falling edge of SMI# as the highest priority interrupt in the system. The CPU responds by entering SMM (System Management Mode). SMI# is deasserted during and following reset.
STPCLK#	O	**STOP CLOCK:** STPCLK# is an active low output that is asserted by the SIO in response to one of many enableable hardware or software events. STPCLK# connects directly to the CPU. The STPCLK# signal is an asynchronous input to the CPU. When the CPU samples STPCLK# asserted it responds by stopping its internal clock. STPCLK# is deasserted during and following reset.
EXTSMI#	I	**EXTERNAL SYSTEM MANAGEMENT INTERRUPT:** EXTSMI# is a falling edge triggered input to the SIO indicating that an external device is requesting the system to enter SMM mode. When enabled, a falling edge on EXTSMI# will result in the assertion of the SMI# signal to the CPU. EXTSMI# is an asychronous input to the SIO. However, when the setup and hold times are met, it is only required to be asserted for one PCICLK. Once deasserted, it must remain deasserted for at least four PCICLKs in order to allow the edge detect logic to reset. This pin includes a weak internal pull-up resistor.
INIT	I	**INIT:** INIT is an input to the SIO indicating that the CPU is actually being soft reset. It is connected to the INIT pin of the CPU. This pin includes a weak internal pull-up resistor.

3.5 ISA Interface Signals

Signal Name	Type	Description
AEN	O	**ADDRESS ENABLE:** AEN is asserted during DMA cycles to prevent I/O slaves from misinterpreting DMA cycles as valid I/O cycles. When negated, AEN indicates that an I/O slave may respond to address and I/O commands. When asserted, AEN informs I/O resources on the ISA Bus that a DMA transfer is occurring. This signal is also driven high during refresh cycles. AEN is driven low upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
BALE	O	**BUS ADDRESS LATCH ENABLE:** BALE is an active high signal asserted by the SIO to indicate that the address (SA[19:0], LA[23:17]), AEN and SBHE# signal lines are valid. The LA[23:17] address lines are latched on the trailing edge of BALE. BALE remains asserted throughout DMA and ISA master cycles. BALE is driven low upon reset.
SYSCLK	O	**SYSTEM CLOCK:** SYSCLK is an output of the SIO component. The frequencies supported are 6 MHz to 8.33 MHz.
IOCHRDY	I/O	**I/O CHANNEL READY:** Resources on the ISA Bus assert IOCHRDY to indicate that additional time (wait-states) is required to complete the cycle. This signal is normally high on the ISA Bus. IOCHRDY is an input when the SIO owns the ISA Bus and a PCI agent is accessing an ISA slave or during compatible DMA transfers (compatible cycles only). IOCHRDY is output when an external ISA Bus Master owns the ISA Bus and is accessing a PCI slave or an SIO register. As an SIO output, IOCHRDY is driven inactive (low) from the falling edge of the ISA commands. After data is available for an ISA master read or the SIO latches the data for a write cycle, IOCHRDY is asserted for 70 ns. After 70 ns, the SIO floats IOCHRDY. The 70 ns includes both the drive time and the time it takes the SIO to float IOCHRDY. The SIO does not drive this signal when an ISA Bus master is accessing an ISA Bus slave. IOCHRDY is tri-stated upon reset.
IOCS16#	I	**16-BIT I/O CHIP SELECT:** This signal is driven by I/O devices on the ISA Bus to indicate that they support 16-bit I/O bus cycles.
IOCHK#	I	**I/O CHANNEL CHECK:** IOCHK# can be driven by any resource on the ISA Bus. When asserted, it indicates that a parity or an un-correctable error has occurred for a device or memory on the ISA Bus. A NMI will be generated to the CPU if the NMI generation is enabled.
IOR#	I/O	**I/O READ:** IOR# is the command to an ISA I/O slave device that the slave may drive data on to the ISA data bus (SD[15:0]). The I/O slave device must hold the data valid until after IOR# is negated. IOR# is an output when the SIO owns the ISA Bus. IOR# is an input when an external ISA master owns the ISA Bus. IOR# is driven high upon reset.
IOW#	I/O	**I/O WRITE:** IOW# is the command to an ISA I/O slave device that the slave may latch data from the ISA data bus (SD[15:0]). IOW# is an output when the SIO owns the ISA Bus. IOW# is an input when an external ISA master owns the ISA Bus. IOW# is driven high upon reset.
LA[23:17]	I/O	**UNLATCHED ADDRESS:** The LA[23:17] address lines are bi-directional. These address lines allow accesses to physical memory on the ISA Bus up to 16 MBytes. LA[23:17] are outputs when the SIO owns the ISA Bus. The LA[23:17] lines become inputs whenever an ISA master owns the ISA Bus. These signals are undefined during DMA type "A", "B", and "F" cycles. The LA[23:17] signals are at an unknown state upon reset.
SA[19:0]	I/O	**SYSTEM ADDRESS BUS:** These bi-directional address lines define the selection with the granularity of one byte within the one Megabyte section of memory defined by the LA[23:17] address lines. The address lines SA[19:17] that are coincident with LA[19:17] are defined to have the same values as LA[19:17] for all memory cycles. For I/O accesses, only SA[15:0] are used. SA[19:0] are outputs when the SIO owns the ISA Bus. SA[19:0] are inputs when an external ISA Master owns the ISA Bus. SA[19:0] are undefined during DMA type "A", 'B", or "F" cycles. SA[19:0] are at an unknown state upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
SBHE#	I/O	**SYSTEM BYTE HIGH ENABLE:** SBHE# indicates, when asserted, that a byte is being transferred on the upper byte (SD[15:8]) of the data bus. SBHE# is negated during refresh cycles. SBHE# is an output when the SIO owns the ISA Bus. SBHE# is an input when an external ISA master owns the ISA Bus. SBHE# is at an unknown state upon reset.
MEMCS16#	OD	**MEMORY CHIP SELECT 16:** MEMCS16# is a decode of LA[23:17] without any qualification of the command signal lines. ISA slaves that are 16-bit memory devices drive this signal low. The SIO ignores MEMCS16# during I/O access cycles and refresh cycles. During DMA cycles, this signal is only used by the byte swap logic. MEMCS16# is an input when the SIO owns the ISA Bus. MEMCS16# is an output when an ISA Bus master owns the ISA Bus. The SIO drives this signal low during ISA master to PCI memory cycles. MEMCS16# is at an unknown state upon reset.
MASTER#	I	**MASTER:** An ISA Bus master asserts MASTER# to indicate that it has control of the ISA Bus. Before the ISA master can assert MASTER#, it must first sample DACK# active. Once MASTER# is asserted, the ISA master has control of the ISA Bus until it negates MASTER#.
MEMR#	I/O	**MEMORY READ:** MEMR# is the command to a memory slave that it may drive data onto the ISA data bus. MEMR# is an output when the SIO is a master on the ISA Bus. MEMR# is an input when an ISA master, other than the SIO, owns the ISA Bus. This signal is also driven by the SIO during refresh cycles. For compatible timing mode DMA cycles, the SIO, as a master, asserts MEMR# if the address is less than 16 MBytes. This signal is not generated for accesses to addresses greater than 16 MByte. MEMR# is not driven active during DMA type "A", "B", or "F" cycles.
MEMW#	I/O	**MEMORY WRITE:** MEMW# is the command to a memory slave that it may latch data from the ISA data bus. MEMW# is an output when the SIO owns the ISA Bus. MEMW# is an input when an ISA master, other than the SIO, owns the ISA Bus. For compatible timing mode DMA cycles, the SIO, as a master, asserts MEMW# if the address is less than 16 MBytes. This signal is not generated for accesses to addresses greater than 16 MByte. MEMW# is not driven active during DMA type "A", "B", or "F" cycles.
SMEMW#	O	**SYSTEM MEMORY WRITE:** The SIO asserts SMEMW# to request a memory slave to accept data from the data lines. If the access is below the 1 MByte range (00000000h–000FFFFFh) during DMA compatible, SIO master, or ISA master cycles, the SIO asserts SMEMW#. SMEMW# is a delayed version of MEMW#. SMEMW# is driven high upon reset.
SMEMR#	O	**SYSTEM MEMORY READ:** The SIO asserts SMEMR# to request a memory slave to accept data from the data lines. If the access is below the 1 MByte range (00000000h–000FFFFFh) during DMA compatible, SIO master, or ISA master cycles, the SIO asserts SMEMR#. SMEMR# is a delay version of MEMR#. Upon PCIRST# this signal is low. SMEMR# is driven high upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
ZEROWS#	I	**ZERO WAIT-STATES:**An ISA slave asserts ZEROWS# after its address and command signals have been decoded to indicate that the current cycle can be shortened. A 16-bit ISA memory cycle can be reduced to two SYSCLKs. An 8-bit memory or I/O cycle can be reduced to three SYSCLKs. ZEROWS# has no effect during 16-bit I/O cycles. If IOCHRDY and ZEROWS# are both asserted during the same clock, then ZEROWS# is ignored and wait states are added as a function of IOCHRDY (i.e., IOCHRDY has precedence over ZEROWS#).
OSC	I	**OSCILLATOR:** OSC is the 14.31818 MHz ISA clock signal. It is used by the internal 8254 Timer, counters 0, 1, and 2.
RSTDRV	O	**RESET DRIVE:** The SIO asserts RSTDRV to reset devices that reside on the ISA Bus. The SIO asserts this signal when PCIRST# (PCI Reset) is asserted. In addition, the SIO can be programmed to assert RSTDRV by writing to the ISA Clock Divisor Register. Software should assert the RSTDRV during configuration to reset the ISA Bus when changing the clock divisor. Note that when RSTDRV is generated via the ISA Clock Divisor Register, software must ensure that RSTDRV is driven active for a minimum of 1 μs.
SD[15:0]	I/O	**System DATA:** SD[15:0] provide the 16-bit data path for devices residing on the ISA Bus. SD[15:8] correspond to the high order byte and SD[7:0] correspond to the low order byte. SD[15:0] are undefined during refresh. The SIO tri-states SD[15:0] during reset.

3.6 DMA Signals

Signal Name	Type	Description
DREQ [3:0,7:5]	I	**DMA REQUEST:** The DREQ lines are used to request DMA service from the SIO's DMA controller or for a 16-bit master to gain control of the ISA expansion bus. The active level (high or low) is programmed via the DMA Command Register (bit 6). When the bit 6 = 0, DREQ[3:0,7:5] are active high and when bit 6 = 1, the signals are active low. All inactive to active edges of DREQ are assumed to be asynchronous. The request must remain active until the appropriate DACK signal is asserted.
DACK# [3:0,7:5]	O	**DMA ACKNOWLEDGE:** The DACK output lines indicate that a request for DMA service has been granted by the SIO or that a 16-bit master has been granted the bus. The active level (high or low) is programmed via the DMA Command Register (bit 7). When bit 7 = 0, DACK#[3:0,7:5] are active low and when bit 7 = 1, the signals are active high. These lines should be used to decode the DMA slave device with the IOR# or IOW# line to indicate selection. If used to signal acceptance of a bus master request, this signal indicates when it is legal to assert MASTER#. If the DMA controller has been programmed for a timing mode other than compatible mode, and another device has requested the bus, and a 4 μs time has elapsed, this line will be negated and the transfer stopped before the transfer is complete. In this case, the transfer is re-started at the next arbitration period that the channel wins the bus. Upon PCIRST#, these lines are set inactive (high).

3.6 DMA Signals (Continued)

Signal Name	Type	Description
EOP	I/O	**END OF PROCESS:** EOP is bi-directional, acting in one of two modes, and is directly connected to the TC line of the ISA Bus. DMA slaves assert EOP to the SIO to terminate DMA cycles. The SIO asserts EOP to DMA slaves as a terminal count indicator. **EOP-IN MODE:** For all transfer types during DMA, the SIO samples EOP. If it is sampled asserted, the transfer is terminated. **TC-OUT MODE:** The SIO asserts EOP after a new address has been output, if the byte count expires with that transfer. The EOP (TC) remains asserted until AEN is negated, unless AEN is negated during an autoinitialization. EOP (TC) is negated before AEN is negated during an autoinitialization. When all the DMA channels are not in use, the EOP signal is in output mode and negated (low). After PCIRST#, EOP is in output mode and inactive.
REFRESH#	I/O	**REFRESH:** As an output, REFRESH# is used by the SIO to indicate when a refresh cycle is in progress. It should be used to enable the SA[15:0] address to the row address inputs of all banks of dynamic memory on the ISA Bus. Thus, when MEMR# is asserted, the entire expansion bus dynamic memory is refreshed. Memory slaves must not drive any data onto the bus during refresh. As an output, this signal is driven directly onto the ISA Bus. This signal is an output only when the SIO DMA refresh is a master on the bus responding to an internally generated request for refresh. As an input, REFRESH# is driven by 16-bit ISA Bus masters to initiate refresh cycles. Upon PCIRST#, this signal is tri-stated.

3.7 Timer Signal

Signal Name	Type	Description
SPKR	O	**SPEAKER DRIVE:** The SPKR signal is the output of counter 2 and is "ANDed" with Port 061h bit 1 to provide Speaker Data Enable. This signal drives an external speaker driver device, which in turn drives the ISA system speaker. SPKR has a 24 mA drive capability. Upon reset, its output state is 0.

3.8 Interrupt Controller Signals

Signal Name	Type	Description
IRQ[15,14, 11:9, 7:3,1]	I	**INTERRUPT REQUEST:** The IRQ signals provide both system board components and ISA Bus I/O devices with a mechanism for asynchronously interrupting the CPU. The assertion mode of these inputs depends on the programming of LTIM, bit 3 of ICW1 on both Controller-1 and Controller-2. When LTIM is programmed to a 0, a low-to-high transition on any of that controller's IRQ lines is recognized as an interrupt request. This is "edge-triggered" mode. Edge-triggered mode is the SIO default. When LTIM is programmed to a 1, a high level on any of that controller's IRQ lines is recognized as an interrupt request. This mode is "level-triggered" mode. Upon PCIRST#, the IRQ lines are placed in edge-triggered mode. An active IRQ input must remain asserted until after the interrupt is acknowledged. If the input goes inactive before this time, a DEFAULT IRQ7 occurs when the CPU acknowledges the interrupt. **NOTE:** Refer to the Utility Bus Signal descriptions for IRQ12 and IRQ13 signal descriptions.
IRQ8#	I	**INTERRUPT REQUEST EIGHT SIGNAL:** IRQ8# is an active low interrupt input. The assertion mode of this input depends on the programming of the LTIM bit of ICW1 on both Controller-1 and Controller-2. When the LTIM = 0, a high-to-low transition on IRQ8# is recognized as an interrupt request. This is "edge-triggered" mode. Edge triggered mode is the SIO default. When the LTIM = 1, a low level on IRQ8# is recognized as an interrupt request. This mode is "level-triggered" mode. Upon PCIRST#, IRQ8# will be placed in edge-triggered mode. IRQ8# must remain asserted until after the interrupt is acknowledged. If the input goes inactive before this time, a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt.
INT	O	**CPU INTERRUPT:** INT is driven by the SIO to signal the CPU that an interrupt request is pending and needs to be serviced. It is asynchronous with respect to SYSCLK or PCICLK and is always an output. The interrupt controller must be programmed following a reset to ensure that INT is at a known state. Upon PCIRST#, INT is driven low.
NMI	O	**NON-MASKABLE INTERRUPT:** NMI is used to force a non-maskable interrupt to the CPU. The SIO generates an NMI when either SERR# or IOCHK# is asserted, depending on how the NMI Status and Control Register is programmed. The CPU detects an NMI when it detects a rising edge on NMI. After the NMI interrupt routine processes the interrupt, the NMI status bits in the NMI Status and Control Register are cleared by software. The NMI interrupt routine must read this register to determine the source of the interrupt. The NMI is reset by setting the corresponding NMI source enable/disable bit in the NMI Status and Control Register. To enable NMI interrupts, the two NMI enable/disable bits in the register must be set to 0, and the NMI mask bit in the NMI Enable/Disable and Real-Time Clock Address Register must be set to 0. Upon PCIRST#, this signal is driven low.

3.9 Utility Bus Signals

Signal Name	Type	Description
UBUSTR	O	**UTILITY DATA BUS TRANSMIT/RECEIVE:** UBUSTR is tied directly to the direction control of a 74F245 that buffers the utility data bus, UD[7:0]. UBUSTR is asserted for all I/O read cycles (regardless if a Utility Bus device has been decoded). UBUSTR is asserted for memory cycles only if BIOS space has been decoded. For PCI and ISA master-initiated read cycles, UBUSTR is asserted from the falling edge of either IOR# or MEMR#, depending on the cycle type (driven from MEMR# only if BIOS space has been decoded). When the rising edge of IOR# or MEMR# occurs, the SIO negates UBUSTR. For DMA read cycles from the Utility Bus, UBUSTR is asserted when DACKx# is asserted and negated when DACKx# is negated. At all other times, UBUSTR is negated. Upon PCIRST#, this signal is driven low.
UBUSOE#	O	**UTILITY DATA BUS OUTPUT ENABLE:** UBUSOE# is tied directly to the output enable of a 74F245 that buffers the utility data bus, UD[7:0], from the system data bus, SD[7:0]. UBUSOE# is asserted anytime a SIO supported Utility Bus device is decoded, and the devices decode is enabled in the Utility Bus Chip Select Enable Registers. UBUSOE# is asserted from the falling edge of the ISA commands (IOR#, IOW#, MEMR#, or MEMW#) for PCI and ISA master-initiated cycles. UBUSOE# is negated from the rising edge of the ISA command signals for SIO-initiated cycles and the SA[16:0] and LA[23:17] address for ISA master-initiated cycles. For DMA cycles, UBUSOE# is asserted when DACK2# is asserted and negated when DACK2# negated. UBUSOE# is not driven active under the following conditions: **NOTES:** 1. During an I/O access to the floppy controller, if DSKCHG is sampled low at reset. 2. If the Digital Output Register is programmed to ignore DACK2#. 3. During an I/O read access to floppy location 3F7h (primary) or 377h (secondary), if the IDE decode space is disabled (i.e. IDE is not resident on the Utility Bus). 4. During any access to a utility bus peripheral in which its decode space has been disabled. Upon a PCIRST#, this signal is driven inactive (high).
ECSADDR [2:0]	O	**ENCODED CHIP SELECTS:** ECSADDR[2:0] are the encoded chip selects and/or control signals for the Utility Bus peripherals supported by the SIO. The binary code formed by the three signals indicates which Utility Bus device is selected. These signals tie to the address inputs of two external 74F138 decoder chips and are driven valid/invalid from the SA[16:0] and LA[23:17] address lines. Upon PCIRST#, these signals are driven high.
ECSEN#	O	**ENCODED CHIP SELECT ENABLE:** ECSEN# is used to determine which of the two external 74F138 decoders is to be selected. ECSEN# is driven low to select decoder 1 and driven high to select decoder 2. This signal is driven valid/invalid from the SA[16:0] and LA[23:17] address lines (except for the generation of RTCALE#, in which case, ECSEN# is driven active based on IOW# falling, and remains active for two SYSCLKs). During a non-valid address or during an access not targeted for the Utility Bus, this signal is driven high. Upon PCIRST#, this signal is driven high.

3.9 Utility Bus Signals (Continued)

Signal Name	Type	Description
ALT_RST#	O	**ALTERNATE RESET:** ALT_RST# is used to reset the CPU under program control. This signal is AND'ed together externally with the reset signal (KBDRST#) from the keyboard controller to provide a software means of resetting the CPU. This provides a faster means of reset than is provided by the keyboard controller. Writing a 1 to bit 0 in the Port 92 Register causes this signal to pulse low for approximately 4 SYSCLKs. Before another ALT_RST# pulse can be generated, bit 0 must be set to 0. Upon PCIRST#, this signal is driven inactive high (bit 0 in the Port 92 Register is set to 0).
ALT_A20	O	**ALTERNATE A20:** ALT_A20 is used to force A20M# to the CPU low for support of real mode compatible software. This signal is externally OR'ed with the A20GATE signal from the keyboard controller and CPURST to control the A20M# input of the CPU. Writing a 0 to bit 1 of the Port 92 Register forces ALT_A20 low. ALT_A20 low drives A20M# to the CPU low, if A20GATE from the keyboard controller is also low. Writing a 1 to bit 1 of the Port 92 Register force ALT_A20 high. ALT_A20 high drives A20M# to the CPU high, regardless of the state of A20GATE from the keyboard controller. Upon reset, this signal is driven low.
DSKCHG	I	**DISK CHANGE:** DSKCHG is tied directly to the DSKCHG signal of the floppy controller. This signal is inverted and driven on SD7 during I/O read cycles to floppy address locations 3F7h (primary) or 377h (secondary) as shown in the table below. Note that the primary and secondary locations are programmed in the Utility Bus Address Decode Enable/Disable Register "A". **FLOPPYCS# Decode** / **IDECSx# Decode** / **State of SD7 (output)** / **State of UBUSOE#** Enabled / Enabled / Tri-stated / Enabled Enabled / Disabled / Driven via DSKCHG / Disabled Disabled / Enabled / Tri-stated / Enabled (note) Disabled / Disabled / Tri-stated / Disabled **NOTE:** For this mode to be supported, extra logic is required to disable the U-bus transceiver for accesses to 3F7/377. This is necessary because of potential contention between the Utility Bus buffer and a floppy on the ISA Bus driving the system bus at the same time during shared I/O accesses. This signal is also used to determine if the floppy controller is present on the Utility Bus. It is sampled on the trailing edge of PCIRST#, and if high, the Floppy is present. For systems that do not support a Floppy via the SIO, this pin should be strapped low. If sampled low, the SD7 function, UBUSOE#, and ECSADDR[2:0] signals will not be enabled for DMA or programmed I/O accesses to the floppy disk controller. This condition overrides the floppy decode enable bits in the Utility Bus Chip Select A.

3.9 Utility Bus Signals (Continued)

Signal Name	Type	Description
FERR#/ IRQ13	I	**NUMERIC COPROCESSOR ERROR/IRQ13:** This signal has two separate functions, depending on bit 5 in the ISA Clock Divisor Register. This pin functions as a FERR# signal supporting coprocessor errors, if this function is enabled (bit 5 = 1), or as an external IRQ13, if the coprocessor error function is disabled (bit 5 = 0). If programmed to support coprocessor error reporting, this signal is tied to the coprocessor error signal on the CPU. If FERR# is asserted by the coprocessor inside the CPU, the SIO generates an internal IRQ13 to its interrupt controller unit. The SIO then asserts the INT output to the CPU. Also, in this mode, FERR# gates the IGNNE# signal to ensure that IGNNE# is not asserted to the CPU unless FERR# is active. When FERR# is asserted, the SIO asserts INT to the CPU as an IRQ13. IRQ13 continues to be asserted until a write to F0h has been detected. If the Coprocessor error reporting is disabled, FERR# can be used by the system as IRQ13. Upon PCIRST#, this signal provides the standard IRQ13 function. This signal should be pulled high with an external 8.2 KΩ pull-up resistor if the IRQ13 mode is used or the pin is left floating.
IGNNE#	O	**IGNORE ERROR:** This signal is connected to the ignore error pin of the CPU. IGNNE# is only used if the SIO coprocessor error reporting function is enabled in the ISA Clock Divisor Register (bit 5 = 1). If FERR# is active, indicating a coprocessor error, a write to the Coprocessor Error Register (F0h) causes the IGNNE# to be asserted. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted when the Coprocessor Error Register is written, the IGNNE# is not asserted. IGNNE# is driven high upon a reset.
IRQ12/M	I	**INTERRUPT REQUEST/MOUSE INTERRUPT:** In addition to providing the standard interrupt function as described in the pin description for IRQ[15,14, 11:9, 7:3, 1], this pin also provides a mouse interrupt function. Bit 4 in the ISA Clock Divisor Register determines the functionality of IRQ12/M. When bit 4 = 0, the standard interrupt function is provided and this pin can be tied to the ISA connector. When bit 4 = 1, the mouse interrupt function is provided and this pin can be tied to the DIRQ12 output of the keyboard controller. When the mouse interrupt function is selected, a low to high transition on this signal is latched by the SIO and an INT is generated to the CPU as IRQ12. An interrupt will continue to be generated until a PCIRST# or an I/O read access to address 60h (falling edge of IOR#) is detected. After a PCIRST#, this pin provides the standard IRQ12 function.

3.10 Test Signals

Signal Name	Type	Description
TEST	I	**TEST:** The TEST signal is used to tri-state all of the SIO outputs. During normal operation, this input should be tied to ground.
TESTO	O	**TEST OUTPUT:** This is the output pin used during NAND tree testing.

4.0 REGISTER DESCRIPTION

The SIO contains both PCI configuration registers and non-configuration registers. The configuration registers (Table 3) are located in PCI configuration space and are only accessible from the PCI Bus. Addresses for configuration registers are offset values that appear on AD[7:2] and C/BE#[3:0]. The configuration registers (Section 4.1) can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities. All multi-byte numeric fields use "little-endian" ordering (i.e., lower addresses contain the least significant parts of the fields).

The non-configuration registers (Table 4) include DMA Registers (Section 4.2), Timer Registers (Section 4.3), Interrupt Controller Registers (Section 4.4), and Control Registers (Section 4.5). All of these registers are accessible from the PCI Bus. In addition, some of the registers are accessible from the ISA Bus. Table 4 indicates the bus access for each register. Except for the DMA scatter/gather registers and the BIOS timer registers, the non-configuration registers can only be accessed as byte quantities. If a PCI master attempts a multi-byte access (i.e., more than one Byte Enable signal asserted), the SIO responds with a target-abort. The scatter/gather registers and BIOS timer registers can be accessed as Byte, Word, or Dword quantities.

Some of the SIO configuration and non-configuration registers contain reserved bits. These bits are labeled "Reserved". Software must take care to deal correctly with bit-encoded fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions, and the data then written back.

In addition to reserved bits within a register, the SIO contains address locations in the PCI configuration space that are marked "Reserved" (Table 3). The SIO responds to accesses to these address locations by completing the PCI cycle. However, reads of reserved address locations yield all zeroes and writes have no affect on the SIO.

The SIO, upon receiving a hard reset (PCIRST# signal), sets its internal registers to pre-determined **default** states. The default values are indicated in the individual register descriptions.

Table 3. Configuration Registers

Configuration Offset	Register	Register Access	Bus Access
00h–01h	Vendor Identification	RO	PCI Only
02h–03h	Device Identification	RO	PCI Only
04h–05h	Command	R/W	PCI Only
06h–07h	Device Status	R/W	PCI Only
08h	Revision Identification	RO	PCI Only
09h–3Fh	Reserved	—	PCI Only
40h	PCI Control	R/W	PCI Only
41h	PCI Arbiter Control	R/W	PCI Only
42h	PCI Arbiter Priority Control	R/W	PCI Only
43h	PCI Arbiter Priority Control Extension Register	R/W	PCI Only
44h	MEMCS# Control	R/W	PCI Only
45h	MEMCS# Bottom of Hole	R/W	PCI Only
46h	MEMCS# Top of Hole	R/W	PCI Only
47h	MEMCS# Top of Memory	R/W	PCI Only
48h	ISA Address Decoder Control	R/W	PCI Only
49h	ISA Address Decoder ROM Block Enable	R/W	PCI Only
4Ah	ISA Address Decoder Bottom of Hole	R/W	PCI Only
4Bh	ISA Address Decoder Top of Hole	R/W	PCI Only
4Ch	ISA Controller Recovery Timer	R/W	PCI Only
4Dh	ISA Clock Divisor	R/W	PCI Only
4Eh	Utility Bus Chip Select Enable A	R/W	PCI Only
4Fh	Utility Bus Chip Select Enable B	R/W	PCI Only
50h–53h	Reserved	—	PCI Only
54h	MEMCS# Attribute Register #1	R/W	PCI Only
55h	MEMCS# Attribute Register #2	R/W	PCI Only
56h	MEMCS# Attribute Register #3	R/W	PCI Only
57h	Scatter/Gather Relocation Base Address	R/W	PCI Only
58h–5Fh	Reserved	—	PCI Only

Table 3. Configuration Registers (Continued)

Configuration Offset	Register	Register Access	Bus Access
60h	PIRQ0# Route Control	R/W	PCI Only
61h	PIRQ1# Route Control	R/W	PCI Only
62h	PIRQ2# Route Control	R/W	PCI Only
63h	PIRQ3# Route Control	R/W	PCI Only
64h–7Fh	Reserved	—	PCI Only
80h–81h	BIOS Timer Base Address	R/W	PCI Only
82h–9Fh	Reserved	—	PCI Only
A0h	SMI Control (SMICNTL)	R/W	PCI Only
A1h	Reserved	—	PCI Only
A2h–A3h	SMI Enable (SMIEN)	R/W	PCI Only
A4h–A7h	System Event Enable (SEE)	R/W	PCI Only
A8h	Fast Off Timer (FTMR)	R/W	PCI Only
A9h	Reserved	—	PCI Only
AAh–ABh	SMI Request (SMIREQ)	R/W	PCI Only
ACh	Clock Throttle STPCLK# Low Timer (CTLTMRL)	R/W	PCI Only
ADh	Reserved	—	PCI Only
AEh	Clock Throttle STPCLK# High Timer (CTLTMRH)	R/W	PCI Only
AFh–FFh	Reserved	—	PCI Only

Table 4. Non-Configuration Registers

Address	Function Unit	Register	Register Access	Bus Access
0000h	DMA	DMA1 CH0 Base and Current Address	R/W	PCI Only
0001h	DMA	DMA1 CH0 Base and Current Count	R/W	PCI Only
0002h	DMA	DMA1 CH1 Base and Current Address	R/W	PCI Only
0003h	DMA	DMA1 CH1 Base and Current Count	R/W	PCI Only
0004h	DMA	DMA1 CH2 Base and Current Address	R/W	PCI Only
0005h	DMA	DMA1 CH2 Base and Current Count	R/W	PCI Only
0006h	DMA	DMA1 CH3 Base and Current Address	R/W	PCI Only
0007h	DMA	DMA1 CH3 Base and Current Count	R/W	PCI Only
0008h	DMA	DMA1 Status(R) Command(W)	R/W	PCI Only
0009h	DMA	DMA1 Write Request	WO	PCI Only
000Ah	DMA	DMA1 Write Single Mask Bit	WO	PCI Only
000Bh	DMA	DMA1 Write Mode	WO	PCI Only
000Ch	DMA	DMA1 Clear Byte Pointer	WO	PCI Only
000Dh	DMA	DMA1 Master Clear	WO	PCI Only
000Eh	DMA	DMA1 Clear Mask	WO	PCI Only
000Fh	DMA	DMA1 Read/Write All Mask Register Bits	R/W	PCI Only
0020h	Interrupt	INT 1 Control	R/W	PCI/ISA
0021h	Interrupt	INT 1 Mask	R/W	PCI/ISA
0040h	Timer	Timer Counter 1–Counter 0 Count	R/W	PCI/ISA
0041h	Timer	Timer Counter 1–Counter 1 Count	R/W	PCI/ISA
0042h	Timer	Timer Counter 1–Counter 2 Count	R/W	PCI/ISA
0043h	Timer	Timer Counter 1 Command Mode	WO	PCI/ISA
0060h(2)	Control	Reset UBus IRQ12	RO	PCI/ISA
0061h	Control	NMI Status and Control	R/W	PCI/ISA
0070h(2)	Control	CMOS RAM Address and NMI Mask	WO	PCI/ISA
0078h-007Bh(3,4,5)	Timer	BIOS Timer	R/W	PCI Only
0080h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0081h	DMA	DMA Channel 2 Page Register	R/W	PCI/ISA
0082h	DMA	DMA Channel 3 Page Register	R/W	PCI/ISA
0083h	DMA	DMA Channel 1 Page Register	R/W	PCI/ISA

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
0084h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0085h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0086h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0087h	DMA	DMA Channel 0 Page Register	R/W	PCI/ISA
0088h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0089h	DMA	DMA Channel 6 Page Register	R/W	PCI/ISA
008Ah	DMA	DMA Channel 7 Page Register	R/W	PCI/ISA
008Bh	DMA	DMA Channel 5 Page Register	R/W	PCI/ISA
008Ch(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Dh(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Eh(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Fh	DMA	DMA Low Page Register Refresh	R/W	PCI/ISA
0090h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0092h(2)	Control	Port 92 Register	R/W	PCI/ISA
0094h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0095h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0096h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0098h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Ch(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Dh(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Eh(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Fh	DMA	DMA Low Page Register Refresh	R/W	PCI/ISA
00A0h	Interrupt	INT 2 Control Register	R/W	PCI/ISA
00A1h	Interrupt	INT 2 Mask Register	R/W	PCI/ISA
00B2h	P.M.	Advanced Power Management Control Port	R/W	PCI Only
00B3h	P.M.	Advanced Power Management Status Port	R/W	PCI Only
00C0h	DMA	DMA2 CH0 Base and Current Address	R/W	PCI Only
00C2h	DMA	DMA2 CH0 Base and Current Count	R/W	PCI Only
00C4h	DMA	DMA2 CH1 Base and Current Address	R/W	PCI Only
00C6h	DMA	DMA2 CH1 Base and Current Count	R/W	PCI Only
00C8h	DMA	DMA2 CH2 Base and Current Address	R/W	PCI Only

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
00CAh	DMA	DMA2 CH2 Base and Current Count	R/W	PCI Only
00CCh	DMA	DMA2 CH3 Base and Current Address	R/W	PCI Only
00CEh	DMA	DMA2 CH3 Base and Current Count	R/W	PCI Only
00D0h	DMA	DMA2 Status(r) Command(w) Register	R/W	PCI Only
00D2h	DMA	DMA2 Write Request Register	WO	PCI Only
00D4h	DMA	DMA2 Write Single Mask Bit Register	WO	PCI Only
00D6h	DMA	DMA2 Write Mode Register	WO	PCI Only
00D8h	DMA	DMA2 Clear Byte Pointer Register	WO	PCI Only
00DAh	DMA	DMA2 Master Clear Register	WO	PCI Only
00DCh	DMA	DMA2 Clear Mask Register	WO	PCI Only
00DEh	DMA	DMA2 Read/Write All Mask Register Bits	R/W	PCI Only
00F0h(2)	Control	Coprocessor Error Register	WO	PCI/ISA
0372h(2)	Control	Secondary Floppy Disk Digital Output Register	WO	PCI/ISA
03F2h(2)	Control	Primary Floppy Disk Digital Output Register	WO	PCI/ISA
040Ah(3)	DMA	Scatter/Gather Interrupt Status Register	RO	PCI Only
040Bh	DMA	DMA1 Extended Mode Register	WO	PCI/ISA
0410h(3,4)	DMA	CH0 Scatter/Gather Command	WO	PCI Only
0411h(3,4)	DMA	CH1 Scatter/Gather Command	WO	PCI Only
0412h(3,4)	DMA	CH2 Scatter/Gather Command	WO	PCI Only
0413h(3,4)	DMA	CH3 Scatter/Gather Command	WO	PCI Only
0415h(3,4)	DMA	CH5 Scatter/Gather Command	WO	PCI Only
0416h(3,4)	DMA	CH6 Scatter/Gather Command	WO	PCI Only
0417h(3,4)	DMA	CH7 Scatter/Gather Command	WO	PCI Only
0418h(3,4)	DMA	CH0 Scatter/Gather Status	RO	PCI Only
0419h(3,4)	DMA	CH1 Scatter/Gather Status	RO	PCI Only
041Ah(3,4)	DMA	CH2 Scatter/Gather Status	RO	PCI Only
041Bh(3,4)	DMA	CH3 Scatter/Gather Status	RO	PCI Only
041Dh(3,4)	DMA	CH5 Scatter/Gather Status	RO	PCI Only
041Eh(3,4)	DMA	CH6 Scatter/Gather Status	RO	PCI Only
041Fh(3,4)	DMA	CH7 Scatter/Gather Status	RO	PCI Only
0420h–0423h(3,4)	DMA	CH0 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
0424h–0427h(3,4)	DMA	CH1 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0428h–042Bh(3,4)	DMA	CH2 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
042Ch–042Fh(3,4)	DMA	CH3 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0434h–0437h(3,4)	DMA	CH5 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0438h–043Bh(3,4)	DMA	CH6 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
043Ch–043Fh(3,4)	DMA	CH7 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0481h	DMA	DMA CH2 High Page Register	R/W	PCI/ISA
0482h	DMA	DMA CH3 High Page Register	R/W	PCI/ISA
0483h	DMA	DMA CH1 High Page Register	R/W	PCI/ISA
0487h	DMA	DMA CH0 High Page Register	R/W	PCI/ISA
0489h	DMA	DMA CH6 High Page Register	R/W	PCI/ISA
048Ah	DMA	DMA CH7 High Page Register	R/W	PCI/ISA
048Bh	DMA	DMA CH5 High Page Register	R/W	PCI/ISA
04D0h	Interrupt	Edge/Level Control Register—INT CNTRL 1	R/W	PCI Only
04D1h	Interrupt	Edge/Level Control Register—INT CNTRL 2	R/W	PCI Only
04D6h	DMA	DMA2 Extended Mode Register	WO	PCI/ISA

NOTES:
1. PCI write cycles to these address locations flow through to the ISA Bus. PCI read cycles to these address locations do not flow through to the ISA Bus.
2. PCI read and write cycles to these address locations flow through to the ISA Bus.
3. The I/O address of this register is relocatable. The value shown in this table is the default address location.
4. This register can be accessed as a Byte, Word, or Dword quantity.
5. If this register location is enabled, PCI accesses to the BIOS Timer Register do not flow through to the ISA Bus. If disabled, accesses to this address location flow through to the ISA Bus.
6. When the DMAAC bit in the PCI Control Register is '0', the 82378 will alias I/O accesses in the 80h–8Fh range to the 90h–9Fh range. Write accesses to these address locations flow through to the ISA Bus. Read cycles to these address locations do not flow through to the ISA Bus. When DMAAC = 1, the SIO will only respond to the 80h-8Fh range and read and write accesses to these addresses in the 90h–9Fh range will be forwarded from the PCI bus to the ISA Bus (I/O port 92h is always a distinct register in the 90h–9Fh range and is always fully decoded, regardless of the setting of the DMAAC bit).

4.1 SIO Configuration Register Description

This section describes the SIO configuration registers. These registers include the Mandatory Header Registers (located in the first 64 bytes of configuration space) and the SIO specific registers (located from configuration offset 40h–56h).

4.1.1 VID—VENDOR IDENTIFICATION REGISTER

Address Offset: 00h, 01h
Default Value: 8086h
Attribute: Read Only
Size: 16 bits

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Bits[15:0]: Vendor Identification Number
This is a 16-bit value assigned to Intel.

4.1.2 DID—DEVICE IDENTIFICATION REGISTER

Address Offset: 02h, 03h
Default Value: 0484h
Attribute: Read Only
Size: 16 bits

The DID Register contains the device identification number. This register, along with the Vendor ID, uniquely identifies the SIO. Writes to this register have no effect.

Bits[15:0]: Device Identification Number
This is a 16-bit value assigned to the SIO.

4.1.3 COM—COMMAND REGISTER

Address Offset: 04h–05h
Default Value: 0007h
Attribute: Read/Write
Size: 16 bits

Bits[15:5]: Reserved
Read 0.

Bit 4: PMWE (Postable Memory Write Enable)
Enable Postable memory write, memory write and invalidate, and memory read Pre-fetch commands. The SIO does not support these commands as a master or slave so this bit is not implemented. This bit will always be read as a 0.

Bit 3: SCE (Special Cycle Enable)
When this bit is set to a "1", the SIO will recognize PCI Special Cycles. When set to "0", the SIO will ignore all PCI Special Cycles. This bit MUST be enabled in the 82378ZB if the STPCLK feature is being used.

Bit 2: BME (Bus Master Enable)
Since the SIO always requests the PCI Bus on behalf of ISA masters, DMA, or line buffer PCI requests, this bit is hardwired to a 1 and will always be read as a 1.

Bit 1: MSE (Memory Space Enable)
Enables SIO to accept a PCI-originated memory cycle. Since the SIO always responds to PCI-originated memory cycles (and ISA-bound cycles) by asserting DEVSEL#, this bit is hardwired to a 1 and will always be read as a 1.

Bit 0: IOSE (I/O Space Enable)
Enable SIO to accept a PCI-originated I/O cycle. Since the SIO always responds to a master I/O cycle, this bit is hardwired to a 1 and will always be read as a 1.

4.1.4 DS—DEVICE STATUS REGISTER

Address Offset: 06h, 07h
Default Value: 0200h
Attribute: Read/Write
Size: 16 bits

DSR is a 16-bit status register that reports the occurrence of a PCI master-abort by the SIO or a PCI target-abort when the SIO is a master. The register also indicates the SIO DEVSEL# signal timing that is hardwired in the SIO.

Bit 15: Reserved
Read as 0.

Bit 14: SERRS (SERR# Status)
This bit is set by the PCI devices that assert the SERR# signal. Since SERR# is only an input to the SIO, this bit is not implemented and will always be read as 0.

Bit 13: MA (Master-Abort Status)
When the SIO, as a master, generates a master-abort, MA is set to a 1. Software sets MA to 0 by writing a 1 to this bit location.

Bit 12: RTA (Received Target-Abort Status)
When the SIO is a master on the PCI Bus and receives a target-abort, this bit is set to a 1. Software sets RTA to 0 by writing a 1 to this bit location.

Bit 11: STA (Signaled Target-Abort Status)
This bit is set to a 1 by the SIO when it generates a target-abort.

Bits[10:9]: DEVT (SIO DEVSEL# Timing Status)
This 2-bit field defines the timing for DEVSEL# assertion. These read only bits indicate the SIO's DEVSEL# timing when performing a positive decode. Since the SIO always generates DEVSEL# with medium timing, DEVT = 01. This DEVSEL# timing does not include Configuration cycles.

Bits[8:0]: Reserved
Read as 0's.

4.1.5 RID—REVISION IDENTIFICATION REGISTER

Address Offset: 08h
Default Value: xxh (dependent on Part Revision)
Attribute: Read Only
Size: 4 bits (upper nibble reserved)

This 4-bit register contains the revision number for the SIO. This number indicates the stepping number of the component. Additionally, the upper nibble of the value is reserved. BIOS should mask the upper nibble when reading this register. These bits are read only. Writes to this register have no effect.

Bits[7:4]: Reserved
These 4 bits are reserved.

Bits[3:0]: Revision Identification Number
This is an 4-bit value that indicates the revision identification number for the SIO. Numbers used so far include:
0h: 82378IB A0-Stepping
1h: 82378IB B0-Stepping
WAS NOT IMPLEMENTED. B0 steppings read 0h also. Read the BIOS Timer Base Address Configuration Register to identify between A0 and B0 steppings.
A0 = 0000h
B0 = 0078h
3h: 82378ZB A0-Stepping

4.1.6 PCICON—PCI CONTROL REGISTER

Address Offset: 40h
Default Value: 20h
Attribute: Read/Write
Size: 8 bits

This 8-bit register controls the Line Buffer operation, the SIO's PCI Posted Write Buffer enabling, and the DEVSEL# signal sampling point. The PCICON Register also controls how the SIO responds to INTA cycles on the PCI Bus and if the reserved DMA page registers are aliased from 80h–8Fh to 90h–9Fh.

Bit 7: Reserved
Read as 0.

Bit 6: DMAAC (DMA Reserved Page Register Aliasing Control)
These register bits control whether the SIO will alias I/O accesses in the 80h–8Fh to the 90h–9Fh range. When DMAAC = 0, the SIO will alias I/O accesses in the 80h–8Fh to the 90h–9Fh range (AD4 is not used for decoding the DMA reserved page registers). When DMAAC = 1, the SIO will only respond to the 80h–8Fh range (AD4 is used for decoding the DMA reserved page registers). Read and write accesses to the 90h–9Fh range will be forwarded from the PCI bus to the ISA bus.

NOTE:
I/O port 92h is always a distinct register in the 90h–9Fh range and is always fully decoded, regardless of the setting of this bit.

Bit 5: IAE (Interrupt Acknowledge Enable)
When IAE = 0, the SIO ignores INTA cycles generated on the PCI Bus. However, when disabled, the SIO still responds to accesses to the 8259's register set and allows poll mode functions. When IAE = 1, the SIO responds to INTA cycles in the normal fashion. This bit defaults to a 1 (respond to INTA cycles).

Bits[4:3]: SDSP (Subtractive Decoding Sample Point)
The SDSP field determines the DEVSEL# sample point, after which an inactive DEVSEL# results in the SIO forwarding the unclaimed PCI cycle to the ISA Bus (subtractive decoding). This setting should match the slowest device in the system.

Bit	4	3	Operation
	0	0	Slow sample point
	0	1	Typical sample point
	1	0	Fast sample point
	1	1	Reserved

Bit 2: PPBE (PCI Posted Write Buffer Enable)
When PPBE = 0, the PCI posted write buffer is disabled. When PPBE = 1, the PCI posted write buffer is enabled. This bit defaults to disabled mode (PPBE = 0).

Bit 1: ILBC (ISA Master Line Buffer Configuration)
When ILBC = 0, the Line Buffer is in single transaction mode. When ILBC = 1, the Line Buffer is in 8-byte mode. This bit applies only to ISA Master transfers. This bit defaults to single transaction mode (ILBC = 0).

Bit 0: DLBC (DMA Line Buffer Configuration)
When DLBC = 0, the Line Buffer is in single transaction mode. When DLBC = 1, the Line Buffer is in 8-byte mode. This bit applies only to DMA transfers. This bit defaults to single transaction mode (DLBC = 0).

4.1.7 PAC—PCI ARBITER CONTROL REGISTER

Address Offset:	41h
Default Value:	00h
Attribute:	Read/Write
Size:	8 bits

This 8-bit register controls the operation of the PCI arbiter. The PAC register enables/disables the guaranteed access time mode, controls bus lock cycles, enables/disables CPU bus parking, and controls the master retry timer.

Bits[7:5]: Reserved
Read as 0's.

Bits[4:3]: MRT (Master Retry Timer)
This 2-bit field determines the number of PCICLKs after the first retry that a PCI initiator's Bus request will be unmasked.

Bit	4	3	Operation
	0	0	Timer disabled, retries never masked.
	0	1	Retries unmasked after 16 PCICLK's.
	1	0	Retries unmasked after 32 PCICLK's.
	1	1	Retries unmasked after 64 PCICLK's.

Bit 2: BP (Bus Park)
Set to a 1 the SIO will park CPUREQ# on the PCI bus when it detects the PCI bus idle. If Bus Park is disabled, the SIO takes responsibility for driving AD, C/BE# and PAR upon detection of bus idle state if the internal arbiter is enabled.

Bit 1: BL (Bus Lock)
This bit selects between bus lock and resource lock. When BL = 1, Bus Lock is selected. The arbiter considers the entire PCI bus locked upon initiation of any locked transaction. When BL = 0, resource lock is enabled. A locked agent is considered a locked resource and other agents may continue normal PCI transactions.

Bit 0: GAT (Guaranteed Access Time)
This bit enables/disables the guaranteed access time mode. When GAT = 1, the SIO is configured for Guaranteed Access Time mode. This mode is available in order to guarantee the 2.5 μs CHRDY time-out specification for the ISA Bus. When the SIO is an Initiator on behalf of an ISA master, the PCI and memory busses are arbitrated for in serial and must be owned before the ISA master is given ownership of the ISA Bus. When GAT = 0, the guaranteed access time mode is disabled. When guaranteed access time mode is disabled, the ISA master is first granted the ISA Bus and then the SIO arbitrates for the PCI Bus.

4.1.8 PAPC—PCI ARBITER PRIORITY CONTROL REGISTER

Address Offset: 42h
Default Value: 04h
Attribute: Read/Write
Size: 8 bits

This register controls the PCI arbiter priority scheme. The arbiter supports six masters arranged through four switching banks. This permits the six masters to be arranged in a purely rotating priority scheme, one of twenty-four fixed priority schemes, or a hybrid combination of the fixed and rotating priority schemes. Bits[4:7] enable/disable rotate priority for the four banks. For each bit, a 1 enables the mode and a 0 disables the mode. If both fixed and rotate modes are enabled for the same bank, the bank will be in rotate mode. For example, if both bits 0 and 4 are set to a 1, bank 0 will be in rotate mode.

Bit 7: Bank 3 Rotate Control

Bit 6: Bank 2 Rotate Control

Bit 5: Bank 1 Rotate Control

Bit 4: Bank 0 Rotate Control

Bit 3: Bank 2 Fixed Priority Mode Select B

Bit 2: Bank 2 Fixed Priority Mode Select A

Bit 1: Bank 1 Fixed Priority Mode Select

Bit 0: Bank 0 Fixed Priority Mode Select

Fixed Priority Mode

The fixed bank control bits select which requester is the highest priority device within that particular bank.

Table 5. Fixed Mode Bank Control Bits

Mode	Bank					Priority					
	3	2b	2a	1	0	Highest					Lowest
00	0	0	0	0	0	SIOREQ#	REQ0#	REQ2#	REQ3#	CPUREQ#	REQ1#
01	0	0	0	0	1	REQ0#	SIOREQ#	REQ2#	REQ3#	CPUREQ#	REQ1#
02	0	0	0	1	0	SIOREQ#	REQ0#	REQ2#	REQ3#	REQ1#	CPUREQ#
03	0	0	0	1	1	REQ0#	SIOREQ#	REQ2#	REQ3#	REQ1#	CPUREQ#
04	0	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ2#	REQ3#
05	0	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ2#	REQ3#
06	0	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ2#	REQ3#
07	0	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ2#	REQ3#
08	0	1	0	0	0	REQ2#	REQ3#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
09	0	1	0	0	1	REQ2#	REQ3#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
0A	0	1	0	1	0	REQ2#	REQ3#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
0B	0	1	0	1	1	REQ2#	REQ3#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
0C–0F	0	1	1	x	x	Reserved					
10	1	0	0	0	0	SIOREQ#	REQ0#	REQ3#	REQ2#	CPUREQ#	REQ1#
11	1	0	0	0	1	REQ0#	SIOREQ#	REQ3#	REQ2#	CPUREQ#	REQ1#
12	1	0	0	1	0	SIOREQ#	REQ0#	REQ3#	REQ2#	REQ1#	CPUREQ#
13	1	0	0	1	1	REQ0#	SIOREQ#	REQ3#	REQ2#	REQ1#	CPUREQ#
14	1	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ3#	REQ2#
15	1	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ3#	REQ2#
16	1	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ3#	REQ2#
17	1	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ3#	REQ2#
18	1	1	0	0	0	REQ3#	REQ2#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
19	1	1	0	0	1	REQ3#	REQ2#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
1A	1	1	0	1	0	REQ3#	REQ2#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
1B		1	0	1	1	REQ3#	REQ2#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
1C–1F	1	1	1	x	x	Reserved					

Rotating Priority Mode
When any Bank Rotate Control bit is set to a one, that particular bank rotates between the two requesting inputs. Any or all banks can be set in rotate mode. If, within a rotating bank, the highest priority input does not have an active request, then the lower priority input will be granted the bus. However, this does not change the rotation scheme. When the bank toggles, the previously lowest priority input will become the highest priority input. Because of this, the maximum latency a device may encounter would be two complete rotations.

4.1.9 ARBPRIX—PCI ARBITER PRIORITY CONTROL EXTENSION REGISTER

Address Offset: 43h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register provides the Fixed Priority Mode select for Bank 3 of the arbiter. The ARBPRIX Register fields are shown.

Bits[7:1]: Reserved
Read as 0.

Bit 0: Bank 3 Fixed Priority Mode Select

0 = REQ2# higher priority
1 = REQ3# higher priority

4.1.10 MEMCSCON—MEMCS# CONTROL REGISTER

Address Offset: 44h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

Bits 0-2 of this register enable MEMCS# blocks. PCI addresses within the enabled blocks result in the generation of MEMCS#. Note that the 0–512 KByte segment does not have RE and WE bits. The 0–512 KByte segment can only be turned off with the MEMCS# Master Enable bit (bit 4). Note also, that when the RE and WE bits are both 0 for a particular segment, the PCI master can not access the segment.

Bits[7:5]: Reserved
Read as 0's.

Bit 4: MEMCS# Master Enable
When the MEMCS# master enable bit is set to a 1, the SIO asserts MEMCS# for all accesses to the defined MEMCS# region (that have been programmed in this register and the MAR1, MAR2, and MAR3 Registers). Also, when this bit is a 1, the positive decoding functions enabled by having the ISA Clock Divisor Register bit 6 = 1 and the Utility Bus Chip Select Register "A" bit 6 = 1 are ignored. Subtractive decoding is provided for these memory areas, instead. When the MEMCS# master enable bit is set to a 0, the entire MEMCS# function is disabled. When this bit is 0, MEMCS# will never be asserted.

Bit 3: Write Enable For 0F0000h–0FFFFFh (Upper 64 KByte BIOS)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory write accesses to the address range 0F0000h–0FFFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the address range 0F0000h–0FFFFFh.

Bit 2: Read Enable For 0F0000h–0FFFFFh (Upper 64 KByte BIOS)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory read accesses to the address range 0F0000h–0FFFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the address range 0F0000h–0FFFFFh.

Bit 1: Write Enable For 080000h–09FFFFh (512 KByte–640 KByte)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory write accesses to the address range 080000h–09FFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the address range 080000h–09FFFFh.

Bit 0: Read Enable For 080000h–09FFFFh (512 KByte–640 KByte)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory read accesses to the address range 080000h–09FFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the address range 080000h–09FFFFh.

4.1.11 MEMCSBOH—MEMCS# BOTTOM OF HOLE REGISTER

Address Offset: 45h
Default value: 10h
Attribute: Read/Write
Size: 8 bits

This register defines the bottom of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSTOH Register. The hole is defined by the following equation: TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by the MCSTOH Register and BOH is the bottom of the MEMCS# hole defined by this register.

For example, to program the BOH at 1 MByte, the value of 10h should be written to this register. To program the BOH at 2 MByte + 64 KByte this register should be programmed to 21h. To program the BOH at 8 MByte this register should be programmed to 80h.

When the TOH < BOH the hole is effectively disabled. It is the responsibility of the programmer to guarantee that the BOH is at or above 1 MB. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH effectively disables the hole.

Bit 7: AD23

Bit 6: AD22

Bit 5: AD21

Bit 4: AD20

Bit 3: AD19

Bit 2: AD18

Bit 1: AD17

Bit 0: AD16

4.1.12 MEMCSTOH—MEMCS# TOP OF HOLE REGISTER

Address Offset: 46h
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

This register defines the top of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSBOH Register. The hole is defined by the following equation: TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by this register and BOH is the bottom of the MEMCS# hole defined by the MCSBOH Register.

For example, to program the TOH at 1 MByte + 64 KByte, this register should be programmed to 10h. To program the TOH at 2 MByte + 128 KByte this register should be programmed to 21h. To program the TOH at 12 MByte this register should be programmed to BFh.

When the TOH < BOH the hole is effectively disabled. It is the responsibility of the programmer to guarantee that the TOH is above 1 MByte. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH effectively disables the hole.

Bit 7: AD23

Bit 6: AD22

Bit 5: AD21

Bit 4: AD20

Bit 3: AD19

Bit 2: AD18

Bit 1: AD17

Bit 0: AD16

4.1.13 MEMCSTOM—MEMCS# TOP OF MEMORY REGISTER

Address Offset: 47h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

This register determines MEMCS# top of memory boundary. The top of memory boundary ranges up to 512 MBytes, in 2 MByte increments. This register is typically set to the top of main memory. Accesses ≥ 2 MByte and ≤ top of memory boundary results in the assertion of the MEMCS# signal (unless the address resides in the hole programmed by the MCSBOH and MCSTOH Registers). A value of 00h disables this 2 MByte-to-top of memory region. A value of 00h assigns the top of memory to include 2 MByte − 1. A value of FFh assigns the top of memory to include 512 MByte − 1.

Bit 7: AD28

Bit 6: AD27

Bit 5: AD26

Bit 4: AD25

Bit 3: AD24

Bit 2: AD23

Bit 1: AD22

Bit 0: AD21

4.1.14 IADCON—ISA ADDRESS DECODER CONTROL REGISTER

Address Offset: 48h
Default value: 01h
Attribute: Read/Write
Size: 8 bits

This register enables the forwarding of ISA or DMA memory cycles to the PCI Bus. In addition, this register sets the top of the "1 MByte to top of main memory" region.

Bits[7:4]:
The top can be assigned in 1 MByte increments from 1 MByte up to 16 MByte. ISA master or DMA accesses within this region are forwarded to PCI unless they are within the hole.

Bits	7	6	5	4	Top of Memory
	0	0	0	0	1 MByte
	0	0	0	1	2 MByte
	0	0	1	0	3 MByte
	0	0	1	1	4 MByte
	0	1	0	0	5 MByte
	0	1	0	1	6 MByte
	0	1	1	0	7 MByte
	0	1	1	1	8 MByte
	1	0	0	0	9 MByte
	1	0	0	1	10 MByte
	1	0	1	0	11 MByte
	1	0	1	1	12 MByte
	1	1	0	0	13 MByte
	1	1	0	1	14 MByte
	1	1	1	0	15 MByte
	1	1	1	1	16 MByte

Bits[3:0]:
ISA and DMA Memory Cycle To PCI Bus Enables. The memory block is enabled by writing a 1 to the corresponding bit position. Setting the bit to 0 disables the corresponding block. ISA or DMA memory cycles to the enabled blocks result in the ISA cycle being forwarded to the PCI Bus. The BIOSCS# enable bit (bit 6 in the UBCSA Register) for the 896K–960K region overrides the function of bit 3 of this register. If the BIOSCS# bit is set to a 1, the ISA or DMA memory cycle is always contained to ISA, regardless of the setting of bit 3 in this register. If the BIOSCS# bit is disabled, the cycle is forwarded to the PCI bus if bit 3 in this register is enabled. Refer to Section 5.5.1.2 for a complete description of BIOS decoding.

Bit	Memory Block
0	0–512 KByte Memory
1	512–640 KByte Memory
2	640–768 KByte VGA Memory
3	896–960 KByte Low BIOS

4.1.15 IADRBE—ISA ADDRESS DECODER ROM BLOCK ENABLE REGISTER

Address Offset: 49h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

ISA addresses within the enabled ranges result in the ISA memory cycle being forwarded to the PCI Bus. For each bit position, the memory block is enabled if the bit is set to 1 and is disabled if the bit is set to 0. If the memory block is disabled, the ISA cycle is not forwarded to the PCI Bus.

Bit 7: 880–896K Memory Enable

Bit 6: 864–880K Memory Enable

Bit 5: 848–864K Memory Enable

Bit 4: 832–848K Memory Enable

Bit 3: 816–832K Memory Enable

Bit 2: 800–816K Memory Enable

Bit 1: 784–800K Memory Enable

Bit 0: 768–784K Memory Enable

4.1.16 IADBOH—ISA ADDRESS DECODER BOTTOM OF HOLE REGISTER

Address Offset: 4Ah
Default value: 10h
Attribute: Read/Write
Size: 8 bits

This register defines the bottom of the ISA Address Decoder hole. The hole is defined by the following equation: TOH ≥ address ≥ BOH, where BOH is the bottom of the hole address programmed into this register and TOH is the top of the hole address programmed into the IADTOH Register. ISA master or DMA addresses falling within the hole will not be forwarded to the PCI Bus. The hole can be sized in 64 KByte increments and placed anywhere between 1 MByte and 16 MByte on any 64 KByte boundary. It is the responsibility of the programmer to guarantee that the BOH is at or above 1 MByte. A[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. When TOH < BOH, the hole is effectively disabled. The default value for the BOH and TOH disables the hole.

For example, to program the BOH at 1 MByte, this register should be set to 10h. To program the BOH at 2 MBytes, this register should be set to 20h. To program the BOH at 8 MBytes, this register should be set to 80h. These settings are shown in Figure 2.

Bit 7: A23

Bit 6: A22

Bit 5: A21

Bit 4: A20

Bit 3: A19

Bit 2: A18

Bit 1: A17

Bit 0: A16

4.1.17 IADTOH—ISA ADDRESS DECODER TOP OF HOLE REGISTER

Address Offset: 4Bh
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

This register defines the top of the ISA Address Decoder hole. The hole is defined by the following equation: TOH ≥ address ≥ BOH, where BOH is the bottom of the hole address programmed into the LADBOH Register and TOH is the top of the hole address programmed into this Register. ISA master or DMA addresses falling within the hole will not be forwarded to the PCI Bus. The hole can be sized in 64 KByte increments and placed anywhere between 1 MByte and 16 MByte on any 64 KByte boundary. It is the responsibility of the programmer to guarantee that the TOH is at or above 1 MByte. A[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. When TOH < BOH, the hole is disabled. The default value for the BOH and TOH disables the hole.

For example, to program the TOH at 1 MByte + 64 KByte, this register should be set to 10h. To program the TOH at 2 MByte + 128 KByte, this register should be set to 21h. To program the TOH at 12 MByte, this register should be set to BFh. These settings are shown in Figure 2.

Bit 7: A23

Bit 6: A22

Bit 5: A21

Bit 4: A20

Bit 3: A19

Bit 2: A18

Bit 1: A17

Bit 0: A16

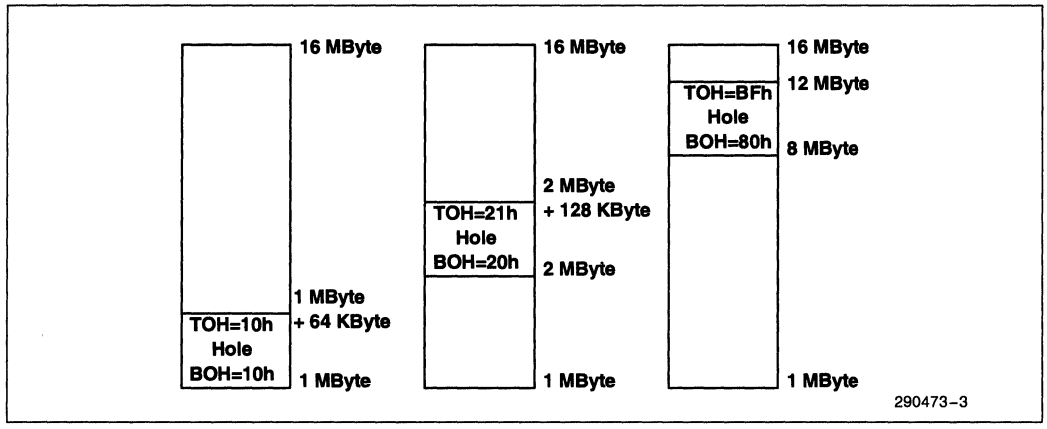


Figure 2. ISA Address Decoder Hole Examples

Table 6. Examples of ISA Decoding

Test Case Description	TOM (48h)	TOH (4Bh)	BOH (4Ah)	Address (hex)	Address	Result
8MB TOM, no hole @ 1M	7xh	0Fh	10h	01000000h 00FFFFFFh 00800000h 007FFFFFh 00100000h 000FFFFFh	16MB 16MB-1 8MB 8MB-1 1MB 1MB-1	To PCI ISA ISA To PCI To PCI ISA (BIOS)
4MB TOM, no hole @ 2M	3xh	1Fh	20h	01000000h 00FFFFFFh 00400000h 003FFFFFh 00200000h 001FFFFFh 00100000h	16MB 16MB-1 4MB 4MB-1 2MB 2MB-1 1MB	To PCI ISA ISA To PCI To PCI To PCI To PCI
1MB TOM, no hole @ 1M	0xh	0Fh	10h	01000000h 00FFFFFFh 00100000h 000FFFFFh	16MB 16MB-1 1MB 1MB-1	To PCI ISA ISA ISA (BIOS)
16MB TOM, 64KB hole @ 15MB	Fxh	F0h	F0h	01000000h 00FFFFFFh 00F10000h 00F0FFFFh 00F00000h 00EFFFFFh 00E10000h 00E0FFFFh 00E00000h 00DFFFFFh	16MB 16MB-1 15MB + 64KB 15MB + 64KB-1 15MB 15MB-1 14MB + 64KB 14MB + 64KB-1 14MB 14MB-1	To PCI To PCI To PCI ISA ISA To PCI To PCI To PCI To PCI To PCI
12MB TOM, 2MB + 128KB hole @ 2MB	Bxh	21h	20h	01000000h 00FFFFFFh 00C00000h 00BFFFFFh 00220000h 0021FFFFh 00210000h 0020FFFFh 00200000h 001FFFFFh 00100000h	16MB 16MB-1 12MB 12MB-1 2MB + 128KB 2MB + 128KB-1 2MB + 64KB 2MB + 64KB-1 2MB 2MB-1 1MB	To PCI ISA ISA To PCI To PCI ISA ISA ISA ISA To PCI To PCI

Table 6. Examples of ISA Decoding (Continued)

Test Case Description	TOM (48h)	TOH (4Bh)	BOH (4Ah)	Address (hex)	Address	Result
5MB TOM, 3MB hole @	4xh	47h	18h	01000000h	16MB	To PCI
				00FFFFFFh	16MB-1	ISA
				00500000h	5MB	ISA
				004FFFFFh	5MB-1	To PCI
				00480000h	4.5MB	To PCI
				0047FFFFh	4.5MB-1	ISA
				00180000h	1.5MB	ISA
				0017FFFFh	1.5MB-1	To PCI
				00100000h	1MB	To PCI

4.1.18 ICRT—ISA CONTROLLER RECOVERY TIMER REGISTER

Address Offset: 4Ch
Default Value: 56h
Attribute: Read/Write
Size: 8 bits

The I/O recovery mechanism in the SIO is used to add additional recovery delay between PCI originated 8-bit and 16-bit I/O cycles to the ISA bus. The SIO automatically forces a minimum delay of five SYSCLKs between back-to-back 8- and 16-bit I/O cycles to the ISA bus. The delay is measured from the rising edge of the I/O command (IOR# or IOW#) to the falling edge of the next BALE. If a delay of greater than five SYSCLKs is required, the ISA I/O Recovery Time Register can be programmed to increase the delay in increments of SYSCLKs. Note that no additional delay is inserted for back-to-back I/O "sub cycles" generated as a result of byte assembly or disassembly. This register defaults to 8- and 16-bit recovery enabled with two clocks added to the standard I/O recovery.

Bit 7: Reserved
Read as 0.

Bit 6: 8-Bit I/O Recovery Enable
This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits[5:3] are enabled. When this bit is set to 0, recovery times are disabled.

Bits[5:3]: 8-Bit I/O Recovery Times
This 3-bit field defines the recovery times for 8-bit I/O. Programmable delays between back-to-back 8-bit PCI cycles to ISA I/O slaves is shown in terms of ISA clock cycles (SYSCLK) added to the five minimum. The selected delay programmed into this field is enabled/disabled via bit 6 of this register.

Bit 5	4	3	SYSCLK Added	Total SYSCLKs
0	0	1	+1	6
0	1	0	+2	7
0	1	1	+3	8
1	0	0	+4	9
1	0	1	+5	10
1	1	0	+6	11
1	1	1	+7	12
0	0	0	+8	13

Bit 2: 16-Bit I/O Recovery Enable
This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits 0 and 1 are enabled. When this bit is set to 0, recovery times are disabled.

Bits[1:0]: 16-Bit I/O Recovery Times
This 2-bit field defines the recovery time for 16-bit I/O. Programmable delays between back-to-back 16-bit PCI cycles to ISA I/O slaves is shown in terms of ISA clock cycles (SYSCLK) added to the five minimum. The selected delay programmed into this field is enabled/disabled via bit 2 of this register.

Bit 1	Bit 0	SYSCLK Added	Total SYSCLKs
0	1	+1	6
1	0	+2	7
1	1	+3	8
0	0	+4	9

4.1.19 ICD—ISA CLOCK DIVISOR REGISTER

Address Offset: 4Dh
Default Value: 40h
Attribute: Read/Write
Size: 8 bits

This register selects the integer value used to divide the PCI clock (PCICLK) to generate the ISA clock (SYSCLK). In addition, this register provides an ISA Reset bit to software control RSTDRV, a bit to enable/disable the MOUSE function, a bit to enable/disable the coprocessor error support, and a bit to disable the positive decode for the upper 64 KBytes of BIOS at the top of 1 MByte (F0000h–FFFFFh) and aliased regions.

Bit 7: Reserved

Bit 6: Positive Decode of Upper 64 KByte BIOS Enable
This bit enables (bit 6 = 1) and disables (bit 6 = 0) the positive decode of the upper 64 KBytes of BIOS area at the top of 1 MByte (F0000h–FFFFFh) and the aliased regions at the top of 4 GBytes (FFFF0000h–FFFFFFFFh) and 4 GByte-1 MByte (FFEF0000–FFEFFFFFh). When bit 6 = 1, these address regions are positively decoded, unless bit 4 in the MEMCS# Control Register is set to a 1 in which case these regions are subtractively decoded. When bit 6 = 0, these address regions are subtractively decoded. The encoded chip selects for BIOSCS# and the UBUSOE# signal will always be generated when these locations are accessed, regardless of the state of this bit. A reset sets this bit to a 1 (positive decode enabled).

Bit 5: Coprocessor Error Enable
This bit is used to enable and disable the Coprocessor error support. When enabled (bit 5 = 1), the FERR# input, when driven active, triggers an IRQ13 to the SIO's interrupt controller. FERR# is also used to gate the IGNNE# output. When disabled (bit 5 = 0), the FERR# signal can be used as IRQ13 and the coprocessor support is disabled. A reset sets this bit to 0 (coprocessor support disabled).

Bit 4: IRQ12/M Mouse Function Enable
When this bit is set to 1, IRQ12/M provides the mouse function. When this bit is set to 0, IRQ12/M provides the standard IRQ12 interrupt function. A hard reset sets this bit to 0.

Bit 3: RSTDRV Enable
This bit is used to enable RSTDRV on the ISA Bus. When this bit is set to 1, RSTDRV is asserted and remains asserted until this bit is set to a 0. When set to 0, normal operation of RSTDRV is provided. This bit should be used during configuration to reset the ISA Bus when changing the clock divisor. For a reset, this bit defaults to 0. Note that the software must ensure that RSTDRV is asserted for a minimum of 1 μs.

Bit[2:0]: PCICLK-to-ISA SYSCLK Divisor
These bits are used to select the integer that is used to divide the PCICLK down to generate the ISA SYSCLK. Upon reset, these bits are set to 000 (divisor of 4 selected). For PCI frequencies less than 33 MHz (not including 25 MHz), a clock divisor value must be selected that ensures that the ISA Bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK specification.

Bit 2	Bit 1	Bit 0	Divisor	SYSCLK
0	0	0	4 (33 MHz)	8.33 MHz
0	0	1	3 (25 MHz)	8.33 MHz
0	1	0	Reserved	
0	1	1	Reserved	
1	0	0	Reserved	
1	0	1	Reserved	
1	1	0	Reserved	
1	1	1	Reserved	

4.1.20 UBCSA—UTILITY BUS CHIP SELECT A REGISTER

Address Offset: 4Eh
Default Value: 07h
Attribute: Read/Write
Size: 8 bits

This register enables/disables accesses to the RTC, keyboard controller, Floppy Disk controller, IDE, and BIOS locations E0000h–EFFFFh and FFF80000h–FFFDFFFFh. Disabling any of these bits prevents the encoded chip select bits (ECSADDR[2:0]) and utility bus transceiver control signal (UBUSOE#) for that device from being generated.

This register is also used to select which address range (primary or secondary) will be decoded for the resident floppy controller and IDE. This ensures that there is no contention with the Utility bus transceiver driving the system data bus during read accesses to these devices.

Bit 7: Extended BIOS Enable
When bit 7 = 1 (enabled), PCI accesses to locations FFF80000h–FFFDFFFFh result in the generation of the encoded signals (ECSADDR[2:0]) for BIOS. When enabled, PCI master accesses to this area are positively decoded and UBUSOE# is generated. When this bit is disabled (bit 7 = 0), the SIO does not generate the encoded (ECSADDR[2:0]) signals or UBUSOE#.

Bit 6: Lower BIOS Enable
When bit 6 = 1 (enabled), PCI or ISA accesses to the lower 64 KByte BIOS block (E0000h–EFFFFh) at the top of 1 MByte, or the aliases at the top of 4 GByte and 4 GByte–1 MByte results in the generation of the encoded (ECSADDR[2:0]) signals for BIOS. When enabled, PCI master accesses to this area are positively decoded to the ISA Bus, unless bit 4 in the MEMCS# Control Register is set to a 1 in which case these regions are subtractively decoded. Also, when enabled, ISA master or DMA master accesses to this region are not forwarded to the PCI Bus. When this bit is disabled (bit 6 = 0), the SIO does not generate the encoded (ECSADDR[2:0]) signals. Also, when this bit is disabled, ISA master or DMA accesses to this region are forwarded to PCI, if bit 3 in the IADCON Register is set to 1.

Bit 4: IDE Decode Enable
Bit 4 enables/disables IDE locations 1F0h–1F7h (primary) or 170h–177h (secondary) and 3F6h, 3F7h (primary) or 376h, 377h (secondary). When bit 4 = 1, the IDE encoded chip select signals and the Utility Bus transceiver signal (UBUSOE#) are generated for these addresses. When bit 4 = 0, the IDE encoded chip select signals and the Utility Bus transceiver signal (UBUSOE#) are not generated for these addresses.

Bit [5, 3:2]: Floppy Disk Address Locations Enable
Bits 2 and 3 are used to enable or disable the floppy locations as indicated below. A PCIRST# sets bit 2 to 1 and bit 3 to 0. Bit 5 is used to select between the primary and secondary address range used by the Floppy Controller and the IDE. Only primary or only secondary can be programmed at any one time. A PCIRST# sets this bit to 0 (primary).

The following table shows how these bits are used to select the floppy controller:

Address	Bit 5	Bit 3	Bit 2	DSKCHG	ECSADDR[2:0]	FLOPPYCS#
X	X	X	X	0	1 1 1	1
3F0h, 3F1h	0	1	X	1	1 0 0	0
3F2h–3F7h	0	X	1	1	1 0 0	0 (note)
370h, 371h	1	1	X	1	1 0 0	0
372h–37Fh	1	X	1	1	1 0 0	0 (note)

NOTE:
If IDE decode is enabled (bit 4 = 1), all accesses to locations 03F6h and 03F7h (primary) or 0376h and 0377h (secondary) result in the ECSADDR[2:0] signals generating a decode for IDECS1# (FLOPPYCS# is not generated). An external AND gate can be used to tie IDECS1# and FLOPPYCS# together to insure that the floppy is enabled for these accesses. If IDE decode is disabled (bit 4 = 0), and the decode for the floppy is enabled, then the encoded chip selects for the floppy locations are generated.

Bit 1: Keyboard Controller Address Location Enable
Enables (1) or disables (0) the Keyboard controller address locations 60h, 62h, 64h, and 66h. When this bit is set to 0, the Keyboard Controller encoded chip select signals (ECSADDR[2:0]) and the Utility Bus transceiver signal (UBUSOE#) are not generated for these locations.

Bit 0: RTC Address Location Enable
Enables (1) or disables (0) the RTC address locations 70h–77h. When this bit is set to 0, the RTC encoded chip select signals (ECSADDR[2:0]), RTCALE#, RTCCS#, and UBUSOE# signals are not generated for these addresses.

4.1.21 UBCSB—UTILITY BUS CHIP SELECT B REGISTER

Address Offset: 4Fh
Default Value: 4Fh
Attribute: Read/Write
Size: 8 bits

This register is used to enable/disable accesses to the serial ports and parallel port locations supported by the SIO. When disabled, the ECSADDR(2:0) encoded chip select bits and Utility Bus Transceiver control signal (UBUSOE#), for that device, are not generated. This register is also used to disable accesses to port 92 and enable or disable configuration RAM decode.

Bit 7: Configuration RAM Decode Enable
This bit is used to enable (bit 7 = 1) or disable (bit 7 = 0) I/O write accesses to location 0C00h and I/O read/write accesses to locations 0800h–08FFh. When enabled, the encoded chip select signals for generating an external configuration page chip select (CPAGECS#) are generated for accesses to 0C00h. The encoded chip select signals for generating an external configuration memory chip select (CFIGMEMCS#) are generated for accesses to 0800h–08FFh. When bit 7 = 0, configuration RAM decode is disabled and the CPAGECS# and CFIGMEMCS# are not generated for the corresponding accesses.

Bit 6: Port 92 Enable
This bit is used to enable/disable access to Port 92. When bit 6 = 1, Port 92 is enabled. When bit 6 = 0, Port 92 is disabled. When a PCIRST# occurs, this bit is set to 1 (enable).

Bits[5:4]: Parallel Port Enable
These bits are used to select the parallel port address range: (LPT1, LPT2, LPT3, or disable). When a PCIRST# occurs, this field is set to 00 (LPT1).

Bit 5	Bit 4	Function
0	0	3BCh–3BFh (LPT1)
0	1	378h–37Fh (LP2)
1	0	278h–27Fh (LPT3)
1	1	Disabled

Bits[3:2]: Serial Port B Enable
These bits are used to assign serial port B address range: (COM1, COM2, or disable). If either COM1 or COM2 address ranges are selected, the encoded chip select signals [ECSADDR(2:0)] for Port B will be generated. A PCIRST# sets bits[3:2] to 11 (Port B disabled).

Bit	3	2	Function
	0	0	3F8h–3FFh (COM1)
	0	1	2F8h–2FFh (COM2)
	1	0	Reserved
	1	1	Port B Disabled

NOTE:
If Serial port A and B are programmed for the same I/O address, the encoded chip select signals, ECSADDR(2:0), for port B are disabled.

Bits[1:0]: Serial Port A Enable
These bits are used to assign serial port A address range: (COM1, COM2, or disable). If either COM1 or COM2 address ranges are selected, the encoded chip select signals (ECSADDR[2:0]) for Port A will be generated. A PCIRST# sets bits[1:0] to 11 (port A disabled).

Bit	1	0	Function
	0	0	3F8h–3FFh (COM1)
	0	1	2F8h–2FFh (COM2)
	1	0	Reserved
	1	1	Port A disabled

NOTE:
If Serial port A and B are programmed for the same I/O address, the encoded chip select signals, ECSADDR[2:0], for port B are disabled.

4.1.22 MAR1—MEMCS# ATTRIBUTE REGISTER #1

Address Offset: 54h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory read accesses to the corresponding segment. When the RE bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

Bit 7: 0CC000h–0CFFFFh Exp. ROM: WE

Bit 6: 0CC000h–0CFFFFh Exp. ROM: RE

Bit 5: 0C8000h–0CBFFFh Exp. ROM: WE

Bit 4: 0C8000h–0CBFFFh Exp. ROM: RE

Bit 3: 0C4000h–0C7FFFh Exp. ROM: WE

Bit 2: 0C4000h–0C7FFFh Exp. ROM: RE

Bit 1: 0C0000h–0C3FFFh Exp. ROM: WE

Bit 0: 0C0000h–0C3FFFh Exp. ROM: RE

4.1.23 MAR2—MEMCS# ATTRIBUTE REGISTER #2

Address Offset: 55h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory read accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

Bit 7: 0DC000h–0DFFFFh Exp. ROM : WE

Bit 6: 0DC000h–0DFFFFh Exp. ROM : RE

Bit 5: 0D8000h–0DBFFFh Exp. ROM : WE

Bit 4: 0D8000h–0DBFFFh Exp. ROM : RE

Bit 3: 0D4000h–0D7FFFh Exp. ROM : WE

Bit 2: 0D4000h–0D7FFFh Exp. ROM : RE

Bit 1: 0D0000h–0D3FFFh Exp. ROM : WE

Bit 0: 0D0000h–0D3FFFh Exp. ROM : RE

4.1.24 MAR3—MEMCS# ATTRIBUTE REGISTER #3

Address Offset: 56h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, ISA master memory read accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master can not access the corresponding segment.

Bit 7: 0EC000h–0EFFFFh Lower 64 KByte BIOS: WE

Bit 6: 0EC000h–0EFFFFh Lower 64 KByte BIOS: RE

Bit 5: 0E8000h–0EBFFFh Lower 64 KByte BIOS WE

Bit 4: 0E8000h–0EBFFFh Lower 64 KByte BIOS: RE

Bit 3: 0E4000h–0E7FFFh Lower 64 KByte BIOS: WE

Bit 2: 0E4000h–0E7FFFh Lower 64 KByte BIOS: RE

Bit 1: 0E0000h–0E3FFFh Lower 64 KByte BIOS: WE

Bit 0: 0E0000h–0E3FFFh Lower 64 KByte BIOS: RE

4.1.25 DMA SCATTER/GATHER RELOCATION BASE ADDRESS REGISTER

Address Offset: 57h
Default Value: 04h
Attribute: Read/Write
Size: 8 bits

The value programmed into this register determines the high order I/O address of the Scatter/Gather Command Registers, Scatter/Gather Status Registers, and the Scatter/Gather Descriptor Table Registers. The default value is 04h so the first S/G register default address is at 0410h.

Bit 7: A15

Bit 6: A14

Bit 5: A13

Bit 4: A12

Bit 3: A11

Bit 2: A10

Bit 1: A9

Bit 0: A8

4.1.26 PIRQ[3:0]#—PIRQ ROUTE CONTROL REGISTERS

Register Name: PIRQ0#, PIRQ1#, PIRQ2#, PIRQ3# Route Control

Address Offset: 60h, 61h, 62h, 63h
Default Value: 80h
Attribute: Read/Write
Size: 8 bits

These registers control the routing of PCI Interrupts (PIRQ[0:3]#) to the PC compatible Interrupts. Each PCI interrupt can be independently routed to 1 of 11 compatible interrupts. Note that two or more PCI interrupts (PIRQ[3:0]#) can be steered into the same IRQ signal (the interrupts are level sensitive and can be shared).

Each IRQ to which a PCI Interrupt is steered into must have its interrupt set to level sensitive in the Edge/Level Control Register.

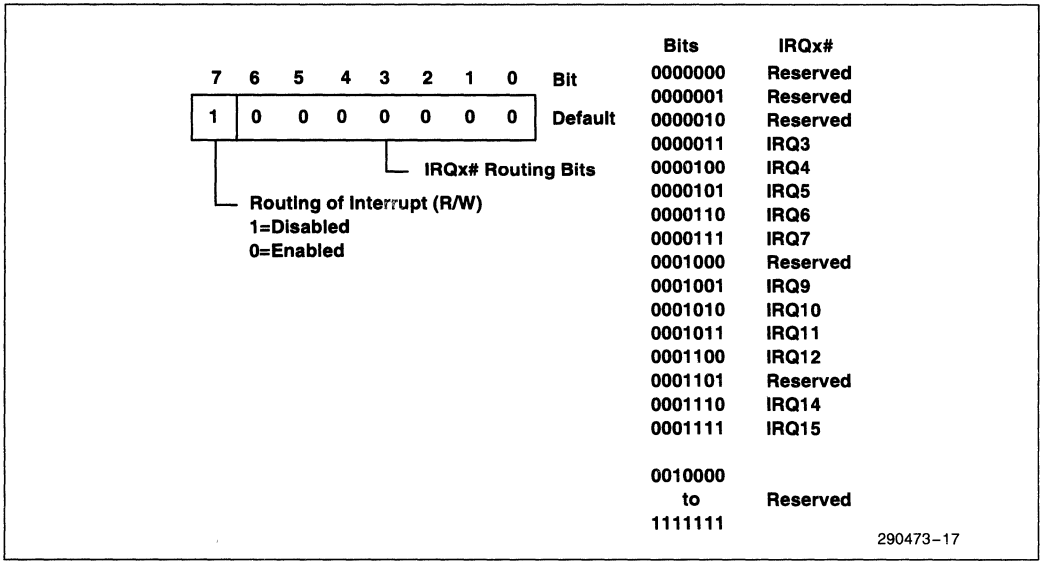


Figure 3. PIRQ Route Control Registers

Bit 7: Routing of Interrupts
When enabled, this bit routes the PCI Interrupt signal to the PC compatible interrupt signal specified in bits[3:0]. At reset, this bit is disabled (set to 1).

Bits[6:4]: Reserved
Read as 0's.

Bits[3:0]: IRQx# Routing Bits
These bits specify which IRQ signal to generate.

4.1.27 BIOS TIMER BASE ADDRESS REGISTER

Address Offset: 80h–81h
Default value: 0078h
Attribute: Read/Write
Size: 16 bits

This register determines the base address for the BIOS Timer Register located in the I/O space. The base address can be set at Dword boundary anywhere in the 64 KByte I/O space. This register also provides the BIOS Timer access enable/disable control bit.

Bits[15:2]: BIOS Timer Base Address
Bits[15:2] correspond to PCI address lines A[15:2].

Bit 1: Reserved

Bit 0: BIOS Timer Access Enable
When bit 0 = 1, access to the BIOS Timer is enabled. When bit 0 = 0, access to the BIOS Timer is disabled. The default value is 0 (disabled).

4.1.28 SMICNTL—SMI CONTROL REGISTER

Address Offset: A0h
Default value: 08h
Attribute: Read/Write
Size: 8 bits

Bit 7: Reserved
Reserved for future Intel use.

Bit 6: Reserved
Reserved for future Intel use.

Bits[5:4]: Reserved
Reserved for future Intel use.

Bit 3: CTMRFRZ
Used to freeze the timers when in SMM. When this bit is set, the Fast Off timer will stop counting. This prevents time-outs from occurring while executing SMM code.

Bit 2: CSTPCLKTH
When set, the STPCLK# throttle is enabled.

Bit 1: CSTPCLKEN
When set, a read from the APMC register will cause STPCLK# to be asserted. CSTPCLKEN will be cleared by writing it to 0 or by any write to the APMC register. Enables SW to put the CPU into a low power state.

Bit 0: CSMIGATE
When this bit is written to "0" SMI# will be deasserted. When this bit is written to a "1", SMI# will be asserted if any SMIs are pending.

4.1.29 SMIEN—SMI ENABLE REGISTER

Address Offset: A2h–A3h
Default value: 0000h
Attribute: Read/Write
Size: 16 bits

The following bits control the enabling of associated hardware events that will generate an SMI. When set to a "1", SMI# will be asserted when the associated event occurs. When bit 7 is set, writes to the APM Control port (APMC) will generate an SMI.

Bits[15:8]: Reserved
Will be read as 0. Writes have no effect.

Bit 7: SAPMCEN
Write to APM Control Port.

Bit 6: SEXTSMIEN
EXTSMI# input asserted.

Bit 5: SFOFFTMREN
Fast Off (Idle) Timer Enable.

Bit 4: SIRQ12EN
PS/2 Mouse Interrupt.

Bit 3: SIRQ8EN
RTC Alarm Interrupt.

Bit 2: SIRQ4EN
COM2/COM4 Interrupt (Mouse).

Bit 1: SIRQ3EN
COM1/COM3 Interrupt (Mouse).

Bit 0: SIRQ1EN
Keyboard Interrupt.

4.1.30 SEE—SYSTEM EVENT ENABLE REGISTER

Address Offset:	A4h, A5h, A6h, A7h
Default value:	00000000h
Attribute:	Read/Write
Size:	32 bits

These bits are used to enable the corresponding hardware events as system events. When set to a "1", anytime the associated hardware event occurs, the Fast Off Timer is reloaded with its initial count. Also, when enabled the associated hardware system event is recognized as a Break Event causing STPCLK# to be deasserted.

Bit 31: FSMIEN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when an SMI occurs.

Bit 30: Reserved
Will be read as 0. Writes have no effect.

Bit 29: FNMIEN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when an NMI occurs (parity error for example).

Bits[28:16]: Reserved
Will be read as 0. Writes have no effect.

Bits[15:3]: FIRQ[15:3]EN
This prevents the system from entering Fast Off and causes STPCLK# to be deasserted when selected hardware interrupts occur.

Bit 2: Reserved
Will be read as 0. Writes have no effect.

Bits[1:0]: FIRQ[1:0]EN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when selected hardware interrupts occur.

4.1.31 FTMR—FAST OFF TIMER

Address Offset:	A8h
Default value:	0Fh
Attribute:	Read/Write
Size:	8 bits

The Fast Off Timer is used to indicate (through an SMI) that the system has been idle for a pre-programmed period of time. The Fast Off Timer consists of a count down timer that is decremented every minute. The value programmed into this register gets loaded into the Fast Off Timer when an enabled system event occurs. Each count represents one minute. When the timer expires, an SMI Special Cycle is generated. Writes to the FTMRLD register cause the Fast Off Timer to be loaded. When this register is read, the value last written to this register is returned.

PROGRAMMER'S NOTE:
Before writing to the FTMRLD register the Fast Off Timer must be stopped by writing a "1" to the CTMRFRZ bit. The Fast Off Timer will begin decrementing when the CTMRFRZ bit is subsequently set to "0".

Bits[7:0]: FTMRLD[7:0]
A write to the FTMRLD register when the Fast Off Timer is stopped (CTMRFRZ = 1) will load the Fast Off Timer with the value being written to FTMRLD register. When the Fast Off Timer is enabled (CTMRFRZ = 0) it counts down from the value loaded into it. When the Fast Off Timer reaches 00h it will trigger an SMI. If an enabled system event occurs before the Fast Off Timer reaches 00h the Fast Off Timer is reloaded with the value stored in the FTMRLD register. A read from the FTMRLD register will return the value last written to this register.

4.1.32 SMIREQ-SMI REQUEST REGISTER

Address Offset: AAh, ABh
Default value: 00h
Attribute: Read/Write
Size: 16 bits

These bits are status bits indicating the cause of the SMI. When an enabled event causes an SMI, the hardware automatically sets the corresponding event's request bit. The request bits are cleared by writing a "0" to them. Only the hardware can set request bits to a "1". In the event that the hardware is trying to set the bit to a "1" at the same time that it is being cleared, the hardware set to "1" will dominate.

Bits[15:8]: Reserved
Reserved for future Intel use.

Bit 7: RAPMC
When set to a "1" indicates that a write to the APM Control Port caused an SMI#.

Bit 6: REXT
When set to a "1", indicates that EXTSMI# was sampled asserted, causing an SMI#.

Bit 5: RFOFFTMR
Fast Off Timer expired causing an SMI#.

Bit 4: RIRQ12
IRQ12 was asserted causing an SMI#.

Bit 3: RIRQ8
IRQ8 was asserted causing an SMI#.

Bit 2: RIRQ4
IRQ4 was asserted causing an SMI#.

Bit 1: RIRQ3
IRQ3 was asserted causing an SMI#.

Bit 0: RIRQ1
IRQ1 was asserted causing an SMI#.

4.1.33 CTLTMRL—CLOCK THROTTLE STPCLK# LOW TIMER

Address Offset: ACh
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The value in this register defines the duration of the STPCLK# asserted period when the CSTPCLKTH bit is set. The value in this register is loaded into the STPCLK# Timer when STPCLK# is asserted. The STPCLK# timer runs off a 32 μs clock. Note that the timer does not begin to count until the Stop Grant Special Cycle is received.

Bits[7:0]: KSTPLOLD[7:0]
The value in this register defines the duration of the STPCLK# asserted period when the CSTPCLKTH bit is set.

4.1.34 CTLTMRH—CLOCK THROTTLE STPCLK# HIGHTIMER

Address Offset: AEh
Default value: 00h
Attribute: Read/Write
Size: 8 bits

The value in this register defines the duration of the STPCLK# deasserted period when the CSTPCLKTH bit is set. The value in this register is loaded into the STPCLK# Timer when STPCLK# is deasserted. The STPCLK# timer runs off a 32 μs clock.

Bits[7:0]: KSTPHILD[7:0]
The value in this register defines the duration of the STPCLK# deasserted period when the CSTPCLKTH bit is set.

4.2 DMA Register Description

The SIO contains DMA circuitry that incorporates the functionality of two 82C37 DMA controllers (DMA1 and DMA2). The DMA registers control the operation of the DMA controllers and are all accessible from the PCI Bus via PCI I/O space. In addition, some of the registers are accessed from the ISA Bus via ISA I/O space. Table 4, at the beginning of Section 4.0 lists the bus access for each register.

This section describes the DMA registers. Unless otherwise stated, a PCIRST# sets each register to its default value. The operation of the DMA is further described in Section 5.4, DMA Controller.

4.2.1 DCOM—DMA COMMAND REGISTER

Address Offset:	Channels 0-3–08h Channels 4–7–0D0h
Default Value:	00h
Attribute:	Write Only
Size:	8 bits

This 8-bit register controls the configuration of the DMA. It is programmed by the microprocessor in the Program Condition and is cleared by PCIRST# or a Master Clear instruction. Note that disabling channels 4–7 also disables channels 0–3, since channels 0–3 are cascaded onto channel 4. The DREQ and DACK# channel assertion sensitivity is assigned by channel group, not per individual channel. For priority resolution, the DMA consists of two logical channel groups—channels 0–3 (Controller 1-DMA1) and channels 4–7 (Controller 2-DMA2). Each group can be assigned fixed or rotating priority. Both groups can be assigned fixed priority, one group can be assigned fixed priority and the second rotating priority, or both groups can be assigned rotating priority. Following a PCIRST# or DMA Master Clear, both DMA1 and DMA2 are enabled in fixed priority, the DREQ sense level is active high, and the DACK# assertion level is active low.

Bit 7: DACK# Assert Level (DACK#[3:0], [7:5])
Bit 7 controls the DMA channel request acknowledge (DACK#) assertion level. Following PCIRST#, the DACK# assertion level is active low. The low level indicates recognition and acknowledgment of the DMA request to the DMA slave requesting service. Writing a 0 to bit 7 assigns active low as the assertion level. When a 1 is written to this bit, a high level on the DACK# line indicates acknowledgment of the request for DMA service to the DMA slave.

Bit 6: DREQ Sense Assert Level (DREQ[3:0], [7:5])
Bit 6 controls the DMA channel request (DREQ) assertion detect level. Following PCIRST#, the DREQ sense assert level is active high. In this condition, an active high level sampled on DREQ is decoded as an active DMA channel request. Writing a 0 to bit 6 assigns active high as the sense assert level. When a 1 is written to this bit, a low level on the DREQ line is decoded as an active DMA channel request.

Bit 5: Reserved
Must be 0.

Bit 4: DMA Group Arbitration Priority
Each channel group is individually assigned either fixed or rotating arbitration priority. At PCIRST#, each group is initialized in fixed priority. Writing a 0 to bit 4 assigns fixed priority to the channel group, while writing a 1 assigns rotating priority to the group.

Bit 3: Reserved
Must be 0.

Bit 2: DMA Channel Group Enable
Writing a 1 to this bit disables the DMA channel group, while writing a 0 to this bit enables the DMA channel group. Both channel groups are enabled following PCIRST#. Disabling channel group 4–7 also disables channel group 0–3, which is cascaded through channel 4.

Bits[1:0]: Reserved
Must be 0.

4.2.2 DCM—DMA CHANNEL MODE REGISTER

Register Name:	DMA Channel Mode
Address Offset:	Channels 0-3–0Bh Channels 4-7–0D6h
Default Value:	Bits[7:2] = 0, Bits[1:0] = undefined
Attribute:	Write Only
Size:	6 bits

Each channel has a 6-bit DMA Channel Mode Register. The Channel Mode Registers provide control over DMA Transfer type, transfer mode, address increment/decrement, and autoinitialization. Bits[1:0] select the appropriate Channel Mode Register and are not stored. Only bits[7:2] are stored in the register. This register is set to its default value upon

PCIRST# or Master Clear. Its default value is Verify transfer, Autoinitialize disable, Address increment, and Demand mode. Channel 4 defaults to cascade mode and cannot be programmed for any mode other than cascade mode.

Bits[7:6]: DMA Transfer Mode
Each DMA channel can be programmed in one of four different modes: single transfer, block transfer, demand transfer and cascade.

Bits	7	6	Transfer Mode
	0	0	Demand mode
	0	1	Single mode
	1	0	Block mode
	1	1	Cascade mode

Bit 5: Address Increment/Decrement Select
Bit 5 controls address increment/decrement during multi-byte DMA transfers. When bit 5 = 0, address increment is selected. When bit 5 = 1, address decrement is selected. Address increment is the default after a PCIRST# cycle or Master Clear command.

Bit 4: Autoinitialize Enable
When bit 4 = 1, the DMA restores the Base Page, Address, and Word count information to their respective current registers following a terminal count (TC). When bit 4 = 0, the autoinitialize feature is disabled and the DMA does not restore the above mentioned registers. A PCIRST# or Master Clear disables autoinitialization (sets bit 4 to 0).

Bits[3:2]: DMA Transfer Type
Verify, write and read transfer types are available. Verify transfer is the default transfer type upon PCIRST# or Master Clear. Write transfers move data from an I/O device to memory. Read transfers move data from memory to an I/O device. Verify transfers are pseudo transfers; addresses are generated as in a normal read or write transfer and the device responds to EOP etc. However, with Verify transfers, the ISA memory and I/O cycle lines are not driven. Bit combination 11 is illegal. When the channel is programmed for cascade ([7:6] = 11) the transfer type bits are irrelevant.

Bits	3	2	Transfer Mode
	0	0	Verify transfer
	0	1	Write transfer
	1	0	Read Transfer
	1	1	Illegal

Bits[1:0]: DMA Channel Select
Bits[1:0] select the DMA Channel Mode Register that will be written by bits[7:2].

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.3 DCEM—DMA CHANNEL EXTENDED MODE REGISTER

Address Offset: Channels 0-3–040Bh
Channels 4-7–04D6h
Default Value: Bits[1:0] = undefined,
Bits[3:2] = 00 for DMA1,
Bits[3:2] = 01 for DMA2,
Bits[7:4] = 0
Attribute: Write Only
Size: 6 bits

Each channel has a 6-bit Extended Mode Register. The register is used to program the DMA device data size, timing mode, EOP input/output selection, and Stop Register selection. Bits[1:0] select the appropriate Channel Extend Mode Register and are not stored. Only bits[7:2] are stored in the register. Four timing modes are available: ISA-compatible, "A", "B", and "F". Timings "A", "B", and "F" are extended timing modes and can only be run to main memory. DMA cycles to ISA expansion bus memory defaults to compatible timing if the channel is programmed in an extended timing mode.

The default bit values for each DMA group are selected upon PCIRST#. A Master Clear or any other programming sequence will not set the default register settings. The default programmed values for DMA1 channels 0-3 are 8-bit I/O count by bytes, compatible timing, and EOP output. The default values for DMA2 channels 4-7 are 16-bit I/O count by words with shifted address, compatible timing, and EOP output.

Bit 7: Reserved
Must be 0.

Bit 6: EOP Input/Output Selection
Bit 6 selects whether the EOP signal is to be used as an output during DMA transfers on this channel or an input. EOP is typically used as an output, as was available on the PC/AT. The input function was added to support data communication and other devices that would like to trigger an autoinitialize when a collision or some other event occurs. The direction of EOP is switched when DACK is changed (when a different channel is granted the bus). There may be some overlap of the SIO driving the EOP signal along with the DMA slave. However, during this overlap, both devices drive the signal to a low level (inactive). For example, assume channel 2 is about to go inactive (DACK negating) and channel 1 is about to go active. In addition, assume that channel 2 is programmed for "EOP OUT" and channel 1 is programmed for "EOP IN". When channel 2's DACK is negated and channel 1's DACK is asserted, the SIO may be driving EOP to a low value on behalf of channel 2. At the same time the device connected to channel 1 is driving EOP in to the SIO, also at an inactive level. This overlap only lasts until the SIO EOP output buffer is tri-stated, and does not effect the DMA operation. Upon PCIRST#, bit 6 is set to 0-EOP output selected.

Bits[5:4]: DMA Cycle Timing Mode
The SIO supports four DMA transfer timings: ISA-compatible, type "A", "B", and "F". Each timing and its corresponding code are described below. Upon PCIRST#, compatible timing is selected and the value of these bits is "00". The cycle timings noted below are for a SYSCLK (8.33 MHz, maximum SYSCLK frequency). DMA cycles to ISA expansion bus memory defaults to compatible timing if the channel is programmed in one of the performance timing modes (type "A", "B", or "F").

Bits[5:4] = 00: Compatible Timing
Compatible timing is provided for DMA slave devices, that, due to some design limitation, cannot support one of the faster timings. Compatible timing runs at 9 SYSCLKs (1080 nsec/single cycle) and 8 SYSCLKs (960 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer.

Bits[5:4] = 01: Type "A" Timing
Type "A" timing is provided to allow shorter cycles to main memory (via the PCI Bus). Type "A" timing runs at 6 SYSCLKs (720 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed main memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter. However, it is expected that the DMA devices that provide the data access time or write data setup time should not require excess IOR# or IOW# command active time. Because of this, most ISA DMA devices should be able to use type "A" timing.

Bits[5:4] = 10: Type "B" Timing
Type "B" timing is provided for 8/16-bit ISA DMA devices that can accept faster I/O timing. Type "B" only works with fast main memory. Type "B" timing runs at 5 SYSCLKs (600 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "B" timing requires faster DMA slave devices than compatible timing. In Type "B" timing the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster.

Bits[5:4] = 11: Type "F" Timing
Type "F" timing provides high performance DMA transfer capability. Type "F" timing runs at 3 SYSCLKs (360 nsec/single cycle) during the repeated portion of a BLOCK or DEMAND mode transfer, resulting in a maximum data transfer rate of 8.33 MBytes/second.

Bits[3:2]: Addressing Mode
The SIO supports both 8- and 16-bit DMA device data sizes. Three data size options are programmable with bits[3:2]. Both the 8-bit I/O, "count by bytes" mode and the 16-bit I/O, "count by words" (address shifted) mode are ISA compatible. The 16-bit I/O, "count by bytes" mode is offered as an extension of the ISA compatible modes. Bits[3:2] = 10 is reserved. Byte assembly/disassembly is performed by the ISA control unit. Each of the data transfer size modes is discussed below.

Bits[3:2] = 00: 8-bit I/O, "Count By Bytes" Mode
In 8-bit I/O, "count by bytes" mode, the Current Address Register can be programmed to any address. The Current Byte/Word Count Register is programmed with the "number of bytes minus 1" to transfer.

Bits[3:2] = 01: 16-bit I/O, "Count By Words" (Address Shifted) Mode
In "count by words" mode (address shifted), the Current Address Register can be programmed to any even address, but must be programmed with the address value shifted right by one bit. The Low Page and High Page Registers are not shifted during DMA transfers. Thus, the least significant bit of the Low Page register is ignored when the address is driven out onto the bus. The Current Byte/Word Count Register is programmed with the number of words minus 1 to be transferred.

Bits[3:2] = 10: Reserved

Bits[3:2] = 11: 16-Bit I/O, "Count By Bytes" Mode
In 16-bit "count by bytes" mode, the Current Address Register can be programmed to any byte address. For most DMA devices, however, it should be programmed only to even addresses. If the address is programmed to an odd address, the DMA controller does a partial word transfer during the first and last transfer, if necessary. The bus controller does the Byte/Word assembly necessary to write any size memory device. In this mode, the Current Address Register is incremented or decremented by two and the byte count is decremented by the number of bytes transferred during each bus cycle. The Current Byte/Word Count Register is programmed with the "number of bytes minus 1" to be transferred. This mode should only be programmed for 16-bit ISA DMA slaves.

Bits[1:0]: DMA Channel Select
Bits[1:0] select the particular channel that will have its DMA Channel Extend Mode Register programmed with bits[7:2].

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.4 DR—DMA REQUEST REGISTER

Address Offset:	Channels 0-3–09h Channels 4-7–0D2h
Default Value:	Bits[1:0] = undefined, Bits[7:2] = 0
Attribute:	Write Only
Size:	4 bits

Each channel has a request bit in one of the two 4-bit DMA Request Registers. The Request Register is used by software to initiate a DMA request. The DMA responds to the software request as though DREQ[x] is asserted. These requests are non-maskable and subject to prioritization by the priority encoder network. Each register bit is set to 1 or 0 separately under software control or is set to 0 upon generation of a TC. The entire register is set to 0 upon PCIRST# or a Master Clear. It is not affected upon a RSTDRV output. To program a bit, the software loads the proper form of the data word. Bits[1:0] determine which channel Request Register will be written. In order to make a software request, the channel must be in Block Mode. The Request Register status for DMA1 and DMA2 is output on bits[7:4] of a Status Register read to the appropriate port.

Bits[7:3]: Reserved
Must be 0.

Bit 2: DMA Channel Service Request
Writing a 0 to bit 2 resets the individual software DMA channel request bit. Writing a 1 to bit 2 sets the request bit. The request bit for each DMA channel is reset to 0 upon a PCIRST# or a Master Clear.

Bits[1:0]: DMA Channel Select
Bits[1:0] select the DMA channel mode register to program with bit 2.

Bits	1	0	Channel
	0	0	Channel 0
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.5 MASK REGISTER—WRITE SINGLE MASK BIT

Address Offset: Channels 0-3–0Ah
Channels 4-7–0D4h
Default Value: Bits[1:0] = undefined, Bit 2 = 1, Bits[7:3] = 0
Attribute: Write Only
Size: 1 bit/channel

Each DMA channel has a mask bit that enables/disables an incoming DMA channel service request DREQ[x]. Two 4-bit registers store the current mask status for DMA1 and DMA2. Setting the mask bit disables the incoming DREQ[x] for that channel. Clearing the mask bit enables the incoming DREQ[x]. A channel's mask bit is automatically set when the Current Byte/Word Count register reaches terminal count (unless the channel is programmed for autoinitialization). Each mask bit may also be set or cleared under software control. The entire register is also set by a PCIRST# or a Master Clear. Setting the entire register disables all DMA requests until a clear mask register instruction allows them to occur. This instruction format is similar to the format used with the DMA Request Register.

Individually masking DMA channel 4 (DMA controller 2, channel 0) will automatically mask DMA channels [3:0], as this channel group is logically cascaded onto channel 4. Setting this mask bit disables the incoming DREQ's for channels [3:0].

Bits[7:3]: Reserved
Must be 0.

Bit 2: Channel Mask Select
When bit 2 is set to a 1, DREQ is disabled for the selected channel. When bit 2 is set to a 0, DREQ is enabled for the selected channel.

Bit[1:0]: DMA Channel Select
Bits[1:0] select the DMA Channel Mode Register to program with bit 2.

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.6 MASK REGISTER—WRITE ALL MASK BITS

Address Offset: Channels 0-3–0Fh
Channels 4-7–0DEh
Default Value: Bit[3:0] = 1, Bit[7:4] = 0
Attribute: Read/Write
Size: 4 bits

Writing to this register enables/disables incoming DREQ assertions. There are four mask bits per register, one for each channel. This permits all four channels to be simultaneously enabled/disabled instead of enabling/disabling each channel individually, as is the case with the Mask Register—Write Single Mask Bit.

Two 4-bit registers store the current mask status for DMA1 and DMA2. Unlike the Mask Register—Write Single Mask Bit, this register and includes a status read to check the current mask status of the selected DMA channel group. A channel's mask bit is automatically set to 1 when the Current Byte/Word Count Register reaches terminal count (unless the channel is programmed for autoinitialization). Bits[3:0] are set to 1 by a PCIRST# or a Master Clear. Setting bits[3:0] to 1 disables all DMA requests until a clear mask register instruction enables the requests.

Two important points should be taken into consideration when programming the mask registers. First, individually masking DMA channel 4 (DMA controller 2, channel 0) will automatically mask DMA channels [3:0], as this channel group is logically cascaded onto channel 4. Second, masking DMA controller 2 with a write to port 0DEh will also mask DREQ assertions from DMA controller 1 for the same reason. When DMA channel 4 is masked, so are DMA channels 0-3.

Bits[7:4]: Reserved
Must be 0.

Bits[3:0]: Channel Mask Bits
Setting the bit(s) to a 1 disables the corresponding DREQ(s). Setting the bit(s) to a 0 enables the corresponding DREQ(s). Bits[3:0] are set to 1 upon PCIRST# or Master Clear. When read, bits[3:0] indicate the DMA channel [3:0] ([7:4]) mask status.

Bit	Channel
0	0 (4)
1	1 (5)
2	2 (6)
3	3 (7)

NOTE:
Disabling channel 4 also disables channels 0-3 due to the cascade of DMA1 through channel 4 of DMA2.

4.2.7 DS—DMA STATUS REGISTER

Address Offset: Channels 0-3–08h
Channels 4-7–0D0h
Default Value: 00h
Attribute: Read Only
Size: 8 bits

Each DMA controller has a read-only DMA Status Register. This register indicates which channels have reached terminal count and which channels have a pending DMA request. Bits[3:0] are set every time the corresponding TC is reached by that channel. Bits[3:0] are set to 0 upon PCIRST# and on each status read. Bits[7:4] are set whenever their corresponding channel is requesting service.

Bits[7:4]: Channel Request Status
When a valid DMA request is pending for a channel (on its DREQ signal line), the corresponding bit is set to 1. When a DMA request is not pending for a particular channel, the corresponding bit is set to 0. The source of the DREQ may be hardware, a timed-out block transfer, or a software request. Note that channel 4 does not have DREQ or DACK lines, so the response for a read of DMA2 status for channel 4 is irrelevant.

Bit	Channel
4	0
5	1 (5)
6	2 (6)
7	3 (7)

Bits[3:0]: Channel Terminal Count Status
When a channel reaches terminal count (TC), its status bit is set to 1. If TC has not been reached, the status bit is set to 0. Note that channel 4 is programmed for cascade, and is not used for a DMA transfer. Therefore, the TC bit response for a status read on DMA2 for channel 4 is irrelevant.

Bit	Channel
0	0
1	1 (5)
2	2 (6)
3	3 (7)

4.2.8 DMA BASE AND CURRENT ADDRESS REGISTERS (8237 COMPATIBLE SEGMENT)

Address Offset: DMA Channel 0–000h
DMA Channel 1–002h
DMA Channel 2–004h
DMA Channel 3–006h
DMA Channel 4–0C0h
DMA Channel 5–0C4h
DMA Channel 6–0C8h
DMA Channel 7–0CCh
Default Value: All bits undefined
Attribute: Read/Write
Size: 16 bits per channel

Each channel has a 16-bit Current Address Register. This register contains the value of the 16 least significant bits of the full 32-bit address used during DMA transfers. The address is automatically incremented or decremented after each transfer and the intermediate values of the address are stored in the Current Address Register during the transfer. This register is written to or read from by the PCI Bus or ISA Bus master in successive 8-bit bytes. The programmer must issue the "Clear Byte Pointer Flip-Flop" command to reset the internal byte pointer and correctly align the write prior to programming the Current Address Register. After clearing the Byte Pointer Flip-Flop, the first write to the Current Address Register programs the low byte, bits[7:0], and the second write programs the high byte, bits[15:8]. This procedure also applies to read cycles. It may also be reinitialized by an Autoinitialize back to its original value. Autoinitialize takes place only after a TC or EOP.

Each channel has a Base Address Register located at the same port address as the corresponding Current Address Register. These registers store the original value of their associated Current Address Registers. During autoinitialize these values are used to restore the Current Address Registers to their original values. The Base Registers are written simultaneously with their corresponding Current Address Register in successive 8-bit bytes. The Base Registers are write-only.

In Scatter/gather mode, these registers store the lowest 16 bits of the current memory address. During a Scatter/gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[15:0]: Base and Current Address [15:0]
These bits represent the 16 least significant address bits used during DMA transfers. Together with the DMA Low Page Register, they form the ISA-compatible 24-bit DMA address. As an extension of the ISA compatible functionality, the DMA High Page Register completes the 32-bit address needed when implementing SIO extensions such as DMA to the PCI Bus slaves that can take advantage of full 32-bit addressability. Upon PCIRST# or Master Clear, the value of these bits is 0000h.

4.2.9 DMA BASE AND CURRENT BYTE/WORD COUNT REGISTERS (8237 COMPATIBLE SEGMENT)

Address Offset:	DMA Channel 0–001h
	DMA Channel 1–003h
	DMA Channel 2–005h
	DMA Channel 3–007h
	DMA Channel 4–0C2h
	DMA Channel 5–0C6h
	DMA Channel 6–0CAh
	DMA Channel 7–0CEh
Default Value:	All bits undefined
Attribute:	Read/Write
Size:	16 bits per channel

Each channel has a 16-bit Current Byte/Word Count Register. This register determines the number of transfers to be performed. The actual number of transfers is one more than the number programmed in the Current Byte/Word Count Register (i.e., programming a count of 100 results in 101 transfers). The Byte/Word count is decremented after each transfer. The intermediate value of the Byte/Word count is stored in the register during the transfer. When the value in the register goes from zero to 0FFFFh, a TC is generated.

Following the end of a DMA service the register may also be re-initialized by an autoinitialization back to its original value. Autoinitialize can only occur when a TC occurs. If it is not autoinitialized, this register has a count of FFFFh after TC.

When the Extended Mode Register is programmed for, or defaulted to, transfers to/from an 8-bit I/O, the Byte/Word count indicates the number of bytes to be transferred.

When the Extended Mode Register is programmed for, or defaulted to, transfers to/from a 16-bit I/O, with shifted address, the Byte/Word count indicates the number of 16-bit words to be transferred.

When the Extended Mode Register is programmed for transfers to/from a 16-bit I/O, the Byte/Word Count indicates the number of bytes to be transferred. The number of bytes does not need to be a multiple of two or four in this case.

Each channel has a Base Byte/Word Count Register located at the same port address as the corresponding Current Byte/Word Count Register. These registers store the original value of their associated Current Byte/Word Count Registers. During Autoinitialize these values are used to restore the Current registers to their original values. The Base registers are written simultaneously with their corresponding Current register in successive 8-bit bytes. The Base registers cannot be read by any external agents.

In Scatter/gather mode, these registers store the 16 bits of the current Byte/Word count. During Scatter/gather transfer, the DMA will load a reserve buffer into the base Byte/Word Count register.

Bits[15:0]: Base and Current Byte/Word Count
These bits represent the 16 byte/word count bits used when counting down a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 0000h.

4.2.10 DMA MEMORY BASE LOW PAGE AND CURRENT LOW PAGE REGISTERS

Register Name:	DMA Memory Current Low Page Register (Read/Write) DMA Memory Base Low Page Register (Write Only)
Address Offset:	DMA Channel 0–087h DMA Channel 1–083h DMA Channel 2–081h DMA Channel 3–082h DMA Channel 5–08Bh DMA Channel 6–089h DMA Channel 7–08Ah
Default Value:	All bits undefined
Size:	8 bits per channel

Each channel has an 8-bit Low Page Register. The DMA memory Low Page Register contains the eight second most-significant bits of the 32-bit address. The register works in conjunction with the DMA controller's High Page Register and Current Address Register to define the complete (32-bit) address for the DMA channel. This 8-bit register is read or written directly. It may also be re-initialized by an autoinitialize back to its original value. Autoinitialize takes place only after a TC or EOP.

Each channel has a Base Low Page Address Register located at the same port address as the corresponding Current Low Page Register. These registers store the original value of their associated Current Low Page Registers. During autoinitialization, these values are used to restore the Current Low Page Registers to their original values. The 8-bit Base Low Page Registers are written simultaneously with their corresponding Current Low Page Register by the microprocessor. The Base Low Page registers are write only.

During Scatter/gather, these registers store the 8 bits from the third byte of the current memory address. During a Scater-Gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[7:0]: DMA Low Page and Base Low Page [23:16]
These bits represent the eight second most significant address bits when forming the full 32-bit address for a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 00h.

4.2.11 DMA MEMORY BASE HIGH PAGE AND CURRENT HIGH PAGE REGISTERS

Register Name:	DMA Memory Current High Page Register (Read/Write) DMA Memory Base High Page Register (Write Only)
Address Offset:	DMA Channel 0–0487h DMA Channel 1–0483h DMA Channel 2–0481h DMA Channel 3–0482h DMA Channel 5–048Bh DMA Channel 6–0489h DMA Channel 7–048Ah
Default Value:	All bits undefined
Size:	8 bits per channel

Each channel has an 8-bit Current High Page Register. The DMA memory Current High Page Register contains the eight most significant bits of the 32-bit address. The register works in conjunction with the DMA controller's Current Low Page Register and Current Address Register to define the complete (32-bit) address for the DMA channels and corresponds to the Current Address Register for each channel. This 8-bit register is read or written directly. It may also be autoinitialized back to its original value. Autoinitialize takes place only after a TC or EOP.

This register is set to 0 during the programming of both the Current Low Page Register and the Current Address Register. Thus, if this register is not programmed after the other address and Low Page Registers are programmed, then its value is 00h. In this case, the DMA channel operates the same as an 82C37 (from an addressing standpoint). This is the address compatibility mode.

If the high 8 bits of the address are programmed after the other addresses, then the channel modifies its operation to increment (or decrement) the entire 32-bit address. This is unlike the 82C37 "Page" register in the original PCs which could only increment to a 64 KByte boundary for 8-bit channels or 128 KByte boundary for 16-bit channels. This is extended address mode. In this mode, the ISA Bus controller generates the signals MEMR# and MEMW# only for addresses below 16 MBytes.

Each channel has a Base High Page Register located at the same port address as the corresponding Current High Page Register. These registers store the original value of their associated Current High Page Registers. During autoinitialize, these values are used to restore the Current High Page Registers to their original values. The 8-bit Base High Page Registers are written simultaneously with their corresponding Current High Page Register. The Base High Page Registers are write only.

During Scatter/Gather, these registers store the 8 bits from the highest byte of the current memory address. During a Scatter/Gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[7:0]: DMA High Page and Base High Page [31:24]
These bits represent the eight most-significant address bits when forming the full 32-bit address for a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 00h.

4.2.12 DMA CLEAR BYTE POINTER REGISTER

Address Offset:	Channels 0–3–00Ch Channels 4–7–0D8h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

Writing to this register executes the clear byte pointer command. This command is executed prior to writing or reading new address or word count information to the DMA. This command initializes the byte pointer flip-flop to a known state so that subsequent accesses to register contents will address upper and lower bytes in the correct sequence.

The clear byte pointer command clears the internal latch used to address the upper or lower byte of the 16-bit Address and Word Count Registers. The latch is also cleared at power on by PCIRST# and by the Master Clear command. The Host CPU may read or write a 16-bit DMA controller register by performing two consecutive accesses to the I/O port. The Clear Byte Pointer command precedes the first access. The first I/O write to a register port loads the least significant byte, and the second access automatically accesses the most significant byte.

When DMA registers are being read or written, two Byte Pointer flip-flops are used. One flip-flop is for Channels 0–3 and one for Channels 4–7. Both of these act independently. There are separate software commands for clearing each of them (0Ch for Channels 0–3, 0D8h for Channels 4–7).

Bits[7:0]: Clear Byte Pointer
No specific pattern. Command enabled with a write to the I/O port address.

4.2.13 DMC—DMA MASTER CLEAR REGISTER

Address Offset:	Channel 0–3–00Dh Channel 4–7–0DAh
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bit

This software instruction has the same effect as the hardware Reset. The Command, Status, Request, and Internal First/Last Flip-Flop registers are cleared and the Mask Register is set. The DMA controller enters the idle cycle. There are two independent Master Clear Commands; 0Dh acts on Channels 0–3, and 0DAh acts on Channels 4–7.

Bits[7:0]: Master Clear
No specific pattern. Command enabled with a write to the I/O port address.

4.2.14 DCM—DMA CLEAR MASK REGISTER

Address Offset:	Channel 0–3–00Eh Channel 4–7–0DCh
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bit

This command clears the mask bits of all four channels, enabling them to accept DMA requests. I/O port 0Eh is used for Channels 0–3 and I/O port 0DCh is used for Channels 4–7.

Bits[7:0]: Clear Mask Register
No specific pattern. Command enabled with a write to the I/O port address.

4.2.15 SCATTER/GATHER COMMAND REGISTER

Register Name: DMA Scatter Gather Command

Address Offset: Channels 0 default address—0410h
Channels 1 default address—0411h
Channels 2 default address—0412h
Channels 3 default address—0413h
Channels 5 default address—0415h
Channels 6 default address—0416h
Channels 7 default address—0417h

Default Value: 00h
Attribute: Write Only, Relocatable
Size: 8 bits

The Scatter/Gather Command Register controls operation of the descriptor table aspect of scatter/gather transfers. This register can be used to start and stop a scatter/gather transfer. The register can also be used to select between IRQ13 and EOP to be asserted following a terminal count. The current scatter/gather transfer status can be read in the scatter/gather channel's corresponding Scatter/Gather Status Register. After a PCIRST# or Master Clear, IRQ13 is disabled and EOP is enabled.

Bit 7: IRQ13/EOP Select
Bit 7, if enabled via bit 6 of this register, selects whether EOP or IRQ13 is asserted at termination caused by a last buffer expiring. The last buffer can be either the last buffer in the list or the last buffer loaded in the DMA while it is suspended. If bit 7 = 1 (and bit 6 = 1), EOP is asserted when the last buffer is completed. If bit 7 = 0 (and bit 6 = 1), IRQ13 is asserted when the last buffer is completed.

EOP can be used to alert an expansion bus I/O device that a scatter/gather termination condition was reached. The I/O device, in turn, can assert its own interrupt request line to invoke a dedicated interrupt handling routine. IRQ13 should be used when the CPU needs to be notified directly.

Following PCIRST#, or Master Clear, the value stored for this bit is "1", and EOP is selected. Bit-6 must be set to a "1" to enable this bit during a S/G Command register write. When bit 6 is a "0" during the write, bit 7 will not have any effect on the current EOP/IRQ13 selection.

Bit 6: IRQ13/EOP Programming Enable
Enabling IRQ13/EOP programming allows initialization or modification of the S/G termination handling bits. When bit 6 = 0, bit 7 does not affect the state of IRQ13 or EOP assertion. When bit 6 = 1, bit 7 determines the termination handling following a terminal count.

Bits[5:2]: Reserved
Must be 0.

Bits[1:0]: Scatter/Gather Commands
This 2-bit field is used to start and stop scatter/gather.

Bits[1:0] = 00: No S/G operation
No S/G command operation is performed. Bits[7:6] may still be used to program IRQ13/EOP selection.

Bits[1:0] = 01: Start S/G Command
The Start command initiates the scatter/gather process. Immediately after the start command is issued (setting bits[1:0] to 01), a request is issued to fetch the initial buffer from the descriptor table to fill the Base Register set in preparation for performing a transfer. The buffer prefetch request has the same priority with respect to other channels as the DREQ it is associated with. Within the channel, DREQ is higher in priority than a prefetch request.

The Start command assumes the Base and Current registers are both empty and will request a prefetch automatically. Note that this command also sets the Scatter/Gather Status Register to S/G Active, Base Empty, Current Empty, not Terminated, and Next Null Indicator to 0. The EOP/IRQ13 bit will still reflect the last value programmed.

Bits[1:0] = 10: Stop S/G Command
The Stop command halts a Scatter/gather transfer immediately. When a Stop command is given, the Terminate bit in the S/G Status register and the DMA channel mask bit are both set.

Bits[1:0] = 11: Reserved

4.2.16 SCATTER/GATHER STATUS REGISTER

Address Offset:	Channels 0 default address—0418h Channels 1 default address—0419h Channels 2 default address—041Ah Channels 3 default address—041Bh Channels 5 default address—041Dh Channels 6 default address—041Eh Channels 7 default address—041Fh
Default Value:	00h
Attribute:	Read Only, Relocatable
Size:	8-bits

The Scatter/Gather Status Register contains information on the scatter/gather transfer status. This register provides dynamic status information on S/G transfer activity, the current and base buffer state, S/G transfer termination, and the End of the List indicator.

An Active bit is set to "1" after the S/G Start command is issued. The Active bit will be "0" before the initial Start command, following a terminal count, and after a S/G Stop command is issued. The Current Register and Base Register Status bits indicate whether the corresponding register has a buffer loaded. It is possible for the Base Register Status to be set while the Current Register Status is cleared. When the Current Register transfer is complete, the Base Register will not be moved into the Current Register until the start of the next data transfer. Thus, the Current Register State is empty (cleared), while the Base Register State is full (set). The Terminate bit is set active after a Stop command, after TC for the last buffer in the list, and both Base and Current Registers have expired. The EOP and IRQ13 bits indicate which end of process indicator will be used to alert the system of an S/G process termination. The EOL status bit is set if the DMA controller has loaded the last buffer of the Link List. Following PCIRST#, or Master Clear, each bit in this register is reset to "0".

Bit 7: Next Link Null Indicator
If the next scatter/gather descriptor fetched from memory during a fetch operation has the EOL value set to 1, the current value of the Next Link Register is not overwritten. Instead, bit 7 of the channel's Scatter/Gather Status Register is set to a 1. If the fetch returns a EOL value set to 0, this bit is set to 0. This status bit is written after every fetch operation. Following PCIRST#, or Master Clear, this bit is set to 0. This bit is also cleared by an S/G Start Command write to the Scatter/Gather Command Register.

Bit 6: Reserved

Bit 5: Issue IRQ13/EOP on Last Buffer
When bit 5 = 0, EOP was either defaulted to at reset or selected through the Scatter/Gather Command Register as the S/G process termination indicator. EOP is issued when a terminal count occurs or following the Stop Command. When bit 5 = 1, an IRQ13 is issued to alert the CPU of this same status.

Bit 4: Reserved

Bit 3: Scatter/Gather Base Register Status
When bit 3 = 0, the Base Register is empty. When bit 3 = 1, the Base Register has a buffer link loaded. Note that the Base Register State may be set while the Current Register state is cleared. This condition occurs when the Current Register expires following a transfer. The Base Register will not be moved into the Current Register until the start of the next DMA transfer.

Bit 2: Scatter/Gather Current Register Status
When bit 2 = 0, the Current Register is empty. When bit 2 = 1, the Current Register has a buffer link loaded and is considered full. Following PCIRST#, bit 2 is set to 0.

Bit 1: Reserved

Bit 0: Scatter/Gather Active
The Scatter/gather Active bit indicates the current S/G transfer status. Bit 0 is set to a 1 after an S/G Start Command is issued. Bit 0 is set to 0 before the Start Command is issued. Bit 0 is 0 after terminal count on the last buffer on the channel is reached. Bit 0 is also 0 after an S/G Stop Command has been issued. Following a PCIRST# or Master Clear, this bit is 0.

4.2.17 SCATTER/GATHER DESCRIPTOR TABLE POINTER REGISTER

Address Offset: Channel 0 default address—0420h–0423h
Channel 1 default address—0424h–0424h
Channel 2 default address—0428h–042Bh
Channel 3 default address—042Ch–042Ch
Channel 5 default address—0434h–0437h
Channel 6 default address—0438h–043Bh
Channel 7 default address—043Ch–043Fh
Default Value: All bits undefined
Attribute: Read/Write, Relocatable
Size: 32 bits

The Scatter/Gather Descriptor Table Pointer Register contains the 32-bit pointer address to the first scatter/gather descriptor entry in the descriptor table in memory. Before the start of a S/G transfer, this register should be programmed to point to the first descriptor in the Scatter/Gather Descriptor Table. Following a S/G Start command, the SIO reads the first SGD entry. Subsequently, at the end of the each buffer block transfer, the contents of the SGD Table pointer registers are incremented by 8 until the end of the SGD Table is reached.

The Scatter/Gather Descriptor Table Pointer Registers can be programmed with a single 32-bit PCI write.

Following a prefetch to the address pointed to by the channel's Scatter/Gather Descriptor Table Pointer Register, the new memory address is loaded into the Base Address Register, the new Byte Count is loaded into the Base Byte Count Register, and the newly fetched next scatter/gather descriptor replaces the current next scatter/gather value.

The end of the Scatter/Gather Descriptor Table is indicated by an End of Table field having a MSB equal to 1. When this value is read during a scatter/gather descriptor fetch, the current scatter/gather descriptor value is not replaced. Instead, bit 7 of the channel's Status Register is set to a 1, when the EOL is read from memory.

Bits[31:0]:
The Scatter/Gather Descriptor Table Pointer Register contains a 32-bit pointer address to the main memory location where the software maintains the Scatter Gather Descriptors for the linked-list buffers. Bits[31:0] correspond to A[31:0] on the PCI.

4.2.18 SCATTER/GATHER INTERRUPT STATUS REGISTER

Address Offset: 040Ah
Default Value: 00h
Attribute: Read Only, Relocatable
Size: 8 bits

The Scatter/Gather Interrupt Status Register is a read only register and is used to indicate the source (channel) of a DMA Scatter/Gather interrupt on IRQ13. The DMA controller drives IRQ13 active after reaching terminal count during a Scatter/Gather transfer. It does not drive IRQ13 active during the initial programming sequence that loads the Base registers.

Bit 7: Channel 7 Interrupt Status
When this bit is set to a 1, Channel 7 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 6: Channel 6 Interrupt Status
When this bit is set to a 1, Channel 6 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 5: Channel 5 Interrupt Status
When this bit is set to a 1, Channel 5 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 4: Reserved
Read as 0.

Bit 3: Channel 3 Interrupt Status
When this bit is set to a 1, Channel 3 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 2: Channel 2 Interrupt Status
When this bit is set to a 1, Channel 2 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 1: Channel 1 Interrupt Status
When this bit is set to a 1, Channel 1 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 0: Channel 0 Interrupt Status
When this bit is set to a 1, Channel 0 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

4.3 Timer Register Description

The SIO contains three counters that are equivalent to those found in the 82C54 Programmable Interval Timer. The Timer registers control these counters and can be accessed from either the ISA Bus via ISA I/O space or the PCI Bus via PCI I/O space.

This section describes the counter/timer registers on the SIO. The counter/timer operations are further described in Section 5.7, Timer Unit.

4.3.1 TCW—TIMER CONTROL WORD REGISTER

Address Offset: 043h
Default Value: All bits undefined
Attribute: Write Only
Size: 8 bits

The Timer Control Word Register specifies the counter selection, the operating mode, the counter byte programming order and size of the count value, and whether the counter counts down in a 16-bit or binary-coded decimal (BCD) format. After writing the control word, a new count can be written at any time. The new value will take effect according to the programmed mode.

There are six programmable counting modes. Typically, the SIO Timer Counters 0 and 2 are programmed for Mode 3, the Square Wave Mode, while Counter 1 is programmed in Mode 2, the Rate Generator Mode.

Two special commands are selected through the Timer Control Word Register. The Read Back Command (see Section 4.3.1.1) is selected when bits[7:6] are both 1 and the Counter Latch Command (see Section 4.3.1.2) is selected when bits[5:4] are both 0. When either of these two commands are selected, the meaning of the other bits in the register changes.

Bits 4 and 5 are also used to select the count register programming mode. The read/write selection chosen with the control word indicates the programming sequence that must follow when initializing the specified counter. If a counter is programmed to read/write two byte counts, note that a program must not transfer control between writing the first and second byte to another routine that also writes into that same counter. Otherwise, the counter will be loaded with an incorrect count. The count must always be completely loaded with both bytes.

Bits 6 and 7 are also used to select the counter for the control word being written.

Following PCIRST#, the control words for each register are undefined. Each timer must be programmed to bring it into a known state. However, each counter OUT signal is set to 0 following PCIRST#. The SPKR output, interrupt controller input IRQ0 (internal), bit 5 of port 061h, and the internally generated refresh request are each set to 0 following PCIRST#.

Bits[7:6]: Counter Select
The Counter Selection bits select the counter the control word acts upon as shown below. The Read Back Command is selected when bits[7:6] are both 1.

Bit	7	6	Function
	0	0	Counter 0 select
	0	1	Counter 1 select
	1	0	Counter 2 select
	1	1	Read Back Command (see Section 4.3.1.1)

Bits[5:4]: Read/Write Select
Bits[5:4] are the read/write control bits. The Counter Latch Command is selected when bits[5:4] are both 0. The read/write options include r/w least significant byte, r/w most significant byte, or r/w the LSB and then the MSB. The actual counter programming is done through the counter I/O port (040h, 041h, and 042h for counters 0, 1, and 2, respectively).

Bit	5	4	Function
	0	0	Counter Latch Command (see Section 4.3.1.2)
	0	1	R/W Least Significant Byte (LSB)
	1	0	R/W Most Significant Byte (MSB)
	1	1	R/W LSB then MSB

Bits[3:1]: Counter Mode Selection
Bits[3:1] select one of six possible modes of operation for the counter as shown below.

Bit 3	2	1	Mode	Function
0	0	0	0	Out signal on end of count (= 0)
0	0	1	1	Hardware retriggerable one-shot
X	1	0	2	Rate generator (divide by n counter)
X	1	1	3	Square wave output
1	0	0	4	Software triggered strobe
1	0	1	5	Hardware triggered strobe

Bit 0: Binary/BCD Countdown Select
When bit 0 = 0, a binary countdown is used. The largest possible binary count is 2^{16}. When bit 0 = 1, a binary coded decimal (BCD) count is used. The largest BCD count allowed is 10^4.

4.3.1.1 Read Back Command

The Read Back Command is used to determine the count value, programmed mode, and current states of the OUT pin and Null count flag of the selected counter or counters. The Read Back Command is written to the Timer Control Word Register which latches the current states of the above mentioned variables. The value of the counter and its status may then be read by I/O access to the counter address.

Status and/or count may be latched on one, two, or all three of the counters by selecting the counter during the register write. The count latched remains latched until read, regardless of further latch commands. The count must be read before newer latch commands latch a new count. The status latched by the Read Back Command also remains latched until after a read to the counter's I/O port by reading the Counter Access Ports Register. Thus, the status and count are unlatched only after a counter read of the Timer Status Byte Format Register, the Counter Access Ports Register, or the Timer Status Byte Register and Counter Access Ports Register in succession.

Both count and status of the selected counter(s) may be latched simultaneously by setting both bit 5 and bit 4 to 0. This is functionally the same as issuing two consecutive, separate Read Back Commands. As mentioned above, if multiple count and/or status Read Back Commands are issued to the same counter(s) without any intervening reads, all but the first are ignored.

If both count and status of a counter are latched, the first read operation from that counter returns the latched status, regardless of which was latched first. The next one or two reads (depending on whether the counter is programmed for one or two byte counts) returns the latched count. Subsequent reads return an unlatched count.

NOTE:
The Timer Counter Register bit definitions are different during the Read Back Command than for a normal Timer Counter Register write.

Bits[7:6]: Read Back Command
When bits[7:6] are both 1, the Read Back Command is selected during a write to the Timer Control Word Register. As noted above, the normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the Read Back Command is selected. Following the Read Back Command, I/O reads from the selected counter's I/O addresses produce the current latch status, the current latched count, or both if bits 4 and 5 are both 0.

Bit 5: Latch Count of Selected Counters
When bit 5 = 1, the current count value of the selected counters will be latched. When bit 4 = 0, the status will not be latched.

Bit 4: Latch Status of Selected Counters
When bit 4 = 1, the status of the selected counters will be latched. When bit 4 = 0, the status will not be latched. The status byte format is described in Section 4.3.2, Interval Timer Status Byte Format Register.

Bit 3: Counter 2 Select
When bit 3 = 1, Counter 2 is selected for the latch command selected with bits 4 and 5. When bit 3 = 0, status and/or count will not be latched.

Bit 2: Counter 1 Select
When bit 2 = 1, Counter 1 is selected for the latch command selected with bits 4 and 5. When bit 2 = 0, status and/or count will not be latched.

Bit 1: Counter 0 Select
When bit 1 = 1, Counter 0 is selected for the latch command selected with bits 4 and 5. When bit 1 = 0, status and/or count will not be latched.

Bit 0: Reserved
Must be 0.

4.3.1.2 Counter Latch Command

The Counter Latch Command latches the current count value at the time the command is received. This command is used to insure that the count read from the counter is accurate (particularly when reading a two-byte count). The count value is then read from each counter's count register (via the Counter Access Ports Register). One, two or all three counters may be latched with one Counter Latch Command.

If a Counter is latched once and then, some time later, latched again before the count is read, the second Counter Latch Command is ignored. The count read will be the count at the time the first Counter Latch Command was issued.

The count must be read according to the programmed format. Specifically, if the Counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other (read, write, or programming operations for other counters may be inserted between the reads).

NOTES:

1. If a counter is programmed to read/write two-byte counts, a program must not transfer control between reading the first and second byte to another routine that also reads from that same counter. Otherwise, an incorrect count will be read. Finish reading the latched two-byte count before transferring control to another routine.
2. The Timer Counter Register bit definitions are different during the Counter Latch Command than for a normal Timer Counter Register write.

Bits[7:6]: Counter Selection
Bits 6 and 7 are used to select the counter for latching.

Bit 7	6	Function
0	0	latch counter 0 select
0	1	latch counter 1 select
1	0	latch counter 2 select
1	1	Read Back Command select

Bits[5:4]: Counter Latch Command
When bits[5:4] are both 0, the Counter Latch Command is selected during a write to the Timer Control Word Register. As noted above, the normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the Counter Latch Command is selected. Following the Counter Latch Command, I/O reads from the selected counter's I/O addresses produce the current latched count.

Bits[3:0]: Reserved
Must be 0.

4.3.2 INTERVAL TIMER STATUS BYTE FORMAT REGISTER

Address Offset:	Counter 0–040h Counter 1–041h Counter 2–042h
Default Value:	Bits[6:0] = X, Bit 7 = 0
Attribute:	Read Only
Size:	8 bits per counter

Each counter's status byte can be read following an Interval Timer Read Back Command. The Read Back Command is programmed through the Timer Control Word Register. If latch status is chosen (bit 4 = 0, Read Back Command) as a read back option for a given counter, the next read from the counter's Counter Access Ports Register returns the status byte. The status byte returns the countdown type, either BCD or binary; the counter operational mode; the read/write selection status; the Null count, also referred to as the count register status; and the current state of the counter OUT pin.

Bit 7: Counter OUT Pin State
When this bit is a 1, the OUT pin of the counter is also a 1. When this bit is a 0, the OUT pin of the counter is also a 0.

Bit 6: Count Register Status
Null Count, also referred to as the Count Status Register, indicates when the last count written to the Count Register (CR) has been loaded into the Counting Element (CE). The exact time this happens depends on the counter mode, but until the count is loaded into the counting element (CE), it can't be read from the counter. If the count is latched or read before the load time, the count value returned will not reflect the new count written to the register. When bit 6 = 0, the count has been transferred from CR to CE and is available for reading. When bit 6 = 1, the Null count condition exists. The count has not been transferred from CR to CE and is not yet available for reading.

Bits[5:4]: Read/Write Selection Status
Bits[5:4] reflect the read/write selection made through bits[5:4] of the control register. The binary codes returned during the status read match the codes used to program the counter read/write selection.

Bit 5	4	Function
0	0	Counter Latch Command
0	1	R/W Least Significant Byte (LSB)
1	0	R/W Most Significant Byte (MSB)
1	1	R/W LSB then MSB

Bits[3:1]: Mode Selection Status
Bits[3:1] return the counter mode programming. The binary code returned matches the code used to program the counter mode, as listed under the bit function above.

Bit 3	2	1	Mode Selected
0	0	0	0
0	0	1	1
X	1	0	2
X	1	1	3
1	0	0	4
1	0	1	5

Bit 0: Countdown Type Status
Bit reflects the current countdown type; either 0 for binary countdown or a 1 for binary coded decimal (BCD) countdown.

4.3.3 COUNTER ACCESS PORTS REGISTER

Address Offset:	Counter 0, System Timer-040h Counter 1, Refresh Request-041h Counter 2, Speaker Tone-042h
Default Value:	All bits undefined
Attribute:	Read/Write
Size:	8 bits per counter

Each of these I/O ports is used for writing count values to the Count Registers; reading the current count value from the counter by either an I/O read, after a counter-latch command, or after a Read Back Command; and reading the status byte following a Read Back Command.

Bits[7:0]: Counter Port Bit[x]
Each counter I/O port address is used to program the 16-bit Count Register. The order of programming, either LSB only, MSB only, or LSB then MSB, is defined with the Interval Counter Control Register at I/O port address 043h. The counter I/O port is also used to read the current count from the Count Register, and return the status of the counter programming following a Read Back Command.

4.3.4 BIOS TIMER REGISTER

Register Location:	Default = 78h–7Bh (Dword aligned)
Default Value:	0000xxxxh
Attribute:	Read/Write, Programmable
Size:	32 bit

A write to the BIOS Timer initiates a counting sequence. The timer can be initiated by writing either a 16-bit data portion or the entire 32-bit register (the upper 16 bits are don't cares). Bits[15:0] can be written with the initial count value to start the timer or read to check the current count value. It is the programmer's responsiblity to ensure that all 16 bits are written at the same time. After data is written into BIOS timer, the timer will start decrementing until it reaches zero. It will "freeze" at zero until the new count value is written.

The BIOS Timer consists of a single 32-bit register mapped in the I/O space on the location determined by the value written into the BIOS Timer Base Address Register. Bit 0 of the BIOS Timer Base Address Register enables/disables accesses to the BIOS Timer and must be 1 to enable access to the BIOS Timer Register. When the BIOS Timer is enabled, PCI accesses to the BIOS Timer Register do not flow through to the ISA Bus. If the BIOS Timer is disabled, accesses to the addresses assigned to the BIOS Timer Register flow through to the ISA bus. Note, however, that the counter continues to count normally.

Bits[31:16]: Reserved
Read as 0.

Bits[15:0]:
Timer count value.

4.4 Interrupt Controller Register Description

The SIO contains an ISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The interrupt registers control the operation of the interrupt controller and can be accessed from the PCI Bus via PCI I/O space. In addition, some of the registers can be accessed from the ISA Bus via ISA I/O space. The bus access for each register is listed in Table 4.

4.4.1 ICW1—INITIALIZATION COMMAND WORD 1 REGISTER

Register Location:	INT CNTRL-1–020h INT CNTRL-2–0A0h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits per controller

A write to Initialization Command Word 1 starts the interrupt controller initialization sequence. Addresses 020h and 0A0h are referred to as the base addresses of CNTRL-1 and CNTRL-2, respectively.

An I/O write to the CNTRL-1 or CNTRL-2 base address with bit 4 equal to 1 is interpreted as ICW1. For SIO-based ISA systems, three I/O writes to "base address + 1" must follow the ICW1. The first write to "base address + 1" performs ICW2, the second write performs ICW3, and the third write performs ICW4.

ICW1 starts the initialization sequence during which the following automatically occur:

a. The edge sense circuit is reset. This means that following initialization, an interrupt request (IRQ) input must make a low-to-high transition to generate an interrupt.

b. The Interrupt Mask register is cleared.

c. IRQ7 input is assigned priority 7.

d. The slave mode address is set to 7.

e. Special Mask Mode is cleared and Status Read is set to IRR.

f. If IC4 was set to 0, then all functions selected by ICW4 are set to 0. However, ICW4 must be programmed in the SIO implementation of this interrupt controller, and IC4 must be set to a 1.

ICW1 has three significant functions within the SIO interrupt controller configuration. ICW4 is needed, so bit 0 must be programmed to a 1. There are two interrupt controllers in the system, so bit 1, SNGL, must be programmed to a 0 on both CNTRL-1 and CNTRL-2, to indicate a cascade configuration. LTIM, the interrupt controller IRQ edge/level detection control bit, defines the IRQ sensing mode for each controller. When bit 3 is a 0, each IRQ line on the selected controller is programmed for edge-triggered mode. This mode is signified by a low-to-high transition on an IRQ input line. When bit 3 is a 1, the controller is programmed in level-triggered mode, where a high level on an IRQ input indicates the presence of an interrupt request. LTIM is global for each controller. The incoming IRQs are either all edge-triggered or all level-triggered. Bit D4 must be a 1 when programming ICW1. OCW2 and OCW3 are also addressed at the same port as ICW1. This bit indicates that ICW1, and not OCW2 or OCW3, will be programmed during the write to this port.

Bit 2, ADI, and bits[7:5], A7-A5, are specific to an MSC-85 implementation. These bits are not used by the SIO interrupt controllers. Bits[7:5,2] should each be initialized to 0.

In the 82378ZB, bit 3, the LTIM bit, is not used by the interrupt controller and is always read as a 1.

Bits[7:5]: ICW/OCW Select
A7–A5 are MCS-85 implementation specific bits. They are not needed by the SIO. These bits should be 000 when programming the SIO.

Bit 4: ICW/OCW Select
Bit 4 must be a 1 to select ICW1. After the fixed initialization sequence to ICW1, ICW2, ICW3, and ICW4, the controller base address is used to write to OCW2 and OCW3. Bit 4 is a 0 on writes to these registers. A 1 on this bit at any time will force the interrupt controller to interpret the write as an ICW1. The controller will then expect to see ICW2, ICW3, and ICW4.

Bit 3: LTIM (Edge/Level Bank Select)
Ignored for the SIO.

Bit 2: ADI
Ignored for the SIO.

Bit 1: SNGL (Single or Cascade)
SNGL must be programmed to a 0 to indicate that two interrupt controllers are operating in cascade mode on the SIO.

Bit 0: IC4 (ICW4 Write Required)
This bit must be set to a 1. IC4 indicates that ICW4 needs to be programmed. The SIO requires that ICW4 be programmed to indicate that the controllers are operating in an 80x86 type system.

4.4.2 ICW2—INITIALIZATION COMMAND WORD 2 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits per controller

ICW2 is used to initialize the interrupt controller with the five most significant bits of the interrupt vector address. The value programmed for bits[7:3] is used by the Host CPU to define the base address in the interrupt vector table for the interrupt routines associated with each IRQ on the controller. Typical ISA ICW2 values are 04h for CNTRL-1 and 70h for CNTRL-2.

Bits[7:3]: Interrupt Vector Base Address
Bits[7:3] define the base address in the interrupt vector table for the interrupt routines associated with each interrupt request level input. For CNTRL-1, a typical value is 00001, and for CNTRL-2, 10000.

The interrupt controller combines a binary code representing the interrupt level to receive service with this base address to form the interrupt vector that is driven out onto the bus. For example, the complete interrupt vector for IRQ[0] (CNTRL-1), would be 0000 1000b (CNTRL-1 [7:3] = 00001b and 000b representing IRQ[0]). This vector is used by the CPU to point to the address information that defines the start of the interrupt routine.

Bits[2:0]: Interrupt Request Level
When writing ICW2, these bits should all be 0. During an interrupt acknowledge cycle, these bits are programmed by the interrupt controller with the interrupt code representing the interrupt level to be serviced. This interrupt code is combined with bits[7:3] to form the complete interrupt vector driven onto the data bus during the second INTA# cycle. Section 5.0, Detailed Function Description, outlines each of these codes. The code is a simple three bit binary code: 000 represents IRQ0 (IRQ8), 001 IRQ1 (IRQ9), 010 IRQ2 (IRQ10), and so on until 111 IRQ7 (IRQ15).

4.4.3 ICW3—INITIALIZATION COMMAND WORD 3 REGISTER

Register Name:	Initialization Command Word 3 (Controller 1-Master Unit)
Address Offset:	INT CNTRL-1–021h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

The meaning of ICW3 differs between CNTRL-1 and CNTRL-2. On CNTRL-1, the master controller, ICW3 indicates which CNTRL-1 IRQ line physically connects the INT output of CNTRL-2 to CNTRL-1. ICW3 must be programmed to 04h, indicating the cascade of the CNTRL-2 INT output to the IRQ[2] input of CNTRL-1.

An interrupt request on IRQ2 causes CNTRL-1 to enable CNTRL-2 to present the interrupt vector address during the second interrupt acknowledge cycle.

Bits[7:3]:
These bits must be programmed to zero.

Bit 2: Cascaded Interrupt Controller IRQ Connection
Bit 2 must always be programmed to a 1. This bit indicates that CNTRL-2, the slave controller, is cascaded on interrupt request line two (IRQ[2]). When an interrupt request is asserted to CNTRL-2, the IRQ goes through the priority resolver. After the slave

controller priority resolution is finished, the INT output of CNTRL-2 is asserted. However, this INT assertion does not go directly to the CPU. Instead, the INT assertion cascades into IRQ[2] on CNTRL-1. IRQ[2] must go through the priority resolution process on CNTRL-1. If it wins the priority resolution on CNTRL-1 and the CNTRL-1 INT signal is asserted to the CPU, the returning interrupt acknowledge cycle is really destined for CNTRL-2. The interrupt was originally requested at CNTRL-2, so the interrupt acknowledge is destined for CNTRL-2, and not a response for IRQ[2] on CNTRL-1.

When an interrupt request from IRQ[2] wins the priority arbitration, in reality an interrupt from CNTRL-2 has won the arbitration. Because bit 2 of ICW3 on the master is set to 1, the master knows which identification code to broadcast on the internal cascade lines, alerting the slave controller that it is responsible for driving the interrupt vector during the second INTA# pulse.

Bits[1:0]:
These bits must be programmed to zero.

4.4.4 ICW3—INITIALIZATION COMMAND WORD 3 REGISTER

Register Name:	Initialization Command Word 3 (Controller 2-Slave Unit)
Address Offset:	INT CNTRL-2–0A1h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

On CNTRL-2 (the slave controller), ICW3 is the slave identification code broadcast by CNTRL-1 from the trailing edge of the first INTA# pulse to the trailing edge of the second INTA# pulse. CNTRL-2 compares the value programmed in ICW3 with the incoming identification code. The code is broadcast over three SIO internal cascade lines. ICW3 must be programmed to 02h for CNTRL-2. When 010b is broadcast by CNTRL-1 during the INTA# sequence, CNTRL-2 assumes responsibility for broadcasting the interrupt vector during the second interrupt acknowledge cycle.

As an illustration, consider an interrupt request on IRQ[2] of CNTRL-1. By definition, a request on IRQ[2] must have been asserted by CNTRL-2. If IRQ[2] wins the priority resolution on CNTRL-1, the interrupt acknowledge cycle returned by the CPU following the interrupt is destined for CNTRL-2, not CNTRL-1. CNTRL-1 will see the INTA# signal, and knowing that the actual destination is CNTRL-2, will broadcast a slave identification code across the internal cascade lines. CNTRL-2 will compare this incoming value with the 010b stored in ICW3. Following a positive decode of the incoming message from CNTRL-1, CNTRL-2 will drive the appropriate interrupt vector onto the data bus during the second interrupt acknowledge cycle.

Bits[7:3]: Reserved
Must be 0.

Bits[2:0]: Slave Identification Code
The Slave Identification code must be programmed to 010b during the initialization sequence. The code stored in ICW3 is compared to the incoming slave identification code broadcast by the master controller during interrupt acknowledge cycles.

4.4.5 ICW4—INITIALIZATION COMMAND WORD 4 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	01h
Attribute:	Write Only
Size:	8 bits

Both SIO interrupt controllers must have ICW4 programmed as part of their initialization sequence. Minimally, the microprocessor mode bit, bit 0, must be set to a 1 to indicate to the controller that it is operating in an 80x86 based system. Failure to program this bit will result in improper controller operation during interrupt acknowledge cycles. Additionally, the Automatic End of Interrupt (AEOI) may be selected, as well as the Special Fully Nested Mode (SFNM) of operation.

The default programming for ICW4 is 01h, which selects 80x86 mode, normal EOI, buffered mode, and special fully nested mode disabled.

Bits 2 and 3 must be programmed to 0 for the SIO interrupt controller to function correctly.

Both bit 1, AEOI, and bit 4, SFNM, can be programmed if the system developer chooses to invoke either mode.

Bits[7:5]: Reserved
Must be 0.

Bit 4: SFNM (Special Fully Nested Mode)
Bit 4, SFNM, should normally be disabled by writing a 0 to this bit. If SFNM = 1, the special fully nested mode is programmed.

Bit 3: BUF (Buffered Mode)
Bit 3, BUF, must be programmed to 0 for the SIO. This is non-buffered mode. As illustrated above under bit functionality, different programming options are offered for bits 2 and 3. However, within the SIO interrupt unit, bits 2 and 3 must always be programmed to 00b.

Bit 2: Master/Slave in Buffered Mode
This bit is not used by the SIO interrupt unit. Bit 2 should always be programmed to 0.

Bit 1: AEOI (Automatic End of Interrupt)
This bit should normally be programmed to 0. This is the normal end of interrupt. If this bit is 1, the automatic end of interrupt mode is programmed.

Bit 0: Microprocessor Mode
The Microprocessor Mode bit must be programmed to 1 to indicate that the interrupt controller is operating in an 80x86-based system. Never program this bit to 0.

4.4.6 OCW1—OPERATIONAL CONTROL WORD 1 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	00h
Attribute:	Read/Write
Size:	8 bits

OCW1 sets and clears the mask bits in the Interrupt Mask Register (IMR). Each interrupt request line may be selectively masked or unmasked any time after initialization. A single byte is written to this register. Each bit position in the byte represents the same-numbered channel: bit 0 = IRQ[0], bit 1 = IRQ[1] and so on. Setting the bit to a 1 sets the mask, and clearing the bit to a 0 clears the mask. Note that masking IRQ[2] on CNTRL-1 will also mask all of controller 2's interrupt requests (IRQ8–IRQ15). Reading OCW1 returns the controller's mask register status.

The IMR stores the bits which mask the interrupt lines to be masked. The IMR operates on the IRR. Masking of a higher priority input will not affect the interrupt request lines of lower priority.

Unlike status reads of the ISR and IRR, for reading the IMR, no OCW3 is needed. The output data bus will contain the IMR whenever I/O read is active and the I/O port address is 021h or 0A1h (OCW1).

All writes to OCW1 must occur following the ICW1–ICW4 initialization sequence, since the same I/O ports are used for OCW1, ICW2, ICW3 and ICW4.

Bits[7:0]: Interrupt Request Mask (Mask [7:0])
When a 1 is written to any bit in this register, the corresponding IRQ[x] line is masked. For example, if bit 4 is set to a 1, then IRQ[4] will be masked. Interrupt requests on IRQ[4] will not set channel 4's interrupt request register (IRR) bit as long is the channel is masked.

When a 0 is written to any bit in this register, the corresponding IRQ[x] mask bit is cleared, and interrupt requests will again be accepted by the controller.

NOTE:
Masking IRQ[2] on CNTRL-1 will also mask the interrupt requests from CNTRL-2, which is physically cascaded to IRQ[2].

4.4.7 OCW2—OPERATIONAL CONTROL WORD 2 REGISTER

Address Offset: INT CNTRL-1–020h
INT CNTRL-2–0A0h
Default Value: Bit[4:0] = undefined,
Bit[7:5] = 001
Attribute: Write Only
Size: 8 bits

OCW2 controls both the Rotate Mode and the End of Interrupt Mode, and combinations of the two. The three high order bits in an OCW2 write represent the encoded command. The three low order bits are used to select individual interrupt channels during three of the seven commands. The three low order bits (labeled L2, L1 and L0) are used when bit 6 is set to a 1 during the command.

Following a PCIRST# and ICW initialization, the controller enters the fully nested mode of operation. Non-specific EOI without rotation is the default. Both rotation mode and specific EOI mode are disabled following initialization.

Bits[7:5]: Rotate and EOI Codes
R, SL, EOI-These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations is listed above under the bit definition.

Bits 7	6	5	Function
0	0	1	Non-Specific EOI Command
0	1	1	Specific EOI Command
1	0	1	Rotate on Non-Specific EOI Command
1	0	0	Rotate in Auto EOI Mode (Set)
0	0	0	Rotate in Auto EOI Mode (Clear)
1	1	1	*Rotate on Specific EOI Command
1	1	0	*Set Priority Command
0	1	0	No Operation

NOTE:
* L0–L2 Are Used

Bits[4:3]: OCW2 Select
When selecting OCW2, bits 3 and 4 must both be 0. If bit 4 is a 1, the interrupt controller interprets the write to this port as an ICW1. Therefore, always ensure that these bits are both 0 when writing an OCW2.

Bits[2:0]: Interrupt Level Select (L2, L1, L0)
L2, L1, and L0 determine the interrupt level acted upon when the SL bit is active. A simple binary code, outlined above, selects the channel for the command to act upon. When the SL bit is inactive, these bits do not have a defined function; programming L2, L1 and L0 to 0 is sufficient in this case.

Bit 2	1	0	Interrupt Level
0	0	0	IRQ 0(8)
0	0	1	IRQ 1(9)
0	1	0	IRQ 2(10)
0	1	1	IRQ 3(11)
1	0	0	IRQ 4(12)
1	0	1	IRQ 5(13)
1	1	0	IRQ 6(14)
1	1	1	IRQ 7(15)

4.4.8 OCW3—OPERATIONAL CONTROL WORD 3 REGISTER

Address Offset: INT CNTRL-1-020h
INT CNTRL-2–0A0h
Default Value: Bit[6,0] = 0,
Bit[7,4:2] = undefined,
Bit[5,1] = 1
Attribute: Read/Write
Size: 8 bits

OCW3 serves three important functions: Enable Special Mask Mode, Poll Mode control, and IRR/ISR register read control.

First, OCW3 is used to set or reset the Special Mask Mode (SMM). The Special Mask Mode can be used by an interrupt service routine to dynamically alter the system priority structure while the routine is executing, through selective enabling/disabling of the other channel's mask bits.

Second, the Poll Mode is enabled when a write to OCW3 is issued with bit 2 equal to 1. The next I/O read to the interrupt controller is treated like an interrupt acknowledge; a binary code representing the highest priority level interrupt request is released onto the bus.

Third, OCW3 provides control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). Either the ISR or IRR is selected for reading with a write to OCW3. Bits 0 and 1 carry the encoded command to select either register. The next I/O read to the OCW3 port address will return the register status specified during the previous write. The register specified for a status read is retained by the interrupt controller. Therefore, a write to OCW3 prior to every status read command is unnecessary, provided the status read desired is from the register selected with the last OCW3 write.

Bit 7: Reserved
Must be 0.

Bit 6: SMM (Special Mask Mode)
If ESMM = 1 and SMM = 1 the interrupt controller enters Special Mask Mode. If ESMM = 1 and SMM = 0, the interrupt controller is in normal mask mode. When ESMM = 0, SMM has no effect.

Bit 5: ESMM (Enable Special Mask Mode)
When ESMM = 1, the SMM bit is enabled to set or reset the Special Mask Mode. When ESMM = 0, the SMM bit becomes a "don't care".

Bits[4:3]: OCW3 Select
When selecting OCW3, bit 3 must be a 1 and bit 4 must be 0. If bit 4 = 1, the interrupt controller interprets the write to this port as an ICW1. Therefore, always ensure that bits[4:3] = 01 when writing an OCW3.

Bit 2: Poll Mode Command
When bit 2 = 0, the Poll command is not issued. When bit 2 = 1, the next I/O read to the interrupt controller is treated as an interrupt acknowledge cycle. An encoded byte is driven onto the data bus, representing the highest priority level requesting service.

Bits[1:0]: Register Read Command
Bits[1:0] provide control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). When bit 1 = 0, bit 0 will not affect the register read selection. When bit 1 = 1, bit 0 selects the register status returned following an OCW3 read. If bit 0 = 0, the IRR will be read. If bit 0 = 1, the ISR will be read. Following ICW initialization, the default OCW3 port address read will be "read IRR". To retain the current selection (read ISR or read IRR), always write a 0 to bit 1 when programming this register. The selected register can be read repeatedly without reprogramming OCW3. To select a new status register, OCW3 must be reprogrammed prior to attempting the read.

Bit	1	0	Function
	0	0	No Action
	0	1	No Action
	1	0	Read IRQ Register
	1	1	Read IS Register

4.5 Control Registers

This section contains NMI registers, a real-time clock register, Port 92 Register, and the Digital Output Register.

4.5.1 NMISC—NMI STATUS AND CONTROL REGISTER

Address Offset: 061h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register is used to check the status of different system components, control the output of the speaker counter (Counter 2), and gate the counter output that drives the SPKR signal.

Bits 4, 5, 6, and 7 are read-only. When writing to this port, these bits must be written as 0's. Bit 6 returns the IOCHK# NMI status. This input signal comes from the ISA Bus. It is used for parity errors on memory cards plugged into the bus, and for other high priority interrupts. The current status of bit 3 enables or disables this NMI source. Bit 5 is the current state of the OUT pin of interval Timer 1, Counter 2. Bit 4 toggles from 1-0 or from 0-1 after every Refresh cycle. Following PCIRST#, bits 4 and 6 are both 0. Bit 5 is undetermined until Counter 2 is properly programmed. Bit 7 returns the PCI System Error status (SERR#). If 0, bit 7 indicates that SERR# was not pulsed active by a PCI agent. If 1, bit 7 indicates that SERR# was pulsed active by a PCI agent and that an NMI will be issued to the Host CPU. This NMI can be disabled with bit 2 of this register.

Bits 0–3 are both read and write. Bit 0 is the GATE input signal for Timer 1, Counter 2. The GATE input is used to disable counting in Counter 2. The Counter 2 output is ANDed with bit 1 to form the SPKR output signal. Bit 1 gates the Counter 2 OUT value. When bit 1 is disabled, the SPKR signal is disabled; when bit 1 is enabled, the SPKR output follows the value at the OUT pin of Counter 2. The Counter 2 OUT pin status can be checked by reading port 061h and checking bit 5. Bit 2 is used to enable the System Error (SERR#) signal. Bit 3 enables or disables the incoming IOCHK# NMI signal from the expansion bus. Each of these bits is reset to 0 following PCIRST#.

Bit 7: SERR# Status
Bit 7 is set if a system board agent (PCI devices or main memory) detects a system board error and pulses the PCI SERR# line. This interrupt is enabled by setting bit 2 to 0. To reset the interrupt, set bit 2 to 0 and then set it to 1. This bit is read-only. When writing to port 061h, bit 6 must be a 0.

Bit 6: IOCHK# NMI Source Status
Bit 6 is set if an expansion board asserts IOCHK# on the ISA/SIO bus. This interrupt is enabled by setting bit 3 to 0. To reset the interrupt, set bit 3 to 0 and then set it to 1. This bit is read-only. When writing to port 061h, bit 6 must be a 0.

Bit 5: Timer Counter 2 OUT Status
The Counter 2 OUT signal state is reflected in bit 5. The value on this bit following a read is the current state of the Counter 2 OUT signal. Counter 2 must be programmed following a PCIRST# for this bit to have a determinate value. Bit 5 is read-only. When writing to port 061h, bit 5 must be a 0.

Bit 4: Refresh Cycle Toggle
The Refresh Cycle Toggle signal toggles from either 0 to 1 or 1 to 0 following every refresh cycle. This read-only bit is a 0 following PCIRST#. When writing to port 061h, bit 4 must be a 0.

Bit 3: IOCHK# NMI Enable
When bit 3 = 1, IOCHK# NMI's are disabled and cleared. When bit 3 = 0, IOCHK# NMI's are enabled. Following PCIRST#, bit 3 is reset to 0.

Bit 2: PCI SERR# Enable
When bit 2 = 1, the PCI System Error (SERR#) is disabled and cleared. When bit 2 = 0, SERR# is enabled. Following PCIRST#, bit 2 is a 0.

Bit 1: Speaker Data Enable
Speaker Data Enable is ANDed with the Counter 2 OUT signal to drive the SPKR output signal. When bit 1 = 0, the result of the AND is always 0 and the SPKR output is always 0. When bit 1 = 1, the SPKR output is equivalent to the Counter 2 OUT signal value. Following PCIRST#, bit 1 is a 0.

Bit 0: Timer Counter 2 Enable
When bit 0 = 0, Counter 2 counting is disabled. Counting is enabled when bit 0 = 1. This bit controls the GATE input to Counter 2. Following PCIRST#, the value of this bit is 0.

4.5.2 NMI ENABLE AND REAL-TIME CLOCK ADDRESS REGISTER

Address Offset: 070h
Default Value: Bit[6:0] = undefined, Bit 7 = 1
Attribute: Write Only
Size: 8 bits

The Mask register for the NMI interrupt is at I/O address 070h shown below. The most significant bit enables or disables all NMI sources including IOCHK# and the NMI Port. Write an 80h to port 70h to mask the NMI signal. This port is shared with the real-time clock. The real-time-clock uses the lower six bits of this port to address memory locations. Writing to port 70h sets both the enable/disable bit and the memory address pointer. Do not modify the contents of this register without considering the effects on the state of the other bits. Reads and writes to this register address flow through to the ISA Bus.

Bit 7: NMI Enable
Setting bit 7 to a 1 disables all NMI sources. Setting the bit to a 0 enables the NMI interrupt. Following PCIRST#, this bit is a 1.

Bits[6:0]: Real Time Clock Address
Used by the Real Time Clock on the Base I/O component to address memory locations. Not used for NMI enabling/disabling.

4.5.3 PORT 92 REGISTER

Address Offset: 92h
Default Value: 24h
Attribute: Read/Write
Size: 8 bits

This register is used to support the alternate reset (ALT__RST#) and alternate A20 (ALT__A20) functions. This register is only accessible if bit 6 in the Utility Bus Chip Select B Register is set to a 1. Reads and writes to this register location flow through to the ISA Bus.

Bits[7:6]: Reserved
Returns 00 when read.

Bit 5: Reserved
Returns a 1 when read.

Bit 4: Reserved
Returns a 0 when read.

Bit 3: Reserved
Returns a 0 when read.

Bit 2: Reserved
Returns a 1 when read.

Bit 1: ALT__A20 Signal Control
Writing a 0 to this bit causes the ALT__A20 signal to be driven low. Writing a 1 to this bit causes the ALT__A20 signal to be driven high.

Bit 0: Alternate System Reset
This read/write bit provides an alternate system reset function. This function provides an alternate means to reset the system CPU to effect a mode switch from Protected Virtual Address Mode to the Real Address Mode. This provides a faster means of reset than is provided by the Keyboard controller. This bit is set to a 0 by a system reset. Writing a 1 to this bit will cause the ALT__RST# signal to pulse active (low) for approximately 4 SYSCLK's. Before another ALT__RST# pulse can be generated, this bit must be written back to a 0.

4.5.4 DIGITAL OUTPUT REGISTER

Address Offset: 03F2h (Primary), 0372h (Secondary)
Default Value: Bit[7:4,2:0] = undefined, Bit 3 = 0
Attribute: Write only
Size: 8 bits

This register is used to prevent UBUSOE# from responding to DACK2# during a DMA read access to a floppy controller on the ISA Bus. If a second floppy (residing on the ISA Bus) is using DACK2# in conjunction with a floppy on the utility bus, this prevents the floppy on the utility bus and the utility bus transceiver from responding to an access targeted for the floppy on the ISA Bus. This register is also located in the floppy controller device. Reads and writes to this register location flow through to the ISA Bus.

Bits[7:4]: Not Used
These bits exist in the floppy controller.

Bit 3: DMA Enable
When this bit is a 1, the assertion of DACK# will result in UBUSOE# being asserted. If this bit is 0, DACK2# has no effect on UBUSOE#. This port bit also exists on the floppy controller. This bit defaults to disable (0).

Bits[2:0]: Not Used
These bits exist in the floppy controller.

4.5.5 RESET UBUS IRQ12 REGISTER

Address Offset: 60h
Default Value: N/A
Attribute: Read only
Size: 8 bits

This address location (60h) is used to clear the mouse interrupt function to the CPU. Reads to this address are monitored by the SIO. When the mouse interrupt function is enabled (bit 4 of the ISA Clock Divisor Register is 1), the mouse interrupt function is provided on the IRQ12/M input signal. In this mode, a mouse interrupt generates an interrupt through IRQ13 to the Host CPU. A read of 60h releases IRQ12. If bit 4 = 0 in the ISA Clock Divisor Register, a read of address 60h has no effect on IRQ12/M. Reads and writes to this register flow through to the ISA Bus. For additional information, see the IRQ12/M description in Section 3.0, Signal Description.

Bits[7:0]: Reset IRQ12
No specific pattern. A read of address 60h executes the command.

4.5.6 COPROCESSOR ERROR REGISTER

Address Offset: F0h
Default Value: N/A
Attribute: Write only
Size: 8 bits

This address location (F0h) is used when the SIO is programmed for coprocessor error reporting (bit 5 of the ISA Clock Divisor Register is 1). Writes to this address are monitored by the SIO. In this mode, the SIO generates an interrupt (INT) to the CPU when it receives an error signal (FERR# asserted) from the CPU's coprocessor. Writing address F0h, when FERR# is asserted, causes the SIO to assert IGNNE# and negate IRQ13. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted, writing to address F0h does not effect IGNNE#. Reads and writes to this register flow through to the ISA Bus. For additional information, see the IGNNE# description in Section 3.0, Signal Description.

Bits[7:0]: Reset IRQ12
No specific pattern. A write to address F0h executes the command.

4.5.7 ELCR—EDGE/LEVEL CONTROL REGISTER

Address Offset: INT CNTRL-1–04D0h
INT CNTRL-2–04D1h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The Edge/Level Control Register is used to set the interrupts to be triggered by either the signal edge or the logic level. INT0, INT1, INT2, INT8, INT13 must be set to edge sensitive. After a reset, all the INT signals are set to edge sensitive. Figure 4 shows which bit numbers represent the various INT signals.

Each IRQ to which a PCI interrupt is steered into (see the PIRQ Route Control Register) must have its interrupt set to level sensitive.

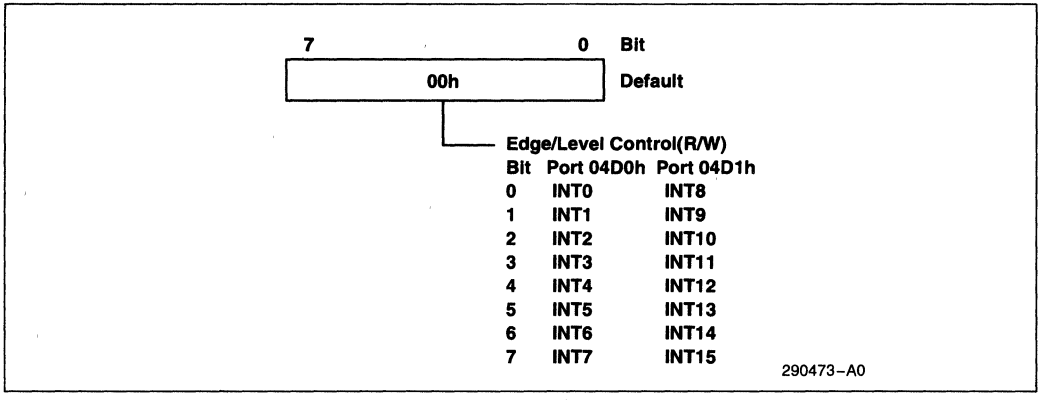


Figure 4. Edge/Level Select Register

Register Location: 04D0h-INT CNTRL-1
04D1h-INT CNTRL-2

04D0h-INT CNTRL-1 Register

Bit[7:0]: Edge/Level Select
These bits select if the interrupts are triggered by either the signal edge or the logic level. A 0 bit represents an edge sensitive interrupt, and a 1 is for level sensitive. The following bits MUST be set to 0:

Port 04D0h (INT-CNTRL-1)

0–INT0	0	Reserved. Read as zero.
1–INT1	0	Reserved. Read as zero.
2–INT2	0	Reserved. Read as zero.
3–INT3	x	
4–INT4	x	
5–INT5	x	
6–INT6	x	
7–INT7	x	

x = selectable to either a 0 or a 1,
0 = edge sensitive, 1 = level sensitive

After reset, this register is set to 00h.

04D1h-INT CNTRL-2 Register

Bit[7:0]: Edge/Level Select
These bits select if the interrupts are triggered by either the signal edge or the logic level. A 0 bit represents an edge sensitive interrupt, and a 1 is for level sensitive. The following bits MUST be set to 0:

Port 04D1h (INT-CNTRL-2)

0–INT8	0	Reserved. Read as zero.
1–INT9	x	
2–INT10	x	
3–INT11	x	
4–INT12	x	
5–INT13	0	Reserved. Read as zero.
6–INT14	x	
7–INT15	x	

x = selectable to either a 0 or a 1,
0 = edge sensitive, 1 = level sensitive

After reset, this register is set to 00h.

4.6. Power Management Registers

This section contains the Power Management Registers located in non-configuration space.

4.6.1 APMC—ADVANCED POWER MANAGEMENT CONTROL PORT

Address Offset: 0B2h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

Bits[7:0]: APMC[7:0]
APM Control Port. Readable/writeable at system I/O address 0B2h. Used to pass an APM command between the OS and the SMI handler. Writes to this port not only store data in the APMC register, but also generate an SMI when the SAPMCEN bit is set. Reads to this port will not generate an SMI. If CSTPCLKEN is set, a read from the APMC will cause STPCLK# to be asserted.

4.6.2 APMS—ADVANCED POWER MANAGEMENT STATUS PORT

Address Offset: 0B3h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

Bits[7:0]: APMS[7:0]
Readable/writeable at system address 0B3h. Used to pass data between the OS and the SMI handler.

5.0 DETAILED FUNCTIONAL DESCRIPTION

5.1 PCI Interface

5.1.1 PCI COMMAND SET

Bus commands indicate to the slave the type of transaction the master is requesting. Bus Commands are encoded on the C/BE[3:0]# lines during the address phase of a PCI cycle.

5.1.2 PCI BUS TRANSFER BASICS

Details of PCI Bus operations can be found in the *Peripheral Component Interconnect (PCI) Specification*. Only details of the PCI Bus unique to the SIO are included in this data sheet.

Table 7. PCI Commands

C/BE[3:0]#	Command Type As Slave	Supported As Slave	Supported As Master
0000	Interrupt Acknowledge	Yes	No
0001	Special Cycle(4)	No/Yes	No
0010	I/O Read	Yes	No
0011	I/O Write	Yes	No
0100	Reserved(3)	No	No
0101	Reserved(3)	No	No
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved(3)	No	No
1001	Reserved(3)	No	No
1010	Configuration Read	Yes	No
1011	Configuration Write	Yes	No
1100	Memory Read Multiple	No(2)	No
1101	Reserved(3)	No	No
1110	Memory Read Line	No(2)	No
1111	Memory Write and Invalidate	No(1)	No

NOTES:
1. Treated as Memory Write.
2. Treated as Memory Read.
3. Reserved Cycles are considered invalid by the SIO and are to be competely ignored. All internal address decoding is ignored and DEVSEL# is never to be asserted.
4. The 82378 responds to a Stop Grant Special Cycle.

5.1.2.1 PCI Addressing

PCI address decoding uses the AD[31:0] signals. AD[31:2] are always used for address decoding while the information contained in the two low order bits AD[1:0] varies for memory, I/O, and configuration cycles.

For I/O cycles, AD[31:0] are decoded to provide a byte address. AD[1:0] are used for generation of DEVSEL# only and indicate the least significant valid byte involved in the transfer. For example, if only BE0# is asserted, AD[1:0] are 00. If only BE3# is asserted, then AD[1:0] are 11. If BE3# and BE2# are asserted, AD[1:0] are 10. If all BEx#'s are asserted, then AD[1:0] are 00. The byte enables determine which byte lanes contain valid data. The SIO requires that PCI accesses to byte-wide internal registers must assert only one byte enable.

When the SIO is the target of any PCI transaction in which BE[3:0]# = 1111, the SIO terminates the cycle normally by asserting TRDY#. No data is written into the SIO during write cycles and the data driven by the SIO during read cycles is indeterminate.

For memory cycles, accesses are decoded as Dword accesses. This means that AD[1:0] are ignored for decoding memory cycles. The byte enables determine which byte lanes contain valid data. When the SIO is a PCI master, it drives 00 on AD[1:0] for all memory cycles.

For configuration cycles, DEVSEL# is a function of IDSEL and AD[1:0]. DEVSEL# is selected during a configuration cycle only if IDSEL is active and both AD[1:0] = 00. The cycle is ignored by the SIO if either AD1 or AD0 is non-zero. Configuration registers are selected as Dwords using AD[7:2]. The byte enables determine which byte lanes contain valid data.

5.1.2.2 DEVSEL# Generation

As a PCI slave, the SIO asserts the DEVSEL# signal to indicate it is the slave of the PCI transaction. DEVSEL# is asserted when the SIO positively or subtractively decodes the PCI transaction. The SIO asserts DEVSEL# (claim the transaction) before it issues any other slave response, i.e., TRDY#, STOP#, etc. After the SIO asserts DEVSEL#, it does not negate DEVSEL# until the same edge that the master uses to negate the final IRDY#.

It is expected that most (perhaps all) PCI target devices will be able to complete a decode and assert DEVSEL# within 1 or 2 clocks of FRAME#. Since the SIO subtractively decodes all unclaimed PCI cycles (except configuration cycles), it provides a configuration option to pull in (by 1 or 2 clocks) the edge when the SIO samples DEVSEL#. This allows faster access to the expansion bus. Use of such an option is limited by the slowest positive decode agent on the bus. This is described in more detail in Section 5.5.1.4, Subtractively Decoded Cycles to ISA.

As a PCI master, the SIO waits for 5 PCICLKs after the assertion of FRAME# for a slave to assert DEVSEL#. If the SIO does not receive DEVSEL# in this time, it will master-abort the cycle. See Section 5.1.3.1, SIO as MasterMaster-Initiated Termination, for further details.

5.1.2.3 Basic PCI Read Cycles (I/O and Memory)

As a PCI master, the SIO only performs memory read transfers (i.e. I/O read transfers are not supported). When reading data from PCI memory, the SIO requests a maximum of 8 bytes via a two data phase burst read cycle to fill its internal 8 byte line buffer. If the line buffer is programmed for single transaction mode, fewer bytes are requested (refer to Section 5.6.1, DMA/ISA Master Line Buffer). Read cycles from PCI memory are generated on behalf of ISA masters and DMA devices.

As a PCI slave, the SIO responds to both I/O read and memory read transfers. For multiple read transactions, the SIO always target-terminates after the first data read transaction by asserting STOP# and TRDY# at the end of the first data phase. For single read transactions, the SIO finishes the cycle in a normal fashion, by asserting TRDY# without asserting STOP#.

5.1.2.4 Basic PCI Write Cycles (I/O and Memory)

As a PCI master, the SIO generates a PCI memory write cycle when it decodes an ISA-originated/PCI-bound memory write cycle. I/O write cycles are never initiated by the SIO. When writing data to PCI memory, the SIO writes a maximum of 4 bytes via a single data transaction write cycle. If the SIO's internal ISA master/DMA line buffer is programmed for single transaction mode, fewer bytes will be generated (refer to Section 5.6.1, DMA/ISA Master Line Buffer). In either case, only one data transaction will be performed. Cycles to PCI memory are generated on behalf of ISA masters, DMA devices, and the SIO when the SIO needs to flush the ISA master/DMA line buffer.

As a PCI master, the SIO drives the AD0 and AD1 signals low during the address phase of the cycle. This is done to indicate to the slave that the address will increment during the transfer. If there is no response on the PCI Bus, the SIO will master-abort due to the DEVSEL# time out.

As a PCI slave, the SIO will respond to both I/O write and memory write transfers. For multiple write transactions, the SIO will always target-terminate after the first data write transaction by asserting STOP# and TRDY# at the end of the first data phase. For single write transactions, the SIO will finish the cycle normally by asserting TRDY# without asserting STOP#.

5.1.2.5 Configuration Cycles

The configuration read or write command defined by the bus control signals C/BE[3:0]# is used to configure the SIO. During the address phase of the configuration read or write command, the SIO will sample its IDSEL (ID select). If IDSEL is active and AD[1:0] are both zero, the SIO generates DEVSEL#. Otherwise, the cycle is ignored by the SIO. During the configuration cycle address phase, bits AD[7:2] and C/BE[3:0]# are used to select particular bytes within a configuration register. Note that IDSEL is normally a "don't care" except during the address phase of a transaction.

NOTE:
An unclaimed configuration cycle is never forwarded to the ISA Bus.

5.1.2.6 Interrupt Acknowledge Cycle

The interrupt acknowledge command is a single byte read implicitly addressed to the SIO's interrupt controller. The address bits are logical "don't cares" during the address phase and the byte enables will indicate to the SIO an 8-bit interrupt vector is to be returned on AD[7:0]. The SIO converts this single cycle transfer into two cycles that the internal 8259 pair can respond to (see Section 5.8, Interrupt Controller). The SIO will hold the PCI Bus in wait states until the 8 bit interrupt vector is returned.

SIO responses to an interrupt acknowledge cycle can be disabled by setting bit 5 in the PCI Control Register to a 0. However, if disabled, the SIO will still respond to accesses to the interrupt register set and allow poll mode functions.

5.1.2.7 Exclusive Access

The SIO marks itself locked anytime it is the slave of the access and LOCK# is sampled negated during the address phase. As a locked slave, the SIO responds to a master only when it samples LOCK# negated and FRAME# asserted. The locking master may negate LOCK# at the end of the last data phase. The SIO unlocks itself when FRAME# and LOCK# are both negated. The SIO will respond by asserting STOP# with TRDY# negated (retry) to all transactions when LOCK# is asserted during the address phase.

Locked cycles are never generated by the SIO.

5.1.2.8 PCI Special Cycle

When the SCE bit (bit 3) in the COM PCI configuration register (configuration offset 04h) is set to a "0", the SIO will ignore all PCI Special Cycles. When the SCE bit is set to a "1", the SIO will recognize PCI Special Cycles.

The only PCI Special Cycle currently recognized is the Stop Grant Special Cycle which is broadcast onto the PCI bus when an S-series processor enters the Stop Grant State. The SCE bit must be set to a "1" when the Stop Clock feature is being used.

5.1.3 TRANSACTION TERMINATION

The SIO supports both Master-initiated Termination as well as Target-initiated Termination.

5.1.3.1 SIO As Master—Master-Initiated Termination

The SIO supports two forms of master-initiated termination:

1. Normal termination of a completed transaction.
2. Abnormal termination due to no slave responding to the transaction (Abort).

Figure 5 shows the SIO performing master-abort termination. This occurs when no slave responds to the SIO's master transaction by asserting DEVSEL# within 5 PCICLK's after FRAME# assertion. This master-abort condition is abnormal and it indicates an error condition. The SIO will not retry the cycle. The Received Master-abort Status bit in the PCI Status Register will be set indicating that the SIO experienced a master-abort condition.

If an ISA master or the DMA is waiting for the PCI cycle to terminate (CHRDY negated), the master-abort condition will cause the SIO to assert CHRDY to terminate the ISA cycle. Note that write data will be lost and the read data will be all 1's at the end of the cycle. This is identical to the way an unclaimed cycle is handled on the "normally ready" ISA Bus. If the line buffer is the requester of the PCI transaction, the master-abort mechanism will end the PCI cycle, but no data will be transferred into or out of the line buffer. The line buffer will not be allowed to retry the cycle.

5.1.3.2 SIO As A Master—Response To Target-Initiated Termination

SIO's response as a master to target-termination:

1. For a target-abort, the SIO will not retry the cycle. If an ISA master or the DMA is waiting for the PCI cycle to complete (CHRDY negated), the target-abort condition will cause the SIO to assert CHRDY and end the cycle on the ISA Bus. If the ISA master or DMA device was reading from PCI memory, the SIO will drive all 1's on the data lines of the ISA Bus. The Received Target-abort Status bit in the PCI Status Register will be set indicating that the SIO experienced a target-abort condition.
2. If the SIO is retried as a master on the PCI Bus, it will remove it's request for 2 PCI clocks before asserting it again to retry the cycle.
3. If the SIO is disconnected as a master on the PCI Bus, it will respond very much as if it had been retried. The difference between retry and disconnect is that the SIO did not see any data phase for the retry. Disconnect may be generated by a PCI slave when the SIO is running a burst memory read cycle to fill it's 8-byte Line Buffer. In this case, the SIO may need to finish a multi-data phase transfer, and thus, must recycle through arbitration as required for a retry. An example of this is when the on-board DMA requests an 8-byte Line Buffer transfer and the SIO is disconnected before the Line Buffer is completely filled.

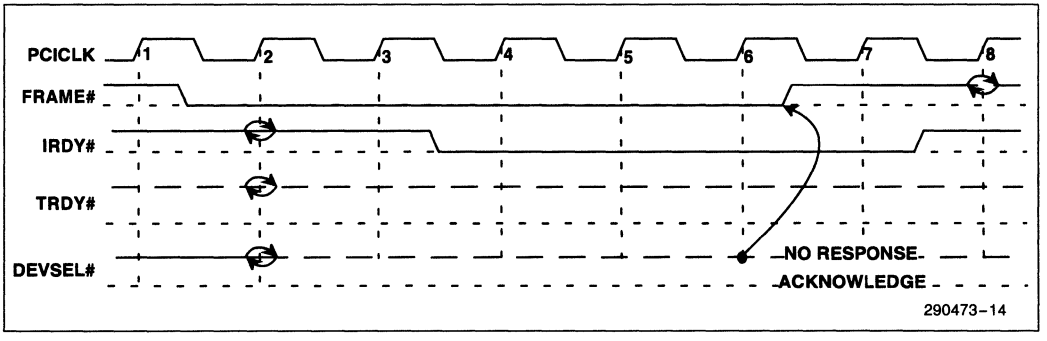


Figure 5. Master—Initiated Termination (Master-Abort)

5.1.3.3 SIO As A Target—Target-Initiated Termination

The SIO supports three forms of Target-initiated Termination:

Disconnect	Disconnect refers to termination requested because the SIO is unable to respond within the latency guidelines of the PCI specification. Note that this is not usually done on the first data phase.
Retry	Retry refers to termination requested because the target is currently in a state which makes it unable to process the transaction.
Abort	Abort refers to termination requested because the target will never be able to respond to the transaction.

The SIO will initiate Disconnect for PCI-originated/ISA-bound cycles after the first data phase due to incremental latency requirements. Since the SIO has only one Posted Write Buffer and every PCI to ISA incremental data phase will take longer than the specified 8 clocks, the SIO will always terminate burst cycles with a disconnect protocol. An example of this is when the SIO receives a burst memory write. Since the SIO only has one Posted Write Buffer, the transaction will automatically be disconnected after the first data phase.

The SIO will retry PCI masters:

1. For memory write cycles when the posted write buffer is full.
2. When the pending PCI cycle initiates some type of buffer management activity.
3. When the SIO is locked as a resource and a PCI master tries to access the SIO without negating the LOCK# signal in the address phase.
4. When the ISA Bus is occupied by an ISA master or DMA.

Target-abort is issued by the SIO when the internal SIO registers are the target of a PCI master I/O cycle and more than one byte enable is active. Accesses to the BIOS Timer Register and the Scatter/Gather Descriptor Table Pointer Registers are exceptions to this rule. Accesses to the Scatter/Gather Descriptor Table Pointer Register must be 32-bits wide and accesses to the BIOS Timer Register must be 16- or 32-bits wide. These accesses will not result in a SIO target-abort. The SIO responds with a target-abort since the registers must be accessed as 8-bit quantities. Target-abort resembles a retry, although the SIO also negates DEVSEL# along with the assertion of STOP#. Bit 11 in the Device Status Register is set to a 1 when the SIO target-aborts.

5.1.4 BUS LATENCY TIME-OUT

5.1.4.1 Master Latency Timer

Because the SIO only bursts a maximum of two Dwords, the PCI master latency timer is not implemented.

5.1.4.2 Target Incremental Latency Mechanism

As a slave, the SIO supports the Incremental Latency Mechanism for PCI to ISA cycles. The PCI specification states that for multi-data phase PCI cycles, if the incremental latency from current data phase (N) to the next data phase (N+1) is greater than 8 PCICLK's, then the slave must manipulate TRDY# and STOP# to stop the transaction upon completion of the current data phase (N). Since all PCI-originated (SIO is a slave)/ISA-bound cycles will require greater than the stated 8 PCICLK's, the SIO will automatically terminate these cycles after the first data phase. Note that latency to the first data phase is not restricted by this mechanism.

5.1.5 PARITY SUPPORT

As a master, the SIO generates address parity for read and write cycles, and data parity for write cycles. As a slave, the SIO generates data parity for read cycles. The SIO does not check parity and does not generate SERR#.

PAR is the calculated parity signal. PAR is "even" parity and is calculated on 36 bits; the 32 AD[31:0] signals plus the 4 C/BE[3:0]# signals. "Even" parity means that the number of 1's within the 36 bits plus PAR are counted and the sum is always even. PAR is always calculated on 36 bits, regardless of the valid byte enables. PAR is only guaranteed to be valid one PCI clock after the corresponding address or data phase.

5.1.6 RESET SUPPORT

The PCIRST# pin acts as the SIO hardware reset pin.

During Reset

AD[31:0], C/BE[3:0]#, and PAR are always driven low by the SIO from the leading edge of PCIRST#. FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/ RESUME# are tri-stated from the leading edge of PCIRST#.

GNT2# and GNT3# are tri-stated from the leading edge of PCIRST#.

After Reset

AD[31:0], C/BE[3:0]#, and PAR are always tri-stated from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO will drive these signals low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), these signals remain tri-stated until the SIO is required to drive them valid as a master or slave.

FRAME#, IRDY#, TRDY#, STOP#, and DEVSEL# remain tri-stated until driven by the SIO as either a master or a slave. MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated until driven by the SIO.

GNT2# and GNT3# are tri-stated until driven by the SIO.

After PCIRST, MEMREQ# and FLSHREQ# are driven inactive asynchronously from PCIRST# inactive. CPUGNT#, GNT0#/SIOREQ#, and GNT1#/ RESUME# are driven based on the arbitration scheme and the asserted REQx#'s.

GNT2# and GNT3# are also driven based on the arbitration scheme and the asserted REQx#'s.

5.1.7 DATA STEERING

Data steering logic internal to the SIO provides the assembly/disassembly, copy up/copy down mechanism for cycles between the 32-bit PCI data bus and the 16-bit ISA Bus. The steering logic ensures that the correct bytes are steered to the correct byte lane and that multiple cycles are run where applicable.

5.2 PCI Arbitration Controller

The 82378 contains a PCI Bus arbiter that supports six PCI masters; the Host Bridge, SIO, and four other masters. The SIO's REQ#/GNT# lines are internal. The integrated arbiter can be disabled by asserting CPUREQ# during PCIRST# (see Section 5.2.7, Power-up Configuration). When disabled, the SIO's REQ#, GNT#, and RESUME# signals become visible for an external arbiter. The internal arbiter is enabled upon power-up.

The internal arbiter contains several features that contribute to system efficiency:

- Use of a RESUME# signal to re-enable a backed-off initiator in order to minimize PCI Bus thrashing when the SIO generates a retry (Section 5.2.4.1).
- A programmable timer to re-enable retried initiators after a programmable number of PCICLK's (Section 5.2.4.2).
- The CPU (host bridge) can be optionally parked on the PCI Bus (Section 5.2.5).
- A programmable PCI Bus lock or PCI resource lock function (Section 5.2.6).

The PCI arbiter is also responsible for control of the Guaranteed Access Time (GAT) mode signals (Section 5.2.3.2).

5.2.1 ARBITRATION SIGNAL PROTOCOL

The internal arbiter follows the PCI arbitration method as outlined in the *Peripheral Component Interconnect (PCI) Specification*. The SIO's arbiter is discussed in this section.

5.2.1.1 Back-To-Back Transactions

The SIO as a master does not generate fast back-to-back accesses since it does not know if it is accessing the same target.

The SIO as a target supports fast back-to-back transactions. Note that for back-to-back cycles, the SIO treats positively decoded accesses and subtractively decoded accesses as different targets. Therefore, masters can only run fast back-to-back cycles to positively decoded addresses or to subtractively decoded addresses. Fast back-to-back cycles must not mix positive and subtractive decoded addresses. See the address decoding section to determine what addresses the SIO positively decodes and subtractively decodes.

5.2.2 PRIORITY SCHEME

The PCI arbitration priority scheme is programmable through the PCI Arbiter Priority Control and Arbiter Priority Control Extension Register. The arbiter consists of four banks that can be configured for the six masters to be arranged in a purely rotating priority scheme, one of twenty-four fixed priority schemes, or a hybrid combination (Figure 6).

Note that SIOREQ#/SIOGNT# are SIO internal signals.

The PCI Arbiter Priority Control (PAPC) and PCI Arbiter Priority Control Extension Register bits are shown below:

PCI Arbiter Priority Control Register Bits (PAPC)

Bit	Description
7	Bank 3 Rotate Control
6	Bank 2 Rotate Control
5	Bank 1 Rotate Control
4	Bank 0 Rotate Control
3	Bank 2 Fixed Priority Mode select B
2	Bank 2 Fixed Priority Mode select A
1	Bank 1 Fixed Priority Mode select
0	Bank 0 Fixed Priority Mode select

PCI Arbiter Priority Control Extension Register Bits (ARBPRIX)

Bit	Description
7:1	Reserved. Read as 0
0	Bank 3 Fixed Priority Mode select

PAPC defaults to 04h and ARBPRIX to 00h at reset selecting fixed mode #10 (Table 8) with the CPU the highest priority device guaranteeing access to BIOS.

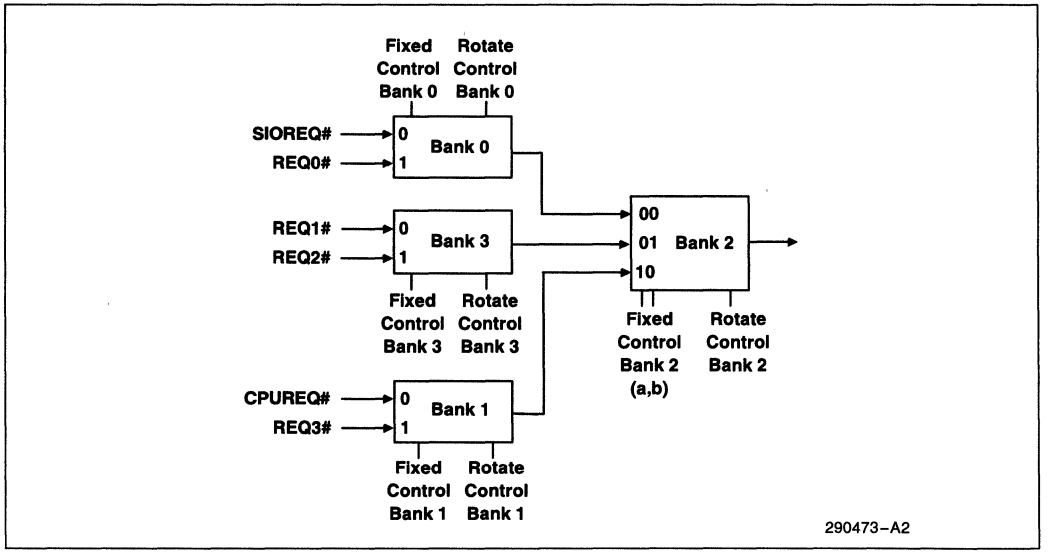


Figure 6. Arbiter Configuration Diagram for 82378ZB

5.2.2.1 Fixed Priority Mode

The 24 selectable fixed priority schemes are listed in Table 8.

Table 8. Fixed Priority Mode Bank Control Bits

Mode	Bank					Priority					
	3	2b	2a	1	0	Highest					Lowest
00	0	0	0	0	0	SIOREQ#	REQ0#	REQ2#	REQ3#	CPUREQ#	REQ1#
01	0	0	0	0	1	REQ0#	SIOREQ#	REQ2#	REQ3#	CPUREQ#	REQ#
02	0	0	0	1	0	SIOREQ#	REQ0#	REQ2#	REQ3#	REQ1#	CPUREQ#
03	0	0	0	1	1	REQ0#	SIOREQ#	REQ2#	REQ3#	REQ1#	CPUREQ#
04	0	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ2#	REQ3#
05	0	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ2#	REQ3#
06	0	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ2#	REQ3#
07	0	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ2#	REQ3#
08	0	1	0	0	0	REQ2#	REQ3#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
09	0	1	0	0	1	REQ2#	REQ3#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
0A	0	1	0	1	0	REQ2#	REQ3#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
0B	0	1	0	1	1	REQ2#	REQ3#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
0C–0F	0	1	1	x	x	Reserved					
10	1	0	0	0	0	SIOREQ#	REQ0#	REQ3#	REQ2#	CPUREQ#	REQ1#
11	1	0	0	0	1	REQ0#	SIOREQ#	REQ3#	REQ2#	CPUREQ#	REQ1#
12	1	0	0	1	0	SIOREQ#	REQ0#	REQ3#	REQ2#	REQ1#	CPUREQ#
13	1	0	0	1	1	REQ0#	SIOREQ#	REQ3#	REQ2#	REQ1#	CPUREQ#
14	1	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ3#	REQ2#
15	1	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ3#	REQ2#
16	1	0	1	1	0	REQ1#	SPUREQ#	SIOREQ#	REQ0#	REQ3#	REQ2#
17	1	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ3#	REQ2#
18	1	1	0	0	0	REQ3#	REQ2#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
19	1	1	0	0	1	REQ3#	REQ2#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
1A	1	1	0	1	0	REQ3#	REQ2#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
1B		1	0	1	1	REQ3#	REQ2#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
1C–1F	1	1	1	x	x	Reserved					

The fixed bank control bit(s) selects which requester is the highest priority device within that particular bank. For fixed priority mode, bits[7:4] of the PAPC Register and bit zero of ARBPRIX must be 0's (rotate mode disabled).

The selectable fixed priority schemes provide 24 of the 64 possible fixed mode permutations possible for the six masters.

5.2.2.2 Rotating Priority Mode

When any bank rotate control bit is set to a one, that particular bank rotates between the requesting inputs. Any or all banks can be set in rotate mode. If all four banks are set in rotate mode, the six supported masters are all rotated and the arbiter is in a pure rotating priority mode. If, within a rotating bank, the highest priority device (a) does not have an active request, the lower priority device (b or c) will be granted the bus. However, this does not change the rotation scheme. When the bank toggles, device b is the highest priority. Because of this, the maximum latency a device can encounter is two complete rotations.

5.2.2.3 Mixed Priority Mode

Any combination of fixed priority and rotate priority modes can be used in different arbitration banks to achieve a specific arbitration scheme.

5.2.2.4 Locking Masters

When a master acquires the LOCK# signal, the arbiter gives that master highest priority until the LOCK# signal is negated and FRAME# is negated. This ensures that a master that locked a resource will eventually be able to unlock that same resource.

5.2.3 MEMREQ#, FLSHREQ#, AND MEMACK# PROTOCOL

Before an ISA master or the DMA can be granted the PCI Bus, it is necessary that all PCI system posted write buffers be flushed (including the SIO's Posted Write Buffer). Also, since the ISA originated cycle could access memory on the host bridge, it's possible that the ISA master or the DMA could be held in wait states (via IOCHRDY) waiting for the host bridge arbitration for longer than the 2.5 μs ISA specification. The SIO has an optional mode called the Guaranteed Access Time Mode (GAT) that ensures that this timing specification is not violated. This is accomplished by delaying the ISA REQ# signal to the requesting master or DMA until the ISA Bus, PCI Bus, and the System Memory Bus are arbitrated for and owned.

Three PCI sideband signals, MEMREQ#, FLSHREQ#, and MEMACK# are used to support the System Posted Write Buffer Flushing and Guaranteed Access Time mechanisms. The MEMACK# signal is the common acknowledge signal for both mechanisms. Note that when MEMREQ# is asserted, FLSHREQ# is also asserted. Table 9 shows the relationship between MEMREQ# and FLSHREQ#:

Table 9. FLSHREQ#, MEMREQ#

FLSHREQ#	MEMREQ#	Meaning
1	1	Idle
0	1	Flush buffers pointing towards PCI to avoid ISA deadlock
1	0	Reserved
0	0	GAT mode, Guarantee PCI Bus immediate access to main memory

5.2.3.1 Flushing the System Posted Write Buffers

Once an ISA master or the DMA begins a cycle on the ISA Bus, the cycle can not be backed-off. It can only be held in wait states via IOCHRDY. In order to know the destination of ISA master cycles, the cycle needs to begin. However, after the cycle has started, no other device can intervene and gain ownership of the ISA Bus until the cycle has completed and arbitration is performed. A potential deadlock condition exists when an ISA originated cycle to the PCI Bus finds the PCI target inaccessible due to an interacting event that also requires the ISA Bus. To avoid this potential deadlock, all PCI posted write buffers in the system must be disabled and flushed before DACK can be returned. The buffers must remain disabled while the ISA Bus is occupied by an ISA master or the DMA.

When an ISA master or the DMA requests the ISA Bus, the SIO asserts FLSHREQ#. FLSHREQ# is an indication to the system to flush all posted write buffers pointing towards the PCI Bus. The SIO also flushes it's own Posted Write Buffer. Once the posted write buffers have been flushed and disabled, the system asserts MEMACK#. Once the SIO receives the MEMACK# acknowledgment signal, it asserts the DACK signal giving the requesting master the bus. FLSHREQ# stays active as long as the ISA master or DMA owns the ISA Bus.

5.2.3.2 Guaranteed Access Time Mode

Guaranteed Access Time (GAT) Mode is enabled/disabled via the PCI Arbiter Control Register. When this mode is enabled, the MEMREQ# and MEMACK# signals are used to guarantee that the ISA 2.5 μs IOCHRDY specification is not violated.

When an ISA master or DMA slave requests the ISA Bus (DREQ# active), the ISA Bus, the PCI Bus, and the memory bus must be arbitrated for and all three must be owned before the ISA master or DMA slave is granted the ISA Bus. After receiving the DREQ# signal from the ISA master or DMA slave, MEMREQ# and FLSHREQ# are asserted (FLSHREQ# is driven active, regardless of GAT mode being enabled or disabled). MEMREQ# is a request for direct access to main memory. MEMREQ# and FLSHREQ# will be asserted as long as the ISA master or the DMA owns the ISA Bus. When MEMACK# is received by the SIO (all posted write buffers are flushed and the memory bus is dedicated to the PCI interface), it will request the PCI Bus. When it is granted the PCI Bus, it asserts the DACK signal releasing the ISA Bus to the requesting master or the DMA.

The use of MEMREQ#, FLSHREQ#, and MEMACK# does not guarantee functionality with ISA masters that don't acknowledge IOCHRDY. These signals just guarantee the IOCHRDY inactive specification.

NOTE:
Usage of an external arbiter in GAT mode will require special logic in the arbiter.

2

5.2.4 RETRY THRASHING RESOLVE

When a PCI initiator's access is retried, the initiator releases the PCI Bus for a minimum of two PCI clocks and will then normally request the PCI Bus again. To avoid thrashing the bus with retry after retry, the PCI arbiter provides REQ# masking. The REQ# masking mechanism differentiates between SIO target retries and all other retries.

For initiators which were retried by the SIO as a target, the masked REQ# is flagged to be cleared upon RESUME# active. All other retries trigger the Master Retry Timer, if enabled. Upon expiration of this timer, the mask is cleared.

5.2.4.1 Resume Function (RESUME#)

The conditions under which the SIO forces a retry to a PCI master and will mask the REQ# are:

1. Any required buffer management
2. ISA Bus occupied by ISA master or DMA
3. The PCI to ISA Posted Write Buffer is full
4. The SIO is locked as a resource and LOCK# is asserted during the address process.

The RESUME# signal is pulsed whenever the SIO has retried a PCI cycle for one of the above reasons and that condition has passed. When RESUME# is asserted, the SIO will unmask the REQ#'s that are masked and flagged to be cleared by RESUME#.

If the internal arbiter is enabled, RESUME# is an internal signal. The RESUME# signal becomes visible as an output when the internal arbiter is disabled. This allows an external arbiter to optionally avoid retry thrashing associated with the SIO as a target. The RESUME# signal is asserted for one PCI clock.

5.2.4.2 Master Retry Timer

To re-enable a PCI master's REQ# which resulted in a retry to a slave other than the SIO, a SIO programmable Master Retry Timer has been provided. This timer can be programmed for 0 (disabled), 16, 32, or 64 PCICLKs. Once the SIO has detected that a PCI slave has forced a retry, the timer will be triggered and the corresponding master's REQ# will be masked. All subsequent PCI retries by this REQ# signal will by masked by the SIO. Expiration of this timer will unmask all of the masked requests. This timer has no effect on the request lines that have been masked due to a SIO retry.

If no other PCI masters are requesting the PCI Bus, all of the REQ#'s masked for the timer will be cleared and the timer will be reset. This is necessary to assist the host bridge in determining when to re-enable any disabled posted write buffers.

5.2.5 BUS PARKING

The SIO arbitration logic supplies a mechanism for PCI Bus parking. Parking is only allowed for the device which is tied to CPUREQ# (typically the system CPU). When bus parking is enabled, CPUGNT# will be asserted when no other agent is currently using or requesting the bus. This achieves the minimum PCI arbitration latency possible. Enabling of bus parking is achieved by programming the Arbiter Control Register. REQ0#, REQ1#, and the internal SIOREQ# are not allowed to park on the PCI Bus.

Upon assertion of CPUGNT# due to bus parking enabled and the PCI Bus idle, the CPU (or the parked agent) must ensure that AD[31:0], C/BE[3:0], and (one PCICLK later) PAR are driven. If bus parking is disabled, the SIO takes responsibility for driving the bus when it is idle.

5.2.6 BUS LOCK MODE

As an option, the SIO arbiter can be configured to run in Bus Lock Mode or Resource Lock Mode. The Bus Lock Mode is used to lock the entire PCI Bus. This may improve performance in some systems that frequently run quick read-modify-write cycles. Bus Lock Mode emulates the LOCK environment found in today's PC by restricting bus ownership when the PCI Bus is locked. With Bus Lock enabled, the arbiter recognizes a LOCK# being driven by any initiator and does not allow any other PCI initiator to be granted the PCI Bus until LOCK# and FRAME# are both negated indicating the master released lock. When Bus Lock is disabled, the default resource lock mechanism is implemented (normal resource lock) and a higher priority PCI initiator could intervene between the read and write cycles and run non-exclusive accesses to any unlocked resource.

5.2.7 POWER-UP CONFIGURATION

The SIO's arbiter is enabled if CPUREQ# is sampled high on the trailing edge of PCIRST#. When enabled, the arbiter is set in fixed priority mode 4 with CPU bus parking turned off. Fixed mode 4 guarantees that the CPU will be able to run accesses to the BIOS in order to configure the system, regardless of the state of the other REQ#'s. Note that the Host Bridge should drive CPUREQ# high during the trailing edge of PCIRST#. When the arbiter is enabled, the SIO acts as the central resource responsible for driving the AD[31:0], C/BE[3:0]#, and PAR signals when no one is granted the PCI Bus and the bus is idle. The SIO is always responsible for driving AD[31:0], C/BE[3:0]#, and PAR when it is granted the bus and as appropriate when it is the master of a transaction. After reset, if the arbiter is enabled, CPUGNT#, GNT0#, GNT1#, and the internal SIOGNT# will be driven based on the arbitration scheme and the asserted REQ#'s.

If an external arbiter is present in the system, the CPUREQ# signal should be tied low. When CPUREQ# is sampled low on the trailing edge of PCIRST#, the internal arbiter is disabled. When the internal arbiter is disabled, the SIO does not drive AD[31:0], C/BE[3:0]#, and PAR as the central resource. In this case, the SIO is only responsible for driving AD[31:0], C/BE[3:0]#, and PAR when it is granted the bus. If the SIO's arbiter is disabled, GNT0# becomes SIOREQ#, GNT1# becomes RESUME#, and REQ0# becomes SIOGNT#. This exposes the normally embedded SIO arbitration signals.

NOTE:
Usage of an external arbiter in GAT mode will require special logic in the arbiter.

5.3 ISA Interface

5.3.1 ISA INTERFACE OVERVIEW

The SIO incorporates a fully ISA Bus compatible master and slave interface. The SIO directly drives six ISA slots without external data or address buffers. The ISA interface also provides byte swap logic, I/O recovery support, wait-state generation, and SYSCLK generation.

The ISA interface supports the following types of cycles:

- PCI-initiated I/O and memory cycles to the ISA Bus.
- DMA compatible cycles between PCI memory and ISA I/O and between ISA I/O and ISA memory, DMA type "A", type "B", and type "F" cycles between PCI memory and ISA I/O.
- ISA Refresh cycles initiated by either the SIO or an external ISA master.
- ISA master-initiated memory cycles to the PCI Bus and ISA master-initiated I/O cycles to the internal SIO registers.

The refresh and DMA cycles are shown and described in Section 5.4.

5.3.2 SIO AS AN ISA MASTER

The SIO executes ISA cycles as an ISA master whenever a PCI initiated cycle is forwarded to the ISA Bus. The SIO also acts as an ISA master on behalf of DMA and refresh.

ISYSCLK is an internal 8 MHz clock.

5.3.3 SIO AS AN ISA SLAVE

The SIO operates as an ISA slave when:

- An ISA master accesses SIO internal registers.
- An ISA master accesses PCI memory on the PCI Bus.

5.3.3.1 ISA Master Accesses To SIO Registers

An ISA Bus master has access to SIO internal registers as shown in Table 19. An ISA master to SIO register cycle will always run as an 8-bit extended cycle (IOCHRDY will be held inactive until the cycle is completed).

Table 10. Arbitration Latency

Bus Condition	Arbitration Latency
Parked	0 PCICLKs for Agent 0, 2 PCICLKs for All Other
Not Parked	1 PCICLK for All Agents

5.3.3.2 ISA Master Accesses to PCI Resource

An ISA master can access PCI memory, but not I/O devices residing on the PCI Bus. The ISA/DMA address decoder determines which memory cycles should be directed towards the PCI Bus. During ISA master read cycles to the PCI Bus, the SIO will return all 1's if the PCI cycle is target-aborted or does not respond.

If the SIO is programmed for GAT mode, the SIO arbiter will not grant the ISA Bus before gaining ownership of both the PCI Bus and system memory. However, if the SIO is not programmed in this mode, the SIO does not need to arbitrate for the PCI Bus before granting the ISA Bus to the ISA master. For more details on the arbitration, refer to Section 5.2.2.

All cycles forwarded to a PCI resource will run as 16-bit extended cycles (i.e. IOCHRDY will be held inactive until the cycle is completed).

Because the ISA bus size is different from the PCI bus size, the data steering logic inside the SIO is responsible for steering the data to the correct byte lanes on both buses, and assembling/disassembling the data as necessary.

5.3.4 ISA MASTER TO ISA SLAVE SUPPORT

During ISA master cycles to ISA slaves, the SIO drives several signals to support the transfer:

BALE:

This signal is driven high while the ISA master owns the ISA Bus.

AEN:

This signal is driven low while the ISA master owns the ISA Bus.

SMEMR# and **SMEMW#:**

These signals are driven active by the SIO whenever the ISA master drives a memory cycle to an address below 1 Mb.

Utility Bus Buffer Control Signals and Chip Select Signals:

These signals are driven active as appropriate whenever an ISA master accesses devices on the Utility Bus. For more details, see Section 5.9.

Data Swap Logic:

The data swap logic inside the SIO is activated as appropriate to swap data between the even and odd byte lanes. This is discussed in further detail in Section 5.3.5.

5.3.5 DATA BYTE SWAPPING

The data swap logic is integrated in the SIO. For slaves that reside on the ISA Bus, data swapping is performed if the slave (I/O or memory) and ISA bus master (or DMA) sizes differ and the upper (odd) byte of data is being accessed. Table 11 shows when data swapping is provided during DMA. Table 12 shows when data swapping is provided during ISA master cycles to 8-bit ISA slaves.

Table 11. DMA Data Swap

DMA I/O Device Size	ISA Memory Slave Size	Swap	Comments I/O	⟷	Memory
8-Bit	8-Bit	No	SD[7:0]	⟷	SD[7:0]
8-Bit	16-Bit	Yes	SD[7:0]	⟷	SD[7:0]
			SD[7:0]	⟷	SD[15:8]
16-Bit	8-Bit	No	Not Supported		
16-Bit	16-Bit	No	SD[15:0]	⟷	SD[15:0]

The SIO monitors the SBHE# and SA0 signals to determine when to swap the data. The SIO ensures that the data is placed on the appropriate byte lane.

Table 12. 16-Bit Master to 8-Bit Slave Data Swap

SBHE#	SA0	SD[15:8]	SD[7:0]	Comments
0	0	Odd	Even	Word Transfer (data swapping not required)
0	1	Odd	Even	Byte Swap(1,2)
1	0	—	Even	Byte Transfer (data swapping not required)
1	1	—	—	Not Allowed

NOTES:
1. For ISA master read cycles, the SIO swaps the data from the lower byte to the upper byte.
2. For ISA master write cycles, the SIO swaps the data from the upper byte to the lower byte.

5.3.6 ISA CLOCK GENERATION

The SIO generates the ISA system clock (SYSCLK). SYSCLK is a divided down version of the PCICLK (see Table 13). The clock divisor value is programmed through the ISA Clock Divisor Register.

Table 13. SYSCLK Generation from PCICLK

PCICLK (MHz)	Divisor (Programmable)	SYSCLK (MHz)
25	3	8.33
33	4 (default)	8.33

NOTE:
For PCI frequencies less than 33 MHz (not including 25 MHz), a clock divisor value must be selected that ensures that the ISA Bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK specification.

5.3.7 WAIT STATE GENERATION

The SIO will add wait states to the following cycles, if IOCHRDY is sampled active low. Wait states will be added as long as IOCHRDY is low.

- During Refresh and SIO master cycles (not including DMA) to the ISA Bus.
- During DMA compatible transfers between ISA I/O and ISA memory only.

For ISA master cycles targeted for the SIO's internal registers or PCI memory, the SIO will always extend the cycle by driving IOCHRDY low until the transaction is complete.

The SIO will shorten the following cycles, if ZEROWS# is sampled active.

- During SIO master cycles (not including DMA) to 8-bit and 16-bit ISA memory.
- During SIO master cycles (not including DMA) to 8-bit ISA I/O only.

For ISA master cycles targeted for the SIO's internal registers or PCI memory, the SIO will not assert ZEROWS#.

NOTE:
If IOCHRDY and ZEROWS# are sampled active at the same time, IOCHRDY will take precedence and wait-states will be added.

5.3.8 I/O RECOVERY

The I/O recovery mechanism in the SIO is used to add additional recovery delay between PCI originated 8-bit and 16-bit I/O cycles to the ISA Bus. The SIO automatically forces a minimum delay of four SYSCLKs between back-to-back 8- and 16-bit I/O cycles to the ISA Bus. This delay is measured from the rising edge of the I/O command (IOR# or IOW#) to the falling edge of the next BALE. If a delay of greater than four SYSCLKs is required, the ISA I/O Recovery Time Register can be programmed to increase the delay in increments of SYSCLKs. No additional delay is inserted for back-to-back I/O "sub cycles" generated as a result of byte assembly or disassembly .

5.4 DMA Controller

5.4.1 DMA CONTROLLER OVERVIEW

The DMA circuitry incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels (Channels 0–3 and Channels 5–7). DMA Channel 4 is used to cascade the two controllers and will default to cascade mode in the DMA Channel Mode (DCM) Register. In addition to accepting requests from DMA slaves, the DMA controller also responds to requests that are initiated by software. Software may initiate a DMA service request by setting any bit in the DMA Channel Request Register to a 1. The DMA controller for Channels 0–3 is referred to as "DMA-1" and the controller for Channels 4–7 is referred to as "DMA-2".

Each DMA channel may be programmed for 8- or 16-bit DMA device size and ISA-compatible, Type "A", Type "B", or Type "F" transfer timing. Each DMA channel defaults to the compatible settings for DMA device size: channels [3:0] default to 8-bit, count-by-bytes transfers, and channels [7:5] default to 16-bit, count-by-words (address shifted) transfers. The SIO provides the timing control and data size translation necessary for the DMA transfer between the PCI and the ISA Bus. ISA-compatible is the default transfer timing.

Full 32-bit addressing is supported as an extension of the ISA-compatible specification. Each channel includes a 16-bit ISA compatible Current Register which holds the 16 least-significant bits of the 32-bit address, and an ISA compatible Low Page Register which contains the eight second most significant bits. An additional High Page Register contains the eight most significant bits of the 32-bit address.

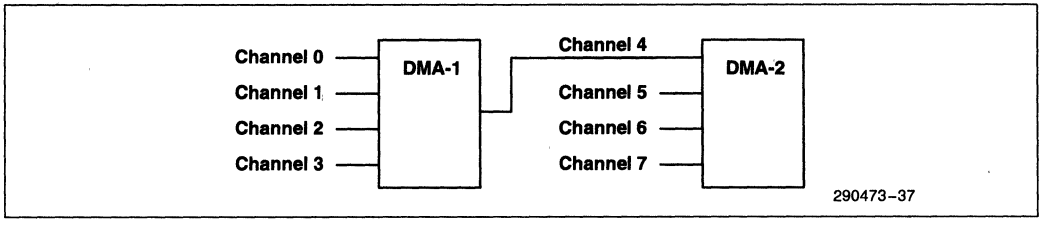


Figure 7. Internal DMA Controller

The DMA controller also features refresh address generation, and auto-initialization following a DMA termination.

The DMA controller receives commands from the ISA Bus arbiter to perform either DMA cycles or refresh cycles. The arbiter determines which requester from among the requesting DMA slaves, the PCI Bus, and refresh should have the bus.

The DMA controller is at any time either in master mode or slave mode. In master mode, the DMA controller is either servicing a DMA slave's request for DMA cycles, or allowing a 16-bit ISA master to use the bus via a cascaded DREQ signal. In slave mode, the SIO monitors both the ISA Bus and the PCI, decoding and responding to I/O read and write commands that address its registers.

Note that a DMA device (I/O device) is always on the ISA Bus, but the memory device is either on the ISA or PCI Bus. If the memory is decoded to be on the ISA Bus, then the DMA cycle will run as a compatible cycle. If the memory is decoded to be on the PCI Bus, the cycle can run as compatible, "A", "B", or "F" type. The ISA controller will not drive a valid address for type "A", "B", and "F" DMA transfers on the ISA Bus.

When the SIO is running a DMA cycle in compatible timing mode, the SIO will drive the MEMR# or MEMW# strobes if the address is less than 16 MBytes (00000000h–00FFFFFFh). These memory strobes will be generated regardless of whether the cycle is decoded for PCI or ISA memory. The SMEMR# and SMEMW# will be generated if the address is less than 1 MBytes (00000000h–000FFFFFh). If the address is greater than 16 MBytes (01000000h–FFFFFFFFh) the MEMR# or MEMW# strobe will not be generated in order to avoid aliasing problems. For type "A", "B", and "F" timing mode DMA cycles, the SIO will only generate the MEMR# or MEMW# strobe when the address is decoded for ISA memory. When this occurs, the cycle converts to compatible mode timing.

During DMA cycles, the ISA controller drives AEN high to prevent the I/O devices from misinterpreting the DMA cycle as a valid I/O cycle. The BALE signal is also driven high during DMA cycles. Also, during DMA memory read cycles to the PCI Bus, the SIO will return all 1's to the ISA Bus if the PCI cycle is either target-aborted or does not respond.

Further details can be found in the 82C37 data sheet.

5.4.2 DMA TIMINGS

ISA Compatible timing is provided for DMA slave devices. Three additional timings are provided for I/O slaves capable of running at faster speeds. These timings are referred to as Type "A", Type "B", and Type "F".

2

5.4.2.1 Compatible Timing

Compatible timing runs at 8 SYSCLKs during the repeated portion of a Block or Demand mode transfer.

5.4.2.2 Type "A" Timing

Type "A" timing is provided to allow shorter cycles to PCI memory. Type "A" timing runs at 6 SYSCLKs (720 ns/cycle) during the repeated portion of a block or demand mode transfer. This timing assumes an 8.33 MHz SYSCLK. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed by system memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter, but it is expected that the DMA devices which provide the data access time or write data setup time should not require excess IOR# or IOW# command active time. Because of this, most ISA DMA devices should be able to use type "A" timing.

5.4.2.3 Type "B" Timing

Type "B" timing is provided for 8-/16-bit ISA DMA devices which can accept faster I/O timing. Type "B" only works with PCI memory. Type "B" timing runs at 5 SYSCLKs (600 ns/cycle) during the repeated portion of a Block or Demand mode transfer. This timing assumes an 8.33 MHz SYSCLK. Type "B" timing requires faster DMA slave devices than compatible timing in that the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster. Some of the current ISA devices should be able to support type "B" timing, but these will probably be more recent designs using relatively fast technology.

5.4.2.4 Type "F" Timing

Type "F" timing provides high performance DMA transfer capability. These transfers are mainly for fast I/O devices (i.e. IDE devices). Type "F" timing runs at 3 SYSCLKs (360 ns/cycle) during the repeated portion of a Block or Demand mode transfer.

5.4.2.5 DREQ and DACK# Latency Control

The SIO DMA arbiter maintains a minimum DREQ to DACK# latency on DMA channels programmed to operate in compatible timing mode. This is to support older devices such as the 8272A. The DREQs are delayed by eight SYSCLKs prior to being seen by the arbiter logic. Software requests will not have this minimum request to DACK# latency.

5.4.3 ISA BUS/DMA ARBITRATION

The ISA Bus arbiter evaluates requests for the ISA Bus coming from several different sources. The DMA unit, the refresh counter, and the PCI Bus (primarily the Host CPU) may all request access to the ISA Bus. Additionally, 16-bit ISA masters may request the bus through a cascaded DMA channel.

The SIO ISA arbiter uses a three-way rotating priority arbitration method. At each level, the devices which are considered equal are given a rotating priority. On a fully loaded bus, the order in which the devices are granted bus access is independent of the order in which they assert a bus request. This is because devices are serviced based on their position in the rotation. The arbitration scheme assures that DMA channels access the bus with minimal latency.

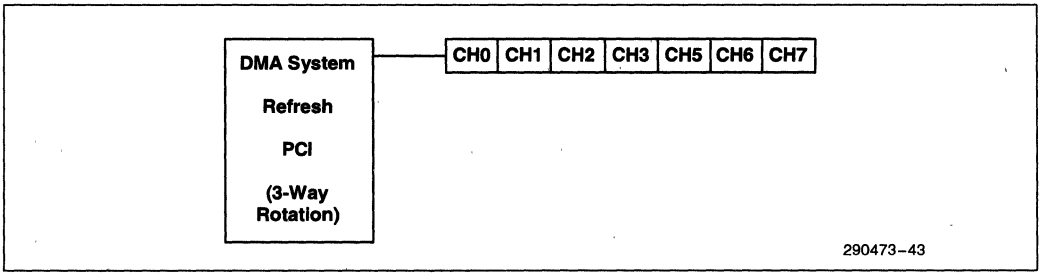


Figure 8. ISA Arbiter with DMA in Fixed Priority

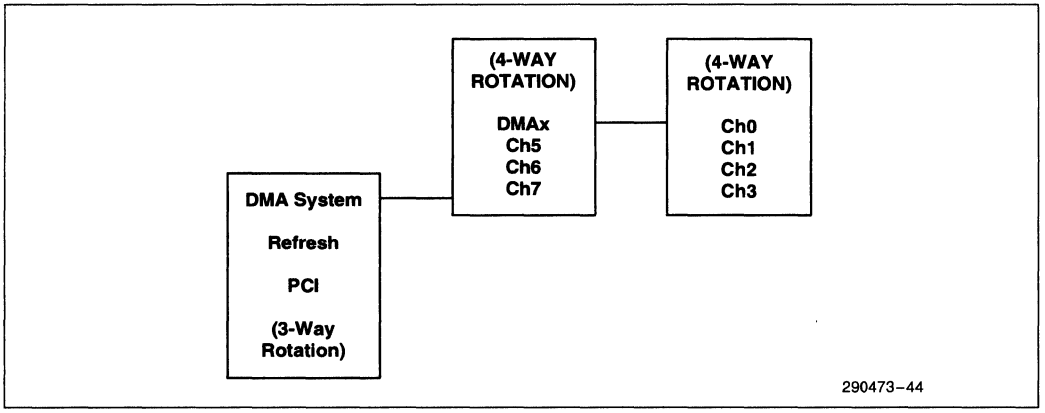


Figure 9. ISA Arbiter with DMA in Rotating Priority

2

5.4.3.1 Channel Priority

For priority resolution the DMA consists of two logical channel groups: channels 0–3 and channels 4–7 (see Figure 7). Each group may be in either fixed or rotate mode, as determined by the DMA Command Register.

For prioritization purposes, the source of the DMA request is transparent. DMA I/O slaves normally assert their DREQ line to arbitrate for DMA service. However, a software request for DMA service can be presented through each channel's DMA Request Register. A software request is subject to the same prioritization as any hardware request. Please see the detailed register description in Section 4.2.4 for Request Register programming information.

Fixed Priority

The initial fixed priority structure is as follows:

Table 14. Initial Fixed Priority Structure

High Priority . . . . . Low Priority
(0, 1, 2, 3), 5, 6, 7

The fixed priority ordering is 0, 1, 2, 3, 5, 6, and 7. In this scheme, Channel 0 has the highest priority, and channel 7 has the lowest priority. Channels [3:0] of DMA-1 assume the priority position of Channel 4 in DMA-2, thus taking priority over channels 5, 6, and 7.

Rotating Priority

Rotation allows for "fairness" in priority resolution. The priority chain rotates so that the last channel serviced is assigned the lowest priority in the channel group (0–3, 5–7).

Channels 0–3 rotate as a group of 4. They are always placed between Channel 5 and Channel 7 in the priority list.

Channel 5–7 rotate as part of a group of 4. That is, channels (5–7) form the first three positions in the rotation, while channel group (0–3) comprises the fourth position in the arbitration.

Table 15 demonstrates rotation priority.

Table 15. Rotating Priority Example

Programmed Mode	Action	Priority High .....Low
Group (0–3) is in Rotation Mode Group (4–7) is in Fixed Mode	1) Initial Setting	(0, 1, 2, 3), 5, 6, 7
	2) After Servicing Channel 2	(3, 0, 1, 2), 5, 6, 7
	3) After Servicing Channel 3	(0, 1, 2, 3), 5, 6, 7
Group (0–3) is in Rotation Mode Group (4–7) is in Rotation Mode	1) Initial Setting	(0, 1, 2, 3), 5, 6, 7
	2) After Servicing Channel 0	5, 6, 7, (1, 2, 3, 0)
	3) After Servicing Channel 5	6, 7, (1, 2, 3, 0), 5
	4) After servicing Channel 6	7, (1, 2, 3, 0), 5, 6
	5) After servicing Channel 7	(1, 2, 3, 0), 5, 6, 7

NOTE:
The first servicing of channel 0 caused double rotation.

5.4.3.2 DMA Preemption In Performance Timing Modes

A DMA slave device that is not programmed for compatible timing will be preempted from the bus by another device that requests use of the bus. This will occur, regardless of the priority of the pending request. For DMA devices not using compatible timing mode, the DMA controller stops the DMA transfer and releases the bus within 32 BCLK (4 μs) of a preemption. Upon the expiration of the 4 μs timer, the DACK will be inactivated after the current DMA cycle has completed. The bus will then be arbitrated for and granted to the highest priority requester. This feature allows flexibility in programming the DMA for long transfer sequences in a performance timing mode while guaranteeing that vital system services such as refresh are allowed access to the ISA Bus.

The 4 μs timer is not used in compatible timing mode. It is only used for DMA channels programmed for Type "A", Type "B", or Type "F" timing. It is also not used for 16-bit ISA masters cascaded through the DMA DREQ lines.

If the DMA channel that was preempted by the 4 μs timer was operating in Block Mode, an internal bit will be set so that the channel will be arbitrated for again, independent of the state of DREQ.

5.4.3.3. Arbitration during Non-Maskable Interrupts

If a non-maskable interrupt (NMI) is pending, and the CPU is requesting the bus, then the DMA controller will be bypassed each time it comes up for rotation. This will give the CPU the bus bandwidth it requires to process the interrupt as fast as possible.

5.4.3.4 Programmable Guaranteed Access Time Mode (GAT Mode)

The PCI Arbiter Register contains a bit for configuring the SIO in "Guaranteed Access Time Mode" (GAT Mode). This mode guarantees that the 2.5 μs CHRDY time-out specification for ISA masters running cycles to PCI will not be exceeded. When an ISA master or DMA slave arbitrates for the ISA Bus, and the SIO is configured in Guaranteed Access Time Mode, the MEMREQ# pin will be asserted by the PCI arbiter in order to gain ownership of main memory. The arbitration for the PCI and then the main memory bus must be completed prior to granting the DACK# to the ISA master or DMA slave. A MEMACK# signal to the SIO indicates that the SIO now owns main memory and can grant the DACK# to the ISA master or DMA slave. A detailed description is contained in Section 5.2.3.2.

5.4.4 REGISTER FUNCTIONALITY

Please see Section 4.2 for detailed information on register programming, bit definitions, and default values/functions of the DMA registers after a PCIRST#.

DMA Channel 4 is used to cascade the two DMA controllers together and should not be programmed for any mode other than cascade. The DMA Channel Mode Register for channel 4 will default to cascade mode. Special attention should also be take when programming the Command and Mask Registers as related to channel 4.

5.4.4.1 Address Compatibility Mode

Whenever the DMA is operating in address compatibility mode, the addresses do not increment or decrement through the High and Low Page Registers, and the High Page Register is set to 00h. This is compatible with the 82C37 and Low Page Register implementation used in the PC/AT. This mode is set when any of the lower three address bytes of a channel are programmed. If the upper byte of a channel's address is programmed last, the channel will go into extended address mode. In this mode, the high byte may be any value and the address will increment or decrement through the entire 32-bit address.

After PCIRST# is negated, all channels will be set to address compatibility mode. The DMA Master Clear command will also reset the proper channels to address compatibility mode.

5.4.4.2 Summary of DMA Transfer Sizes

Table 16 lists each of the DMA device transfer sizes. The column labeled "Current Byte/Word Count Register" indicates that the register contents represents either the number of bytes to transfer or the number of 16-bit words to transfer. The column labeled "Current Address Increment/Decrement" indicates the number added to or taken from the Current Address register after each DMA transfer cycle. The DMA Channel Mode Register determines if the Current Address Register will be incremented or decremented.

5.4.4.3 Address Shifting when Programmed for 16-Bit I/O Count by Words

To maintain compatibility with the implementation of the DMA in the PC/AT which used the 82C37, the DMA will shift the addresses when the DMA Channel Extended Mode Register is programmed for, or defaulted to, transfers to/from a 16-bit device count-by-words. Note that the least significant bit of the Low Page Register is dropped in 16-bit shifted mode. When programming the Current Address Register when the DMA channel is in this mode, the Current Address must be programmed to an even address with the address value shifted right by one bit. The address shifting is shown in Table 17.

5.4.4.4 Autoinitialize

By programming a bit in the DMA Channel Mode Register, a channel may be set up as an autoinitialize channel. During Autoinitialize initialization, the original values of the Current Page, Current Address and Current Byte/Word Count Registers are automatically restored from the Base Page, Address, and Byte/Word Count Registers of that channel following TC. The Base Registers are loaded simultaneously with the Current Registers by the microprocessor and remain unchanged throughout the DMA service. The mask bit is not set when the channel is in autoinitialize. Following Autoinitialize, the channel is ready to perform another DMA service, without CPU intervention, as soon as a valid DREQ is detected.

Table 16. DMA Transfer Size

DMA Device Date Size and Word Count	Current Byte/Word Count Register	Current Address Increment/Decrement
8-Bit I/O, Count by Bytes	Bytes	1
16-Bit I/O, Count by Words (Address Shifted)	Words	1
16-Bit I/O, Count by Bytes	Bytes	2

Table 17. Address Shifting in 16-Bit I/O DMA Transfers

Output Address	8-Bit I/O Programmed Address	16-Bit I/O Programmed Address (Shifted)	16-Bit I/O Programmed Address (No Shift)
A0	A0	0	A0
A[16:1]	A[16:1]	A[15:0]	A[16:1]
A[31:17]	A[31:17]	A[31:17]	A[31:17]

NOTE:
The least significant bit of the Low Page Register is dropped in 16-bit shifted mode.

5.4.5 SOFTWARE COMMANDS

There are three additional special software commands which can be executed by the DMA controller. The three software commands are:

1. Clear Byte Pointer Flip-Flop
2. Master Clear
3. Clear Mask Register

They do not depend on any specific bit pattern on the data bus.

5.4.5.1 Clear Byte Pointer Flip-Flop

This command is executed prior to writing or reading new address or word count information to/from the DMA controller. This initializes the flip-flop to a known state so that subsequent accesses to register contents by the microprocessor will address upper and lower bytes in the correct sequence.

When the Host CPU is reading or writing DMA registers, two Byte Pointer Flip-Flops are used; one for channels 0–3 and one for channels 4–7. Both of these act independently. There are separate software commands for clearing each of them (0Ch for channels 0–3, 0D8h for channels 4–7).

5.4.5.2 DMA Master Clear

This software instruction has the same effect as the hardware reset. The Command, Status, Request, and Internal First/Last Flip-Flop Registers are cleared and the Mask Register is set. The DMA controller will enter the idle cycle.

There are two independent master clear commands, 0Dh which acts on channels 0–3, and 0DAh which acts on channels 4–7.

5.4.5.3 Clear Mask Register

This command clears the mask bits of all four channels, enabling them to accept DMA requests. I/O port 00Eh is used for channels 0–3 and I/O port 0DCh is used for channels 4–7.

5.4.6 TERMINAL COUNT/EOP SUMMARY

This is a summary of the events that will happen as a result of a terminal count or external EOP when running DMA in various modes. (See Table 18.)

5.4.7 ISA REFRESH CYCLES

Refresh cycles are generated by two sources: the refresh controller inside the SIO component or by ISA bus masters other than the SIO. The ISA bus controller will enable the address lines SA[15:0] so that when MEMR# goes active, the entire ISA system memory is refreshed at one time. Memory slaves on the ISA Bus must not drive any data onto the data bus during the refresh cycle.

Counter 1 in the timer register set should be programmed to provide a request for refresh about every 15 μs.

Table 18. Terminal Count/EOP Summary Table

Conditions AUTOINIT	No		Yes	
Event				
Word Counter Expired	Yes	X	Yes	X
EOP Input	X	Asserted	X	Asserted
Result				
Status TC	set	set	set	set
Mask	set	set	—	—
SW Request	clr	clr	clr	clr
Current Register	—	—	load	load

NOTES:
load = load current from base
— = no change
X = don't care
clr = clear

5.4.8 SCATTER/GATHER DESCRIPTION

Scatter/Gather (S/G) provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In Scatter/Gather, the DMA can read the memory address and word count from an array of buffer descriptors, located in system memory (ISA or PCI), called the Scatter/Gather Descriptor (SGD) Table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD Table are transferred.

The S/G Command and Status Registers are used to control the operational aspect of S/G transfers. The SGD Table Pointer Register holds the address of the next buffer descriptor in the SGD Table.

The next buffer descriptor is fetched from the SGD Table by a DMA read transfer. DACK# will not be asserted for this transfer because the IO device is the SIO itself. The SIO will fetch the next buffer descriptor from either PCI memory or ISA memory, depending on where the SGD Table is located. If the SGD table is located in PCI memory, the memory read will use the line buffer to temporarily store the PCI read before loading it into the DMA S/G registers. The line buffer mode (8 byte or single transaction) for the S/G fetch operation will be the same as what is set for all DMA operations. If set in 8 byte mode, the SGD Table fetches will be PCI burst memory reads. The SGD Table PCI cycle fetches are subject to all types of PCI cycle termination (retry, disconnect, target-abort, master-abort). The fetched SGD Table data is subject to normal line buffer coherency management and invalidation. EOP will be asserted at the end of the complete link transfer.

To initiate a typical DMA Scatter/Gather transfer between memory and an I/O device, the following steps are required:

1. Software prepares a SGD Table in system memory. Each SGD is 8 bytes long and consists of an address pointer to the starting address and the transfer count of the memory buffer to be transferred. In any given SGD Table, two consecutive SGDs are offset by 8 bytes and are aligned on a 4-byte boundary.

   Each Scatter/Gather Descriptor for the linked list contains the following information:

   a. Memory Address (buffer start) 4 bytes

   b. Transfer Size (buffer size) 2 bytes

   c. End of Link List 1 bit (MSB)
2. Initialize the DMA Channel Mode and DMA Channel Extended Mode Registers with transfer specific information like 8-/16-bit I/O device, Transfer Mode, Transfer Type, etc.
3. Software provides the starting address of the SGD Table by loading the SGD Table Pointer Register.
4. Engage the Scatter/Gather function by writing a Start command to the S/G Command Register.
5. The Mask register should be cleared as the last step of programming the DMA register set. This is to prevent the DMA from starting a transfer with a partially loaded command description.
6. Once the register set is loaded and the channel is unmasked, the DMA will generate an internal request to fetch the first buffer from the SGD Table.

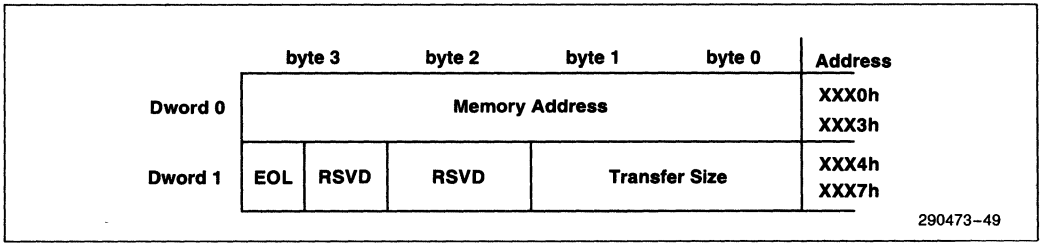


Figure 10. SGD Format

After the above steps are finished, the DMA will then respond to DREQ or software requests. The first transfer from the first buffer moves the memory address and word count from the Base register set to the Current register set. As long as S/G is active and the Base register set is not loaded and the last buffer has not been fetched, the channel will generate a request to fetch a reserve buffer into the Base register set. The reserve buffer is loaded to minimize latency problems going from one buffer to another. Fetching a reserve buffer has a lower priority than completing DMA transfers for the channel.

The DMA controller will terminate a Scatter/Gather cycle by detecting an End of List (EOL) bit in the SGD Table. After the EOL bit is detected, the channel transfers the buffers in the Base and Current register sets, if they are loaded. At terminal count the channel asserts EOP or IRQ13, depending on its programming and set the terminate bit in the S/G Status Register. If the channel asserted IRQ13, then the appropriate bit is set in the S/G Interrupt Status Register. The active bit in the S/G Status Register will be reset and the channel's Mask bit will be set.

5.5 Address Decoding

The SIO contains two address decoders; one to decode PCI master cycles and one to decode DMA/ISA master cycles. Two decoders are required to support the PCI and ISA Buses running concurrently. The PCI address decoder decodes the address from the multiplexed PCI address/data bus. The DMA/ISA master address decoder decodes the address from the ISA address bus for DMA and ISA master cycles. The address decoders determine how the cycle is handled.

5.5.1 PCI ADDRESS DECODER

PCI address decoding is always a function of AD[31:2]. The information contained in the two low order bits (AD[1:0]) varies for memory, I/O, and configuration cycles.

2

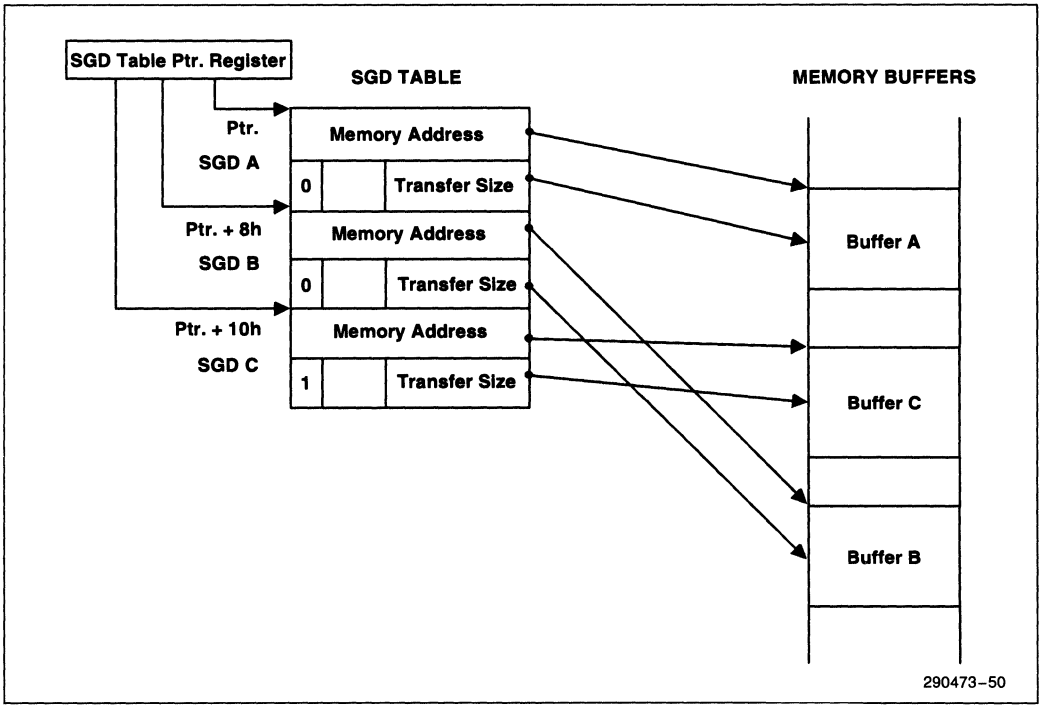


Figure 11. Link List Example

For I/O cycles, AD[31:0] are all decoded to provide a byte address. The byte enables determine which byte lanes contain valid data. The SIO requires that PCI accesses to byte-wide internal registers must assert only one byte enable.

For memory cycles, accesses are decoded as Dword accesses. This means that AD[1:0] are ignored for decoding memory cycles. The byte enables are used only to determine which byte lanes contain valid data.

For configuration cycles, DEVSEL# is a function of IDSEL# and AD[1:0]. DEVSEL# is generated only when AD[1:0] are both zero. If either AD[1:0] are non-zero, the cycle is ignored by the SIO. Individual bytes of a configuration register can be accessed with the byte enables. A particular configuration register is selected using AD[7:2]. Again, the byte enables determine which byte lanes contain valid data.

All PCI cycles decoded in one of the following ways result in the SIO generating DEVSEL#. The PCI master cycle decoder decodes the following addresses based on the settings of the relevant configuration registers:

SIO I/O Addresses: Positively decodes I/O addresses for registers contained within the SIO (exceptions: 60h, 92h, 3F2h, 372h, and F0h).

BIOS Memory Space: Positively decodes BIOS memory space.

MEMCS# Address Decoding: Decodes memory addresses that reside on the other side of the Host bridge and generates the MEMCS# signal. (SIO does not generate DEVSEL# in this case). The address range(s) used for this decoding is selected via the MEMCSCON, MEMCSBOH, MEMCSTOH, MEMCSTOM, MAR1, MAR2, and MAR3 Registers (see Section 4.1).

Subtractively Decoding Cycles to ISA: Subtractively decodes cycles to the ISA Bus. Accesses to registers 60h, 92h, 3F2h, 372h, and F0h are also subtractively decoded to the ISA Bus.

One of the PCI requirements is that, upon power-up, PCI agents do not respond to any address. Typically, the only access to a PCI agent is through the IDSEL configuration mechanism until it is enabled through configuration. The SIO is an exception to this, since it controls access to the BIOS boot code. All addresses decoded by the PCI address decoder, that are enabled after chip reset, are accessible immediately after power-up.

5.5.1.1 SIO I/O Addresses

These addresses are the internal, non-configuration SIO register locations and are shown in the SIO Address Decoding Table, Table 19. These addresses are fixed. Note that the Configuration Registers, listed in Table 3, are accessed with PCI configuration cycles as described in Section 5.1.2.5

In general, PCI accesses to the internal SIO registers will not be broadcast to the ISA Bus. However, PCI accesses to addresses 70h, 60h, 92h, 3F2h, 372h, and F0h are exceptions. Read and write accesses to these SIO locations are broadcast onto the ISA Bus. PCI master accesses to SIO registers will be retried if the ISA Bus is owned by an ISA master or the DMA controller. All of the registers are 8 bit registers. Accesses to these registers must be 8 bit accesses. Target-abort is issued by the SIO when the internal SIO non-configuration registers are the target of a PCI master I/O cycle and more than one byte enable is active. Refer to Table 19 for the SIO Address Decoding Map.

Accesses to the BIOS Timer Register (78h–7Bh) are broadcast to the ISA bus only if this register is disabled. If this register is enabled, the cycle is not broadcast to the ISA bus.

The address decoding logic includes the read/write cycle type. For example, read cycles to write only registers are not positively decoded and get forwarded to the ISA bus via subtractive decoding.

Table 19. SIO Address Decoding

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
0000h	0000	0000	000x	0000	r/w	DMA1 CH0 Base and Current Address	DMA
0001h	0000	0000	000x	0001	r/w	DMA1 CH0 Base and Current Count	DMA
0002h	0000	0000	000x	0010	r/w	DMA1 CH1 Base and Current Address	DMA
0003h	0000	0000	000x	0011	r/w	DMA1 CH1 Base and Current Count	DMA
0004h	0000	0000	000x	0100	r/w	DMA1 CH2 Base and Current Address	DMA
0005h	0000	0000	000x	0101	r/w	DMA1 CH2 Base and Current Count	DMA
0006h	0000	0000	000x	0110	r/w	DMA1 CH3 Base and Current Address	DMA
0007h	0000	0000	000x	0111	r/w	DMA1 CH3 Base and Current Count	DMA
0008h	0000	0000	000x	1000	r/w	DMA1 Status(r) Command(w) Register	DMA
0009h	0000	0000	000x	1001	wo	DMA1 Write Request Register	DMA
000Ah	0000	0000	000x	1010	wo	DMA1 Write Single Mask Bit	DMA
000Bh	0000	0000	000x	1011	wo	DMA1 Write Mode Register	DMA
000Ch	0000	0000	000x	1100	wo	DMA1 Clear Byte Pointer	DMA
000Dh	0000	0000	000x	1101	wo	DMA1 Master Clear	DMA
000Eh	0000	0000	000x	1110	wo	DMA1 Clear Mask Register	DMA
000Fh	0000	0000	000x	1111	r/w	DMA1 Read/Write All Mask Register Bits	DMA
0020h	0000	0000	001x	xx00	r/w	INT 1 Control Register	PIC
0021h	0000	0000	001x	xx01	r/w	INT 1 Mask Register	PIC
0040h	0000	0000	010x	0000	r/w	Timer Counter 1—Counter 0 Count	TC
0041h	0000	0000	010x	0001	r/w	Timer Counter 1—Counter 1 Count	TC
0042h	0000	0000	010x	0010	r/w	Timer Counter 1—Counter 2 Count	TC
0043h	0000	0000	010x	0011	wo	Timer Counter 1 Command Mode Register	TC
0060h	0000	0000	0110	0000	ro	Reset UBus IRQ12	Control
0061h	0000	0000	0110	0xx1	r/w	NMI Status and Control	Control
0070h	0000	0000	0111	0xx0	wo	CMOS RAM Address and NMI Mask Register	Control
0078h(1)	0000	0000	0111	10xx	r/w	BIOS Timer	TC

2

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
0080h	0000	0000	100x	0000	r/w	DMA Page Register (Reserved)	DMA
0081h	0000	0000	100x	0001	r/w	DMA Channel 2 Page Register	DMA
0082h	0000	0000	1000	0010	r/w	DMA Channel 3 Page Register	DMA
0083h	0000	0000	100x	0011	r/w	DMA Channel 1 Page Register	DMA
0084h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0085h	0000	0000	100x	0101	r/w	DMA Page Register (Reserved)	DMA
0086h	0000	0000	100x	0110	r/w	DMA Page Register (Reserved)	DMA
0087h	0000	0000	100x	0111	r/w	DMA Channel 0 Page Register	DMA
0088h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0089h	0000	0000	100x	1001	r/w	DMA Channel 6 Page Register	DMA
008Ah	0000	0000	100x	1010	r/w	DMA Channel 7 Page Register	DMA
008Bh	0000	0000	100x	1011	r/w	DMA Channel 5 Page Register	DMA
008Ch	0000	0000	100x	1100	r/w	DMA Page Register (Reserved)	DMA
008Dh	0000	0000	100x	1101	r/w	DMA Page Register (Reserved)	DMA
008Eh	0000	0000	100x	1110	r/w	DMA Page Register (Reserved)	DMA
008Fh	0000	0000	100x	1111	r/w	DMA Low Page Register Refresh	DMA
0090h	0000	0000	100x	0000	r/w	DMA Page Register (Reserved)	DMA
0092h	0000	0000	1001	0010	r/w	System Control Port	Control
0094h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0095h	0000	0000	100x	0101	r/w	DMA Page Register (Reserved)	DMA
0096h	0000	0000	100x	0110	r/w	DMA Page Register (Reserved)	DMA
0098h	0000	0000	100x	1000	r/w	DMA Page Register (Reserved)	DMA
009Ch	0000	0000	100x	1100	r/w	DMA Page Register (Reserved)	DMA
009Dh	0000	0000	100x	1101	r/w	DMA Page Register (Reserved)	DMA
009Eh	0000	0000	100x	1110	r/w	DMA Page Register (Reserved)	DMA
009Fh	0000	0000	100x	1111	r/w	DMA low page Register Refresh	DMA
00A0h	0000	0000	101x	xx00	r/w	INT 2 Control Register	PIC
00A1h	0000	0000	101x	xx01	r/w	INT 2 Mask Register	PIC
00B2h	0000	0000	1011	0010	r/w	Advanced Power Management Control Port	PM
00B3h	0000	0000	1011	0011	r/w	Advanced Power Management Status Port	PM
00C0h	0000	0000	1100	000x	r/w	DMA2 CH0 Base and Current Address	DMA

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
00C2h	0000	0000	1100	001x	r/w	DMA2 CH0 Base and Current Count	DMA
00C4h	0000	0000	1100	010x	r/w	DMA2 CH1 Base and Current Address	DMA
00C6h	0000	0000	1100	011x	r/w	DMA2 CH1 Base and Current Count	DMA
00C8h	0000	0000	1100	100x	r/w	DMA2 CH2 Base and Current Address	DMA
00CAh	0000	0000	1100	101x	r/w	DMA2 CH2 Base and Current Count	DMA
00CCh	0000	0000	1100	110x	r/w	DMA2 CH3 Base and Current Address	DMA
00CEh	0000	0000	1100	111x	r/w	DMA2 CH3 Base and Current Count	DMA
00D0h	0000	0000	1101	000x	r/w	DMA2 Status(r) Command(w) Register	DMA
00D2h	0000	0000	1101	001x	wo	DMA2 Write Request Register	DMA
00D4h	0000	0000	1101	010x	wo	DMA2 Write Single Mask Bit	DMA
00D6h	0000	0000	1101	011x	wo	DMA2 Write Mode Register	DMA
00D8h	0000	0000	1101	100x	wo	DMA2 Clear Byte Pointer	DMA
00DAh	0000	0000	1101	101x	wo	DMA2 Master Clear	DMA
00DCh	0000	0000	1101	110x	wo	DMA2 Clear Mask Register	DMA
00DEh	0000	0000	1101	111x	r/w	DMA2 Read/Write All Mask Register Bits	DMA
00F0h	0000	0000	1111	0000	wo	Coprocessor Error	Control
0372h	0000	0011	0111	0010	wo	Secondary Floppy Disk Digital Output Reg.	Control
03F2h	0000	0011	1111	0001	wo	Primary Floppy Disk Digital Output Reg.	Control
040Ah	0000	0100	0000	1010	ro	Scatter/Gather Interrupt Status Register	DMA
040Bh	0000	0100	0000	1011	wo	DMA1 Extended Mode register	DMA
0410h(1)	0000	0100	0001	0000	wo	CH0 Scatter/Gather Command	DMA
0411h(1)	0000	0100	0001	0001	wo	CH1 Scatter/Gather Command	DMA
0412h(1)	0000	0100	0001	0010	wo	CH2 Scatter/Gather Command	DMA
0413h(1)	0000	0100	0001	0011	wo	CH3 Scatter/Gather Command	DMA
0415h(1)	0000	0100	0001	0101	wo	CH5 Scatter/Gather Command	DMA
0416h(1)	0000	0100	0001	0110	wo	CH6 Scatter/Gather Command	DMA
0417h(1)	0000	0100	0001	0111	wo	CH7 Scatter/Gather Command	DMA
0418h(1)	0000	0100	0001	1000	ro	CH0 Scatter/Gather Status	DMA
0419h(1)	0000	0100	0001	1001	ro	CH1 Scatter/Gather Status	DMA
041Ah(1)	0000	0100	0001	1010	ro	CH2 Scatter/Gather Status	DMA
041Bh(1)	0000	0100	0001	1011	ro	CH3 Scatter/Gather Status	DMA

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
041Dh(1)	0000	0100	0001	1101	ro	CH5 Scatter/Gather Status	DMA
041Eh(1)	0000	0100	0001	1110	ro	CH6 Scatter/Gather Status	DMA
041Fh(1)	0000	0100	0001	1111	ro	CH7 Scatter/Gather Status	DMA
0420h(1)	0000	0100	0010	00xx	r/w	CH0 Scatter/Gather Descriptor Table Pointer	DMA
0424h(1)	0000	0100	0010	01xx	r/w	CH1 Scatter/Gather Descriptor Table Pointer	DMA
0428h(1)	0000	0100	0010	10xx	r/w	CH2 Scatter/Gather Descriptor Table Pointer	DMA
042Ch(1)	0000	0100	0010	11xx	r/w	CH3 Scatter/Gather Descriptor Table Pointer	DMA
0434h(1)	0000	0100	0011	01xx	r/w	CH5 Scatter/Gather Descriptor Table Pointer	DMA
0438h(1)	0000	0100	0011	10xx	r/w	CH6 Scatter/Gather Descriptor Table Pointer	DMA
043Ch(1)	0000	0100	0011	11xx	r/w	CH7 Scatter/Gather Descriptor Table Pointer	DMA
0481h	0000	0100	1000	0001	r/w	DMA CH2 High Page Register	DMA
0482h	0000	0100	1000	0010	r/w	DMA CH3 High Page Register	DMA
0483h	0000	0100	1000	0011	r/w	DMA CH1 High Page Register	DMA
0487h	0000	0100	1000	0111	r/w	DMA CH0 High Page Register	DMA
0489h	0000	0100	1000	1001	r/w	DMA CH6 High Page Register	DMA
048Ah	0000	0100	1000	1010	r/w	DMA CH7 High Page Register	DMA
048Bh	0000	0100	1000	1011	r/w	DMA CH5 High Page Register	DMA
04D0	0000	0100	1101	0000	r/w	INT CNTRL-1 Edge Level Control Register	Control
04D1	0000	0100	1101	0001	r/w	INT CNTRL-2 Edge Level Control Register	Control
04D6h	0000	0100	1101	0010	wo	DMA2 Extended Mode Register	DMA

NOTE:
1. The I/O address of this register is relocatable. The value shown in this table is the default address location.

5.5.1.2 BIOS Memory Space

The 128 Kb BIOS memory space is located at 000E0000h to 000FFFFFh (top of 1 Mb), and is aliased at FFFE0000h to FFFFFFFFh (top of 4 Gb) and FFEE0000h to FFEFFFFFh (top of 4 Gb-1 Mb). The aliased regions account for the CPU reset vector and the uncertainty of the state of A20GATE when a software reset occurs. This 128 Kb block is split into two 64 Kb blocks. The top 64 Kb is always enabled while the bottom 64 Kb can be enabled or disabled (the aliases automatically match). Enabling the lower 64 Kb BIOS space (000E0000h to 000EFFFFh, 896 Kb-960 Kb) results in positively decoding this region and enables the BIOSCS# signal generation. The upper 64 Kb is positively decoded only if bit 6 = 1 in the ISA Clock Divisor Register. Otherwise this region is subtractively decoded. Positively decoding these cycles expedites BIOS cycles to the ISA Bus. Note that both of these regions are subtractively decoded if bit 4 in the MEMCS# Control Register is set to a 1.

When PCI master accesses to the 128 Kb BIOS space at 4 Gb-1 Mb are forwarded to the ISA Bus, the LA20 line is driven to a 1 to avoid aliasing at the 15 Mb area. The 4 Gb-1 Mb BIOS decode area accounts for the condition when A20M# is asserted and an ALT-CTRL-DEL reset is generated. The CPU's reset vector will access 4 Gb-1 Mb. When this gets forwarded to ISA, AD[32:24] are truncated and the access is aliased to 16 Mb-1 Mb = 15 Mb space. If ISA memory is present at 15 Mb, there will be contention. Forcing LA20 high aliases this region to 16 Mb. The alias here is permissible since this is the 80286 reset vector location.

In addition to the normal 128 Kb BIOS space, the SIO supports an additional 384 Kb BIOS space. The SIO can support a total of 512 Kb BIOS space. The additional 384 Kb region can only be accessed by PCI masters and resides at FFF80000h to FFFDFFFFh. When enabled via the UBCSA Register, memory accesses within this region will be positively decoded, forwarded to the ISA Bus, and encoded BIOSCS# will be generated. When forwarded to the ISA Bus, the PCI AD[23:20] signals will be propagated to the ISA LA[23:20] lines as all 1's which will result in aliasing this 512 Kb region at the top of the 16 Mb space. To avoid contention, ISA add-in memory must not be present in this space.

All PCI cycles positively decoded in the enabled BIOS space will be broadcast to the ISA Bus. Since the BIOS device is 8 or 16 bits wide and typically has very long access times, PCI burst reads from the BIOS space will invoke "disconnect target termination" semantics after the first data transaction in order to meet the PCI incremental latency guidelines.

The following tables and diagrams describe the operation of the SIO in response to PCI BIOS space accesses.

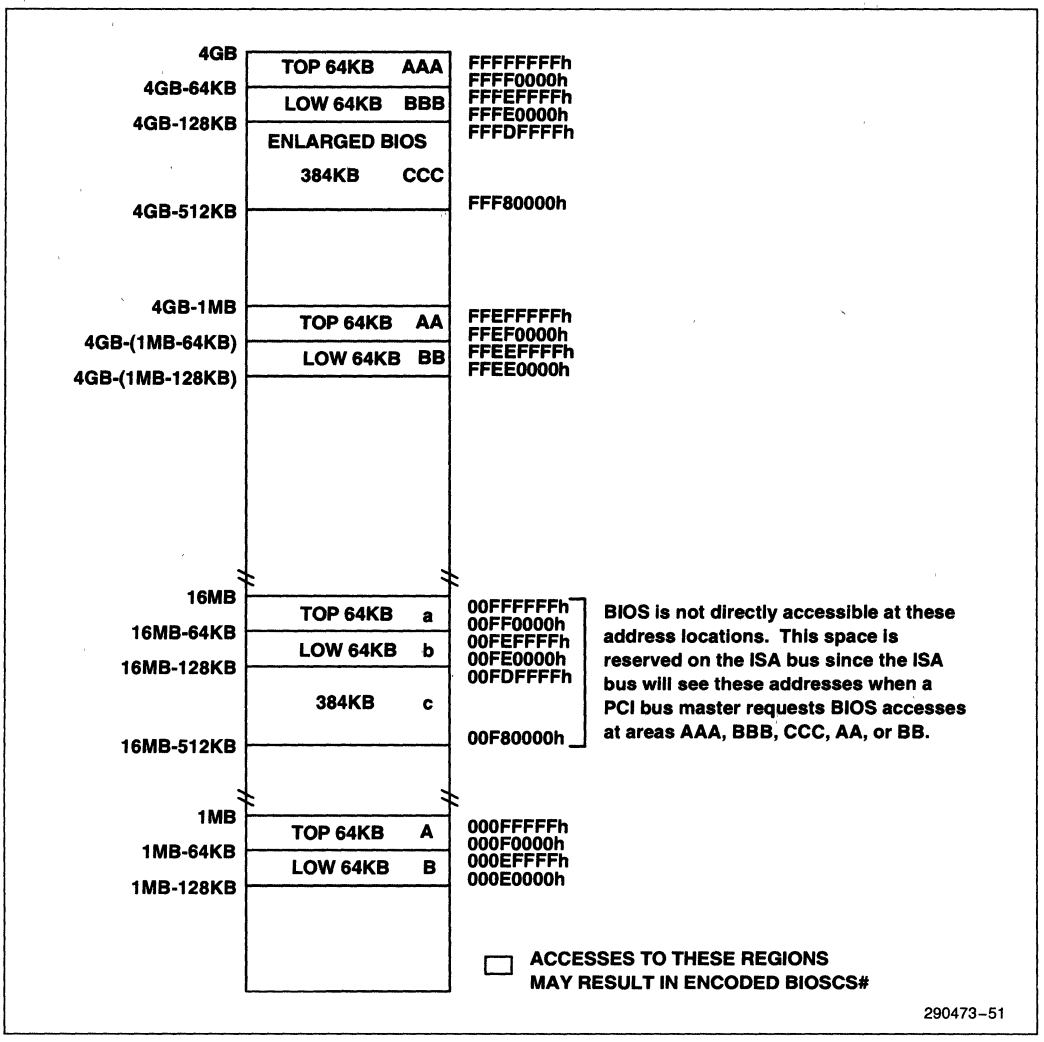


Figure 12. BIOS Space Decode Map

The BIOS space decode map, Figure 12, shows the possible BIOS spaces and the aliases throughout the memory space. The various regions are designated with code letters; "a's" for the top 64 Kb, "b's" for the low 64 Kb, and "c's" for the enlarged space.

Table 20 indicates the SIO's response to PCI BIOS space accesses based on its configuration state.

Table 20. PCI Master BIOS Space Decoding

Master	Region	Top 64 Kb BIOS Positive Decode Enabled(1)	Low 64 Kb BIOS Enabled(2)	Enlarged BIOS Enabled(3)	Encoded BIOSCS# Generated	LA20	Positive PCI Decode	Subtractive PCI Decode
PCI	A	0	x	x	Yes	Pass (0)	No	Yes
PCI	A	1	x	x	Yes	Pass (0)	Yes(5)	No(5)
PCI	B	x	0	x	No	Pass (0)	No	Yes
PCI	B	x	1	x	Yes	Pass (0)	Yes(5)	No(5)
PCI	a	1	x	x	No	Pass (1)	No	Yes
PCI	b	x	0	x	No	Pass (1)	No	Yes
PCI	c	x	x	0	No	Pass (1)	No	Yes
PCI	AA	0	x	x	Yes	1	No	Yes(4)
PCI	AA	1	x	x	Yes	1	Yes(4,5)	No(5)
PCI	BB	x	0	x	No	x	No	No
PCI	BB	x	1	x	Yes	1	Yes(4,5)	No(5)
PCI	AAA	0	x	x	Yes	Pass (1)	No	Yes(4)
PCI	AAA	1	x	x	Yes	Pass (1)	Yes(4,5)	No(5)
PCI	BBB	x	0	x	No	x	No	No
PCI	BBB	x	1	x	Yes	Pass (1)	Yes(4,5)	No(5)
PCI	CCC	x	x	0	No	x	No	No
PCI	CCC	x	x	1	Yes	Pass (1)	Yes(4)	No

NOTES:
1. The column labeled "Top 64 Kb BIOS Positive Decode Enable" shows the value of the ISA Clock Register bit 6. This bit determines the decoding for memory regions A, AA, and AAA (1 = positive, 0 = negative decoding).
   Note that if bit 4 in the MEMCS# Control Register is set to a 1 (Global MEMCS# decode enabled), the positive decoding function enabled by having ISA Clock Register bit 6 = 1 is ignored. Subtractive decoding is provided, instead.
2. The column labeled "Low 64 Kb BIOS Enable" shows the value of the Utility Bus Chip Select Enable A Bit 6. This bit determines whether memory regions B, BB, and BBB are enabled (bit = 1) or disabled (bit = 0).
3. The column labeled "Enlarged BIOS Enabled" shows the value of the Utility Bus Chip Select Enable A Bit 7. This bit determines whether memory region CCC is enabled (bit = 1) or disabled (bit = 0).
4. ISA memory is not allowed to be enabled at the corresponding aliased areas or contention will result.
5. When bit 4 in the MEMCS# Control Register is set to a 1 (Global MEMCS# decode enabled), positive decoding for these areas will be disabled. The SIO will only provide subtractive decoding in this case.

5.5.1.3 MEMCS# Decoding

For MEMCS# decoding, the SIO decodes sixteen ranges. Fourteen of these ranges can be enabled or disabled independently for both read and write cycles. The fifteenth range (0 KB–512 KB) and sixteenth range (programmable from 1 MB up to 512 MB in 2 MB increments) can be enabled or disabled only. Addresses within these enabled regions generate a MEMCS# signal that can be used by the host bridge to know when to forward PCI cycles to main memory. A seventeenth range is available that can be used to identify a "memory hole". Addresses within this hole will not generate a MEMCS#. The address regions are summarized:

- 0 KB to 512 KB Memory (can only be disabled if MEMCS# is completely disabled)
- 512 KB to 640 KB Memory
- (1 MB–64 KB) to 1 MB Memory (BIOS Area)
- 768 KB to 918 KB in 16 KB sections (total of 8 sections)
- 918 KB to 983 KB in 16 KB sections (total of 4 sections)
- 1M-to-programmable boundary on 2 MB increments from 2 MB up to 512 MB
- programmable "memory hole" in 64 KB increments between 1 MB and 16 MB

Table 21 and Figure 13 show the registers and decode areas for MEMCS#.

Table 21. MEMCS# Decoding Register Summary

MAR Registers	Attribute	Memory Segments	Comments
MCSCON[4] = 0	Disable	Disable MEMCS# Function	Enable/Disable MEMCS# Function
MCSCON[4] = 1	Enable	Enable MEMCS# Function	When Enabled, 0 KB to 512 KB Range is also Automatically Enabled (RE/WE)
MCSTOH/ MCSBOH	MEMCS# Hole	100000h–0FFFFFFh	1 MB to 16 MB Hole in MEMCS# Region
MCSTOM	MEMCS# Top	200000h–1FFFFFFFh	2 MB to 512 MB Top of MEMCS# Region
MCSCON[1:0]	[0] = RE[1] = WE	080000h–09FFFFh	512 KB to 640 KB R/W Enable
MCSCON[3:2]	[2] = RE[3] = WE	0F0000h–0FFFFFh	BIOS Area R/W Enable
MAR1[1:0]	[0] = RE[1] = WE	0C0000h–0C3FFFh	ISA Add-On BIOS R/W Enable
MAR1[3:2]	[2] = RE[3] = WE	0C4000h–0C7FFFh	ISA Add-On BIOS R/W Enable
MAR1[5:4]	[4] = RE[5] = WE	0C8000h–0CBFFFh	ISA Add-On BIOS R/W Enable
MAR1[7:6]	[6] = RE[7] = WE	0CC000h–0CFFFFh	ISA Add-On BIOS R/W Enable
MAR2[1:0]	[0] = RE[1] = WE	0D0000h–0D3FFFh	ISA Add-On BIOS R/W Enable
MAR2[3:2]	[2] = RE[3] = WE	0D4000h–0D7FFFh	ISA Add-On BIOS R/W Enable
MAR2[5:4]	[4] = RE[5] = WE	0D8000h–0DBFFFh	ISA Add-On BIOS R/W Enable
MAR2[7:6]	[6] = RE[7] = WE	0DC000h–0DFFFFh	ISA Add-On BIOS R/W Enable
MAR3[1:0]	[0] = RE[1] = WE	0E0000h–0E3FFFh	BIOS Extension R/W Enable
MAR3[3:2]	[2] = RE[3] = WE	0E4000h–0E7FFFh	BIOS Extension R/W Enable
MAR3[5:4]	[4] = RE[5] = WE	0E8000h–0EBFFFh	BIOS Extension R/W Enable
MAR3[7:6]	[6] = RE[7] = WE	0EC000h–0EFFFFh	BIOS Extension R/W Enable

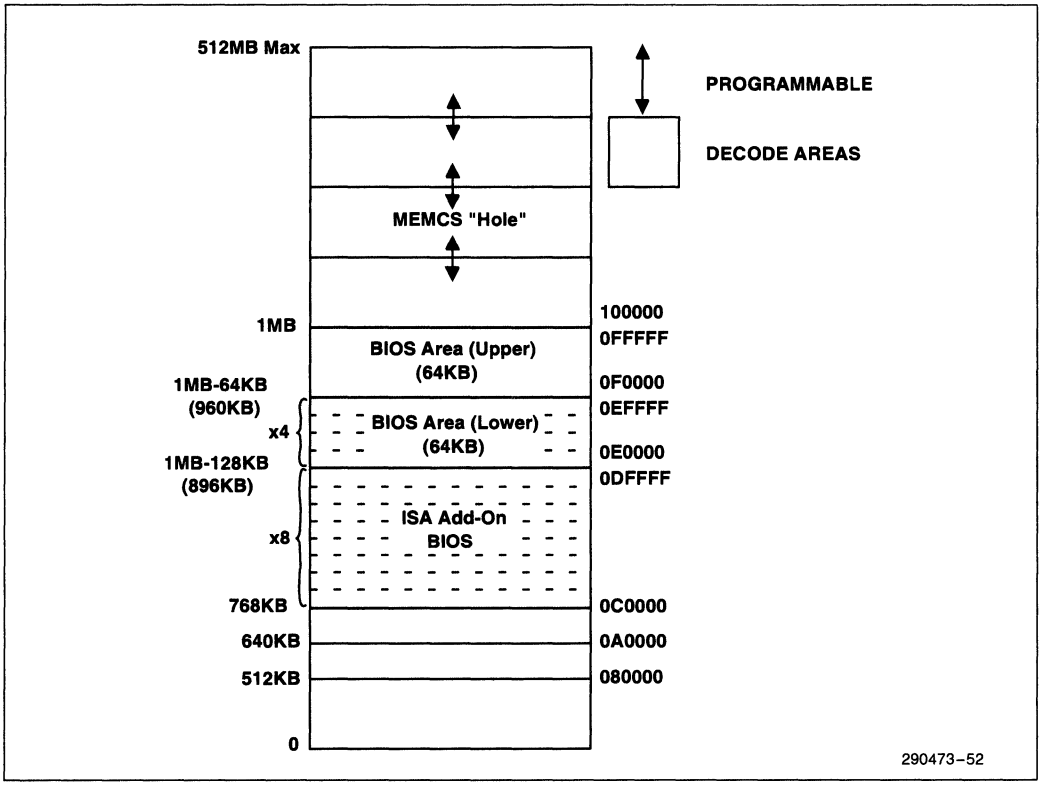


Figure 13. MEMCS# Decode Areas

The SIO generates MEMCS# from the PCI address. MEMCS# is generated from the clock edge after FRAME# is sampled active. MEMCS# will only go active for one PCI clock period. The SIO does not take any other action as a result of this decode other than generating MEMCS#. It is the responsibility of the device using the MEMCS# signal to generate DEVSEL#, TRDY# and any other cycle response. The device using MEMCS# will always generate DEVSEL# on the next clock. This fact can be used to avoid an extra clock delay in the subtractive decoder described in the next section.

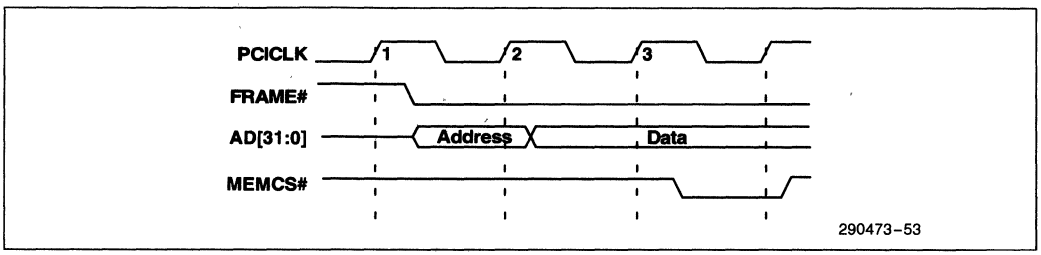


Figure 14. MEMCS# Generation

5.5.1.4 Subtractively Decoded Cycles to ISA

The addresses that reside on the ISA Bus could be highly fragmented. For this reason, subtractive decoding is used to forward PCI cycles to the ISA Bus. An inactive DEVSEL# will cause the SIO to forward the PCI cycle to the ISA Bus. The DEVSEL# sample point can be configured for three different settings. If the "fast" point is selected, the cycle will be forwarded to ISA when DEVSEL# is inactive at the F sample point as shown in Figure 15. If the "typical" point is selected, DEVSEL# will be sampled on both F and T, and if inactive, will be forwarded to the ISA Bus. Likewise, if the "slow" point is selected, DEVSEL# will be sampled at F, T, and S. The sample point should be configured to match the slowest PCI device in the system. This capability reduces the latency to ISA slaves when all PCI devices are "fast" and also allows for devices with slow decoding. Note that when these unclaimed cycles are forwarded to the ISA Bus, the SIO will drive the DEVSEL# active.

Since an active MEMCS# will always result in an active DEVSEL# at the "Slow" sample point, MEMCS# is used as an early indication of DEVSEL#. In this case, if the device using MEMCS# is the only "slow" agent in the system, the sample point can be moved in to the "typical" edge.

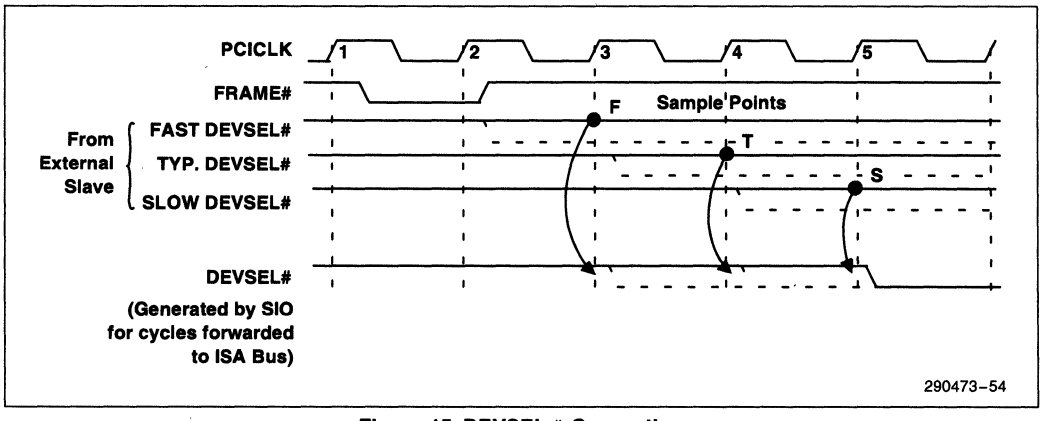


Figure 15. DEVSEL# Generation

Unclaimed PCI cycles with memory addresses above 16M and I/O addresses above 64K will not be forwarded to the ISA Bus. The SIO will not respond with DEVSEL# (BIOS accesses are an exception to this). This is required to avoid the possibility of aliasing. Under this condition, these unclaimed cycles will be recognized as such by the originating master and the master will use "master-abort" semantics to terminate the PCI cycle.

5.5.2 DMA/ISA MASTER CYCLE ADDRESS DECODER

The SIO also contains a decoder which is used to determine the destination of ISA master and DMA master cycles. This decoder provides:

Positive Decode to PCI: Positively decodes addresses to be forwarded to the PCI Bus. This includes addresses residing directly on PCI as well as addresses that reside on the back side of PCI bridges (Host Bridges).

Access to SIO Internal Registers: Positively decodes addresses to registers within the SIO.

BIOS Accesses: Positively decodes BIOS memory accesses and generates encoded BIOSCS#.

Utility Bus Chip Selects: Positively decodes utility bus chip selects.

Subtractive Decode: Subtractively decodes cycles to be contained to the ISA Bus.

5.5.2.1 Positive Decode to PCI

ISA master or DMA addresses that are positively decoded by this decoder will be propagated to the PCI Bus. This is the only way to forward a cycle from an ISA master or the DMA to the PCI Bus. If the cycle is not decoded by this decoder it will *not* be forwarded to the PCI Bus.

This decoder has several memory address regions to positively decode cycles that should be forwarded to the PCI Bus. These regions are listed below. Regions "a" through "e" are fixed and can be enabled or disabled independently. Region "f" defines a space starting at 1M with a programmable upper boundary up to 16 MB. Within this region a hole can be opened. Its size and location are programmable to allow a hole to be opened in the memory space. A memory address above 16 MB will be forwarded to the PCI Bus automatically. This is possible only during DMA cycles in which the DMA has been programmed for 32-bit addressing above 16 MB.

a. Memory: 0 KB–512 KB

b. Memory: 512 KB–640 KB

c. Memory: 640 KB–768 KB (Video buffer)

d. Memory: 768 KB–896 KB in eight 16K sections (Expansion ROM)

e. Memory: 896 KB–960 KB (lower BIOS area)

f. Memory: 1 MB-to-X MB (up to 16 MB) within which a hole can be opened. Accesses to the hole are not forwarded to PCI. The top of the region can be programmed on 64 KByte boundaries up to 16 MB. The hole can be between 64 KB and 8 MB in size in 64 KB increments located on any 64 KB boundary. (Refer to the ISA Address Decoder Register in the register description section, Section 5.5.2)

g. Memory: > 16 MB automatically forwarded to PCI

Figure 16 shows a map of the ISA master/DMA decode regions and Table 22 summarizes the registers used to configure the decoder.

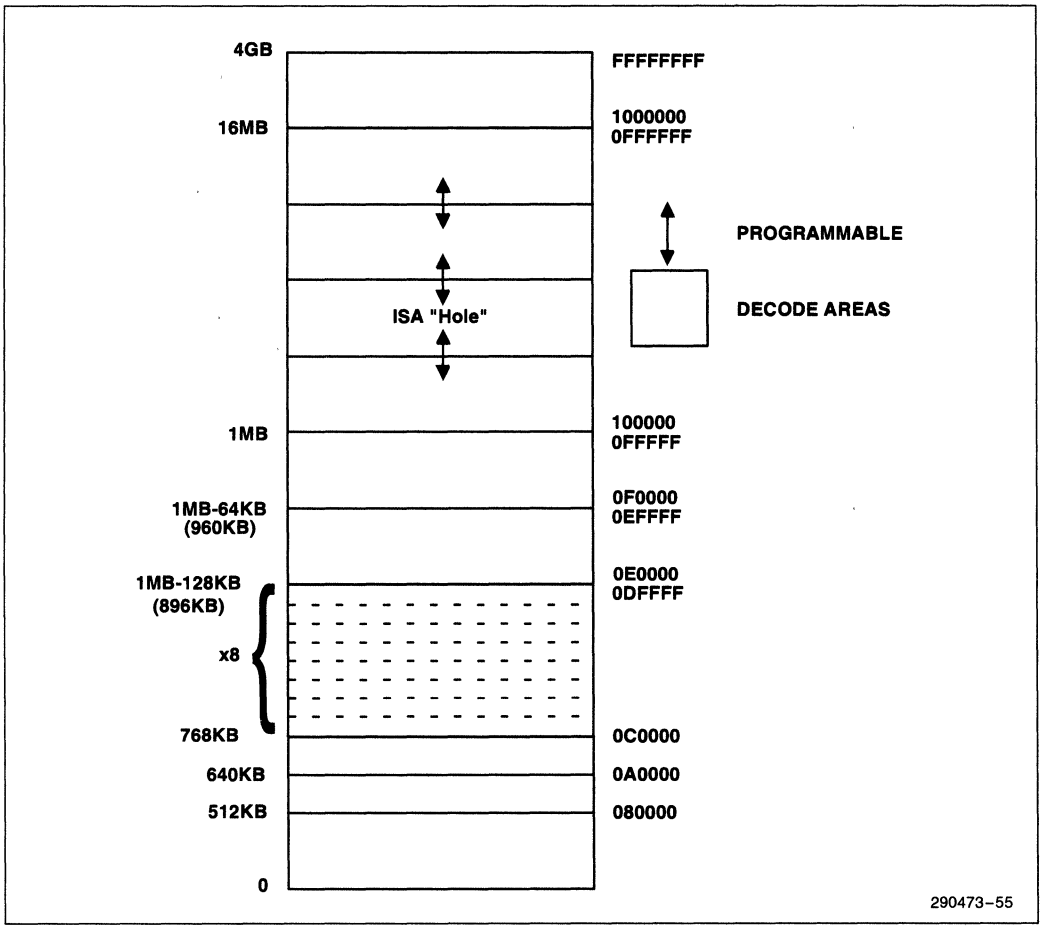


Figure 16. ISA Master/DMA to PCI Bus Decoder Regions

Table 22. ISA Master/DMA to PCI Bus Decoding Register Summary

MAR Registers	Attribute	Memory Segments	Comments
IADCON[7:4]	ISA Memory Top	100000h–0FFFFFFh	1 MB to 16 MB Top of ISA Region
IADTOH/IADBOH	ISA Hole	100000h–0FFFFFFh	1 MB to 16 MB Hole in ISA Region
IADCON[0]	Enable/Disable	000000h–07FFFFh	0 to 512 KB Enable/Disable
IADCON[1]	Enable/Disable	080000h–09FFFFh	512 KB to 640 KB Enable/Disable
IADCON[2]	Enable/Disable	0A0000h–0BFFFFh	640 KB to 768 KB Enable/Disable
IADCON[3]*	Enable/Disable	0E0000h–0EFFFFh	896 KB to 960 KB Lower BIOS Enable/Disable
IADRBE[0]	Enable/Disable	0C0000h–0C3FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[1]	Enable/Disable	0C4000h–0C7FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[2]	Enable/Disable	0C8000h–0CBFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[3]	Enable/Disable	0CC000h–0CFFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[4]	Enable/Disable	0D0000h–0D3FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[5]	Enable/Disable	0D4000h–0D7FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[6]	Enable/Disable	0D8000h–0DBFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[7]	Enable/Disable	0DC000h–0DFFFFh	ISA Add-On BIOS (Expansion ROM) Enable

NOTE:
* This can be overridden by bit 6 of the UBCSA Register being set to a 1.

5.5.2.2 SIO Internal Registers

Most of the internal SIO registers are accessible by ISA masters. Table 19 lists the registers that are not accessible by ISA masters. Registers accessed by ISA masters are run as 8-bit extended I/O cycles.

5.5.2.3 BIOS Accesses

The 128K BIOS memory space is located at 000E0000h to 000FFFFFh, and is aliased at FFFE0000h to FFFFFFFFh (top of 4 GB) and FFEE0000h to FFEFFFFFh (top of 4 GB–1 MB). The aliased regions account for the CPU reset vector and the uncertainty of the state of A20GATE when a software reset occurs. This 128K block is split into two 64K blocks. The top 64K is always enabled while the bottom 64K can be enabled or disabled (the aliases automatically match). ISA masters can only access BIOS in the 000E0000 to 000FFFFFh region.

ISA originated accesses to the enabled 64K sections of the BIOS space (000E0000h–000FFFFFh) will activate the encoded BIOSCS# signal. ISA originated cycles will not be forwarded to the PCI Bus. Encoded BIOSCS# is combinatorially generated from the ISA, SA, and LA address bus. Encoded BIOSCS# is disabled during refresh and DMA cycles. The ISA Master/DMA BIOS Decoding Table indicates the SIO's response to BIOS accesses based on the configuration state.

Table 23. ISA Master/DMA BIOS Decoding

Cycle		SIO Configuration			SIO Response		
Master	Region(1)	Top 64 KB PCI Positive Decode Enabled(2)	Low 64 KB BIOS Enabled(3)	Forward Low 64 KB to PCI Enabled(4)	Encoded BIOSCS# Generated	Forward to PCI	Contain to ISA
ISA/DMA	A	x	x	x	Yes	No	Yes
ISA/DMA	B	x	0(5)	0	No	No	Yes
ISA/DMA	B	x	0(5)	1	No	Yes	No
ISA/DMA	B	x	1	x	Yes	No	Yes
ISA/DMA	a	These cycles will be forwarded to PCI dependent on the state of the ISA Address Decoder Configuration Registers. Encoded BIOSCS# will not be generated for any of these cycles.					
ISA/DMA	b						
ISA/DMA	c						

NOTES:
1. The memory sections referenced can be found in Figure 12.
2. The column labeled "Top 64 KB BIOS Positive Decode Enabled" shows the value of the ISA Clock Divisor Configuration Register bit 6. This bit determines how the memory region is decoded (0 = subtractively decoded, 1 = positively decoded).
3. The column labeled "Low 64 KB BIOS Enable" shows the value of the Utility Bus Chip Select Enable A Configuration Register bit 6. This bit determines if the memory region is enabled (bit = 1) or disabled (bit = 0).
4. The column labeled "Forward Low 64 KB to PCI Enables" shows the value of the ISA Address Decoder Control Configuration Register Bit 3. This bit determines whether PCI Bus forwarding is enabled (bit = 1) or disabled (bit = 0).
5. Forward to PCI if IADCON Bit 6 = 1.

5.5.2.4 Utility Bus Encoded Chip Selects

The SIO generates encoded chip selects for certain functions that are located on the utility bus (formerly X-Bus). The encoded chip selects are generated combinatorially from the ISA SA[15:0] address bus. The encoded chip selects are decoded externally (see Figure 19).

The encoded chip select table (Table 24) shows the addresses that result in encoded chip select generation. Chip selects can be enabled or disabled via configuration registers. In general, the addresses shown in Table 24 do not reside in the SIO itself. Write only addresses 70h, 372h, 3F2h are exceptions since particular bits from these registers reside in the SIO. For ISA master cycles, the SIO will respond to writes to address 70h, 372h, and 3F2h by generating IOCHRDY and writing to the appropriate bits.

Note that the SIO monitors read accesses to address 60h to support the mouse function. In this case, IOCHRDY is not generated.

Table 24. Encoded Chip Select Table

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0060h	0000	0000	0110	00x0	r/w	Keyboard Controller	KEYBRDCS#
0064h	0000	0000	0110	01x0	r/w	Keyboard Controller	KEYBRDCS#
0070h	0000	0000	0111	0xx0	w	Real Time Clock Address	RTCALE#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0071h	0000	0000	0111	0xx1	r/w	Real Time Clock Data	RTCCS#
0170h	0000	0001	0111	0000	r/w	Secondary Data Register	IDECS0#
0171h	0000	0001	0111	0001	r/w	Secondary Error Register	IDECS0#
0172h	0000	0001	0111	0010	r/w	Secondary Sector Count Register	IDECS0#
0173h	0000	0001	0111	0011	r/w	Secondary Sector Number Register	IDECS0#
0174h	0000	0001	0111	0100	r/w	Secondary Cylinder Low Register	IDECS0#
0175h	0000	0001	0111	0101	r/w	Secondary Cylinder High Register	IDECS0#
0176h	0000	0001	0111	0110	r/w	Secondary Drive/Head Register	IDECS0#
0177h	0000	0001	0111	0111	r/w	Secondary Status Register	IDECS0#
01F0h	0000	0001	1111	0000	r/w	Primary Data Register	IDECS0#
01F1h	0000	0001	1111	0001	r/w	Primary Error Register	IDECS0#
01F2h	0000	0001	1111	0010	r/w	Primary Sector Count Register	IDECS0#
01F3h	0000	0001	1111	0011	r/w	Primary Sector Number Register	IDECS0#
01F4h	0000	0001	1111	0100	r/w	Primary Cylinder Low Register	IDECS0#
01F5h	0000	0001	1111	0101	r/w	Primary Cylinder High Register	IDECS0#
01F6h	0000	0001	1111	0110	r/w	Primary Drive/Head Register	IDECS0#
01F7h	0000	0001	1111	0111	r/w	Primary Status Register	IDECS0#
0278h	0000	0010	0111	1x00	r/w	LPT3 PP Data Latch	LPTCS#
0279h	0000	0010	0111	1x01	r	LPT3 PP Status	LPTCS#
027Ah	0000	0010	0111	1x10	r/w	LPT3 PP Control	LPTCS#
027Bh	0000	0010	0111	1x11	r/w		LPTCS#
02F8h	0000	0010	1111	1000	r/w	COM2 SP Transmit/Receive Register	COM2CS#
02F9h	0000	0010	1111	1001	r/w	COM2 SP Interrupt Enable Register	COM2CS#
02FAh	0000	0010	1111	1010	r	COM2 SP Interrupt Identification Register	COM2CS#
02FBh	0000	0010	1111	1011	r/w	COM2 SP Line Control Register	COM2CS#
02FCh	0000	0010	1111	1100	r/w	COM2 SP Modem Control Register	COM2CS#
02FDh	0000	0010	1111	1101	r	COM2 SP Line Status Register	COM2CS#
02FEh	0000	0010	1111	1110	r	COM2 SP Modem Status Register	COM2CS#
02FFh	0000	0010	1111	1111	r/w	COM2 SP Scratch Register	COM2CS#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0370h	0000	0011	0111	0000	r/w	Secondary Floppy Disk Extended Mode Register	FLOPPYCS#
0371h	0000	0011	0111	0001	r/w	Secondary Floppy Disk Extended Mode Register	FLOPPYCS#
0372	0000	0011	0111	0010	w	Secondary Floppy Disk Digital Output Register	FLOPPYCS#
0373h	0000	0011	0111	0011	r/w	Reserved	FLOPPYCS#
0374h	0000	0011	0111	0100	r/w	Secondary Floppy Disk Status Register	FLOPPYCS#
0375h	0000	0011	0111	0101	r/w	Secondary Floppy Disk Data Register	FLOPPYCS#
0376h	0000	0011	0111	0110	r/w	Secondary Alternate Status Register	IDECS1#
0377h	0000	0011	0111	0111	r	Secondary Drive Address Register	IDECS1#
0377h*	0000	0011	0111	011x	r/w	Secondary Floppy Disk Digital Input Register	FLOPPYCS#
0378h	0000	0011	0111	1x00	r/w	LPT2 PP Data Latch	LPTCS#
0379h	0000	0011	0111	1x01	r	LPT2 PP Status	LPTCS#
037Ah	0000	0011	0111	1x10	r/w	LPT2 PP Control	LPTCS#
037Bh	0000	0011	0111	1x11	r/w		LPTCS#
03BCh	0000	0011	1011	1100	r/w	LPT1 PP Data Latch	LPTCS#
03BDh	0000	0011	1011	1101	r	LPT1 PP Status	LPTCS#
03BEh	0000	0011	1011	1110	r/w	LPT1 PP Control	LPTCS#
03BFh	0000	0011	1011	1111	r/w		LPTCS#
03F0h	0000	0011	1111	0000	r/w	Primary Floppy Disk Extended Mode Register	FLOPPYCS#
03F1h	0000	0011	1111	0001	r/w	Primary Floppy Disk Extended Mode Register	FLOPPYCS#
03F2h	0000	0011	1111	0010	w	Primary Floppy Disk Digital Output Register	FLOPPYCS#
03F3h	0000	0011	1111	0011	r/w	Reserved	FLOPPYCS#
03F4h	0000	0011	1111	0100	r/w	Primary Floppy Disk Status Register	FLOPPYCS#
03F5h	0000	0011	1111	0101	r/w	Primary Floppy Disk Data Register	FLOPPYCS#
03F6h	0000	0011	1111	0110	r/w	Primary Drive Alternate Status Register	IDECS1#
03F7h	0000	0011	1111	0111	r	Primary Drive Address Register	IDECS1#
03F7h*	0000	0011	1111	011x	r/w	Primary Floppy Disk Digital Input Register	FLOPPYCS#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
03F8h	0000	0011	1111	1000	r/w	COM1 SP Transmit/Receive Register	COM1CS#
03F9h	0000	0011	1111	1001	r/w	COM1 SP Interrupt Enable Register	COM1CS#
03FAh	0000	0011	1111	1010	r	COM1 SP Interrupt Identification Register	COM1CS#
03FBh	0000	0011	1111	1011	r/w	COM1 SP Line Control Register	COM1CS#
03FCh	0000	0011	1111	1100	r/w	COM1 SP Modem Control Register	COM1CS#
03FDh	0000	0011	1111	1101	r	COM1 SP Line Status Register	COM1CS#
03FEh	0000	0011	1111	1110	r	COM1 SP Modem Status Register	COM1CS#
03FFh	0000	0011	1111	1111	r/w	COM1 SP Scratch Register	COM1CS#
0800h–08FFh	0000	1000	xxxx	xxxx	r/w		CFIGMEMCS#
0C00h	0000	1100	0000	0000	r/w		CPAGECS#

NOTE:
*If both the IDE and Floppy Drive are located on the UBUS, FLOPPYCS# will not be generated, IDECS1# will be generated.

5.5.2.5 Subtractive Decode to ISA

ISA master and DMA cycles not positively decoded by the ISA decoder are contained to the ISA Bus.

5.6 Data Buffering

The SIO contains data buffers to isolate the PCI Bus from the ISA Bus. The buffering is described from two perspectives: PCI master accesses to the ISA Bus (Posted Write Buffer) and DMA/ISA master accesses to the PCI Bus (Line Buffer). Temporarily buffering the data requires buffer management logic to ensure that the data buffers remain coherent.

5.6.1 DMA/ISA MASTER LINE BUFFER

An 8-byte Line Buffer is used to isolate the ISA Bus's slower I/O devices from the PCI Bus. The Line Buffer is bi-directional and is used by ISA masters and the DMA controller to assemble and disassemble data. Only memory data written to or read from the PCI Bus by an ISA master or DMA is assembled/disassembled using this 8 byte line buffer. I/O cycles do not use the buffer.

Bits 0 and 1 of the PCI Control Register set the buffer to operate in either single transaction mode (bit = 0) or 8-byte mode (bit = 1). Note that ISA masters and DMA controllers can have their buffer modes configured separately.

In single transaction mode, the buffer will store only one transaction. For DMA/ISA master writes, this single transaction buffer looks like a posted write buffer. As soon as the ISA cycle is complete, a PCI cycle is scheduled. Subsequent DMA/ISA master writes are held off in wait-states until the buffer is empty. For DMA/ISA master reads, only the data requested is read over the PCI Bus. For instance, if the DMA channel is programmed in 16-bit mode, 16 bits of data will be read from PCI. As soon as the requested data is valid on the PCI bus, it is latched into the Line Buffer and the ISA cycle is then completed, as timing allows. Single transaction mode will guarantee strong read and write ordering through the buffers.

In 8 byte mode, for write data assembly, the Line Buffer acts as two individual 4 byte buffers working in ping pong fashion. For read data disassembly, the Line Buffer acts as one 8 byte buffer.

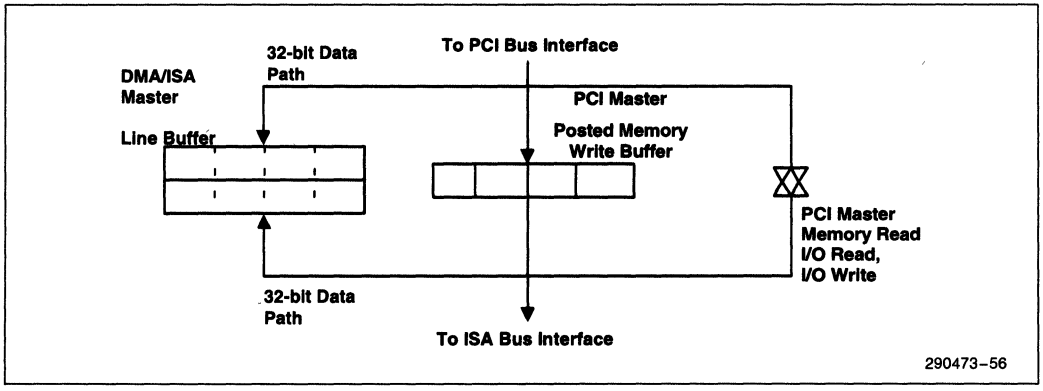


Figure 17. SIO Buffer Diagram

5.6.2 PCI MASTER POSTED WRITE BUFFER

PCI master memory write cycles destined to ISA memory are buffered in a 32-bit Posted Write Buffer. The PCI Memory Write and Memory Write and Invalidate commands are all treated as a memory write and can be posted, subject to the Posted Write Buffer status. The Posted Write Buffer has an address associated with it. A PCI master memory write can be posted any time the posted write buffer is empty and write posting is enabled (bit 2 of the PCI Control Configuration Register is set to a 1). Also, the ISA Bus must not be occupied. If the posted write buffer contains data, the PCI master write cycle is retried. If the posted write buffer is disabled, the SIO's response to a PCI master memory write is dependent on the state of the ISA Bus. If the ISA Bus is available and the posted write buffer is disabled, the cycle will immediately be forwarded to the ISA Bus (TRDY# will not be asserted until the ISA cycle has completed). If the ISA Bus is busy and the posted write buffer is disabled, the cycle is retried.

Memory read and I/O read and I/O write cycles do not use the 32-bit Posted Write Buffer.

5.6.3 BUFFER MANAGEMENT

Any time data is temporarily stored in the buffers between the ISA Bus and the PCI Bus, there are potential data coherency problems.

The SIO contains buffer management circuitry which guarantees data coherency by intercepting synchronization protocol between the buses and managing the buffers before synchronization communication between the buses is complete. The buffers are flushed or invalidated as appropriate before a bus cycle is allowed to occur in cases where data coherency could be lost.

5.6.3.1 DMA/ISA Master Line Buffer—Write State

When the DMA/ISA Master Line Buffer contains data that is to be written to the PCI Bus, it is in the Write State. The 8-byte line buffer is flushed when the line becomes full, when a subsequent write is a line miss, when a subsequent write would overwrite an already valid byte, or when a subsequent cycle is a read. The ISA master or DMA cycle that triggers the buffer flush will be held in wait-states until the flush is complete. The buffer is also flushed whenever there is a change in ISA Bus ownership as indicated by any DACK# signal going inactive.

Once the buffer is scheduled to be flushed to PCI, any PCI cycle to the SIO or ISA Bus will get retried by the SIO.

5.6.3.2 DMA/ISA Master Line Buffer—Read State

When the DMA/ISA Master Line Buffer contains data that has been read from the PCI Bus, it is in the Read State. The data in the buffer will be invalidated when the SIO accepts a PCI memory or I/O write cycle. The line buffer in the read state is also invalidated when a subsequent read is a line miss, or when a subsequent cycle is a write. The line buffer in the read state is not invalidated on a change of ISA ownership. Note that as bytes are disassembled from the line buffer, they are invalidated so that subsequent reads to the same byte will cause a line buffer miss.

5.6.3.3 PCI Master Posted Write Buffer

As soon as a PCI master has posted a memory write into the posted write buffer, the buffer is scheduled to be written to the ISA Bus. Any subsequent PCI master cycles to the SIO (including ISA Bus) will be retried until the posted write buffer is empty.

Prior to granting the ISA Bus to an ISA master or the DMA, the PCI master posted memory write buffer is flushed. Also, as long as the ISA master or DMA owns the ISA Bus, the posted write buffer is disabled. A PCI master write can not be posted while an ISA master or the DMA owns the ISA Bus.

5.7 SIO Timers

5.7.1 INTERVAL TIMERS

The SIO contains three counters that are equivalent to those found in the 82C54 programmable interval timer. The three counters are contained in one SIO timer unit, referred to as Timer-1. Each counter output provides a key system function. Counter 0 is connected to interrupt controller IRQ0 and provides a system timer interrupt for a time-of-day, diskette time-out, or other system timing functions. Counter 1 generates a refresh request signal and Counter 2 generates the tone for the speaker. Note that the 14.31818 MHz counters use OSC for a clock source.

Full details of this counter can be found in the 82C54 data sheet.

2

Table 25. Interval Timer Functions Table

Interval Timer Functions	
Function	**Counter 0—System Timer**
Gate	Always On
Clock In	1.193 MHz (OSC/12)
Out	INT-1 IRQ0
Function	**Counter 1—Refresh Request**
Gate	Always On
Clock In	1.193 MHz (OSC/12)
Out	Refresh Request
Function	**Counter 2—Speaker Tone**
Gate	Programmable-Port 61h
Clock In	1.193 MHz (OSC/12)
Out	Speaker

5.7.1.1 Interval Timer Address Map

Table 26 shows the I/O address map of the interval timer counters.

Table 26. Interval Timer Counters I/O Address Map

I/O Address	Register Description
040h	System Timer (Counter 0)
041h	Refresh Request (Counter 1)
042h	Speaker Tone (Counter 2)
043h	Control Word Register

Counter 0, System Timer

This counter functions as the system timer by controlling the state of IRQ0 and is typically programmed for Mode 3 operation. The counter produces a square wave with a period equal to the product of the counter period (838 ns) and the initial count value. The counter loads the initial count value one counter period after software writes the count value to the counter I/O address. The counter initially asserts IRQ0 and decrements the count value by two each counter period. The counter negates IRQ0 when the count value reaches 0. It then reloads the initial count value and again decrements the initial count value by two each counter period. The counter then asserts IRQ0 when the count value reaches 0, reloads the initial count value, and repeats the cycle, alternately asserting and negating IRQ0.

Counter 1, Refresh Request Signal

This counter provides the refresh request signal and is typically programmed for Mode 2 operation. The counter negates refresh request for one counter period (833 ns) during each count cycle. The initial count value is loaded one counter period after being written to the counter I/O address. The counter initially asserts refresh request, and negates it for 1 counter period when the count value reaches 1. The counter then asserts refresh request and continues counting from the initial count value.

Counter 2, Speaker Tone

This counter provides the speaker tone and is typically programmed for Mode 3 operation. The counter provides a speaker frequency equal to the counter clock frequency (1.193 MHz) divided by the initial count value. The speaker must be enabled by a write to port 061h (see Section 4.5.1 on the NMI Status and Control Register).

5.7.2 BIOS TIMER

5.7.2.1 Overview

The SIO provides a system BIOS Timer that decrements at each edge of its 1.04 MHz clock (derived by dividing the 8.33 MHz SYSCLK by 8). Since the state of the counter is undefined at power-up, it must be programmed before it can be used. Accesses to the BIOS Timer are enabled and disabled through the BIOS Timer Base Address Register. The timer continues to count even if accesses are disabled.

A BIOS Timer Register is provided to start the timer counter by writing an initial clock value. The BIOS Timer Register can be accessed as a single 16-bit I/O port or as a 32-bit port with the upper 16-bits being "don't care" (reserved). It is up to the software to access the I/O register in the most convenient way. The I/O address of the BIOS Timer Register is software relocatable. The I/O address is determined by the value programmed into the BIOS Timer Base Address Register.

The BIOS Timer clock has a value of 1.04 MHz using an 8.33 MHz SYSCLK input (an 8 to 1 ratio will always exist between SYSCLK and the timer clock). This allows the counting of time intervals from 0 ms to approximately 65 ms. Because of the PCI clock rate, it is possible to start the counter and read the value back in less than 1 μs. The expected value of the expired interval is 0, but depending on the state of the internal clock divisor, the BIOS Timer might indicate that 1 ms has expired. Therefore, accuracy of the counter is $\pm$ 1 μs.

5.7.2.2 BIOS Timer Operations

A write operation to the BIOS Timer Register will initiate the counting sequence. The timer can be initiated by writing either the 16-bit data portion or the whole 32-bit register (upper 16 bits are "don't care"). After initialization, the BIOS timer will start decrementing until it reaches zero. Then it will stop decrementing (and hold a zero value) until initialized again.

After the timer is initialized, the current value can be read at any time and the timer can be reprogrammed (new initial value written), even before it reaches zero.

All write and read operations to the BIOS timer Register should include all 16 counter bits. Separate accesses to the individual bytes of the counter must be avoided since this can cause unexpected results (wrong count intervals).

5.8 Interrupt Controller

The SIO provides an ISA compatible interrupt controller which incorporates the functionality of two 82C59 interrupt controllers. The two controllers are cascaded so that 14 external and two internal interrupts are possible. The master interrupt controller provides IRQ[7:0] and the slave interrupt controller provides IRQ [15:8] (see Figure 18). The two internal interrupts are used for internal functions only and are not available to the user. IRQ2 is used to cascade the two controllers together and IRQ0 is used as a system timer interrupt and is tied to Interval Timer 1, Counter 0. The remaining 14 interrupt lines (IRQ1, IRQ3–IRQ15) are available for external system interrupts. Edge or level sense selection is programmable on a by-controller basis.

The Interrupt Controller consists of two separate 82C59 cores. Interrupt Controller 1 (CNTRL-1) and Interrupt Controller 2 (CNTRL-2) are initialized separately and can be programmed to operate in different modes. The default settings are: 80x86 Mode, Edge Sensitive (IRQ0-15) Detection, Normal EOI, Non-Buffered Mode, Special Fully Nested Mode disabled, and Cascade Mode. CNTRL-1 is connected as the Master Interrupt Controller and CNTRL-2 is connected as the Slave Interrupt Controller.

Note that IRQ13 is generated internally (as part of the coprocessor error support) by the SIO when bit 5 in the ISA Clock Divisor Register is set to a 1. When this bit is set to a 0, then the FERR#/IRQ13 signal is used as an external IRQ13 signal and has the same functionality as the normal IRQ13 signal. IRQ12/M is generated internally (as part of the mouse support) by the SIO when bit 4 in the ISA Clock Divisor Register is set to a 1. When set to a 0, the standard IRQ12 function is provided.

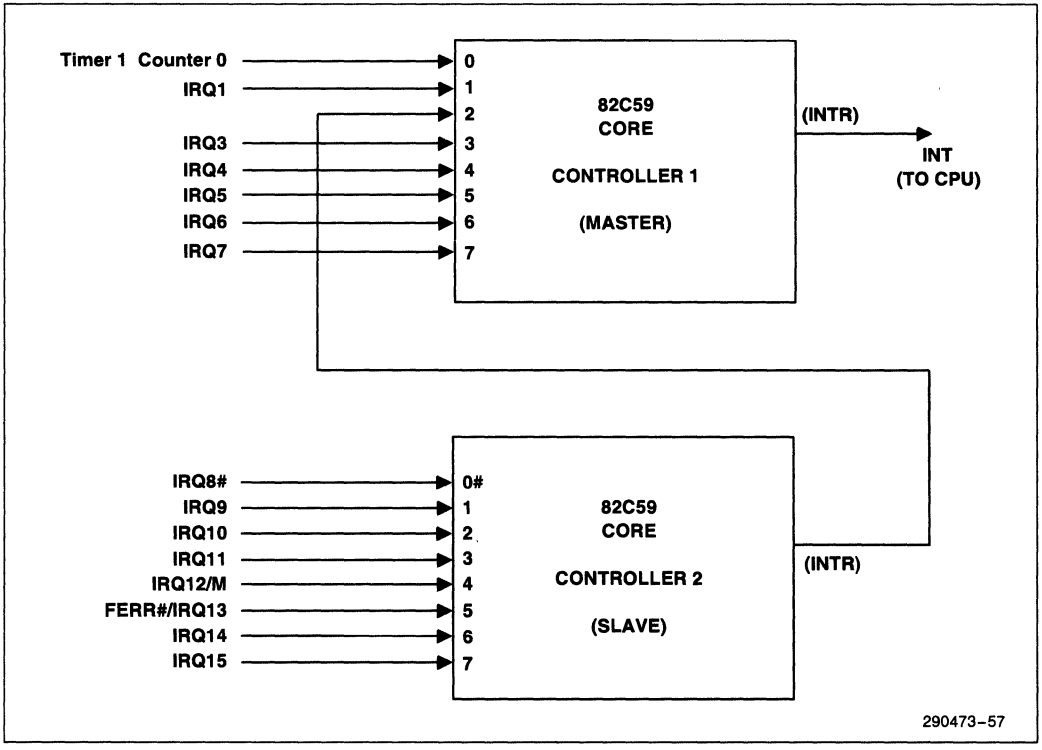


Figure 18. Block Diagram of the Interrupt Controller

Table 27 lists the I/O port address map for the interrupt registers:

Table 27. Interrupt Registers I/O Port Address Map

Interrupts	I/O Address	# of Bits	Register
IRQ[7:0]	0020h	8	CNTRL-1 Control Register
IRQ[7:0]	0021h	8	CNTRL-1 Mask Register
IRQ[15:8]	00A0h	8	CNTRL-2 Control Register
IRQ[15:8]	00A1h	8	CNTRL-2 Mask Register

IRQ0, IRQ2, (and possibly IRQ13 and IRQ12 if the "mouse" or floating point error logic is disabled in the ISA Clock Divisor Register), are connected to the interrupt controllers internally. The other interrupts are always generated internally and their typical functions are shown in Table 28:

Table 28. Typical Interrupt Functions

Priority	Label	Controller	Typical Interrupt Source
1	IRQ0	1	Interval timer 1, Counter 0 OUT
2	IRQ1	1	Keyboard
3–10	IRQ2	1	Interrupt from Controller 2
3	IRQ8#	2	Real Time Clock
4	IRQ9	2	Expansion Bus Pin B04
5	IRQ10	2	Expansion Bus Pin D03
6	IRQ11	2	Expansion Bus Pin D04
7	IRQ12/M	2	Mouse Interrupt
8	FERR#/IRQ13	2	Coprocessor Error
9	IRQ14	2	Fixed Disk Drive Controller Expansion Bus Pin D07
10	IRQ15	2	Expansion Bus Pin D06
11	IRQ3	1	Serial Port 2, Expansion Bus B25
12	IRQ4	1	Serial Port 1, Expansion Bus B24
13	IRQ5	1	Parallel Port 2, Expansion Bus B23
14	IRQ6	1	Diskette Controller, Expansion Bus B22
15	IRQ7	1	Parallel Port 1, Expansion Bus B21

5.8.1 EDGE AND LEVEL TRIGGERED MODES

There are two ELCR registers, one for each 82C59 bank. They are located at I/O ports 04D0h (for the Master Bank, IRQ[0:1,3:7]#) and 04D1h (for the Slave Bank, IRQ[8:15])#. They allow the edge and level sense selection to be made on an interrupt by interrupt basis instead of on a complete bank. Interrupts reserved for ISA use MUST be programmed for edge sensitivity (to ensure ISA compatibility). That is, IRQ (0,1,2,8#,13) must be programmed for edge sensitive operation. The LTIM bit (Edge/Level Bank select, offsets 20h, A0h) is disabled in the SIO. The default programming is equivalent to programming the LTIM bit (ICW1 bit 3) to a 0.

If an ELCR bit is equal to "0", an interrupt request will be recognized by a low to high transition on the corresponding IRQ input. The IRQ input can remain high without generating another interrupt.

If an ELCR bit is equal to "1", an interrupt request will be recognized by a "low" level on the corresponding IRQ input, and there is no need for an edge detection. For level triggered interrupt mode, the interrupt request signal must be removed before the EOI command is issued or the CPU interrupt must be disabled. This is necessary to prevent a second interrupt from occurring.

In both the edge and level triggered modes the IRQ inputs must remain active until after the falling edge of the first INTA#. If the IRQ input goes inactive before this time a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt. This can be a useful safeguard for detecting interrupts caused by spurious noise glitches on the IRQ inputs. To implement this feature the IRQ7 routine is used for "clean up" simply executing a return instruction, thus ignoring the interrupt. If IRQ7 is needed for other purposes a default IRQ7 can still be detected by reading the ISR. A normal IRQ7 interrupt will set the corresponding ISR bit, a default IRQ7 won't. If a default IRQ7 routine occurs during a normal IRQ7 routine, however, the ISR will remain set. In this case it is necessary to keep track of whether or not the IRQ7 routine was previously entered. If another IRQ7 occurs it is a default.

5.8.2 REGISTER FUNCTIONALITY

For a detailed description of the Interrupt Controller register set, please see Section 4.4, Interrupt Controller Register Description.

5.8.3 NON-MASKABLE INTERRUPT (NMI)

An NMI is an interrupt requiring immediate attention and has priority over the normal interrupt lines (IRQx). The SIO indicates error conditions by generating a non-maskable interrupt.

NMI interrupts are caused by the following conditions:

1. System Errors on the PCI Bus. SERR# is driven low by a PCI resource when this error occurs.
2. Parity errors on the add-in memory boards on the ISA expansion bus. IOCHK# is driven low when this error occurs.

The NMI logic incorporates two different 8-bit registers. These registers are addressed at locations 061h and 070h. The status of Port (061h) is read by the CPU to determine which source caused the NMI. Bits set to 1 in these ports show which device requested an NMI interrupt. After the NMI interrupt routine processes the interrupt, the NMI status bits are cleared by the software. This is done by setting the corresponding enable/disable bit high. Port (070H) is the mask register for the NMI interrupts. This register can mask the NMI signal and also disable or enable all NMI sources.

The individual enable/disable bits clear the NMI detect flip-flops when disabled.

All NMI sources can be enabled or disabled by setting Port 070h bit 7 to a 0 or 1. This disable function does not clear the NMI detect flip-flops. This means, if NMI is disabled then enabled via Port 070h, then an NMI will occur when Port 070h is re-enabled if one of the NMI detect flip-flops had been previously set.

To ensure that all NMI requests are serviced, the NMI service routine software needs to incorporate a few very specific requirements. These requirements are due to the edge detect circuitry of the host microprocessor, 80386 or 80486. The software flow would need to be the following:

1. NMI is detected by the processor on the rising edge of the NMI input.
2. The processor will read the status stored in port 061h to determine what sources caused the NMI. The processor may then set to 0 the register bits controlling the sources that it has determined to be active. Between the time the processor reads

the NMI sources and sets them to a 0, an NMI may have been generated by another source. The level of NMI will then remain active. This new NMI source will not be recognized by the processor because there was no edge on NMI.

3. The processor must then disable all NMI's by setting bit 7 of port 070H to a 1 and then enable all NMI's by setting bit 7 of port 070H to a 0. This will cause the NMI output to transition low then high if there are any pending NMI sources. The CPU's NMI input logic will then register a new NMI.

Section 4.5 Control Registers, contains a detailed description of the NMI Status and Control Register (port 061h) and the NMI Enable and Real-Time Clock Address Register at port 070h.

5.9 Utility Bus Peripheral Support

The Utility Bus is a secondary bus buffered from the ISA Bus used to interface with peripheral devices that do not require a high speed interface. The buffer control for the lower 8 data signals is provided by the SIO via two control signals; UBUSOE# and UBUSTR. Figure 19 shows a block diagram of the external logic required as part of the decode and Utility Bus buffer control.

The SIO provides the address decode and three encoded chip selects to support:

1. Floppy Controller
2. Keyboard Controller
3. Real Time Clock
4. IDE Drive
5. 2 Serial Ports (COM1 and COM2)
6. 1 Parallel Port (LPT1, 2, or 3)
7. BIOS Memory
8. Configuration Memory (8 Kbyte I/O Mapped)

The SIO also supports the following functions:

1. Floppy DSKCHG Function
2. Port 92 Function (Alternate A20 and Alternate Reset)
3. Coprocessor Logic (FERR# and IGNNE# Function)

The binary code formed by the three Encoded Chip Selects determines which Utility Bus device is selected. The SIO also provides an Encoded Chip Select Enable signal (ECSEN#) that is used to select between the two external decoders. A zero selects decoder 1 and a one selects decoder 2. The table below shows the address decode for each of the Utility Bus devices.

Table 29. NMI Source Enable/Disable and Status Port Bits

NMI Source	I/O Port Bit for Status Reads	I/O Port Bit for Enable/Disable
IOCHK#	Port 061h, Bit 6	Port 061h, Bit 3
SERR#	Port 061h, Bit 7	Port 061h, Bit 2

Table 30. Encoded Chip Select Summary Table

ECSADDR2	ECSADDR1	ECSADDR0	ECSEN#	Address Decoded	External Chip Select	Note	Cycle Type
Decoder 1							
0	0	0	0	70h, 72h, 74, 76h	RTCALE#		I/O W
0	0	1	0	71h, 73h, 75h, 77h	RTCCS#		I/O R/W
0	1	0	0	60h, 62h, 64h, 66h	KEYBRDCS#		I/O R/W
0	1	1	0	000E0000h–000FFFFFh FFFE0000h–FFFFFFFFh FFF80000h–FFFDFFFFh	BIOSCS#	1	MEM R/W
1	0	0	0	3F0h–3F7h (primary) 370h–377h (secondary)	FLOPPYCS#	2	I/O R/W
1	0	1	0	1F0h–1F7h (primary) 170h–177h (secondary)	IDECS0#	2	I/O R/W
1	1	0	0	3F6h–3F7h (primary) 376h–377h (secondary)	IDECS1#	2	I/O R/W
1	1	1	0	Reserved			
Decoder 2							
0	0	0	1	Reserved			
0	0	1	1	0C00h	CPAGECS#	3	I/O R/W
0	1	0	1	0800h–08FFh	CFIGMEMCS#	3	I/O R/W
0	1	1	1	3F8h–3FFh (COM1) -or- 2F8h–37Fh (COM2)	COMACS#	4	I/O R/W
1	0	0	1	3F8h–3FFh (COM1) -or- 2F8h–37Fh (COM2)	COMBCS#	4	I/O R/W
1	0	1	1	3BCh–3BFh (LPT1) 378h–37Fh (LPT2) 278h–27Fh (LPT3)	LPTCS#	5	I/O R/W
1	1	0	1	Reserved			
1	1	1	1	Idle State			

NOTES:
1. The encoded chip select signals for BIOSCS# will always be generated for accesses to the upper 64 KB at the top of 1 MByte (F0000h–FFFFFh) and its aliases at the top of the 4 GB and 4 GB-1 MByte. Access to the lower 64 KByte (E0000h–EFFFFh) and its aliases at the top of 4 GB and 4GB–1MB can be enabled or disabled through the SIO. An additional 384 KB of BIOS memory at the top of 4 GB (FFFD0000h–FFFDFFFFh) can be enabled for BIOS use.
2. The primary and secondary locations are programmable through the SIO. Only one location range can be enabled at any one time. The floppy and IDE share the same enable and disable bit (i.e. if the floppy is set for primary, the IDE is also set for primary).
3. These signals can be used to select additional configuration RAM.
4. COM1 and COM2 address ranges can be programmed for either port A (COMACS#) or port B (COMBCS#).
5. Only one address range (LPT1, LPT2, or LPT3) can be programmed at any one time.

Port 92h Function

The SIO integrates the Port 92h Register. This register provides the alternate reset (ALTRST) and alternate A20 (ALT_A20) functions. Figure 19 shows how these functions are tied into the system.

DSKCHG Function

DSKCHG is tied directly to the DSKCHG signal of the floppy controller. This signal is inverted and driven onto system data line 7 (SD7) during I/O read cycles to floppy address locations 3F7h (primary) or 377 (secondary) as indicated by Table 31.

Table 31. DSKCHG Summary Table

FLOPPYCS# Decode	IDECSx# Decode	State of SD7 (Output)	State of UBUSOE#
Enabled	Enabled	Tri-stated	Enabled
Enabled	Disabled	Driven via DSKCHG	Disabled
Disabled	Enabled	Tri-stated	Enabled(1)
Disabled	Disabled	Tri-stated	Disabled

NOTE:
1. For this mode to be supported, extra logic is required to disable the U-bus transceiver for accesses to 3F7/377. This is necessary because of potential contention between the Utility bus buffer and a floppy on the ISA Bus driving the system bus at the same time during shared I/O accesses.

Coprocessor Error Support

If bit 5 in the ISA Clock Divisor Register is set to a one, the SIO will support coprocessor error reporting through the FERR#/IRQ13 signal.

FERR# is tied directly to the Coprocessor error signal of the CPU. If FERR# is driven active in this mode (coprocessor error detected by the CPU), an internal IRQ13 is generated and the INT output from the SIO is driven active. When a write to I/O location F0h is detected, the SIO negates IRQ13 and drives IGNNE# active. IGNNE# remains active until FERR# is driven inactive. Note that IGNNE# is not generated unless FERR# is active.

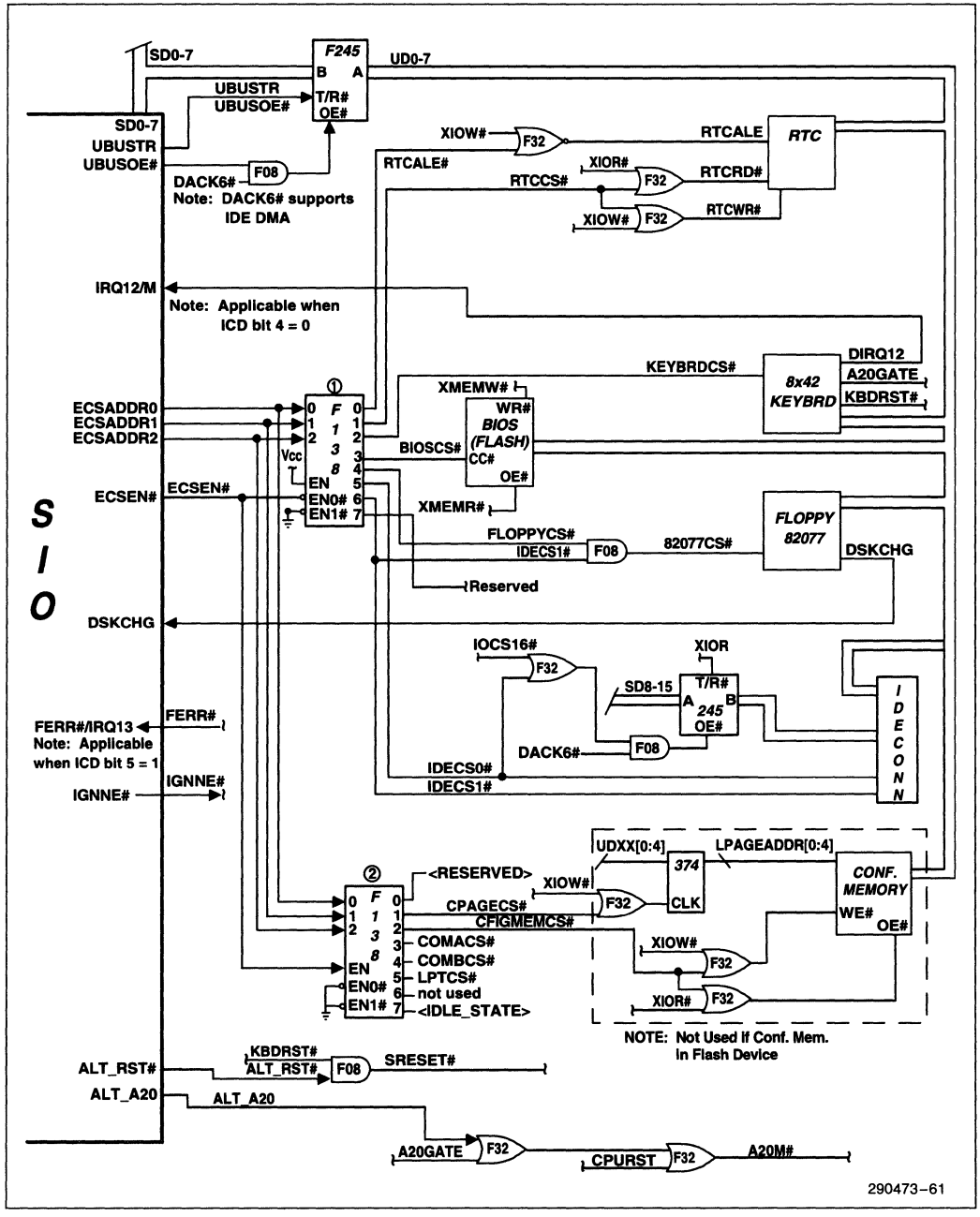


Figure 19. Utility Bus External Support Logic

Utility Bus accesses by the SIO, by an ISA master, and by the DMA is shown in Figure 20 and Figure 21. UBUSOE# and UBUSTR are driven differently for DMA cycles as shown in Figure 21.

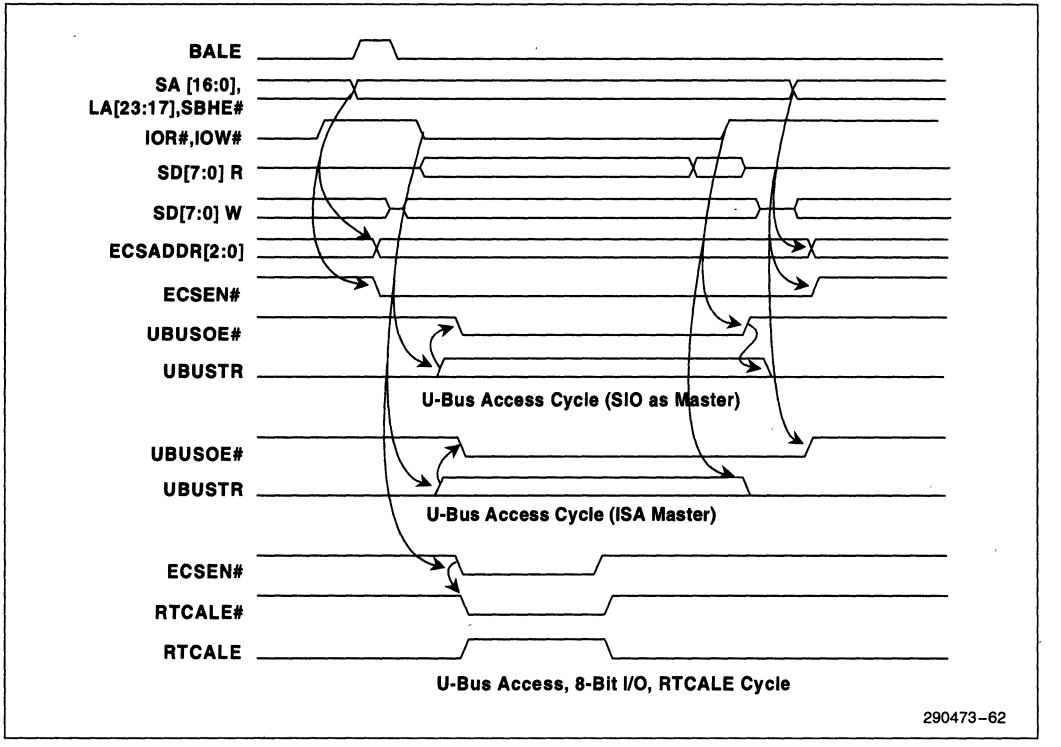


Figure 20. Utility Bus Access (SIO and ISA Master)

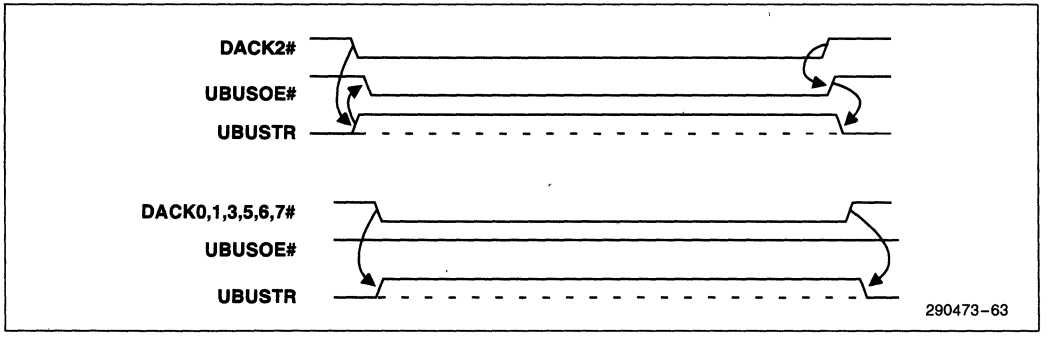


Figure 21. Utility Bus Access (DMA)

5.10 Power Management

The SIO's power management architecture is based around three core functions:

1. SMM (System Management Mode)
2. Clock Throttling
3. APM (Advanced Power Management Interface)

SMM is a mode during which an S Series Processor is executing SMM code from a secure memory space (SMRAM). SMM is invoked through the assertion of an SMI (System Management Interrupt). Physically, this is signaled over the SMI# pin. SMI's are triggered by various hardware and software events. SMRAM is used to store the SMM code which is really the SMI interrupt handler routine.

Clock Throttling will be used to reduce the power consumption of the CPU. STPCLK# is the physical signal used to control the CPU's clock.

APM creates an interface to allow the Operating System to communicate with the SMM code.

Figure 22 shows how the power management signals are connected in a Saturn based system with an S-series CPU.

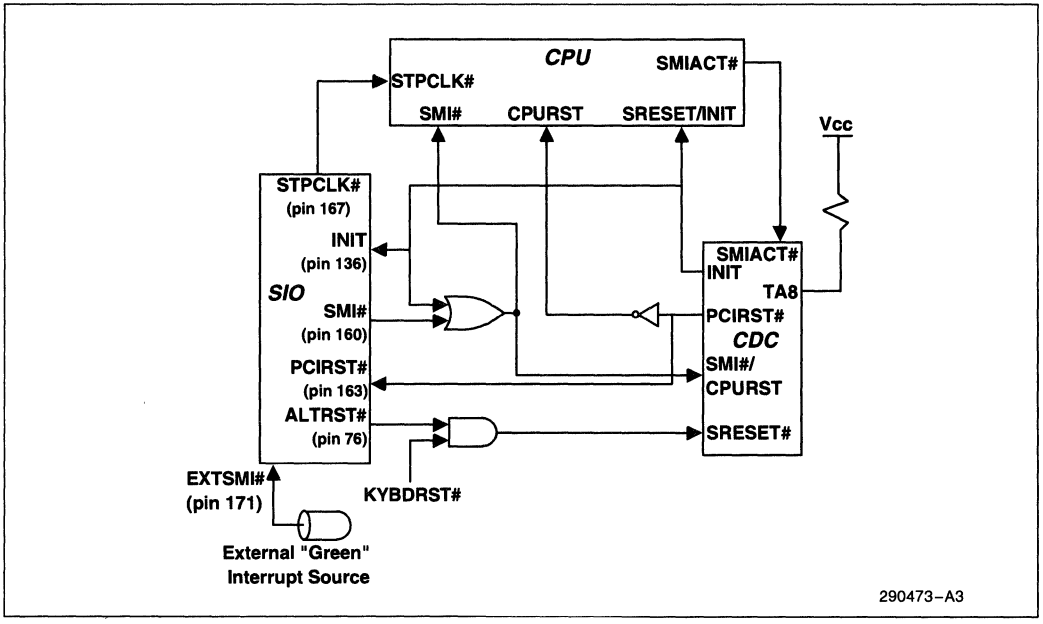


Figure 22. Power Management

6.0 ELECTRICAL CHARACTERISTICS

6.1 Absolute Maximum Ratings*

Case Temperature under Bias . . . −65°C to +110°C
Storage Temperature . . . . . −65°C to +150°C
Supply Voltages
with Respect to Ground . . . −0.5V to V_{CC} + 0.5V
Voltage On Any Pin . . . . . −0.5V to V_{CC} + 0.5V

** WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

7.0 MECHANICAL SPECIFICATIONS

7.1 Package Diagram

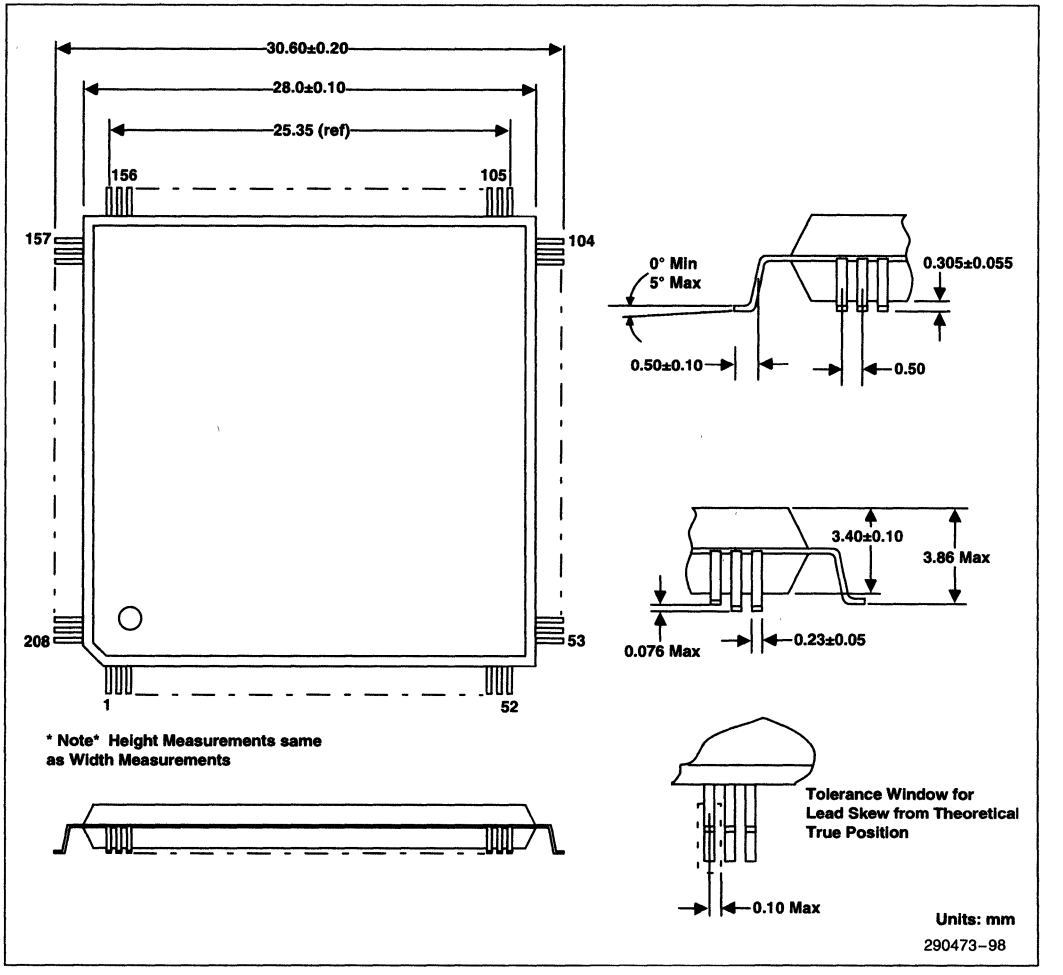


Figure 23. 208-Pin Quad Flat Pack (QFP) Package Dimensions

7.2 Thermal Specifications

Table 32. 82378 QFP Package Thermal Characteristics

Thermal Resistance-°C/Watt			
	Air Flow Rate (Ft./Min)		
Parameter	**0**	**200**	**400**
$\theta_{Junction\ to\ Case}$	6.6	6.6	6.6
$\theta_{Case\ to\ Ambient}$	36.6	27.4	24

8.0 TESTABILITY

The TEST and TESTO pins are used to test the SIO. During normal operations, the TEST pin must be grounded. The test output TESTO may be left as a no-connect (NC).

8.1 Global Tri-State

The TEST pin and IRQ3 are used to provide a high-impedance tri-state test mode. When the following input combination occurs, all outputs and bi-directional pins are tri-stated, with the exception of TESTO:

TEST = "1"
IRQ3 = "1"

The SIO must be reset after the bi-directional and output pins have been tri-stated in this manner.

8.2 NAND Tree

A NAND Tree is provided primarily for V_{IL}/V_{IH} testing. The NAND Tree is also useful for ATE at board level testing. The NAND Tree allows the tester to test the solder connections for each individual signal pin.

The TEST pin, along with IRQ5 or IRQ6, activates the NAND Tree. All bi-directional pins, and certain pure output pins using bi-directional buffers for performance reasons, are tri-stated when the following input combinations occur:

TEST = "1"
IRQ5 = "1"
- or -
TEST = "1"
IRQ6 = "0"

In the 82378, the output pulse train is observed at the TESTO test output. Pure output pins are not included directly in the NAND Tree. As noted in Section 8.3, each output can be expected to toggle after the corresponding node noted next to the pin name toggles from a "1" to a "0".

The sequence of the ATE test is as follows:

1. Drive TEST and IRQ5 high or TEST high and IRQ6 low.
2. Drive each input and bi-directional pin noted in Section 8.3 high.
3. Starting with the pin farthest from TESTO (SA8), individually drive each pin low. Expect TESTO to toggle with each pin. Expect each pure output noted in Section 8.3 to toggle after each corresponding input pin has been driven low.
4. Turn off tester drivers before driving TEST low.
5. Reset the SIO prior to proceeding with further testing.

8.3 NAND Tree Cell Order

Table 33. NAND Tree Cell Order

Tree Output #	Pin #	Pin Name	Notes
	14	IRQ4	Reserved
	21	TESTO	Test Mode Output
1	11	IRQ5	Cell Closest to TESTO
2	10	SA9	
3	9	IRQ6	
4	8	SA10	
5	7	IRQ7	
6	6	SA11	
7	5	SA12	
8	4	REFRESH#	
9	3	SA13	
10	207	SA14	
11	206	MASTER#	
12	205	SA15	
13	204	MEMW#	
14	203	MEMR#	
15	202	SA16	
16	201	SA17	
17	200	IOR#	
18	199	SA18	
19	198	IOW#	
20	197	SA19	
21	196	SMEMR#	
22	193	AEN	
23	192	SMEMW#	
24	191	IOCHRDY	
25	190	SD0	
26	189	SD1	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
27	188	ZEROWS#	
28	187	SD2	
29	186	SD3	
30	185	SD4	
31	184	IRQ9	
32	180	SD5	
33	179	SD6	
34	178	SD7	
35	177	RSTDRV	
36	176	IOCHK#	
	175	ECSADDR0	NAND Tree Output of Tree Cell 28
	174	ECSADDR1	NAND Tree Output of Tree Cell 29
	173	ECSADDR2	NAND Tree Output of Tree Cell 30
37	172	IRQ8#	
38	171	EXTSMI#	
	170	ECSEN#	NAND Tree Output of Tree Cell 32
	169	TEST	PI = > VCC, TEST must be '1'
39	168	IRQ1	
	167	STPCLK#	
40	166	SYSCLK	
	165	UBUSTR	NAND Tree Output of Tree Cell 33
	164	UBUSOE#	NAND Tree Output of Tree Cell 34
41	163	PCIRST#	
42	161	DSKCHG	
	160	SMI#	
43	159	AD0	
44	155	AD1	
45	154	AD2	
46	153	AD3	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
47	152	AD4	
48	151	AD5	
49	150	AD6	
50	149	AD7	
51	148	AD8	
52	147	C/BE0#	
53	146	AD9	
54	143	AD10	
55	142	AD11	
56	141	AD12	
57	140	AD13	
58	139	AD14	
59	138	AD15	
60	137	C/BE1#	
61	136	INIT	
62	135	PAR	
63	134	SERR#	
64	133	LOCK#	
65	132	STOP#	
66	128	DEVSEL#	
67	127	TRDY#	
68	126	IRDY#	
69	125	FRAME#	
70	124	C/BE2#	
71	123	AD16	
72	122	AD17	
73	121	AD18	
74	120	AD19	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
75	119	AD20	
76	118	AD21	
77	115	AD22	
78	114	AD23	
79	113	C/BE3#	
80	112	AD24	
81	111	AD25	
82	110	AD26	
83	109	AD27	
84	108	AD28	
85	107	AD29	
86	106	AD30	
87	102	AD31	
88	101	IDSEL	
89	100	REQ3#	
90	98	REQ1#	
91	97	REQ2#	
92	96	CPUREQ#	
	95	CPUGNT#	NAND Tree Output of Tree Cell 93
	94	GNT1#	NAND Tree Output of Tree Cell 95
93	93	REQ0#	
	92	GNT0#	NAND Tree Output of Tree Cell 100
94	90	PCICLK	
	89	FLSHREQ#	NAND Tree Output of Tree Cell 102
95	88	MEMACK#	
	87	MEMREQ#	NAND Tree Output of Tree Cell 103

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
	86	MEMCS#	NAND Tree Output of Tree Cell 104
	85	ALT_A20	NAND Tree Output of Tree Cell 105
96	84	PIRQ[3]#	
97	83	PIRQ[2]#	
98	82	PIRQ[1]#	
99	81	PIRQ[0]#	
100	80	OSC	
	76	ALT_RST#	NAND Tree Output of Tree Cell 23
	75	INT	NAND Tree Output of Tree Cell 24
	74	NMI	NAND Tree Output of Tree Cell 25
101	73	SPKR	
	72	IGNNE#	NAND Tree Output of Tree Cell 26
102	71	FERR#	
103	70	SD15	
104	69	SD14	
105	68	SD13	
106	67	SD12	
107	65	DREQ7	
108	64	SD11	
109	63	DACK7#	
110	62	SD10	
111	61	DREQ6	
112	60	SD9	
113	59	DACK6#	
114	58	DREQ3	
115	57	DREQ2	
116	56	DREQ1	
117	55	SD8	
118	51	DREQ5	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
119	50	DACK5#	
120	49	DACK3#	
121	48	DACK1#	
122	47	DREQ0	
123	46	LA17	
124	45	DACK0#	
125	44	LA18	
126	43	IRQ14	
127	42	LA19	
128	41	IRQ15	
129	40	LA20	
130	39	IRQ12/M	
131	38	LA21	
132	37	IRQ11	
133	36	LA22	
134	35	IRQ10	
135	34	LA23	
136	33	IOCS16#	
137	32	SBHE#	
138	31	MEMCS16#	
139	30	SA0	
140	29	SA1	
141	28	SA2	
142	24	SA3	
143	23	BALE	
144	22	SA4	
145	20	EOP	
146	19	SA5	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
147	18	DACK2#	
148	17	SA6	
149	16	IRQ3	Output signals will transition from high-impedance state to driving state after this pin is driven low.
150	15	SA7	
151	13	SA8	Cell furthest from TESTO Start of NAND Tree

8.4 NAND Tree Diagram

Figure 24 shows the NAND Tree Diagram.

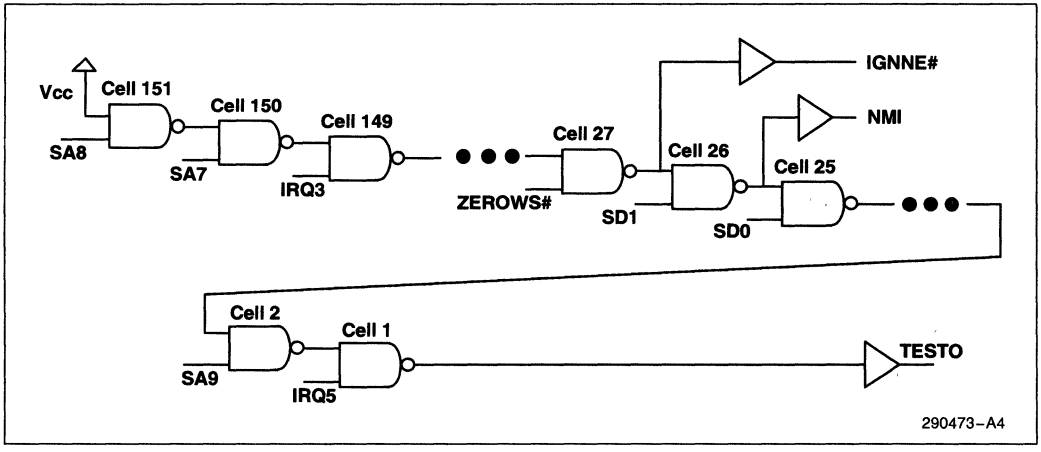


Figure 24. NAND Tree Diagram for 82378

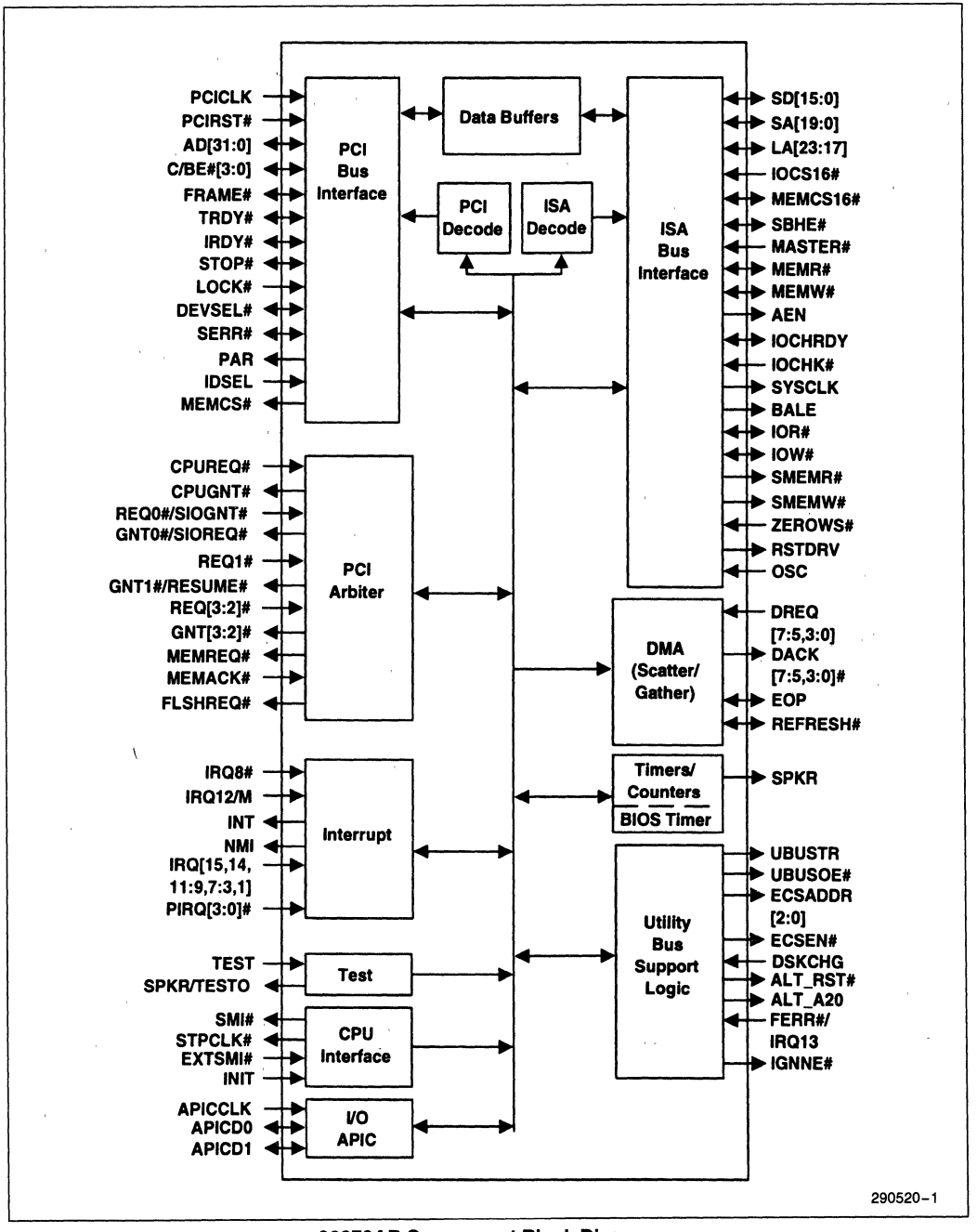
82379AB SYSTEM I/O-APIC (SIO.A)

- Provides the Bridge between the PCI Bus and ISA Bus
- 100% PCI and ISA Compatible
  - PCI and ISA Master/Slave Interface
  - Directly Drives 10 PCI Loads and 6 ISA Slots
  - Supports PCI at 25 MHz and 33 MHz
  - Supports ISA from 6 MHz to 8.33 MHz
- Enhanced DMA Functions
  - Compatible DMA Transfers
  - 27-Bit Addressability
  - Seven Independently Programmable Channels
  - Functionality of Two 82C37A DMA Controllers
- Integrated Data Buffers to Improve Performance
  - 8-Byte DMA/ISA Master Line Buffer
  - 32-Bit Posted Memory Write Buffer to ISA
- Integrated 16-Bit BIOS Timer
- Non-Maskable Interrupts (NMI)
  - PCI System Errors
  - ISA Parity Errors
- Four Dedicated PCI Interrupts
  - Level Sensitive
  - Can be Mapped to Any Unused Interrupt
- Arbitration for ISA Devices
  - ISA Masters
  - DMA and Refresh
- Arbitration for PCI Devices
  - Six PCI Masters Are Supported
  - Fixed, Rotating, or a Combination of the Two
- Utility Bus (X-Bus) Peripheral Support
  - Provides Chip Select Decode
  - Controls Lower X-Bus Data Byte Transceiver
- Integrates the Functionality of One 82C54 Timer
  - System Timer
  - Refresh Request
  - Speaker Tone Output
- Integrates the Functionality of Two 82C59 Interrupt Controllers
  - 14 Interrupts Supported
  - Edge/Level Selectable Interrupts: Each Interrupt Individually Programmable
- Complete Support for SL Enhanced Intel486™ CPU's
  - SMI# Generation Based on System Hardware Events
  - STPCLK# Generation to Power Down the CPU
- Integrated I/O Advanced Programmable Interrupt Controller (APIC)

2

The 82379AB System I/O-APIC (SIO.A) component provides the bridge between the PCI bus and the ISA expansion bus. The 82379AB also integrates many of the common I/O functions found in today's ISA based PC systems. The 82379AB incorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller that supports data buffers to isolate the PCI bus from the ISA bus and to enhance performance, PCI and ISA arbitration, 14 level interrupt controller, a 16-bit BIOS timer, three programmable timer/counters, and Non-Maskable Interrupt (NMI) Control Logic. The 82379AB also provides decode for peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive. The 82379AB supports several Advanced Power Management features such as SMI# Interrupt. The 82379AB also supports a total of 6 PCI Masters, and can support up to 4 PCI Interrupts. The 82379AB incorporates an Advanced Programmable Interrupt Controller (APIC) that communicates with the processor via a dedicated two data bit bus.

December 1994
Order Number: 290520-001



82379AB Component Block Diagram

The complete document for this product is available from Intel's Literature Center at 1-800-548-4725.

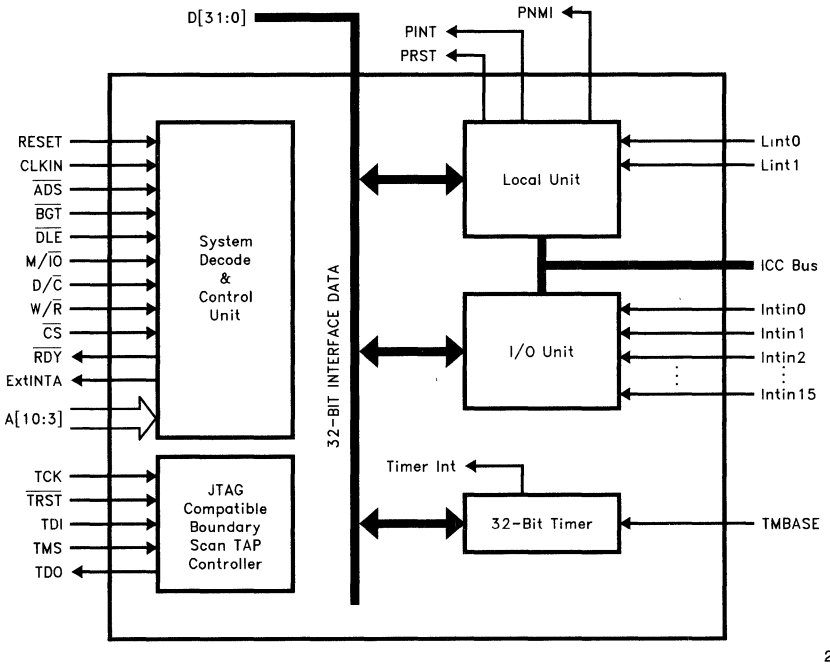
82489DX
ADVANCED PROGRAMMABLE INTERRUPT CONTROLLER

82489DX FEATURES OVERVIEW

- Advanced Interrupt Controller for 32-Bit Operating Systems
- Solution for Multiprocessor Interrupt Management
- Dynamic Interrupt Distribution for Load Balancing in MP Systems
- Separate Nibble Bus (Interrupt Controller Communications (ICC) Bus) for Interrupt Messages
- Inter-Processor Interrupts
- Various Addressing Schemes—Broadcast, Fixed, Lowest Priority, etc.
- Compatibility Mode with 8259A
- 32-Bit Internal Registers
- Integrated Timer Support
- 33 MHz Operation
- 132-Lead PQFP Package, Package Type KU
  (See Packaging Specification. Order Number 240800)

2

82489DX Block Diagram



290446-1

Refer to Application Note AP-388: 82489DX User's Manual (Order Number 292116) when evaluating your design needs.

1.0 INTRODUCTION

The 82489DX Advanced Programmable Interrupt Controller provides multiprocessor interrupt management, providing both static and dynamic symmetrical Interrupt distribution across all processors.

The main function of the 82489DX is to provide interrupt management across all processors. This dynamic interrupt distribution includes routing of the interrupt to the lowest-priority processor. The 82489DX works in systems with multiple I/O subsystems, where each subsystem can have its own set of interrupts. This chip also provides inter-processor interrupts, allowing any processor to interrupt any processor or set of processors. Each 82489DX I/O unit Interrupt Input pin is individually programmable by software as either edge or level triggered. The interrupt vector and interrupt steering information can be specified per pin. A 32-bit wide timer is provided that can be programmed to interrupt the local processor. The timer can be used as a counter to provide a time base to software running on the processor, or to generate time slice interrupts locally to that processor. The 82489DX provides 32-bit software access to its internal registers. Since no 82489DX register reads have any side effects, the 82489DX registers can be aliased to a user read-only page for fast user access (e.g., performance monitoring timers).

The 82489DX supports a generalized naming/addressing scheme that can be tailored by software to fit a variety of system architectures and usage models. It also supports 8259A compatibility by becoming virtually transparent with regard to an externally connected 8259A style controller, making the 8259A visible to software.

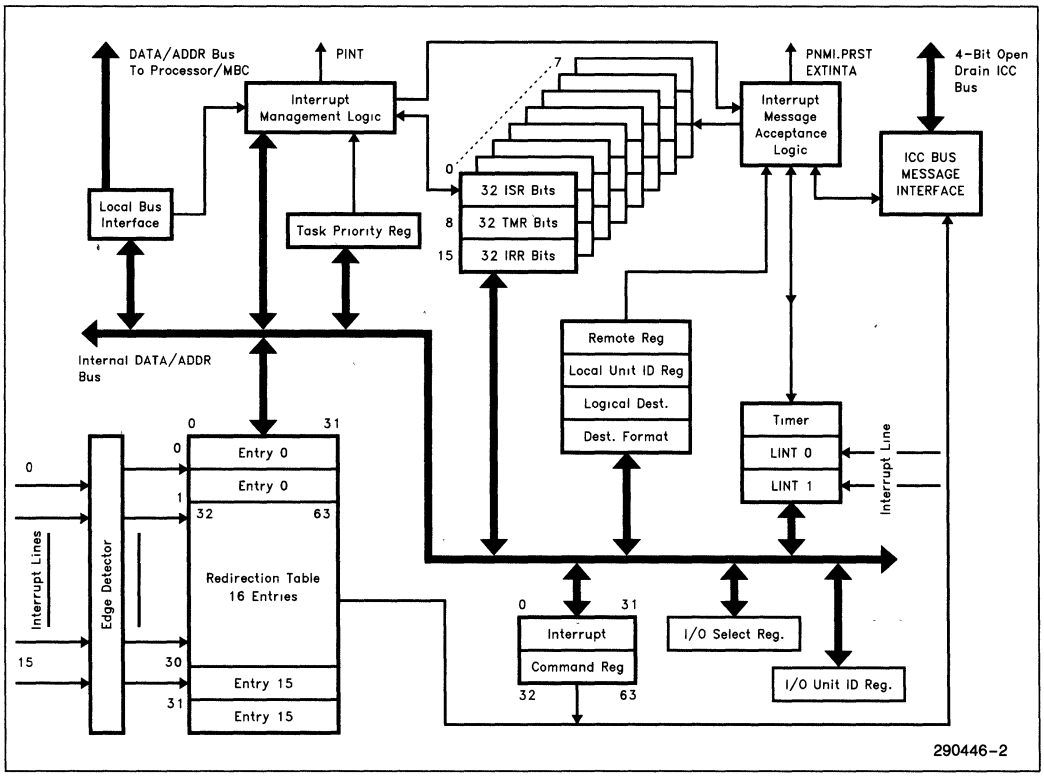


Figure 1. 82489DX Architecture

2.0 FUNCTIONAL OVERVIEW

82489DX Functional Blocks

82489DX contains one Local Unit, one I/O unit and a timer. The ICC bus is used to pass interrupt messages.

ICC BUS

The ICC bus is a 5-wire synchronous bus connecting all 82489DXs (all I/O Units and all Local Units). The Local Units and I/O Units communicate over this ICC bus. Four of these five wires are used for data transmissions and arbitration, and one wire is a clock.

LOCAL UNIT

The Local Unit contains the necessary intelligence to determine whether or not its processor should accept interrupt messages sent on the ICC bus by other Local Units and I/O Units. The Local Unit also provides local pending of interrupts, nesting and masking of interrupts, and handles all interactions with its local processor such as the INT/INTA/EOI protocol. The Local Unit further provides inter-processor interrupt functionality and a timer to its local processor. The interface of a processor to its 82489DX Local Unit is identical for every processor.

I/O UNIT

The I/O Unit provides the interrupt input pins on which I/O devices inject interrupts into the system in the form of an edge or a level. The I/O unit also contains a Redirection Table for the interrupt input pins. Each entry in the Redirection Table can be individually programmed to indicate whether an interrupt on the pin is recognized as either an edge or a level; what vector and also what priority the interrupt has; and which of all possible processors should service the interrupt and how to select that processor (statically or dynamically). The information in the table is used to send interrupt messages to all 82489DX Units via the ICC bus.

TIMER

The 82489DX provides a 32-bit wide timer that can be programmed to interrupt the local processor. The timer can be used as a counter to provide a time-base to software running on the processor, or to generate time-slice interrupts local to that processor.

3.0 PIN DESCRIPTION

The 82489DX pin description is organized in a small number of functional groups. Pin definitions and protocols have been designed to minimize interface issues. In particular, they support the notion of independently controlled address and data phases. The primary host interface is synchronous in nature.

In the following pin definition table if the signal name has (__) over it, the signal is in its active state when it has a low level. The signal direction column identifies output only signals as a continuous drive (O), tristate (T/S), or open drain (O/D). All bi-directional (BI-D) signals have tri-stating outputs.

Pin Definition Table

Symbol	Pin No.	Type	Function
SYSTEM PINS			
RESET	65	I	The **RESET INPUT** forces 82489DX to enter its initial state. The 82489DX Local Unit in turn asserts it PRST (Processor Reset) output. All tri-state outputs remain in high impedance until explicitly enabled.
ExtINTA	41	O	The **EXTERNAL INTERRUPT ACKNOWLEDGE** output is asserted (high) when an external interrupt controller (e.g., 8259) is expected to respond to the current INTA cycle. If deasserted (low), 82489DX will respond, and the INTA cycle must not be delivered to the external controller.
CLKIN	57	I	**CLOCK INPUT** provides reference timing for most of the bus signals.
$\overline{\text{TRST}}$	56	I	**TEST RESET** is the JTAG compatible boundary scan TAP controller reset pin. A weak pull-up keeps the pin high if not driven.
TCK	55	I	**TEST CLOCK** is the clock input for the JTAG compatible boundary scan controller and latches.
TDI	53	I	**TEST DATA INPUT** is the test data input pin for the JTAG compatible boundary scan chain and TAP controller. A weak pull-up keeps this pin high if not driven.
TDO	52	O	**TEST DATA OUTPUT** is the test data output for the JTAG compatible boundary scan chain.
TMS	54	I	**TEST MODE SELECT** is the test mode select pin for the JTAG boundary scan TAP controller. A weak pull-up keeps this pin high if not driven.
TIMER PIN			
TMBASE	59	I	The **TIME BASE** input provides a standard frequency that is only used by the 82489DX timer and that is independent of the system clock.
INTERRUPT PINS			
INTIN[15:0]	82–97	I	These 16 **INTERRUPT INPUT** pins accept edge or level sensitive interrupt requests from I/O or other devices. The pin numbers are specified respectively. INTIN15 corresponds to pin number 82, INTIN14 corresponds to pin number 83 etc., and INTIN0 corresponds to pin number 97. These pins are active high.
LINTIN[1] LINTIN[0]	80 81	I I	Two **LOCAL INTERRUPT INPUT** pins accept edge or level sensitive interrupt requests that can only be delivered to the connected processor. These pins are active high.
REGISTER ACCESS PINS			
$\overline{\text{ADS}}$	64	I	**ADDRESS STROBE** signal indicating the start of a bus cycle. 82489DX does not commit to start the cycle internally until BUS GRANT is detected active.

Pin Definition Table (Continued)

Symbol	Pin No.	Type	Function
REGISTER ACCESS PINS (Continued)			
M/$\overline{IO}$, D/$\overline{C}$, W/$\overline{R}$	63 61 62	I I I	Bus cycle definition signals. Note that since the 82489DX registers can be mapped in either memory or I/O space, the M/$\overline{IO}$ pin is not used for register access cycles; it is only used to decode interrupt acknowledge cycles. 82489DX does not respond to code read cycles.
$\overline{BGT}$	66	I	The **BUS GRANT** input is optional and is used to indicate the address phase of a bus cycle in configurations where address timing cannot be inferred from $\overline{ADS}$. This signal is really used as an address latch enable, but is named as it is to indicate that it can normally be connected to the Intel Cache Controller generated signal of the same name. Must be tied low if not used.
$\overline{CS}$	74	I	The **CHIP SELECT** input indicates that the 82489DX registers are being addressed.
A3 A4 A5 A6 A7 A8 A9 A10	31 29 28 27 26 24 22 21	BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D	The address pins are used as inputs in addressing internal register space. Output function is reserved. They are also used to latch local unit ID on reset.
$\overline{DLE}$	73	I	**DATA LATCH/ENABLE** is optional and is used to indicate committing the data phase of a bus cycle in configurations where data timing cannot be inferred from other cycle timings. Must be tied low if not used.
D31 D30 D29 D28 D27 D26 D25 D24 D23 D22 D21 D20 D19 D18 D17 D16 D15 D14 D13 D12 D11	105 107 109 110 111 112 114 115 116 118 119 121 122 123 124 125 128 129 130 131 2	BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D	The DATA BUS is for all register accesses and interrupt vectoring.

Pin Definition Table (Continued)

Symbol	Pin No.	Type	Function
REGISTER ACCESS PINS (Continued)			
D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0	3 4 7 8 9 11 12 13 14 16 18	BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D BI-D	
DP3 DP2 DP1 DP0	101 102 103 104	BI-D BI-D BI-D BI-D	One Data Parity pin for each byte on the data bus. EVEN parity is generated any time the data bus is driven by the 82489DX.
$\overline{RDY}$	43	O	**READY** output indicates that the current bus cycle is complete. In the case of a read cycle, valid data and the return to inactive state after going active low may be delayed till $\overline{DLE}$ goes active.
PROCESSOR PINS			
PINT	35	T/S	The **PROCESSOR INTERRUPT OUTPUT** indicates to the processor that one or more maskable interrupts are pending. This pin is tri-stated at reset, and has an internal pull-down resistor to prevent false signaling to the processor until the 82489DX Local Unit is enabled and this pin is actively driven.
PRST	38	O	The **PROCESSOR RESET OUTPUT** is asserted/de-asserted upon 82489DX reset, and also in response to ICC bus messages with "RESET" delivery mode. This pin should be used with care.
PNMI	37	T/S	The **NON-MASKABLE INTERRUPT** output is signaled in respone to ICC bus messages with "NMI" delivery mode. This pin is tri-stated at reset, and has an internal pull-down resistor to prevent false signaling to the processor until the Local Unit is enabled and this pin is actively driven.
ICC BUS PINS			
ICLK	60	I	The **ICC BUS CLOCK** input provides synchronous operation of the ICC bus.
MBI[3:0]	76–79	I	The four **ICC BUS IN** inputs are used for incoming ICC bus messages. In smaller configurations the ICC bus input and outputs may be tied directly together at the pins. Pin number for MBI3 is 76, MBI2 is 77, MBI1 is 78 and MBI0 is 79.
MBO3 MBO2 MBO1 MBO0	45 48 49 51	O/D	The four **ICC BUS OUT** outputs are used for outgoing ICC bus messages. The current capacity is only 4 mA. So external buffers will be needed.

Pin Definition Table (Continued)

Symbol	Pin No.	Type	Function
RESERVED PINS			
Reserved	34, 42	NC	These pins **MUST BE LEFT OPEN.**
Reserved	70, 72, 75		**Reserved** by Intel. **These pins should be strapped to V_{CC}.**
Reserved	71, 19, 20		**Reserved** by Intel. **These pins should be strapped to GND.**
POWER AND GROUND PINS			
V_{CC}	1, 32, 69, 98	POWER	Nominally +5V. These pins along with V_{SS} and V_{SSI} should be separately bypassed.
V_{CCP}	6, 15, 25,100, 108, 117, 126	POWER	Nominally +5V. These pins along with V_{SSP} should be separately bypassed.
V_{CCPO}	39, 46	POWER	Nominally +5V. These pins along with V_{SSPO} should be separately bypassed.
V_{SS}	5, 33, 67, 68, 99	GND	Nominally 0V. These pins along with V_{CC} should be separately bypassed.
V_{SSP}	10, 17, 23, 30, 106, 113, 120, 127, 132,	GND	Nominally 0V. These pins along with V_{CCP} should be separately bypassed.
V_{SSPO}	36, 40, 44, 47, 50	GND	Nominally 0V. These pins along with V_{CCPO} should be separately bypassed.
V_{SSI}	58	GND	Nominally 0V. These pins along with V_{CC} should be separately bypassed.

NOTE:
V_{CC}, V_{CCP} and V_{CCPO} should be of same voltage. V_{SS}, V_{SSP}, V_{SSPO} and V_{SSI} should be 0V.

4.0 FUNCTIONAL DESCRIPTION

As far as interrupt management is concerned, the 82489DX's interrupt control function spans over two functional units, the I/O Unit of which there is one per I/O subsystem, and the Local Unit of which there is one per processor. 82489DX has one I/O unit and one Local Unit in a single package. This section takes a detailed look at both local and I/O Units.

I/O Unit

The I/O Unit consists of a set of Interrupt Input pins, an Interrupt Redirection Table, and a message unit for sending and receiving messages from the ICC bus. The I/O Unit is where I/O devices inject their interrupts, the I/O Unit selects the corresponding entry in the Redirection Table and uses the information in that entry to format an interrupt request message. The message unit then broadcasts this message over the ICC bus. The content of the Redirection Table is under software control and is assigned benign defaults upon reset. The masks in the Redirectional Table entries are set to 1 at *hardware reset* to disable the interrupts.

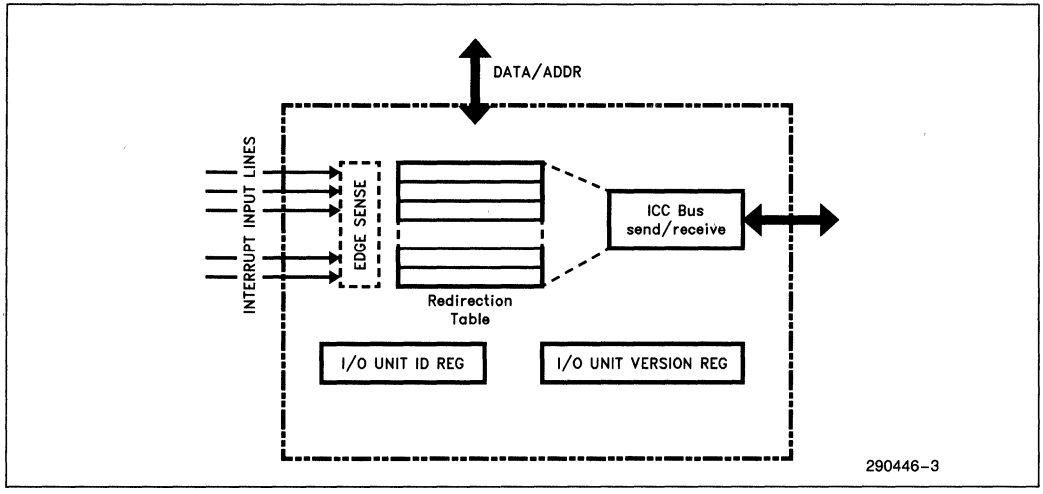


Figure 2. 82489DX I/O Unit Block Diagram

Local Unit

Interrupt Management of the Local Unit is responsible for local interrupt sources, interrupt acceptance, dispensing interrupts to the processor, and sending inter-processor interrupts. Depending on the delivery mode of the interrupt, zero, one or more units can accept an interrupt. A Local Unit accepts an interrupt only if it will deliver the interrupt to its processor. Accepting an interrupt is purely an inter-82489DX matter; dispensing an interrupt to the local processor only involves a 82489DX and its local processor.

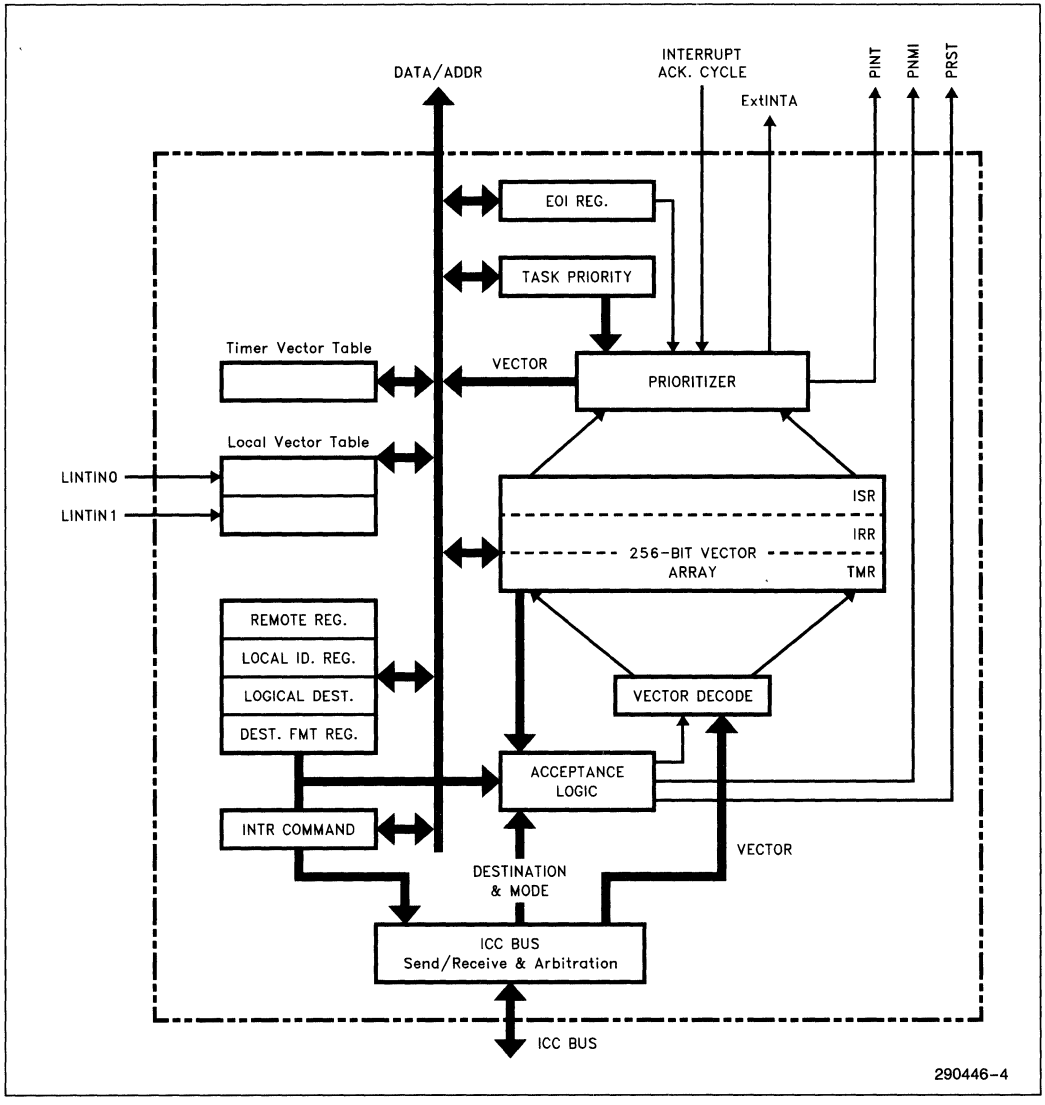


Figure 3. 82489DX Local Unit Block Diagram

2

5.0 INTERRUPT CONTROL MECHANISM

This section describes briefly the interrupt control mechanism in the 82489DX.

5.1 Interrupts

The interrupt control function of all 82489DXs are collectively responsible for delivering interrupts from interrupt sources to interrupt destinations in the multiprocessor system. When a processor accepts an interrupt, it uses the vector to locate the entry point of the handler in its interrupt table. The 82489DX architecture allows for 16 possible interrupt priorities; zero being the lowest priority and 15 being the highest. Priority of interrupt A "is higher than" the priority of interrupt B if servicing A is more urgent than servicing B. An interrupt's priority is implied by its vector; namely priority = vector/16.

With 256 vectors and 16 different priorities, this implies that 16 different interrupt vectors can share a single interrupt priority.

TOTAL ALLOWED INTERRUPT VECTORS

Out of 256 vectors, interrupt vectors 0 to 15 should not be used in the 82489DX. Only 240 interrupt vectors (vectors from 16 to 255) are supported in the 82489DX.

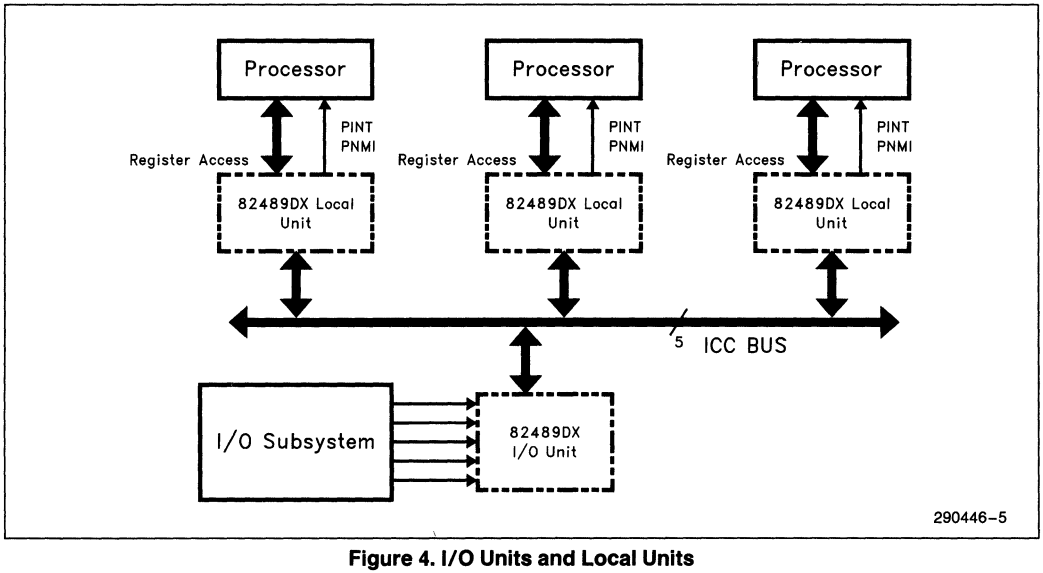


Figure 4. I/O Units and Local Units

INTERRUPT SOURCES

Interrupts are generated by a number of different interrupt sources in the system.

Possible interrupt sources are:

- Externally connected (I/O) devices. Interrupts from these external sources manifest themselves as edges or levels on interrupt input pins and can be redirected to any processor.
- Locally connected devices. These originate as edges or levels on interrupt pins, but they are always directed to the local processor only.
- 82489DX timer generated interrupts. Like locally connected devices, 82489DX timer can only interrupt its local processor.
- Processors. A processor can interrupt any individual processor or sets of processors. This supports software self-interrupts, preemptive scheduling, TLB flushing, and interrupt forwarding. A processor generates interrupts by writing to the interrupt command register in its Local Unit.

INTERRUPT DESTINATIONS

I/O Units can only source interrupts whereas Local Units can both source and accept interrupts, so whenever "interrupt destination" is discussed, it is implied that the Local Unit is the destination of the interrupt. In physical mode the destination processor is specified by a unique 8-bit 82489DX local ID. Only a single destination or a broadcast to all (LOCAL ID of all ones) can be specified in physical destination mode.

In logical mode destinations are specified using a 32-bit destination field. All Local Units contain a 32-bit Logical Destination register against which the destination field of the interrupt is matched to determine if the receiver is being targeted by the interrupt. An additional 32-bit Destination Format register in each Local Unit enables the logical mode addressing.

INTERRUPT DELIVERY

The description of interrupt delivery makes frequent use of the following terms:

- Each processor has a processor priority that reflects the relative importance of the code the processor is currently executing. This code can be part of a process or thread, or can be an interrupt handler. A processor's priority fluctuates as a processor switches threads, a thread or handler raises and lowers its priority level to mask out interrupt, and the processor enters an interrupt handler and returns from an interrupt handler to previously interrupted activity.
- A processor is lowest priority within a given group of processors if its processor priority is the lowest of all processors in the group. Note that more than one processor can be the lowest priority in a given group.
- A processor is the focus of an interrupt if it is currently servicing that interrupt, or if it currently has a request pending for the interrupt.

Interrupt delivery begins with an interrupt source injecting its interrupt into the interrupt system at one of the 82489DX. Delivery is complete only when the servicing processor tells its 82489DX Local Unit it is complete by issuing an end-of-interrupt (EOI) command to its 82489DX Local Unit. Only then has all (relevant) internal state regarding that occurrence of the interrupt been erased. The interrupt system guarantees exactly-once delivery semantics of interrupts to the specified destinations. Exactly-once guaranteed delivery implies a number of things:

- The interrupt system never rejects interrupts; it never NAKs interrupt injection, interrupts are never lost, and the same interrupt (occurrence) is never delivered more than once.

Clearly a single edge interrupt or level interrupt counts as a single occurrence of an interrupt. In uniprocessor systems, an occurrence of an interrupt that is already pending (IRR) cannot be distinguished from the previous occurrence. All occurrences are recorded in the same IRR bit. They are therefore treated as "the same" interrupt occurrence.

For lowest-priority delivery mode, by delivering an interrupt first to its focus processor (if it currently has one), the identical behavior can be achieved in a MP (Multiprocessor) system. If an interrupt has a focus processor then the interrupt will be delivered to the interrupt's focus processor independent of priority information. This means that even if there is a lower priority processor compared to the focus processor, the interrupt still gets delivered to the focus processor.

Each edge occurring on an edge triggered interrupt input pin is clearly a one-shot event; each occurrence of an edge is delivered. An active level on a level triggered interrupt input pin represents more of a "continuous event". Repeatedly broadcasting an interrupt message while the level is active would cause flooding of the ICC bus, and in effect transmits very little useful information since the same processor (the focus) would have to be the target.

Instead, for level triggered interrupts the 82489DX merely recreate the state of the interrupt input pin at the destination . The source 82489DX accomplishes this by tracking the state of the appropriate destination 82489DX's Interrupt Request Register (or pending bit) and only sending inter-82489DX messages when the state of the interrupt input pin and the destination's interrupt request enter a disagreement. Unlike edge triggered interrupts, when a level interrupt goes into service, the interrupt request at the servicing 82489DX is not automatically removed. If the handler of a level sensitive interrupt executes an EOI then that interrupt will immediately be raised to the processor again, unless the processor has explicitly raised its task priority, or the source of that interrupt has been removed.

5.2 Interrupt Redirection

This section specifically talks about how a processor is picked during interrupt delivery. The 82489DX supports two modes for selecting the destination processor: Fixed and Lowest Priority.

- *Fixed Delivery Mode*
  In fixed delivery mode, the interrupt is unconditionally delivered to all local 82489DXs that match in the destination information supplied with the interrupt. Note that for I/O device interrupts typically only a single 82489DX would be listed in the destination. Priority and focus information are ignored. If the priority of a destination processor equal to or higher than the priority of the interrupt, then the interrupt is held pending locally in the destination processor's Local Unit, until the processor priority becomes low enough at which time the interrupt is dispensed to the processor. More than one processor can be the destination in fixed-delivery mode.
- *Lowest Priority Delivery Mode*
  Under the lowest priority delivery mode, the processor to handle the interrupt is the one in the specified destination with the lowest processor priority value. If more than one processor is at the lowest priority, then a unique arbitration ID is used to break ties. For lowest priority dynamic delivery, the interrupt will always be taken by its focus processor if it has one. The lowest priority delivery method assures minimum interruption of high priority tasks. Since each Local Unit only knows its own processor priority, determining the lowest priority processor is done by arbitration on the ICC bus. Only one processor can be the destination in lowest-priority delivery mode.

INTER-82489DX COMMUNICATION

All I/O and Local Units communicate during interrupt delivery. Interrupt information is exchanged between different units on a dedicated five wire ICC bus in the form of broadcast messages. A 82489DX Unit's 8-bit ID is used as its name for the purpose of using the ICC bus, and all 82489DX units using one ICC bus should be assigned a different ID. The Arbitration ID of the Local Units used to resolve tles during lowest priority arbitration is also derived from the Local Unit's ID.

16.0 GUIDELINES FOR 82489DX USERS

16.1 Initialization

This section outlines one possible initialization scenario. Other scenarios are certainly possible, and one would be selected as part of a platform standard initialization scheme. The intent of this section is to illustrate that the initialization support provided by the 82489DX is adequate to support MP (Multiprocessor) system initialization.

Each 82489DX has a RESET input pin connected to a common Reset line. Upon system reset, this common reset line is activated, causing all the 82489DXs to go through reset. All 82489DX local units (note: only local units and not I/O units) latch their ID from their address bus on reset. The ID can be provided by the bus control agent based on slot number.

The local units next assert their processor's Reset pin, holding the processor in reset, and next perform their internal reset, setting all registers to their initial state. The initial state of all 82489DX Units (both local and I/O units) is "all masks set" and all Local

Units disabled; registers are otherwise initialized to zero. Note that the PINT and PNMI output pins are in tri-state mode when the local unit is disabled. After this, each 82489DX local unit will deassert its processor's Reset pin, allowing the processors to come out of reset and perform self test and start executing initialization code.

Note that while connecting PRST pin it should be noted that whenever PRST pin is activated by 82489DX either because of software reset message or hardware reset, the 82489DX itself is reset. It should be taken care in the cases of Warm reset where only processors need to be reset and not the interrupt controller. In brief, the usage of PRST depends upon the system requirement on various reset.

Somewhere in this code sequence, the processors that are "alive" will enable their 82489DX local units, and attempt to force all the other processors back into Reset. Forcing the other processors into reset is performed by sending them the inter-processor interrupt with Destination Mode = "Physical", Delivery Mode = "Reset", Trigger Mode = "Level", Level = "1", and Destination Shorthand = "All Excl Self". Only the first processor to get the ICC bus will succeed in sending this signal and reset all other 82489DXs and their processors. The other processors are kept in reset until such time that an MP operating system decides they can become active again. The only running processor next performs the rest of system initialization.

Eventually, an MP operating system will be booted at which time the operating system would send "deassert reset" interprocessor signals to activate the other processors in the system. A mechanism must be provided by the platform that allows the added processors to differentiate the very first reset from a subsequent one.

16.2 Compatibility

COMPATIBILITY LEVELS

The 82489DX can be used in conjunction with standard 8259A-style interrupt controllers to provide a range of compatibility levels.

At the lowest level we have "PC shrink-wrap" compatibility. This level effectively creates a uniprocessor hardware environment within the MP platform capable of booting/running DOS shrinkwrap software. In this mode, only the 8259A generates interrupts and the 82489DX becomes a virtual wire. The interrupt latency can be minimized by connecting the 8259A interrupt to local unit directly.

The next level preserves the software compatible view of an 8259A but it allows more than one processor to be active in the system. This results in an asymmetrical arrangement, with one processor fielding all 8259A interrupts but with added inter-processor interrupt capability. In this mode, 82489DX "merges" 8259A interrupts with inter-processor interrupts. Existing I/O drivers would be bound to the compatible CPU and interface directly with the 8259A.

At the next compatibility level, 8259A compatible drivers can be mixed with native 82489DX drivers. Devices can generate interrupts at either 8259A or an 82489DX. This provides for partial symmetry as individual drivers migrate from the 8259A to native 82489DXs.

Another 8259A compatible point can be defined for MP systems. Each processor could have its own compatible 8259A controllers, allowing multiple processors to run compatible I/O drivers, but statically spreading the load across the available processors.

82489DX/8259A INTERACTION

The principle of compatible operation is very straightforward; the 82489DX(s) become a virtual wire connecting the 8259A's INT output through to the processor, while at the same time making 8259A visible to the processor.

The two connection schemes described only differ in the number of 82489DX(s) (one or two) that are located in the path from the 8259A to the processor. In the one 82489DX example illustrated in Figure 37, the INT output of the 8259A connects to one of the Interrupt Input pins of the 82489DX through an edge generation logic. This could be an interrupt pin on the 82489DX's I/O unit or local unit; assume a local interrupt input is used. The Local Vector Table entry for the interrupt pin that connects to the 8259A is set up with a Delivery Mode of "ExtINT" and edge trigger mode. This indicates that the interrupt is generated by an external controller. The processor's INT pin connects to the 82489DX PINT pin.

This setup enables the 82489DX local unit to detect assertions (up-edges) of the 8259A's INT output pin and pass this on to the processor's INT input. 82489DX asserts ExtINTA pin along with (one clock prior to) PINT pin to indicate "8259" interrupt. When the processor performs its INTA cycle the 82489DX itself does not respond other than deasserting PINT to the processor. At the third clock after $\overline{ADS}$ in the second bus cycle of INTA cycle ExtINTA is deasserted. External logic should make use of the ExtINTA signal to make the INTA cycle visible to the 8259A and the 8259A should provide the vector. At the same time, the local unit considers the external request as delivered, and need not wait for the external 8259A's INT to be deasserted. *A new up-edge must be generated on the 8259A INT pin before the local unit will assert the processor's INT pin on behalf of the 8259A. External edge generation logic should be used for this.* Compatible software interacts directly with the 8259A.

The mechanism is essentially the same in the two-82489DX scheme. The difference is that the 8259A connects to an interrupt input pin of the 82489DX I/O unit in the I/O system. The Redirection Table entry for this pin is again programmed with an "ExtINT" Delivery Mode, and the (single) 82489DX destination local ID corresponding to the compatible DOS processor. Capturing the up-edges of the 8259A's INT pin by the 82489DX local unit now involves sending messages from the 82489DX I/O unit to the 82489DX local unit via the ICC bus. The "virtual wire" now includes messages over the ICC bus.

Adding inter-processor ICC interrupts (or any other 82489DX generated interrupts) to the compatible operation is accomplished by having the 82489DX internally OR the 8259A's INT request with any 82489DX interrupt request.

Before the 82489DX actually sends the interrupt signal to the processor, the 82489DX decides whether it does this for an 82489DX interrupt or whether it does this on behalf of the external controller. When the processor performs the corresponding INTA cycle, only the 82489DX knows whether it should respond with a vector, or whether the external 8259A should.

If the 82489DX needs to respond, then it will enable an externally implemented trap that prevents the 8259A from seeing the INTA cycle. If the 8259A needs to respond, then the 82489DX will not enable the INTA trap, and the INTA will be allowed to reach the 8259A. 82489DX implements this by asserting its EXTINTA pin to indicate external 8259A should respond with the vector. The 82489DX local unit controls the INTA trap via its "ExtINTA" output pin; the 82489DX does not actually provide the trap itself.

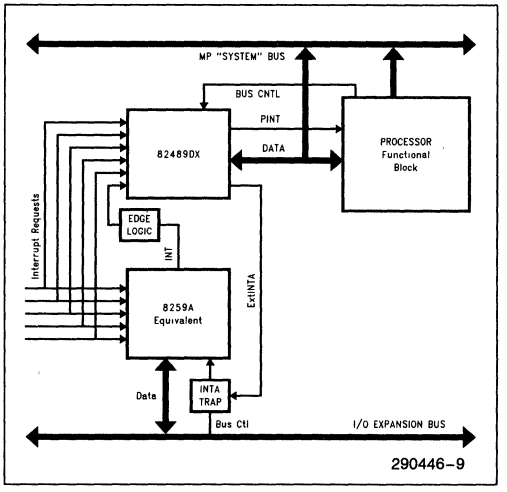


Figure 37. Edge Logic

82489DX/8259A DUAL MODE CONNECTION

In systems that can be booted either as a configuration with compatible 8259A or without, device interrupt lines are connected to both the Interrupt Request pins of the 8259A and Interrupt Input pins of the 82489DX with all interrupts either masked at the 82489DX or at the 8259A. Some EISA and Micro-Channel chip sets that include on-chip 8259As also have internally connected interrupt requests. For example, the 82357 (the ISP of the EISA chipset) generates timer and DMA chaining interrupts internally. These are not available as separate interrupts outside the ISP. In non-compatible mode the ISP timers are not used, since each local 82489DX unit provides its own timer. Therefore, the ISP's 8259A is configured to mask out all interrupts except the DMA chaining interrupt which is configured in level-sensitive, auto EOI mode. This causes the 8259A's INT output to track the state of the internal DMA interrupt request. The 8259A's INT output is then connected to one of the 82489DX interrupt input pins programmed to generate a regular (i.e., not "ExtINT") level-sensitive interrupt. The ISP 8259A then no longer functions as an external interrupt controller; it has been logically disabled, and it needs no interrupt acknowledge or EOI. The INTA and EOI cycles occur only at the 82489DX. It should be noted that 82489DX accepts only active high level/edge interrupt inputs. External programmable logic should take care of polarity reversal that may be needed in EISA system for sharing of interrupts.

intel®

AP-388

APPLICATION NOTE

82489DX User's Manual

2

December 1995

PRELIMINARY

Order Number: 292116-002

82489DX User's Manual

CONTENTS

CONTENTS

INTRODUCTION

82489DX is the new interrupt controller for high performance systems and 32-bit OS. Some important considerations for hardware designers are given. This application note will provide information of all registers in 82489DX and their bits and bytes organization. The control word for various programming options are given in a tabular format. Some programming hints are given to facilitate a quick understanding of the interrupt architecture and the priority model in 82489DX.

The programming model discusses the registers, their data structure like fields, bits, bytes and default register values. The system considerations and key points to be noted while programming 82489DX are discussed next. Typical examples of initialization, interrupt service routine and Spl() routines are given. The notes discuss important hardware design considerations.

Related Reference Materials

1) 82489DX Data Book, Order Number 290446.
2) An APIC based Symmetric Multiprocessor System Design AP-474, Order Number 241521.

REGISTER ORGANIZATION

The 82489DX contains both the local unit and I/O unit. I/O unit has its own Unit ID and local unit has its own Unit ID. Both units are operational at all times once they are enabled and the access can be done to both units. It should be noted that the local unit has its own version register, and I/O unit has its own version register, namely, I/O version register. The unit enable bit is provided for local unit and it is not provided for I/O unit. However, I/O unit has mask bit for each redirection table entry to mask the interrupts. Functionally I/O unit can only transmit interrupt messages whereas local unit can both transmit and receive interrupt messages. In summary, 82489DX should be viewed as an integrated chip having a local unit and an I/O unit both capable of operating at the same time.

INITIAL REGISTER VALUES AFTER HARDWARE RESET

The local unit ID register latches the value on the address pins A3 to A10 after hardware reset whereas the I/O unit ID register gets cleared to 0 after hardware reset. The local unit Version Register is cleared to 0 whereas the I/O unit Version Register contains 1111 in its Max Redir Entry field. The interrupt masks in the local timer vector table register and in the I/O redirection table entry(31:0) registers are set so that after reset all the interrupts are masked. The spurious vector register's unit enable bit is cleared so that local unit is disabled after hardware reset. Since all the interrupts are masked after hardware reset, the I/O unit will not transmit any interrupt after hardware reset until mask is cleared specifically by software and the interrupt is active.

All other registers are cleared to 0 after hardware reset.

SYSTEM CONSIDERATIONS WHILE PROGRAMMING THE 82489DX

The 82489DX register data structure contains different fields to specify the mode of operations and the options available within each mode. Since certain options are applicable to specific modes only (for example "Remote Read" mode applies only to Interrupt Command Register, it does not have any relevance to I/O unit's redirection tables) the following programming hints are provided.

82489DX and Memory Mapping

The 82489DX is a 32-bit high performance interrupt controller. It allows the CPU to do 32-bit read and write to it. By memory mapping the 82489DX, system performance can be enhanced. Even though the 82489DX can be memory mapped, its functionality as an interrupt controller should be kept in mind while programming the virtual memory management control data structure. The caching policy for the page where an 82489DX is mapped should also be done with the functionality of the 82489DX in mind. For example, the reads to an 82489DX should not be cached and writes should be write-through. Since 82489DX registers are aligned at 128-bit boundaries, memory mapping the 82489DX with interleaved memory system should not be a problem. However, it should be noted that the 82489DX does not support pipelining.

Unique ID Requirement

All the local units and I/O units hooked on an I_{CC} bus should have an unique ID before they can use the bus. This should be ensured by the programmer, since for I_{CC} bus arbitration the units (whether it is local unit or I/O unit) arbitrate with their unit ID.

PROGRAMMING THE LOCAL UNIT

Dos and Don'ts

1. The local interrupt vector table entry (and the I/O unit redirection table entry) should not be programmed for "Remote Read" Delivery mode. In other words, only Interrupt Command register supports "Remote Read" Delivery mode.
2. Local Interrupts should not be programmed with "Lowest Priority" Delivery mode.
3. Local Interrupts should not be programmed with "Reset" Delivery mode.
4. It is not recommended to use level triggered mode except for "Reset Deassert" messages.

Atomic Write Read to Task Priority Register

This section discusses issues regarding write buffer flushing and necessity of atomicity of task priority register programming.

Typically, the task priority register is written with higher priority to mask certain low level interrupts before entering into a critical section code. In a system where an 82489DX is memory mapped the CPU may buffer this task priority register write to its on chip write buffer. The following scenario can happen in such situation: CPU posts task priority register write to its on chip write buffer and enters into the critical code. A lower priority interrupt (which should not enter the critical code) interrupts the CPU before the write buffer gets flushed into task priority register). The CPU now erraneously accepts the lower priority interrupt. To avoid the situation, atomic write and read to task priority register should be done. The read following write ensures that the write buffer is flushed to task priority register and the atomicity ensures that no interrupt will be accepted by the CPU during its write to task priority. In case if the CPU itself takes care of flushing its write buffers before INTA cycle, there is no problem. However, if there are system posted write buffers then external logic should make sure to flush the system write buffers before INTA-cycle.

Task Priority Register and Total Usable Vectors

Task priority register is used to specify the priority of the task the processor is executing. In 8259 the priority is defined only among the interrupts that it handles. 82489DX goes further ahead in handling priority. In multitasking system, in addition to device interrupts, various tasks have different priority and 82489DX allows consideration of the priority at system level. The processor specifies the priority of the task it executes by writing to task priority register. Now any interrupts *at or below* the task priority will be masked until the task priority gets lowered. The masking granularity is at priority level. Out of 256 interrupt vectors 16 priority levels are specified and 16 vectors share one priority level. Since the masking granularity by the task priority register is at priority level, group of 16 vectors get masked when a local unit increases its task priority by one level.

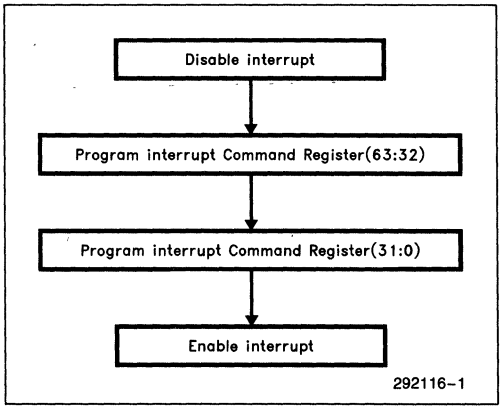
When task priority register is at its minimum level of 0, interrupt vectors having level 1 to 16 are passed to CPU. Stated in other words, even when the task priority register is at its minimum (level of 0), interrupt vectors at level 0 will be masked. This means that the interrupts should not be programmed with vectors 0 to 15. So out of 256 interrupt vectors, only 240 interrupt vectors (vector 16 to 255) can be used in 82489DX.

ISR/IRR/TMR

1. Bits 0–15 of IRR/ISR/TMR do not track interrupt. No interrupt of vector number from 0–15 can be posted. The total interrupts supported are 240. This can be easily explained by the way the priority mechanism is defined. When reading the lowest 32 bits of this register, 0 will always be returned for the lower 16 bits.

Interrupt Command Register Programming Considerations

The interrupt command register (31:0) has the side effect of sending interrupt once it is written. There is no mask bit associated with Interrupt Command Register. Once interrupt command register (31:0) is written, the interrupt is sent from the local unit. The interrupt destination is provided in the interrupt command register (63:32). So, the interrupt command register (63:32) should always be programmed before the interrupt command register (31:0) is programmed.



CRITICAL REGIONS AND MUTUAL EXCLUSION

This section discusses the reasons for mutual exclusion to be exercised when writing to interrupt command register. Each 82489DX has a single Interrupt Command Register that is used to send interrupts to other processors. The programmer should make sure to synchronize access to this register. Specifically, (1) writing all fields of the register (MSB), (2) Sending the interrupt message (by writing the LSB register), and (3) waiting for Delivery State to become Idle again, should occur as a single atomic operation. For example, if interrupt handlers are also allowed to send inter processor interrupts, then interrupt dispensing to the processor must be disabled for the duration of these activities so that interrupt handlers are excluded from accessing the ICR. This is explained as follows. Let us assume in a typical MP system preemptive scheduling (on another processor) is implemented by sending inter processor interrupts (IPIs). IPI can be also used for clock distribution in an asymmetric system where the timer interrupts only one processor and that processor notifies all other processors in the system through IPI. Inter processor interrupts are implemented by using interrupt command register. If we allow interrupts during writing interrupt command register the following erraneous operation may result. If interrupts are enabled (they should not be) during writing to interrupt command register, interrupts can come after writing to MSB portion and before writing to LSB portion of interrupt command register. Now in the interrupt service routine, if ICR is used (for distribution of interrupt to other processor(s), for example) then this ISR also starts writing to the Interrupt Command Register. That means the ISR will overwrite the MSB portion just written by the previous IPI. After returning from the ISR when the previous IPI continous writing to the remaining LSB portion, the message will be delivered to wrong address since MSB is modified by the module which interrupted. **The inference is that while accessing ICR interrupts should be disabled.** Also it should be noted that except for "Reassert Deassert Messages", IPI should only use edge triggered mode.

BUFFERING IN INTERRUPT COMMAND REGISTER

The Interrupt Command Register provides one level of buffering which should be kept in mind while programming an 82489DX. The ICR (Interrupt Command Register) becomes busy as soon as inter processor message is written into it. It hands the message over to I_{CC} bus transmit unit which in turn tries to send through I_{CC} bus. Since the ICR has passed the command to transmit Unit (whose responsibility is to send it through I_{CC} bus) it becomes free. The software before writing next inter processor message reads the flag to be free and writes next message. Thus there is a possibility of next message being written into the 82489DX before the first message is really sent out. The programmer should be aware of this.

INTERRUPT COMMAND REGISTER DO'S AND DON'TS

1. "ExtINTA" delivery mode should not be used for all destination shorthand.
2. "Remote Read" should always be programmed as "Edge" triggered interrupt.
3. "Remote Read" should always be programmed with physical Destination mode (and not with Logical Destination mode).
3. Only Fixed Delivery Mode should be used for "Self" destination shorthand. Stated otherwise, "lowest priority", "Remote Read", "Reset", "NMI" delivery modes do not apply for "Self".
4. For "All incl. self" and "All excl Self" destination shorthands, "Remote Read" delivery mode should not be used.
5. For "All incl. self" and "Self" destination shorthands "Reset" Assert mode should not be used.
6. For "All exclusive self" destination shorthand if "Reset ASSERT" delivery mode is used, it should be ensured at system level that only one processor executes this instruction at any time. To explain this, let us consider the following situation. Let us assume that two CPUs, CPU A and CPU B are executing "Reset ASSERT, All Exclusive Self". The message of CPU A puts every CPU except CPU A in reset state. After the message is written by CPU A it typically takes 2.9 μs for the message to flow through the I_{CC} bus to reach other local units to reset all other processors. Before this message resets, let us assume another processor also, say CPU B, issues the "Reset ASSERT, All Exclusive Self" message. The following CPU B message (which was sent out before CPU B itself got reset because of CPU A reset message) will reset every CPU, which will include CPU A, except CPU B. But CPU B will eventually get reset by the message sent by CPU A and CPU A will also get reset by the message sent by CPU B. Thus all the CPUs in the system goes into reset state and this is an irrecoverable state. To avoid this, only one processor should execute this instruction at any time. This can be achieved, for example, by spinlock or mutex implemented as shared variables between multiprocessors.
7. Messages could be sent out in "Logical" or "Physical" mode with destination ID of all 1's depending on the way Destination Mode entry is programmed. In brief, "All incl. self" and "All excl. self" supports both "Logical" and "Physical" addressing mode.
8. When destination shorthand (*i.e.*, broadcast) is used with "lowest priority" destination mode, then even though all participates in arbitrating for destination, only the lowest priority gets the message. So even though the addressing is broadcast since the destination mode is lowest priority only one gets the message.

9. When destination shorthand (*i.e.,* broadcast) is used with "Fixed" destination mode, then all the units get the message. So to send messages to all units Fixed destination mode should be used in addition to using destination shorthand.
10. It is recommended that all IPI messages, except for "Reset Deassert", use only edge triggered interrupt mode.

IPI THROUGH INTERRUPT COMMAND REGISTER

Interrupt command register can be used to send inter processor interrupt. Inter processor interrupts can be used for preemptive scheduling, TLB flushing, clock distribution, etc. IPIs can also used in asymmetric systems to pass certain work to another processor who has exclusive access to certain piece of hardware. Let us consider a dual processor system which uses only two 82489DX in the whole system. Since the local units should be accessible only by their respective processor a local unit should be selected only when the arbitrator grants the bus to its processor. Since 82489DX has common chip select for its local unit and I/O unit, for logical simplicity, system hardware may select a 82489DX when the corresponding processor is granted bus. Because of this, processor A can access only I/O unit and local unit that are available in its 82489DX. It is not possible to access the I/O unit of the other 82489DX. The same thing holds good for the other processor. Since I/O unit should be globally visible to both processors, there may be situations when a processor may want to access the other I/O unit. This is typically the case for enabling and disabling the I/O interrupt. IPI can be used to pass that task to the other processor which can access that I/O unit. This is just one example for using IPI.

ExtINTA INTERRUPT POSTING

ExtINTA interrupts are used to support 8259 in a 82489DX based system. The external interrupts (ExtINTA) are specific in their characteristics in that they do not have any priority relationship with rest of the interrupt structure. But when posting an interrupt to the processor, if both an external interrupt and a 82489DX interrupt are pending, 82489DX could post either one to the processor. In 82489DX implementation, it would post external interrupt whenever there is no other 82489DX interrupt that can be posted to the processor. It should be also noted that External Interrupts can not be masked by raising task priority. However, they can be masked by the mask bit in the table entry for that (ExtINTA) interrupt.

Since ExtINTA interrupts do not have any priority relationship, ISR and IRR bits are not maintained for external interrupts. As far as interrupt acceptance is concerned, if more than one ExtINTA interrupts are directed towards a local unit, that local unit treats all the ExtINTA interrupts directed to it as only one ExtINTA interrupt. This leads to an important point that in a system no more than one interrupt should be programmed as ExtINTA interrupt type with the **same destination**. However, it should be noted that there can be more than one ExtINTA type of interrupt in a system with each having different local unit as destination.

LOWEST PRIORITY

Under the lowest priority delivery method, the processor to handle the interrupt is the *one* in the specified destination with the lowest processor priority value. If more than one processor is at the lowest priority, then a unique arbitration ID is used to break ties. To have unique arbitration ID in the system (which is mandatory for the lowest priority algorithm to work) all the arbitration ID of local 82489DXs in the system should be in sync. On reset, arbitration ID is reset to zero by the hardware. Hence all the local units in the system after reset will have same arbitration ID (namely zero). For lowest priority arbitration to work properly we need to have unique arbitration ID in the system. This means after local unit IDs are written in all local units (obviously, each unit ID should be different from other IDs) a RESET DEASSERT message should be sent in ALL INCLUSIVE mode. The important side effect of RESET DEASSERT message is that it copies the unitID into the respective arbitration ID. Since unit IDs are unique, the RESET DEASSERT message ensures that the arbitration ID also are unique in the system. This RESET DEASSERT message should be sent before system is used for lowest priority arbitration.

The RESET DEASSERT message, if not sent, only once delivery semantics may not be guaranteed. If RESET DEASSERT message is not sent then all the arbID in the system will be same. When a message is sent in the lowest priority arbitration, the participating local units use their processor priority concatenated with arbitration ID to decide the destination. Processor priority is derived from the task priority. There is a chance that two local units can have same task priority depending on the code they are executing and thereby same processor priority. In addition since arbID are also same if RESET DEASSERT message WAS NOT sent, all the processors in the same priority may accept the message in lowest priority arbitration. This violates the only once delivery semantics. The inference is that RESET DEASSERT message in ALL INCLUSIVE SELF mode should be sent as part of initialization before enabling interrupt in the lowest priority destination scheme.

It should be noted that only once delivery semantics for a group destination is guaranteed only if multiple fixed delivery of the same interrupt vector are not mixed.

DISABLING LOCAL UNIT

Once the 82489DX is enabled by setting bit 8 of spurious vector register to 1, the user should not disable the local unit by resetting the bit to 0. The result will put the local unit in an inconsistent state. However, a local unit can be disabled by getting "reset" interrupt message from any other local unit across the I_{CC} bus.

ISSUING EOI

EOI, End of Interrupt issuing indicates end of service routine to 82489DX. Always the highest priority ISR bit which is set during INTA cycle gets cleared by EOI. This section discusses the relevance of EOI to the specific types of interrupts and its timing related to interrupt deassertion.

EXTERNAL INTERRUPTS AND EOI

External Interrupts (ExtINTA) should be programmed as edge type. INTA cycles to external interrupts are taken automatically as EOI by 82489DX. This is similar to AEOI, Automatic End of Interrupt of 8259A. So EOI should not be issued to 82489DX for ExtINTA interrupt servicing. For ExtINTA type of interrupts, there is no need to have interrupt service routines since the main purpose of ExtINTA interrupt itself is to have software transparency in the compatible mode. The existing interrupt service routines written for 8259 will be executed by the processor for ExtINTA interrupts.

SPURIOUS INTERRUPTS AND EOI

Spurious Interrupts do not have any priority relationship to other interrupts in the system. So IRR is not set for spurious interrupts. EOI should not be issued for spurious interrupts. It is advisable not to share the spurious interrupt vector with any interrupt.

If spurious interrupt vector is shared with some other interrupt then the following guidelines should be followed. If the source is spurious interrupt (for which the corresponding ISR is **not** set) then EOI should **not** be issued. If the source is a valid interrupt sharing the spurious interrupt vector (for which the corresponding ISR is set) then EOI should be issued.

NMI AND EOI

For NMI type of interrupt no IRR bit is set. So, obviously EOI should not be issued while servicing NMI type of interrupts.

PROGRAMMING I/O UNIT

Interrupt Sharing Considerations

Two different interrupts should not be programmed with the same interrupt vector. This means that each redirection table in a system should have unique vector. Interrupt sharing can be done electrically. Interrupts connected at different interrupt input pins of 82489DX CAN NOT share interrupt by having same vector. 82489DX does not support active low interrupts. So sharing interrupts should have polarity logic support externally.

I/O Unit and Priority

The 82489DX partitions its interrupt control function among two different units:

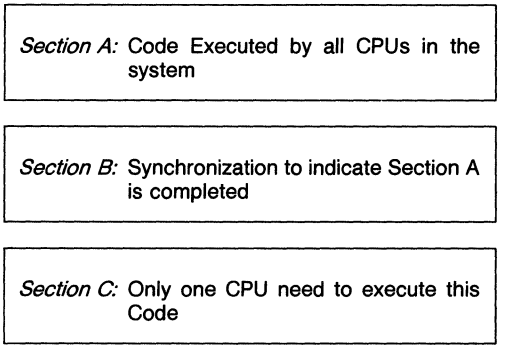
1. I/O unit
2. local unit

The priority resolving is done at local unit. The I/O unit does not involve itself in the priority mechanism. The I/O unit takes a snapshot of interrupts pending at the INTIN interrupt input pins. If interrupts are active, it starts sending the interrupt messages over I_{CC} bus. It starts sending the lowest numbered interrupt input first. That is if INTIN0 and INTIN5 are found active in a snapshot, interrupt message corresponding to INTIN0 is sent first regardless of the priority of the vectors that are associated with these interrupts. It sequentially sends all the interrupts found active in a snapshot. Before sending, it checks whether the corresponding INTIN is still active. **This is the reason why interrupts, both edge and level triggered, should be kept active until CPU acknowledges it.** The difference between edge triggered and level triggered interrupt is that edge triggered interrupts ensure only one activation of interrupt per low to high edge whereas the level triggered interrupt allows to have multiple interrupts as long as the interrupt is held high. It should be noted that both edge and level triggered interrupts are active high.

MP SYSTEM

Initialization Sequence

This section assumes the system with multiple CPUs with each CPU having its own 82489DX local units and local interrupts (like local secondary cache data parity interrupt, coprocessor interrupt etc.,) connected to the respective local units. The system additionally assumes symmetric multiprocessing in the sense that I/O system is symmetric and it can be initialized by any CPU in the system.

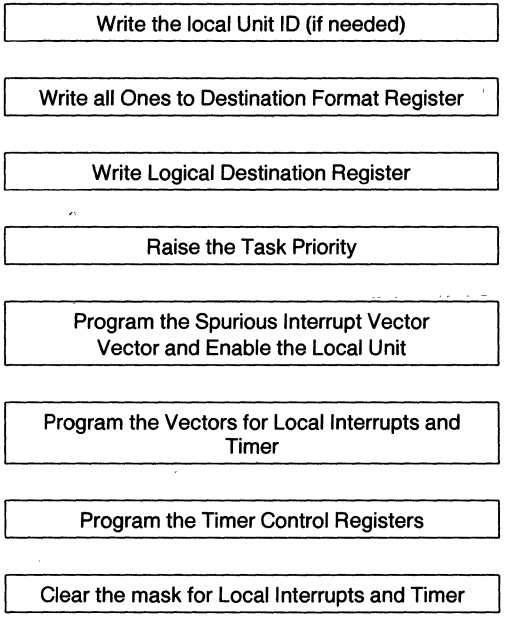


Each local unit is visible (through address mapping) only to that CPU to which local unit is attached. So each local unit will be programmed by its own CPU. Thus the code specified as section A will be executed by all CPUs in the system.

Section B of the initialization code is also executed by all the CPUs. This section of the code ensures that all the CPUs have completed execution of their "Section A" so that all Local units are properly initialized with different IDs, the system is in a consistent state, etc.,

Section C initializes system wide I/O unit and enables the interrupt mechanism to start functioning. Since the I/O unit is system wide, only one CPU need to program the I/O unit part of the 82489DX.

Section A



It should be noted that the interrupt descriptors, interrupt service routine, spurious interrupt service routine and other interrupt related structure should have been initialized before the Section A. This is because Section A code enables respective local interrupts and timer interrupt vectors and when the interrupts arrive from these devices Section A ensures that 82489DX will provide the vector. But the code executed before Section A should ensure the interrupt structure is initialized. Spurious interrupts are bound to occur because of the asynchronous interaction between interrupts and software writing to task priority register. So spurious interrupt service routine has to be initialized before Section A.

WRITE THE LOCAL UNIT ID (IF NEEDED)

Each local unit can get the ID latched by reset from 82489DX address pins A3–A10. If the hardware ensures that during reset each local unit in the system gets different pattern on the address pins A3–A10 then all the local units are initialized automatically with different IDs. In that case writing to the local unit ID by software is not mandatory. If software writes the local unit ID then it should be read from some address space which is same for all CPUs but have different IDs for different CPUs. This will ensure that the same code when executed by different CPUs will initialize respective local unit with different ID.

WRITE ALL ONES TO DESTINATION FORMAT REGISTER

All the 32 bits of Destination Format Register are written with 1. This is to support single level logical addressing mode. This mode is explained in the following paragraph.

WRITE TO LOGICAL DESTINATION REGISTER

The logical Destination Register should be written with the logical destination address. It should be noted that since each CPU needs to assign a different logical Destination address to its own 82489DX local unit and since this code is executed by all CPUs the logical Destination address should be read from some address space which is the same for all CPUs but contains different Destination address values. Since logical Destination address is in bit decoding format, typically this can be achieved by shifting the CPUID.

The logical destination register with Destination format register can be used to support flat model. In this model, bits 24 through 31 of the destination address of the interrupt message vector are interpreted as decoded field. Intel strongly recommends for future compatibility to use only bits 31 to 24 of logical destination register. To have binary compatibility with future APIC implementations, any code written for 82489DX should not use bits 0 to 23 of logical destination register. This field is compared against the logical destination register of the local unit. If there is a bit match (i.e., if at least one of the corresponding pair of bits of the destination field and logical destination register match) this local unit is selected for interrupt delivery. Each bit position in the destination field corresponds to an individual local unit. For future compatibility, only bits 0 to 23 of logical destination register should be zero. This scheme allows the specification of arbitrary groups of 82489DXs simply by setting the member's bit to one, but allows a maximum of 8 local units in the system since bits 0 to 23 of the logical destination register is zero. Broadcast to all is achieved by setting all 8 bits of destination to ones. This selects all 82489DXs in the system.

If more than 8 units are to be addressed in the system (and if future compatibility is not a major issue) then all the bits of the logical distination register can be used as a bit map thereby increasing the number of CPUs addressable in logical addressing to 32.

RAISE THE TASK PRIORITY

Before enabling the local interrupts and timer interrupts the task priority is raised to maximum priority in the system so that these interrupts are masked temporarily.

PROGRAM THE SPURIOUS INTERRUPT VECTOR AND ENABLE THE LOCAL UNIT

The spurious interrupt vector register is programmed with the corresponding vector. This vector will be pointing to a dummy routine with just an IRET. The unit is enabled so that the tristate pin PINT can come out of tristate state to pass the interrupts.

PROGRAM THE VECTORS FOR LOCAL INTERRUPTS AND TIMER

The interrupt vectors for Local Interrupts and timer are initialized with corresponding vector.

PROGRAM THE TIMER CONTROL REGISTERS

The timer registers such as divider configuration register, initial count, mode of operation and source of the timer clock are programmed.

CLEAR THE INTERRUPT MASK FOR TIMER AND LOCAL INTERRUPT

The interrupt mask is cleared for timer and local interrupts by clearing the interrupt mask bit in their respective interrupt vector register

INITIALIZE THE LOCAL INTERRUPT SOURCE

The local interrupt sources are also programmed for proper system operation. This involves enabling the interrupt from the sources. The order of enabling the interrupt is very important. First the 82489DX entries should be cleared and then the sources connected to the pins should be enabled. If done the other way, interrupts may get lost. This is true particularly in edge triggered interrupt inputs where if 82489DX mask is cleared after enabling the source interrupt, 82489DX may not have a chance to capture the low to high edge which might have produced immediately after the source interrupt is enabled and before 82489DX mask is cleared.

BROADCAST ALL INCL. SELF RESET DEASSERT MESSAGE

This is done so that all the local unit's ArbIDs are in sync. It should be noted that for breaking the tie during lowest priority arbitration ArbID is used. ArbID is copied from local unit ID during reset. Since local unit IDs can be written through software and at that time ArbID is not updated there may be a case where all ArbID in a system to have same value. To avoid such situation Reset Deassert message is sent to ALL INCL. SELF so that the ArbIDs are different in the system.

LOWER THE TASK PRIORITY

Task priority is lowered so that the interrupts can be armed to the CPU.

Section B

Synchronization

There are many methods available for synchronization. **Test - and - set** is a simple primitive, for example, available for synchronization. **Counting semaphores** can be built using this test - and - set primitive and synchronization can be achieved.

The main idea is to achieve global synchronization among the processors to indicate the local unit portion is programmed.

Section C

SYSTEM WIDE RESOURCES PROGRAMMING

This portion needs to be programmed by one CPU only. It should be noted that since the system environment we are assuming is shared memory symmetric MP system, CPU specific coding is not possible. System wide resource programming can be achieved by many ways depending on simplicity and performance (since this is only initialization routines, performance should not matter much) tradeoff. The following sequence illustrates a simple approach to program. The assumption here is that 82489DX will get reset both during cold reset and warm reset.

Locked access to the system wide resource, I/O unit

Read a **specific** MASK from Redirection Table Entry

If the mask is set, Jump to **Prog. I/O unit**

If the mask is not set, **Release lock** and Jump to **I/O unit Done**

Prog. I/O Unit: Write to the index register to select unit ID reg

Write I/O unit ID in the ID register

Write to index register to select I/O unit Version Reg

Read the Version reg. to know no. of RedirTable Entries, **N**

RedirTble: Write to index register to address MSB Redir.Table **n**

Write to MSB Redirection Table Entry **n** Destination local unit ID

Write to index register to address LSB Redir.Table **n**

Write to LSB Redirection Table **n** mode,dest.,mask, Vector of INT

Loop to RedirTble: till all N (here N = 16) Entries are done

Release the lock

I/O unit done: Remaining system init like I/O system etc.,

The first CPU getting the lock will find the mask to be set (since after reset, 82489DX mask is set). It locks the I/O unit for programming. The mask selected is specific in that the system initializes such that the mask is cleared during initialization. So by reading that mask mutual exclusion is achieved. If the system requirement is such that no mask can be cleared during system initialization some other register can be read. For example, the I/O unit IDs are reset to 0 on reset. Since the system initialization will have all the local unit IDs starting from 0 and I/O unit will be initialized by the system to non Zero ID, I/O unit can be read and if 0 can be assumed that programming is not yet done (so that it can gain control of lock and start programming) and if found non Zero, then that CPU can skip programming the I/O units by jumping to I/O unit done.

The I/O unit registers are organized as index register and data register. Other portions of section C are self explanatory. After I/O unit done, the I/O system initialization can be started so that the interrupts can start flowing in the system.

INTERRUPT SERVICE ROUTINE

ISR (x)

Save Stack and frame pointer for parameter referring

Save hardware context and software context

Service: Service the source, modify shared data structure etc.,

EOI: Issue EOI to reset ISR of level x in 82489DX

Restore hardware context and software context

Restore Stack and Return from Interrupt

The above ISR x shows the interrupt service routine of interrupt level x. The stack, hardware context like CPU registers and software context like task specific variables are saved. The Servicing is done as specific to the interrupting source. This may involve reading a status register or initiating a thread to read a "full" buffer, initiating a thread to write data to some "empty" register, or acknowledging an interrupt from another CPU. This is the point at which the interrupting source is supposed to deactivate its request. Next EOI is issued to reset the ISR bit corresponding to the interrupt level x. Till now the interrupts from same level and lower levels were masked. Once EOI is written, interrupts from all the levels can start coming. The hardware context and software context are restored followed by stack cleaning and a proper Return from Interrupt is executed.

There are couple of timing issues that can be considered here. The time delay between **Service** and **EOI** is referred here. This timing and its relevance to edge/level triggered interrupt is discussed as follows: In the case of edge triggered interrupts, for each edge one Interrupt message is sent by I/O unit to local unit over I_{CC} bus whereas for level triggered interrupts there are two interrupt messages sent, one during assertion of level interrupt, and another during deassertion of level interrupt. In edge triggered interrupts, since the deassertion of interrupt does not result in any interrupt message, there are not many issues with the timing delay between **Service** and **EOI**, even though in general delaying EOI means interrupts from the same interrupt source are kept pending from interrupting CPU. In level triggered interrupts after **service** the I/O device starts deasserting its interrupt request. This results in an interrupt message to clear IRR bit in the local unit. This may take some time because the minimum possible time in I_{CC} bus is 2.3 μs (10 MHz I_{CC} clock assumed). If the I_{CC} bus is occupied by some other messages already then this IRR clearing message has to wait to get its turn which means additional delay. If EOI is issued before this happens then ISR gets cleared and IRR for this "done interrupt" is still alive to erroneously set ISR again. This will result in another interrupt. So "Early Servicing" is advisable in level triggered interrupts.

DOS Environment

In the DOS environment the initialization portion is the only routine to be coded since the 82489DX acts as a virtual wire once initialized and needs no more programming. Since it is uniprocessor environment there is no need for synchronization.

The interrupt from 8259 is programmed as type ExtINTA and other redirection table entries are not accessed since their masks are set by reset and hence disabled.

In the Interrupt service routine, since EOI is not needed for ExtINTA type of interrupts, no programming is needed for 82489DX. Since ExtINTA type of interrupts do not have any relationship to task priority, Spl routines do not apply for DOS configurations.

Transition from 8259 to 82489DX

Typically, platforms with 82489DX will support 82489DX in virtual wire mode. The BIOS in the EPROM will program the 82489DX in "Virtual Wire" mode. Typically systems boot DOS and then the 32 bit high performance OS is given control. There are also situations where after BIOS code is executed the high performance OS is given control. In both the situations, the 8259 will be operational during the DOS or BIOS portion of the code and interrupts will be flowing in the system. When the high performance OS is given control, it may want to disable the interrupt during initialization. This will involve disabling 8259. After disabling 8259, the 32 bit OS initializes and then it may want to enable interrupt mechanism which involves enabling 82489DX. The sequence we are encountering here is 8259 (and one input of 82489DX enabled in "ExtINTA" mode) enabled, 8259 disabled and then 82489DX enabled. When 82489DX is enabled in 32 bit OS all the interrupt inputs are enabled as opposed to the only one interrupt enabled in "Virtual Wire" mode. The additional difference is that 82489DX is no more a "virtual wire" but it is functioning as an interrupt controller.

In the above situation, consider the following scenario. The 82489DX is functioning as "virtual wire" and passing the 8259 interrupts as "ExtINTA" mode to the local unit. When interrupt mechanism is disabled by CLI (Clear interrupt) or masking the 8259 interrupt, there may be a possibility that already 8259 originated interrupt may be pending at the local unit asserting interrupt to the CPU. Now since the CPU has executed CLI, the interrupt is not serviced and the interrupt is kept pending. It should be noted that the pending interrupt is of type "ExtINTA". After this, 32 bit OS gets loaded which configures 82489DX redirection tables and interrupt is enabled. Now the "old pending" interrupt is delivered and since it is "ExtINTA" the external hardware will typically pass the interrupt acknowledge cycle to 8259. But at this point of time 8259 has been masked by 32 bit OS. Hence the "masked" 8259 responds with IR7 vector. So the 32 bit OS should reserve IR7 vector for both master and slave 8259 for "emptying" the old pending interrupt since the "Virtual Wire" remembers the previous interrupt.

Sequence of Enabling: In the case of enabling interrupt controllers in "ExtINTA" mode the 82489DX should be enabled before the 8259 interrupt Controller is enabled. This is because ExtINTA is "edge triggered" and if 8259 is enabled before 82489DX, 8259 might have

given an interrupt request by activating its interrupt output while 82489DX is still not enabled. When 82489DX is enabled the interrupt input has a high level and 82489DX had no chance of capturing the low to high edge of 8259.

Spurious Interrupt Service Routine

It is advisable not to share spurious interrupt vector with any genuine interrupt source. This section assumes that spurious interrupt vector is not shared with any other interrupt.

82489DX does not set ISR in response to spurious interrupt, NMI type of interrupt, Reset type of interrupt and ExtINTA type of interrupts. For all these interrupts EOI should not be issued.

Some systems have a variable count in the supervisor data structure to count number of spurious interrupts raised in the system. This can be used to study the reliability and "noise level" of the system. But in 82489DX architecture, spurious interrupt can occur even by a dynamic write to task priority register, frequency of spurious interrupt does not mean anything related to "noise level".

Return from interrupt

Spl(x) Routines

The processor handles the I/O system through device driver interface. The device driver consists of two entries to access the I/O system: 1) Call entry and 2) Interrupt entry. The interrupt entry is the one that we have been discussing for a while, i.e., interrupt service routine. The Call entry is the way the I/O system is accessed to initiate and service devices. The call entry has its own task priority and interrupt entry has the priority that is associated with the device interrupt level. The call entry and interrupt entry processes have I/O data structure like linked list, buffer pointers in common which they share. Mutual exclusion is needed to ensure the integrity of I/O system.

To ensure the mutual exclusion between these two processes running in the same processor, Spl(x) routine is used. The call entry routine (which is normally at a lower priority than the interrupt entry routine) calls Spl(x) routine to elevate its own priority above (or equal to) that of the corresponding device's interrupt priority. At this priority the interrupts from the device are masked out and the shared data structures can be accessed (exclusively).

Once this is done the priority is restored back to original value so that other interrupts won't suffer for relatively long time.

Spl() is used to save the current task priority and Spl(x) is used to elevate the task priority.

Spl()

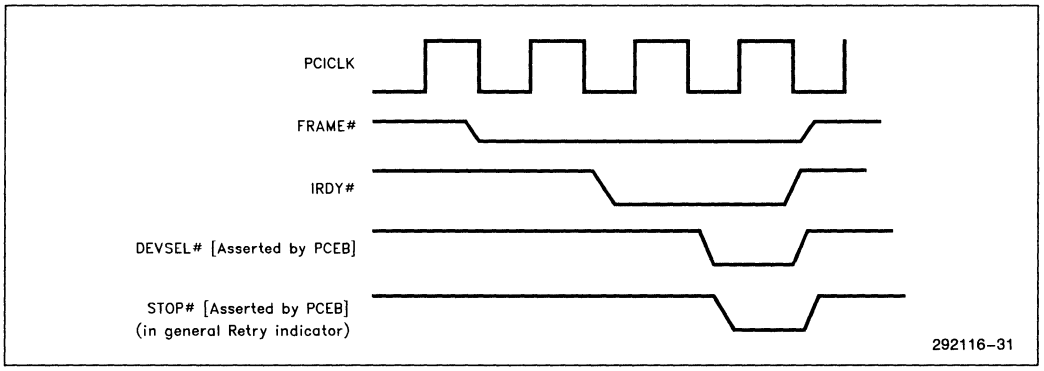
Read and return the 82489DX local unit task priority register

Spl(x)

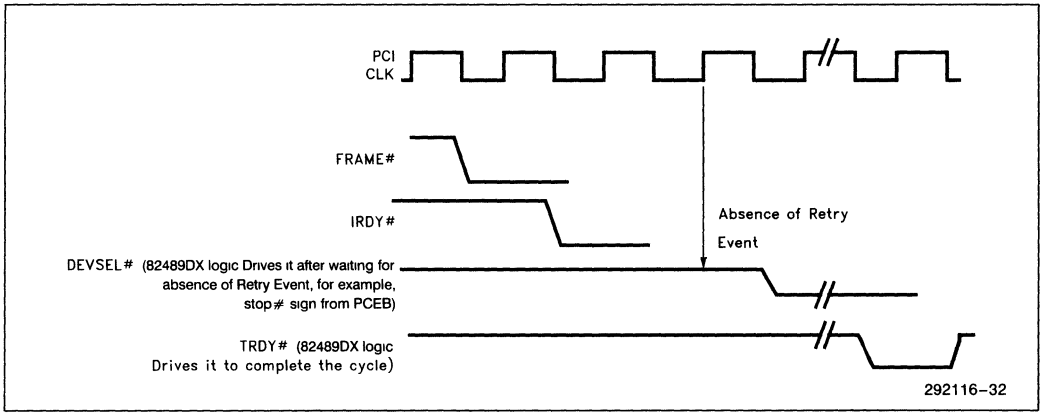
Write x to task priority register to raise priority to x

Typical usage of these routines

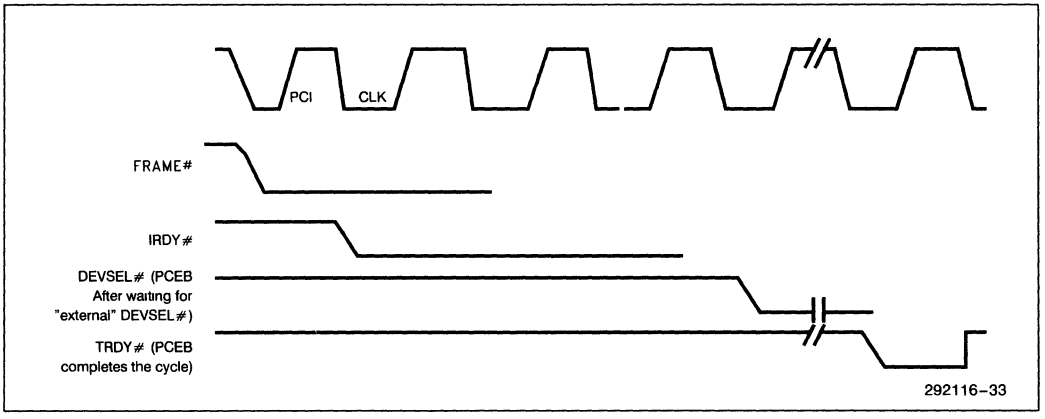
```
y = spl()    //Save the current task priority register value//
spl(x)       //Raise the task priority value                 //
:
:
:            // Access the shared data structure             //
spl(y)       // Restore the task priority register           //
```



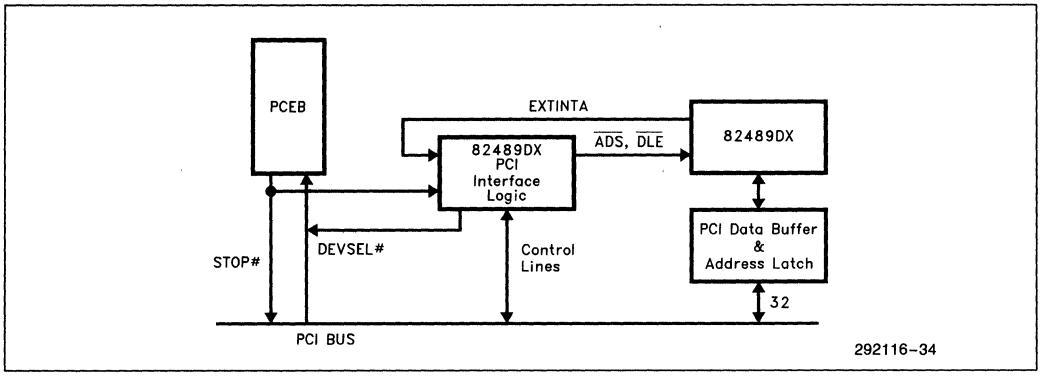
Case 1) Any INTA# Cycle Buffer Management and Retry



Case 2) Interrupt Acknowledge Cycle
Source of the Interrupt and Vector is 82489DX



Case 3) Interrupt Acknowledge Cycle
Source of the Interrupt and Vector is ESC 8259



82489DX-PCI Interface

82489DX AND PCI-EISA BRIDGE INTEROPERABILITY

82489DX gives performance benefits in multithreaded operating systems. Thus in both uniprocessor and multiprocessor systems 82489DX enhances the system level performance. This is because of its advantage in task priority management. Intel's PCIset EISA bridge component PCEB (B-stepping onwards) can interoperate with 82489DX. In this section, we will go over the "hook" provided by PCEB to connect 82489DX in a PCI system with some external glue logic.

From the Interrupt acknowledge timings of PCEB (B-stepping onwards) it can be inferred that **whenever the internal data buffers are empty**, on an interrupt acknowledge cycle, PCEB waits one clock cycle so that PCI interface logic for 82489DX can activate DEVSEL#. But, if the internal data buffers are not empty, then PCEB drives STOP# to retry the INTA cycle so that it can flush the buffers.

If DEVSEL# is seen active and if the PCEB's internal data buffers are empty, then PCEB allows 82489DX to own the INTA cycle. Thus, the external 82489DX glue logic, on an INTA cycle, should first sample STOP#. If STOP# is driven, then the glue logic should ignore the cycle. Because PCEB has some data in the buffers it wants to flush them before INTA cycle is run. So, the 82489DX glue logic should not start the cycle to 82489DX. If STOP# is not active and if the "ExtINTA" pin from 82489DX is inactive (to indicate that the cycle is for 82489DX) then it should drive DEVSEL# immediately to own the INTA cycle. At the same time it can start the cycle to 82489DX. It should be noted that 82489DX needs two INTA cycles whereas PCI bus has only one INTA cycle. So, the external logic is responsible for splitting one PCI INTA cycle into two 82489DX INTA cycles back to back but pass only one READY to the system.

During INTA cycle on PCI bus if "ExtINTA" pin is active (to indicate that it is 8259 INTA cycle), then the 82489DX glue logic should not drive DEVSEL#. Thus by finding DEVSEL# inactive, the PCEB will respond to the INTA cycle.

Thus with minimal external glue logic, it is possible to design an APIC based PCI system. Since PCI local bus will improve the I/O performance of the system, APIC will enhance the improvement of the overall system performance in a multithreaded environment.

HARDWARE DESIGN CONSIDERATIONS

Design Consideration 0

Any edge triggered interrupt creating an active edge while the interrupt is masked at 82489DX is lost. The 82489DX samples the edge triggered interrupt input only when it is unmasked. If an edge occurs while the interrupt is masked, that interrupt is lost. The software should always unmask the interrupt at 82489DX and then enable at the device. By this, it is made sure that 82489DX will have a chance to find the active going edge.

Design Consideration 1

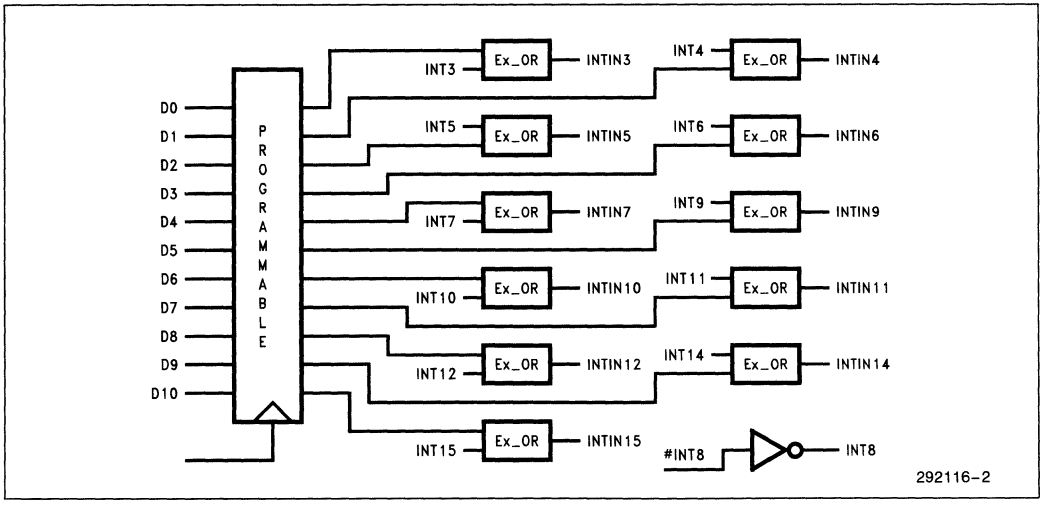
Description: The following design consideration has to be taken care of when using ISP (82357) as external interrupt controller. 82489DX allows connecting external 8259 type interrupt controller at one of its inputs. The mode associated with the interrupt input which has 8259 connected to it is called ExtINTA mode. 82489DX allows only EDGE TRIGGERED program-

ming option for ExtINTA mode. But in the case of 82357, the INT output from ISP stays high in case more than one interrupt is pending at its inputs. It does not always inactivate its INT output after INTA cycle. This will lead to a situation where ISP keeps the interrupt at high level continously and waits for INTA cycle. But since 82489DX expects an edge for interrupt sensing (for ExtINTA interrupts) it does not pass the interrupt to CPU and further interrupts are lost. So External circuitry should monitor the end of SECOND CYCLE of INTA cycle and force an inactive state at 82489DX's input. This can be done by ANDing ISP's output with a forced brief low going pulse at the end of second INTA cycle. This will generate an edge for each interrupt at 82489DX's input. For more refined edge generating logic, refer to data book, Order Number 290446.

Design Consideration 2

Description: The following design consideration has to be taken care of when using 82489DX in EISA systems. EISA ISP(82357) chip integrates 8259A. It additionally allows sharing of interrupts. To facilitate this sharing it has a programmable register, ELCR (Edge / Level trigger control register) by which certain interrupt inputs can be programmed as edge (low to high except for RTC) or level (the level is active low). The determination of edge or level is done during initial configuration of EISA system by reading EISA add in boards from the interrupt description data structures. The solution is to have programmable logic at the interrupt inputs so that 82489DX is compatible with EISA ISP. This will introduce one more register and logic to support this. This should be an 11 bit programmable register and an array of ExOR logic (12 ExOR gates or equivalent PLD). The ISP allows programmability of the following interrupts. It is highly recommended to use the same address of ELCR (and also bit definitions) for this polarity register, if possible, so that when ELCR is written this register will also be written. By this there is no separate programming needed for this polarity register. This will help to maintain compatibility with future APIC implementations which may use the existing ELCR register itself for polarity control. This is true for integrated APIC.

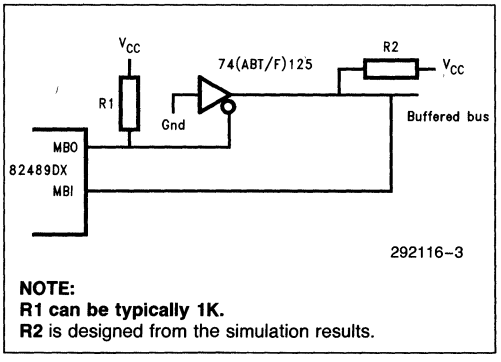
INT3 INT4 INT5 INT6 INT7 INT9 INT10 INT11 INT12 INT14 INT15. In addition to the above 11 interrupts, it fixes **INT8** to be active low edge triggered interrupt. INT8 is the only case where it is active low edge triggered type. So the following logic can be used to add programmability in 82489DX based EISA system. Before connecting these 11 interrupt lines directly (#INT8 which is from Real Time Clock is always active low edge triggered. #INT8 can be passed through an inverter since there is no need for programmability) to the 82489DX they should pass through an array of 11 Ex__OR gates. One input of Ex__OR gate connects to the corresponding INT pin and other input connects to a bit of programmable register. The output of Ex__OR gate is connected to 82489DX. The idea of Ex__OR is to use as a controlled inverter.



INTIN are the interrupt inputs to the 82489DX and INT are the system interrupt. The Ex_OR gating register is programmed after EISA configuration is found from add in boards as how these interrupt lines are going to be used in that particular configuration. If a particular input is edge triggered, then the corresponding bit in the register is written with 0. If a particular input is level triggered, then the corresponding bit in the register is written with 1.

Design Consideration 3

I_{CC} bus drive is an open drain bus with drive capacity of 4 mA only. Since data is transmitted at each I_{CC} clock, the "charging" of I_{CC} bus should be fast enough to ensure proper logic level at each clock edge. The I_{CC} bus needs pull up resistors since it is open drain bus. Since the drive is only 4 mA, the pull up resistor value can not be less than 5V/4mA. This being the limit of the resistor value, the length and the characteristics of the I_{CC} trace forces a capacitance value. Both the resistor and capacitance brings a RC time constant to the I_{CC} bus waveform. So, Electrical consideration has to be given to and practice of controlled impedence should be exercised for layout of the I_{CC} bus. The length of the trace should be kept as minimum as possible. If the length of the I_{CC} bus can't be kept less, than say 6 inch, because of mechanical design of the system, the external line drivers should be added to I_{CC} bus and I_{CC} bus should be simulated with the added driver characteristics.



NOTE:
R1 can be typically 1K.
R2 is designed from the simulation results.

Design Consideration 4

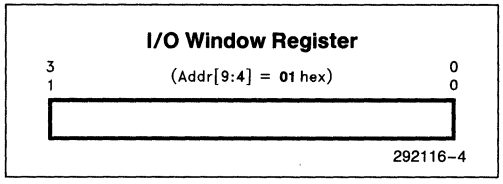
This is related to ADS#, BGT# and CS# timings. For bus cycles not intended for 82489DX, (CS# = 1 where 82489DX is supposed to sample it), any change in CS# line while the ADS# is still active, may erroneously cause a RDY# returned from 82489DX. Anomolous behavior may result if for BGT# ties low cases

a) BGT# goes away just one clock after ADS# or

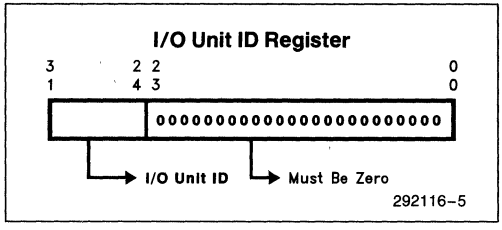
b) ADS# is still active, and CS# changes during this period.

For other cases anomolous behavior results if CS# changes when ADS# is still active. The following considerations are important from timing point of view. Always limit the pulse width of 82489DX ADS# to one CLKIN. Also avoid changing levels on BGT#/CS# line, when ADS# is active for cases being identified as BGT# tied low (BGT# sampled low when ADS# goes active). Also avoid changing levels on CS# line when BGT# is active.

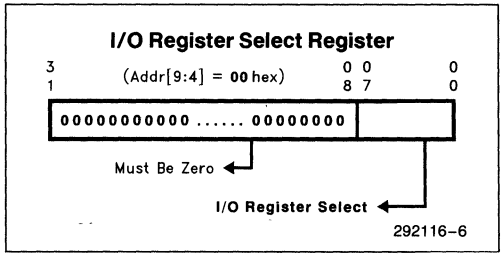
REGISTER PROGRAMMING DETAILS



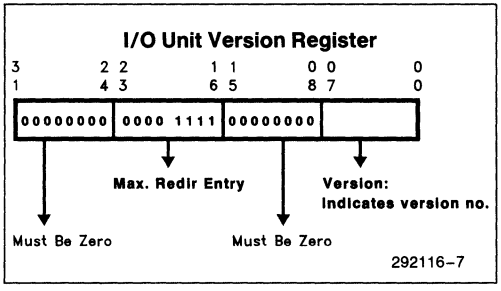
Data access to the register selected by I/O register select Register.



For example, for a Unit ID of 0A hex, the I/O unit ID register should be written with **0A00 0000.**



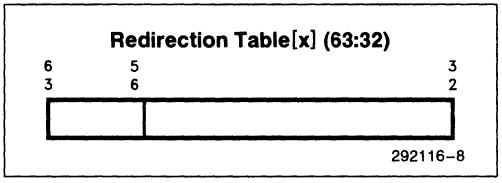
I/O Register Select	Register Selected
00 hex	I/O Unit ID Register
01 hex	I/O Unit Version Register
10 hex	Redirection Table[0] (31:0)
11 hex	Redirection Table[0] (63:32)
12 hex	Redirection Table[1] (31:0)
13 hex	Redirection Table[1] (63:32)
14 hex	Redirection Table[2] (31:0)
15 hex	Redirection Table[2] (63:32)
•	•
•	•
1E hex	Redirection Table[7] (31:0)
1F hex	Redirection Table[7] (63:32)
20 hex	Redirection Table[8] (31:0)
21 hex	Redirection Table[8] (63:32)
•	•
•	•
2E hex	Redirection Table[15] (31:0)
2F hex	Redirection Table[15] (63:32)



I/O Unit Version Register is read only register. It reads as **000F 00XX** where XX is version.

Max.Redir Entry: This is equal to the number of interrupt input pins minus 1 of this I/O unit. Read as 15 in 82489DX.

Version: The version number that identifies this version.



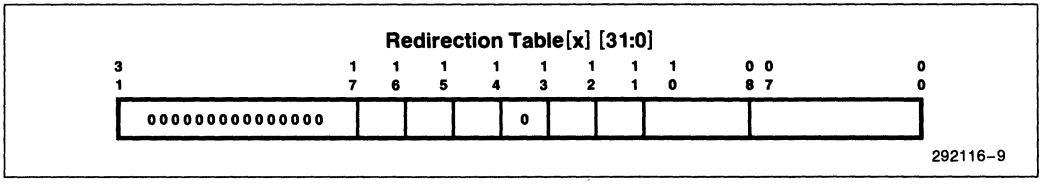
Destination: If the destination mode of this entry is "Physical Mode", then the 8 MSB (bits 56 through 63) contain an 82489DX local unit ID.

If logical mode, then all the 32 bits (bits 63 through 32) of the Destination field potentially defines a set of processors.

NOTE:
The same format holds good for Redirection Tables 0 to 15 (x = 0 to 15) for bits 63 to 32.

If the destination is to a local unit with ID, say, 05 in physical mode, then the redirection table [63:32] should be programmed as hex **0500 0000.**

Redirection Table [63:32] should be programmed first before programming Redirection Table [31:0].



Bits [31:17]: Reserved. Should be written 0.

Bit 16: MASK
0 — Not masked
1 — Masked

Bit [15]: TRIGGER MODE
0 — Edge Triggered
1 — Level Triggered

Bit 14: Remote IRR Status (Read only)
0 — Remote IRR is clear
1 — Remote IRR is set

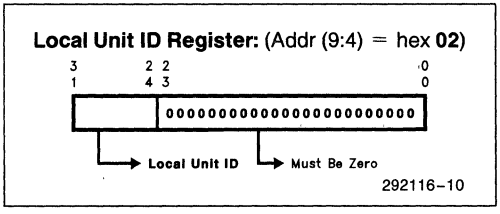
Bit [13]: Reserved. Should be written 0.

Bit 12: Delivery Status (Read only)
0 — Idle
1 — Send Pending

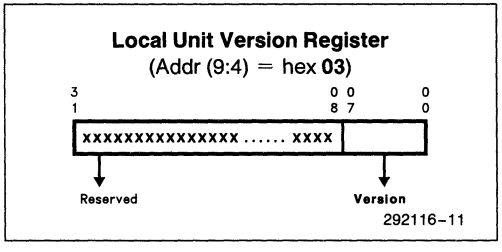
Bit [11]: Destination Mode
0 — Physical
1 — Logical

Bits [10:8] Delivery Mode
000: Fixed
001: Lowest Priority
100: NMI
101: Reset
111: ExtINTA

Bits [7:0] Vector
Vector for this interrupt

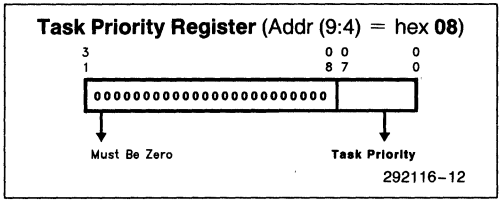


For example, for a local Unit ID of 0A hex, the local unit ID register should be written with **0A00 0000.**

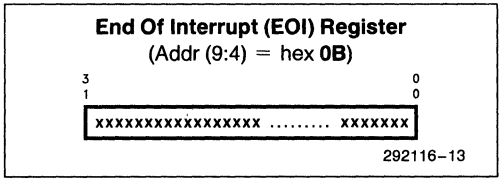


Local Unit Version Register is **read only register.** It reads as **0000 00YY** where YY is version number.

Bits [7:0] Version: The version number that identifies this version.

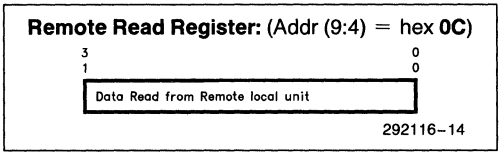


Bits [0:7] Task Priority: Should be written with task priority.



Bits [31:0]: Data written to EOI is don't care.

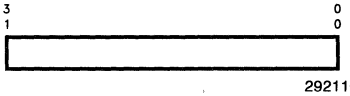
Before returning from the interrupt handler, software must issue an End-Of-Interrupt (EOI) command to the 82489DX local unit. For NMI and ExtINTA and Spurious interrupts EOI SHOULD **NOT** be issued.



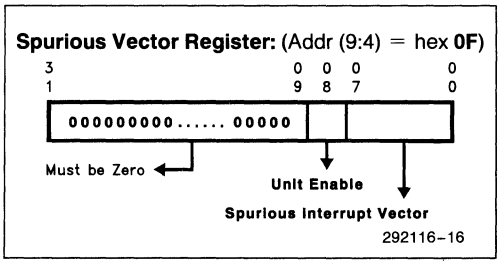
The data read from remote local unit is latched in Remote Read Register. The software should qualify this data with "Remote Read Status bit" in the ICR register.

Interrupt Status Register [ISR]:

Register	Address[9:4]
ISR[31:0]	hex 10
ISR[63:32]	hex 11
ISR[95:64]	hex 12
ISR[127:96]	hex 13
ISR[159:128]	hex 14
ISR[191:160]	hex 15
ISR[223:192]	hex 16
ISR[255:224]	hex 17



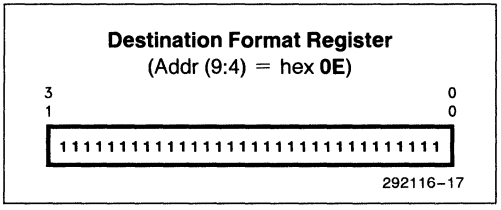
Interrupt Status Register is read only. It marks the interrupts that have been delivered to the processor and waiting for EOI.



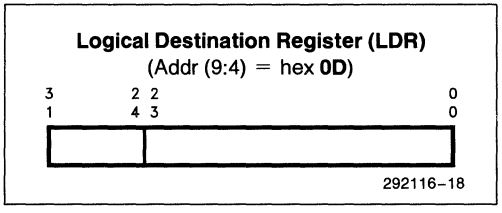
Bits [31:09]: Reserved bits. Must be zero.

Bit 8: Unit Enable: When this bit is 0, the local unit is disabled with regard to transmit and responding messages on I_{CC} bus. It only responds to messages with delivery mode set to "Reset". Reading a 0 at this bit indicates that the unit is disabled. When a 1 is written to the bit, the local unit is enabled for both transmitting and receiving messages. **Once enabled, it should not be disabled by software. Only further resets can take the unit into disabled condition.**

Bits [7:0] Spurious Interrupt Vector: For future compatibility, the bits [3:0] should be written with 1111. A spurious interrupt service routine should be existing in the address corresponding to the spurious interrupt vector.



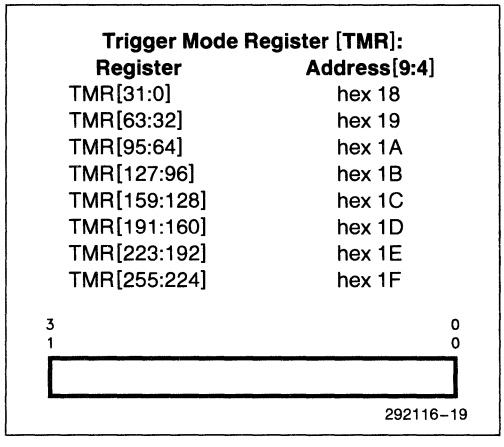
The destination format register enables logical addressing by specifying the bit map in logical destination register. For future compatibility, all the 32 bits of Destination Format Register should be 1.



Each local unit can be addressed either physically using physical ID or logically using logical destination register. In physical addressing, either only one local unit can be addressed at a time or broadcast to all local units can be done. In logical addressing, a group of local units can be addressed through bit mapping in destination addressing and the logical destination register.

For future compatibility, bits [0:23] of the logical destination register and bits [0:23] of the destination address in the message should be zero. Bits 24 through 31 of destination information in the interrupt message received are interpreted as decoded field. This field is compared against the logical destination register of the local unit. If there is a bit match (i.e., at least one of the corresponding pair of bits of the destination field and LDR match) that unit is selected for interrupt delivery. Each bit position in the destination field corresponds to an individual Local unit. This scheme allows the specification of arbitrary groups of local units by setting the member's bits to 1, but allows a maximum of 8 local units in a system since only bits 24 through 31 (of the Logical Destination Register and logical destination address in interrupt message) are used. Broadcast to all is achieved by setting all 8 bits of destination to ones. This selects all local units in the system.

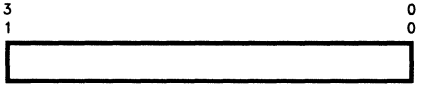
In a very large multiprocessor system where future compatibility is not a main problem, all the 32 bits of the Logical Destination Register and all the 32 bits of the destination address in the interrupt message can be used as a bit map to address the processors. When message addresses the destination using logical addressing scheme, the local unit compares the logical address in the interrupt message with its own logical Destination Register. Thus it is possible to support 32 processors in logical addressing mode.



If a bit corresponding to an interrpt vector number is 0, then it is assumed as edge triggered interrupt. For edge triggered interrupt, the corresponding IRR bit is automatically cleared when interrupt service starts. If 1 (level triggered) this is not the case. Instead, the source 82489DX (source I/O unit or Source Local Unit) must explicitly request the IRR bit be cleared (upon deassert of the interrupt input pin or upon sending an appropriate interprocessor interrupt). Upon acceptance of interrupt, the TMR bit is cleared for edge triggered interrupts and set for level triggered interrupts. This information was carried in the accepted interrupt message. The source 82489DX I/O unit also tracks the state of the destination unit's IRR bit (Remote IRR bit in the redirection table). When a level triggered interrupt input is deasserted, the source 82489DX I/O unit detects the discrepancy between the input pin state and the Remote IRR, and automatically sends a message telling destination 82489DX to clear IRR for the interrupt.

Interrupt Request Register [IRR]:

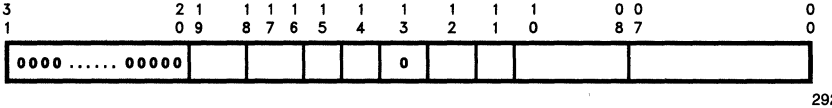
Register	Address [9:4]
IRR[31:0]	hex 20
IRR[63:32]	hex 21
IRR[95:64]	hex 22
IRR[127:96]	hex 23
IRR[159:128]	hex 24
IRR[191:160]	hex 25
IRR[223:192]	hex 26
IRR[255:224]	hex 27



292116-20

It contains the active interrupt requests that have been accepted, but not yet dispensed by this 82489DX local unit. A bit in IRR is set when 82489DX local unit accepts the interrupt. When TMR is 0, it is cleared when the interrupt is serviced; when TMR is 1, it is cleared when the 82489DX local unit receives a message to clear it.

Interrupt Command Register [31:0] (Addr [9:4] = **30** hex)



292116-21

Bits [31:20]: Reserved. Should be written 0.

Bits [19:18]: Destination Shorthand. This field indicates whether a shorthand notation is used to specify the destination of the interrupt and if so, which shorthand is used. Destination shorthands do not use the 32-bit Destination field, and can be sent by software with a single 32-bit write to the 82489DX's interrupt command register. Shorthands are defined for the following cases: Software self interrupt, interrupt to all processors in the system including the sender, interrupts to all processors in the system excluding the sender.

00: (dest field) means that no shorthand is used. The destination is specified in the 32-bit Destination field in the second word (bits 32 to 63) of the interrupt control register.

01: (self) means that the current local unit is the single destination of the interrupt. This is useful for software interrupts. The destination field in the interrupt command register is ignored. RESET assert Delivery mode should not be used with self destination. Only FIXED delivery mode should be used with SELF.

10: (all incl. self) means that the interrupt is to be sent to "all" processors in the system including the processor sending the interrupt. The 82489DX will broadcast a message with destination unit ID field set to all ones. RESET assert Delivery mode should not be used with "all incl. self" destination.

11: (all excl. self) means that the interrupt is to be sent to all processors in the system excluding the processor sending the interrupt. The 82489DX will broadcast a message with destination unit ID field set to all ones.

Bits [17:16]: Remote Read Status. This field indicates the status of the data contained in the Remote Read register. This field is read only to software. Whenever software writes to the interrupt command register using Delivery mode "Remote Read" the Remote Read Status becomes "in progress" (waiting for the remote data to arrive). The remote 82489DX local unit is expected to respond in a fixed amount of time. If the remote 82489DX local unit is unable to do so, then the remote read status becomes "invalid". If successful, the Remote Read status resolves to "Valid". Software should poll this field to determine completion and success of the Remote Read command.

00: (invalid): The content of the Remote Read register is invalid. This is the case when after a Remote Read command is issued and the remote 82489DX Local unit was unable to deliver the Register content in time.

01: (in progress): a remote read command has been issued and this 82489DX is waiting for the data to arrive from remote 82489DX local unit

10: (valid): the most recent Remote Read command has completed and the remote read register content is valid.

11: reserved.

Bit [15]: TRIGGER MODE

0 — Edge Triggered

1 — Level Triggered

Software should use this bit in conjunction with Level Assert/Deassert to generate interrupts that behave as edges or levels. **For future compatibility, send ICR messages only in edge triggered mode.**

Bit [14]: LEVEL. Software should use this bit in conjunction with the Trigger mode bit when issuing an inter-processor interrupt to simulate assertion/deassertion of level sensitive interrupts.

To assert: Trigger mode = 1 and Level = 1.

To deassert: Trigger mode = 1 and Level = 0.

For example, a message with Delivery mode of "Reset", a trigger mode of "Level", and Level bit of 0 deasserts reset to the processor of the addressed 82489DX Local unit(s). As a side effect, this will also cause all 82489DX to reset their Arbitration ID to their unit ID. (The Arb ID is used for tie breaking in lowest priority arbitration.) **For future compatibility, only edge triggering should be used in ICR.**

Bit [13]: Reserved. Should be written 0.

Bit [12]: Delivery Status (Read only)

0 — Idle

1 — Send pending

Delivery status is software read-only. Software can read to find out if the current interrupt has been sent, and the Interrupt command register is available to send the next interrupt. If the interrupt command register is overwritten before the Delivery status is "idle", then the destiny of that interrupt is undefined; the interrupt may have been lost.

Bit [11]: Destination Mode

0 — Physical

1 — Logical

In physical mode, a destination 82489DX is identified by its Local Unit ID. Bits 56 through 63 (8 MSB of the destination field) specify the 8-bit 82489DX Local unit ID.

In logical mode, destinations are identified by matching on Logical Destination under the control of the Destination Format Register in each Local 82489DX. The 32-bit Destination field is the logical destination. **For future compatibility, use only bits [31:24] of the logical destination address.** Bits [23:0] should be zero.

Bits [10:8]: Delivery Mode

000: **(Fixed)** means deliver the signal on the INT pin of all processors listed in the destination. Trigger mode for "fixed" Delivery Mode can be edge or level.

001: **(Lowest Priority)** means deliver the signal on the INT pin of the processor that is executing at the lowest priority among all the processors listed in the specified destination; Trigger mode for "lowest priority" Delivery mode can be edge or level.

011: **(Remote Read)** is a request to a remote 82489DX local unit to send the value of one of its registers over the I_{CC} bus. The register is selected by providing its address in the vector field. The register value is latched by the requesting 82489DX and stored in the Remote Register where it can be read by the local processor. A Delivery Mode of "Remote Read" requires an "Edge" Triggered mode.

100: **(NMI)** means deliver the signal on the NMI pin of all processors listed in the destination. Vector information is ignored. A delivery mode equal to "NMI" requires a "LEVEL" Trigger mode.

101: **(Reset)** means deliver the signal to all processors listed in the destination by asserting/deasserting the 82489DX local unit's PRST output pin. All addressed 82489DX local units will assume their reset state but preserve their ID. One side effect of a message with Delivery mode equal to "Reset" that results in a deassert of reset is that all Local Units (whether listed in the destination or not) will reset their lowest-priority tie breaker arbitration ID to their Local unit ID. A delivery mode of "Reset" requires a "level" Trigger mode. "Reset" should not be used with "Self" or "all incl.Self" Shorthand mode since it will leave the system in non-recoverable reset state. If "RESET" is used with "all exc.Self" mode, software should make sure that only one CPU executes this instruction in an MP system.

Delivery mode options **010,110,111** are Intel reserved. They should not be used.

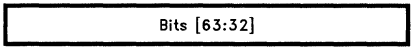
Bits [7:0] Vector. The vector identifies the interrupt being sent. If the Delivery mode is "Remote Read", then the Vector field contains the address of the register to be read in the remote 82489DX's Local unit.

NOTE:
In cases where Destination field in Interrupt Command Register [63:32] is used, Interrupt Command Register [31:0] should be programmed only **AFTER** programming Interrupt Command Register [63:32], since writing to [31:0] will start sending the message.

The following are the control words for interrupt command register [31:0] for different modes. The interrupt vector, for example, is illustrated with **AA hex**. In the remote Read request command **RR in the vector field** specify address of the register to be read. The **XX** in the vector field means the vector is don't care.

CONTROL WORD	PHYSICAL Destination Mode	LOGICAL Destination Mode
Fixed INT, Edge triggered int, *dest. field specified*	0000 00AA hex	0000 08AA hex
Lowest priority INT, Edge trigg. int, *dest. field specified*	0000 01AA hex	0000 09AA hex
Remote Read (only Edge Triggered), *dest. field specified*	0000 03RR hex	NOT SUPPORTED
NMI (Only Level) Level ASSERT, *dest. field specified*	0000 C4XX hex	0000 CCXX hex
NMI (Only Level) Level DEASSERT, *dest. field specified*	0000 84XX hex	0000 8CXX hex
Reset (Only Level) Level ASSERT, *dest. field specified*	0000 C5XX hex	0000 CDXX hex
Reset (Only Level) Level DEASSERT, *dest. field specified*	0000 85XX hex	0000 8DXX hex
Fixed INT, Edge triggered int, *Self*	0004 00AA hex	0004 08AA hex
Fixed INT, Edge trigg. int, *All inclusive Self*	0008 00AA hex	0008 08AA hex
Lowest priority INT, Edge trigg. int, *All inclusive Self*	0008 01AA hex	0008 09AA hex
NMI, Level ASSERT, *All inclusive Self*	0008 C4XX hex	0008 CCXX hex
NMI, Level DEASSERT, *All inclusive Self*	0008 84XX hex	0008 8CXX hex
Reset, Level DEASSERT, *All inclusive Self*	0008 85XX hex	0008 8DXX hex
Fixed INT, Edge trigg. int, *All exclusive self*	000C 00AA hex	000C 08AA hex
Lowest priority INT, Edge trigg. int, *All exclusive self*	000C 01AA hex	000C 09AA hex
NMI, Level ASSERT, *All exclusive Self*	000C C4XX hex	000C CCXX hex
NMI, Level DEASSERT, *All exclusive Self*	000C 84XX hex	000C 8CXX hex
Reset, Level ASSERT, *All exclusive Self*	000C C5XX hex	000C CDXX hex
Reset, Level DEASSERT, *All exclusive Self*	000C 85XX hex	000C 8DXX hex

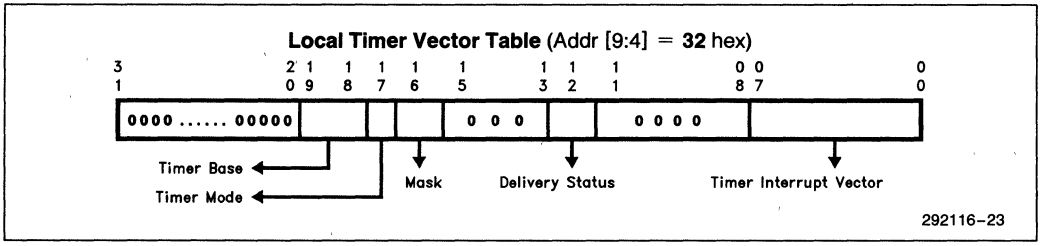
Interrupt Command Register [63:32]
(Addr [9:4] = **31** hex)



292116-22

Bits [63:32] Destination

This field is only used when the Destination Shorthand field is set to "Destination Field". If Destination field is physical mode, **then the 8 MSB contain an Destination Unit ID.** If logical mode, the full 32-bit Destination field contains the logical address. This register should be programmed for proper destination before programming Interrupt Command Register [31:0]. If the destination to a local unit with ID, say, 05 in physical mode, then the Interrupt Command Register [63:32] should be programmed as hex **0500 0000**.



Bits [31:20] Reserved. Should be written Zero.

Bits [19:18] Timer Base: This field selects the time base input to be used by timer.

00: (Base 0): Uses "CLKIN" as input.

01: (Base 1): Uses "TMBASE".

10: (Base 2): Uses the output of the divider (Base 2).

Bit 17: Timer Mode: This field indicates the operation mode of timer.

0 — ONE- SHOT;

1 — PERIODIC

In *ONE-SHOT*, the current count register remains at Zero after the timer reaches zero and software needs to reassign the timer's initial count register to rearm the timer.

In *PERIODIC* mode, when the timer reaches zero, the Current Count Register is automatically reloaded with the value in the initial Count Register, and the timer counts down again.

Bit 16: Mask: This bit serves to mask timer interrupt generation.

0 — Not masked;

1 — Masked.

Bits [15:13] Reserved. Should be written Zero.

Bit [12] Delivery Status: Delivery status indicates the current status of the delivery status of this interrupt.

0 — IDLE means that there is currently no activity for this interrupt.

1 — SEND PENDING indicates that the interrupt has been injected, but its delivery is temporarily held up by other recently injected interrupts that are in the process of being delivered; Delivery status is software read only.

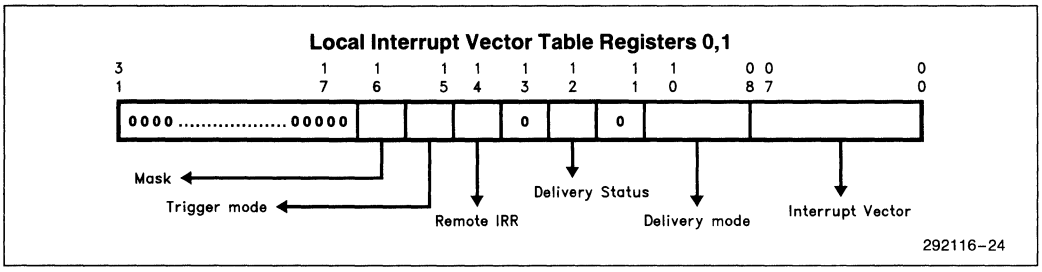
Bits [11:8] Reserved. Should be written Zero.

Bits [7:0] Timer Interrupt vector: This is the 8-bit interrupt vector to be used when timer generates an interrupt.

NOTE:
TIMER interrupts are always treated as EDGE triggered interrupts.

The following is the control word for various modes to be used in Local Timer Vector Table. For illustration purpose, the interrupt vector for Timer is shown as AA hex.

Control Word	CLKIN Input (Base 0)	TMBASE Input (Base 1)	Divider Input (Base 2)
PERIODIC timer, MASK cleared	0002 00AA hex	0006 00AA hex	000A 00AA hex
PERIODIC timer, MASK set	0003 00AA hex	0007 00AA hex	000B 00AA hex
ONE SHOT timer, MASK cleared	0000 00AA hex	0004 00AA hex	0008 00AA hex
ONE SHOT timer, MASK set	0001 00AA hex	0005 00AA hex	0009 00AA hex



Register	Address [9:4]
Local Int0 Vector table register	**35** hex
Local Int1 Vector table register	**36** hex

The same format applies to both Local Int0 and Local Int1 registers.

Bits [31:17]: Reserved: Must be Zero.

Bit 16: MASK:

0 — enables interrupt by clearing mask

1 — masks the interrupt.

Bit 15: Trigger mode:

0 — Edge Triggered

1 — Level Triggered

Bit 14: Remote IRR: This bit is used for level triggered local interrupts. Its meaning is undefined for edge triggered interrupts. Remote IRR mirrors the interrupt's IRR bit of this local unit. Remote IRR is software read only.

Bit 13: Reserved. Must be Zero.

Bit 12: Delivery Status: Software read only. Indicates the current status of the delivery of this interrupt.

0 — **IDLE** means that there is currently no activity for this interrupt.

1 — **Send Pending** indicates that the interrupt has been injected, but its delivery is temporarily held up by the recently injected interrupts that are in the process of being delivered.

Bit 11: Reserved. Must be Zero.

Bits [10:8]: Delivery mode

000 — Fixed INT

100 — NMI

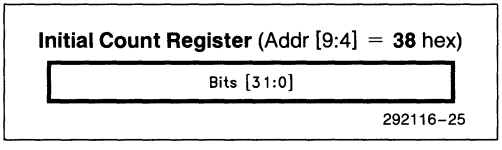
111 — ExtINTA

All other options of Bits [10:8] are reserved. Should not be used.

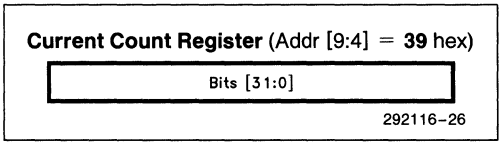
Bits [7:0]: Vector: This is the interrupt vector to use when generating interrupt for this entry.

The following are the control words for local interrupt [0 as well as 1] vector tables for different modes. The interrupt vector, for example, is illustrated with **AA hex**. The XX in the vector field means the vector is don't care.

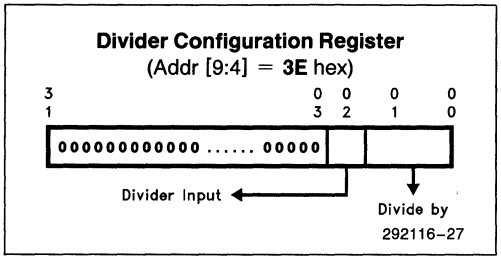
Interrupt Option	Control Word
Fixed INT, Edge triggered	0000 00AA hex
Fixed INT, Level trigg. int	0000 80AA hex
NMI (Only Level)	0000 84XX hex
ExtINTA (Only Edge)	0000 07XX hex



Bits [31:0] Initial Count: Software writes to this register to set the initial count for timer. This register can be written at any time. When written, the value is copied to the current count Register and countdown starts or continues from there. The initial count register is read-write by software.



Bits [31:0] Current Count: This is the current count of timer. It is read only by software and can be read any time.



Configuration	Control Word
Divide CLKIN by 2	0000 0000 hex
Divide CLKIN by 4	0000 0001 hex
Divide CLKIN by 8	0000 0002 hex
Divide CLKIN by 16	0000 0003 hex
Divide TMBASE by 2	0000 0004 hex
Divide TMBASE by 4	0000 0005 hex
Divide TMBASE by 8	0000 0006 hex
Divide TMBASE by 16	0000 0007 hex

Bits [31:3] Reserved. Must be Zero.

Bit [2]: Divider Input: Selects whether divider's input connects to the 82489DX local unit's CLKIN pin or TMBASE.

0 — means the divider takes its input signal from CLKIN.

1 — means use TMBASE

Bits [1:0]: Divide by: Selects by how much the divider divides.

00 — divide by 2

01 — divide by 4

10 — divide by 8

11 — divide by 16

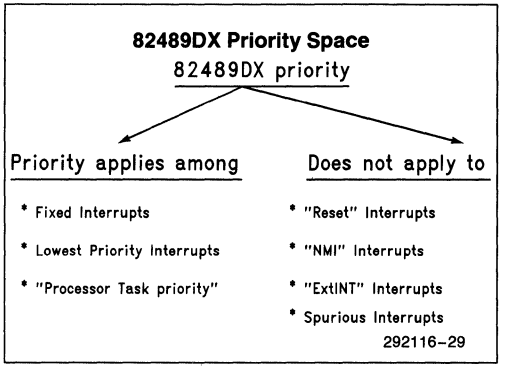
Programming Guidelines

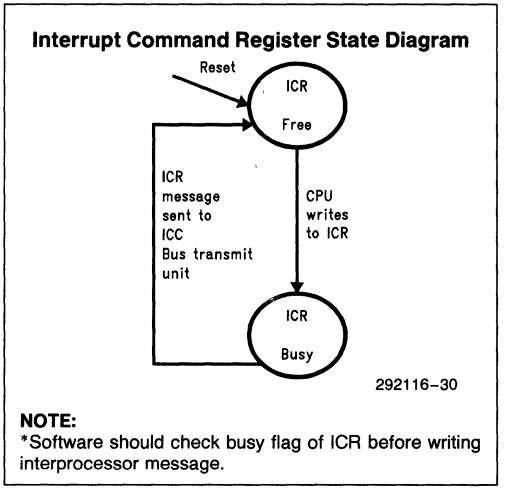
A) Modes of Interrupt in 82489DX:

Trigger Mode	Delivery Mode				
	Fixed Destination	Lowest Priority Delivery	NMI	Reset	ExtINT
Edge	✓	✓			✓
Level	✓	✓	✓	✓	

NOTE:

- RESET delivery mode should not be used for Local Interrupts.
- EOI should not be issued for NMI and ExtINT delivery mode.





CONCLUSION

82489DX has simple and powerful programming model. It has programmable priority and it supports task priority in the light of interrupt priority. It reduces the SPL() overhead which is very useful in uniprocessor system. The system performance is improved by using interrupt priority model to prioritize interrupts and by using task priority register for SPL() calls. It provides an easy migration path from 8259 by providing ExtINTA mode for DOS compatibility.

intel®

AP-479

APPLICATION NOTE

Pentium® Processor Clock Design

2

November 1995

Order Number: 241574-002

Pentium® Processor Clock Design

CONTENTS

PAGE

CONTENTS

PAGE

FIGURES

CONTENTS

PAGE

FIGURES

CONTENTS

PAGE

TABLES

1.0 INTRODUCTION

Today's high speed microprocessors place a heavy demand on clock generation and distribution. To maintain a synchronous system, well-controlled and precise clocking solutions are required. Pentium® processor, with operating frequencies of 60 MHz and 66 MHz, has tight system clock specifications. In order to bring clock signals of acceptable quality and minimal skew to the Pentium processor and the rest of the system, system designers have to contend with high speed issues for clock distribution and limited number of precise clock driver devices. In this application note, the key issues in the design of a 60 MHz or 66 MHz clock for a Pentium processor-based system will be discussed, available clock drivers will be listed and discussed, and detailed design examples of a clock solution for the Pentium processor with 256K second-level cache subsystem, using the 82496 Cache Controller and the 82491 Cache SRAMs, are provided.

The Pentium processor, 82496 Cache Controller, and 82491 Cache SRAM form a CPU-Cache core or chip set. Along with a memory bus controller (MBC), the chip set provides a CPU-like interface for many types of memory buses.

This application note is intended for system designers concerned with clock generation and distribution for the Pentium processor and CPU-Cache chip set based systems. It reflects data collected from several quarters of characterization of the Pentium processor and experience with some of the clock driver devices, as well. This application note gives readers a good understanding of the issues and solutions of high speed clocking, particularly that for the Pentium processor. The reader should be familiar with the Pentium processor and CPU-Cache chip set electrical and mechanical specifications, *Clock Design in 50 MHz Intel486™Systems*, and transmission line theory. If not, please read materials listed in Section 9.0 before proceeding.

1.1 General Clocking Issues

There are two major problems with distributing clock signals at 66 MHz: clock signal quality and clock skew. At high speed, one set of effects which has been minor in slower designs is now significant—the effects of transmission line. At high frequencies and fast edge rates, long traces behave like transmission lines. The "lumped" circuit assumption which assumes instantaneous signal transmission is no longer valid. Instead, signals travel in a finite time. When a transmission line is not properly terminated, one can observe severe overshoot, undershoot and ringback, all of which degrade logical signals. Bad signal quality can cause false switching or multiple switching, and can in extreme cases damage the devices. To maintain a clean clock signal, designers must consider clock driver characteristics, signal routing, load characteristics, and transmission line termination.

There are four basic ways to terminate a transmission line, series, parallel, Thevenin, and AC terminations (Figure 1). Series termination is recommended when driver output impedance is less than the transmission line characteristic impedance (true for most TTL drivers) and the line is driving a small number of devices. Series termination consumes low power and uses only one device; however, the termination method increases signal rise and fall times. Series termination ensures good signal quality by eliminating secondary reflection off the driver end. The rest of the termination methods eliminate reflection at the load end. All of the termination methods can provide good, clean clock signals at the load. Both parallel and Thevenin terminations consume a large amount of power. Thevenin termination consumes less power than parallel but requires one more device. AC termination consumes low power but adds capacitive load to the driver and delay due to RC time constant. Design examples provided with this application note use series termination. For more information on transmission line effects and design issues, please refer to [ref. 3, ref. 4, ref. 5]

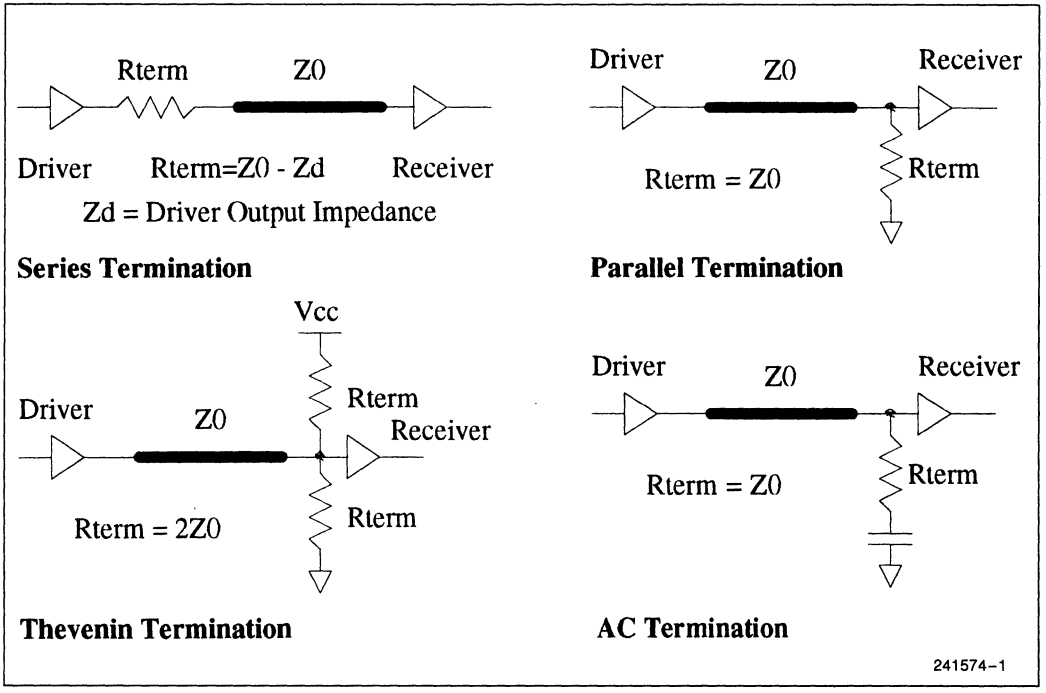


Figure 1. Common Termination Techniques

Skew is defined as the time difference between when the clock signal reaches each component. As frequency increases, there is less and less time for computation in a given clock period for a synchronous design. For a typical design, the time from one rising edge to the next is composed of the largest path-delay, setup time, propagational delay through logic elements, and skew. Clock skew then, takes away from the time available for propagational delay, thereby restricting the amount of logic done in a clock cycle. For high speed designs, skew must be minimized.

To minimize skew, designers must tune clock traces so that the propagational delay from driver through each trace to load is the same for each load. For balanced loads, tuned traces have same lengths. For unbalanced loads, trace lengths can be adjusted to make up for loading differences. If possible, designers should try to keep the loading on each clock line the same.

2.0 PENTIUM® PROCESSOR, 82496 AND 82491 SYSTEM CLOCK SPECIFICATIONS

System clock specifications can be divided into 2 categories: signal quality requirements and skew specifications. Clock signal quality requirements are the same for the Pentium processor and CPU-Cache chip set. Skew specifications are only required for CPU-Cache chip set.

Signal quality requirements define boundaries for acceptable signal shapes and levels. There are two parts to signal quality requirements: signal quality specifications (Table 1) and guidelines (Table 2). Please refer to the latest revision of the Pentium processor and CPU-Cache chip set specifications for more details and for the most up-to-date information.

Table 1. Clock Signal Quality Specifications

Symbol (5)	Parameter	Minimum	Maximum	Unit	Notes
	CLK Frequency	33.33	66.66	MHz	(1)
t2	CLK Period	15		ns	
t3	CLK High Time	4		ns	(2)
t4	CLK Low Time	4		ns	(3)
t5	CLK Rise Time	0.15	1.5	ns	(4)
t6	CLK Fall Time	0.15	1.5	ns	(4)
	CLK Stability		±250	ps	(6), (7), (8), (9)
	V_{IH}	2	V_{CC} + 0.3	V	
	V_{IL}	−0.3	0.8	V	

NOTES:
1. Below 66 MHz only functionality is guaranteed.
2. High times are measured between 2.0V crossing points.
3. Low times are measured between 0.8V crossing points.
4. Rise and fall times are measured between 0.8V and 2.0V.
5. Symbols in Figure 2.
6. Functionality is guaranteed by design/characterization.
7. Measured on rising edge of adjacent CLKs at 1.5V.
8. To ensure a 1:1 relationship between the amplitude of the input jitter and the internal and external clocks, the jitter frequency spectrum should not have any power spectrum peaking between 500 KHz and 1/3 of the CLK operating frequency.
9. The amount of jitter present must be accounted for as a component of CLK skew between devices.

Table 2. Clock Signal Quality Guidelines

Parameter	Maximum	Unit	Notes
Overshoot	1.6	V	(1)
Undershoot	1.6	V	(1)
Ringback	0.8	V	(2)

NOTES:
1. Overshoot (undershoot) is the absolute value of the maximum voltage above V_{CC} (or below V_{SS}). The guideline assumes the absence of diodes on the input.
2. Ringback is the absolute value of the maximum voltage at the receiving pin below V_{CC} (or above V_{SS}) relative to V_{CC} (or V_{SS}) level after the signal has reached its maximum voltage level. The input diodes are assumed present.

The overshoot guideline should be used in simulations, without diodes present, to ensure overshoot (undershoot) is within the acceptable range. The ringback guideline is provided for verification in an actual system. System designers do not have to worry about ringback if the signal does not overshoot or undershoot, respectively. Figure 2 summarizes clock waveform requirements listed in Table 1.

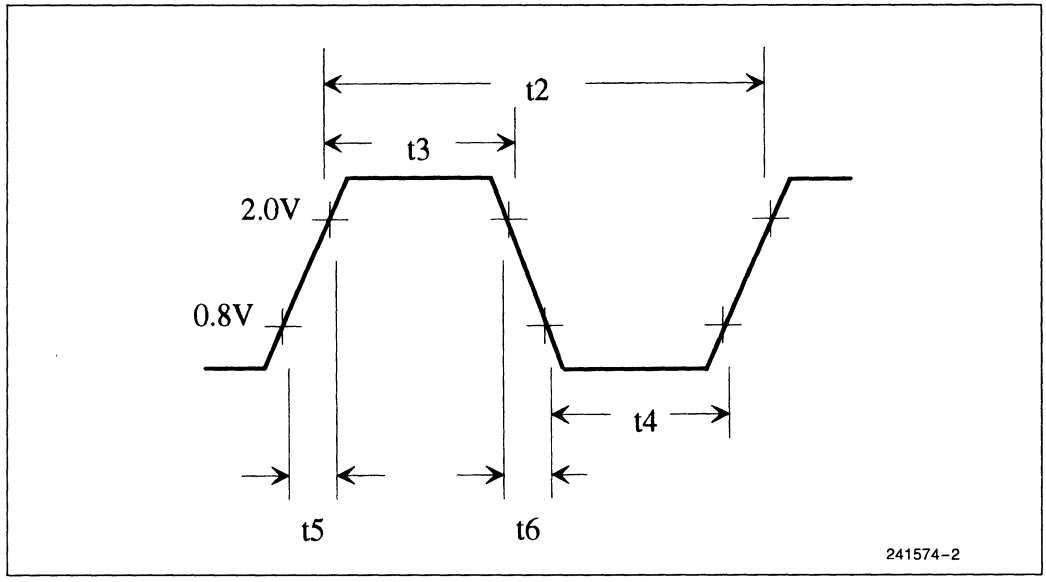


Figure 2. Clock Requirements for the Pentium® Processor and CPU-Cache Chip Set

Figure 3 to Figure 5 illustrates examples of acceptable and unacceptable clock waveforms. Waveform in Figure 3 is for an input model without diodes. Waveform in Figure 4 is for an input model with diodes. The diodes clamp the voltage and prevent it from going more than a diode drop above V_{CC} or below V_{SS}. Waveform in Figure 5 is for an input model without diodes. The waveform is not acceptable for several reasons. It violates the minimum low time specification (4 ns), the maximum fall time specification (1.5 ns), and it does not follow the maximum undershoot guideline (1.6V).

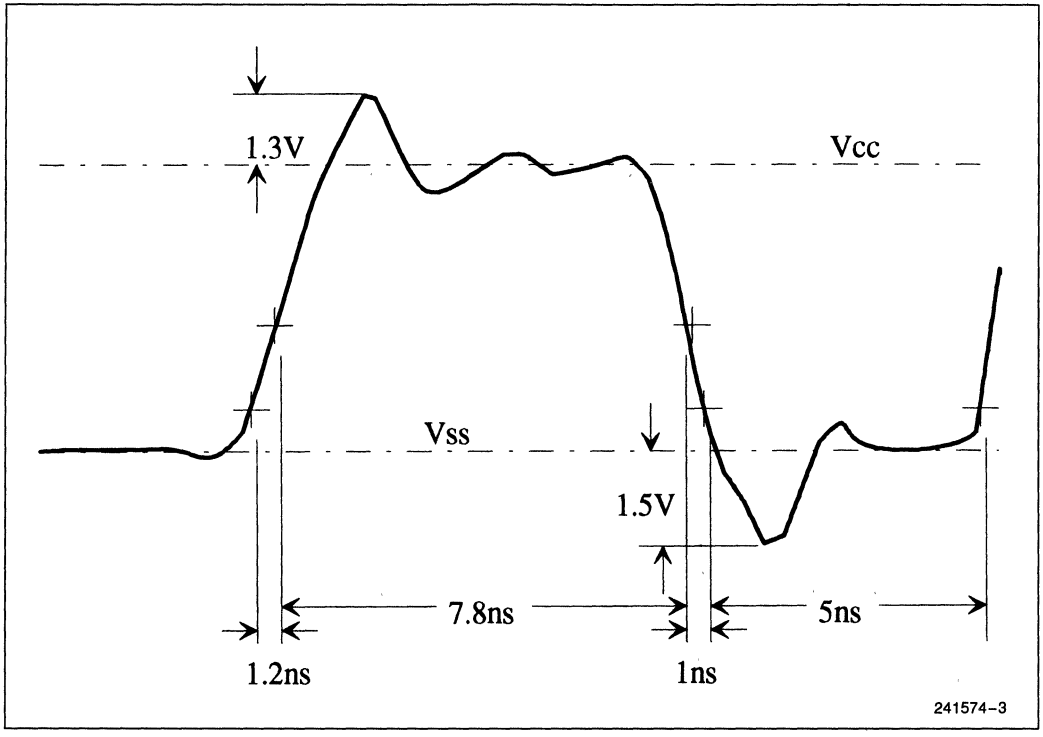


Figure 3. An Example of an Acceptable Clock Waveform (Diodes are Absent from the Input Model)

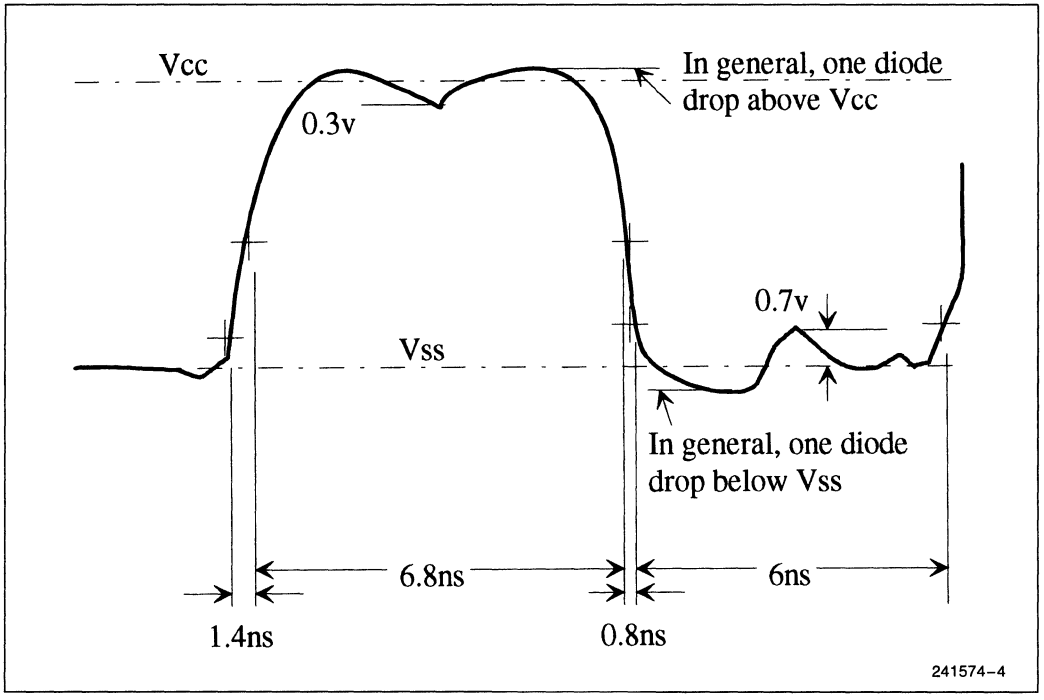


Figure 4. An Example of an Acceptable Clock Waveform (Diodes are Present in the Input Model)

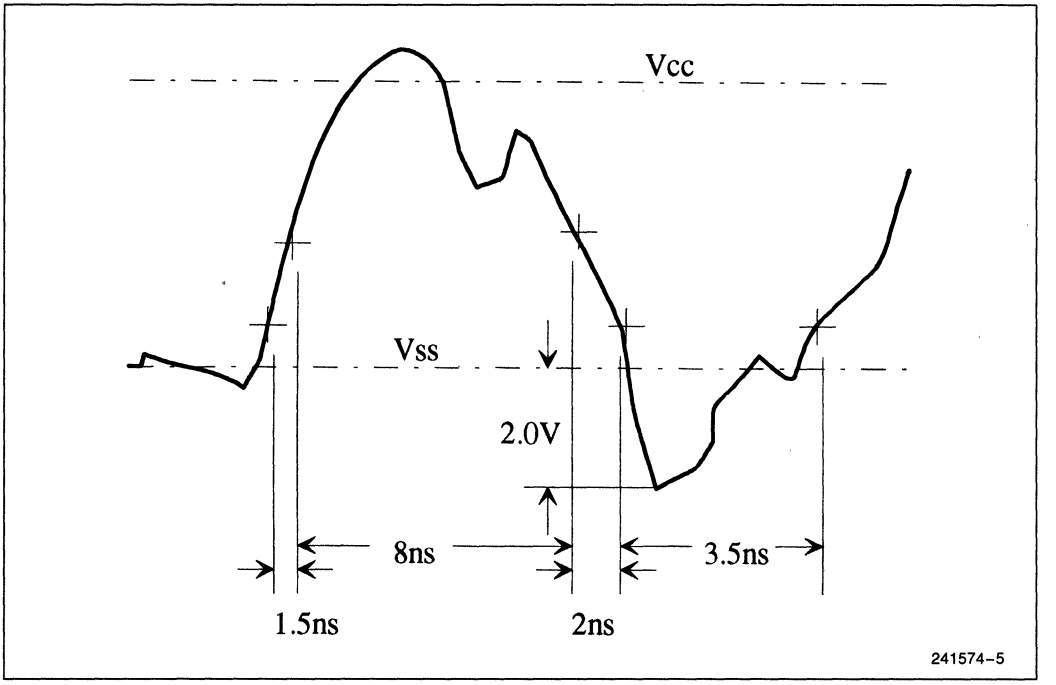


Figure 5. An Example of an Unacceptable Clock Waveform (Diodes Are Absent from the Input Model)

Clock skews for the CPU-Cache chip set are measured at 0.8V, 1.5V, and 2.0V on the rising edge. Worst case skew between the Pentium processor and the 82496 is 0.2 ns, and worst case skew between any 82491 and either the Pentium processor or the 82496 is 0.7 ns.

3.0 AVAILABLE CLOCK DRIVERS

Intel has held discussions with many clock driver component companies. The intent has been to enable these companies to offer clock driver solutions that meet the Pentium processor specifications. It has also been to ensure that the super set of these companies can provide support and distribution worldwide on a schedule that closely matches the Pentium processor's availability. Based on information available, Table 3 lists a number of companies who are planning to offer solutions to meet these requirements. All the clock drivers listed in Table 3 have maximum output frequency equal or above 66 MHz. Preliminary data sheets show that solutions listed in Table 3 meet the CPU or CPU-Cache chip set requirements. The specifications listed are based on preliminary data provided by each company and may be subject to change. Designers should contact each company for the latest specifications and availability. Some evaluation has been done by simulating an example clock layout using output models supplied by a subset of the companies listed, along with interconnect models and preliminary clock input model of the CPU-Cache chip set. For more detail on the simulations and example routing, please see Section 5.0. Intel has been and will be working closely with the listed companies to ensure they have the latest specifications for the Pentium processor. With published preliminary data sheets, all the listed parts meet either the CPU or the chip set clock specifications (including the signal quality and skew specifications). Please contact individual manufacturers for data sheets and sample availability.

Table 3. Clock Driver Options

Mfgr	Part #	Driver Type	Level In/Out	Pin-to-Pin Skew (ns)	Part-to-Part Skew (ns)	t_r/t_f (0.8V–2.0V) (ns)	Clock Stability	# of Outputs (per pkg)	Spec'd Loading	Max. Freq.
Intel Spec			TTL inputs	(1)		1.5/1.5	±250 ps (2)			66 MHz and 60 MHz
AMCC	SC35XX-1	Buffer	PECL or TTL/TTL	0.5	1.0	1.5/1.5		20 Outputs Which Vary with Part #	10 pF	80 MHz
	SC44XX-80	PLL	TTL/TTL	±0.2	1.0 (9)	1.5/1.5		6–12 Outputs	35 pF	80 MHz
AT&T (7)	DA400	PLL	PECL or TTL/TTL	0.2	0.5	1.5/1.5		8@1, 0.5X (Prog Shift)	50 pF	100 MHz
Cypress	CY7B991	PLL	TTL/TTL	0.5	1.2 (6)	1.5/1.5	0.5%	4@1X 4@1, 0.5, 0.25X	50Ω/ 30 pF	80 MHz
ICS	ICS2686	PLL		0.5	0.6	1.5/1.5		5@1, 0.5X	20 pF	
Intel	85C224-100	Divider/ Buffer/PLD	TTL/CMOS	Divider 0.4 Buffer 0.5	NA	Divider 1.2/1.1 Buffer 1.4/1.1		8 @ + 1X, −1X, 0.5X	70Ω/ 50 pF	Divider 100/50 Buffer 133
Intel	85C224-10	Buffer/ Divider/PLD	TTL/CMOS	Divider 0.4 Buffer 0.5	NA	Divider 12./1.1 Buffer 1.4/1.1		8 @+1X, −1X, 0.5X	70Ω/ 50 pF	Divider 58/29 Buffer 100
Motorola	MC10H646	Buffer	PECL or TTL/TTL	0.5	1.0	1.2/1.2	NA	8	50Ω/ 50 pF	100 MHz
	88915	PLL	TTL/CMOS	0.5	NA	2.5/2.5(11)	NA	5@1X 1@2X 1@.5X 1 Inverted X	50Ω	66 MHz

Table 3. Clock Driver Options (Continued)

Mfgr	Part #	Driver Type	Level In/Out	Pin-to-Pin Skew (ns)	Part-to-Part Skew (ns)	t_r/t_f (0.8V–2.0V) (ns)	Clock Stability	# of Outputs (per pkg)	Spec'd Loading	Max. Freq.
National	CGS74CT2524	Buffer	TTL/CMOS	0.45	NA	1.5/1.5		4	50 pF	100 MHz
	CGS74CT2527	Buffer	TTL/CMOS	0.45	NA	1.5/1.5		8	50 pF	100 MHz
	CGS74B2528	Buffer	TTL/TTL	0.55	NA	1.5/1.5		10	50 pF	70 MHz
Pioneer	PI6B2407	PLL	TTL/TTL	±0.25	NA	1.5/1.5	100 ps	12@1, 0.5, 2X (Prog Shift)		80 MHz
TI (8)	CDC328	Buffer	TTL/TTL	0.7	NA	1.2/0.5	NA	6	500Ω/ 50 pF	Not Spec'd.
Triquint (10)	GA1085	PLL	TTL/TTL	0.25	NA	1.4/1.4	75 ps (typ.)	5@1X 4@0.5X 2@0.5X (Prog Shift)	50Ω	66 MHz
	GA1086	PLL	TTL/TTL	0.25	NA	1.4/1.4	75 ps (typ.)	9@1X 1@0.5X	50Ω	66 MHz
	GA1087	PLL	TTL/TTL	0.25	NA	1.4/1.4	75 ps (typ.)	6@1X 4@0.5X	50Ω	66 MHz
Vitesse	VSL4485	PLL	TTL/TTL	0.5	NA	1.5/1.5		6@1X 2@1, 2, 4X	50 pF	70 MHz
	VSL4586	PLL	TTL/TTL	0.5	NA	1.5/1.5		2@1X 6@1, 2, 4X	50 pF	70 MHz

NOTES:
1. 0.7 ns between Pentium processor-82491, 82496-82491, 82491-82491. 0.2 ns between Pentium processor-82496. Assumed 0.5 ns between clock driver outputs, leaving 0.2 ns for routing or trace skew.
2. See complete specification in Table 1 or the data book.
3. Manufacturers listed in alphabetical order.
4. Contact manufacturers for price and availability information.
5. Intel does not guarantee specifications for other manufacturer's devices. All clock driver specifications listed were provided by the manufacturer and are subject to change. Designers should contact the manufacturer for the latest specification/data sheet information.
6. As low as 0.75 ns in some configurations.
7. First samples in March '93. Specifications may improve during characterization.
8. Other Solutions are under development. Contact TI for preliminary details.
9. Maximum phase erro quoted in the manufacturer's data sheet for the entire frequency range.
10. Other configurations available. Contact Triquint for details.
11. Between 0.2 V_{CC} and 0.8 V_{CC}. Contact Motorola for details between 0.8 and 2.0V.

intel®

AMCC offers the SC35XX-1 series of buffered clock drivers and the SC44XX-80 series of PLL based clock drivers. The SC35XX-1 series must be driven with a TTL or PECL 2X frequency input. Each member of the series provides 20 outputs. Depending on the specific part within the series, these 20 outputs can be configured to provide the primary frequency, 1/2, or 1/4 the primary frequency. The SC3502-1 even provides 5 inverted outputs of the primary frequency. The SC44XX-80 series must be driven with a TTL input. The PLL design allows for very low skew (±200 ps) between the outputs.Different members of the series offer different numbers and configurations of outputs. Between 4 and 8 outputs are available at the primary frequency. These devices also allow a subset of the outputs to be configured for 1/2X or 2X the primary frequency. In addition, the PLL allows the outputs to be skewed in phase from one another.

AT&T DA400 is a PLL clock driver. Its inputs can be driven by TTL or PECL levels. Eight outputs are provided. They can be configured for the primary frequency or 1/2X the primary frequency. In addition each output has a programmable delay line which allows 1/32 or 1/64 increments of the clock period of delay between outputs.

Cypress's CY7B991 is a PLL clock driver. It requires a TTL input and is able to drive 8 outputs. A subset of the outputs can be configured as 1/2X, 1/4X, or inverted outputs. As with other PLL solutions, the skew between outputs is small and the outputs can be configured for a fixed amount of delay or skew between outputs.

ICS's ICS2686 is a PLL clock driver. Five outputs are available. Both primary and 1/2X frequencies are available. The ICS2686 has been designed to work with the 74ABT240 type buffer to provide more than 5 outputs. A unique feature of the ICS2686 is the multiple feedback inputs. This feature allows synchronizing multiple outputs at their destination or load with the input clock.

Intel's 85C224-100 is a "20V8" architecture programmable logic device. From its TTL inputs it provides 8 TTL outputs which can be configured to provide 1X, 1X inverted, and 1/2X versions of the primary frequency, in any combination. When programmed to function as a frequency divider, the primary frequency can be as high as 100 MHz and the 1/2X frequency outputs will maintain output skew below 400 ps. When programmed to operate as a straight 1X buffer, it supports frequencies of up to 133 MHz with less than 500 ps of output skew. The 85C224-100 provides a combination of superior output signal quality including fast rise and fall times and low output skew. A particularly unique feature of the 85C224-100 is in its programmable logic circuitry. Its flexibility satisfies programmable logic needs such as control line signals and widespread glue logic. With this minimized output skew PLD, a single 28-pin PLCC can provide low output skew clock distribution, frequency division, and programmable logic; for the low price of a 20V8 PLD.

Motorola offers both a buffered and a PLL clock solution. Motorola's 10H646 is a buffered clock driver. It offers both TTL and ECL inputs which supports backplane routing using ECL levels. The clock driver's outputs are clamped to 3V, not V_{CC}. 10H646's output stage has similar rise and fall output resistances. Similar rise and fall output resistances makes series termination easier since the termination resistance is the difference between the characteristic impedance of the transmission line connecting the output to the load and the driver's output impedance. 10H646 has 8 1x outputs. As a straight buffer, 10H646 does not offer any multiples of the input besides 1x. The Motorola 88915 is a PLL clock driver. It provides a 0.5 ns skew between outputs. The 88915 provides 5 1X outputs along with 1 2X, 1 0.5X, and 1 inverted X outputs.

National's clock buffers are packaged to function reliably at high frequencies. Their output rise and fall resistances are approximately equal. The CGS74CT2524 and 2527 provide 0.45 ns of output skew. The CGS74CT2528's output skew, 0.55 ns, allows for only 0.15 ns skew due to board traces or any unbalanced loading effects when using the 82496/82491 cache, however this amount may be sufficient for other cache solutions. These parts offer a range of 4, 8, and 10 outputs. The CGS74CT2524 and 2527 have CMOS level outputs, which transition from rail to rail.

Pioneer's PI6B2407 is a PLL clock driver. From its TTL input, it provides twelve TTL outputs, which can be configured to operate at 1X or 2X the input frequency. In addition, the outputs can be phase adjusted from the input clock. The PI6B2407 is able to provide ±0.25 ns of skew between outputs while maintaining the fast 1.5 ns rise and fall times.

Texas Instruments' ABT328 driver provides six outputs with an output skew of 0.7 ns. Please contact Texas Instruments for the availability of 0.5 ns output skew parts. 0.7 ns output skew is too large for the chip set application. In the design example on Section 5.0, 0.5 ns output skew is assumed. As a buffered driver, the ABT328 offers only 1x outputs.

TriQuint's GA1086 is a Gallium Arsenide-based product. It takes a 66 MHz input and produces nine 66 MHz outputs and one 33 MHz output. The availability of a low skew 33 MHz output facilitates clock distribution for systems that have synchronous 33 MHz memory buses. Since the part is phase-lock-loop based, one of the outputs can be fed back to the input so that all the outputs are synchronized with the input clock. Such a set up is ideal for cascading clock drivers to achieve maximum fanout. The specified output skew of the GA1086 is 0.25 ps, the smallest skew number available. Triquint also offers the GA1085 and GA1087. These products are similar to the GA1086, however, they offer different combinations of outputs between 1X and 0.5X.

Vitesse's VSL4485 is also a Gallium Arsenide-based product. It offers 1x, 2x, and 4x options on two of its eight outputs. Thus, to obtain both 33 MHz and 66 MHz signals with low skew, for example, the clock input frequency of the VSL4485 can be 33 MHz. For the chip set application, two 66 MHz outputs are not enough, and thus cascading another driver is necessary. Alternatively, the input can be 66 MHz and all of its outputs can be at 66 MHz. It offers 0.5 ns output skew, and a low effective delay. In addition, VSL4485 can generate programmable, multiple phase relationships among its outputs.

4.0 CLOCK GENERATION FOR THE Pentium® PROCESSOR AND THE CPU-CACHE CHIP SET

Clock generation is the generation of copies of clock signals from a signal oscillator or any other source which then are distributed to the various loads. The function of a clock driver is to generate multiple copies of clocks from a single source. In general, Pentium processor-based systems have three types of memory interface: fully synchronous, divided synchronous, and asynchronous. Each interface requires different methods of clock generation. The basic setup of a processor card is illustrated symbolically in Figure 6. Depending on the configuration, the Clock In signal can come from the memory bus or a separate oscillator.

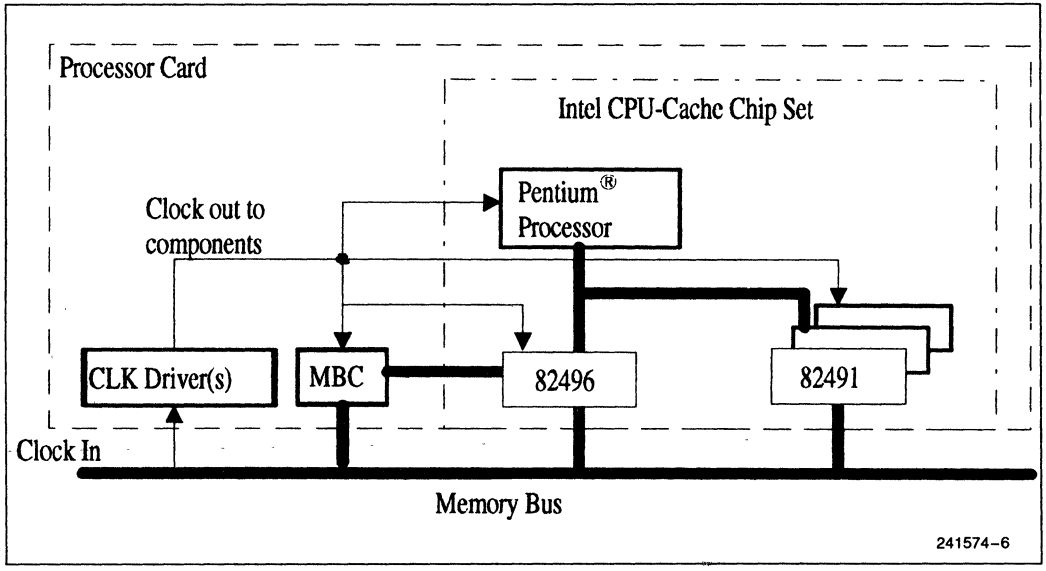


Figure 6. A CPU Module with the Pentium® Processor, 82496 and 82491 CPU-Cache Chip Set

4.1 Clock Generation for Fully Synchronous Systems

A fully synchronous system is one which everything in the system runs synchronous to the CPU. In particular, the memory bus interface is synchronous to the CPU. In Figure 6, the memory bus is at 66 MHz, synchronous to the CPU module. Clock In signal must be synchronous to the memory bus. Clocking for this case involves the generation of tightly controlled copies of clock signals that are distributed to all the clocked parts. The task of clock generation and distribution is the most difficult for this type of set up. All copies of clock signals must come from a single source, and must be deskewed appropriately. For Pentium processor-based systems that run at 66 MHz, the most critical parameter in choosing a clock driver is its output skew, as well as its part-to-part skew if more than one driver is needed. Since all the clock signals are at 66 MHz, only 1x outputs are needed. All of the drivers listed in Section 3.0 can be used here.

For a fully synchronous configuration, it is likely that a single clock driver cannot provide enough copies of clock signals. Then, some kind of cascading of drivers is necessary. Figure 7 shows two ways of clock generation by cascading drivers. Tskew is the total worst case skew at outputs of CD2 and CD3. Tpp23 is the worst case part-to-part skew between CD2 and CD3. Tos2 is the worst case output skew of CD2, assuming the worst case output skew of CD3 is the same as Tos2. Tos1 is the worst case output skew of CD1. Ttol2 is the feedback tolerance of CD2. Feedback tolerance is the phase tolerance between the feedback input and the reference clock. Typically, Ttol2 is a small number. For the examples in Figure 7, it is assumed that only the second level drivers feed the clock signals to the loads. Otherwise, for part a, signals from CD2 will be later than signals from CD1 by the propagational delay of CD2 which is typically between 6 ns to 8 ns.

For the examples in Figure 7 clock signals for the CPU-Cache chip set must be derived from one clock driver outputs only so that the 0.2 ns and 0.7 ns skew specifications can be met. In part a, Tskew, the sum of Tpp23, Tos2, and Tos1 is the worst case skew which is the skew between an output of CD2, and an output of CD3. The output skew of CD1 (Tos1) causes the inputs to CD2 and CD3 to arrive at different times. The difference in propagational delay which is Tpp23, further skews the outputs of CD2 and CD3. If the part-to-part skew does not include output skew, different outputs from CD2 and CD3 can also be skewed by the output skew. For part b, Tskew, the sum of Ttol2, Tos2, and Tos1, is also the worst case skew between the outputs of CD2 and the outputs of CD3. Once again, Tos1 causes the inputs to CD2 and CD3 to arrive at different times. The feedback in CD2 synchronizes all its outputs in the input. The feedback output of CD2 is different from the input reference clock only by Ttol2. All the other outputs are further skewed from the feedback output by Tos2. The analysis for CD3 is the same.

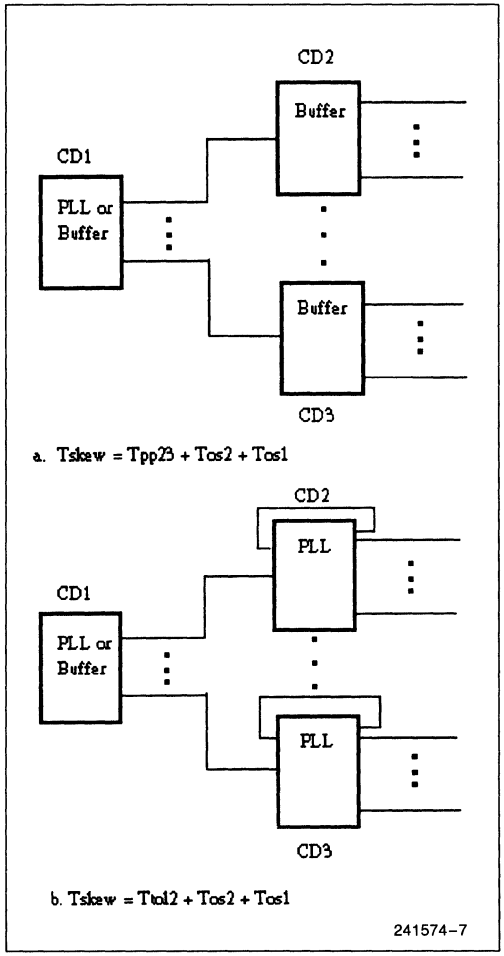


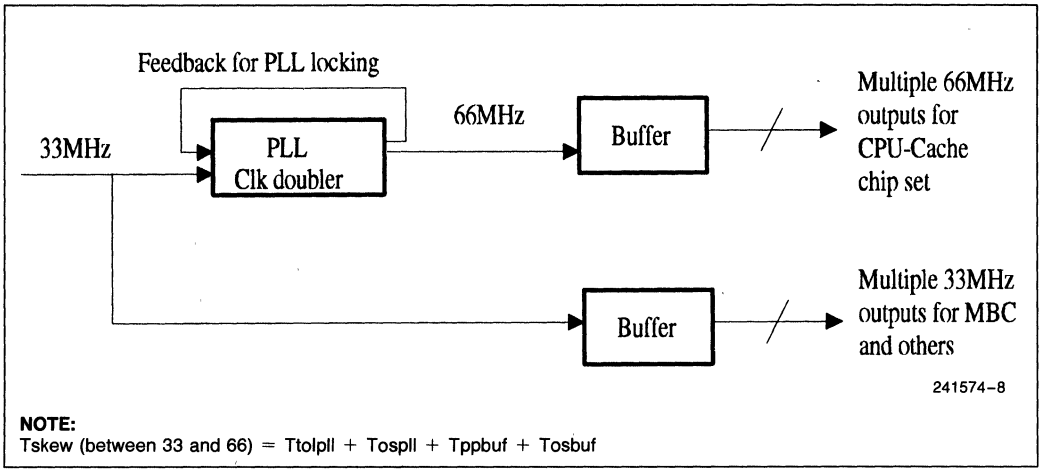
Figure 7. Examples of Clock Generation

4.2 Clock Generation for Divided Synchronous Systems

For a divided synchronous system, the memory bus is at half the speed of the CPU-Cache chip set; i.e.,

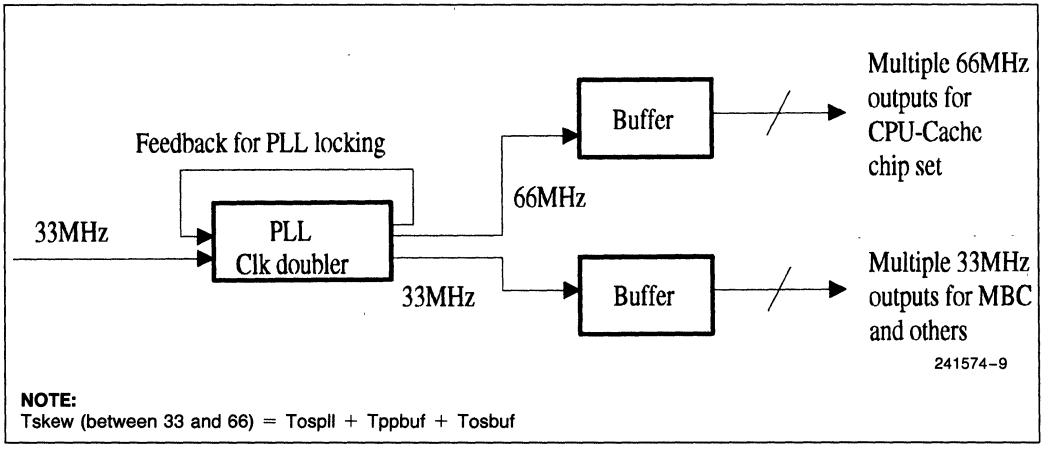
the memory bus runs at 33 MHz for the Pentium processor or the CPU-Cache chip set based systems. A 33 MHz reference clock (Clock In) can come from the backplane from which all the clocks serving the CPU-cache module (Figure 6) must be synchronized. The memory bus controller (MBC) itself requires both 33 MHz and 66 MHz clocks. For this configuration, clock drivers that can provide both 33 MHz and 66 MHz outputs are needed.

There are several ways of providing the two frequencies. They are shown in Figure 8 through Figure 12. Tskew is the worst skew between 33 MHz signals and 66 MHz signals. The skews among 66 MHz signals or among 33 MHz signals are simply the output skew of the driving devices. Ttolpll is the PLL CLK doubler or PLL CLK divider's feedback tolerance. Tospll is the PLL CLK doubler or PLL CLK divider's worst case output skew. Tppbuf is the worst case part-to-part skew of the second level buffers. Those buffers can be phase-lock-loops also in which case Tppbuf is the feedback tolerance of the PLLs if feedback is used. Tosbuf is the worst case output skew of the second level buffers. Tos1 is the output skew of CD1, Ttol2 is the feedback tolerance of CD2, and Tos2 is the output skew of CD2.



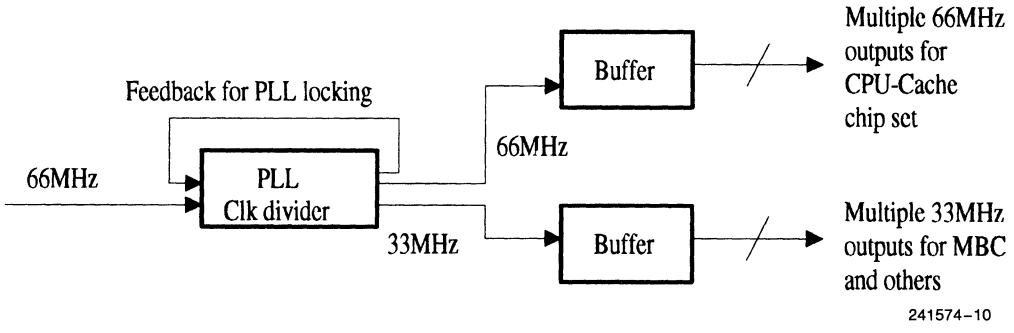
NOTE:
Tskew (between 33 and 66) = Ttolpll + Tospll + Tppbuf + Tosbuf

Figure 8. Clock Generation Using Clock Doubler



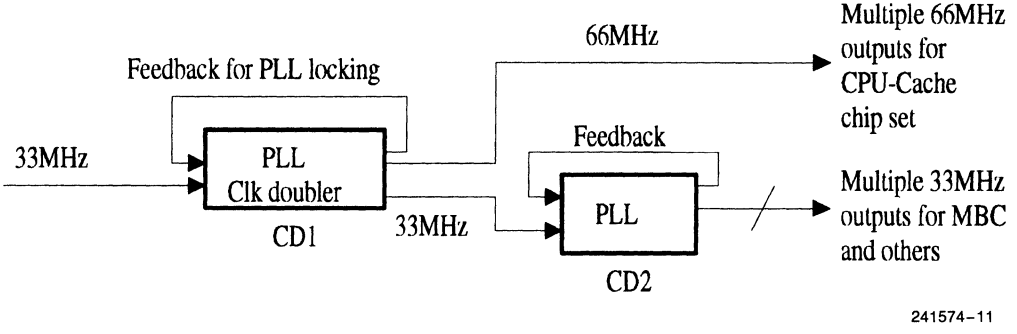
NOTE:
Tskew (between 33 and 66) = Tospll + Tppbuf + Tosbuf

Figure 9. Clock Generation Using Clock Doubler



NOTE:
Tskew (between 33 and 66) = Tospll + Tppbuf + Tosbuf

Figure 10. Clock Generation Using Clock Divider



NOTE:
Tskew (between 33 and 66) ~ Tos1 + Ttol2 + Tos2

Figure 11. Clock Generation Using Two PLLs

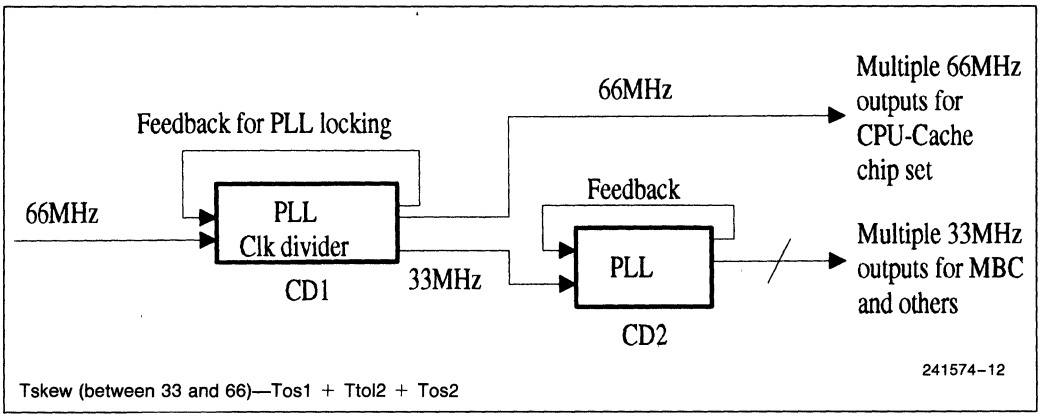


Figure 12. Clock Generation Using Two PLLs

Since the outputs of the first level PLL CLK doublers and dividers go directly to inputs of another clock driver, the signal quality requirements of these outputs are not as stringent as if the outputs drive the loads of the Pentium processor and others. One of the functions of a clock driver is to buffer and clean up a clock signal in addition to generating multiple copies of the same. However, the output skew of the PLL used for the first level is very important. Depending if feedback is used, the feedback tolerance is of importance. When choosing a clock driver, also be sure that its maximum output frequency is greater than 66 MHz for 66 MHz outputs and 33 MHz for 33 MHz outputs. The parts listed in Table 4 and Table 5 are examples of devices that can be used as first level drivers as illustrated in Figure 8 through Figure 12.

Table 4. List of Clock Doubler Parts

Manufacturer	Part #	Driver Type	# of Outputs (per pkg)
Motorola	88915	PLL	1 @ 2x 6 @ 1x (1) 1 @ 0.5x
Motorola	88916	PLL	1 @ 2x 4 @ 1x (1) 1 @ 0.5x
TI	ABT338	PLL	1 @ 2x 4 @ 1x 1 @ 0.5x
Vitesse	VSL4485	PLL	6 @ 1x 2 @ 1, 2, or 4x

NOTES:
1. One of the outputs is inverted.
2. This list is not meant to be complete. Other solutions may be available.

The phase-lock-loop drivers listed in Table 4 can be used to drive the Pentium processor loads directly if only one copy of 66 MHz clock signal is needed. In this case, the second level buffers are not necessary if the driver used can provide enough 33 MHz copies. Intel has not done formal analysis on these parts.

Table 5. List of Clock Divider Parts

Manufacturer	Part #	Driver Type	# of Outputs (per pkg)
Motorola	88915	PLL	1 @ 2x 6 @ 1x (1) 1 @ 0.5x
Motorola	88916	PLL	1 @ 2x 4 @ 1x (1) 1 @ 0.5x
Texas Instruments	ABT338	PLL	1 @ 2x 4 @ 1x 1 @ 0.5x
Texas Instruments	ABT337	Buffer	4 @ 1x 4 @ 0.5x
Texas Instruments	ABT339	Buffer	4 @ 1x 4 @ 0.5x
TriQuint	GA1086	PLL	9 @ 1x 1 @ 0.5x

NOTES:
1. One of the outputs is inverted.
2. This list is not meant to be complete. Other solutions may be available.

Table 5 lists examples of clock drivers that offer divided by 2 outputs. These devices can be used as the first level drivers illustrated in Figure 8 through Figure 12. Depending on the number of 66 MHz copies and 33 MHz copies needed, the second level buffers may not be necessary. Again, Intel has not performed any formal analysis on these parts.

4.3 Clock Generation for Asynchronous Systems

If the memory bus is not synchronized with the CPU or CPU-cache module, clock generation for the system is easier compared with the two configurations above. However, clock synchronization for the Pentium processor, 82496, and 82491, as well as the clocks for the MBC is still a concern. In order for the MBC to communicate properly with the CPU-Cache chip set, some synchronized clocks at 66 MHz are needed. Since the system is asynchronous to the CPU-Cache chip set, the number of synchronous MBC clock signals is less than the synchronous case. The examples in Section 5.0 illustrate how the synchronization is done. Since the system is asynchronous, one can use a different clock source for the CPU-Cache chip set from the rest of the system.

5.0 Pentium® PROCESSOR WITH 256K 82496/82491 SECOND LEVEL CACHE CLOCK DISTRIBUTION DESIGN EXAMPLES

After a clock generation scheme is determined, careful analysis must be done on clock distribution to ensure minimal skew and proper rise and fall times. Clock distribution is the connection between clock driver outputs and clock inputs of the components that need clocking. Preliminary analysis has been done on several of the drivers listed in Section 3.0. The following examples show in detail how to terminate transmission lines properly, tune clock traces to minimize board trace skew, and validate the usefulness of the drivers to the CPU-Cache chip set using models from the manufacturers. The examples have been done using preliminary or typical models for the devices involved. They are meant as an example of the process designers can use when selecting and routing a clock circuit. Although the examples only show the clock distribution for the CPU-Cache chip set, the same principles can be applied to distribution to the memory bus controller (MBC) and other parts.

5.1 Clock Routing for the 256K CPU-Cache Chip Set

Analysis for CPU-Cache chip set clock routing is done using first order input models for the three chips, shown in Figure 13. Specific to CLK inputs, the models shown are **typical models** based on on-going simulation efforts, and are subject to change. Refer to the Intel data books or contact Intel for the latest models (including minimum and maximum conditions). The models include package inductance, package capacitance, and input buffer capacitance of the clock pins.

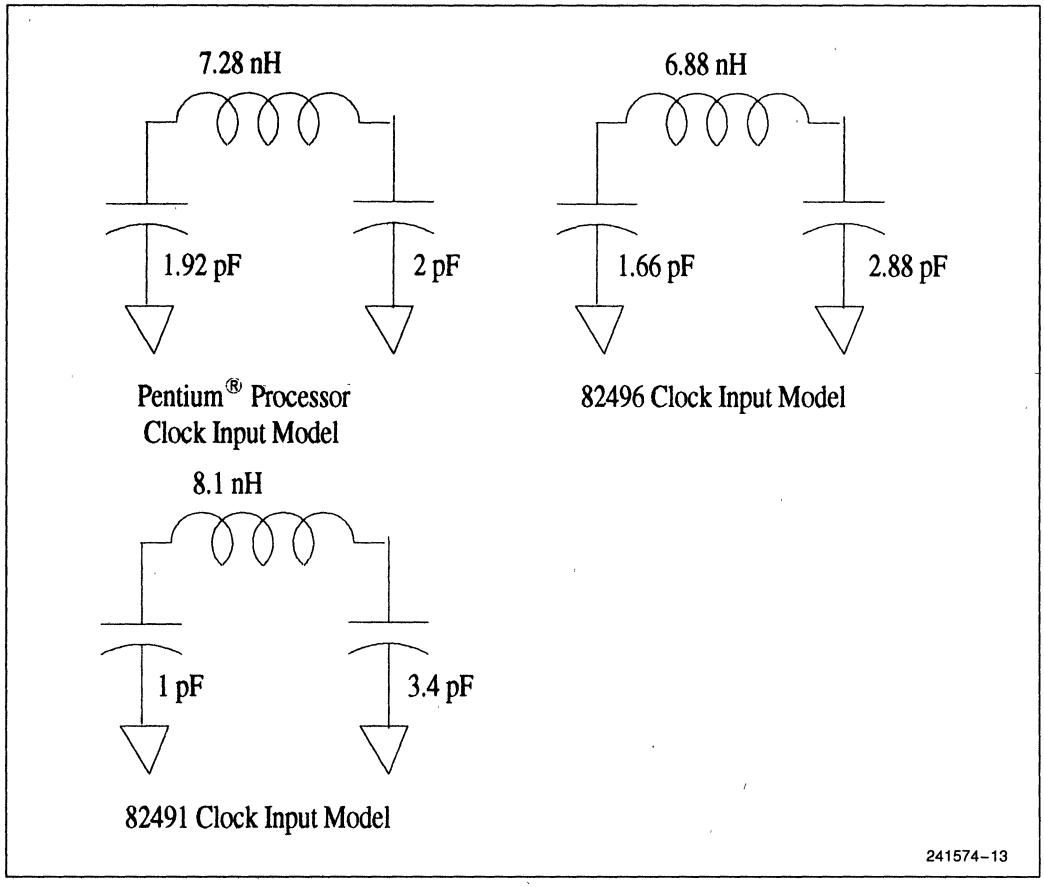
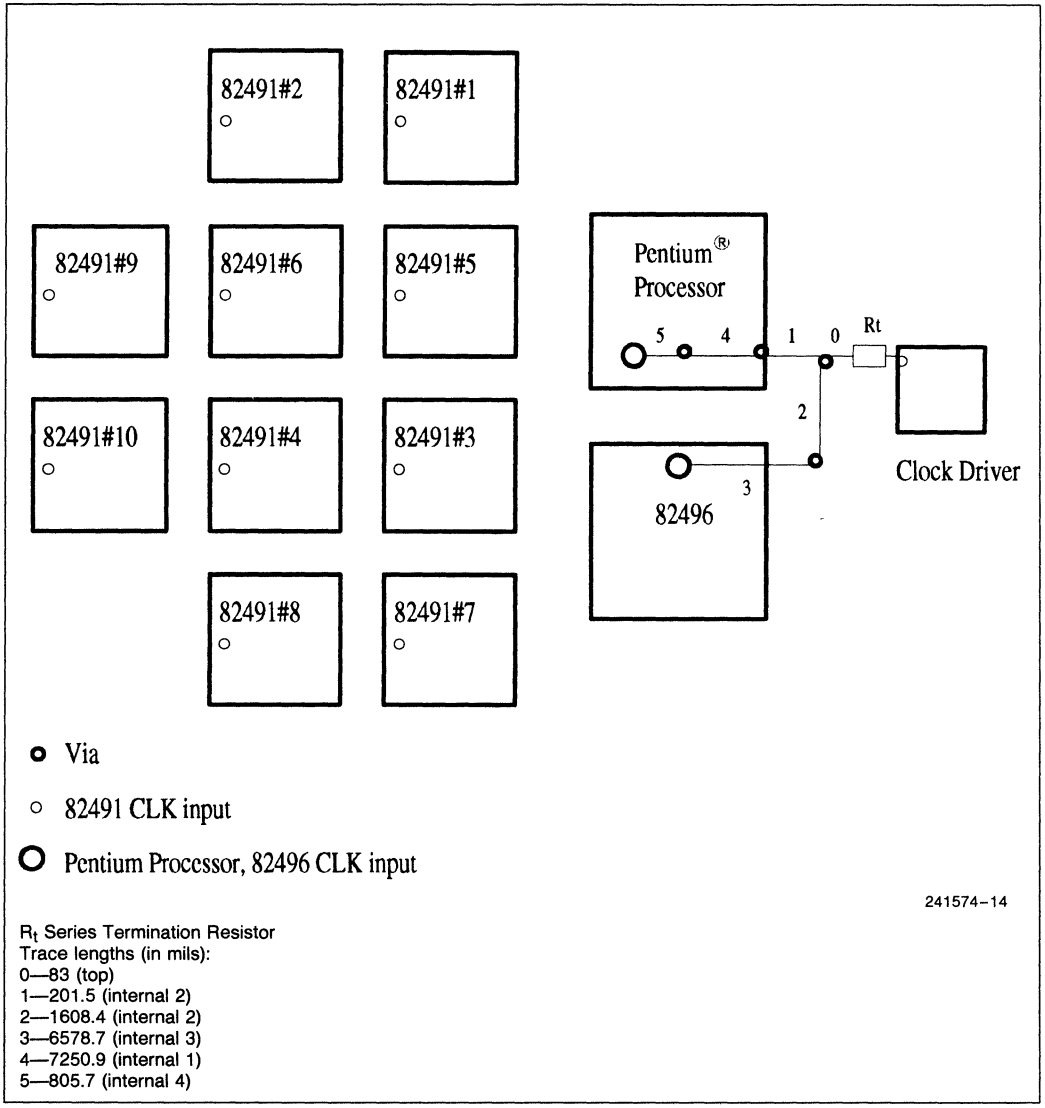
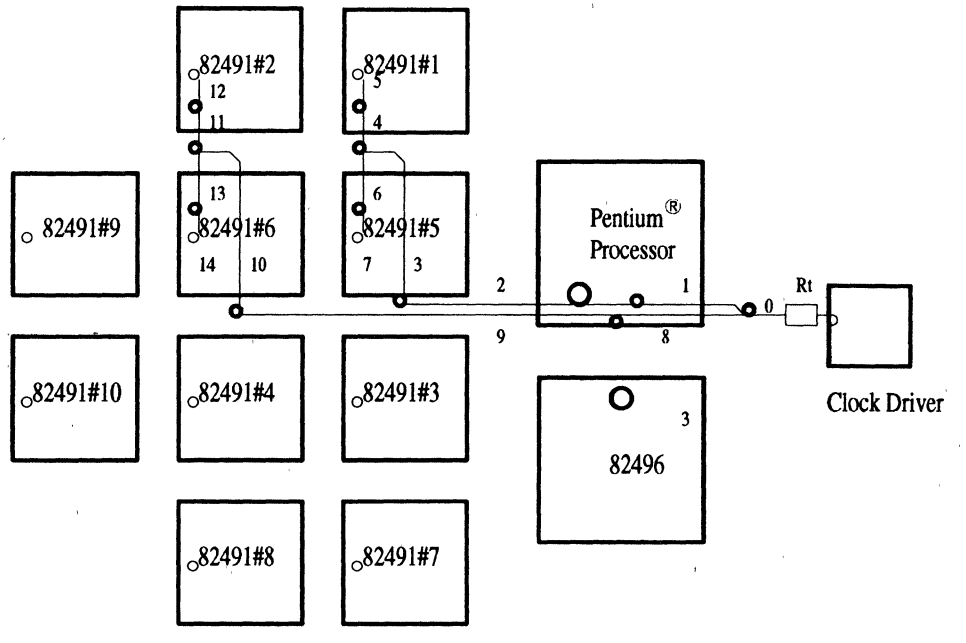


Figure 13. Pentium® Processor, 82496 and 82491 Clock Input Models



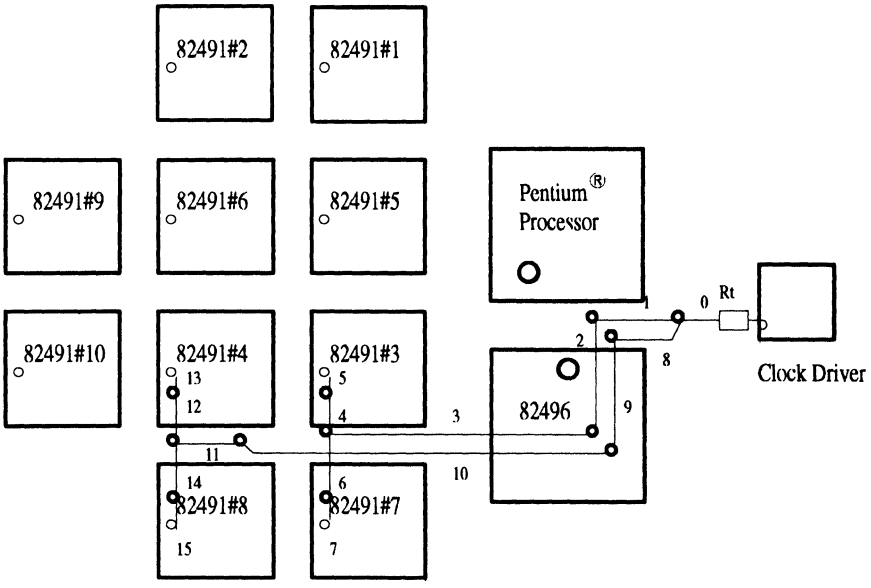
R_t Series Termination Resistor
Trace lengths (in mils):
0—83 (top)
1—201.5 (internal 2)
2—1608.4 (internal 2)
3—6578.7 (internal 3)
4—7250.9 (internal 1)
5—805.7 (internal 4)

Figure 14. CLK0 Layout for 256K Chip Set with Parity



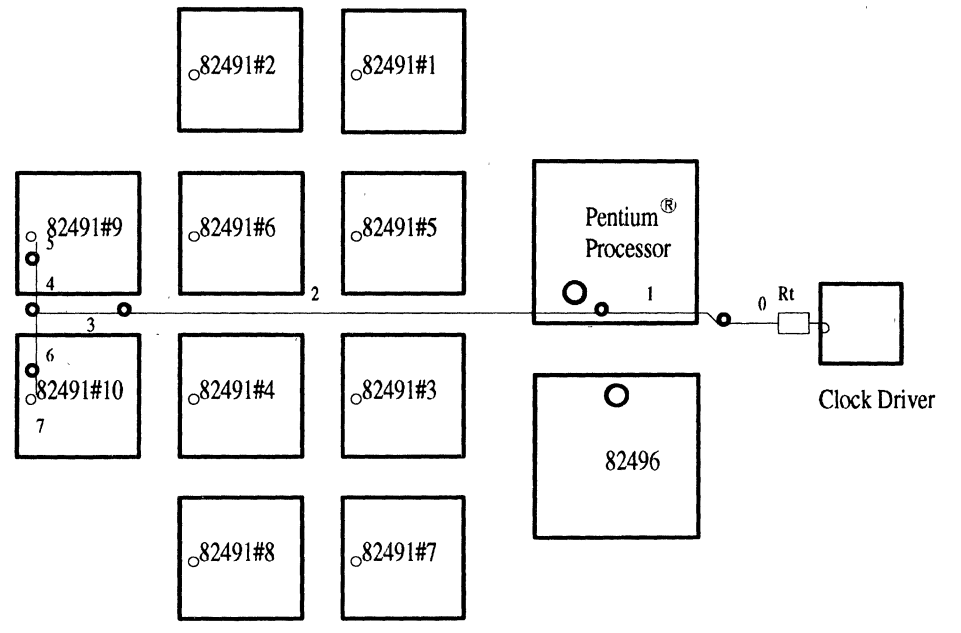
R_t Series Termination Resistor
Trace lengths (in mils):
0—78 (top)
1—455.3 (internal 2)
2—3639.3 (internal 1)
3—2180.7 (internal 4)
4—530.7 (internal 4)
5—75 (top)
6—727.2 (internal 4)
7—75 (top)
8—462.4 (internal 2)
9—4650 (internal 1)
10—1156.7 (internal 4)
11—526.5 (internal 4)
12—75 (top)
13—520.7 (internal 4)
14—75 (top)

Figure 15. CLK1 Layout for 256K Chip Set with Parity



R_t Series Termination Resistor
Trace lengths (in mils):
0—78 (top)
1—206.5 (internal 2)
2—3480.5 (internal 1)
3—2550.9 (internal 4)
4—560.2 (internal 4)
5—75 (top)
6—527.2 (internal 4)
7—75 (top)
8—160 (internal 2)
9—1236.0 (internal 2)
10—4290.9 (internal 3)
11—631.0 (internal 1)
12—530.7 (internal 4)
13—75 (top)
14—534.3 (internal 4)
15—75 (top)

Figure 16. CLK2 Layout for 256K Chip Set with Parity



R_t Series Termination Resistor
Trace lengths (in mils):
0—75 (top)
1—562.4 (internal 2)
2—5897.2 (internal 3); 7147.2 for National 74CT2527 as clock driver
3—631.0 (internal 4)
4—526.5 (internal 4)
5—75 (top)
6—520.7 (internal 4)
7—75 (top)

Figure 17. CLK3 Layout for 256K Chip Set with Parity

Figure 14 through Figure 17 show how clock signals are distributed from the clock driver to each component in the CPU-Cache chip set. Each clock line is tuned to minimize skew. Series termination is used on each line. Since the 82491, numbers 9 and 10, are parity chips, they are grouped onto the same driver output. Since the skew requirement between the Pentium processor and the 82496 is very tight (0.2 ns), they are also on the same driver output. All the loads on the same driver output can be tuned to have close to zero skew. Loads on different outputs, however, must contend with the output skew of the driver. CLK0 through CLK2 are layed out such that they branch off very close to the driver. For these lines, the transmission line characteristic impedance that the clock driver output sees can be treated as two resistors in parallel. In other words, the driver output sees half the impedance for CLK0, CLK1, and CLK2. The advantage of this scheme is that the value of the termination resistor is reduced dramatically. A smaller termination resistor helps faster rise and fall times. For CLK3, the branch off is at the end of the line, and thus the driver output sees the full characteristic impedance.

To achieve minimal skew, all the loads should be balanced, i.e., all the loads should be the same. For this design example, CLK1 and CLK2 have about twice the loads of CLK0 and CLK3. The load imbalance will add to the output skew quoted by manufacturers since on data sheets, manufacturers generally quote output skew for balanced loads. Since heavier loads see the transmitted signal later than lighter loads assuming the transmission lines are of the same length, traces for the lighter loads can be made longer to compensate for the discrepancy. CLK0 and CLK3 have longer traces from driver output to load than CLK1 and CLK2 traces. Since heavier loads (higher capacitance) have a longer rise time, and since for the CPU-Cache chip set skew measurements are taken at 0.8V, 1.5V, and 2.0V, to minimize skew at all free points, the termination resistors for CLK1 and CLK2 should be smaller than the termination resistors on CLK0. However, a smaller termination resistor than the value needed to perfectly terminate the line will result in a larger undershoot. When choosing termination values, it is a trade off among rise/fall times, skew, and undershoot.

When choosing a termination value, it is important to know the output impedance of the driver. For many TTL drivers, output rise impedance is different from output fall impedance. [Reference 3, Section 9] shows how to measure output impedance, or the driver manufacturer can be contacted for the information. Typically, output fall impedance is 5Ω–10Ω, and rise impedance is 5Ω–50Ω.

Figure 14 through Figure 17 are extensions to the layout topologies in the *Pentium® Processor, 82496, and 82491 256K CPU-Cache Chip Set Layout Example.* The routing topologies 15–18 shown in the example route the clock signals from the Pentium processor, 82496, and all the 82491's to the outside. The layout examples shown in this application note (Figure 14 through Figure 17) take the layout all the way to the clock driver, complete with termination. Although in the example, clock signals are routed toward the bottom and in the examples here, the clock signals are routed toward the side, the same principles apply. If routing toward the bottom is preferred, the same layouts as illustrated in Figure 14 through Figure 17 can be used with little or no modification.

5.2 Analysis of Drivers Used in Examples

Output models for MC10H646, CGS74CT2527, GA1086, and VSL4485 are used to drive the clock network described in Section 5.1. The clock networks shown in Figure 14 through Figure 17 were used. The simulations assume no variation in characteristic impedance and propagation speed for the board traces. Fast and slow simulations were performed. Three sigma clock driver models are used when available. Board traces are assumed to have plus/minus 10% variation in characteristic impedance and propagational speed. Table 6 shows the range of trace characteristics. Slow simulations assume the highest operating temperature the drivers expect to see, and slow interconnect characteristics. Fast simulations assume operating temperature to be zero and fast interconnect characteristics.

Table 6. Interconnect Characteristics

Corner	Trace Type	Z0(1) (Ω)	TD(2) (ns/ft)
Slow	Inner	58.5	2.41
Fast	Inner	71.5	1.85
Slow	Surface	72	2.05
Fast	Surface	108	1.35

NOTES:
1. Characteristic Impedance
2. Propagational Speed

Since simulation can only account for skew due to board trace and load imbalance, total skew is assumed to be the sum of the worst case output skew published by driver manufacturers and skew from simulation. Skew from simulation is derived by using identical driver models for each driver output, thus assuming zero

output skew. The board traces and termination resistances are tuned with 0.5 ns output skew in mind which leaves 0.2 ns for trace skew. For TriQuint's GA1086, the output skew is 0.25 ns; thus, there is a larger window for trace skew. Table 7 summarizes simulation results of the tightest parameters for each driver. All of the drivers can meet the 4 ns minimum high and low times easily. Most clock drivers guarantee a 45/55 duty cycle, which exceeds Intel's requirement.

Series termination resistors are chosen to minimize skew and undershoot. To achieve similar rise time for each load, termination resistance values are smaller for heavier loaded lines such as CLK1 and CLK2 compared to the resistance values for CLK0. Since CLK3 splits off at the end of the line, its termination value is about twice as CLK0's. Table 8 lists the termination values for each line and for each driver. Waveforms for each driver are attached. Notice the signals at the CLK input for each load is relatively clean whereas the signals at the driver side are not. Since the clock signals are only important to the component receiving the signal, how dirty the signal is at the driver end is not important, providing that the signal does not cause any damage or other ill effects on the driver.

Figures attached in this section show some waveforms from the simulations. V(201) is the voltage at the Pentium processor clock input, V(202) is the voltage at the 82496 clock input, and V(213), V(214), V(217), and V(218) are voltages at the 82491 clock inputs for the 82491s on CLK1 line. For 74CT2527, V(8) is the voltage at driver output, V(100) is the voltage at the junction of the series termination resistor and the beginning of board trace. For 10H646, V(9) is the voltage at driver output, V(20) is the voltage at the junction of the series termination resistor and the beginning of board trace.

Table 7. Compilation of Simulation Data

Mfr.	Clock Driver	Worst Skew P5-C5C (ns)(1)			Worst Skew C8C-Others (ns)			Worst Skew C8C (No Parity) (ns)(2)			Undershoot (−mV)(3)			Tr/Tf (ns)(4)		
		Slow	Fast	Spec	Slow	Fast	Spec	Slow	Fast	Spec	Slow	Fast	Spec	Slow	Fast	Spec
Motorola	10H646	0.021	0.023	0.2	0.65	0.67	0.7	0.65	0.67	0.7	468	816	1600	0.90/ 1.13	0.74/ 0.67	1.5/ 1.5
National	74CT2527	0.0071	(5)	0.2	0.67	(5)	0.7	0.61	(5)	0.7	285	(5)	1600	1.14/ 0.42	(5)	1.5/ 1.5
TriQuint	GA1086	0	0	0.2	0.55	0.45	0.7	0.45	0.45	0.7	150	400	1600	0.9/ 1.9 (6)	0.6/ 1.2	1.5/ 1.5
Vitesse	VSL4485	0.02	0.05	0.2	0.7	0.57	0.7	0.66	0.57	0.7	275	800	1600	0.95/ 0.78	0.78/ 0.6	1.5/ 1.5

NOTES:
1. All Skews are worst case numbers
2. Not using the parity chips
3. Worst Undershoot of all the CLK nodes
4. Slowest rise and fall times of all the CLK nodes
5. Only typical model at 25°C is available. Thus, only simulation performed is with slow interconnect corner
6. Simulation done on driver slow corner. Device specification for t_f is 1.4 ns worst case. Device was still under development when simulation was done. Please contact TriQuint for more information.

Clock distribution method for the memory bus controller (MBC) is very similar to that of the chip set. When distributing clocks for the MBC, be sure to load each driver output with similar loads as for the chip set, and route clock traces with similar lengths as for the chip set. For example, CLK1 and CLK2 have an aggregate load of about 20 pF, and the total clock trace length is about 7" from driver output to a load. To minimize the clock skew of the MBC clock from loads on CLK1 and CLK2 lines, the clock lines should fan out 2.9 pF per inch. Also, be sure to terminate the line properly. It is important to keep the loading similar to the loadings on clock lines of the chip set if skew is to be kept close to 0.7 ns. Adjusting trace lengths and termination resistance can compensate for load imbalance to a degree, but not perfectly and not always.

Simulations results provided here are based on best available models at the time. Some models were for parts still under development at the time of simulation. Therefore, the simulation results are subject to change.

Table 8. Series Termination Resistor Values for Each Line

Manufacturer	Clock	CLK Line	R_t (Ω)
Motorola	10H646	CLK0	26
		CLK1	20
		CLK2	20
		CLK3	59
National	74CT2527	CLK0	20
		CLK1	15
		CLK2	15
		CLK3	45
TriQuint	GA1086	CLK0	30
		CLK1	30
		CLK2	30
		CLK3	50
Vitesse	VSL4485	CLK0	32
		CLK1	23
		CLK2	23
		CLK3	48

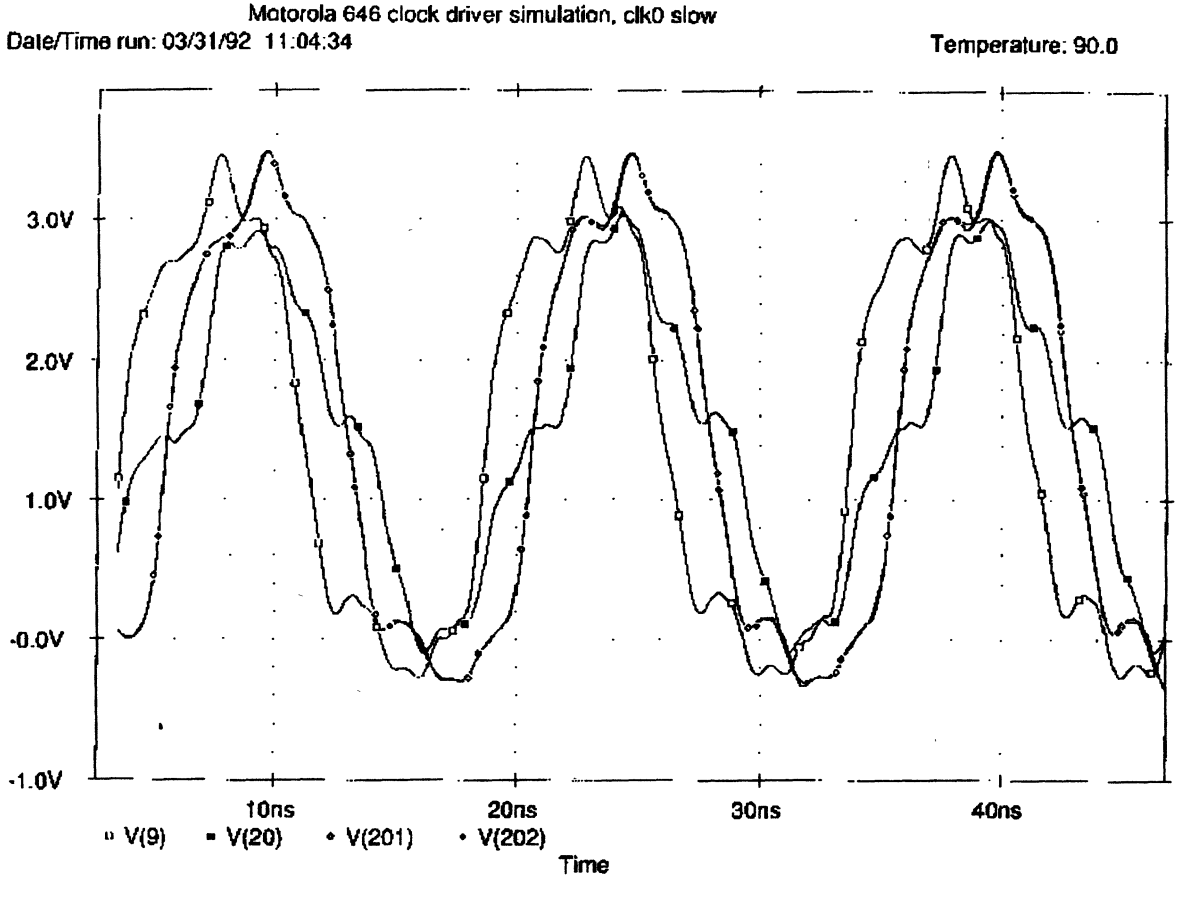


Figure 18. Motorola Waveform

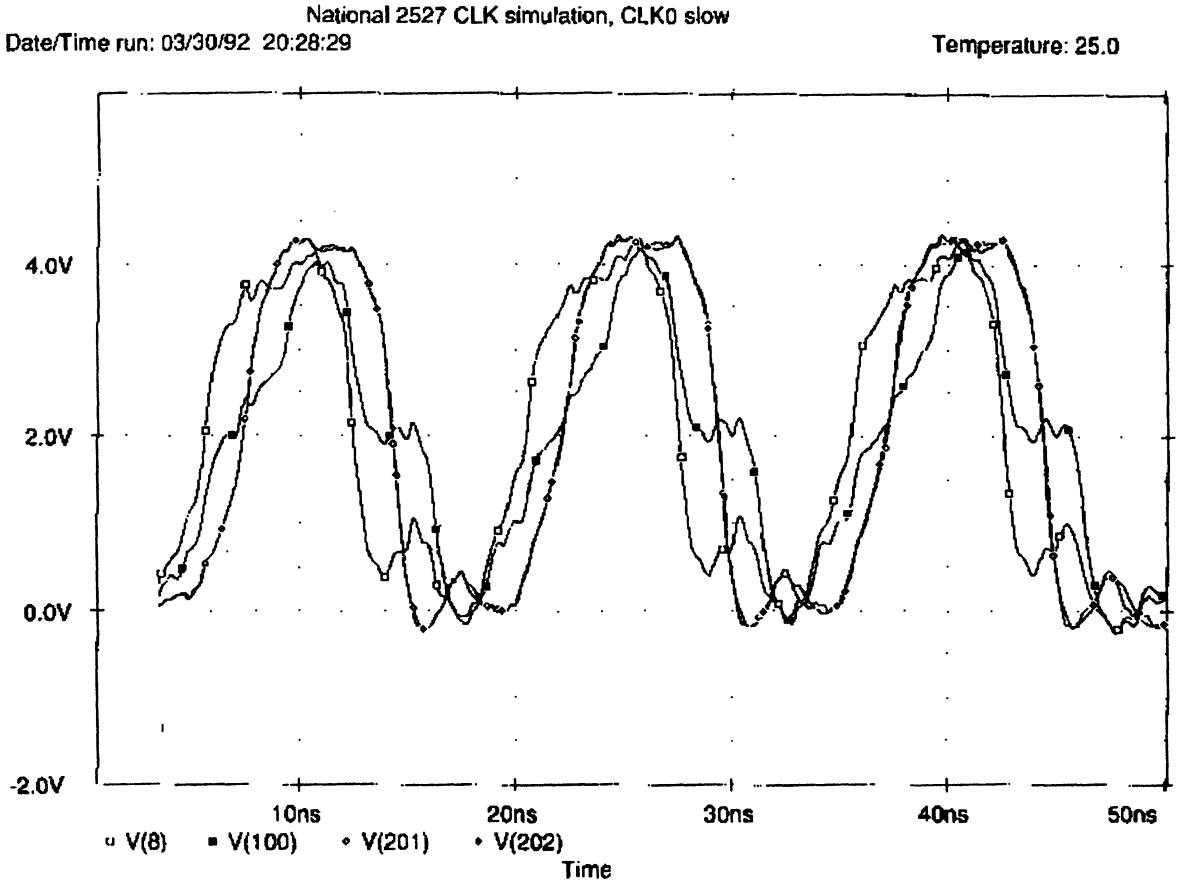


Figure 19. National Waveform

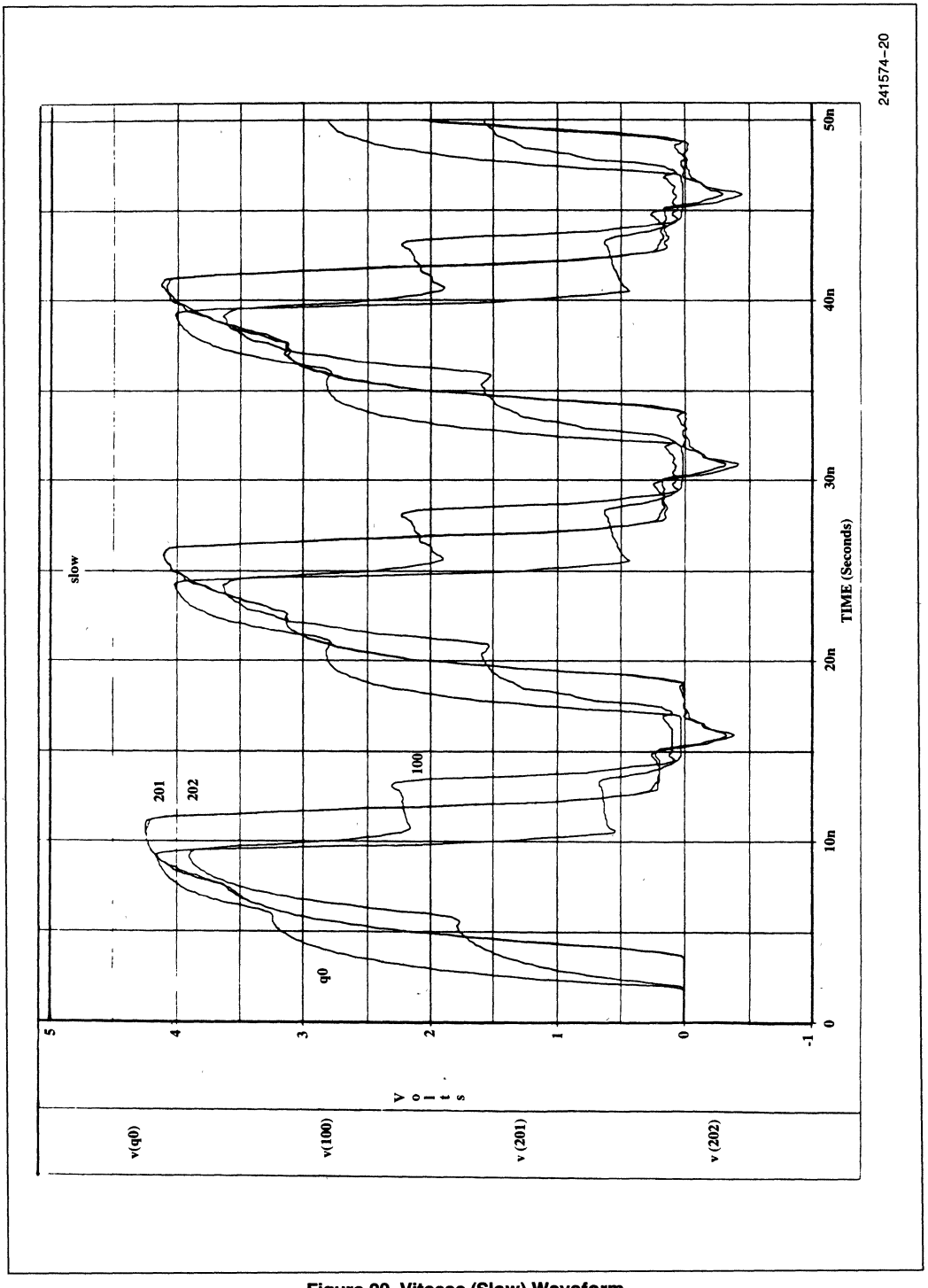


Figure 20. Vitesse (Slow) Waveform

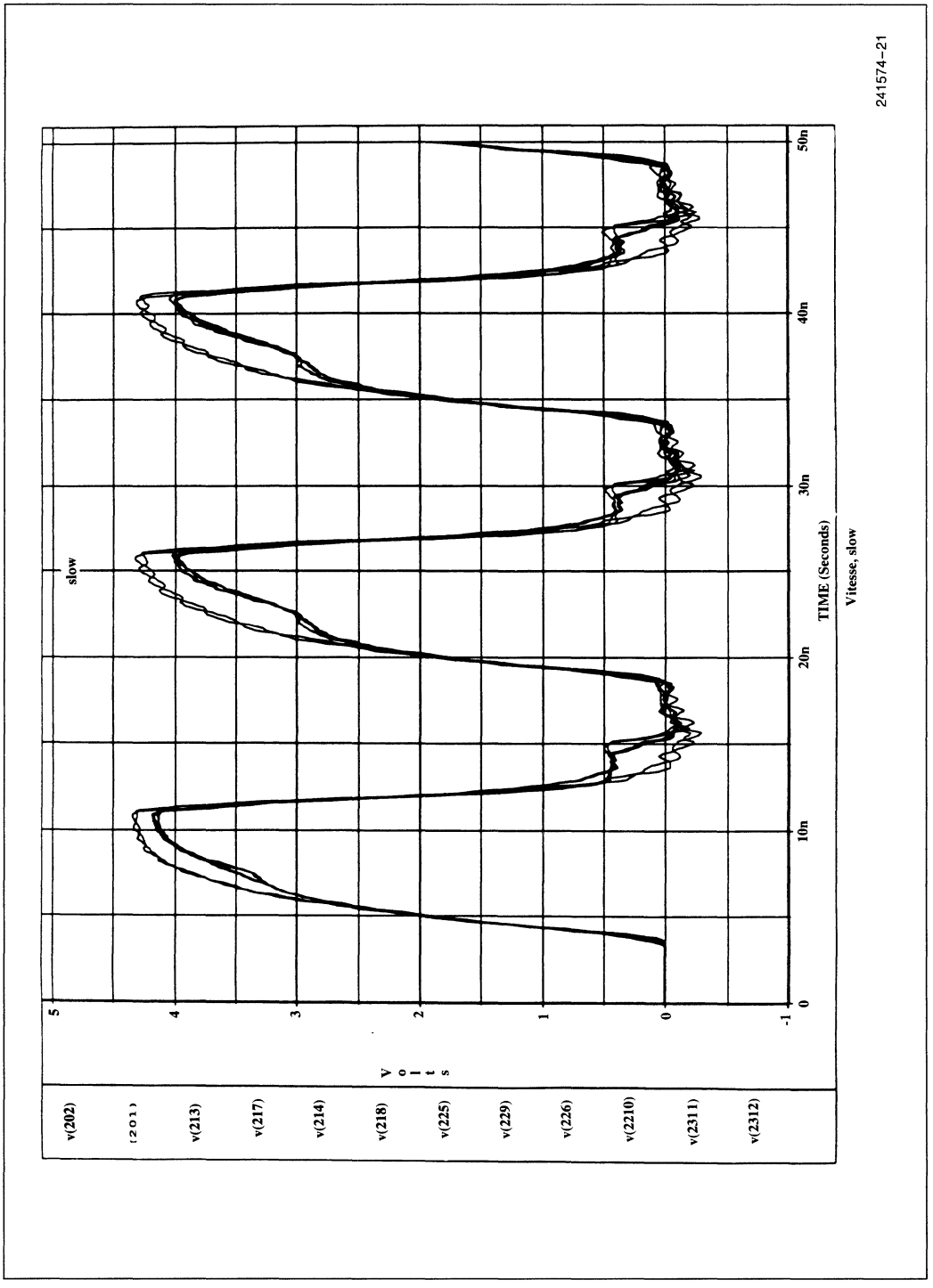


Figure 21. Vitesse (Slow) Waveform (Continued)

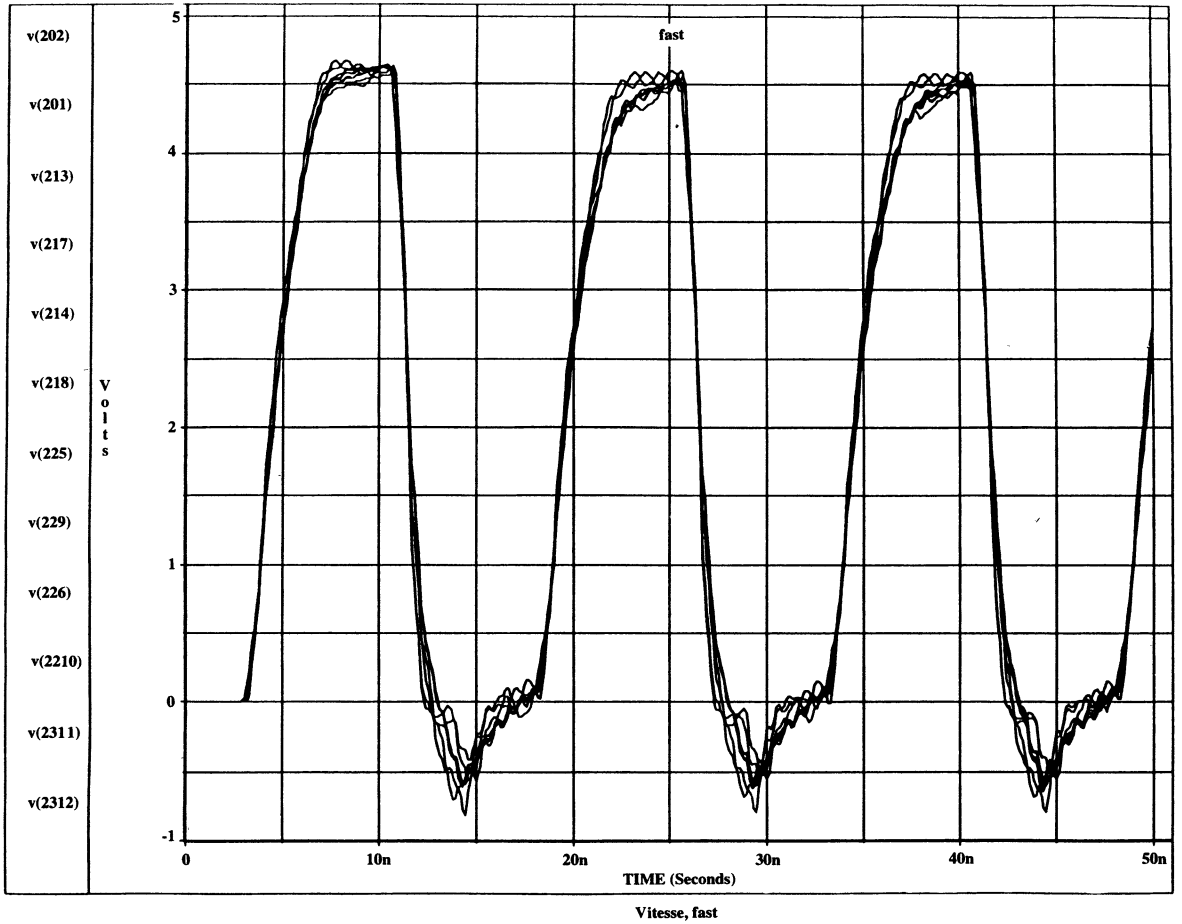


Figure 22. Vitesse (Fast) Waveform

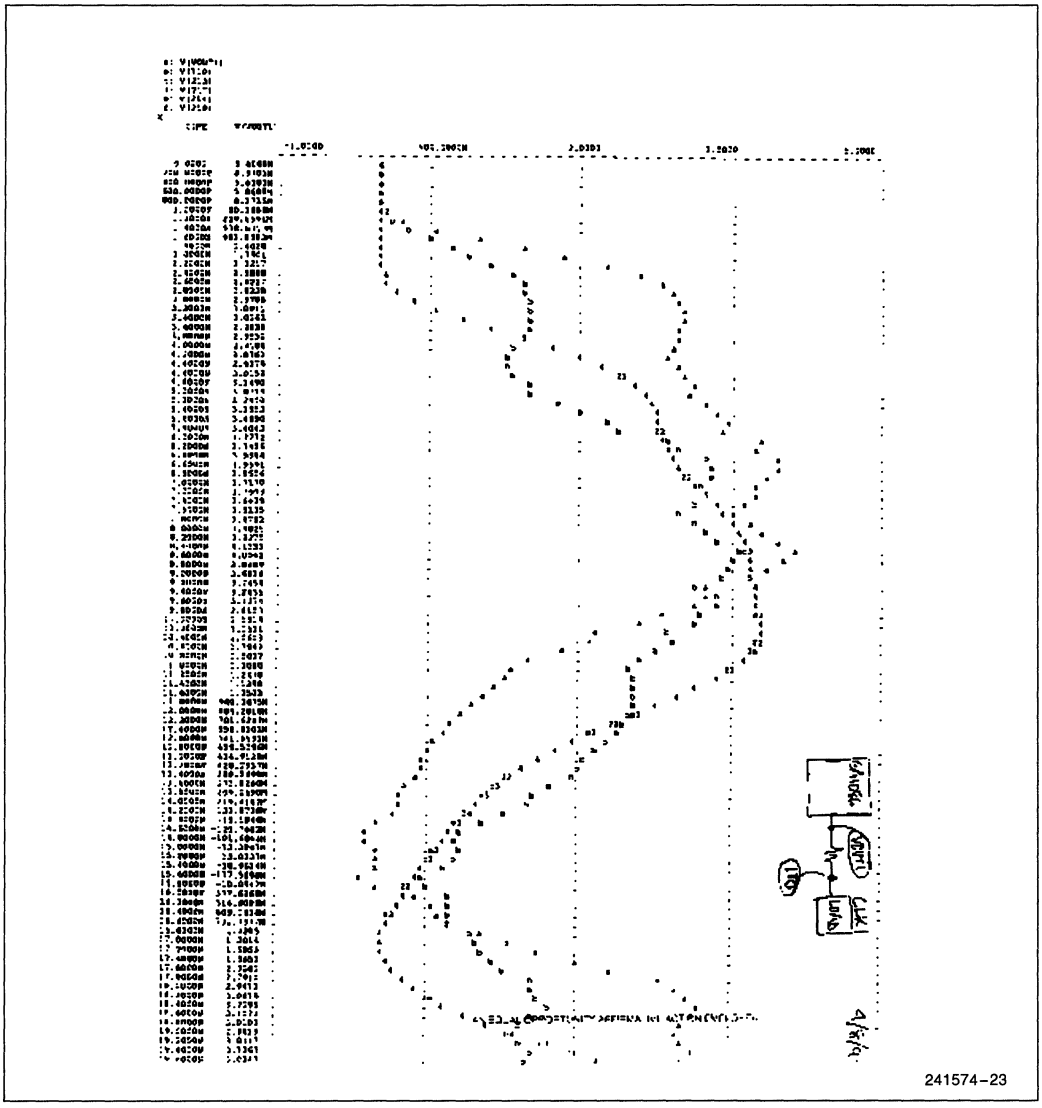
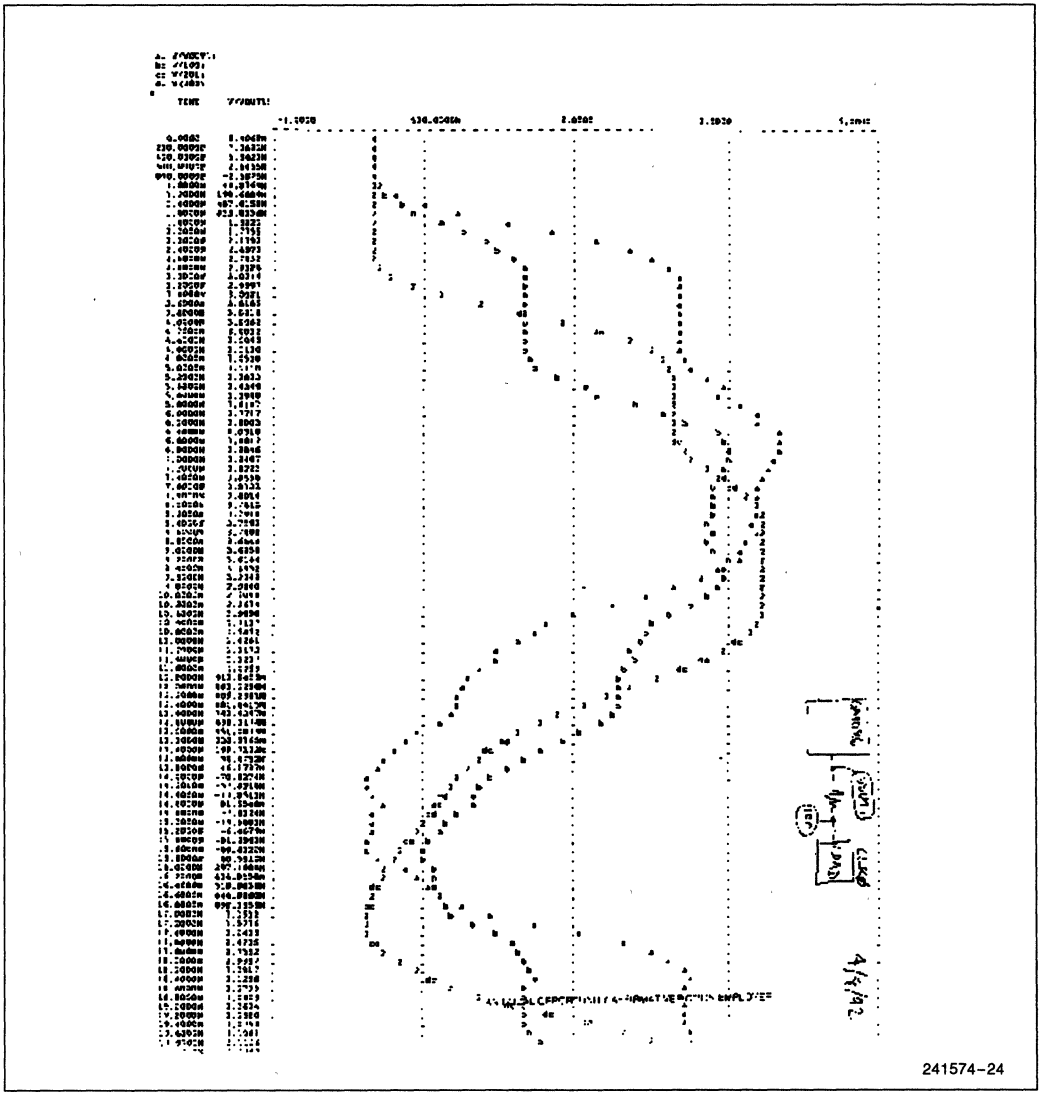


Figure 23. Triquint Waveform



241574-24

Figure 24. Triquint Waveform (Continued)

6.0 Pentium® PROCESSOR WITH 512K 82496/82491 SECOND LEVEL CACHE CLOCK DISTRIBUTION ISSUES

Clock distribution for 512K CPU-Cache chip set can be done in the same way as for the 256K chip set. Since there are more SRAM chips for the 512K cache, there are more loads that need clocking. Including parity, there are 18 82491s. Once again, the same principles apply. Keep the driver loading as close to balanced as possible. Tune traces and adjust termination resistance so that skew is minimized.

7.0 CLOCK DISTRIBUTION FOR THE Pentium® PROCESSOR WITH OTHER SECOND LEVEL CACHES

The Pentium processor can be used with cache configurations other than with the 82496 and 82491, as well as, without a second level cache. With other caches, the first thing that must be done is to decide how much skew is tolerable. Then, decide on which clock driver to use and carefully layout clock signals for distribution. If skew requirements do not exceed the CPU-Cache chip set requirements, the same drivers and the same distribution can be used. Design examples in Section 5.0 serve as a guide to how to distribute clocks for Pentium processor systems with tight skew.

If the Pentium processor is used without a second level cache, and only a small number of 66 MHz signals are needed, there are a few more options for clock drivers. For example, Motorola's 88915 has one 2x output that can run to maximum 70 MHz. Texas Instruments has the ABT337, 338, and 339 that can provide four copies of 66 MHz signals.

8.0 SUMMARY

At high speeds, clock synchronization becomes a difficult problem. Clock traces must be treated as transmission lines. Proper termination must be given to the lines to ensure good signal quality. The Pentium processor, with operating frequencies of 60 MHz and 66 MHz, has tight clock requirements. Together with the 82496 Cache Controller and 82491 Cache SRAM, the CPU-Cache chip set must be synchronized with minimal skew.

For the Pentium processor clocking, the most critical parameters are skew and rise and fall times. Depending on the memory interface to the CPU-Cache chip set, there are many ways of generating multiple copies of clock signals.

Fully synchronous designs need to route 66 MHz only, but with minimal skew for all of them. Divided synchronous designs require both 66 MHz and 33 MHz signals. Asynchronous designs need to worry about the CPU-Cache chip set clock generation and distribution as well as the MBC.

Several clock drivers have been analyzed in detail with carefully tuned clock routing and the proper termination such that the clock signals transmitted to the Pentium processor, 82496, and 82491 meet all the timing requirements of the Intel chip set parts. Loading on a clock driver should be as balanced as possible. Clock traces should have equivalent length from driver output to load. The clock lines should be terminated properly to minimize reflections.

The same design principles used in the 256K CPU-Cache chip set clocking example can be applied to other CPU-cache configurations, or to a cacheless interface.

This application note has listed a number of devices from several different manufacturers. The purpose of this list is to supply a starting point for finding a clocking solution that meets each system's specific requirements. The lists provided are not meant as an endorsement or guarantee of the devices listed. In addition, these lists are not a complete listing of devices. These or other manufacturers may offer additional devices that meet the clock specifications for the Pentium processor.

9.0 REFERENCES

1. Intel Corporation, *Pentium® Processor Family Developer's Manual*, Order Number: 241563.
2. Intel Corporation, *Designing with the Pentium® Processor, 82496, and 82491 256K CPU-Cache Chip Set*, Order Number: 241576.
3. Jolly, Rich, *Clock Design in 50 MHz Intel486™ Systems*, Application Note AP-453, 1991, Intel Corporation, Santa Clara, CA.
4. Blood, William R., Jr., *MECL System Design Handbook*, 1988, Motorola Inc.
5. Hanke, Chris and Tharalson, Gary, *Low Skew Clock Drivers and their System Design Considerations*, Application Note AP-1091, Motorola Inc., 1990.

APPENDIX A
CLOCK DRIVER MANUFACTURERS

The following is a list of contacts for the clock driver manufacturers listed in this application note. It is not meant to be an exhaustive list of all possible solutions. It is meant as a starting point for system designers to assist in finding a clock solution that meets their system requirements.

AMCC

United States:
Headquarters
6195 Lusk Boulevard
San Diego, CA 92121-2793
Ph: 619-450-9333 or 800-PLL-AMCC (755-2622)
FAX: 619-450-9885

Europe:

Amega Electronics
Basingstoke, RG24OPF, U.K.
Ph: 011/44-256-843166

Japan:

Teksel Co., Ltd.
Kawasaki 213
Tokyo, Japan
Ph: 011/81-448127430

Israel:

EIM
Petach Tiqva, Israel
Ph: 011/972-3-9233257

AT&T Microelectronics

AT&T Customer Response Center
Ph: 800-372-2447 x773

Danny George
555 Union Blvd.
Allentown, PA 18103
50N2G2100
Ph: 215-439-6697

Cypress

Sean Dingman
3901 N. 1st St.
San Jose, CA 95134
408-943-2743

ICS

Bruce Rogers
Technical Marketing Manager
2626 Van Buren Ave.
P.O. Box 968
Valley Forge, PA 19482
215-666-1900

Intel PLD BU International Contact List

United States:

John Van Sack
Intel Corporation
FM4-42
1900 Prairie City Road
Folsom, CA 95630
Ph: (916) 356-3964
FAX: (916) 356-6949

Europe:

Tony O'Sullivan
Intel Corporation GmbH
Dornacher Str. 1
PostFach 213
D-8016 FeldKirchen/Munchen
Germany
Ph: (49) 89/90992-340
FAX: (49)89/9043948

Japan:

Norikazu Aoki
5-6 Tokodia, Tsukuba-shi
Ibaraki-Ken 300-26
Japan
Ph: 0298-47-0721
FAX: 0298-47-8819

APAC:

Eric Chan
Intel Technology SDN BHD
Bayan Lepas Free Trade Zone,
Box 121
11900 Penang
Malaysia
Ph: 604-820-7271
FAX: 604-836-405

Motorola Inc.

Todd Pearson
Motorola Inc
2200 W. Broadway Rd.
Mesa, Arizona 85202
USA
Ph: (602) 962-3410

Masanori Matsubara
Nippon Motorola LTD
3-20-1, Minami-Azabu
Minato Ku, Tokyo 106
Japan
Ph: 81-33-280-8383

Axel Krepil
Motorola GMBH
Schatzbogen 7
8000 Munchen 81
Germany
Ph: 49-89-92103-167

Derek Leung
Motorola Hong Kong LTD
Silicon Harbour Center
2 Dai King Street
Taipo Industrial Estate
Taipo N. T. Hong Kong
Ph: 852-666-8194

National Semiconductor

National Semiconductor
Santa Clara, CA
Tony Ochoa
Ph: 408-721-6804
Ph: 800-272-9959

Pioneer Semiconductor

Joe Kraus
2343 Bering Dr.
San Jose, CA 95131
Ph: 408-435-0800
FAX: 408-435-1100

Texas Instruments

United States:

Steve Plote
Program Manager
CLOCK DRIVERS
8330 LBJ Freeway, Center 3
P.O. Box 655303
Dallas, Texas 75265
Ph: 214-997-5214

Brett Clark
Applications Engineer
Ph: 903-868-5836

Japan:

Mich Komatsu
Texas Instruments Japan LTD.
M.S. Shibaura Bldg. 13-23
Shibaura 4-Chome
Minato-ku Tokyo, 108 Japan
Ph: 033-769-8717

Asia Pacific Region:

Eric Wey
Texas Instruments Taiwan LTD.
Taipei Branch
10F Bank Tower, 205 Tung Hua N.
Taipei, Taiwan ROC
Ph: 886-2-713-9311

Europe:

Lothar Katz
Texas Instruments
8050 Freising, Fed. Rep. of Ger.
Deutschland GMBH
Haggertystr. 1
Ph: 49-816-80314

TriQuint Semiconductor

United States:

Marketing, Sunil Sanghavi
(408) 982-0900 x142, FAX (408) 982-0222
Western Sales, Mark Wu
(408) 982-0900 x113, FAX (408) 982-0222
Central Sales, John Watson
(214) 422-2532, FAX (214) 423-4947
East Sales, Mike Zyla
(215) 493-6944, FAX (215) 493-7418
International, Mike Kilgore
(503) 644-3535 x228, FAX (503) 644-3198

Europe - GiGA A/S

Fin Helmer, President
45-4-343-1588, FAX 45-4-343-5967

Japan - Japan Macnica Corp.

Shin Ishikawa, Product Manager
045-939-9140, FAX 045-939-6141

Vitesse Semiconductor

United States:

Corporate Headquarters
Vitesse Semiconductor Corporation
741 Calle Plano
Camarillo, CA 93012
Ph: 805-388-7501
FAX: 805-388-7565

Europe:

Thomson Composants Micronodes
Route Departementale 128 B.P. 48
91401 Orsay Cedex France
Ph: 33-1-60-19-7000
FAX: 33-1-60-19-7140

Japan:

H.Y. Associates Co., LTD.
3-1-10, Sekimachikita, Nerima-Ku
Tokyo, 177 Japan
Ph: 81-33-929-7111
FAX: 81-33-928-0301

Korea:

Beaver International, Inc.
3601 Deauville Court
Calabasas, CA 91302
Ph: 818-591-0356
FAX: 818-591-0753

Taiwan:

Tamarack Microelectronics
16 Fl., No. 1, Fu-Hsing N. Road
Taipei, Taiwan, ROC
Ph: 886-2-772-7400
FAX: 886-2-776-0545

intel®

AP-480

APPLICATION NOTE

Pentium® PROCESSOR THERMAL DESIGN GUIDELINES REV. 2.0

2

November 1995

Order Number: 241575-003

Pentium® PROCESSOR THERMAL DESIGN GUIDELINES REV. 2.0

CONTENTS

CONTENTS

1.0 INTRODUCTION

In a system environment, the Pentium® processor's temperature is a function of both the system and component thermal characteristics. The system level thermal constraints imposed on the package are local ambient temperature and thermal conductivity (i.e., airflow over the device). The Pentium processor thermal characteristics depend on the package (size and material), the type of interconnection to the printed circuit board (PCB), the presence of a heat sink, and the thermal conductivity and the power density of the PCB.

All of these parameters are aggravated by the continued push of technology to increase the operating speeds and the packaging density. As operating frequencies increase and packaging size decreases the power density increases and the heat sink size and airflow become more constrained. The result is an increased importance on system design to ensure that thermal design requirements are met for each component in the system.

In addition to heat sinks and fans, there are other solutions for cooling integrated circuit devices. A few of these solutions are: fan mounted on heat sink, heat pipes, thermoelectric (peltier) cooling, liquid cooling, etc. While these alternatives are capable of dissipating additional heat, they have disadvantages in terms of system cost, complexity, reliability, and efficiency. These techniques are more expensive than a passive heat sink and fan. The introduction of active devices can also decrease reliability. Finally, the power efficiency of some of these techniques is poor, and gets worse as the amount of power being dissipated increases. Despite these disadvantages, each of these solutions may be the right one for particular system implementations.

However, for the purpose of this application note, Intel has focused its efforts on describing solutions using passive heat sinks and fans.

1.1 Document Goal

The goal of this document is to provide thermal performance information for the Pentium processor and recommendations for meeting the thermal requirements imposed on systems. This application note attempts to provide an understanding of the thermal characteristics of the Pentium processor and some examples of how the thermal requirements can be met.

2.0 IMPORTANCE OF THERMAL MANAGEMENT

Thermal management of an electronic system encompasses all of the thermal processes and technologies that must be employed to remove and transfer heat from individual components to the system's thermal sink in a controlled manner.

The objective of thermal management is to ensure that the temperature of all components is maintained within functional and absolute maximum limits. The functional temperature limit is the range within which the electrical circuits can be expected to meet their specified performance requirements. Operation outside the functional limit can degrade system performance or cause logic errors. The absolute maximum temperature limit is the highest temperature that a portion of the component may be safely exposed. Temperatures exceeding the limit can cause physical destruction or may result in irreversible changes in operating characteristics. Higher temperatures result in earlier failure of the devices in the system. Every 10°C rise above the operating range means a halving of the mean time between failures.

3.0 Pentium® PROCESSOR POWER SPECIFICATIONS

The Pentium processor's power dissipation and case temperature specs for 60 MHz and 66 MHz are shown in Table 1.

To ensure functionality and reliability of the Pentium processor, maximum device junction temperature must remain below 90°C. Considering the power dissipation levels and typical ambient environments of 40°C to 45°C, the Pentium processor's junction temperatures cannot be maintained below 90°C without additional thermal enhancement to dissipate the heat generated by this level of power consumption.

The thermal characterization data described in Table 2 illustrates that both a heat sink and airflow are needed. The size of heat sink and the amount of airflow are interrelated and can be traded off against each other. For example, an increase in heat sink size decreases the amount of airflow required. In a typical system, heat sink size is limited by board layout, spacing, and component placement. Airflow is limited by the size and number of fans along with their placement in relation to the components and the airflow channels. In addition, acoustic noise constraints may limit the size or types of fans limiting the airflow.

To develop a reliable thermal solution, all of the above variables must be considered. Thermal characterization and simulation should be carried out at the entire system level accounting for the thermal requirements of each component.

4.0 THERMAL PARAMETERS

Component power dissipation results in a rise in temperature relative to the temperature of a reference point. The amount of rise in temperature depends on the net thermal resistance between the junction and the reference point. Thermal resistance is the key factor in determining the power handling capability of any electronic package.

Thermal resistance from junction to case (θ_{JC}), and from junction to ambient (θ_{JA}) are the two most often specified thermal parameters for integrated circuit packages.

4.1 Ambient Temperature

Ambient temperature is the temperature of the undistributed ambient air surrounding the package. Denoted T_A, ambient temperature is usually measured at a specified distance away from the package. In the laboratory test environment, ambient temperature is measured 12 inches upstream from the package under investigation. In a system environment, ambient temperature is the temperature of the air upstream to the package and in its close vicinity.

Table 1. Pentium® Processor Power Dissipation

	Package Type	Total Pins	Pin Array	Package Size	Power (Typical)	Power (Max)	Max Case Temp (°C)
Pentium Processor 60 MHz	PGA	273	21 x 21	2.16" x 2.16"	11.9W	14.6W	80
Pentium Processor 66 MHZ	PGA	273	21 x 21	2.16" x 2.16"	13W	16W	70

4.2 Case Temperature

Case temperature, denoted T_C, is measured at the center of the top surface (on top of the heat spreader, see Figure 1) of the package, typically the hottest point on the package case. Special care is required when measuring the case temperature to ensure an accurate temperature measurement. Thermocouples are often used to measure T_C. Before any temperature measurements, the thermocouples have to be calibrated. When measuring the temperature of a surface which is at a different temperature from the surrounding ambient air, errors could be introduced in the measurements. The measurement errors could be due to having a poor thermal contact between the thermocouple junction and the surface, heat loss by radiation or by conduction through thermocouple leads. To minimize the measurement errors, it is recommended to use the following approach:

- Use 36 gauge or finer diameter K, T, or J type thermocouples. The laboratory testing was done using a thermocouple made by Omega (part number: 5TC-TTK-36-36).
- Attach the thermocouple bead or junction to the center of the package top surface using high thermal conductivity cements. The laboratory testing was done by using Omega Bond (part number: OB-100).
- The thermocouple should be attached at a 90° angle as shown in Figure 1. When a heat sink is attached a hole (no larger than 0.15") should be drilled through the heat sink to allow probing the center of the package as shown in Figure 1.
- If the case temperature is measured with a heat sink attached to the package, drill a hole through the heat sink to route the thermocouple wire out.

4.3 Junction Temperature

Junction temperature, denoted T_J, is the average temperature of the die within the package.

The junction temperature for a given junction-to-ambient thermal resistance, power dissipation, and ambient temperature is given by the following formula:

$$T_J = P_D * \theta_{JA} + T_A$$

If a heat sink with thermal resistance of θ_{SA} (sink-to-ambient) is used, then the thermal resistance from the junction-to-case, θ_{JC}, is given by the following formula:

$$T_J = P_D * (\theta_{JC} + \theta_{CS} + \theta_{SA}) + T_A$$

where:

θ_{CS} is the thermal resistance from the component (case) to the heat sink.

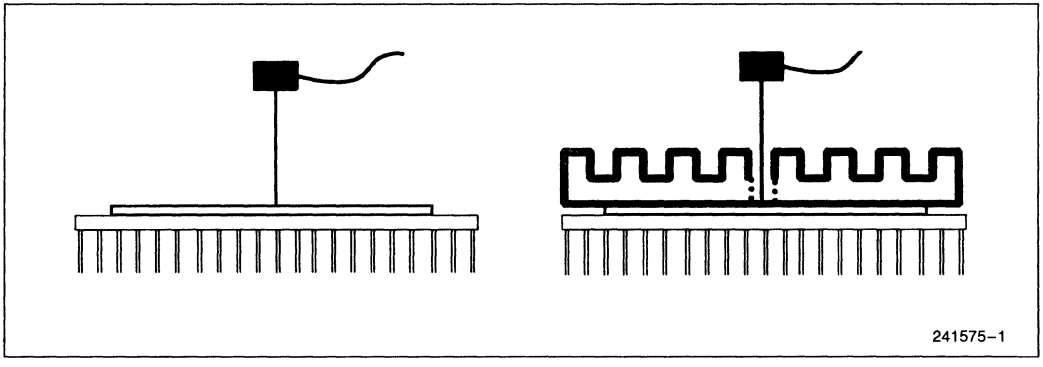


Figure 1. Thermocouple Attachment

4.4 Thermal Resistance

Thermal resistance (Figure 2) values for junction-to-ambient, θ_{JA} and junction-to-case, θ_{JC}, are used as measures of IC package thermal performance. θ_{JC} is a measure of the package's internal thermal resistance along the major heat flow path from silicon die to package exterior. This value is strongly dependent on the material, thermal conductivity, and geometry of the package. θ_{JC} values also depend on the location of the reference point (in this case center of the package top surface), the external cooling configurations and the heat flow paths from the package to the ambient. For example, if a heat sink is attached to the package top surface or more heat is pulled into the board through the pins, the θ_{JC} values measured with reference to the center of the package top surface will change. θ_{JA} values include not only internal thermal resistance, but also the radiative and convective thermal resistance from the package exterior to ambient air. θ_{JA} values depend on the material, thermal conductivity, and geometry of the package and also on ambient conditions such as airflow rates and coolant physical properties.

In order to obtain thermal resistance values, junction temperature is measured using the temperature sensitive parameter (TSP) method. With this method, special design thermal test structures are used which are approximately the same size as the Pentium processor die. The test structure consists of resistors and diodes. Resistors are used to simulate the Pentium processor power dissipation and thereby heat up the package. Diodes, which are located at the center of the thermal test die, are used to measure the die temperature. The measurements are carried out in a wind tunnel environment. The air flow rate and the ambient temperature are measured 12 inches away from the package in the upstream air.

The parameters are defined by the following relationships:

$$\theta_{JA} = (T_J - T_A)/P_D$$
$$\theta_{JC} = (T_J - T_C)/P_D$$
$$\theta_{JA} = \theta_{JC} + \theta_{CA}$$

where:

θ_{JA} = junction-to-ambient thermal resistance (°C/W)
θ_{JC} = junction-to-case thermal resistance (°C/W)
θ_{CA} = case-to-ambient thermal resistance (°C/W)
T_J = average die (junction) temperature (°C)
T_C = case temperature at a pre-defined location (°C)
T_A = ambient temperature (°C)
P_D = device power dissipation (W)

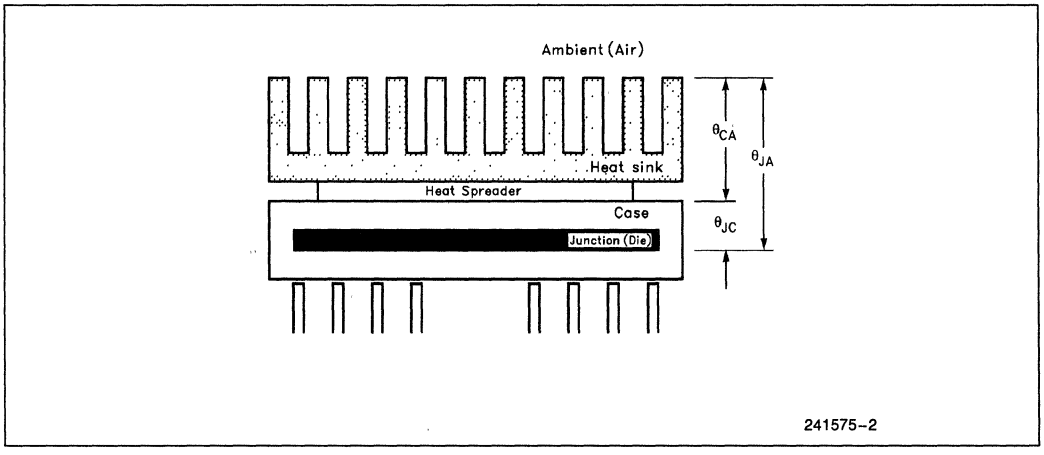


Figure 2. Thermal Resistance Parameters

Table 2 lists the junction-to-case and case-to-ambient thermal resistances for the Pentium processor (with and without a heat sink).

Table 2. Thermal Characterization Data

	θ_{JC}	θ_{CA} **vs Airflow (ft/min.)					
		0	200	400	600	800	1000
With 0.25" Heat Sink	0.6	8.3	5.4	3.5	2.6	2.1	1.8
With 0.35" Heat Sink	0.6	7.4	4.5	3.0	2.2	1.8	1.6
With 0.65" Heat Sink	0.6	5.9	3.0	1.9	1.5	1.2	1.1
Without Heat Sink	1.2	10.5	7.9	5.5	3.8	2.8	2.4

NOTE:
Heat Sink: 2.1 in^2 base, omni-directional pin Al heat sink with 0.050 in. pin width, 0.143 in. pin-to-pin center spacing and 0.150 in. base thickness. Heat sinks are attached to the package with a 2 to 4 mil thick layer of typical thermal grease. The thermal conductivity of this grease is about 1.2 w/m c.
** θ_{CA} values shown in this table are typical values. The actual θ_{CA} values depend on the air flow in the system (which is typically unsteady, non-uniform and turbulent) and thermal interactions between Pentium CPU and surrounding components through PCB and the ambient.

2

5.0 DESIGNING FOR THERMAL PERFORMANCE

At this point the application note turns from describing the characteristics that define thermal performance to describing how designers should use these characteristics to assess thermal requirements of PC system designs. The Pentium processor specifies a maximum case temperature, T_C, of 70°C @ 66 MHz. This case temperature limit along with the Pentium processor's power and thermal resistance characteristics can be used to determine the ambient temperature required to keep the Pentium processor operating within its specified limits. Using these parameters in the following equations:

$$T_A = T_C - (P^* \theta_{CA})$$

$$T_A = 70°C - (16W^* 10.5°C/W)$$

$$T_A = -98°C$$

The maximum ambient temperature required in a Pentium processor system without any additional thermal enhancement is −98°C at 66 MHz. Obviously, this ambient temperature is impractical and unachievable in a PC system. In order to be able to maintain the case temperature at 70°C in a typical system ambient with air temperature of 40°C to 45°C, the thermal resistance between the case and the ambient must be reduced.

5.1 Heat Sinks

The most common way to improve the package thermal performance is to increase the surface area of the device by attaching a large piece of metal (a heat sink) to the package. The heat sink is usually made of Aluminum and is chosen for its price/thermal-performance ratio. There are materials that offer higher conductivity such as copper, but cost becomes prohibitive. To maximize the flow of heat for a given junction temperature rise over the ambient temperature, the thermal resistance from heat sink to air can be reduced by a) maximizing the surface area, and b) maximizing the air flow across the surface area (maximizing air flow **through** heat sink fins in most cases).

Intel has used test data to determine what size of heat sink and airflow is needed to properly cool a Pentium processor system. The data was derived assuming an adhesive attach process that offers thermal resistance of about 0.2°C/W.

The testing was done in a wind tuunel in the configuration (in Figure 3) where the heat sink was mounted on a real Pentium processor package with a thermal die mounted inside to generate the 16W of power. The package is then mounted in a socket which is soldered to a 2-layer PCB that brings power to the die.

Based on these tests, three specific heat sink and airflow combinations have been identified that properly dissipate the Pentium processor's 16W and maintains a case temperature below 70°C @ 66 MHz. The three heat sinks are shown in Figure 4.

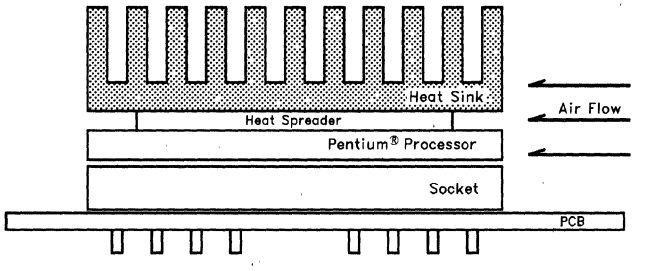


Figure 3. Improving Thermal Performance

Heat sink A: **H = 1.4" for 200 LFMs of Air**
Heat sink B: **H = 1.05" for 300 LFMs of Air**
Heat sink C: **H = 0.85" for 400 LFMs of Air**
Assumption: Air Temp = 45°C

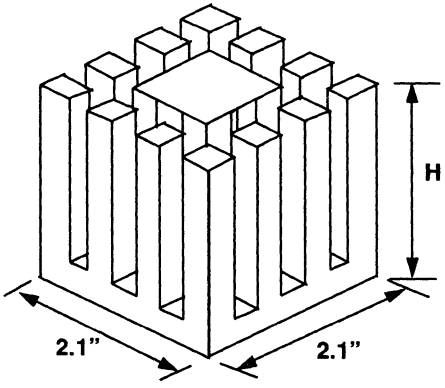


Figure 4. Recommended Combinations

In addition, testing has been done to provide more general guidelines which allow deviating from the above conditions. These guidelines allow systems to derive various combinations of heat sink size and airflow that ensure the Pentium processor thermal specifications are met. For example, by increasing the heat sink x-y dimensions and extending it over the package footprint, the heat sink height can be reduced while maintaining the same thermal performance as the taller heat sink with the same footprint as that of the package. The first three charts (Figures 5, 6, 7) show the thermal resistance as a function of heat sink size and airflow. The last three charts (Figures 8, 9, 10) show the power dissipation achievable with a given heat sink size and airflow. The power dissipation calculations assume T_C = 70°C @ 66 MHz, T_A = 45°C, and θ_{JC} = 0.6°C/W.

$$P_{max} = (T_C - T_A)/\theta_{CA} = 25/\theta_{CA}$$

A key assumption in all of these calculations is that a perfect thermal connection can be achieved between the case and the heat sink. One ban extrapolate the heat sink solutions by adding the additional thermal resistance of any chosen heat sink attach process. See Appendix B for case to heat sink thermal interface options.

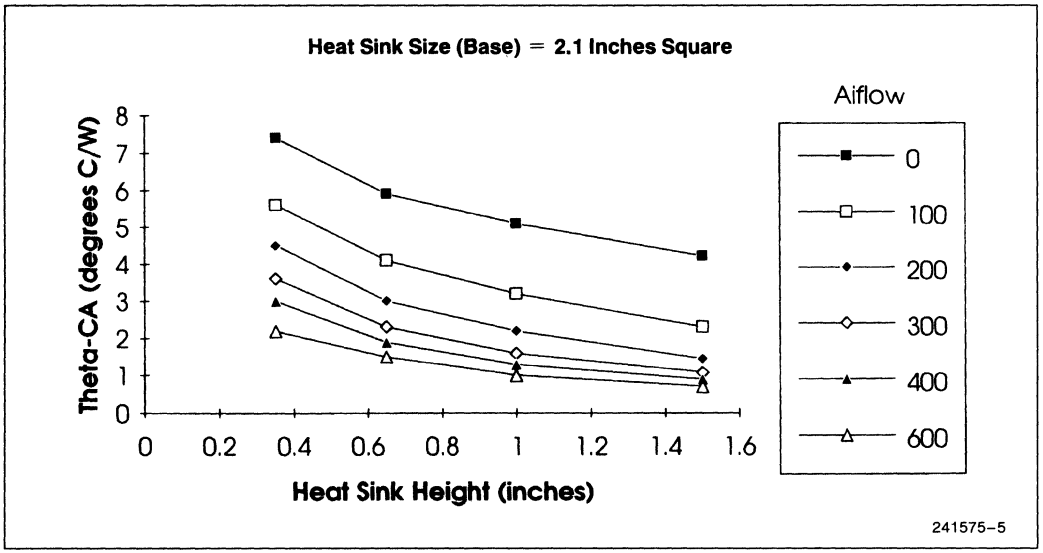


Figure 5. Thermal Resistance

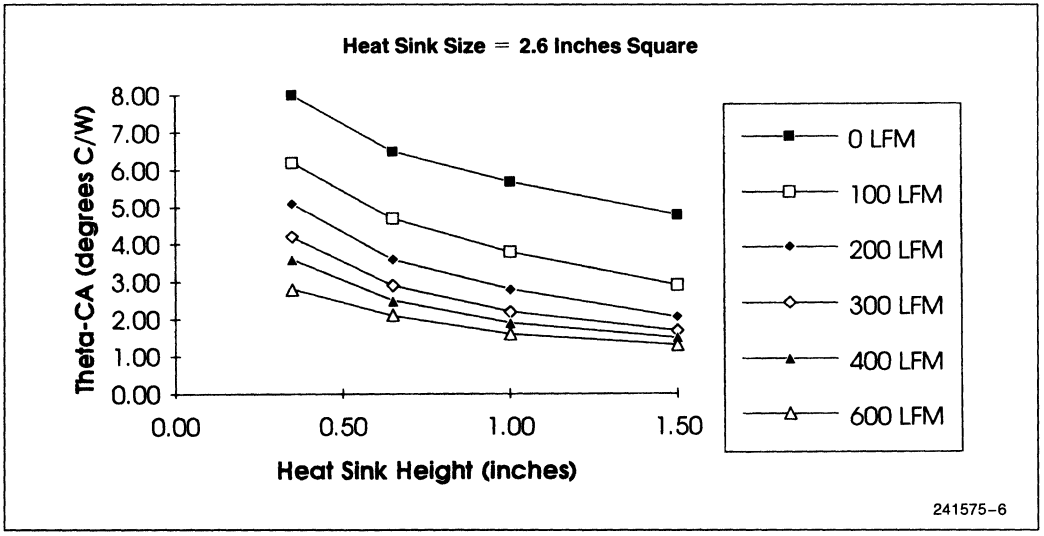


Figure 6. Thermal Resistance

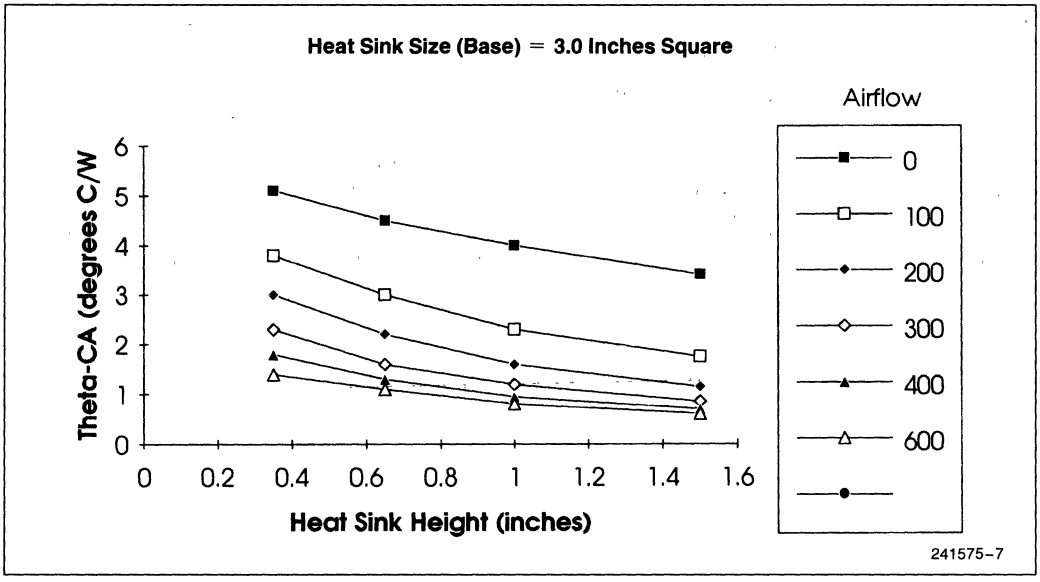


Figure 7. Thermal Resistance

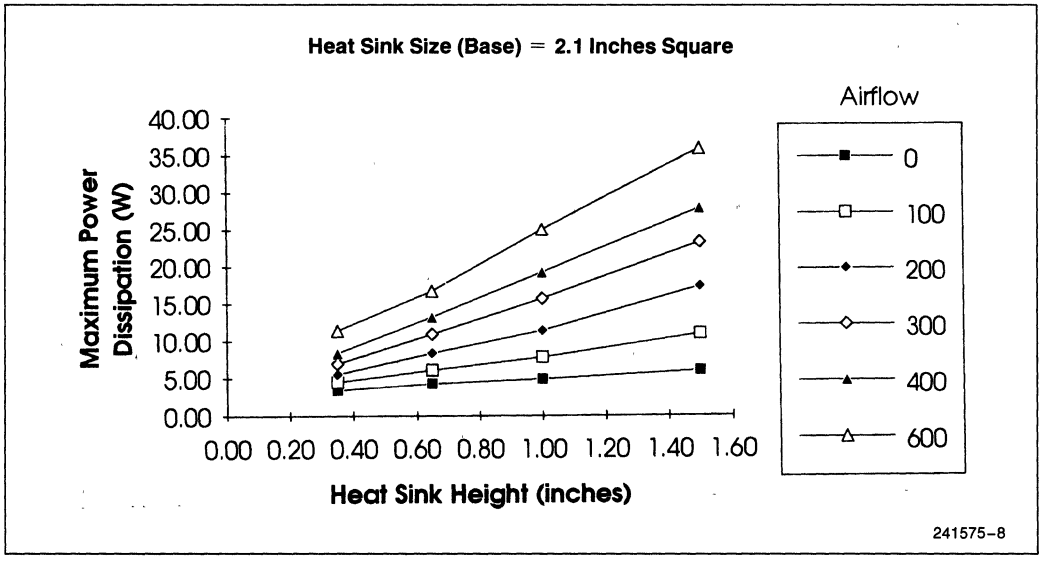


Figure 8. Power Dissipation

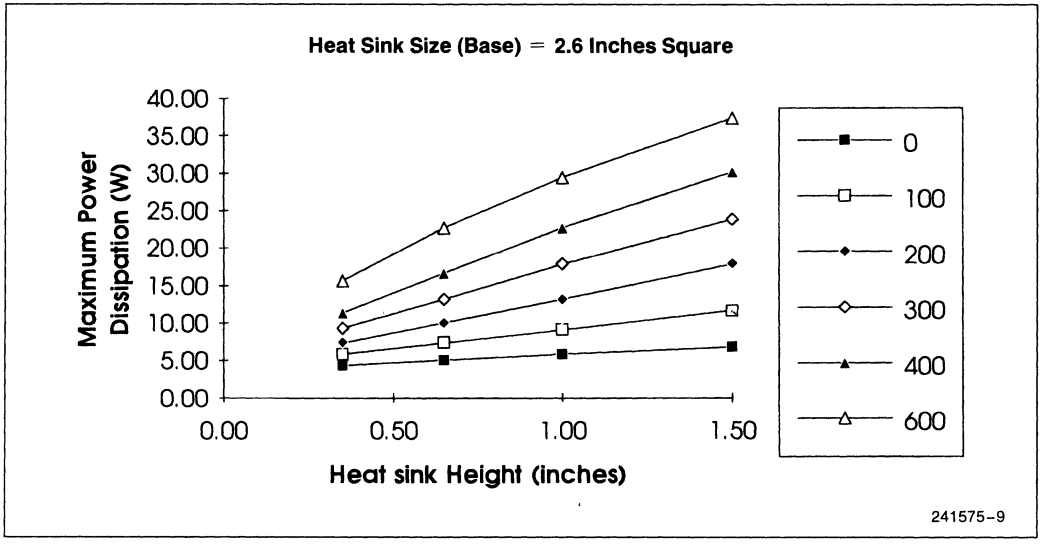


Figure 9. Power Dissipation

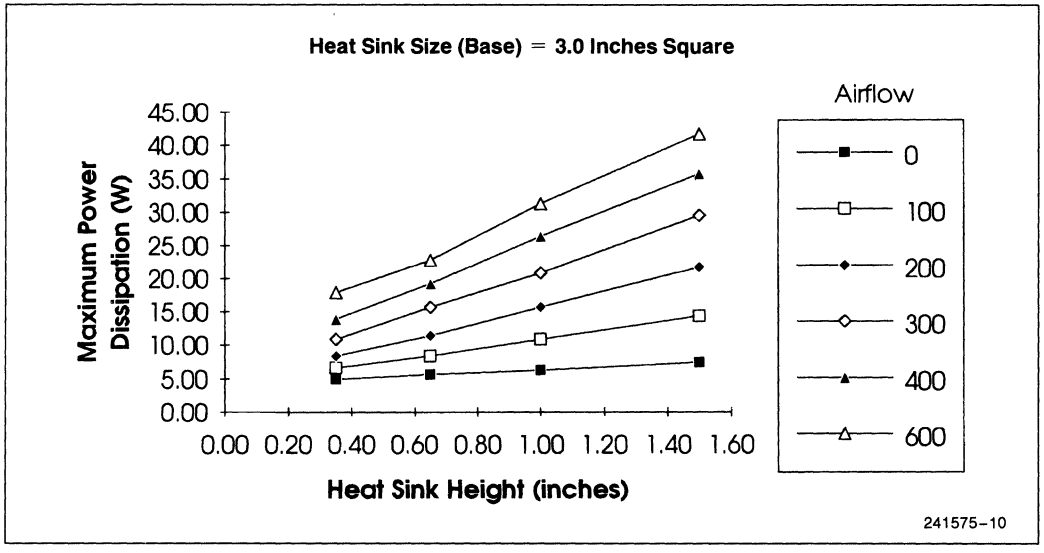


Figure 10. Power Dissipation

5.2 Airflow

To improve the effectiveness of heat sinks it is important to manage the airflow so as to maximize the amount of air that flows over the device or heat sink's surface area. In the system, the air flow around the processor can be increased by providing an additional fan or increasing the output of existing fan. If this is not possible, baffling the airflow to direct it across the device may help. This means the addition of sheet metal or objects to guide the air to the target device. Often the addition of simple baffles can eliminate the need for an extra fan. In addition, the order in which air passes over devices can impact the amount of heat dissipated.

5.3 Fans

Fans are often needed to assist in moving the air inside a chassis. A typical variable speed fan capable of up to 100 CFM of air can be found for approximately $10.

The airflow rate is usually directly related to the acoustic noise level of the fan and system. Therefore maximum acceptable noise levels may limit the fan output or the number of fans selected for a system.

A fan may be placed at the top of a heat sink to produce direct air impingement on the heat sink for efficient heat removal. A key issue with fans is their reliability. Although many fans are rated for approximately 50,000 hours of operation, operating conditions such as operating temperature, pressure drop across the fan and the particles in the air can significantly reduce the fan's useful life.

5.4 Thermal Performance Validation

The recommended thermal solutions in Section 5.1–5.3 are only design guidelines. Since there are many variables that can effect thermal performance in an actual system, thermal performance should always be validated experimentally, following the case temperature measurement technique described in Section 4.2.

6.0 CONCLUSION

As the complexity of today's microprocessors continues to increase so do the power dissipation requirements. Care must be taken to ensure the additional heat resulting from the power is properly dissipated. As documented, the heat can be dissipated using passive heat sinks, fans and/or active cooling devices.

The simplest and probably most cost effective method is to use a heat sink and a fan. The size of the heat sink and the output of the fan can be varied to balance the tradeoffs between size and space constraints versus noise. For example, if space is available a 1.4″ high heat sink can be used with only 200 LFM to cool the 66 MHz Pentium processor and the 16W it dissipates. Another example in which space is restricted shows a 0.7″ high heat sink can be used if approximately 500 LFM of airflow is provided.

These are not the only valid solutions, but both provide adequate cooling to maintain the Pentium processor case temperature at or below the 70°C @ 66 MHz specified limit. By maintaining this specification, the system can guarantee proper functionality and reliability of the Pentium processor.

APPENDIX A
EXAMPLES

Thermal management examples, designing with the Pentium processor. Using the best known methods.

Appendix Goal

The goal of this appendix is to measure the operating temperatures in a real system versus the wind tunnel laboratory measurements. These experiments are done with heat sinks that are similar to the ones suggested in Section 5.1 of the main document. The thermocouples and attachment methods suggested in Section 4.2 of the main document are also used. The appendix begins by reviewing the variables that the system designer has control over and uses tables to describe thermal resistance in the context of where the system designer can have the most effect The importance of the case to heat sink thermal interface and correct attachment methods are reviewed and different options given. The appendix proceeds to describe the system used for these tests and the tools and equipment needed. The lab set up procedures are discussed in detail and the measurements are presented with comments at the conclusion.

WHAT ARE THE VARIABLES?

Table A-1 shows the cooling options that customers can control when designing a system. From Table A-1 it is obvious that changing the heat sink and air flow are the two most effective ways for a system designer to affect the thermal performance of a system.

Table A-1. Variables

COOLING OPTIONS UNDER CUSTOMER CONTROL	
Variables	**Options for Cooling**
Device	• Use Power Management SW in the System • Clock the Device at a Lower Speed
Heat Sink	• Increase Heat Sink Area, Width or Height • Use Interface Materials with Lower Thermal Resistance • Use Active Cooling Devices • Use a Combination of Active and Passive Cooling Devices
Air Flow	• Increase Air Flow • Manage Air Flow • Place Hottest IC's near Highest Airflow
Ambient Temperature	• Place System in a Controlled Climate

Figure A-1 sums up the thermal tradeoff picture succinctly. Looking at Figure A-1 reiterates the previous statement that increasing the heat sink size and air flow rate provide the largest thermal performance improvements. In addition it shows the variables that are constant. Note that the θ_{JC} (junction-to-case thermal resistance) of the Pentium processor is fixed and a system designer can have no effect on this parameter. Also note that the θ_{CS} (case-to-heat sink thermal resistance) is a constant. Even though θ_{CS} is shown as a constant in Figure A-1 it can move up and down the Y axis depending on the interface material chosen. The case to heat sink interface is critical to the overall success of the thermal solution and cannot be overlooked. The next section will go into detail on this subject.

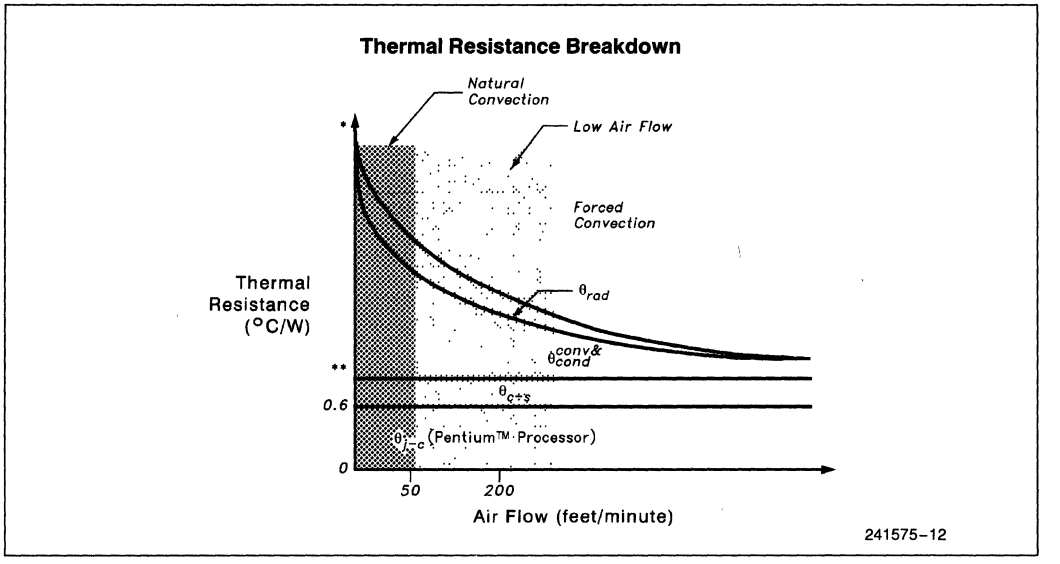


Figure A-1. Thermal Resistance

The main of purpose Figure A-2 is to show that packages and heat sinks are not perfectly flat and this requires that the air gap be filled with an interface material that has a lower thermal resistance than air. The whole point is to try and minimize the contact thermal resistance. The different types of thermal interface materials are listed to show the wide array of materials available to the system designer. Intel's data books have a mechanical section that list the flatness of the package. Heat sink vendors should be able to provide specifications for their heat sink offerings.

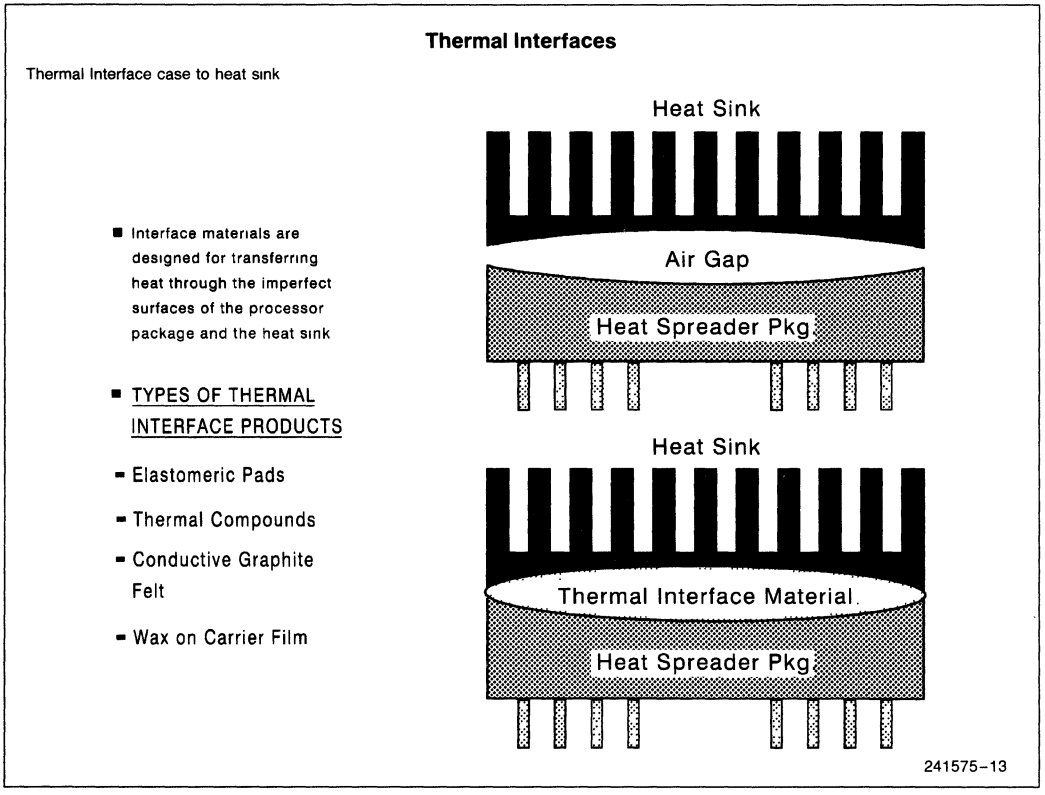


Figure A-2. Thermal Interfaces

The next section (Figures A-3 through A-5) covers attachment methods which generally fall into the categories shown; epoxies, double sided tapes or manual clips to either chip or socket.

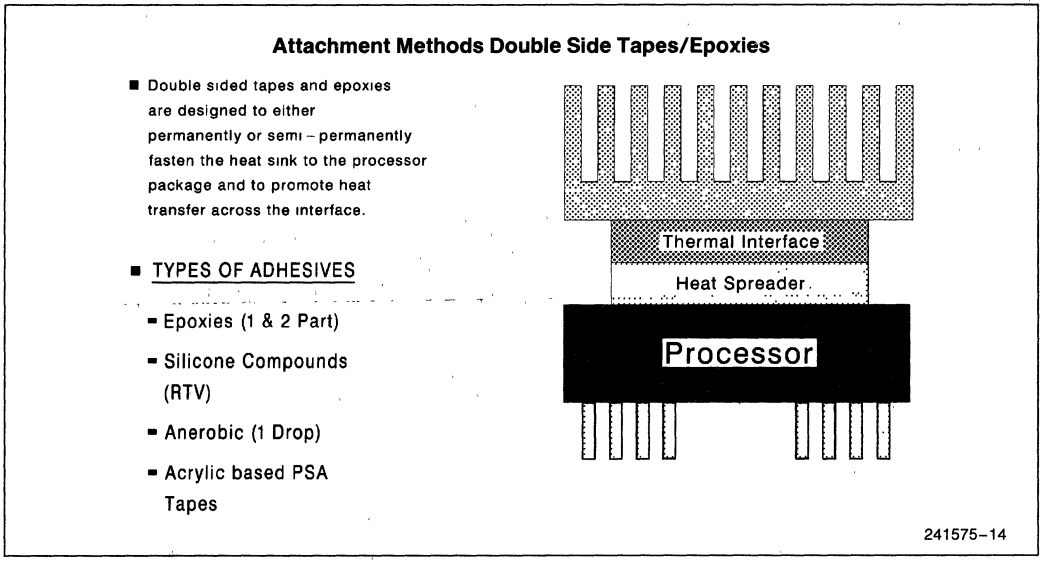


Figure A-3. Attachment Methods

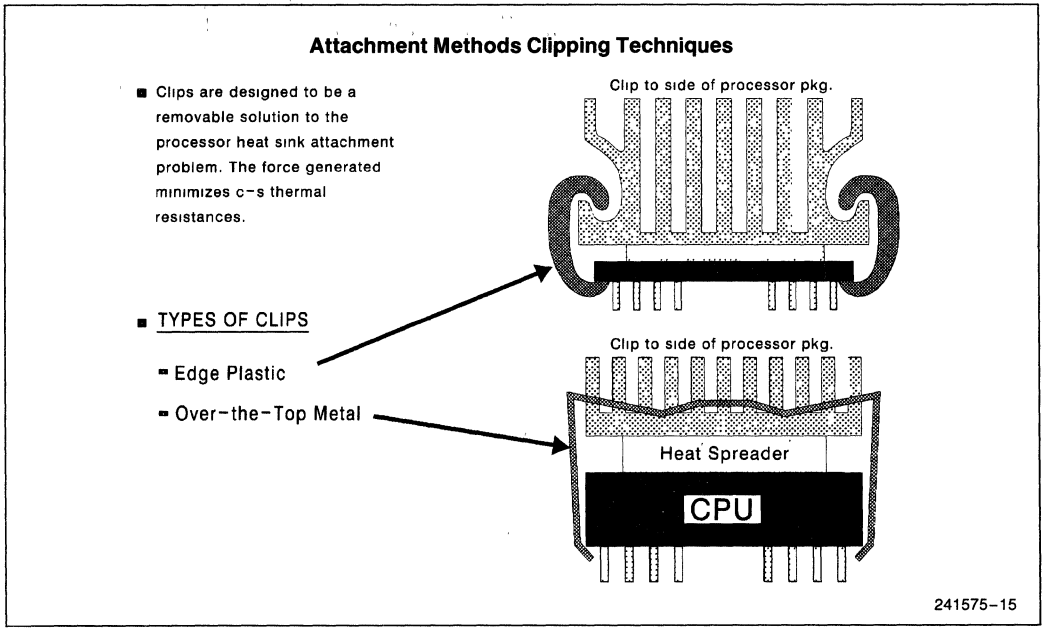


Figure A-4. Attachment Methods

Note that some clips don't allow the package to be pushed all the way into the socket and this could be a problem with short lead packages. The main advantage of this type of system is that a low profile socket can be used to lower the height of the processor heat sink assembly.

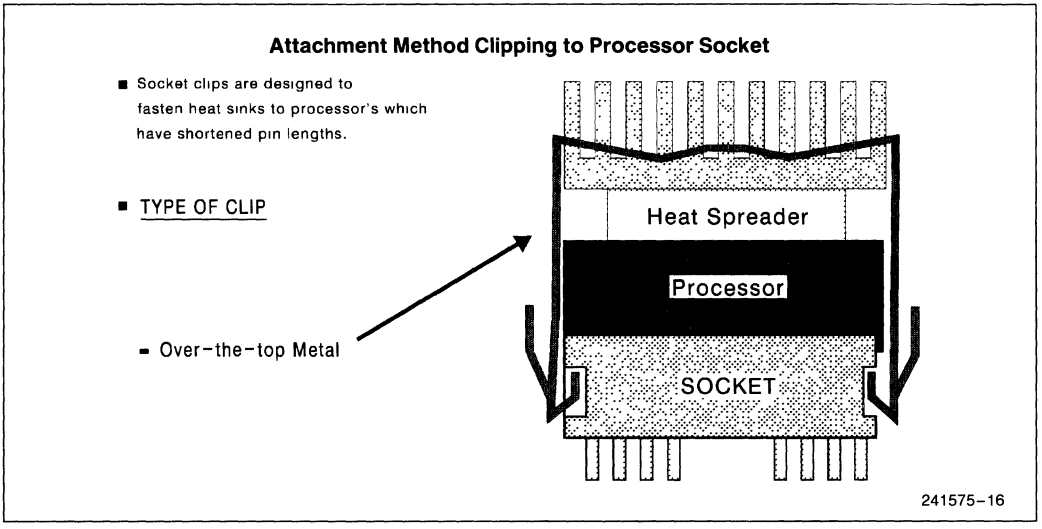


Figure A-5. Attachment Methods

Table A-2 lists pros and cons of the different attachment methods covered.

Table A-2. Attachment Methods

ATTACHMENT METHODS	ADVANTAGES	DISADVANTAGES
DOUBLE SIDED TAPES	• Quick to Use • Low Installed Cost • Compliant	• High Thermal Resistance • Requires Flat Interfaces • Assembly Contact Pressure
EPOXIES	• Low Potential Thermal Resistance • Low Contact Pressure	• Mixing, Curing, Messy • Timing Consuming (if not automated) • CTE Stress, High Rigidity • Variable Thickness (theta)
CLIPS	• Centralized Pressure Points • Removable • Easily Installed • Solution to Upgrade • Accommodates Wide Tolerance	• Removable • Force Limits vs Assembly • Insufficient Shock and Vibration Data • Potential for Loss of Pressure

The materials chart Figure A-6 shows the performance of each type of thermal interface material. Note that even though thermal grease has a deserved reputation for being messy and harder to control it still performs well as a thermal interface. All the examples that are shown in Appendix A use thermal grease as the case to heat sink interface.

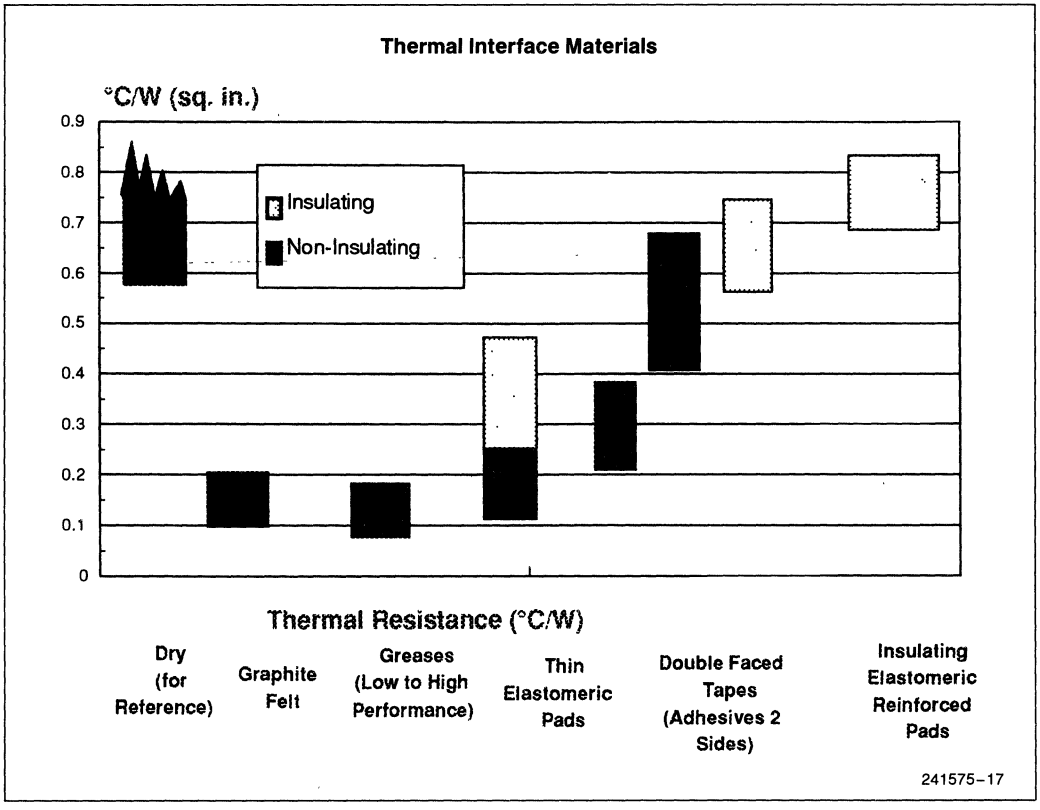


Figure A-6. Materials

The next step is to chose a heat sink. Figure A-7 shows the wide range of choices and the cost associated with each technology.

Now that all the variables and options are known for this problem we can proceed on to do some real system measurements using the recommendations and data shown in the first part of this application note.

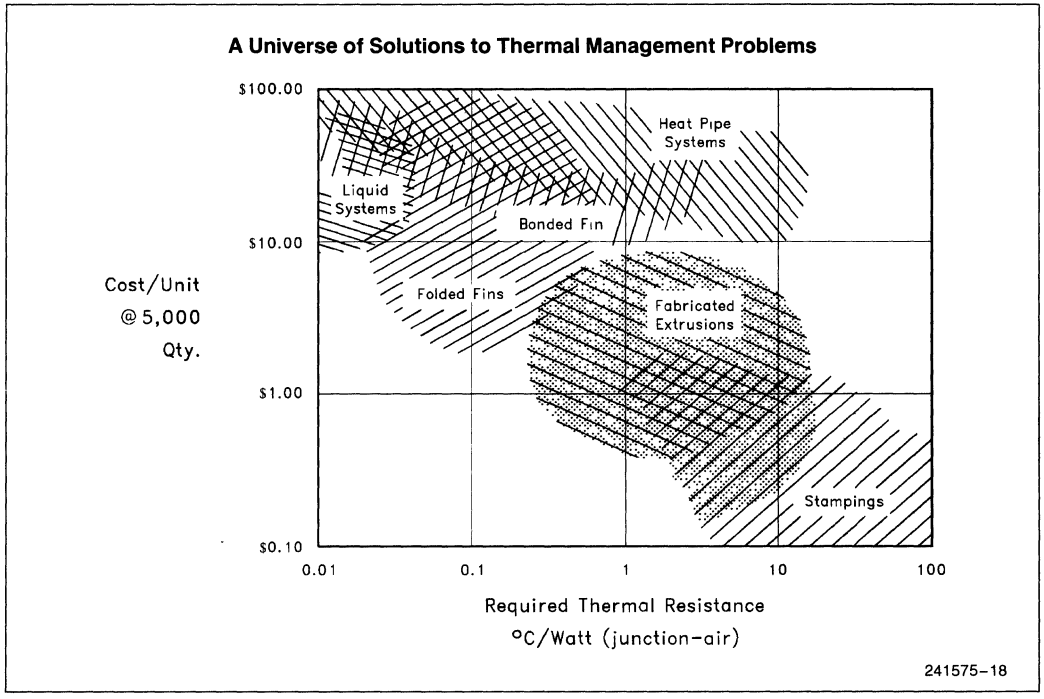


Figure A-7. Solutions

Examples

For all the examples in this section we used a 40 MHz system with a Pentium processor and 256K cache. A picture of the system under test is shown in Figure A-8 with the covers off to show the placement of the Pentium processor and the associated cache components. A 40 MHz system was used because it was the only one available at the time the testing was done.

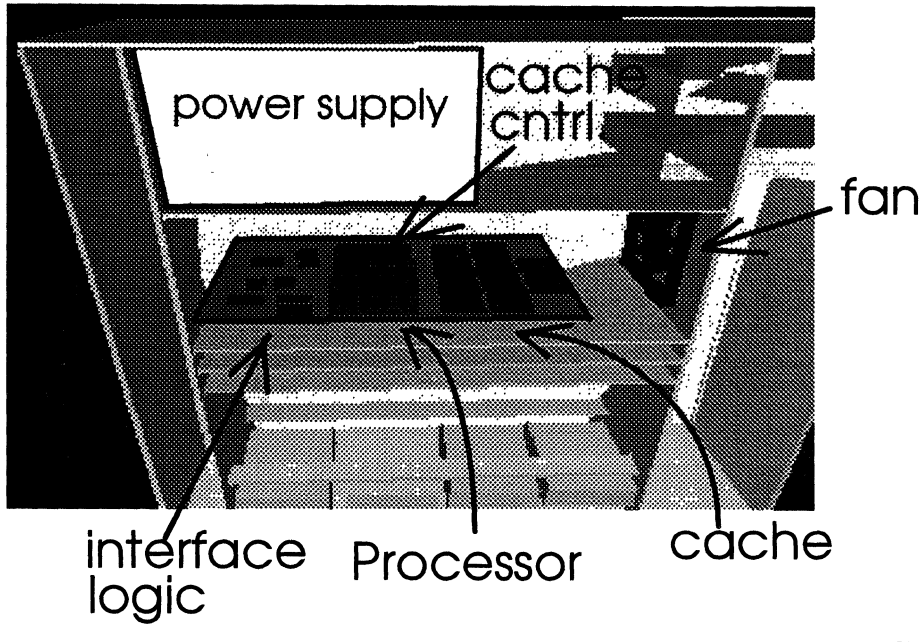


Figure A-8. Pentium® Processor System

Objectives

- To measure a Pentium processor system operating under real working conditions.
- To compare the measured results to the predicted results shown in the beginning of this application note. The reader should always keep the main goal in mind; **the main goal is always to meet the case temperature specification for the Pentium processor.** Any combination of heat sink and air flow rate is fine as long as the case temperature specification is met. The heat sinks used in test #1 thru #4 will match the suggested heat sinks as close as possible to accurately correlate with the wind tunnel data. This is meant to illustrate how a system designer might start by using the suggested heat sinks and air flow rates as starting points to thermally tune their particular system. Test #5 uses a heat sink and a fan combination. The fan heat sink is best described as a fan attached directly to the heat sink on the Pentium processor. It is an active device used for spot cooling ICs. We will concentrate on traditional passive heat sink solutions with only one set of measurements being done for a fan heat sink assembly.

Tools and Equipment

1. Pentium processor-based system running at 40 MHz.
2. Hot wire anemometer to measure airflow rate.
3. Thermocouples and high thermal conductivity cement as recommended in the application note.
4. Homemade jig for accurate and repeatable attachment of the thermocouples to the package.
5. Homemade power supply isolation socket for setting the V_{CC} and reading the I_{CC} of the processor independently of the rest of the system.
6. Adjustable power supply with adequate current capabilities and both current and voltage read out.
7. Multimeter to read the voltage and current.
8. Cables to connect everything up.
9. Software test suite that simulates "worst case conditions for a typical real application." In this case it was Microsoft Excel and Word for Windows test suites.
10. Drill and drill bits.
11. Thermal grease.

The lab procedure was as follows:

Preparing the System

1. Load the test software on the system disk (or floppy) and make sure everything runs correctly before you start. After everything works satisfactorily proceed to the next step.
2. Remove the covers, choose several places (random) around the processor to measure the air flow of the system. Then drill holes large enough to allow the anemometer to be inserted. Five holes were drilled in the system cover.
3. In this case we had a 12″ long ¼″ diameter directional anemometer. To get more repeatable measurements the shaft of the probe was marked with a pencil to get the same depth, into the box, for each measurement
4. We then removed the processor card from the chassis (use anti-static procedures to prevent IC damage).
5. Remove the Pentium processor from the card and install the isolation socket.

Preparing the Pentium Processor for Testing

1. Using the jig carefully attach the thermocouple to the center of the processor package using cement and let it cure as recommended by the manufacturer of the cement.
2. Drill holes no larger than 0.125″ in the centers of the heat sinks to be tested just large enough to get the thermocouple wires through the hole. In the case of the fan heat sink, the fan was removed and the heat sink was drilled the same as the others and then re-assembled. Each of the holes were counter sunk on the bottom to better conform to the tear drop shape the thermocouple and cement naturally forms into. The idea is to not disturb or break the contact between the cement and the package. If it is broken or cracked the measurements will be incorrect
3. Apply the thermal grease (less than 0.004″ thick) evenly, with no voids, to the processor package.
4. Slide the heat sink down the thermocouple wires being careful not to disturb the thermocouple while at the same time firmly seating the heat sink to the package. Attach the plug for the temperature meter to the other end of the thermocouple wire terminals.
5. Re-install the processor/thermocouple/heat sink assembly into the isolation socket on the processor board, again being careful not to disturb the thermocouple connection.

Preparing for Measurements

1. Re-install the processor card into the system.
2. Connect the power supply wires to the power supply and the isolation socket.
3. Connect the multimeter to the the power supply to monitor the V_{CC} and set the power supply meter to measure I_{CC}.
4. Connect the thermocouple to the meter.
5. Turn on the processor power supply and then the system supply.
6. Wait for the system to boot and then run the test software.

Thermal Measurements

The next step was to determine the baseline airflow in the system without a heat sink attached to the processor. Measure the airflow at several locations using the access holes in the system and the marks on the probe to ensure accurate placement of the probe and repeatability of the measurements. Table A-3 shows the results. Be cautious when placing the fan in a system relative to the processor. All fans have a dead spot (low airflow) in the center of the fan. Avoid the dead spot. Even several inches away from the fan the dead spot can influence airflow considerably.

Test #1

The next step is to compare how close the suggested values and tables are to the measured results. Use the formulas described in the beginning of the application note and the values from Table A-4.

$P_D = V_{CC} * I_{CC} = (1.82*4.89) = 8.827W$

$\theta_{JC} = (T_J - T_C)/P_D = 0.6$

$\theta_{JA} = (T_J - T_A)/P_D$

$\theta_{CA} = \theta_{JA} - \theta_{JC} = [(T_J - T_A) - (T_J - T_C)]/P_D$
$= [T_C - T_A]/P_D$

$\theta_{CA} = (55.3 - 29)/(1.8*4.85) = 24/8.827 = 2.97$

θ_{CA} = 2.7°C/W is the measured value in the system for this configuration.

Table A-3. Baseline Airflow

Airflow Measured, LFM	120–160
Location of probe	~ 2 inches (upstream from the fan) @ center of the processor (above the heat sink)

Table A-4. Test Conditions Test #1

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
1.2 x 2.1 sq.	23	29	55.3	1.82	4.85	100–150

The graph (Figure A-9) from the application note, for heat sink size size of 2.1″ x 2.1″, is used to compare the predicted θ_{CA}, for a 1.2″ tall heat sink, to the measured value of θ_{CA}.

The predictions from the graph (Figure A-9) are:

θ_{CA} = 2.9°C/W @ 100 LFM

θ_{CA} = 2.4°C/W @ 150 LFM

θ_{CA} = 1.9°C/W @ 200 LFM

And the measured value is:

θ_{CA} = 2.97°C/W @ 100–150 LFM

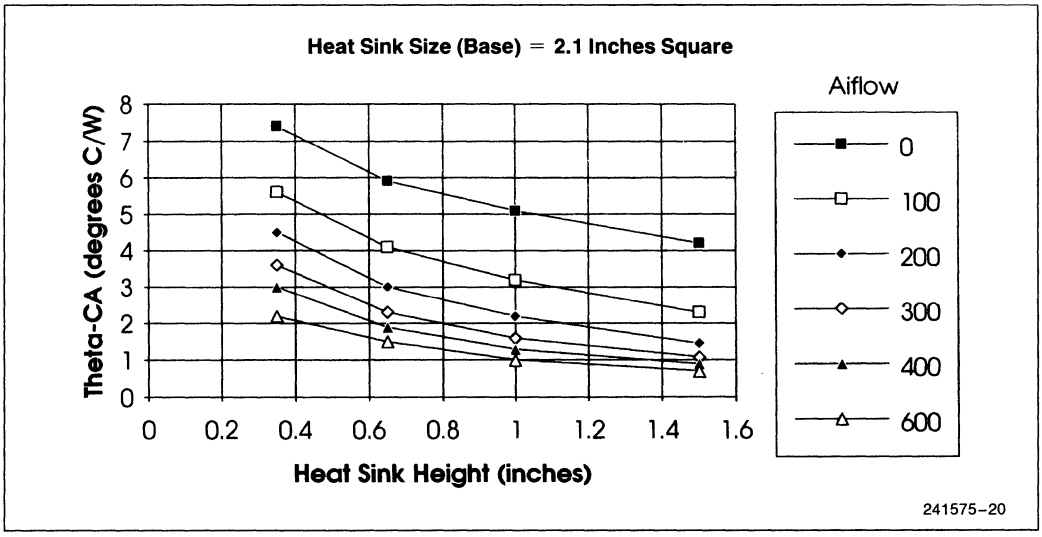


Figure A-9. Thermal Resistance

Table A-5. Test Conditions Test #2

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
0.5 x 2.1 sq.	22	29	58.3	1.81	4.85	100–150

Test #2

The same test only the heat sink height was reduced to 0.5 inch height.

$\theta_{CA} = (T_C - T_A)/P_D$

$\theta_{CA} = 58.3 - 29/8.79 = 3.33°C/W$

The 2. 1" x 2.1" graph (Figure A-9) from test #1 is used again and it predicts:

$\theta_{CA} = 4.9°C/W$ @ 100 LFM

$\theta_{CA} = 4.3°C/W$ @ 150 LFM

$\theta_{CA} = 3.8°C/W$ @ 200 LFM

And the measured value is:

$\theta_{CA} = 3.33°C/W$ @ 100–150 LFM

Test #3

This test is the same as test #2 except that processor board to board spacing was reduced to 0.6 inches using a cardboard baffle to simulate a system with very tight board spacing. An existing system that is upgrading from an Intel 486 processor to the Pentium processor might have this type of spacing. Note that this particular configuration actually has more airflow than test #2. It could have just as easily been lower. It all depends on the particular system being measured.

$\theta_{CA} = (T_C - T_A)/P_D$

$\theta_{CA} = 70.3 - 27/8.79 = 4.9°C/W$

The 2.1 " x 2.1" graph (Figure A-9) from test #1 is used again to predict the θ_{CA}:

$\theta_{CA} = 4.3°C/W$ @ 150 LFM

$\theta_{CA} = 3.8°C/W$ @ 200 LFM

$\theta_{CA} = 3.0°C/W$ @ 300 LFM

And the measured value is:

$\theta_{CA} = 4.9°C/W$ @ 175–200 LFM

Table A-6. Test Conditions Test #3

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
0.5 x 2.1 sq.	23	27	70.3	1.81	4.85	175–200

Table A-7. Test Conditions Test #4

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
0.65 x 3.1 sq.	23	29	55.3	1.8	4.85	100–140

Test #4

This test uses a 0.65″ tall heat sink that is 3.1″ sq. This type of heat sink might be used when height is limited and there is room to spread out by adding more area to the heat sink base.

$\theta_{CA} = (T_C - T_A)/P_D$

$\theta_{CA} = (55.3 - 29)/8.73 = 3.0$

The 3.0″ x 3.0″ graph (Figure A-10) from the application note is used since it is similar to the heat sink used. The 3.0″ x 3.0″ graph predicts:

θ_{CA} = 3.0°C/W @ 100 LFM

θ_{CA} = 2.6°C/W @ 150 LFM

And the measured value is:

θ_{CA} = 3.0°C/W @ 100–140 LFM

Test #5

The last test was done using a fan/heat sink assembly that has become popular for prototyping, debug and spot cooling in some situations. We were not able to measure the airflow on the processor with this configuration because the air flow is not directional enough to get a reading with the probe available. The case temperature however was monitored by mounting a thermocouple in the same manner used above. We did modify the setup by bringing the thermocouple wires out the side to clear the fan. This will change the measurements the thermocouple produces and should be factored into any data. We do not have any wind tunnel data on the fan/heat sink combination. Note that the case temperature is within specification.

2

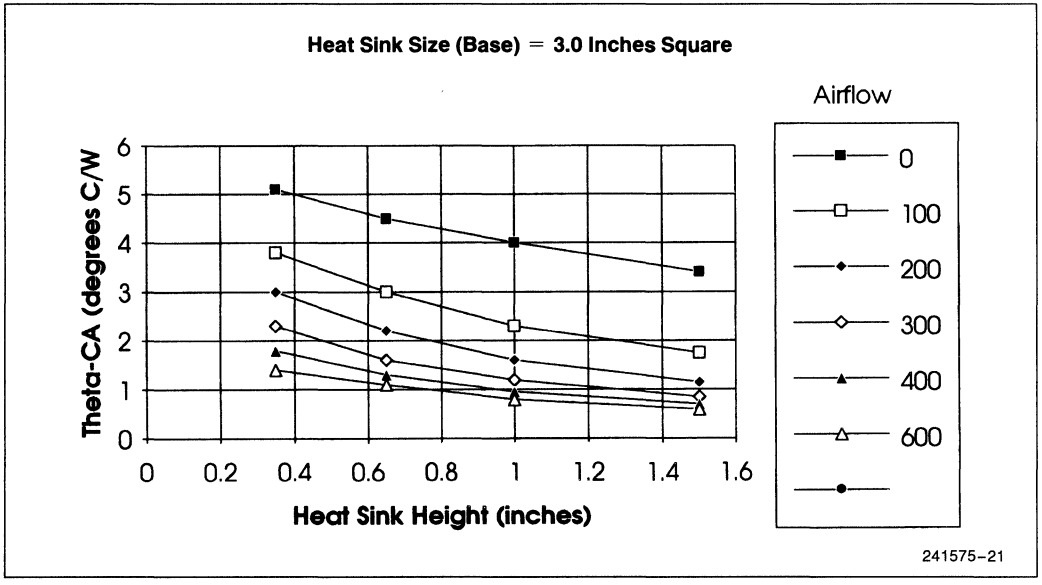


Figure A-10. Thermal Resistance

Table A-8. Test Conditions Test #5

Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	V_{CC} Volts	Air Flow LFM
	Room T_A	System T_A	Case T_C			
HS/Fan	23	29	46	1.8	4.85	120–160

Conclusion

Table A-9 shows all the tests in one table. The data shows that the suggestions in the application note are a very good starting point to begin tuning any Pentium processor system and that there is no one cookbook answer that fits all systems because of the complexity of air flow and variations from each type of system. Indeed the results show that airflow can be changed dramatically even in the same system by changing one variable. For example test #2 and #3 are exactly the same except that board to board spacing was reduced significantly. Note that case temperature rose significantly even though the airflow sensor was reading a higher value. This suggests that the airflow through the heat sink was lower even though the anemometer, 2 inches away, was reading higher airflow at its position. Note also that test #2 more closely approximates the wind tunnel test setup because it has open space above the board instead of a board nearby. This is also why the predicted data versus the measured data is so far off for test #3, while test #2 is very close to the predicted results.

Test #1 and #4 demonstrate a fundamental principle of the physics involved. If you have the same airflow and must reduce the height of the heat sink, you have to spread out the area of the heat sink to compensate for the reduced height. Test #1 uses a 1.2" height heat sink that is the same size as the package. Test #4 was able to produce the same case temperature with a shorter heat sink and more area.

Test #5 demonstrates that a fan/heat sink assembly can spot cool effectively if you have enough space above and around it to allow the required back pressure. This is the only active device tested. If you look back at the "A Universe of Solutions to Thermal Management Problems" (Figure A-7) chart you will see the reason why. While the Pentium processor is at the outer envelope of passive cooling, this method of cooling still offers lower cost, power usage and reliability in most cases.

Most of all the system designer should **never lose sight of the real goal which is to keep the junction temperature within the operating limit.** Since the designer cannot measure junction temperature they must use the case temperature, which can be measured to ensure proper operation for the component.

Table A-9

Test #	Heat Sink Size, Inches H x W	Temperature, degrees C			I_{CC} Amps	Air Flow LFM
		Room T_A	System T_A	Case T_C		
1	1.2 x 2.1 sq.	23	29	55.3	1.82	100–150
2	0.5 x 2.1 sq.	22	29	58.3	1.81	100–150
3	0.5 x 2.1 sq.	23	27	70.3	1.81	175–200
4	0.65 x 3.1 sq.	23	29	55.3	1.8	100–140
5	HS/Fan	23	29	46	1.8	120–160

V_{CC} = 4.85V for all tests.

APPENDIX B
HEAT SINK VENDORS

Aavid Engineering

One Kool Path
P.O. Box 400
Laconia, NH 03247
(603) 528-3400
(603) 525-1478 (FAX)

Contact: Gary F. Kuzmin (Product Marketing Manager)

EG&G Wakefield Engineering

60 Audubon Rd.
Wakefield, MA 01880
(617) 245-5900
(617) 246-0874 (FAX)

Contact: David Saums (Marketing Manager)

IERC

135 W. Magnolia Blvd.
Burbank, CA 91502
(818) 842-7277
(818) 848-8872 (FAX)

Contact: Guy R. Addis (Western Region Applications Engineer)

Thermalloy

2021 W. Valley View Lane
Dallas, TX 75234-8993
(214) 243-4321

Contact: Larry Tucker (VP of Sales and Marketing)

intel®

AP-481

APPLICATION NOTE

Designing with the Pentium® Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set

December 1995

Order Number: 241576-002

Designing with the Pentium® Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set

CONTENTS

CONTENTS

Designing with the Pentium® Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set

CONTENTS

CONTENTS

1.0 INTRODUCTION

The Pentium® processor, 82496 cache controller, and 82491 cache SRAM CPU-Cache Chip Set has been designed to take advantage of the high performance available from the 66-MHz operating frequency. In addition, it has been designed to obtain this performance without severely adding to the complexity of the system design. These benefits are accomplished by dividing the chip set into two interfaces. The first is the External Interface which is the interface between the CPU-Cache core and the rest of the system. It consists of the memory bus and the memory bus controller. This interface has been designed to operate at a fraction of the CPU's frequency or asynchronous to the CPU. These options simplify the system design by minimizing the portions that must deal with the high frequency signals. The second interface is the Optimized Interface which is between the Pentium processor and the 82496 cache controller and 82491 cache SRAM. This interface is tuned for the known configuration options of the chip set and includes specially designed input and output buffers optimized for the defined electrical environment of each signal path. The specification of the optimized interface is defined to accommodate the tuning involved.

The purpose of this application note is to provide additional explanation of the steps involved in completing a chip set design. It includes:

1. Describing the specifications and how they should be used.
2. Describing Intel's I/O buffer models.
3. Describing the characteristics of printed circuit boards and transmission lines.
4. Stepping through an example of using the information and tools to complete the layout of one signal and verify that it meets the published chip set specifications.

In addition, all of the information associated with a completed chip set optimized interface design example is included. This information allows system designers to minimize their effort in implementing the same or similar design.

2.0 A/C SPECIFICATIONS OF THE CHIP SET

The AC/DC specifications for the chip set are published in the latest revision of the *Pentium® Processor Family Developer's Manual, Volume 2*. It includes the specifications for both the optimized and external interfaces.

2.1 Optimized Interface

The optimized interface is the high-performance interconnect between the Pentium processor and the 82496 cache controller and 82491 cache SRAM. The input and output buffers have been tuned for the defined configuration and electrical environment (loading, etc.). This tuning is what allows the chip set's CPU-Cache core to operate synchronously at 66 MHz.

There are two types of specifications for signals in the optimized interface. The first is Flight Time which is used to guarantee that signal timings are met. The second is Signal Quality to guarantee reliable operation. I/O buffer models have been provided as a tool to ensure these specifications are met. In this section, flight time and signal quality will be discussed. I/O buffer models will be left for the next section.

2

2.1.1 FLIGHT TIME SPECIFICATION

2.1.1.1 Purpose of Flight Time Specification

The purpose of the flight time specification is to guarantee that a signal supplied by a driving component is available at the receiving component for sampling.

It replaces the output valid delay and input setup time. The two methods are analogous, except that flight time allows the input and output buffer behavior to be matched without major impact to designers or the documentation. In other words, if component A's output is slower than expected, but component B's input is faster than expected, these two can be traded off without having to change the flight time specification. However, if output valid delay and input setup time specifications had been used the specifications would have to be changed. Note that in both cases the time available to the system designer to move the signal from one component to the other is the same.

2.1.1.2 Definition of Flight Time

Flight Time is the propagation delay of a signal from a driving component to any receiving component. It is defined as the time difference between the $V_{CC}/2$ (50%) level of an unloaded output signal and the $V_{CC}/2$ (50%) level of a receiving signal whose 50% Vcc to 65% Vcc rise time is greater than or equal to 1V/ns. Figure 1 shows the flight time measurement between the 50% Vcc points on the unloaded driver and receiver waveforms.

If the rise time between the 50% Vcc and 65% Vcc points is less than 1V/ns, the determination of flight time is slightly more difficult and requires more calculation. In this case the 65% Vcc point is extrapolated back to the 50% Vcc point using a 1V/ns reference slope (i.e., subtract 0.75 ns when Vcc = 5V). Figure 2 shows the extrapolation from the 65% Vcc point and the resulting flight time measurement.

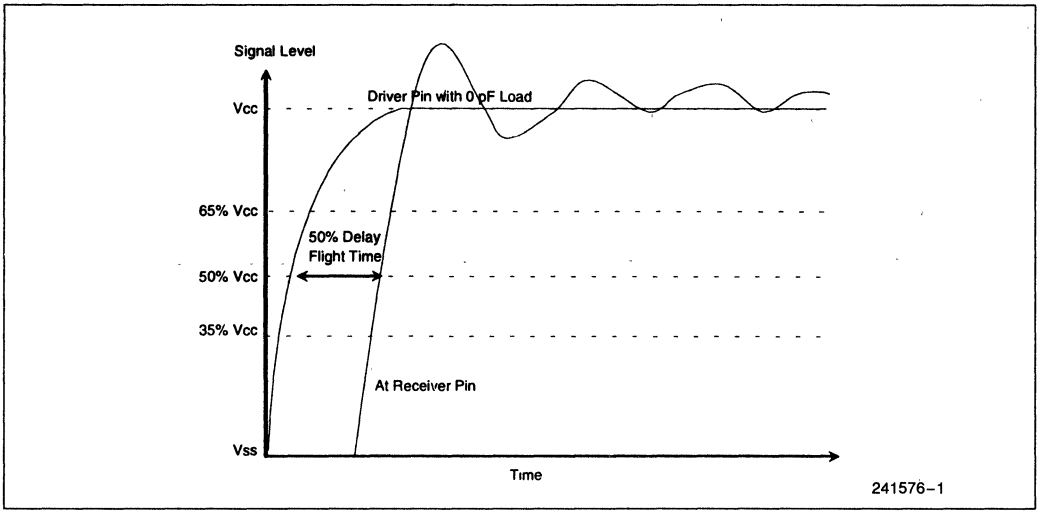


Figure 1. Flight Time Measurement

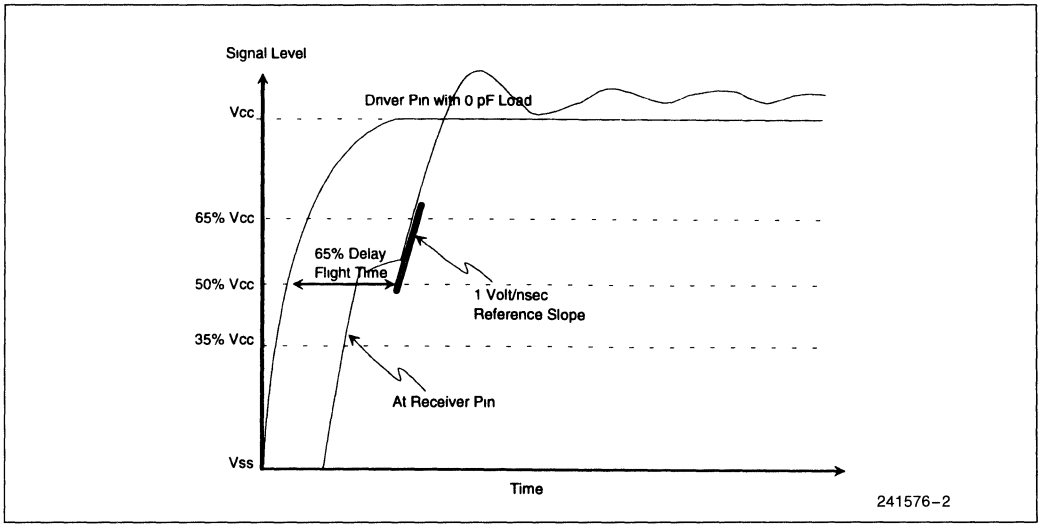


Figure 2. Flight Time Extrapolated from 65% Point

Thus flight time is the longer of T50-50 or Textrapolated. Although Figure 1 and 2 only show low-to-high transitions, flight time is the worst case of low-to-high and high-to-low transitions. Note on high-to-low transitions, 65% Vcc is replaced with 35% Vcc.

In a system environment it will not usually be possible to measure the delay of an unloaded driver. Figure 3 shows the method for measuring flight time in a system environment. As shown, the voltage measured at the pin of the loaded driver will have a ledge near the center of the transition. According to Transmission Line Theory, the time required to reach half the voltage level of the ledge is equivalent to the time required for an unloaded driver to reach the 50% Vcc level. The oscillation (if any) seen at the ledge defines the measurement uncertainty for this technique.

To measure flight time via this technique, first measure the maximum and minimum voltages of the ledge and take the average of these two values, (Vmax + Vmin)/2, to arrive at the ledge voltage. Finally, divide the ledge voltage by two, (Vmax + Vmin)/4. The result is the voltage level that approximately corresponds to the point in time at which an unloaded driver's signal would reach the 50% Vcc level. The flight time is determined by measuring the difference in time between the (Vmax + Vmin)/4 point and the extrapolated 50% point on the receiver, Textrapolated. The uncertainty of this technique is the time difference between the Vmin/2 and Vmax/2 points.

NOTE:

Figure 3 uses Textrapolated. This assumes the rise time between 50% Vcc and 65% Vcc is less than 1V/ns. If the rise time is equal to or greater than 1V/ns, the flight time should be measured to the 50% Vcc point, T0-50.

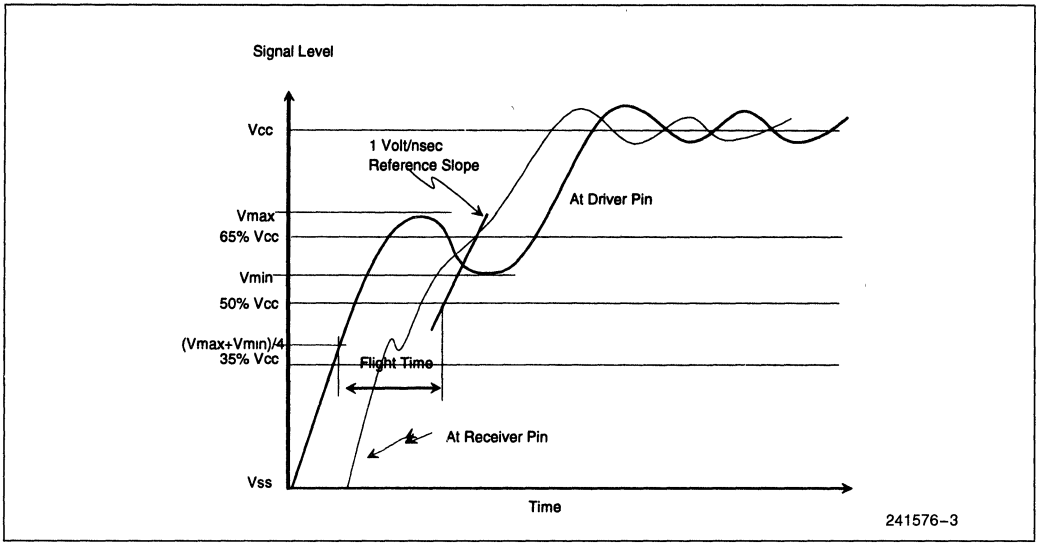


Figure 3. In-System Measurement of Flight Time

2.1.1.3 Clock Skew

Clock skew has generally been included in system design timing analysis. However, as frequencies increase, controlling clock skew becomes more important. Clock skew is the difference in time of the clock signal arriving at different components. It is measured at 0.8V, 1.5V, and 2.0V.

In synchronous devices, the clock signal defines the point in time in which signals are driven or sampled. It is important that all devices have a common reference. If the reference varies from component to component, the difference must be accounted for to ensure the devices function properly. In other words, clock skew must be subtracted from the clock period when performing a timing analysis of a system.

In the CPU-Cache Chip Set, the maximum clock skew between components in the optimized interface is a specification. This specification is required as a complement of flight time to ensure proper functionality. If clock skew exceeds the specified limit, the excess must be subtracted from the available flight time or the clock period must be increased.

2.1.2 SIGNAL QUALITY SPECIFICATIONS

Acceptable signal quality must be maintained over the entire operating range to insure reliable operation of the chip set. Signal quality consists of four parameters: Over/Undershoot, Time Beyond Supply, Ringback, and Settling Time. Figure 4 illustrates these signal quality parameters and how each is measured for a low-to-high transition. In addition to the absolute maximums associated with individual signals, each of the signal groups defined in the specification must meet maximum group average of Over/Undershoot and Time Beyond Supply. The following sections explain each of these in more detail.

Reliable operation means the signals are sampled correctly, do not exhibit false transitions, and that the long-term reliability of the component is not effected by overdriving the inputs.

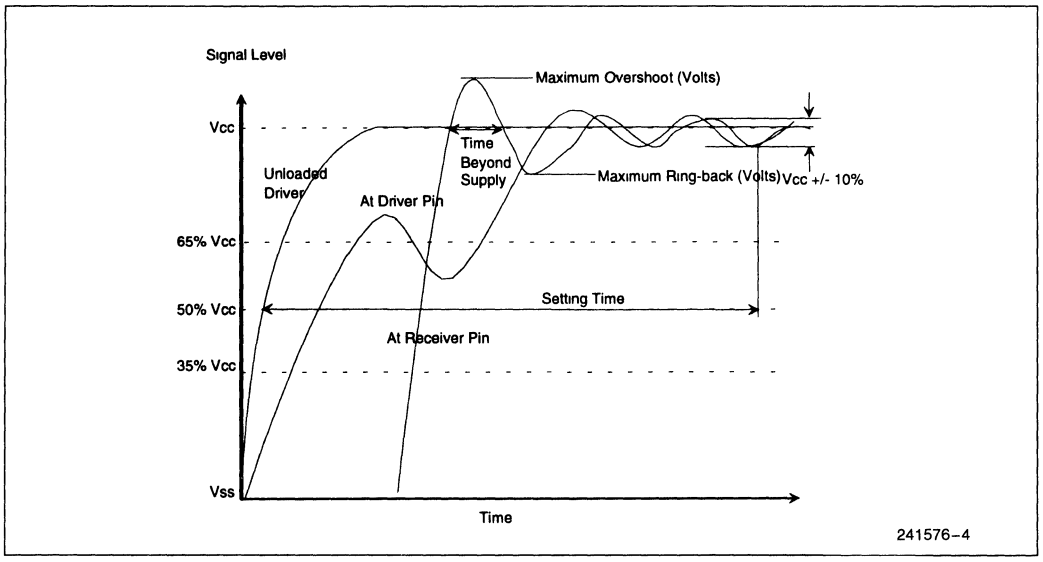


Figure 4. Signal Quality Parameter Measured for Low-to-High Transitions

2.1.2.1 Overshoot

Overshoot is the maximum absolute voltage a signal extends above Vcc or below Vss at the pin of the receiving component. Figure 4 shows the above Vcc case of overshoot. The overshoot specification is defined for use in simulation and assumes that input diodes are not included in the input model during simulation.

The overshoot specification maintains signal reliability by limiting the amount of energy that is injected into the component. Excessive energy being driven into the component can cause both short-and long-term reliability problems. They include Vcc or Vss plane shifts, electromigration, excessive ringback when the input diodes turn off, etc.

2.1.2.2 Time Beyond Supply

Time beyond supply is the maximum time a signal exceeds Vcc or Vss at the pin of the receiving component as shown in Figure 4. If the overshoot voltage is less than or equal to 0.5V, time beyond supply can be ignored. When time beyond supply is being ignored, a value of 0 ns should be used for that signal in calculating the group average time beyond supply.

Time beyond supply is a complement to overshoot when looking at signal quality. The time the signal is beyond the supply is a second factor in limiting the energy driven into the component. By limiting the time beyond supply, system designers are avoiding or minimizing the risk of the same reliability issues as described in the section on overshoot.

2.1.2.3 Ringback

Ringback is the maximum absolute voltage at the pin of the receiving component below Vcc or above Vss relative to Vcc or Vss after the signal has reached its maximum voltage level. Figure 4 illustrates how to measure ringback.

Eliminating ringback maintains signal quality by preventing a signal from re-crossing the threshold, causing a false transition to be detected.

2.1.2.4 Settling Time

Settling Time is the time required for a signal to settle within 10% of its final value referenced from the time unloaded driver's initial crossing of the 50% Vcc threshold. Figure 4 shows how settling time is measured on both low-to-high and high-to-low transitions.

The settling time specification is defined to ensure that a signal transition has completed and is no longer oscillating prior to the next transition. This is important to avoid forcing a signal to transition a distance significantly greater than Vcc/2. For example, if a signal is still not settled at the time of its next transition, it may be at a voltage above Vcc such as 6.5V. Assuming Vcc = 5V, the transition requires the voltage to swing from 6.5V (instead of 5V) to 2.5V. This added voltage distance translates into added flight time.

2.1.2.5 Group Averages

A Maximum Group Average specification is defined for Overshoot and Settling Time for each signal group. The group average is calculated by summing the maximum overshoot level or settling time for each signal in the group and dividing by the number of signals in the group.

The purpose of the group average specifications is to limit the amount of energy driven into the device. While each input can handle a certain amount of energy as limited by the Overshoot and Time Beyond Supply specifications, the overall part or portions of the part also have limits. These limits are maintained by ensuring the energy driven into these portions does not exceed a certain level.

2.2 External Interface

The External Interface is the interface between the CPU-Cache core and the rest of the system. It consists of the memory bus and the memory bus controller. This interface has been designed so that it can operate at a fraction of the CPU's frequency or asynchronous to the CPU. These options simplify the system design by minimizing the portions that must deal with the high frequency signals.

The specifications for the external interface are defined to allow system designers to connect the CPU and Cache components to other devices (ASICs, PLDs, memory, etc.). The external interface signal specifications are the more traditional output valid delay and float time and input setup and hold time. In addition, I/O buffer models have been provided as a tool to assist system designers.

2.2.1 OUTPUT VALID DELAY AND FLOAT TIME

Output Valid Delay is the amount of time it takes the signal to transition referenced from the clock edge. The maximum output valid delay is the earliest point in time that the signal can be assumed valid at the pin of the device. This timing is referenced from the clock edge and is measured at 1.5V as illustrated in Figure 5.

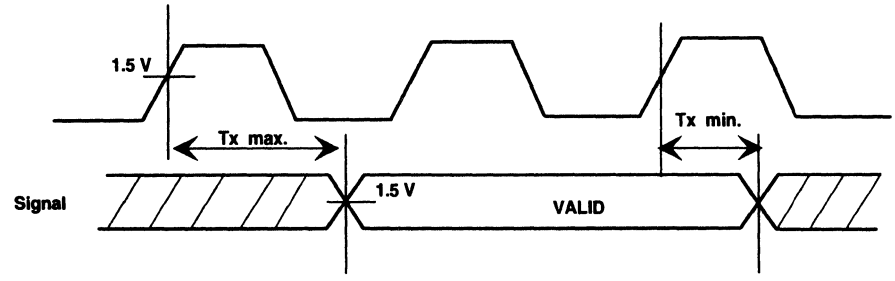
NOTE:
This specification is defined assuming C_L = 0 pF; therefore, system designers must account for all delay added by the signal traveling to the receiving device.

The maximum output valid delay is used by system designers to perform a worst case timing analysis to ensure that setup times are met at receiving devices.

The minimum output valid delay is the earliest valid data from the previous clock will begin transition after the clock edge. This timing is also referenced from the clock edge and is measured at 1.5V as illustrated in Figure 5.

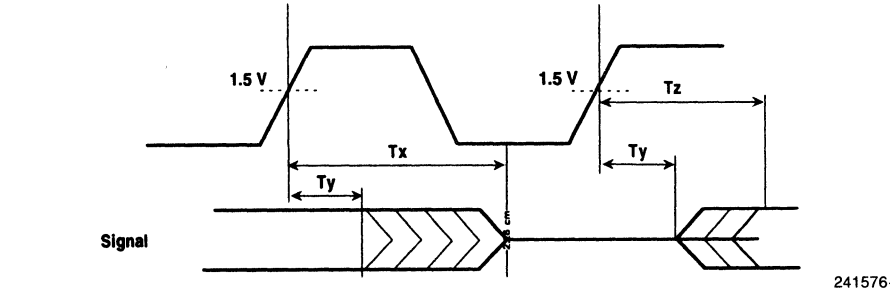
The minimum output valid delay is used by system designers to perform a worst case timing analysis to ensure that hold times are met at receiving devices. In addition, on I/O or tristate pins, it is used to ensure no bus contention exists due to multiple devices driving the bus simultaneously.

The maximum Output Float Time is the amount of time it takes to float a signal that was driven in the previous clock. This timing is also referenced from the clock edge and is illustrated in Figure 6. Note that the minimum valid delay determines how long data from the previous clock remains valid as shown in Figure 6.



Tx = valid delay

Figure 5. Output Valid Delay



Tx = max. float delay
Ty = min. valid delay
Tz = max. valid delay

Figure 6. Output Float Time

2.2.2 INPUT SETUP AND HOLD TIME

Input Setup Time is the amount of time a signal must be valid at the component's input pin prior to the clock edge it is sampled. The minimum input setup time is the latest point in time that the signal can be assumed valid at the pin of the device. This timing is referenced to the clock edge and is measured at 1.5V as illustrated in Figure 7.

The input setup time is used by system designers to perform a worst case timing analysis to verify that fast enough drivers have been chosen for ASICs or other devices and that the signals are able to travel across the board layout in the allotted amount of time.

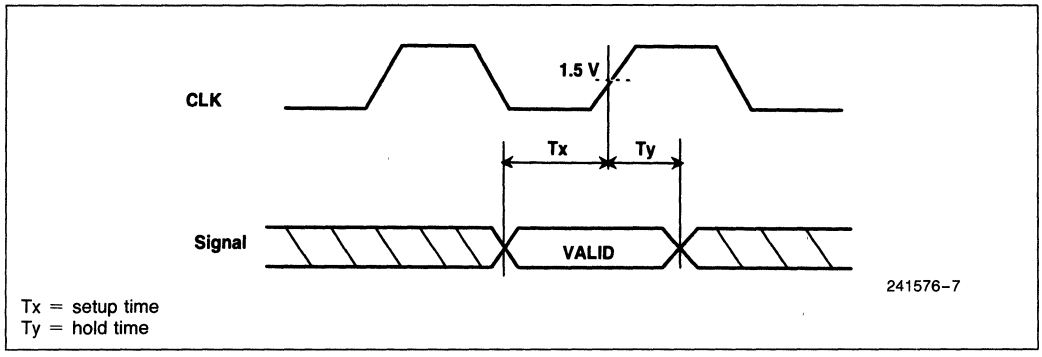


Figure 7. Input Setup and Hold Time

Input Hold Time is the amount of time a signal must be valid at the component's input pin after the clock edge is sampled. The minimum input hold time is the earliest point in time that the signal can start its next transition at the pin of the device. This timing is referenced to the clock edge and is measured at 1.5V as illustrated in Figure 7.

3.0 I/O BUFFER MODELS

3.1 Description of the First Order I/O Buffer Model

The first order I/O buffer model is a simplified representation of the complex input and output buffers used in the Pentium processor, 82496 cache controller, and 82491 cache SRAM. Figure 8 shows the structure of the input buffer model and Figure 9 shows the output buffer model. Tables 1 and 2 show the parameters used to specify these models.

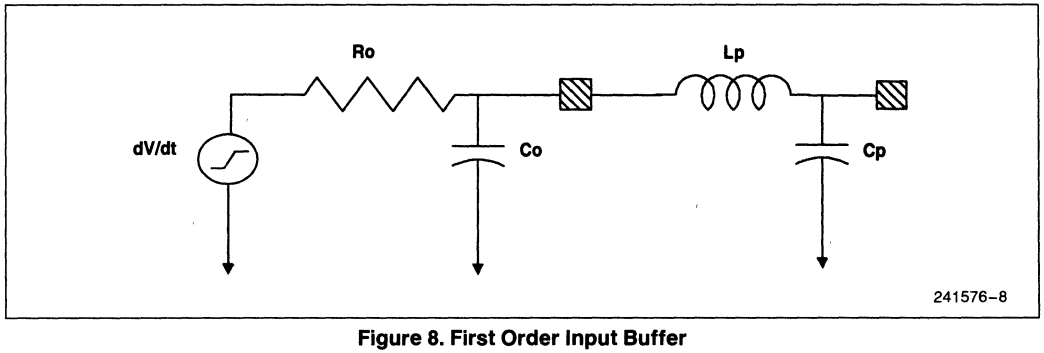


Figure 8. First Order Input Buffer

Table 1. The Parameters Used in the Specification of the First Order Input Buffer Model

Parameter	Description
Cin	Minimum and maximum value of the capacitance of the input buffer model
Lp	Minimum and maximum value of the package inductance
Cp	Minimum and maximum value of the package capacitance

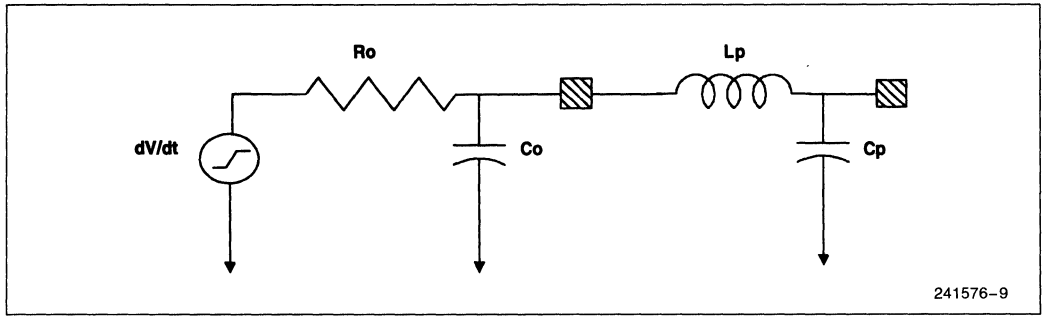


Figure 9. First Order Output Buffer

Table 2. The Parameters Used in the Specification of the First Order Output Buffer Model

Parameter	Description
dV/dt	Minimum and maximum value of the rate of change of the open circuit voltage source used in the output buffer model
Ro	Minimum and maximum value of the output impedance of the output buffer model
Co	Minimum and maximum value of the capacitance of the output buffer model
Lp	Minimum and maximum value of the package inductance
Cp	Minimum and maximum value of the package capacitance

The first order I/O buffer parameters for the chip set are published in the latest revision of the *Pentium® Processor Family User's Manual, Volume 2*. It includes the minimum and maximum values for each parameter allowing simulations for both the fast and slow condition to be performed.

The key to the first order model is that it is a simplistic representation of the input and output buffers. The parameters are easy to use in a variety of simulators and provide the needed accuracy to complete a chip set design. In addition, the simplicity greatly reduces the compute time required to perform simulations as compared to full transistor and process models.

4.0 HIGH FREQUENCY DESIGN CONSIDERATIONS

Any board interconnection is a transmission line by definition. However, as a general rule, the effects of modeling an interconnect/trace as a transmission line are negligible at low frequencies. This is because the reflections get masked since the propagation is short with respect to the signal rise time. In this case, system interconnects can be modelled as lumped loads with no sacrifice to accuracy. As the frequency at which signals change increases, the rise time becomes shorter and transmission line properties become significant and must be considered. Reflections, interference, and noise cause measureable changes in the appearance or behavior of signals at the higher frequencies. They can slow the transition time or cause signal quality violations. Figures 10 and 11 illustrate the effect of modeling traces as lumped loads or as transmission lines. Figure 10 shows a block diagram of a lumped load model and the resulting simulation and Figure 11 shows a block diagram of a transmission line model and the resulting waveform. The transmission line case shows the effects of reflections and how the signal varies at each component. These details are not shown in the lumped load case.

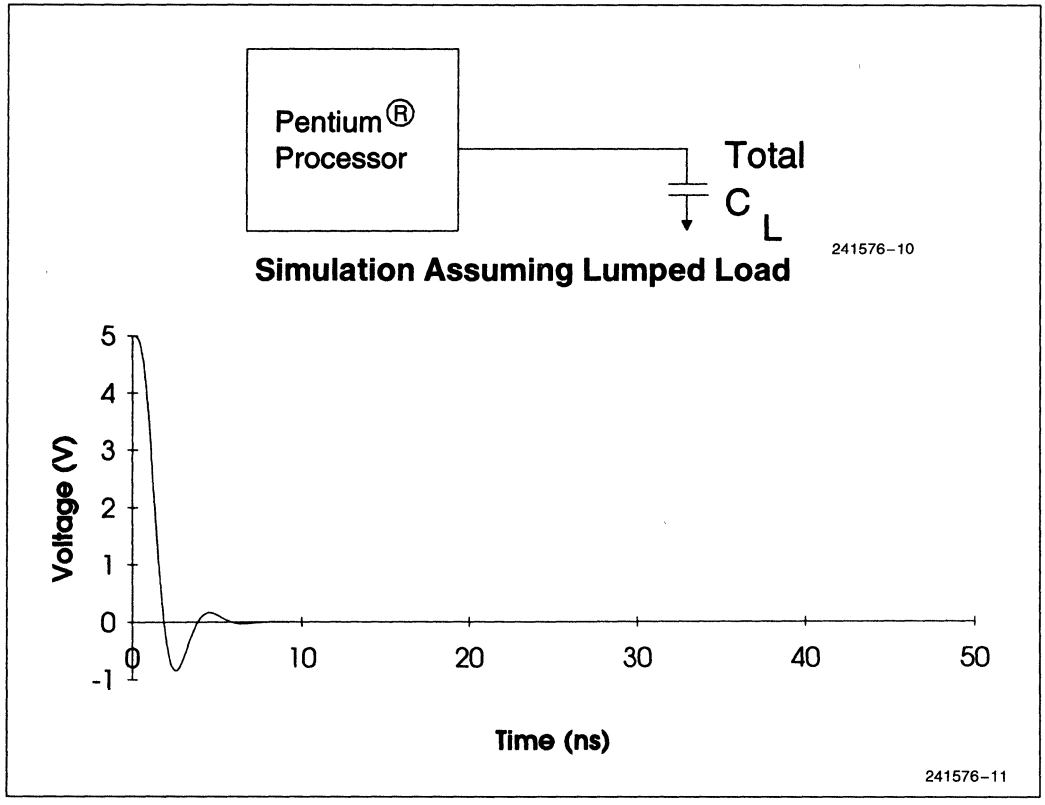


Figure 10. Lumped Load Simulations

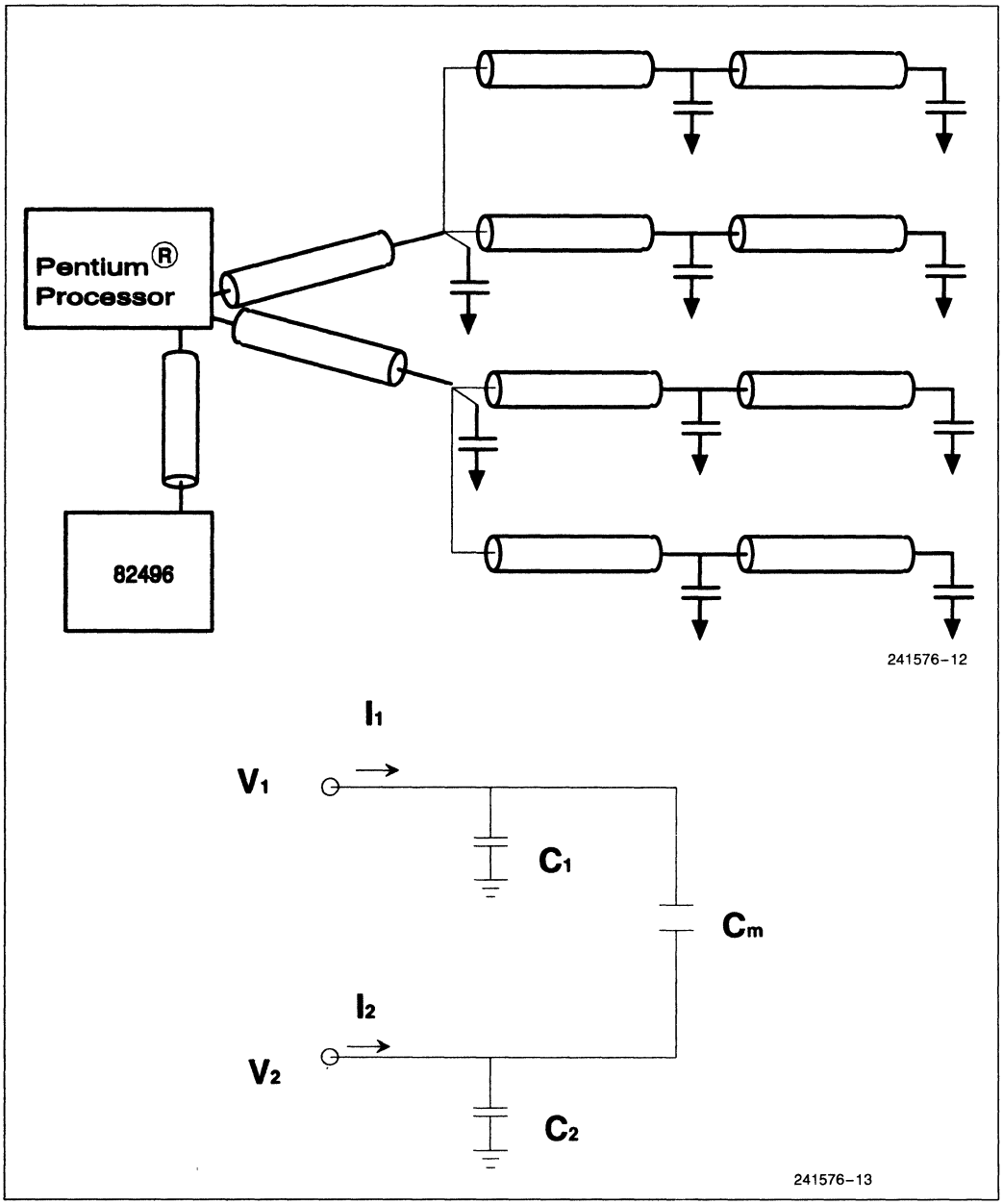


Figure 11. Transmission Line Simulation

To predict a trace's behavior and eliminate the negative effects, it is important to properly model board interconnects as transmission lines. A model consists of the driving component's output buffer model, the characteristics of the trace, and the receiving component's input buffer model. The I/O buffers are modelled using

the parameters published in the latest revision of the *Pentium® Processor Family Developer's Manual, Volume 2*. The trace characteristics are a function of the actual board and the material used in its construction. Characteristic impedance (Zo) and propagation delay (Tpd) are the primary characteristics needed. These two parameters, along with length, can be used in many simulators to represent the transmission line as shown in Figure 12(a).

Some simulators may not have a single representation for a transmission line. These simulators require the user to model the transmission line in a more basic form. In addition to characteristic impedance and propagation delay, transmission lines can be characterized by their characteristic inductance (Lo) and characteristic capacitance (Co). The following equations can be used to convert between these four parameters:

$$Lo = Zo * Tpd$$

$$Co = Tpd/Zo.$$

Figure 12(b) illustrates the model of a transmission line using Co and Lo. The number of L-C links/inch in the chain should be chosen such that $(LC)^{1/2} \ll Tr$. As a good rule of thumb, two to four links per inch is a good starting point. The value of L and C is determined by dividing the characteristic impedance by the number of links/inch. A transmission line also has a resistive component, Ro. However, the Ro value is usually assumed negligible and omitted from the model. Its only effect is to cause a voltage loss between the driver and the receiver. Since Ro's value is negligible, the voltage loss is very small and can also be ignored.

$$L = Lo/n$$

$$C = Co/n$$

$$n = \text{number of links/inch}$$

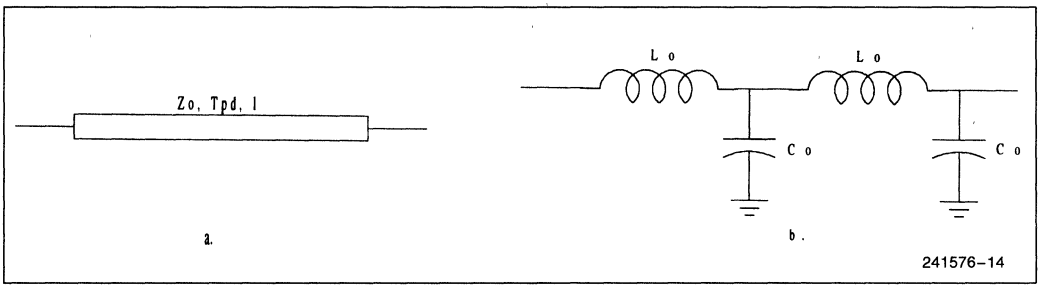


Figure 12. Representation of a Transmission Line

In order to complete the model, the actual parameter values must be determined. That means the next step is to determine the values of Zo and Tpd. To do this the designer must understand the construction of the transmission line. In particular that means understanding how the printed circuit board is constructed and the geometry of the various layers and traces.

4.1 Printed Circuit Board

A printed circuit board consists of some number of signal layers and power and ground planes separated by a dielectric material. An example is illustrated in Figure 13.

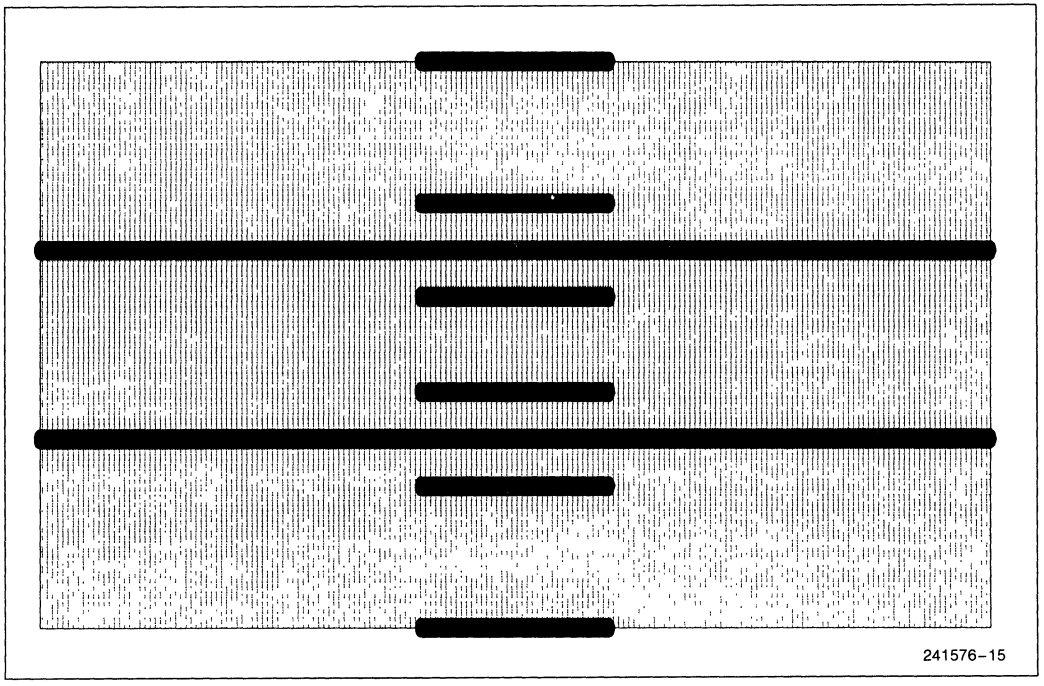


Figure 13. Printed Circuit Board

Although there are many types of transmission lines, those most commonly used on printed circuit boards are microstrip lines and striplines. Microstrip lines consist of a signal trace that is separated from a power or ground plane by a dielectric as shown in Figure 14.

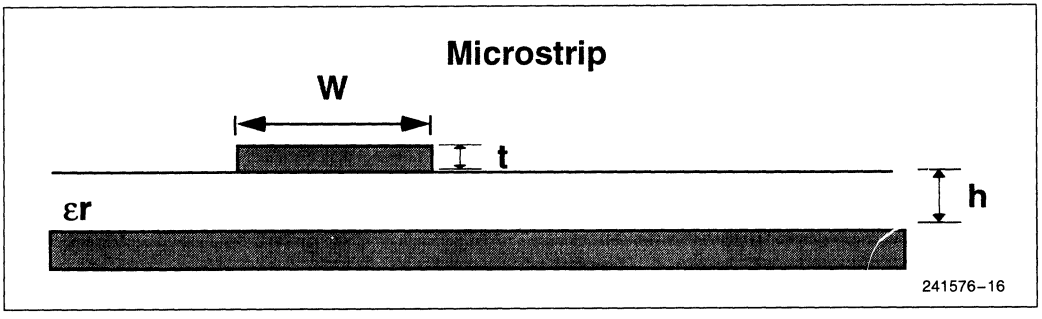


Figure 14. Microstrip Trace

The characteristic impedance is a function of the dielectric constant and the board geometry. For a microstrip trace this is given by:

$$Zo = \frac{87}{(\epsilon_r + 1.41)^{1/2}} \ln\left(\frac{5.98h}{0.8W + t}\right) \text{ Ohms}$$

where e_ϵ is the relative dielectric constant of the board material. The propagation delay is a function of the dielectric constant only. For a microstrip trace this is given by:

$$tpd = 1.017\ (0.475\ \epsilon_r + 0.67)^{1/2}\ ns / ft$$

Striplines consist of a signal trace that is located in a dielectric material between two power or ground planes as shown in Figure 15.

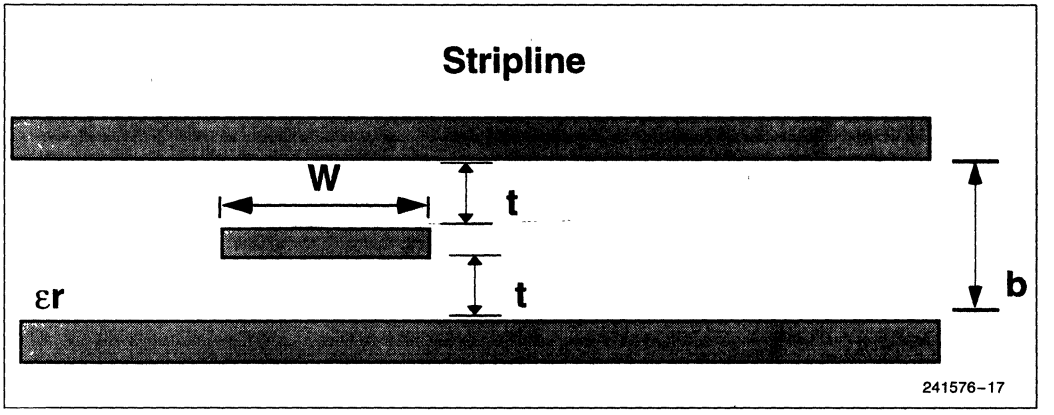


Figure 15. Stripline Trace

The characteristic impedance is a function of the dielectric constant and the board geometry. For a stripline trace this is given by:

$$Zo = \frac{60}{(\epsilon_r)^{1/2}} \ln\left(\frac{4b}{0.67\,\pi(t + 0.8W)}\right) \text{ Ohms}$$

where e_r is the relative dielectric constant of the board material. The propagation delay is a function of the dielectric constant only. For a stripline trace this is given by:

$$tpd = 1.017\ (\epsilon_r)^{1/2}\ ns/ft$$

Signal traces that are located on the external layers of the board can be modelled using the microstrip characteristics. Signal traces that are located on internal layers between the power and ground planes can be modelled by using the stripline characteristics.

4.2 Transmission Line Behavior

Now that the parameters needed to represent a transmission line are complete, the next step is to look at how the transmission line behaves or effects signals that are transmitted on it. The four primary effects are signal propagation and reflection, crosstalk, and skew.

4.2.1 SIGNAL PROPAGATION AND REFLECTION

Ideally, a signal that is driven by one device to another across a trace will reach the receiving device instantly and without any distortion. In reality, this is not the case. Because of the resistive, capacitive, and inductive components of a transmission line and the attached loads, the voltage and current of a signal may change as the signal travels along the trace and may vary over time.

For simplicity, the examples that follow will assume lossless transmission lines. In other words, the transmission line itself does not cause an attenuation of the voltage as it travels along the trace. The voltage loss seen at the receiving end is a function of the resistive component of the transmission line which is negligible.

An example of a signal propagating along a transmission line is shown in Figure 16. The voltage source at the left launches a wave onto the transmission line. The voltage of the wave is equal to the voltage division of the voltage source resistance and the line inductance.

$$Vi = Vin * Zo/(Zo + Rs)$$

This waveform is propagated down the transmission line without any voltage loss or change in the waveform. The only effect of the transmission line is to delay the signal from reaching the receiving end. When the wave reaches the receiving end, it is reflected back towards the source. The reflection is proportional to the initial wave by an amount called the Reflective Coefficient. A reflective coefficient exists for both the source and the load. The values are a function of the source or load resistance and the lines characteristic impedance.

$$\rho_L = (R_L - Zo)/(R_L + Zo)$$
$$\rho_S = (R_S - Zo)/(R_S + Zo)$$

The value of the initial reflection at the load is:

$$Vr = Vi * \rho_L$$

The reflections can be caused by any discontinuity on the line. The discontinuity can be caused by a mismatch in impedance between the source or load and the characteristic impedance of the trace, branches in the trace, vias, or bends and angles in the trace. Here the discontinuity between the source and load are used as an example because they are probably the most prominent. Each reflection can attenuate or reinforce the wave depending on the phase of the reflection. The reflections continue indefinitely; however, with each reflection the magnitude of the voltage decreases and the line approaches a steady state value. A rule of thumb is that by the third or fourth reflection the value is negligible.

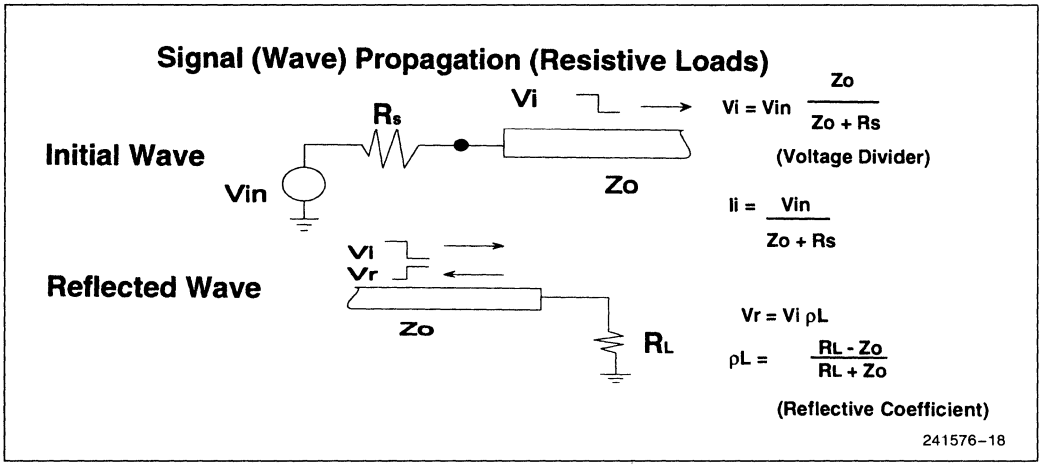


Figure 16. Signal Propagation Along a Transmission Line with Resistive Loads

Now that the voltage of each reflected waveform can be calculated, the next step is to sum these values to determine the voltage measured on the line at any point in time. The superposition principle comes into play. It states that the voltage/current at any point on a transmission line equals the sum of the voltages/currents of all the signals (waves) that have passed that point. In other words, as each reflection passes the point of measurement, it is added to the previous voltage seen at that point. Figure 17 illustrates the voltage seen at the midpoint of the circuit shown in Figure 16 over time.

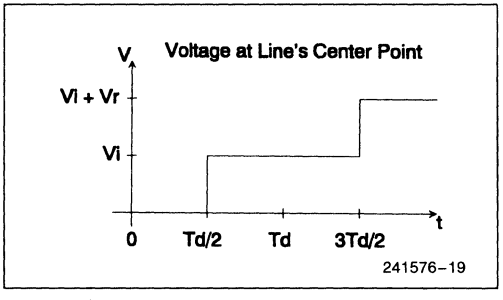


Figure 17. Voltage at the Midpoint of Circuit in Figure 16

From time 0 until Td/2 the voltage is 0V at the midpoint because the initial wave has not yet reached the midpoint. At time Td/2, the initial wave reaches the midpoint and the voltage is Vi. This wave travels down the trace until it reaches the load and a reflection occurs. The reflection begins traveling back towards the source, but does not reach the midpoint until time 3Td/2. At that time the voltage increases by Vr, the reflections voltage as shown in Figure 17. The following equation determines the voltage at any given point on a trace at the given time.

$$
\begin{aligned}
V(x,t) = [Zo / (Zo + Zs)] * \{ & Vin\,[t - (t - tpd * x)] * U(t - tpd * x) \\
& + \rho_L * Vin\,[t - (t - tpd * (2L - x))] * U(t - tpd * (2L - x)) \\
& + \rho_L * \rho_S * Vin\,[t - (t - tpd * (2L + x))] * U(t - tpd * (2L + x)) \\
& + \rho_L 2 * \rho_S * Vin\,[t - (t - tpd * (4L - x))] * U(t - tpd * (4L x)) \\
& + \rho_L 2 * \rho_S 2 * Vin\,[t - (t - tpd * (4L + x))] * U(t - tpd * (4L + x)) \\
& + \ldots\}
\end{aligned}
$$

U(x) = unit step function
Tpd = propagation delay of signal traveling along the transmission line (ns/ft)

As reflections occur on the line they can cause slower signal transitions, overshoot, undershoot, ringing, and other undesirable effects. Although many of the effects of reflections are negative, sometimes designers take advantage of constructive reflections to decrease the time it takes for the voltage to reach its final value at the destination. In general, designers try to minimize the magnitude of reflections.

System designers can do several things to reduce or minimize reflections: reduce angles (specifically 90° angles in traces, minimize the number of vias, and use termination when necessary).

Figure 18 illustrates how angles can be reduced by using 135° bends instead of 90° bends. The 135° bend approximates a smooth curve more closely than the 90° bend. The discontinuity occurs in the 90° bend because the trace is wider through the bend and therefore the impedance is altered by the geometry of the trace.

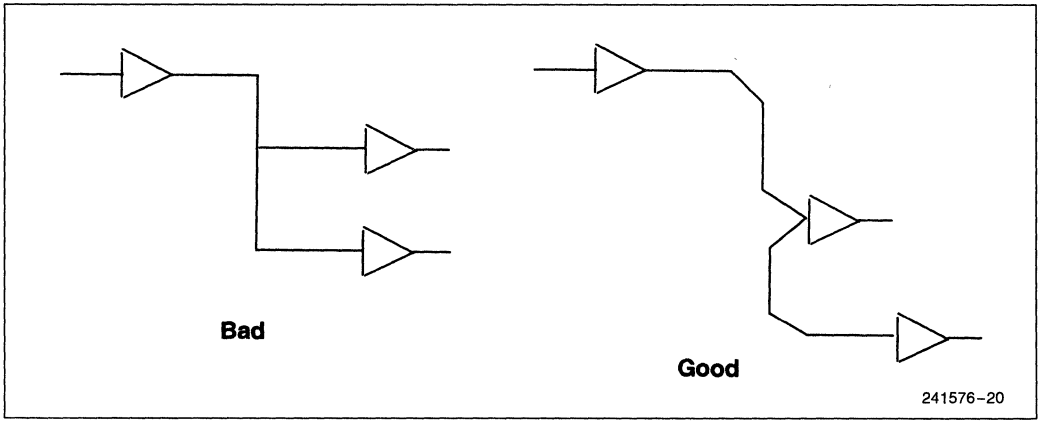


Figure 18. Eliminate 90° Angles

Figure 19 illustrates a way to reduce the number of discontinuities by minimizing the number of vias. Once again, a via causes a change in the path of a signal much like the 90° angles do. In addition, the geometry of a via is generally wider or thicker than the rest of the trace resulting in a different impedance for that portion of the interconnect. The change of impedance in the path causes the discontinuity and the resulting reflections.

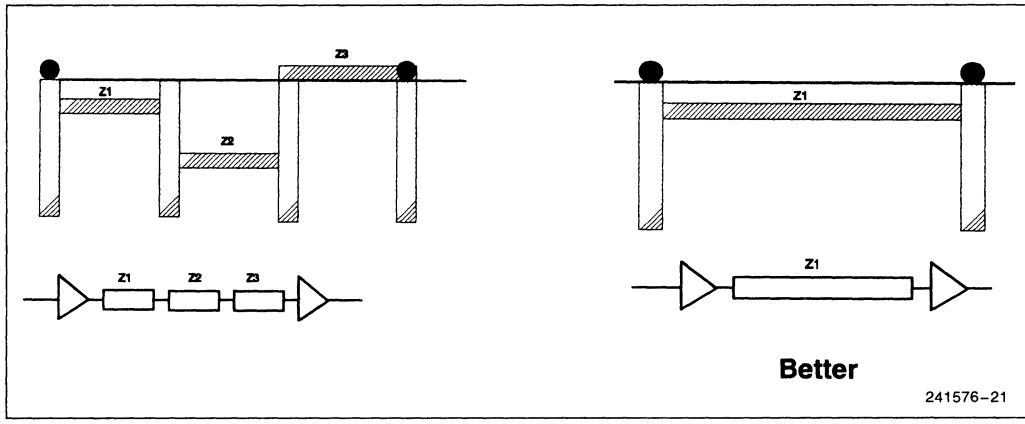


Figure 19. Minimize Vias

It is not always possible to eliminate all the discontinuities or mismatches in impedance. When this is the case, it is sometimes necessary to use a technique called *Termination* to artificially make the mismatched impedances appear matched. This technique is normally used to match a traces characteristic impedance with either the load or sources impedance.

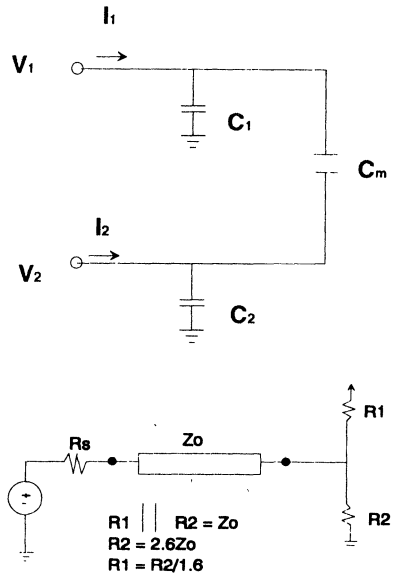
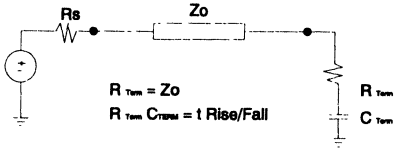
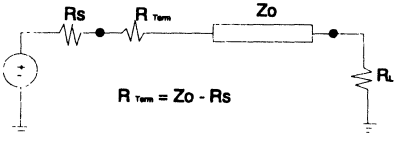
$$Z_o = Z_{term} + Z_L$$

and

$$Z_o = Z_{term} + Z_S$$

Several techniques of termination exist. They are Parallel, AC or RC, and Series termination. Each technique has its own advantages and disadvantages as summarized in Table 3.

Table 3. Termination Techniques

Type of Termination	Figure	Advantages	Disadvantages
Parallel		• eliminates reflections at receiver • good overshoot suppression • no added delay	• high power dissipation • requires Zo > 100Ω to avoid exceeding dc current limit • reduced voltage swing
AC or RC		• low power consumption • full voltage swing • eliminates initial reflection at receiver	• Cterm adds capacitive Load to driver • added delay due to RC time constant • component size and count
Series		• no additional loading on driver • no additional charging time • low power consumption • eliminates secondary reflection at source	• added delay

4.2.2 CROSSTALK

Crosstalk is another side effect of transmission line interconnnects. Crosstalk is the result of fields from adjacent traces interacting with each other. The interaction can alter the characteristics of a driven line or cause noise to be coupled into passive lines.

Crosstalk can be characterized by two parameters: Mutual Inductance, L_m, and Mutual Capacitance, C_m, as shown in Figure 20 and 21, respectively. These two parameters represent the inductive and capacitive values that exist between two adjacent lines. The inductance allows a current in one line to induce a voltage in a second line.

$$V_{m2} = L_m * \Delta I1/\Delta t$$

The capacitance allows a voltage on one line to induce a current in the second.

$$I_{M2} = C_m * \Delta(V1V2)/\Delta t$$

These mutual components have an additive effect to the L and C used to characterize each transmission line. To see what effect this has, examine the two components separately. First, Figure 20 illustrates two parallel traces with their inductive components and a mutual inductance between the two.

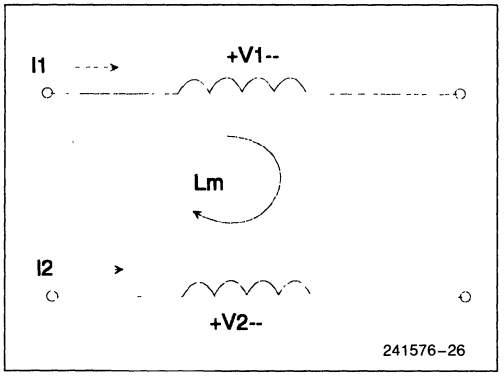


Figure 20. Inductive Components of Two Parallel Traces

The voltage seen on each line is given by:

$$V_1 = V_{L1} + V_m = (L_1 * \Delta I_1/\Delta t) + (L_m * \Delta I_2/\Delta t)$$
$$V_2 = V_{L2} + V_m = (L_2 * \Delta I_2/\Delta t) + (L_m * \Delta I_1/\Delta t)$$

To see what effect this has assume $L_1 = L_2$ and the magnitude of $\Delta I_1/\Delta t$ = the magnitude of $\Delta I_2/\Delta t$. This allows the above equations to be simplified to:

$$V_1 = V_2 = (L_1 + L_m) * \Delta I_1/\Delta t$$

(Current in same direction)

and

$$V_1 = -V_2 = (L_1 - L_m) * \Delta I_1/\Delta t$$

(Current in opposite direction).

From these equations the effective inductance seen on either trace is:

$$L_{eff} = L_1 + L_m$$

(Current in same direction)

and

$$L_{eff} = L_1 - L_m$$

(Current in opposite direction).

Therefore, if the currents are flowing in the same direction the effective inductance of each trace is increased. If the currents are in opposite directions the effective inductance of each trace is decreased.

Secondly, Figure 21 illustrates two parallel traces with their capacitive components and a mutual capacitance between the two.

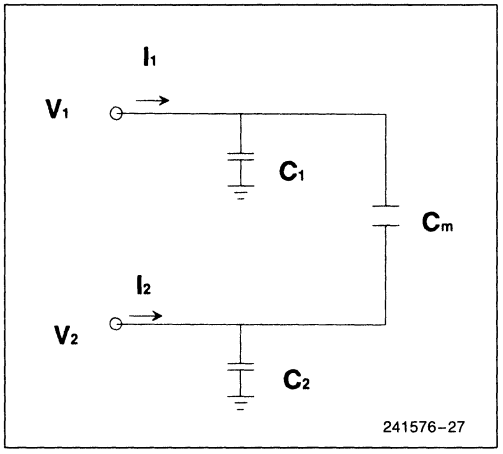


Figure 21. Capacitive Components of Two Parallel Traces

The current seen in each line is given by:

$$I_1 = I_{C1} + I_M$$
$$= (C_1 * \Delta V_1/\Delta t) + (C_m * \Delta(V_1 - V_2)/\Delta t)$$
$$= ((C_1 + C_m) * \Delta V_1/\Delta t) - (C_m * \Delta V_2/\Delta t)$$

$$I_2 = I_{C2} + I_M$$
$$= (C_2 * \Delta V_2/\Delta t) + (C_m * \Delta(V_2 - V_1)/\Delta t)$$
$$= ((C_2 + C_m) * \Delta V_2/\Delta t) - (C_m * \Delta V_1/\Delta t)$$

Using the same assumptions that $C_1 = C_2$ and that the magnitude of $\Delta V_1/\Delta t$ = the magnitude of $\Delta V_2/\Delta t$ allows the equations to be simplified to:

$I_1 = I_2 = C_1 * \Delta V_1/\Delta t$ (Voltage change in the same direction on both traces)

and

$I_1 = -I_2 = (C_1 + 2C_m) * \Delta V_1/\Delta t$ (Voltage change in the opposite direction on each trace).

2

From these equations the effective capacitance seen on either trace is:

$C_{eff} = C_1$ (Voltage change in same direction)

and

$C_{eff} = C_1 + 2C_m$ (Voltage change in opposite directions).

Therefore, if the voltages are changing in the same direction the effective capacitance of each trace is unchanged or decreases. If the voltages are changing in opposite directions the effective capacitance of each trace is increased. See Figure 22.

If L_{eff} and C_{eff} are used to determine Zo and Tpd, the following results:

$$Zo = (Leff/Ceff)^{1/2} \quad Tpd = (Leff * Ceff)^{1/2}$$

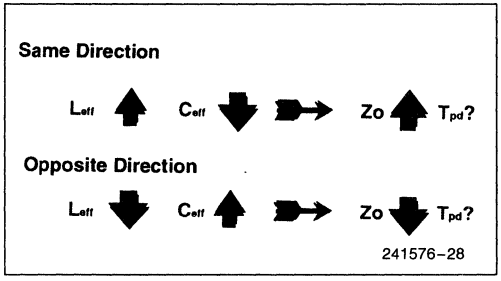


Figure 22. Effect of Changing Voltages in the Same or Opposite Directions

Electrons travel at the speed of light, so Tpd can never decrease. Therefore, Tpd either remains constant or increases.

This altering of Zo and Tpd by crosstalk explains why termination is never 100% effective. The crosstalk leads to a variation between the targeted Zo and the actual Zo. Termination is usually defined to match the targeted Zo's. The result is an interconnect that is not perfectly matched via termination.

5.0 CHIP SET DESIGN

As simulation tools have improved it has become easier to test design assumptions before actually spending the time or money to build a printed circuit board. In addition, the frequencies at which signals switch have also increased and complicated the process of designing a board that ensures signals reach their destination at the correct point in time and maintain a reasonable level of signal quality. To better predict signal behavior and minimize the need for board rework or revision, many designers are simulating their board layouts before building a board. The complexity of the optimized interface of the CPU-Cache Chip Set is a prime candidate for this type of approach. In this interface designers must ensure that the signals accurately travel along the interconnects at very high frequencies (i.e., 66 MHz). As discussed, transmission line effects become a more dominant influence on signals switching at these frequencies. It is important to take these effects into account to ensure that no specification violations occur.

A possible scenario for designing the optimized interface is shown in Figure 23. As always the first step is to understand the specifications. This document along with the published specifications should help complete this step. Based on these specifications, system geometry requirements, and an understanding of the board's basic electrical characteristics, a first pass component placement and routing can be completed. Once the routing is complete or possibly as part of the routing, individual traces should be simulated to determine their electrical behavior. This includes examining both flight time and signal quality for each signal and determining if it meets the specification. Any signals that violate the specification must be modified. Portions of this document will provide some information and guidelines on how to modify or route the traces to meet the specifications. With each change, the routing should be re-simulated to ensure the specifications are still met.

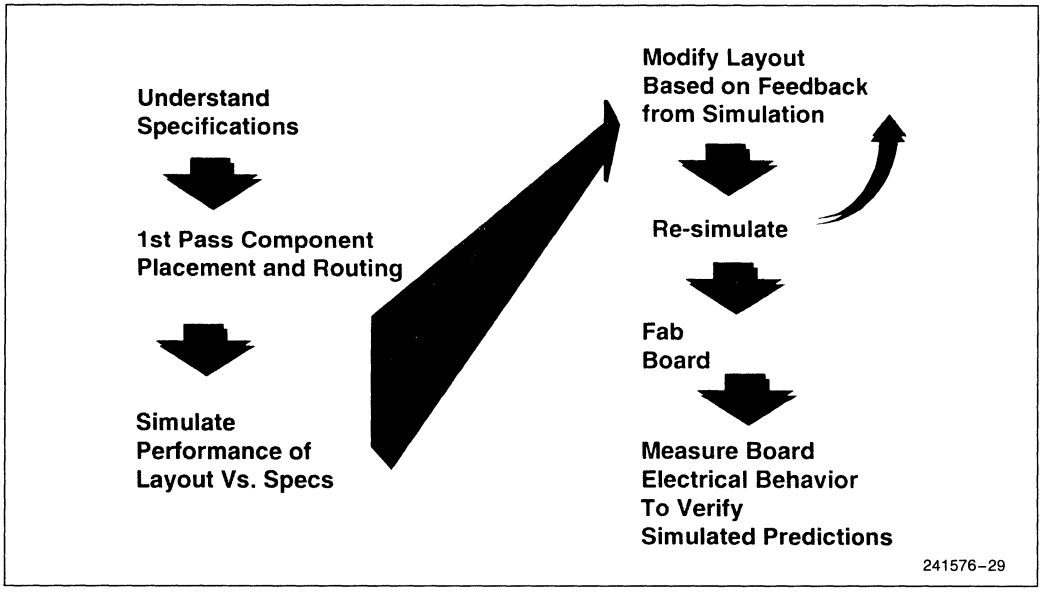


Figure 23. Process for Completing Optimized Interface Design

Once all the specifications are met, it is time to build the board. The goal is that once this board is manufactured and the components are installed, it will meet specification without any changes. However, this must be verified by making actual measurements on the board to verify all of the flight time and signal quality specifications are met. It is also beneficial to make sure that the actual measurements correlate to the predicted results from simulation. This is especially helpful if any corrections are required to bring the board within specification.

The next couple of sections will describe the requirements and guidelines that should be followed while making these simulations and measurements.

5.1 Simulation Environment

The environment chosen to simulate the optimized interface is very critical. A number of different options are available on the market today. It is the system designer's responsibility to select the option best suited for their design requirements. These requirements will include the accuracy of the results; as well as, how easy it is to import schematics, layout routing, or modeling parameters.

5.1.1 SIMULATION REQUIREMENTS

When simulating the optimized interface to determine flight time and signal quality, it is important that the appropriate modeling parameters are used. The I/O models are provided with minimum and maximum values for each parameter. Using these values the fast and slow corners of the buffer's behavior can be modeled. In addition, the printed circuit board can be modeled for its fast and slow corners. Table 4 restates the characteristics of a printed circuit board.

Flight time is determined by simulating with the slow corner used for all parameters. In this corner signals require the longest amount of time to transition and reach their destination. The fast corner is used to simulate signal quality. In the fast corner, signals transition their fastest and are therefore their noisiest. Table 5 summarizes the parameter values used to simulate for flight time and signal quality.

Table 4. Parameters Used to Specify Printed Circuit Board Characteristics

Parameter	Symbol	Description
Characteristic Impedance	Zo (Ω)	Minimum and maximum impedance for signal traces on each layer
Propagation Delay	S (ns/ft)	Minimum and maximum propagation delay for signal traces on each layer
Via Capacitance	Cvia (pF)	Minimum and maximum capacitance of a via used to pass a signal from one layer to another of the PC board

Table 5. Parameter Values Used to Simulate Flight Time and Signal Quality

Device	Modeling Parameter	Flight Time	Signal Quality
Input Buffer	Cp	Max	Min
	Lp	Max	Min
	Cin	Max	Min
Output Buffer	dV/dt	Min	Max
	Ro	Max	Min
	Co	Max	Min
	Lp	Max	Min
	Cp	Max	Min
Printed Circuit Board	Zo	Min	Max
	S	Max	Min
	Cvia	Max	Min
Other	Temperature	Max	Min
	Vcc	Min	Max

These values should be used to define the simulation model files used to simulate for flight time and signal quality.

While simulating the two corners it should become obvious that there will be trade-offs in optimizing for one or the other. Some sacrifices in signal quality may be required to ensure flight time specifications are met, or vice versa.

5.2 Routing Signal Traces for Their Optimal Performance

Priority should be given to optimizing the performance of the signals in the optimized interface. For the 256K byte layout example that Intel completed, the signals have been divided into the categories listed in Table 6. These categories are based on fanout and connectivity characteristics.

Table 6. Optimized Interface Signal Categories

Category	Signal
Low Address (connected to PP, CC, and CS)	A3–A16, HITM#, W/R#
High Address (connected to PP and CC)	A17–31, BT0–3
PP-CC Control (connected to PP and CC)	Driven by PP: ADSC#, AP, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC Driven by CC: AHOLD, BRDYC1#, EADS#, INV, KEN#, NA#, WB/WT#
PP-CS Control (connected to PP and CSs)	ADS#
CC-CS Control (connected to CC and CSs)	BLAST#, BRDYC2#, BUS#, MAWEA#, MCYC#, WBA, WBTYP, WBWE#, WRARR#, WAY
Other CC Control	BLEC#, BOFF#
Byte Enables	BE0#–BE7#
CPU Data and Parity	CD0–CD63, CP0–CP7

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

Within each category the routing or topology should be defined to minimize delay while maintaining acceptable signal quality. To do this and maintain the manufacturability of the board, rules were defined to govern the line lengths for each segment of a topology. To develop these rules some analysis of board characteristics and signal behavior is necessary.

5.2.1 RULES FOR OPTIMIZING SIGNAL ROUTING

Both the fast and slow corners must be considered to ensure both flight time and signal quality are met by optimizing a signal's routing.

Flight time is minimized by optimizing each interconnect to minimize the distance the signal must travel and the loading presented to the driver. The dominant opposition to minimizing these factors is the printed circuit board's geometry requirements (i.e., physical distance between components and component placement) and electrical characteristics (propagation delay and characteristic impedance).

The strategy used to optimize each interconnect for signal quality is to make each net's routing electrically symmetric. This is especially important on heavily loaded nets.

Electrically symmetric means the delays of each branch within the net are equal when viewed from the driver. Figure 24 shows a topology from the 256 Kbyte layout example that illustrates this principle. For this topology with the Pentium processor driving, the symmetry is best when the delay from the Pentium processor to the 82496 cache controller is equal to the delay from the Pentium processor to the farthest 82491 cache SRAM. By making these delays equal, the round-trip delays are also equal, and therefore any reflections return to the Pentium processor simultaneously. By returning simultaneously, the reflections can rapidly cancel each other, resulting in the waveform settling quickly.

If these two delays are not equal, asymmetric reflections return to the Pentium processor at different times, and do not cancel each other. The result is a complex interference pattern that generates considerable ringing. In some cases this ringing can last for more than one clock cycle.

5.2.2 DETERMING THE OPTIMAL NET

There are two methods of optimizing the line lengths and relationships of traces within a net. One uses an asymmetry factor [5] to identify the optimal relationship. The other uses settling time to find this relationship.

Using the asymmetry factor to optimize the symmetry of the net in Figure 24, the input impedance (frequency-domain) of each branch was calculated from the driver's point of view. The branch impedances were compared to each other and the difference was integrated over all frequencies, resulting in a symmetry energy factor which quantifies the amount of symmetry in the topology. Figure 25 also shows a plot of this factor as a function of the lengths of segments L__a and L__b.

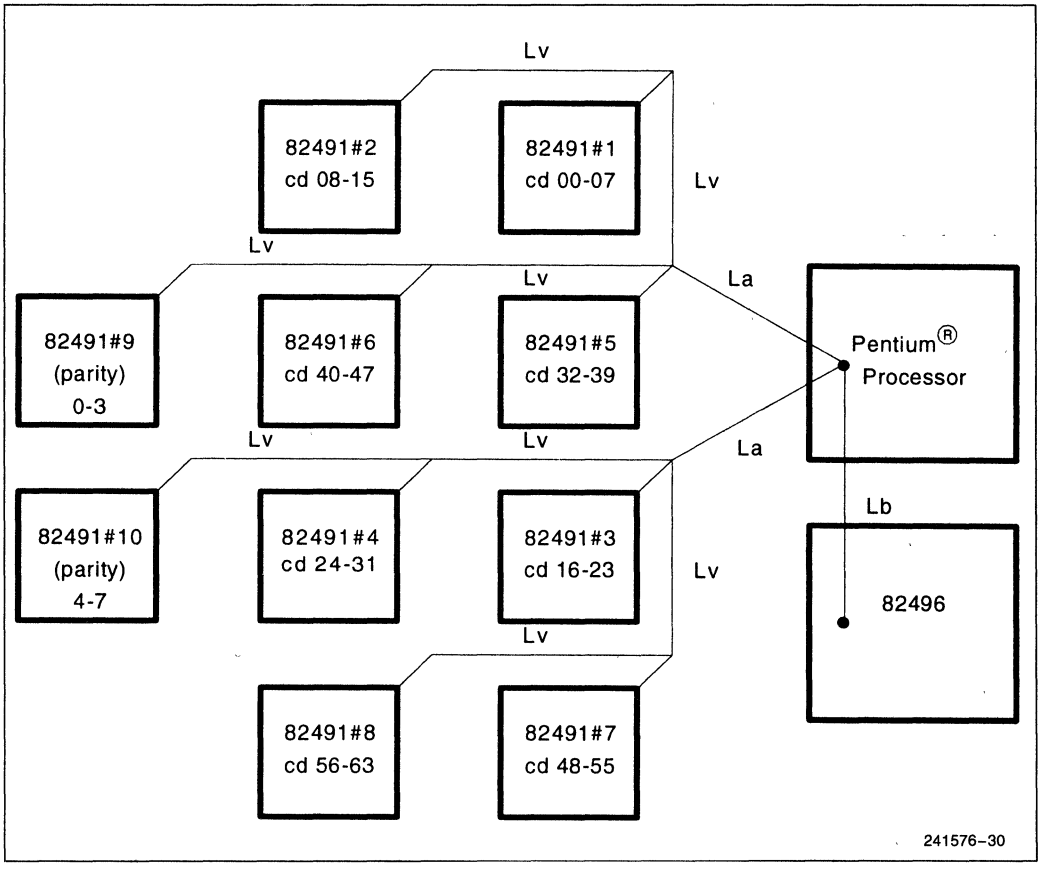


Figure 24. Energy Minimization for a Given Topology

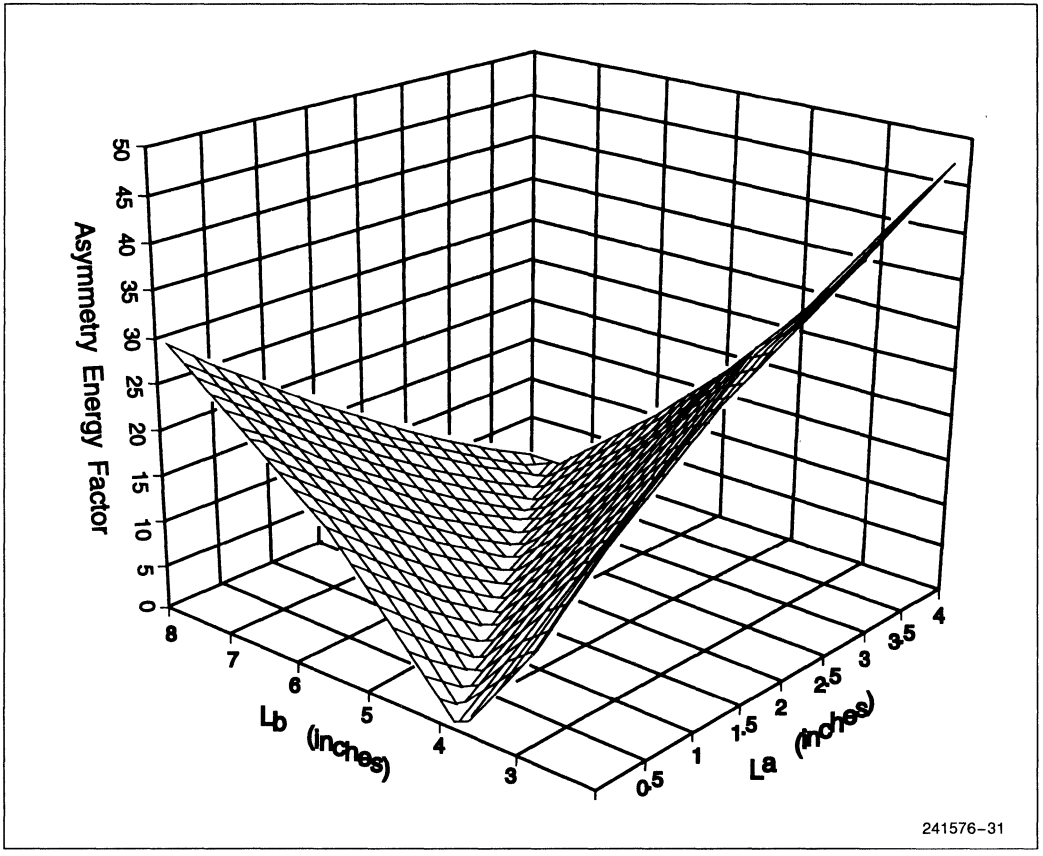


Figure 25. Energy Minimization for a Given Topology

There is a strong correlation between the asymmetry factor and the net's signal quality. This is reinforced by simulating the topology using values for L_a and L_b from the plot in Figure 25. Figure 26 shows the waveforms obtained by simulating the topology with symmetric values for L_a and L_b from along the energy minimum. The line of points in the plot of Figure 25 where the energy minimum occurs corresponds to topologies that are electrically symmetric. An asymmetric topology is obtained by using values for L_a and L_b that lie away from the energy minimum. The waveform obtained by simulating this asymmetric case is also shown in Figure 26. Notice the difference in signal quality in the two plots. The symmetric case is much better than the asymmetric.

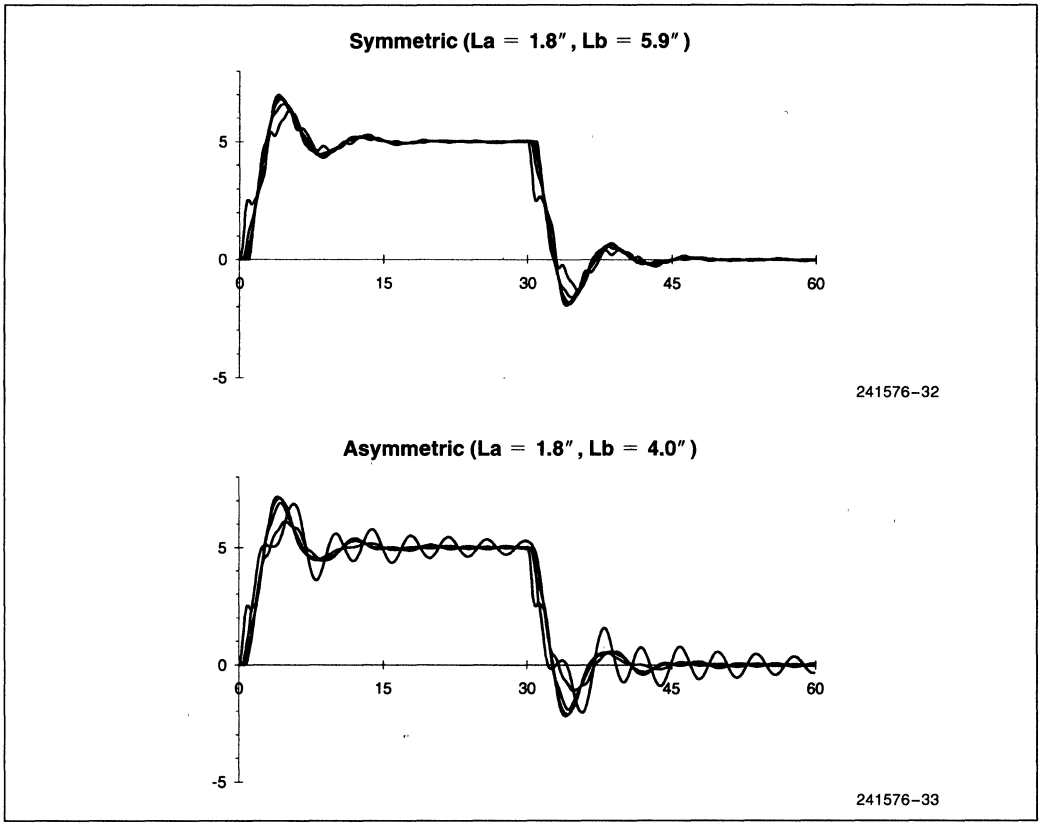


Figure 26. Symmetric Versus Asymmetric Values for La and Lb

This technique can be used to optimize the routing of all heavily loaded signals in a chip set design. From the energy factor plot rules can be defined to govern the segment lengths needed to minimize the energy factor and obtain the specified signal quality.

The 256K layout example that follows used this technique extensively to route the heavily loaded signals. For each signal group or topology, the asymmetry energy factor was calculated as a function of the topology's segment lengths and a set of rules defined to govern the segment lengths required to provide a routing that meets the signal quality specifications.

Similar results can be determined by using settling time to the optimal routing. To optimize the symmetry of a net, the settling time is plotted against line length. The minimum settling time occurs at the point where the net is balanced. Figure 27 shows a settling time plot for the net in Figure 24. Settling time is plotted against the true length for the segment between the Pentium processor and the 82496 for a given length between the Pentium processor and 82491. For La = 1.8 inches the settling time approach recommends $L_b = 5.8-6.0$ inches.

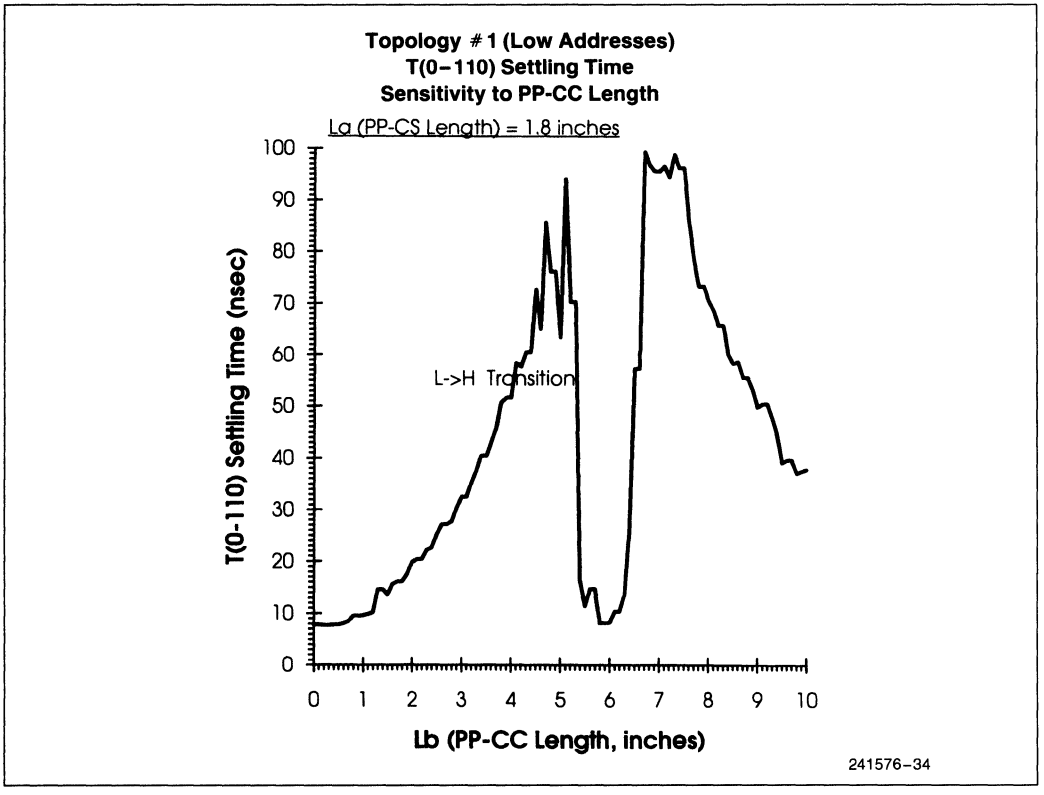


Figure 27. Settling Time Versus Line Length

5.2.3 SERPENTINE STRUCTURES

Serpentine structures are one design technique that can be used to assist in balancing the interconnect delays between the Pentium processor, 82496 cache controller, and 82491 cache SRAM components. The structure is used to add length to specific traces within the nets. The goal of adding this length is to make the net "balanced" or electrically symmetrical. In particular, this technique has been used to add length to the trace between the Pentium processor and 82496 cache controller so that it is electrically symmetric with the traces between the Pentium processor and 82491s of the same net as shown in Figure 28.

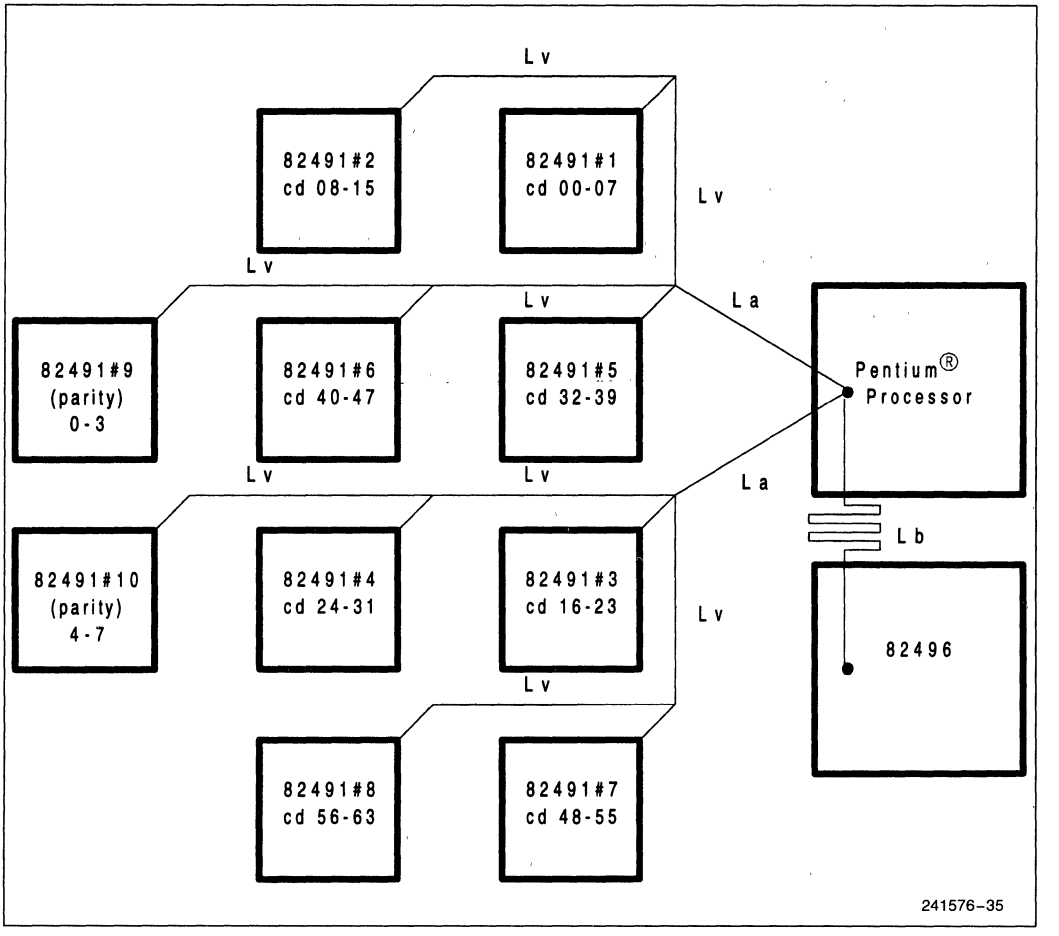


Figure 28. Balancing Nets Using Serpentine Structures

Due to the parallel traces that make up the serpentine, cross-coupling may occur between the individual portions of the serpentine. The cross-coupling may cause the propagation velocity and characteristic impedance of the serpentine to differ from those of a straight line of equivalent length. In general, the propagation velocity may be greater and the characteristic impedance less for the serpentine structure. To simplify the simulation environment for the CPU-cache-chip set design example, the added trace length was assumed to be equal to the length of trace used to make the serpentine. See Figure 29.

Experiments were performed to confirm that this assumption was valid. The experiments involved Time Domain Reflectometry (TDR) and Time Domain Transmission (TDT) measurements on various serpentine configurations. The height of the serpentine, h, and the separation, s, were varied.

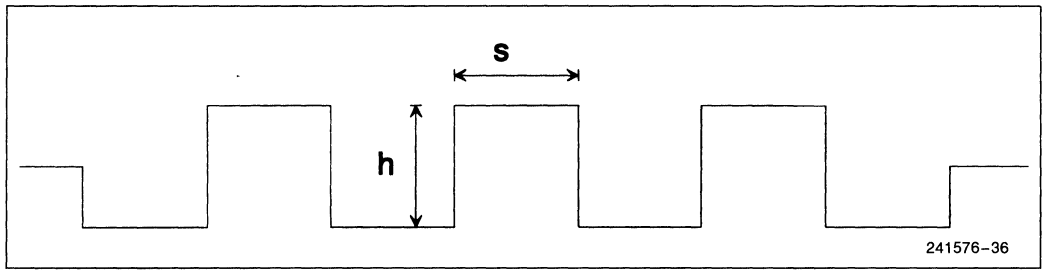


Figure 29. Parameters of a Serpentine Structure

It was found that as the separation was increased or the height decreased the propagation velocity increased. Amount of increase varied from being almost negligible to being approximately 40% for short, closely spaced serpentines. Also, it was observed that as the height decreases the magnitude of the decrease in impedance gets smaller, with the largest decrease in impedance, approximately 12%, seen when the height and separation are at their smallest. The serpentines used in the CPU-cache-chip set design example were not the worst case configuration. Based on the experiments, the serpentines should cause less than 10% decrease in the characteristic impedance and less than 30% decrease in the propagation delay.

Both of these variations appear considerable at first glance. However, serpentines, as used in the CPU-cache-chip set design example, account for only a small percentage of the entire trace length of a net. For example, if the serpentine is only 25% of the total trace length and the total propagation delay is 2 ns, representing the trace as a straight line length only introduces a maximum error of about 150 ps. This is determined by assuming the 25% of trace accounts for 0.5 ns delay and a 30% decrease is 150 ps. Based on the small amount of error introduced, it was decided that a straight line representation was accurate enough and simplified the simulations.

Note the variation in effects caused by the serpentines. Each design should perform similar analysis if a different serpentine structure than that used in the CPU-cache-chip set design example is used.

If the designer chooses to include the propagation velocity and characteristic impedance variations in the simulations, the only change is to represent the serpentine length of trace as a separate length with the different characteristics.

6.0 EXAMPLE: DESIGNING THE A12 NET FOR THE CPU-CACHE CHIP SET

The A12 net is one of the more complex nets in the optimized interface of the CPU-Cache Chip Set. The signal is driven by the Pentium processor to the 82496 cache controller and all the 82491 cache SRAMs during memory reads. In addition, the 82496 cache controller drives this signal to the Pentium processor during inquire cycles.

In routing A12 net all of the guidelines and techniques described were used. An initial routing of the A12 net was made using an H-type routing and attempting to make all of the interconnects as short as possible. The resulting topology is shown in Figure 30.

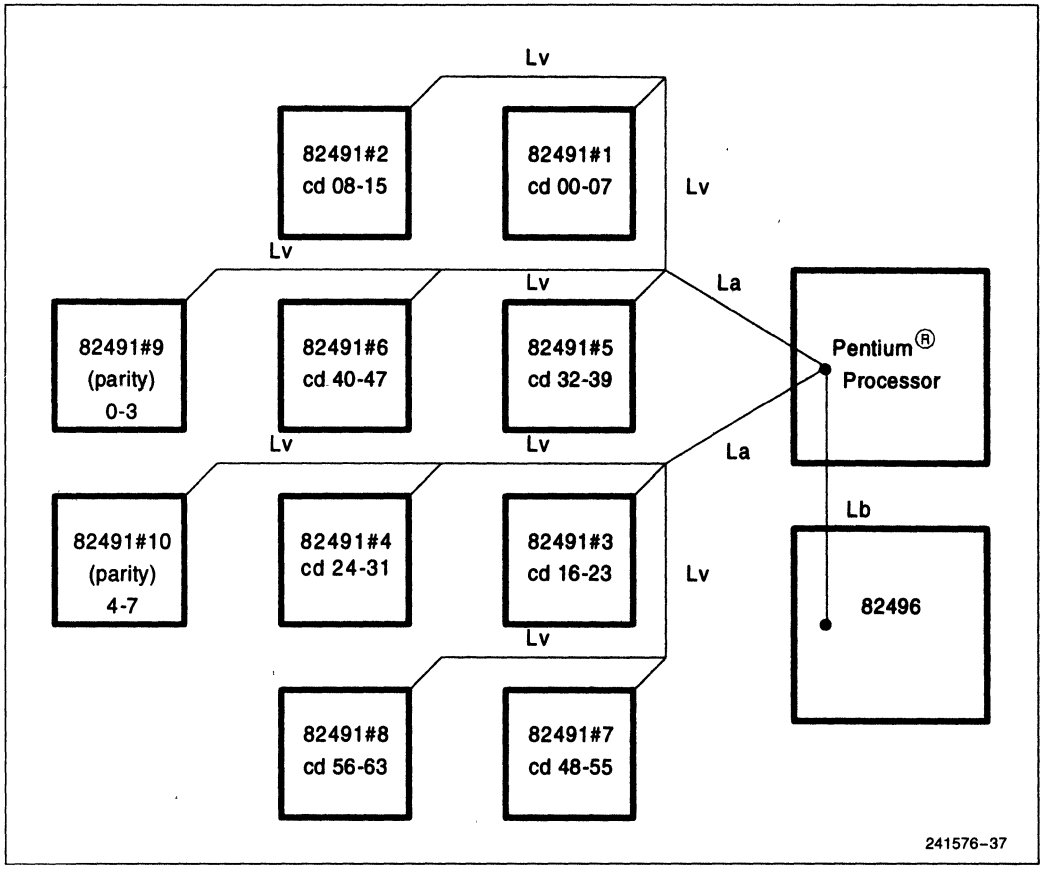


Figure 30. Initial Topology for A12 Net

The A12 net was simulated assuming the Pentium processor is driving the net. Quad Design's TLC was used to simulate the A12 net. For flight time the slow corner is used. The slow corner uses the model parameters as defined in Table 5. The actual values can be obtained from the *Pentium® Processor User's Manual.* A TLC control file calls the appropriate model and topology files along with setting the needed measurement points to complete the flight time simulations.

Initially, the line lengths or segments between components was assumed to be the straight line distance. In other words, the initial routing conserved space and used the shortest line possible to connect the components. The rising and falling waveforms resulting from the TLC simulations of this routing are shown in Figure 31.

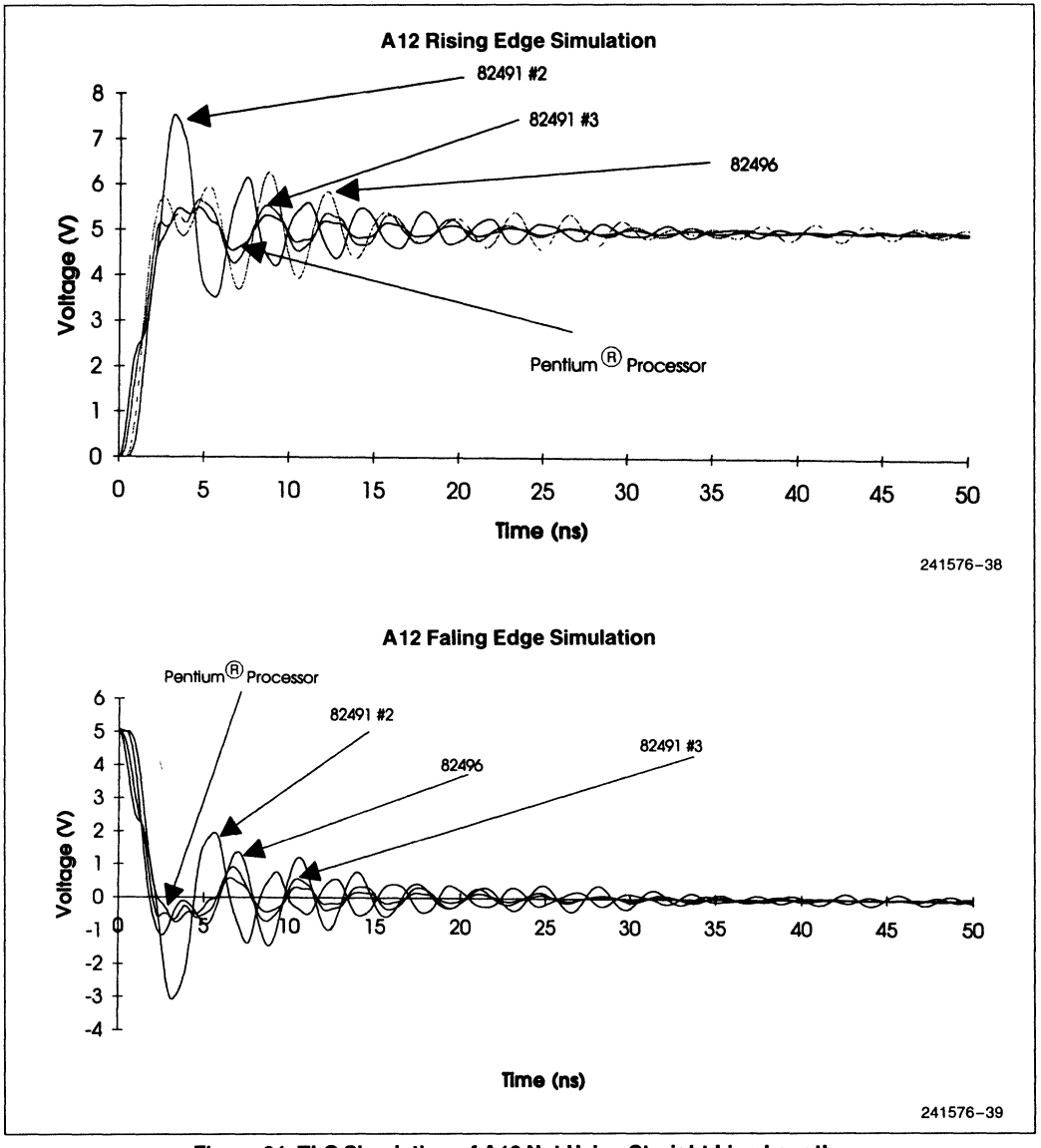


Figure 31. TLC Simulation of A12 Net Using Straight Line Lengths

Notice the large amount of ringing that occurs in this routing of the net. The excessive ringing can cause failures in both the signal quality and flight specifications. To bring the net within specification the routing must be improved to better "balance" the net. At first glance, the loading on the 82496 cache controller branch is much less than the 82491 cache SRAM branch. Splitting the 82491 cache SRAM branch into two branches and continuing the H-type routing along those branches and lengthening the 82496 cache controller branch should improve the "balance." The asymmetry energy factor, described in Section 5.2, was used to derive the relationship between individual trace segments needed to balance the net. The relationship is that:

$$Lb = 2 * La + Lc + 800 \text{ mils} ; La \leq 2000 \text{ mils}$$

$$1.6 * La + Lc + 1500 \text{ mils} ; La > 2000 \text{ mils}$$

Following these rules ensures that the A12 net is electrically symmetric. Figure 32 illustrates this improved routing.

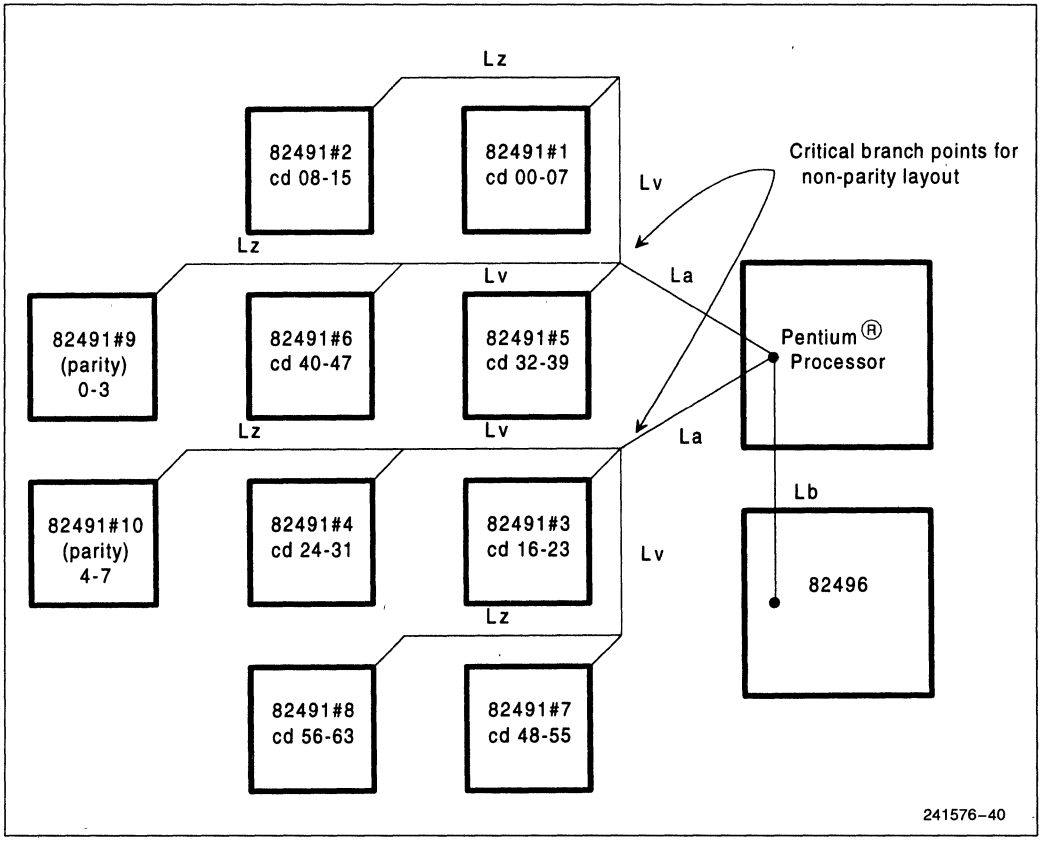


Figure 32. Improved Topology of A12 Net

The improved net was also simulated using TLC and the resulting waveforms are shown in Figure 33 through Figure 35. Notice the reduction in ringing. With the "balanced" or electrically symmetric routing the net exhibits better flight time and signal quality. In fact these parameters are now within specification as summarized in Table 7.

The technique for measuring flight time and signal quality are described in detail in Section 2.0. To measure flight time one must first determine the 50% point of the unloaded Pentium processor driver as shown in Figure 33. In the example this occurs at 2.26 ns. Next one must determine where the waveform crosses the 50% Vcc point at the receiver as shown in Figure 34. This crossing occurs at 4.54 ns. The time difference between these two points is the 50–50 flight time. The 50–50 flight time for the A12 net example is 2.28 ns.

Flight time also assumes the waveform continues through the 50% Vcc point with a slope of at least 1 V/ns through the 65% Vcc point. To ensure this, first determine the 65% point on the A12 flight time simulation as shown in Figure 35. The 65% Vcc point is 5.32 ns. Next extrapolate using the 1V/ns line to find where it crosses the 50% voltage level by subtracting 0.68 ns from the 65% Vcc number. The extrapolated 50% Vcc point for the example is 4.64 ns. The time difference between the unloaded buffer's 50% point and the extrapolated crossing of the 50% point is the 50–65 flight time. The 50–65 flight time is 2.38 ns.

The greater of the 50–50 and 50–65 flight times is the flight time for the net. In this case, the flight time is 2.38 ns, the 50–65 flight time.

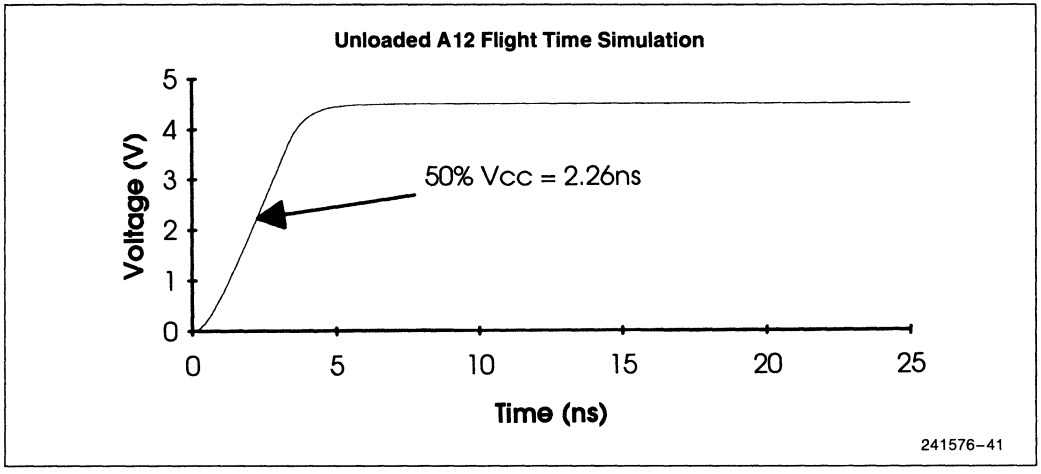


Figure 33. Measuring the 50% Vcc Point of the Unloaded Output

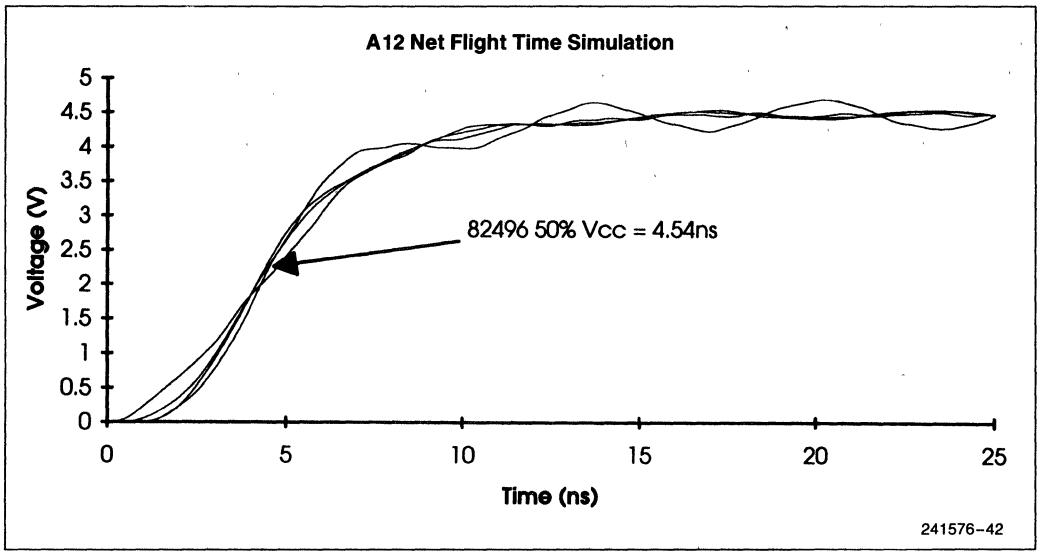


Figure 34. Measuring the 50% Vcc Point at the A12 Input of the 82496

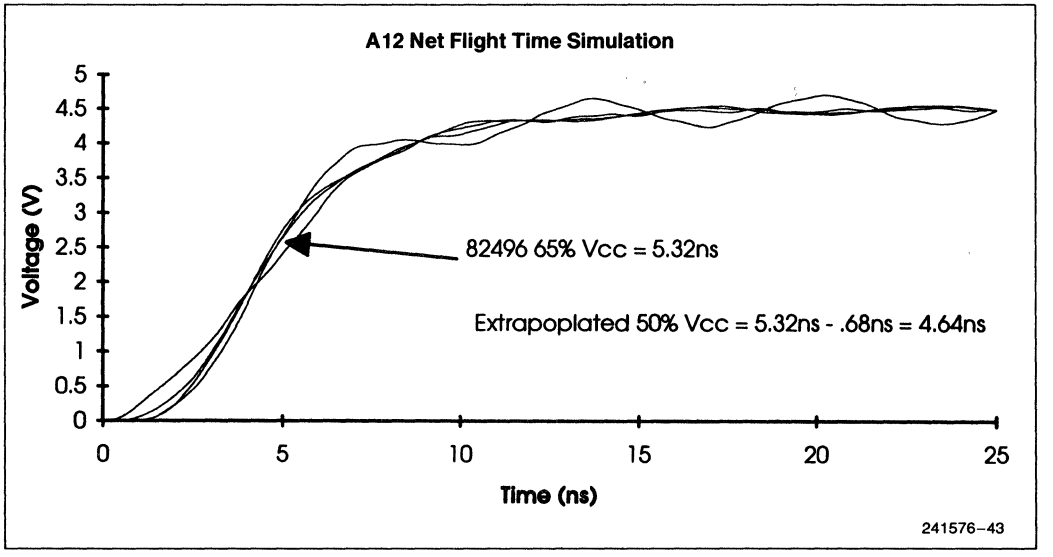


Figure 35. Measuring the 65% Vcc Point at the A12 Input of the 82496

Figure 36 and 37 illustrate the measurement of the signal quality parameters for the A12 net. Overshoot is the maximum voltage above Vcc. Time beyond supply is measured between points A and B. Ringback is the maximum voltage amount that the signal cross back across Vcc. Settling time is measured from point C and D.

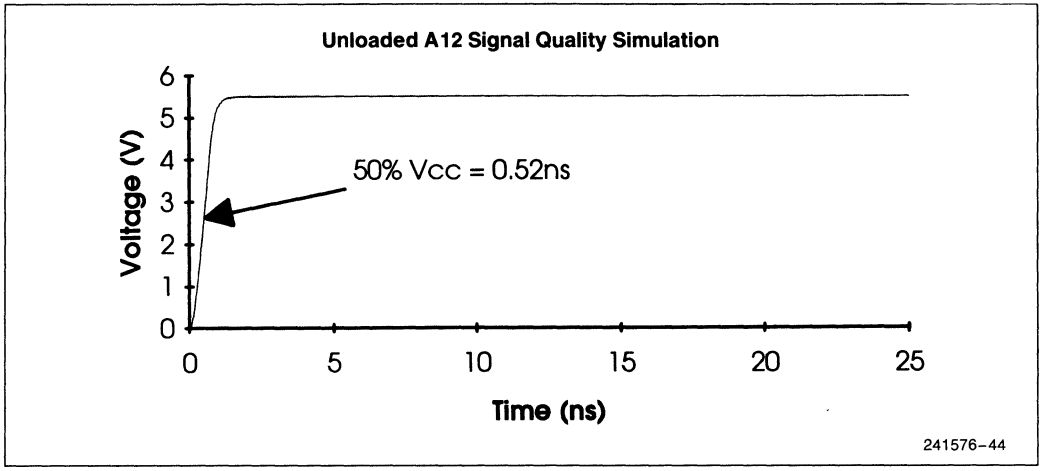


Figure 36. Measuring the 50% Vcc Point of the Unloaded Output at the Fast Corner

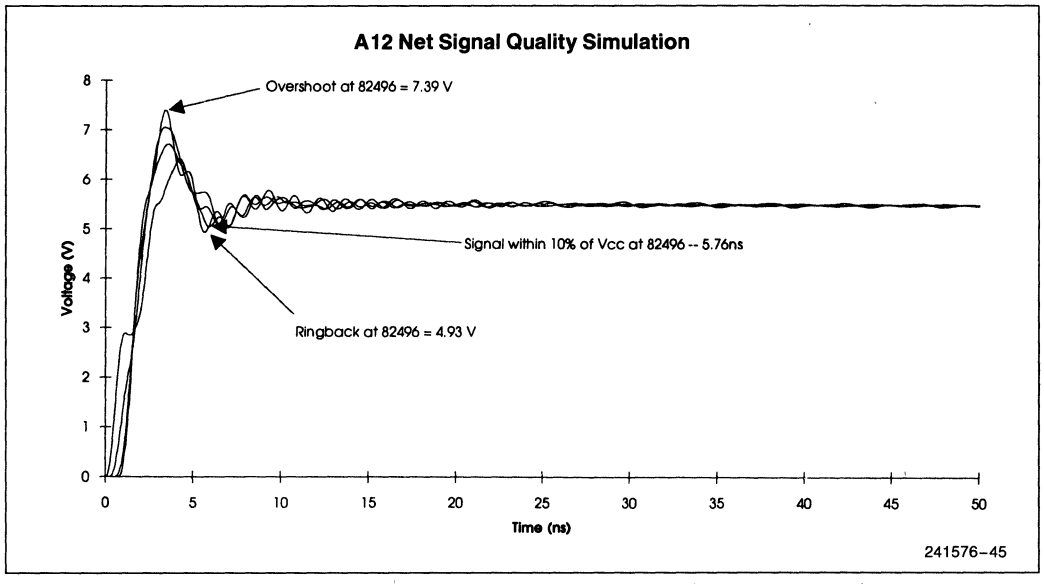


Figure 37. Measuring the Signal Quality of the 82496

The flight time and signal quality specifications for this net are listed in Table 7 along with the values measured in the simulation of the net.

Table 7. Flight Time and Signal Quality Simulated Values

Signal	Flight Time		Overshoot		Ringback		Settling Time	
	Spec	TLC	Spec	TLC	Spec	TLC	Spec	TLC
A12	28 ns	2.38 ns	3.0V	1.89V	35% Vcc	0.57V	12.5 ns	5.76 ns

7.0 256K CPU-CACHE CHIP SET OPTIMIZED INTERFACE LAYOUT DESIGN EXAMPLE

This chapter contains an example layout design for Intel's 256 Kbyte CPU-Cache Chip Set's optimized interface. Intel has simulated and verified the example layout using the latest information. Work is currently underway to validate the design by measuring the flight time and signal quality parameters on boards based on the design example. As updated information becomes available on the components and the boards, Intel plans to update this example accordingly.

The intent of the design example is to provide system designers a starting point. It provides one solution of how the Pentium processor, 82496 cache controller, and 82491 cache SRAM components can be placed and routed to ensure flight time and signal quality specifications are met. It is not the only solution. System designers can alter the layout to meet their system requirements as long as the flight time and signal quality specifications are met.

7.1 Layout Objectives

The 256K layout is an example of a CPU-Cache chip set arrangement that meets Intel's chip set specifications. The layout consists of 1 Pentium processor, 1 82496 cache controller, and 10 82491 cache SRAMs for a 256K second-level cache with parity. Although the layout is specifically designed for a chip set with parity, we will also discuss conversion to a non-parity layout.

This example layout follows the chip set's flight time and signal quality specifications. In addition to meeting those specifications, we had the following objectives:

1. To design the optimized portion of the interface so that the layout is not limited by interconnect performance. By not artificially creating any critical paths, the interface can yield maximum performance of the chip set.
2. To be consistent with EMI and thermal requirements.
3. To have the layout be used as a validation and correction vehicle by Intel. Intel will use the layout to validate the optimized interface of the chip set, measure flight times and signal quality, and tune input and output buffers.

Provided are complete specifications for a board layout: part lists, board layer plots, and the electronic files in Gerber format. Also provided are a set of topologies and line lengths so it will be easy to understand how the layout was generated.

7.2 Component Placement

To meet flight time with clock skew restrictions we placed the parts in relative proximity to each other. At the same time, we ensured that the layout's Memory Bus Controller (MBC) interface signals are routable. Figure 38 illustrates how the chip set components are placed in the layout example. The dot indicates the location of pin 1. Figure 38 component side view of the layout.

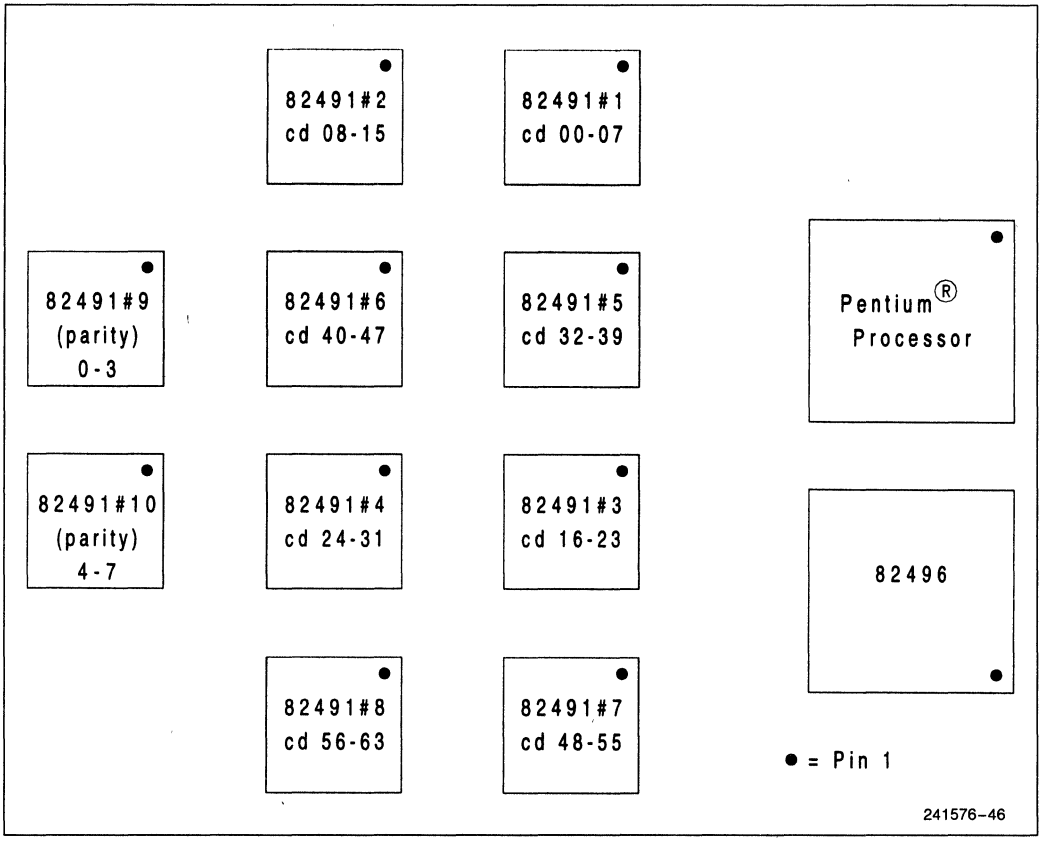


Figure 38. Component Placement

7.3 Signal Routing/Topologies

Tables 8 and 9 list the signal nets and their corresponding topologies for the optimized and external interfaces of the CPU-Cache Chip Set.

All chip set signals in the optimized interface fall into six groups: low addresses, high addresses, Pentium processor control, 82496 control, CPU data, and byte enables. Within each group are subsets of signals that share common origination and destination points. Each subset has a unique routing called a "topology." Groups, subsets, and topologies are listed in Table 8.

Topologies are given only for signals that are routed to multiple chips. It is the system designer's responsibility for routing the "point-to-point" signals such as CADS#.

Topologies are also supplied for the external interface. These topolgies provide channels for routing signals from the chip set components to the periphery where they can be connected to the memory bus and memory bus controller (MBC). However, topologies are not supplied for point to point signals in the MBC interface (e.g. CRDY#). Instead, the system designer must optimize these for the particular application.

Table 9 lists the topologies provided for the MBC interface signals which are not point to point.

Table 8. Optimized Interface Signal Net/Topology Assignments

Grouping	Routing Requirements	Topology
Low Addresses		
(PA3–PA16)	Bused to all core components. Must be routed to optimize delay and signal quality at all points.	1
High Addresses		
(PA17–PA31, PBT0–PBT3)	Point to point links. Must be kept as short as possible.	4
Pentium® Processor Control		
(HITM#, W/R#)	Same as low addresses.	1
(ADS#)		3b
(ADSC#, AP, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC)	Same as high addresses.	4
CC Control		
(BRDYC2#, WRARR#, MCYC#, WAY, BUS#, MAWEA#, WBWE#, WBTYP#, WBA, BLAST#)	Must be routed to optimize delay and signal quality at the CS.	3
(BLEC#)	Not routed to parity CSs.	3a
(BOFF#)		1b
(AHOLD, EADS#, KEN#, BRDYC1#, INV, EWBE#, NA#, WB/WT#)	Same as high addresses.	4
CPU Data		
(CD0–CD63)	Point to point signals. Keep as short as possible. Keep the total length of each trace within 1/2" of each other to minimize skew.	4
Byte Enables		
(CBE0#–CBE7#)		5

Table 9. External Interface Signal Net/Topology Assignments

Signal	Topology
RESETC50, CRDY0#	10
CRDY#, RESETC51	11
MBRDY#, MOCLK, MDOE#	12, 13
MFRZ#, MSEL#, MZBT#, MCLK	14
BRDY#, CLK0	15
CLK1, BRDY1#, MEOC1#	16
CLK2, BRDY2#, MEOC2#	17
CLK3, BRDY3#, MEOC3#	18
MDATA0–63	19

Figures 39 through Figure 58 are the topologies which are described in Tables 8 and 9. A topology is a graphical representation how specific sets of signals are routed. A topology shows the components that share a specific signal and the relative lengths of the traces between components.

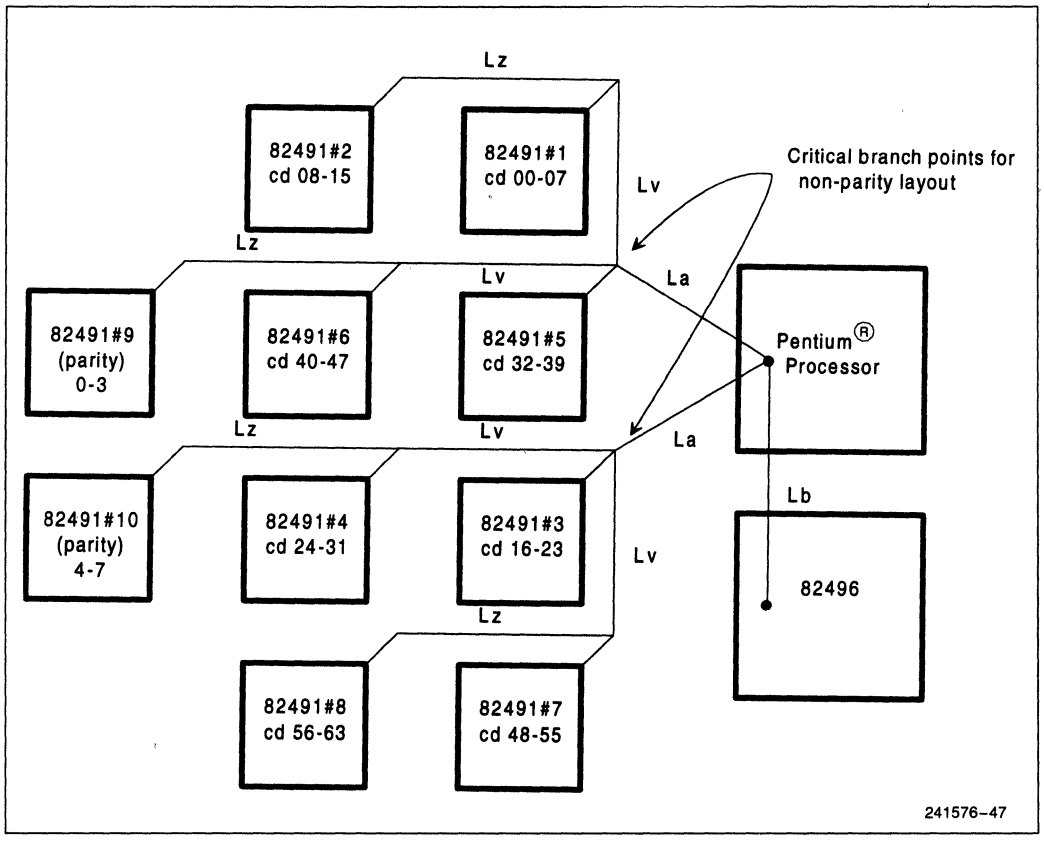


Figure 39. Topology 1

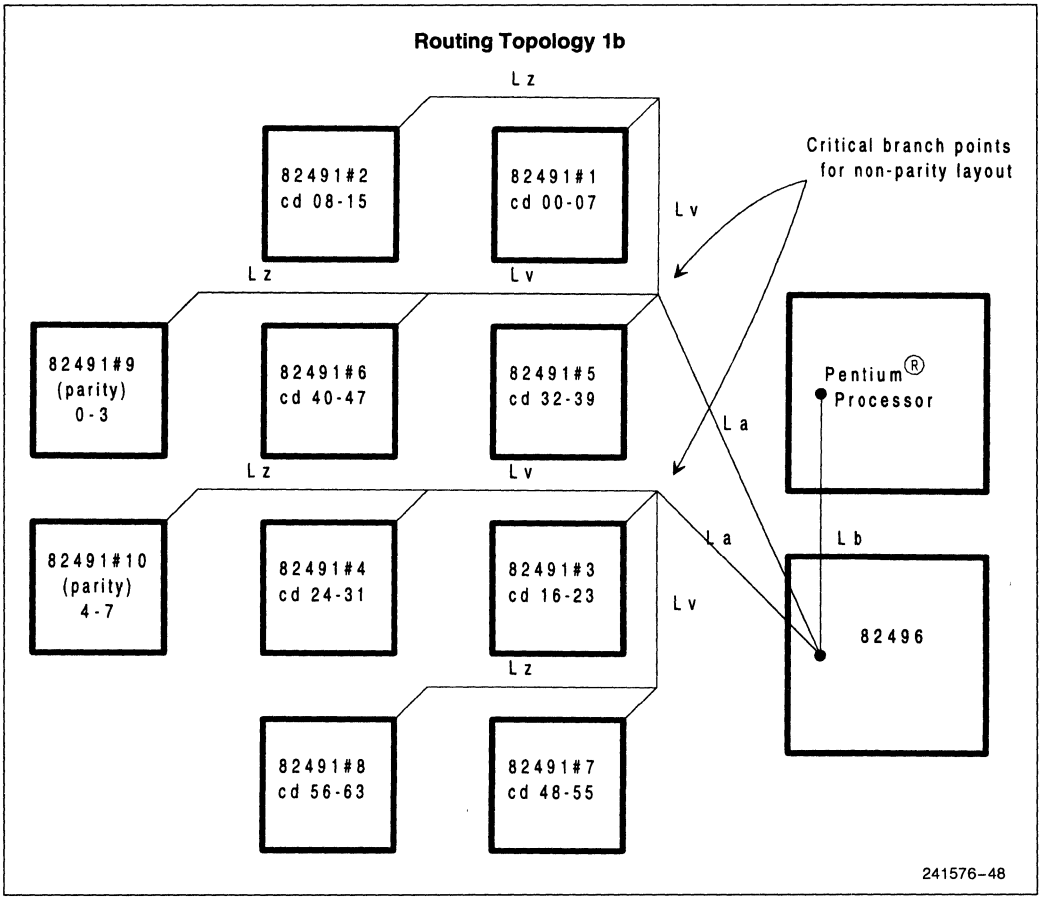


Figure 40. Topology 1b

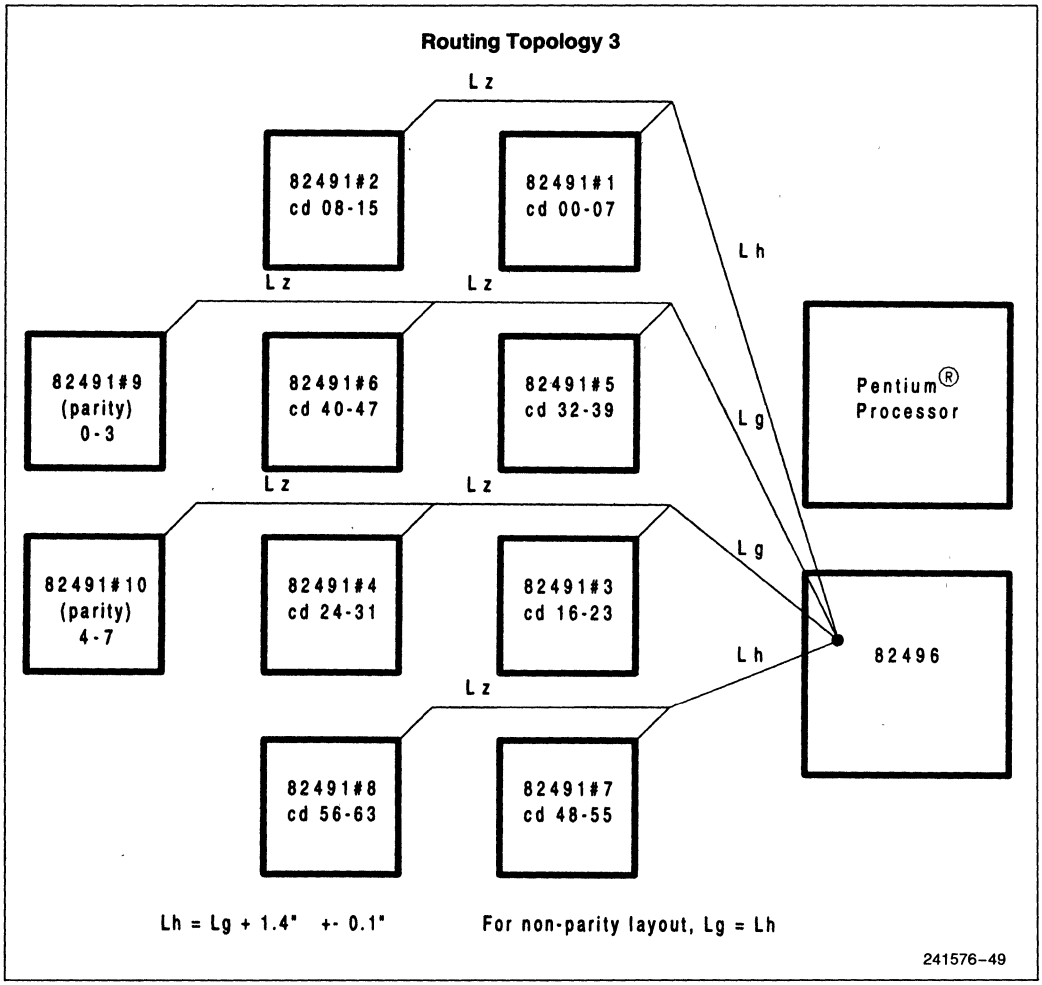


Figure 41. Topology 3

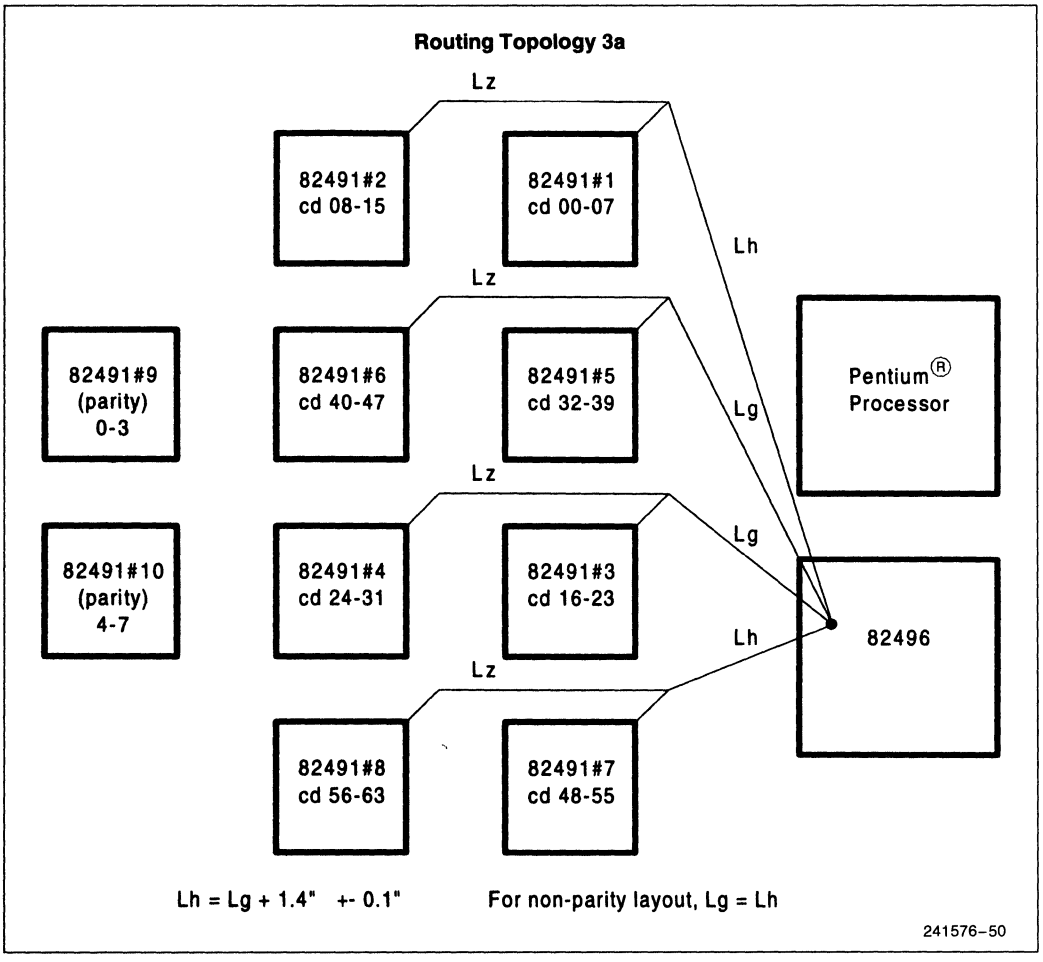


Figure 42. Topology 3a

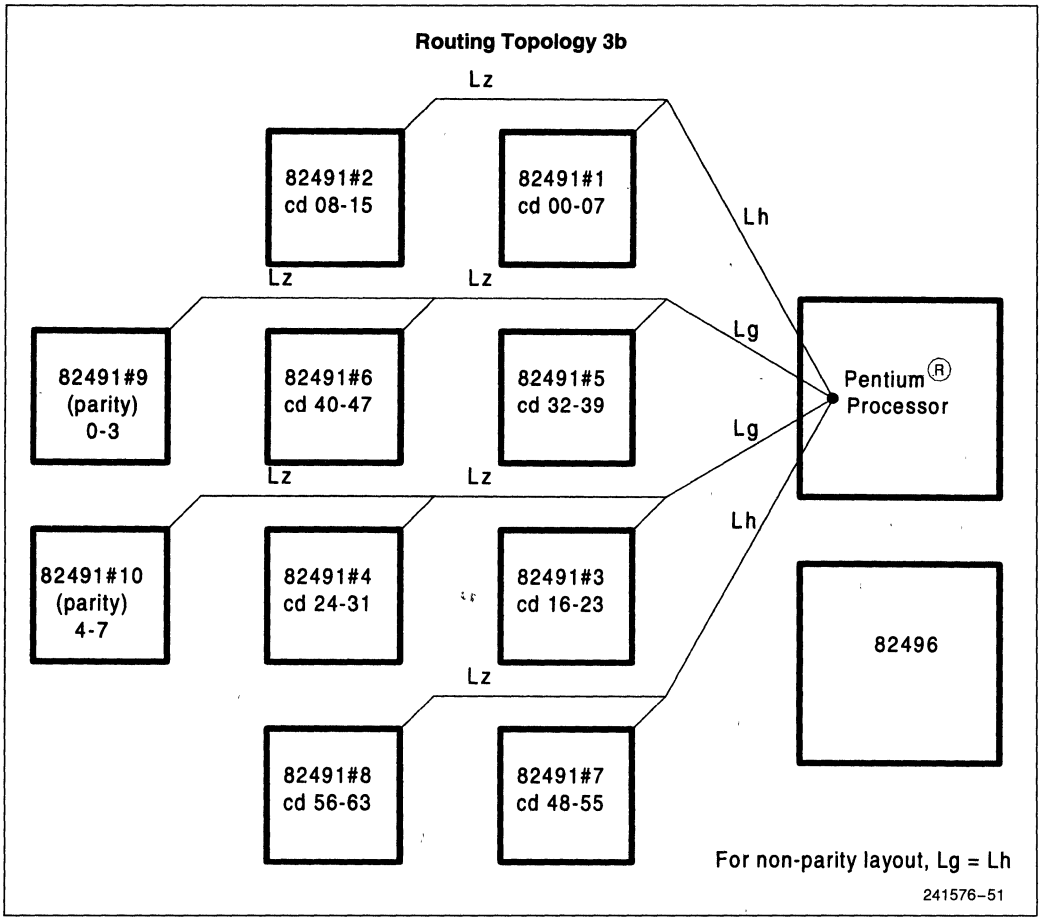


Figure 43. Topology 3b

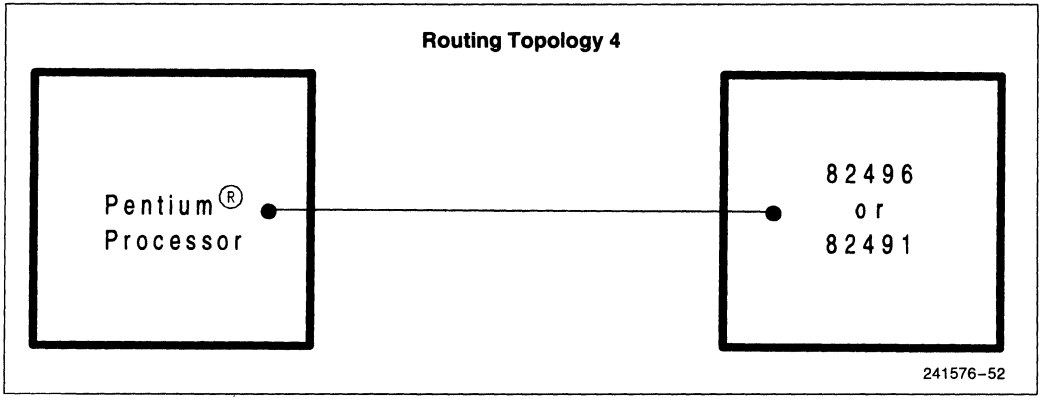


Figure 44. Topology 4

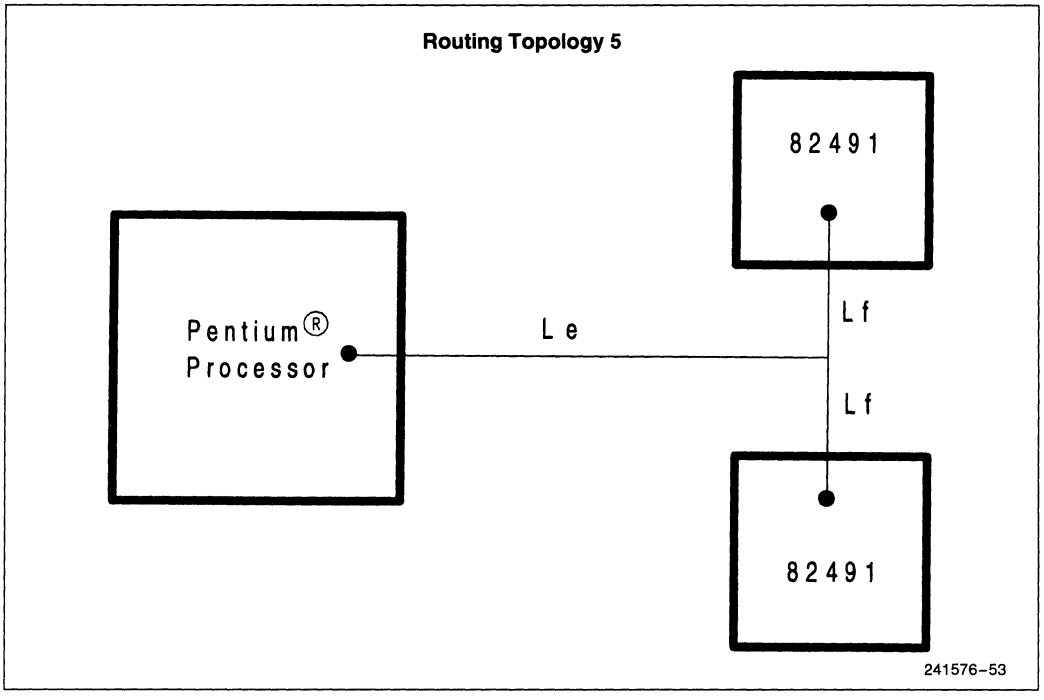


Figure 45. Topology 5

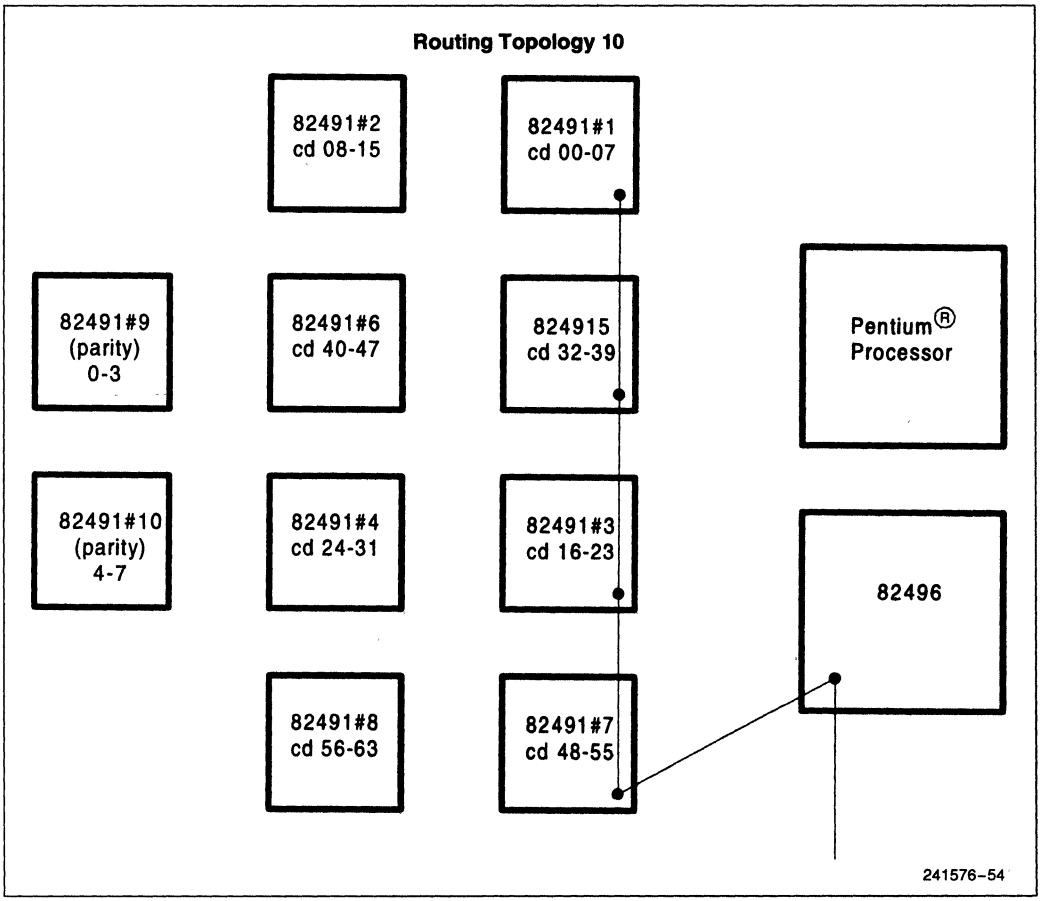


Figure 46. Topology 10

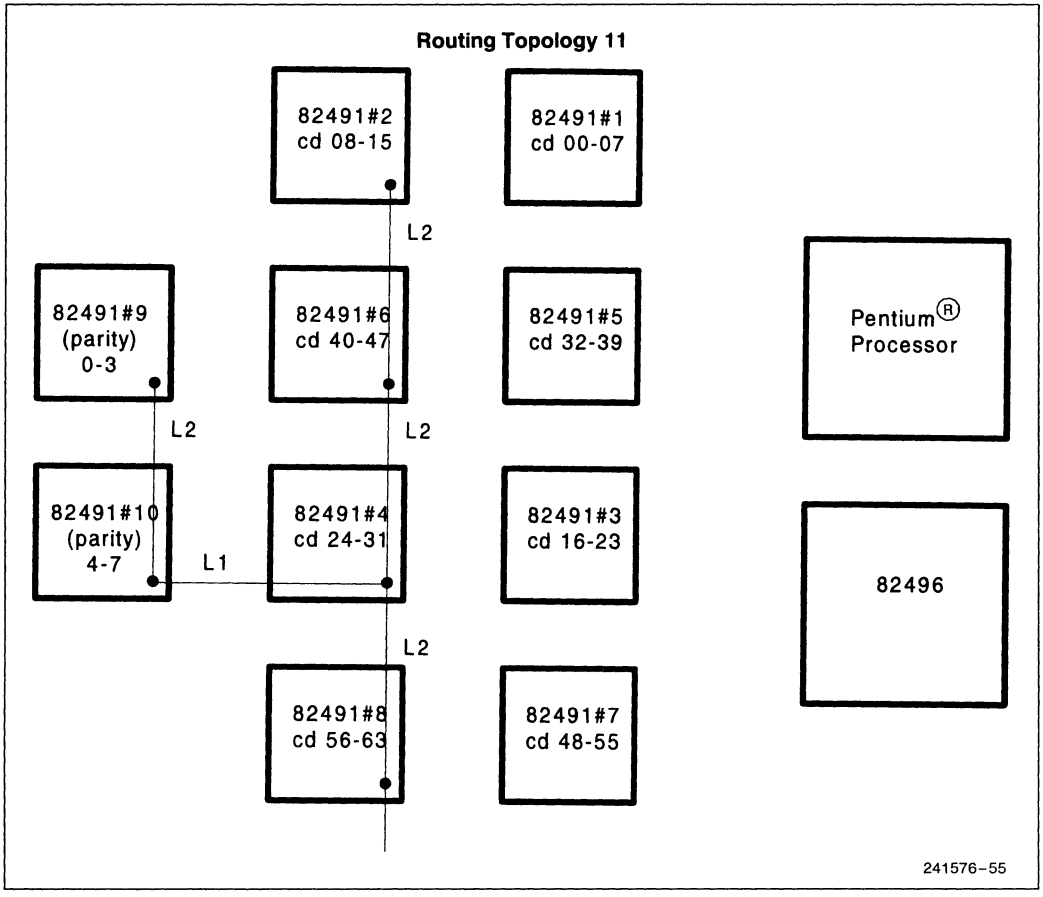


Figure 47. Topology 11

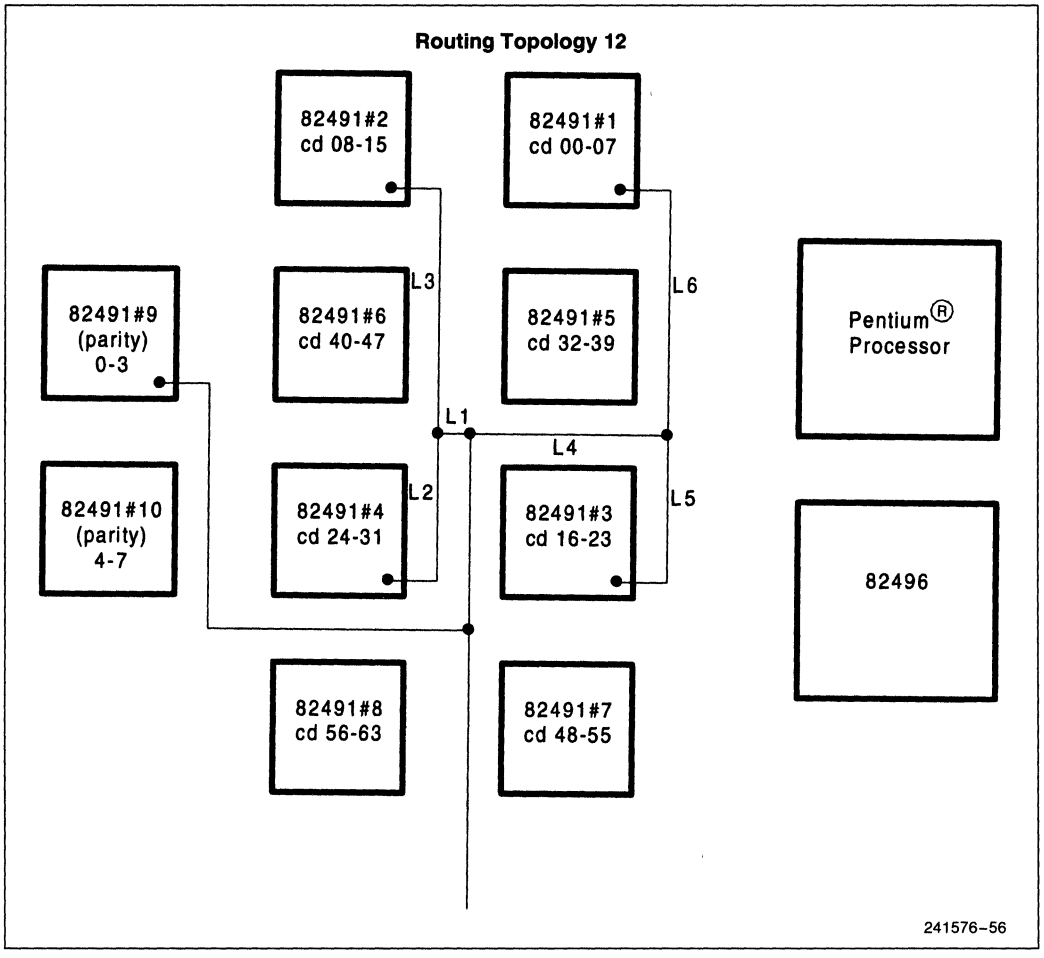


Figure 48. Topology 12

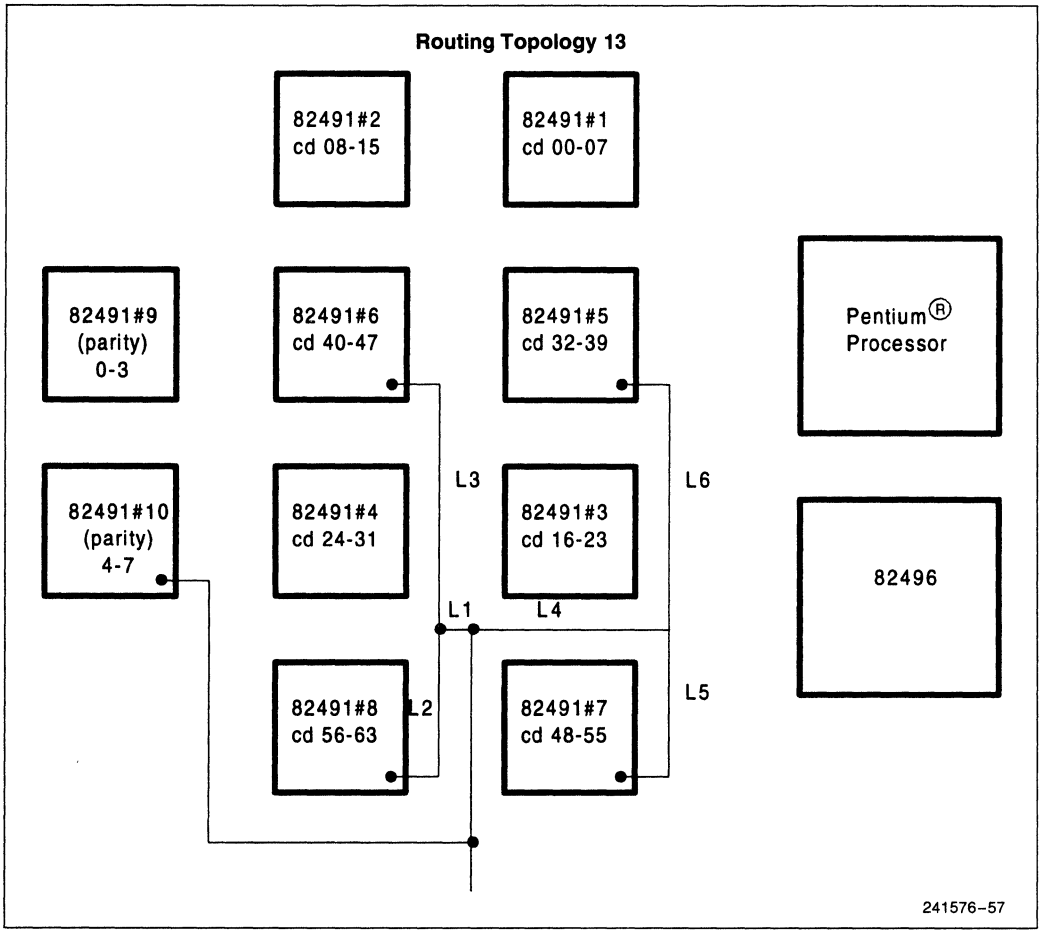


Figure 49. Topology 13

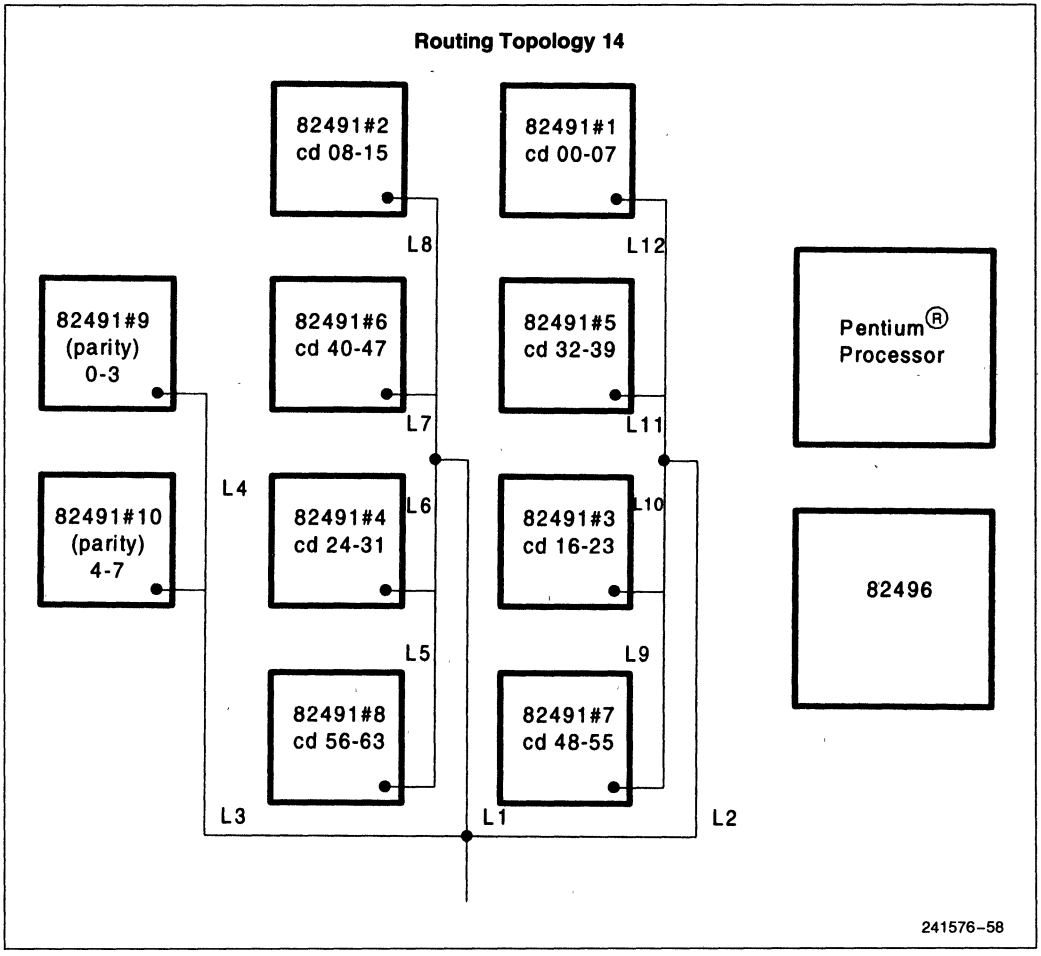


Figure 50. Topology 14

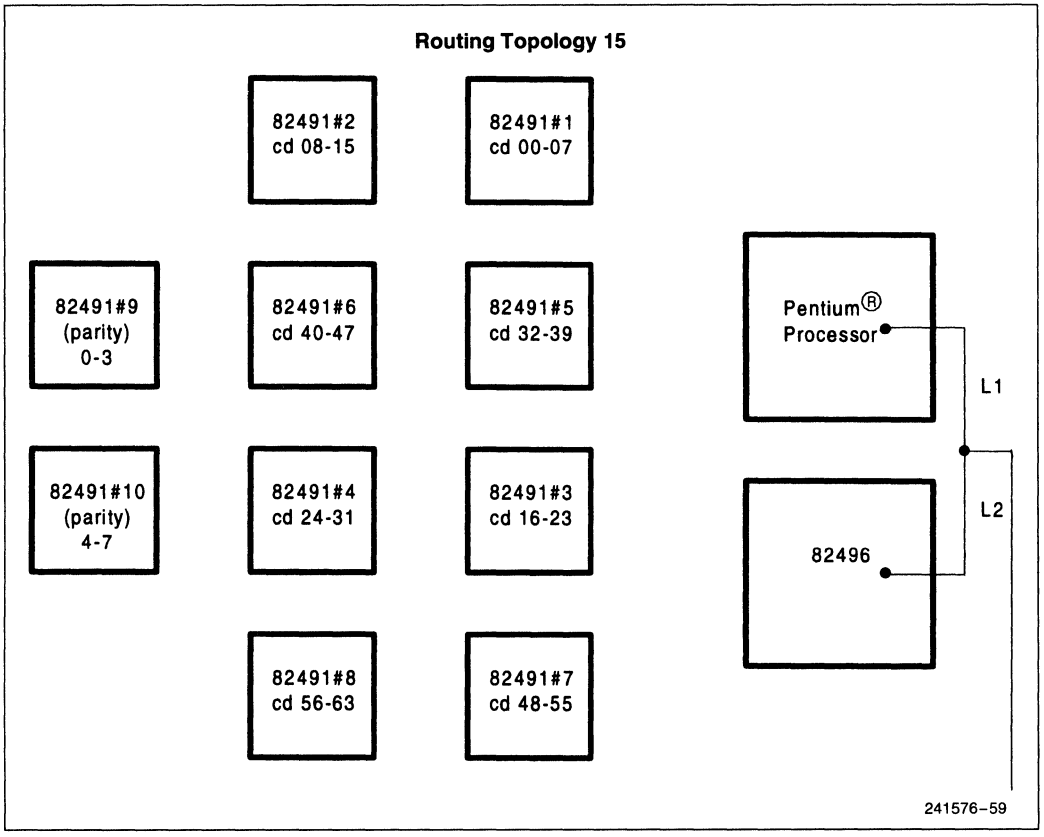


Figure 51. Topology 15

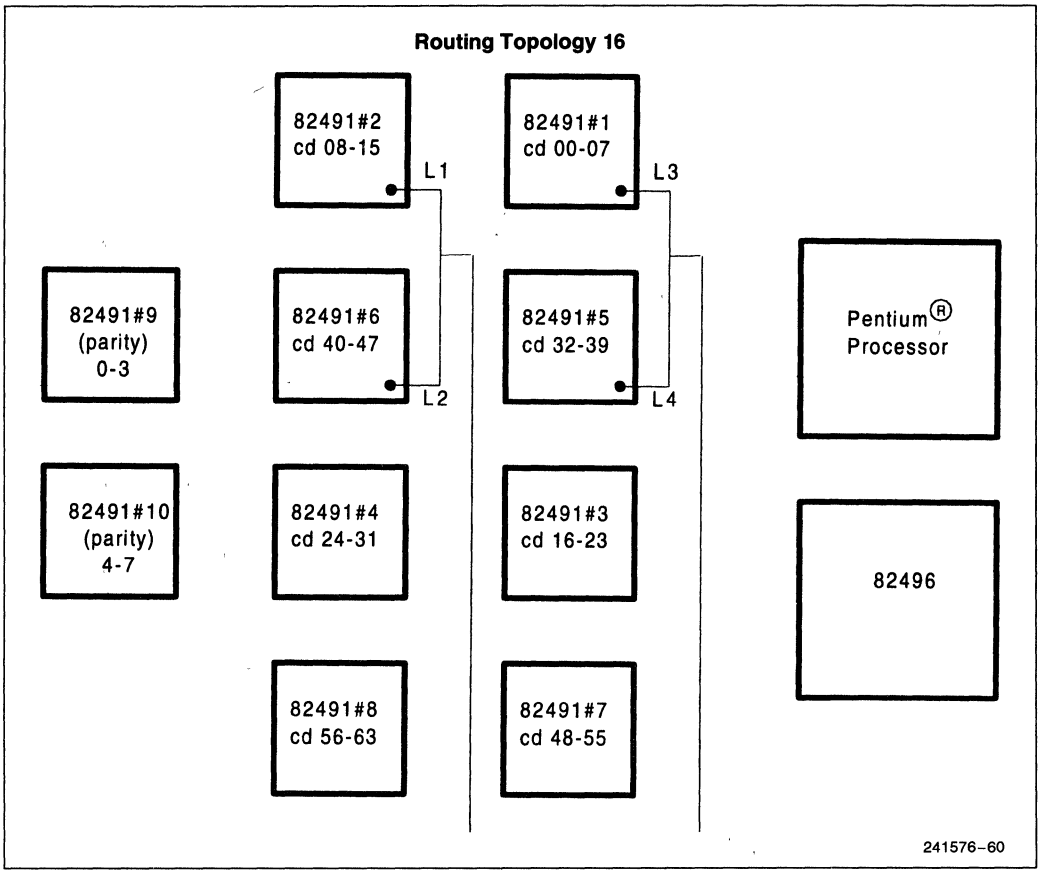


Figure 52. Topology 16

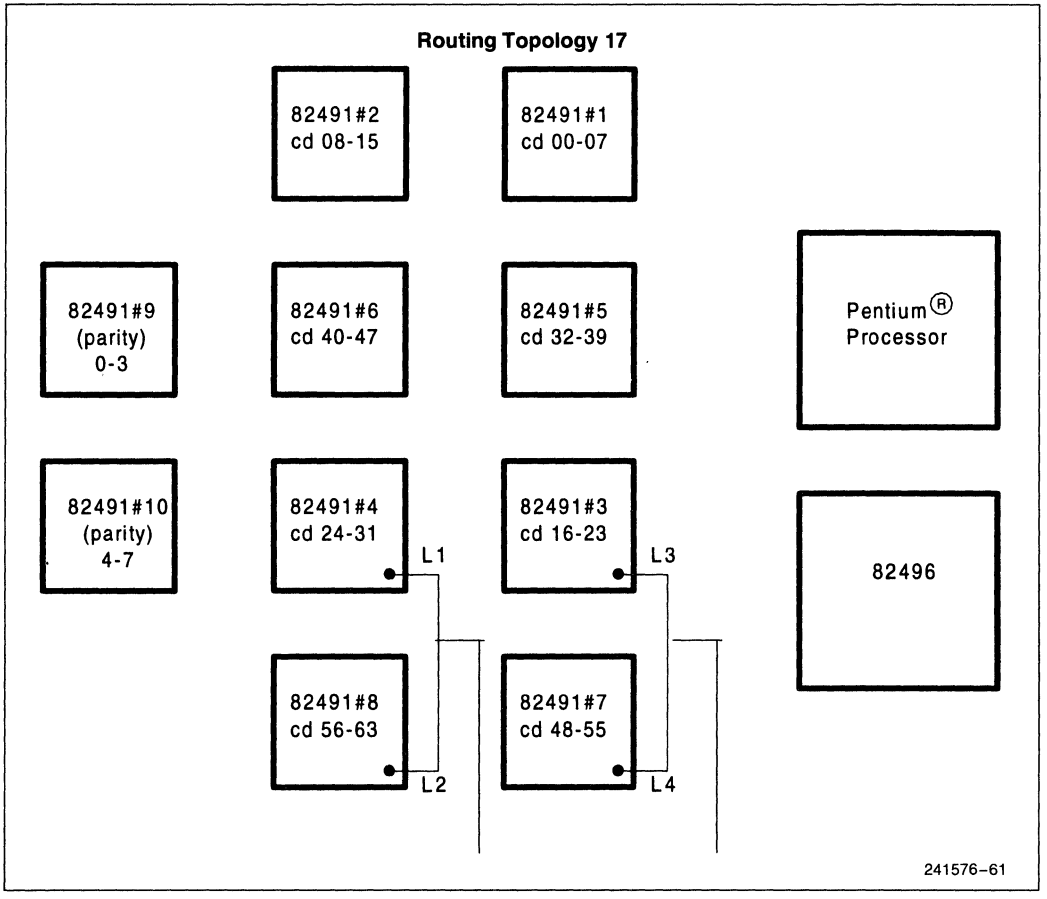


Figure 53. Topology 17

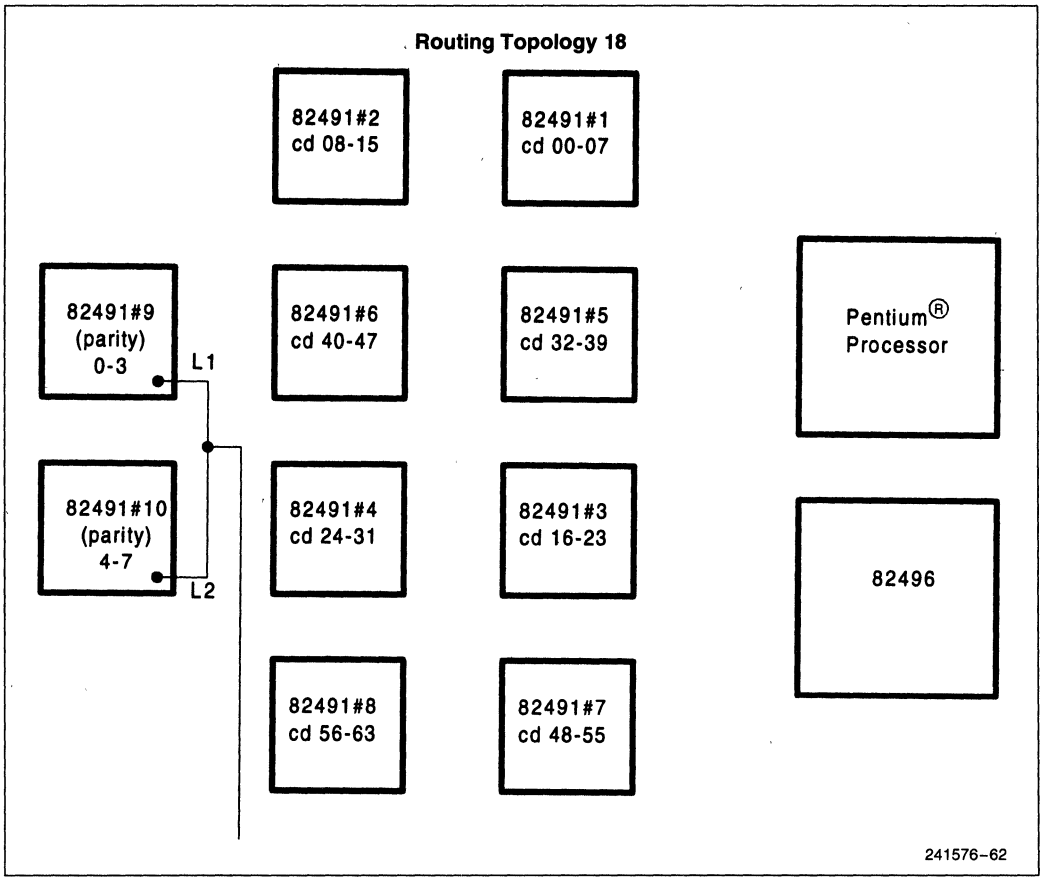


Figure 54. Topology 18

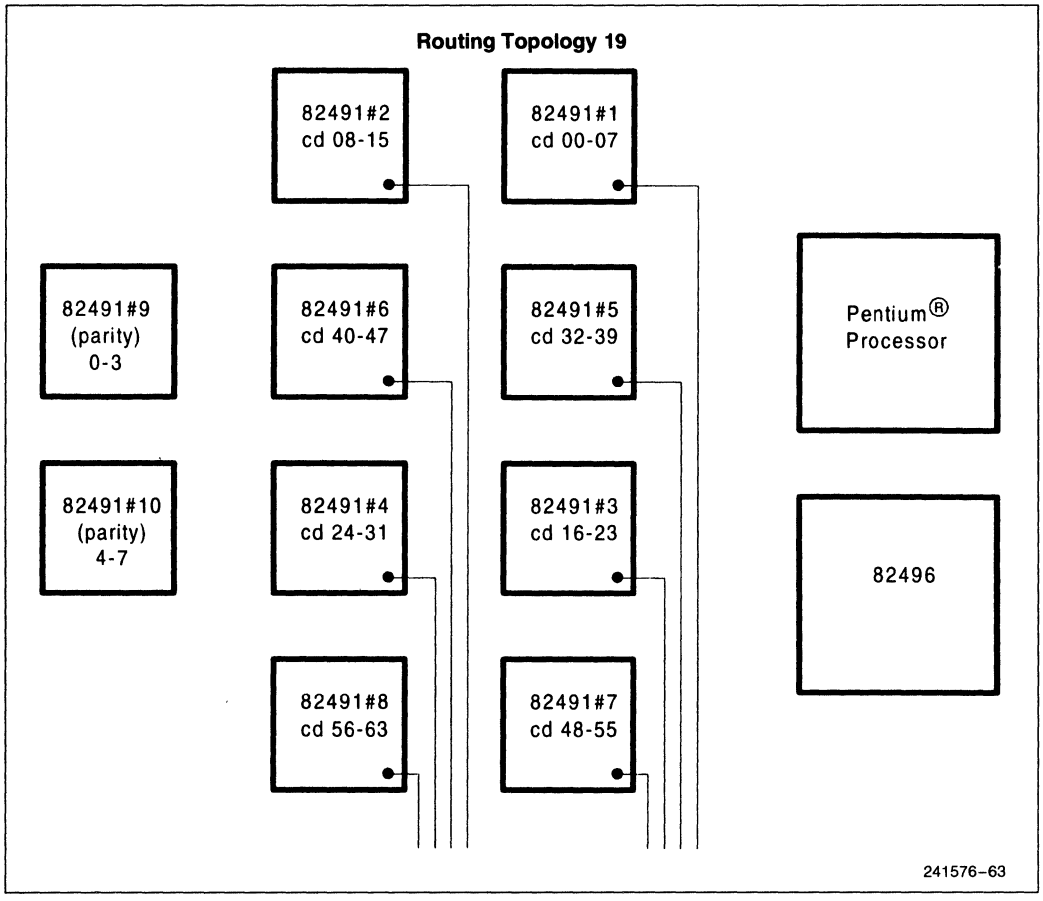


Figure 55. Topology 19

Figures 56 and 57 provide topologies for the non-parity configuration of the 256 Kbyte CPU-Cache Chip Set. Refer to Section 7.7.1 for more details on the non-parity configuration.

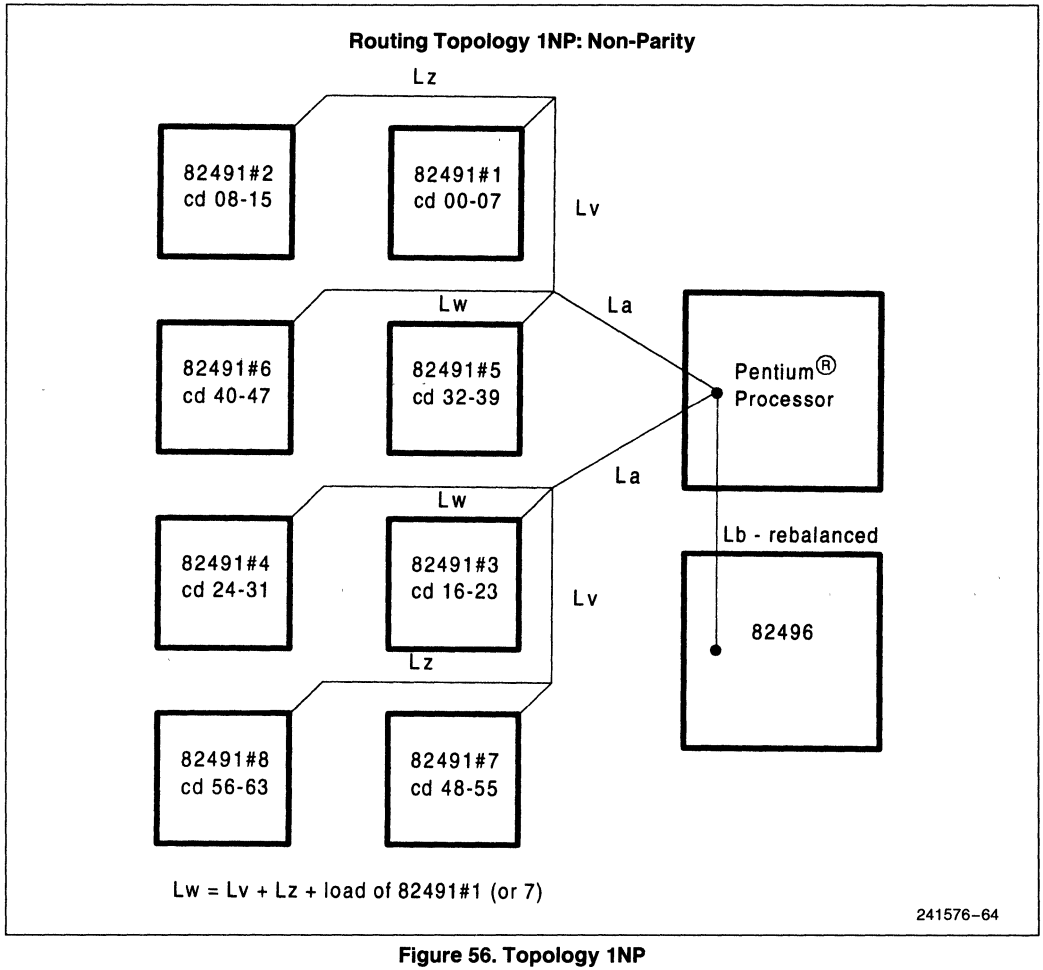


Figure 56. Topology 1NP

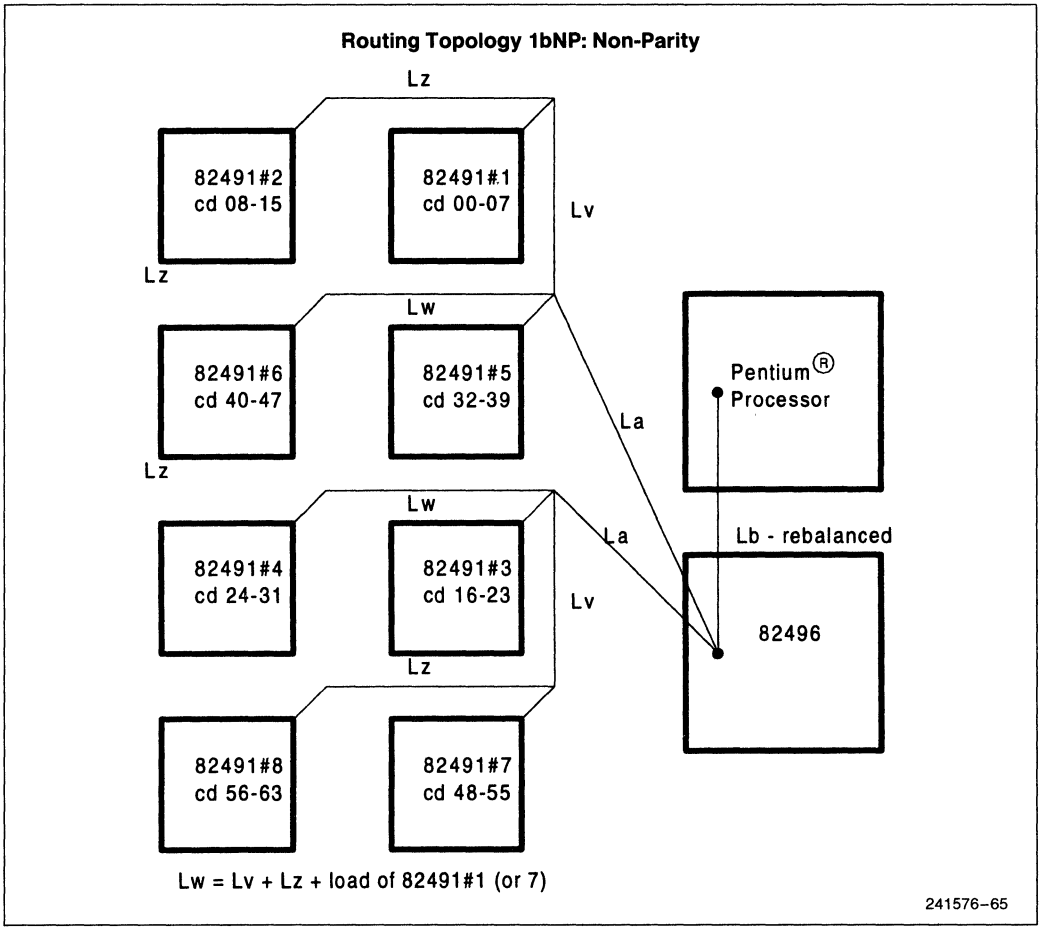


Figure 57. Topology 1bNP

7.4 Board/Trace Properties

Specific board and trace properties were assumed while performing the simulations to optimize the chip set layout. These properties were used as the specification or guideline the board manufacturer was to use in building boards. Figure 58 provides the board layer stackup.

Table 10 lists the minimum and maximum trace characteristics. These parameters along with the board material determine the spacing between layers and the total board thickness. See Table 11.

Only the inner layers of the board are impedance controlled. The top and bottom layers are not impedance controlled.

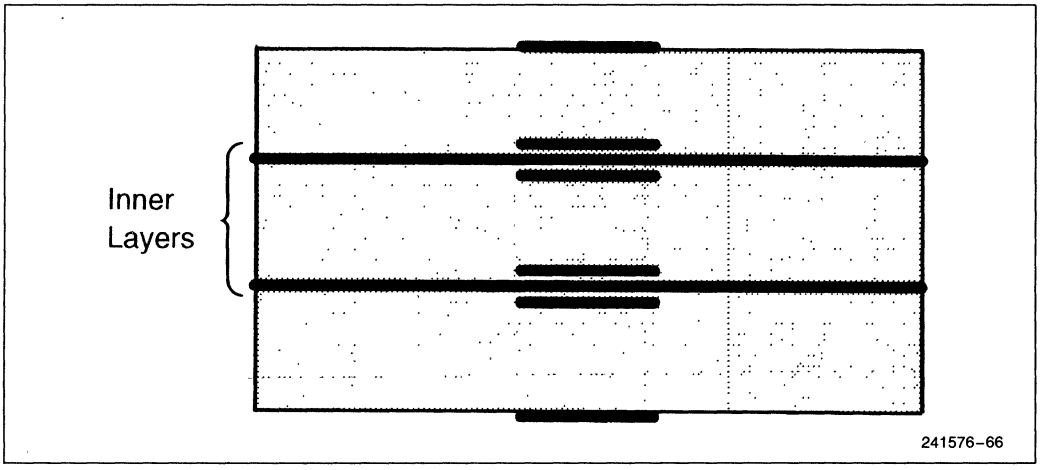


Figure 58. Board Layer Stackup

Table 10. Trace Characteristics

	4 Inner Layers	2 Outer Layers
Width/Space	5/5 Mils	8/8.5 Mils
Z_0	65Ω ± 10%	90Ω ± 20%
Velocity	1.85 to 2.41 ns/ft	1.35 to 2.05 ns/ft

Table 11. Other Printed Circuit Board Geometries

Via Pad	25 Mils
Via Hole	10 Mils
PGA Pad	55 Mils
PGA Hole	38 Mils
Layout Grid	5 Mils

7.5 Design Notes

The following design notes accompany this layout example:

1. The layout did not specifically address heat dissipation except to allow space for heat sinks to be attached. Please see the *Pentium® Processor User's Manual* for the devices' thermal specifications. The *Pentium Processor Thermal Design Guidelines* application note provides some examples of possible thermal solutions.
2. All fast-switching signals are routed near the power and ground planes on inner layers of the board to minimize EMI effects. However, two sets of signals are routed on the top layer of the board: BRDYC1#, and JTAG signals. BRDYC1# is routed on top to take advantage of the higher trace velocity there. JTAG signals are routed on the top layer because they are low-speed signals and will probably be re-routed by each customer to suit individual needs.

3. Resistor R1 (0) is used to set the Pentium processor configurable output buffers (A3–A20, ADS#, W/R#, and HITM#). When the resistor is included the buffers are set to the Extra Large size. When it is not included (BUSCHK# internally pulled high) the buffers are set to Large size. Intel currently recommends the large buffers be used for the 256K layout example. The 0Ω resistor should be designed into your design as Intel may change the recommended buffer size once silicon and the system design have been characterized.
4. The 82496 output buffers that drive the 82491 inputs must also be configured to be Large. This is done by driving 82496 CLDRV[BGT#] (pin N04) high during reset. 82496 and 82491 Memory Bus buffer sizes must be controlled by the Memory Bus Controller.
5. Series termination resistors were added to the nets PA17, PA18, PA19, and PA20 to control overshoot. A value of 24Ω is currently recommended, but that value may change when overshoot is measured on an actual board.

7.6 Explanation of Information Provided

The following sections outline the design files associated with the 256Kbyte CPU-Cache Chip Set design example that are available from Intel. These files are provided to simplify the task of porting the design example into a specific design. By using these files, designers may eliminate or minimize the amount of duplicate effort when using the design example as the basis for their design. The following items are available:

- Schematics
- I/O Model Files
- Board Files
- Bill of Materials
- Photoplot Log
- Netlist Report
- Placed Component Report
- Artwork for Each Board Layer
- Trace Segment Line Lengths

Hard copies of the schematics and trace segment line lengths are provided in the following sections. ASCII or soft copies of all the information are available from Intel by requesting order number 241663, *AP-481 Design Diskettes.*

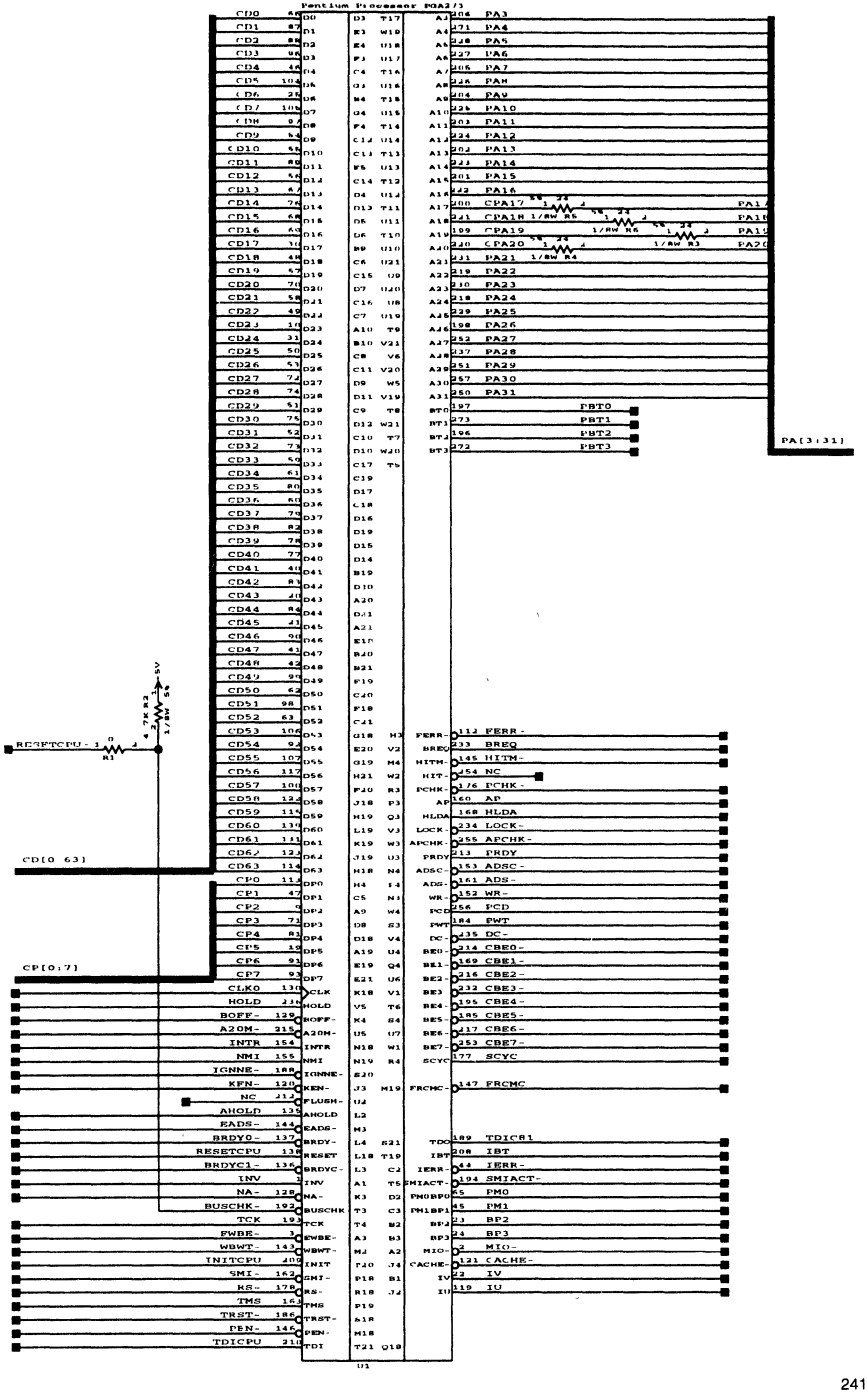
7.6.1 SCHEMATICS

Schematics for the 256 Kbyte CPU-Cache Chip Set design example were created using ViewLogics's Workview V4.1. The schematics are 14 pages long. Both the Workview and the postscript files are available from Intel as described above.

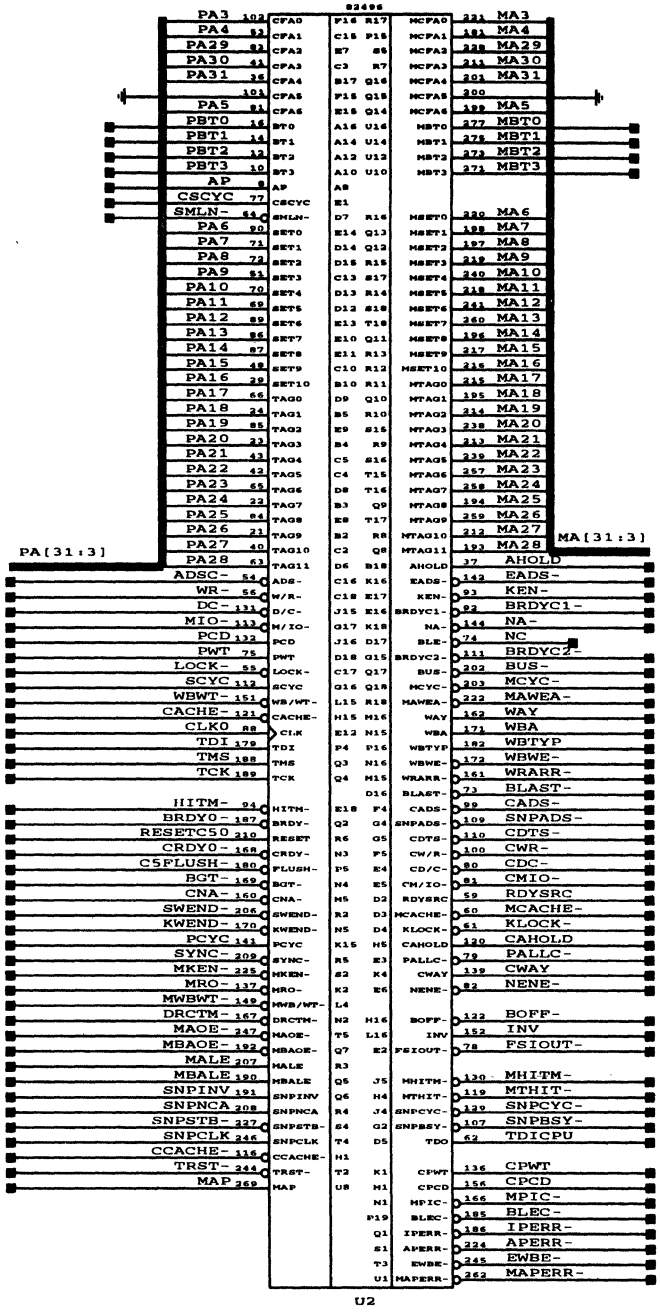
Pentium® Processor/82496/82491 256 KB OPTIMIZED INTERFACE
1-20-92
REV 2.1

NOTES: (Unless Otherwise Specified)
1. Capacitor values are in microfarads.
2. Resistor values are in ohms.
3. An "-" following a signal name denotes negation.
4. VCC = +5V.
5. This document also exists on electronic media.

intel®

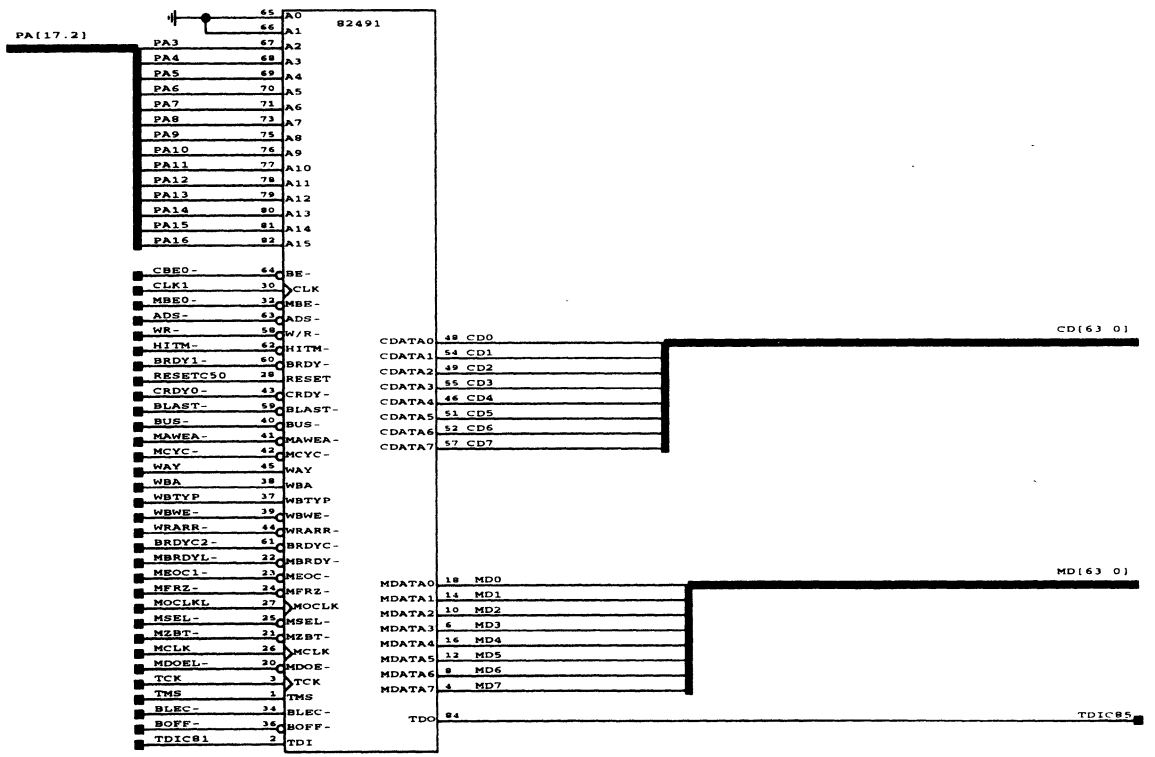

Pentium Processor PGA2/3
CD[0 63]
CP[0:7]
PA[3:31]
RESETCPU
5V
PBT0
PBT1
PBT2
PBT3
FERR-
BREQ
HITM-
NC
PCHK-
AP
HLDA
LOCK-
APCHK-
PRDY
ADSC-
ADS-
WR-
PCD
PWT
DC-
CBE0-
CBE1-
CBE2-
CBE3-
CBE4-
CBE5-
CBE6-
CBE7-
SCYC
FRCMC-
TDIC81
IBT
IERR-
SMIACT-
PM0
PM1
BP2
BP3
MIO-
CACHE-
IV
IU
CLK
HOLD
BOFF-
A20M-
INTR
NMI
IGNNE-
KEN-
FLUSH-
AHOLD
EADS-
BRDY0-
RESETCPU
BRDYC1-
INV
NA-
BUSCHK-
TCK
FWBE-
WBWT-
INITCPU
SMI-
RS-
TMS
TRST-
PEN-
TDICPU

241576-67


82496
U2
PA[31:3]
MA[31:3]
PA3 103
PA4 83
PA29 81
PA30 41
PA31 39
101
PA5 81
PBT0 16
PBT1 14
PBT2 12
PBT3 10
AP 8
CSCYC 77
SMLN- 64
PA6 80
PA7 71
PA8 72
PA9 81
PA10 70
PA11 68
PA12 89
PA13 86
PA14 87
PA15 48
PA16 29
PA17 66
PA18 24
PA19 85
PA20 23
PA21 43
PA22 42
PA23 65
PA24 22
PA25 84
PA26 21
PA27 40
PA28 63
ADSC- 54
WR- 56
DC- 131
MIO- 133
PCD 132
PWT 75
LOCK- 55
SCYC 112
WBWT- 151
CACHE- 121
CLK0 88
TDI 179
TMS 188
TCK 189
HITM- 94
BRDY0- 187
RESETC50 210
CRDY0- 168
C5FLUSH- 180
BGT- 169
CNA- 160
SWEND- 206
KWEND- 170
PCYC 141
SYNC- 209
MKEN- 225
MRO- 137
MWBWT- 149
DRCTM- 167
MAOE- 247
MBAOE- 192
MALE 207
MBALE 190
SNPINV 191
SNPNCA 208
SNPSTB- 227
SNPCLK 246
CCACHE- 116
TRST- 244
MAP 269
CFA0 CFA1 CFA2 CFA3 CFA4 CFA5 CFA6
BT0 BT1 BT2 BT3
AP CSCYC SMLN-
SET0 SET1 SET2 SET3 SET4 SET5 SET6 SET7 SET8 SET9 SET10
TAG0 TAG1 TAG2 TAG3 TAG4 TAG5 TAG6 TAG7 TAG8 TAG9 TAG10 TAG11
ADS- W/R- D/C- M/IO- PCD PWT LOCK- SCYC WB/WT- CACHE- CLK TDI TMS TCK
HITM- BRDY- RESET CRDY- FLUSH- BGT- CNA- SWEND- KWEND- PCYC SYNC- MKEN- MRO- MWB/WT- DRCTM- MAOE- MBAOE- MALE MBALE SNPINV SNPNCA SNPSTB- SNPCLK CCACHE- TRST- MAP
F16 R17 C15 P15 E7 B6 C3 R7 B17 Q16 F15 Q15 E15 Q14
A15 U16 A14 U14 A12 U12 A10 U10 A8
E1 D7 R16 E14 Q13 D14 Q12 D15 R15 C13 S17 D13 R14 D12 S18 E13 T18 E10 Q11 E11 R13 C10 R12
B10 R11 D9 Q10 B5 R10 E9 S15 B4 R9 C5 S16 C4 T15 D8 T16 B3 Q9 E8 T17 B2 R8 C2 Q8
D6 B18 C16 K16 C18 E17 J15 E16 G17 K18 J16 D17 D18 G15 C17 Q17 G16 Q18 L15 R18 H15 M16 E12 N15
P4 P16 Q3 N16 Q4 M15 D16 E18 F4 Q2 G4 R6 G5 N3 P5 P5 E4 N4 E5 M5 D2 R2 D3 N5 D4 K15 H5 R5 E3 S2 K4 K2 E6 L4
N2 H16 T5 L16 Q7 E2 R3 Q5 J5 Q6 H4 R4 J4 S4 G2 T4 D5 H1
T2 K1 U8 M1 N1 P19 Q1 S1 T3 U1
MCFA0 MCFA1 MCFA2 MCFA3 MCFA4 MCFA5 MCFA6
MBT0 MBT1 MBT2 MBT3
MSET0 MSET1 MSET2 MSET3 MSET4 MSET5 MSET6 MSET7 MSET8 MSET9 MSET10
MTAG0 MTAG1 MTAG2 MTAG3 MTAG4 MTAG5 MTAG6 MTAG7 MTAG8 MTAG9 MTAG10 MTAG11
AHOLD EADS- KEN- BRDYC1- NA- BLE- BRDYC2- BUS- MCYC- MAWEA- WAY WBA WBTYP WBWE- WRARR- BLAST- CADS- SNPADS- CDTS- CWR- CDC- CM/IO- RDYSRC MCACHE- KLOCK- CAHOLD PALLC- CWAY NENE-
BOFF- INV FSIOUT-
MHITM- MTHIT- SNPCYC- SNPBSY- TDO
CPWT CPCD MPIC- BLEC- IPERR- APERR- EWBE- MAPERR-
221 MA3
181 MA4
228 MA29
211 MA30
201 MA31
200
199 MA5
277 MBT0
278 MBT1
273 MBT2
271 MBT3
220 MA6
188 MA7
197 MA8
218 MA9
240 MA10
218 MA11
241 MA12
260 MA13
196 MA14
217 MA15
216 MA16
215 MA17
195 MA18
214 MA19
238 MA20
213 MA21
239 MA22
257 MA23
258 MA24
194 MA25
259 MA26
212 MA27
193 MA28
37 AHOLD
143 EADS-
93 KEN-
92 BRDYC1-
144 NA-
74 NC
111 BRDYC2-
202 BUS-
203 MCYC-
222 MAWEA-
162 WAY
171 WBA
182 WBTYP
172 WBWE-
161 WRARR-
73 BLAST-
99 CADS-
109 SNPADS-
110 CDTS-
100 CWR-
80 CDC-
81 CMIO-
59 RDYSRC
60 MCACHE-
61 KLOCK-
139 CAHOLD
79 PALLC-
139 CWAY
82 NENE-
122 BOFF-
152 INV
78 FSIOUT-
120 MHITM-
119 MTHIT-
129 SNPCYC-
107 SNPBSY-
62 TDICPU
136 CPWT
156 CPCD
166 MPIC-
185 BLEC-
186 IPERR-
224 APERR-
245 EWBE-
262 MAPERR-

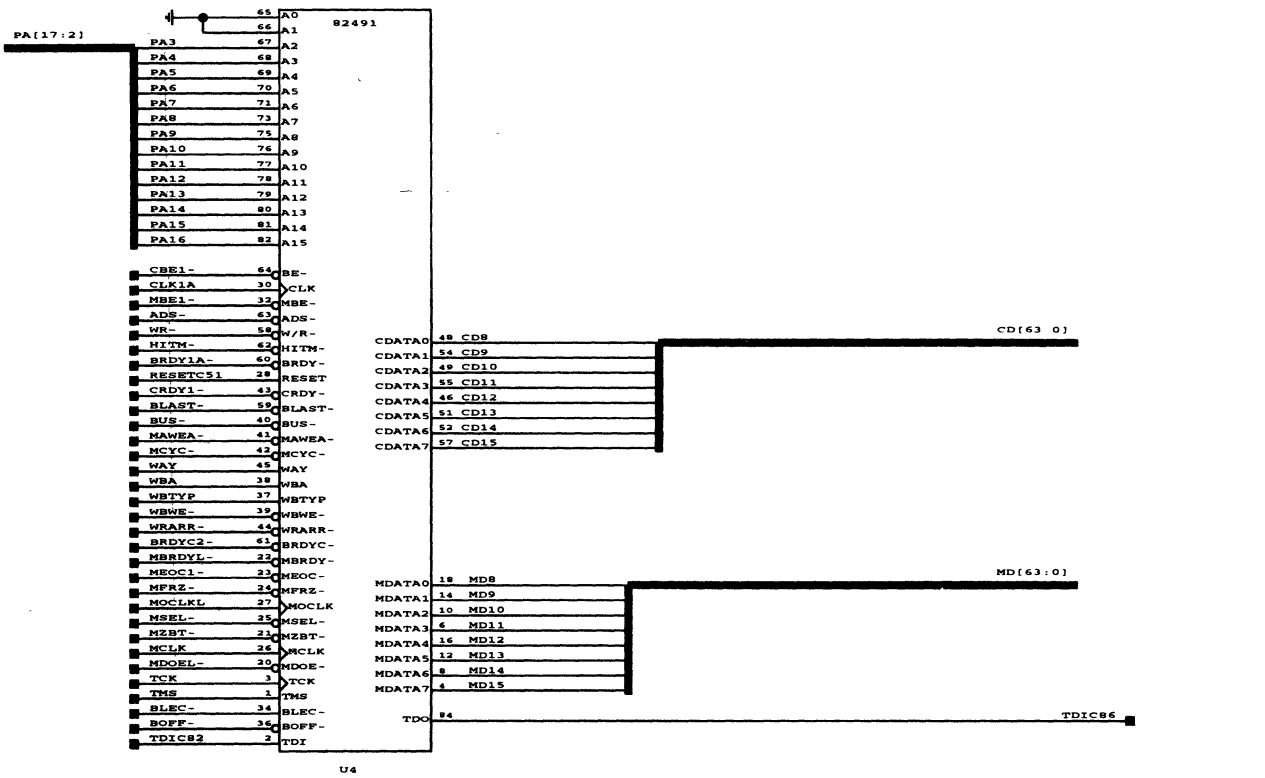
241576-68

82491 Byte 0



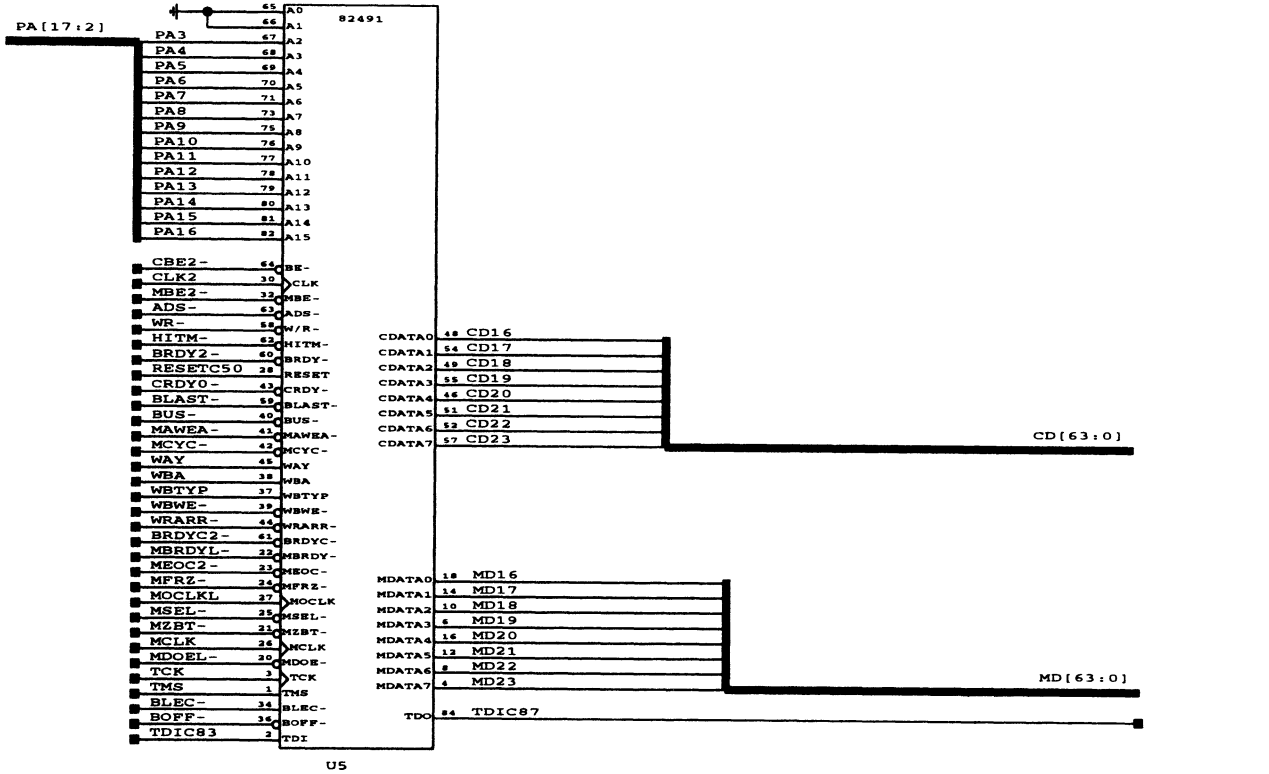
241576-69

82491 Byte 1



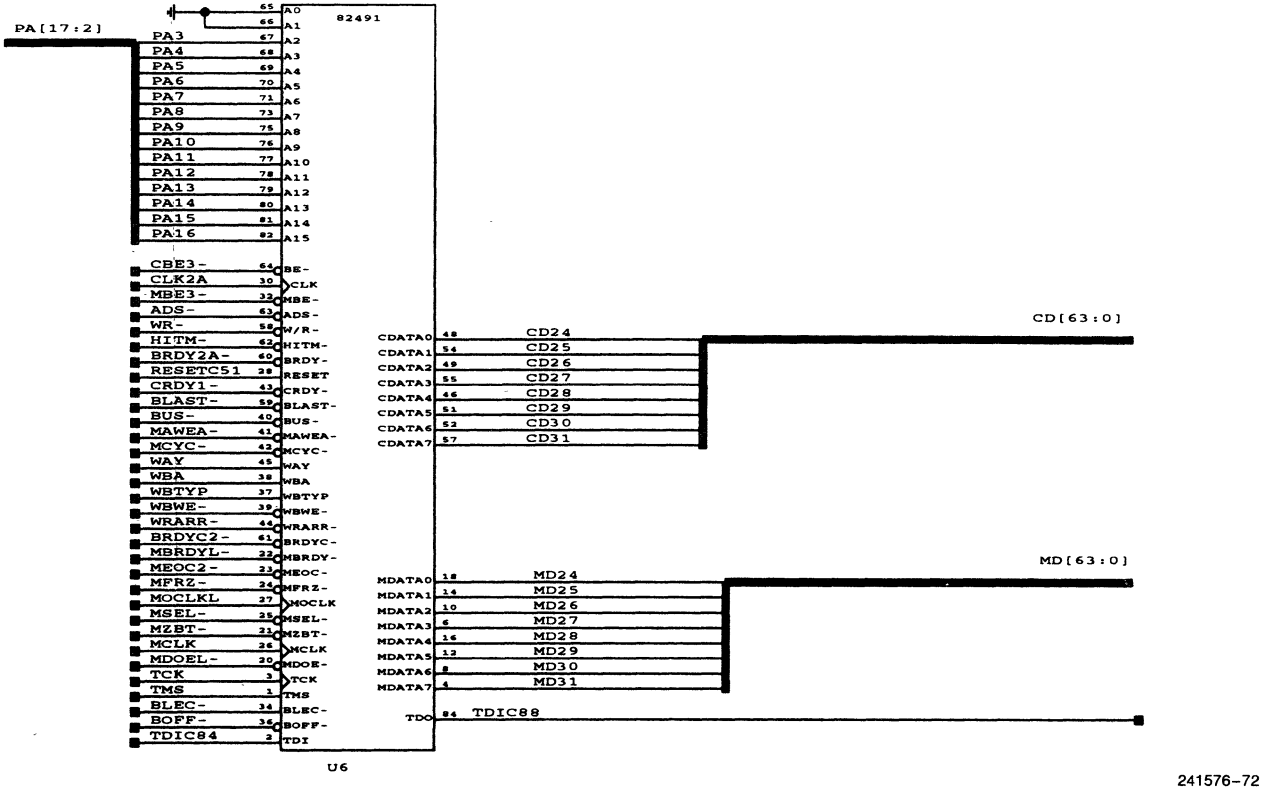
241576-70

82491 Byte 2



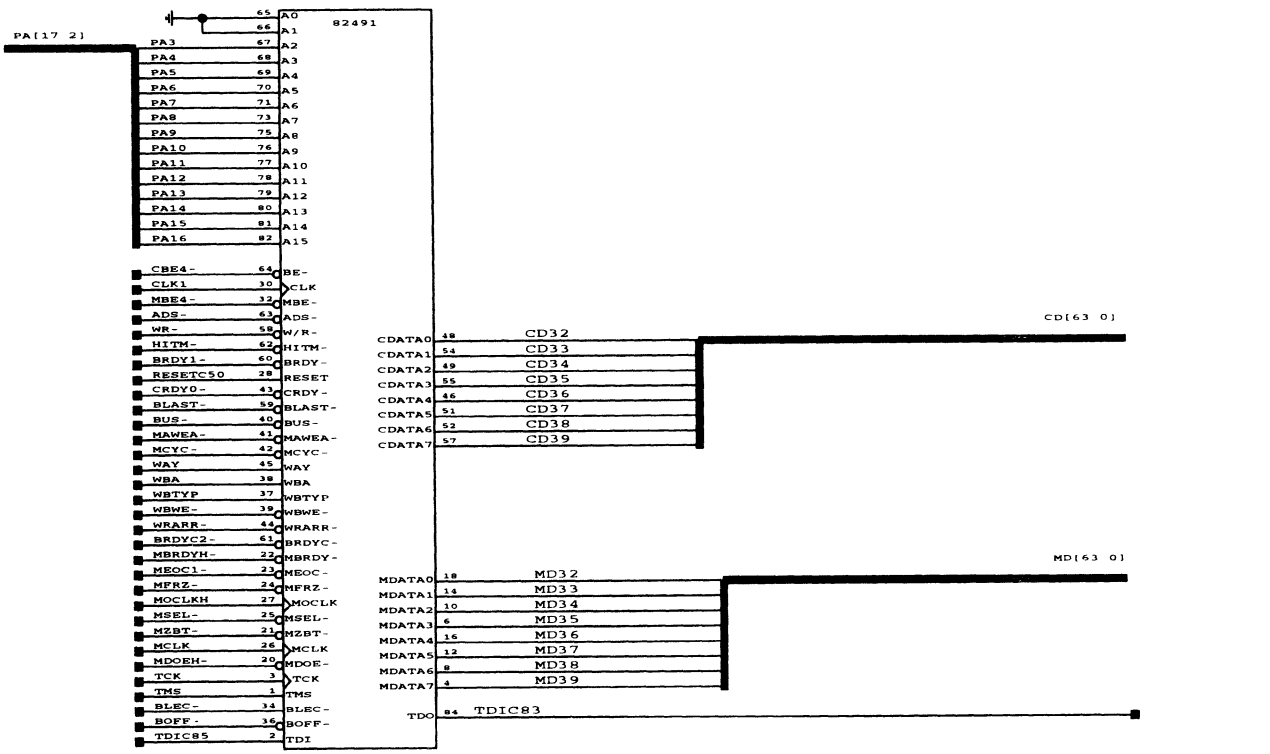
241576-71

82491 Byte 3



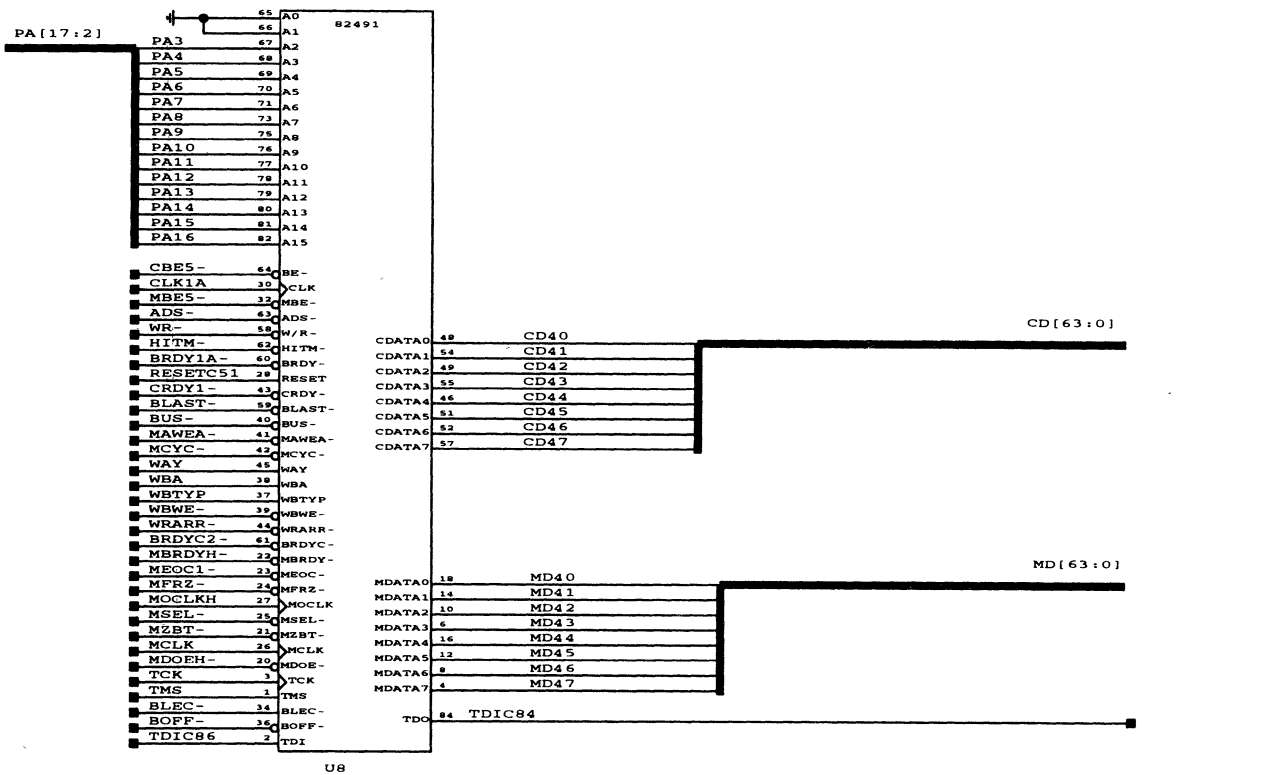
241576-72

82491 Byte 4



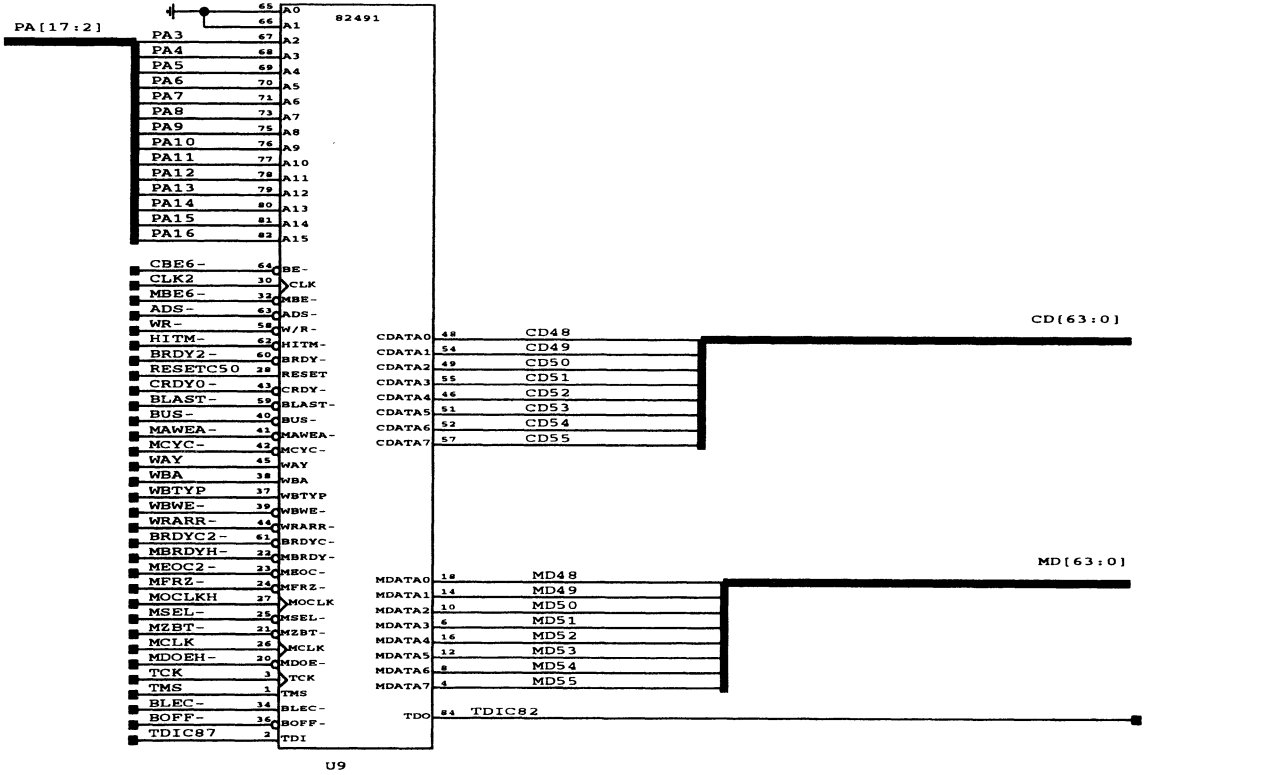
241576-73

82491 Byte 5



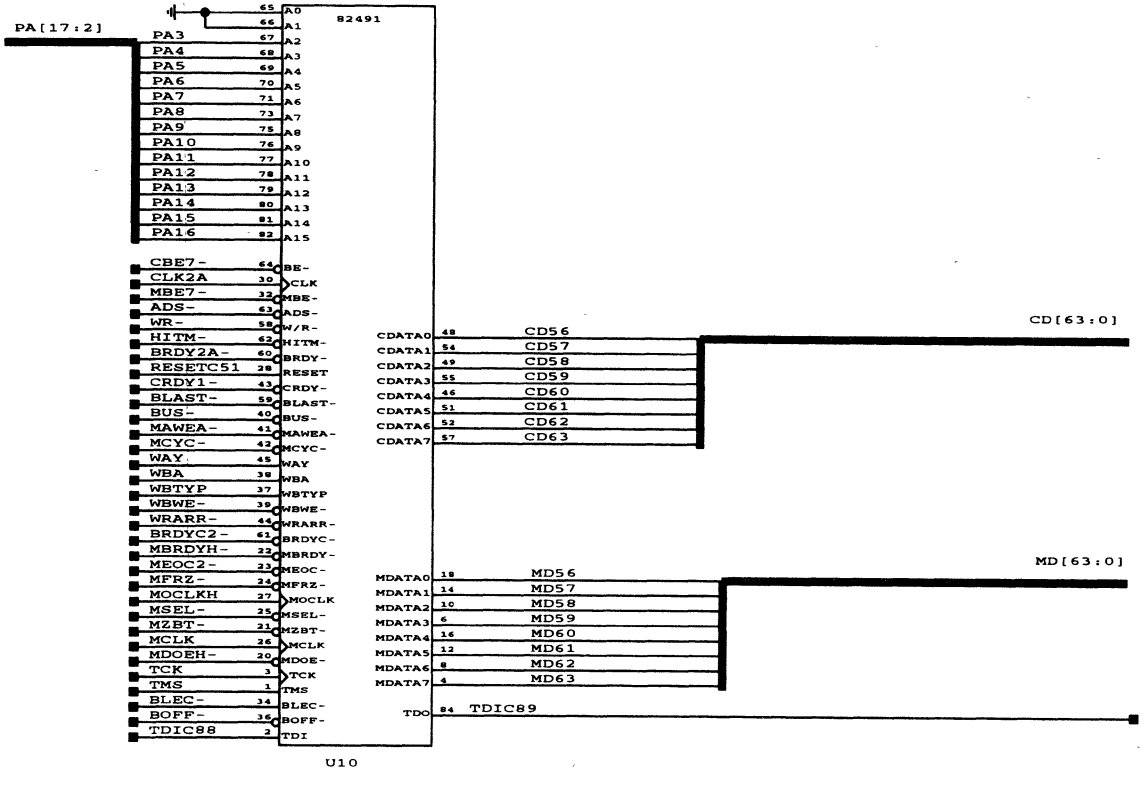
241576-74

82491 Byte 6



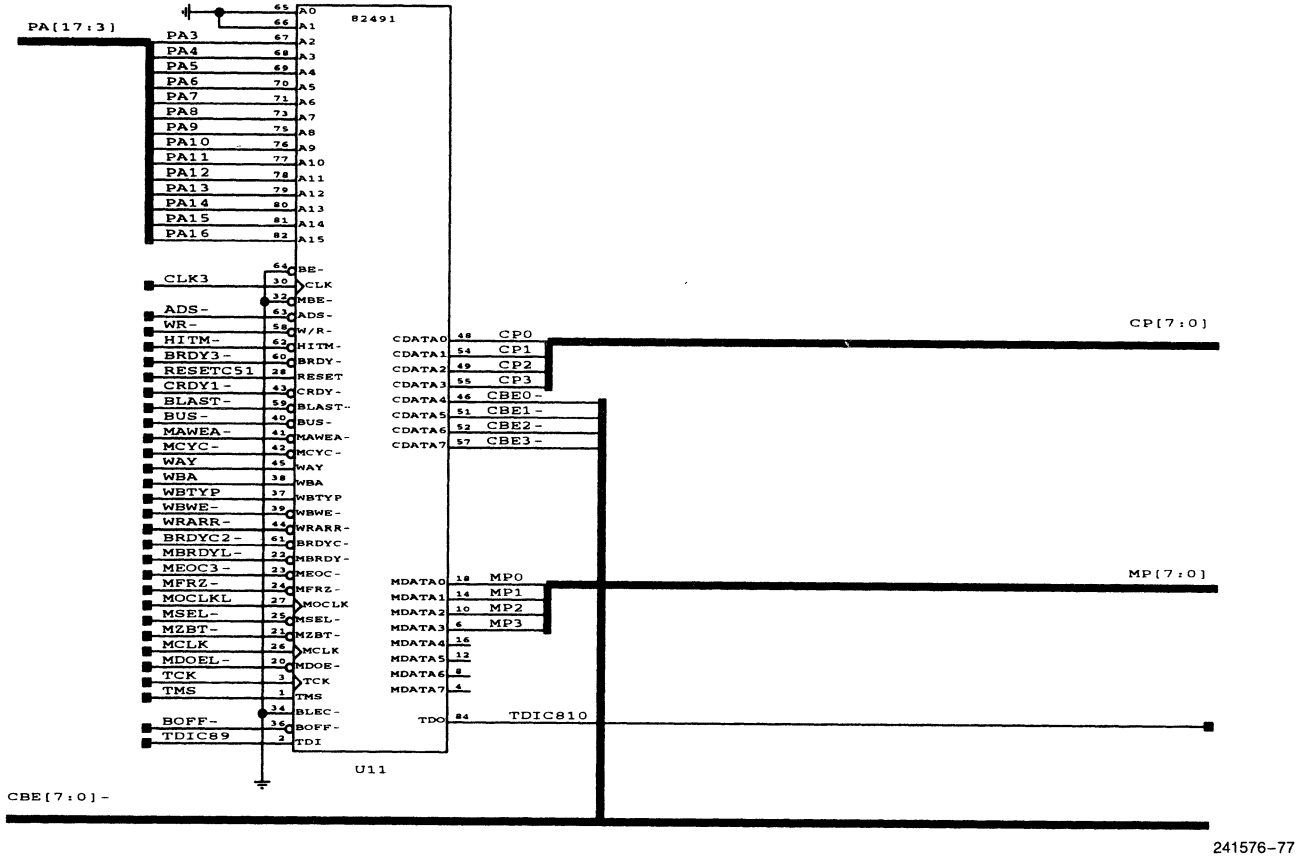
241576-75

82491 Byte 7



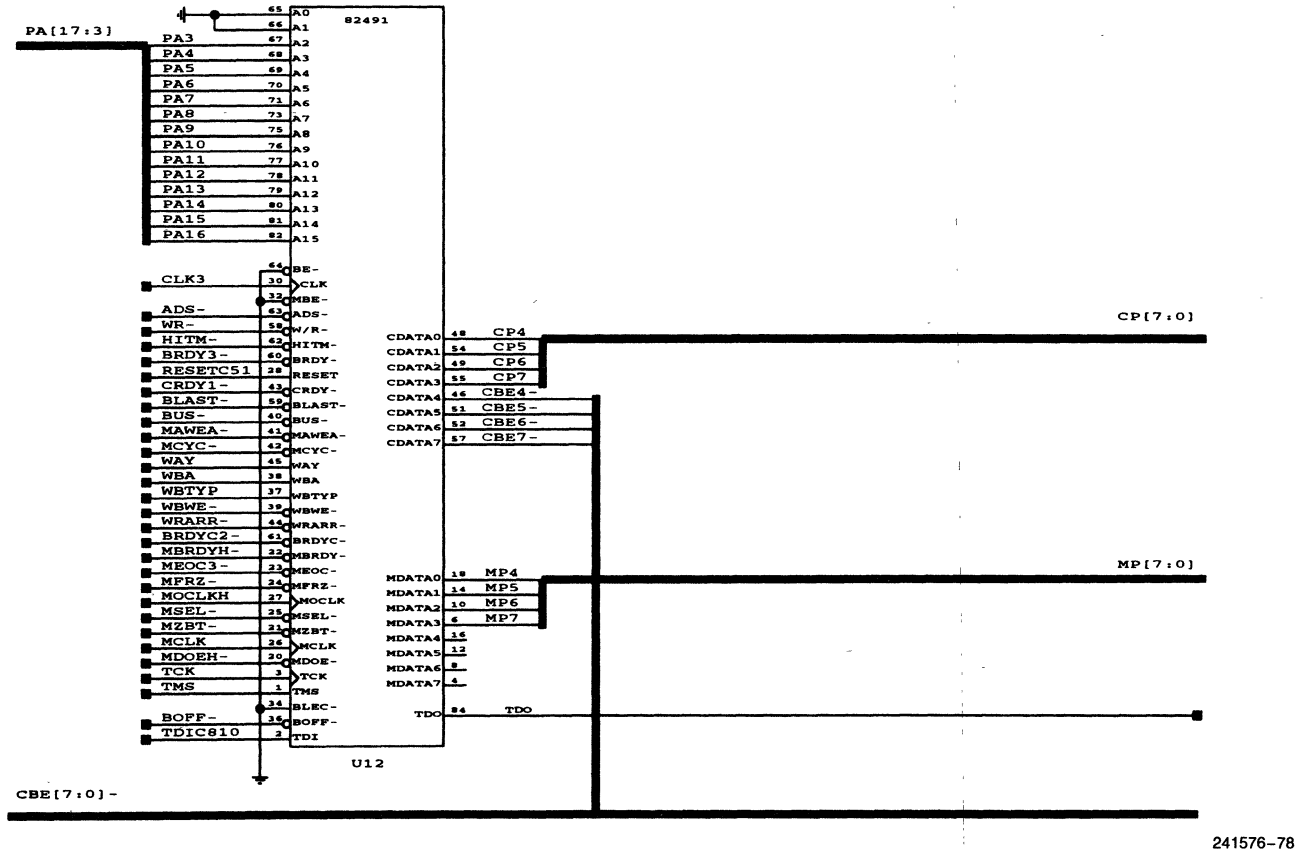
241576-76

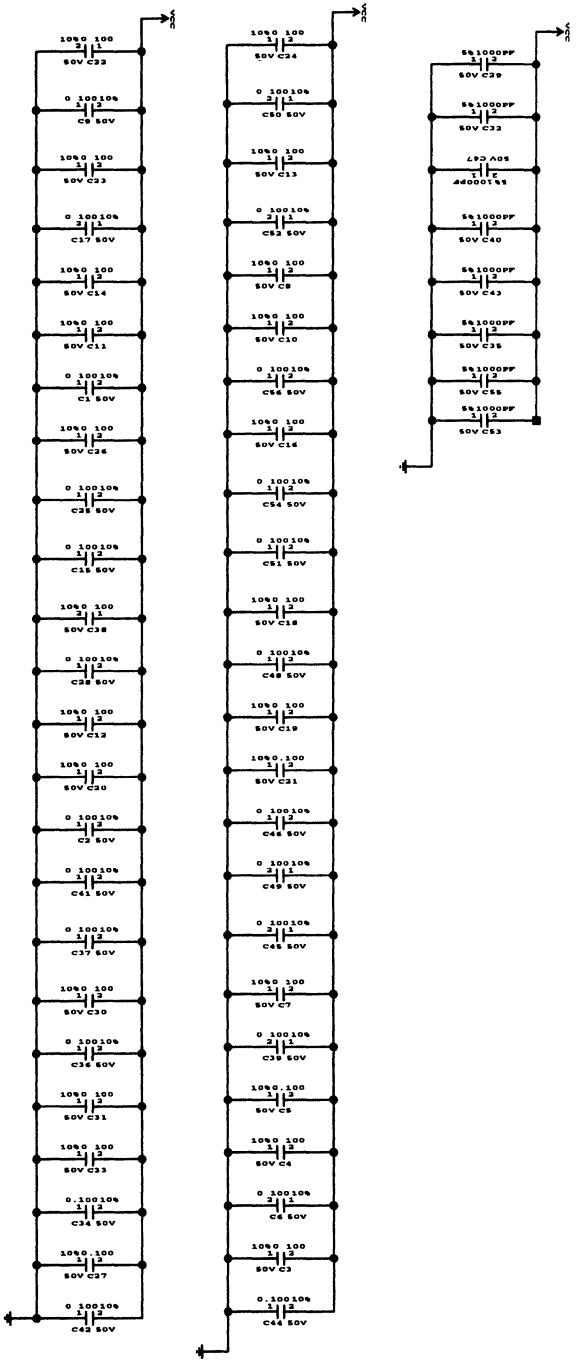
82491 Parity 0-3



241576-77

82491 Parity 4-7




241576-79

7.6.2 I/O MODEL FILES

All electrical I/O simulations were performed using TLC V4.1.13 from Quad Design Technology, Inc. The simulations were performed at the fast and slow corners to verify all signal quality and flight time specifications are met. The files used for these simulations are available from Intel as described above. These files include the topology, model, and control files needed to run the simulations for all nets in the optimized interface.

7.6.3 BOARD FILES

The board files for the design example were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel as described above. These files may be used to import the design example into a specific system design. Note: some changes to the layout and nets may be necessary to complete importing these files into a specific system design.

7.6.4 BILL OF MATERIALS

The bill of materials file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

7.6.5 PHOTOPLOT LOG

The photoplot log file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

7.6.6 NETLIST REPORT

The netlist report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

7.6.7 PLACED COMPONENT REPORT

The placed component report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

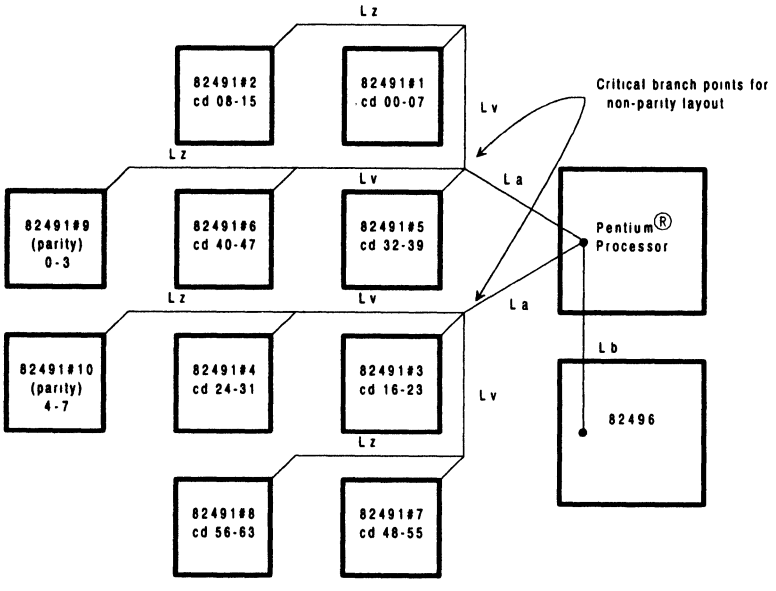
7.6.8 ARTWORK FOR EACH BOARD LAYER

The artwork for the six board layers were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel in a Gerber format as described above.

7.6.9 TRACE SEGMENT LINE LENGTHS

Sections 7.6.9.1 to 7.6.9.10 list the segment line lengths for each net of the optimized interface. All lengths are provide in mils (1/1000 inch). The stubs listed in the following tables are associated with the pin escapes required for the 82491s.

7.6.9.1 Low Addresses (Topology 1)



NET	PP-CS#5	PP-CS#3	PP-CC	CS#5-CS#1	CS#1-CS#2	CS#5-CS#6	CS#6-CS#9
PA3	1761.4	1768.6	5513	1184.6	936.5	1186.5	936.5
PA4	1259.6	1278.4	4673.2	1113.9	936.5	1113.6	936.5
PA5	1553.6	1573.9	5193.7	1164.6	936.5	1176.5	936.5
PA6	1543.1	1540	5123.2	1152.9	936.5	1156.5	936.5
PA7	1691.9	1692.4	5367.6	1152.9	936.5	1156.5	936.5
PA8	1590.7	1590.2	5243.7	1215.3	936.5	1216.5	936.5
PA9	1660.7	1663.1	5365.2	1210.5	936.5	1206.5	936.5
PA10	1543.6	1543.1	5233	1264.1	936.5	1266.5	936.5
PA11	1701.9	1695.5	5474.2	1264.1	936.5	1266.5	936.5
PA12	1586.5	1594.3	5324.1	1292.4	936.5	1296.5	936.5
PA13	1741.9	1745.5	5497.9	1295.3	936.5	1296.5	936.5
PA14	1633.6	1633.1	5403.8	1317.8	936.5	1316.5	936.5
PA15	1791.9	1792.6	5500.4	1314.8	936.5	1316.5	936.5
PA16	1623.6	1619	5359.1	1288.0	936.5	1286.5	936.5

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

NET	CS#3-CS#4	CS#4-CS#10	CS#3-CS#7	CS#7-CS#8	Stubs
PA3 (cont)	1181.4	936.5	1184.6	936.5	135.3–135.3
PA4 (cont)	1111.4	936.5	1113.9	936.5	75.0–75.0
PA5 (cont)	1166.5	936.5	1170.5	936.5	135.3–135.3
PA6 (cont)	1156.5	936.5	1158.8	936.5	75.0–75.0
PA7 (cont)	1156.5	936.5	1158.8	936.5	135.3–135.3
PA8 (cont)	1216.5	936.5	1212.4	936.5	135.3–135.3
PA9 (cont)	1206.5	936.5	1207.6	936.5	135.3–135.3
PA10 (cont)	1266.5	936.5	1261.2	936.5	75.0–75.0
PA11 (cont)	1266.5	936.5	1261.2	936.5	135.3–135.3
PA12 (cont)	1296.5	936.5	1292.4	936.5	75.0–75.0
PA13 (cont)	1296.5	936.5	1295.3	936.5	135.3–135.3
PA14 (cont)	1316.5	936.5	1317.8	936.5	75.0–75.0
PA15 (cont)	1316.5	936.5	1314.8	936.5	135.3–135.3
PA16 (cont)	1286.5	936.5	1288.0	936.5	75.0–75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.2 High Addresses (Topology 4, Point-to-Point)

Routing Topology 4

241576-81

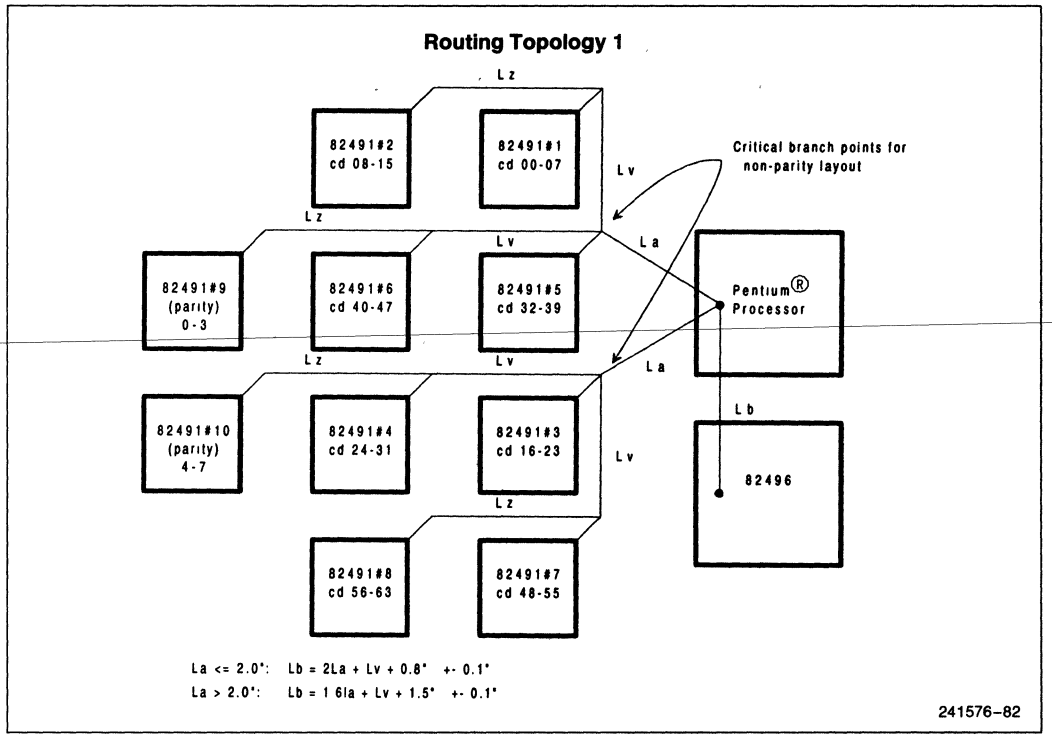
NET	PP-CC
PA17*	689.3 + 2841.7
PA18*	647.6 + 3683.1
PA19*	731.0 + 2763.3
PA20*	601.7 + 718.6
PA21	3376.6
PA22	4347.5
PA23	3111.5
PA24	4661.2
PA25	3029.7

NET	PP-CC
PA26	4495.1
PA27	3885.5
PA28	4689.3
PA29	3136.1
PA30	5177.1
PA31	2518.5
PA32	3604.4
PA33	2686.8
PA34	3952
PA35	3189.9

NET	PP-CC
ADSC#	3791.7
AP	4531.6
CACHE#	3719.8
DC#	3613.1
LOCK#	4710.4
MIO#	5062
PCD	3461.3
PWT	4295.6
SCYC	3848.2
WBWT#	3493.3

***NOTE:**
24Ω resistor included on PA[17–20].
Lengths are Pentium processor-resistor + resistor-82486, respectively.
PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.3 Pentium® Processor Control (Topology 1)

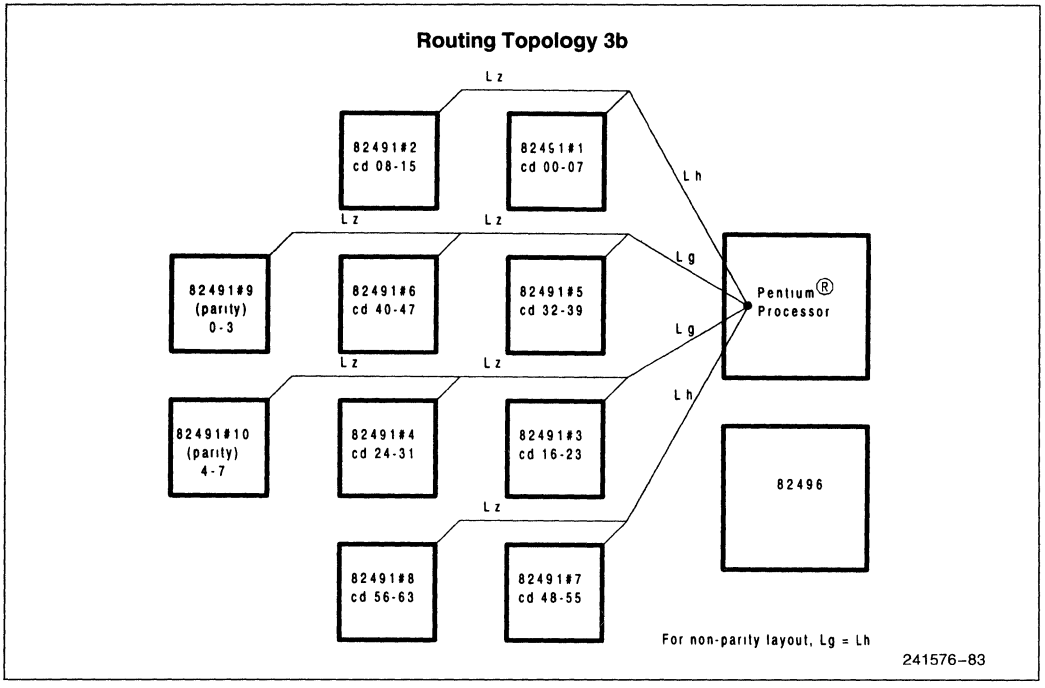


NET	PP–CS#5	PP–CS#3	PP–CC	CS#5–CS#1	CS#1–CS#2	CS#5–CS#6	CS#6–CS#9
HITM#	4205.2	4199.7	9147.1	1141.3	936.5	1146.5	936.5
WR#	4091.1	4088.2	9149.4	1193.0	936.5	1192.1	936.5

NET	CS#3–CS#4	CS#4–CS#10	CS#3–CS#7	CS#7–CS#8	Stubs
HITM# (cont)	1146.2	944.8	1146.6	944.8	95.3–95.3
WR# (cont)	1196.7	953.1	1193.0	953.1	95.3–95.3

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.4 Pentium® Processor Control (Topology 3b No 82496)

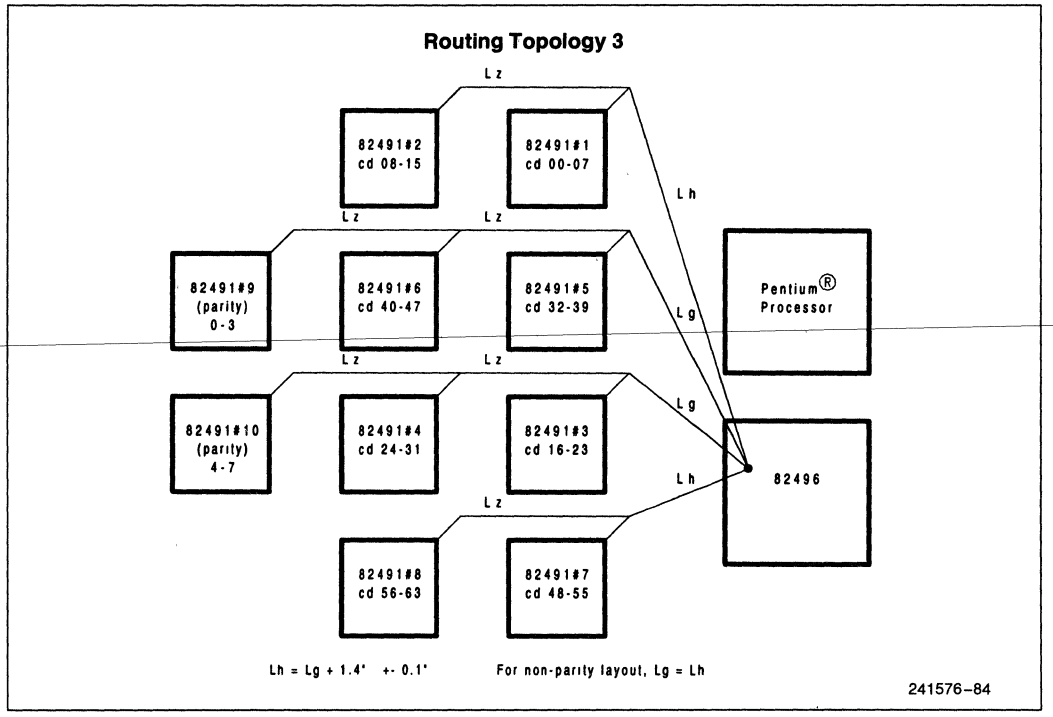


NET	PP–CS#1	PP–CS#5	PP–CS#3	PP–CS#7	CS#1–CS#2	CS#5–CS#6	CS#6–CS#9
ADS#	5113.0	3728.2	3738.5	5102.0	936.5	964.8	983.8

NET	CS#3–CS#4	CS#4–CS#10	CS#7–CS#8	Stubs
ADS# (cont)	936.5	936.5	936.5	75.0–75.0

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.5 82496 Control (Topology 3)

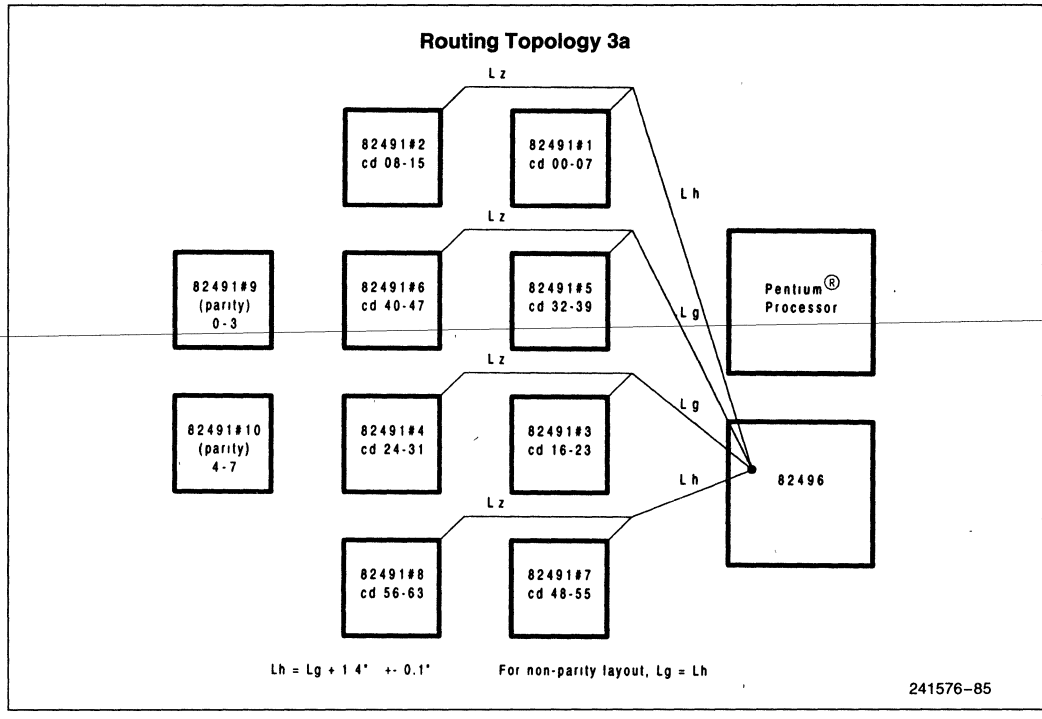


NET	CC-CS#1	CC-CS#5	CC-CS#3	CC-CS#7	CS#1-CS#2	CS#5-CS#6
BLAST#	4222.1	2820.0	2819.8	4219.2	986.7	1034.5
BRDYC2#	4186.4	2802.4	2775.0	4171.6	969.7	1037.4
BUS#	4607.6	3215.7	3210.0	4611.3	936.5	936.5
MAWEA#	4476.4	3058.9	3088.8	4481.3	936.5	936.5
MCYC#	4913.9	3507.0	3523.8	4921.5	936.5	961.4
WBA	5114.2	3747.8	3719.5	5119.3	936.5	936.5
WBTYP	4365.4	2986.2	2973.5	4376.0	936.5	936.5
WBWE#	4502.4	3109.1	3113.0	4511.7	936.5	936.5
WRARR#	4198.9	2735.0	2802.1	4199.8	936.5	969.7
WAY	4818.9	3348.4	3417.4	4816.3	936.5	936.5

NET	CS#6–CS#9	CS#3–CS#4	CS#4–CS#10	CS#7–CS#8	Stubs
BLAST# (cont)	965.5	936.5	936.5	936.5	85.0–85.0
BRDYC2# (cont)	965.5	936.5	936.5	936.5	85.0–85.0
BUS# (cont)	936.5	936.5	936.5	936.5	75.0–75.0
MAWEA# (cont)	936.5	936.5	936.5	936.5	135.3–135.3
MCYC# (cont)	936.5	936.5	936.5	936.5	75.0–75.0
WBA (cont)	936.5	936.5	936.5	936.5	75.0–75.0
WBTYP (cont)	936.5	936.5	936.5	936.5	135.3–135.3
WBWE# (cont)	936.5	936.5	936.5	936.5	135.3–135.3
WRARR# (cont)	936.5	936.5	936.5	936.5	135.3–135.3
WAY (cont)	936.5	936.5	936.5	936.5	75.0–75.0

PP = Pentium® processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.6 82496 Control (Topology 3a Not Connected to Parity 82491's)

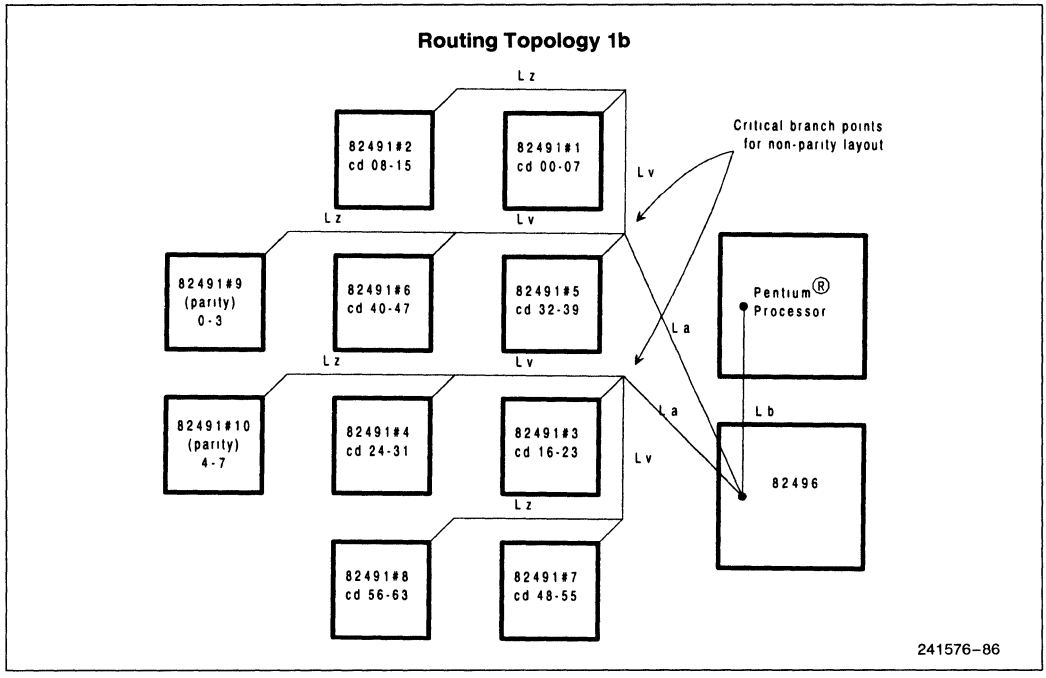


NET	CC-CS#1	CC-CS#5	CC-CS#3	CC-CS#7	CS#1-CS#2	CS#5-CS#6
BLEC#	4222.7	4219.0	4205.8	4206.4	936.5	936.5

NET	CS#6-CS#9	CS#3-CS#4	CS#4-CS#10	CS#7-CS#8	Stubs
BLEC# (cont)	n/a	936.5	n/a	936.5	75.0-75.0

PP = Pentium® processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.7 82496 Control (Topology 1b Pentium® Processor and 82496 Switch Positions)

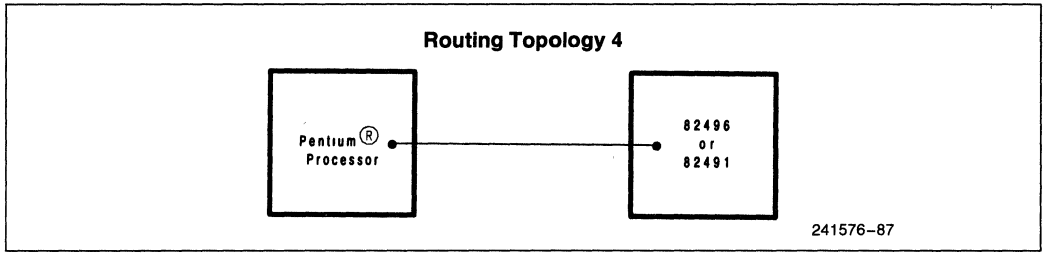


NET	CC-CS#5	CC-CS#3	CC-PP	CS#5-CS#1	CS#1-CS#2	CS#5-CS#6
BOFF#	3612.7	3596.1	7605.8	936.5	936.5	936.5

NET	CS#6-CS#9	CS#3-CS#4	CS#4-CS#10	CS#3-CS#7	CS#7-CS#8	Stubs
BOFF# (cont)	936.5	936.5	936.5	944.8	936.5	75.0-75.0

PP = Pentium® processor
CC = 82496 cache controller
CS = 82491 cache SRAM

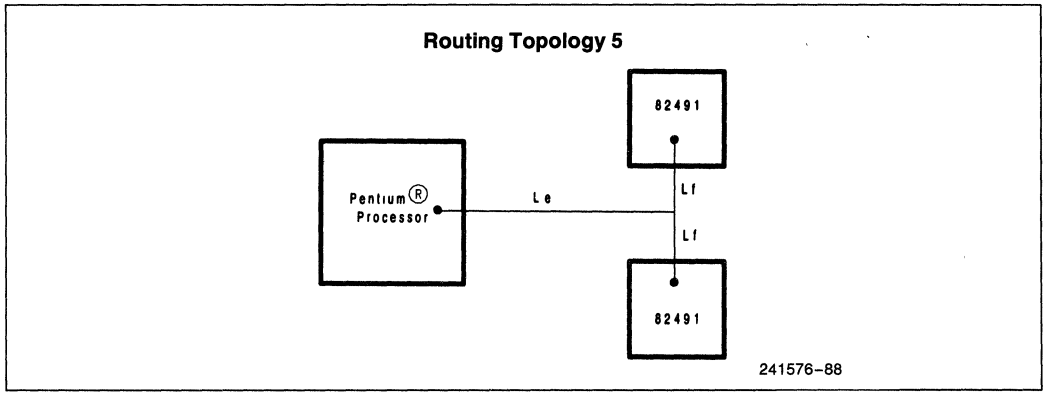
7.6.9.8 82496 Control (Topology 4, Point-to-Point)



NET	CC-PP
AHOLD	4549
BRDYC1#	3648.5
EADS#	3656.4
INV	4603.5
KEN#	4136.9
NA#	3770.3

PP = Pentium® processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.9 Byte Enables (Topology 5)

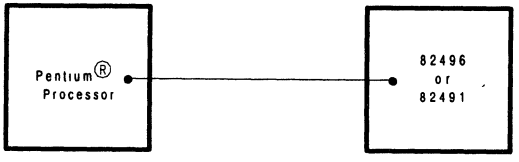


NET	PP-Tee	Tee-CS#1	Tee-CS#9	Stubs
CBE0#	3035.4	1634.4	1633.9	112.4 135.3
NET	**PP-Tee**	**Tee-CS#2**	**Tee-CS#9**	**Stubs**
CBE1#	4098.7	1294.8	1293.1	75.0–121.0
NET	**PP-Tee**	**Tee-CS#3**	**Tee-CS#9**	**Stubs**
CBE2#	3412.9	1732.0	1682.3	75.0–95.3
NET	**PP-Tee**	**Tee-CS#4**	**Tee-CS#9**	**Stubs**
CBE3#	3547.8	1194.8	1192.1	75.0–75.0
NET	**PP-Tee**	**Tee-CS#5**	**Tee-CS#10**	**Stubs**
CBE4#	3600.9	2243.5	2242.0	75.0–135.3
NET	**PP-Tee**	**Tee-CS#6**	**Tee-CS#10**	**Stubs**
CBE5#	4811.9	1339.7	1338.9	75.0–75.0
NET	**PP-Tee**	**Tee-CS#7**	**Tee-CS#10**	**Stubs**
CBE6#	4662.0	1663.0	1662.6	75.0–135.3
NET	**PP-Tee**	**Tee-CS#8**	**Tee-CS#10**	**Stubs**
CBE7#	4230.0	1167.7	1165.5	75.0–75.0

PP = Pentium® processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.9.10 CDATA and Parity (Point-to-Point)

Routing Topology 4



241576-89

Net	PP-CS#9	Stub
CP0	5722.4	135.3
CP1	5842.1	135.3
CP2	5854.7	75.0
CP3	5712.4	75.0

NET	PP-CS#10	Stub
CP4	5831.1	135.3
CP5	5849.8	135.3
CP6	5563.8	75.0
CP7	5558.6	75.0

NET	PP-CS#1	Stub	NET	PP-CS#3	Stub	NET	PP-CS#5	Stub
CD0	5523.3	135.3	CD16	5541.4	135.3	CD32	5507.8	135.3
CD1	5376.2	135.3	CD17	5781.0	135.3	CD33	5419.5	135.3
CD2	5476.9	75.0	CD18	5550.7	75.0	CD34	5433.2	75.0
CD3	5363.3	75.0	CD19	5558.2	75.0	CD35	5756.3	75.0
CD4	5547.9	135.3	CD20	5449.7	135.3	CD36	5654.1	135.3
CD5	5393.5	75.0	CD21	5545.2	75.0	CD37	5551.6	75.0
CD6	5357.9	135.3	CD22	5410.4	135.3	CD38	5357.3	135.3
CD7	5582.3	75.0	CD23	5533.2	75.0	CD39	5451.6	75.0

NET	PP-CS#2	Stub	NET	PP-CS#4	Stub	NET	PP-CS#6	Stub
CD8	5448.4	135.3	CD24	5804.5	135.3	CD40	5430.0	135.3
CD9	5516.1	135.3	CD25	5594.4	135.3	CD41	5757.8	135.3
CD10	5629.1	75.0	CD26	5754.8	75.0	CD42	5378.1	75.0
CD11	5468.9	75.0	CD27	5705.4	75.0	CD43	5703.8	75.0
CD12	5451.5	135.3	CD28	5774.5	135.3	CD44	5462.3	135.3
CD13	5543.2	75.0	CD29	5737.7	75.0	CD45	5755.1	75.0
CD14	5707.3	135.3	CD30	5380.7	135.3	CD46	5568.6	135.3
CD15	5420.3	75.0	CD31	5608.3	75.0	CD47	5757.2	75.0

NET	PP-CS#7	Stub
CD48	5488.3	135.3
CD49	5551.8	135.3
CD50	5697.9	75.0
CD51	5581.1	75.0
CD52	5499.5	135.3
CD53	5704.4	75.0
CD54	5493.6	135.3
CD55	5627.9	75.0

NET	PP-CS#8	Stub
CD56	5553.9	135.3
CD57	5663.3	135.3
CD58	5602.7	75.0
CD59	5718.3	75.0
CD60	5451.6	135.3
CD61	5533.9	75.0
CD62	5550.4	135.3
CD63	5766.2	75.0

PP = Pentium® processor
CC = 82496 cache controller
CS = 82491 cache SRAM

7.6.10 Pentium® PROCESSOR TO 82496 SEGMENT LENGTH AND ROUTING CHANGES

The example layout described in this application note was completed using early revisions to the I/O buffer models. This process was necessary to ensure that a board was available for the arrival of first silicon. After the models were improved based on the model validation and silicon characterization, the board layout was resimulated. These simulations have resulted in the recommendation to change the line length between the Pentium processor and 82496 for several nets. These changes result in a better tuned routing that meets the specifications. In particular, these changes reduce the amount of ringback and the ringing that leads to long settling times. Table 12 summarizes the recommended segment length changes.

Table 12. Summary of Segment Lengths

Net/Signal Name	Segment	Original Length (in.)	Recommended Length (in.)
WRARR#	CC-CS#3	2.802	2.9
WRARR#	CC-CS#5	2.735	2.9
PA4	PP-CC	4.673	4.3
PA6	PP-CC	5.123	4.9
PA7	PP-CC	5.368	5.1
PA10	PP-CC	5.233	4.9
PA12	PP-CC	5.324	5.0
PA16	PP-CC	5.359	4.9

PP = Pentium® processor
CC = 82496 cache controller
CS = 82491 cache SRAM

Actual system measurements have shown that the original segment lengths do not violate the specifications. The reduction in ringing is probably due to transmission line losses which are not accounted for in the simulation. Therefore for completed designs using the example layout these changes are not necessary; however, Intel does recommend that all future designs that use the layout example use these new lengths.

In addition, a layer change to the BRDYC1# routing is recommended. Section 7.5 Design Notes, describes that BRDYC1# was routed on an outer layer to reduce the propagation delay; however, this resulted in a signal quality violation. Since the original routing of the board, the flight time specification was relaxed and BRDYC1# can now be routed on an inner layer which allows it to meet both signal quality and flight time specifications.

7.6.11 I/O SIMULATION RESULTS FOR EACH NET

Electrical simulations were performed on each net within the optimized interface of the 256 Kbyte CPU-Cache Chip Set design example. The simulations were done at the fast and slow corners to verify that signal quality and flight time specifications are met. The simulations were done using TLC V4.1.13 from Quad Design Technology, Inc. using the files described in Section 7.6.2. Table 13 summarizes the simulation results assuming all the segment length changes listed in Section 7.6.10 have been implemented along with the layer change to the BRDYC1# routing.

Table 13. Summary of Simulation Results

Net	Flight Time (ns)	Signal Quality			
		Over/ Undershoot (V)	Ringback (V)	Settling Time (ns)	Time Beyond Supply (ns)
Specs.	**Vary by Pin**	**3.0**	**1.75**	**12.5**	**6.0**
82496 Driving					
A3–16 at CPU	7.2	2.1	1.0	16.5	7.4
A3–16 at SRAM	7.0	2.1	1.0	16.5	7.4
A17–31 at CPU	2.6	3.0	1.75	10.3	3.6
BT0–3	2.6	3.0	1.75	10.3	3.6
AHOLD	1.1	3.0	1.75	9.0	2.1
AP	1.4	3.0	1.75	9.0	2.1
BRDYC1#	0.9	2.9	1.7	8.3	1.8
EADS#	0.9	2.9	1.7	7.8	1.8
EWBE#	0.7	2.5	1.4	6.1	1.5
INV	1.6	2.8	1.5	12.4	2.9
KEN#	1.0	2.9	1.75	8.5	2.0
NA#	0.9	2.9	1.7	7.9	1.9
WB/WT#	0.9	2.8	1.7	7.5	1.8
BLAST#	2.5	2.2	1.0	6.2	3.3
BLEC#	2.2	2.1	0.9	5.9	3.1
BOFF# at CPU	2.5	2.6	1.1	12.1	3.0
BOFF# at SRAM	2.9	2.6	1.1	12.1	3.0
BRDYC2#	2.5	2.2	1.1	6.2	3.7

Table 13. Summary of Simulation Results (Continued)

Net	Flight Time (ns)	Signal Quality			
		Over/ Undershoot (V)	Ringback (V)	Settling Time (ns)	Time Beyond Supply (ns)
BUS#	2.5	2.1	0.9	6.5	3.3
MAWEA#	2.6	2.2	1.0	8.1	3.4
MCYC#	2.6	2.1	0.8	6.5	3.3
WAY	2.5	2.1	0.9	6.4	3.6
WBA	2.7	2.2	1.0	6.7	3.5
WBTYP	2.6	2.2	1.0	6.1	3.6
WBWE#	2.6	2.1	0.9	6.2	3.3
WRARR#	2.5	2.4	1.1	8.1	3.3
Pentium® Processor Driving					
A3–16 at Controller	2.5	2.6	1.2	12.8	3.2
A3–16 at SRAM	2.8	2.6	1.2	12.8	3.2
A17–31	1.5	3.0	1.8	10.6	2.3
BT0–3	1.5	3.0	1.8	10.6	2.3
D0–63, DP0–7	1.2 min. 1.4 max.	3.0	1.8	11.5	2.5
ADS#	2.6	2.2	1.0	7.3	3.7
HITM# at Controller	3.0	3.2	1.6	20.3	4.1
HITM# at SRAM	3.2	3.2	1.6	20.3	4.1
W/R# at Controller	3.0	3.1	1.6	20.7	4.1
W/R# at SRAM	3.2	3.1	1.6	20.7	4.1
ADSC#	1.2	3.0	1.75	7.3	1.7
AP	1.3	3.0	1.75	8.8	2.1
CACHE#	1.1	2.9	1.75	7.2	1.7
D/C#	1.1	2.9	1.75	7.1	1.7
LOCK#	1.3	3.0	1.8	8.6	2.0
M/IO#	1.4	3.0	1.8	9.1	2.1

Table 13. Summary of Simulation Results (Continued)

Net	Flight Time (ns)	Signal Quality			
		Over/ Undershoot (V)	Ringback (V)	Settling Time (ns)	Time Beyond Supply (ns)
PCD	1.1	2.9	1.7	6.9	1.6
PWT	1.2	3.0	1.8	8.7	1.9
SCYC	1.2	3.0	1.75	7.4	1.7
BE0–7#	1.9	3.5	2.0	14.7	3.3
82491 Driving					
D0–63, DP0–7	1.7 min. 1.9 max.	3.0	1.9	12.1	2.7

Shading indicates a spec. violation.

The shaded entries indicate specification violations with the example design layout. Intel is continuing to work on addressing these violations by either relaxing specifications or improving the layout design. Note that no violations have been measured on the actual board. The violations have all been in simulation.

7.7 Possible Modification to the Layout

7.7.1 NON-PARITY LAYOUT

Intel has not simulated a non-parity layout example. The following suggestions will assist in modifying the design example for non-parity implementations. You must simulate all paths that are altered when the parity components are removed to ensure that flight time and signal quality specifications are still met.

Modify the following aspects of the layout example:

1. Remove the two leftmost 82491 components, U9 and U10. These are the parity components.
2. Rework Topologies 1 and 1b. Balance the array so that the two critical branch points branch out to electrically equivalent traces, i.e., adjust Lw to be electrically equivalent to Lv + Lz. Keep the trace leading to the Pentium processor length La. Topologies 1NP and 1bNP illustrate this (NP = Non-Parity). Also, retune Lb to be electrically equivalent with these new trace lengths. Topologies 1 and 1b indicate exactly where the critical branch points are.
3. Rework topologies 3 and 3b. Make the four traces branching from the 82496 electrically equivalent. This may be accomplished by making Lg = Lh for these topologies.
4. Remove the Byte Enable traces that connect to the parity chips.

Making traces electrically equivalent means that reflections from all branches return to the source at the same point in time. In simple cases, electrically equivalent traces are the same length. In all cases, simulate the effects of changing trace lengths to find the proper trace length and routing.

8.0 512K CPU-CACHE CHIP SET OPTIMIZED INTERFACE LAYOUT DESIGN EXAMPLE

This chapter contains an example layout design for Intel's 512 Kbyte CPU-Cache Chip Set's optimized interface. Intel has simulated and verified the example layout using the latest information. Work is currently underway to validate the design by measuring the flight time and signal quality parameters on boards based on the design example. As updated information becomes available on the components and the boards, Intel plans to update this example accordingly.

The intent of the design example is to provide system designers a starting point. It provides one solution of how the Pentium processor, 82496, and 82491 components can be placed and routed to ensure flight time and signal quality specifications are met. It is not the only solution. System designers can alter the layout to meet their system requirements as long as the flight time and signal quality specifications are met.

8.1 Layout Objectives

The 512K layout is an example of a CPU-Cache chip set arrangement that meets Intel's chip set specifications. The layout consists of 1 Pentium processor, 1 82496 cache controller, and 18 82491 cache SRAMs for a 512K second-level cache with parity. Although the layout is specifically designed for a chip set with parity, we will also discuss conversion to a non-parity layout.

This example layout targets the chip set's flight time and signal quality specifications. In addition to meeting those specifications, we had the following objectives:

1. To design the optimized portion of the interface so that the layout is not limited by interconnect performance. By not artificially creating any critical paths, the interface can yield maximum performance of the chip set.
2. To be consistent with EMI and thermal requirements.
3. To have the layout be used as a validation and correction vehicle by Intel. Intel will use the layout to validate the optimized interface of the chip set, measure flight times and signal quality, and tune input and output buffers.

Provided are complete specifications for a board layout: part lists, board layer plots, and the electronic files in Gerber format. Also provided are a set of topologies and line lengths so it will be easy to understand how the layout was generated.

8.2 Component Placement

To meet flight time with clock skew restrictions we placed the parts in relative proximity to each other. At the same time, we ensured that the layout's Memory Bus Controller (MBC) interface signals are routable. Figure 59 illustrates how the chip set components are placed in the layout example. The dot indicates the location of pin 1. Figure 59 shows a component side view of the layout.

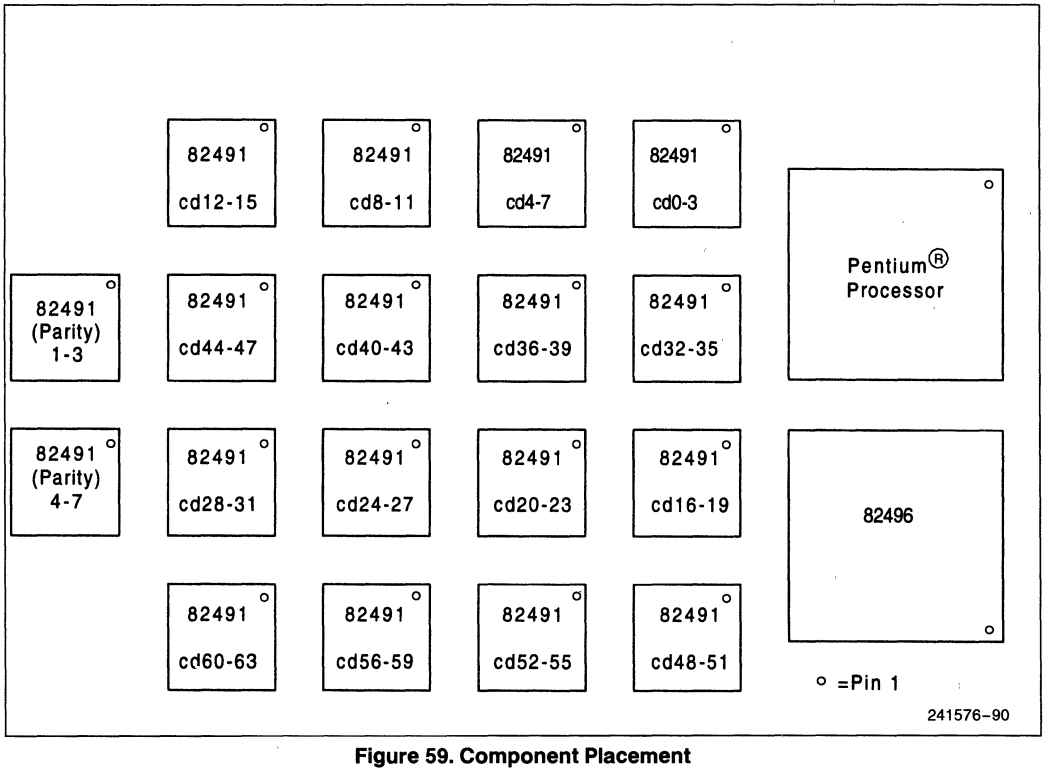


Figure 59. Component Placement

8.3 Signal Routing/Topologies

Table 14 and Table 15 list the signal nets and their corresponding topologies for the optimized and external interfaces of the CPU-Cache Chip Set.

All chip set signals in the optimized interface fall into six groups: low addresses, high addresses, Pentium processor control, 82496 control, CPU data, and byte enables. Within each group are subsets of signals that share common origination and destination points. Each subset has a unique routing called a "topology." Groups, subsets, and topologies are listed in Table 14.

Topologies are given only for signals that are routed to multiple chips. It is the system designer's responsibility for routing the "point-to-point" signals such as CADS#.

Topologies are also supplied for the external interface. These topolgies provide channels for routing signals from the chip set components to the periphery where they can be connected to the memory bus and memory bus controller (MBC). However, topologies are not supplied for point to point signals in the MBC interface (e.g. CRDY#). Instead, the system designer must optimize these for the particular application.

Table 15 lists the topologies provided for the MBC interface signals which are not point to point.

Figures 60 through 77 are the topologies which are described in Table 14 and 15. A topology is a graphical representation how specific sets of signals are routed. A topology shows the components that share a specific signal and the relative lengths of the traces between components.

2

Table 14. Optimized Interface Signal Net/Topology Assignments

Grouping	Routing Requirements	Topology
Low Addresses		
(PA3–PA17)	Bused to all core components. Must be routed to optimize delay and signal quality at all points.	1
High Addresses		
(PA18–PA31, PBT0–PBT3)	Point to point links. Must be kept as short as possible.	6
Pentium® Processor Control		
(HITM#, W/R#)	Same as low addresses.	1
(ADS#)		5
(ADSC#, AP, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC)	Same as high addresses.	6
CC Control		
(BUS#, MAWEA#, WBWE#, WBTYP#, WBA, BLAST#)	Must be routed to optimize delay and signal quality at the CS.	3
(BLEC#)	Not routed to parity CSs.	4
(BOFF#)		1
(AHOLD, EADS#, KEN#, BRDYC1#, INV, EWBE#, NA#, WB/WT#)	Same as high addresses.	6
(BRDYC2#, WRARR#, MCYC#, WAY)	Routed differently.	15
CPU Data		
(CD0–CD63, CP0–CP7)	Point to point signals. Keep as short as possible. Keep within 3" of each other to minimize skew.	6
Byte Enables		
(CBE0#–CBE7#)		7

Table 15. External Interface Signal Net/Topology Assignments

Signal	Topology
MDATA0–63, Parity0–7	8
BRDY0#, CLK0	9
CRDY#	10
RESETC	17
MBRDY#, MOCLK, MDOE#	11
BRDY1–3#	12
MEOC1–3#	18
CLK1–3	13
MFRZ#, MSEL#, MZBT#, MCLK	14
RESETCPU	16

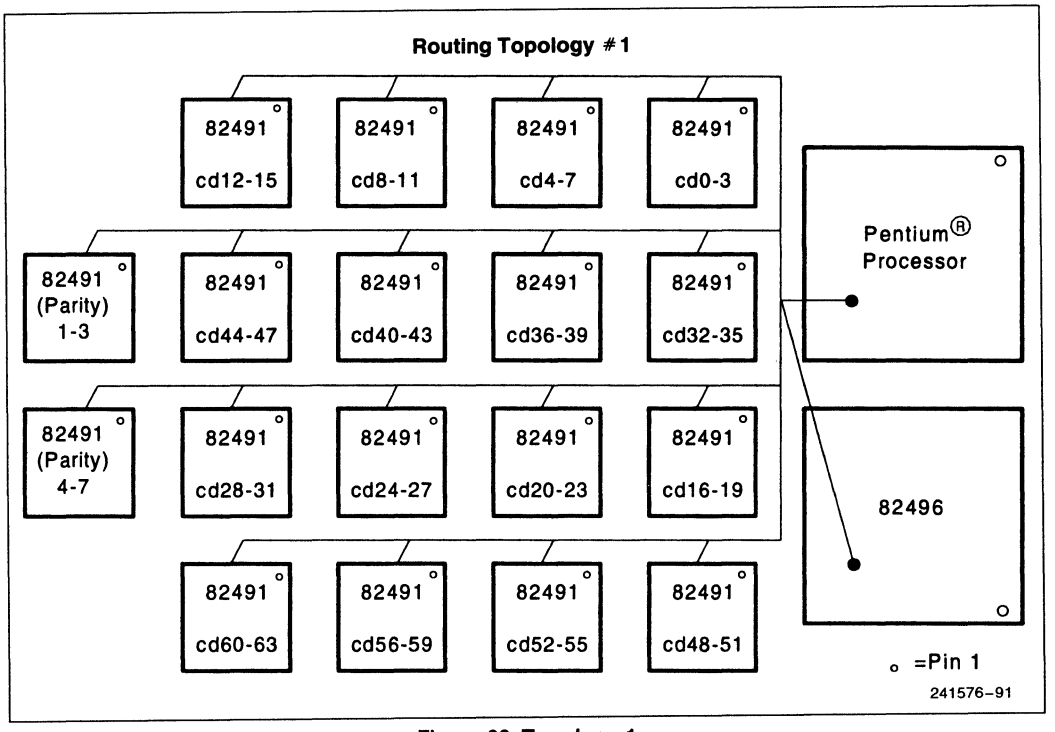


Figure 60. Topology 1

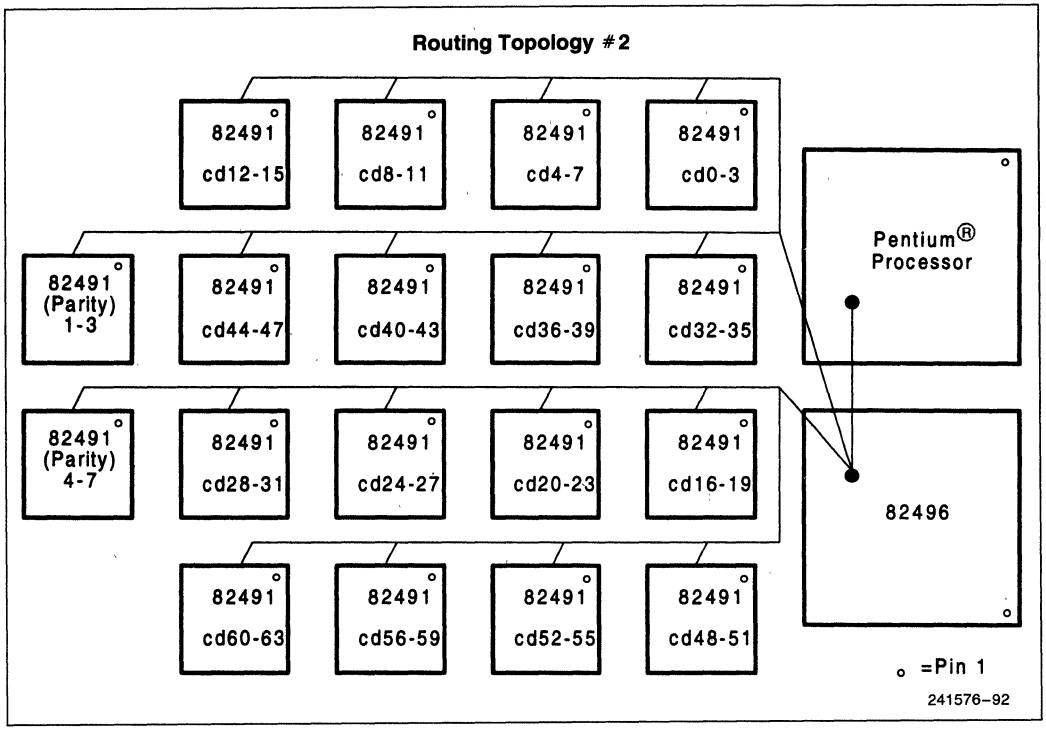


Figure 61. Topology 2

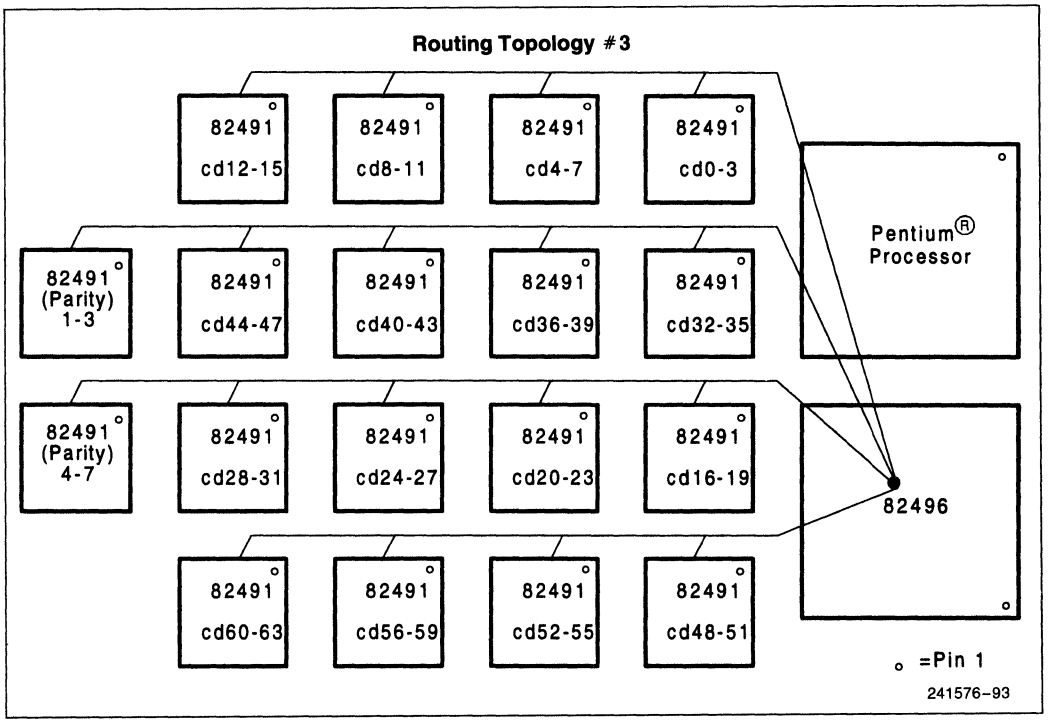


Figure 62. Topology 3

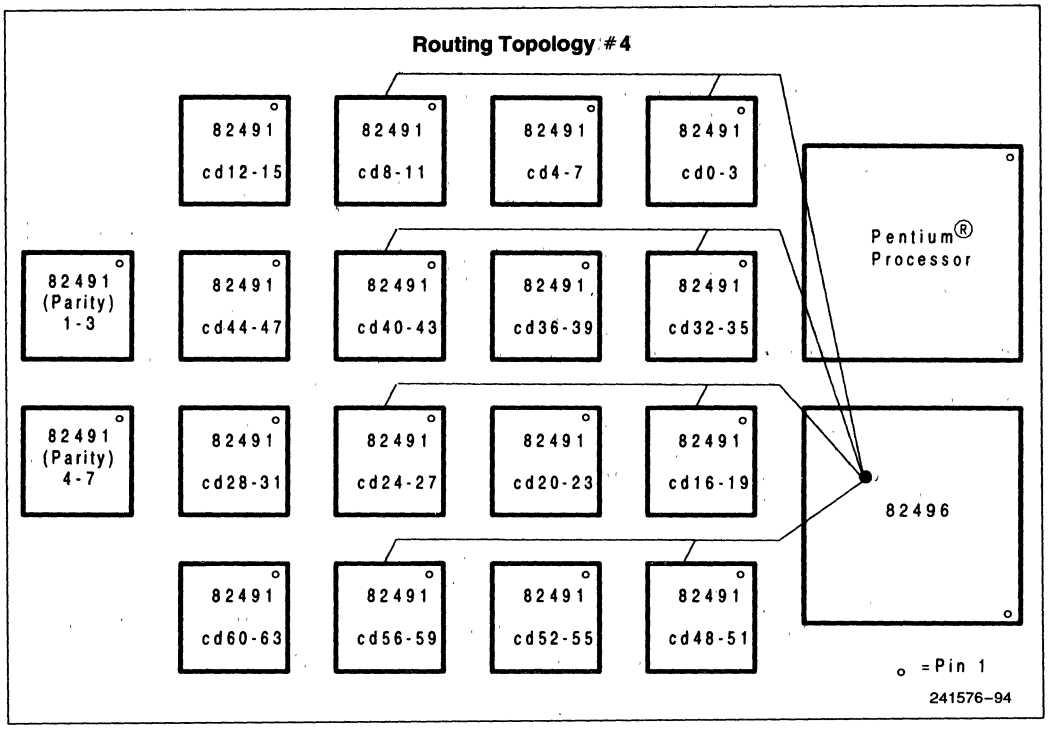


Figure 63. Topology 4

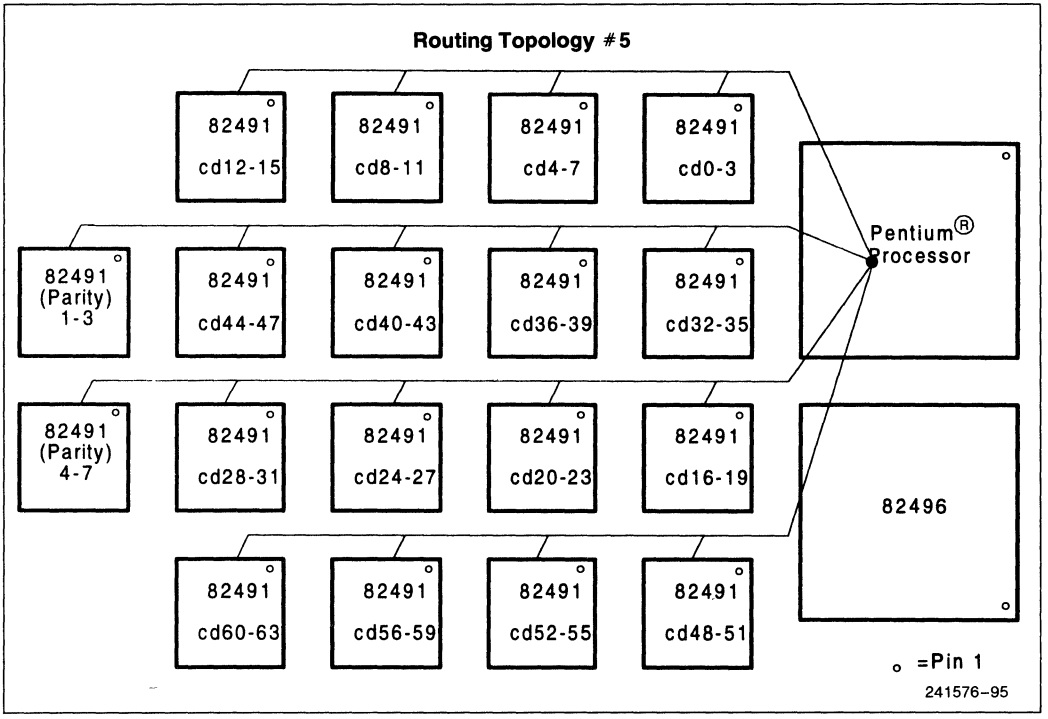


Figure 64. Topology 5

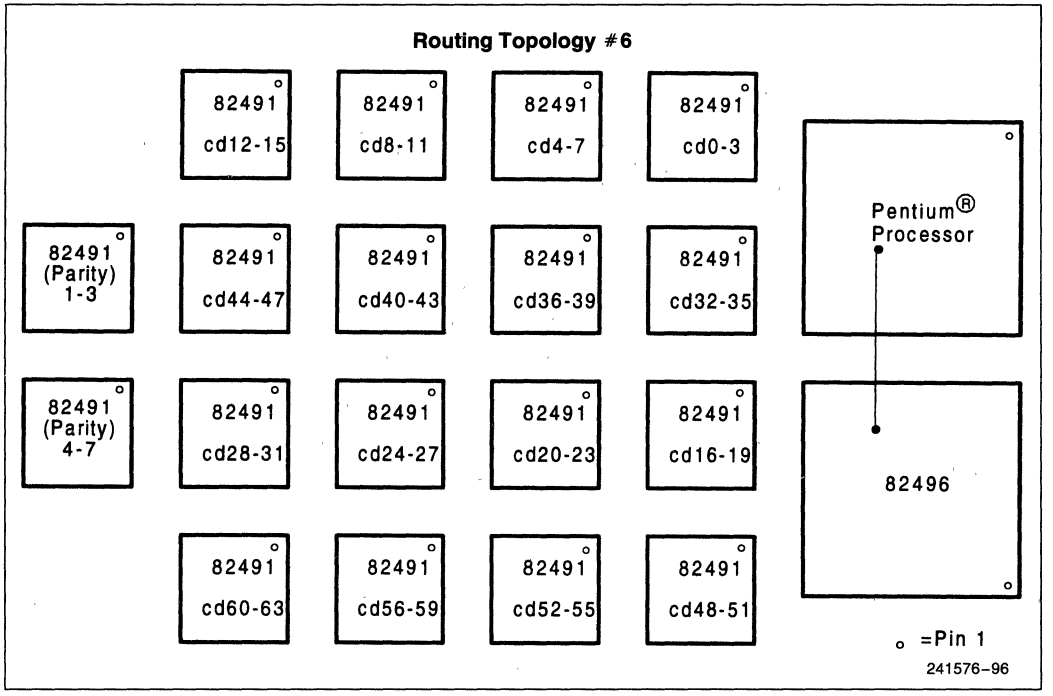


Figure 65. Topology 6

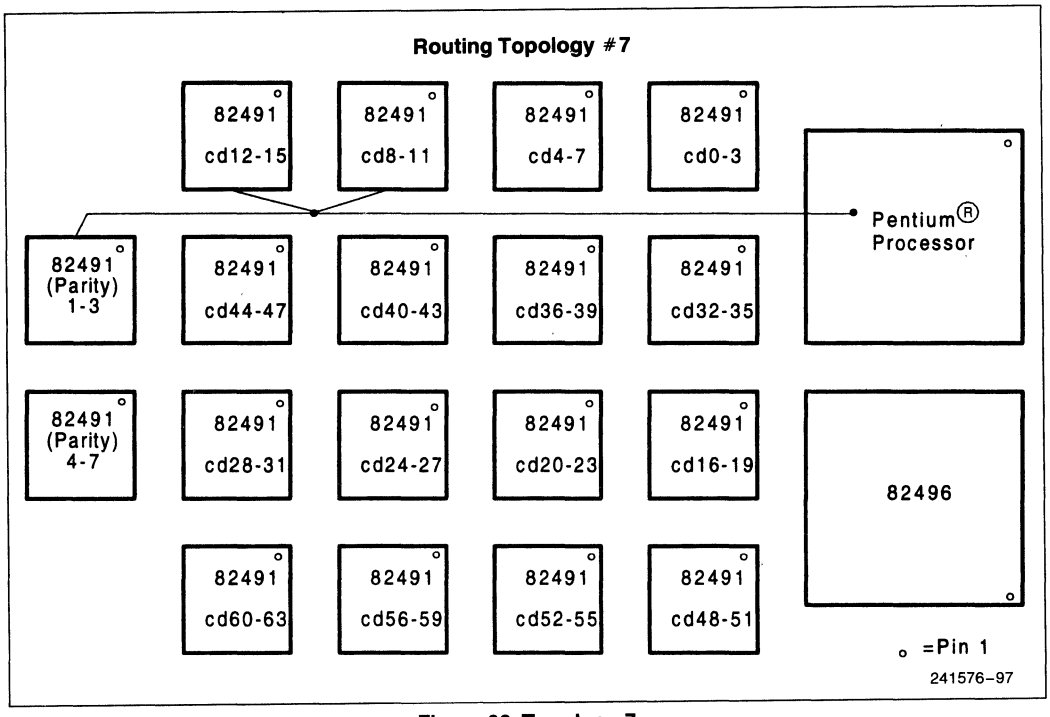


Figure 66. Topology 7

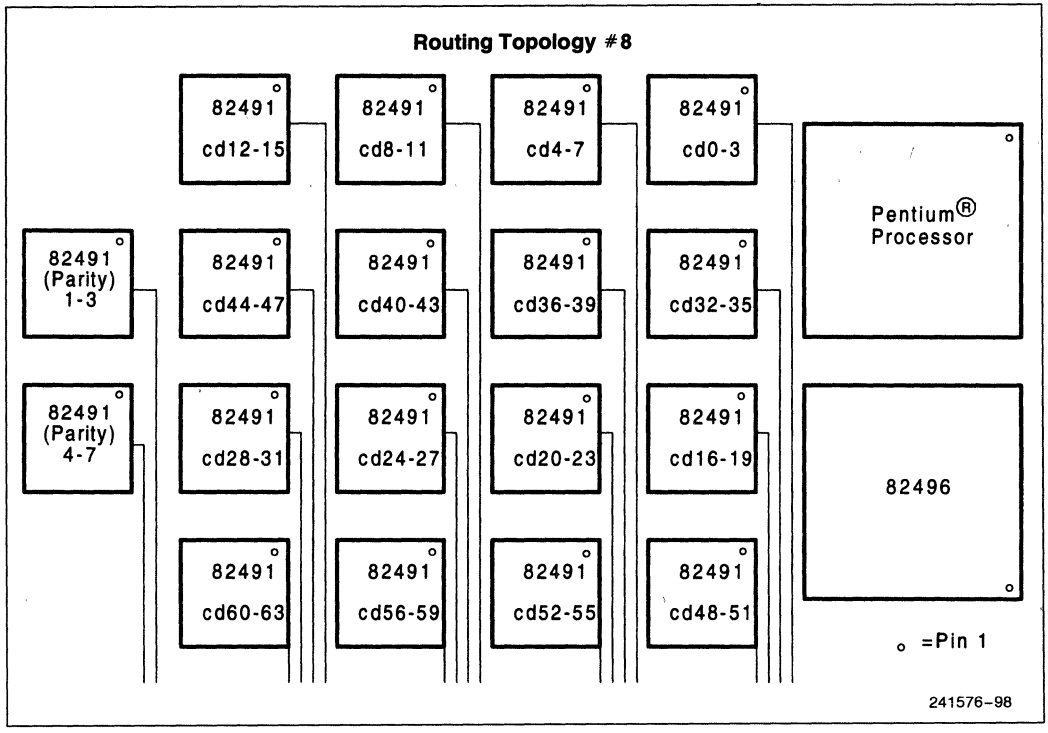


Figure 67. Topology 8

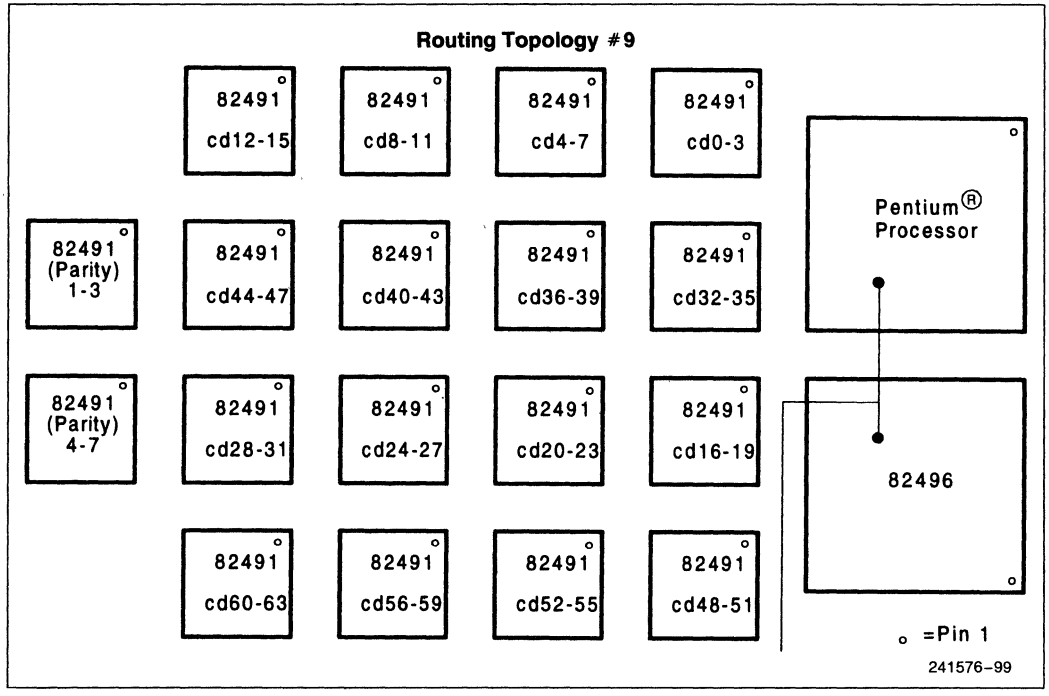


Figure 68. Topology 9

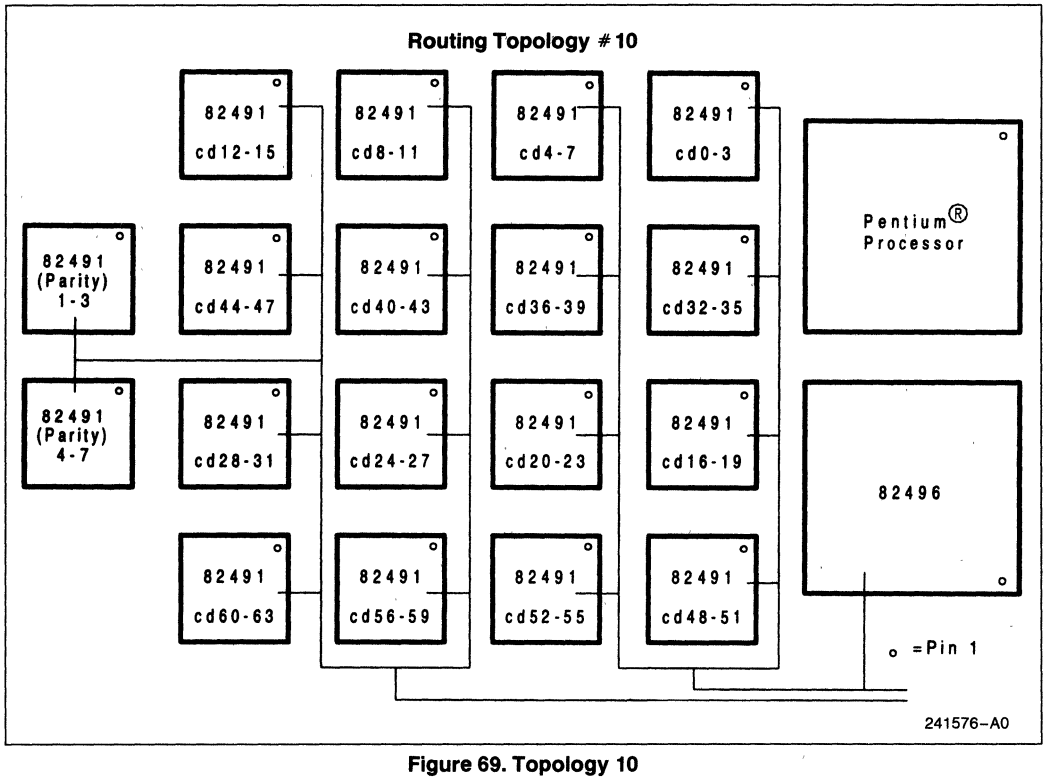


Figure 69. Topology 10

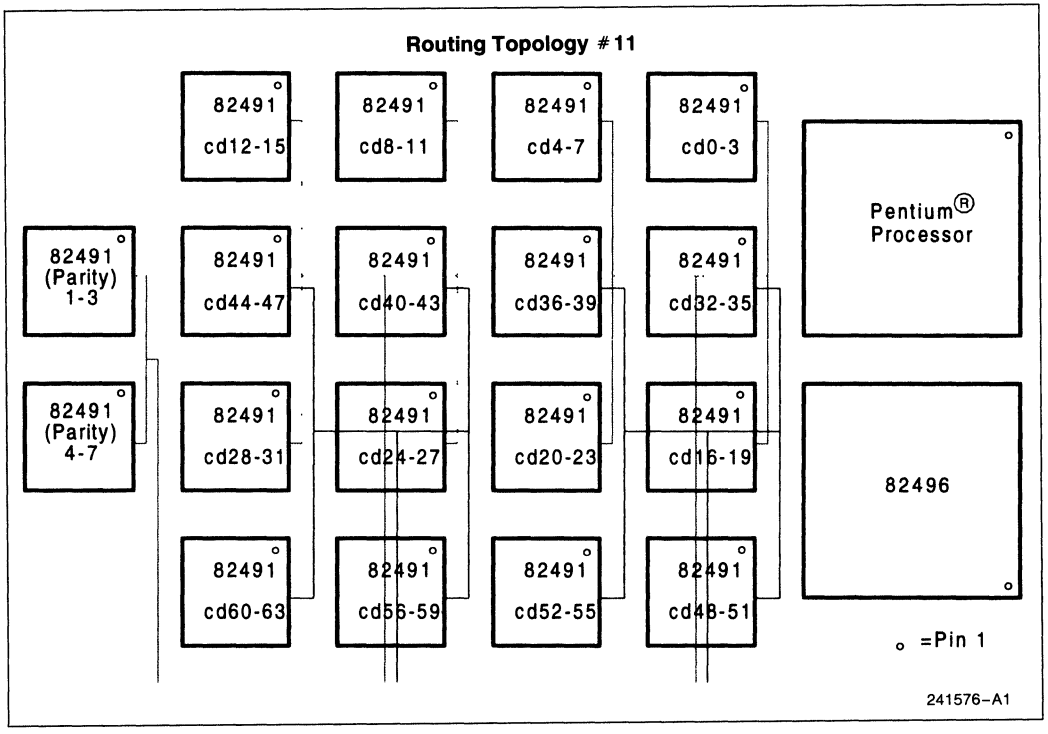


Figure 70. Topology 11

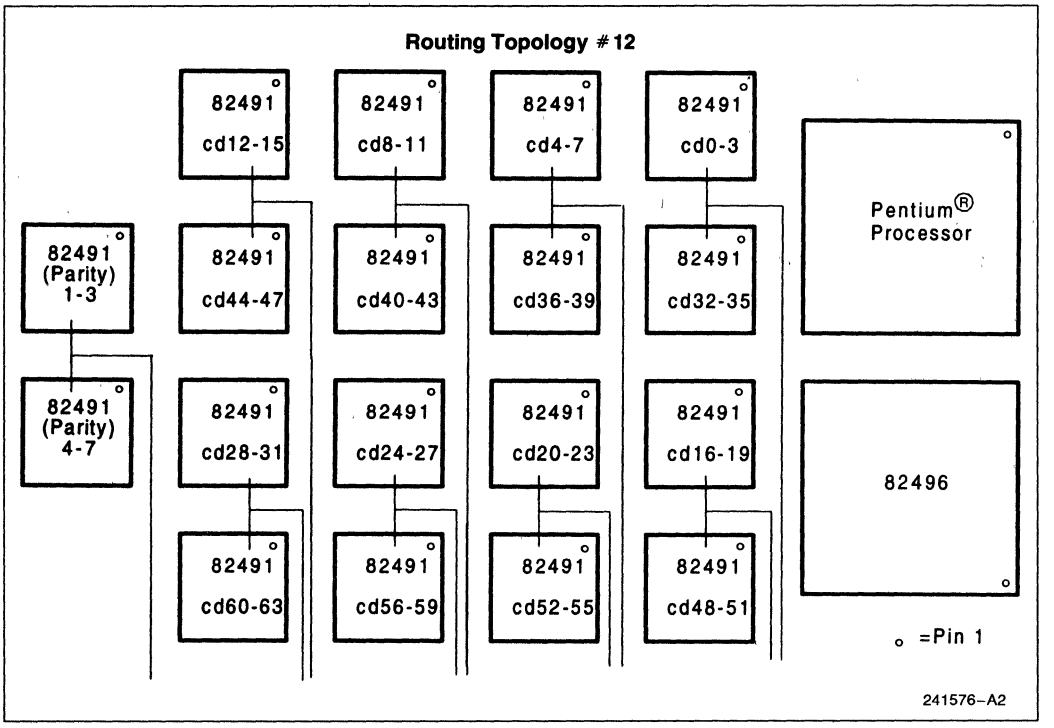


Figure 71. Topology 12

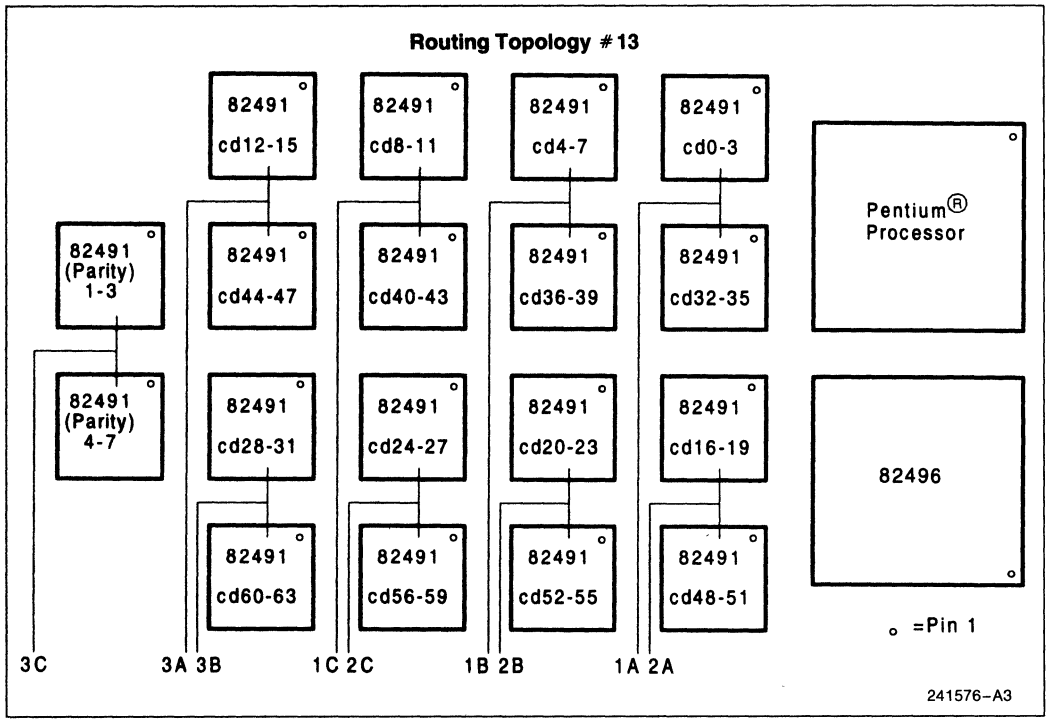


Figure 72. Topology 13

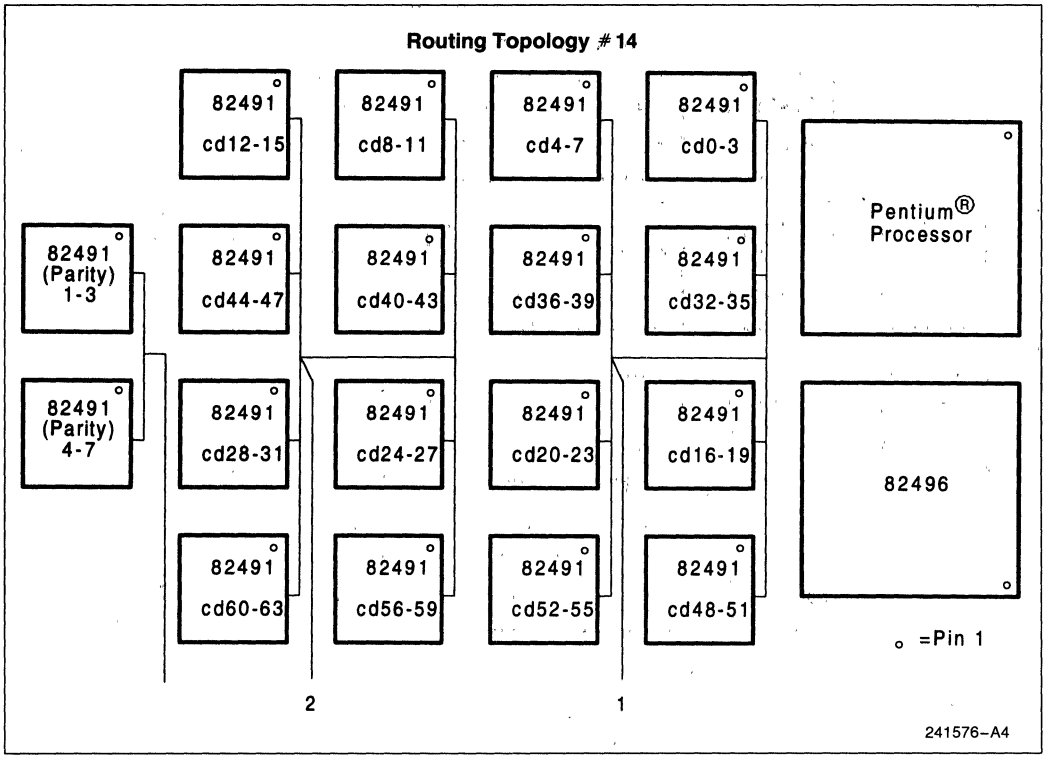


Figure 73. Topology 14

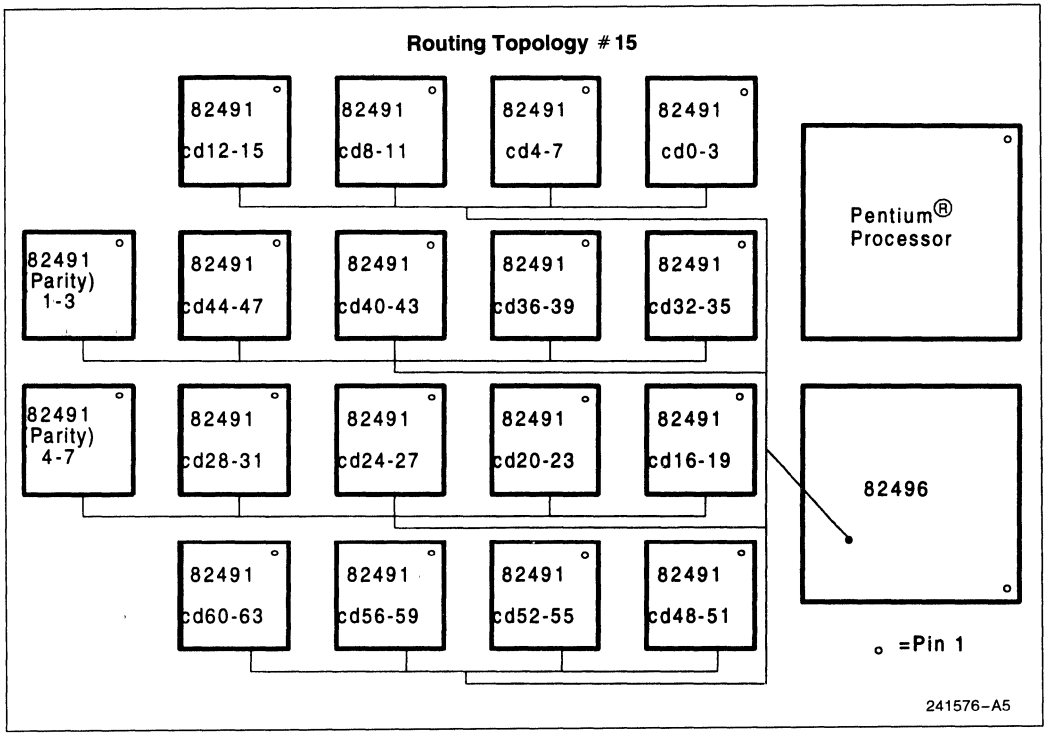


Figure 74. Topology 15

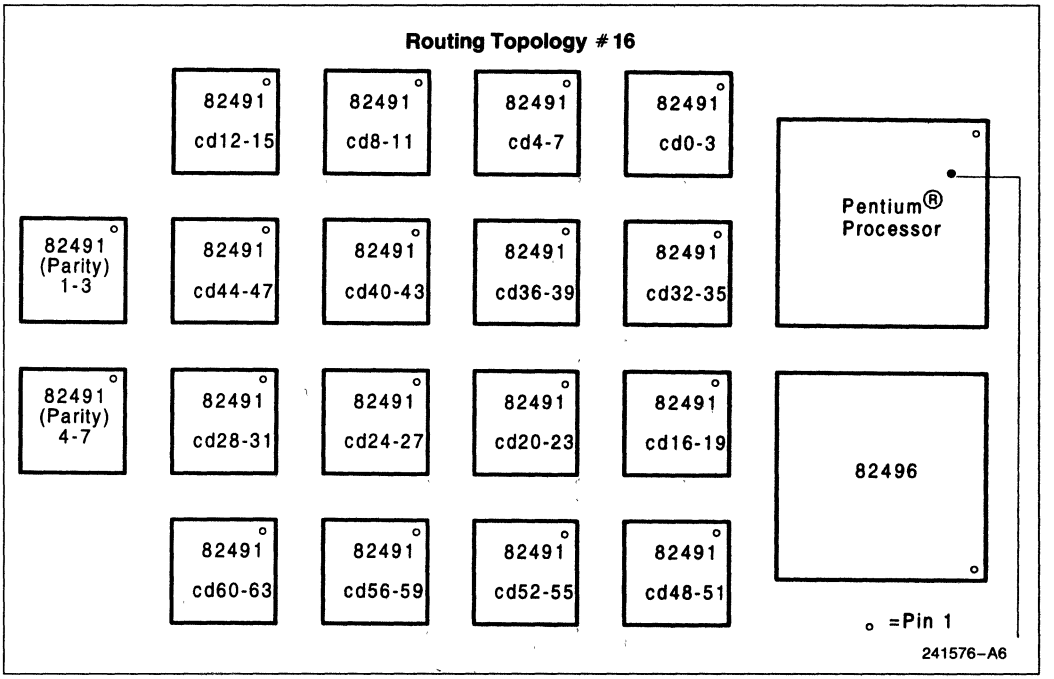


Figure 75. Topology 16

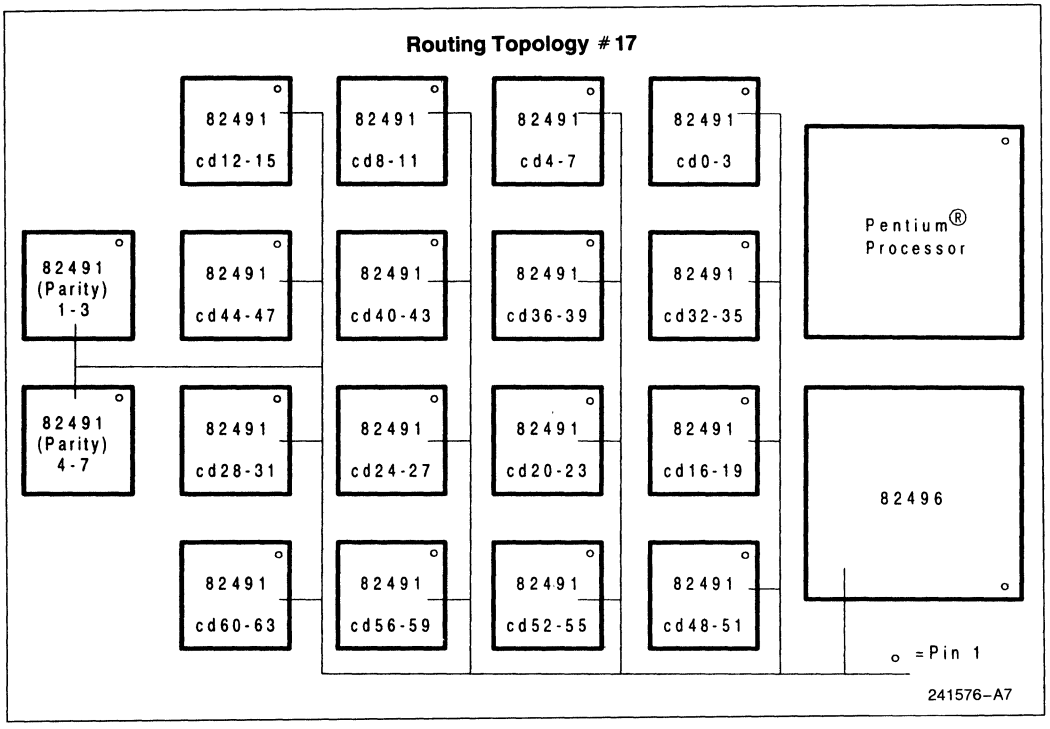


Figure 76. Topology 17

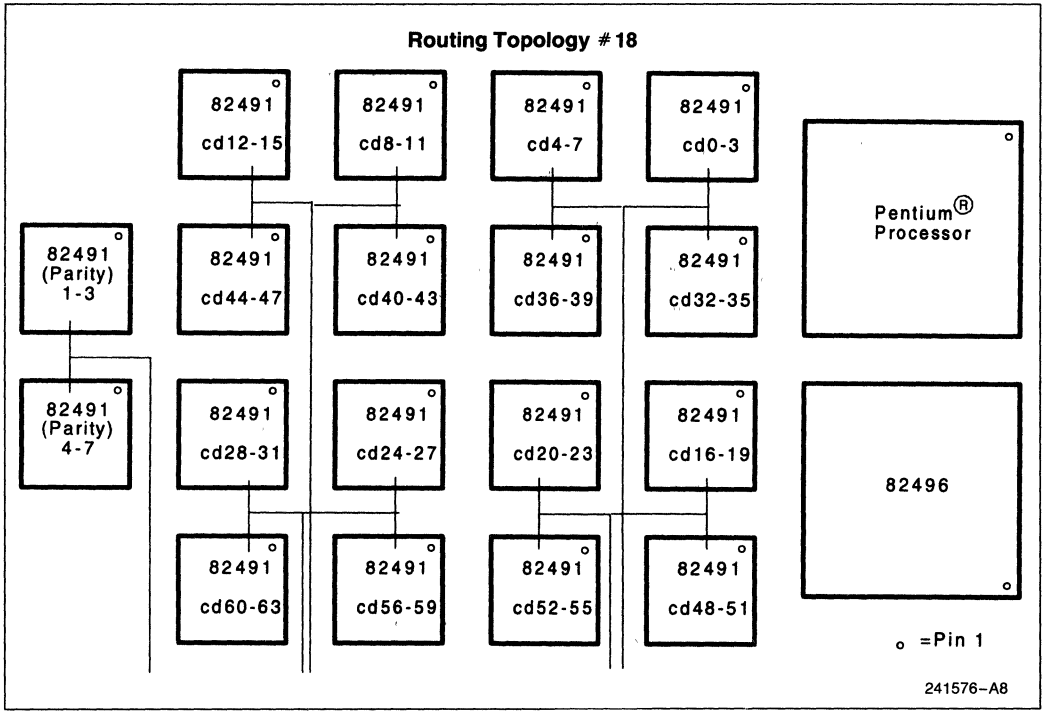


Figure 77. Topology 18

8.4 Board/Trace Properties

Specific board and trace properties were assumed while performing the simulations to optimize the chip set layout. These properties were used as the specification or guideline the board manufacturer was to use in building boards. Figure 78 provides the board layer stackup.

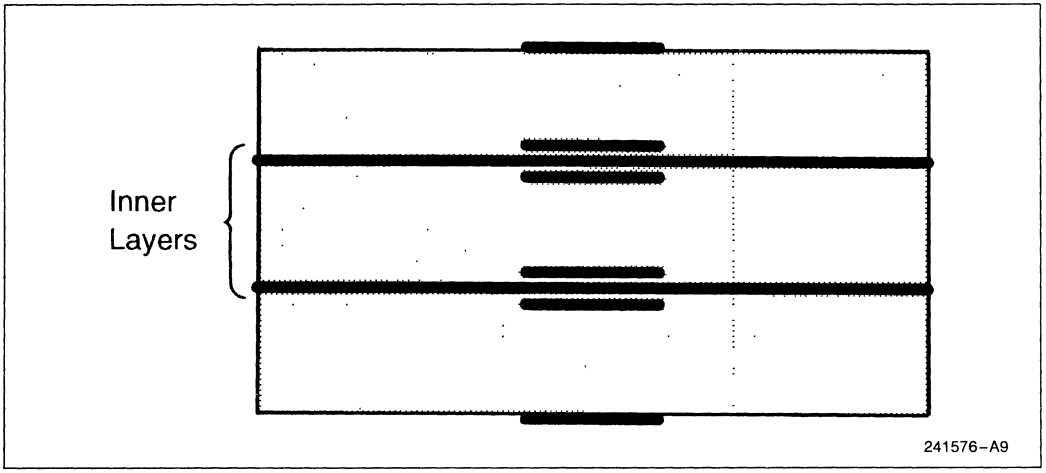


Figure 78. Board Layer Stackup

Table 16 lists the minimum and maximum trace characteristics. These parameters along with the board material determine the spacing between layers and the total board thickness. See Table 17.

Table 16. Trace Characteristics

	4 Inner Layers	2 Outer Layers
Width/Space	5/5 Mils	8/8.5 Mils
Z_0	65W ±10%	90W ±20%
Velocity	1.85 to 2.41 ns/ft	1.35 to 2.05 ns/ft

Table 17. Other Printed Circuit Board Geometries

Via Pad	25 Mils
Via Hole	10 Mils
PGA Pad	55 Mils
PGA Hole	38 Mils
Layout Grid	5 Mils

Only the inner layers of the board are impedance controlled. The top and bottom layers are not impedance controlled.

8.5 Design Notes

The following design notes accompany this layout example:

1. The layout did not specifically address heat dissipation except to allow space for heat sinks to be attached. Please see the *Pentium® Processor Family Developer's Manual* for the devices' thermal specifications. The *Pentium® Processor Thermal Design Guidelines* application note provides some examples of possible thermal solutions.
2. All fast-switching signals are routed near the power and ground planes on inner layers of the board to minimize EMI effects. However, two sets of signals are routed on the top layer of the board: BRDYC1#, and JTAG signals. BRDYC1# is routed on top to take advantage of the higher trace velocity there. JTAG signals are routed on the top layer because they are low-speed signals and will probably be re-routed by each customer to suit individual needs.
3. Resistor R5 (0Ω) is used to set the Pentium processor configurable output buffers (A3–A20, ADS#, W/R#, and HITM#). When the resistor is included the buffers are set to the Extra Large size. When it is not included (BUSCHK# internally pulled high) the buffers are set to Large size. Intel currently recommends the x-large buffers be used for the 512K layout example. The 0Ω resistor should be designed into your design as Intel may change the recommended buffer size once silicon and the system design have been characterized.

4. The 82496 output buffers that drive the 82491 inputs must also be configured to be x-large. This is done by using a pulldown resistor on 82496 CLDRV[BGT#] (pin N04). 82496 and 82491 Memory Bus buffer sizes must be controlled by the Memory Bus Controller.
5. Series termination resistors were added to the nets PA18, PA19, and PA20 to control overshoot. A value of 24W is currently recommended, but that value may change when overshoot is measured on an actual board.

8.6 Explanation of Information Provided

The following sections outline the design files associated with the 512 Kbyte CPU-Cache Chip Set design example that are available from Intel. These files are provided to simplify the task of porting the design example into a specific design. By using these files, designers may eliminate or minimize the amount of duplicate effort when using the design example as the basis for their design. The following items are provided:

- Schematics
- I/O Model Files
- Board Files
- Bill of Materials
- Photoplot Log
- Netlist Report
- Placed Component Report
- Artwork for Each Board Layer
- Trace Segment Line Lengths

Hard copies of the schematics and trace segment line lengths are provided in the following sections. ASCII or soft copies of all the information are available from Intel by requesting order number 241663, *AP-481 Design Diskettes*.

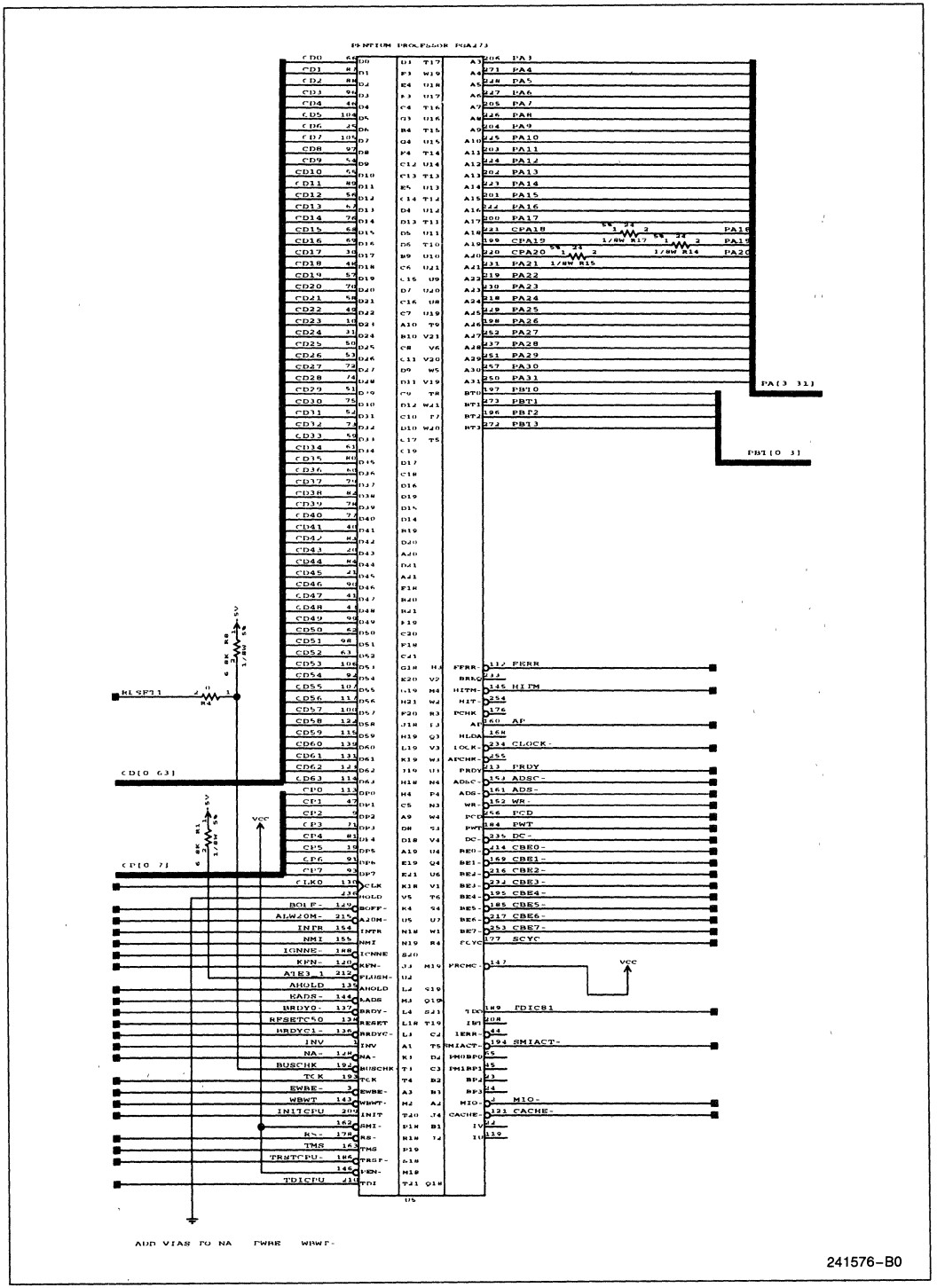
8.6.1 SCHEMATICS

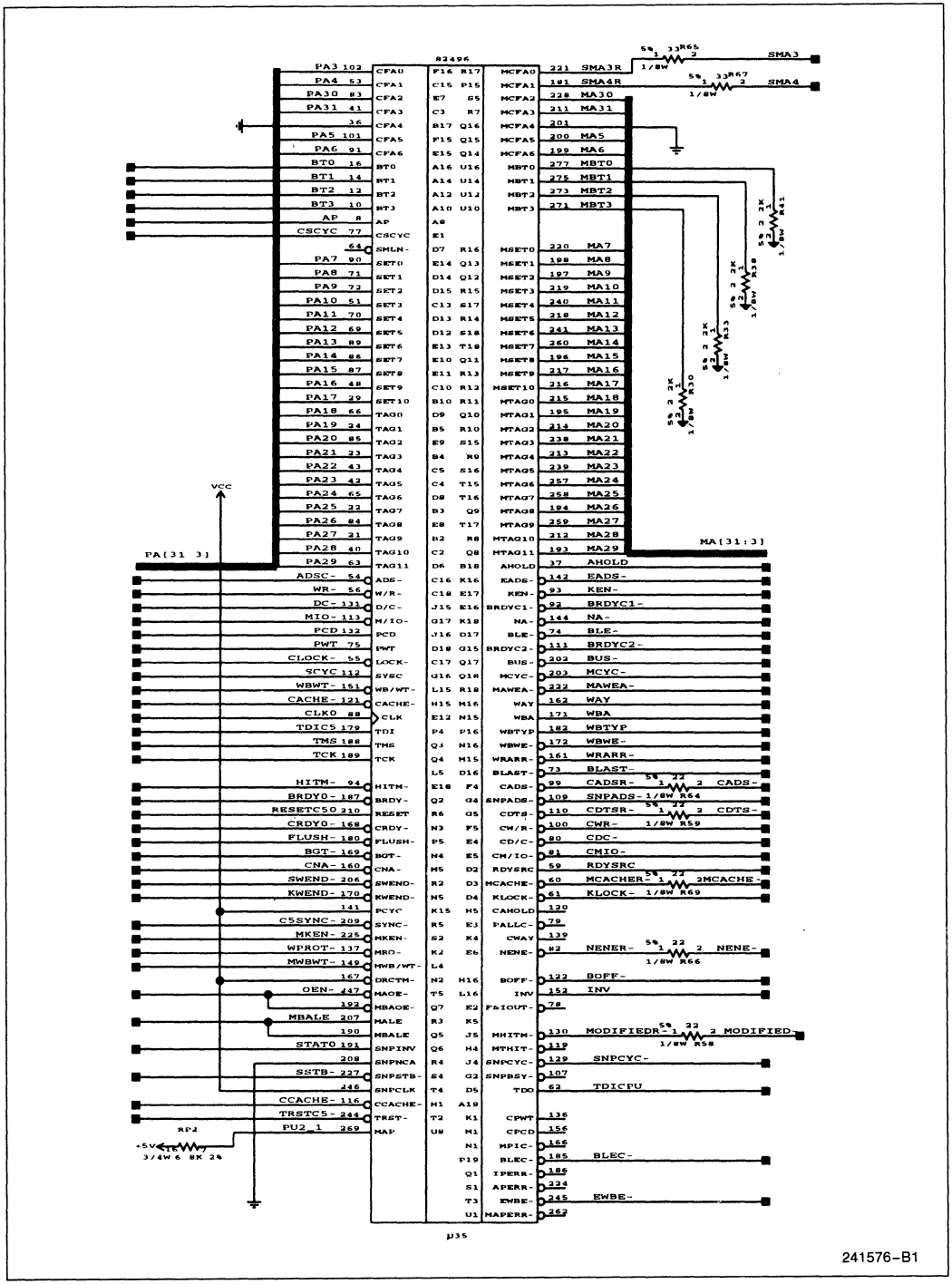
Schematics for the 512 Kbyte CPU-Cache Chip Set design example were created using ViewLogics's Workview V4.1. The schematics are 13 pages long. Both the Workview and the postscript files are available from Intel as described above.

PENTIUM® PROCESSOR/82496/82491 512KB MODULE
7-1-92
REV 2.0

(Tied MBZT- on all 82491 to Vcc)
(Removed series resistor on clk7 and added dummy load on clk7 at crdy pal, pin 27)

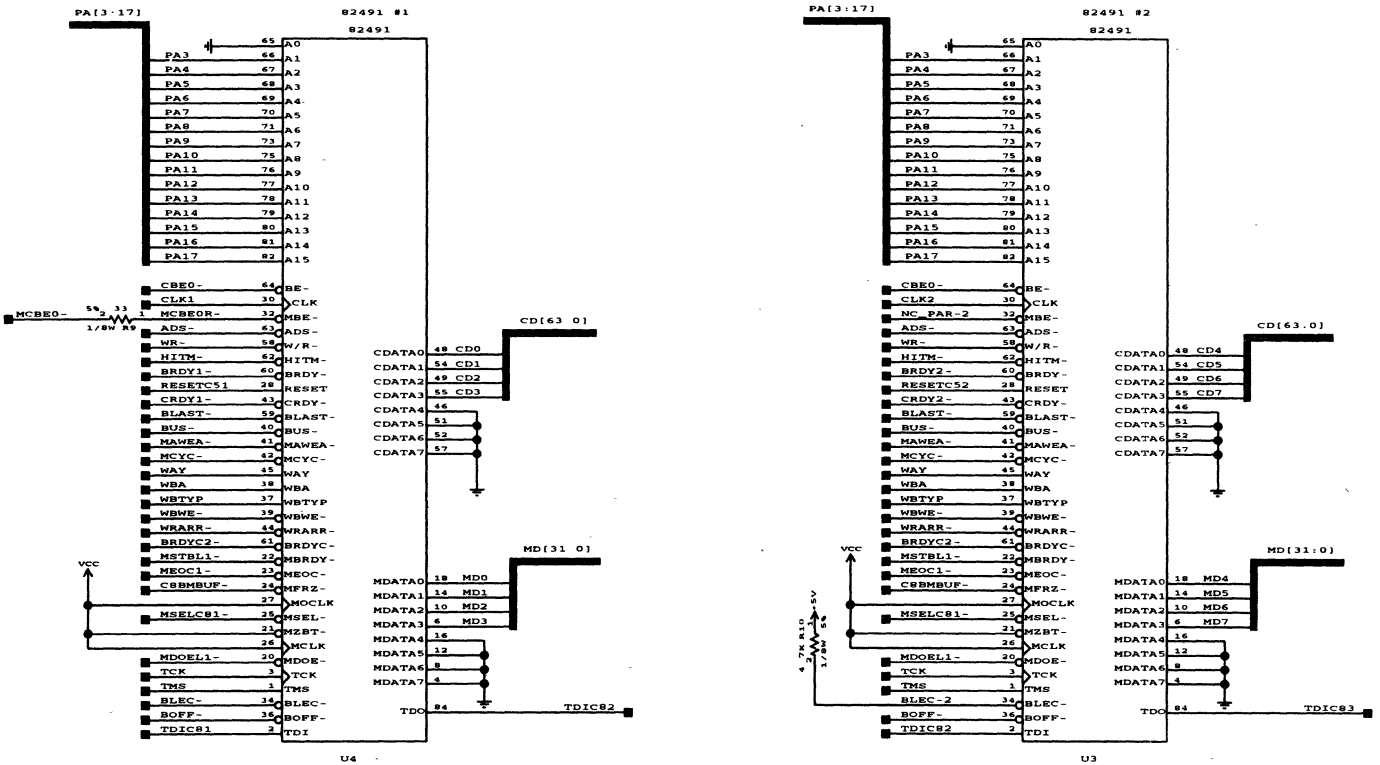
NOTES:
(Unless Otherwise Specified)
1. Capacitor values are in microfarads.
2. Resistor values are in ohms.
3. An "-" following a signal name denotes negation.
4. VCC = +5V.
5. This document also exists on electronic media.


PENTIUM PROCESSOR PGA273
CD[0 63]
CP[0 7]
RESET
VCC
5V
ADD VIAS TO NA FWBE WBWT-
241576-B0

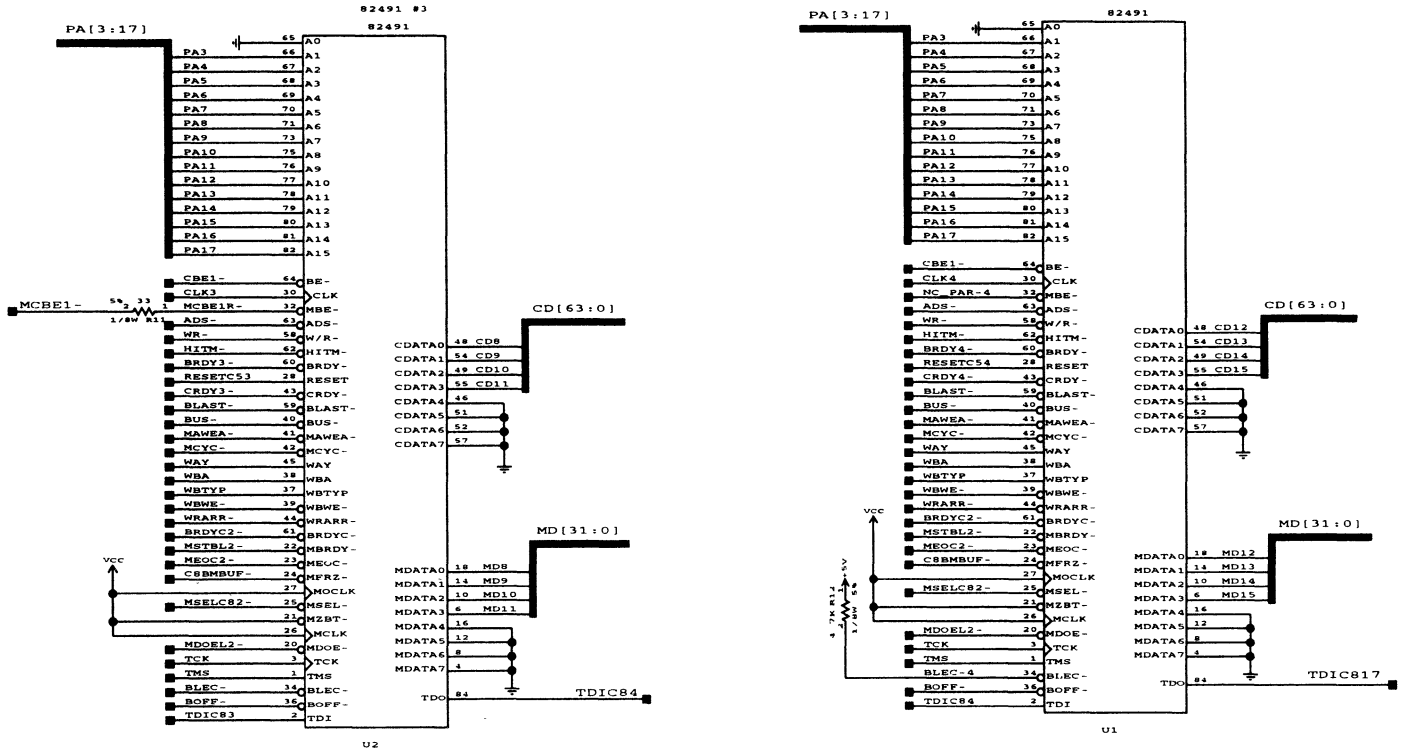

82496
PA3 102
PA4 53
PA30 83
PA31 41
36
PA5 101
PA6 91
BT0 16
BT1 14
BT2 13
BT3 10
AP 8
CSCYC 77
64
PA7 90
PA8 71
PA9 72
PA10 51
PA11 70
PA12 69
PA13 89
PA14 86
PA15 87
PA16 48
PA17 29
PA18 66
PA19 24
PA20 85
PA21 23
PA22 43
PA23 42
PA24 65
PA25 22
PA26 84
PA27 21
PA28 40
PA29 63
ADSC- 54
WR- 56
DC- 131
MIO- 113
PCD 132
PWT 75
CLOCK- 55
SCYC 112
WBWT- 151
CACHE- 121
CLK0 88
TDIC5 179
TMS 188
TCK 189
HITM- 94
BRDY0- 187
RESETC50 210
CRDY0- 168
FLUSH- 180
BGT- 169
CNA- 160
SWEND- 206
KWEND- 170
141
C5SYNC- 209
MKEN- 225
WPROT- 137
MWBWT- 149
167
OEN- 147
192
MBALE 207
190
STAT0 191
208
SSTB- 227
245
CCACHE- 116
TRSTC5- 244
PU2_1 269
RP2
+5V
3/4W 8 8K 2%
CFA0
CFA1
CFA2
CFA3
CFA4
CFA5
CFA6
BT0
BT1
BT2
BT3
AP
CSCYC
SMLN-
SET0
SET1
SET2
SET3
SET4
SET5
SET6
SET7
SET8
SET9
SET10
TAG0
TAG1
TAG2
TAG3
TAG4
TAG5
TAG6
TAG7
TAG8
TAG9
TAG10
TAG11
ADS-
W/R-
D/C-
M/IO-
PCD
PWT
LOCK-
SYSC
WB/WT-
CACHE-
CLK
TDI
TMS
TCK
HITM-
BRDY-
RESET
CRDY-
FLUSH-
BGT-
CNA-
SWEND-
KWEND-
PCYC
SYNC-
MKEN-
MRO-
MWB/WT-
DRCTM-
MAOE-
MBAOE-
MALE
MBALE
SNPINV
SNPNCA
SNPSTB-
SNPCLK
CCACHE-
TRST-
MAP
F16 R17
C15 P15
E7 S5
C3 R7
B17 Q16
F15 Q15
E15 Q14
A16 U16
A14 U14
A12 U12
A10 U10
A8
E1
D7 R16
E14 Q13
D14 Q12
D15 R15
C13 S17
D13 R14
D12 S18
E13 T18
E10 Q11
E11 R13
C10 R12
B10 R11
D9 Q10
B5 R10
E9 S15
B4 R9
C5 S16
C4 T15
D8 T16
B3 Q9
E8 T17
B2 R8
C2 Q8
D6 B18
C16 K16
C18 E17
J15 E16
G17 K18
J16 D17
D18 G15
C17 Q17
G16 Q18
L15 R18
H15 M16
E12 N15
P4 P16
Q3 N16
Q4 M15
L5 D16
E18 F4
Q2 G4
R6 G5
N3 F5
P5 E4
N4 E5
M5 D2
R2 D3
N5 D4
K15 H5
R5 E3
S2 K4
KJ E6
L4
N2 H16
T5 L16
Q7 E2
R3 K5
Q5 J5
Q6 H4
R4 J4
E4 G2
T4 D5
H1 A19
T2 K1
U8 M1
N1
P19
Q1
S1
T3
U1
MCFA0
MCFA1
MCFA2
MCFA3
MCFA4
MCFA5
MCFA6
MBT0
MBT1
MBT2
MBT3
MSET0
MSET1
MSET2
MSET3
MSET4
MSET5
MSET6
MSET7
MSET8
MSET9
MSET10
MTAG0
MTAG1
MTAG2
MTAG3
MTAG4
MTAG5
MTAG6
MTAG7
MTAG8
MTAG9
MTAG10
MTAG11
AHOLD
EADS-
KEN-
BRDYC1-
NA-
BLE-
BRDYC2-
BUS-
MCYC-
MAWEA-
WAY
WBA
WBTYP
WBWE-
WRARR-
BLAST-
CADS-
SNPADS-
CDTS-
CWR-
CDC-
CMIO-
RDYSRC
MCACHE-
KLOCK-
CAHOLD
PALLC-
CWAY
NENE-
BOFF-
INV
FSIOUT-
MHITM-
MTHIT-
SNPCYC-
SNPBSY-
TDO
CPWT
CPCD
MPIC-
BLEC-
IPERR-
APERR-
EWBE-
MAPERR-
221 SMA3R
191 SMA4R
228 MA30
211 MA31
201
200 MA5
199 MA6
277 MBT0
275 MBT1
273 MBT2
271 MBT3
220 MA7
198 MA8
197 MA9
219 MA10
240 MA11
218 MA12
241 MA13
260 MA14
196 MA15
217 MA16
216 MA17
215 MA18
195 MA19
214 MA20
238 MA21
213 MA22
239 MA23
257 MA24
258 MA25
194 MA26
259 MA27
212 MA28
193 MA29
37
142
93
92
144
74
111
202
203
222
162
171
182
172
161
73
99
109
110
100
80
81
59
60
61
120
79
139
82
122
152
78
130
119
129
107
62
136
156
166
185
186
224
245
262
SMA3
SMA4
MA[31:3]
PA[31 3]
VCC
5% 1 33R65 2 1/8W
5% 1 33R67 2 1/8W
5% 2 2K 1/8W R41
5% 2 2K 1/8W R39
5% 2 2K 1/8W R33
5% 2 2K 1/8W R30
CADSR- 5% 1 22 2 CADS-
CDTSR- 5% 1 22 2 CDTS-
MCACHER- 5% 1 22 2 MCACHE-
NENER- 5% 1 22 2 NENE-
MODIFIEDR- 5% 1 22 2 MODIFIED
1/8W R64
1/8W R59
1/8W R69
1/8W R66
1/8W R58
BLEC-
EWBE-
BOFF-
INV
SNPCYC-
TDICPU
μ35

241576-B1

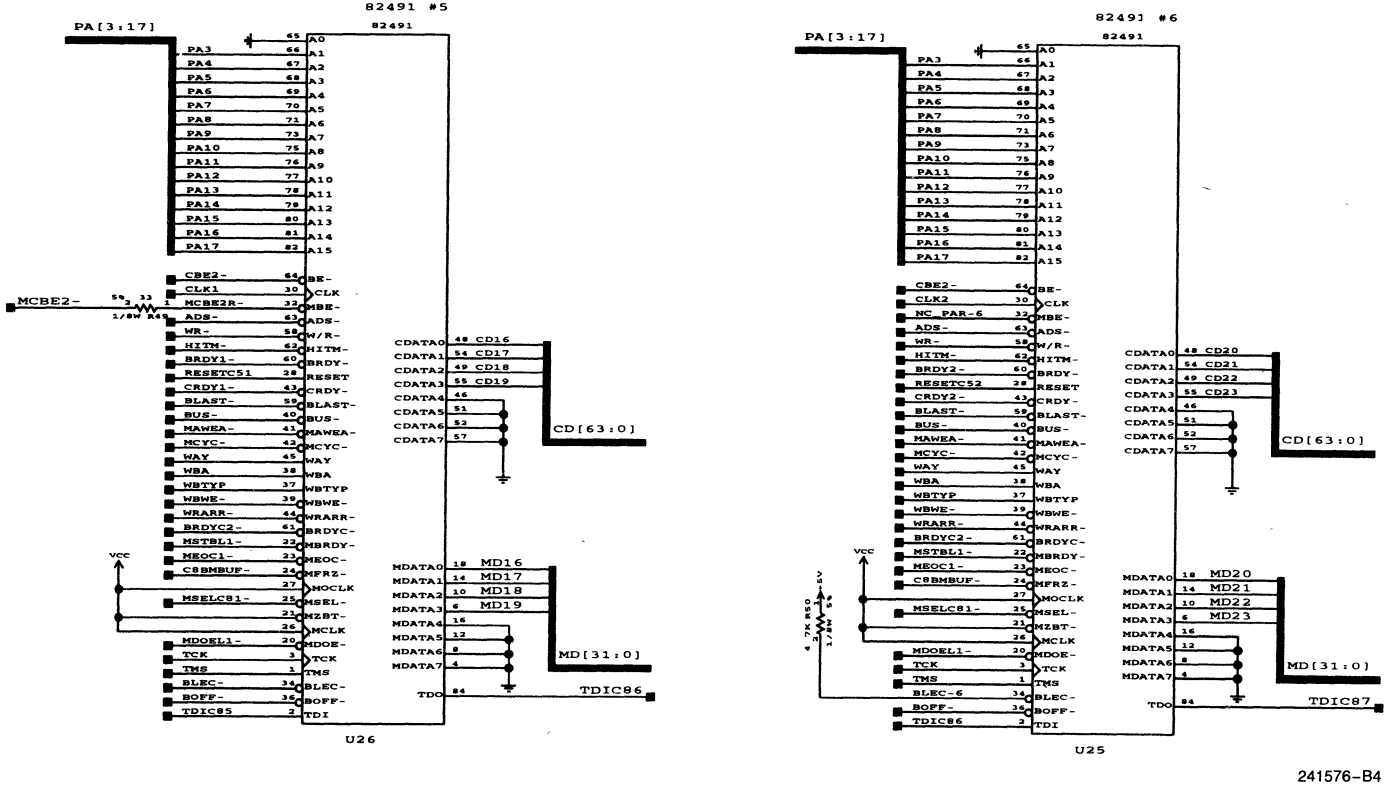
82491 Byte 0


PA[3:17]
82491 #1
82491
MCBE0-
CBE0-
CLK1
MCBE0R-
ADS-
WR-
HITM-
BRDY1-
RESETC51
CRDY1-
BLAST-
BUS-
MAWEA-
MCYC-
WAY
WBA
WBTYP
WBWE-
WRARR-
BRDYC2-
MSTBL1-
MEOC1-
C8BMBUF-
MSELC81-
MDOEL1-
TCK
TMS
BLEC-
BOFF-
TDIC81
CD[63 0]
CDATA0 48 CD0
CDATA1 54 CD1
CDATA2 49 CD2
CDATA3 55 CD3
MD[31 0]
MDATA0 18 MD0
MDATA1 14 MD1
MDATA2 10 MD2
MDATA3 6 MD3
TDO 84 TDIC82
U4
PA[3:17]
82491 #2
82491
CBE0-
CLK2
NC_PAR-2
ADS-
WR-
HITM-
BRDY2-
RESETC52
CRDY2-
BLAST-
BUS-
MAWEA-
MCYC-
WAY
WBA
WBTYP
WBWE-
WRARR-
BRDYC2-
MSTBL1-
MEOC1-
C8BMBUF-
MSELC81-
MDOEL1-
TCK
TMS
BLEC-2
BOFF-
TDIC82
CD[63.0]
CDATA0 48 CD4
CDATA1 54 CD5
CDATA2 49 CD6
CDATA3 55 CD7
MD[31:0]
MDATA0 18 MD4
MDATA1 14 MD5
MDATA2 10 MD6
MDATA3 6 MD7
TDO 84 TDIC83
U3
241576-B2

82491 Byte 1

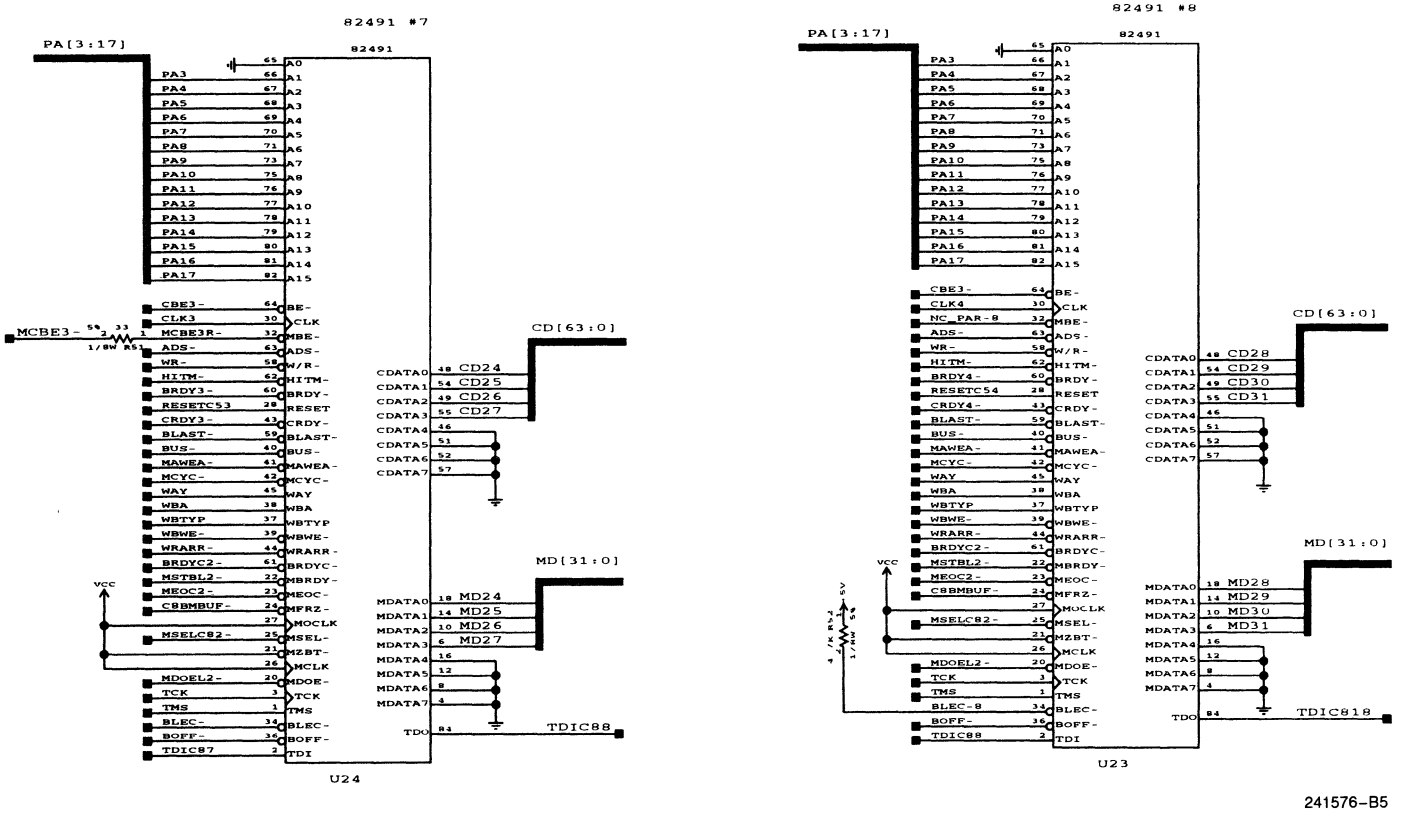

82491 #3
82491
PA[3:17]
MCBE1-
CBE1-
CLK3
MCBE1R-
ADS-
WR-
HITM-
BRDY3-
RESETC53
CRDY3-
BLAST-
BUS-
MAWEA-
MCYC-
WAY
WBA
WBTYP
WBWE-
WRARR-
BRDYC2-
MSTBL2-
MEOC2-
C8BMBUF-
MSELC82-
MDOEL2-
TCK
TMS
BLEC-
BOFF-
TDIC83
CD[63:0]
MD[31:0]
TDO
TDIC84
U2
82491 #4
82491
PA[3:17]
CBE1-
CLK4
NC_PAR-4
ADS-
WR-
HITM-
BRDY4-
RESETC54
CRDY4-
BLAST-
BUS-
MAWEA-
MCYC-
WAY
WBA
WBTYP
WBWE-
WRARR-
BRDYC2-
MSTBL2-
MEOC2-
C8BMBUF-
MSELC82-
MDOEL2-
TCK
TMS
BLEC-4
BOFF-
TDIC84
CD[63:0]
MD[31:0]
TDO
TDIC817
U1
VCC
241576-B3

82491 Byte 2



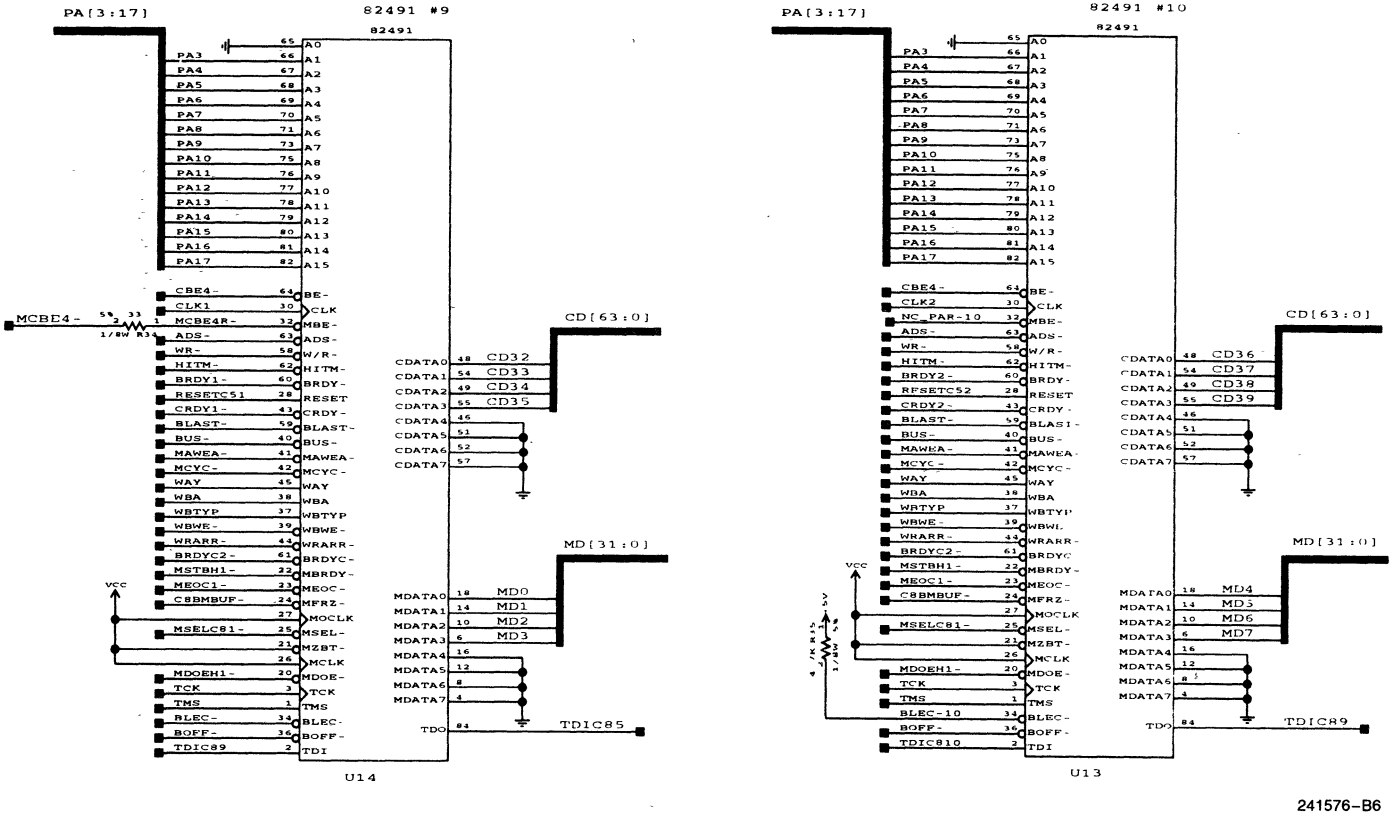
241576-B4

82491 Byte 3



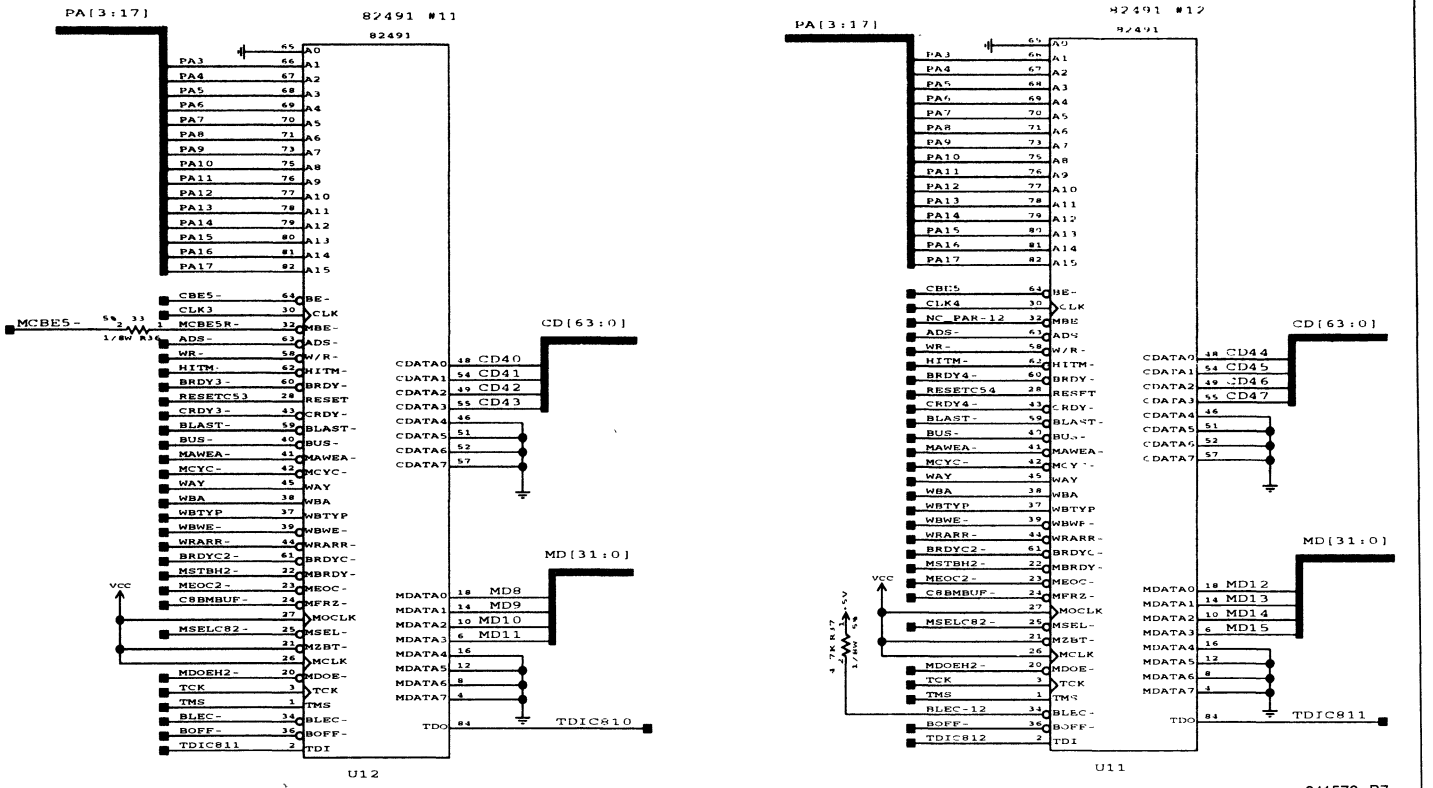
241576-B5

82491 Byte 4

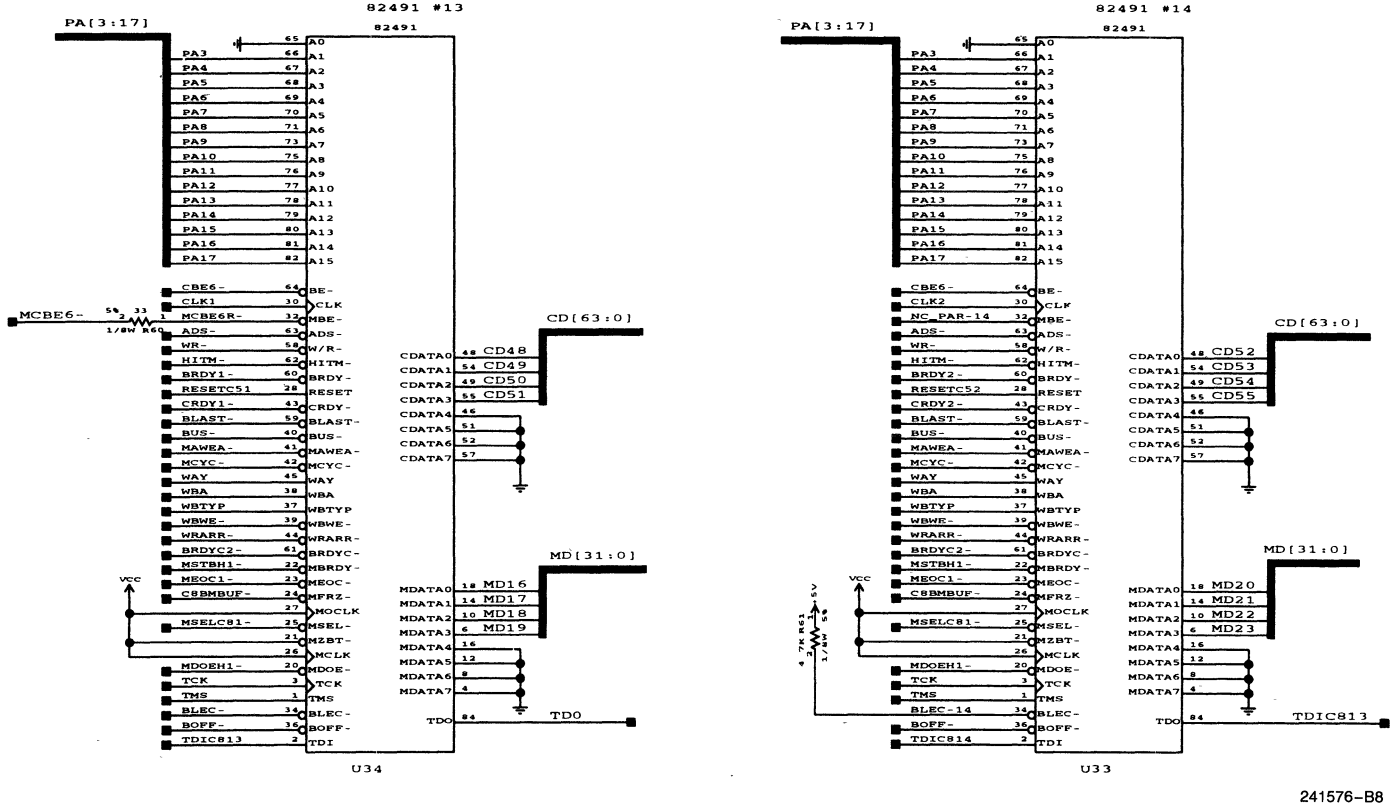


241576-B6

82491 Byte 5

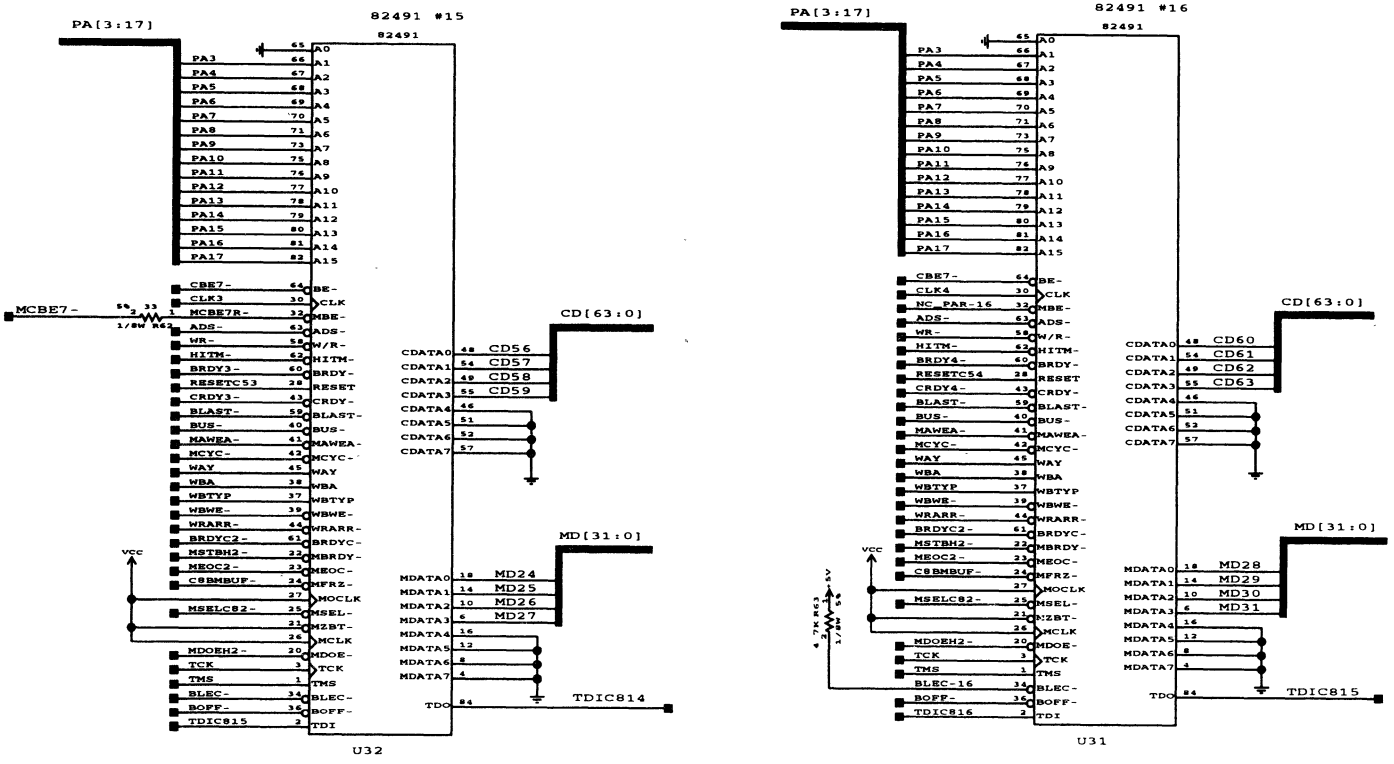

82491 #11
82491
PA[3:17]
MCBE5-
CD[63:0]
MD[31:0]
VCC
MSELC82-
MDOEH2-
TCK
TMS
BLEC-
BOFF-
TDIC811
TDO
TDIC810
U12
82491 #12
82491
PA[3:17]
CD[63:0]
MD[31:0]
VCC
MSELC82-
MDOEH2-
TCK
TMS
BLEC-12
BOFF-
TDIC812
TDI
TDO
TDIC811
U11
241576-B7

82491 Byte 6

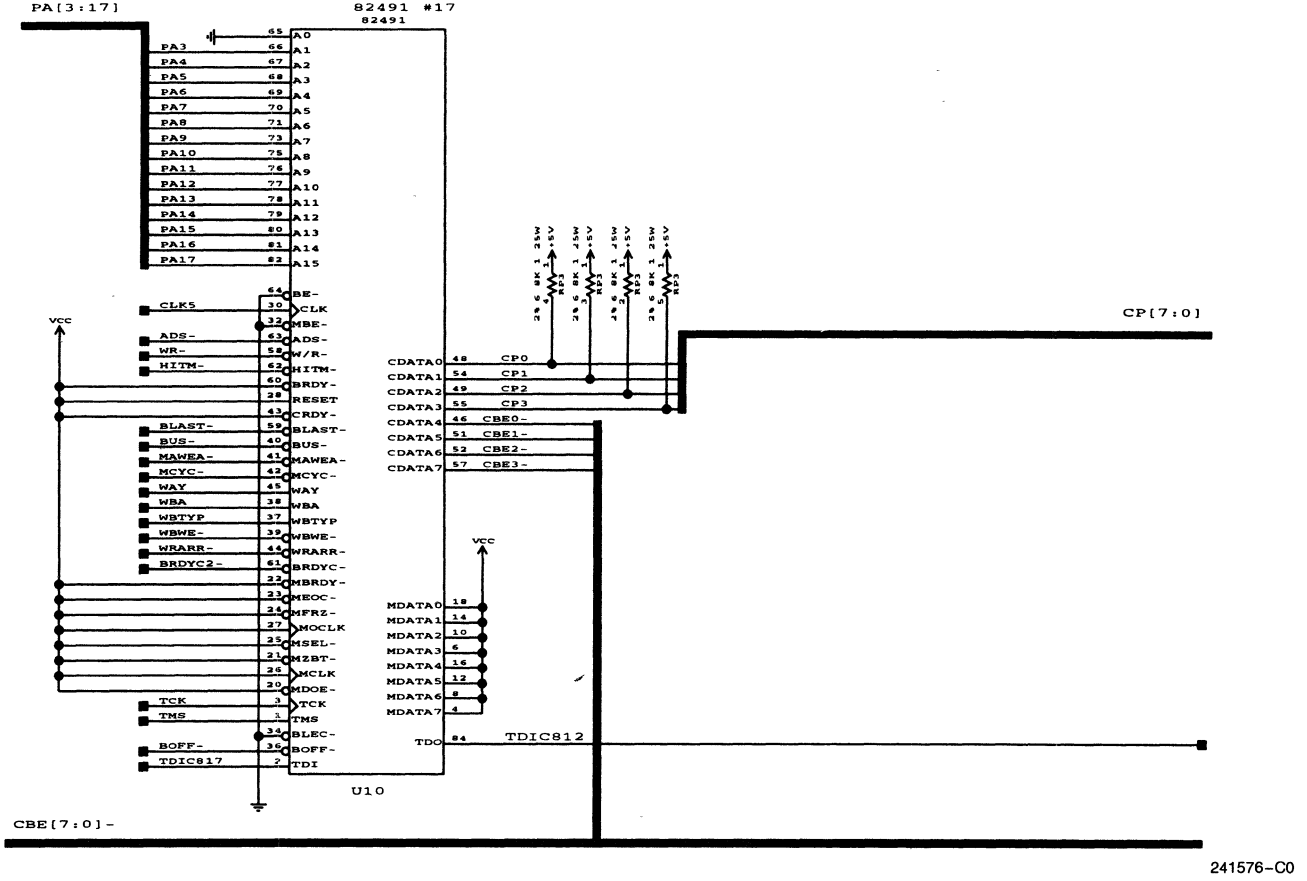


241576-B8

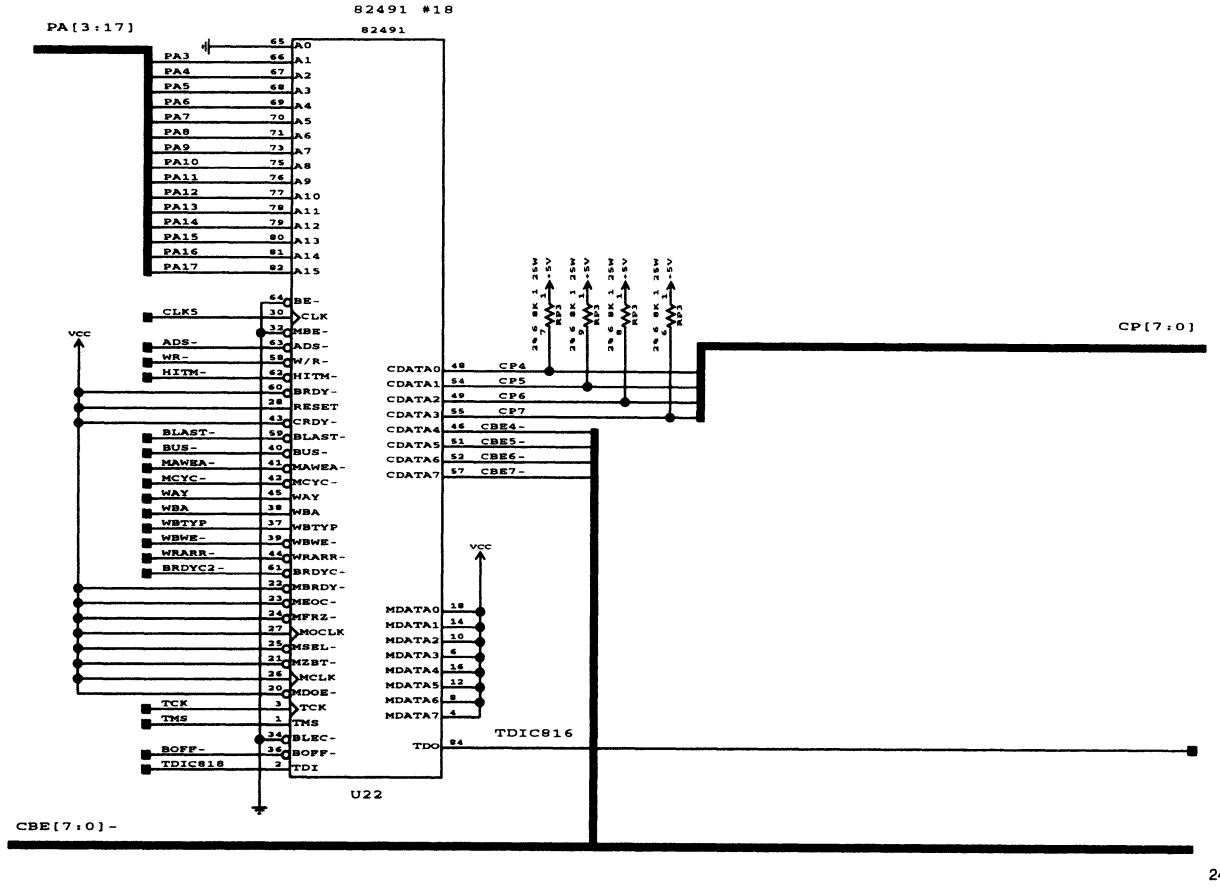
82491 Byte 7


82491 #15
82491
PA[3:17]
MCBE7-
CD[63:0]
MD[31:0]
TDIC814
U32
82491 #16
82491
PA[3:17]
CD[63:0]
MD[31:0]
TDIC815
U31
241576-B9

82491 Parity 0–3



82491 Parity 4–7



8.6.2 I/O MODEL FILES

All electrical I/O simulations were performed using TLC V4.1.13 from Quad Design Technology, Inc. The simulations were performed at the fast and slow corners to verify all signal quality and flight time specifications are met. The files used for these simulations are available from Intel as described above. These files include the topology, model, and control files needed to run the simulations for all nets in the optimized interface.

8.6.3 BOARD FILES

The board files for the design example were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel as described above. These files may be used to import the design example into a specific system design. Note: some changes to the layout and nets may be necessary to complete importing these files into a specific system design.

8.6.4 BILL OF MATERIALS

The bill of materials file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.5 PHOTOPLOT LOG

The photoplot log file was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.6 NETLIST REPORT

The netlist report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.7 PLACED COMPONENT REPORT

The placed component report was created using Allegro V4.2 from Cadence Design Systems, Inc. The file is available from Intel as described above.

8.6.8 ARTWORK FOR EACH BOARD LAYER

The artwork for the six board layers were created using Allegro V4.2 from Cadence Design Systems, Inc. The files are available from Intel in a Gerber format as described above.

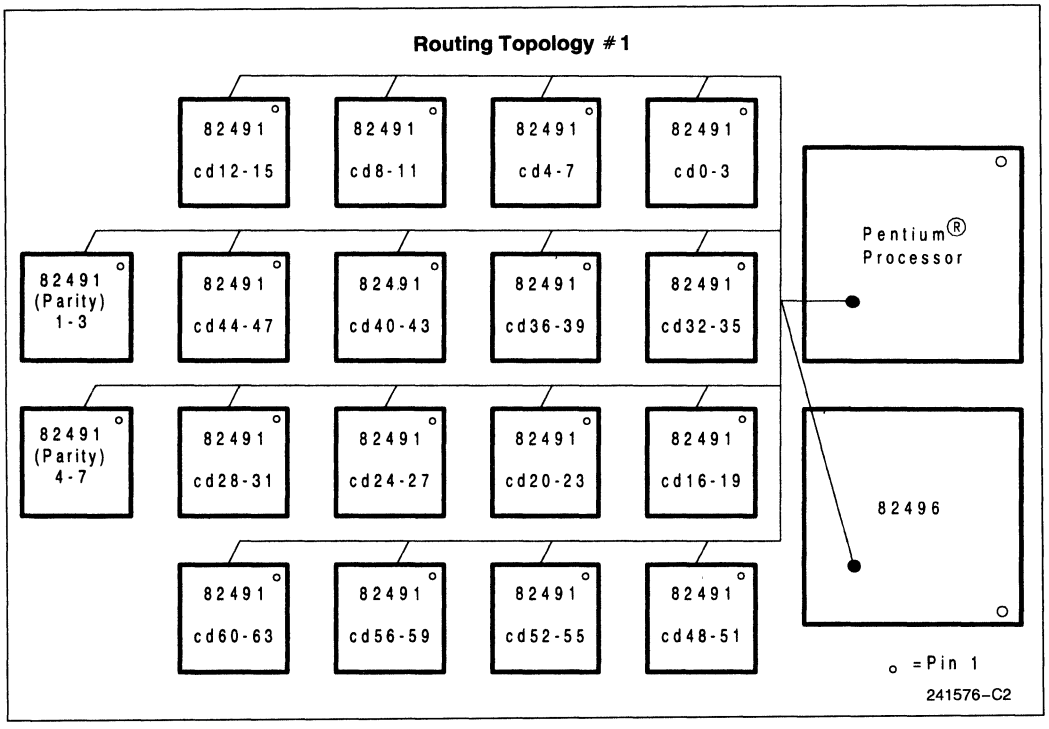
8.6.9 TRACE SEGMENT LINE LENGTHS

Sections 8.6.9.1 to 8.6.9.7 list the segment line lengths for each net of the optimized interface. All lengths are provide in mils (1/1000 inch). The stubs listed in the following tables are associated with the pin escapes required for the 82491s.

The following abbreviations have been used to represent the different devices:

PP = Pentium processor
CC = 82496 cache controller
CS = 82491 cache SRAM

8.6.9.1 Low Addresses and Pentium® Processor Control



NET	PP-Tee	CC-Tee	Tee-CS#5	Tee-CS#9
PA3	1211	4398	614.9	634.9
PA4	771.3	4403	620.8	634.9
PA5	920.1	4402	577.8	637.8
PA6	1026	4405	577.8	634.9
PA7	1132	4443	569.1	642
PA8	1032	4410	563.2	640.8
PA9	1122	4449	576.7	606.6
PA10	1000	4478	582.5	597.8
PA11	1060	4332	585.8	629.9
PA12	1019	4393	585.8	635.8
PA13	1077	4434	597.8	637.8
PA14	968.3	4653	689.1	743.2
PA15	1233	4402	524.2	580.8
PA16	1092	4384	597.8	614.9
PA17	1186	4396	600.8	602.3
HITM#	3575	4409	430.7	530.7
W/R#	3913	4397	674.9	676.6

NET	CS#9–CS#1	CS#1–CS#2	CS#2–CS#3	CS#3–CS#4	CS#9–CS#10	CS#10–CS#11	CS#11–CS#12	CS#12–CS#17
PA3	1111	944.9	936.6	961.4	1124	936.6	944.9	936.6
PA4	1210	964.9	936.6	975.6	1207	936.6	961.9	936.6
PA5	1131	944.9	936.6	961.4	1104	936.6	944.9	936.6
PA6	1165	936.6	936.6	936.6	1177	936.6	936.6	936.6
PA7	1170	936.6	936.6	936.6	1157	936.6	936.6	936.6
PA8	1170	936.6	936.6	936.6	1157	936.6	936.6	936.6
PA9	1224	936.6	936.6	936.6	1217	936.6	936.6	936.6
PA10	1211	936.6	936.6	936.6	1207	936.6	936.6	936.6
PA11	1264	936.6	936.6	936.6	1267	936.6	936.6	936.6
PA12	1264	936.6	936.6	936.6	1267	936.6	936.6	936.6
PA13	1293	936.6	936.6	936.6	1297	936.6	936.6	936.6
PA14	1295	936.6	936.6	936.6	1297	936.6	936.6	936.6
PA15	1312	936.6	936.6	936.6	1317	936.6	936.6	936.6
PA16	1309	936.6	936.6	936.6	1317	936.6	936.6	936.6
PA17	1297	936.6	936.6	936.6	1287	936.6	936.6	936.6
HITM#	1141	953.1	953.1	953.1	1147	936.6	996.9	944.9
W/R#	1193	953.1	953.1	1023	1192	936.6	1003	953.1

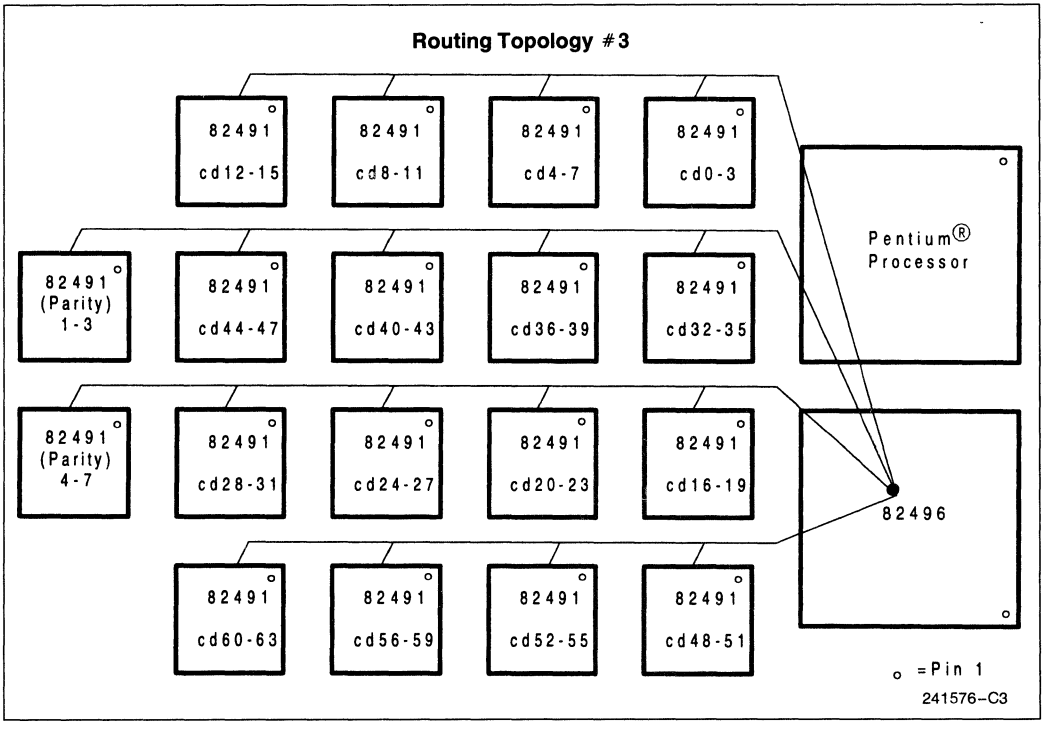
NET	CS#5-CS#6	CS#6-7 CS#	CS#7-CS#8	CS#8-CS#18	CS#5-CS#13	CS#13-CS#14	CS#14-CS#15	CS#15-CS#16	Stubs
PA3	1129	961.4	953.1	936.6	1122	944.9	936.6	944.9	75.0- 75.0
PA4	1207	969.7	967.3	944.9	1185	959	936.6	964.9	135.4- 135.4
PA5	1136	961.4	953.1	936.6	1145	944.9	936.6	944.9	75.0- 75.0
PA6	1190	936.6	936.6	936.6	1179	936.6	936.6	936.6	135.4- 135.4
PA7	1176	936.6	936.6	936.6	1179	936.6	936.6	936.6	75.0- 75.0
PA8	1177	936.6	936.6	936.6	1176	936.6	936.6	936.6	135.4- 135.4
PA9	1217	936.6	936.6	936.6	1213	936.6	936.6	936.6	135.4- 135.4
PA10	1207	936.6	936.6	936.6	1208	936.6	936.6	936.6	135.4 -135.4
PA11	1267	936.6	936.6	936.6	1261	936.6	936.6	936.6	75.0- 75.0
PA12	1267	936.6	936.6	936.6	1261	936.6	936.6	936.6	135.4 -135.4
PA13	1297	936.6	936.6	936.6	1281	936.6	936.6	936.6	75.0- 75.0
PA14	1297	936.6	936.6	936.6	1284	936.6	936.6	936.6	135.4 -135.4
PA15	1317	936.6	936.6	936.6	1318	936.6	936.6	936.6	75.0- 75.0
PA16	1317	936.6	936.6	936.6	1315	936.6	936.6	936.6	135.4 -135.4
PA17	1287	936.6	936.6	936.6	1288	936.6	936.6	936.6	75.0-75.0
HITM#	1146	944.9	944.9	964.9	1138	944.9	936.6	936.6	95.4 -95.4
W/R#	1197	975.6	961.4	953.1	1193	953.1	936.6	936.6	95.4 -95.4

NET	CC-Tee	PP-Tee	Tee-CS#5	Tee-CS#9	CS#9-CS#1	CS#1-CS#2	CS#2-CS#3	CS#3-CS#4
BOFF#	2405.6	4401.9	919.7	925.8	936.6	936.6	936.6	936.6

NET	CS#9-CS#10	CS#10-CS#11	CS#11-CS#12	CS#12-CS#17	CS#5-CS#6	CS#6-CS#	CS#7-CS#8	CS#8-CS#18
BOFF#	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6

NET	CS#5-CS#13	CS#13-CS#14	CS#14-CS#15	CS#15-CS#16	Stubs
BOFF#	944.9	936.6	936.6	936.6	75.0-75.0

8.6.9.2 82496 Control

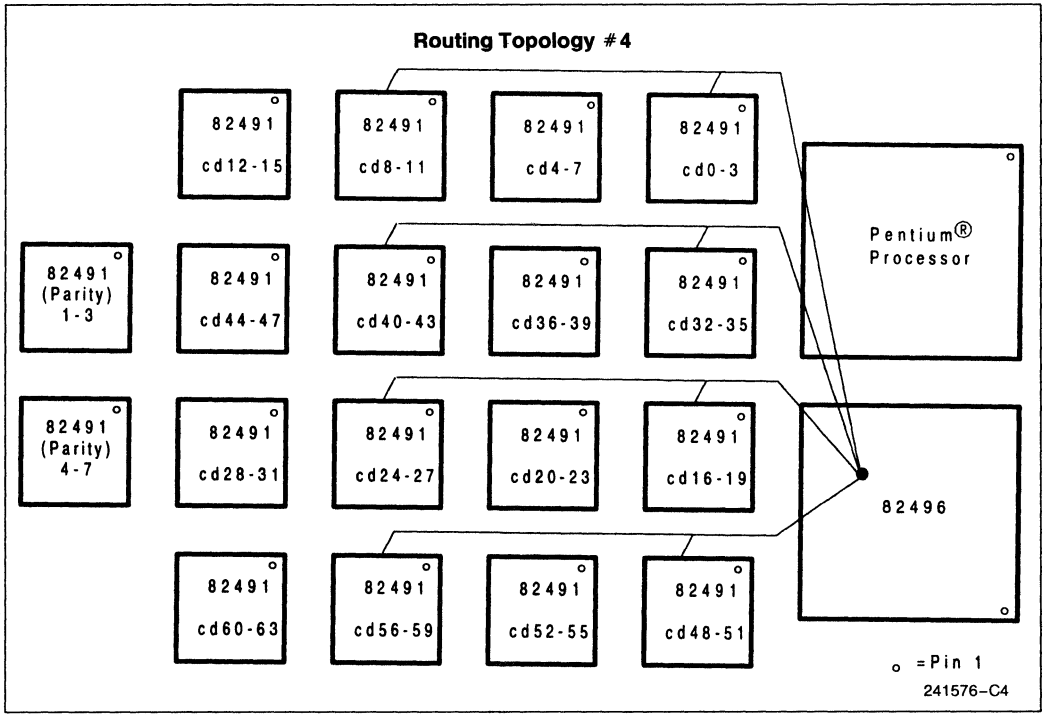


NET	CC–CS#1	CC–CS#9	CC–CS#5	CC–CS#13	CS#1–CS#2	CS#2–CS#3	CS#3–CS#4	CS#9–CS#10
BLAST#	4225	3085	3089	4186	961.4	967.3	961.4	1020
BUS#	4318	3216	3210	4311	936.6	936.6	936.6	936.6
MAWEA#	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
WBA	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
WBTYP	4087	2986	2974	4086	936.6	936.6	936.6	936.6
WBWE	4210	3109	3113	4215	936.6	936.6	936.6	936.6

NET	CS#10–CS#11	CS#11–CS#12	CS#12–CS#17	CS#5–CS#6	CS#6–CS#7	CS#7–CS#8	CS#8–CS#18	CS#13–CS#14
BLAST#	1011	969.7	969.7	964.9	936.6	961.4	944.9	936.6
BUS#	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
MAWEA#	4187	3059	3089	4178	936.6	936.6	936.6	936.6
WBA	4846	3748	3680	4845	936.6	936.6	936.6	936.6
WBTYP	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6
WBWE	936.6	936.6	936.6	936.6	936.6	936.6	936.6	936.6

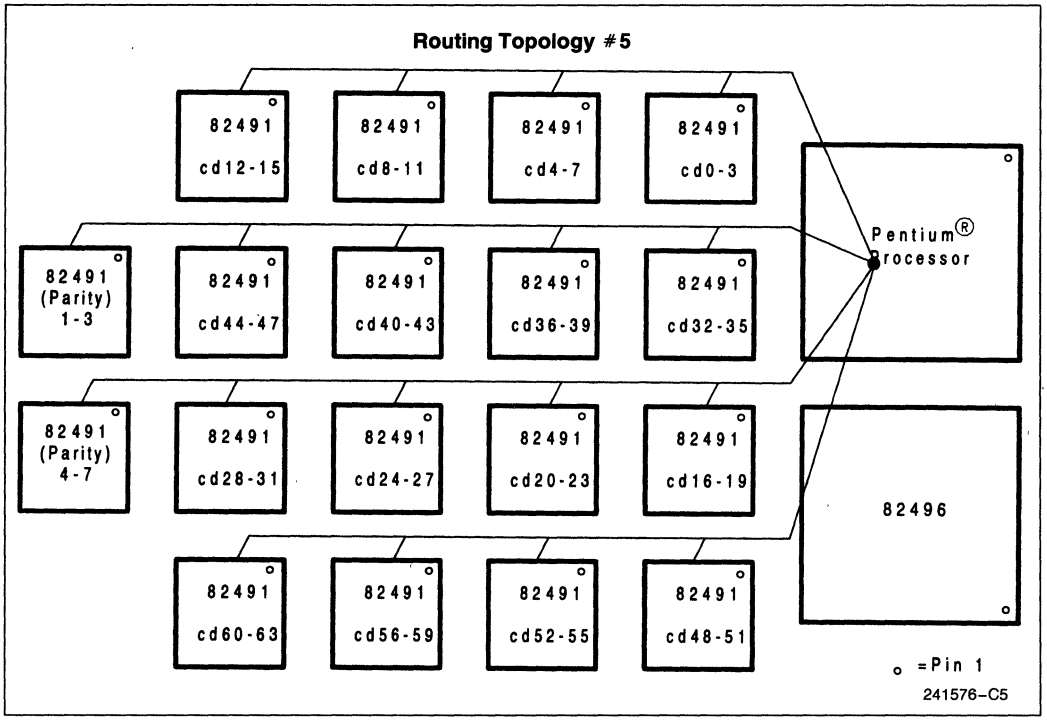
NET	CS#14–CS#15	CS#15–CS#16	Stubs
BLAST#	936.6	944.9	85.0–85.0
BUS#	936.6	936.6	75.0–75.0
MAWEA#	936.6	936.6	135.4–135.4
WBA	936.6	936.6	75.0–75.0
WBTYP	936.6	936.6	135.4–135.4
WBWE	936.6	936.6	135.4–135.4

8.6.9.3 82496 Control



NET	CC–CS#1	CC–CS#9	CC–CS#5	CC–CS#13	CS#1–CS#3	CS#9–CS#11	CS#5–CS#7	CS#13–CS#15	Stubs
BLEC–	4222.8	4219.1	4205.9	4206.5	1856.6	1856.6	1856.6	1856.6	75.0 –75.0

8.6.9.4 Pentium® Processor Control

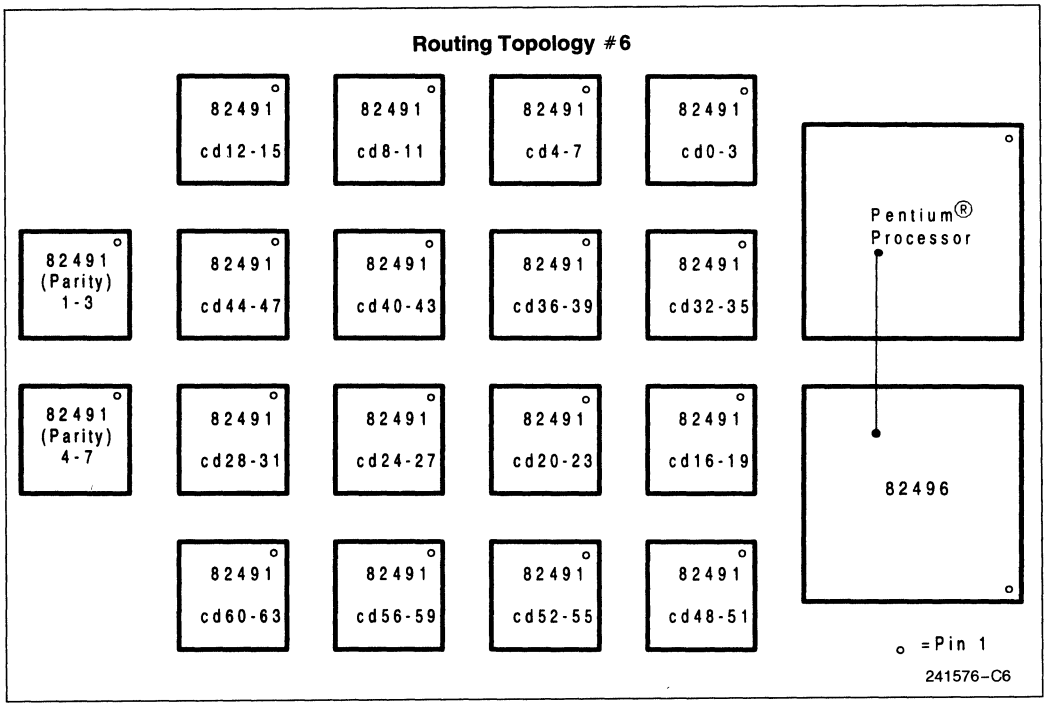


NET	PP-CS#1	PP-CS#9	PP-CS#5	PP-CS#13	CS#1-CS#2	CS#2-CS#3	CS#3-CS#4	CS#9-CS#10	CS#10-CS#11
ADS#	5213.8	4111.6	4108.4	5207.4	936.6	936.6	936.6	964.9	1003.8

NET	CS#11-CS#12	CS#12-CS#17	CS#5-CS#6	CS#6-CS#7	CS#7-CS#8	CS#8-CS#18	CS#13-CS#14	CS#14-CS#15	CS#15-CS#16
ADS#	964.9	978	936.6	936.6	936.6	969	961.9	959	961.9

NET	Stubs
ADS#	75.0–75.0

8.6.9.5 Pentium® Processor and 82496 Control, High Addresses, Pentium® Processor Data



NET	PP-res	res-CC
PA18	797.1	2907.3
PA19	824.3	3440.5
PA20	682.6	2881.1

NET	PP-CC
PA21	3505.2
PA22	4234.7
PA23	3478.8
PA24	4011.7
PA25	3756.6
PA26	3672.9
PA27	3807.5
PA28	4963.9
PA29	3318.8
PA30	4535.8
PA31	4097.2
BT0	3604.5
BT1	2677
BT2	3952.2
BT3	3185.9

NET	PP-CC
ADSC#	3880.6
AP	4556.5
CACHE#	3798.4
D/C#	3647.4
LOCK#	4721.2
M/IO#	5062.1
PCD	3366
PWT	4264.2
SCYC	3798
AHOLD	4604.8
BRDYC1#	3686
EADS#	3656.5
INV	4603.5
KEN#	4144.8
NA#	3770.4
WB/WT#	3493.3
EWBE#	2475

NET	PP-CS#17	Stubs
CP0	7558	135.4
CP1	7685.9	135.4
CP2	7675.3	75
CP3	7307.4	75

NET	PP-CS#5	Stubs
CD16	7121.5	135.4
CD17	6878.5	135.4
CD18	6974.5	75
CD19	7021.5	75

NET	PP-CS#11	Stubs
CD40	692.8	135.4
CD41	7567.4	135.4
CD42	6922.3	75
CD43	7613.3	75

NET	PP-CS#18	Stubs
CD4	7641.3	135.4
CD5	7698.3	135.4
CD6	7416.4	75
CD7	6946.9	75

NET	PP-CS#6	Stubs
CD20	7089.3	135.4
CD21	6948.1	135.4
CD22	7064.9	75
CD23	6927.4	75

NET	PP-CS#12	Stubs
CD44	7018	95.4
CD45	6997.1	135.4
CD46	6956.1	75
CD47	7491.6	75

NET	PP-CS#1	Stubs
CD0	6920.6	135.4
CD1	6964.1	135.4
CD2	7130.1	75
CD3	6910.9	75

NET	PP-CS#7	Stubs
CD24	6968.1	107.1
CD25	6877.9	135.4
CD26	7040.2	75
CD27	6891.7	75

NET	PP-CS#13	Stubs
CD48	6961	135.4
CD49	6884.2	135.4
CD50	7072.2	75
CD51	6968.3	75

NET	PP-CS#2	Stubs
CD4	7037.3	135.4
CD5	6869.4	135.4
CD6	6925	75
CD7	6886.5	75

NET	PP-CS#31	Stubs
CD28	7688.3	135.4
CD29	7872	135.4
CD30	7395.9	75
CD31	7433.3	75

NET	PP-CS#1	Stubs
CD52	7277.5	135.4
CD53	6979.9	135.4
CD54	6921.7	75
CD55	6920.2	75

NET	PP-CS#1	Stubs
CD8	6945.5	135.4
CD9	7448.5	135.4
CD10	7790.2	75
CD11	6924.1	75

NET	PP-CS#9	Stubs
CD32	7586.8	135.4
CD33	7271.6	135.4
CD34	7424.6	75
CD35	6944.1	75

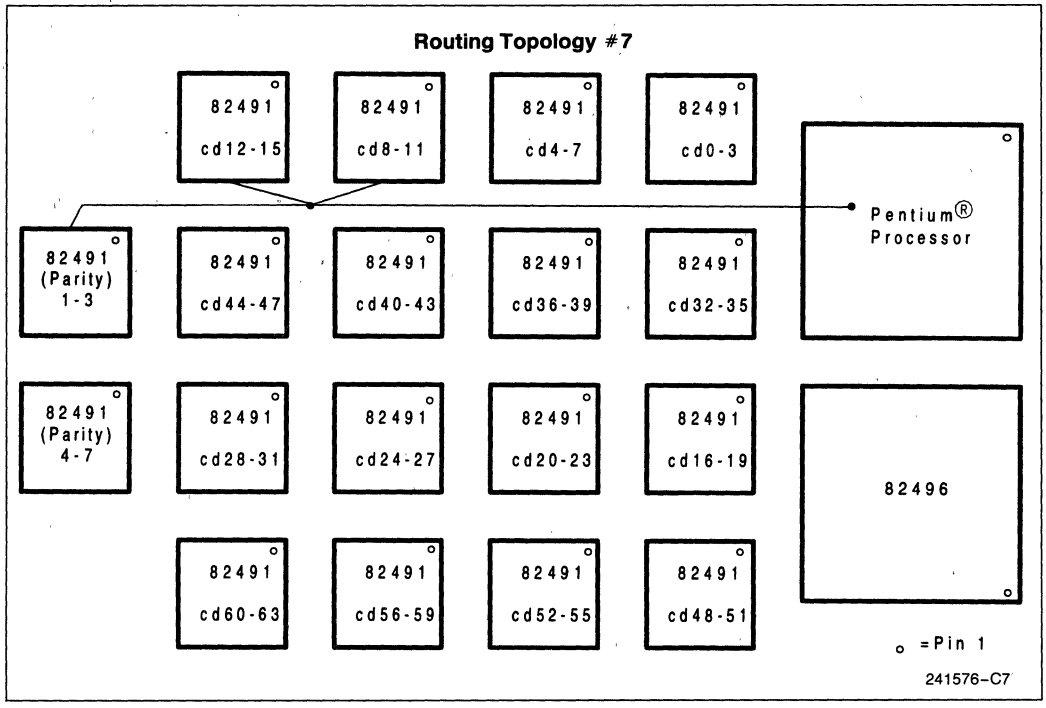
NET	PP-CS#15	Stubs
CD56	6777.1	135.4
CD57	6997.1	135.4
CD58	6881.1	75
CD59	7130.3	75

NET	PP-CS#4	Stubs
CD12	7275.6	135.4
CD13	7344.7	135.4
CD14	7692.1	75
CD15	7438.1	75

NET	PP-CS#10	Stubs
CD36	7343	135.4
CD37	6952.2	135.4
CD38	7520	75
CD39	7403	75

NET	PP-CS#16	Stubs
CD60	6919.5	135.4
CD61	6971.3	135.4
CD62	7274.5	75
CD63	7019.4	75

8.6.9.6 Byte Enables



NET	PP-Tee	Tee-CS# 1	Tee-CS# 2	Tee-CS#17	Stubs
CBE0#	1552.2	708.5	707.3	4813.3	75.0–135.4

NET	PP-Tee	Tee-CS# 3	Tee-CS# 4	Tee-CS#17	Stubs
CBE1#	4358.7	545.6	543.1	2627.4	75.0–75.0

NET	PP-Tee	Tee-CS# 5	Tee-CS# 6	Tee-CS#17	Stubs
CBE2#	3476.4	562.1	563.1	4330.5	75.0–135.4

NET	PP-Tee	Tee-CS# 7	Tee-CS# 8	Tee-CS#17	Stubs
CBE3#	5352.9	537.3	440.7	3011.3	75.0–75.0

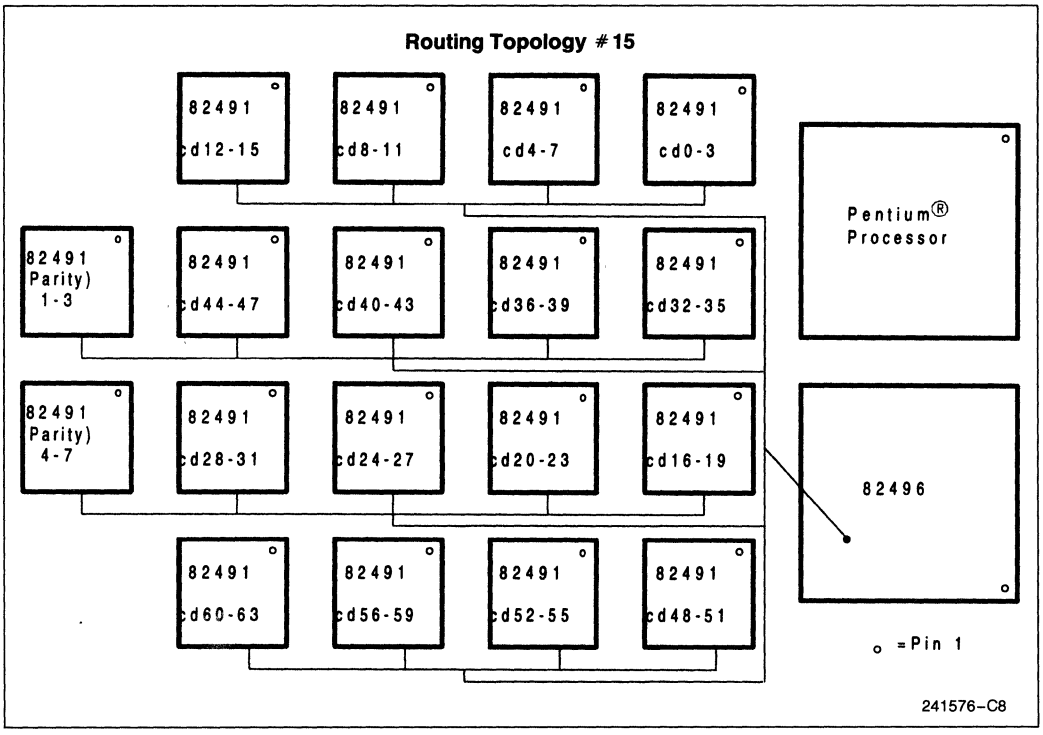
NET	PP-Tee	Tee-CS# 9	Tee-CS#10	Tee-CS#18	Stubs
CBE4#	2709.8	520.7	440.7	4762.9	75.0–135.4

NET	PP-Tee	Tee-CS#11	Tee-CS#12	Tee-CS#18	Stubs
CBE5#	5035.3	520.7	440.7	3065.4	75.0–75.0

NET	PP-Tee	Tee-CS#13	Tee-CS#14	Tee-CS#18	Stubs
CBE6#	4215.7	532.4	440.7	4338.7	75.0–95.4

NET	PP-Tee	Tee-CS#15	Tee-CS#16	Tee-CS#18	Stubs
CBE7#	6132.2	520.7	440.7	2225.6	75.0–105.7

8.6.9.7 82496 Control



NET	CC-Tee	Tee-CS#9	Tee to CS#5	CS#5 to CS#13	CS#1 to CS#9	CS#1-CS#2	CS#2-CS#3	CS#3-CS#4	CS#9-CS#10	CS#10-CS#11
BRDYC2	2077	598	595	935	932	940	940	940	540	940
MCYC	1707	540	527	1038	1041	940	940	940	1047	940
WRARR#	1820	573	576	976	976	940	940	940	940	940
WAY	1969	749	747	944	944	940	940	940	940	940

NET	CS#11–CS#12	CS#12–CS#17	CS#5–CS#6	CS#6–CS#7	CS#7–CS#8	CS#8–CS#18	CS#13–CS#14	CS#14–CS#15	CS#15–CS#16	Stubs
BRDYC2	940	940	940	940	940	940	940	940	940	85
MCYC	940	940	1047	940	940	940	940	940	940	75
WRARR#	940	940	940	940	940	940	940	940	940	135
WAY	940	940	940	940	940	940	940	940	940	75

References

1. Intel Corporation, *Pentium® Processor Family Developer's Manual*, Order#: 241563.
2. Intel Corporation, *Pentium® Processor Thermal Design Guide*, Application Note: AP-480, Order# 241575.
3. Intel Corporation, *Pentium® Processor Clock Design*, Application Note: AP-479, Order# 241574.
4. Blood, William R., Jr., *MECL System Design Handbook*, 1988, Motorola Inc.
5. T. Rahal-Arabi and R. Suarez-Gartner, "*An Efficient Methodology for the Design of Optimum Electrical Performance of Interconnects in High Performance Systems*," IEEE Topical Meeting on Electrical Performance of Electronic Packaging, Tucson AZ, April 1992.
6. Huck, Scott, "*Simulating Intel's Optimized Interface with the Intel486™ DX CPU-Cache Chip-Set*," 1992, Intel Corp.

intel®

AP-485

APPLICATION NOTE

Intel Processor Identification with the CPUID Instruction

November 1995

Order Number: 241618-003

INTEL PROCESSOR IDENTIFICATION WITH THE CPUID INSTRUCTION

CONTENTS

CONTENTS

1.0 INTRODUCTION

As the Intel Architecture evolves, with the addition of new generations and models of processors (8086, 8088, Intel 286, Intel386™, Intel486™, and Pentium® processors), it is essential that Intel provides an increasingly sophisticated means with which software can identify the features available on each processor. This identification mechanism has evolved in conjunction with the Intel Architecture as follows:

- Originally, Intel published code sequences that could detect minor implementation differences to identify processor generations.
- Later, with the advent of the Intel386 processor, Intel implemented processor signature identification, which provided the processor family, model, and stepping numbers to software at reset.
- As the Intel Architecture evolved, Intel extended the processor signature identification into the CPUID instruction. The CPUID instruction not only provides the processor signature, but also provides information about the features supported by and implemented on the Intel processor.

The evolution of processor identification was necessary because, as the Intel Architecture proliferates, the computing market must be able to tune processor functionality across processor generations and models that have differing sets of features. Anticipating that this trend will continue with future processor generations, the Intel Architecture implementation of the CPUID instruction is extensible.

This Application Note explains how to use the CPUID instruction in software applications, BIOS implementations, and tools. By taking advantage of the CPUID instruction, software developers can create software applications and tools that can execute compatibly across the widest range of Intel processor generations and models, past, present, and future.

1.1 Update Support

New Intel processor signature and feature bits information can be obtained from the user's manual, programmer's reference manual or appropriate documentation for a processor. In addition, Intel can provide you with updated versions of the programming examples included in this application note; contact your Intel representative for more information.

2.0 DETECTING THE CPUID INSTRUCTION

Intel has provided a straightforward method for detecting whether the CPUID instruction is available. This method uses the ID flag in bit 21 of the EFLAGS register. If software can change the value of this flag, the CPUID instruction is available. The program examples at the end of this Application Note show how to use the PUSHFD instruction to change the value of the ID flag.

3.0 OUTPUTS OF THE CPUID INSTRUCTION

Figure 1 summarizes the outputs of the CPUID instruction.

The CPUID instruction can be executed multiple times, each time with a different parameter value in the EAX register. The output depends on the value in the EAX register, as specified in Table 1. To determine the highest acceptable value in the EAX register, the program should set the EAX register parameter value to 0. In this case, the CPUID instruction returns the highest value that can be recognized in the EAX register. CPUID instruction execution should always use a parameter value that is less than or equal to this highest returned value. Currently, the highest value recognized by the CPUID instruction is 1. Future processors might recognize higher values.

The processor type, specified in bits 12 and 13, indicate whether the processor is an original OEM processor, an OverDrive® processor, or is a dual processor (capable of being used in a dual processor system). Table 2 shows the processor type values that can be returned in bits 12 and 13 of the EAX register.

While any imitator of the Intel Architecture can provide the CPUID instruction, no imitator can legitimately claim that its part is a genuine Intel part. Therefore, the presence of the `GenuineIntel` string is an assurance that the CPUID instruction and the processor signature are implemented as described in this document.

3.1 Vendor-ID String

If the EAX register contains a value of 0, the vendor identification string is returned in the EBX, EDX, and ECX registers. These registers contain the ASCII string `GenuineIntel`.

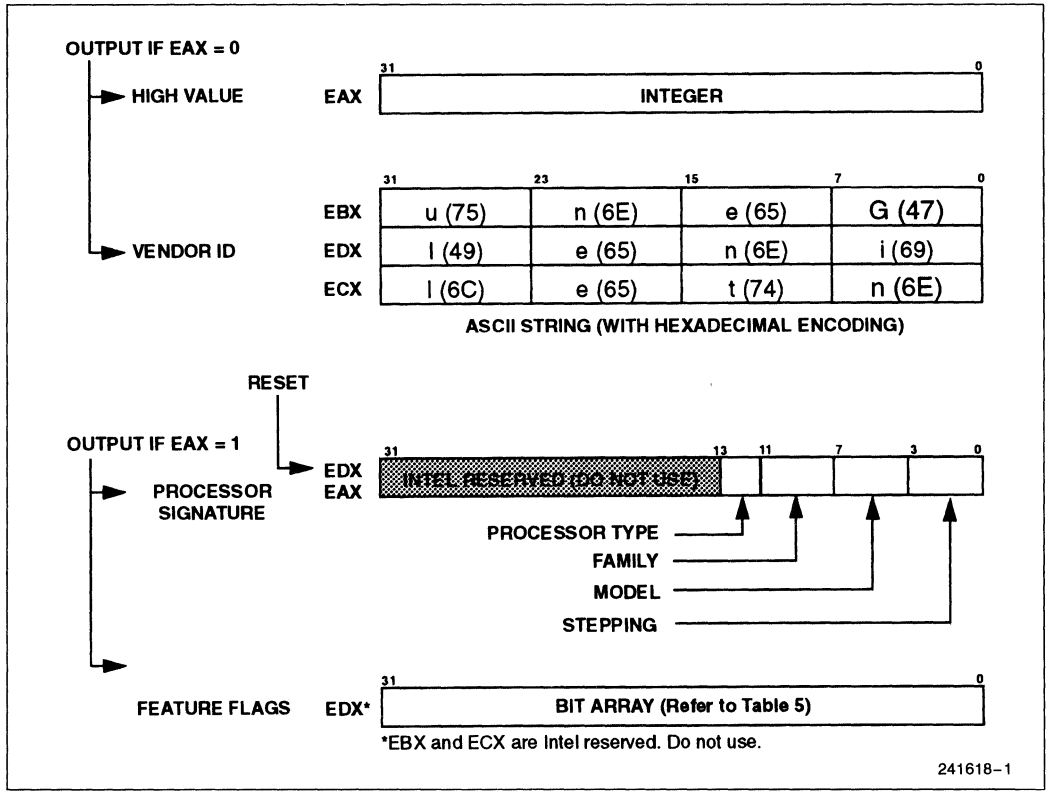


Figure 1. CPUID Instruction Outputs

Table 1. Effects of EAX Contents on CPUID Instruction Output

Parameter	Outputs of CPUID
EAX = 0	EAX ← Highest value recognized
	EBX:EDX:ECX ← Vendor identification string
EAX = 1	EAX ← Processor signature
	EDX ← Feature flags
	EBX:ECX ← Intel reserved (Do not use.)
1 < EAX ≤ highest value	Currently undefined
EAX > highest value	EAX:EBX:ECX:EDX ← Undefined (Do not use.)

Table 2. Processor Type

Bit Position	Value	Description
13,12	00	Original OEM Processor
	01	OverDrive® Processor
	10	Dual Processor(1)
	11	Intel reserved (Do not use.)

NOTE:
1. Not applicable to Intel386 and Intel486 processors.

3.2 Processor Signature

Beginning with the Intel386 processor family, the processor signature has been available at reset. With processors that implement the CPUID instruction, the processor signature is available both upon reset and upon execution of the CPUID instruction. Figure 1 shows the format of the signature for the Intel486 and Pentium processor families. Table 3 shows the values that are currently defined. (The high-order 18 bits are undefined and reserved.)

Older versions of Intel486 SX, Intel486 DX and IntelDX2 processors do not support the CPUID instruction. Therefore, the processor signature is only available upon reset for these processors. Refer to the programming examples at the end of this Application Note to determine which processors support the CPUID instruction.

On Intel386 processors, the format of the processor signature is somewhat different, as Figure 2 shows. Table 4 gives the current values.

Table 3. Intel486™ and Pentium® Processor Signatures

Family	Model	Stepping(1)	Description
0100	0000 and 0001	xxxx	Intel486™ DX Processors
0100	0010	xxxx	Intel486 SX Processors
0100	0011	xxxx	Intel487™ Processors(2)
0100	0011	xxxx	IntelDX2™ and Intel DX2 OverDrive® Processors
0100	0100	xxxx	Intel486 SL Processor(2)
0100	0101	xxxx	IntelSX2™ Processors
0100	0111	xxxx	Write-Back Enhanced IntelDX2 Processors
0100	1000	xxxx	IntelDX4™ and IntelDX4 OverDrive Processors
0101	0001	xxxx	Pentium® Processors (510\60, 567\66)
0101	0010	xxxx	Pentium Processors (735\90, 815\100)
0101	0011	xxxx	Pentium OverDrive Processors
0101	0101	xxxx	Reserved for Pentium OverDrive Processor for IntelDX4 Processor
0101	0010	xxxx	Reserved for Pentium OverDrive Processor for Pentium Processor (510\60, 567\66)
0101	0100	xxxx	Reserved for Pentium OverDrive Processor for Pentium Processor (735\90, 815\100)

NOTES:
1. Intel releases information about stepping numbers as needed.
2. This processor does not implement the CPUID instruction.

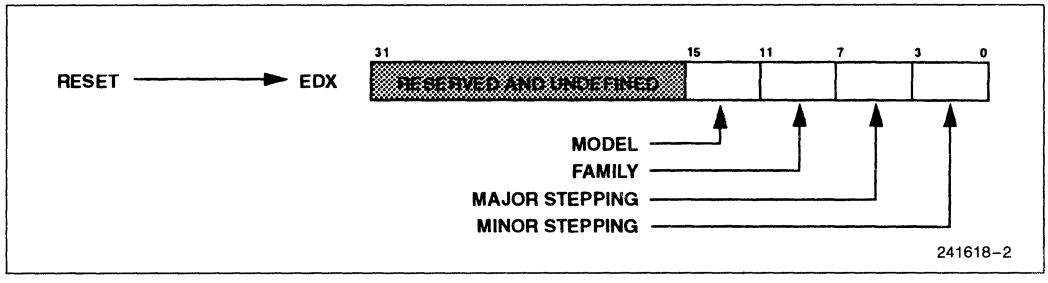


Figure 2. Processor Signature Format on Intel386™ Processors

Table 4. Intel386™ Processor Signatures

Model	Family	Major Stepping	Minor Stepping(1)	Description
0000	0011	0000	xxxx	Intel386™ DX Processor
0010	0011	0000	xxxx	Intel386 SX Processor
0010	0011	0000	xxxx	Intel386 CX Processor
0010	0011	0000	xxxx	Intel386 EX Processor
0100	0011	0000 and 0001	xxxx	Intel386 SL Processor
0000	0011	0100	xxxx	RAPIDCAD™ Coprocessor

NOTE:
1. Intel releases information about minor stepping numbers as needed.

3.3 Feature Flags

When a value of 1 is placed in the EAX register, the CPUID instruction loads the EDX register with the feature flags. The feature flags indicate which features the processor supports. A value of 1 in a feature flag can indicate that a feature is either supported or not supported, depending on the implementation of the CPUID instruction for a specific processor. Table 5 lists the currently defined feature flag values. For future processors, refer to the programmer's reference manual, user's manual, or the appropriate documentation for the latest feature flag values.

Developers should use the feature flags in applications to determine which processor features are supported. By using the CPUID feature flags to predetermine processor features, software can detect and avoid incompatibilities that could result if the features are not present.

Table 5. Feature Flag Values

Bit	Name	Description When Flag = 1	Comments
0	FPU	Floating-Point Unit On-Chip	The processor contains an FPU that supports the Intel387 floating-point instruction set.
1	VME	Virtual Mode Extension	The processor supports extensions to virtual-8086 mode.
2(1)			(See note)
3	PSE	Page Size Extension	The processor supports 4-Mbyte pages.
4–6(1)			(See note)
7	MCE	Machine Check	Exception 18 is defined for Pentium processor style machine checks, including CR4.MCE for controlling the feature. This feature does not define the model-specific implementation of the machine-check error logging reporting and processor shutdowns. Machine-check exception handlers may have to depend on processor version to do model-specific processing of the exception or test for the presence of the standard machine-check feature.
8	CX8	CMPXCHG8B	The 8-byte (64-bit) compare and exchange instructions is supported (implicitly locked and atomic).
9	APIC	On-Chip APIC	Indicates that an integrated APIC is present and hardware enabled. (Software disabling does not affect this bit.)
10–31(1)			(See note)

NOTE:

1. Some non-essential information regarding Intel486 and Pentium processors is considered Intel confidential and proprietary and is not documented in this publication. This information is provided in the *Supplement to the Pentium® Processor User's Manual* and is available with the appropriate non-disclosure agreements in place. Contact Intel Corporation for details.

4.0 USAGE GUIDELINES

This document presents Intel-recommended feature-detection methods. Software should not try to identify features by exploiting programming tricks, undocumented features, or otherwise deviating from the guidelines presented in this Application Note. The following is a list of guidelines that can help programmers maintain the widest range of compatibility for their software.

- Do not depend on the absence of an invalid opcode trap on the CPUID opcode to detect CPUID. Do not depend on the absence of an invalid opcode trap on the PUSHFD opcode to detect a 32-bit processor. Test the ID flag, as described in Section 2.0 and shown in Section 6.0.
- Do not assume that a given family or model has any specific feature. For example, do not assume that, because the family value is 5 (Pentium processor), there must be a floating-point unit on-chip. Use the feature flags for this determination.
- Do not assume that the features in the OverDrive processors are the same as those in the OEM version of the processor. Internal caches and instruction execution might vary.
- Do not use undocumented features of a processor to identify steppings or features. For example, the Intel386 processor A-step had bit instructions that were withdrawn with B-step. Some software attempted to execute these instructions and depended on the invalid-opcode exception as a signal that it was not running on the A-step part. This software failed to word correctly when the Intel486 processor used the same opcodes for different instructions. That software should have used the stepping information in the processor signature.
- Do not assume that a value of 1 in a feature flag ind icates that a given feature is present, even though that is the case in the first models of the Pentium processor in which the CPUID instruction is implemented. For some feature flags that might be defined in the future, a value of 1 can indicate that the corresponding feature is not present.
- Programmers should test feature flags individually and not make assumptions about undefined bits. It would be a mistake, for example, to test the FPU bit by comparing the feature register to a binary 1 with a compare instruction.
- Do not assume that the clock of a given family or model runs at a specific frequency and do not write clock-dependent code, such as timing loops. For instance, an OverDrive Processor could operate at a higher internal frequency and still report the same family and/or model. Instead, use the system's timers to measure elapsed time.
- Processor model-specific registers may differ among processors, including in various models of the Pentium processor. Do not use these registers unless identified for the installed processor.

5.0 BIOS RECOGNITION FOR INTEL OVERDRIVE® PROCESSORS

A system's BIOS will typically identify the processor in the system and initialize the hardware accordingly. In many cases, the BIOS identifies the processor by reading the processor signature, comparing it to known signatures, and, upon finding a match, executing the corresponding hardware initialization code.

The Pentium OverDrive processor is designed to be an upgrade to any Intel486 family processor. Because there are significant operational differences between these two processor families, processor misidentification can cause system failures or diminished performance. Major differences between the Intel486 processor and the Pentium OverDrive processor include the type of on-chip cache supported (write-back or write-through), cache organization and cache size. The OverDrive processor also has an enhanced floating point unit and System Management Mode (SMM) that may not exist in the OEM processor. Inability to recognize these features causes problems like those described below.

In many BIOS implementations, the BIOS reads the processor signature at reset and compares it to known values. If the OverDrive processor's signature is not among the known values, a match will not occur and the OverDrive processor will not be identified. Often the BIOS will drop out of the search and initialize the hardware based on a default case such as initializing the chipset for an Intel486 SX processor. Following are two common examples of system failures and how to avoid them.

Example 1

If (for the Pentium OverDrive processor) the system's hardware is configured to enable the write-back cache but the BIOS fails to detect the Pentium OverDrive processor signature, the BIOS may incorrectly cause the chipset to support a write-through processor cache. This results in a data incoherency problem with the bus masters. When a bus master accesses a memory location (which was also in the processor's cache in a modified state), the processor will alert the chipset to allow it to update this data in memory. But the chipset is not programmed for such an event and the bus master instead receives stale data. This usually results in a system failure.

Example 2

If the BIOS does not recognize the OverDrive processor's signature and defaults to an Intel486 SX processor, the BIOS can incorrectly program the chipset to ignore, or improperly route, the assertion of the floating point error signaled by the processor. The result is that floating point errors will be improperly handled by the Pentium OverDrive processor. The BIOS may also completely disable math exception handling in the OverDrive processor. This can cause installation errors in applications that require hardware support for floating point instructions.

Hence, when programming or modifying a BIOS, be aware of the impact of future OverDrive processors. Intel recommends that you include processor signatures for the OverDrive processors in BIOS identification routines to eliminate diminished performance or system failures. The recommendations in this application note can help a BIOS maintain compatibility across a wide range of processor generations and models.

6.0 PROPER IDENTIFICATION SEQUENCE

The `cpuid3a.asm` program example demonstrates the correct use of the CPUID instruction. (See Example 1.) It also shows how to identify earlier processor generations that do not implement the processor signature or CPUID instruction. This program example contains the following two procedures:

- `get_cpu_type` identifies the processor type. Figure 3 illustrates the flow of this procedure.
- `get_fpu_type` determines the type of floating-point unit (FPU) or math coprocessor (MCP).

This procedure has been tested with 8086, 80286, Intel386, Intel486, and Pentium processors. This program example is written in assembly language and is suitable for inclusion in a run-time library, or as system calls in operating systems.

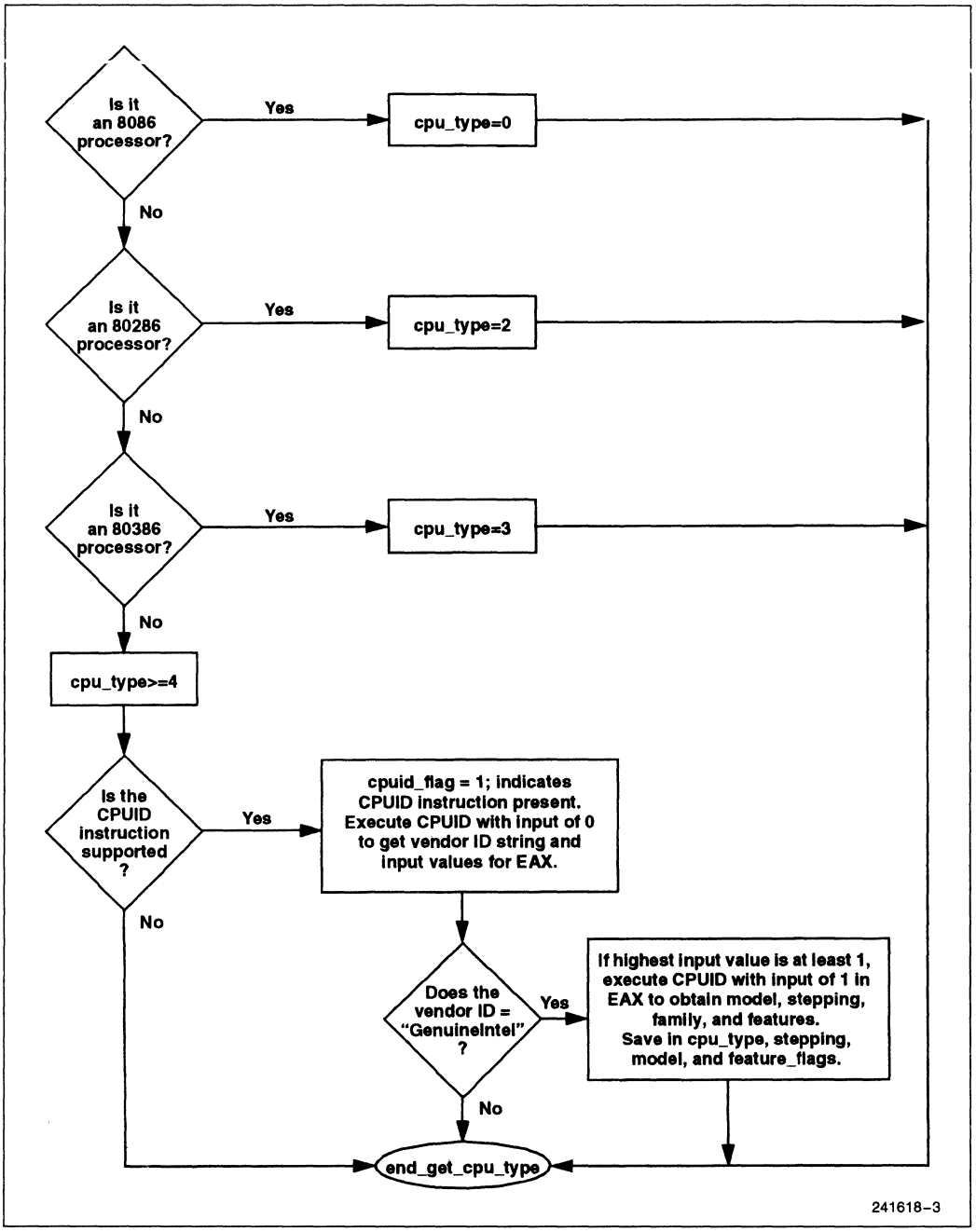


Figure 3. Flow of Processor get__cpu__type Procedure

7.0 USAGE PROGRAM EXAMPLE

The `cpuid3b.asm` and `cpuid3b.c` program examples demonstrate applications that call `get_cpu_type` and `get_fpu_type` procedures and interpret the returned information. The results, which are displayed on the monitor, identify the installed processor and features. The `cpuid3b.asm` example is written in assembly language and demonstrates an application that displays the returned information in the DOS environment. The `cpuid3b.c` example is written in the C language. (See Examples 2 and 3.)

Figure 4 presents an overview of the relationship between the three program examples.

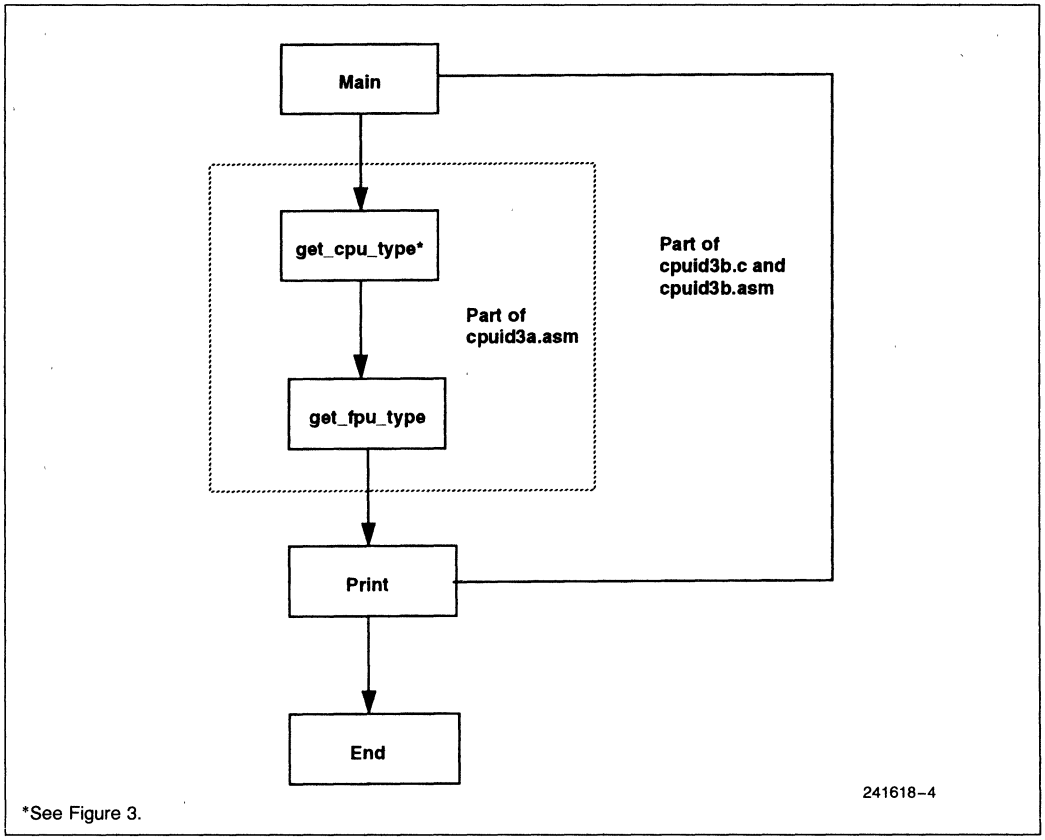


Figure 4. Flow of Processor Identification Extraction Procedures

Example 1. Processor Identification Extraction Procedure

```
;       Filename:       cpuid3a.asm
;       Copyright 1993, 1994 by Intel Corp.
;
;       This program has been developed by Intel Corporation.  You
;       have Intel's permission to incorporate this source code into
;       your product, royalty free.  Intel has intellectual property
;       rights which it may assert if another manufacturer's processor
;       mis-identifies itself as being "GenuineIntel" when the CPUID
;       instruction is executed.
;
;       Intel specifically disclaims all warranties, express or
;       implied, and all liability, including consequential and other
;       indirect damages, for the use of this code, including
;       liability for infringement of any proprietary rights, and
;       including the warranties of merchantability and fitness for a
;       particular purpose.  Intel does not assume any responsibility
;       for any errors which may appear in this code nor any
;       responsibility to update it.
;
;       This code contains two procedures:
;       _get_cpu_type: Identifies processor type in _cpu_type:
;               0=8086/8088 processor
;               2=Intel 286 processor
;               3=Intel386(TM) family processor
;               4=Intel486(TM) family processor
;               5=Pentium® family processor
;
;       _get_fpu_type: Identifies FPU type in _fpu_type:
;               0=FPU not present
;               1=FPU present
;               2=287 present (only if _cpu_type=3)
;               3=387 present (only if _cpu_type=3)
;
;       This program has been tested with the MASM assembler.
;       This code correctly detects the current Intel 8086/8088,
;       80286, 80386, 80486, and Pentium® processors in the
;       real-address mode.
;
;       To assemble this code with TASM, add the JUMPS directive.
;       jumps                   ; Uncomment this line for TASM


        TITLE   cpuid3a.asm
        DOSSEG
        .model  small

CPU_ID  MACRO
```

241618-5

```
        db      0fh             ; Hardcoded CPUID instruction
        db      0a2h
ENDM

        .data
        public  _cpu_type
        public  _fpu_type
        public  _cpuid_flag
        public  _intel_CPU
        public  _vendor_id
        public  _cpu_signature
        public  _features_ecx
        public  _features_edx
        public  _features_ebx
_cpu_type       db      0
_fpu_type       db      0
_cpuid_flag     db      0
_intel_CPU      db      0
_vendor_id      db      "------------"
intel_id        db      "GenuineIntel"
_cpu_signature  dd      0
_features_ecx   dd      0
_features_edx   dd      0
_features_ebx   dd      0
fp_status       dw      0

        .code
        .8086

;*************************************************************************

        public  _get_cpu_type
_get_cpu_type   proc

;       This procedure determines the type of processor in a system
;       and sets the _cpu_type variable with the appropriate
;       value.  If the CPUID instruction is available, it is used
;       to determine more specific details about the processor.
;       All registers are used by this procedure, none are preserved.
;       To avoid AC faults, the AM bit in CR0 must not be set.

;       Intel 8086 processor check
;       Bits 12-15 of the FLAGS register are always set on the
;       8086 processor.

check_8086:
        pushf                   ; push original FLAGS
        pop     ax              ; get original FLAGS
        mov     cx, ax          ; save original FLAGS
        and     ax, 0fffh       ; clear bits 12-15 in FLAGS
```

241618-6

```
        push    ax              ; save new FLAGS value on stack
        popf                    ; replace current FLAGS value
        pushf                   ; get new FLAGS
        pop     ax              ; store new FLAGS in AX
        and     ax, 0f000h      ; if bits 12-15 are set, then
        cmp     ax, 0f000h      ;   processor is an 8086/8088
        mov     _cpu_type, 0    ; turn on 8086/8088 flag
        je      end_cpu_type    ; jump if processor is 8086/8088

;       Intel 286 processor check
;       Bits 12-15 of the FLAGS register are always clear on the
;       Intel 286 processor in real-address mode.

        .286
check_80286:
        or      cx, 0f000h      ; try to set bits 12-15
        push    cx              ; save new FLAGS value on stack
        popf                    ; replace current FLAGS value
        pushf                   ; get new FLAGS
        pop     ax              ; store new FLAGS in AX
        and     ax, 0f000h      ; if bits 12-15 are clear
        mov     _cpu_type, 2    ; processor=80286, turn on 80286 flag
        jz      end_cpu_type    ; if no bits set, processor is 80286

;       Intel386 processor check
;       The AC bit, bit #18, is a new bit introduced in the EFLAGS
;       register on the Intel486 processor to generate alignment
;       faults.
;       This bit cannot be set on the Intel386 processor.

        .386                    ; it is safe to use 386 instructions
check_80386:
        pushfd                  ; push original EFLAGS
        pop     eax             ; get original EFLAGS
        mov     ecx, eax        ; save original EFLAGS
        xor     eax, 40000h     ; flip AC bit in EFLAGS
        push    eax             ; save new EFLAGS value on stack
        popfd                   ; replace current EFLAGS value
        pushfd                  ; get new EFLAGS
        pop     eax             ; store new EFLAGS in EAX
        xor     eax, ecx        ; can't toggle AC bit, processor=80386
        mov     _cpu_type, 3    ; turn on 80386 processor flag
        jz      end_cpu_type    ; jump if 80386 processor

        push    ecx
        popfd                   ; restore AC bit in EFLAGS first

;       Intel486 processor check
;       Checking for ability to set/clear ID flag (Bit 21) in EFLAGS
;       which indicates the presence of a processor with the CPUID
```

241618-7

```
;       instruction.

        .486
check_80486:
        mov     _cpu_type, 4    ; turn on 80486 processor flag
        mov     eax, ecx        ; get original EFLAGS
        xor     eax, 200000h    ; flip ID bit in EFLAGS
        push    eax             ; save new EFLAGS value on stack
        popfd                   ; replace current EFLAGS value
        pushfd                  ; get new EFLAGS
        pop     eax             ; store new EFLAGS in EAX
        xor     eax, ecx        ; can't toggle ID bit,
        je      end_cpu_type    ; processor=80486

;       Execute CPUID instruction to determine vendor, family,
;       model, stepping and features.  For the purpose of this
;       code, only the initial set of CPUID information is saved.

        mov     _cpuid_flag, 1  ; flag indicating use of CPUID inst.
        push    ebx             ; save registers
        push    esi
        push    edi
        mov     eax, 0          ; set up for CPUID instruction
        CPU_ID                  ; get and save vendor ID

        mov     dword ptr _vendor_id, ebx
        mov     dword ptr _vendor_id[+4], edx
        mov     dword ptr _vendor_id[+8], ecx

        mov     si, ds
        mov     es, si

        mov     si, offset _vendor_id
        mov     di, offset intel_id
        mov     cx, 12          ; should be length intel_id
        cld                     ; set direction flag
        repe    cmpsb           ; compare vendor ID to "GenuineIntel"
        jne     end_cpuid_type  ; if not equal, not an Intel processor

        mov     _intel_CPU, 1   ; indicate an Intel processor
        cmp     eax, 1          ; make sure 1 is valid input for CPUID
        jl      end_cpuid_type  ; if not, jump to end
        mov     eax, 1
        CPU_ID                  ; get family/model/stepping/features
        mov     _cpu_signature, eax
        mov     _features_ebx, ebx
        mov     _features_edx, edx
        mov     _features_ecx, ecx

        shr     eax, 8          ; isolate family
```

241618-8

```
        and     eax, 0fh
        mov     _cpu_type, al    ; set _cpu_type with family

end_cpuid_type:
        pop     edi              ; restore registers
        pop     esi
        pop     ebx

        .8086
end_cpu_type:
        ret
_get_cpu_type   endp

;*************************************************************************

        public  _get_fpu_type
_get_fpu_type   proc

;       This procedure determines the type of FPU in a system
;       and sets the _fpu_type variable with the appropriate value.
;       All registers are used by this procedure, none are preserved.

;       Coprocessor check
;       The algorithm is to determine whether the floating-point
;       status and control words are present.  If not, no
;       coprocessor exists.  If the status and control words can
;       be saved, the correct coprocessor is then determined
;       depending on the processor type.  The Intel386 processor can
;       work with either an Intel287 NDP or an Intel387 NDP.
;       The infinity of the coprocessor must be checked to determine
;       the correct coprocessor type.

        fninit                   ; reset FP status word
        mov     fp_status, 5a5ah; initialize temp word to non-zero
        fnstsw  fp_status        ; save FP status word
        mov     ax, fp_status    ; check FP status word
        cmp     al, 0            ; was correct status written
        mov     _fpu_type, 0     ; no FPU present
        jne     end_fpu_type

check_control_word:
        fnstcw  fp_status        ; save FP control word
        mov     ax, fp_status    ; check FP control word
        and     ax, 103fh        ; selected parts to examine
        cmp     ax, 3fh          ; was control word correct
        mov     _fpu_type, 0
        jne     end_fpu_type     ; incorrect control word, no FPU
        mov     _fpu_type, 1

;       80287/80387 check for the Intel386 processor
```

241618-9

```
check_infinity:
        cmp     _cpu_type, 3
        jne     end_fpu_type
        fld1                        ; must use default control from FNINIT
        fldz                        ; form infinity
        fdiv                        ; 8087/Intel287 NDP say +inf = -inf
        fld     st                  ; form negative infinity
        fchs                        ; Intel387 NDP says +inf <> -inf
        fcompp                      ; see if they are the same
        fstsw   fp_status           ; look at status from FCOMPP
        mov     ax, fp_status
        mov     _fpu_type, 2        ; store Intel287 NDP for FPU type
        sahf                        ; see if infinities matched
        jz      end_fpu_type        ; jump if 8087 or Intel287 is present
        mov     _fpu_type, 3        ; store Intel387 NDP for FPU type
end_fpu_type:
        ret
_get_fpu_type   endp

        end
```

241618-10

Example 2. Processor Identification Procedure in Assembly Language

```
;       Filename:       cpuid3b.asm
;       Copyright 1993, 1994 by Intel Corp.
;
;       This program has been developed by Intel Corporation.  You
;       have Intel's permission to incorporate this source code into
;       your product, royalty free.  Intel has intellectual property
;       rights which it may assert if another manufacturer's processor
;       mis-identifies itself as being "GenuineIntel" when the CPUID
;       instruction is executed.
;
;       Intel specifically disclaims all warranties, express or
;       implied, and all liability, including consequential and other
;       indirect damages, for the use of this code, including
;       liability for infringement of any proprietary rights, and
;       including the warranties of merchantability and fitness for a
;       particular purpose.  Intel does not assume any responsibility
;       for any errors which may appear in this code nor any
;       responsibility to update it.
;
;       This program contains three parts:
;       Part 1: Identifies processor type in the variable _cpu_type:
;
;       Part 2: Identifies FPU type in the variable _fpu_type:
;
;       Part 3: Prints out the appropriate message.  This part is
;               specific to the DOS environment and uses the DOS
;               system calls to print out the messages.
;
;       This program has been tested with the MASM assembler.
;       If this code is assembled with no options specified and linked
;       with the cpuid3a.asm module, it correctly identifies the
;       current Intel 8086/8088, 80286, 80386, 80486, and Pentium(tm)
;       processors in the real-address mode.
;
;       To assemble this code with TASM, add the JUMPS directive.
;       jumps                   ; Uncomment this line for TASM


        TITLE   cpuid3b.asm
        DOSSEG
        .model  small
        .stack  100h

        .data
        extrn   _cpu_type: byte
        extrn   _fpu_type: byte
        extrn   _cpuid_flag: byte
```

241618-11

```
        extrn   _intel_CPU: byte
        extrn   _vendor_id: byte
        extrn   _cpu_signature: dword
        extrn   _features_ecx: dword
        extrn   _features_edx: dword
        extrn   _features_ebx: dword

;       The purpose of this code is to identify the processor and
;       coprocessor that is currently in the system.  The program
;       first determines the processor type.  Then it determines
;       whether a coprocessor exists in the system.  If a
;       coprocessor or integrated coprocessor exists, the program
;       identifies the coprocessor type.  The program then prints
;       the processor and floating point processors present and type.

        .code
        .8086
start:  mov     ax, @data
        mov     ds, ax          ; set segment register
        mov     es, ax          ; set segment register
        and     sp, not 3       ; align stack to avoid AC fault
        call    _get_cpu_type   ; determine processor type
        call    _get_fpu_type
        call    print
        mov     ax, 4c00h       ; terminate program
        int     21h

;*************************************************************************

        extrn   _get_cpu_type: proc

;*************************************************************************

        extrn   _get_fpu_type: proc

;*************************************************************************


FPU_FLAG        equ     0001h
VME_FLAG        equ     0002h
PSE_FLAG        equ     0008h
MCE_FLAG        equ     0080h
CMPXCHG8B_FLAG  equ     0100h
APIC_FLAG       equ     0200h

        .data
id_msg          db      "This system has a$"
cp_error        db      "n unknown processor$"
cp_8086         db      "n 8086/8088 processor$"
cp_286          db      "n 80286 processor$"
```

241618-12

```
cp_386          db      "n 80386 processor$"

cp_486          db      "n 80486DX, 80486DX2 processor or
                db      " 80487SX math coprocessor$"
cp_486sx        db      "n 80486SX processor$"

fp_8087         db      " and an 8087 math coprocessor$"
fp_287          db      " and an 80287 math coprocessor$"
fp_387          db      " and an 80387 math coprocessor$"

intel486_msg    db      " Genuine Intel486(TM) processor$"
intel486dx_msg  db      " Genuine Intel486(TM) DX processor$"
intel486sx_msg  db      " Genuine Intel486(TM) SX processor$"
inteldx2_msg    db      " Genuine IntelDX2(TM) processor$"
intelsx2_msg    db      " Genuine IntelSX2(TM) processor$"
inteldx4_msg    db      " Genuine IntelDX4(TM) processor$"
inteldx2wb_msg  db      " Genuine Write-Back Enhanced"
                db      " IntelDX2(TM) processor$"
pentium_msg     db      " Genuine Intel Pentium® processor$"
unknown_msg     db      "n unknown Genuine Intel processor$"

; The following 16 entries must stay intact as an array
intel_486_0     dw      offset intel486dx_msg
intel_486_1     dw      offset intel486dx_msg
intel_486_2     dw      offset intel486sx_msg
intel_486_3     dw      offset inteldx2_msg
intel_486_4     dw      offset intel486_msg
intel_486_5     dw      offset intelsx2_msg
intel_486_6     dw      offset intel486_msg
intel_486_7     dw      offset inteldx2wb_msg
intel_486_8     dw      offset inteldx4_msg
intel_486_9     dw      offset intel486_msg
intel_486_a     dw      offset intel486_msg
intel_486_b     dw      offset intel486_msg
intel_486_c     dw      offset intel486_msg
intel_486_d     dw      offset intel486_msg
intel_486_e     dw      offset intel486_msg
intel_486_f     dw      offset intel486_msg
; end of array

family_msg      db      13,10,"Processor Family:  $"
model_msg       db      13,10,"Model:             $"
stepping_msg    db      13,10,"Stepping:          "
cr_lf           db      13,10,"$"

turbo_msg       db      13,10,"The processor is an OverDrive®"
                db      " processor$"
dp_msg          db      13,10,"The processor is the upgrade processor"
                db      " in a dual processor system$"
fpu_msg         db      13,10,"The processor contains an on-chip FPU$"
```

241618-13

2

```
mce_msg         db      13,10,"The processor supports Machine Check"
                db      " Exceptions$"
cmp_msg         db      13,10,"The processor supports the CMPXCHG8B"
                db      " instruction$"
vme_msg         db      13,10,"The processor supports Virtual Mode"
                db      " Extensions$"
pse_msg         db      13,10,"The processor supports Page Size"
                db      " Extensions$"
apic_msg        db      13,10,"The processor contains an on-chip"
                db      " APIC$"

not_intel       db      "t least an 80486 processor."
                db      13,10,"It does not contain a Genuine Intel"
                db      " part and as a result, the",13,10,"CPUID"
                db      " detection information cannot be determined"
                db      " at this time.$"

ASC_MSG MACRO   msg
        LOCAL   ascii_done              ; local label
        add     al, 30h
        cmp     al, 39h                 ; is it 0-9?
        jle     ascii_done
        add     al, 07h
ascii_done:
        mov     byte ptr msg[20], al
        mov     dx, offset msg
        mov     ah, 9h
        int     21h
ENDM

        .code
        .8086
print   proc

;       This procedure prints the appropriate cpuid string and
;       numeric processor presence status.  If the CPUID instruction
;       was used, this procedure prints out the CPUID info.
;       All registers are used by this procedure, none are preserved.

        mov     dx, offset id_msg       ; print initial message
        mov     ah, 9h
        int     21h

        cmp     _cpuid_flag, 1          ; if set to 1, processor
                                        ;   supports CPUID instruction
        je      print_cpuid_data        ; print detailed CPUID info

print_86:
        cmp     _cpu_type, 0
        jne     print_286
```

241618-14

```
        mov     dx, offset cp_8086
        mov     ah, 9h
        int     21h
        cmp     _fpu_type, 0
        je      end_print
        mov     dx, offset fp_8087
        mov     ah, 9h
        int     21h
        jmp     end_print

print_286:
        cmp     _cpu_type, 2
        jne     print_386
        mov     dx, offset cp_286
        mov     ah, 9h
        int     21h
        cmp     _fpu_type, 0
        je      end_print
print_287:
        mov     dx, offset fp_287
        mov     ah, 9h
        int     21h
        jmp     end_print

print_386:
        cmp     _cpu_type, 3
        jne     print_486
        mov     dx, offset cp_386
        mov     ah, 9h
        int     21h
        cmp     _fpu_type, 0
        je      end_print
        cmp     _fpu_type, 2
        je      print_287
        mov     dx, offset fp_387
        mov     ah, 9h
        int     21h
        jmp     end_print

print_486:
        cmp     _cpu_type, 4
        jne     print_unknown           ; Intel processors will have
        mov     dx, offset cp_486sx     ; CPUID instruction
        cmp     _fpu_type, 0
        je      print_486sx
        mov     dx, offset cp_486
print_486sx:
        mov     ah, 9h
        int     21h
        jmp     end_print
```

241618-15

```
print_unknown:
        mov     dx, offset cp_error
        jmp     print_486sx

print_cpuid_data:
        .486
        cmp     _intel_CPU, 1           ; check for genuine Intel
        jne     not_GenuineIntel        ;   processor
print_486_type:
        cmp     _cpu_type, 4            ; if 4, print 80486 processor
        jne     print_pentium_type
        mov     ax, word ptr _cpu_signature
        shr     ax, 4
        and     eax, 0fh                ; isolate model
        mov     dx, intel_486_0[eax*2]
        jmp     print_common
print_pentium_type:
        cmp     _cpu_type, 5            ; if 5, print Pentium processor
        jne     print_unknown_type
        mov     dx, offset pentium_msg
        jmp     print_common
print_unknown_type:
        mov     dx, offset unknown_msg  ; if neither, print unknown

print_common:
        mov     ah, 9h
        int     21h

; print family, model, and stepping

print_family:
        mov     al, _cpu_type
        ASC_MSG family_msg              ; print family msg

print_model:
        mov     ax, word ptr _cpu_signature
        shr     ax, 4
        and     al, 0fh
        ASC_MSG model_msg               ; print model msg

print_stepping:
        mov     ax, word ptr _cpu_signature
        and     al, 0fh
        ASC_MSG stepping_msg            ; print stepping msg

print_upgrade:
        mov     ax, word ptr _cpu_signature
        test    ax, 1000h               ; check for turbo upgrade
        jz      check_dp
```

241618-16

```
        mov     dx, offset turbo_msg
        mov     ah, 9h
        int     21h
        jmp     print_features

check_dp:
        test    ax, 2000h               ; check for dual processor
        jz      print_features
        mov     dx, offset dp_msg
        mov     ah, 9h
        int     21h

print_features:
        mov     ax, word ptr _features_edx
        and     ax, FPU_FLAG            ; check for FPU
        jz      check_MCE
        mov     dx, offset fpu_msg
        mov     ah, 9h
        int     21h

check_MCE:
        mov     ax, word ptr _features_edx
        and     ax, MCE_FLAG            ; check for MCE
        jz      check_CMPXCHG8B
        mov     dx, offset mce_msg
        mov     ah, 9h
        int     21h

check_CMPXCHG8B:
        mov     ax, word ptr _features_edx
        and     ax, CMPXCHG8B_FLAG      ; check for CMPXCHG8B
        jz      check_VME
        mov     dx, offset cmp_msg
        mov     ah, 9h
        int     21h

check_VME:
        mov     ax, word ptr _features_edx
        and     ax, VME_FLAG            ; check for VME
        jz      check_PSE
        mov     dx, offset vme_msg
        mov     ah, 9h
        int     21h

check_PSE:
        mov     ax, word ptr _features_edx
        and     ax, PSE_FLAG            ; check for PSE
        jz      check_APIC
        mov     dx, offset pse_msg
        mov     ah, 9h
```

241618-17

```
        int     21h

check_APIC:
        mov     ax, word ptr _features_edx
        and     ax, APIC_FLAG           ; check for APIC
        jz      end_print
        mov     dx, offset apic_msg
        mov     ah, 9h
        int     21h

        jmp     end_print

not_GenuineIntel:
        mov     dx, offset not_intel
        mov     ah, 9h
        int     21h

end_print:
        mov     dx, offset cr_lf
        mov     ah, 9h
        int     21h
        ret
print   endp

        end     start
```

241618-18

Example 3. Processor Identification Procedure in the C Language

```
/* Filename:    cpuid3b.c                                              */
/* Copyright 1994 by Intel Corp.                                       */
/*                                                                     */
/* This program has been developed by Intel Corporation.  You          */
/* have Intel's permission to incorporate this source code into        */
/* your product, royalty free. Intel has intellectual property         */
/* rights which it may assert if another manufacturer's processor*/
/* mis-identifies itself as being "GenuineIntel" when the CPUID        */
/* instruction is executed.                                            */
/*                                                                     */
/* Intel specifically disclaims all warranties, express or             */
/* implied, and all liability, including consequential and other */
/* indirect damages, for the use of this code, including               */
/* liability for infringement of any proprietary rights, and           */
/* including the warranties of merchantability and fitness for a */
/* particular purpose.  Intel does not assume any responsibility */
/* for any errors which may appear in this code nor any                */
/* responsibility to update it.                                        */
/*                                                                     */
/* This program contains three parts:                                  */
/* Part 1: Identifies CPU type in the variable _cpu_type:              */
/*                                                                     */
/* Part 2: Identifies FPU type in the variable _fpu_type:              */
/*                                                                     */
/* Part 3: Prints out the appropriate message.                         */
/*                                                                     */
/* This program has been tested with the Microsoft C compiler.         */
/* If this code is compiled with no options specified and linked */
/* with the cpuid3a.asm module, it correctly identifies the            */
/* current Intel 8086/8088, 80286, 80386, 80486, and                   */
/* Pentium(tm) processors in the real-address mode.                    */

#define FPU_FLAG         0x0001
#define VME_FLAG         0x0002
#define PSE_FLAG         0x0008
#define MCE_FLAG         0x0080
#define CMPXCHG8B_FLAG   0x0100
#define APIC_FLAG        0x0200

extern char cpu_type;
extern char fpu_type;
extern char cpuid_flag;
extern char intel_CPU;
extern char vendor_id[12];
extern long cpu_signature;
extern long features_ecx;
extern long features_edx;
```

241618-19

```
extern long features_ebx;
main() {
    get_cpu_type();
    get_fpu_type();
    print();
}
print() {
    printf("This system has a");
    if (cpuid_flag == 0) {
        switch (cpu_type) {
        case 0:
            printf("n 8086/8088 processor");
            if (fpu_type) printf(" and an 8087 math coprocessor");
            break;
        case 2:
            printf("n 80286 processor");
            if (fpu_type) printf(" and an 80287 math coprocessor");
            break;
        case 3:
            printf("n 80386 processor");
            if (fpu_type == 2)
                printf(" and an 80287 math coprocessor");
            else if (fpu_type)
                printf(" and an 80387 math coprocessor");
            break;
        case 4:
            if (fpu_type) printf("n 80486DX, 80486DX2 processor or \
80487SX math coprocessor");
            else printf("n 80486SX processor");
            break;
        default:
            printf("n unknown processor");
        }
    } else {
        /* using cpuid instruction */
        if (intel_CPU) {
            if (cpu_type == 4) {
                switch ((cpu_signature>>4)&0xf) {
                case  0:
                case  1:
                    printf(" Genuine Intel486(TM) DX processor");
                    break;
                case  2:
                    printf(" Genuine Intel486(TM) SX processor");
                    break;
                case  3:
                    printf(" Genuine IntelDX2(TM) processor");
                    break;
                case  4:
                    printf(" Genuine Intel486(TM) processor");
```

241618-20

```
                    break;
                case  5:
                    printf(" Genuine IntelSX2(TM) processor");
                    break;
                case  7:
                    printf(" Genuine Write-Back Enhanced \
IntelDX2(TM) processor");
                    break;
                case  8:
                    printf(" Genuine IntelDX4(TM) processor");
                    break;
                default:
                    printf(" Genuine Intel486(TM) processor");
                }
            } else if (cpu_type == 5)
                printf(" Genuine Intel Pentium® processor");
            else
                printf("n unknown Genuine Intel processor");
            printf("\nProcessor Family: %X", cpu_type);
            printf("\nModel:            %X", (cpu_signature>>4)&0xf);
            printf("\nStepping:         %X\n", cpu_signature&0xf);
            if (cpu_signature & 0x1000)
                printf("\nThe processor is an OverDrive® upgrade
          \processor");
            else if (cpu_signature & 0x2000)
                printf("\nThe processor is the upgrade processor \
in a dual processor system");
            if (features_edx & FPU_FLAG)
                printf("\nThe processor contains an on-chip FPU");
            if (features_edx & MCE_FLAG)
                printf("\nThe processor supports Machine Check \
Exceptions");
            if (features_edx & CMPXCHG8B_FLAG)
                printf("\nThe processor supports the CMPXCHG8B \
instruction");
            if (features_edx & VME_FLAG)
                printf("\nThe processor supports Virtual Mode \
Extensions");
            if (features_edx & PSE_FLAG)
                printf("\nThe processor supports Page Size \
Extensions");
            if (features_edx & APIC_FLAG)
                printf("\nThe processor contains an on-chip APIC");
        } else {
            printf("t least an 80486 processor.\nIt does not \
contain a Genuine Intel part and as a result, the\nCPUID detection \
information cannot be determined at this time.");
        }
    }
    printf("\n");
}
```

241618-21

Revision	Revision History	Date
-001	Original Issue.	05/93
-002	Modified Table 2. Intel486™ and Pentium® Processor Signatures.	10/93
-003	Updated to accommodate new processor versions. Program examples modified for ease of use, section added discussing BIOS recognition for OverDrive® processors, and feature flag information updated.	09/94

intel®

AP-519

APPLICATION NOTE

Pentium® Processor With Voltage Reduction Technology:

Power Supply Design Considerations for Mobile Systems

Rev. 1.0

2

December 1995

Order Number: 242558-001

Pentium® PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY

CONTENTS

CONTENTS

1.0. INTRODUCTION

As the mobile market segment has closed the gap to desktop performance with equivalent notebook performance, the challenges to the system designer have increased tremendously. As the performance requirements have increased, so have the power consumption requirements. This results in the need for careful attention to thermal design to mitigate the potentially higher temperature within the system, to power supply designs which can supply the higher current needs, and to good power management design to extend battery life to an acceptable level. The focus of this application note is the power supply design considerations.

With the addition of the Pentium® processor with Voltage Reduction Technology to the processor roadmap, Intel is able to offer the higher performance level of a 90-MHz Pentium processor at a lower power consumption level than the Pentium processor 610\75 MHz. This allows the Pentium processor notebook designs to migrate to a higher performance Pentium processor without requiring major changes to existing designs. This allows the Pentium processor to migrate into the subnotebook market, which requires lower power consumption processors as well. The Pentium processor with Voltage Reduction Technology achieves lower processor power consumption by lowering the core voltage required by the processor while maintaining compatibility to existing I/O and memory through a separate 3.3 volt supply. The major advantage provided by this splitting of processor voltage requirements is that existing Pentium processor thermal designs will accept the higher performance while requiring minimal or no thermal changes.

The notebook design areas affected by the splitting of processor voltages are in the design of power supplies, the layout of printed circuit boards and the type, quality and quantity of capacitive decoupling provided at the processor. This application note will discuss these areas and make recommendations to allow the designer to minimize the changes to existing designs. This note does not address platform issues such as chip set support, graphics controllers and memory improvements needed to support the faster 60-MHz bus of the Pentium processor 90/60 MHz.

This application note was written for mobile system designers planning to migrate from Pentium processor 610\75 MHz designs to the Pentium processor with Voltage Reduction Technology designs. It is assumed that the reader is familiar with the documents listed in the reference section. It is highly recommended that these be read and understood before using the information in this note since this application note builds on the foundation created by these reports. Section 2 defines the differences between the Pentium processor 610\75 MHz and Pentium processor with Voltage Reduction Technology and outlines the areas to be aware of while designing a system for a Pentium processor with Voltage Reduction Technology. Section 3 discusses power supply design considerations resulting from split voltage requirements between the core and I/O logic. Section 4 discusses the power plane design considerations associated with printed circuit board development for the Pentium processor with Voltage Reduction Technology. Section 5 discusses the requirements for bulk and high frequency decoupling at the processor for both the core and I/O logic. Section 6 summarizes the results of empirical measurements of V_{CC} tolerance using the capacitance values recommended in Section 5.0.

The data presented in this application note reflects studies on a selected system. It is recommended that the following information be used as a starting point for designing a Pentium processor with Voltage Reduction Technology system and that measurements be made on individual systems to ensure a robust design.

2.0. DIFFERENCES BETWEEN PENTIUM® PROCESSOR 610\75 MHZ AND PENTIUM PROCESSOR WITH VOLTAGE REDUCTION TECHNOLOGY

The architecture and internal features of the Pentium processor with Voltage Reduction Technology are identical to the Pentium processor 610\75 MHz, thereby maintaining hardware and software compatibility. The Pentium processor with Voltage Reduction Technology is offered in two versions, 75/50 and 90/60 MHz where the Pentium processor 610\75 MHz was only available in 75/50 MHz version. The Pentium processor with Voltage Reduction Technology will use the same packaging, Tape Carrier Package (TCP), used for the Pentium processor 610\75 MHz as well as the SPGA package. Experience gained in the manufacture of the Pentium processor 610\75 MHz TCP designs can be applied to a Pentium processor with Voltage Reduction Technology design.

The Pentium processor with Voltage Reduction Technology lowers processor power requirements by lowering the voltage required by the core logic of the

processor. This results in an approximately 20 percent savings in power over the Pentium processor 610\75 MHz at the same frequency. The Pentium processor with Voltage Reduction Technology also separates the voltage required (V_{CC}) for core functions (2.9 volts) and that required for I/O functions (3.3 volts). The Pentium processor with Voltage Reduction Technology maintains the same V_{CC} tolerance as required on the Pentium processor 610\75 MHz for both the core V_{CC} as well as I/O V_{CC}.

The separation of the processor voltages puts new requirements on the power supply design, layout of PCBs and decoupling requirements at the processor. These areas will be covered in the next sections.

As processors migrate to lower and lower core voltages, it is important that power supply designs take into account this trend and provide flexibility in design to handle these future voltages. This will minimize changes required in the future.

3.0. POWER SUPPLY DESIGN CONSIDERATIONS

With the separation of V_{CC} between the processor core logic and I/O logic, an additional voltage regulator is required to supply the 2.9 volts required by the core. The load handling capability of this regulator will support the majority of the processor requirements (approximately 90 percent of total current) while the processor load handling requirement on the 3.3 volt regulator drops to approximately 10 percent of the total processor current requirement.

The designer can choose between two types of regulators to implement the required core voltage. Section 3.2 discusses the use of commonly available switching and linear regulators from Linear Technology Corporation and Maxim Integrated Products to supply this voltage. Regulators from other power supply vendors are also available.

3.1 Power Supply Current Requirements

The power supply current specifications are shown in the table below. This value should be used for power supply design. It was determined using a worst-case instruction mix and V_{CC} + 165 mV. Power supply transient response and decoupling capacitors must be sufficient to handle the instantaneous current charges occurring during transitions from stop clock to full active modes.

Table 1. Pentium® Processor with Voltage Reduction Technology Maximum Power Supply Current Specifications

	Pentium® Processor 75 MHz	Pentium Processor 90 MHz
2.9 Volt Max $I_{CC}2$	2096 mA	2515 mA
3.3 Volt Max $I_{CC}3$	265 mA	318 mA

3.2 Voltage Regulator Options

To supply the additional voltage required for the core, designers have a choice of adding a linear or switching regulator to their design and regulating down from the battery output voltage. These two power supply regulators yield different ranges of efficiency. A linear regulator is much like a voltage divider which steps down the input voltage to a specified output level (e.g., from 12V or 5V to 2.9V). The maximum efficiency offered by a 5V to 2.9V linear regulator is 58 percent, as shown by the following equation:

$$\text{Efficiency} = \frac{V_{OUT}}{V_{IN}} \times 100$$

$$= \frac{2.9V}{5.0V} \times 100$$

$$= 58\%$$

In this case, the remaining 42 percent is dissipated as heat. A switching regulator, however, generates an output voltage by pulsing the input voltage through a MOSFET switch and an inductor. Since there are no voltage divider-type losses as in a linear regulator and the AC switching losses are small, there is much less dissipated heat. Switching regulators commonly offer up to 95 percent efficiency. For this reason, it is recommended that a switching regulator be used to generate the 2.9 volt requirement of the core.

There are trade-offs for the increased efficiency. Switching regulators have a larger component count than linear regulators and also a higher cost. However, the efficiency of the switching regulator with its minimum power loss makes it more ideal for the mobile application.

A switching regulator and associated support components require approximately 1.5 square inches of board space to implement. The maximum current supplied by any design will depend on the supporting components used in the design. Most manufacturers supply a reference design to make the design process as easy as possible.

In providing the voltage source for the processor core, the power supply should be designed to provide this output by adding an additional regulator in parallel with the existing 3.3 volt source as indicated in Figure 1a. This will ensure the 2.9 volt regulator will not be affected by any inefficiencies in a previous regulation stage. Figure 1b is an example of an inefficient design since a cumulative effect results from the inefficiency of the 3.3 volt source being supplied to the input of the 2.9 volt regulator.

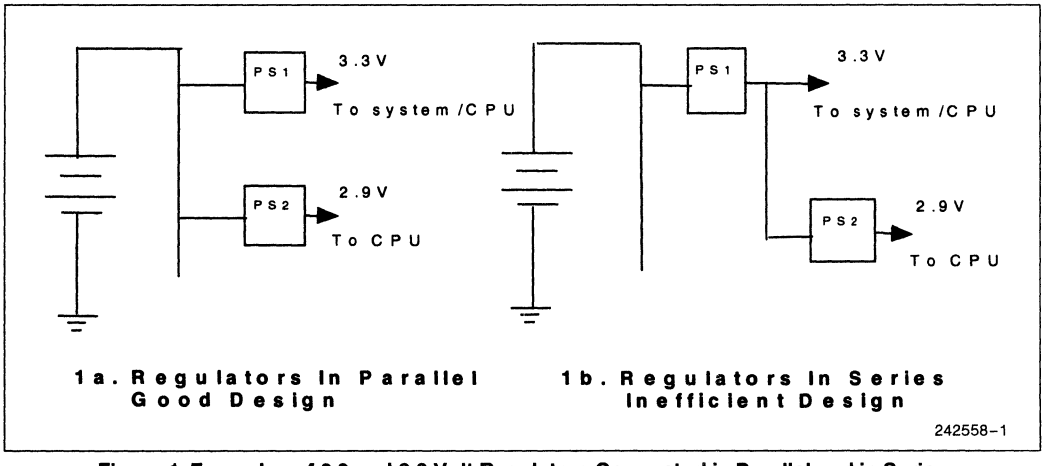


Figure 1. Examples of 2.9 and 3.3 Volt Regulators Connected in Parallel and in Series

Linear Technology Corporation and Maxim Integrated Products offer switching regulators that can be designed into power supplies meeting the maximum current and V_{CC} tolerance specifications of a Pentium processor mobile system (see Appendix C for vendor information). Other solutions may also be available from additional power supply vendors.

4.0. PCB LAYOUT CONSIDERATIONS

In most Pentium processor system board designs, the 3.3 volts for the processor is supplied on the board through a dedicated layer. With the requirement for a new 2.9 volt supply for the processor, it is not necessary to add a completely new power supply layer to the circuit board. It is possible to create a 2.9 volt "island" around the processor in the existing 3.3 volt power plane. The "island" needs to be large enough to include the processor, the required power supply decoupling capacitance (see Section 5.0), and the necessary connections to the 2.9 volt source. The power supply regulator should be physically close to the processor.

Figure 2 is an example of a TCP power plane which contains several power islands. A large core voltage island surrounds the TCP package completely on three sides and partially on a fourth. The remainder of the fourth side provides access to I/O voltage pins for creating an I/O voltage island. Similar techniques can be used on SPGA packages as well.

The 2.9 volt voltage source, along with the desired amount of bulk and high frequency capacitance, should be located close to the processor to minimize inductance due to trace length. Trace widths should be kept as wide as possible to provide maximum current capability with minimal inductance.

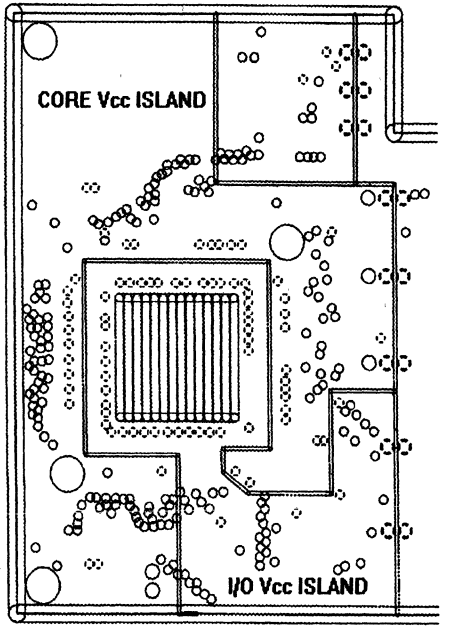


Figure 2. TCP Power Plane With Power Islands For Core and I/O V_{CC}

5.0. PROCESSOR CAPACITIVE DECOUPLING

Processor power supply decoupling is critical for reliable operation. There are two areas that the designer needs to address: high frequency decoupling resulting from small current changes in very fast transition periods (1–10 nS) and lower frequency decoupling resulting from large current changes in fast transition periods (50–300 nS).

The *Pentium® Processor 610\75 MHz Power Supply Considerations for Mobile Systems* application note indicates that the processor experiences rapid current changes transitioning between active operation and low power operation (Auto Halt Powerdown mode or STOP Grant state). These transitions occur in less than 100 nS with as much as 2.0A in current change. Even during active operation, large current changes can occur from clock cycle to clock cycle, depending on the instruction mix of the application being executed. These rapid current changes result in large load changes to the power supply. When such load transients occur, current must be supplied from the power supply decoupling capacitors, since the power supply cannot change its output current instantly. Negative transients (voltage droop) occur when current load increases, while positive transients (voltage surges) occur when load current decreases.

The response time of a switching regulator power supply to a large transient current is limited by the value of the energy storing inductor, and is typically on the order of 10 μs or less. Adequate bulk capacitance, located as close as possible to the processor, is needed to provide current until the power supply can respond to the change in load. Capacitors with low ESR and ESL values are required to keep the processor supply voltage within the V_{CC} tolerance spec. Suggestions for the type and quantity of bulk decoupling capacitors to be used are provided in Section 5.1. The voltage transients caused by the rapid transitions between active and low power modes are additive to any DC voltage drops between the power supply regulator and the processor. Minimizing board level DC drops provides more margin for transient effects. This can be accomplished by using short, wide power traces to minimize resistance and inductance.

Due to the high internal speed of the processor and rapid transitions on the external bus, high frequency decoupling is also required. High frequency capacitors connected between the power and ground planes near the processor are required to filter these high frequency components of noise. Section 5.2 provides recommendations for the type and quantity of high frequency decoupling capacitors to be used.

2

5.1 Bulk Decoupling

As pointed out in Section 5.0, bulk decoupling is required with the Pentium processor, since the processor switches between normal and low power states very quickly, causing large instantaneous current changes. The resulting power supply voltage transient is illustrated in Figure 3.

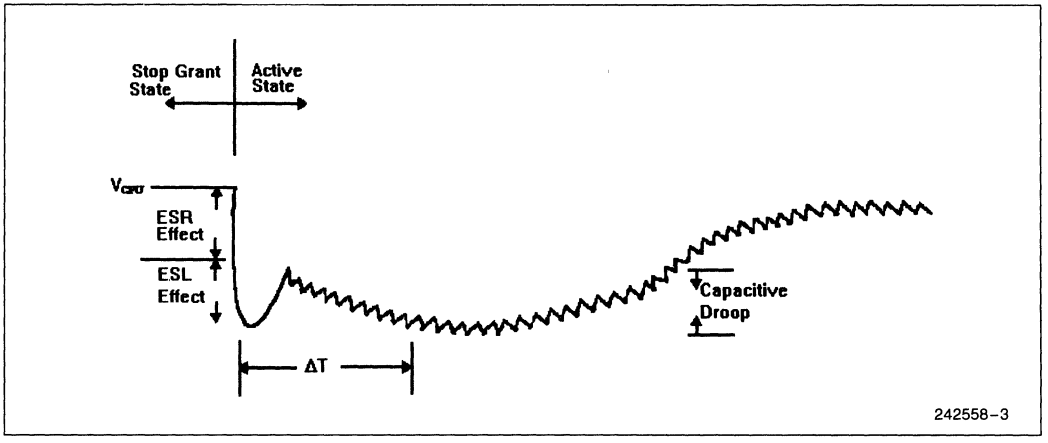


Figure 3. Example of Power Supply Voltage Transient

The voltage initially drops as the current encounters the decoupling capacitor's equivalent series resistance (ESR) and further undershoots by a voltage equal to the decoupling capacitor's equivalent series inductance (ESL) times the rate of change of current (di/dt). Then as the decoupling capacitor discharges, the power supply voltage droops until the power supply regulator reacts and is able to supply the full processor current. Since the processor is now active, high frequency noise also appears on the waveform. The system then settles to the initial voltage regulator setting, offset by any DC voltage drop. The total voltage dip must not exceed the voltage tolerance spec.

Part of the voltage tolerance specification is consumed by regulator tolerance (±2 percent), high frequency noise and any board level DC drops. The remaining amount is available to handle the processor active/low power transitions. Allocating about ½ of this amount for the ESR drop and the other ½ for the ESL undershoot and capacitive droop results in reasonable required values of ESR and ESL. If we allow too much for ESR, then the budget for ESL becomes prohibitive.

The following examples show how to determine the required component values, using a 60/40 ratio for ESR to capacitive droop. The solution can be fine-tuned to meet design requirements by adjusting the ratio between ESR and capacitive droop.

Core Supply Example

Power source: Lithium ion battery, 5.5 volts.

Switching regulator: 300 KHz, Vin = 5.5 volts,
Vout = 2.9 volts,
buck inductor L = 7.5 μH

ΔI = (Max active I_{CC} − Stop Grant I_{CC}*) for core V_{CC} pins

= 1.95 amps Pentium® Processor 90 MHz with Voltage Reduction Technology

= 1.78 amps Pentium Processor 75 MHz with Voltage Reduction Technology

ΔT = Regulator response time (ΔI = 1.95 amps, L = 7.5 μH) = 7 μs

ΔV = Voltage budget for ΔI

= V_{CC} tolerance specification (165 mV) - Regulator tolerance (2 percent) - high frequency noise (20 mV)*

= 165 mV − 2.9V x (0.02) − 20 mV = 87 mV

= $\Delta V_{ESR} + \Delta V_{CAPDROOP}$

* Adequately decoupled high frequency noise measures about 40 mV peak-to-peak.

Assume that the board DC drop is negligible.

Maximum allowable ESR:

$$ESR_{MAX} = \Delta V_{ESR} / \Delta I$$

Bulk decoupling can be calculated as:

$$C_{MIN} = (\Delta I \times \Delta T) / \Delta V_{CAPDROOP}$$

Allocating 60 percent of ΔV for ESR effects (ΔV_{ESR} = (0.6) x 87 mV, $\Delta V_{CAPDROOP}$ = (0.4) x 87 mV) gives the following results:

$$ESR_{MAX} = 0.027 \text{ ohm}$$

$$C_{MIN} = 392 \ \mu F$$

Note the low ESR value required for this application. This will require multiple low ESR capacitors in parallel to achieve this value. In this example, 4 x 100 μF / 0.1 ohm ESR capacitors will satisfy both maximum ESR and minimum bulk decoupling requirements.

This has now ensured that the maximum power supply voltage droop will not exceed the processor voltage tolerance specification for transitions between active and Stop Grant states. An additional factor that must also be comprehended is the inductive undershoot resulting from the rapid rate of current change times the ESL of the decoupling capacitors. As the processor resumes activity, there are many large, instantaneous demands for power supply current as the various internal circuits start switching. There is a need to ensure that the resulting change in power supply voltage remains within limits. Since this inductive voltage effect is additive to the ESR drop, its magnitude must be less than or equal to the capacitive droop.

Maximum allowable ESL:

$$ESL_{MAX} = (\Delta V_{ESL} \times \Delta T) / \Delta I$$

where

ΔV_{ESL} = voltage drop across the capacitor due to parasitic inductance

ΔI = current change within a specified DT

ΔT = period in which current changes

A current slew rate of 40 mA / nS will be used for this calculation. With 40 percent of the previously calculated 87 mV budget as the maximum allowed value for **DV**, an overall ESL of less than or equal to 0.87 nH is needed.

$$ESL_{MAX} = (0.4) \times 87 \text{ mV} / (40 \text{ mA} / \text{nS}) = 0.87 \text{ nH}$$

Surface mount capacitors with short lead lengths are available with an ESL value of 2 nH. Since four capacitors in parallel are needed to achieve the required overall ESR value, using these 2 nH ESL capacitors will result in an overall value of 0.5 nH, and will meet the overall requirement with margin left over for any additional inductance due to board traces.

I/O Supply Example

Switching regulator: 300 KHz, Vin = 5.5 volts,
Vout = 3.3 volts,
buck inductor L = 7.5 μH

ΔI = IO supply current
= 318 mA Pentium® Processor 90 MHz with Voltage Reduction Technology
= 265 mA Pentium Processor 75 MHz with Voltage Reduction Technology

ΔT = Regulator response time
(ΔI = 318 mA, L = 7.5 uH) = 3.0 μs

ΔV = Voltage budget for ΔI
= V_{CC} tolerance specification (165 mV) - Regulator tolerance (2 percent) - high frequency noise (20 mV)*
= 165 mV − 3.3V x (0.02) − 20 mV = 79 mV
= $\Delta V_{ESR} + \Delta V_{CAPDROOP}$

* High frequency noise (about 40 mV peak-to-peak) from other devices on the 3.3 volt supply. Assume that the board DC drop is negligible.

NOTE:
This calculation is for the Pentium processor I/O supply bulk decoupling only. Other high power components, including the L2 cache, may require additional bulk decoupling to meet voltage tolerance specs for transitions between active and Stop Grant states.

Maximum allowable ESR:

$$ESR_{MAX} = \Delta V_{ESR} / \Delta I$$

Minimum bulk decoupling:

$$C_{MIN} = (\Delta I \times \Delta T) / \Delta V_{CAPDROOP}$$

Allocating 60 percent of ΔV for ESR effects (ΔV_{ESR} = (0.6) x 79 mV, $\Delta V_{CAPDROOP}$ = (0.4) x 79 mV) gives the following results:

ESR_{MAX} = 0.150 ohm C_{MIN} = 30 μF

Again, please note the significance of low ESR. Use a 33 μF / 0.15 ohm ESR capacitor.

The current transients associated with I/O pin switching are small relative to the processor core and also much faster. Small, high frequency capacitors are needed to handle these current demands. The bulk decoupling capacitor is less important in this respect, and so there is no specific calculation of required ESL. Regardless, low ESL (short lead, surface mount, ESL = 2 nH) capacitors are recommended for I/O supply bulk decoupling.

Table 2. Pentium® Processor with Voltage Reduction Technology Bulk Decoupling Estimates (300-KHz Regulator)

	Pentium® Processor 75 MHz	Pentium Processor 90 MHz
2.9 Volt Core Decoupling	4x 100 μF / 0.1 ohm ESR / 2 nH ESL	4x 100 μF / 0.1 ohm ESR / 2 nH ESL
3.3 Volt I/O Decoupling	33 μF / 0.15 ohm ESR / 2 nH ESL	33 μF / 0.15 ohm ESR / 2 nH ESL

5.2 High Frequency Decoupling

High frequency decoupling is also critical for reliable operation. High frequency transients can be minimized by the use of multiple 1.0 μF and 0.01 μF bypass capacitors. Ceramic high frequency capacitors have the best high frequency performance and should be placed as close as possible to the processor between the processor power and ground pins. As a first approximation it is recommended that in a Pentium processor with voltage reduction technology design, the number of high frequency capacitors to be used should be equivalent to the number of capacitors used in a Pentium processor 610\75 MHz design. These capacitors should be split between the core V_{CC} and I/O V_{CC}.

In the testing to be described in the next section, it was found that using eight ceramic capacitors on the core V_{CC} and eight ceramic capacitors on the I/O provided adequate high frequency decoupling to maintain V_{CC} tolerance limits.

6.0. EXPERIMENTAL VERIFICATION OF CAPACITANCE ESTIMATES

A 90-MHz Pentium processor was used to verify the performance of the processor bulk capacitance recommendations. The 2.9V to the core was supplied by a 300-KHz switching regulator, and the 3.3V was supplied to the I/O by another 300-KHz switching regulator.

The method of measuring worst-case transients used for the experiments required the assertion of the STPCLK# input pin. Mobile chip sets provide support for STPCLK# control through programming a register which toggles STPCLK#. Oscilloscope probes were connected to STPCLK#, $V_{CC}2$ (core V_{CC}), $V_{CC}3$ (I/O V_{CC}) and STPCLK# was used as the trigger source.

Since the STPCLK# pin is toggled through the chip set hardware or by using a pulse generator connected to processor STPCLK# pin, additional software can be executed by the processor to produce high load conditions. DOS Edit with menu pulldown is an example of code which causes steady state high power consumption by the processor (about 5.1W for the Pentium processor 90 MHz, depending on the system configuration). DOS Edit was used for these experiments. A high-power, FPU-intensive code loop can also be acquired from Intel (contact your local Intel sales representative) which will provide similar results. It should be noted that in making a measurement, sufficient time should be allowed to elapse before the power state is toggled. For example, a 300-KHz switching regulator typically takes about 7 μs to respond to a load change. If the power state is toggled before 7 μs, the overall response of the regulator cannot be viewed since the regulator has not had adequate time to respond.

The following discussions cover three different configurations of decoupling/filtering capacitance applied around the core and I/O V_{CC} pins of the Pentium processor on the daughter card. The first of these traces (Figure 4 below) illustrates the results from the recommended capacitance configuration and shows the overall pattern of STPCLK# used to drive the experiments.

The middle signal in Figure 4 is the I/O V_{CC} and the upper signal is the core V_{CC}. Note that the core voltage increases by 80 mV to 90 mV in the STPCLK# active (low) region, since the current drawn is greatly reduced, and thus so are the voltage drops along the power buses. (In our experimental setup, we had greater resistance in the power busses than recommended for production systems, and thus larger DC offsets between the power states.) Also as expected, the noise level on the V_{CC} pins is reduced in the STPCLK# active region, especially for the core V_{CC}, since the core activity is essentially turned off and most of the noise on the core V_{CC} pins is generated inside the processor itself.

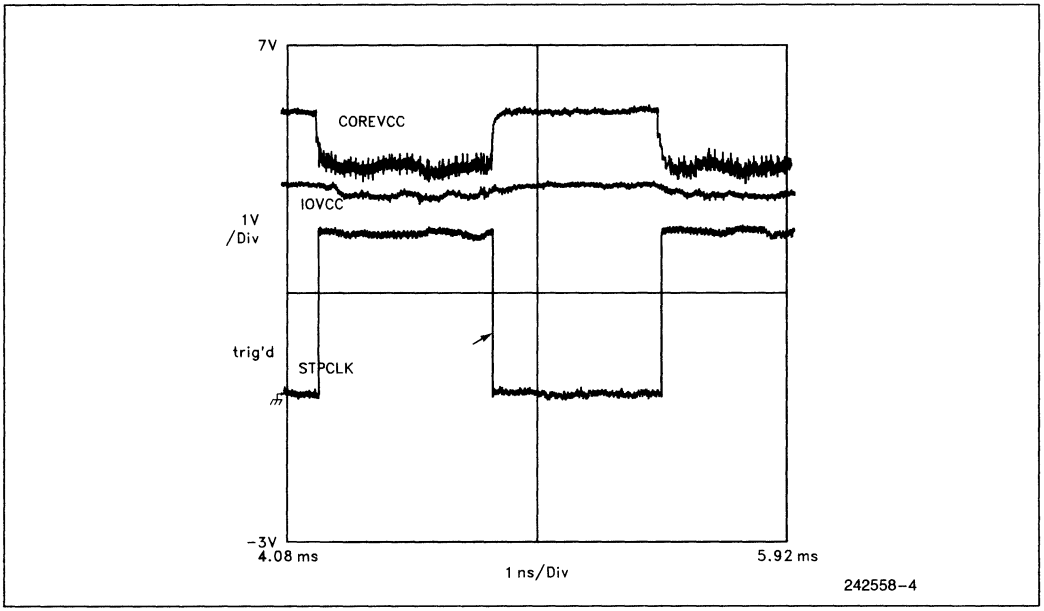


Figure 4. Example of Adequate Bulk Decoupling Capacitance for Core and High Frequency Capacitance

Using the recommended capacitance:

Bulk capacitors: core: 4 - 100 μF, I/O: 1- 47 μF

High frequency capacitor: core: 6 - 0.1 μF, 2 - 0.01 μF, I/O: 2 - 0.1 μF, 6 - 0.01 μF

The example system always operates within V_{CC} specifications and always works reliably. The largest variation from core V_{CC} minimum to maximum (peak-to-peak variation, including the DC offset) is about 150 mV peak-to-peak, which is well within the 214 mV tolerance (core V_{CC} tolerance - core regulator tolerance). The largest peak-to-peak variation in the I/O V_{CC} is about 50 mV, which is also well within the 198 mV tolerance (I/O V_{CC} tolerance - I/O regulator tolerance). (Note that in a production model, more guardband would be obtained by reducing the DC offset between the power states.)

Figures 5 and 6 illustrate the transition into and out of the Stop Grant state with a finer time resolution. The time scale of 200 ns will allow viewing undershoot and overshoot if they occur. The pictures show that using the recommended capacitive decoupling, there is no significant V_{CC} overshoot when STPCLK# is asserted or undershoot when it is deasserted.

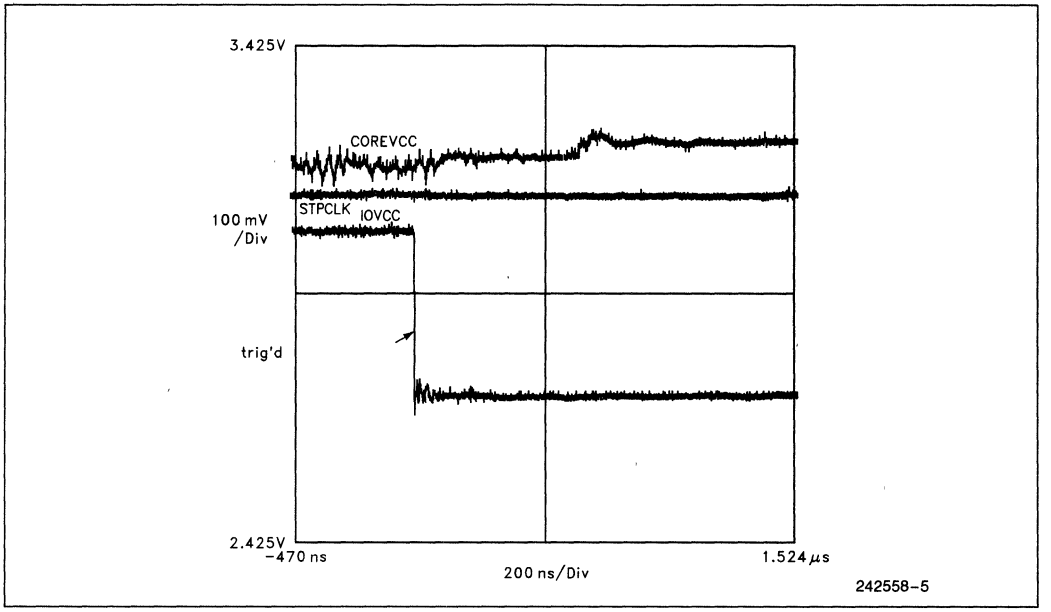


Figure 5. Example of Adequate Bulk Decoupling Capacitance for Core and I/O V_{CC} Entering Stop Grant

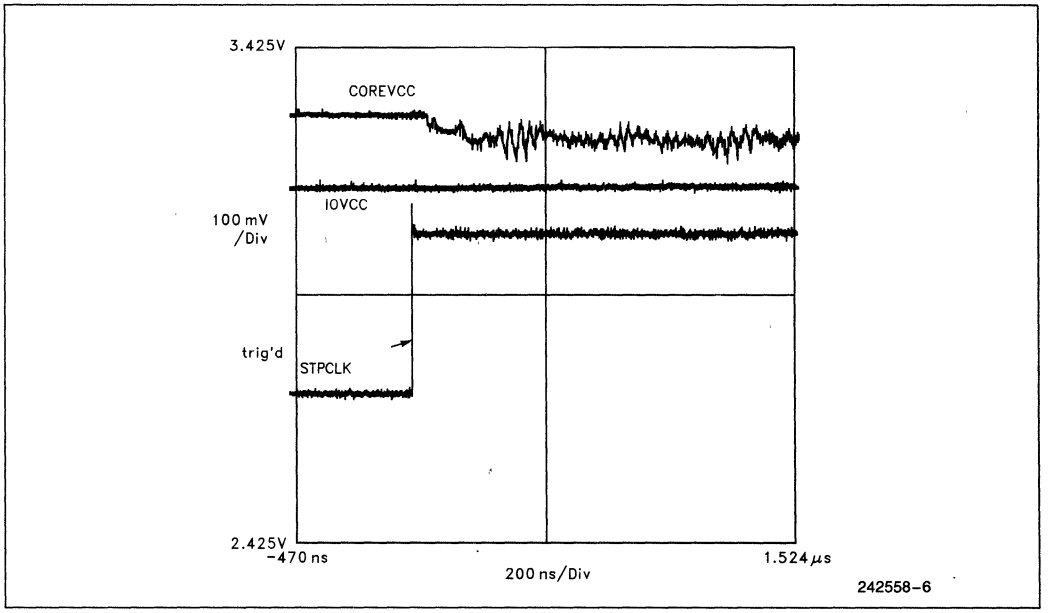


Figure 6. Example of Adequate Bulk Decoupling Capacitance for Core and I/O V_{CC} Exiting Stop Grant

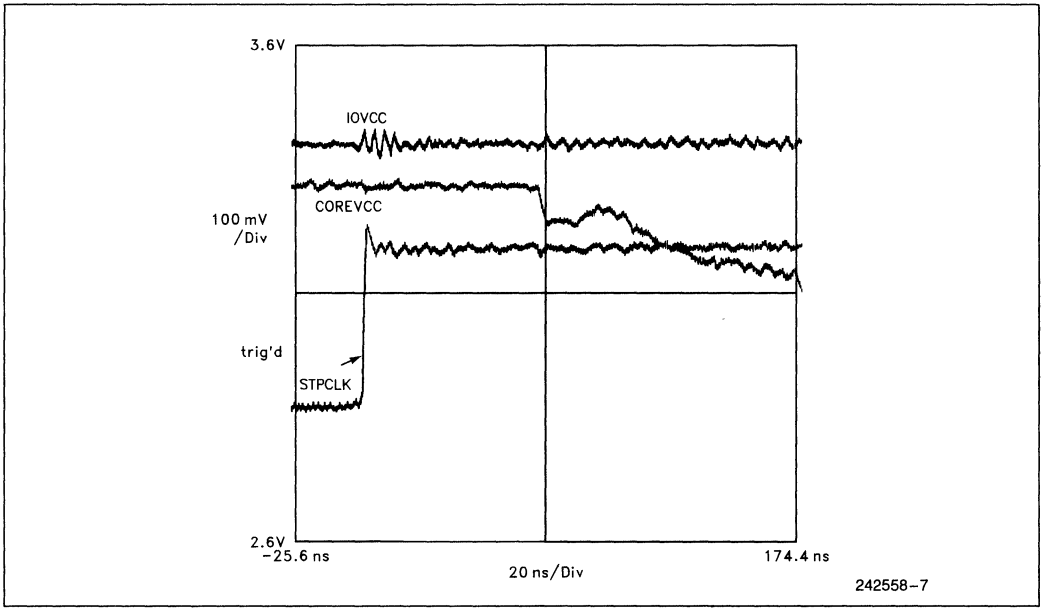


Figure 7. Example of No Bulk Decoupling Capacitance for Core and I/O V_{CC} Exiting Stop Grant

Using only high frequency filter capacitors:

Bulk Capacitors: 0; Standard high frequency filter Capacitors

Frequent large violations of specifications were observed on both I/O and core V_{CC}s. Figure 7 shows sizable ringing in the V_{CC} values after the processor enters the Stop Grant state with a peak-to-peak range of 210 mV on V_{CC} core, and about 65 mV on V_{CC} I/O. As expected, there is a severe droop in the core V_{CC} due to the lack of bulk capacitors to handle the abrupt change in load and the voltage tolerance specification is violated. The picture also illustrates the importance of the bulk decoupling for the core V_{CC}, where the largest load transients occur. Note also that although these experiments were done with no bulk capacitors added to the processor card, there was still capacitance in the power supply, small capacitance from loads and board effects, and a total of 0.88 μF from the high frequency capacitors.

Using *no* high frequency filter capacitors, with recommended bulk capacitors:

Bulk Capacitors: 400 μF core; 47 μF I/O. High frequency Capacitors: None

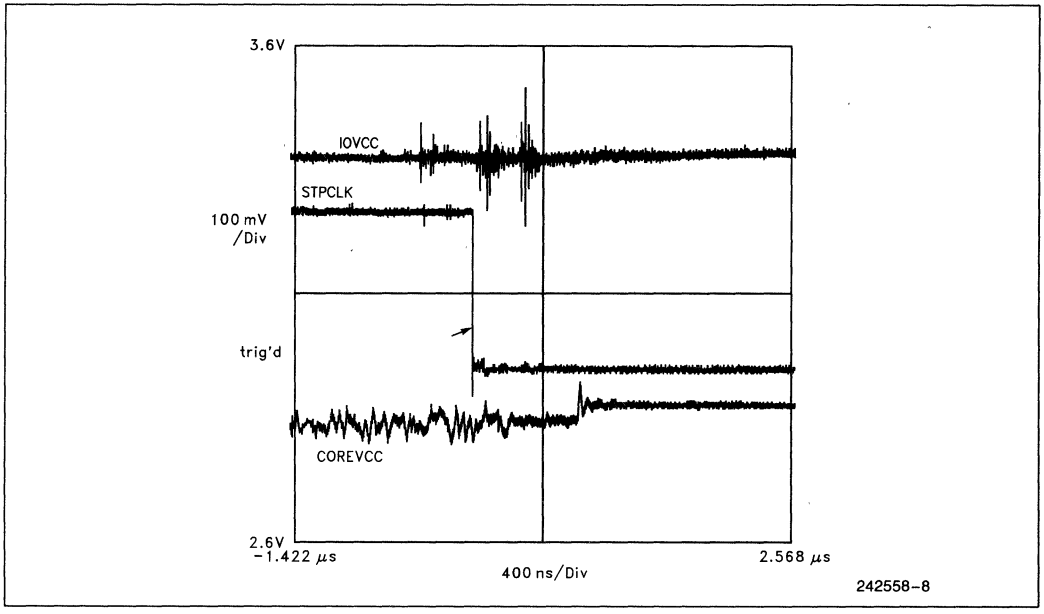


Figure 8. Example of No High Frequency Decoupling Capacitance for Core and I/O V_{CC} Exiting Stop Grant

A high magnitude of high frequency noise was observed on both the core and I/O V_{CC}. Figure 8 shows almost 300 mV of peak-to-peak noise on the I/O V_{CC}, and about 125 mV of peak-to-peak noise on the core V_{CC}. In this case, the system does not meet the voltage tolerance specifications and is not stable.

The data presented in this application note reflects studies on a selected system. It is recommended that the following information be used as a starting point for doing a Pentium processor with voltage reduction technology system and that measurements be made on individual systems to ensure a robust design.

7.0. SUMMARY

With the addition of the Pentium processor 90 MHz to the microprocessors a designer has to choose from, additional design issues must be addressed. The separation of core and I/O voltages requires the use of an additional regulator to supply the 2.9 volt requirement of the core. This in turn requires the separation of power planes on the system board or daughter card. Appropriate decoupling of the core V_{CC} and I/O V_{CC} is also required to ensure the V_{CC} tolerance specifications are met.

Using power islands for core V_{CC} and I/O V_{CC} offer the designer an option to avoid using an additional power plane for the system board or daughter card. There are switching regulator controllers commercially available which can be used to design power supplies to source the worst case current for the Pentium processor with Voltage Reduction Technology. Transients during large current changes can be overcome by using adequate bulk decoupling capacitance. Additional high frequency decoupling will help reduce high frequency noise to an acceptable level. It is recommended that the bulk and high frequency capacitance estimates provided in this note, be used as a starting point for selecting the correct amount of decoupling used in a Pentium processor with Voltage Reduction Technology. It is further recommended that the V_{CC} tolerance for both core and I/O be verified through the use of applications (such as DOS Edit, and I/O intensive benchmarks) which provide a worst case operational scenario. This will ensure that the final design will meet V_{CC} tolerance specifications.

APPENDIX A
EXPERIMENTAL PLATFORM

All measurement data and oscilloscope traces were taken on the following platform/test equipment:

Neptune 82430 PCI Evaluation System
TCP-to-SPGA Converter Nehemiah Card
300-KHz Switching Regulator Power Supply
Tektronix TLS216 Logic Scope (16 Channel, 2 Hz)
Tektronix Pulse Generator PG2011

APPENDIX B
REFERENCES

Pentium™ Processor at iComp™ Index 610\75 MHz (Order Number 242323)†

Pentium™ Processor (610\75) Power Supply Considerations for Mobile Systems Application Note, (Order Number 242415)

AP517 Pentium™ Processor (610\75) Power Consumption Application Note, (Order Number 242416)

AP515 Pentium™ Processor (610\75) TCP System Thermal Design Application Note, (Order Number 242414)

APPENDIX C
VENDORS PROVIDING VOLTAGE REGULATORS AND CAPACITORS

The list below is meant to be representative only, and does not include all vendors of a particular type. Intel has not tested all of the components listed below and cannot guarantee that these components will meet every PC manufacturers specific requirements.

Voltage Regulators:

Linear Technology Corporation
1630 McCarthy Blvd.
Milpitas, CA 95035-7487
Tel. (408) 432-1900

Maxim Integrated Products
120 San Gabriel Drive
Sunnyvale, CA 94086
Tel. (408) 737-7600

Decoupling Capacitors:

AVX Corporation TPSE Series
Myrtle Beach, South Carolina 29577 USA

KEMET Electronics Corporation T Series
P.O.Box 5928
Greenville, South Carolina 29606 USA
Tel. (803) 963-6348

Nichicon (American) Corporation PL Series
927 East State Parkway,
Schaumburg, Illinois 60173 USA

Sanyo Video Components OS-CON Series
2001 Sanyo Ave.
San Diego, California 92073 USA
Tel. (619) 661-6835

intel®

AP-521

APPLICATION NOTE

Designing with the Pentium® Processor (735\90, 815\100, 1000\120, 1110\133), 82498 Cache Controller and 82493 Cache SRAM

December 1995

Order Number: 242640-001

DESIGNING WITH THE Pentium® PROCESSOR (735\90, 815\100, 1000\120, 1110\133), 82498 CACHE CONTROLLER AND 82493 CACHE SRAM

CONTENTS

CONTENTS

1.0. INTRODUCTION

This document describes two sample Pentium® processor 82498/82493 layouts. The 82498 cache controller and the 82493 cache SRAM are second-level cache components which are optimized, both electrically and functionally, for use with the Pentium processor (735\90, 815\100, 1000\120, 1110\133). In order to achieve maximum performance from a Pentium processor 82498/82493 configuration, system designers must tune the placement and routing of the components on their board. This document provides information about sample layouts which have already been simulated. The sample layouts described in this document may be used as a starting point for designing a Pentium processor 82498/82493 layout. The theories and simulation methods required to create an optimized Pentium processor 82498/82493 design are contained in a separate application note, *Designing with the Pentium® Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set, AP-481* (Order Number 241576). Please reference the previously mentioned application note for all necessary background information since this document is designed to be used in conjunction with AP-481.

Note that the Pentium processor 82498/82493 1-Mbyte layout is intended to be extremely similar in placement and topology to the Pentium processor 82496/82491 256K layout as long as flight time requirements are met. Simulation data indicates that only a minimal amount of tuning should be required. The Pentium processor (735\90, 815\100, 1000\120, 1110\133) 82498/82493 2-Mbyte layout is a new and improved design which is more compact than the Pentium processor 82496/82491 512K layout. The 2-Mbyte layout utilizes standard board design methods that were not used in the 1-Mbyte layout. For instance variable trace widths are used to decrease impedance in line lengths, thereby decreasing signal flight times. Both layouts include a revised pinout and split plane for mixed voltage operation. New simulations are required for all topologies in both the 1 and 2-Mbyte Pentium processor 82498/82493 layouts.

2.0. INTERFACE PARAMETERS

The 82498/82493 Cache Chip Set has been designed to take advantage of the high performance available from the 66-MHz bus frequency. In addition, it has been designed to obtain this performance without severely adding to the complexity of the system design. These benefits are accomplished by dividing the chip set into two interfaces. The first is the External Interface which is the interface between the CPU-Cache core and the rest of the system. It consists of the memory bus and the memory bus controller. This interface has been designed to operate at a fraction of the CPU's frequency or asynchronous to the CPU. These options simplify the system design by minimizing the portions that must deal with the high frequency signals. The second interface is the Optimized Interface which is between the Pentium processor (735\90, 815\100, 1000\120, 1110\133) and the 82498 cache controller and 82493 cache SRAM. This interface is tuned for the known configuration options of the chip set and includes specially designed input and output buffers optimized for the defined electrical environment of each signal path.

AC parameters must be taken into account due to the fact that Optimized Interface must be tuned for each design. The purpose of this chapter is to give a brief description of the AC specifications of the chip set which were taken into account when designing the External and Optimized Interfaces of the examples contained in this document. Once again, if more information is desired please refer to the application note, *Designing with the Pentium® Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set, AP-481*. It contains details on how to measure the various AC parameters of a design. In addition, it has information on I/O buffer models, high frequency design considerations, and chip set layout design.

To obtain the AC/DC specifications and flight times for the chip set, please refer to the *Pentium® Family Developer's Manual Volume 2: 82496/82497/82498 Cache Controller and 82491/82492/82493 Cache SRAM* (Order Number 241429).

2.1. External Interface

The External Interface is the interface between the CPU-Cache core and the rest of the system. It consists of the memory bus and the memory bus controller. This interface has been designed so that it can operate at a fraction of the CPU's frequency or asynchronous to the CPU. These options simplify the system design by minimizing the portions that must deal with the high frequency signals.

The specifications for the external interface are defined to allow system designers to connect the CPU and Cache components to other devices (ASICs, PLDs, memory, etc.). The external interface signal specifications are the more traditional output valid delay and flight time and input setup and hold time. In addition, I/O buffer models have been provided as a tool to assist system designers.

2.1.1. OUTPUT VALID DELAY AND FLIGHT TIME

Output valid delay is the amount of time it takes the signal to transition referenced from the clock edge. The maximum output valid delay is the earliest point in time that the signal can be assumed valid at the pin of the device. This timing is referenced from the clock edge and is measured at 1.5 volts as illustrated in Figure 1.

NOTE

This specification is defined assuming C_L = 0 pF; therefore, system designers must account for all delay added by the signal traveling to the receiving device.

The maximum output valid delay is used by system designers to perform a worst case timing analysis to ensure that setup times are met at receiving devices.

The minimum output valid delay is the earliest time data will begin transition after the clock edge. This timing is also referenced from the clock edge and is measured at 1.5 volts as illustrated in Figure 1.

The minimum output valid delay is used by system designers to perform a worst case timing analysis to ensure that hold times are met at receiving devices. In addition, on I/O or tri-state pins, it is used to ensure no bus contention exists due to multiple devices driving the bus simultaneously.

The maximum output flight time is the amount of time it takes to propagate a signal that was driven in the previous clock. This timing is also referenced from the clock edge and is illustrated in Figure 2. Note that the minimum valid delay determines how long data from the previous clock remains valid as shown in Figure 2.

2.1.2. INPUT SETUP AND HOLD TIME

Input setup time is the amount of time a signal must be valid at the component's input pin prior to the clock edge during which it is sampled. The minimum input setup time is the latest point in time that the signal can be assumed valid at the pin of the device. This timing is referenced to the clock edge and is measured at 1.5 volts as illustrated in Figure 3.

The input setup time is used by system designers to perform a worst case timing analysis to verify that fast enough drivers have been chosen for ASICs or other devices and that the signals are able to travel across the board layout in the allotted amount of time.

Input hold time is the amount of time a signal must be valid at the component's input pin after the clock edge it is sampled. The minimum input hold time is the earliest point in time that the signal can start its next transition at the pin of the device. This timing is referenced to the clock edge and is measured at 1.5 volts as illustrated in Figure 3.

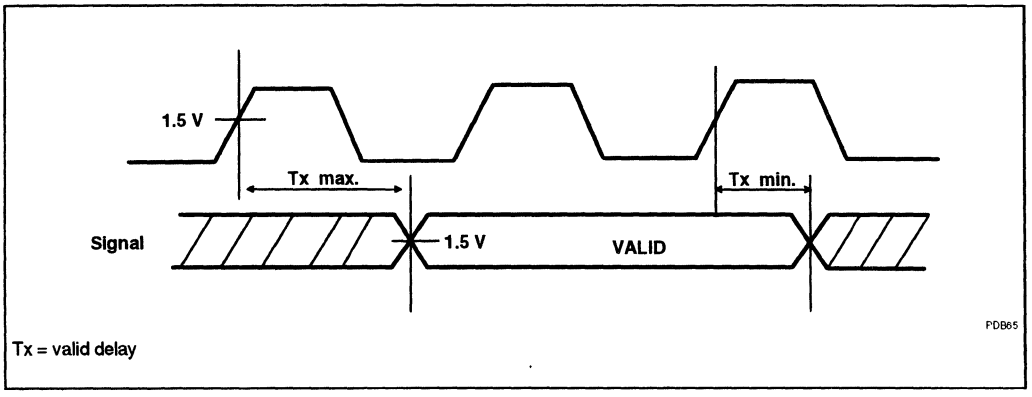


Figure 1. Output Valid Delay

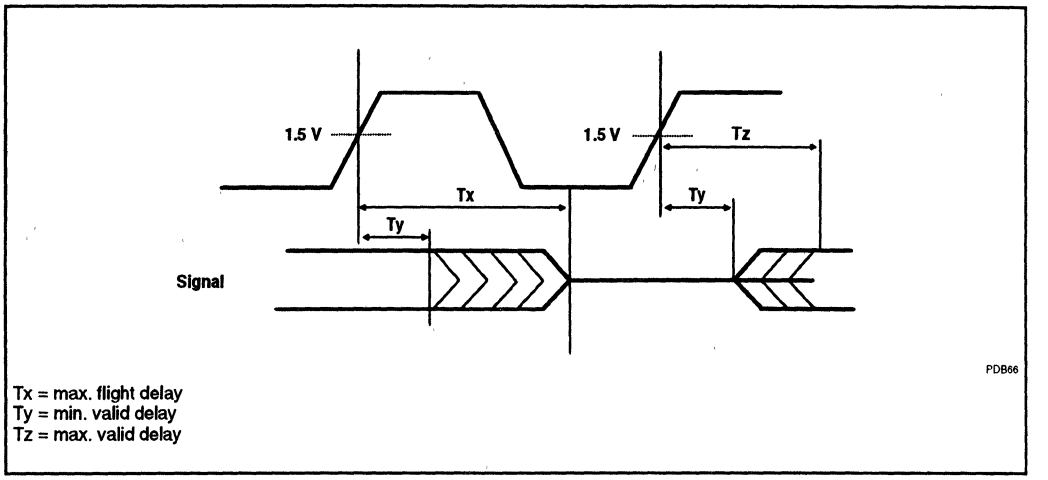


Figure 2. Output Flight Time

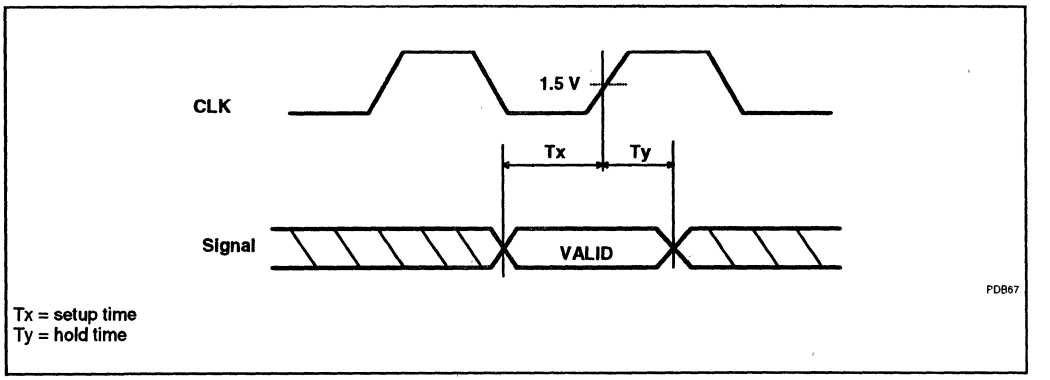


Figure 3. Input Setup and Hold Time

2.2. Optimized Interface

The optimized interface is the high-performance interconnect between the Pentium Processor (735\90, 815\100, 1000\120, 1110\133) and the 82498 cache controller and 82493 cache SRAM. The input and output buffers have been tuned for the defined configuration and electrical environment (loading, etc.). This tuning is what allows the chip set's CPU-Cache core to operate synchronously at 60 MHz/66 MHz.

There are two types of specifications for signals in the optimized interface. The first is Flight Time which is used to guarantee that signal timings are met. The second is Signal Quality to guarantee reliable operation. I/O buffer models can be used as a tool to ensure these specifications are met. In this section, flight time and signal quality will be discussed. Details on I/O buffer models can be found in the application note, AP-481 and the *Pentium® Family Developer's Manual Volume 2: 82496/82497/82498 Cache Controller and 82491/82492/82493 Cache SRAM.*

When simulating the optimized interface for either Flight Time or Signal Quality, it is critical to use the appropriate buffer model specification. Table 1 shows the correct specifications to use in the Flight Time or Signal Quality simulation.

Table 1. Specifications to be Used for Simulation of Flight Time or Signal Quality

	Flight Time	Signal Quality
Driver:		
dV/dt	Min	Max
Co	Max	Min
Ro	Max	Min
Cp	Max	Min
Lp	Max	Min
Receiver:		
Cin	Max	Min
Cp	Max	Min
Lp	Max	Min
Other:		
Temperature	Max	Min
V_{CC}	Min	Max
Board Zo	Min	Max
tpd	Max	Min
Via Capacitance	Max	Min

2.2.1. FLIGHT TIME SPECIFICATION

The purpose of the flight time specification is to guarantee that a signal supplied by a driving component is available at the receiving component for sampling. It replaces the output valid delay and input setup time. The two methods are analogous, except that flight time allows the input and output buffer behavior to be matched without major impact to designers or the documentation. In other words, if component A's output is slower than expected, but component B's input is faster than expected, these two can be traded off without having to change the flight time specification. However, if output valid delay and input setup time specifications had been used the specifications would have to be changed. Note that in both cases the time available to the system designer to move the signal from one component to the other is the same.

Flight time is the propagation delay of a signal from a driving component to any receiving component. It is defined as the time difference between the V_{CC} /2 (50%) level of an unloaded output signal and the V_{CC} /2 (50%) level of a receiving signal whose 50% V_{CC} to 65% V_{CC} rise time is greater than or equal to 1 volt/ns. Figure 4 shows the flight time measurement between the 50% V_{CC} points on the unloaded driver and receiver waveforms.

If the rise time between the 50% V_{CC} and 65% V_{CC} points is less than 1 volt/ns, the determination of flight time is slightly more difficult and requires more calculation. In this case the 65% V_{CC} point is extrapolated back to the 50% V_{CC} point using a 1 volt/ns reference slope (i.e., subtract 0.75 ns when V_{CC} = 5V). Figure 5 shows the extrapolation from the 65% V_{CC} point and the resulting flight time measurement.

The calculation of maximum flight time for the 5V tolerant inputs is slightly different than from when both the driving and receiving components have the same V_{CC} levels. For 5V tolerant inputs, flight time is always measured using the falling edge since this will be the worst case. The falling edge is assumed to be the worst case since 5V and 3.3V signals start at the same low logic level, whereas the 5V signal must drop an initial 1.7V just to have the same initial high logic level as a 3.3V signal. Therefore, maximum flight time for the 5V tolerant input is the difference between the 50% 5V V_{CC} level of an unloaded output signal and the 50% 3.3V V_{CC} (or extrapolated from 35% 3.3V V_{CC} if the fall time is less than 1 volt/ns) of the receiving signal on a falling edge.

The minimum flight time for 3.3V and 5V signals is the difference between the 50% V_{CC} level of an unloaded output signal and the 50% V_{CC} level of the receiving signal on a rising edge.

The minimum flight time for 5V tolerant inputs is the difference between the 50% 5V V_{CC} level of an unloaded output signal and the 50% 3.3V level of a receiving signal on a rising edge.

2.2.1.1. Clock Skew

Clock skew has generally been included in system design timing analysis. However, as frequencies increase, controlling clock skew becomes more important. Clock skew is the difference in time of the clock signal arriving at different components. It is measured at 0.8V, 1.5V, and 2.0V.

In synchronous devices, the clock signal defines the point in time in which signals are driven or sampled. It is important that all devices have a common reference. If the reference varies from component to component, the difference must be accounted for to ensure the devices function properly. In other words, clock skew must be subtracted from the clock period when performing a timing analysis of a system.

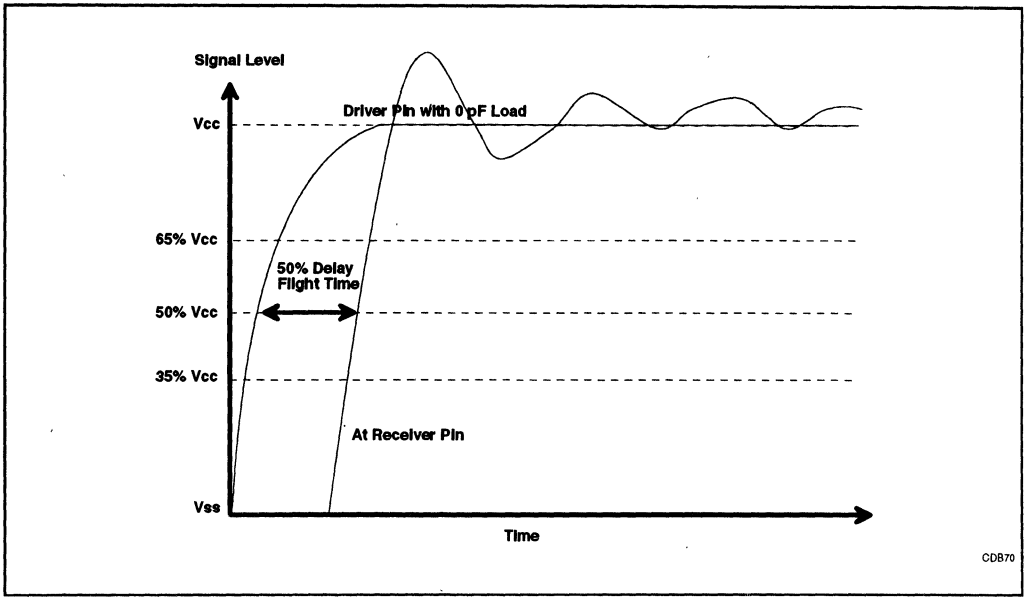


Figure 4. Flight Time Measurement

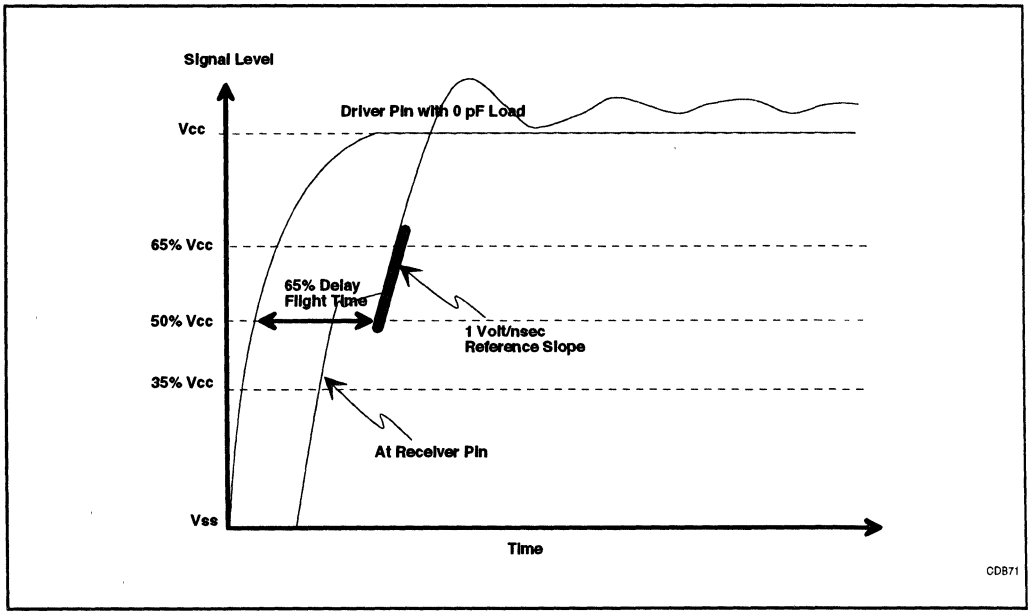


Figure 5. Flight Time Extrapolated from the 65% Point

In the CPU-Cache Chip Set, the maximum clock skew between components in the optimized interface is a specification. This specification is required as a complement of flight time to ensure proper functionality. If clock skew exceeds the specified limit, the excess must be subtracted from the available flight time or the clock period must be increased.

2.2.2. SIGNAL QUALITY SPECIFICATIONS

Acceptable signal quality must be maintained over the entire operating range to insure reliable operation of the chip set. Signal quality consists of two parameters: Ringback, and Settling Time. Figure 6 illustrates these signal quality parameters and how each is measured for a low-to-high transition. The following sections explain each of these in more detail.

Reliable operation means the signals are sampled correctly, do not exhibit false transitions, and that the long term reliability of the component is not affected by overdriving the inputs.

2.2.2.1. Ringback

Excessive ringback can contribute to long-term reliability degradation of the chip set. Ringback is simulated at the input pin of a component using the input buffer model. Ringback can be simulated with or without the diodes that are in the input buffer model.

If simulated without the input diodes, follow the maximum overshoot/undershoot specification. By meeting the overshoot/undershoot specification, the signal is guaranteed not to ringback excessively.

If simulated with the diodes present in the input model, follow the maximum ringback specification.

Overshoot (undershoot) is the absolute value of the maximum voltage above V_{CC} (below V_{SS}). The specification assumes the absence of diodes on the input.

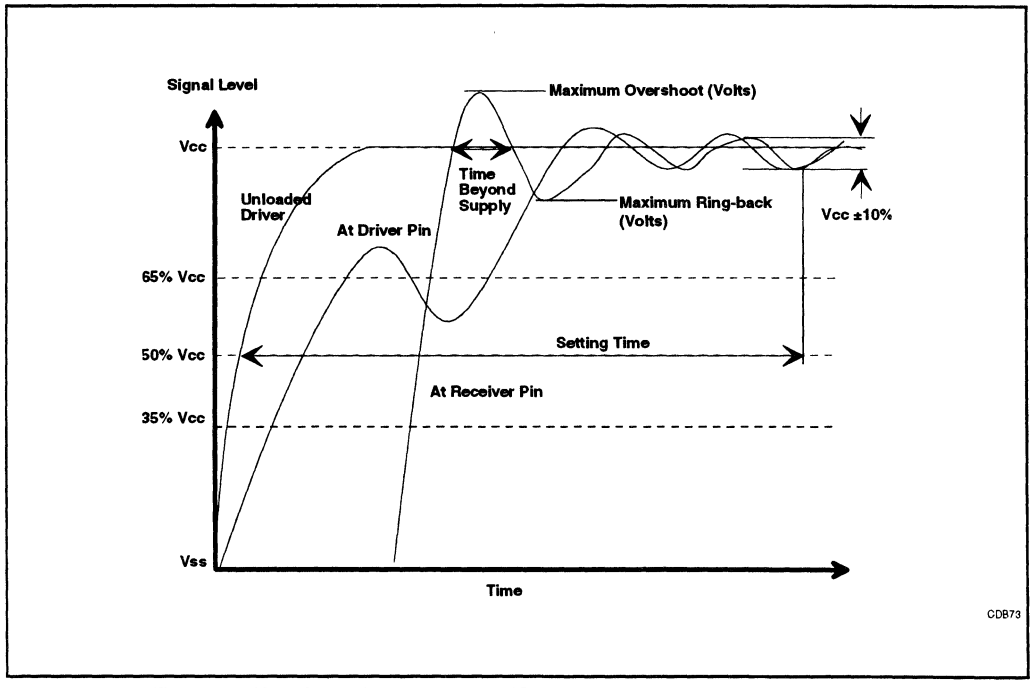


Figure 6. Signal Quality Parameters Measured for Low-to-High Transitions

Maximum overshoot/undershoot on 5V chip set (CLK and PICCLK only) inputs = 1.6V above $V_{CC}5$ (without diodes). Maximum overshoot/undershoot on 3.3V Pentium processor (735\90, 815\100, 1000\120, 1110\133) inputs (not 5V tolerant CLK and PICCLK) = 1.4V above $V_{CC}3$ (without diodes).

Ringback is the maximum absolute voltage at the pin of the receiving component below V_{CC} (or above V_{SS}) relative to V_{CC} (or V_{SS}) level after the signal has reached its maximum voltage level. The input diodes are assumed present. Maximum ringback on inputs = 0.8V (with diodes). Figure 6 illustrates how to measure ringback. Eliminating ringback maintains signal quality by preventing a signal from re-crossing the threshold, causing a false transition to be detected.

2.2.2.2. Settling Time

Settling time is the maximum time required for a signal to settle within 10% of its final value referenced from the time unloaded driver's initial crossing of the 50% V_{CC} threshold. Figure 6 shows how settling time is measured on low-to-high transitions.

Most available simulation tools are unable to simulate settling time so that it accurately reflects silicon measurements. On a physical board, second order effects and other effects serve to dampen the signal at the receiver. Because of all these concerns, settling time is a recommendation or a tool for layout tuning and not a specification.

Settling time is simulated at the slow corner, to make sure that there is no impact on the flight times of the signals if the waveform has not settled. Settling time may be simulated with the diodes included or excluded from the input buffer model. If diodes are included, settling time recommendation will be easier to meet.

The settling time recommendation should be met in order to ensure that a signal transition has completed and is no longer oscillating prior to the next transition. This is important to avoid forcing a signal to transition a distance significantly greater than V_{CC} /2. For example, if a signal is still not settled at the time of its next transition, it may be at a voltage above V_{CC} such as 4.0 volts. Assuming V_{CC} = 3.3 volts, the transition requires the voltage to swing from 4.0 volts (instead of 3.3 volts) to 1.65 volts. This added voltage distance translates into added flight time.

Although simulated settling time has not shown good correlation with physical, measured settling time, settling time simulations can still be used as a tool to tune layouts. Use the following procedure to verify board simulation and tuning with concerns for settling time.

1. Simulate settling time at the slow corner for a particular signal.
2. If settling time violations occur (signal requires more than 12.5 ns to settle to ±10% of its final value), simulate signal trace with DC diodes in place at the receiver pin.
3. If settling time violations still occur, simulate flight times for five consecutive cycles for that particular signal.
4. If flight time values are consistent over the five simulations, settling time should not be a concern. If however, flight times are not consistent over the five simulations, tuning of the layout is required.
5. Note that, for signals that are allocated two cycles for flight time, the recommended settling time is doubled.

Maximum Settling Time to within 10% of V_{CC} is: 12.5 ns at 66 MHz, and 14.2 ns at 60 MHz.

3.0. DIFFERENCES FROM THE PENTIUM® PROCESSOR (735\90, 815\100, 1000\120, 1110\133) CHIP SET LAYOUT

3.1. New Package

The 82498 uses a 296 pin Staggered PGA package, while the 82496 uses a 273 pin standard PGA. The 82498 package is the same as the Pentium processor (735\90, 815\100, 1000\120, 1110\133), without the heat spreader. The 82498 pin out has almost the same pin locations as the 82496. The package change causes pin parasitics to be altered which affects physical routing optimization. In addition, the new package may also make routing more complex as Figure 7 illustrates. The 82498 and 82493 V_{CC} is 3.3V, except for three V_{CC} pins on 82498 and one pin on 82493 which can be set to 5V for 5V tolerant external bus interface inputs.

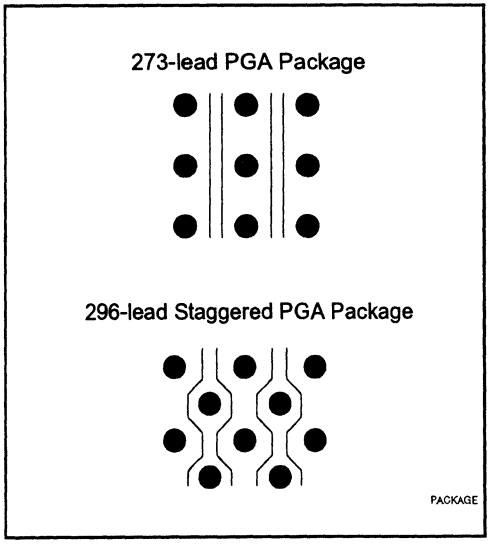


Figure 7. Routing Differences Between the PGA and SPGA Packages

3.2. Power Supply Issues

The Pentium processor (735\90, 815\100, 1000\120, 1110\133) in standard SPGA ceramic package requires 3.3V on all of its $V_{CC}3$ and V_{CCcore} inputs. Both $V_{CC}3$ and V_{CCcore} are power supply pins on the Pentium processor (735\90, 815\100, 1000\120, 1110\133) and should be connected to the same power supply. Therefore, the inputs and outputs of the Pentium processor (735\90, 815\100, 1000\120, 1110\133) are JEDEC standard 3.3V levels. Both inputs and outputs are also TTL-compatible, although the inputs cannot tolerate voltage swings above 3.3V V_{IN3} max. In addition, CLK and PICCLK are also 5V-tolerant so standard clock drivers may be used. Future processors may require mixed voltage power supplies and/or 3.3V clock drivers.

3.3. Board Issues

Split power planes on the board are recommended; one power plane for V_{CCcore} and the other for $V_{CC}3$. Dividing the board into a $V_{CC}3$ island and a V_{CCcore} island does not increase the cost of the board. Extensive simulation and modeling indicate there is no impact to timing or signal quality. Simulation shows that for a gap between the two power planes approximately equal to as much as a half inch, crosstalk noise is still negligible. To minimize the EMI of the split power planes, clock or clock related signal (i.e. strobes) traces should not cross the gap when routed on a layer adjacent to the power plane. Use of vias to connect between signal planes should be minimized, and the signal planes must be within 8 mils of the reference plane. High speed signals such as data lines and low address bits can be routed on any layer, but the number of those traces crossing the gaps on an adjacent layer should be minimized (ideally to none). A bypass capacitor should be placed across the gap no more than 1 inch away from the signals that do cross the gap. Liberal decoupling should be placed around the processor core and I/O; refer to the reference schematics.

3.4. Routing Topologies

Minor changes have been made to some of the routing topologies from application note AP-481: *Designing with the Pentium® Processor, 82496 Cache Controller and 82491 Cache SRAM CPU-Cache Chip Set.* The following tables give a quick reference to what changes, if any, have been made. For some of the topologies, the only change is the number. These tables are provided to enable a designer to create a new 82498/82493 design by using a previous 82496/82491 design as a reference. Table 2 refers to the topologies of the 1-Mbyte reference design as compared to the 256K 82496/82491 reference design. Table 3 refers to the topologies of the 2-Mbyte reference design as compared to the 512K 82496/82491 reference design.

Table 2. 1-Mbyte Reference Design Topology Differences

82496/ 82491 Topology #	82498/ 82493 Topology #	Changes
1	1	Only one trace from the Pentium® processor
1b	3b	Only one trace from the cache controller
3	3b	Only one trace from the cache controller
3a	4	Only one trace from the Pentium processor
3b	3	None
4	2	Different routing
5	5	None
10–19	10–19	

Table 3. 2-Mbyte Reference Design Topology Difference

82496/ 82491 Topology #	82498/ 82493 Topology #	Changes
1	1	Different routing, diodes added
2	—	—
3	3b	Different routing, diodes added
4	4	Different routing, diodes added
5	3	Different routing, diodes added
6	2	None
7	5	Slightly different routing
8–14	8–14	None
15	16	Different routing
16	15	None
17	10	None
18	11	Different routing

4.0. 1-MBYTE PENTIUM® PROCESSOR (735\90, 815\100, 1000\120, 1110\133) 82498/82493 CPU-CACHE CHIP SET LAYOUT EXAMPLE

This chapter contains an example layout design for Intel's 1-Mbyte CPU-Cache Chip Set. Intel has simulated and validated the design by measuring the flight time and signal quality parameters in the board design example.

The intent of the design example is to provide system designers a starting point. It provides one solution of how Pentium processor (735\90, 815\100, 1000\120, 1110\133), 82498 cache controller, and 82493 cache SRAM components can be placed and routed to ensure flight time and signal quality specifications are met. It is not the only solution. System designers can alter the layout to meet their system requirements as long as the flight time and signal quality specifications are met.

4.1. Layout Objectives

The 1-Mbyte layout is an example of a CPU-Cache chip set arrangement that meets Intel's chip set specifications. The layout consists of 1 Pentium processor (735\90, 815\100, 1000\120, 1110\133), 1 82498 cache controller, and 10 82493 cache SRAMs for a 1-Mbyte second-level cache with parity. Although the layout is specifically designed for a chip set with parity, conversion to a non-parity layout will also be discussed.

This example layout follows the chip set's flight time and signal quality specifications. In addition to meeting those specifications, these objectives were used:

1. To design the optimized portion of the interface so that the layout is not limited by interconnect performance. By not artificially creating any critical paths, the interface can yield maximum performance of the chip set.
2. To be consistent with EMI and thermal requirements.
3. To have the layout be used as a validation and correlation vehicle by Intel. Intel used the layout to validate the optimized interface of the chip set, and measure flight times and signal quality.

Provided are complete specifications for a board layout: part lists, board layer plots, and the electronic files in Gerber format. Also provided are a set of topologies and

line lengths so it will be easy to understand how the layout was generated.

4.2. Component Placement

To meet flight time with clock skew restrictions the parts should be placed in relative proximity to each other. At the same time, ensure that the layout's Memory Bus Controller (MBC) interface signals are routable. Figure 8 illustrates how the chip set components are placed in the layout example. The dot indicates the location of pin 1. Figure 8 is a component top view of the layout:

4.3. Signal Routing/Topologies

Tables 4 and 5 list the signal nets and their corresponding topologies for the optimized and external interfaces of the CPU-Cache Chip Set.

All chip set signals in the optimized interface fall into six groups: Low addresses, High addresses, Pentium processor (735\90, 815\100, 1000\120, 1110\133) control, 82498 control, CPU data and parity, and Byte enables. Within each group are subsets of signals that share common origination and destination points. Each subset has a unique routing called a "topology". Groups, subsets, and topologies are listed in Table 4.

Topologies are given only for signals that are routed to multiple chips. It is the system designer's responsibility for routing the "point-to-point" signals such as CADS#.

Topologies are also supplied for the external interface. These topologies provide channels for routing signals from the chip set components to the periphery where they can be connected to the memory bus and memory bus controller (MBC). However, topologies are not supplied for point to point signals in the MBC interface (e.g. CRDY#). Instead, the system designer must optimize these for the particular application.

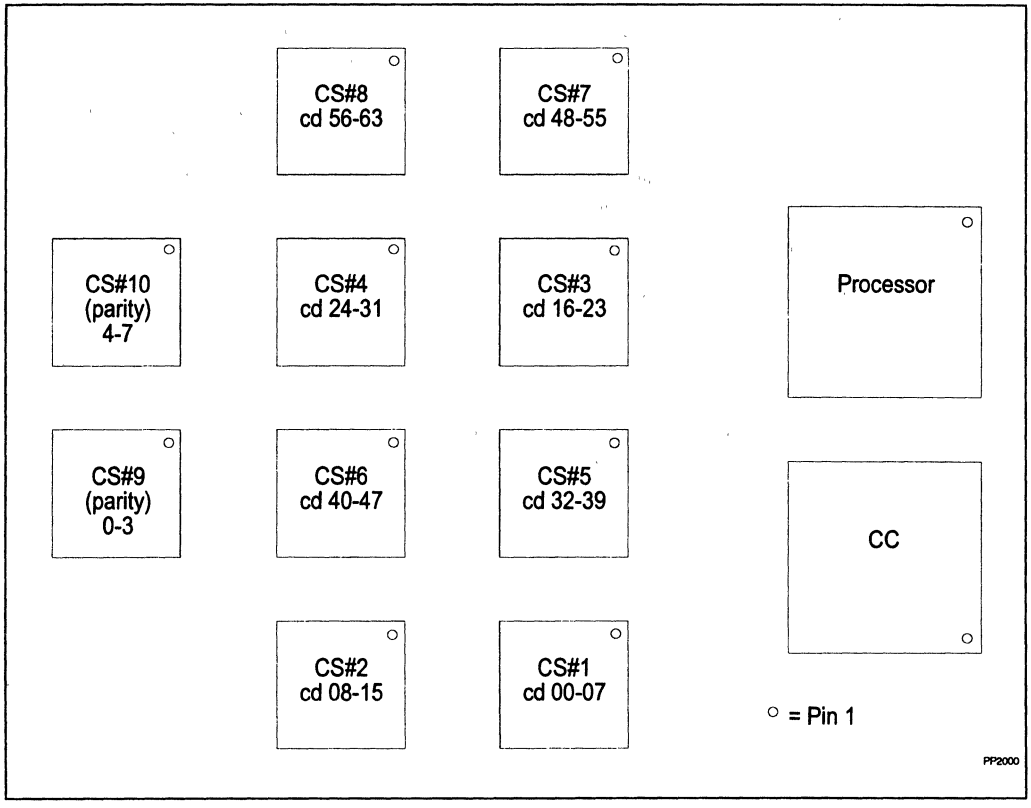


Figure 8. Component Placement

Table 4. TT_1MbOptInterface1-Mbyte Optimized Interface Signal Net/Topology Assignments

Grouping	Routing Requirements	Topology
Low Addresses		
(PA3-PA18)	Connected to all isolated interface components. Must be routed to optimize delay and signal quality at all points. La, Lb as short as possible.	1
High Addresses		
(PA19-21)	Resistors on PA19-P21 La as short as possible but >5" (see sect. 4.5).	2b
(PA22-PA31)	Point to point links. Must be kept as short as possible but longer than 5".	2

Table 4. TT_1MbOptInterface1-Mbyte Optimized Interface Signal Net/Topology Assignments (Contd.)

Grouping	Routing Requirements	Topology
(HITM#, W/R#)	Same as low addresses. La, Lb as short as possible.	1
(ADS#)	Same as low addresses. La as short as possible.	3
(ADSC#, AP, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC)	Same as high addresses. La as short as possible, but longer than 5".	2
82498 Control		
(BLAST#, BOFF#, BUS#, MAWEA#, WBA, WBTYP, WBWE#, BRDYC2#, MCYC#, WRARR#, WAY)	Must be routed to optimize delay and signal quality at the 82493s. Lb as short as possible but longer than 5".	3b
(BLEC#)	Not routed to parity 82493s. La as short as possible.	4
(AHOLD, BRDYC1#, EADS#, INV, KEN#, NA#, WB/WT#, EWBE#)	Same as high addresses.	2
CPU Data and Parity		
(CD0-CD63, CP0-CP7)	Same as high addresses.	2
Byte Enables		
(CBE0#-CBE7#)	Connected to the Pentium processor (735\90, 815\100, 1000\120, 1110\133), two 82493s and one parity 82493. La as short as possible but longer than 5".	5

Table 5. TT_1MbExtInterface1-Mbyte External Interface Signal Net/Topology Assignments

Signal	Topology
RESETCPU	10
CRDY#, RESETC5	11
MBRDY#, MOCLK, MDOE#, MSEL#, MEOC#	12 & 13
MFRZ#, MZBT#, MCLK	14
BRDY0#, CLK0	15
CLK, BRDY#, MEOC#	16
MD0-MD63, Parity0-7	19

Table 5 lists the topologies provided for the MBC interface signals which are not point to point.

Figures 9 through 23 contain the topologies which are described in Tables 4 and 5. A topology is a graphical representation how specific sets of signals are routed. A topology shows the components that share a specific signal and the relative lengths of the traces between components.

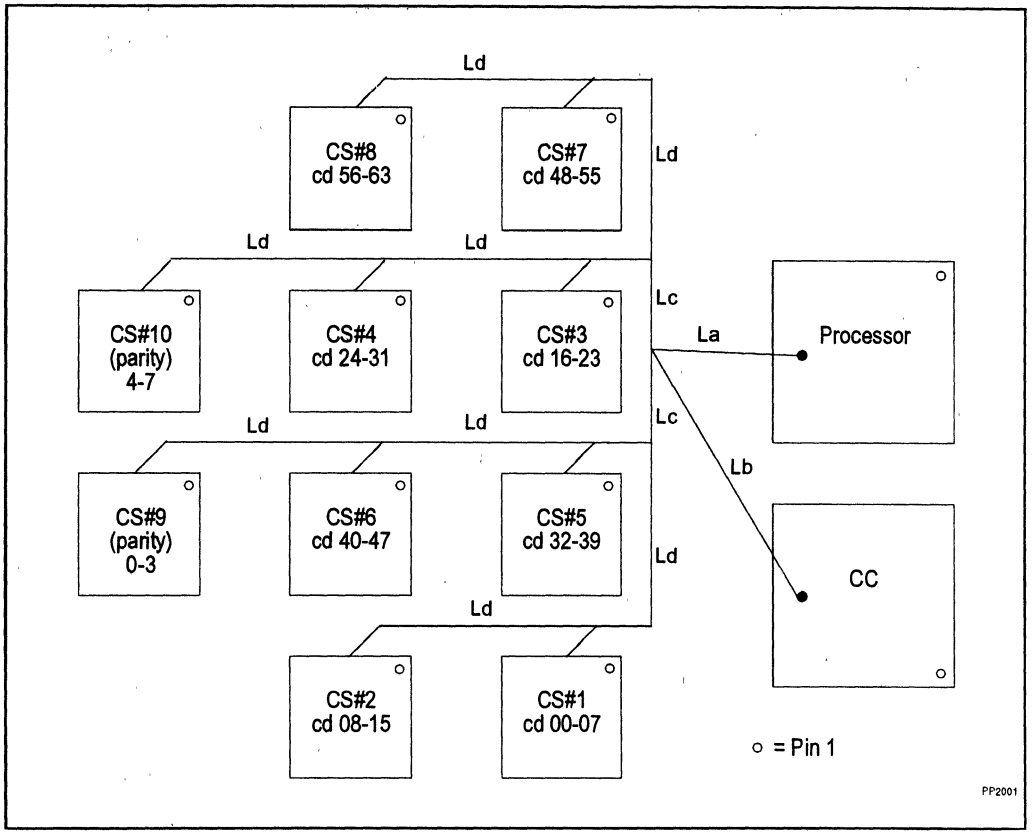


Figure 9. Topology 1

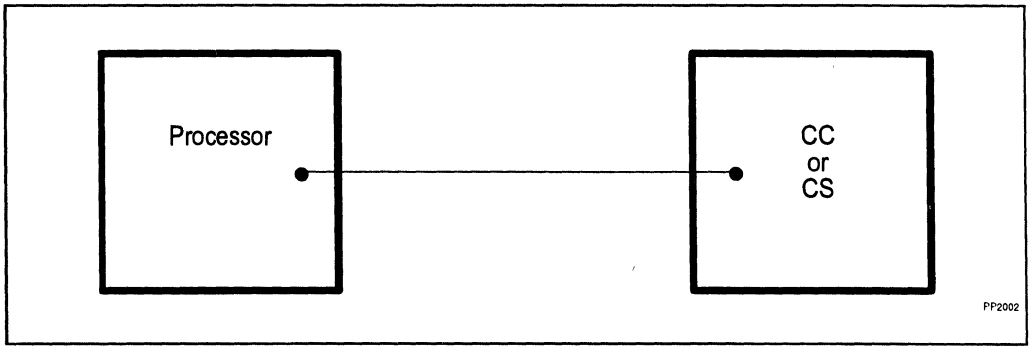


Figure 10. Topology 2

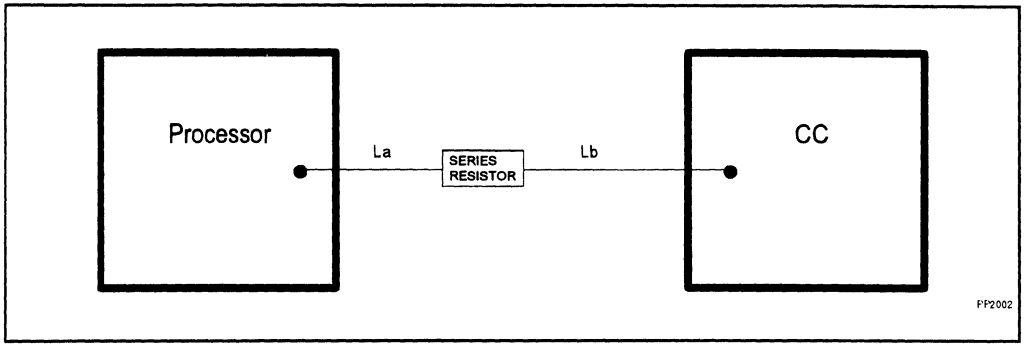


Figure 11. Topology 2b

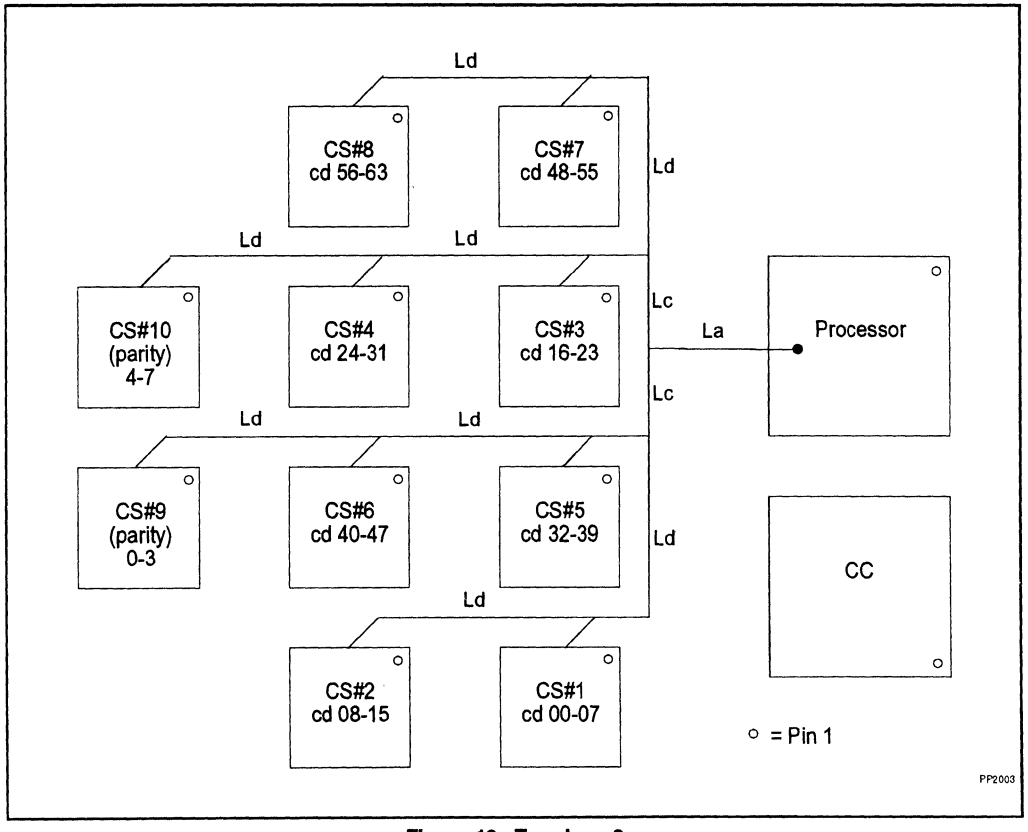


Figure 12. Topology 3

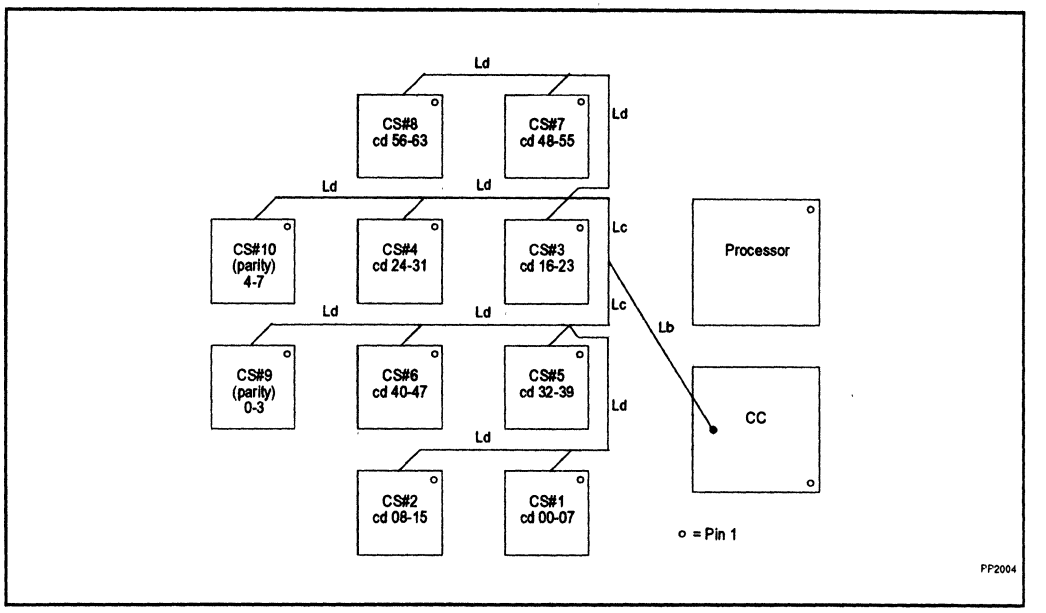


Figure 13. Topology 3b

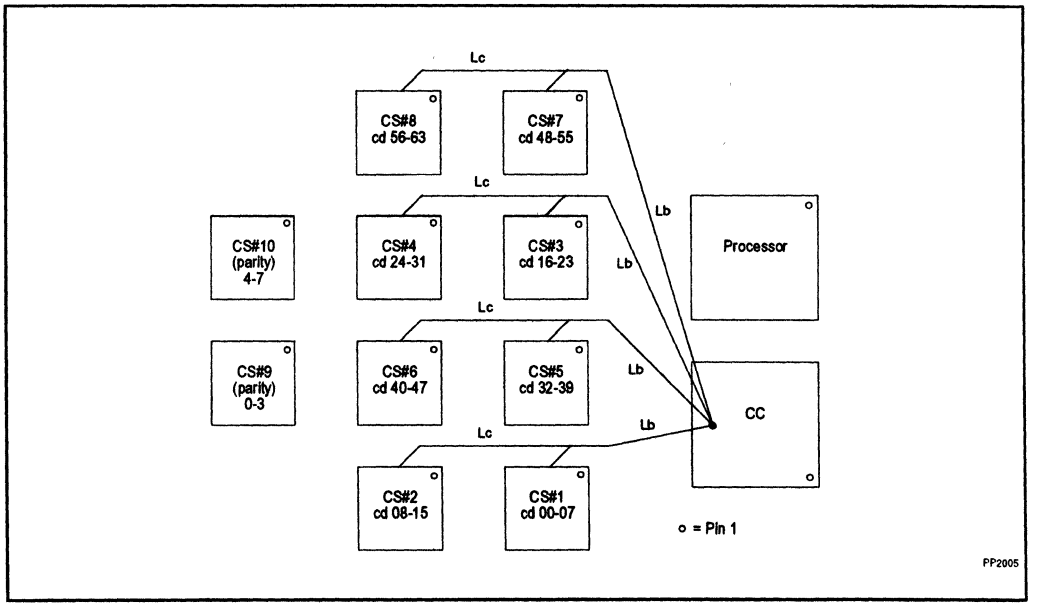


Figure 14. Topology 4

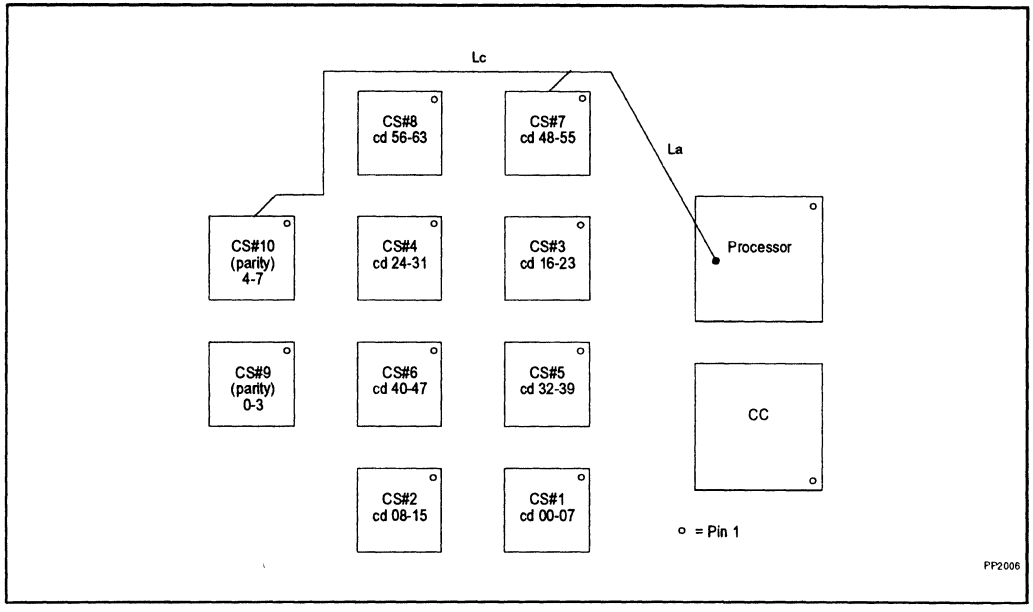


Figure 15. Topology 5

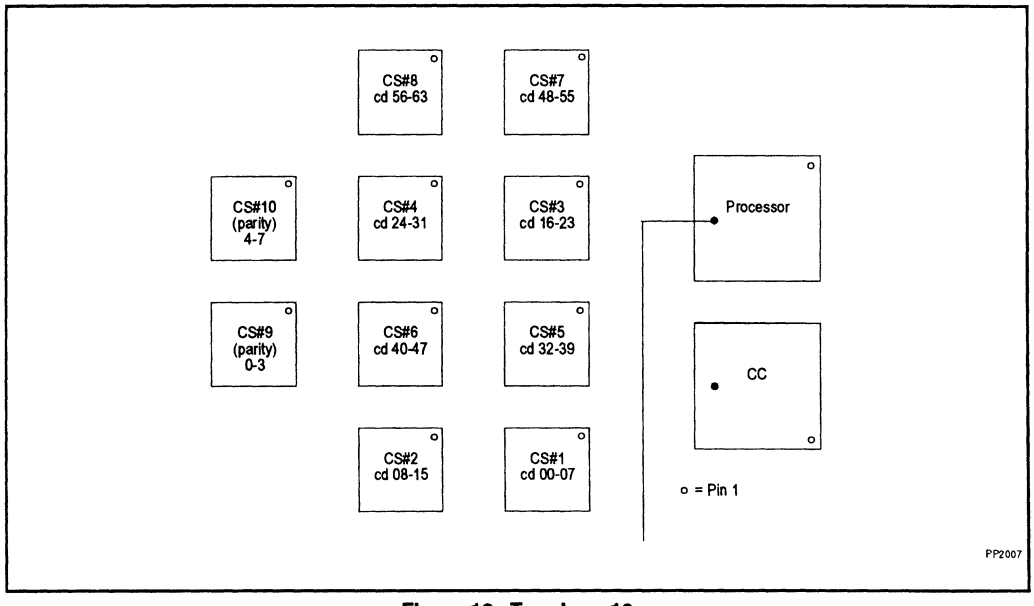


Figure 16. Topology 10

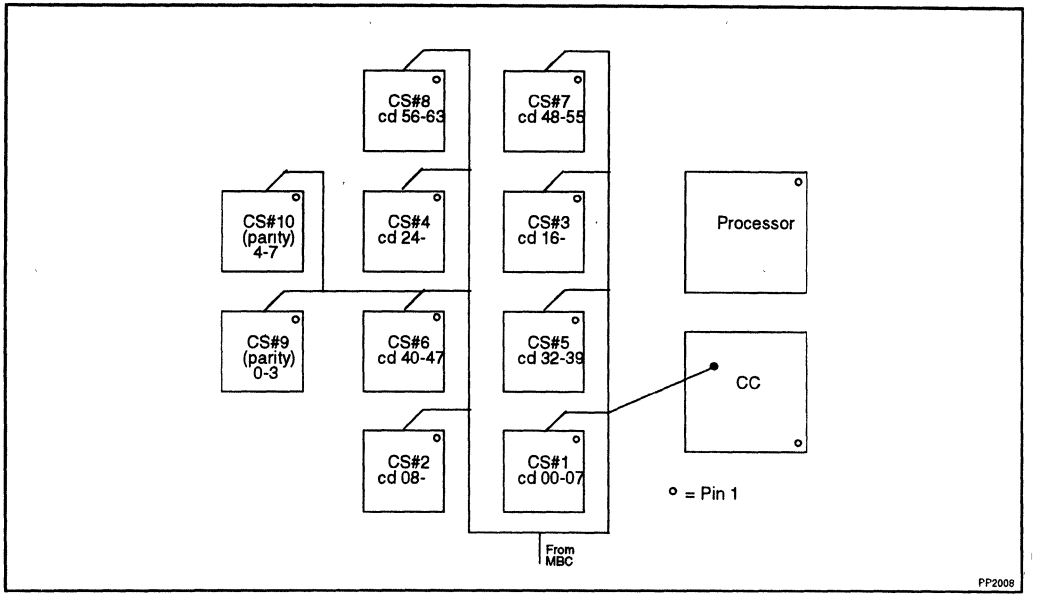


Figure 17. Topology 11

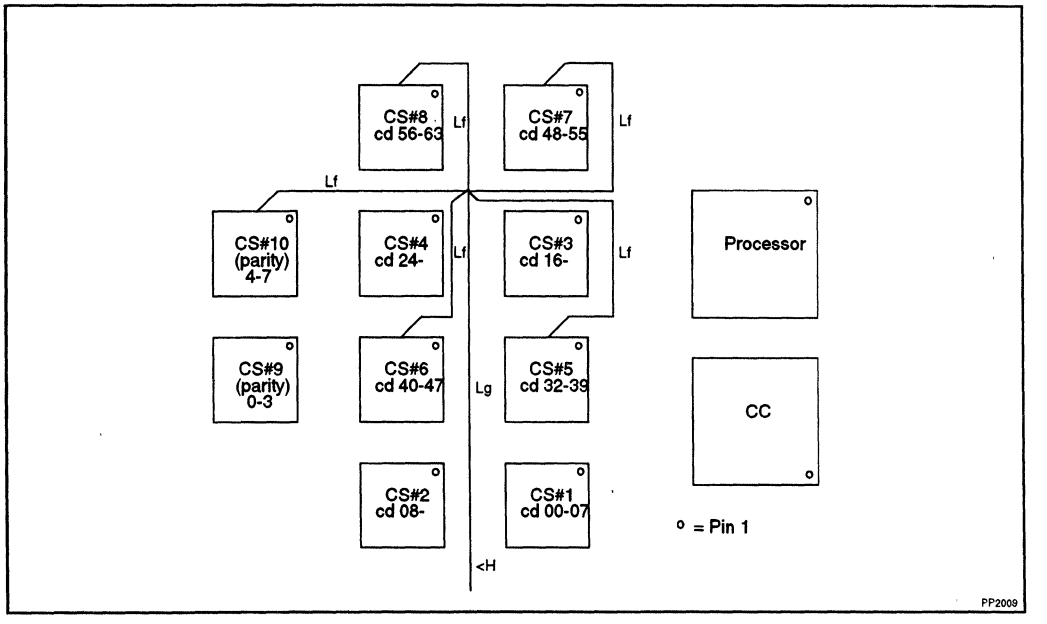


Figure 18. Topology 12

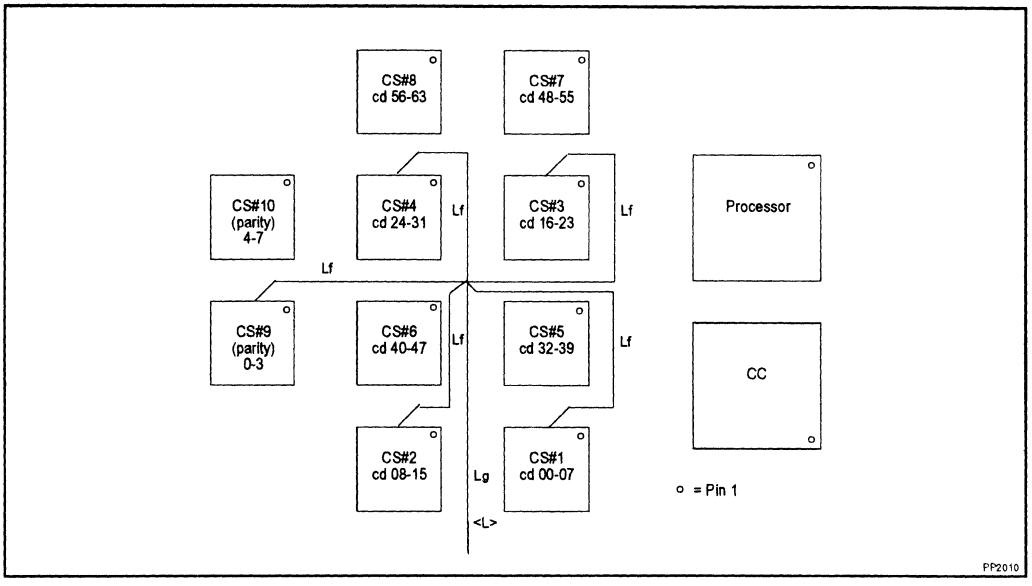


Figure 19. Topology 13

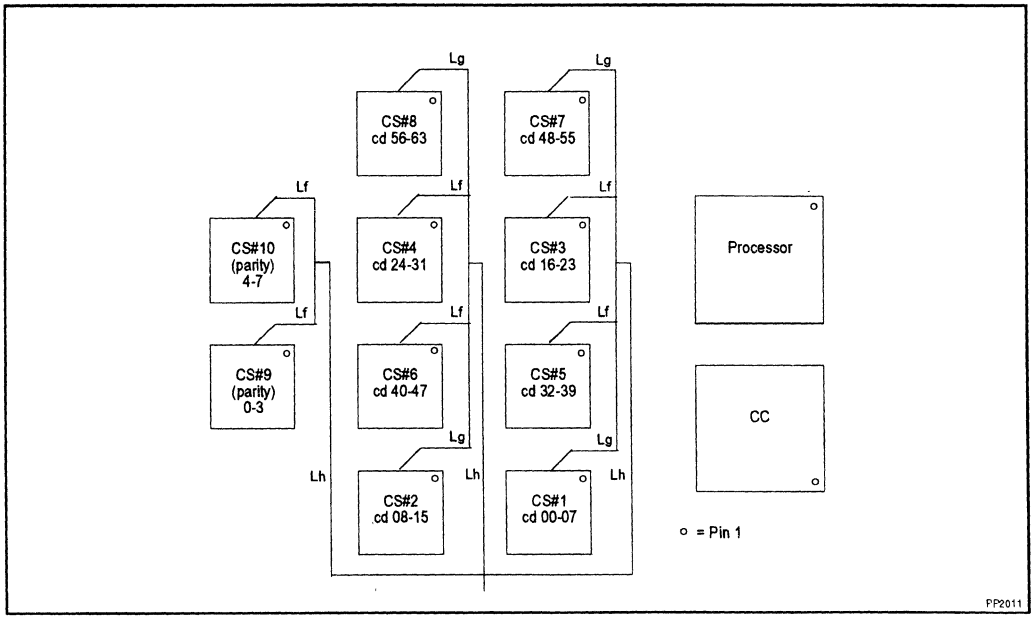


Figure 20. Topology 14

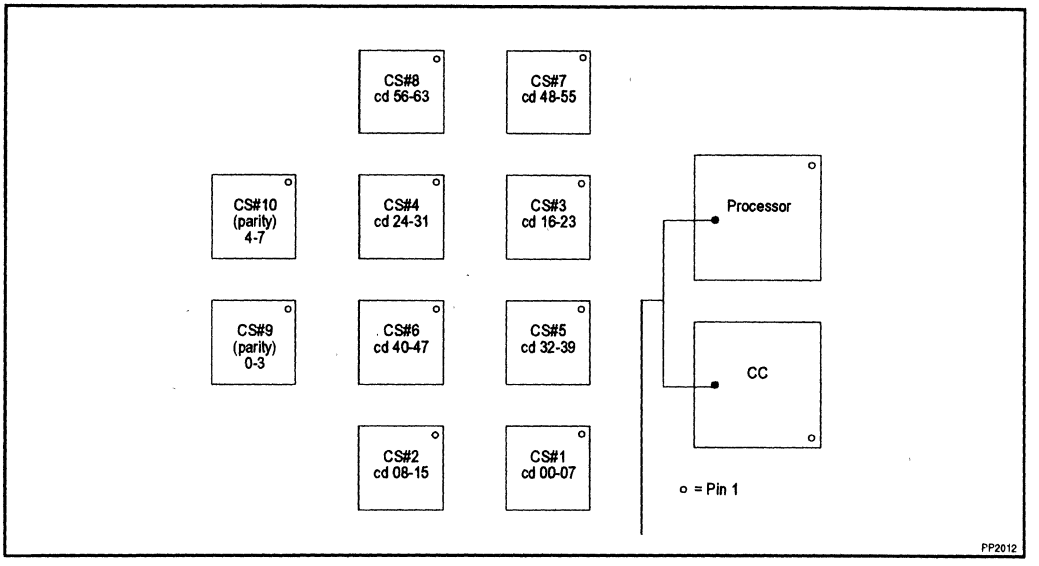


Figure 21. Topology 15

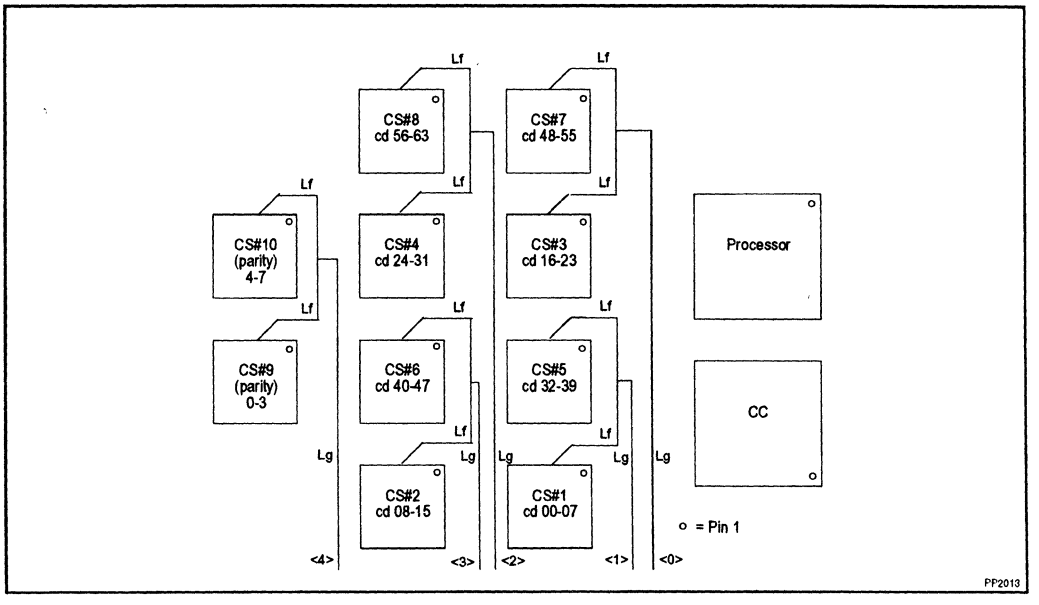


Figure 22. Topology 16

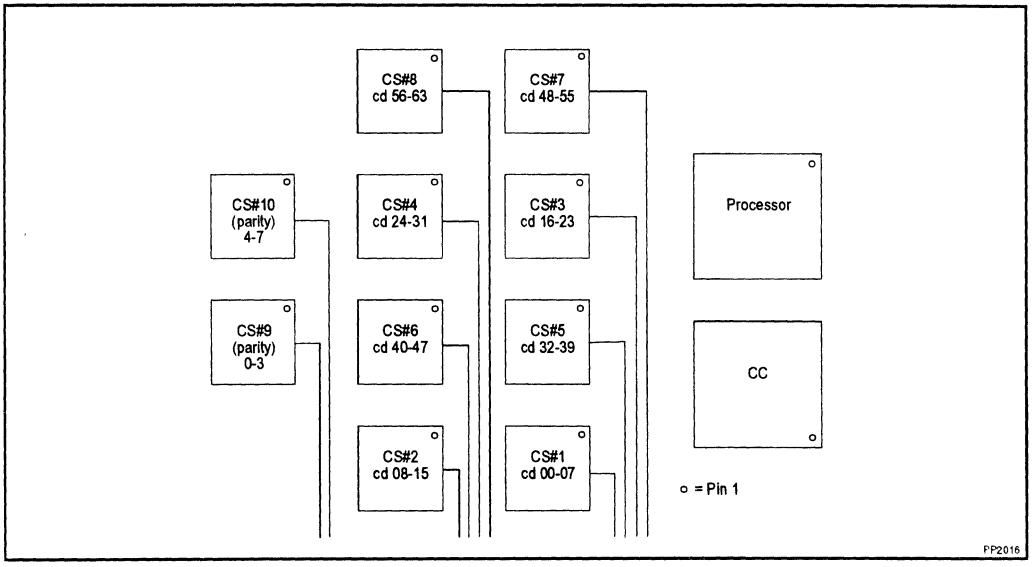


Figure 23. Topology 19

4.4. Board/Trace Properties

Specific board and trace properties were assumed while performing the simulations to optimize the chip set layout. These properties were used as the specification or guideline the board manufacturer should use in building boards. Figure 24 provides the board layer stackup.

Table 6 lists the minimum and maximum trace characteristics. These parameters along with the board material determine the spacing between layers and the total board thickness. See Table 7.

4.5. Design Notes

1. All fast-switching signals are routed near the power and ground planes on inner layers of the board to minimize EMI effects. However, two sets of signals are routed on the top layer of the board: BRDYC1#, and JTAG signals. BRDYC1# is routed on top to take advantage of the higher trace velocity. JTAG signals can be routed on any layer because they are low-speed signals and will probably be rerouted by each customer to suit individual needs.
2. Resistor R1(0Ω) on page 2 of the 1-Mbyte and 2-Mbyte schematics is used to set the Pentium processor's (735\90, 815\100, 1000\120, 1110\133) configurable output buffers (A3-A20, ADS#, W/R#, and HITM#). When the resistor is included the buffers are set to the extra large size. When it is not included (BUSCHK# internally pulled high) the buffers are set to the large size. Intel currently recommends the large buffers be used for the 1-Mbyte layout example. The 0Ω resistor should be designed in your design for flexibility.
3. The 82498 output buffers that drive the 82493 inputs must also be configured to be large. This is done by driving 82498 CLDRV [BGT#] high during reset. The 82498 and 82493 memory bus buffer sizes must be controlled by the Memory Bus Controller. Please refer to the *Pentium® Family Developer's Manual Volume 2: 82496/82497/82498 Cache Controller and 82491/82492/82493 Cache SRAM.*
4. Series termination resistors were added to the nets PA19, PA20, PA21 and several other nets to control overshoot. A value of 24Ω is recommended, but the value depends on overshoot measurement on the actual design.
5. Several Topologies require different line widths on the same trace. This information can be extracted automatically using a Quad Design translation

tool. For extraction procedure, please refer to Quad Design's user's manual, "Preparing PCB Design Databases for Simulation with TLC/XTK."

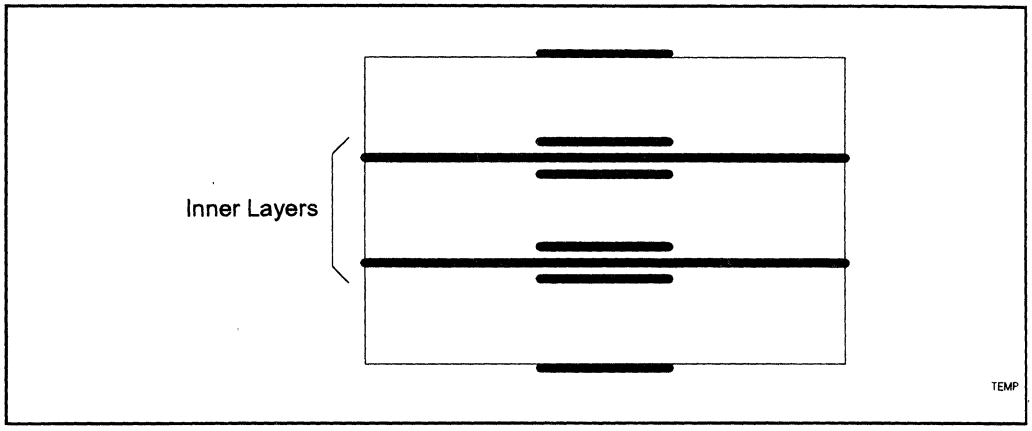


Figure 24. Board Layer Stackup

Table 6. Trace Characteristics

	4 Inner Layers	2 Outer Layers
Width/Space	5/5 Mils	8/8 Mils
Z_0	65Ω ± 10%	75Ω ± 10%
Velocity	1.85 to 2.41 ns/ft	1.5 to 2.1 ns/ft

Table 7. Other Printed Circuit Board Geometries

Via Pad	25 Mils
Via Hole	10 Mils
Pentium® Processor (735\90, 815\100, 1000\120, 1110\133) SPGA Pad	45 Mils
Pentium Processor (735\90, 815\100, 1000\120, 1110\133) SPGA Hole	29 Mils
82498 PGA Pad	55 Mils
82498 PGA Hole	38 Mils
82493 Surface Mount Package Pad	70x13 Primary Side
Layout Grid	5 Mils

6. Layout Rules/Common Definitions/Explanations: Identical symbols for transmission lines means that the lines should be identical in Length, Width, and Configuration (e.g. Strip, Microstrip etc.). The term "as short as possible but greater than a certain length" is defined as follows. In order to satisfy the maximum flight time requirements, generally, it is desired that the transmission lines be as short as possible. Many signals however have tight minimum flight time requirements. For these signals it is necessary to lengthen the lines, but not too long so as to violate the maximum flight times. Whenever possible, a length window is given so that both min timings and max timings are specified. When both margins are very tight and the signals are heavily loaded with too many transmission lines, it is very difficult to specify a range of lengths for every segment on that topology without being too restrictive. Instead the strategy was to lengthen a specific segment to satisfy min timings and shorten all other segments as short as possible to meet max timings. The exact lengths can be extracted from the layout.

4.6. Explanation of Information Provided

The following sections outline the design files associated with the 1-Mbyte CPU-Cache Chip Set design example that are available from Intel. These files are provided to simplify the task of porting the design example into a specific design. By using these files, designers may eliminate or minimize the amount of duplicate effort when using the design example as the basis for their design. The following items are available:

- Schematics
- I/O Model Files
- Board Files
- Bill of Materials
- Photoplot Log
- Netlist Report
- Placed Component Report
- Artwork for Each Board Layer
- Trace Segment Line Lengths

Hard copies of the schematics are provided in the following section. ASCII or soft copies of all the information are available from Intel Order Number 242658.

4.6.1. SCHEMATICS

Schematics for the 1-Mbyte CPU-Cache Chip Set design example were created using Cadence/Valid Concept Schematic Capture program V1.5-p3. The schematics are 11 pages long and are provided in A-size format. Both the Cadence and the postscript files (A-size format) are available from Intel as described above.

4.6.2. I/O MODEL FILES

All electrical I/O simulations were performed using TLC* V4.2.8 from Quad Design Technology, Inc. The simulations were performed at the fast and slow corners to verify that all signal quality and flight time specifications were met. The files used for these simulations are available from Intel as described above. These files include the topology, model, and control files needed to run the simulations for all nets in the optimized interface.

4.6.3. BOARD FILES

The board files for the design example were created using Allegro* V6.1 from Cadence Design Systems, Inc. The files are available from Intel as described above. These files may be used to import the design example into a specific system design.

NOTE

Some changes to the layout and nets may be necessary to complete importing these files into a specific system design.

4.6.4. BILL OF MATERIALS

The bill of materials file was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

4.6.5. PHOTOPLOT LOG

The photoplot log file was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

4.6.6. NETLIST REPORT

The netlist report was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

4.6.7. PLACED COMPONENT REPORT

The placed component report was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

4.6.8. ARTWORK FOR EACH BOARD LAYER

The artwork for the six board layers was created using Allegro V6.1 from Cadence Design Systems, Inc. The files are available from Intel in a Gerber* format as described above.

4.6.9. TRACE SEGMENT LINE LENGTHS

The file containing the trace segment line lengths was created from the artwork files of section 4.6.8. All lengths are provide in mils (1/1000 inch). The stubs listed in the tables are associated with the pin escapes required for the 82493s. The file is available from Intel as described above.

4.6.10. I/O SIMULATION RESULTS FOR EACH NET

Electrical simulations were performed on each net within the optimized interface of the 1-Mbyte CPU-Cache Chip Set design example. The simulations were done at the fast and slow corners to verify that signal quality and flight time specifications are met. The simulations were done using TLC V4.2.8 from Quad Design Technology, Inc. using the files described in section 4.6.2. Table 8 summarizes the slow corner simulation results and Table 9 summarizes the fast corner simulation results.

The flight time specifications listed in Tables 8 and 9 are the 60/66-MHz flight times.

Table 8. 1-Mbyte Slow Corner Simulation Results

Net	Flight Time (ns)			Signal Quality	
	Maximum Flight Time	Maximum Flight Time Specification	Flight Time Margin	Settling Time (ns)	Overshoot (V)
82498 Driving					
PA3-18	5.72	11.2	5.48	15.91	0
PA19-31	2.47	3.2	0.73	7.1	0
AHOLD	0.77	1.5	0.73	3.88	0.11
AP	1.23	3.2	1.97	4.5	0.1
BRDYC1#	0.72	1.5	0.78	3.86	0.09
EADS#	0.8	1.5	0.73	3.9	0.06
EWBE#	0.85	1.5	0.65	3.95	0.18
INV	1.39	1.5	0.11	4.53	0.27
KEN#	0.77	1.5	0.73	3.88	0.11
NA#	0.74	1.5	0.76	3.79	0.09
WB/WT#	0.92	1.5	0.58	3.91	0.19
BLAST#	2.42	3.2	0.78	5.55	0.46
BLEC#	1.89	2.5	0.61	5.73	0.06
BOFF1# at CPU		1.5			
BOFF2# at SRAM	1.87	3.2	1.33	4.87	0.36
BRDYC2#	2.3	3.2	0.9	5.42	0.4
BUS#	2.14	3.2	1.06	5.24	0.35
MAWEA#	2.13	3.2	1.07	5.24	0.26
MCYC#	2.14	3.2	1.06	5.21	0.33
WAY	2.27	3.2	0.93	5.36	0.44
WBA	2.1	3.2	1.1	5.24	0.36
WBTYP	2.19	3.2	1.01	5.22	0.47
WBWE#	2.26	3.2	0.94	5.41	0.29
WRARR#	2.37	3.2	0.83	5.53	0.37

Table 8. 1-Mbyte Slow Corner Simulation Results (Contd.)

Net	Flight Time (ns)			Signal Quality	
	Maximum Flight Time	Maximum Flight Time Specification	Flight Time Margin	Settling Time (ns)	Overshoot (V)
Pentium® Processor (735\90, 815\100, 1000\120, 1110\133) Driving					
PA3-18	2.94	3.7	0.76	22.19	0.41
PA19-31	1.66	1.8	0.14	4.99	0.09
CP0-7, CD0-63	1.47	2.0	0.53	4.75	0.17
ADS#	2.27	3.3	1.03	5.27	0.45
HITM#	2.37	3.6	1.23	5.43	0.5
WR#	2.43	3.6	1.17	5.52	0.52
ADSC#	1.36	1.5	0.14	4.7	0.16
AP	1.45	1.5	0.05	4.62	0.09
CACHE#	1.45	1.5	0.05	4.79	0.17
D/C#	1.36	1.5	0.14	4.7	0.16
LOCK#	1.36	1.5	0.14	4.7	0.16
M/IO#	1.46	1.5	0.04	4.8	0.18
PCD	1.36	1.5	0.14	4.7	0.16
PWT	1.36	1.5	0.14	4.7	0.16
SCYC	1.34	1.5	0.16	4.68	0.16
CBE0-7#	2.08	2.8	0.72	6.16	0.07
82493 Driving					
CP0-7, CD0-63	1.93	2.3	0.37	4.91	0.15

Table 9. 1-Mbyte Fast Corner Simulation Results

Net	Minimum Flight Time (ns)			Signal Quality (V)		
	Minimum Flight Time	Minimum Flight Time Specification	Minimum Flight Time Margin	Overshoot	Overshoot Specification	Overshoot Margin
82498 Driving						
PA3-18	1.53	0.8	0.73	0.77	2.6	1.83
PA19-31	0.86	0.6	0.26	2.03	2.6	0.57
AHOLD	0.65	0.6	0.05	2.49	2.6	0.11
AP	0.86	0.6	0.26	2.01	2.6	0.59
BRDYC1#	0.61	0.6	0.01	2.45	2.6	0.15
EADS#	0.68	0.6	0.08	2.34	2.6	0.26
EWBE#	0.72	0.6	0.12	2.53	2.6	0.07
INV	1.03	0.6	0.43	2.43	2.6	0.17
KEN#	0.65	0.6	0.05	2.48	2.6	0.12
NA#	0.63	0.6	0.03	2.45	2.6	0.15
WB/WT#	0.95	0.6	0.35	2.61	2.6	-0.01
BLAST#	1.33	0.8	0.53	2.29	2.6	0.31
BLEC#	0.94	0.8	0.14	1.59	2.6	1.01
BOFF# at CPU	0.84	0.8	0.04	2.57	2.6	0.03
BOFF# at SRAM	1.06	0.8	0.26	2.11	2.6	0.49
BRDYC2#	1.27	0.8	0,47	2.25	2.6	0.35
BUS#	1.13	0.8	0.33	2.02	2.6	0.58
MAWEA#	1.11	0.8	0.31	1.93	2.6	0.67
MCYC#	1.14	0.8	0.34	2.04	2.6	0.56
WAY	1.26	0.8	0.46	2.25	2.6	0.35
WBA	1.12	0.8	0.32	1.99	2.6	0.61
WBTYP	1.27	0.8	0.47	2.34	2.6	0.26
WBWE#	1.19	0.8	0.39	2.01	2.6	0.59
WRARR#	1.28	0.8	0.48	2.18	2.6	0.42

Table 9. 1-Mbyte Fast Corner Simulation Results (Contd.)

Net	Minimum Flight Time (ns)			Signal Quality (V)		
	Minimum Flight Time	Minimum Flight Time Specification	Minimum Flight Time Margin	Overshoot	Overshoot Specification	Overshoot Margin
Pentium® Processor (735\90, 815\100, 1000\120, 1110\133) Driving						
PA3-18	1.07	0.8	0.27	1.49	2.6	1.11
PA19-31	1.04	0.6	0.44	2.11	2.6	0.49
CP0-7, CD0-63	1.1	0.8	0.3	2.01	2.6	0.59
ADS# at SRAM	1.33	0.8	0.53	2.28	2.6	0.32
HITM#	1.35	0.8	0.55	2.42	2.6	0.18
WR#	1.11	0.8	0.06	2.02	2.6	0.58
ADSC#	0.86	0.6	0.26	2.02	2.6	0.58
AP	0.93	0.6	0.33	2.08	2.6	0.52
CACHE#	0.93	0.6	0.33	2.04	2.6	0.56
D/C#	0.86	0.6	0.26	2.02	2.6	0.58
LOCK#	0.86	0.6	0.26	2.02	2.6	0.58
M/IO#	0.94	0.6	0.34	2.04	2.6	0.56
PCD	0.86	0.6	0.26	2.02	2.6	0.58
PWT	0.86	0.6	0.26	2.02	2.6	0.58
SCYC	0.85	0.6	0.25	2.05	2.6	0.55
CBE0-7#	1	0.6	0.4	2.21	2.6	0.39
82493 Driving						
CP0-7, CD0-63	1.05	0.8	0.25	2.07	2.6	0.53

Note: An overshoot specification voltage in parentheses denotes the value for V_{SS} if different from V_{CC}.

PP/CC/CS 1MB INTERFACE

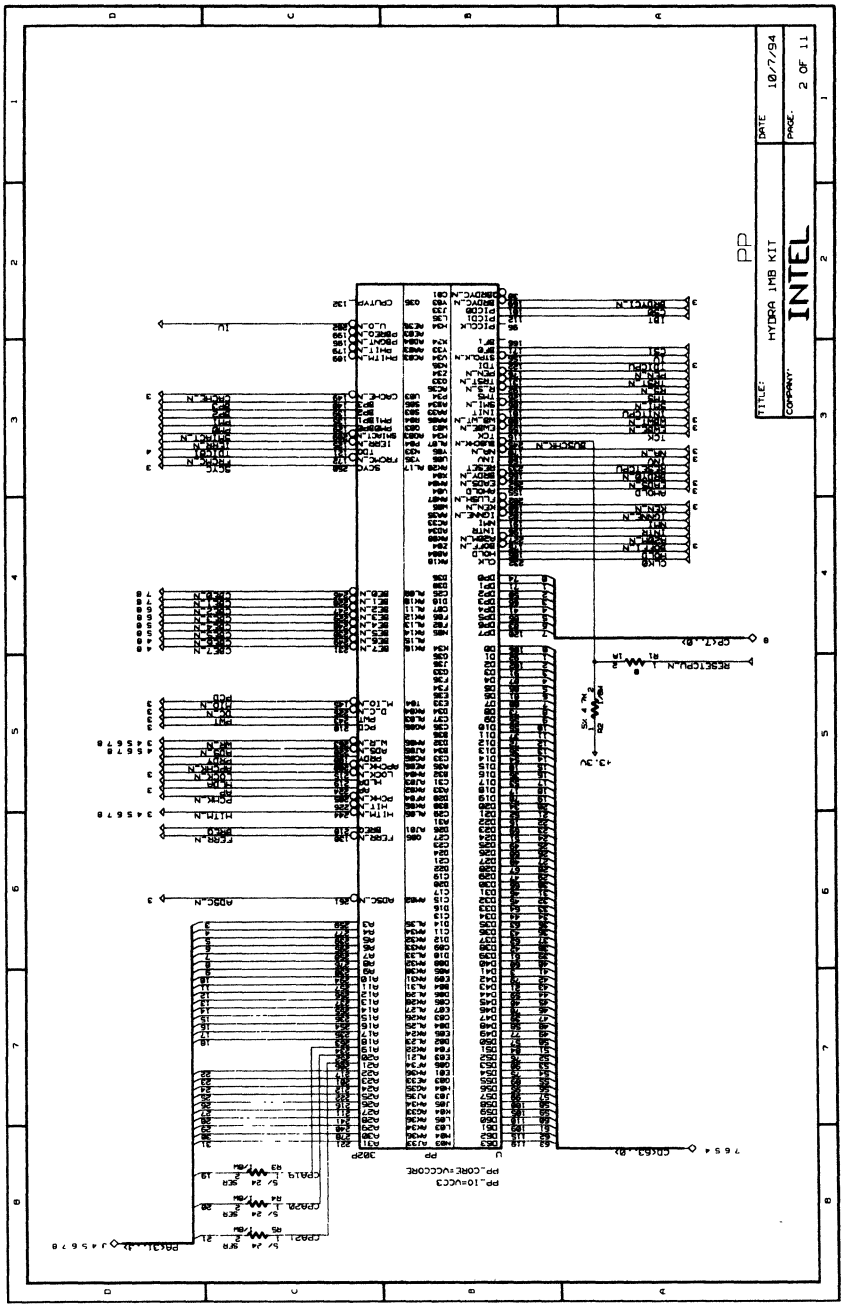
LIST OF POWER SUPPLY PINS AND NO CONNECT PINS:

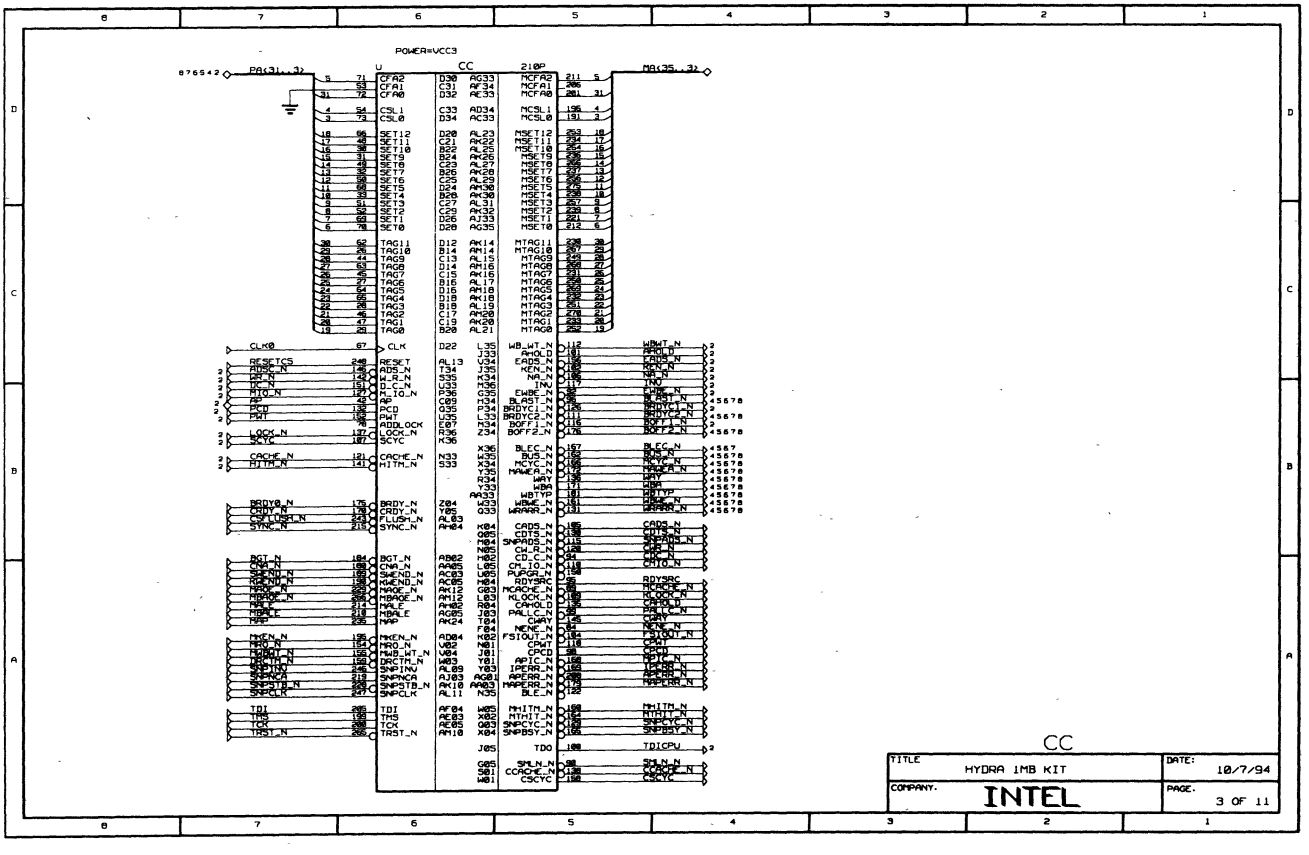
REF DES	NAME	POWER SUPPLY	GROUND	NC
U1	PP	VCC3: 10,11,12,13,14,15,82,93,103 111,113,123,133,143,146,151,153 163,173,183,193,203,213,269,290 291,292,293 VCCCORE: 4,5,6,7,8,9,88,98,108 118,128,138,148,158,168,178,188 198,208,263,264,265,266,267,268	22,23,24,25,26,27,28,29,30,31,32 33,34,97,104,107,114,117,124,127 134,137,144,147,152,154,157,164 167,174,177,184,187,194,197,204 207,214,223,260,264,265,266,267 268,270,271,272,273,274,275,297	2,19,20,135,141 142,161,162,242 251,279,280,296
U2	CC	VCC3: 2,4,6,7,9,10,12,13,17,19,22 34,36,38,40,41,43,57,60,61,74,76 77,81,85,86,88,93,103,119,123,124 128,143,144,148,153,163,183,186 188,192,193,202,203,213,218,222 225,227,240,244,260,261,271,273 275,278,280,284,286,288,290,292 293,296 VCCS: 264,282,283	3,5,8,11,14,15,16,18,20,21,23,24 25,35,37,39,55,56,58,59,75,78,79 80,82,83,87,91,97,100,115,114 129,133,134,139,140,147,149,150 157,173,174,177,178,182,185,187 194,197,198,204,207,209,216,217 220,223,224,230,241,242,245,250 259,262,263,272,274,277,279,281 285,287,289,291,294,295,297	
U3-U8 U11-U14	CS	VCC3: 5,9,13,29,35,58,56,74 VCCS: 17	7,11,15,19,31,33,47,53,72	

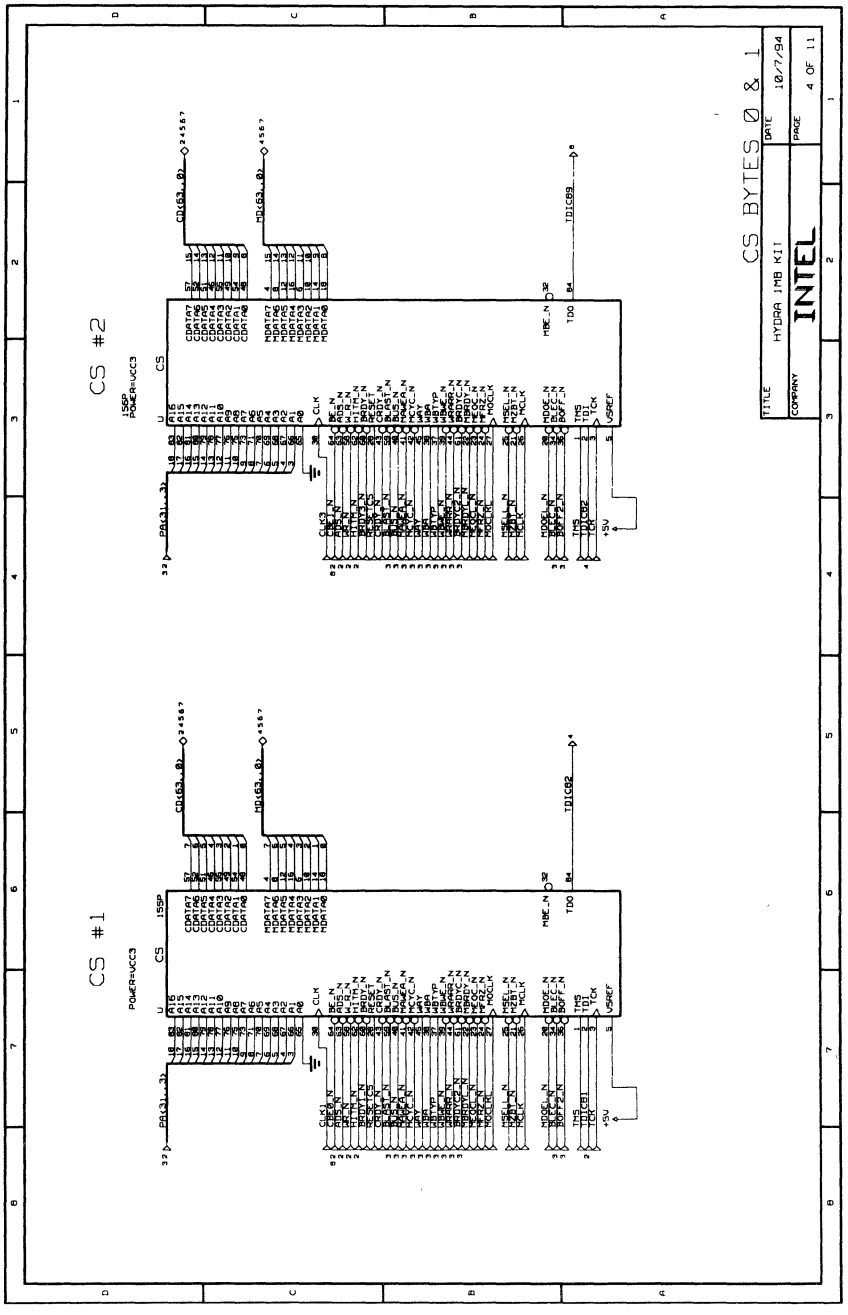
DRAWING:
TITLE=HYDRA1
ABBREV=PAGE_1

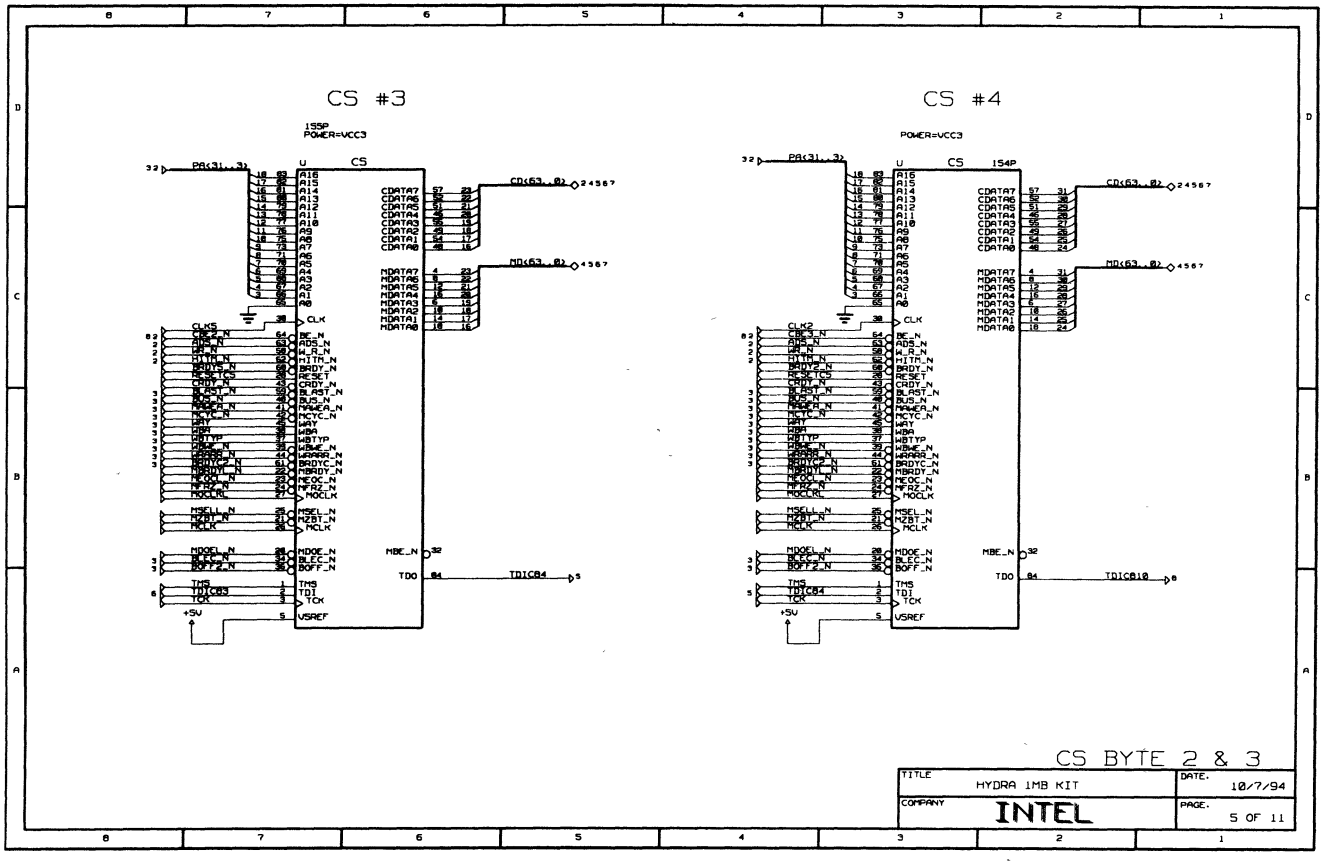
Thu Jun 1 15:28:25 1995

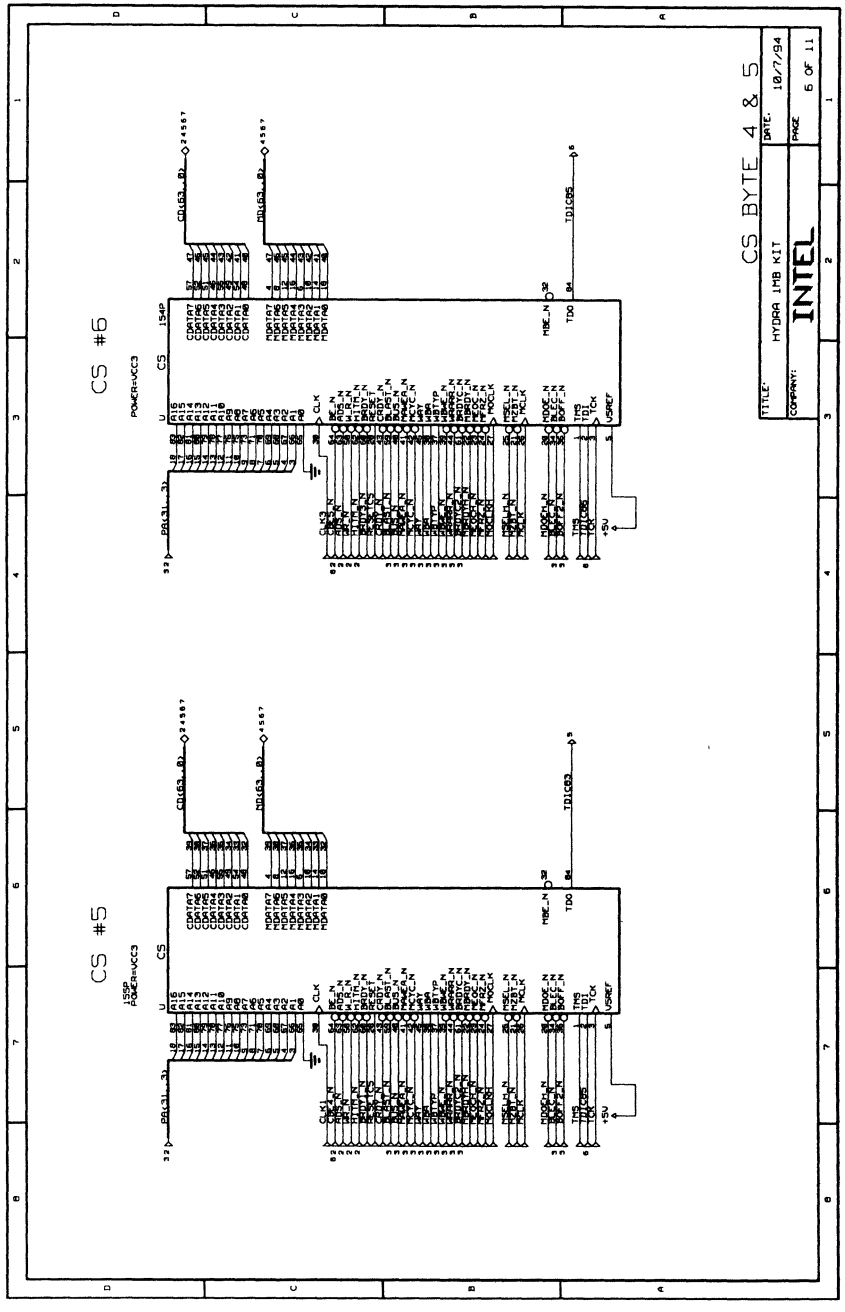
TITLE: HYDRA 1MB KIT		DATE: 10/7/94
COMPANY: INTEL		PAGE: 1 OF 11

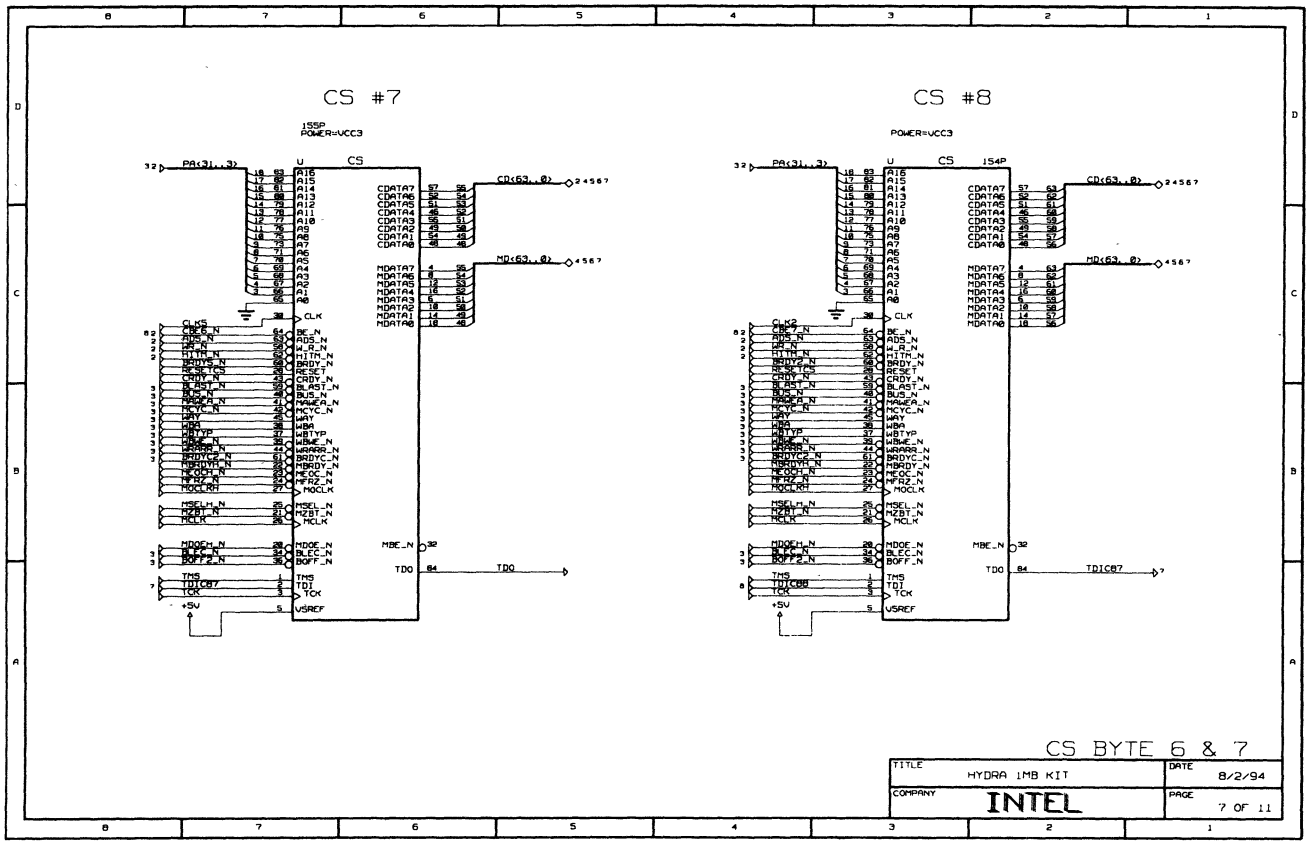

TITLE: HYDRA 1MB KIT
COMPANY: INTEL
DATE: 10/7/94
PAGE: 2 OF 11
PP

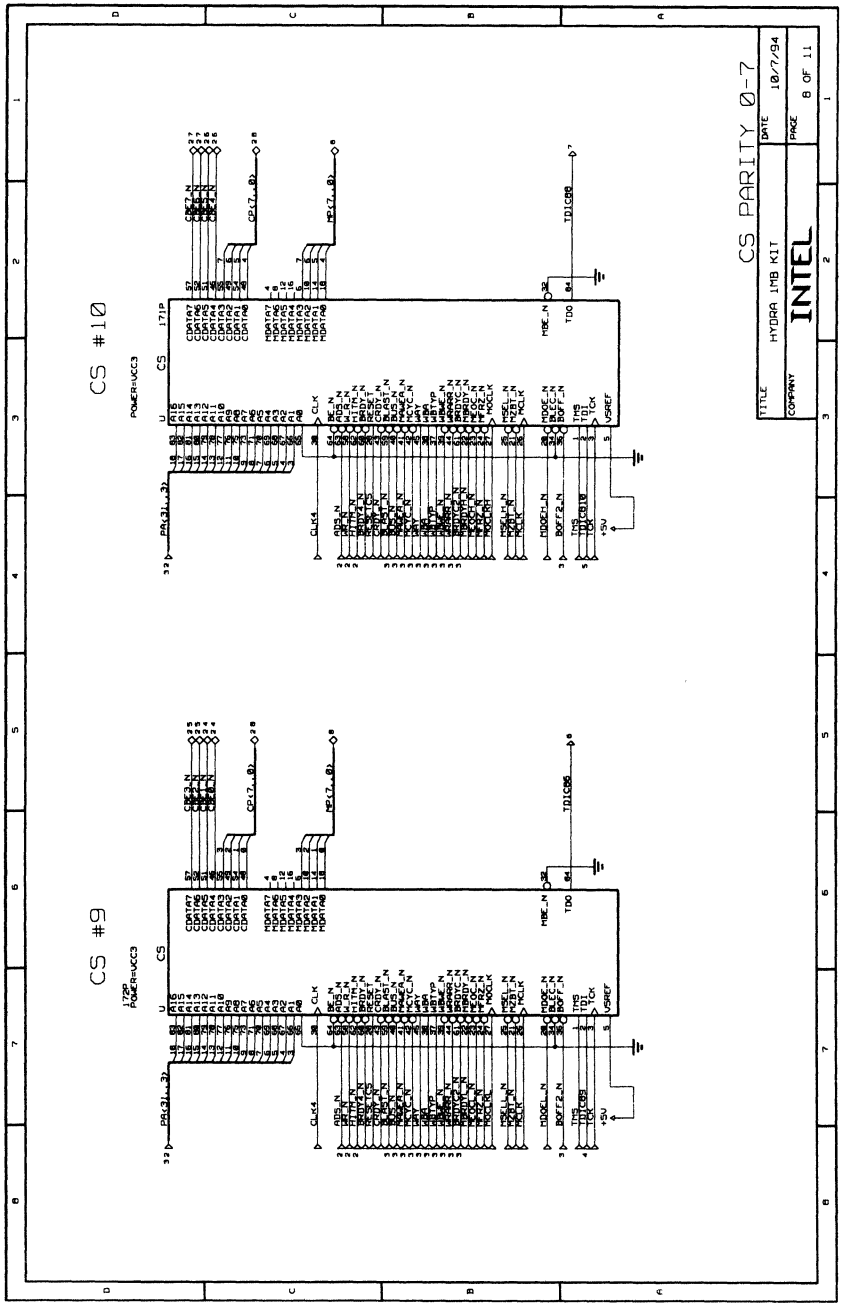

POWER=VCC3
CC
210P
PA(31..3)
MA(35..3)
CLK0
RESETCS
ADS_N
BLAST_N
D_C_N
M_IO_N
W_R_N
LOCK_N
SCYC
CACHE_N
MITM_N
BRDY0_N
CRDY_N
CLOCK_N
SYNC_N
TDI
TMS
TCK
TRST_N
TDO
CC
TITLE
HYDRA 1MB KIT
DATE:
10/7/94
COMPANY.
INTEL
PAGE.
3 OF 11

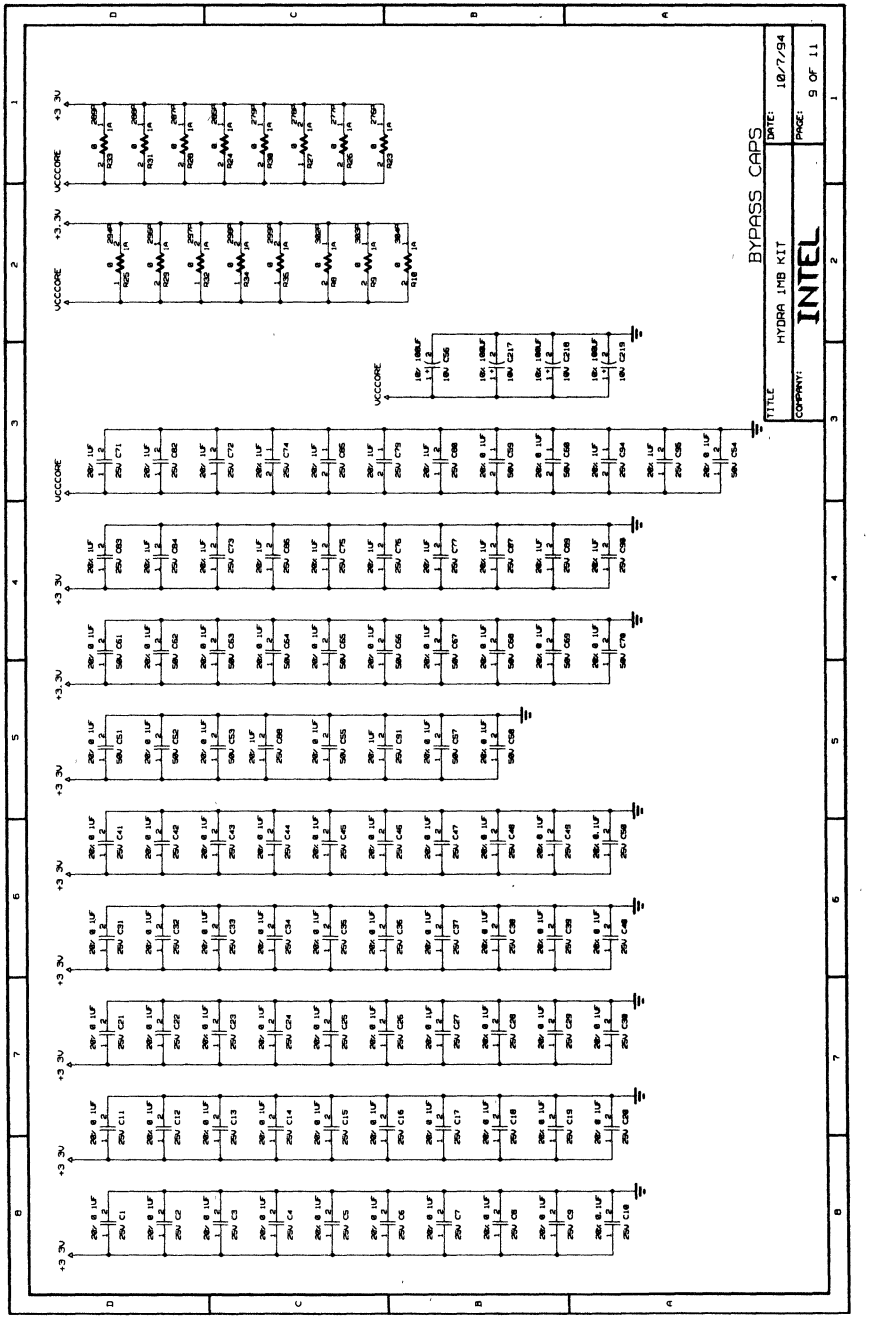

CS BYTES 0 & 1
TITLE
HYDRA IMB KII
DATE
10/7/94
COMPANY
INTEL
PAGE
4 OF 11
CS #2
CS #1
ISSP
CS
POWER=VCC3
+5V
VSREF
TDO
TDI
TMS
TCK
TDICS2
TDICS9

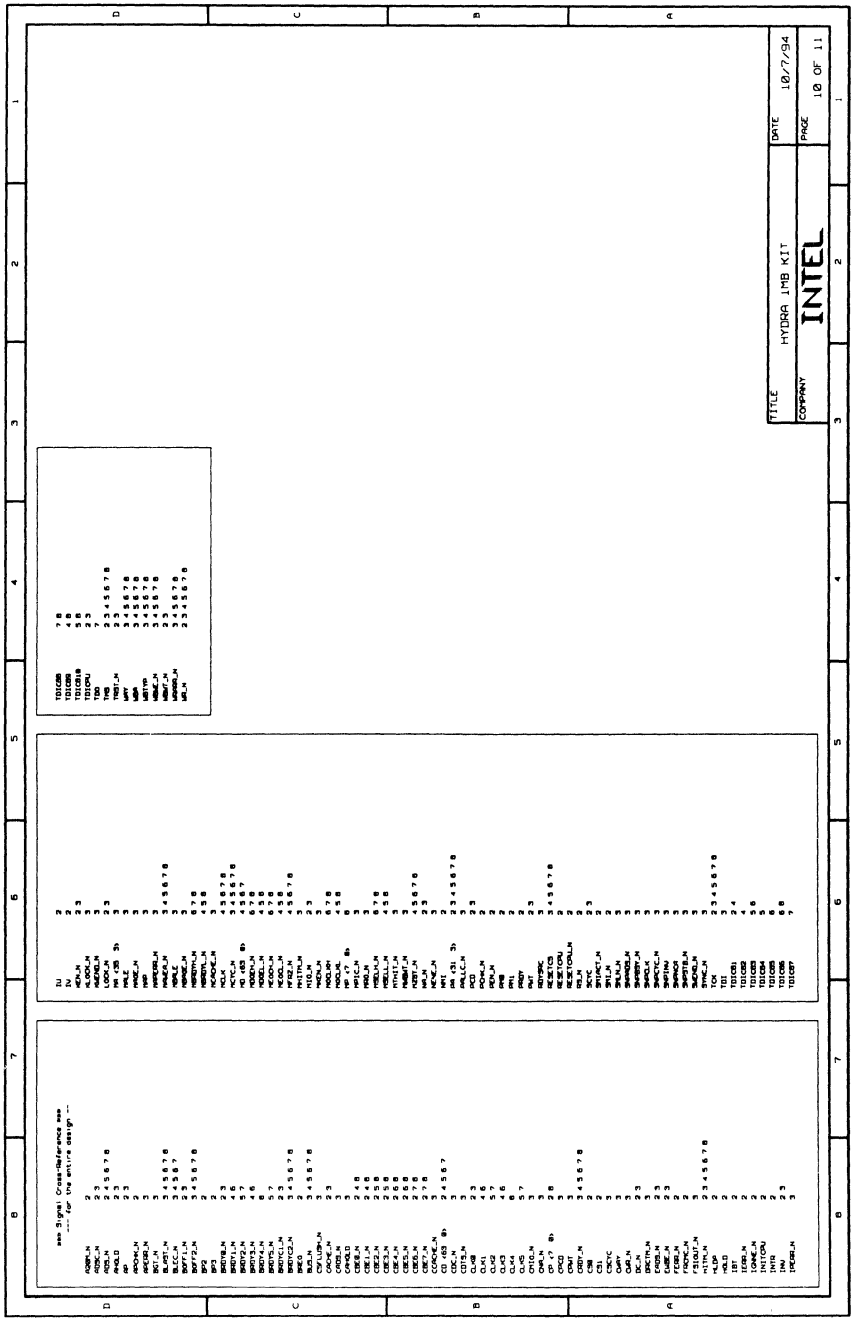

CS #3
155P
POWER=VCC3
U
CS
PA<31..3>
CD<63..0>
MD<63..0>
CLK
MBE_N
TDO
TDIC84
TMS
TDI
TCK
VSREF
+5V
CS #4
154P
POWER=VCC3
CLK2
TDIC810
CS BYTE 2 & 3
TITLE
HYDRA 1MB KIT
DATE.
10/7/94
COMPANY
INTEL
PAGE.
5 OF 11

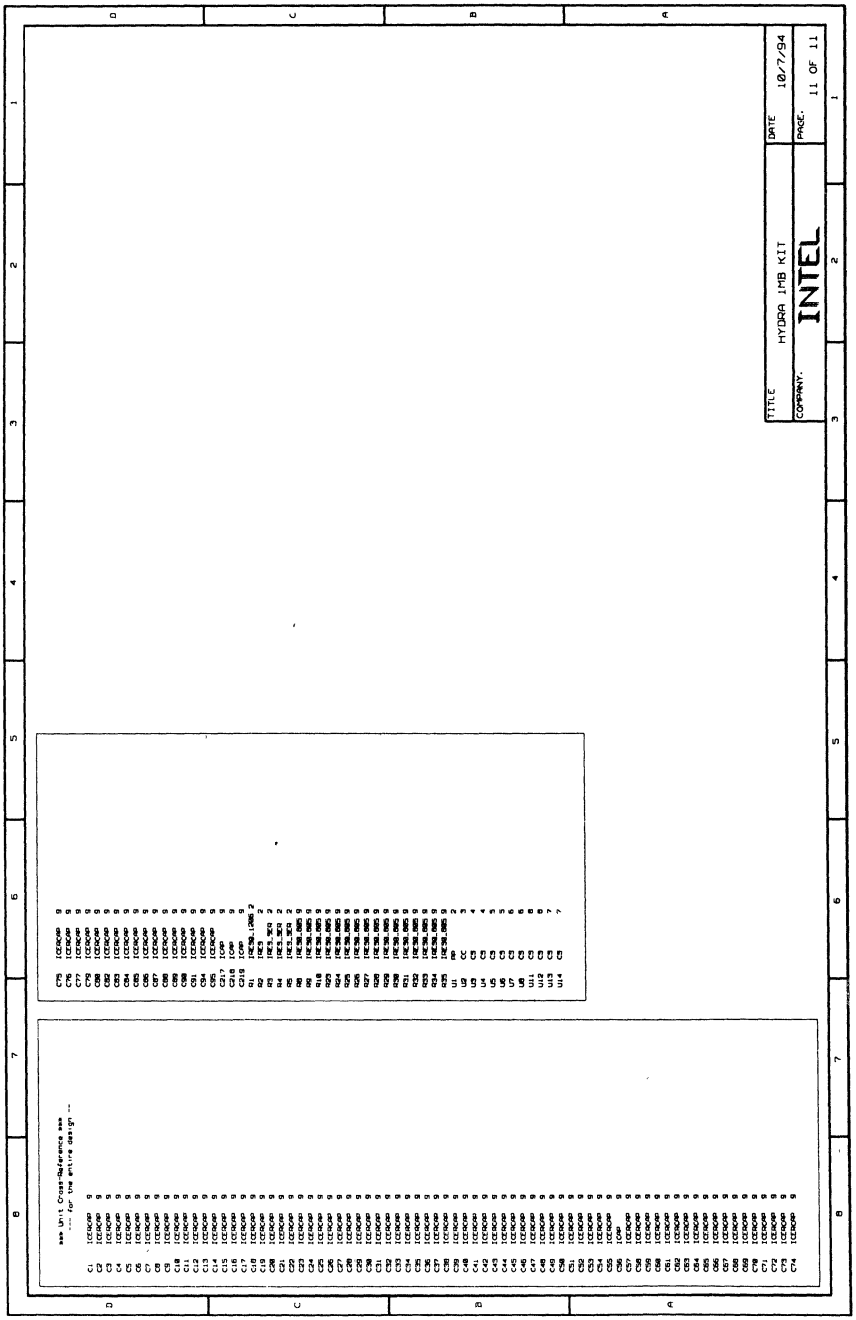

CS BYTE 4 & 5
TITLE: HYDRA 1MB KIT
DATE: 10/7/94
COMPANY: INTEL
PAGE 6 OF 11
CS #6
CS #5
POWER=VCC3
CS
154P
155P
TDI
TDO
TMS
TCK
+5V
VSREF


CS #7
155P
POWER=VCC3
CS #8
154P
POWER=VCC3
CS BYTE 6 & 7
TITLE
HYDRA 1MB KIT
DATE
8/2/94
COMPANY
INTEL
PAGE
7 OF 11


CS #10
CS #9
CS PARITY 0-7
HYDRA 1MB KIT
INTEL
10/7/94
8 OF 11


BYPASS CAPS
TITLE: HYDRA 1MB KIT
COMPANY: INTEL
DATE: 10/7/94
PAGE: 9 OF 11




TITLE HYDRA IMB KIT
DATE 10/7/94
COMPANY: INTEL
PAGE: 11 OF 11

PP/CC/CS 2MB INTERFACE

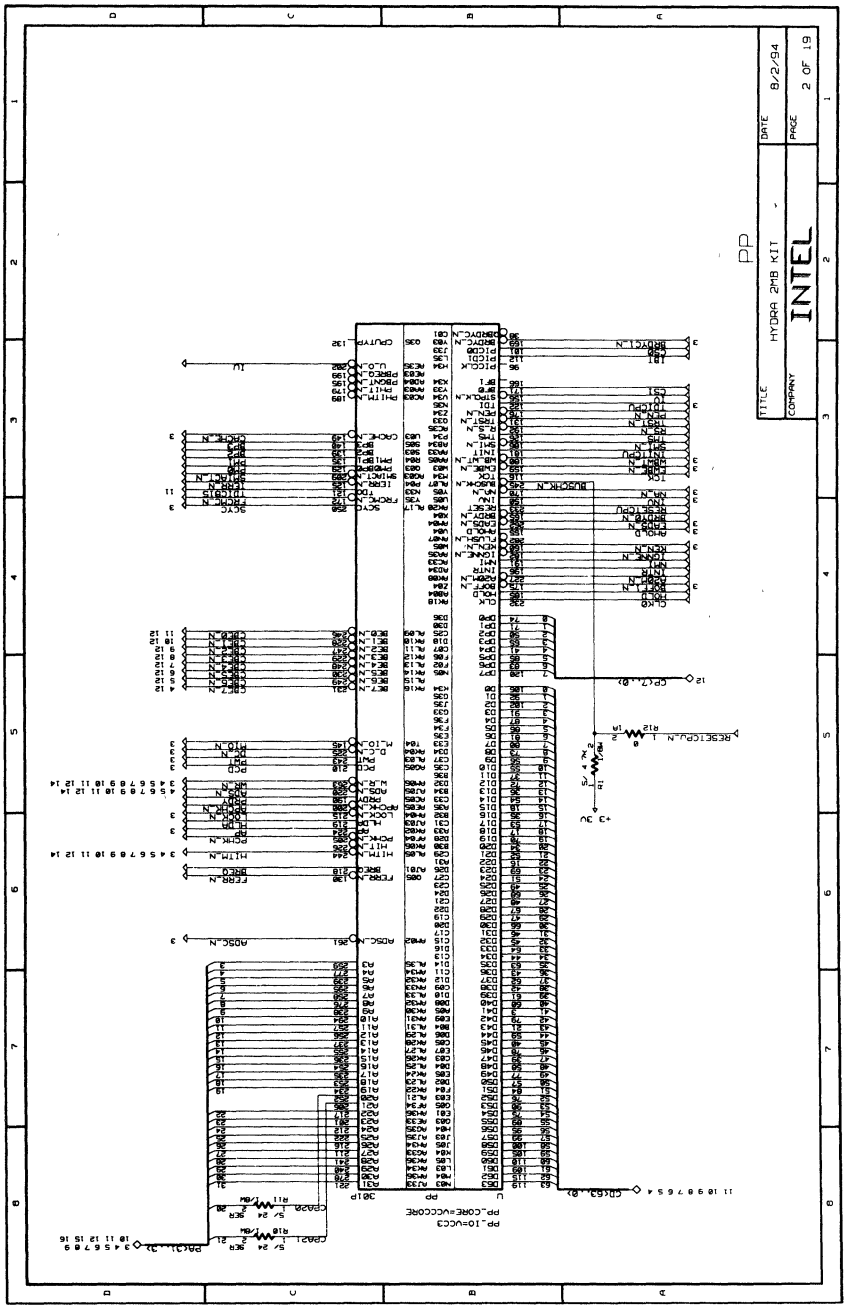
LIST OF POWER SUPPLY PINS AND NO CONNECT PINS:

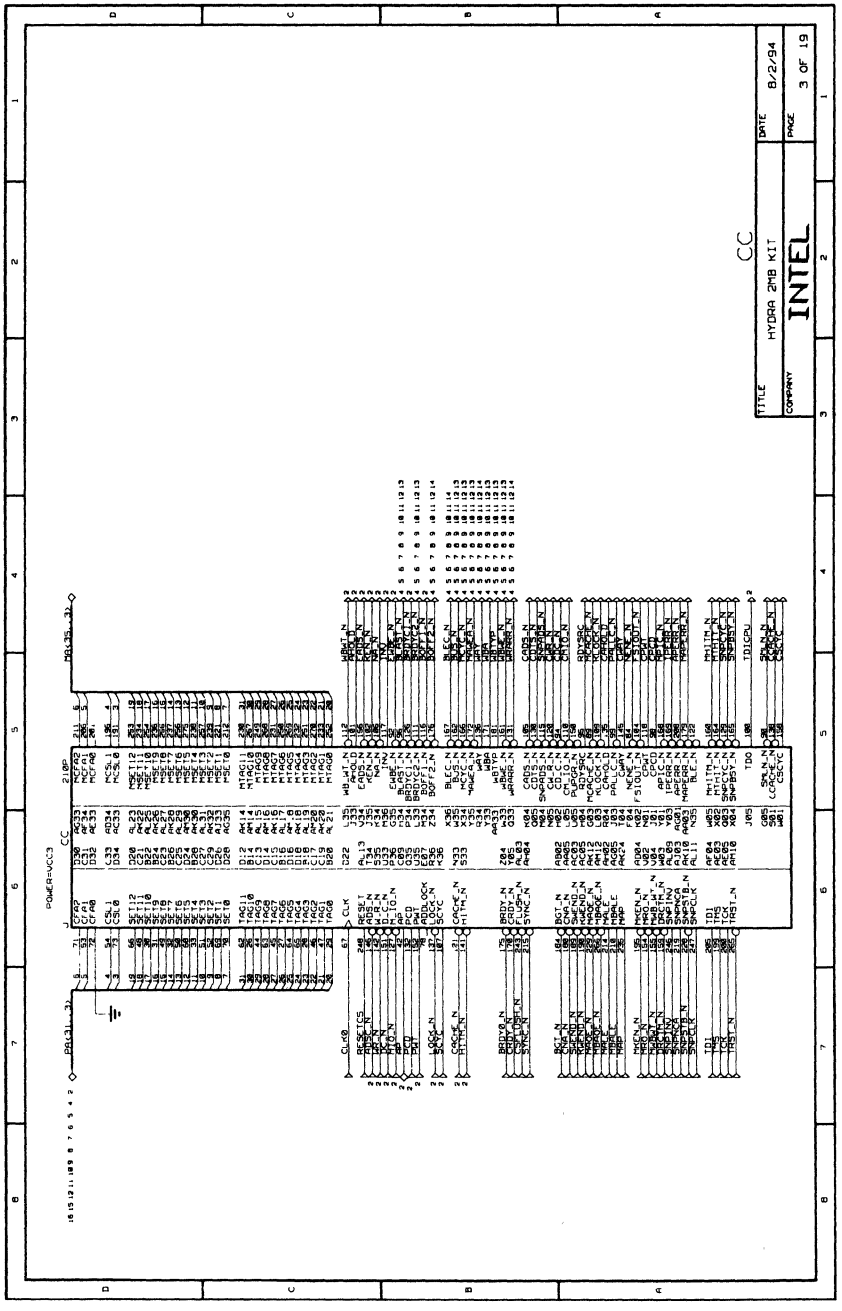
REF DES	NAME	POWER SUPPLY	GROUND	NC
U12	PP	VCC3 10,11,12,13,14,15,82,93 103,111,113,123,133,143,146,151 153,163,173,183,193,203,213,289 290,291,292,293 VCCCORE 4,5,6,7,8,9,88,90,108,118 128,136,146,158,168,178,188,198 200,203,204,205,206,207,208	22,23,24,25,26,27,28,29,30,31,32 33,94,97,104,107,114,117,124,127 134,137,144,147,152,154,157,164 167,174,177,184,187,194,197,204 207,214,223,260,264,265,266,267 268,270,271,272,273,274,275,297	2,19,20,136,141 142,161,162,242 251,279,280,296
U8	CC	VCC3 2,4,6,7,9,10,12,,3,17,19,22 34,36,38,40,41,43,57,60,61,74,76 77,81,85,86,88,93,103,119,123,,24 128,143,144,148,153,163,183,186 188,192,193,202,203,213,218,222 225,227,240,244,260,261,271,273 276,278,280,284,286,288,290,292 293,296 VCC5 264,282,283	3,5,8,11,14,15,16,18,20,21,23,24 25,35,37,39,55,56,58,59,75,78,79 80,82,83,87,91,97,100,113,114 120,133,134,139,140,147,149,150 157,173,174,177,178,182,185,187 194,197,198,204,207,209,216,217 218,223,224,225,241,242,245,250 259,262,263,272,274,277,279,281 285,287,289,291,294,295,297	
U1-U7,U9-U11 U13-20	C5	VCC3 5,9,13,29,35,50,56,74 VCC5 17	7,11,15,19,31,33,47,53,72	
U21-U28,U30	7451053	VCC3 1,20	10,11	

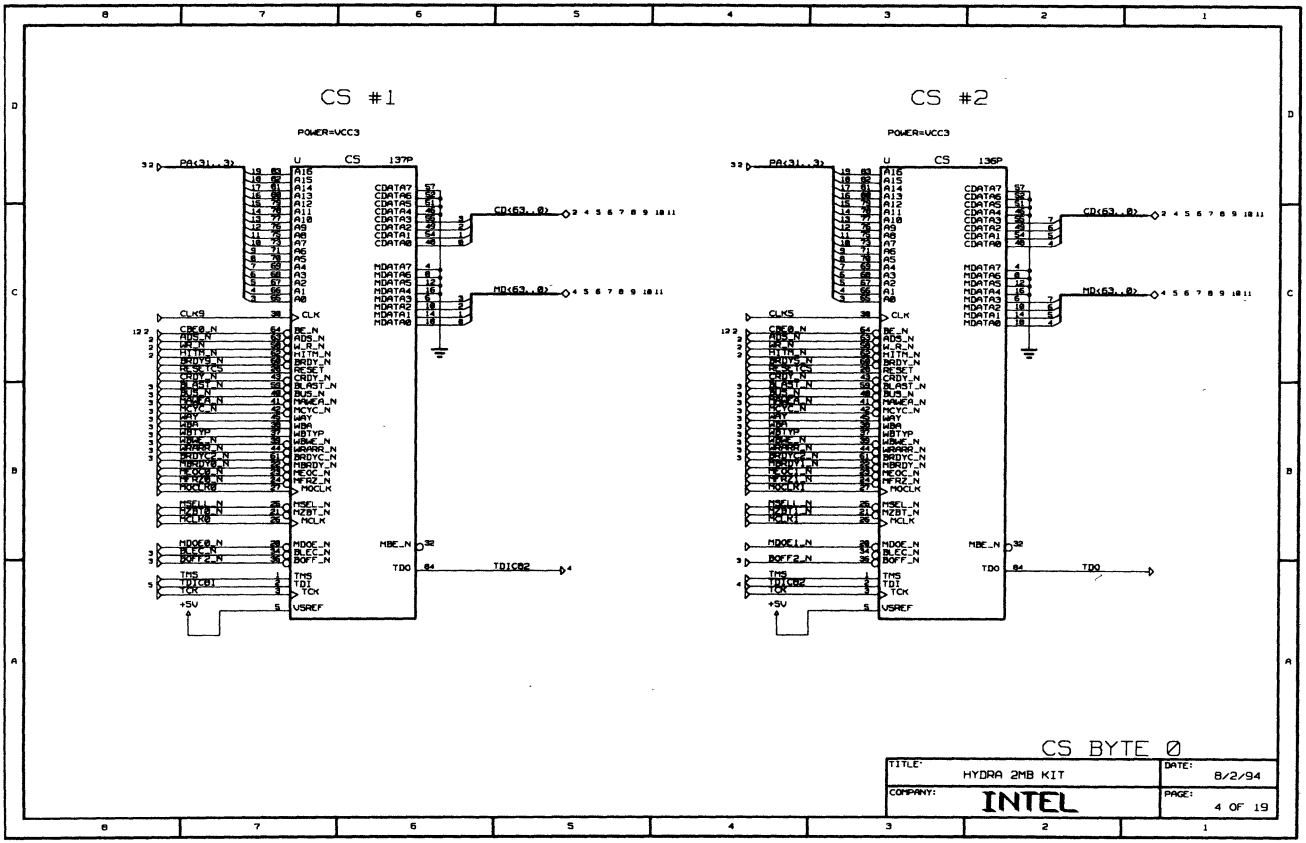
DRAWING:
TITLE=HYDRA2
ABBREV=PAGE_1

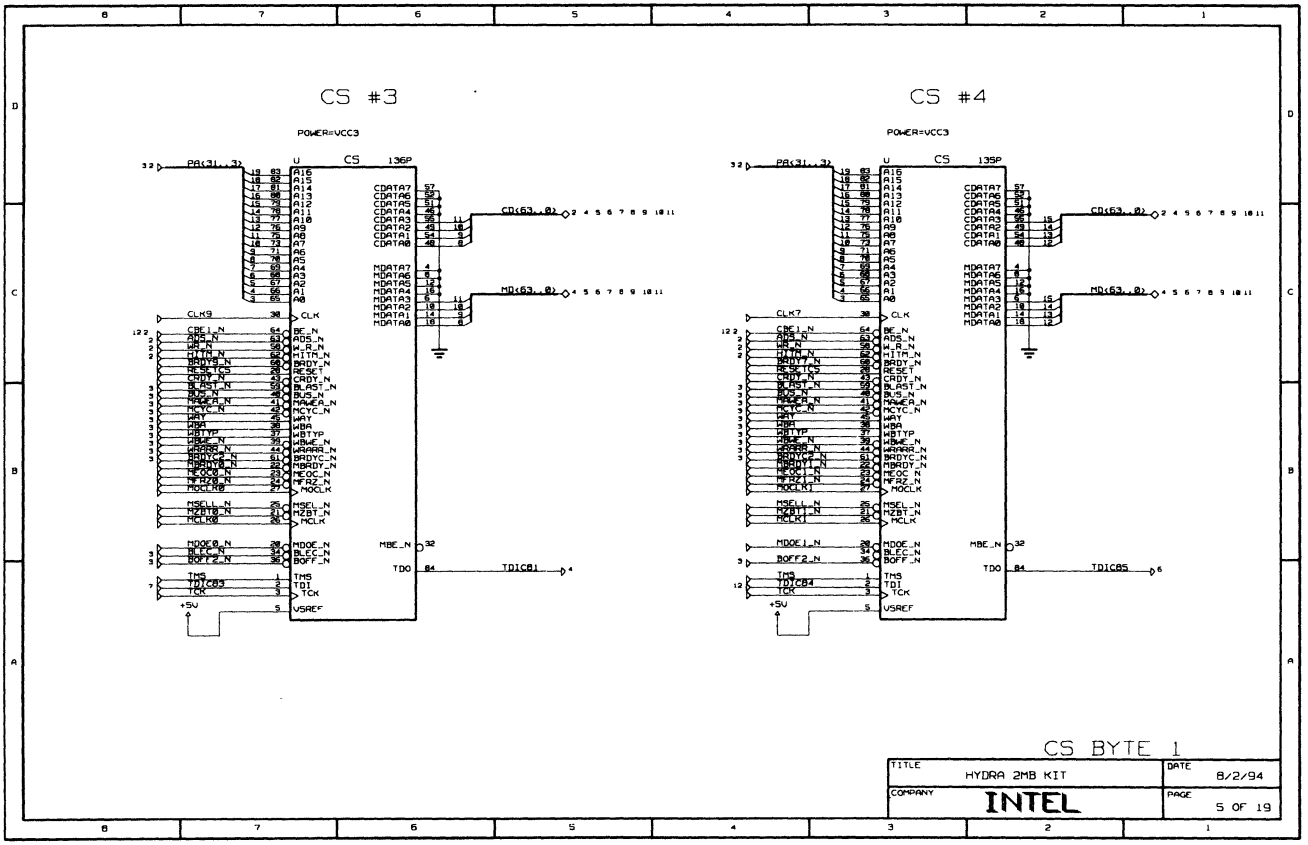
Thu Jun 1 16 07 23 1995

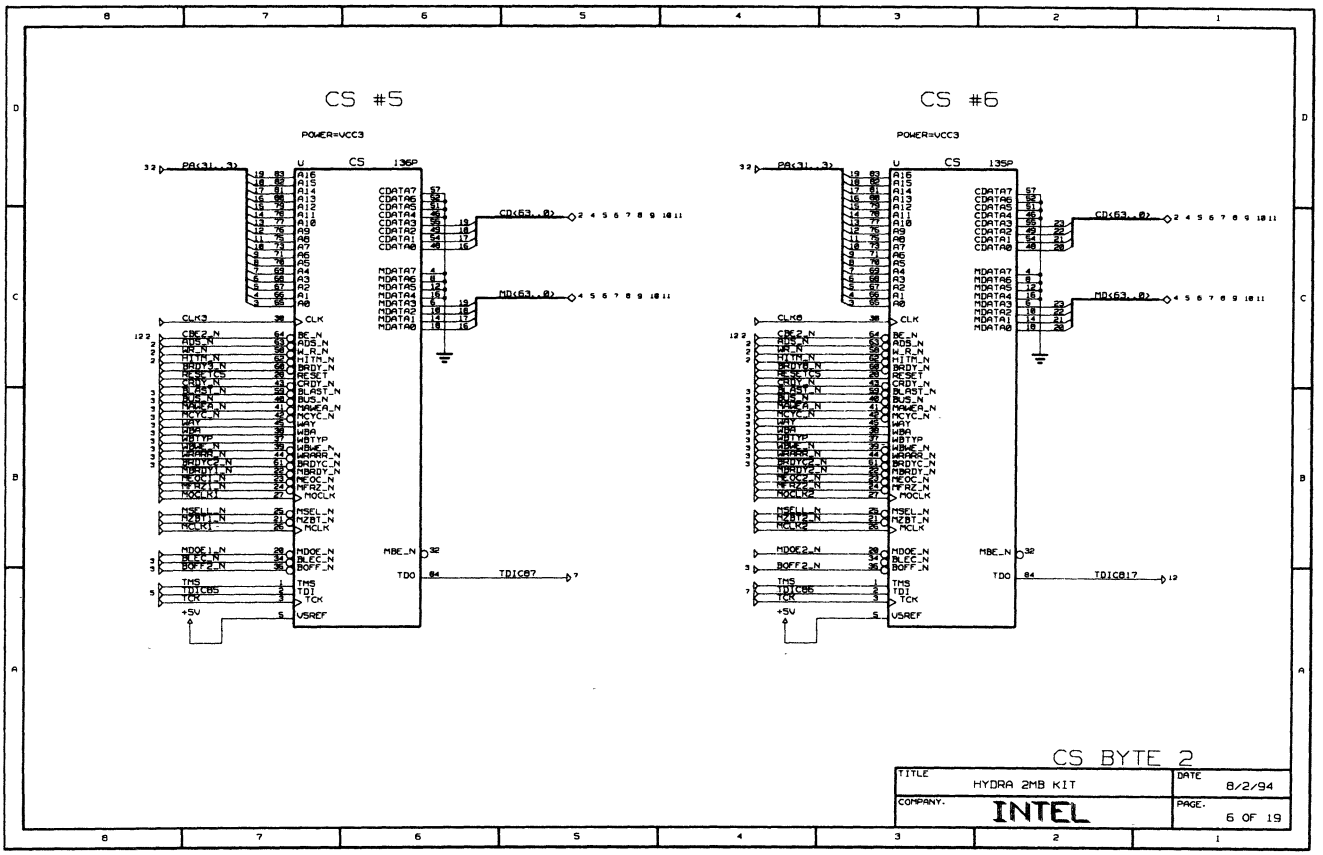
TITLE: HYDRA 2MB KIT		DATE: 8/2/94
COMPANY: INTEL		PAGE: 1 OF 19

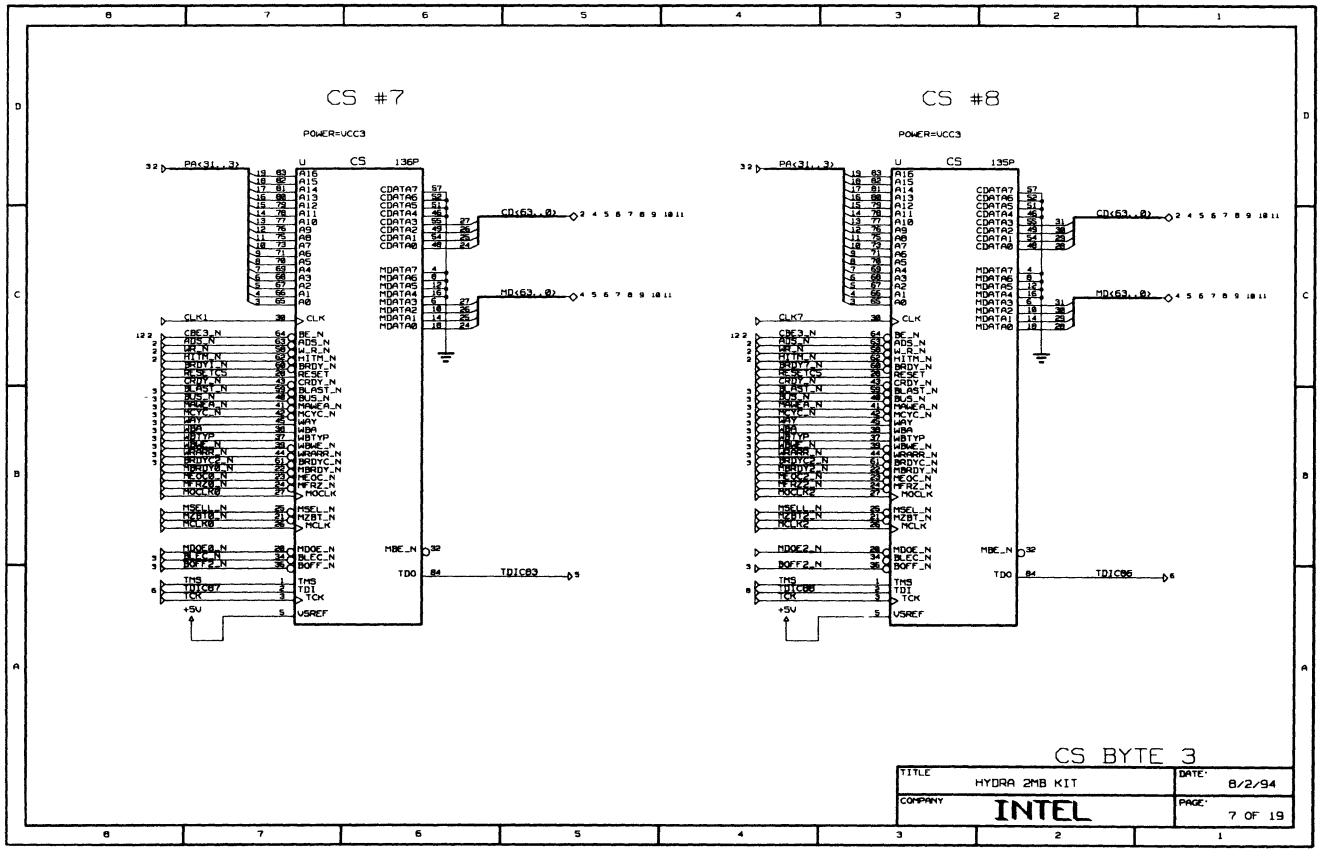

TITLE
HYDRA 2MB KIT
COMPANY
INTEL
DATE
8/2/94
PAGE
2 OF 19

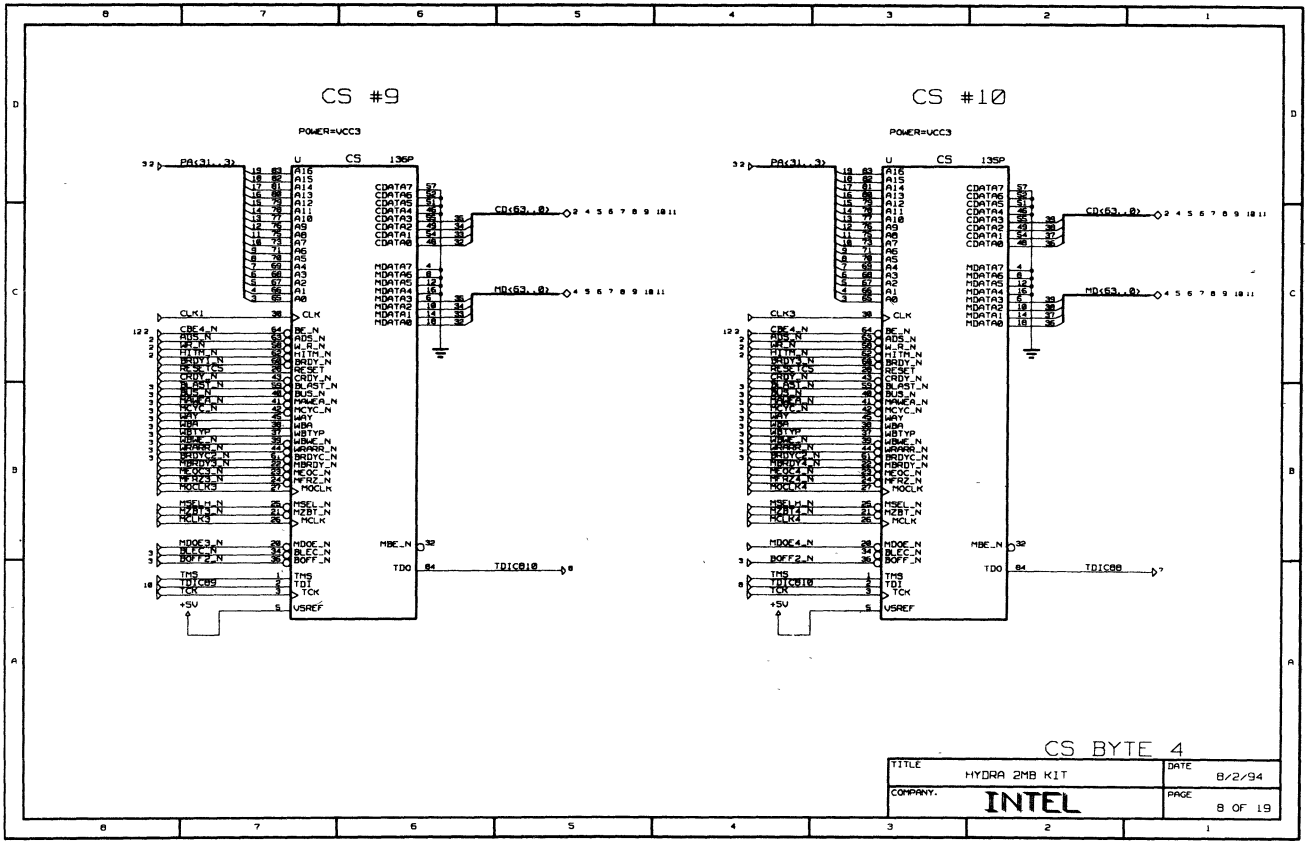

CC
HYDRA 2MB KIT
INTEL
DATE 8/2/94
PAGE 3 OF 19

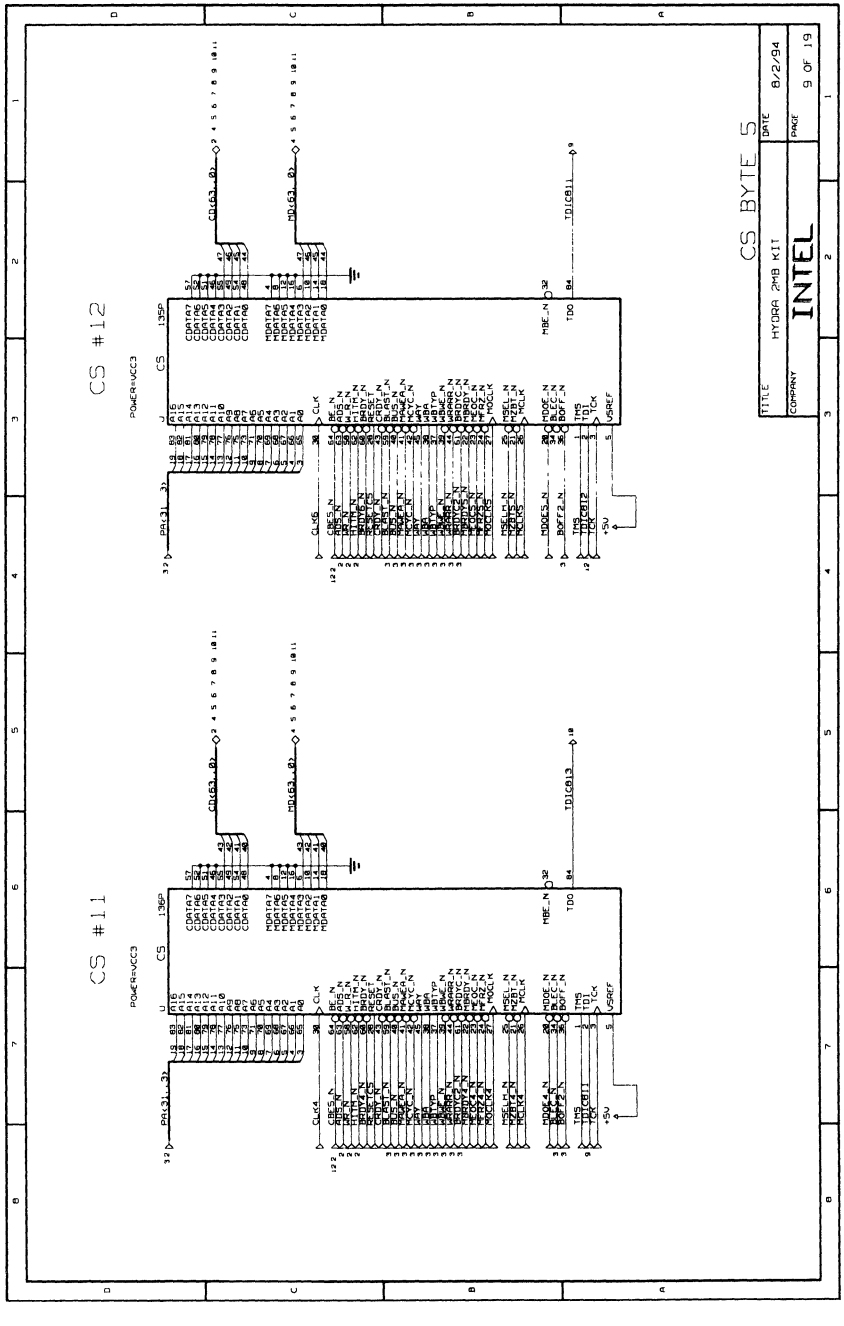

CS #1
POWER=VCC3
CS #2
POWER=VCC3
CS BYTE 0
TITLE: HYDRA 2MB KIT
DATE: 8/2/94
COMPANY: INTEL
PAGE: 4 OF 19

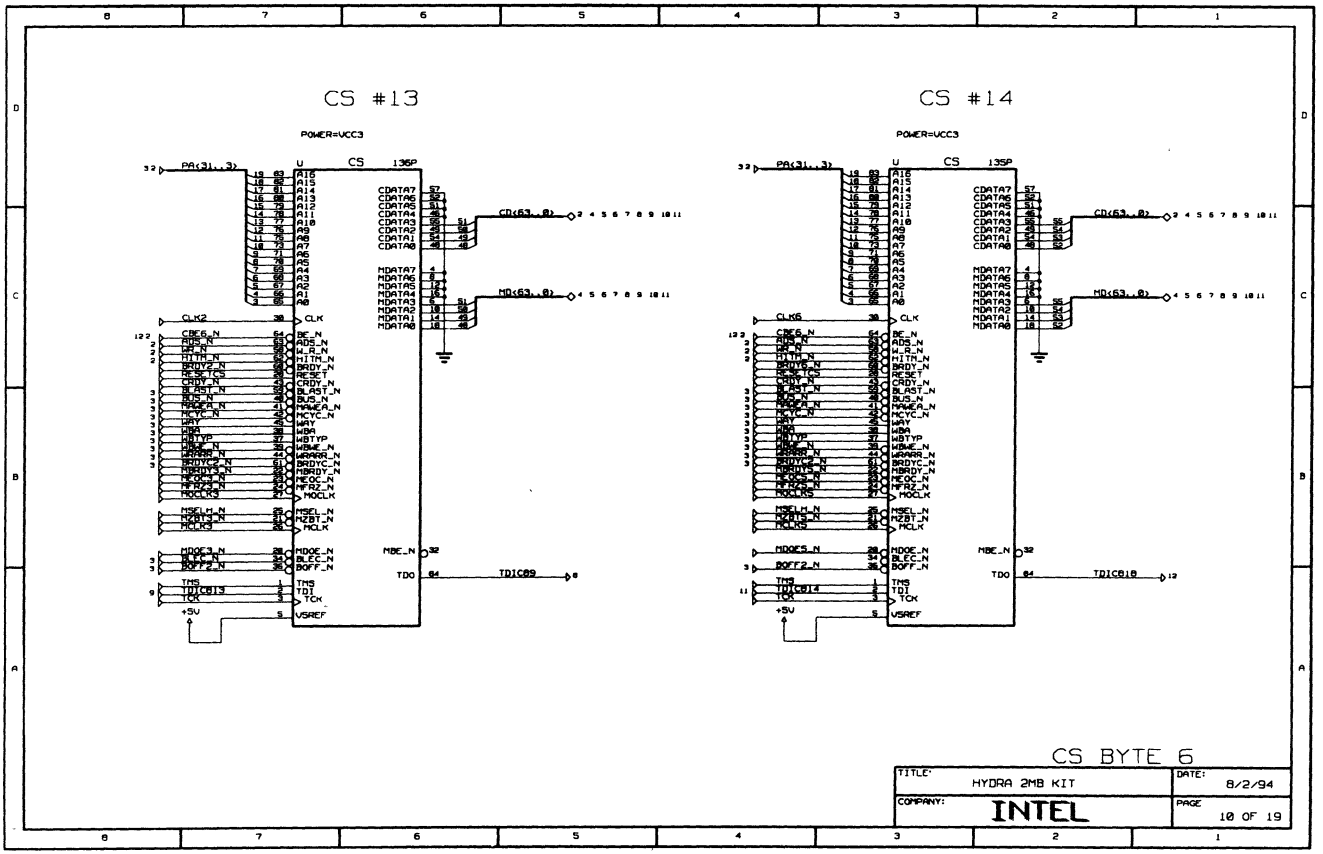

CS #3
CS #4
POWER=VCC3
CS BYTE 1
TITLE
HYDRA 2MB KIT
DATE
8/2/94
COMPANY
INTEL
PAGE
5 OF 19

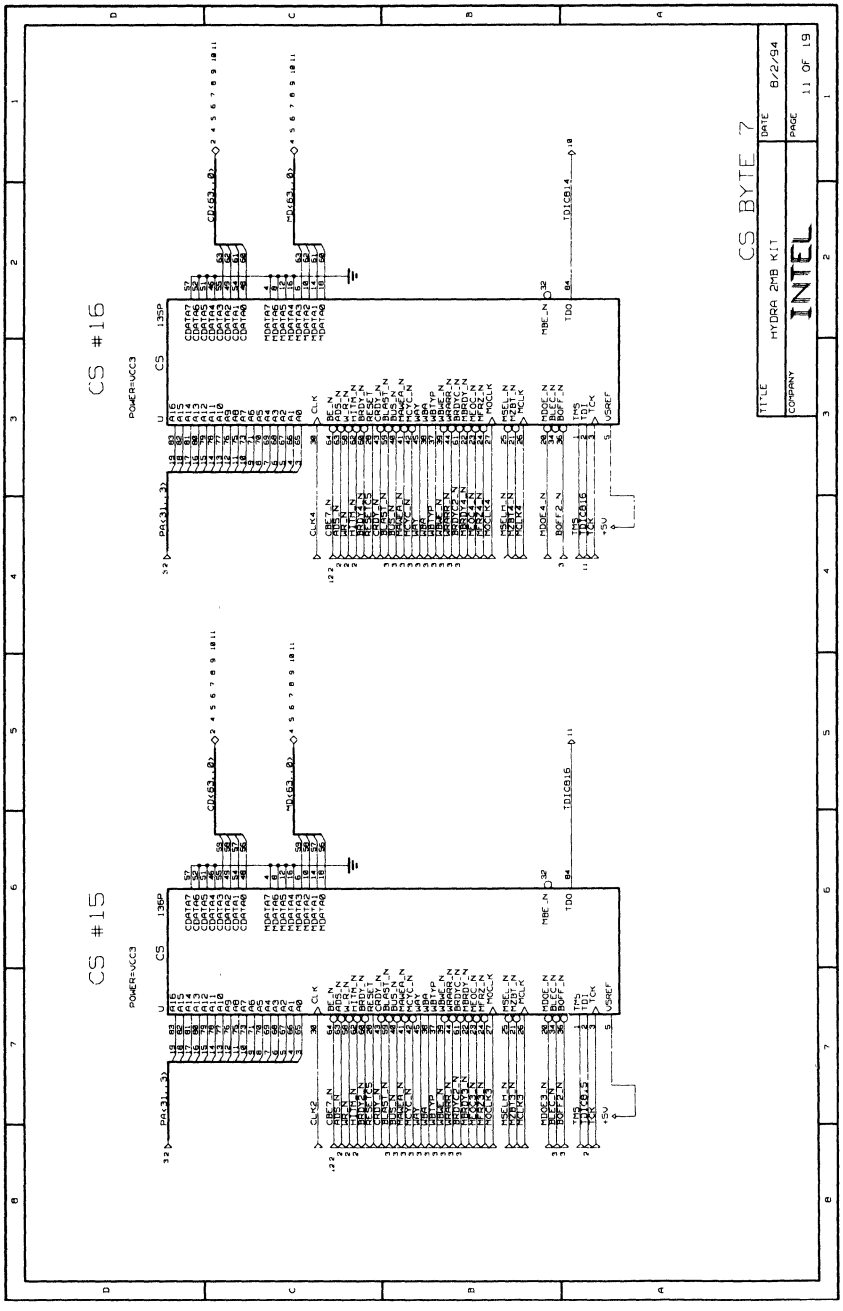

CS #5
POWER=VCC3
CS #6
POWER=VCC3
CS BYTE 2
TITLE
HYDRA 2MB KIT
DATE
8/2/94
COMPANY.
INTEL
PAGE.
6 OF 19

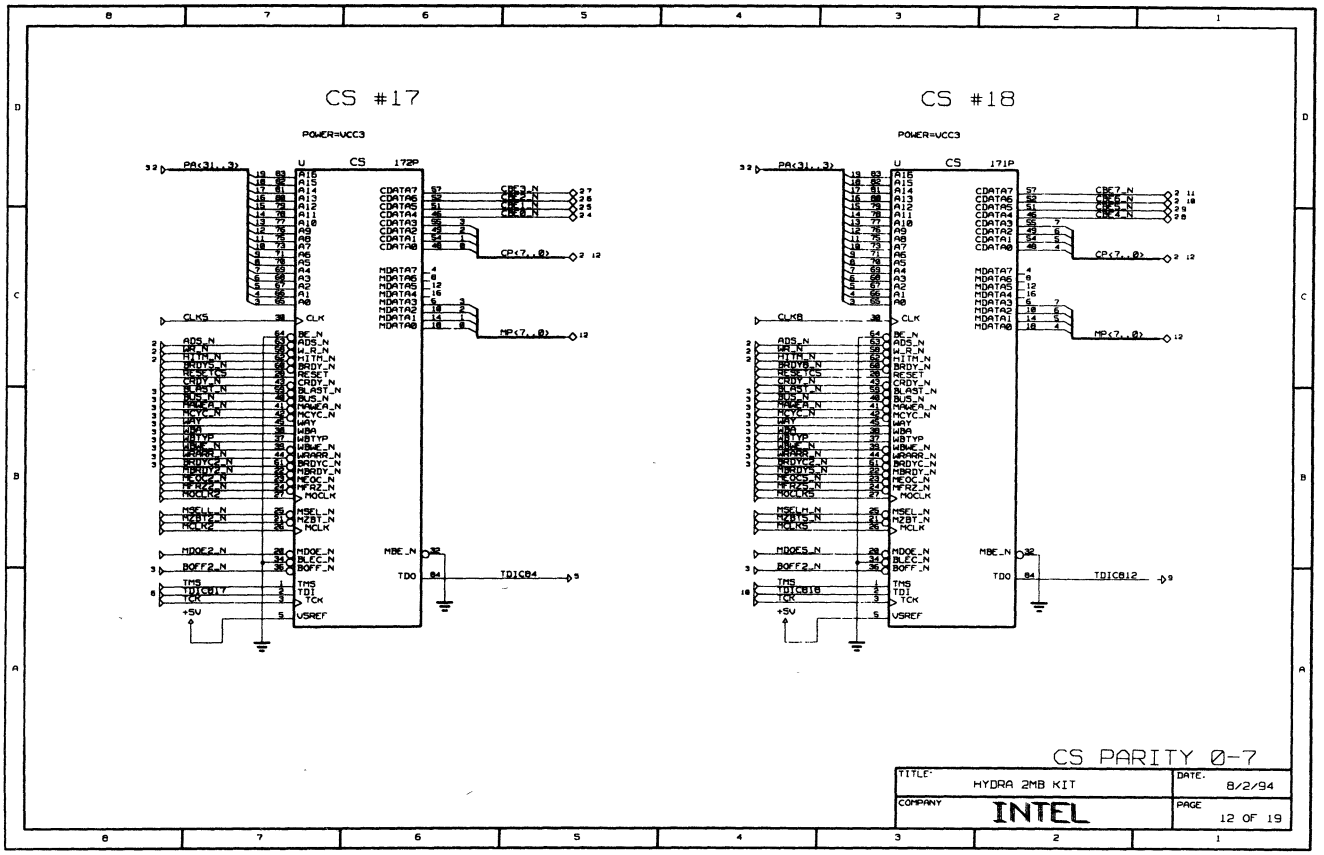

CS #7
POWER=VCC3
CS #8
POWER=VCC3
CS BYTE 3
TITLE
HYDRA 2MB KIT
DATE
8/2/94
COMPANY
INTEL
PAGE
7 OF 19

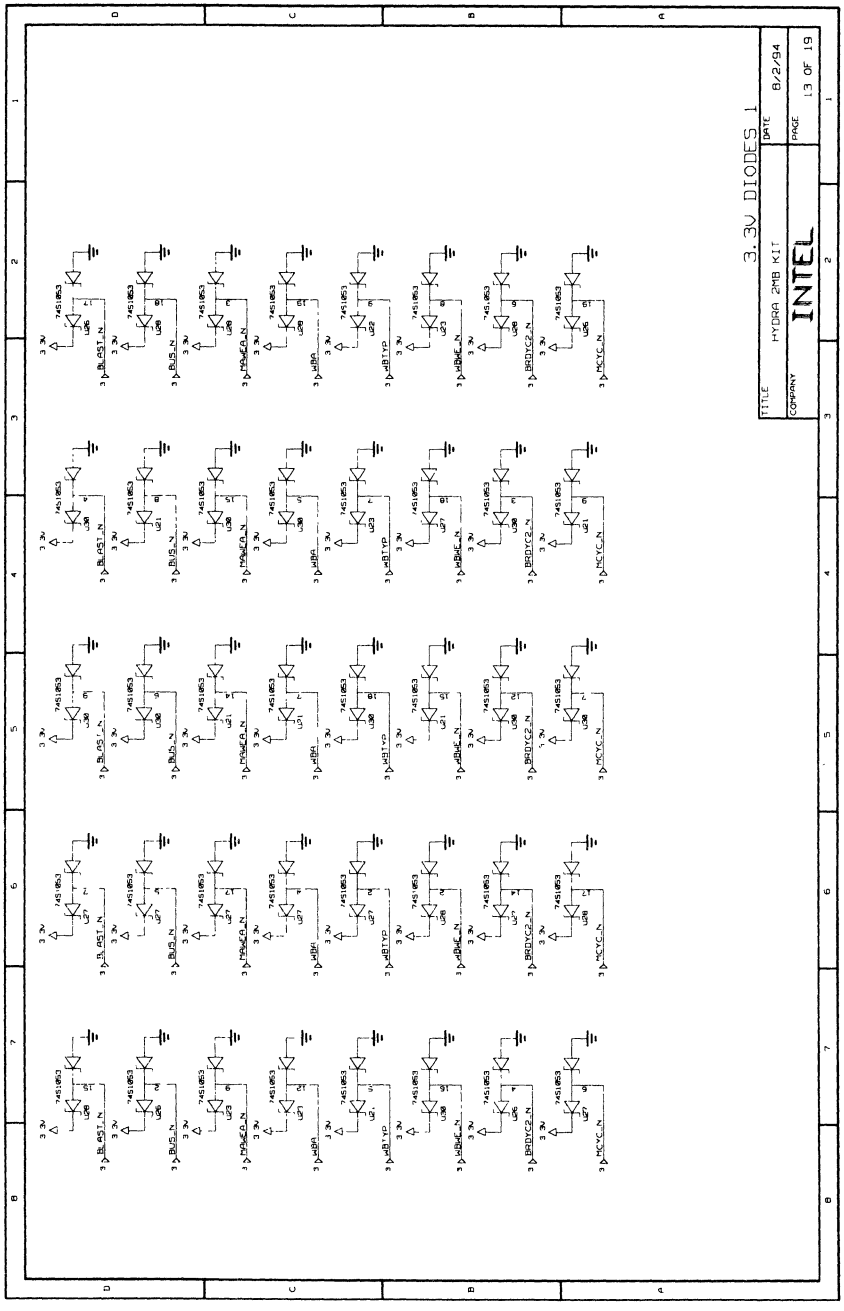

CS #9
POWER=VCC3
CS #10
POWER=VCC3
CS BYTE 4
TITLE
HYDRA 2MB KIT
DATE
8/2/94
COMPANY.
INTEL
PAGE
8 OF 19

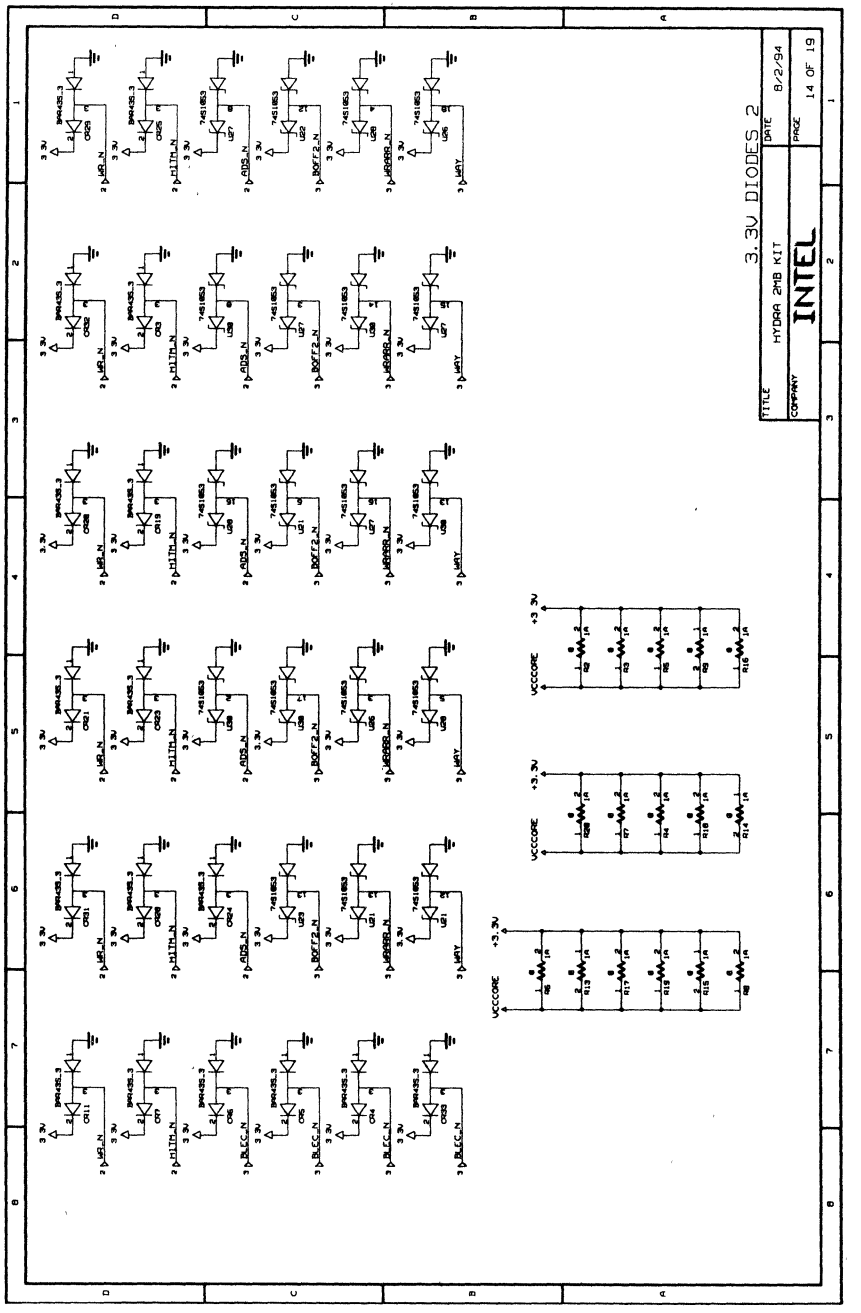

CS #12
CS #11
CS BYTE 5
TITLE
HYDRA 2MB KIT
COMPANY
INTEL
DATE
8/2/94
PAGE
9 OF 19

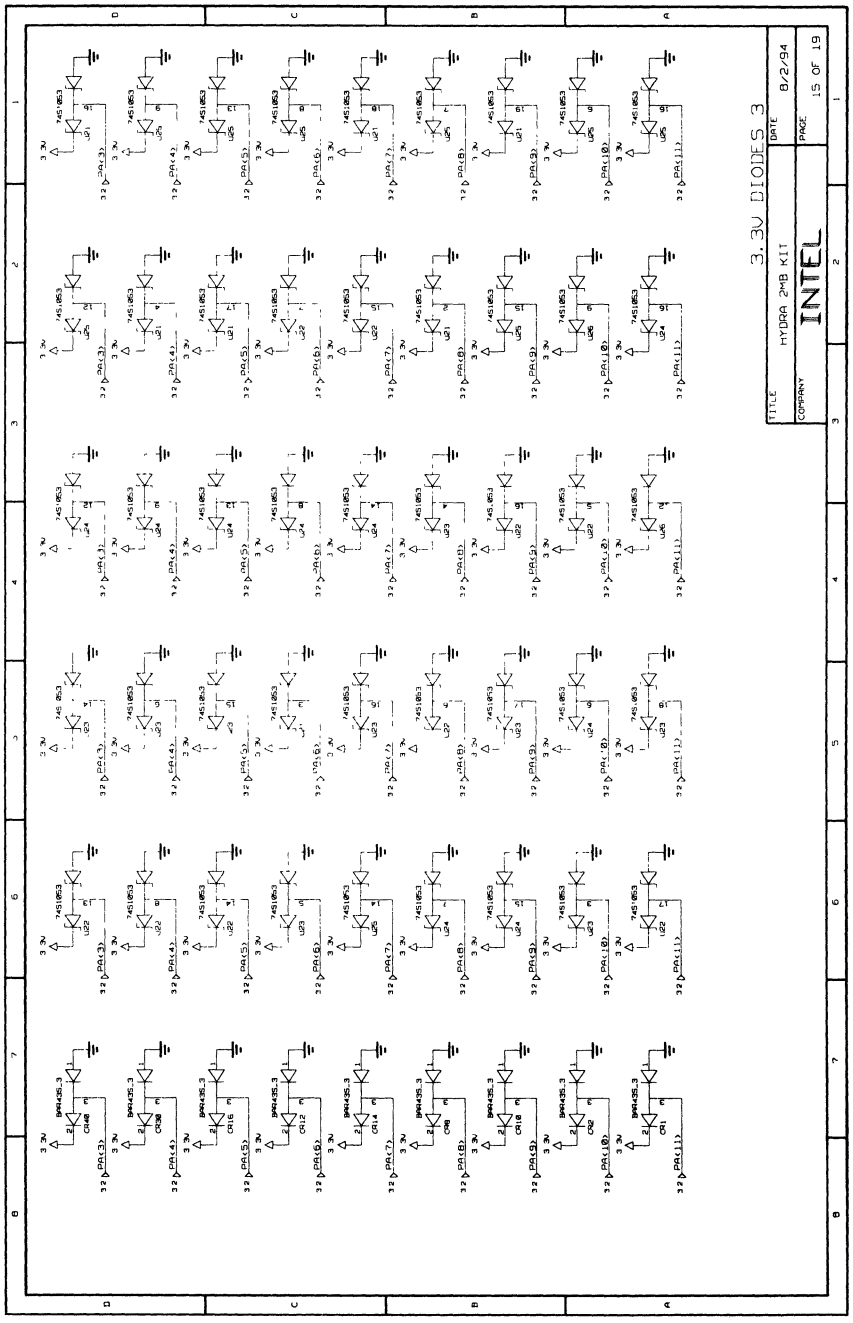

CS #13
CS #14
POWER=VCC3
CS BYTE 6
TITLE: HYDRA 2MB KIT
DATE: 8/2/94
COMPANY: INTEL
PAGE 10 OF 19

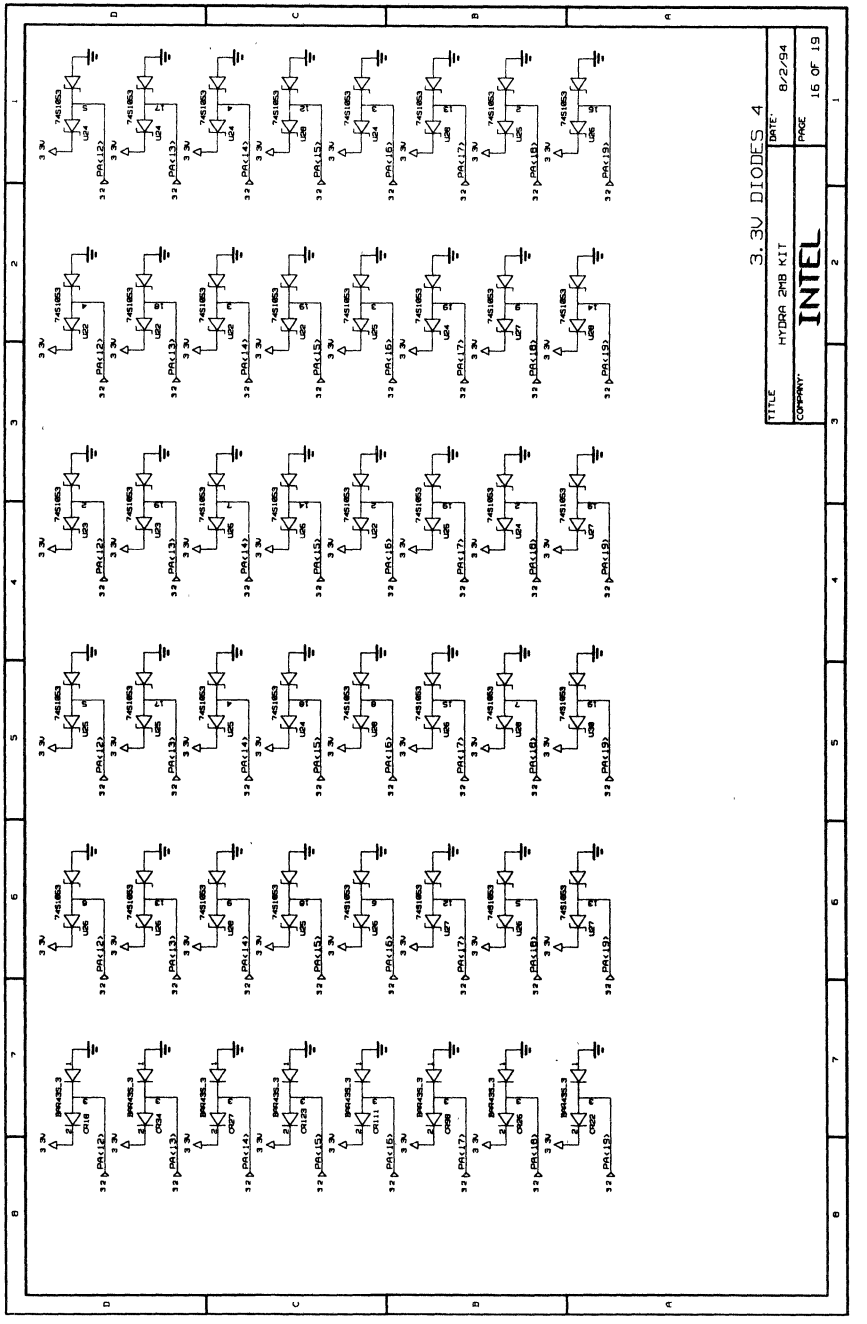

CS BYTE 7
TITLE
HYDRA 2MB KIT
COMPANY
INTEL
DATE
8/2/94
PAGE
11 OF 19
CS #16
CS #15
POWER=VCC3
CS
135P
P0(31..3)
CD(63..0)
MD(63..0)
IDIC814
IDIC816
MBE_N
TDO
CLK4
CLK2
CLK
+5V
VSREF

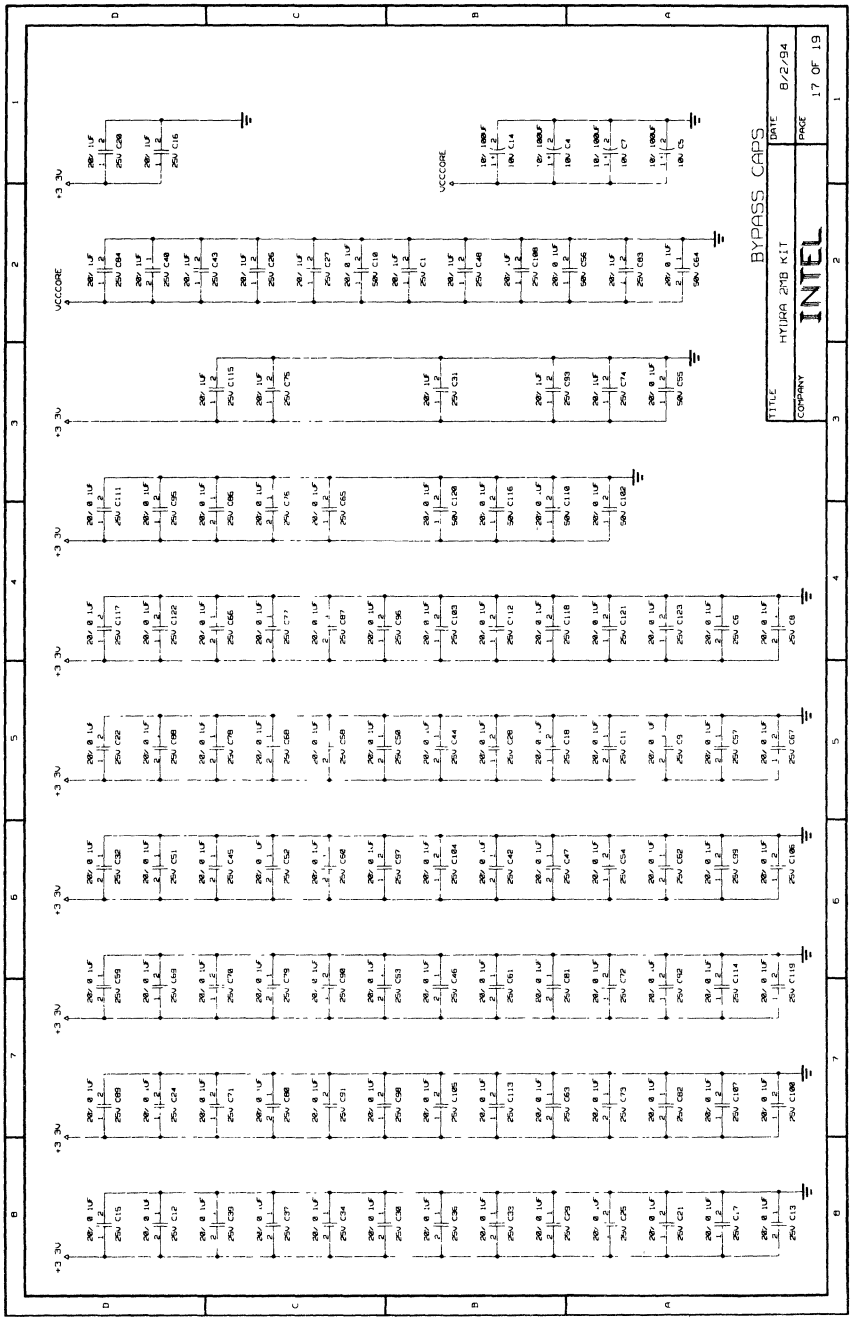

CS #17
POWER=VCC3
CS #18
POWER=VCC3
CS PARITY 0-7
TITLE: HYDRA 2MB KIT
DATE: 8/2/94
COMPANY: INTEL
PAGE: 12 OF 19

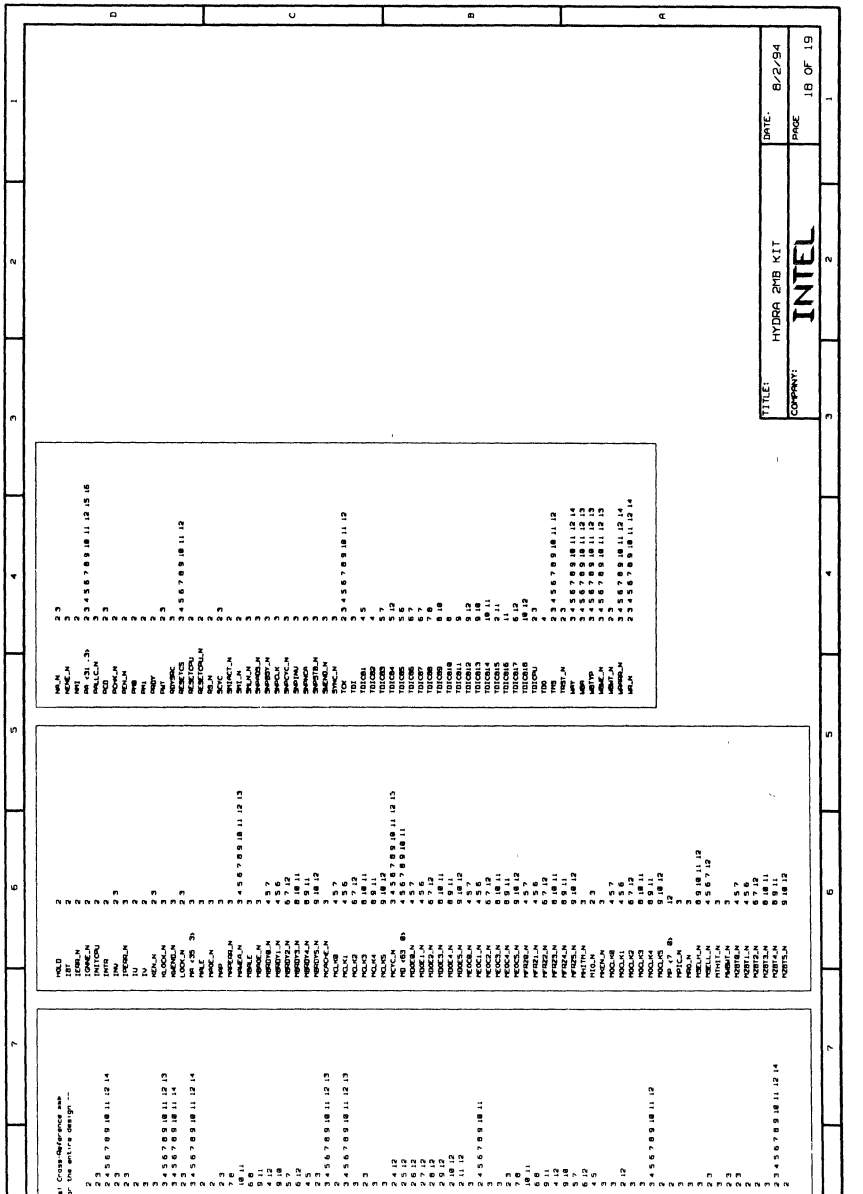

3.3V DIODES 1
TITLE
HYDRA 2MB KIT
COMPANY
INTEL
DATE
8/2/94
PAGE
13 OF 19

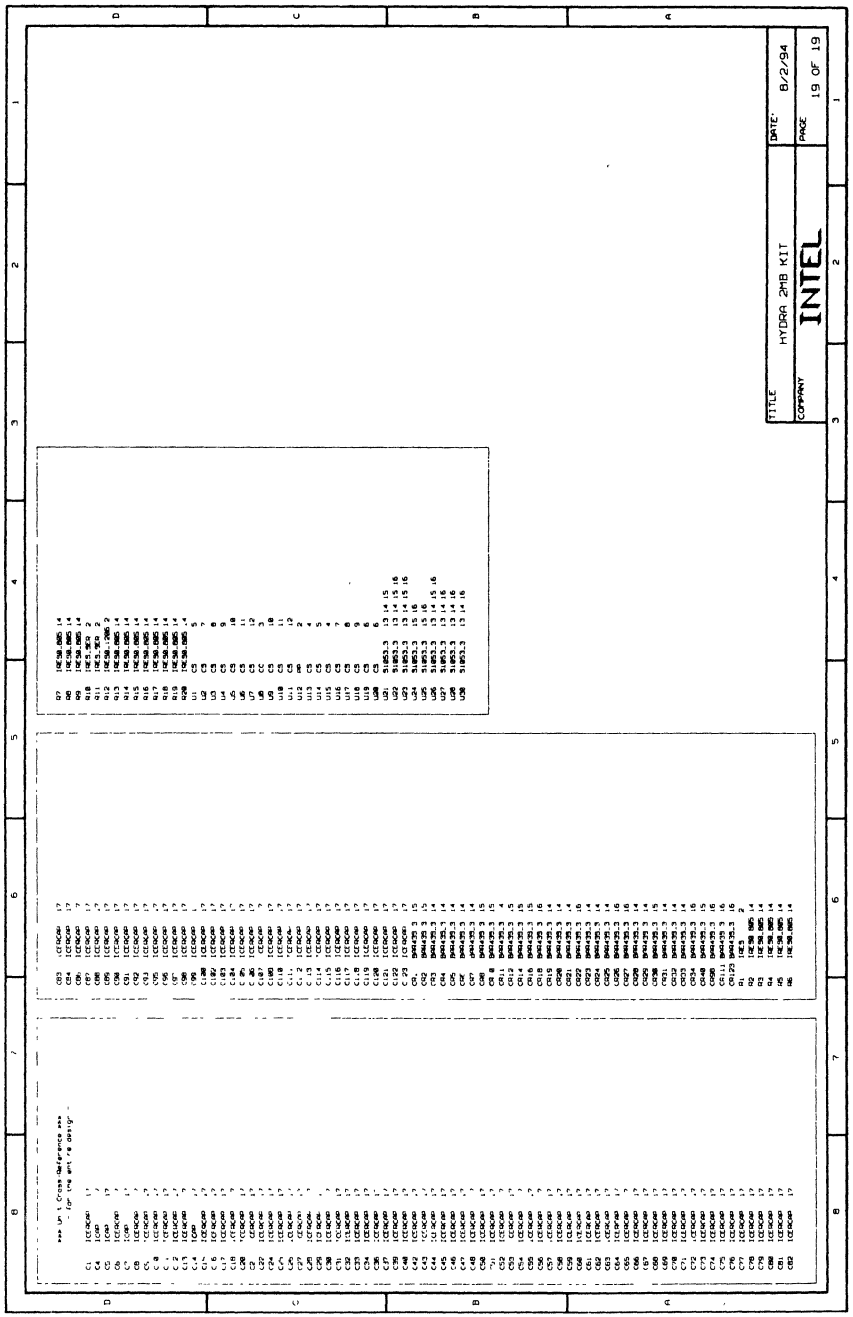

3.3V DIODES_2
TITLE
HYDRA 2MB KIT
COMPANY
INTEL
DATE
8/2/94
PAGE
14 OF 19


3.3V DIODES 3
TITLE HYDRA 2MB KIT
COMPANY INTEL
DATE 8/2/94
PAGE 15 OF 19


3.3V DIODES 4
TITLE HYDRA 2MB KIT
DATE 8/2/94
COMPANY INTEL
PAGE 15 OF 19


BYPASS CAPS
TITLE
HYJRA 2MB KIT
COMPANY
INTEL
DATE
8/2/94
PAGE
17 OF 19
VCCCORE
+3.3V


TITLE: HYDRA 2MB KIT
DATE: 8/2/94
COMPANY: INTEL
PAGE 18 OF 19


TITLE HYDRA 2MB KIT
DATE 8/2/94
COMPANY INTEL
PAGE 19 OF 19

4.7. Non-Parity Layout

Intel has not simulated a non-parity layout example. The following suggestions will assist in modifying the design example for non-parity implementations. You must simulate all paths that are altered when the parity components are removed to ensure that flight time and signal quality specifications are still met.

Modify the following aspects of the layout example:

1. Remove the two leftmost 82493 components. These are the parity components.
2. Rework Topology 1. Keep the trace leading to the Pentium processor (735\90, 815\100, 1000\120, 1110\133) length La. Retune (if needed) the trace to the 82498 of length Lb to be electrically equivalent.
3. Rework topologies 3 and 3b. Remove the traces which led to the parity components which have already been removed.
4. Remove the Byte Enable traces that connect to the parity chips.

Making traces electrically equivalent means that reflections from all branches return to the source at the same point in time. In simple cases, electrically equivalent traces are the same length. In all cases, simulate the effects of changing trace lengths to find the proper trace length and routing.

5.0. 2-MBYTE PENTIUM® PROCESSOR (735\90, 815\100, 1000\120, 1110\133)/82498/82493 CPU-CACHE CHIP SET LAYOUT EXAMPLE

This chapter contains an example layout design for Intel's 2-Mbyte CPU-Cache Chip Set. Intel has simulated and validated the design by measuring the flight time and signal quality parameters on boards based on the design example.

The intent of the design example is to provide system designers a starting point. It provides one solution of how the Pentium processor (735\90, 815\100, 1000\120, 1110\133), 82498 cache controller, and 82493 cache SRAM components can be placed and routed to ensure flight time and signal quality specifications are met. It is not the only solution. System designers can alter the layout to meet their system requirements as long as the flight time and signal quality specifications are met.

5.1. Layout Objectives

The 2-Mbyte layout is an example of a CPU-Cache chip set arrangement that meets Intel's chip set specifications. The layout consists of one Pentium processor (735\90, 815\100, 1000\120, 1110\133), one 82498 cache controller, and eighteen 82493 cache SRAMs for a 2-Mbyte second-level cache with parity. Although the layout is specifically designed for a chip set with parity, conversion to a non-parity layout will also be discussed.

This example layout follows the chip set's flight time and signal quality specifications. In addition to meeting those specifications, these objectives were used:

1. To design the optimized portion of the interface so that the layout is not limited by interconnect performance. By not artificially creating any critical paths, the interface can yield maximum performance of the chip set.
2. To be consistent with EMI and thermal requirements.
3. To have the layout be used as a validation and correlation vehicle by Intel. Intel used the layout to validate the optimized interface of the chip set, measure flight times and signal quality, and tune input and output buffers.

Provided are complete specifications for a board layout: parts list, board layer plots, and the electronic files in Gerber format. Also provided are a set of topologies and line lengths so it will be easy to understand how the layout was generated.

5.2. Component Placement

To meet flight time with clock skew restrictions the parts should be placed in relative proximity to each other. At the same time, we ensured that the layout's Memory Bus Controller (MBC) interface signals are routable. Figure 25 illustrates how the chip set components are placed in the layout example. The dot indicates the location of pin 1. Figure 25 is a component top view of the layout.

5.3. Signal Routing/Topologies

Tables 10 and 11 list the signal nets and their corresponding topologies for the optimized and external interfaces of the CPU-Cache Chip Set.

All chip set signals in the optimized interface fall into six groups: Low addresses, High addresses, Pentium

processor (735\90, 815\100, 1000\120, 1110\133) control, 82498 control, CPU data and parity, and Byte enables. Within each group are subsets of signals that share common origination and destination points. Each subset has a unique routing called a "topology". Groups, subsets, and topologies are listed in Table 10.

Topologies are given only for signals that are routed to multiple chips. It is the system designer's responsibility for routing the "point-to-point" signals such as CADS#.

Topologies are also supplied for the external interface. These topologies provide channels for routing signals from the chip set components to the periphery where they can be connected to the memory bus and memory bus controller (MBC). However, topologies are not supplied for point to point signals in the MBC interface (e.g. CRDY#). Instead, the system designer must optimize these for the particular application.

Table 11 lists the topologies provided for the MBC interface signals which are not point to point.

Figures 26 through 41 contain the topologies which are described in Tables 10 and 11. A topology is a graphical representation of how specific sets of signals are routed. A topology shows the components that share a specific signal and the relative lengths of the traces between components. A small box with a capital D inside represents a clamping diode in the topology diagrams. Refer to the schematics (section 5.6.1) for details on the type of diode required.

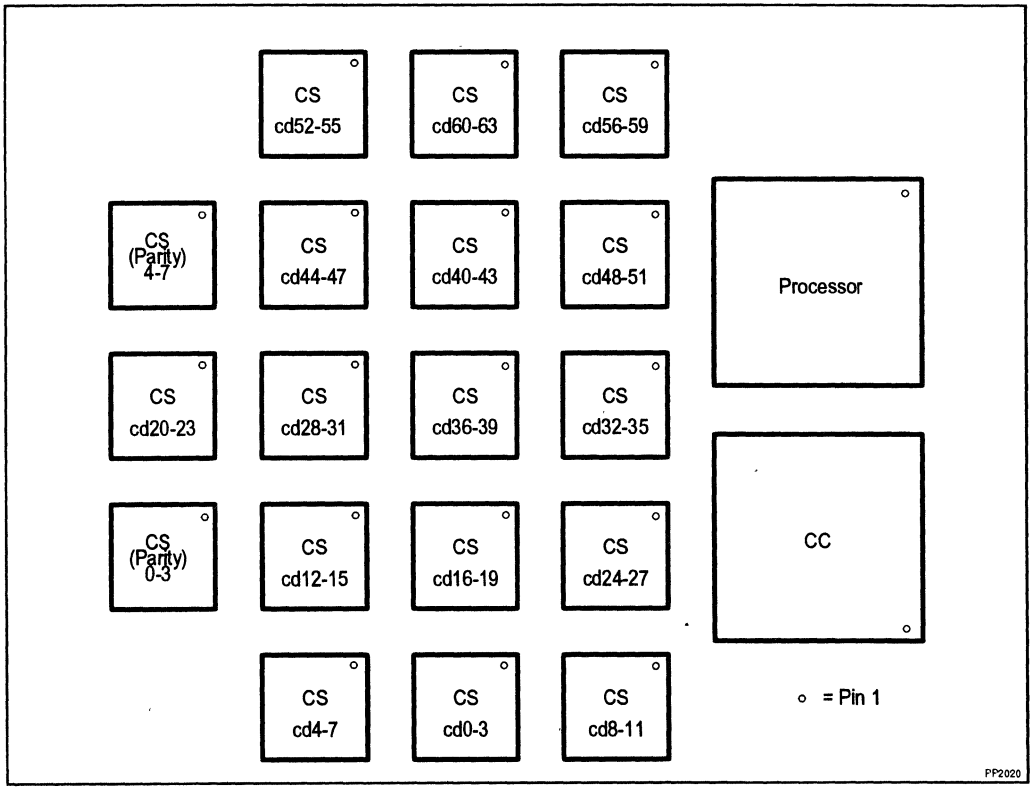


Figure 25. Component Placement

Table 10. 2-Mbyte Optimized Interface Signal Net/Topology Assignments

Grouping	Routing Requirements	Topology
Low Addresses		
(PA3-PA19)	Lines as short as possible. Line segments between all 82493s in the array are equal to retain symmetry. La,Lc: as short as possible. Diodes at the end of the 82493 array and at the 82498 cache controller. La: 0.9~1.2", W=15 Mils. Lb: 3.7~3.9". Lc: 1.0~1.17". Ld: <1.2". Le: <1.0".	1
High Addresses		
(PA20-PA21)	Point to point links with resistors. PA20: La + Lb = 4.270". PA21: La + Lb=4.150".	2b
(PA22-PA31)	Point to point links. L: 4.0~4.2".	2
Pentium® Processor (735\90, 815\100, 1000\120, 1110\133) Control		
(HITM#, W/R#)	Lines as short as possible. Line segments between all 82493s in the array are equal to retain symmetry. La, Lc as short as possible. La: Surface layer, W = 50 Mils or as wide as possible. Lc: 1.0~1.17", Ld: <1.2", Le: <1.0". For HITM#, La: 2.7", Lb: 5.075". For W/R#, La: 2.65", Lb: 4.6".	1
(ADS#)	Lines as short as possible. Line segments between all 82493s in the array are equal to retain symmetry. La: 3.94" and as short as possible, W = 15 Mils or as wide as possible. All traces on surface layer. Diodes at the end of the 82493 array. Ld: 1.2". Le: 1.0".	3
(ADSC#, AP, BOFF1#, CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC, WB/WT#)	Point to point links. L: 4.0~4.2". BOFF1# and BRDYC1# L=5.5", AP: >4", surface layer, and as short as possible. WB/WT# is surface layer.	2
82498 Control		
(BLAST#, BOFF2#, BUS#, MAWEA#, WBA, WBTYP, WBWE#, BRDYC2#, MCYC#, WRARR#, WAY)	Lines as short as possible. Line segments between all 82493s in the array are equal to retain symmetry. Lb: <2.587" and as short as possible. Diodes at the end of the 82493 array. Ld: <1.2". Le: <1.0". For these signals (priority top to bottom), Lb is surface layer and greater than or equal to 15 Mils (or if not possible, then inner layer is greater than or equal to 5 Mils wide) or as wide as possible: BRDYC2#, BOFF2#, BLAST#, WBTYP, WBWE#, WBA, MWEA#, BUS#	3b
(BLEC#)	Connected to the Cache controller and eight of the 82493s. La and La: 2.72" and Ld: 1.795" and as short as possible.	4
(AHOLD, BRDYC1#, EADS#, INV, KEN#, NA#, EWBE#)	Point to point links. Must be kept as short as possible.	2
CPU Data and Parity		
(CD0-CD63, CP0-CP7)	Point to point links. L: 5.50~7.4"	2
Byte Enables		
(CBE0#-CBE7#)	Connected to the Pentium processor (735\90, 815\100, 1000\120, 1110\133), two cache 82493s and one parity 82493. La: 5~9".	5

Table 11. 2-Mbyte External Interface Signal Net/Topology Assignments

Signal	Topology
MD0-63, Parity0-7	8
BRDY0#, CLK0	9
CRDY#, RESETC	10
MBRDY#, MOCLK, MDOE#, MEOC#	11
BRDY#'s, CLK's	12
MFRZ#, MZBT#, MCLK	14
RESETCPU	15
MSEL#	16 & 17

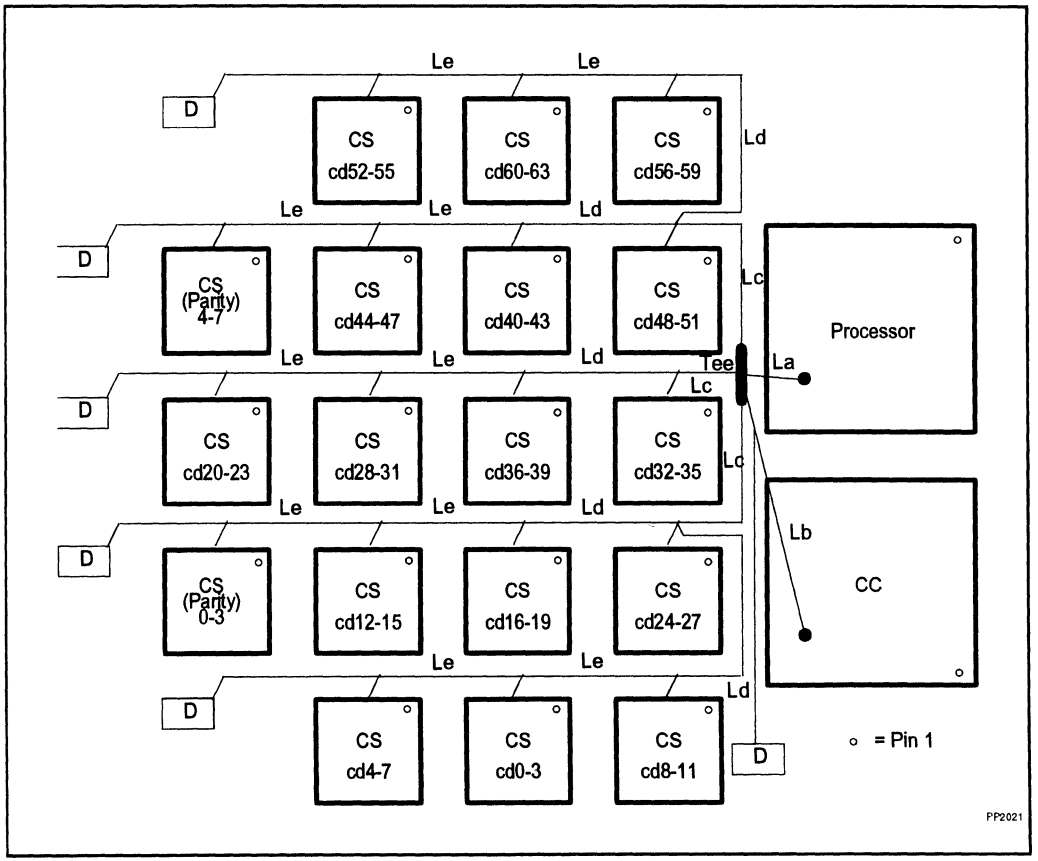


Figure 26. Topology 1

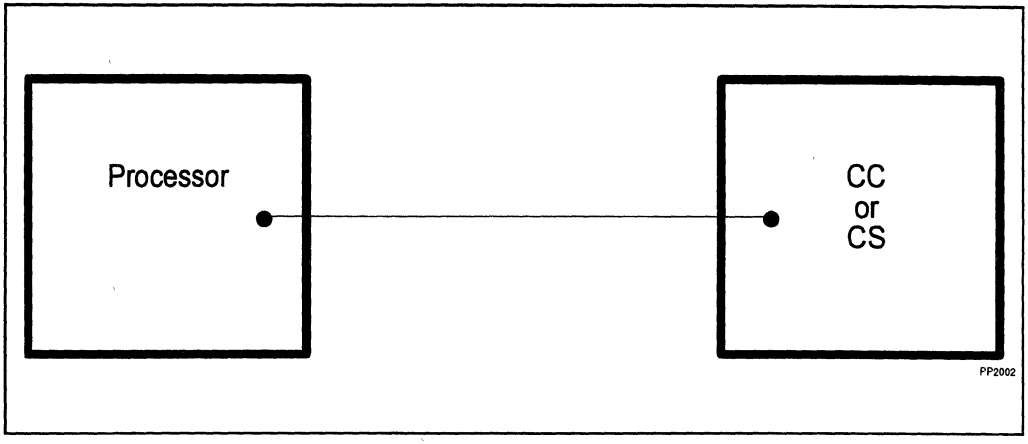


Figure 27. Topology 2

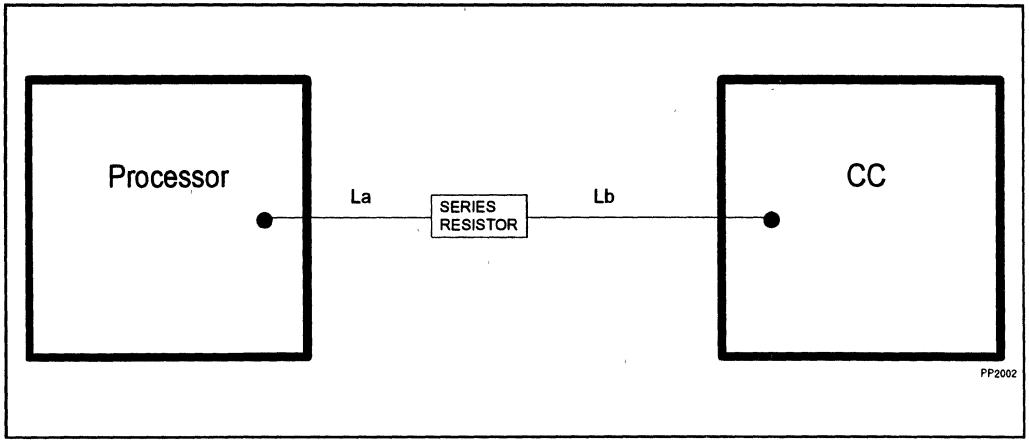


Figure 28. Topology 2b

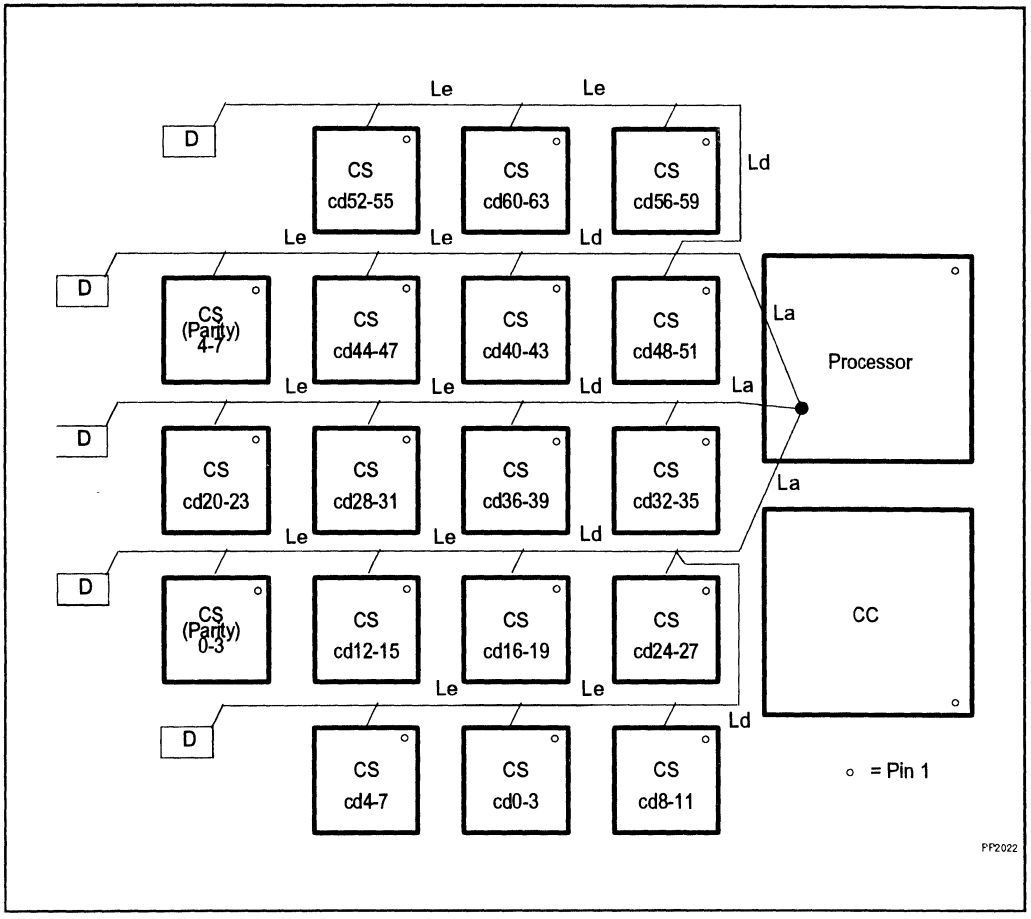


Figure 29. Topology 3

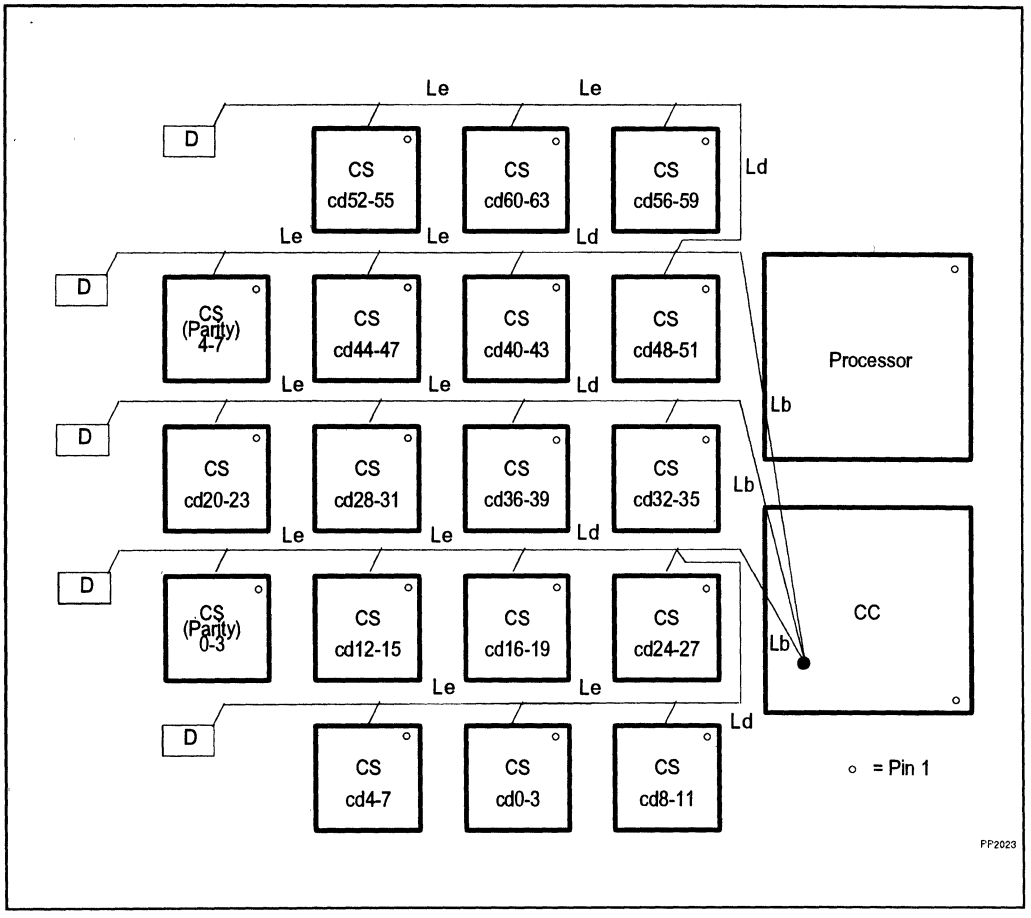


Figure 30. Topology 3b

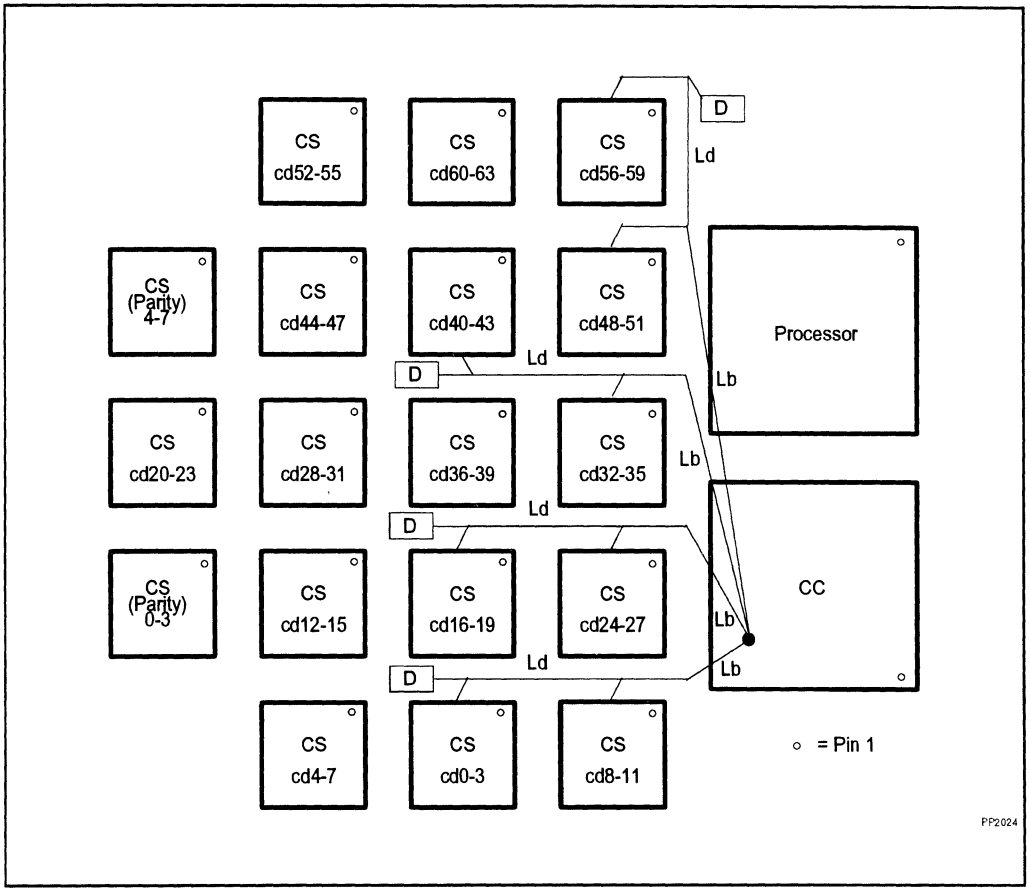


Figure 31. Topology 4

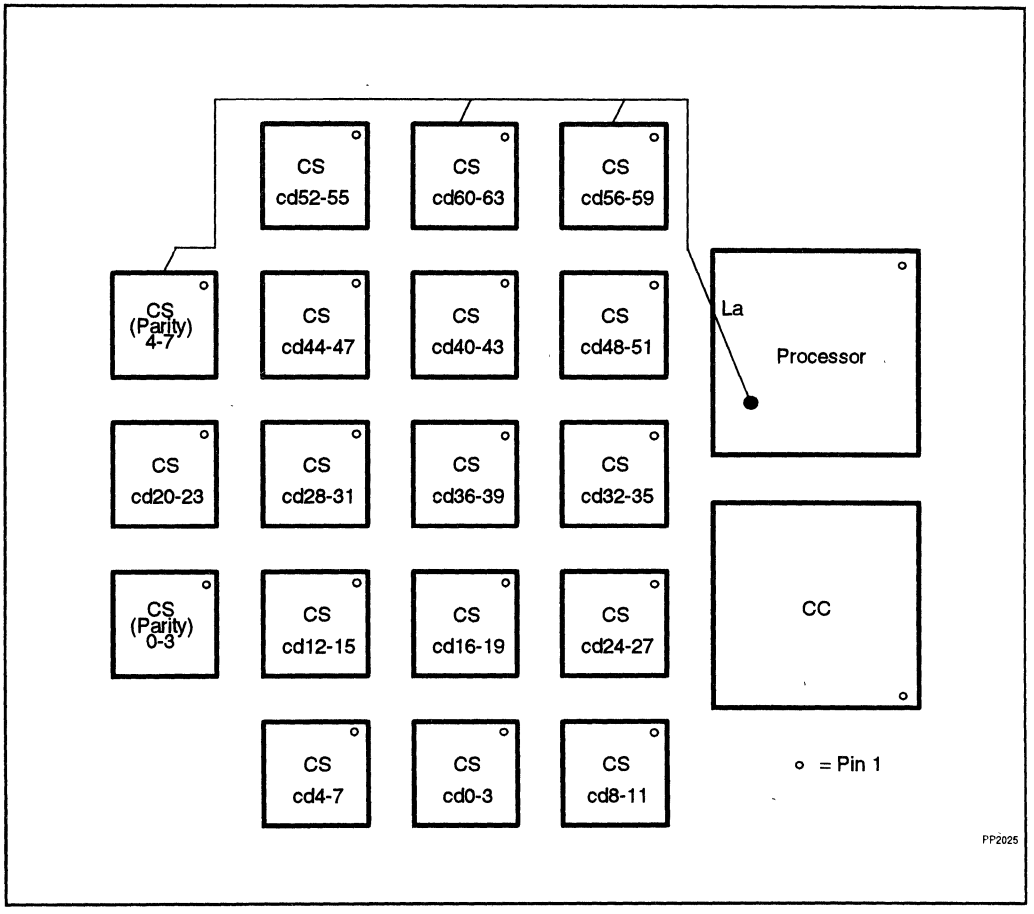


Figure 32. Topology 5

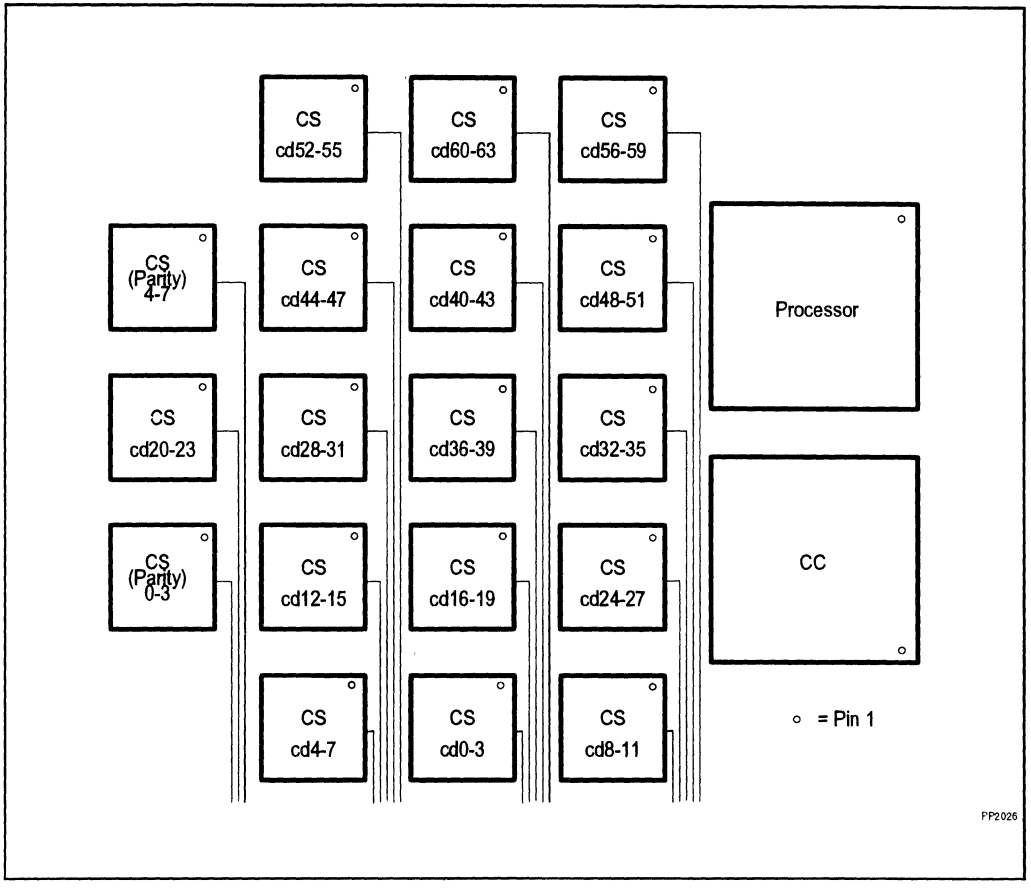


Figure 33. Topology 8

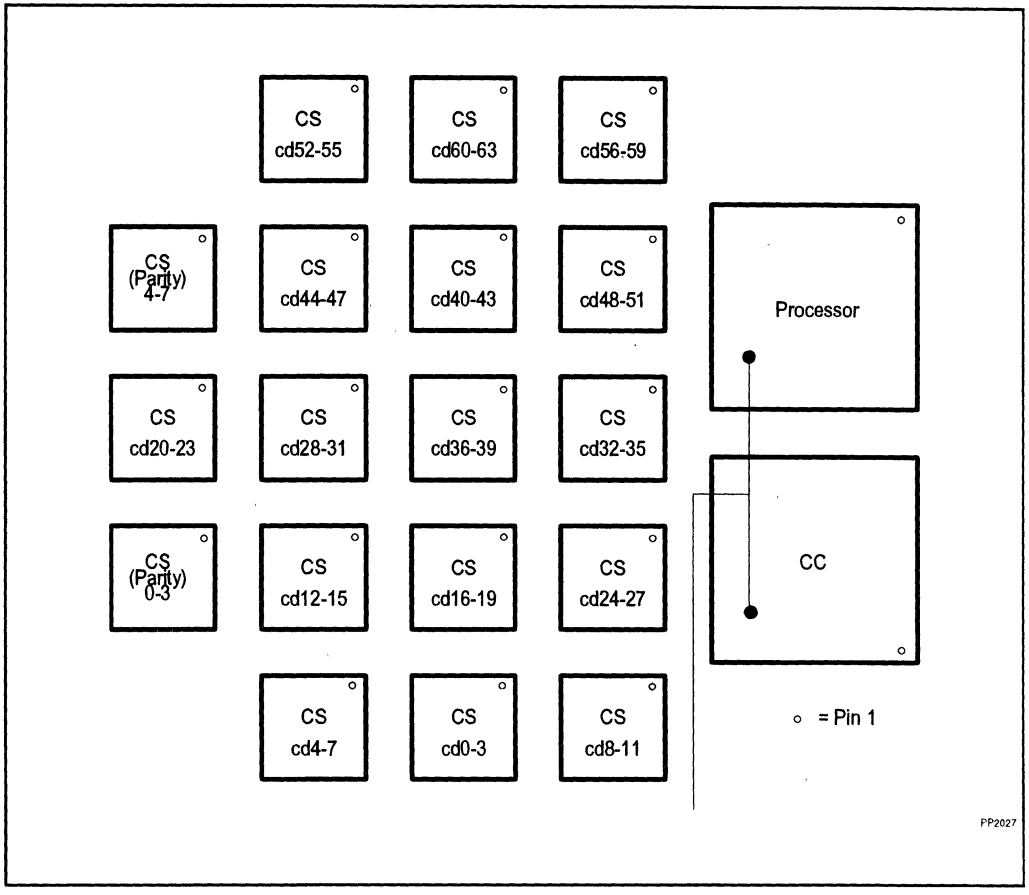


Figure 34. Topology 9

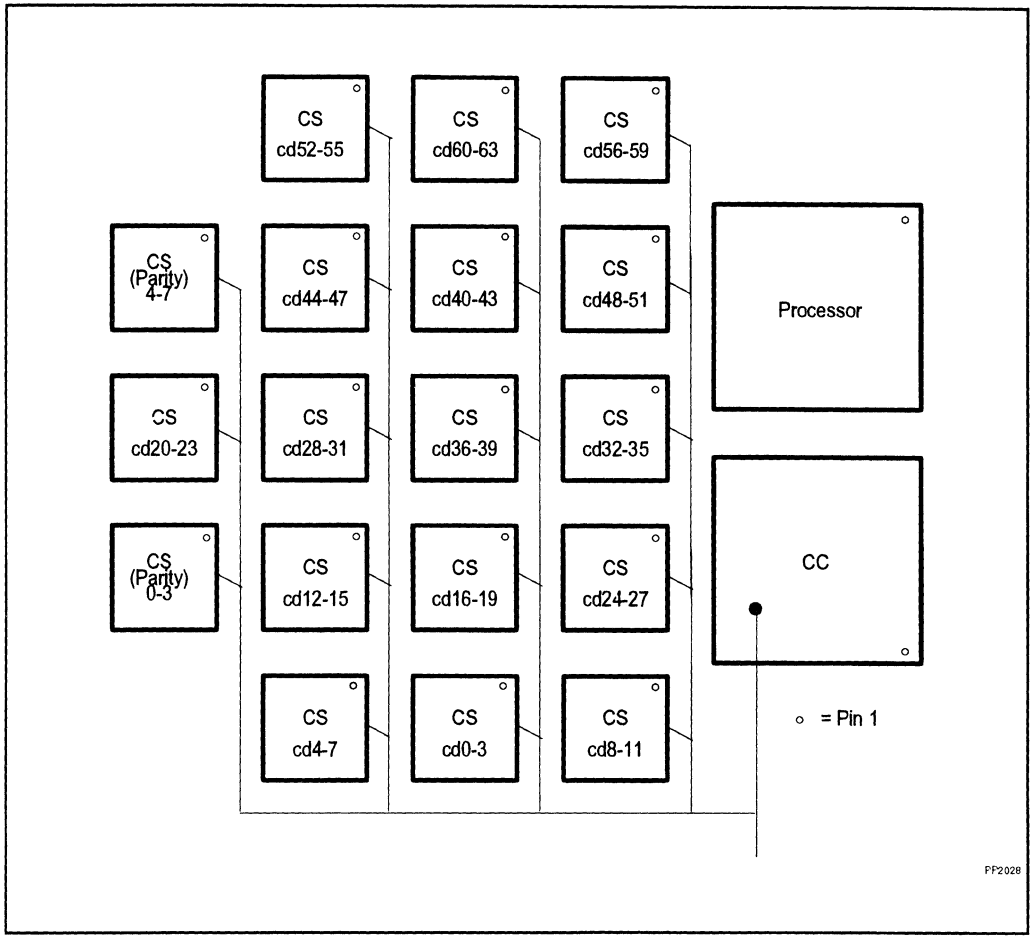


Figure 35. Topology 10

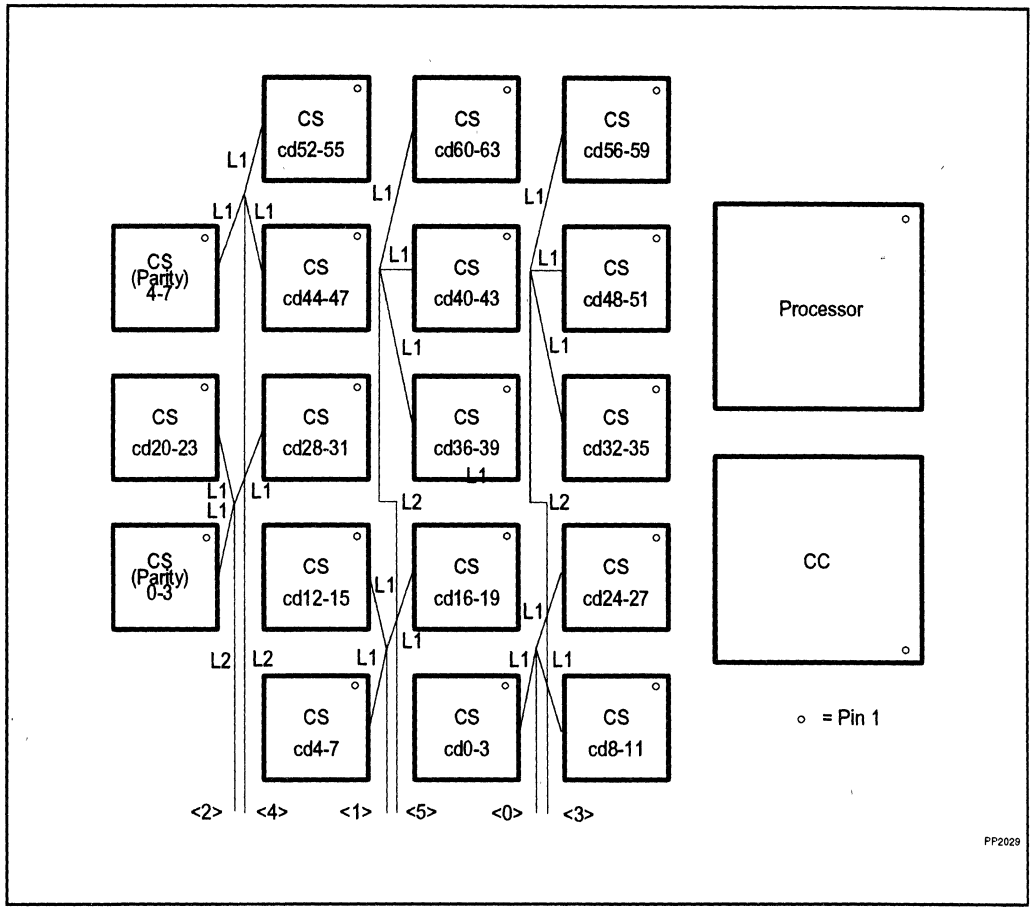


Figure 36. Topology 11

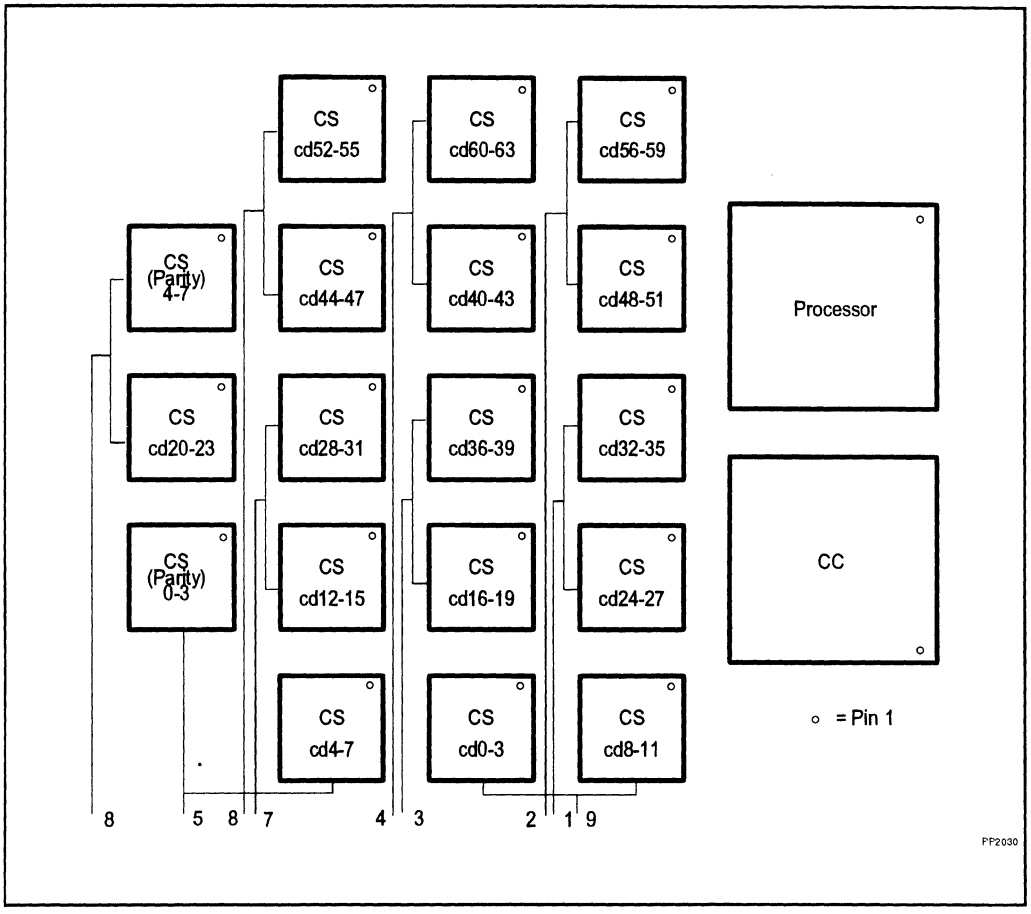


Figure 37. Topology 12

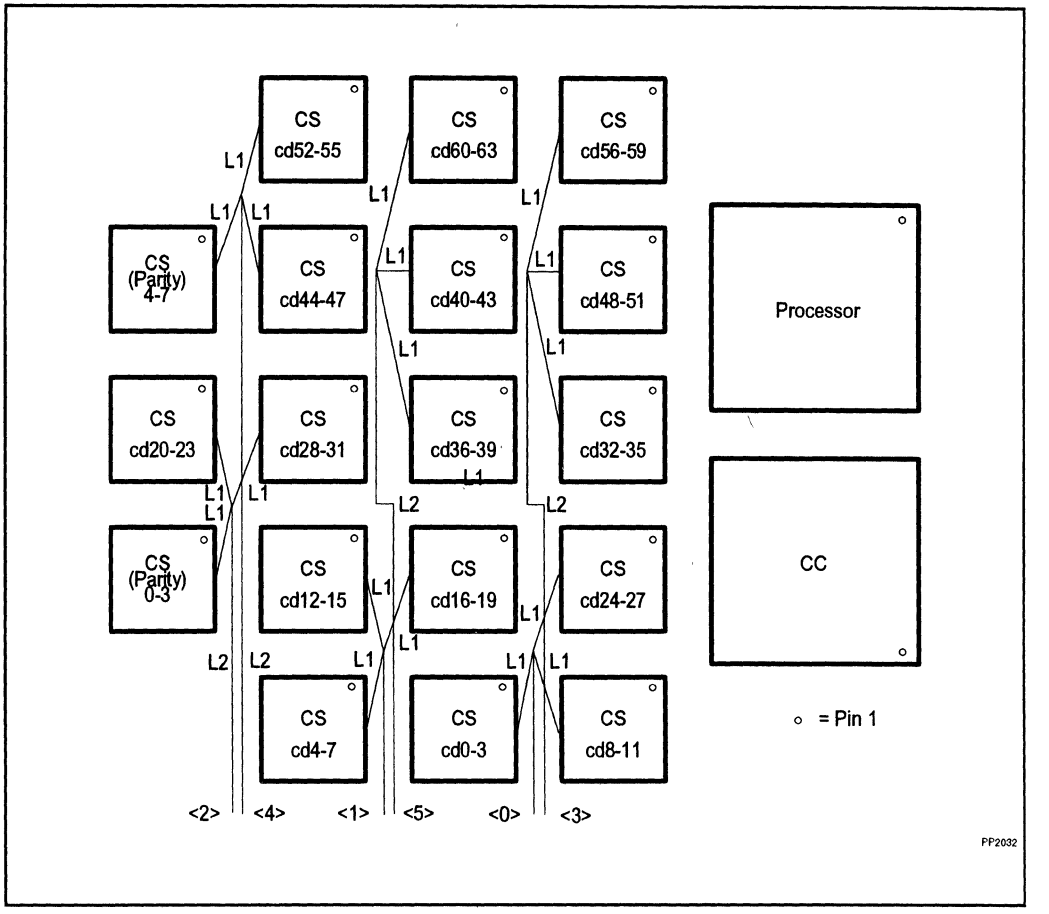


Figure 38. Topology 14

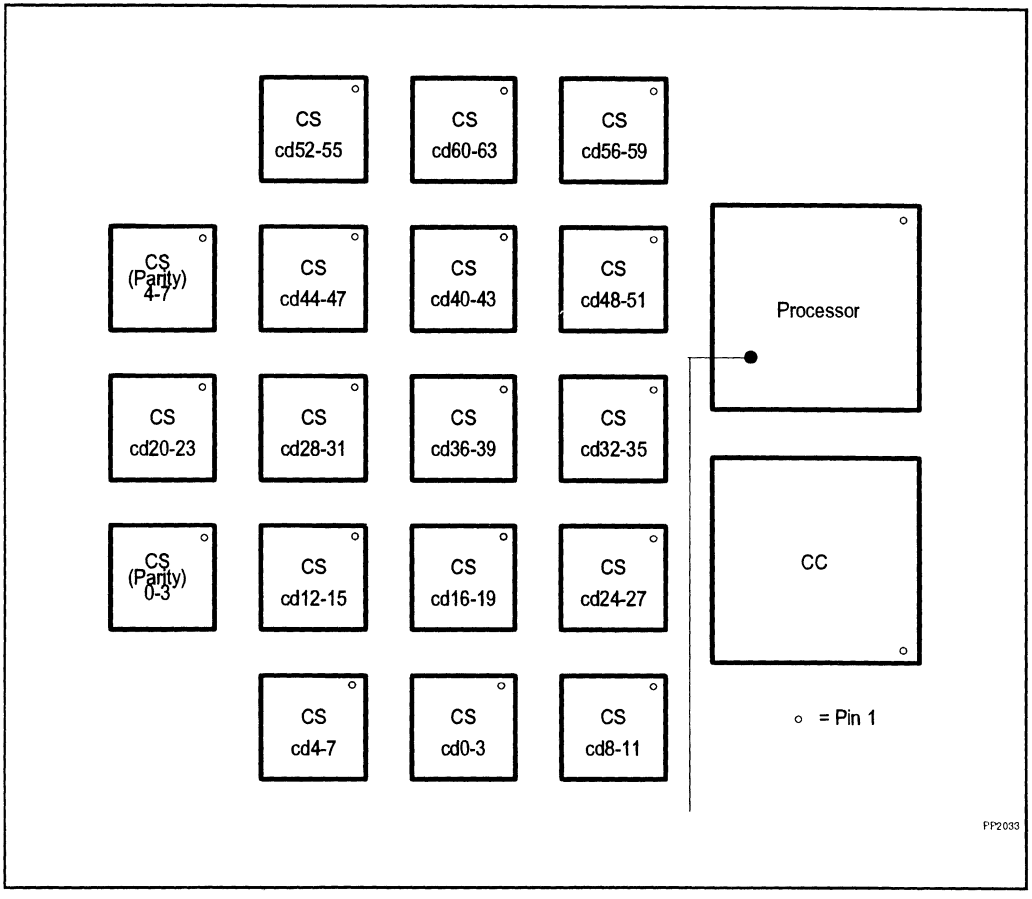


Figure 39. Topology 15

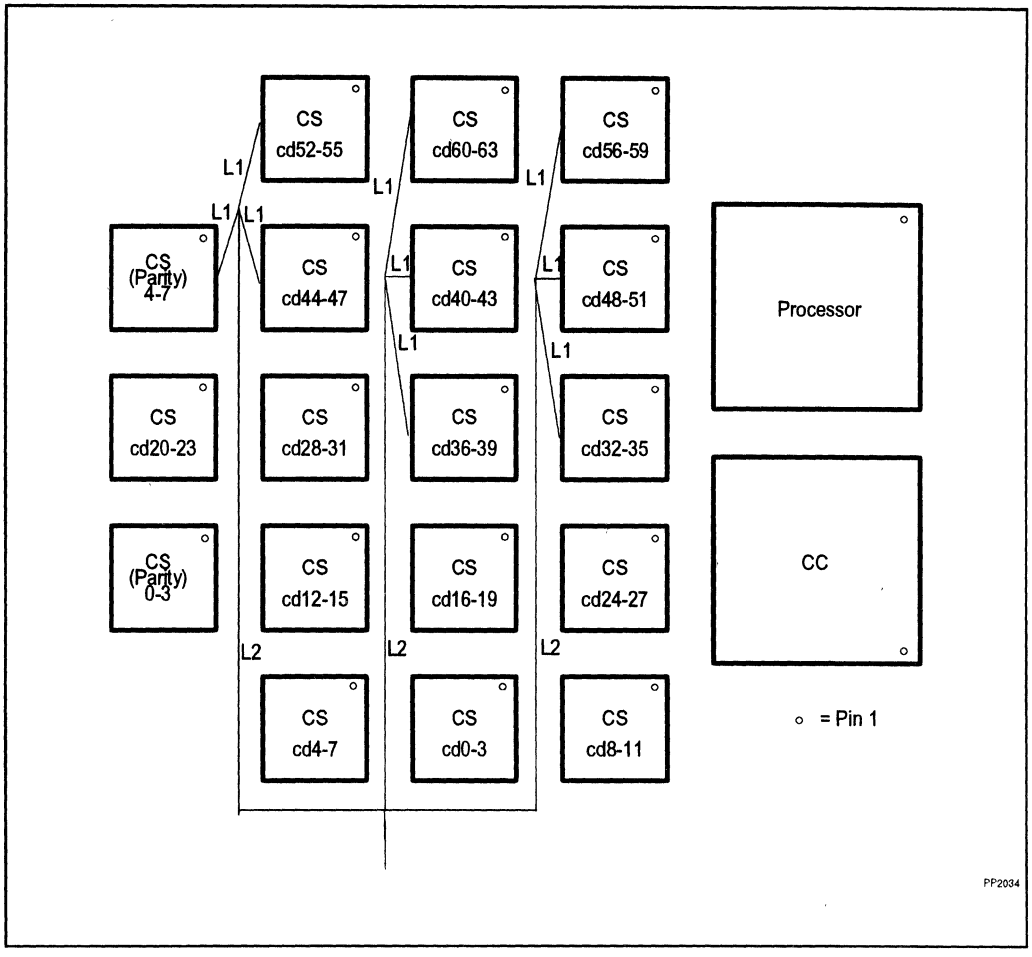


Figure 40. Topology 16

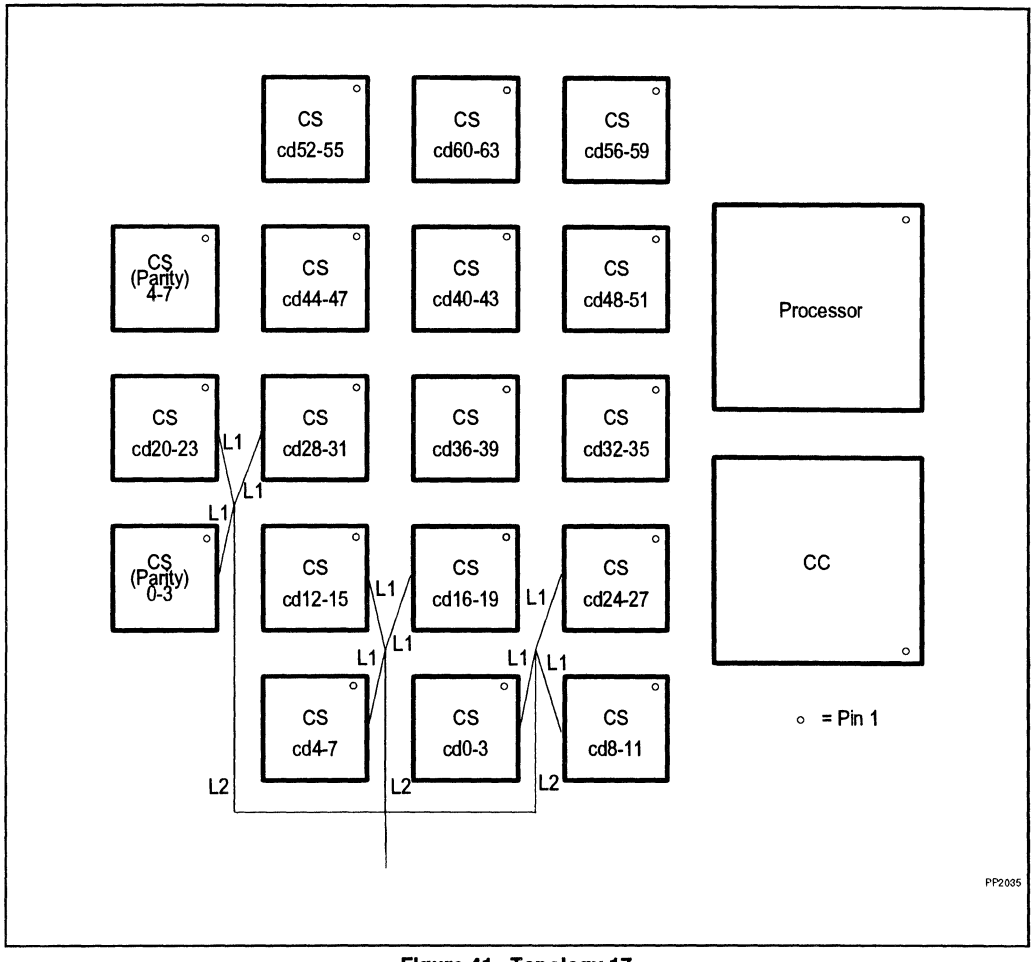


Figure 41. Topology 17

5.4. Board/Trace Properties

Specific board and trace properties were assumed while performing the simulations to optimize the chip set layout. These properties were used as the specification or guideline the board manufacturer should use in building boards. Figure 42 provides the board layer stackup.

Table 12 lists the minimum and maximum trace characteristics. These parameters along with the board material determine the spacing between layers and the total board thickness. See Table 13.

5.5. Variable Line Widths

Variable Line Widths are used for impedance matching between driver and receiver, consequently increasing a signal's velocity. Tables 14 and 15 show the impedance and velocities for a given line width.

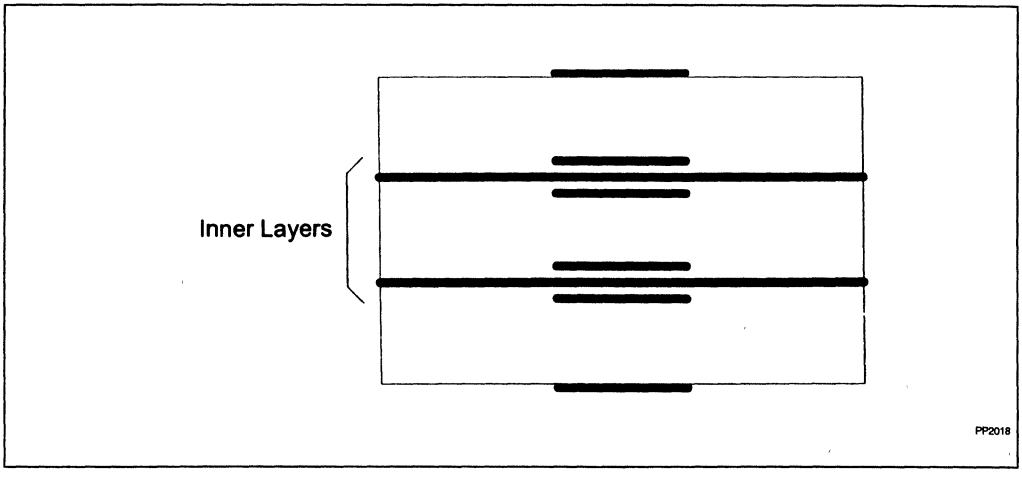


Figure 42. Board Layer Stackup

Table 12. TT_TraceChar2Trace Characteristics

	4 Inner Layers	2 Outer Layers
Width/Space	5/5 Mils	8/8 Mils
Z_0	65Ω ± 10%	75Ω ± 10%
Velocity	1.85 to 2.41 ns/ft	1.5 to 2.1 ns/ft

Table 13. TT_OtherPCBGeos2Other Printed Circuit Board Geometries

Via Pad	25 Mils
Via Hole	10 Mils
Pentium® Processor (735\90, 815\100, 1000\120, 1110\133) SPGA Pad	45 Mils
Pentium Processor (735\90, 815\100, 1000\120, 1110\133) SPGA Hole	29 Mils
82498 PGA Pad	55 Mils
82498 PGA Hole	38 Mils
82493 Surface Mount Package Pad	70x13 Primary Side
Layout Grid	5 Mils

Table 14. TT_NomLineWidthMicrostrip Line Impedance and Velocity as a Function of Nominal Line Width

W_{NOM}	Z_{MIN}	Z_{MAX}	V_{MIN}	V_{MAX}
8.00	69.60	105.00	1.61	1.98
10.00	62.80	94.70	1.64	2.01
15.00	76.10	52.00	1.64	2.03
20.00	44.20	65.60	1.67	2.06
25.00	34.30	50.60	1.67	2.08
30.00	30.90	45.90	1.70	2.08
35.00	28.20	41.60	1.70	2.11
40.00	25.70	38.40	1.70	2.13
45.00	23.80	35.30	1.73	2.13
50.00	22.20	32.80	1.73	2.13

Table 15. Stripline Impedance and Velocity as a Function of Line Width

W_{NOM}	Z_{MIN}	Z_{MAX}	V_{MIN}	V_{MAX}
5.00	48.00	89.20	1.90	2.36
10.00	35.10	60.00	1.90	2.36
15.00	49.20	27.90	1.90	2.36
20.00	23.20	41.10	1.90	2.36
25.00	19.80	35.30	1.90	2.36
30.00	17.40	31.10	1.90	2.36
35.00	15.50	28.00	1.90	2.36
40.00	14.10	25.30	1.90	2.36
45.00	12.80	23.10	1.90	2.36
50.00	11.80	21.20	1.90	2.36

5.6. Design Notes

1. All fast-switching signals are routed near the power and ground planes on inner layers of the board to minimize EMI effects. However, two sets of signals are routed on the top layer of the board: BRDYC1#, and JTAG signals. BRDYC1# is routed on top to take advantage of the higher trace velocity. JTAG signals can be routed on any layer because they are low-speed signals and will probably be rerouted by each customer to suit individual needs.
2. Resistor R1(0Ω) on page 2 of the 1-Mbyte and 2-Mbyte schematics is used to set the Pentium processor (735\90, 815\100, 1000\120, 1110\133) configurable output buffers (A3-A20, ADS#, W/R#, and HITM#). When the resistor is included the buffers are set to the extra large size. When it is not included (BUSCHK# internally pulled high) the buffers are set to the large size. Intel currently recommends the extra-large buffers be used for the 2-Mbyte layout example. The 0Ω resistor should be designed in your design for flexibility.
3. The 82498 output buffers that drive the 82493 inputs must also be configured to be large. This is done by driving 82498 CLDRV [BGT#] high during reset. The 82498 and 82493 memory bus buffer sizes must be controlled by the Memory Bus Controller. Please refer to the *Pentium® Family Developer's Manual Volume 2: 82496/82497/82498 Cache Controller and 82491/82492/82493 Cache SRAM.*
4. Series termination resistors were added to the nets, PA20, PA21 and several other nets to control overshoot. A value of 24Ω is recommended, but the value depends on overshoot measurement on the actual design.
5. Several Topologies require different line widths on the same trace. This information can be extracted automatically using a Quad Design translation tool. For extraction procedure, please refer to Quad Design's user's manual, "Preparing PCB Design Databases for Simulation with TLC/XTK."
6. Layout Rules/Common Definitions/Explanations: Identical symbols for transmission lines means that the lines should be identical in Length, Width, and Configuration (e.g. Strip, Microstrip etc.). The term "as short as possible but greater than a certain length" is defined as follows. In order to satisfy the maximum flight time requirements, generally, it is desired that the transmission lines be as short as possible. Many signals however have tight minimum flight time requirements. For these signals it is necessary to lengthen the lines, but not too long so as to violate the maximum flight times. Whenever possible, a length window is given so that both min timings and max timings are specified. When both margins are very tight and the signals are heavily loaded with too many transmission lines, it is very difficult to specify a range of lengths for every segment on that topology without being too restrictive. Instead the strategy was to lengthen a specific segment to satisfy min timings and shorten all other segments as short as possible to meet max timings. The exact lengths can be extracted from the layout.

5.7. Explanation of Information Provided

The following sections outline the design files associated with the 2-Mbyte CPU-Cache Chip Set design example that are available from Intel. These files are provided to simplify the task of porting the design example into a specific design. By using these files, designers may eliminate or minimize the amount of duplicate effort when using the design example as the basis for their design. The following items are available:

- Schematics

- I/O Model Files
- Board Files
- Bill of Materials
- Photoplot Log
- Netlist Report
- Placed Component Report
- Artwork for Each Board Layer
- Trace Segment Line Lengths

Hard copies of the schematics are provided in the following section. ASCII or soft copies of all the information are available from Intel Order Number 242658.

5.7.1. SCHEMATICS

Schematics for the 2-Mbyte CPU-Cache Chip Set design example were created using Cadence/Valid "Concept" Schematic Capture program V1.5-p3. The schematics are 19 pages long and are provided in A size format. Both the Cadence and the postscript files (A size format) are available from Intel as described above.

5.7.2. I/O MODEL FILES

All electrical I/O simulations were performed using TLC V4.2.8 from Quad Design Technology, Inc. The simulations were performed at the fast and slow corners to verify that all signal quality and flight time specifications were met. The files used for these simulations are available from Intel as described above. These files include the topology, model, and control files needed to run the simulations for all nets in the optimized interface.

5.7.3. BOARD FILES

The board files for the design example were created using Allegro V6.1 from Cadence Design Systems, Inc. The files are available from Intel as described above. These files may be used to import the design example into a specific system design.

NOTE

Some changes to the layout and nets may be necessary to complete importing these files into a specific system design.

5.7.4. BILL OF MATERIALS

The bill of materials file was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

5.7.5. PHOTOPLOT LOG

The photoplot log file was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

5.7.6. NETLIST REPORT

The netlist report was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

5.7.7. PLACED COMPONENT REPORT

The placed component report was created using Allegro V6.1 from Cadence Design Systems, Inc. The file is available from Intel as described above.

5.7.8. ARTWORK FOR EACH BOARD LAYER

The artwork for the six board layers was created using Allegro V6.1 from Cadence Design Systems, Inc. The files are available from Intel in a Gerber format as described above.

5.7.9. TRACE SEGMENT LINE LENGTHS

The file containing the trace segment line lengths was created from the artwork files of section 5.6.8. All lengths are provided in mils (1/1000 inch). The stubs listed in the tables are associated with the pin escapes required for the 82493s. The file is available from Intel as described above.

5.7.10. I/O SIMULATION RESULTS FOR EACH NET

Electrical simulations were performed on each net within the optimized interface of the 2-Mbyte CPU-Cache Chip Set design example. The simulations were done at the fast and slow corners to verify that signal quality and flight time specifications are met. The simulations were done using TLC V4.2.8 from Quad Design Technology, Inc. using the files described in section 5.6.2. Table 16 summarizes the slow corner

simulation results and Table 17 summarizes the fast corner simulation results.

The flight time specifications listed in Tables 16 and 17 are the 60/66-MHz flight times.

Table 16. 2-Mbyte Slow Corner Simulation Results

Net	Flight Time (ns)			Signal Quality	
	Maximum Flight Time	Maximum Flight Time Specification	Flight Time Margin	Settling Time (ns)	Overshoot (V)
82498 Driving					
PA3-19	11.41	11.2	-0.21	24.78	0
PA20-31	2.35	3.2	0.85	7.49	0
AHOLD	0.71	1.5	0.79	3.85	0.08
AP	1.17	3.2	2.03	4.43	0.17
BRDYC1#	0.97	1.5	0.53	7.35	0.27
EADS#	0.64	1.5	0.86	3.83	0.06
EWBE#	0.79	1.5	0.71	3.9	0.13
INV	1.32	1.5	0.18	4.46	0.24
KEN#	0.71	1.5	0.79	3.85	0.08
NA#	0.67	1.5	0.83	3.77	0.06
WB/WT#	0.86	1.5	0.64	3.85	0.17
BLAST#	3	3.2	0.2	24.16	0.76
BLEC#	1.73	2.5	0.77	3.93	0.68
BOFF1# at CPU	1.21	1.5	0.29	4.55	0.15
BOFF2# at SRAM	2.73	3.2	0.47	18.96	1.95
BRDYC2#	2.76	3.2	0.44	21.59	0.8
BUS#	2.87	3.2	0.33	24.41	0.73
MAWEA#	2.88	3.2	0.32	24.56	0.73
MCYC#	2.91	3.2	0.29	23.84	0.81
WAY	2.91	3.2	0.29	23.77	0.82
WBA	2.93	3.2	0.27	24.89	0.72
WBTYP	2.93	3.2	0.27	24.96	0.72
WBWE#	2.89	3.2	0.29	24.83	0.7
WRARR#	2.98	3.2	0.22	24.78	0.75

Table 16. 2-Mbyte Slow Corner Simulation Results (Contd.)

Net	Flight Time (ns)			Signal Quality	
	Maximum Flight Time	Maximum Flight Time Specification	Flight Time Margin	Settling Time (ns)	Overshoot (V)
Pentium® Processor (735\90, 815\100, 1000\120, 1110\133) Driving					
PA3-19	3.45	3.7	0.25	6.76	0.56
PA20-31	1.61	1.8	0.19	3.49	0.76
CP0-7, CD0-63	1.77	2.0	0.23	5.06	0.18
ADS#	2.77	3.3	0.53	24.93	0.78
HITM#	3.4	3.6	0.2	6.73	0.65
WR#	3.2	3.6	0.4	6.52	0.63
ADSC#	1.32	1.5	0.18	4.65	0.14
AP	1.39	1.5	0.11	4.55	0.22
CACHE#	1.17	1.5	0.33	4.49	0.1
D/C#	1.16	1.5	0.34	4.48	0.1
LOCK#	1.16	1.5	0.34	4.48	0.1
M/IO#	1.17	1.5	0.33	4.49	0.09
PCD	1.17	1.5	0.33	4.49	0.09
PWT	1.17	1.5	0.33	4.49	0.1
SCYC	1.17	1.5	0.33	4.49	0.09
CBE0-7#	2.57	2.8	0.23	6.73	0.04
82493 Driving					
CP0-7, CD0-63	2.24	2.3	0.06	5.21	0.17

Table 17. 2-Mbyte Fast Corner Simulation Results

Net	Minimum Flight Time (ns)			Signal Quality (V)		
	Minimum Flight Time	Minimum Flight Time Specification	Minimum Flight Time Margin	Overshoot	Overshoot Specification	Overshoot Margin
82498 Driving						
PA3-19	2.24	0.8	1.44	0.33	2.6	2.27
PA20-31	0.7	0.6	0.1	2.05	2.6	0.55
AHOLD	0.58	0.6	-0.02	2.38	2.6	0.22
AP	0.65	0.6	0.05	2.45	2.6	0.15
BRDYC1#	0.8	0.6	0.2	2.58	2.6	0.02
EADS#	0.52	0.6	-0.08	2.3	2.6	0.3
EWBE#	0.63	0.6	0.03	2.46	2.6	0.3
INV	0.95	0.6	0.35	2.35	2.6	0.25
KEN#	0.58	0.6	-0.02	2.39	2.6	0.21
NA#	0.56	0.6	-0.04	2.36	2.6	0.24
WB/WT#	0.7	0.6	0.1	2.51	2.6	0.09
BLAST#	0.65	0.8	-0.15	1.36	2.6	1.24
BLEC#	0.71	0.8	-0.09	0.97	2.6	1.63
BOFF# at CPU	1	0.8	0.2	2.03	2.6	0.57
BOFF# at SRAM	0.51	0.8	-0.29	1.38	2.6	1.22
BRDYC2#	0.57	0.8	-0.23	1.35	2.6	1.25
BUS#	0.61	0.8	-0.19	1.42	2.6	1.18
MAWEA#	0.61	0.8	-0.19	1.42	2.6	1.18
MCYC#	0.61	0.8	-0.19	1.4	2.6	1.2
WAY	0.61	0.8	-0.19	1.4	2.6	1.2
WBA	0.67	0.8	-0.13	1.34	2.6	1.26
WBTYP	0.63	0.8	-0.17	1.39	2.6	1.21
WBWE#	0.61	0.8	-0.19	1.42	2.6	1.18
WRARR#	0.69	0.8	-0.11	1.26	2.6	1.34

Table 17. 2-Mbyte Fast Corner Simulation Results (Contd.)

Net	Minimum Flight Time (ns)			Signal Quality (V)		
	Minimum Flight Time	Minimum Flight Time Specification	Minimum Flight Time Margin	Overshoot	Overshoot Specification	Overshoot Margin
Pentium® Processor (735\90, 815\100, 1000\120, 1110\133) Driving						
PA3-19	1.2	0.8	0.4	1.09	2.6	1.51
PA20-31	0.89	0.6	0.29	2.07	2.6	0.53
CP0-7, CD0-63	0.96	0.8	0.16	2.01	2.6	0.59
ADS#	0.89	0.8	0.09	1.2	2.6	1.4
HITM#	1.31	0.8	0.51	1.03	2.6	1.57
WR#	1.13	0.8	0.33	1	2.6	1.6
ADSC#	0.83	0.6	0.23	2	2.6	0.6
AP	0.85	0.6	0.25	2.44	2.6	0.16
CACHE#	0.72	0.6	0.12	2.02	2.6	0.58
D/C#	0.71	0.6	0.11	2.02	2.6	0.58
LOCK#	0.71	0.6	0.11	2.02	2.6	0.58
M/IO#	0.71	0.6	0.11	1.99	2.6	0.61
PCD	0.72	0.6	0.12	2.02	2.6	0.58
PWT	0.71	0.6	0.11	2.02	2.6	0.58
SCYC	0.71	0.6	0.11	2	2.6	0.6
CBE0-7#	0.69	0.6	0.09	1.89	2.6	0.71
82493 Driving						
CP0-7, CD0-63	0.91	0.8	0.11	2.08	2.6	0.52

Note: An overshoot specification voltage in parentheses denotes the value for V_{SS} if different from V_{CC}.

intel®

AP-522

APPLICATION NOTE

Implementation Guidelines for 3.3V Pentium® Processors with VRE Specifications

2

for Desktop and Server Designs

Rev. 2.0

December 1995

Order Number: 242687-001

IMPLEMENTATION GUIDELINES FOR 3.3V Pentium® PROCESSORS WITH VRE SPECIFICATIONS FOR DESKTOP AND SERVER DESIGNS

CONTENTS

CONTENTS

1.0. INTRODUCTION

In addition to standard 3.3V Pentium® processors, Intel is offering 3.3V Pentium processors with VRE specifications to enable quicker time-to-market cycles, and higher-performance desktop and server systems. **This document will explain the voltage specifications, recommend solutions for supplying consistent power, and suggest validation techniques to ensure robust 3.3V Pentium processor-based desktop or server systems.** VRE (Voltage Regulated Extension) components have stricter supply voltage specifications than standard components, and as such VRE-based designs bring additional challenges to power regulation. Although this document focuses on VRE designs, the system design and voltage measurement concepts also apply to designs for standard components.

This document contains five key sections:

Chapter 2.0 discusses the standard and VRE specifications. The VRE and standard voltage ranges have been adjusted for the C2-step and all subsequent processors. These changes are reflected in the chapter (and the entire document). Chapter 2.0 also gives an overview of some important system design and voltage measurement considerations associated with VRE components. The consequences of specification violations are also discussed.

Chapter 3.0 deals with the power supply and regulation. It contains power implementation recommendations to ensure a robust system design. In addition, this chapter contains detailed low cost bulk and high speed decoupling recommendations for Socket 5, Socket 7, and standard 3.3V designs.

Chapter 4.0 explains the proper measurement techniques to verify that systems meet their respective voltage specifications. These measurement techniques apply to all Pentium processors. Measurement results from the *Pentium® Processor Flexible Motherboard Reference Design* are shown in this chapter.

Appendix A provides information about tools to assist in both simulating decoupling solutions, and in taking voltage measurements. This section also contains information on how to obtain and use the recommended "stress" code for voltage noise measurement.

Finally, Appendix B provides a list of third party vendors. These vendors include suppliers of regulators, resistors, capacitors, and sockets.

2.0. SPECIFICATIONS

2.1. The VRE Specification

The main difference between the VRE, and standard 3.3V specifications are the voltage supply requirements (see Table 1). Since the VRE specification has a stricter V_{cc} requirement than the standard voltage specification, VRE-based systems are very sensitive to voltage supply noise and transients. **Any overshoot or undershoot beyond the voltage range (at a measurement bandwidth of 20MHz) is not permitted.** Any transient excursion beyond the specified voltage range may result in unstable system behavior. Note that socket type and measuring techniques are also specified and should be followed to ensure consistent and accurate measurements.

Complete S-Specs, and availability of the Pentium processor family may be found in the latest Pentium processor stepping information or *Pentium® Processor Specification Update* (Order Number 242480).

The complete specifications shown above must be met to ensure a robust VRE-based platform. All measurements must be made and guaranteed at the back of the motherboard at the CPU socket pins. The voltage specifications assume an oscilloscope measurement bandwidth of 20MHz. Socket 5 or Socket 7 (or an equivalent socket of less than 5nH) should be used to ensure upgradability to future Pentium OverDrive® processors and to ensure accurate transient measurements.

Note that standard voltage range encompasses the VRE range, hence standard parts will operate in VRE systems (see Figure 1).

Table 1. Comparison of Standard Specifications to VRE Specifications

Specifications	Standard	VRE
V_{CC}	**3.135V[1] to 3.6V** No overshoot or undershoot allowed	**3.4V[2] to 3.6V** No overshoot or undershoot allowed
Timings	Standard and MD	MD only
Thermals	Same Maximum I_{CC} and Maximum Power Dissipation	
Socket	Pentium® OverDrive® Processor Upgrade Socket 5 or Socket 7	
Measurement	Transients must be measured and guaranteed at the back of the motherboard at the CPU socket pins. The measurement should be taken with a bandwidth of at least 20MHz (see section 4.2).	

Notes: [1]Applies to the C2 stepping, and all subsequent Pentium® processors.
[2]Applies to the C2 stepping, and all subsequent Pentium® processors with the VRE specification.

2.1.1. The VRE Supply Voltage Range

VRE components allow less transient tolerance than standard components. To compensate for the smaller transient tolerance, VRE-based platforms must use accurate voltage regulators and adequate local decoupling capacitors. During worst-case transient conditions (transition into and out of Stop Grant Mode or Halt Power Down Mode), current supplied to the processor can change by several amperes in tens of nanoseconds. Since power supply units and voltage regulators can at best respond in a time frame on the order of milliseconds, bulk decoupling capacitors are required to act as current reservoirs until the power supply unit or voltage regulators regulate to the new load. Due to the high operation speed of the internal core of the Pentium processor, high frequency capacitors are also required to filter the excessive noise components. Failure to provide adequate power regulation during this transition may result in undershoot and overshoot beyond the voltage specifications of the processor.

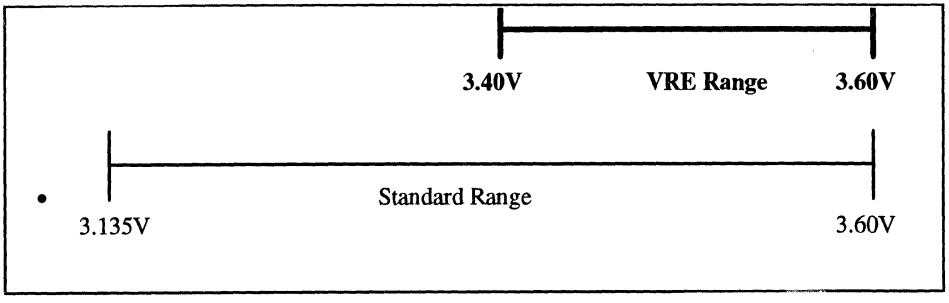


Figure 1. A Comparison of Standard and VRE Specifications

2.2. Typical Application Behavior

Poorly designed desktop and server systems may violate VRE specifications during normal operation. An unusual application instruction mix can cause large current spikes from clock cycle to clock cycle. Figure 2 shows the rapid fluctuations of system power during execution of a BAPCo93* benchmark trace (BAPCo93 is a system benchmark used for measuring system performance). These quick transitions in current occur in a shorter time frame than that in which the power supply unit or voltage regulator may be able to respond.

Worst-case transients occur during power management. Figure 3 shows an oscilloscope trace of a system leaving

the low-power Stop Grant State (via a deassertion of STPCLK#). The supply voltage "droops" due to ESL and ESR effects (see section 3.5), and because the voltage regulator cannot respond quickly enough to the large, instantaneous change in current. Droops and surges also occur in systems with proper decoupling, but to a lesser extent. The system also has high frequency noise due to high operation speed of the internal core. Figure 3 shows the "droop" due to ESR/ESL effects when exiting the Stop Grant state. The longer term voltage variations (on the order of milliseconds) are not shown in this plot. **Violating the VRE specifications by undershooting or overshooting the voltage range may result in unreliable and unstable behavior.** The consequences of voltage specification violations are explained in the next section. Chapter 3.0 will recommend techniques for providing accurate regulation and proper decoupling to ensure a robust VRE-based platform.

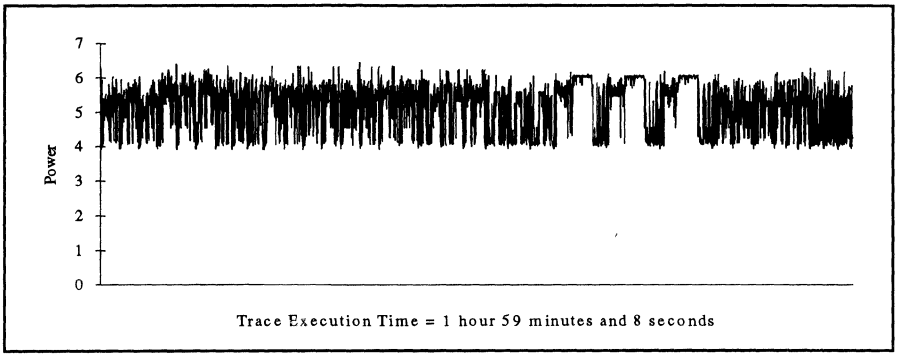


Figure 2. Rapid Fluctuations of System Power While During Active Operation (BAPCo93*)

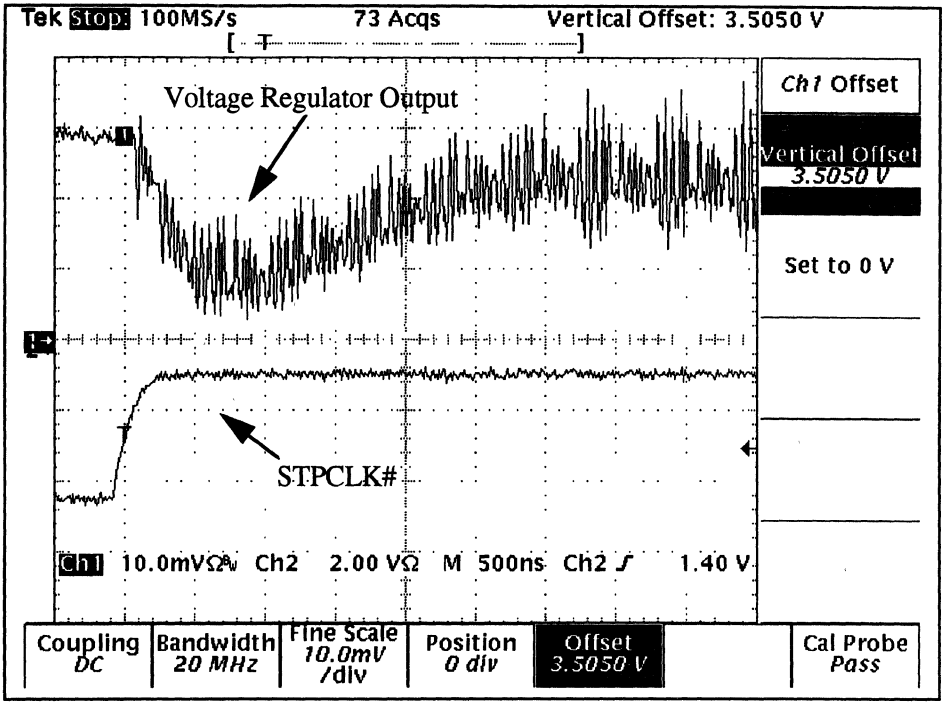


Figure 3. Voltage "Droop" when Exiting Stop Grant State

2.3. Voltage Specification Violations

Overshooting the voltage specification can cause certain signals to violate their Minimum Valid Delay timing specifications. This timing violation will in turn lead to a failure in the system. Excessive and sustained overshooting can also cause hot electron related effects which can compromise the reliability of the part.

Undershooting causes a reduction in the performance of the component, and may also lead to timing related failures. The processor will not function properly at its correct clock frequency.

Extensive die probing experiments show that high frequency overshooting and undershooting of the voltage specification are filtered by the processor's package parasitics, and are accounted for during the testing of the processor. As a result, the recommended oscilloscope measurement bandwidth has been adjusted to 20MHz (see section 4.2 for details).

3.0. THE POWER SUPPLY

Until traditional power supply units with 3.3V DC outputs are widely available, supplying power to the 3.3V Pentium processor requires a 5V-to-3.3V voltage regulator. In addition, robust local decoupling must be provided to accommodate the transition to and from low-power modes. It is important to select the components to be as accurate as possible. A platform based on an inaccurate power supply unit must be compensated with a more accurate regulator and extra local decoupling. Similarly, a platform based on an inaccurate regulator requires accurate supporting components and additional decoupling capacitors. As shown next, selecting accurate components will

maximize the voltage transient allowed.

The VRE specification allows a total voltage budget of 200mV. It is important to understand the voltage budget must include any deviation in the voltage regulator, the inaccuracy of its supporting components, and other non-ideal behavior of real components. When designing for a VRE-based platform, these DC factors must be subtracted from the total VRE budget. The remaining allowance should be targeted when measuring the voltage transient. It is hence important to select accurate voltage regulators and precise support components to allow maximum voltage transients.

VRE Specifications =	Σ Voltage Regulator Accuracy
+	Support Component Accuracy
+	Thermal Drift and Aging Effects
+	**Measured Voltage Transient**

3.1. Selecting an Accurate Power Supply Unit

The power supply unit must provide a minimum setpoint equal to, or higher than the minimum input voltage required by the regulator (see Figure 4). Off-the-shelf 5V power supply units with a 5% accuracy specification can meet the typical 4.75V requirement of most regulators. However, a 5V power supply unit with an accuracy of 10% may provide a setpoint as low as 4.5V and fail the minimum input requirement. Similarly, an accurate power supply unit may also fail if the voltage regulator has minimum input voltages higher than 4.75V. If using a less accurate power supply unit, the minimum setpoint must be raised to meet or exceed the minimum input voltage required by the voltage regulator. If a voltage regulator requires an input voltage higher than 4.75V, consider choosing a more accurate power supply unit to raise the minimum setpoint.

Sufficient decoupling must be provided between the power supply unit and the voltage regulator to minimize any noise. The disturbance on the 5V power supply unit may exceed the specification of TTL logic devices if the decoupling capacitance is insufficient.

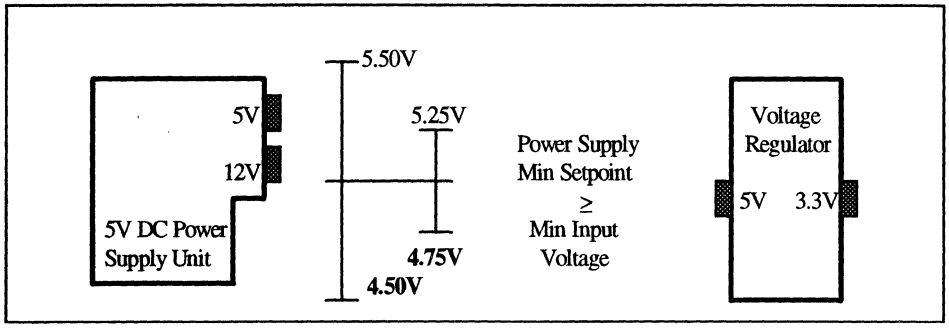


Figure 4. Setpoint Requirement of Power Supply

3.2. Selecting an Accurate Voltage Regulator

There are two types of voltage regulators: switching and linear. Switching regulators provide power by pulsing the voltages and currents to the load, thus resulting in lower heat dissipation and higher efficiency. Switching regulators are however generally more expensive and require more supporting components than linear regulators. Linear regulators essentially are voltage dividers and provide power by "dividing down" the 5V inputs to 3.3V outputs. Linear regulators dissipate more power, but are less expensive and typically require only two additional (feedback) resistors. Unless the system has strict thermal requirements, linear regulators generally are suited for high-volume designs. Both types of regulators can meet the VRE voltage range if they have accurate outputs and precise supporting components. Table 2 below compares the two types of voltage regulators:

An inaccurate regulator leaves little room for transient tolerance. For example, VRE specifications allow a voltage regulator solution to deviate only two percent (3.4V to 3.6V) from the desired regulator setpoint of 3.5V. Static specifications such as line regulation, temperature drift, and the initial setpoint must be held to 1% if any transient is to be permitted at all. Table 3 recommends the voltage regulator module accuracy required to ensure a robust VRE-based platform.

There are direct tradeoffs between the accuracy of the regulator and the amount of local decoupling. Using an inaccurate regulator requires more accurate dividing resistors and more decoupling. Conversely, using high-ESR, quick-aging capacitors necessitates accurate regulators. The next section recommends the bulk and high-speed decoupling required to ensure a robust VRE-based platform. The recommendations were based on extensive simulations and empirical measurements.

Table 2. Comparison of Voltage Regulators

Characteristics (Typical)	Linear Regulator	Switching Regulator
Maximum Efficiency	67%	95%
Maximum Power Dissipation	33%	5%
Supporting Components	2 to 6 (feedback resistors)	5 to 12 (feedback resistors, MOSFET switches, inductor, diode, caps)
Approximate Total Cost	Moderate	Moderately High

Table 3. Recommendations for Linear Voltage Regulator

Parameters	Total Accuracy	Maximum Deviation
Voltage Regulator Setpoint + Feedback Resistors + Thermal Drift, Aging Effects	± 1 % (VRE)	±35mV (VRE)
	± 2% (STD)	± 66mV (STD)

3.3. Bulk Decoupling Recommendations

The Pentium processor can be shut down and restarted very quickly with either the STPCLK# signal, or the HALT instruction. Switching the supply current on and off in very short time may cause serious power supply surges and droops in systems with inadequate bulk decoupling. Adequate bulk decoupling capacitors, located between the power and ground planes, near the processor, are necessary to filter these surges and droops. Adequate bulk capacitance is necessary to provide a current reservoir until the regulator can respond to the new load. It is important to use tantalum capacitors to minimize any aging effect. Electrolytic capacitors age faster, are inaccurate and are not stable over a wider temperature range. Capacitors with long leads add inductance and increase transients. Figure 5 shows the bulk decoupling required and a layout example to ensure the effectiveness of bulk decoupling.

Table 4 shows the decoupling recommendations for Socket 5, Socket 7 and standard 3.3V designs. Socket 7 is backward-compatible with Socket 5, and in addition it allows for upgradability to a future OverDrive processor. Note that the recommendations in Table 4 have already been optimized for cost efficiency.

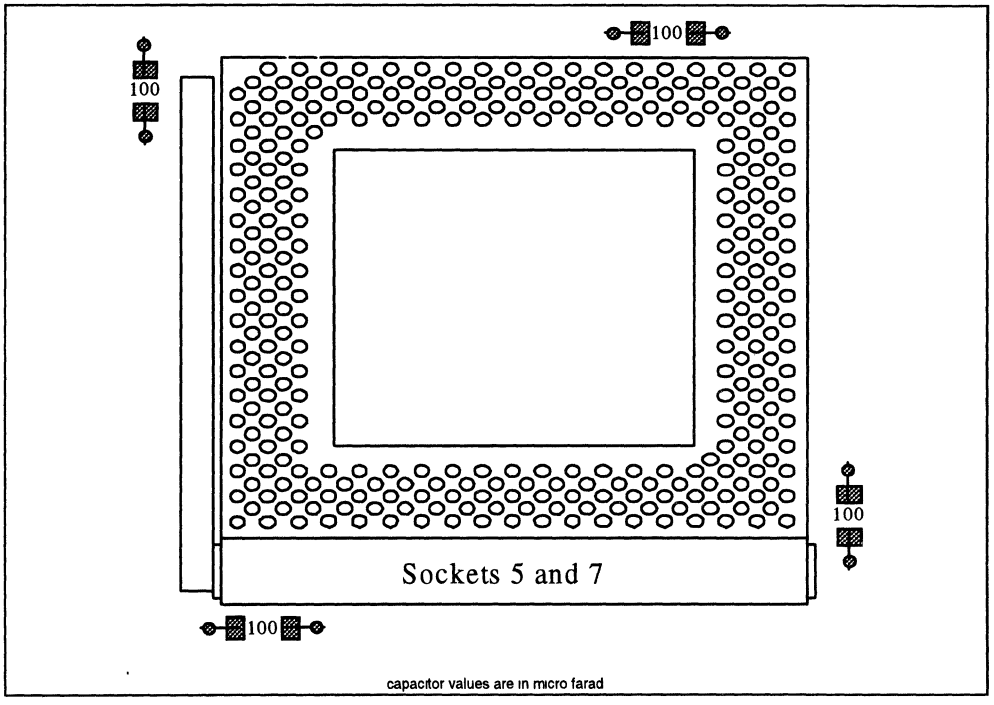


Figure 5. Recommended Bulk Decoupling Capacitor Values and Layout for VRE-based Design

3.4. High Speed Decoupling Recommendations

Due to its high speed core activity, the Pentium processor generates high frequency noise components and higher current spikes in the power supply. High frequency capacitors between the power and ground planes and near the processor, are required to filter these high frequency noise components. Since the inductive effects of circuit board traces and component leads become more critical at higher frequencies, it is critical to place high frequency capacitors as near as possible to the processor, using short traces to minimize inductance. Surface mount capacitors should be placed inside and around the socket cavity as shown in Figure 6.

The recommendations shown in Table 4 and Table 5 were based on extensive simulations and experiments. They provide a robust solution to accommodate various Pentium processors and Pentium Overdrive processors. During system design cycles, questions may arise about reducing cost by reducing the amount of decoupling, substituting with different capacitor dielectrics, and using less accurate resistors. Before committing to any deviations from the recommendations, it is highly recommended that the solution be simulated and certified for the variety in components, temperatures, and lifetime degradations. **The use of fewer, lower quality decoupling than indicated in this section is discouraged, even if voltage measurements indicate that the margin exists at 20MHz.**[1]

Table 4. Bulk Decoupling Recommendations for 3.3V Platforms

Design	Qty	Value	Type	Maximum ESR	Maximum ESL
Socket 5	4	100 µF	Tantalum	25 mΩ (100 mΩ/cap)	0.68 nH (2.7 nH/cap)
Socket 7	4	100 µF	Tantalum	25 mΩ (100 mΩ/cap)	0.68 nH (2.7 nH/cap)
Standard (low cost, non -VRE)	4	100 µF	Tantalum	25 mΩ (100 mΩ/cap)	0.68 nH (2.7 nH/cap)

Footnotes

[1] The product test environment assumes a certain minimum amount of decoupling.

Table 5. High Speed Decoupling Recommendations for 3.3V Platforms

Socket	Qty	Value	Type	Maximum ESR	Maximum ESL
Socket 5	18	1 µF	**X7R/X7S ceramic caps**	**0.83 mΩ** (15 mΩ/cap)	**0.117 nH** (2.1 nH/cap)
Socket 7	25	1 µF	**X7R/X7S ceramic caps**	**0.6 mΩ** (15 mΩ/cap)	**0.084 nH** (2.1 nH/cap)
Standard	12	1 µF	**X7R/X7S ceramic caps**	**1.25mΩ** (15 mΩ/cap)	**0.175 nH** (2.1 nH/cap)

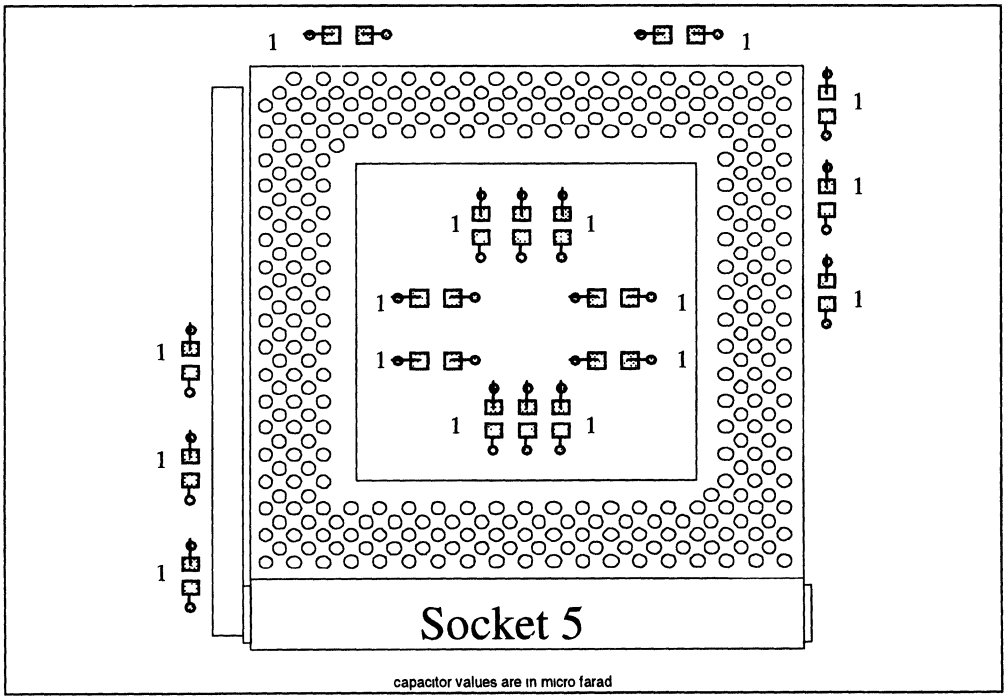


Figure 6. Recommended High Speed Decoupling Capacitors and Layout for VRE-based Unified-Plane Designs

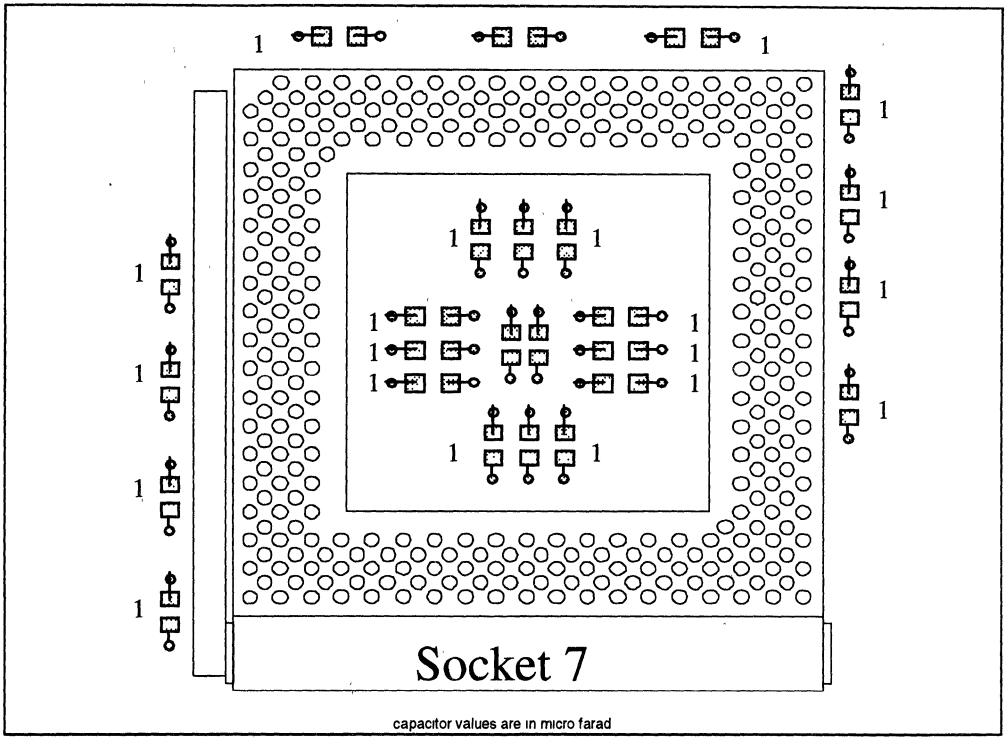


Figure 7. Recommended High Speed Decoupling Capacitors and Layout for VRE-based Unified-Plane Designs

3.5 ESR and ESL: Why Less is Better?

Effective Series Resistance (ESR) and Effective Series Inductance (ESL) are elements of non-ideal behavior of real components. The ESR and ESL determine how quickly a capacitor can source current to regulate a new load. More importantly, the ESR must be low enough at high frequencies to not offset the desired filtering effects of bulk decoupling capacitors. For a given current transient, the voltage transient is proportional to the ESL and ESR. The use of capacitors with high ESR and ESL hence contributes to higher voltage transients and may cause overshooting or undershooting. Aluminum electrolytic capacitors degrade at a relatively low frequency. Low ESR tantalum caps can retain ESR specifications up to about 1-10 MHz. Low ESR ceramic capacitors can retain ESR specifications up to 100MHz. Do not reduce the quantity of capacitors shown in Table 4 and Table 5 if substituting with capacitors with a larger value. When placed in parallel, two 220µF tantalum capacitors may have higher ESR than four 100 µF capacitors. Placing capacitors in parallel reduces the maximum overall ESR. The maximum overall ESR specifications listed in Table 4 are the same for capacitors with values of 100µF, 220µF, and 330µF.

3.6. Decoupling Recommendations for Split-Plane Designs

A split-plane design using Socket 7 should have all the decoupling capacitors recommended for Socket 7 (in Tables 4 and 5) placed on the core power plane. In addition 12 capacitors, each with a value of 0.1µF, should be used to decouple the I/O power plane.

4.0. TAKING VOLTAGE MEASUREMENTS

4.1. Creating Worst-Case Transient Excursion

The recommendations for regulators and local decoupling can be validated by creating worst-case supply transient conditions and measuring accurately as shown in Figure 8. Worst-case transients may be generated by executing the "stress" program on a Pentium processor test sample (refer to Appendix A for directions on obtaining test samples and the "stress" program). Table 6 explains the steps to create the worst-case transient conditions. Table 7 provides the measurement technique summary.

It is necessary to assert and deassert the STPCLK# signal while executing the "stress" program to create the worst-case transient conditions. Asserting the STPCLK# signal will place the processor into the Stop Grant mode (consuming about 15% of active current). Deasserting the STPCLK# signal will return the processor to the Normal state. To simulate actual system behavior, V_{cc} should be stabilized before asserting or deasserting STPCLK# as shown below. Asserting and deasserting STPCLK# too rapidly may generate unrealistic voltage transients. There are no minimum time specifications required to stabilize V_{cc} since the estimated time is highly dependent on the system (length of current instruction, outstanding write cycles, response time of voltage regulator, and accuracy and quantity of decoupling). However, based on experiments from the *Pentium® Processor Flexible Motherboard Reference Design*, STPCLK# should be asserted and deasserted at a rate of 10-100KHz. As part of their power saving features, certain BIOS are able to assert/deassert STPCLK# during execution of a batch file such as described in Table 6.

2

Table 6. Directions to Generate Worst-Case Transient

Step One	A	Install Pentium® processor
Step Two	B	Insert diskette containing the "stress" program
Step Three	C	Copy "STR4Y.EXE" to the C drive
Step Four	D	Create a batch file with an infinite loop that executes "STRY4.EXE" once in every loop. See Appendix A.
Step Five	E	Run the batch file created in step four.
Step Six	F	Measure V_{cc} and set oscilloscope as shown in Table 8 to obtain the voltage transient. The voltage transient must not overshoot 3.6V, or dip lower than 3.4V for VRE systems.
Step Seven	G	Assert and deassert STPCLK# while the "stress" program is executing.

Table 7. Measurement Technique Summary

Measurement Bandwidth[1]	≥ 20 MHz
Probe Bandwidth	≥ 250 MHz
Board Location	At the back of the board, at Socket pins
Pin Locations	12 Pins (listed above)

Note: [1] Signals should be attenuated by no more than 3dB at 20MHz, and 6dB at 40MHz.

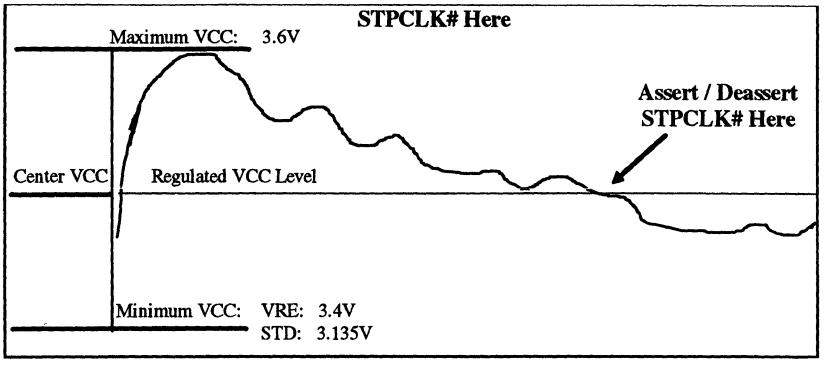


Figure 8. Method for Generating Worst-Case Transient via STPCLK#

4.2. Measurement Technique

All transient measurements must be taken at the back of the motherboard at the socket pins on an iPSL-certified Socket 5, Socket 7, or an equivalent socket of 5nH or less. Measuring transients on an unspecified socket or at a different location will result in inaccurate readings. For accurate readings, all probe connections must be clean. Shorten the ground lead of the probe to minimize any extra inductance. A specially-made probe as shown in Figure 9A will ensure accurate readings by connecting the probe tip directly to the V_{ss} signal and connecting four standoffs to the V_{ss} plane. Figure 9B proposes an alternate solution by providing a short loop of wire around the ground shield of the probe. Figure 9C is a good example of how not to perform measurements. The ground cable of the probe will add significant noise to the transient measurements.

The following V_{cc}/V_{ss} pairs should be measured, and must all meet the voltage specification: AN13/AM10, AN21/AM18, . AN29/AM26, AC37/Z36, U37/R36, L37/H36, A25/B28, A17/B20, A7/B10, G1/K2, S1/V2, AC1/Z2. These pins are a subset of all V_{cc}/V_{ss} pairs, and hence should not be singled out when placing decoupling capacitors.

The scope settings shown in Table 8 are recommended for accurate measurements. Although the measurement bandwidth of the scope should be set at 20MHz, a probe with a bandwidth of at least 250MHz should be used. This high bandwidth probe ensures a total effective bandwidth of 20MHz. The trigger point should be set in the middle of the range and slowly moved to both the high and low ends of the VRE range.

Table 8. Recommended Oscilloscope Configurations to Capture Voltage Transient

Bandwidth[1]	20 MHz[2]
Sampling Rate	≥ 100 Million Samples / Second
Vertical Reading	≤ 20 mV/division
Horizontal Reading	≥ 500 nS/division
Display	Infinite Persistence

Notes: [1] Signals should be attenuated by no more than 3dB at 20MHz, and 6dB at 40MHz.
[2] A probe with a bandwidth of at least 250mhz should be used.

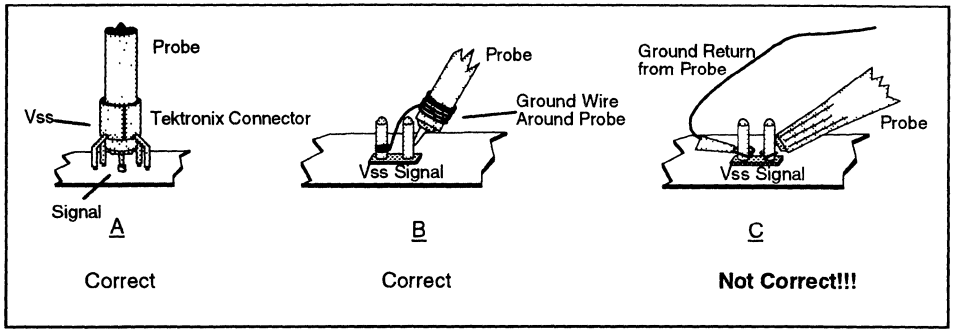


Figure 9. Correct and Incorrect Probe Connections for Measuring Transient

4.3. Measurement Results

The voltage transient measurement shown in Figure 10 was taken on the Pentium Processor Flexible Motherboard Reference Design (FMB) using the technique shown in Figure 9B. The Pentium processor FMB is an actual motherboard designed to accommodate various Pentium Processors and Pentium OverDrive processors, regardless of specifications. The Pentium processor FMB ensures accurate voltage regulation and proper decoupling through the Voltage Regulator Module (VRM), a small add-on module. To allow maximum flexibility, a variety of VRM models are available to accommodate all voltage specifications on the Pentium processors. For more detailed specifications of the Pentium processor FMB or VRM, please refer to the *Pentium® Processor Flexible Motherboard Design Guidelines*, Revision 2.0 (Reference Number SC-0990).

Figure 10 shows the Pentium processor exiting the Stop Grant Mode. Measurements were taken with a Tektronix TDS-684A oscilloscope and P6245 probe, while running the "stress" program with STPCLK# toggling to potentially create worst case transients. STPCLK# was also used to trigger the measurement. The platform used a VRE Spec VRM.

The VRE specification can tolerate voltage transients from 3.4V to 3.6V. The available tolerance of 200mV allows for voltage deviations due to transients and for VRM setpoint accuracy. VRM setpoint accuracy refers to the range in which the VRM maintains the output voltage (i.e. DC offset, noise, regulation tolerance including reference resistor tolerance under line and temperature variations). In this case, the setpoint accuracy is 70mV. The maximum voltage transients measured was about 58mV. This demonstrates that the Pentium processor flexible motherboard reference design meets the VRE voltage specification. It is important to note that the VRM and motherboard decoupling should allow for the main processor or Pentium OverDrive processor with the worst case current ramp.

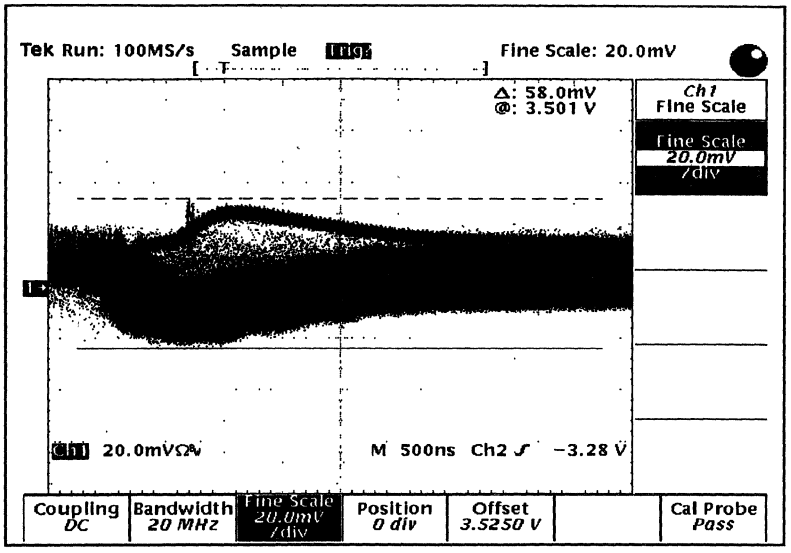


Figure 10. Transient Measurements with VRE Spec VRM

Appendix A: Test Samples and "Stress" Code

The "stress" program and may be obtained through a local Intel Field Sales representative or by calling Intel Technical Support at 1-800-628-8686. Intel Field Sales representatives may obtain the stress program from Sales Library Database (Technical Documents).

After the "stress" program has been obtained and installed on the C drive, a batch file should be written to run an infinite loop. The following is an example. The file is called **STRESS.BAT**:

```
:loop

STRY4.exe

echo on

goto loop
```

At the DOS* prompt, type STRESS and take measurements as explained in Chapter 4.0.

Appendix B: Third Party Components

The following vendors offer various solutions to ensure a robust VRE-based platform. Please contact the following vendors for specifications, samples, and design support.

Table 9. Voltage Regulator Modules

Vendor	North America	Europe	APAC	Japan
Linear Tech	Bob Scott Tel: (408) 432-1900 Fax: (408) 434-0507	Fred Killinger Tel: (49) 89-3197410 Fax: (49) 89-3194821	Dick Collins Tel: (65) 293-5322 Fax: (65) 292-0398	
Maxim/ Ambit[1]	David Timm (Maxim) Tel: (408) 737-7600 Fax: (408) 737-7194		Jacob Huang (Ambit) Tel: 886 35-784975 Fax: 886-35-775100	David Timm (Maxim) Tel: (408) 737-7600 Fax: (408) 737-7194
Power Trends[1]	Phil Lulewicz Tel: (708) 406-0900 Fax: (708) 406-0901		Joe Ywli Tel: (408) 737-7600 Fax: (408) 737-7194	Ken Katsumoto Tel: (81) 3-5367-9000 Fax: (81) 3-5467-0777
Semtech[1]	Art Fury Tel: (805) 498-2111 Fax: (805) 498-3804	Julian Foster Tel: (44) 592-630350 Fax: (44) 592-774781	Art Fury Tel: (805) 498-2111 Fax: (805) 498-3804	
Siliconix	Howard Chen Tel: (408) 970-4151 Fax: (408) 970-3910	Eric Williams Tel: (44) 344-485757 Fax: (44) 344-427371	Howard Chen Tel: (408) 970-4151 Fax: (408) 970-3910	Tony Grizelj Tel: (81) 3-5562-3321 Fax: (81) 3-5562-3316

Note: [1] DP VRM available

Table 10. Socket 7

Vendor	North America	Europe	APAC	Japan
Amp	Jim Crompton Tel: (910) 855-2338 Fax: (910) 855-2224	Rob Rix Tel: (44) 753-67-6892 Fax: (44) 753-67-6808	H. Itoh Tel: (81) 44-844-8086 Fax: (81) 44-812-3203	
Appros	Tony Goulart Tel: (415) 548-1636 Fax: (415) 548-1124		Appros Taiwan Inc. Tel: (886) 2-718-4774 Fax: (886) 2-718-4344	Appros Inc. Tel: (03) 3358-4857 Fax: (03) 3358-5734
Augat	David M. Barnum Tel: (508) 699-9890 Fax: (508) 695-8111	Arif Shahab Tel: (44) 952-670-281 Fax: (44) 952-670-342	Atsushi Sasaki Tel: (81) 44-853-5400 Fax: (81) 44-853-1113	
Foxconn	Julia Jang or Paul Fitting Tel: (408) 749-1228 Fax: (408) 749-1266		Ronny Chiou or Ivan Liaw Tel: (886) 2-268-3466 Fax: (886) 2-268-3225	
Yamaichi	Ann Sheperd Tel: (408) 456-0797 Fax: (408) 456-0779	Mr. Matsuda Tel: (49) 89-451021-43 Fax: (49) 89-451021-10	Alan Liu Tel: (886) 02-546-0507 Fax: (886) 02-546-0509	Mr. Shiwaku Tel: (81) 3-3778-6161 Fax: (81) 3-3778-6181

Table 11. Decoupling Capacitors

Vendor	Part No.	Type	North America	APAC
AVX	1206YZ105KAT1A	1μF, X7S	Dennis Lieberman Tel: (803) 946-0616 Fax: (803) 448-2606	Singapore Steve Chan Tel: (65) 258-2833 Fax: (65) 258-8221
	TPSD107K010R0100	100μF, Tantalum		Korea K.J. Kim Tel: (82) 2-785-6504 Fax: (82) 2-784-5411
Johanson Dielectrics	160R18W105K4	1μF, X7R	Sales Department Tel: (818) 364-9800 Fax: (818) 364-6100	Taiwan Nanco Electronics Bill Yu Tel: (886) 2-758-4650 Fax: (886) 2-729-4209 Hong Kong Tel: (852) 765-3029 Fax: (852) 330-2560
KEMET Electronics	T495X107K010AS	100μF, Tantalum	Richey-Cypress Electronics Tel: (408) 956-8010 Fax: (408) 956-8245	Internation Accounts Warren Marshall Tel: (800) 421-7258 Fax: (714) 895-0060
Murata Electronics	GRM40X7R105J016	1μF, X7R	Sales Department Tel: (404) 436-1300 Fax: (404) 436-3030	Taiwan Tel: (886) 2-562-4218 Fax: (886) 2-536-6721 Hong Kong Tel: (852) 782-2618 Fax: (852) 782-1545 Korea Tel: (82) 2-730-7605 Fax: (82) 2-739-5483
TDK	CC1206HX7R105K	1 μF, X7R/X7S	Sales Department Tel: (708) 803-6100 Fax: (708) 803-6296	Korea Tel: (82) 2-554-6633 Fax: (82) 2-712-6631 Taiwan Tel: (886) 2-712-5090 Fax: (886) 2-712-3090 Hong Kong Tel: (852) 736-2238 Fax: (852) 736-2108

Table 12. Header 7

Vendor	North America	Europe	APAC	Japan
Amp	Larry Freeland Tel: (717) 780-6045 Fax: (717) 780-7027	Rob Rix Tel: (44) 753-67-6892 Fax: (44) 753-67-6808	H. Itoh Tel: (81) 44-844-8086 Fax: (81) 44-812-3203	
Foxconn	Julia Jang or Paul Fitting Tel: (408) 749-1228 Fax: (408) 749-1266		Ronny Chiou or Ivan Liaw Tel: (886) 2-268-3466 Fax: (886) 2-268-3225	

Table 13. Shorting Blocks

Vendor	North America	Europe	APAC	Japan
Amp	Larry Freeland Tel: (717) 780-6045 Fax: (717) 780-7027	Rob Rix Tel: (44) 753-67-6892 Fax: (44) 753-67-6808	H. Itoh Tel: (81) 44-844-8086 Fax: (81) 44-812-3203	
Foxconn	Julia Jang or Paul Fitting Tel: (408) 749-1228 Fax: (408) 749-1266		Ronny Chiou or Ivan Liaw Tel: (886) 2-268-3466 Fax: (886) 2-268-3225	
Molex	Micheal Gits Tel: (708) 527-4801 Fax: (708) 969-1352	(Molex) Tel: (49) 89-413092-0 Fax: (49) 89-401527	(Molex) Tel: (65) 268-6868 Fax: (65) 265-6044	(Molex) Tel: (81) 427-21-5539 Fax: (81) 427-21-5562

Table 14. Resistors

Vendor	Size	Type	Accuracy/ Value	Contact
Beckman	0805	thin	0.1%, 10K-100K ohms	Cathy Whittaker
Industrial		thick	1-5%, 10-1M ohms	(214) 392-7616
	0603	thick	1-5%, 10-1M ohms	
Dale Electronic	0603	thin	0.5%, 10-100K ohms	Gary Bruns
		thick	1%,2%, 10-1M ohms	(402) 371-0080
	0805	thin	0.1%, 100-100K ohms	
Koa Spear	0805	thin	0.1%, 100-100K ohms	T. Yogi
		thick	0.5-5%, 10-1M ohms	(814) 362-5536
Thin Film Technology	1206	thin	0.1%, 100-250K ohms	Thin Film TECH. (607) 625 8445 Regional Sales Managers
			0.5%, 10-250K ohms	Patrick J Lyons ext. 14 All states W. of Mississippi except Texas and S. California
	0805	thin	0.1%, 100-100K ohms	
			0.5%, 10-1M ohms	Mark Porisch ext. 12 Southern U.S. E. of Mississippi including Texas
	0603	thin	0.1%, 100-33K ohms	
			0.5%, 10-330K ohms	Tim Goertzen ext. 13 Northen U.S. E. of Mississippi & Canada
	0402	thin	0.5%, 10-100K ohms	Mike Smith (310) 768-8923 Southern California

NORTH AMERICAN SALES OFFICES

ARIZONA

Intel Corp.
410 North 44th Street
Suite 470
Phoenix 85008
Tel: (800) 628-8686
FAX: (602) 244-0446

CALIFORNIA

Intel Corp.
26707 W. Agoura Road
Suite 203
Calabasas, CA 91302
Tel: (800) 628-8686
FAX: (818)-880-1820

Intel Corp.
3550 Watt Avenue
Suite 140
Sacramento 95821
Tel: (800) 628-8686
FAX: (916) 979-7011

Intel Corp.
9655 Granite Ridge Drive
3rd Floor
Suite 4A
San Diego 92123
Tel: (800) 628-8686
FAX: (619) 467-2460

Intel Corp.
1781 Fox Drive
San Jose 95131
Tel: (800) 628-8686
FAX: (408) 441-9540

Intel Corp.
1551 North Tustin Avenue
Suite 800
Santa Ana 92701
Tel: (800) 628-8686
TWX: (910) 595-1114
FAX: (714) 541-9157

Intel Corp.
514 Via de la Valle
Suite 208-RCO
Solana Beach 92075

Intel Corp.
1960 E. Grand Avenue
Suite 150
El Segundo, CA 90245
Tel: (800) 628-8686
FAX: (310) 640-7133

COLORADO

Intel Corp.
600 South Cherry Street
Suite 700
Denver 80222
Tel: (800) 628-8686
TWX: 910-931-2289
FAX: (303) 322-8670

CONNECTICUT

Intel Corp.
40 Old Ridgebury Road
Suite 311
Danbury 06811
Tel: (800) 628-8686
FAX: (203) 778-2168

FLORIDA

Intel Corp.
600 West Hillsboro Blvd.
Suite 348
Deerfield Beach 33441
Tel: (800) 628-8686
FAX: (305) 421-2444

Intel Corp.
2250 Lucien Way
Suite 100
Suite 8
Maitland 32751
Tel: (800) 628-8686
FAX: (407) 660-1283

GEORGIA

Intel Corp.
20 Technology Park
Suite 150
Norcross 30092
Tel: (800) 628-8686
FAX: (404) 448-0875

IDAHO

Intel Corp.
910 W. Main Street
Suite 236
Boise 83702
Tel: (800) 628-8686
FAX: (208) 331-2295

ILLINOIS

Intel Corp.
300 North Martingale Road
Suite 400
Schaumburg 60173
Tel: (800) 628-8686
FAX: (708) 605-9762

INDIANA

Intel Corp.
8041 Knue Road
Indianapolis 46250
Tel: (800) 628-8686
FAX: (317) 577-4939

MARYLAND

Intel Corp.
131 National Bus. Pkwy
Suite 200
Annapolis Junction 20701
Tel: (800) 628-8686
FAX: (301) 206-3678

MASSACHUSETTS

Intel Corp.
Nagog Park
125 Nagog Park
Acton 01720
Tel: (800) 628-8686
FAX: (508) 266-3867

MICHIGAN

Intel Corp.
32255 North Western Hwy.
Suite 212, Tri Atria
Farmington Hills 48334
Tel: (800) 628-8686
FAX: (313) 851-8770

MINNESOTA

Intel Corp.
3500 West 80th Street
Suite 360
Bloomington 55431
Tel: (800) 628-8686
TWX: 910-576-2867
FAX: (612) 831-6497

NEW JERSEY

Intel Corp.
2001 Route 46
Suite 310
Parsippany 07054
Tel: (800) 628-8686
FAX: (201) 402-4893

Intel Corp.
Lincroft Center
125 Half Mile Road
Red Bank 07701
Tel: (800) 628-8686
FAX: (908) 747-0983

NEW YORK

Intel Corp.
850 Cross Keys Office Pk
Fairport 14450
Tel: (800) 628-8686
TWX: 510-253-7391
FAX: (716) 223-2561

Intel Corp.
2950 Expressway Drive
Islandia 11722
Tel: (800) 628-8686
TWX: 510-227-6236
FAX: (516) 348-7939

OHIO

Intel Corp.
56 Milford Drive
Suite 205
Hudson 44236
Tel: (800) 628-8686
FAX: (216) 528-1026

***†Intel Corp.**
3401 Park Center Drive
Suite 220
Dayton 45414
Tel: (800) 628-8686
TWX: 810-450-2528
FAX: (513) 890-8658

OKLAHOMA

Intel Corp.
6801 North Broadway
Suite 115
Oklahoma City 73162
Tel: (800) 628-8686
FAX: (405) 840-9819

OREGON

Intel Corp.
15254 NW Greenbrier
Pkwy
Building B
Beaverton 97006
Tel: (800) 628-8686
TWX: 910-467-8741
FAX: (503) 645-8181

PENNSYLVANIA

Intel Corp.
925 Harvest Drive
Suite 200
Blue Bell 19422
Tel: (800) 628-8686
FAX: (215) 641-0785

SOUTH CAROLINA

Intel Corp.
7403 Parklane Road
Suite 4
Columbia 29223
Tel: (800) 628-8686
FAX: (803) 788-7999

Intel Corp.
100 Executive Center Dr
Suite 109, B183
Greenville 29615
Tel: (800) 628-8686
FAX: (803) 297-3401

TEXAS

Intel Corp.
8911 Capital of Texas Hwy
Suite 4230
Austin 78759
Tel: (800) 628-8686
FAX: (512) 338-9335

Intel Corp.
5000 Quorum Drive
Suite 750
Dallas 75240
Tel: (800) 628-8686
FAX: (214) 233-1325

Intel Corp.
20405 State Hwy 249
Suite 880
Houston 77070
Tel: (800) 628-8686
TWX: 910-881-2490
FAX: (713) 376-2891

UTAH

Intel Corp.
428 East 6400 South
Suite 135
Murray 84107
Tel: (800) 628-8686
FAX: (801) 268-1457

WASHINGTON

Intel Corp.
2800 156th Avenue SE
Suite 105
Bellevue 98007
Tel: (800) 628-8686
FAX: (206) 746-4495

WISCONSIN

Intel Corp.
400 North Executive Drive
Suite 401
Brookfield 53005
Tel: (800) 628-8686
FAX: (414) 789-2746

CANADA

BRITISH COLUMBIA

Intel of Canada, Ltd.
999 Canada Place
Suite 404
Suite 11
Vancouver V6C 3E2
Tel: (800) 628-8686
FAX: (604) 844-2813

ONTARIO

Intel of Canada, Ltd.
2650 Queensview Drive
Suite 250
Ottawa K2B 8H6
Tel: (800) 628-8686
FAX: (613) 820-5936

Intel of Canada, Ltd.
190 Attwell Drive
Suite 500
Rexdale M9W 6H8
Tel: (800) 628-8686
FAX: (416) 675-2438

QUEBEC

Intel of Canada, Ltd.
1 Rue Holiday, Tour West
Suite 320
Pt. Claire H9R 5N3
Tel: (800) 628-8686
FAX: 514-694-0064

NORTH AMERICAN DISTRIBUTORS

ALABAMA

Anthem Electronics
600 Boulevard South
Suite 104F & H
Huntsville 35802
Tel: (205) 890-0302

Arrow/Schweber Electronics
1015 Henderson Road
Huntsville 35805
Tel: (205) 837-6955
FAX: (205) 721-1581

Hall-Mark Computer
4890 University Square
Huntsville 35816
Tel: (800) 409-1483

Hamilton Hallmark
4890 University Square
Suite 1
Huntsville 35816
Tel: (205) 837-8706
FAX: (205) 830-2565

MTI Systems Sales
4950 Corporate Drive
Suite 120
Huntsville 35805
Tel: (205) 830-9526
FAX: (205) 830-9557

Pioneer Technologies Group
4835 University Square
Suite 5
Huntsville 35805
Tel: (205) 837-9300
FAX: (205) 837-9358

Wyle Electronics
7800 Governers Dr., W.
Tower Building, 2nd Floor
Huntsville 35807
Tel: (205) 830-1119
FAX: (205) 830-1520

ARIZONA

Alliance Electronics
7550 East Redfield Rd
Scottsdale 85260
Tel: (602) 261-7988

Anthem Electronics
1555 West 10th Place
Suite 101
Tempe 85281
Tel: (602) 966-6600
FAX: (602) 966-4826

Arrow/Schweber Electronics
2415 West Erie Drive
Tempe 85282
Tel: (602) 431-0030
FAX: (602) 431-9555

Avnet Computer
1626 South Edwards Dr
Tempe 85281
Tel: (800) 426-7999

Hall-Mark Computer
4637 South 37th Place
Phoenix 85040
Tel: (800) 409-1483

Pioneer Standard
1438 West Broadway
Suite B-140
Tempe 85282
Tel: (602) 350-9335

Hamilton Hallmark
4637 South 36th Place
Phoenix 85040
Tel: (602) 437-1200
FAX: (602) 437-2348

Wyle Electronics
4141 East Raymond
Phoenix 85040
Tel: (602) 437-2088
FAX: (602) 437-2124

CALIFORNIA

Anthem Electronics
9131 Oakdale Avenue
Chatsworth 91311
Tel: (818) 775-1333
FAX: (818) 775-1302

Anthem Electronics
1 Oldfield Drive
Irvine 92718-2809
Tel: (714) 768-4444
FAX: (714) 768-6456

Anthem Electronics
580 Menlo Drive
Suite 8
Rocklin 95677
Tel: (916) 624-9744
FAX: (916) 624-9750

Anthem Electronics
9369 Carroll Park Drive
San Diego 92121
Tel: (619) 453-9005
FAX: (619) 546-7893

Anthem Electronics
1160 Ridder Park Drive
San Jose 95131
Tel: (408) 453-1200
FAX: (408) 441-4504

Arrow/Schweber Electronics
26707 West Agoura Road
Calabasas 91302
Tel: (818) 880-9686
FAX: (818) 772-8930

Arrow/Schweber Electronics
48834 Kato Road
Suite 103
Fremont 94538
Tel: (510) 490-9477
FAX: (510) 490-1084

Arrow/Schweber Electronics
6 Cromwell
Suite 100
Irvine 92718
Tel: (714) 581-4622
FAX: (714) 454-4206

Arrow/Schweber Electronics
9511 Ridgehaven Court
San Diego 92123
Tel: (619) 565-4800
FAX: (619) 279-8062

Arrow/Schweber Electronics
1180 Murphy Avenue
San Jose 95131
Tel: (408) 441-9700
FAX: (408) 453-4810

Avnet Computer
1 Mauchley
Irvine 92718
Tel: (800) 426-7999

Avnet Computer
371 Van Ness Way
Torrance 90501
Tel: (800) 426-7999

Avnet Computer
15950 Bernardo Ctr Dr
Suite 6
San Diego 92127
Tel: (800) 426-7999

Avnet Computer
1175 Bordeaux Drive
Suite A
Sunnyvale 94089
Tel: (800) 426-7999

Hall-Mark Computer
21150 Califa Street
Woodland Hills 91367
Tel: (800) 409-1483

Hall-Mark Computer
15950 Bernardo Ctr Dr
Suite C
San Diego 92127
Tel: (800) 409-1483

Hall-Mark Computer
1175 Bordeaux Drive
Sunnyvale 94089
Tel: (800) 409-1483

Hall-Mark Computer
1 Mauchly
Irvine 92718
Tel: (800) 409-1483

Hall-Mark Computer
580 Menlo Drive
Suite 2
Rocklin 95765
Tel: (800) 409-1483

Hamilton Hallmark
3170 Pullman Street
Costa Mesa 92626
Tel: (714) 641-4100
FAX: (714) 641-4122

Hamilton Hallmark
2105 Lundy Avenue
San Jose 95131
Tel: (408) 435-3500
FAX: (408) 435-3720

Hamilton Hallmark
4545 Viewridge Avenue
San Diego 92123
Tel: (619) 571-7540
FAX: (619) 277-6136

Hamilton Hallmark
21150 Califa Street
Woodland Hills 91367
Tel: (818) 594-0404
FAX: (818) 594-8234

Hamilton Hallmark
580 Menlo Drive
Suite 2
Rocklin 95762
Tel: (916) 624-9781
FAX: (916) 961-0922

Pioneer Standard
5126 Clareton Drive
Suite 106
Agoura Hills 91301
Tel: (818) 865-5800

Pioneer Standard
217 Technology Drive
Suite 110
Irvine 92718
Tel: (714) 753-5090

Pioneer Technologies Group
134 Rio Robles
San Jose 95134
Tel: (408) 954-9100
FAX: (408) 954-9113

Pioneer Standard
4370 La Jolla Village Drive
San Diego 92122
Tel: (619) 546-4906

Wyle Electronics
15370 Barranca Pkwy
Irvine 92713
Tel: (714) 753-9953
FAX: (714) 753-9877

Wyle Electronics
15360 Barranca Pkwy
Suite 200
Irvine 92713
Tel: (714) 753-9953
FAX: (714) 753-9877

Wyle Electronics
2951 Sunrise Blvd.
Suite 175
Rancho Cordova 95742
Tel: (916) 638-5282
FAX: (916) 638-1491

Wyle Electronics
9525 Chesapeake Dr.
San Diego 92123
Tel: (619) 565-9171
FAX: (619) 365-0512

Wyle Electronics
3000 Bowers Avenue
Santa Clara 95051
Tel: (408) 727-2500
FAX: (408) 727-5896

Wyle Electronics
17872 Cowan Avenue
Irvine 92714
Tel: (714) 863-9953
FAX: (714) 263-0473

Wyle Electronics
26010 Mureau Road
Suite 150
Calabasas 91302
Tel: (818) 880-9000
FAX: (818) 880-5510

Zeus Arrow Electronics
6276 San Ignacio Avenue
Suite E
San Jose 95119
Tel: (408) 629-4689
FAX: (408) 629-4792

Zeus Arrow Electronics
6 Cromwell Street
Suite 100
Irvine 92718
Tel: (714) 581-4622
FAX: (714) 454-4355

COLORADO

Anthem Electronics
373 Inverness Dr. S.
Englewood 80112
Tel: (303) 790-4500
FAX: (303) 790-4532

Arrow/Schweber Electronics
61 Inverness Dr East
Suite 105
Englewood 80112
Tel: (303) 799-0258
FAX: (303) 799-0730

Avnet Computer
9605 Maroon Circle
Englewood 80111
Tel: (800) 426-7999

Hall-Mark Computer
9605 Maroon Circle
Englewood 80111
Tel: (800) 409-1483

Hamilton Hallmark
12503 East Euclid Dr
Suite 20
Englewood 80111
Tel: (303) 790-1662
FAX: (303) 790-4991

Hamilton Hallmark
710 Wooten Road
Suite 28
Colorado Springs 80915
Tel: (719) 637-0055
FAX: (719) 637-0088

Pioneer Technologies
5600 Greenwood Plaza Blvd.
Suite 201
Englewood 80111
Tel: (303) 773-8090

Wyle Electronics
451 East 124th Avenue
Thornton 80241
Tel: (303) 457-9953
FAX: (303) 457-4831

CONNECTICUT

Anthem Electronics
61 Mattatuck Heights Road
Waterburg 06705
Tel: (203) 575-1575
FAX: (203) 596-3232

Arrow/Schweber Electronics
860 N. Main St. Ext.
Wallingford 06492
Tel: (203) 265-7741
FAX: (203) 265-7988

Hall-Mark Computer
Still River Corporate Ctr
55 Federal Road
Danbury 06810
Tel: (800) 409-1483

Hamilton Hallmark
125 Commerce Court, Unit 6
Cheshire 06410
Tel: (203) 271-2844
FAX: (203) 272-1704

Pioneer Standard
2 Trap Falls Road
Shelton 06484
Tel: (203) 929-5600

FLORIDA

Anthem Electronics
5200 NW 3rd Avenue
Suite 206
Ft. Lauderdale 33309
Tel: (305) 484-0990

Anthem Electronics
598 S. Northlake Blvd.
Suite 1024
Altamonte Sprgs 32701
Tel: (813) 797-2900
FAX: (813) 796-4880

Arrow/Schweber Electronics
400 Fairway Drive
Suite 102
Deerfield Beach 33441
Tel: (305) 429-8200
FAX: (305) 428-3991

Arrow/Schweber Electronics
37 Skyline Drive
Suite 3101
Lake Mary 32746
Tel: (407) 333-9300
FAX: (407) 333-9320

Arrow/Schweber Electronics
4010 Boy Scout Dr.
Suite 295
Tampa 33607
Tel: (813) 873-1030
FAX: (813) 873-0077

Avnet Computer
541 S. Orlando Ave.
Suite 203
Maitland 32751
Tel: (800) 426-7999

Hall-Mark Computer
10491 72nd St. North
Largo 34647
Tel: (800) 409-1483

Hall-Mark Computer
13700 58th St. North
Suite 206
Clearwater 34620
Tel: (800) 409-1483

Hamilton Hallmark
3350 N.W. 53rd Street
Suite 105-107
Ft. Lauderdale 33309
Tel: (305) 484-5482
FAX: (305) 484-2995

NORTH AMERICAN DISTRIBUTORS (Cont'd)

Hamilton Hallmark
10491 72nd St. North
Largo 34647
Tel: (813) 541-7440
FAX: (813) 544-4394

Hamilton Hallmark
7079 University Blvd.
Winter Park 32792
Tel: (407) 657-3300
FAX: (407) 678-4414

Pioneer Technologies Group
337 Northlake Blvd
Suite 1000
Alta Monte Spgs 32701
Tel: (407) 834-9090
FAX: (407) 834-0865

Pioneer Technologies Group
674 South Military Trail
Deerfield Beach 33442
Tel: (305) 428-8877
FAX: (305) 481-2950

Wyle Electronics
1000 112th Circle North
St. Petersburg 33716
Suite 800
Tel: (813) 579-1518
FAX: (813) 579-1518

Zeus Arrow Electronics
37 Skyline Drive
Bldg D., Suite 3101
Lake Mary 32746
Suite 800
Tel: (407) 333-3055
FAX: (407) 333-9681

GEORGIA

Anthem Electronics
2400 Pleasant Hill Rd
Suites 9 & 10
Duluth 30136
Tel: (404) 931-3900
FAX: (404) 931-3902

Arrow/Schweber Electronics
4250 E Rivergreen Pkwy
Suite E
Duluth 30136
Tel: (404) 497-1300
FAX: (404) 476-1493

Avnet Computer
3425 Corporate Way
Suite G
Duluth 30136
Tel: (800) 426-7999

Hall-Mark Computer
3425 Corporate Way
Suite G
Duluth 30136
Tel: (800) 409-1483

Hamilton Hallmark
3425 Corporate Way
Suite G & A
Duluth 30136
Tel: (404) 623-5475
FAX: (404) 623-5490

Pioneer Technologies Group
4250C Rivergreen Pkwy
Duluth 30136
Tel: (404) 623-1003
FAX: (404) 623-0665

Wyle Electronics
6025 The Corners Pkwy
Suite 111
Norcross 30092
Tel: (404) 441-9045
FAX: (404) 441-9086

ILLINOIS

Anthem Electronics
1300 Remington Road
Suite A
Schaumberg 60173
Tel: (708) 884-0200
FAX: (708) 885-0480

Arrow/Schweber Electronics
1140 W Thorndale Rd
Itasca 60143
Tel: (708) 250-0500

Avnet Computer
1124 Thorndale Ave
Bensenville 60106
Tel: (800) 426-7999

Hall-Mark Computer
1124 Thorndale Ave
Bensenville 60106
Tel: (800) 409-1483

Hamilton Hallmark
1130 Thorndale Ave
Bensenville 60106
Tel: (800) 426-7999

MTI Systems Sales
1140 West Thorndale Avenue
Itasca 60143
Tel: (708) 250-8222
FAX: (708) 250-8275

Pioneer Standard
2171 Executive Drive
Suite 200
Addison 60101
Tel: (708) 495-9680
FAX: (708) 495-9831

Wyle Electronics
2055 Army Trail Road
Suite 140
Addison 60101
Tel: (800) 853-9953

Zeus Arrow Electronics
1140 W Thorndale Ave
Itasca 60143
Tel: (708) 250-0500

INDIANA

Arrow/Schweber Electronics
7108 Lakeview
Parkway West Drive
Indianapolis 46268
Tel: (317) 299-2071
FAX: (317) 299-2379

Avnet Computer
655 West Carmel Drive
Suite 160
Carmel 46032
Tel: (800) 426-7999

Hall-Mark Computer
655 West Carmel Drive
Carmel 46032
Tel: (800) 409-1483

Hamilton Hallmark
655 West Carmel Drive
Suite 160
Carmel 46032
Tel: (317) 575-3500
FAX: (317) 575-3535

Pioneer Standard
9350 Priority Way W Dr
Indianapolis 46250
Tel: (317) 573-0880
FAX: (317) 573-0979

KANSAS

Arrow/Schweber Electronics
9801 Legler Road
Lenexa 66219
Tel: (913) 541-9542
FAX: (913) 541-0328

Hall-Mark Computer
10809 Lakeview Ave
Lenexa 66219
Tel: (800) 409-1483

Hamilton Hallmark
10809 Lakeview Avenue
Lenexa 66215
Tel: (913) 888-4747
FAX: (913) 888-0523

MARYLAND

Anthem Electronics
7168A Columbia
Gateway Drive
Columbia 21046
Tel: (800) 239-6039

Arrow/Schweber Electronics
9800J Patuxent Woods Drive
Columbia 21046
Tel: (301) 596-7800
FAX: (301) 596-7821

Avnet Computer
7172 Columbia
Gateway Drive
Suite G
Columbia 21045
Tel: (800) 426-7999

Hall-Mark Computer
7172 Columbia
Gateway Drive
Suite G
Columbia 21046
Tel: (800) 409-1483

Hamilton Hallmark
10240 Old Columbia Road
Columbia 21046
Tel: (410) 988-9800
FAX: (410) 381-2036

North Atlantic Industries
Systems Division
7125 River Wood Drive
Columbia 21046
Tel: (301) 312-5800
FAX: (301) 312-5850

Pioneer Technologies Group
15810 Gaither Road
Gaithersburg 20877
Tel: (301) 921-0660
FAX: (301) 670-6746

Wyle Electronics
9101 Guilford Road
Suite 120
Columbia 21046
Tel: (301) 490-2170
FAX: (301) 490-2190

MASSACHUSETTS

Anthem Electronics
200 Research Drive
Wilmington 01887
Tel: (508) 657-5170
FAX: (508) 657-6008

Arrow/Schweber Electronics
25 Upton Drive
Wilmington 01887
Tel: (508) 658-0900
FAX: (508) 694-1754

Avnet Computer
10 D Centennial Drive
Peabody 01960
Tel: (800) 426-7999

Hall-Mark Computer
10 D Centennial Drive
Peabody 01960
Tel: (800) 409-1483

Hamilton Hallmark
10 D Centennial Drive
Peabody 01960
Tel: (508) 531-7430
FAX: (508) 532-9802

Pioneer Standard
44 Hartwell Avenue
Lexington 02173
Tel: (617) 861-9200
FAX: (617) 863-1547

Wyle Electronics
5 Oak Park Drive
Bedford 01803
Tel: (617) 271-9953
FAX: (617) 275-3809

Zeus Arrow Electronics
25 Upton Drive
Wilmington 01887
Tel: (508) 658-4776
FAX: (508) 694-2199

MICHIGAN

Arrow/Schweber Electronics
44720 Helm Street
Plymouth 48170
Tel: (313) 462-2290
FAX: (313) 462-2686

Avnet Computer
41650 Garden Brk Rd
Suite 120
Novi 48375
Tel: (800) 426-7999

Hall-Mark Computer
41650 Garden Brk Rd
Suite 120
Novi 48375
Tel: (800) 409-1483

Hamilton Hallmark
44191 Plymouth Oaks Blvd.
Suite 1300
Plymouth 48170
Tel: (313) 416-5806
FAX: (313) 416-5811

Hamilton Hallmark
41650 Garden Brk Rd
Suite 100
Novi 49418
Tel: (313) 347-4271
FAX: (313) 347-4021

Pioneer Standard
4505 Broadmoor S.E.
Grand Rapids 49512
Tel: (616) 698-1800
FAX: (616) 698-1831

Pioneer Standard
44190 Plymouth Oaks Blvd.
Plymouth 48170
Tel: (313) 525-1800
FAX: (313) 427-3720

MINNESOTA

Anthem Electronics
7646 Golden Triangle Drive
Eden Prairie 55344
Tel: (612) 944-5454
FAX: (612) 944-3045

Arrow/Schweber Electronics
10100 Viking Drive
Suite 100
Eden Prairie 55344
Tel: (612) 941-5280
FAX: (612) 942-7803

Avnet Computer
9800 Bren Roaqd East
Suite 410
Minnetonka 55343
Tel: (800) 426-7999

Hall-Mark Computer
9800 Bren Road East
Suite 410
Minnetonka 55343
Tel: (800) 409-1483

Hamilton Hallmark
9401 James Ave South
Suite 140
Bloomington 55431
Tel: (612) 881-2600
FAX: (612) 881-9461

Pioneer Standard
7625 Golden Triangle Drive
Suite G
Eden Prairie 55344
Tel: (612) 944-3355
FAX: (612) 944-3794

Wyle Electronics
1325 East 79th Street
Suite 1
Bloomington 55425
Tel: (612) 853-2280
FAX: (612) 853-2298

MISSOURI

Arrow/Schweber Electronics
2380 Schuetz Road
St. Louis 63141
Tel: (314) 567-6888
FAX: (314) 567-1164

Avnet Computer
3783 Rider Train South
Earth City 63045
Tel: (800) 426-7999

Hall-Mark Computer
3783 Rider Trail South
Earth City 63045
Tel: (800) 409-1483

Hamilton Hallmark
3783 Rider Trail South
Earth City 63045
Tel: (314) 291-5350
FAX: (314) 291-0362

NEW HAMPSHIRE

Avnet Computer
2 Executive Park Drive
Bedford 03102
Tel: (800) 426-7999

NEW JERSEY

Anthem Electronics
26 Chapin Road, Unit K
Pine Brook 07058
Tel: (201) 227-7960
FAX: (201) 227-9246

Arrow/Schweber Electronics
4 East Stow Road
Unit 11
Marlton 08053
Tel: (609) 596-8000
FAX: (609) 596-9632

Arrow/Schweber Electronics
43 Route 46 East
Pine Brook 07058
Tel: (201) 227-7880
FAX: (201) 227-2064

Avnet Computer
1-B Keystone Avenue
Building 36
Cherry Hill 08003
Tel: (800) 426-7999

Hall-Mark Computer
1-B Keystone Avenue
Building 36
Cherry Hill 08003
Tel: (800) 409-1483

Hall-Mark Computer
10 Lanidex Plaza West
Parsippany 07054
Tel: (800) 409-1483

Hamilton Hallmark
1 Keystone Avenue
Building 36
Cherry Hill 08003
Tel: (609) 424-0110
FAX: (609) 751-2552

Hamilton Hallmark
10 Lanidex Plaza West
Parsippany 07054
Tel: (201) 515-5300
FAX: (201) 515-1601

MTI Systems Sales
43 Route 46 East
Pinebrook 07058
Tel: (201) 882-8780
FAX: (201) 539-6430

NORTH AMERICAN DISTRIBUTORS (Cont'd)

PioneerStandard
14-A Madison Road
Fairfield 07006
Tel: (201) 575-3510
FAX: (201) 575-3454

Wyle Electronics
115 Route 46, Bldg F
Mountain Lakes 07046
Tel: (201) 402-4970

NEW MEXICO

Alliance Electronics, Inc.
3411 Bryn Mawr N.E.
Albuquerque 87101
Tel: (505) 292-3360
FAX: (505) 275-6392

Avnet Computer
7801 Academy Road
Building 1, Suite 204
Albuquerque 87109
Tel: (800) 426-7999

NEW YORK

Anthem Electronics
47 Mall Drive
Commack 11725
Tel: (516) 864-6600
FAX: (516) 493-2244

Arrow/Schweber Electronics
3375 Brighton Henrietta Townline Road
Rochester 14623
Tel: (716) 427-0300
FAX: (716) 427-0735

Arrow/Schweber Electronics
20 Oser Avenue
Hauppauge 11788
Tel: (516) 231-1000
FAX: (516) 231-1072

Avnet Computer
2 Penn Plaza
Suite 1245
New York 10121
Tel: (800) 426-7999

Avnet Computer
1057 E. Henrietta Road
Rochester 14623
Tel: (800) 426-7999

Hall-Mark Computer
2 Penn Plaza
New York 10121
Tel: (800) 409-1483

Hall-Mark Computer
1057 E Henrietta Road
Rochester 14623
Tel: (800) 409-1483

Hamilton Hallmark
933 Motor Parkway
Hauppauge 11788
Tel: (516) 434-7470
FAX: (516) 434-7491

Hamilton Hallmark
1057 E Henrietta Road
Rochester 14623
Tel: (716) 475-9130
FAX: (716) 475-9119

Hamilton Hallmark
3075 Veterans Memorial Hwy.
Ronkonkoma 11779
Tel: (516) 737-0600
FAX: (516) 737-0838

MTI Systems Sales
1 Penn Plaza
250 West 34th Street
New York 10119
Tel: (212) 643-1280
FAX: (212) 643-1288

Pioneer Standard
68 Corporate Drive
Binghamton 13904
Tel: (607) 722-9300
FAX: (607) 722-9562

Pioneer Standard
60 Crossway Pk West
Woodbury, Long Island
11797
Tel: (516) 921-8700
FAX: (516) 921-2143

Pioneer Standard
840 Fairport Park
Fairport 14450
Tel: (716) 381-7070
FAX: (716) 381-5955

Zeus Arrow Electronics
100 Midland Avenue
Port Chester 10573
Tel: (914) 937-7400
FAX: (914) 937-2553

NORTH CAROLINA

Anthem Electronics
4805 Greenwood
Suite 100
Raleigh 27604
Tel: (919) 782-3550

Arrow/Schweber Electronics
5240 Greensdairy Road
Raleigh 27604
Tel: (919) 876-3132
FAX: (919) 878-9517

Avnet Computer
4421 Stuart Andrew Boulevard
Suite 600
Charlotte 28217
Tel: (800) 426-7999

Hall-Mark Computer
3510 Spring Forest Rd
Suite B
Raleigh 27604
Tel: (800) 409-1483

Hamilton Hallmark
3510 Spring Forest Rd
Suite B
Raleigh 27604
Tel: (800) 409-1483

Hamilton Hallmark
5234 Greens Dairy Rd
Raleigh 27604
Tel: (919) 878-0819

Pioneer Technologies Group
2200 Gateway Ctr. Blvd
Suite 215
Morrisville 27560
Tel: (919) 460-1530

OHIO

Arrow/Schweber Electronics
6573 Cochran Road
Suite E
Solon 44139
Tel: (216) 248-3990
FAX: (216) 248-1106

Arrow/Schweber Electronics
8200 Washington Village Drive
Centerville 45458
Tel: (513) 435-5563
FAX: (513) 435-2049

Avnet Computer
7764 Washington Village Drive
Dayton 45459
Tel: (800) 426-7999

Avnet Computer
2 Summit Park Drive
Suite 520
Independence 44131
Tel: (800) 426-7999

Hall-Mark Computer
5821 Harper Road
Solon 44139
Tel: (800) 409-1483

Hall-Mark Computer
777 Dearborn Pk Lane
Suite L
Worthington 43085
Tel: (800) 409-1483

Hamilton Hallmark
5821 Harper Road
Solon 44139
Tel: (216) 498-1100
FAX: (216) 248-4803

Hamilton Hallmark
777 Dearborn Pk Lane
Suite L
Worthington 43085
Tel: (614) 888-3313
FAX: (614) 888-0767

MTI Systems Sales
23404 Commerce Pk Road
Beachwood 44122
Tel: (216) 464-6688
FAX: (216) 464-3564

Pioneer Standard
4433 Interpoint Blvd
Dayton 45424
Tel: (513) 236-9900
FAX: (513) 236-8133

Pioneer Standard
4800 East 131st Street
Cleveland 44105
Tel: (216) 587-3600
FAX: (216) 663-1004

Wyle Electronics
6835 Cochran Rd.
Solon 44139
Tel: (216) 248-9996

OKLAHOMA

Arrow/Schweber Electronics
12101 East 51st Street
Suite 106
Tulsa 74146
Tel: (918) 252-7537
FAX: (918) 254-0917

Hamilton Hallmark
5411 S. 125th E. Ave
Suite 305
Tulsa 74146
Tel: (918) 254-6110
FAX: (918) 254-6207

Pioneer Standard
9717 East 42nd Street
Suite 105
Tulsa 74146
Tel: (918) 665-7840
FAX: (918) 665-1891

OREGON

Almac Arrow Electronics
9500 S.W. Nimbus Ave
Suite E
Beaverton 97008
Tel: (503) 629-8090
FAX: (503) 645-0611

Anthem Electronics
9090 SW Gemini Drive
Beaverton 97005
Tel: (503) 643-1114
FAX: (503) 626-7928

Avnet Computer
9750 SW Nimbus Ave.
Beaverton 97005
Tel: (800) 426-7999

Hall-Mark Computer
9750 SW Nimbus Ave.
Beaverton 97005
Tel: (800) 409-1483

Hamilton Hallmark
9750 SW Nimbus Ave.
Beaverton 97005
Tel: (503) 526-6200
FAX: (503) 641-5939

Pioneer Technologies
8905 Southwest Nimbus Ave.
Suite 160
Beaverton 97005
Tel: (503) 626-7300
FAX: (503) 626-5300

Wyle Electronics
9640 Sunshine Court
Building G
Suite 200
Beaverton 97005
Tel: (503) 643-7900
FAX: (503) 646-5466

PENNSYLVANIA

Anthem Electronics
355 Business Ctr Drive
Horsham 19044
Tel: (215) 443-5150
FAX: (215) 675-9875

Avnet Computer
213 Executive Drive
Suite 320
Mars 16046
Tel: (800) 426-7999

Arrow/Schweber Electronics
2681 Mosside Blvd
Suite 204
Monroeville 15146
Tel: (412) 856-9490

Pioneer Technologies Group
259 Kappa Drive
Pittsburgh 15238
Tel: (412) 782-2300
FAX: (412) 963-8255

Pioneer Technologies Group
500 Enterprise Road
Keith Valley Bus.Ctr
Horsham 19044
Tel: (215) 674-4000

Wyle Electronics
1 Eves Drive
Suite 111
Marlton 08053-3185
Tel: (609) 985-7953
FAX: (609) 985-8757

TEXAS

Anthem Electronics
651 N. Plano Road
Suite 401
Richardson 75081
Tel: (214) 238-7100
FAX: (214) 238-0237

Anthem Electronics
14050 Summit Drive
Suite 119
Tel: (512) 388-0049
FAX: (512) 388-0271

Arrow/Schweber Electronics
Brake Ctr III, Bldg M1
11500 Metric Boulevard
Suite 160
Austin 78758
Tel: (512) 835-4180
FAX: (512) 832-5921

Arrow/Schweber Electronics
3220 Commander Drive
Carrollton 75006
Tel: (214) 380-6464
FAX: (214) 248-7208

Arrow/Schweber Electronics
19416 Park Row
Suite 190
Houston 77084
Tel: (713) 647-6868
FAX: (713) 492-8722

Avnet Computer
4004 Beltline
Suite 200
Dallas 75244
Tel: (800) 426-799

Avnet Computer
1235 North Loop West
Suite 525
Houston 77008
Tel: (800) 426-7999

Hall-Mark Computer
12211 Technology Blvd
Austin 78727
Tel: (800) 409-1483

Hall-Mark Computer
4004 Beltline Road
Suite 200
Dallas 75244
Tel: (800) 409-1483

Hall-Mark Computer
1235 North Loop West
Houston 77008
Tel: (800) 409-1483

Hamilton Hallmark
12211 Technology Boulevard
Austin 78727
Tel: (512) 258-8848
FAX: (512) 258-3777

Hamilton Hallmark
11420 Page Mill Road
Dallas 75243
Tel: (214) 553-4300
FAX: (214) 553-4395

Hamilton Hallmark
8000 Westglen
Houston 77063
Tel: (713) 781-6100
FAX: (713) 953-8420

Pioneer Standard
1826D Kramer Lane
Austin 78758
Tel: (512) 835-4000
FAX: (512) 835-9829

Pioneer Standard
13765 Beta Road
Dallas 75244
Tel: (214) 263-3168
FAX: (214) 490-6419

Pioneer Standard
10530 Rockley Road
Suite 100
Houston 77099
Tel: (713) 495-4700
FAX: (713) 495-5642

Wyle Electronics
1810 Greenville Ave
Richardson 75081
Tel: (214) 235-9953
FAX: (214) 644-5064

Wyle Electronics
9208 Waterford Center Blvd
Suite 150
Austin 78750
Tel: (512) 345-8853
FAX: (512) 345-9330

Wyle Electronics
2901 Wilcrest
Suite 120
Houston 77099
Tel: (713) 879-9953
FAX: (713) 879-9953

Zeus Arrow Electronics
3220 Commander Dr
Carrollton 75006
Tel: (214) 380-4330
FAX: (214) 447-2222

UTAH

Anthem Electronics
1279 West 2200 South
Salt Lake City 84119
Tel: (801) 973-8555
FAX: (801) 973-8909

Arrow/Schweber Electronics
1946 West Parkway Boulevard
Salt Lake City 84119
Tel: (801) 973-6913
FAX: (801) 972-0200

NORTH AMERICAN DISTRIBUTORS (Cont'd)

Avnet Computer
1100 East 6600 South
Suite 150
Salt Lake City 84121
Tel: (800) 426-7999

Hall-Mark Computer
1100 East 6600 South
Suite 150
Salt Lake City
Tel: (800) 409-1483

Hamilton Hallmark
1100 East 6600 South
Suite 120
Salt Lake City 84121
Tel: (801) 266-2022
FAX: (801) 263-0104

Wyle Electronics
1325 West 2200 South
Suite E
West Valley 84119
Tel: (801) 974-9953
FAX: (801) 972-2524

WASHINGTON

Almac Arrow Electronics
14360 S.E. Eastgate Way
Bellevue 98007
Tel: (206) 643-9992
FAX: (206) 643-9709

Anthem Electronics
19017 120th Ave N.E.
Suite 102
Bothell 98011
Tel: (206) 483-1700
FAX: (206) 486-0571

Avnet Computer
8630 154th Ave, NE
Redmond 98052
Tel: (800) 426-7999

Hamilton Hallmark
8630 154th Avenue
Redmond 98052
Tel: (206) 881-6697
FAX: (206) 867-0159

Pioneer Technologies
2800 156th Ave S.E.
Suite 100
Bellevue 98007
Tel: (206) 644-7500

Wyle Electronics
15385 NE 90th St
Redmond 98052
Tel: (206) 881-1150
FAX: (206) 881-1567

WISCONSIN

Arrow/Schweber Electronics
200 N. Patrick
Suite 100
Brookfield 53045
Tel: (414) 792-0150
FAX: (414) 792-0156

Avnet Computer
2440 South 179th St
New Berlin 53416
Tel: (800) 426-7999

Hall-Mark Computer
2440 South 179th St
New Berlin 53146
Tel: (800) 409-1483

Hamilton Hallmark
2440 South 179th St
New Berlin 53146
Tel: (414) 797-7844
FAX: (414) 797-9259

Pioneer Standard
120 Bishops Way
Suite 163
Brookfield 53005
Tel: (414) 780-3600
FAX: (414) 780-3613

Wyle Electronics
150 North Patrick
Building 7, Suite 150
Brookfield 53045
Tel: (414) 879-0434
FAX: (414) 879-0474

ALASKA

Avnet Computer
1400 W Benson Blvd
Suite 400
Anchorage 99503
Tel: (800) 426-7999

CANADA

ALBERTA

Avnet Computer
1144 29th Avenue NE
Suite 108
Calgary T2E 7P1
Tel: (800) 387-3406

Pioneer/Pioneer
560, 1212-31 Ave. NE
Calgary T2E 7S8
Tel: (403) 291-1988
FAX: (403) 295-8714

BRITISH COLUMBIA

Almac Arrow Electronics
8544 Baxter Place
Burnaby V5A 4T8
Tel: (604) 421-2333
FAX: (604) 421-5030

Hamilton Hallmark
8610 Commerce Court
Burnaby V5A 4N6
Tel: (604) 420-4101
FAX: (604) 420-5376

Pioneer/Pioneer
4455 North 6 Road
Rochmond V6V 1P6
Tel: (604) 273-5575
FAX: (604) 273-2413

MANITOBA

Pioneer/Pioneer
540 Marjorie Street
Winnipeg R3H 0S9

ONTARIO

Arrow/Schweber Electronics
36 Antares Drive
Unit 100
Nepean K2E 7W5
Tel: (613) 226-6903
FAX: (613) 723-2018

Arrow/Schweber Electronics
1093 Meyerside, Unit 2
Mississauga L5T 1M4
Tel: (416) 670-2010
FAX: (416) 670-5863

Avnet Computer
Canada System Engineering Group
151 Superior Blvd.
Mississauga L5T 2L1
Tel: (800) 387-3406

Avnet Computer
190 Colonade Road
Nepean K2E 7J5
Tel: (800) 387-3406

Canada System Engineering Group
151 Superior Boulevard
Mississauga L5T 2L1
Tel: (800) 387-3406

Hamilton Hallmark
151 Superior Blvd.,
Unit 1-6
Mississauga L5T 2L1
Tel: (416) 564-6060
FAX: (416) 564-6033

Hamilton Hallmark
190 Colonade Road
Nepean K2E 7J5
Tel: (613) 226-1700
FAX: (613) 226-1184

Pioneer/Pioneer
3415 American Drive
Mississauga L4V 1T6
Tel: (416) 507-2600
FAX: (416) 507-2831

Pioneer/Pioneer
155 Colonnade Rd., S.
Suite 17
Nepean K2E 7K3
Tel: (613) 226-8840
FAX: (613) 226-6352

QUEBEC

Arrow/Schweber Electronics
1100 Street Regis Blvd
Dorval H9P 2T5
Tel: (514) 421-7411
FAX: (514) 421-7430

Gates Arrow Electronics
500 Boul.
St-Jean-Baptiste Ave
Quebec H2E 5R9
Tel: (418) 871-7500
FAX: (418) 871-6816

Avnet Computer
7575 Trans Canada
Suite 601
St. Laurent H4T 1V6
Tel: (800) 265-1135

Hamilton Hallmark
7575 Transcanada Hwy
Suite 600
Street Laurent H4T 2V6
Tel: (514) 335-1000
FAX: (514) 335-2481

Pioneer/Pioneer
520 McCaffrey
Street Laurent H4T 1N1
Tel: (514) 737-9700
FAX: (514) 737-5212

NORTH AMERICAN SERVICE OFFICES

Computervision

Intel Corporation's North American Preferred Service Provider

Central Dispatch: 1-800-876-SERV (1-800-876-7378)

ALABAMA
- Birmingham
- Huntsville

ALASKA
- Anchorage

ARIZONA
- Phoenix
- Tucson

ARKANSAS
- North Little Rock

CALIFORNIA
- Concord
- Los Angeles
- Ontario
- Orange
- Redwood City
- Sacramento
- San Diego
- San Francisco
- Van Nuys

COLORADO
- Colorado Springs
- Denver

CONNECTICUT
- E. Windsor
- Middlebury

FLORIDA
- Ft. Lauderdale
- Jacksonville
- Miami
- Orlando
- Pensacola
- Tampa

GEORGIA
- Atlanta

HAWAII
- Honolulu

ILLINOIS
- Chicago
- Wood Dale

INDIANA
- Carmel
- Evansville
- Ft. Wayne
- South Bend

IOWA
- Cedar Rapids
- Davenport
- West Des Moines

KANSAS
- Kansas City
- Wichita

KENTUCKY
- Louisville
- Madisonville

LOUISIANA
- Baton Rouge
- New Orleans

MAINE
- Auburn

MARYLAND
- Baltimore

MASSACHUSETTS
- Bedfoird
- S. Easton

MICHIGAN
- Detroit
- Flint
- Grand Rapids
- Lansing
- Troy

MINNESOTA
- Minneapolis

MISSOURI
- Springfield
- Street Louis

MISSISSIPPI
- Jackson

NEW HAMPSHIRE
- Manchester*

MONTANA
- Butte

NEBRASKA
- Omaha

NEW JERSEY
- Cherry Hill
- Hamilton Township
- Westfield

NEW MEXICO
- Albuquerque

NEW YORK
- Albany
- Binghampton
- Buffalo
- Farmingdale
- New York City
- Rochester
- Dryden

NORTH CAROLINA
- Ashville
- Charlotte
- Greensboro
- Raleigh
- Wilmington

OHIO
- Cincinnati
- Cleveland
- Columbus
- Dayton

OKLAHOMA
- Oklahoma City
- Tulsa

OREGON
- Beaverton

PENNSYLVANIA
- Camp Hill
- Erie
- Pittsburgh
- Wayne

SOUTH CAROLINA
- Charleston
- Columbia
- Greenville

TENNESSEE
- Chattanooga
- Knoxville
- Memphis
- Nashville

TEXAS
- Austin
- Houston
- Dallas
- Tyler

UTAH
- Salt Lake City

VERMONT
- White River Junction

VIRGINIA
- Charlottesville
- Richmond
- Roanoke
- Virginia Beach

WASHINGTON
- Renton
- Richland

WASHINGTON D.C.*

WEST VIRGINIA
- Charleston

WISCONSIN
- Milwaukee

CANADA
- Calgary
- Edmonton
- Fredericton
- Halifax
- Mississauga
- Montreal
- Ottawa
- Toronto
- Vancouver, BC*
- Winnipeg
- Quebec City
- Regina
- St.John's

Pentium® and Pentium Pro Processors and Related Products

In 1993, Intel introduced the Pentium® Processor, the fifth generation processor based on the Intel Architecture. The Pentium processor is fully code-compatible with previous members of the Intel Architecture family of microprocessors, thus preserving the value of user's software investments. Up to 3-1/2 times faster than the IntelDX2™ 66MHz processor, the Pentium processor extends the Intel processor performance continuum while maintaining full compatibility with existing software. The Pentium processor utilizes advanced design techniques, superscalar technology and enhanced floating-point to speed up today's applications and enable tomorrow's emerging technologies. The family is designed for mainstream desktops and powerful mobiles.

In November of 1995, Intel introduced the Pentium Pro processor, Intel's next generation of performance for high-end desktops, workstations and servers. The family is easily scalable up to four microprocessors in a multiprocessor system. The Pentium Pro processor delivers more performance than previous generation processors through an innovation called Dynamic Execution. This is the next step beyond superscalar architecture. Dynamic Execution makes possible the advanced 3D visualization and interactive capabilities required by high-end commercial and technical applications. The Pentium Pro processor includes advanced data integrity, reliability, and serviceability features for mission critical applications.

ISBN 1-55512-251-5

90000

EAN

9 781555 122515

Order Number 241732-003
Printed in USA/0196/27k/RRD/PG
Microprocessors

intel®

Printed on Recycled Material