CHAPTERS (Continued)

APPENDIXES

FIGURES

FIGURES (Continued)

TABLES

TABLES (Continued)

PREFACE

This manual provides the information necessary for the user to operate and maintain a Perkin-Elmer Model 3205 System.

This manual is intended for use by the technician who is required to operate and maintain the system.

Chapter 1 contains a general description of the system components. Chapter 2 describes the operation of the consolette. Chapter 3 provides an analysis of the control unit section of the central processing unit (CPU) board. Chapter 4 provides an analysis of the execution unit section of the CPU board. Chapter 5 describes the input/output (I/O) section of the CPU board. Chapter 6 provides an analysis of the memory unit section of the CPU board. Chapter 7 describes the power up/down section of the CPU board. Appendix A contains the mnemonics for the CPU board.

This manual is to be used in conjunction with the Model 3205 System Schematic and Assembly Drawings Manual, Publication Number 47-068.

For information on the contents of all Perkin-Elmer 32-bit manuals, see the 32-Bit Systems User Documentation Summary.

CHAPTER 1
GENERAL DESCRIPTION

1.1 INTRODUCTION

The Perkin-Elmer Model 3205 System is one of the lowest priced 32-bit minicomputers available today. Perkin-Elmer is able to achieve this breakthrough by using high technology components, space-saving memory packaging and a single-board processing unit.

The Model 3205 System is a fully configured, 32-bit processing system. The unbundled or stripped down version of the Model 3205 System is referred to as the Model 3205 Processor Subsystem and is targeted to the needs of the original equipment manufacturers (OEM).

Figure 1-1 shows the basic Model 3205 System in a 76.2cm (30") cabinet consisting of an eight-slot chassis, a power subsystem, a system consolette and a 50Mb cartridge disk drive (CDD50). The chassis contains a processor board with floating point capabilities and 512kb or 1Mb of memory, a multiperipheral controller (MPC) board capable of servicing eight full-duplex data communications channels, a line printer port and an intelligent disk controller (IDC). The basic system can be expanded to include a memory expansion board with 1, 2 or 3Mb of memory. This gives the system a total of 4Mb of real memory and an additional 50Mb CDD50.

Figure 1-2 shows an unbundled Model 3205 Processor Subsystem that contains an eight slot chassis, a system consolette, a processor board configured with 512kb of memory and an MPC board.

1.2 PROCESSOR BOARD COMPONENTS

The central processing unit (CPU) board is comprised of six functional units: control unit, execution unit, memory, power up/down, consolette interface and input/output (I/O).

These six units are synchronized by a 5MHz clock (200ns cycle). This cycle is divided into four 50ns states (0 to 3) with most microinstruction execution being completed within one cycle. These four states are used to divide the operation into two or more tasks (see Figure 1-3).

058-1

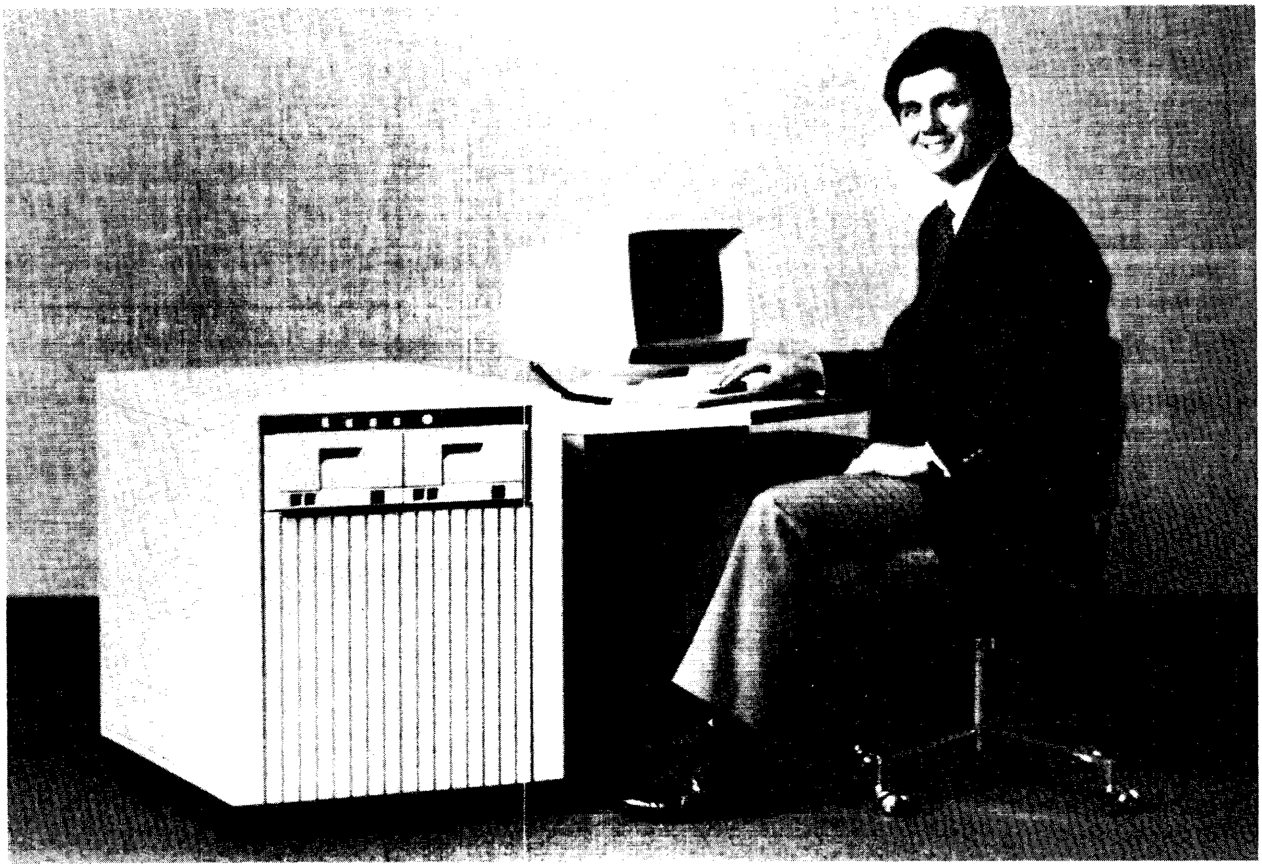


Figure 1-1 Model 3205 System

058-2

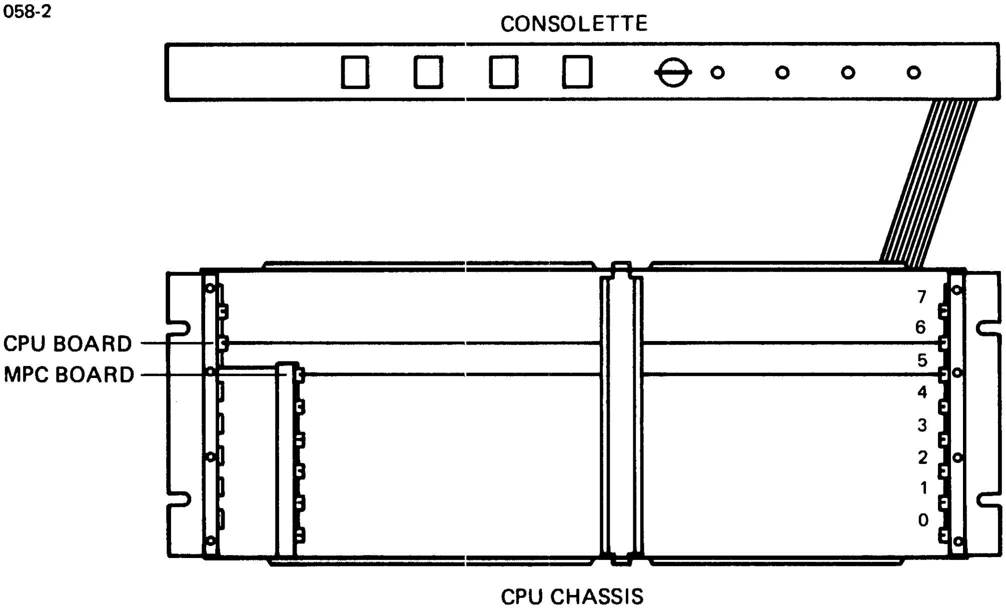


Figure 1-2 Model 3205 Processor Subsystem

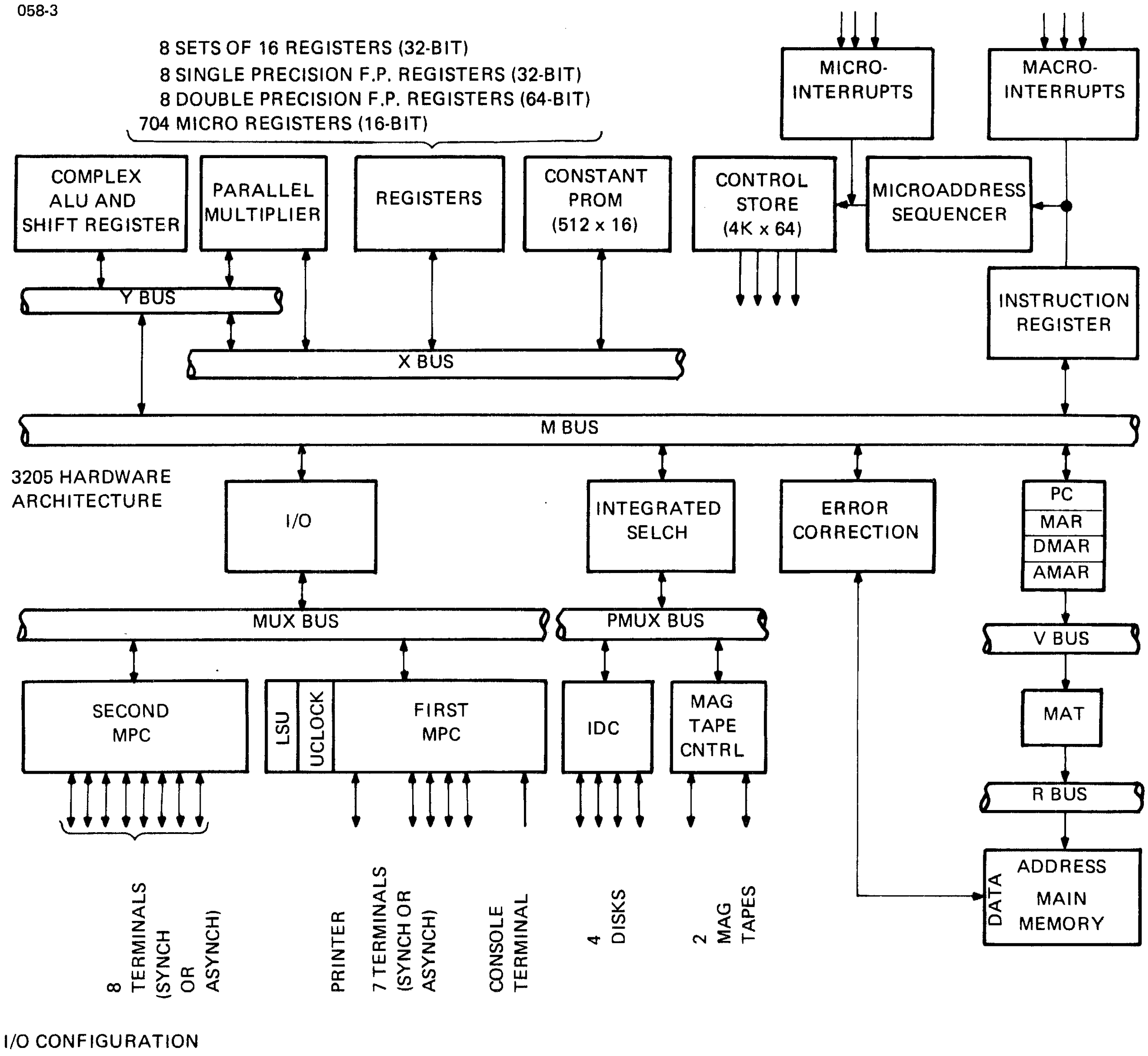


Figure 1-3 Model 3205 Processor Board Block Diagram

1.2.1 Control Unit

The control unit contains the control store (CS), control register (CR), microaddress sequencer, N register (NR), condition code multiplexor (CCMUX), instruction registers, macrointerrupt and microinterrupt circuitry.

1.2.2 Execution Unit

The execution unit contains the arithmetic logic unit (ALU), multiplier, external registers (constant programmable read-only memory (PROM), register external (REX)), P register (PR) and shift registers.

1.2.3 Memory

The Model 3205 System supports up to 4Mb of directly addressable main memory by using 64K random access memory (RAM) technology.

The memory unit contains the address registers, memory address translator (MAT), error check and correction (ECC), error logger, direct memory access (DMA) word counter and memory refresh.

1.2.4 Input/Output (I/O) Interface

The Model 3205 System incorporates two communications buses. The man/machine multiplexor (MUX) bus and the machine/machine private MUX (PMUX) bus.

- MUX bus

  The MUX bus can address up to 1,023 medium-speed devices such as printers, consoles, card readers, etc.

- PMUX bus

  The PMUX bus interfaces high-speed secondary storage devices such as disks and magnetic tapes to the integrated selector channel (ISELCH). The PMUX bus supports five device controllers. Once the PMUX bus has been activated and interfaced to the ISELCH for a DMA transfer, it functions in a completely autonomous fashion, with one controller or device at any one time. When the PMUX bus is not interfaced to the ISELCH, it is placed in the idle mode and functions as an extension of the MUX bus.

  The ISELCH interfaces with the PMUX and provides for DMA transfers.

1.2.5 Consolette Interface

The consolette logic section of the CPU board interfaces the consolette to the processor. The interface circuitry generates three macrointerrupts and a single instruction mode signal.

1.2.6 Power Up/Down

The power up/down section of the CPU board monitors both the P5 and P5U voltage levels and generates power fail signals, which are sent to the consolette interface logic. These signals are gated with other signals and generate power fail macrointerrupts, which are sent to the processor for appropriate action.

1.3 CAPABILITIES

The capabilities of the Model 3205 System are discussed in the following sections.

1.3.1 Comprehensive Instruction Set

The Model 3205 System uses the Perkin-Elmer Series 3200 instruction set, which is a comprehensive array of general-purpose processing instructions. The Model 3205 System includes a standard repertoire of single and double precision floating point instructions. The instruction set performs the following classes of operations:

- Load/store halfwords, fullwords and multiple words
- Fixed point arithmetic on halfwords and fullwords
- Logical operations (AND, OR, exclusive OR, compare and test) on halfwords or fullwords
- Logical and arithmetic shifts and rotation of halfwords and fullwords
- Extensive bit manipulation
- Floating point arithmetic on single (32-bit) and double (64-bit) precision operands
- List operations
- Data handling operation
- I/O
- Byte manipulations
- Privileged system functions

- Storage-to-storage functions

- Decimal conversion

1.4 INTERNAL ARCHITECTURE

The Model 3205 System is structured around internal buses, which are discussed further in the following sections.

1.4.1 Processor Internal Buses

The X, Y and S buses interconnect major units on the CPU board.

- The X bus is the destination bus containing the results of the operation.

- The Y bus is the first operand bus.

- The S bus (sequencer bus) is used for inputting address data for branching, user instruction decode and loading the repeat counter (RPCT) in the microaddress sequencer.

1.4.2 Processor/Device Controller Buses

The MUX and PMUX buses connect the device controllers to the processor for I/O request/response signals and data communications.

1.4.3 Processor/Memory Buses

There are two processor/memory buses:

- The C bus passes addresses to the virtual address register.

- The M bus passes data between the processor and memory.

1.4.4 Internal Memory Buses

The three internal memory buses are:

- The E bus passes data to or from the memory data register (MDR), ECC and the memory RAMs.

- The MA bus passes addresses from the real address bus to local memory RAMs.

- The memory MUX bus passes addresses from the real address bus to the expansion memory RAMs.

CHAPTER 2
CONSOLETTE

2.1 INTRODUCTION

This chapter describes the consolette operation and handshaking.

2.2 OPERATION

The operation of the consolette controls and indicators is described in the following sections. The front of the consolette is shown in Figure 2-1.

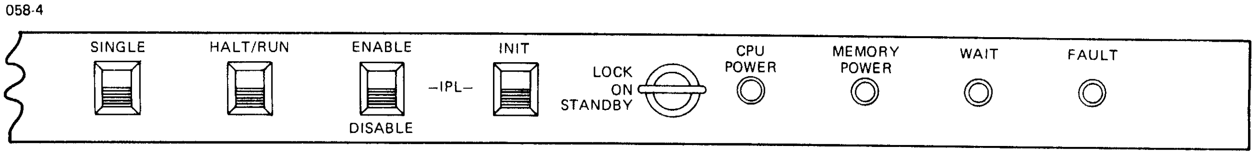


Figure 2-1 Consolette Function Switches and Indicators

2.2.1 Function Switches

The operation of the five consolette function switches is described in the following sections.

2.2.1.1 Key-Operated Security Lock

The key-operated security lock switch is a three-position (STANDBY/ON/LOCK) switch that controls the primary power to the system. The key-operated switch in the corresponding position performs the following functions:

STANDBY	Processor 5 volts (P5) is OFF. When equipped with P5U, STANDBY is ON.
ON	Primary power is ON and all switches are active.

LOCK — Primary power is ON, and the INITIALIZE HALT/RUN and SINGLE switches on the consolette are disabled.

This switch is normally in the LOCK position.

2.2.1.2 Initialize (INIT) Switch (Momentary Switch)

This momentary action switch, when depressed, causes system initialization. After system initialization, all device controllers on the system's multiplexor (MUX) bus are cleared and certain processor functions are reset. This switch is disabled when the key-operated switch is in the LOCK position.

2.2.1.3 Initial Program Load (IPL) ENABLE/DISABLE Switch

When the two-position IPL switch is in the ENABLE position, and the key-operated security lock switch is placed in the ON position from the STANDBY position, and one of the following two conditions occurs:

- the INIT switch is depressed, or
- AC power is restored to the system,

the system's bootloader is reloaded from the loader storage unit (LSU). The IPL switch is normally in the DISABLE position.

2.2.1.4 HALT/RUN Switch

When depressed, this single action switch forces a running system to halt and enter the processor console service state, or forces a halted system in the processor console service state to enter the run mode.

2.2.1.5 SINGLE Switch

When enabled, this two-position switch places the processor in the single instruction cycle mode and the running program in the processor console service mode. When in the single instruction cycle mode, the processor is returned to the processor control mode after each user instruction is executed and the location counter (LOC) displays the address of the next user instruction to be executed. The status portion of the program status word (PSW) reflects the execution of the previous user instruction. This switch normally remains in the OFF position.

2.2.2 Indicators

The following sections describe the four consolette indicators.

- Central processing unit (CPU) power

  When lit, indicates that P5 is on.

- Memory power

  When lit, indicates that P5U is on.

- WAIT

  A running program can place the processor into the wait state by setting the wait bit of the current PSW. The WAIT indicator is lit to inform the operator of this condition and when the processor is in the console service mode.

- FAULT

  This indicator is lit during system initialization and remains lit until the microcode power-up test is successfully completed. It remains lit if the processor self-test fails.

2.3 CONSOLETTE HANDSHAKING

Figure 2-2 shows consolette handshaking with the processor board.

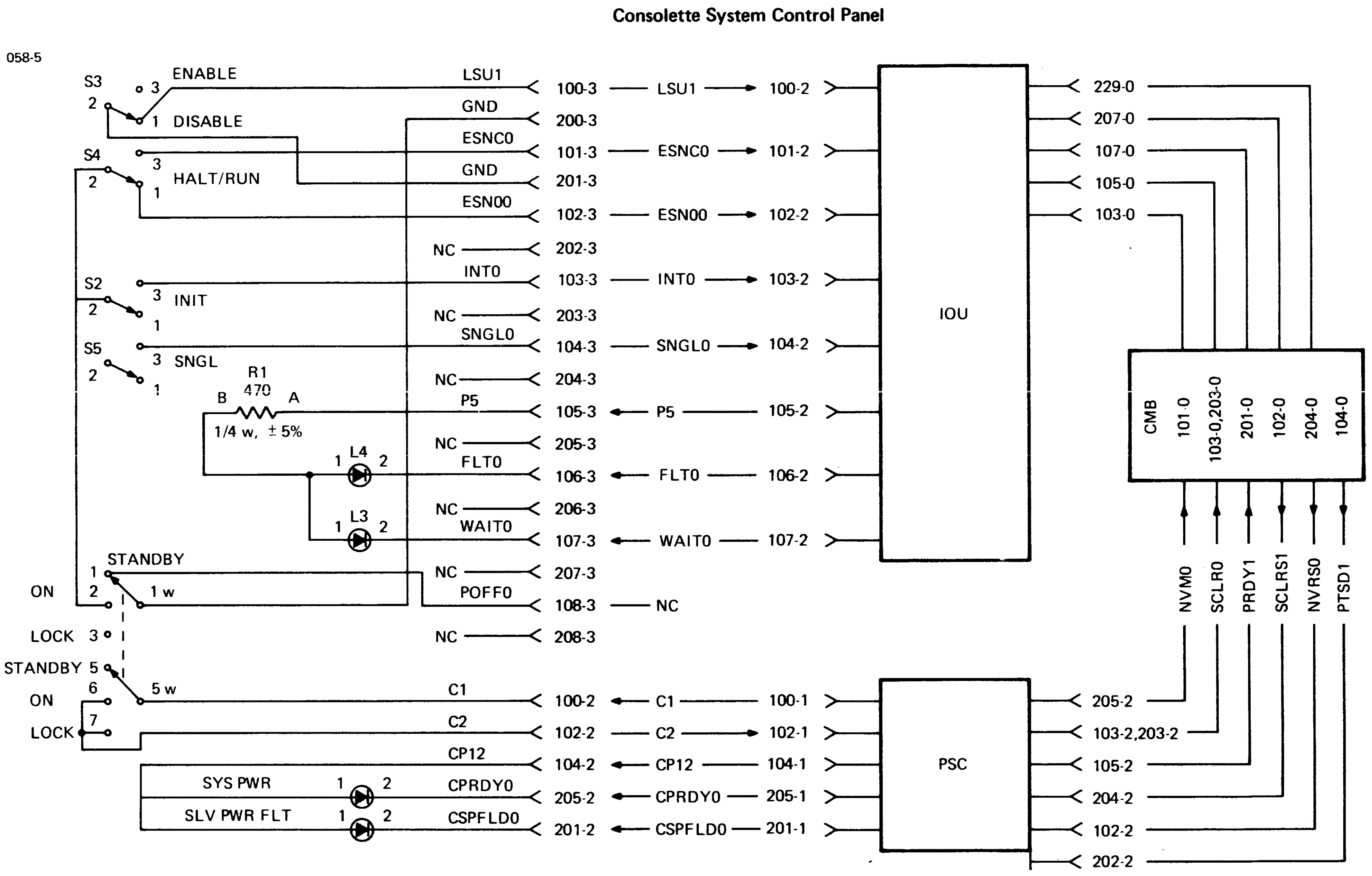


Figure 2-2 Consolette Handshaking Diagram

CHAPTER 3
CONTROL UNIT
THEORY OF OPERATION

3.1 INTRODUCTION

This chapter provides an analysis of the control unit section of the Model 3205 System central processing unit (CPU) board. The control unit consists of the following units (see Figure 3-1).

- Microaddress sequencer
- Control store (CS)
- Control registers (CR)
- Microinterrupt logic
- Instruction register
- Macrointerrupt logic
- N register (NR)
- Condition code multiplexor (CCMUX)

The function of the control unit is to generate the 64-bit microinstruction (microword) for the system.

The CR is loaded with the microinstruction which is addressed by either the microaddress sequencer or the microinterrupt logic. The microaddress sequencer is sourced by combinations of the macrointerrupt logic, the instruction register, NR and a 12-bit CR field. The CCMUX provides the true or false conditions used by the microaddress sequencer for branching decisions.

All schematic references (Sheet XX) refer to Functional Schematic 35-864 D08.

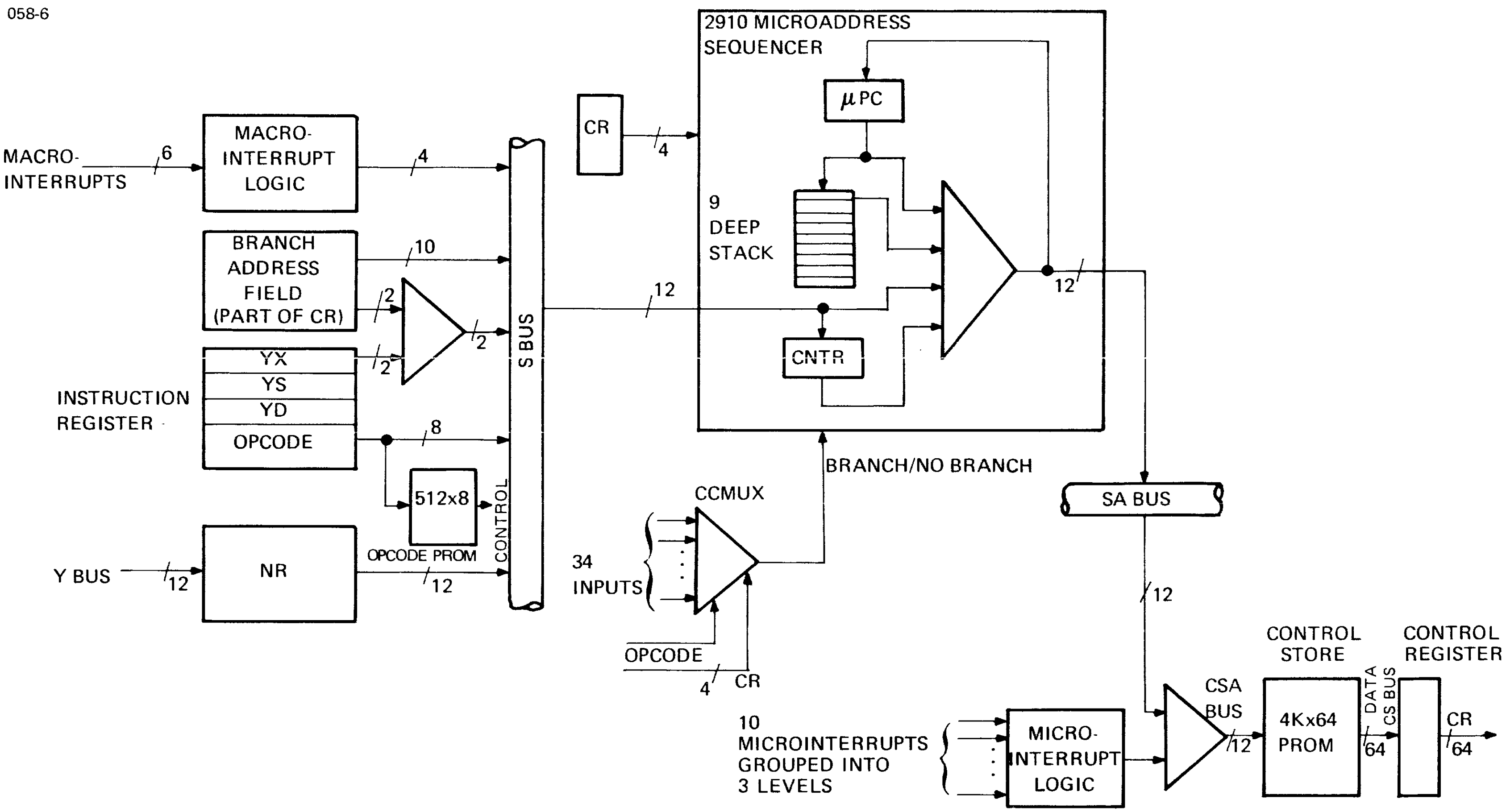


Figure 3-1 Control Unit Block Diagram

3.2 MICROADDRESS SEQUENCER

The microaddress sequencer (Sheet 21) controls the sequence of execution of the microinstructions stored in the CS. The microaddress sequencer provides both sequential and conditional branching within a 4,096 microword range. A nine word, last-in/ first-out (LIFO) stack provides microsubroutine nesting and looping capability. Microinstruction looping count control is provided with a 4,096 count capacity. The microaddress sequencer instruction format and definitions are listed in Table 3-1 (see Figure 3-2).

During each microinstruction, the microaddress sequencer provides a 12-bit address to the sequencer address (SA) bus from one of four possible sources.

- The internal 12-bit microprogram counter (μPC) has two functions.

  - When the carry-in (pin 32) is high, the μPC is loaded with the SA bus value + 1 for sequential microinstruction execution.

  - When the carry-in (pin 32) is low, the μPC reloads the current SA bus value unmodified. This feature is only used during a microinterrupt when the SA bus data is not used and becomes the return point from the interrupt service routine (ISR).

- An internal register/counter (CNTR) is used to store a future branch address or a loop count value.

- An internal nine word deep LIFO stack

- The S bus is the only external source. S bus drivers are selected according to the type of sequencer instruction being executed. The S bus drivers are:

  - macrointerrupt logic

  - NR

  - instruction register

  - control register next address field CR11:00

058-7

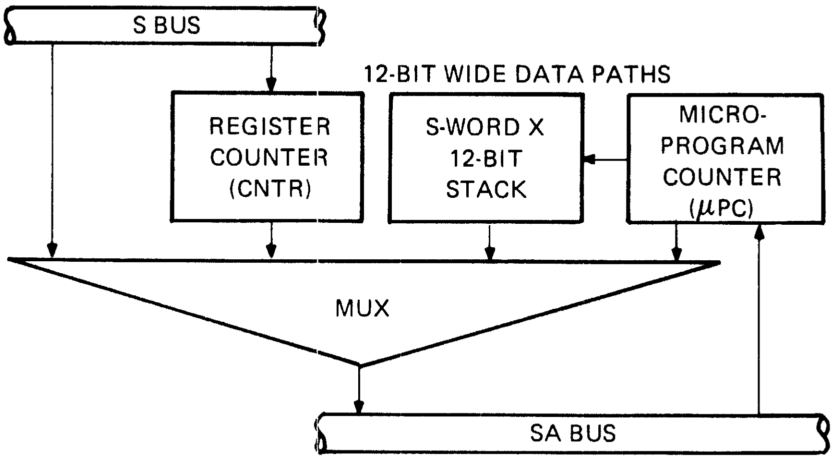


Figure 3-2 Microaddress Sequencer Block Diagram

3.2.1 Microaddress Sequencer Instruction Field

The microaddress sequencer is controlled by CS bits 15:12. These correspond to instruction bits I3:0 of the microaddress sequencer. This field selects one of 16 sequencer instructions defined in Table 3-1.

The microaddress sequencer instruction field is reset (zeroed) on a system clear (SCLR0) or a system BOMB.

TABLE 3-1 MICROADDRESS SEQUENCER INSTRUCTIONS

I3:0	MNEMONIC	INSTRUCTION	NOTES
0	JZ	Jump zero unconditional SA bus <= ZERO	Only used during system clear (SCLR0) or system BOMB. Clears all sequencer functions.

1	CJS	Conditional jump to subroutine
		
		IF condition TRUE
		THEN SA bus <= S bus
		push μPC to top of stack
		ELSE SA bus <= μPC	

2	JMAP	Jump map unconditional SA bus <= S bus	Only used for the four-way branch in RX handler. S bus sourced by ten CR and two M bus bits indicating RX type.

TABLE 3-1 MICROADDRESS SEQUENCER INSTRUCTIONS (Continued)

I3:0	MNEMONIC	INSTRUCTION	NOTES

3	CJP	Conditional jump pipeline
		
		IF condition TRUE
		  THEN SA bus <= S bus
		  ELSE SA bus <= µPC	

4	PUSH	Push/conditional load CNTR
		
		IF condition TRUE
		  THEN SA bus <= µPC
		    push µPC to top of stack
		    CNTR <= S bus
		  ELSE SA bus <= µPC
		    push µPC to top of stack
		unconditional push stack
			Used in setting up loops.

5	JSRP	Conditional jump to subroutine IF condition TRUE THEN SA bus <= S bus push µPC to top of stack ELSE SA bus <= CNTR push µPC to top of stack unconditional push stack	CNTR value must be loaded previously.

6	CJV	Conditional jump vector
		
		IF condition TRUE
		  THEN SA bus <= S bus
		  ELSE SA bus <= µPC
			S bus sourced by IRH (opcode) or macrointerrupt logic.

7	JRP	Conditional jump to CNTR or S bus
		
		IF condition TRUE
		  THEN SA bus <= S bus
		  ELSE SA bus <= CNTR
			CNTR value must be loaded previously.

8	RFCT	Repeat loop until CNTR = 0
		
		IF CNTR = 0
		  THEN SA bus <= µPC
		      pop stack
		  ELSE SA bus <= top of stack
		      decrement CNTR
			CNTR must have been loaded previously.
			Branch address must have been pushed onto the stack previously.

9	RPCT	Repeat loop until CNTR = 0
		
		IF CNTR = 0
		  THEN SA bus <= µPC
		  ELSE SA bus <= S bus
		    decrement CNTR
			CNTR must have been loaded previously.

TABLE 3-1 MICROADDRESS SEQUENCER INSTRUCTIONS (Continued)

I3:0	MNEMONIC	INSTRUCTION	NOTES

A	CRTN	Conditional return from subroutine
		IF condition TRUE
		THEN SA bus <= top of stack
		pop stack
		ELSE SA bus <= μPC	

B	CJPP	Conditional jump and pop stack
		IF condition TRUE
		THEN SA bus <= S bus
		pop stack
		ELSE SA bus <= μPC	

C	LDCT	Load CNTR and continue unconditional CNTR <= S bus SA bus <= μPC	

D	LOOP	Test end loop
		IF condition TRUE
		THEN SA bus <= μPC
		pop stack
		ELSE SA bus <= top of stack
			Conditional exiting of a loop at the bottom.

E	CONT	Continue SA bus <= μPC unconditional	

F	TWB	Three-way branch
		IF conditon TRUE
		THEN SA bus <= μPC
		pop stack,
		decrement CNTR
		ELSE IF CNTR = 0
		THEN SA bus <= S bus
		pop stack
		ELSE SA bus <= top of stack
		decrement CNTR
			CNTR must have been previously loaded. Branch address must have been PUSHed onto the stack previously.

NOTES

Unless otherwise specified:

- The (μPC) is loaded with the present SA bus value + 1 at the end of every cycle.
- The stack remains the same.
- The internal/register counter (CNTR) remains unchanged.

3.3 CONTROL STORE (CS)

The CS read-only memory (ROM) (Sheet 17) contains 4,096 microinstructions (microwords). Each microinstruction is 64-bits wide. The CS is addressed by a 12-bit field from the microaddress sequencer or the microinterrupt logic. Table 3-2 is a list of bit numbers and definitions.

TABLE 3-2 CS BIT DEFINITIONS

CS BIT	DEFINITION	NOTES
11:00	Next microinstruction address, sequencer internal CNTR data, register external (REX) address, or constant programmable read-only memory (PROM) address	
15:12	Microsequencer instruction bit I(3:0)	See Table 3-1
19:16	CCMUX selection MSB:LSB	See Table 3-7
20	FP/G 1=floating point register 0=general register	
21	Memory address translator (MAT) enable	See Table 6-1
22	REX halfword select. 0=lower halfword 1=upper halfword	
22	R/W 0=write, 1=read	
24	H/B 0=byte, 1=halfword	
27:25	Memory, I/O, control (MIC) field MSB:LSB	
28	MEMOP 0=no memory access, 1=memory access	
30:29	Selects memory address source, PC, MAR, DMAR or AMAR	
32:31	Selects type of memory operation	
33	Condition code register loading enable 0=load, 1=no load	See Table 4-14
34	Controls condition code conversion, ALU status to Perkin-Elmer status 0=convert status to Perkin-Elmer form 1=no conversion	
35	Condition code register halfword control, 1=halfword 0=multiple halfwords	

TABLE 3-2 CS BIT DEFINITIONS (Continued)

CS BIT	DEFINITION	NOTES
36	REX write enable, 0=write, 1=no write	
37	NR output enable	
38	YS and YD count direction	See Table 3-4
39	YS or YD count enable	
41:40	REX address select MSB:LSB	See Table 4-6
43:42	ALU register internal (RIN) address MUX select MSB:LSB	See Table 4-4
59:44	ALU instruction (I15:00)	See Table 4-3
61:60	Shift register load enable control MSB:LSB	See Table 4-9
63:62	Y bus source control MSB:LSB	See Table 4-8

3.4 CONTROL REGISTERS (CRs)

The CRs (Sheet 25) stores the 64-bit microinstruction being executed. The CR is clocked at the beginning of every cycle except during a clock stop (refresh freeze or memory error). CR bits 15:12 (microaddress sequencer instruction bits) can be reset by either a SCLR0 or BOMB (see Table 3-1).

3.5 MICROINTERRUPTS

Microinterrupts (Sheet 22) allow the system to halt the present user instruction in response to a condition that requires immediate attention. The microinterrupt logic:

- produces the microcode vector address needed for branching to the correct interrupt handler,
- determines the priority of coincidental interrupts so that the highest priority interrupt can be serviced while the lower priority interrupts are queued, and
- detects invalid conditions called BOMBs.

A system BOMB is caused by one of three unique hardware error conditions:

- BOMBA occurs if a microinterrupt goes active when the same level microinterrupt has already been queued or is being serviced. BOMBA is not detected on level zero.
- BOMBB is the product of any two coincidental, level two microinterrupts.
- BOMBC is the product of a microaddress sequencer stack overflow (attempting a Push Stack when the stack .is full).

When an interrupt or a queued interrupt is serviced, the service flag for that level is set. When an interrupt is queued, the queue flag for that level is set.

The general rules followed by the microinterrupt logic are:

- The status is updated at the end of every cycle.
- When the STOP signal (STP0) is active, all incoming interrupts are queued.
- When the STOP signal (STP0) is reset, all rules apply.
- System clear (SCLR0) resets the microinterrupt logic.

The specific rules governing the microinterrupt logic are:

- An interrupt is serviced when that interrupt level goes active and if no higher interrupt is active, queued or being serviced.
- An interrupt is serviced if a higher level service flag is being reset at the same time the present interrupt goes active.
- A queued interrupt is serviced when no higher interrupt is present, no higher interrupt is queued and only one higher service flag is being reset. When a queued interrupt is serviced, the queue flag is reset.
- A service flag is reset when the reset level signal (RLV20:00) for that level goes active. Reset level signals are activated by microcode at the end of the service routine.
- An interrupt is queued when at least one higher interrupt is present at the same time, at least one higher interrupt is queued or at least one higher service flag is already set.

3.5.1 Microinterrupt Priority

The ten microinterrupts are divided into three priority levels. The levels, a description, signal names and microcode vector addresses are listed in descending order in Table 3-3.

TABLE 3-3 MICROINTERRUPTS

PRIORITY LEVEL	DESCRIPTION	SIGNAL NAME	MICROCODE VECTOR ADDRESS
2	REX parity error	REXPO	X'D80'
2	Alignment fault	AFLTO	X'D88'
2	Nonconfigured memory	NMEMO	X'D90'
2	Error check and correction (ECC) double-bit error	ECCDO	X'D98'
1	DMA read/write	DMARDO	X'DA0'
1	DMA interrupt	DMAINTO	X'DA8'
0	MAT fault	MFLTO	X'DC0'
0	MAT miss	MISSO	X'DC8'
0	Memory dirty bit	DIRTO	X'DD0'
0	Refresh interrupt	REFINTO	X'DD8'

3.5.1.1 Register External Parity Error (REXPO)

An odd parity bit is stored with each byte written into REX. These parity bits are checked whenever REX is read. If either is incorrect, the REX parity error signal (REXPO) is generated.

3.5.1.2 Alignment Fault (AFLTO)

An AFLTO interrupt occurs when a memory read is attempted on an incorrect boundary. For example, a fullword read on a halfword boundary.

3.5.1.3 Nonconfigured Memory (NMEMO)

An NMEMO interrupt occurs when the real address attempts to access memory above the available amount. For example, a system configured with 1Mb of memory can access up to and include memory address X'FFFFF'. A memory address of X'100000' in this system causes an error. The RAS PAL (A9) contains the logic for determining system memory size.

3.5.1.4 Error Check and Correction Double-Bit Error (ECCDO)

An ECCDO interrupt occurs when two or more bits of the word currently being read from memory are incorrect. This error cannot be corrected.

3.5.1.5 Direct Memory Access Read/Write (DMARDO)

When a DMA interrupt (DMAINTO) occurs, DMARDO allows the vector address PAL to direct the microcode via the CS address bus to do a memory read (0) or a memory write (1) operation.

3.5.1.6 Direct Memory Access Interrupt (DMAINTO)

A DMA interrupt occurs when the integrated selector channel (ISELCH) requests a read from or a write to memory. The DMA4QO signal automatically routes a DMA interrupt to the queue during a memory refresh.

3.5.1.7 Memory Address Translator Fault (MFLTO)

A MFLTO interrupt occurs when a read, write or an instruction fetch is attempted on a particular segment that is protected against such an operation in its segment table entry. When a memory segment access is attempted by a task that does not have correct access level, MFLTO occurs. A MFLTO also occurs when the virtual address used to access a memory segment has no corresponding real address in the segment.

3.5.1.8 Memory Address Translator Miss (MISSO)

A MISSO interrupt occurs when the specified virtual address has not yet been loaded into the hardware MAT look-up table.

3.5.1.9 Memory Dirty Bit (DIRTO)

Every segment table entry in system memory contains a dirty bit. The dirty bit is set if any location within the segment has been modified. An example of dirty bit use is the case of restoring a disk file previously loaded to main memory.

If the dirty bit of a particular segment is not set, then that segment has not been modified and need not be restored.

3.5.1.10 Refresh Interrupt (REFINTO)

A REFINTO occurs when a memory refresh must be done and the execution unit is continually accessing memory. If the execution unit is not continually going to memory, REFINTO is not necessary and refresh is transparent.

3.6 INSTRUCTION REGISTERS

The instruction register (Sheet 19) is used to store and manipulate the 24 most significant bits (MSBs) of the present user instruction.

3.6.1 Instruction Register High (IRH)

The IRH is a 16-bit register that stores bits 15:00 of the user instruction. The IRH consists of an 8-bit opcode latch and two 4-bit up/down counters. The opcode latch contains the most significant eight bits of the user instruction. The counters contain the destination (YD) and source (YS) fields of the user instruction. The opcode latch provides the S bus with the first microaddress of the microcode routine that emulates the user instruction. The two register/counters provide the 1 of 16 register internal (RIN) and REX addresses in the current register set. The counters can be incremented, decremented or reloaded during a user instruction by microcode commands. The counters are reloaded using the memory data register (MDR) when signal IRHLEO is active. Table 3-4 describes the CS bits that control the IRH function.

TABLE 3-4 IRH UP/DOWN COUNT CONTROL

CS39	CS38	CS21	MEANING
1	X	X	Disable count
0	0	0	Decrement YS
0	0	1	Decrement YD
0	1	0	Increment YS
0	1	1	Increment YD

3.6.2 Instruction Register Low (IRL)

The IRL is an 8-bit latch that stores bits 23:16 of the user intruction. These eight bits are divided into two 4-bit fields. The function (F) field is the four MSBs (19:16) that specify which RX format (RX1, RX2 or RX3) the instruction is in. The second index field is the four least significant bits (LSBs) (23:20) that select the general register containing the second index value. Table 3-5 further defines the F field.

TABLE 3-5 F FIELD FORMAT

F FIELD 19:16	FORMAT
X X 0 0	RX1 short absolute
X X X 1	RX2 relative
0 0 1 0	RX3 long double indexed

For more information, consult the Model 3205 System Instruction Set Reference Manual.

3.7 MACROINTERRUPTS

A macrointerrupt (Sheet 20) is an interrupt of the user program (not the microprogram) that acknowledges a system condition requiring attention. The six macrointerrupts are input to the macrointerrupt priority encoder (MPE), which produces a unique 3-bit code for each interrupt. Along with the 3-bit code, the interrupt generates the signal MINTO, which alerts the system to the macrointerrupt condition. When MINTO is serviced, the 3-bit code becomes part of the microcode vector address for that interrupt. When active, signal PSW171 masks macrointerrupts.

3.7.1 Macrointerrupt Priority

Table 3-6 shows the macrointerrupts in descending priority.

TABLE 3-6 MACROINTERRUPTS

PRIORITY LEVEL	DESCRIPTION	SIGNAL NAME	MICROCODE VECTOR ADDRESS
5	Console attention	FSCATNO	X'DF5'
4	Primary power fail	FSPPFO	X'DFD'
3	Early power fail	FSEPFO	X'DF3'
2	I/O attention	I/OATNO	X'DFB'
1	DMA bad status	DMABDSTO	X'DF7'
0	DMA end of transfer	ELFINO	X'DFF'

3.7.1.1 Console Attention (FSCATNO)

The FSCATNO macrointerrupt results from one of the following conditions when the system is in the console service mode.

- The breakpoint (BRK) instruction is executed by a running program when program status word (PSW) bit 23 is zero.
- Execution of an instruction is completed while in the single-step mode.
- The HALT/RUN switch is depressed momentarily while the processor is in the run mode.
- Following the system initialization sequence, when the following conditions exist:
    - Memory back-up power was lost.
    - The loader storage unit (LSU) is not enabled when the initialization sequence is completed.

- Following the system initialization sequence, when the following conditions exist:

  - Memory back-up power was maintained.
  - The LSU is not enabled when the initialization sequence is completed.
  - Contents of physical memory location X'000028' indicate that the processor was in the console mode when system initialization occurred.

- An attempt to fetch a machine malfunction interrupt new PSW results in a noncorrectable memory error.

The FSCATNO interrupt requests that the system console terminal and other input/output (I/O) devices remain queued until the run mode is entered.

3.7.1.2 Primary Power Fail (FSPPFO)

Early power fail (EPF) detect occurs when the primary power fail (PPF) sensor detects a low voltage, the power switch is turned from the ON to the STANDBY position, or the INIT switch is depressed.

At the end of execution of the current instruction or the current iteration of the current interruptible instruction, a machine malfunction interrupt is taken if PSW bit 18 is set.

Following EPF detect, software has one millisecond before the automatic shutdown procedure of the processor takes control as a result of PPF. During this procedure, the following actions occur:

1. The fullword power fail save area pointer is fetched from location X'000084'.
2. The information listed in Table 3-7 is saved by firmware in the power fail save area.

TABLE 3-7 PREPOWER FAIL INFORMATION SAVED

DATA	OFFSET IN SAVE AREA (IN BYTES)
Current PSW	0-7
Eight general register sets (in order 0-F)	8-519
Interruptible instruction state (scratchpad registers)	520-583
All floating point registers	584-679
*** P5 GOES DEAD ***	

3. The processor waits for power restore.

NOTES

- If the pointer found in location X'000084' does not specify a save area aligned to a fullword boundary, the processor forces correct alignment by replacing the two LSBs of the pointer with zeros.

- The floating point masked mode bit in the PSW has no effect on the saving of the floating point registers.

- The IIP bit in the PSW has no effect on the saving of the scratchpad registers.

3.7.1.3 Early Power Fail (FSEPFO)

The FSEPFO signal is generated when the high voltage area in the power supply drops below a specified "low voltage" threshold and PSW bit 18 is set. FSEPFO can be initiated in one of three ways:

- The low voltage comparator inside the power supply goes active.

- The power switch is turned from the ON to the STANDBY position.

- The INIT switch is depressed.

Whenever FSEPF0 is acknowledged, all lower priority macrointerrupts are ignored.

3.7.1.4 Input/Output Attention (I/OATN0)

The I/OATN0 signal is generated by a device controller on either the MUX bus (ATN0) or private multiplexor (PMUX) bus (PATN0) when a device requires bus access for data transfer. The signal I/OATN0 can be masked by PSW1'/1. The ATN0 signal is received from pin 223-0 and PATN0 is received from pin 223-1 (Sheet 29).

3.7.1.5 Direct Memory Access (DMA) Termination

The DMA or PMUX operation can be terminated in one of two ways: bad status (DMABST0) or good status (ELFIN0). Signal DMABST0 also has the task of informing the processor that the original termination, good or bad, was not serviced when it occurred because interrupts were disabled. When the interrupts are reenabled, DMABST0 causes the original termination interrupt to be serviced.

- DMA bad status (DMABST0)

  The DMABST0 signal indicates bad termination of a DMA transfer or that the control unit should now service the original termination, good or bad. DMABST0 can be generated in one of three ways:

  - The microcode can issue a BADSTATUS command when a double-bit error (ECCD0) microinterrupt occurs during an ISELCH transfer.

  - The device controller currently addressed returns bad status on a STATUS command from the ISELCH.

  - The device controller currently addressed activates the signal CHK1 during a data available (DA) or data request (DR) command from the ISELCH.

- DMA end of transfer (ELFIN0)

  The ELFIN0 signal indicates good termination of a DMA transfer. ELFIN0 is generated when the device controller returns SYNC and the signal DMAEND0 is active at the same time. DMAEND0 is generated by the DMA word counter when the transfer is complete.

3.8 N REGISTER (NR)

The NR (Sheet 14) is a 12-bit register loaded from the X bus that has two functions. The NR can be used to source the microaddress sequencer S bus during a load internal counter instruction or it can source the N field of the arithmetic logic unit (ALU) instruction. CS bit 37 controls the register output enable.

3.9 CONDITION CODE MULTIPLEXOR (CCMUX)

The CCMUX (Sheet 21) selects one of 34 conditions to be tested by the microaddress sequencer. The condition is selected using CR bits 19:16, signals CCXH1 and CCXL1 from the opcode PROM, and input S (select). CR bits 19:16 are part of the microinstruction being executed and signals CCXH1 and CCXL1 form the select inputs to the CCMUX extension. The CCMUX extension allows positions 14 and 15 to become one of four inputs. CR bits 35:33 are logically ANDed to form the S input. The S input, when active (high), selects the ALU status (ALUCT1), which is the output of a MUX whose inputs are C (carry), V (overflow), N (negative), Z (zero), link, flag 1, flag 2 and flag 3 (see Figure 3-3). Table 3-8 shows all possible CCMUX inputs and how they are selected.

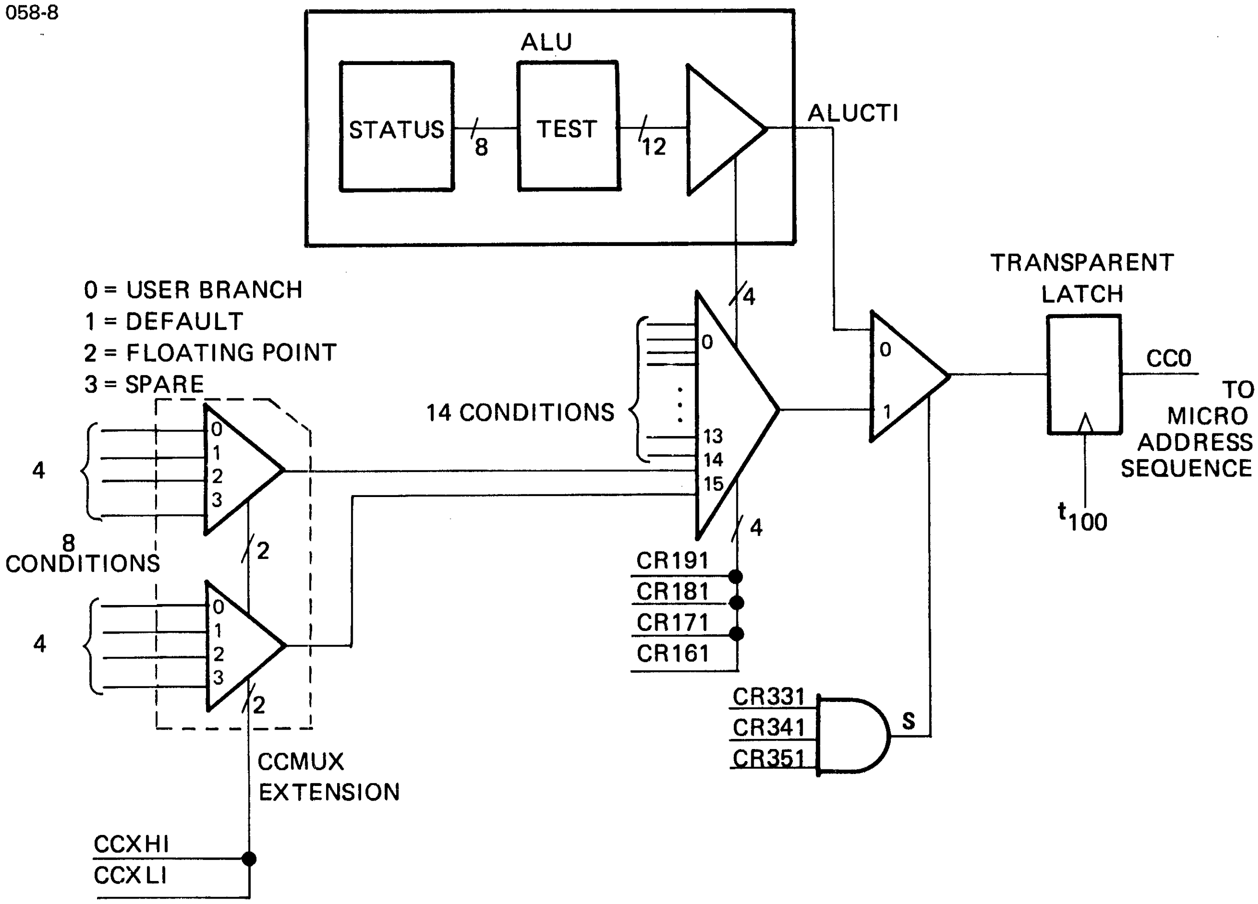


Figure 3-3 34-Input CCMUX

TABLE 3-8 CCMUX SELECTION

CONTROL STORE BITS					CCXH1	CCXL1	SIGNAL NAME	DESCRIPTION
S	19	18	17	16				
0	0	0	0	0	X	X	YXZF1	Test for index 1=no index, 0=index
0	0	0	0	1	X	X	Y15F1	Sign of bus from last ucycle 1=negative, 0=positive
0	0	0	1	0	X	X	YSZF1	Test YS field 1=YS zero, 0=YS not zero
0	0	0	1	1	X	X	P5RA	Fixed to logical 1, always branch
0	0	1	0	0	X	X	I/OHW1	Test I/O device configuration 1=halfword, 0=byte device
0	0	1	0	1	X	X	I/OSYN	Test for sync
0	0	1	1	0	X	X	MDR151	Test for sign of MDR 1=negative, 0=positive
0	0	1	1	1	X	X	MATPRT1	Odd parity over 32-bits 1=odd, 0=even
0	1	0	0	0	X	X	CCR00	ALU zero status 0=zero, 1=not zero
0	1	0	0	1	X	X	CCR10	ALU carry status 0=carry, 1=not carry
0	1	0	1	0	X	X	CCR20	ALU negative status 0=negative, 1=not negative
0	1	0	1	1	X	X	CCR30	ALU overflow status 0=overflow, 1=no overflow
0	1	1	0	0	X	X	MINT0	Macrointerrupt 0=macrointerrupt 1=no macrointerrupt
0	1	1	0	1	X	X	PSW130	Floating point instruction mask 0=disable, 1=enable
0	1	1	1	0	0	0	MSKT1	Mask true 1=branch, 0=no branch

TABLE 3-8 CCMUX SELECTION (Continued)

CONTROL STORE BITS					CCXH1	CCXL1	SIGNAL NAME	DESCRIPTION
S	19	18	17	16				
0	1	1	1	0	0	1	YDZF1	Test YD field status 1=zero, 0=not zero
0	1	1	1	0	1	0	XNZF0	X, Y bus bits 11:08 status 1=not zero; normalized 0=zero; unnormalized
0	1	1	1	0	1	1		*
0	1	1	1	1	0	0	MSKF1	Mask false 0=no branch, 1=branch
0	1	1	1	1	0	1	PSW231	Relocation protection 1=protect mode 0=not in protect mode
0	1	1	1	1	1	0	STICKY	Set on 4-bit shift right when carry out = 1.+
0	1	1	1	1	1	1		*
1	0	0	0	0	X	X	ALUCT1	(N EX-OR V) +Z
1	0	0	0	1	X	X	ALUCT1	N EX-OR V
1	0	0	1	0	X	X	ALUCT1	Z
1	0	0	1	1	X	X	ALUCT1	V
1	0	1	0	0	X	X	ALUCT1	Low (fixed to logical zero)
1	0	1	0	1	X	X	ALUCT1	C (carry)
1	0	1	1	0	X	X	ALUCT1	Z + Cnot
1	0	1	1	1	X	X	ALUCT1	N (negative)
1	1	0	0	0	X	X	ALUCT1	Link
1	1	0	0	1	X	X	ALUCT1	Flag 1
1	1	0	1	0	X	X	ALUCT1	Flag 2
1	1	0	1	1	X	X	ALUCT1	Flag 3

X = DON'T-CARE
* = SPARE-NC
+ = Used for floating point operations

3.10 OPCODE PROGRAMMABLE READ-ONLY MEMORY (PROM)

The opcode PROM (Sheet 23) generates a 5-bit field used as control signals to the CCMUX and condition code register (CCR). The device is addressed via the S bus by the 8-bit opcode latch (see Section 3.6.1). The PROM outputs are stored in the opcode PROM register and clocked on the first microcycle of every user instruction.

The five opcode PROM outputs are divided and used as follows:

- Two bits (CH1 and CL1) are sent to the CCR and they select one of four conversion methods. See Section 4.6.
- Two bits (CCXH1 and CCXL1) are sent to the CCMUX and are the select inputs to the CCMUX extension, choosing one of four conditions for positions 14 and 15. If the final carry output comes from the shift register, then CCXH1 and CCXL1 also select one of three shift register carry outputs.
- The fifth bit selects where the final carry output comes from, either the shift register or the ALU.

CHAPTER 4
EXECUTION UNIT
THEORY OF OPERATION

4.1 INTRODUCTION

This chapter provides an analysis of the execution unit section of the Model 3205 System central processing unit (CPU) board (see Figure 4-1). The execution unit consists of the following units:

- Arithmetic logic unit (ALU)
- Register external (REX)
- Constant programmable read-only memory (PROM)
- Shift registers (SR0, SR1, SR2)
- Multiplier
- Condition code register (CCR)
- Program status word condition code (PSWCC)
- P register (PR)
- N register (NR)
- Direct memory address (DMA) word counter
- M register (MR)

The execution unit performs arithmetic and logical operations. It is also used for calculating the address of operands in memory. All user registers are located in the execution unit.

At the beginning of every user instruction, the currently selected general register set is in ALU register internal (RIN) and an exact copy resides in register external (REX).

While emulating the user instruction, either register set (RIN or REX) may be altered. Upon completion of the user instruction, the copy is guaranteed.

All schematic references (Sheet XX) refer to Functional Schematic 35-864 D08.

4.1.1 Execution Unit Timing

There are two types of timings described in this section.

- Y bus timing
  - Data is sourced to the ALU in the first half microcycle (t0-100ns) via the Y bus.
  - The ALU result is transferred to its destination in the last half microcycle (t100-200ns). This result is always available on the X bus.

NOTE

> ALU sources and destinations are listed in Tables 4-1 and 4-2.

- X bus timing
  - In the first half microcycle, REX or constant PROM source the X bus (see Section 4.3.3).
  - The ALU result always sources the X bus in the last half microcycle. (ALU result <= Y bus, X bus)

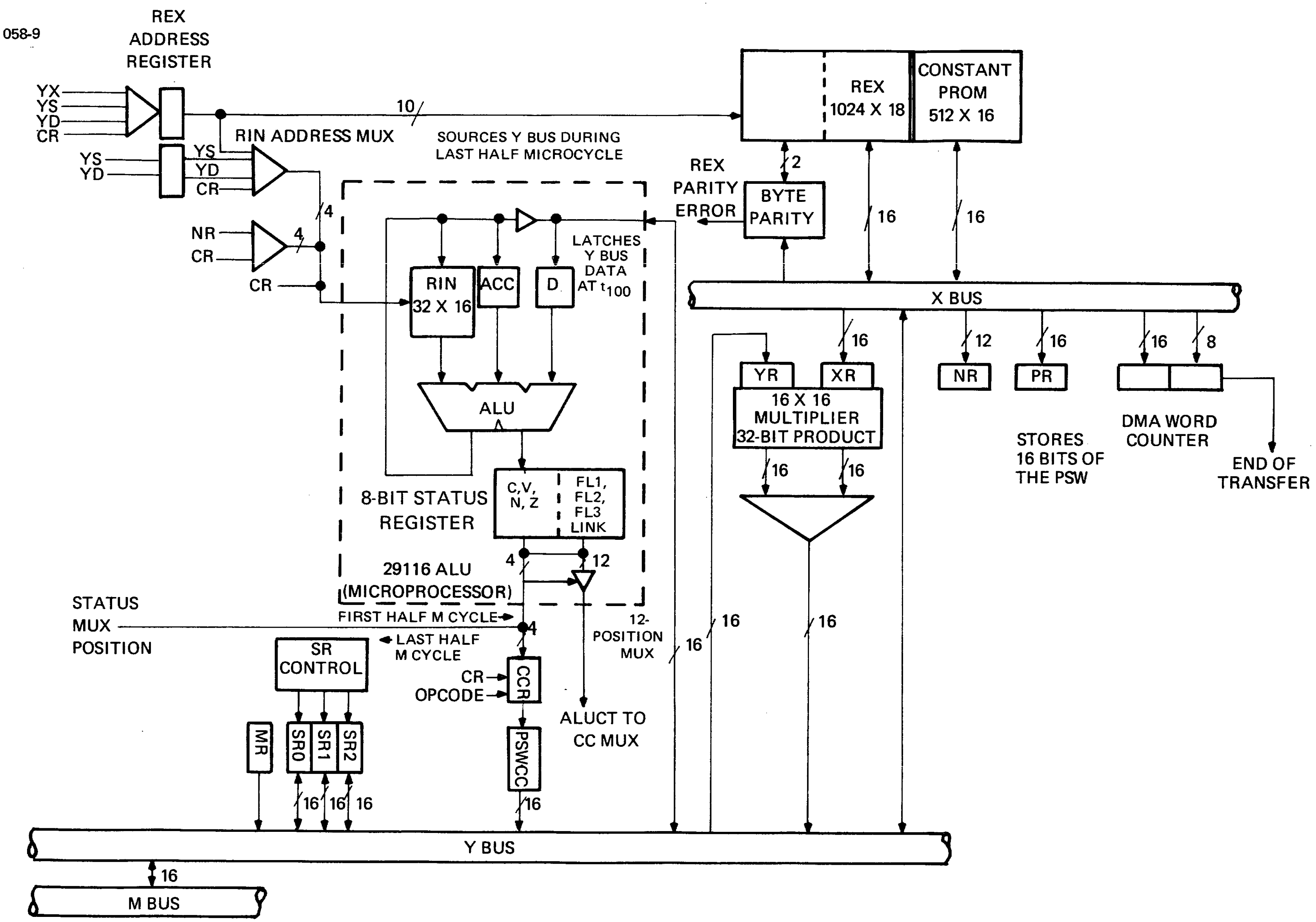


Figure 4-1 Execution Unit Block Diagram

TABLE 4-1 ALU SOURCES

FUNCTION	BUS	FUNCTIONAL UNIT
RIN (inside ALU)	-	Execution unit
ACC (inside ALU)	-	Execution unit
PSWCC	Y	Execution unit
PRODH	Y	Execution unit
PRODL	Y	Execution unit
MR	Y	Execution unit
SR0	Y	Execution unit
SR1	Y	Execution unit
SR2	Y	Execution unit
REX	X	Execution unit
Constant PROM	X	Execution unit
Memory data	M	Memory system
PC (address register)	M	Memory system
MAR (address register)	M	Memory system
DMAR (address register)	M	Memory system
AMAR (address register)	M	Memory system
Error logger	M	Memory system
MDR	M	Memory system
MUX	M	I/O system
PMUX	M	I/O system

TABLE 4-2 ALU DESTINATIONS

CATE-GORY	FUNCTION	BUS	FUNCTIONAL UNIT
1	RIN (inside ALU)	-	Execution unit
2	ACC (inside ALU)	-	Execution unit
3	SR0, SR1, SR2	Y	Execution unit
4	REX	X	Execution unit
5	NR, PR, DCTL, DCTH	X	Execution unit
	YR	Y	Execution unit
	PC, MAR, DMAR, AMAR, MEMORY	M	Memory
	MUX, PMUX	M	I/O

NOTES

1. One destination in each of the five categories may be selected within a microinstruction.

2. NULL is used when a destination is not specified.

4.2 ARITHMETIC LOGIC UNIT (ALU)

The 16-bit ALU (Sheet 12) (A184) provides complex arithmetic and logical operations. Both 16-bit (halfword) and 8-bit (byte) instructions are supported. All ALU results are presented to the Y bus in the last half microcycle, regardless if the ALU instruction specifies the Y bus or not. The most significant eight-bits of the Y bus are undetermined for ALU byte instructions. The device consists of the following major elements (see Figure 4-2).

- Register internal (RIN)

  RIN is a 32-word by 16-bit internal single port random access memory (RAM) with a 16-bit latch on its output. For byte instructions, only the lower eight bits are written to; for halfword instructions, all 16 bits are written to. With the use of the RIN address multiplexor (MUX) (see Section 4.2.2), it is possible to select separate read and write RIN addresses for the same instruction.

- Accumulator

  The 16-bit accumulator is an edge-triggered register. For byte instructions, only the lower eight bits of the accumulator are written to. For halfword instructions, all 16 bits are written to.

- Data latch

  The 16-bit data latch holds the Y bus data being sourced to the ALU. The Y bus data is always latched midway through every microcycle.

- Barrel rotator

  The barrel rotator is used as one of the ALU inputs. This permits rotating data up to 15 positions from either the RIN, accumulator or the data latch. In the halfword mode, the barrel rotator rotates a 16-bit word, and in the byte mode, rotates only the lower eight bits.

- Arithmetic logic unit (ALU)

  The ALU is capable of operating on either one, two or three operands, depending upon the instruction being executed. The ALU has the ability to execute all conventional one and two operand operations such as move, complement, two's complement, add, subtract, increment, AND, NAND, OR, NOR, EXOR and EXNOR. In addition, the ALU can also execute three operand instructions such as rotate/merge and rotate/compare with a mask. All ALU operations can be performed on either a halfword or byte, with byte operations being performed on the lower eight bits.

  The carry input to the ALU is generated by the carry MUX, which can select an input of 0, 1 or the stored carry bit from the ALU status register. Using the stored carry bit as the carry input allows execution of multiprecision addition and subtraction.

- Priority encoder

  The priority encoder produces a binary-weighted code to indicate the location of the highest order one at its input. The input to the priority encoder is generated by the ALU, which performs an AND operation on the operand to be prioritized, and a mask. The mask determines which bit locations to eliminate for prioritization.

- Eight-bit status register

  The status register is updated at the end of all instructions except NO-OP, Save-Status and Test-Status instructions. The lower four bits of the status register contain the ALU status bits of zero (Z), carry (C), negative (N) and overflow (OVR). The upper four bits contain a link bit and three definable status bits (flag 1, flag 2 and flag 3).

  The lower four status bits are updated after each instruction except those mentioned previously (NO-OP, Save-Status, Test-Status) and Status Set/Reset instruction for the upper four bits. The upper four status bits are changed only during their respective Status Set/Reset instructions and during Status Load instructions in the halfword mode. The link-status bit is also updated after each shift instruction.

  The status register can be loaded from the internal Y bus and can also be selected as a source to the internal Y bus. When the status register is loaded in the halfword mode, all eight bits are updated; in the byte mode, only the lower four bits (Z, C, N, OVR) are updated.

- Twelve inputs to the CCMUX

  Twelve inputs to the CCMUX (see Section 3.9) originate from the ALU and are used for conditional microbranching. These inputs consist of various combinations of the internal 8-bit status register.

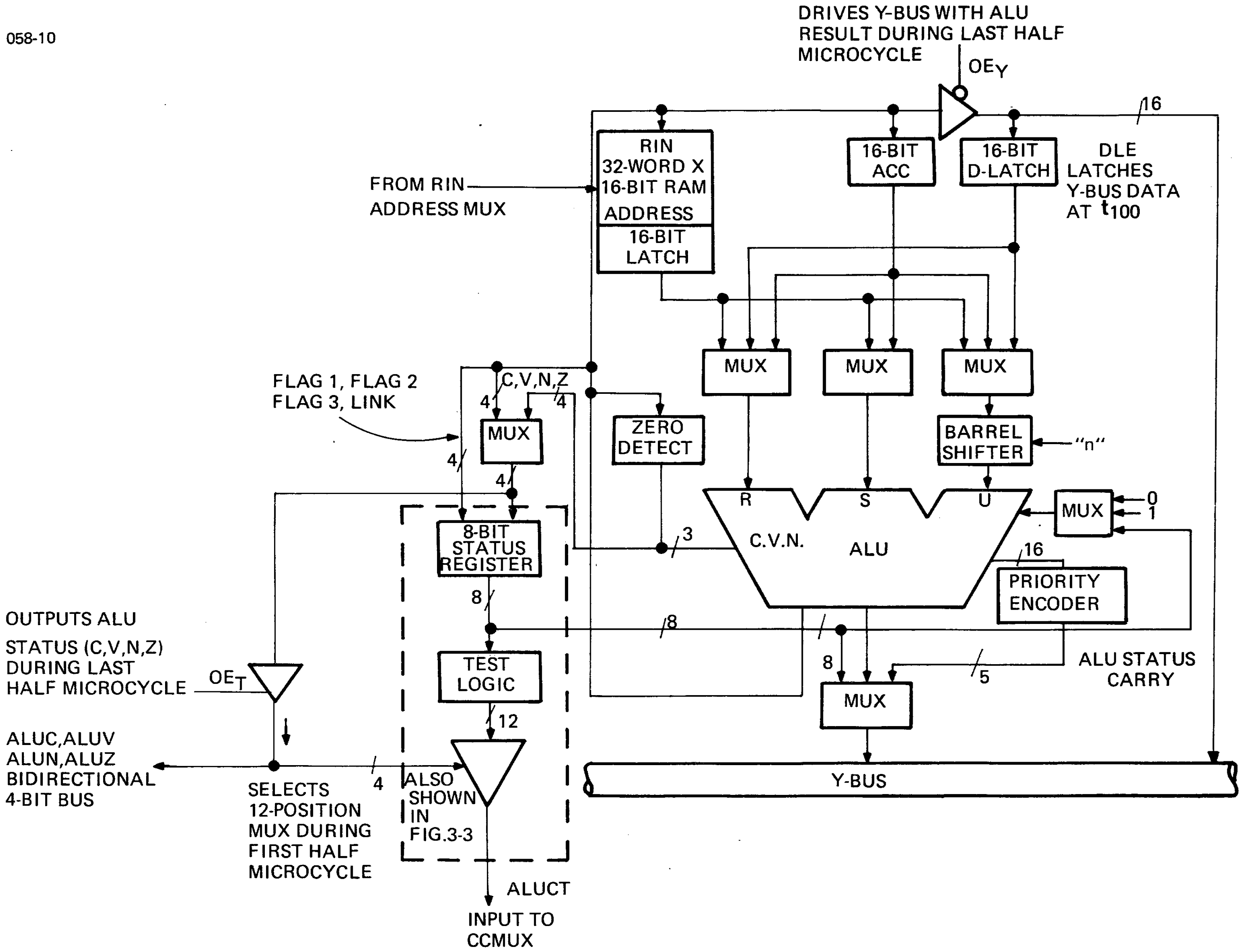


Figure 4-2 ALU

4.2.1 Arithmetic Logic Unit (ALU) Instructions

The ALUs powerful instruction repertoire includes many instructions that are not traditionally performed by an ALU. These instructions are: Rotate (up to 15 bits) and Merge, Rotate and Compare, Shift (one bit), Prioritize, Status and Complex bit-oriented instructions.

The following ALU instructions do not use the "N" field as data.

Move
Complement (one's complement)
Negate (two's complement)
Increment
Add
Add with Carry
Subtract
Subtract with Carry
Logical AND
Logical NAND
Logical OR
Logical NOR
Logical EXOR
Logical EXNOR
Shift Up One Bit (programmable carry-in)
Shift Down One Bit (programmable carry-in)
Prioritize
CRC Generation
Set Status (C, V, N, Z, flag 1, flag 2, flag 3 and link)
Reset Status (C, V, N, Z, flag 1, flag 2, flag 3 and link)

The following ALU instructions use the "N" field as data.

Set Bit 2^n
Reset Bit 2^n
Add 2^n
Subtract 2^n
Rotate Left n
Rotate/Merge n
Rotate/Compare n
Test Bit 2^n

NOTE

"n" ranges from 0 to 15.

The 16-bit ALU instruction (I15:00) is divided into the four fields (F0, F1, N and F2) listed in Table 4-3 (see Section 4.2.1.1).

TABLE 4-3 ALU INSTRUCTION BIT DEFINITIONS

	ALU INSTRUCTION		
SOURCE	FIELD	BIT	DEFINITION
RXA[03:00] or YS Field or YD Field or CR 47:44	F0	03:00	Specifies RIN address if instruction uses RIN; otherwise, it is part of the ALU instruction.
CR 45	F0	04	Specifies RIN address if instruction uses RIN; otherwise, it is part of the ALU instruction.
CR[52:49]	F1	08:05	Part of the ALU instruction
NR[03:00] or CR[56:53]	N	12:09	Part of the ALU instruction (sometimes referred to as the "N" field).
CR[58:57]	F2	14:13	Part of the ALU instruction
CR 59	F2	15	Part of the ALU instruction byte/halfword select: 0=8-bit ALU operation 1=16-bit ALU operation

4.2.1.1 Arithmetic Logic Unit (ALU) Instruction Format

The 16-bit ALU instruction (I15:00) is created with the four fields defined below.

```
 15     13 12          9 8            4 3     0
 -------------------------------------------
|   F2    |      N      |      F1      |  F0  |
 -------------------------------------------
```

These four fields are used to specify the type of instruction. The following examples illustrate how these fields are used.

Example:

increment accumulator

```
     15                                          0
     --------------------------------------------
    |                                            |
    |               INSTRUCTION                  |
     --------------------------------------------
F0 )
F1  >         specifies increment accumulator.
N   |
F2 )
```

Example:

increment RIN

```
     15                               5 4        0
     --------------------------------------------
    |                                  |   RIN   |
    |           INSTRUCTION            | ADDRESS |
     --------------------------------------------

F0            specifies RIN address.
F1 )
N   >         specifies increment RIN.
F2 )
```

Example:

rotate accumulator "N" bits

```
     15     13 12          9 8                   0
     --------------------------------------------
    |INSTRUC-|            |                      |
    |TION    |  "N" DATA  |     INSTRUCTION      |
     --------------------------------------------

F0 )
F1  >         specifies rotate accumulator.
F2 )
N             specifies rotate amount.
```

Example:

rotate RIN "N" bits

15 13	12 9	8 4	3 0
INSTRUCTION	"N" DATA	INSTRUCTION	RIN ADDRESS

F0 specifies RIN address.

F1, F2 specifies rotate RIN.

N specifies rotate amount.

ALU instruction bits 6 and 7 may be altered when performing signed division (see Section 4.5.2.1).

4.2.2 Register Internal (RIN) Address Multiplexor (MUX)

The RIN address MUX (Sheet 19) selects one of four sources to the ALU instruction bits I03:00. This field specifies a RIN address when the ALU instruction uses RIN as a source or destination (position 0, 1 or 2). Otherwise, this field is a part of the instruction.

TABLE 4-4 ALU RIN ADDRESS SELECTION

CR BIT 43:42	SELECTION DESCRIPTION
0 0	Output of REX address MUX. This is used when writing an ALU result into both RIN and REX.
0 1	YS Field user source IRH[03:00]
1 0	YD Field user destination IRH[07:04]
1 1	CR[19:16] standard ALU instruction bits

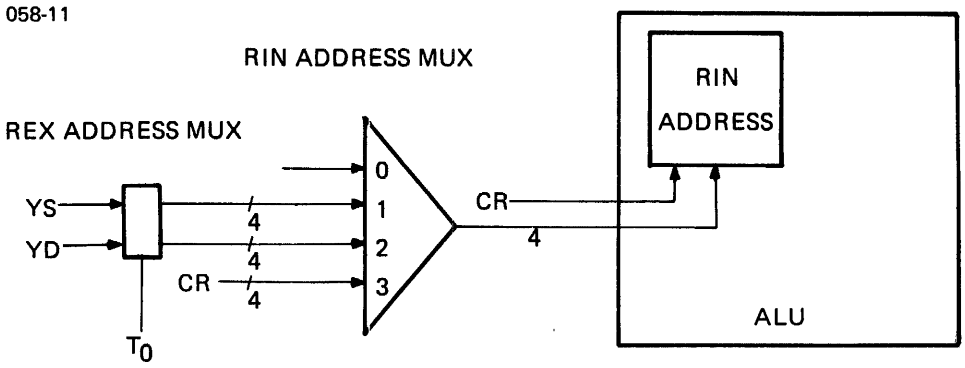


Figure 4-3 RIN Address MUX

The RIN source address may differ from the RIN destination address. This is referred to as double addressing RIN.

Double addressing RIN occurs when the microinstruction specifies:

- RIN address MUX position "1" (YS-user source field)
- REX address MUX position "2" (YD-user destination field)

DOUBLE ADDRESSING RIN TIMING

- YS specifies RIN address (source) in first half microcycle
- YD specifies RIN address (destination) in last half microcycle

Example:

```
    double addressing RIN

   REX (destination) + RIN (source)  ---->   RIN (destination)

|  REX address MUX   |  RIN address MUX |   REX address
|  position 2           position 1      |   position 0
|                                       |
|      First-half microcycle            | Last-half microcycle
```

4.3 REGISTER EXTERNAL (REX) and CONSTANT PROGRAMMABLE READ-ONLY MEMORY (PROM)

REX and constant PROM (Sheet 18) are described in the following sections.

4.3.1 Register External (REX)

The REX is a 1,024 x 20-bit RAM. REX contains general registers, single precision floating point registers, double precision floating point registers, scratchpad registers and microregisters (see Table 4-5).

TABLE 4-5 REX REGISTER SETS

QUANTITY	TYPE OF REGISTER	NUMBER OF LOC-ATIONS	ADDRESS (HEX)
8 Sets	16 general registers (32-bit)	256	000-FFF
8	Single precision registers (32-bit)	32	200-11F
8	Double precision registers (64-bit)	32	300-21F
32	Scratchpad ("single") registers (16-bit)	32	220-23F
32	Scratchpad ("double") registers (16-bit)	32	320-33F
640	Standard microregisters (16-bit)	640	100-1FF 240-2FF 340-3FF

REX is comprised of five 1,024 x 4 RAMs (A250, A251, A252, A253, A254) (Sheet 18) to form a 1,024 x 20-bit register. The 20-bit width consists of 16-bit data, 2-bit byte parity and the remaining 2-bits are unused. REX is addressed by a 10-bit address (RXA 91:01) produced by the REX constant PROM address register. REX is connected to the X bus and can source either the ALU or the multipliers X input register (XR).

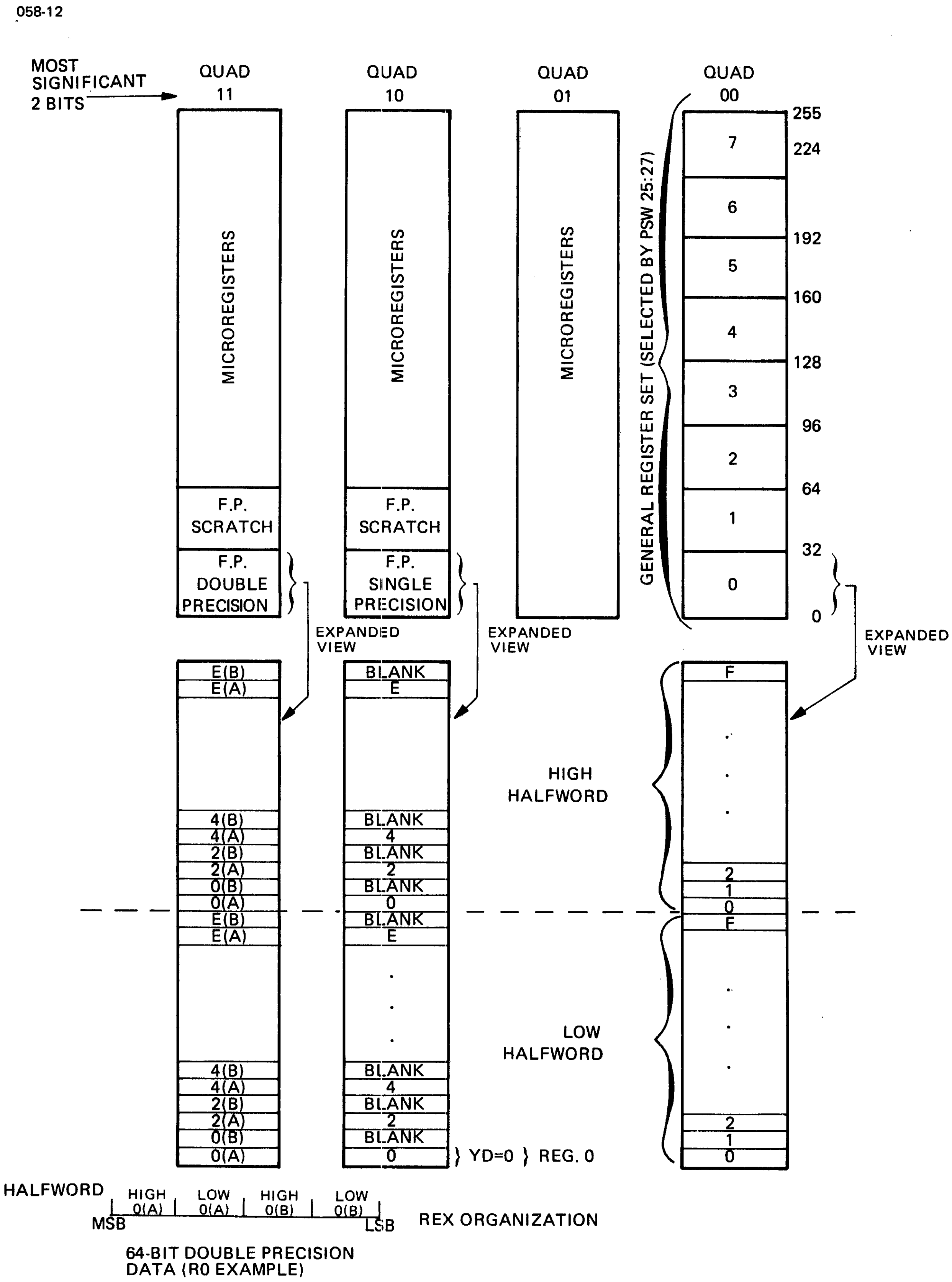


Figure 4-4 REX Organization

4.3.2 Constant Programmable Read-Only Memory (PROM)

The constant PROM stores 512, 16-bit data constants, which are used in various processor functions. The constant PROM is comprised of two 512 x 8 PROMs (A187, A217) (Sheet 18). The constant PROM is addressed by a 9-bit address (RXA 81:01) produced by the REX constant PROM address register. The 16-bit constant PROM output can source either the ALU or the multipliers X input register (XR).

4.3.3 Register External (REX) and Constant Programmable Read-Only Memory (PROM) Address Register

The field RXA[9:0] is used to address REX and constant PROM. It is updated at the end of each microcycle.

TABLE 4-6 REX REGISTER SELECTION

	SPECIFY REGISTER					RXA ADDRESS			
						QUAD	REGISTER SET	HALF-WORD SELECT	
REX REGISTER	41 40	39 38	20	11 10	DBL	9:8	7:5	4	3:0
General	0 0	X X	0	X X	X	0 0	PSW27:25	CS22	YX3:0
	0 1	X X	0	X X	X	0 0	PSW27:25	CS22	YS3:0
	1 0	X X	0	X X	X	0 0	PSW27:25	CS22	YD3:0
Single precision	0 0	X X	1	X X	0	1 0	0 0 0	CS22	YX3:0
	0 1	X X	1	X X	0	1 0	0 0 0	CS22	YX3:0
	1 0	X X	1	X X	0	1 0	0 0 0	CS22	YX3:0
Double precision	0 0	X X	1	X X	1	1 1	0 0 0	CS22	YX3:0
	0 1	X X	1	X X	1	1 1	0 0 0	CS22	YX3:0
	1 0	X X	1	X X	1	1 1	0 0 0	CS22	YX3:0
Single precision scratch-pad	0 0	1 0	1	X X	0	1 0	0 0 1	CS22	YX3:0
	0 1	1 0	1	X X	0	1 0	0 0 1	CS22	YX3:0
	1 0	1 0	1	X X	0	1 0	0 0 1	CS22	YX3:0
Double precision scratch-pad	0 0	1 0	1	X X	1	1 1	0 0 1	CS22	YX3:0
	0 1	1 0	1	X X	1	1 1	0 0 1	CS22	YX3:0
	1 0	1 0	1	X X	1	1 1	0 0 1	CS22	YX3:0
Micro	1 1	X X	X	0 1	X	CS9:8	CS7:5	CS4	CS3:0
Micro (YD)	1 1	X X	X	0 1	X	CS9:8	CS7:5	CS4	YD3:0
Constant PROM	1 1	X X	X	1 0	X	CS9:8	CS7:5	CS4	CS3:0
Constant PROM (YD)	1 1	X X	X	1 1	X	CS9:8	CS7:5	CS4	YD3:0

TABLE 4-7 REX ADDRESS SOURCES

CS BIT 41:40	SOURCE	DEFINITION

0 0	YX/F	YX = User source index; IRL[11:08] (P.E. only)
		F = User base register; IRL[15:12] (CCM only)

0 1	YS	User source IRH[03:00]
1 0	YD	User destination IRH[07:04]
1 1	CS[03:00]	Selecting this position switches REX address directly from the CS as explained below.

CR36 is dedicated to controlling REX writes. When CR36 is a "0", the ALU results are written into REX at the end of the microcycle.

4.4 SHIFT REGISTERS (SR0, SR1 and SR2)

SR0, SR1 and SR2 (Sheet 13) are 16-bit shift registers that can be used as a contiguous 48-bit shift register or as three versatile 16-bit storage registers (see Figure 4-5).

One of the three SRs may source the Y bus, independent of the SR that is specified as a destination.

A 2-bit encoded field (CS63:62) determines one of three possible SR or REX sources to the Y bus (see Table 4-8).

TABLE 4-8 SR SOURCE TO Y BUS (FIRST-HALF MICROCYCLE)

CS63	CS62	Y BUS SOURCE
0	0	SR0
0	1	SR1
1	0	SR2
1	1	REX/PROM*

*The selection of REX or PROM is determined by REX constant PROM address register (Sheet 18) shown in Figure 4-4.

The signal NOS1 inhibits SR0, SR1, SR2, REX and constant PROM from sourcing the Y bus. It is active when any of the following devices are specified as a Y bus source.

MR
PRDL
PRDH
PSWCC
Error logger
MUX
PMUX
MDR
Address register (PC, MAR, DMAR or AMAR)*
Memory data*

*Both NOS1 and M2YN0 go active

A 2-bit encoded field (CR61:60) determines one of four possible SR destinations from the Y bus (see Table 4-9).

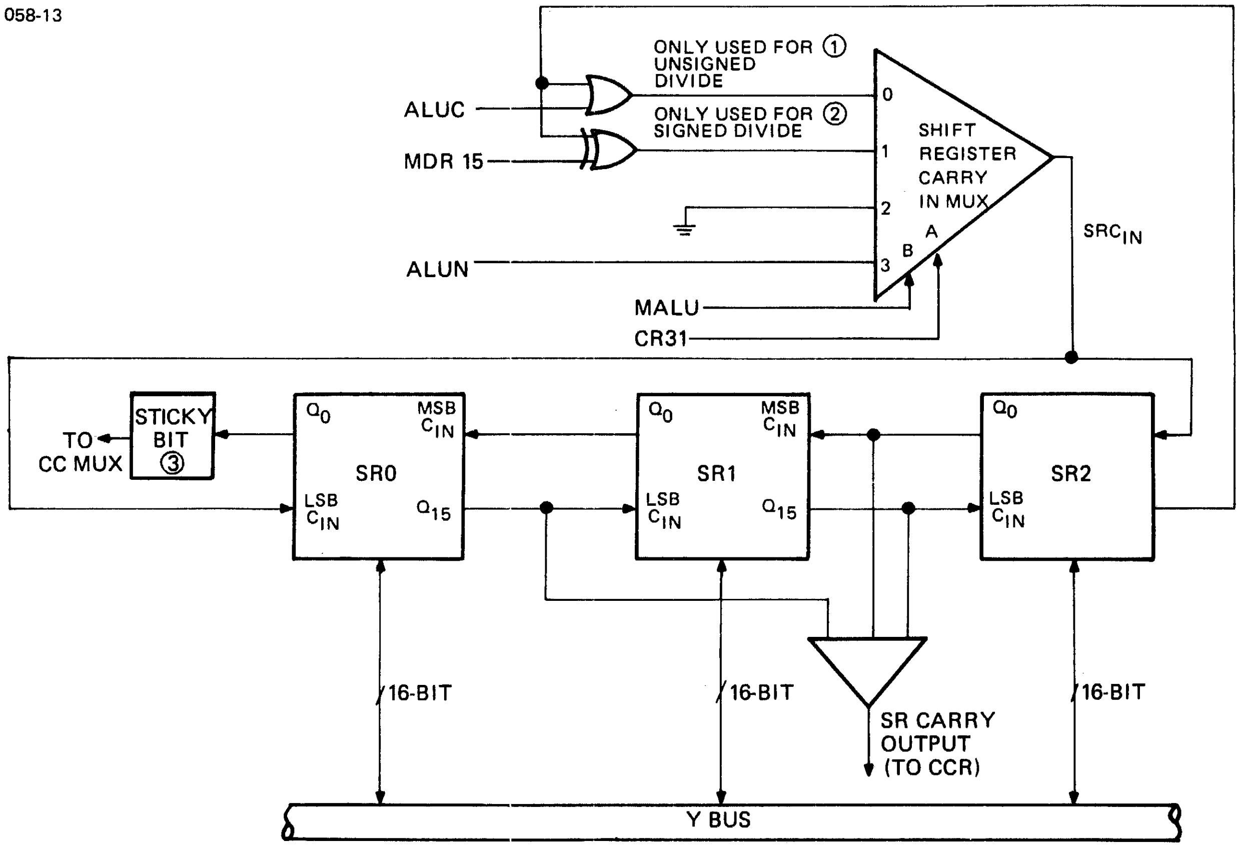
TABLE 4-9 Y BUS SOURCE TO SR

CR61	CR60	MEANING
0	0	Load SR0
0	1	Load SR1
1	0	Load SR2
1	1	Do not load any SR

Within one microcycle, the SRs can perform 1- or 4-bit shifts in either direction. Carry input is either zero, ALU negative status (ALUN) or specially derived through the divide instructions (see Section 4.5.2). Whenever a shift is specified, SR2, SR1 and SR0 are shifted together as a contiguous 48-bit shifter. Table 4-10 is SR operations and the CR bits that control them.

TABLE 4-10 SR OPERATIONS

CONTROL REGISTER BITS				
32	31:28	27:24	23	SR OPERATION
X	X 0 0 0	1 1 0 X	X	No Shift
X	X 0 1 0	1 1 0 X	0	Shift right one bit
X	X 1 0 0	1 1 0 X	0	Shift left one bit
X	X 1 1 0	1 1 0 X	X	Reset SR0, SR1 and SR2
X	X 0 1 0	1 1 0 X	1	Shift right four bits
X	X 1 0 0	1 1 0 X	1	Shift left four bits
0	0 X X 0	1 1 0 X	X	Zero carry input
0	1 X X 0	1 1 0 X	X	ALU negative status carry input
1	X X X X	X X X X	X	See Section 4.5.2 on division



① USED ONLY IN DOUBLE PRECISION DIVIDE AND CONVERT BINARY TO BCD.
② USED ONLY FOR INTEGER FULLWORD AND AFTERWORD DIVIDE.
③ USED ONLY FOR SINGLE AND DOUBLE PRECISION ADD/SUBTRACT.

Figure 4-5 Shift Registers

4.4.1 Sticky Bit

The sticky bit is used for routing the results of floating point add/subtract operations (user instructions; AE, AER, SE, SER, AD, ADR, SD and SDR). It consists of one flip-flop internal to the microsequencer instruction PAL (A147), which is reset when performing a user instruction decode. The signal STICKY1 becomes active when the shift registers (SR0, SR1 and SR2) perform a 4-bit right shift and a logical "1" is shifted out of SR0 (see Figure 4-5).

4.5 MULTIPLIER

The multiplier (Sheet 12) performs high-speed 16- x 16-bit multiplication of signed or unsigned numbers, which results in a 32-bit product. It is used for the following user instructions:

- MH, MHR (signed 16 x 16; 32-bit product)
- M, MR (signed 32 x 32; 64-bit product)
- ME, MER (unsigned 24 x 24; 48-bit product)
- MD, MDR (unsigned 56 x 56; 112-bit product)

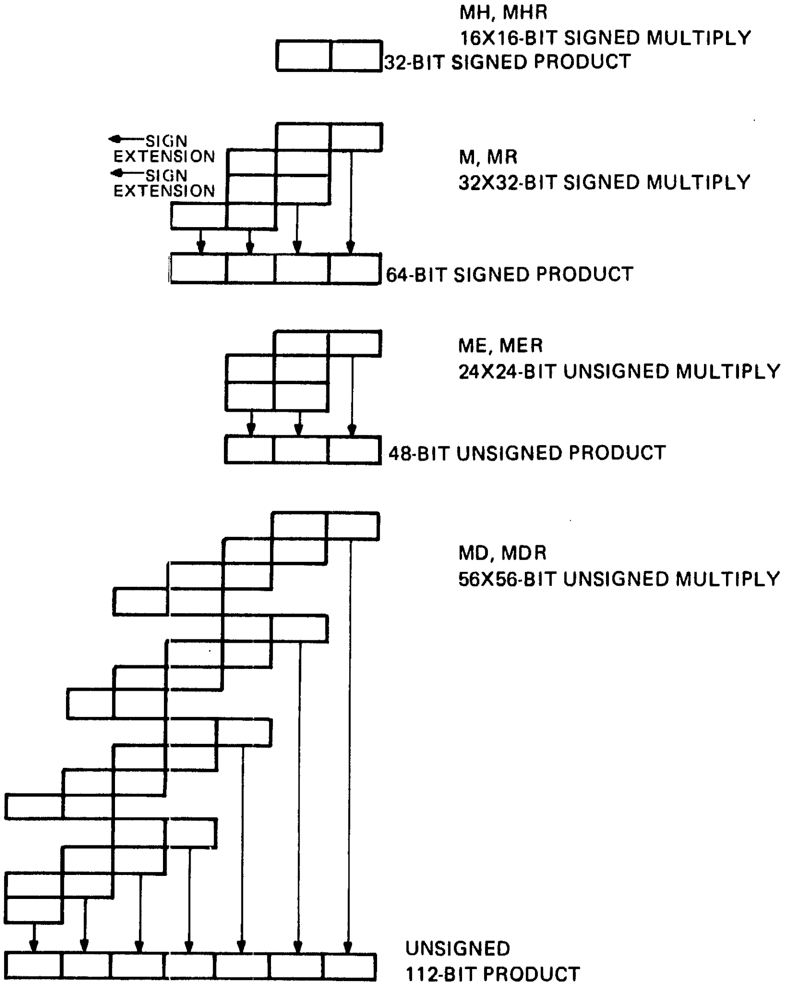
One of the operands is sourced from the X bus (usually from REX) and loaded into the multipliers X bus input register (XR) midway through the microcycle. The second operand is sourced from the ALU and is loaded into the multipliers Y bus input register (YR) at the end of the microcycle. Either one or both input registers may be loaded in one cycle. The product is available in the next microcycle.

A 16-bit output multiplexor selects either the most significant 16-bit product (PRDH) or the least significant 16-bit product (PRDL). These products only source the ALU. Either product (PRDL or PRDH) may be read during the same microcycle that the input registers are loaded. Table 4-11 lists the CR bits and the operation they control.

TABLE 4-11 MULTIPLY CONTROL FIELD

CONTROL REGISTER BITS				OPERATION
32	31:28	27:24	23	
X	X X 0 0	1 1 1 X	X	Unsigned MPY
X	X X 1 0	1 1 1 X	X	Signed MPY
X	1 X X 0	1 1 1 X	X	XR <= REX or Constant PROM
1	X X X 0	1 1 1 X	X	YR <= ALU
X	X X X 0	1 1 1 1	1	Y bus <= PRDL
X	X X X 0	1 1 1 0	1	Y bus <= PRDH

058-14



32-BIT PARTIAL PRODUCT ACCUMULATION FOR EXECUTING USER INSTRUCTIONS

Figure 4-6 32-Bit Multiplier Product Accumulation

4.5.1 Multiply Operations

The following examples display two multiplication operations.

Example:

16 x 16 **MULTIPLY**

	FIRST MICROCYCLE		SECOND MICROCYCLE		THIRD MICROCYCLE	
X BUS	Load XR From REX			Store PRDL in REX		Store PRDH in REX
Y BUS		Load YR from ALU	Read PRDL	Store PRDL in RIN	Read PRDH	Store PRDH in RIN

Example:

1 x 32-bit UNSIGNED MULTIPLY

```
                     16-BIT     32-BIT
                                                          ---------------
                      ---      --------         A x B     | PRDH  | PRDL  | 32-BIT PARTIAL PRODUCT
| INPUT DATA          | A |    | B    B |                 ---------------
|                     ---      --------
                                                   ---------------
|                     |                 A x B      | PRDH  | PRDL  |           32-BIT PARTIAL PRODUCT
|                     |                            ---------------
|                     |                        +
|                     |                     ---------------------------------
|                     |
|                     |                            -----------------------
|                     |                            |  P    |  P    |  P    | COMPLETE 48-BIT PRODUCT
|                     |                            -----------------------
```

	FIRST MICROCYCLE		SECOND MICROCYCLE		THIRD MICROCYCLE		FOURTH MICROCYCLE		FIFTH MICROCYCLE	
X BUS	Load B from REX			Store P in REX		Load P from REX		Store P in REX		Store P in REX
Y BUS		Load A from ALU	Read PRDL (P)	Store P in RIN	Read PRDL	Store in ACCUM.	Read PRDL	Store P (PRDL +ACCUM) in RIN	Read PRDH	Store P (PRDH +0+CAM) in RIN

4.5.2 Divide Operations

The execution unit performs signed and unsigned iterative division. The microinstruction used in the divide loop is specified by a MALU command. This command is used with a shift left (SR instruction) and subtract (ALU instruction).

4.5.2.1 Signed Division

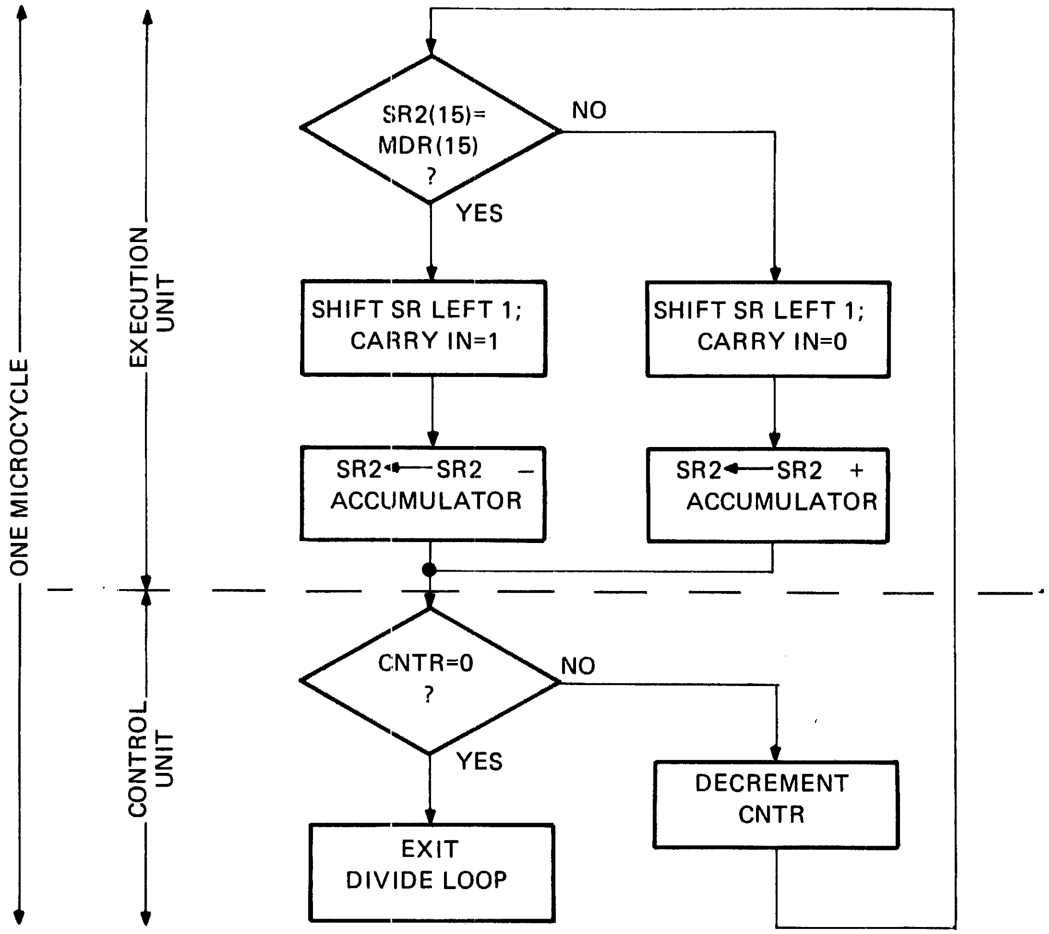
Signed division is used only for integer fullword and halfword divide operations (D, DR, DH, DHD).

Signed division is executed by performing an iterative left shift and add/subtract operation, based on a nonrestoring division algorithm. The shift carry-in and ALU operation (add or subtract) is determined by the sign of the divisor (MDR bit 15) and the sign of the partial remainder (SR2 bit 15).

Example:

Iterative signed division

058-15



The logic that makes these decisions is shown on Schematic Sheet 13

SRCIN1 = Shift register carry-in
I071 = ALU instruction bit 07
I061 = ALU instruction bit 06

The microinstruction specifies an ALU subtract (or subtract with carry) instruction; however, I[07:06] is modified to perform an add or subtract. The final quotient is found in SR0. The 16-bit remainder is found in SR2 (the additional 16-bit remainder is found in SR1, for 32-bit operations).

4.5.2.2 Unsigned Division

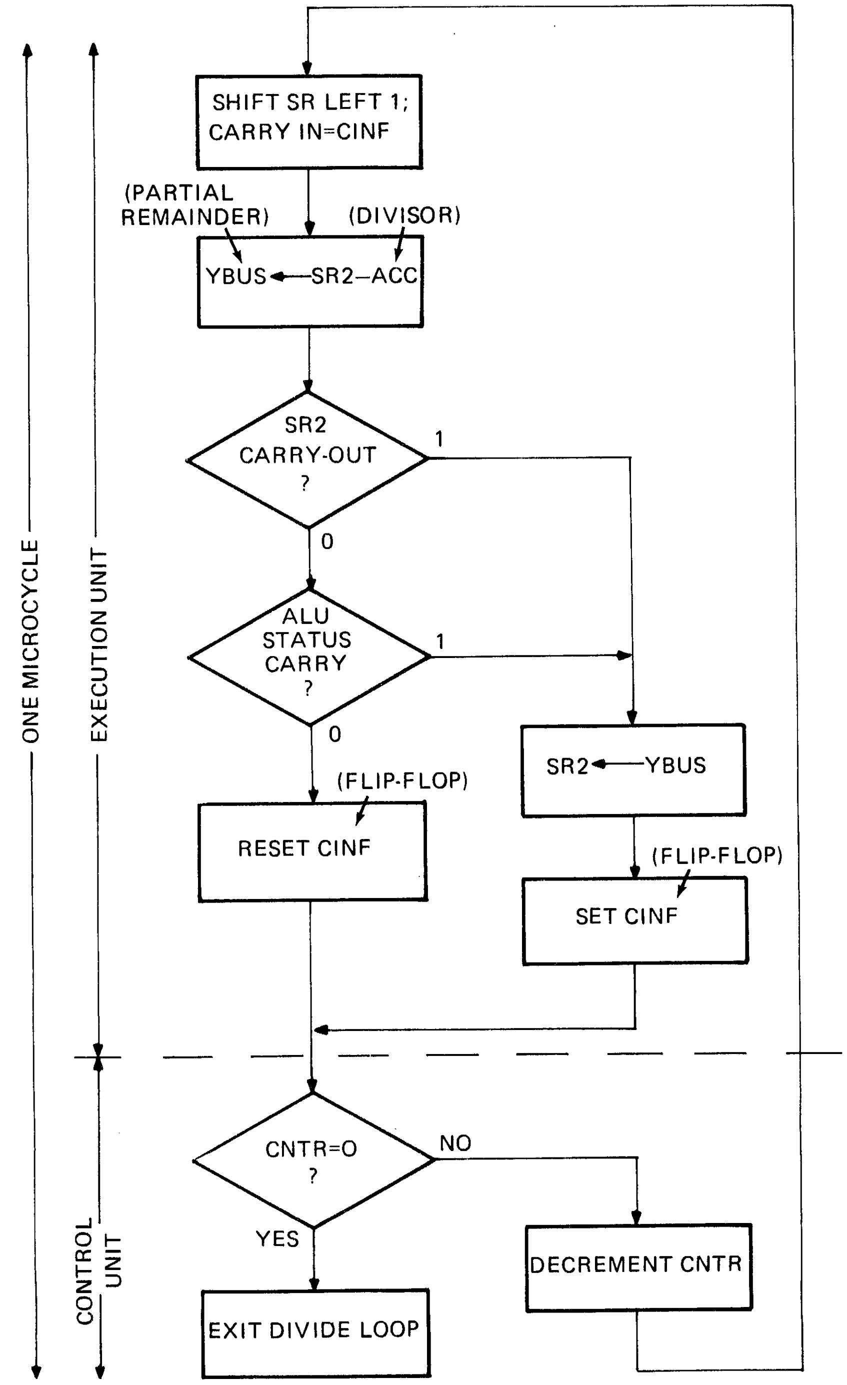
Unsigned division is used only for floating point division (DE, DER, DD, DDR) and Store Binary as Packed Decimal (STBP) User instructions.

Unsigned division is based on a simple, nonperforming division algorithm. This is an iterative shift left and subtract method. It is called nonperforming because the subtraction result (partial remainder) is conditionally loaded into SR2.

Example:

Iterative unsigned division

058-16



The shift register carry-in (SRCIN1) and SR2 load-disable (DISLD1) are determined by logic shown on Schematic Sheet 13.

The final quotient is found in SR0. The 16-bit remainder is found in SR2 and the additional 16-bit remainder is found in SR1 for 32-bit operations.

TABLE 4-12 SR AND ALU CONTROL FIELD

MIC FIELD				POSITION OF SR CARRY-IN MUX	SR2 LOAD	ALU INSTRUCTION	GENERAL OPERATION
33	31:28	27:24	23				
1	0 X X 0	1 1 0 1	0	0	CONDITIONAL	UNAFFECTED	UNSIGNED DIVISION
1	1 X X 0	1 1 0 1	0	1	UNAFFECTED	I07,I06 ALTERED	SIGNED DIVISION
0	0 X X 0	1 1 0 1	0	2 (Cin=0)	UNAFFECTED	UNAFFECTED	ALL OPERATIONS EXCEPT DIVISION
0	1 X X 0	1 1 0 1	0	3 (Cin=ALUN)	UNAFFECTED	UNAFFECTED	ALL OPERATIONS EXCEPT DIVISION

4.6 CONDITION CODE REGISTER (CCR)

The CCR (Sheet 23) is a registered PAL used for storing intermediate ALU status information. This status is used for creating the final 4-bit condition codes (Carry (C), Overflow (V), Greater (G) and Less Than (L)). Table 4-16 contains the CCR control field (CS35:33) and definitions of operations.

The CCR converts ALU status C, V, N and Z to Perkin-Elmer status C, V, G and L. Often, 32-bit status is derived from two 16-bit operations. The ALU zero status must reflect the entire 32-bits. This is accomplished by ANDing previous ALU zero status to the current ALU zero status.

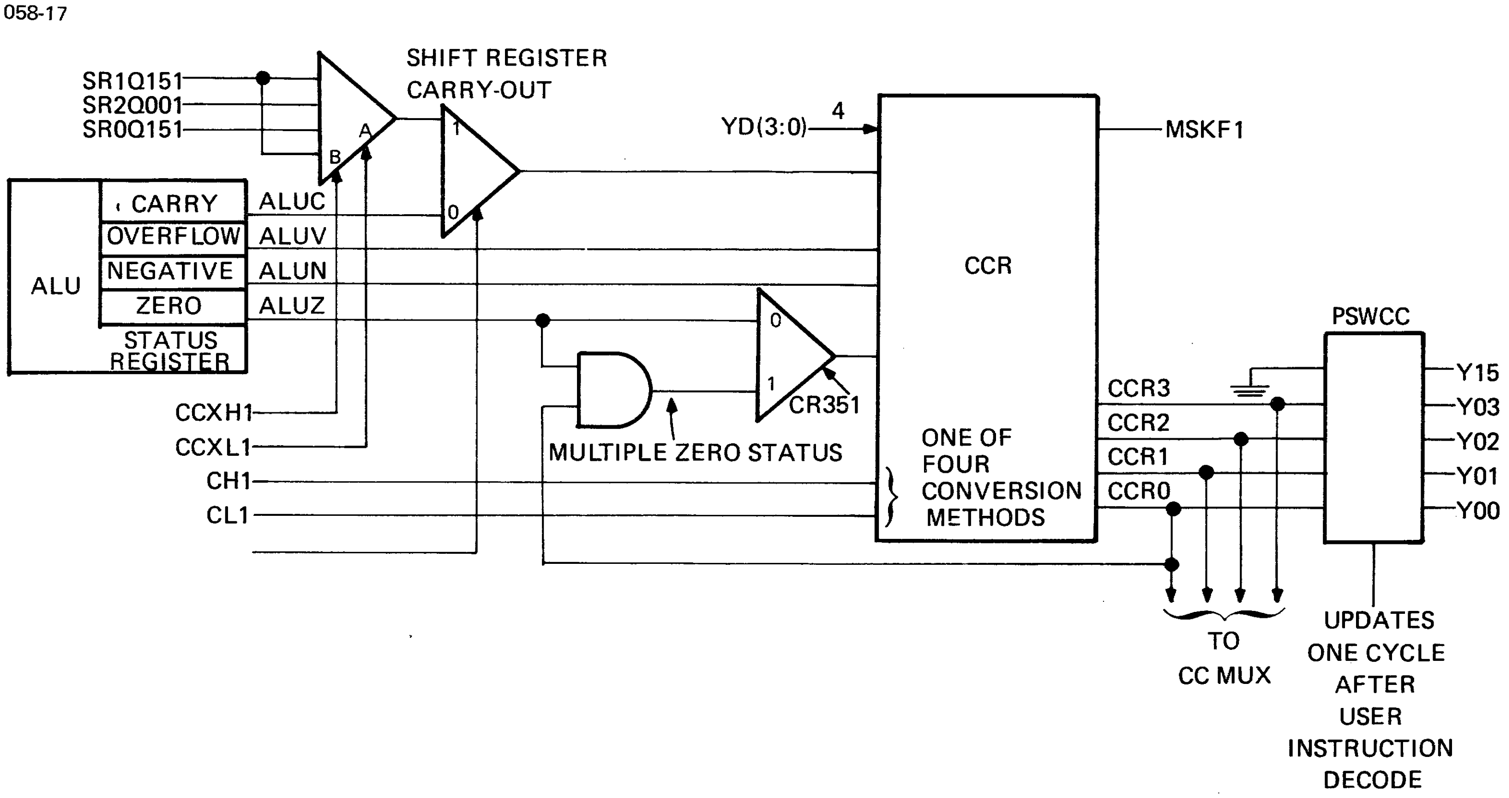


Figure 4-7 CCR

The PSW carry status originates from either the ALU or the SRs. This selection is done by the data from the opcode PROM (CCXH1 and CCXL1). PSW carry status originates from the SRs when executing User Shift instructions. This carry status originates from the ALU for all other user instructions (see Table 4-13).

TABLE 4-13 USER SHIFT INSTRUCTIONS

USER SHIFT INSTRUCTIONS		SOURCE OF PSW CARRY	CCXH1	CCXL1
DIRECTION	TYPE			
Shift Right	Halfword Logical	SR1 Q151	1	1
Shift Right	Fullword Logical	SR0 Q151	1	0
Shift Right	Halfword Arithmetic	SR1 Q151	1	1
Shift Right	Fullword Arithmetic	SR0 Q151	1	0
Shift Left	Halfword Logical	SR2 Q001	0	1
Shift Left	Fullword Logical	SR2 Q001	0	1
Shift Left	Halfword Arithmetic	SR1 Q151	0	0
Shift Left	Fullword Arithmetic	SR1 Q151	0	0
All other instructions		ALU Status Carry	-	-

The CCR can be loaded with the ALU status directly or it may be converted to reflect Perkin-Elmer status (see Table 4-14).

TABLE 4-14 CCR LOAD

SIGNAL	LOAD DIRECT	LOAD CONVERTED
CCR3	Overflow	Carry
CCR2	Negative	Overflow
CCR1	Carry	Greater than zero
CCR0	Zero	Less than zero

One of four conversion methods can be used when converting ALU status. The opcode PROM selects the conversion method using signals CH1 and CL1 (Sheet 23).

The CCR also makes branching decisions (see Table 4-15) for user branch instructions using signals MSKT1 and MSKF1 (MSKT1 is the inverse of MSKF1).

058-18

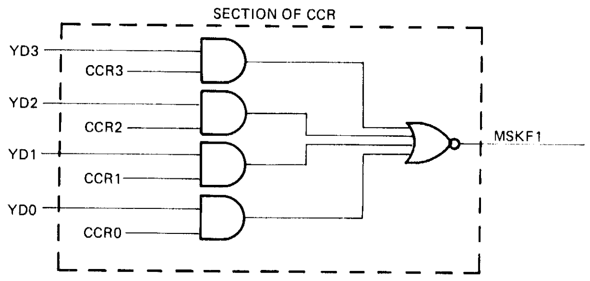


Figure 4-8 CCR MSKF1 Diagram

TABLE 4-15 CCR BRANCH

BRANCH TYPE	VALUE OF MSKT1	VALUE OF MSKF1	DECISION
TRUE	1	X	Branch
TRUE	0	X	No branch
FALSE	X	1	Branch
FALSE	X	0	No branch

TABLE 4-16 CCR CONTROL FIELD

CS BITS 33	34	35	DEFINITION
0	0	0	Load CCR using conversion type "A, B, C or D" multiple halfword
0	0	1	Load CCR using conversion type "A, B, C or D" single halfword
0	1	0	Load direct (multiple halfword)
0	1	1	Load direct (single halfword)
1	0	0	No change to CCR (default position)

TABLE 4-16 CCR CONTROL FIELD (Continued)

CS BITS 33 34 35	DEFINITION
1 0 1	No change to CCR (reset micro-interrupt level one)
1 1 0	Y bus <= PSWCC
1 1 1	Condition code multiplexor (CCMUX) extension

4.7 PROGRAM STATUS WORD CONDITION CODE (PSWCC)

The PSWCC (Sheet 23) is loaded with the final CCR data at the beginning of every user instruction. It always stores the current PSW condition code (least significant four bits of the PSW).

4.8 P REGISTER (PR)

The PR stores 16 bits of the 64-bit PSW necessary for continual operation of the system (see Table 4-17). For example, PSW bits 25:27 are required for determining the current general register set.

TABLE 4-17 P REGISTER CONTENTS

Y-BUS BIT	PSW BIT	DEFINITION
03:00	27:24	"R" general register set select field (LSB:MSB)
04	23	"P" protect mode
05	22	"Q" system queue service interrupt mask
06	21	"R/P" relocation/protect mask
07	20	"I" I/O interrupt mask
08	19	"FLU" floating point underflow mask

TABLE 4-17 P REGISTER CONTENTS (Continued)

Y-BUS BIT	PSW BIT	DEFINITION
09	18	"M" machine malfunction interrupt mask
10	17	"I" I/O interrupt mask
11	16	"W" wait state
12	13	"FLM" floating point instruction mask
13	12	Not used
15:14	11:10	"LVL" memory access levels

4.9 N REGISTER (NR)

The NR (Sheet 14) is a 12-bit register loaded from the X bus. It is used by the control unit (see Section 3.8) and the execution unit.

4.10 DIRECT MEMORY ACCESS (DMA) WORD COUNTER

The 24-bit DMA word counter (Sheet 14) determines the end of a DMA operation by counting the number of transfers. It consists of an 8-bit DMA word counter high (DCTH) and a 16-bit DMA word counter low (DCTL) to form a contiguous 24-bit counter. Each counter is loaded separately from the ALU via the X bus (see Section 5.3.1.6).

4.11 M REGISTER (MR)

The MR (Sheet 8) is a 16-bit memory status register attached to the Y bus in the execution unit (see Section 6.4).

CHAPTER 5
INPUT/OUTPUT (I/O) INTERFACE
THEORY OF OPERATION

5.1 INTRODUCTION

I/O operations provide a versatile means for exchange of data between the execution or memory unit and system devices. The I/O unit achieves this link by generating two standard Series 3200 buses, the multiplexor (MUX) bus and the private multiplexor (PMUX) bus (see Figure 5-1).

The MUX bus is used for data transfer between the execution unit and an I/O device. The data and handshaking signals on the MUX bus respond to commands from the control unit.

The PMUX bus is used for direct memory access (DMA) or as an extension of the MUX bus. In the DMA mode, the PMUX bus is controlled by the DMA sequencer. The DMA sequencer controls data flow and handshaking signals, leaving the control unit free to perform other processing functions. When the PMUX is not being used for DMA transfers, the sequencer is in the idle state and the PMUX bus emulates the protocols of the MUX bus responding to commands from the control unit. The I/O unit consists of the following sections:

- Bus structure
  - MUX bus
  - PMUX bus
  - M bus
- I/O hardware and control
  - Hardware
  - Control
  - DMA sequencer bus
- I/O bus operation
  - Writes
  - Reads
- Integrated selector channel (ISELCH)
  - Initialization
  - DMA block transfer
  - Termination

058-19

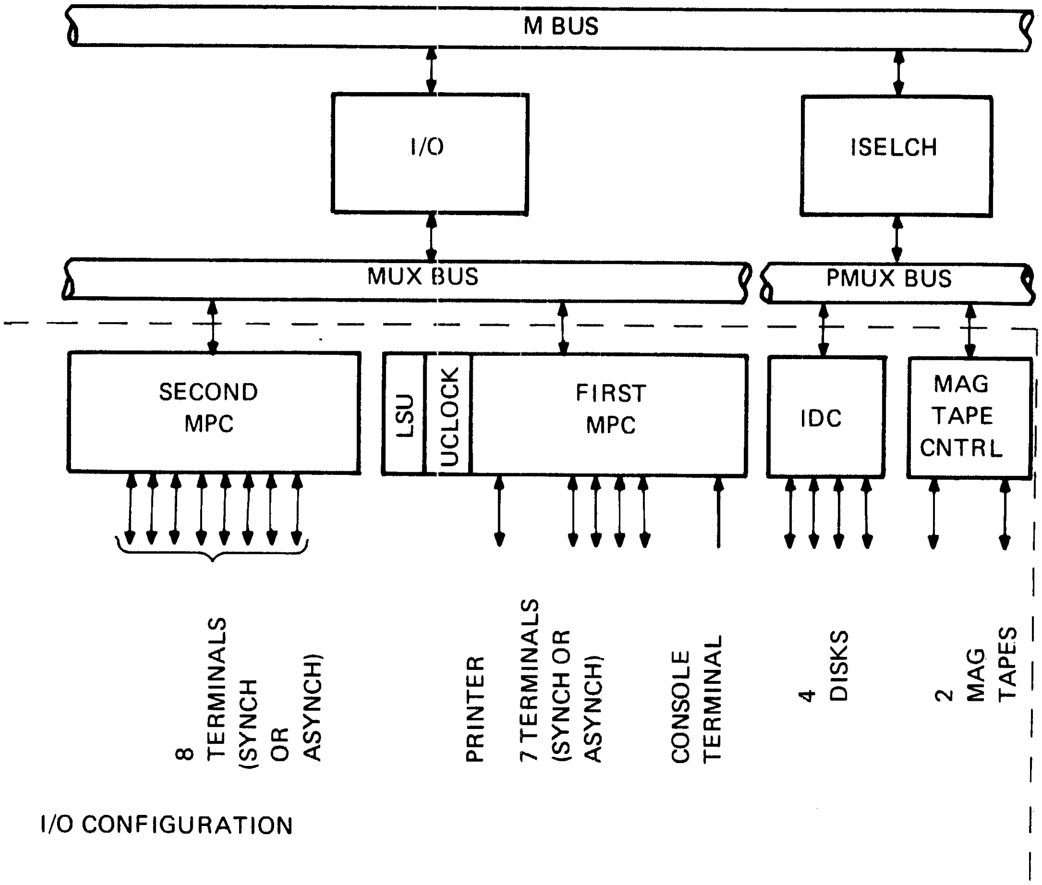


Figure 5-1 I/O Unit Block Diagram

5.2 BUS STRUCTURE

The following sections describe the MUX bus, PMUX bus and M bus structures.

The PMUX and the MUX buses operate in the same manner, except during block transfers when the PMUX bus disconnects from the MUX bus. The DMA sequencer then controls the PMUX operation until the transfer is complete. Both PMUX and MUX operations interface the other system units through the M bus. Detailed bus operations are described in Sections 5.2.1.1 and 5.2.2.1.

5.2.1 Multiplexor (MUX) Bus

The MUX bus is the primary data/control channel between the central processing unit (CPU) board and system devices. The CPU board initiates, monitors and responds to all system devices via MUX operation sequences. The MUX bus is a byte- or halfword-oriented I/O system that can address a maximum of 1,023 peripheral devices. The MUX bus consists of 27 signal lines: 16 bidirectional data lines, 7 control lines, 3 test lines and an initialize line, all on connector 0 of every backpanel slot.

5.2.1.1 Multiplexor (MUX) Bus Operations

There are six types of operations that occur on the MUX bus.

OPERATION	PURPOSE
Address	Selects the desired I/O device.
Command	Transfers a command byte from the CPU board to the selected I/O device. The command byte is the primary element for control of I/O devices by the CPU board.
Status	Transfers a status byte from the selected I/O device to the CPU board. The status byte is the primary element for interrogation and monitoring of the I/O device by the CPU board.
Data Available (data byte/halfword)	Transfers a data byte or halfword from the CPU board to the selected I/O device.
Data Request (data byte/halfword)	Transfers a data byte or halfword from the selected I/O device to the CPU board.
Interrupt Acknowledge	Transfers the address of an interrupting I/O device to the CPU board.

5.2.1.2 Multiplexor (MUX) Bus Signal Line Definitions

The 27 MUX bus signal lines are described in detail in the following paragraphs. Table 5-1 provides an overview.

TABLE 5-1 MUX BUS SIGNAL LINES

		DIRECTION	
TYPE	MNEMONIC	PROCESSOR <-> DEVICE	NUMBER
Data lines	D000:150	<-------------------->	16 lines
Control lines	ADRSO	--------------------->	1 line
	SRO	--------------------->	1 line
	DRO	--------------------->	1 line
	DAO	--------------------->	1 line
	CMDO	--------------------->	1 line
	RACKO/ TACKO	-----(daisy-chain)-->	1 line
	CLO70	--------------------->	1 line
Test lines	SYNO	<---------------------	1 line
	ATNO	<---------------------	1 line
	HWO	<---------------------	1 line
Initialize line	SCLRO	--------------------->	1 line

NOTE

All of the signal lines listed in Table 5-1 are applicable to the PMUX bus. Each signal mnemonic (except SCLRO) on the private bus has a P prefix for identification.

Signal line definitions:

The signal lines consist of the data and control lines. With the exception of the daisy-chained receive acknowledge/transmit acknowledge (RACKO/TACKO) line, all I/O bus signal lines are connected in parallel to all device controllers on the I/O bus.

Data lines:

- D150:D000

  The 16 low-active data lines (D150 through D000) are used to transfer, in parallel, a byte or a halfword of data between the CPU board and the device controller. In the case of a byte transfer, the data is transferred on the least significant eight data lines (D150 through D080). The MUX bus data lines occupy backpanel pins 111-0 through 218-0.

Control lines:

- ADRS0 (Address)

  This low-active control line is activated by the CPU board to all the device controllers. It is accompanied by an 8- or 10-bit device address on the data lines, to select one device controller for subsequent I/O transfers (backpanel pin 219-0).

- SR0 (Status Request)

  This low-active control line is activated by the CPU board to the previously selected device controller. The device controller gates its status byte onto the data lines (backpanel pin 219-0).

- DR0 (Data Request)

  This low-active control line is activated by the CPU board to the previously selected device controller. The device controller gates a byte or halfword of data onto the data lines (backpanel pin 120-0).

- DA0 (Data Available)

  This low-active control line is activated by the CPU board to the previously selected device controller, accompanied by a byte or halfword of data on the data lines (backpanel pin 221-0).

- CMD0 (Command)

  This low-active control line is activated by the CPU board to the previously selected device controller, accompanied by a command byte on the data lines (backpanel pin 220-0).

- RACK0/TACK0 (Receive Acknowledge/Transmit Acknowledge)

  This low-active control line is activated by the CPU board in response to signal ATN0 from a device controller. RACK0/TACK0 is propagated down the backpanel in serial "daisy-chain" fashion. The TACK0 from every board is connected to the RACK0 of the next board in the chain. The first I/O board receives TACK0 from the CPU board. If it has an interrupt pending, (ATN0 active), the board puts its device address on D000 through D150 and inhibits propagation of TACK0. If it has no interrupt pending, the first board sends TACK0 to the second board in the chain and the process is repeated. In this manner, I/O interrupt priority is maintained and bus contention problems are eliminated.

- CLO70 (Early Power Fail Warning)

  This low-active control line is activated by the CPU board to all the device controllers when a power fail condition is detected by the processor. This control line is held active until initialize (SCLRO) is activated (backpanel pin 121-0).

Test lines:

- SYNO (Synchronize)

  This low-active test line is activated by the device controller to the CPU board to inform the CPU that the device has properly recognized and responded to a control line signal. For an address operation, SYNO is only activated by the device controller being addressed. SYNO is not activated by any device controller in response to the CLO70 control line. For status request, data request, command and data available operations, SYNO is only activated by the previously selected device controller (backpanel pin 123-0).

- ATNO (Attention)

  This low-active test line is activated by any device controller to the CPU board to inform the CPU that an interrupt is pending. The device controller holds this test line active until it has received an interrupt acknowledge (RACKO) control line signal. Several device controllers may activate ATNO concurrently (backpanel pin 223-0).

- HWO (Halfword)

  This low-active test line is activated by a halfword-oriented device controller whenever it is selected. A byte-oriented device controller must not activate this test line (backpanel pin 226-0).

Initialize line:

- SCLRO (Initialize)

  This low-active line is activated during a system shutdown, power-up or initialization operation (backpanel pin 126-0).

5.2.1.3 Multiplexor (MUX) Bus Protocol

Communications over the I/O bus are performed on a request/response basis.

1. The I/O unit selects a device controller by placing a 10-bit address on the data lines and activating the address (ADRS0) control line. The device controller corresponding to the 10-bit address sets its address flip-flop and returns a SYN0 signal to the I/O unit. This controller is now selected. All other device controllers reset their address flip-flops to ensure that only one controller is selected at a time. Once a device controller is addressed, it remains so until another device is addressed or until the system is initialized. The selected device controller responds to all subsequent activity on the I/O bus until another controller is addressed.

2. The I/O unit issues the I/O instruction by placing the command on the data lines (D150:080) and activating the command (CMD0) control line. The device returns SYN0 to signify that it has received the command.

   If the I/O instruction involves transferring data from the I/O unit to the device controller, the I/O unit places data on the data lines and activates the data available (DA0) control line. The addressed device responds with a SYN0 after it has received the data. The I/O unit removes DA0 when it receives SYN0.

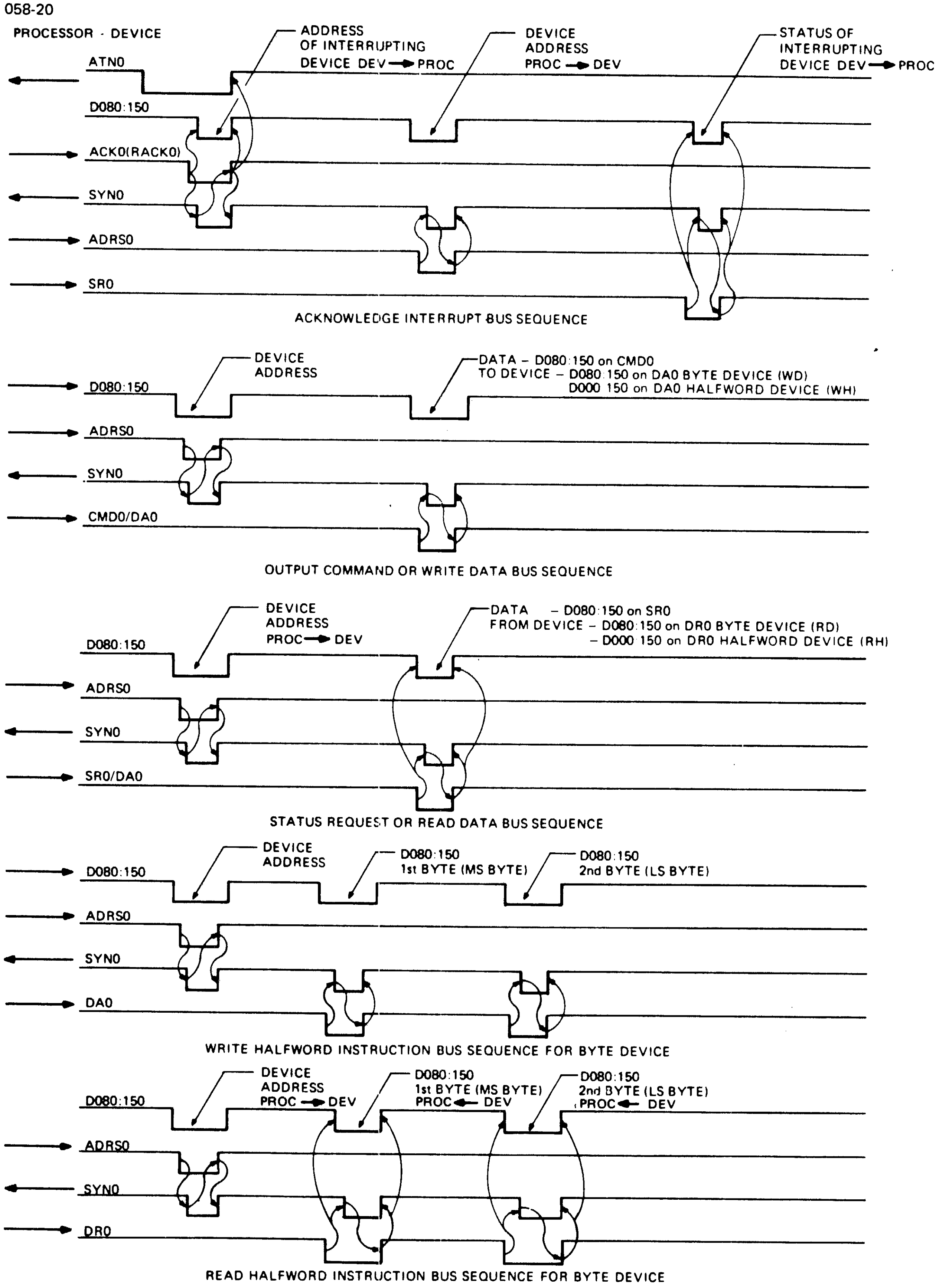
3. If the I/O instruction involves transferring data from the device controller to the I/O unit, the I/O unit activates the data request (DR0) control line. It then waits for the device controller to respond by placing a byte or halfword of data on the data lines and activating SYN0. When the I/O unit receives SYN0, it accepts the data and removes the DR0.

   In all cases, the device controller removes the SYNC (SYN0) whenever the I/O unit removes the control line.

NOTE

> In all cases, whenever an I/O unit control line (DR0, DA0,...etc.) is activated on the backpanel, the microaddress sequencer internal counter (described in Chapter 3) is loaded. The counter counts down and times-out after approximately 50μs, at which time the microcode vectors to the false sync routine and returns to the user.

Figure 5-2 illustrates the timing for the MUX bus. These timing diagrams apply for halfword (D000:150) or byte (D080:150) devices.



058-20
PROCESSOR - DEVICE
ATNO
D080:150
ACKO(RACKO)
SYNO
ADRSO
SRO
ADDRESS OF INTERRUPTING DEVICE DEV → PROC
DEVICE ADDRESS PROC → DEV
STATUS OF INTERRUPTING DEVICE DEV → PROC
D080:150
ADRSO
SYNO
CMDO/DAO
DEVICE ADDRESS
DATA – D080:150 on CMDO
TO DEVICE – D080:150 on DAO BYTE DEVICE (WD)
D000:150 on DAO HALFWORD DEVICE (WH)
D080:150
ADRSO
SYNO
SRO/DAO
DEVICE ADDRESS PROC → DEV
DATA – D080:150 on SRO
FROM DEVICE – D080:150 on DRO BYTE DEVICE (RD)
– D000:150 on DRO HALFWORD DEVICE (RH)
D080:150
ADRSO
SYNO
DAO
DEVICE ADDRESS
D080:150 1st BYTE (MS BYTE)
D080:150 2nd BYTE (LS BYTE)
D080:150
ADRSO
SYNO
DRO
DEVICE ADDRESS PROC → DEV
D080:150 1st BYTE (MS BYTE) PROC ← DEV
D080:150 2nd BYTE (LS BYTE) PROC ← DEV

ACKNOWLEDGE INTERRUPT BUS SEQUENCE

OUTPUT COMMAND OR WRITE DATA BUS SEQUENCE

STATUS REQUEST OR READ DATA BUS SEQUENCE

WRITE HALFWORD INSTRUCTION BUS SEQUENCE FOR BYTE DEVICE

READ HALFWORD INSTRUCTION BUS SEQUENCE FOR BYTE DEVICE

Figure 5-2 Bus Sequence for a Byte or Halfword Device

5.2.2 Private Multiplexor (PMUX) Bus

The PMUX bus provides a communication link between the I/O unit and the device controllers on the private side. When the ISELCH is idle (State 00), the PMUX bus performs regular MUX bus operations under microcode control between the on-board data latches and the system device controllers.

When the ISELCH is active, the PMUX is disconnected from the MUX bus during an ISELCH initialize operation (see Section 5.5.1). The PMUX is then controlled by the DMA sequencer and performs block data transfers with minimal execution unit interruption. The PMUX bus consists of 30 signal lines: 16 bidirectional data lines, 8 control lines, 5 test lines and an initialize line (SCRLO), all on connector 1 of every backpanel slot.

5.2.2.1 Private Multiplexor (PMUX) Bus Operations

When the ISELCH is idle and the PMUX and MUX buses are tied together, the PMUX performs all regular MUX bus operations outlined in Section 5.2.1.1.

When the ISELCH is busy and the PMUX and MUX buses are separate, the PMUX bus only performs the following operations:

- Status request
- Data available
- Data request

These three operations are described in Section 5.2.1.1.

5.2.2.2 Private Multiplexor (PMUX) Bus Signal Line Definitions

The PMUX bus signal line definitions are all identical to those of the MUX bus except for the addition of the P prefix to all signal names (example: ATN0 => PATN0). The PMUX bus has three additional lines connected in parallel to all devices on the PMUX.

Control line definitions:

- ISELCH Busy (SBUSY0)

  This low-active control line is activated by the ISELCH to the previously selected device controller to indicate that a ISELCH block data transfer is in progress. This line uses backpanel pin 224-1.

Test lines definitions:

- Switch to New Sequence (SENS0)

  This low-active test line is activated by the previously selected device controller to the ISELCH to specify that the controller supports the new protocol. It remains active at all times while the device controller is selected. This line uses backpanel pin 124-1.

- Status Check (SCHK0)

  This low-active test line is activated by the previously selected device controller to the ISELCH to indicate the occurrence of a bad status condition. Bad status is determined by the output of a NAND gate with inputs PD130, PD140 and PD150. Any time an input is active (low), a bad status condition exists. SCHK0 may be activated only while PDA0 or PDR0 is active. SCHK0 must be active at least 50ns before the device controller activates PSYN0. Once activated, it must remain active until after SBSY0 deactivates. This line uses backpanel pin 225-1.

  Signals PD120:150 perform multiple tasks on a DMA transfer.

- PD120 is the device controller busy bit. This bit is gated through the dual ranking circuitry and becomes signal PBZY1. The DMA sequencer checks PBZY1 before accessing the device controller.

- PD130:150 are the inputs to a NAND gate. The gate output is checked by the DMA sequencer on a STATUS REQUEST operation to the selected device. If BADSTATUS is indicated, the transfer enters the termination sequence (see Section 5.5.3).

5.2.2.3 Normal Integrated Selector Channel (ISELCH) Handshake Procedure

Normal ISELCH handshake I/O timing is shown in Figure 5-3. As the ISELCH transfer begins, the ISELCH activates private status request (PSR0) to the selected device controller. The controller responds by gating its status byte onto the PMUX bus data lines and then activating private SYNC return (PSYN0). BUSY (bit 12 of the status byte) controls the period between individual data transfers. BUSY is active at all times when the controller is not ready to transfer data. The ISELCH holds PSR0 active until PD120 deactivates (indicating the controller is now ready to transfer data); until PD130, PD140 or PD150 activate (indicating bad status); or until the processor halts the ISELCH.

After PD12O deactivates, a data transfer takes place. In read mode, private data request (PDRO) is activated and the controller gates a byte or halfword of data onto the PMUX bus. It then activates PSYNO. In the write mode, the byte or halfword of data is gated onto the private data lines by the ISELCH; private data available (PDAO) is then activated. The controller activates PSYNO after it accepts the data. In both read and write modes, the state of private halfword (PHWO) controls, whether a byte or halfword data transfer, takes place. The selected device controller activates PHWO if it is a halfword-oriented device. With PHWO active, the ISELCH transfers successive halfwords of data to or from the controller. With PHWO inactive, the ISELCH transfers successive bytes of data to or from the controller.

After the data transfer takes place, the ISELCH continues by repeating the status check/data transfer sequence until the memory data buffer is completed or the transfer is prematurely terminated due to bad controller status.

5.2.2.4 High-Speed Integrated Selector Channel (ISELCH) Protocol

The high-speed handshake protocol supports device controllers that sustain higher throughputs than are achievable using the normal procedure. This is achieved by eliminating the status check operation and streamlining the data transfer procedure. Table 5-2 shows the three additional control/test signals required for the high-speed protocol.

TABLE 5-2 HIGH-SPEED PROTOCOL SIGNAL LINES

TYPE	MNEMONIC	DIRECTION ISELCH <---> DEVICE	NUMBER
Control line	SBUSYO	---------------------->	1 line
Test lines	SNSO	<----------------------	1 line
	SCHKO	<----------------------	1 line

- Read

  Figure 5-3 shows the high-speed protocol data transfer timing. As a read mode, ISELCH transfer begins, the ISELCH activates private data request (PDRO). When the device controller is ready to transfer a byte or halfword of data (PD12O inactive), the controller gates the data onto the private ISELCH bus and then activates PSYNO. The ISELCH responds by accepting the data and deactivating PDRO, causing the controller to deactivate the private data lines and PSYNO. This handshake procedure is repeated until the data transfer terminates.

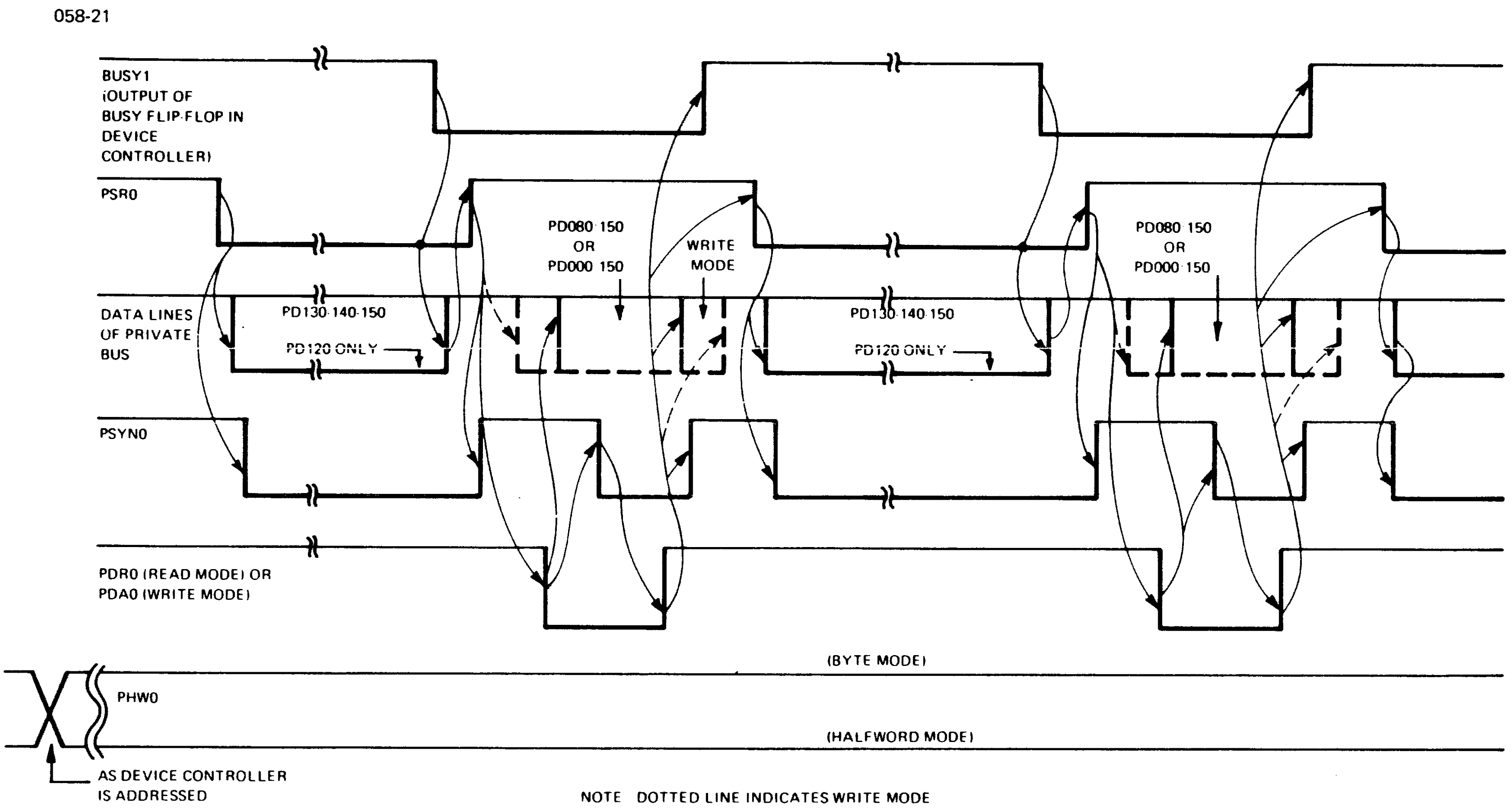


Figure 5-3 Normal ISELCH Handshake Timing

- Write

  As a write mode, ISELCH transfer begins, the ISELCH gating the data byte or halfword onto the private data lines and then activates private data available (PDA0). When the controller is ready to accept the data (PD120 inactive), the controller latches the data and then activates PSYN0. The ISELCH responds by deactivating PDA0, causing the controller to deactivate PSYN0. This handshake procedure is repeated until the data transfer terminates.

In both read and write modes, the device controller may terminate the data transfer by activating status check (SCHK0) after PDR0 or PDA0 is activated, but before PSYN0 is activated (see Figure 5-3). SCHK0 must remain active until the ISELCH deactivates ISELCH busy (SBUSY0). As in normal protocol, the private halfword (PHW0) line controls whether the data transfer is performed in byte or halfword mode.

5.2.3 M Bus Definition

Data on the M bus is valid at either 100ns or 200ns of a 200ns cycle depending on the source or destination of the data.

- If the ALU is a source to the M bus, the data is valid from 175ns to 200ns.

- If the M bus destination is the memory, the data is available from 175ns to 200ns.

- If the M bus destination is the ALU, the data is valid from 80ns to 100ns.

- If memory is a source to the M bus, the data is valid from 80ns to 100ns.

The control unit enables the data onto the M bus and latches data off the M bus by issuing the appropriate commands. The timing is derived from the system's free running clock.

5.3 INPUT/OUTPUT (I/O) HARDWARE AND CONTROL

The following sections describe the I/O hardware and control.

5.3.1 Input/Output (I/O) Unit Hardware

The I/O unit contains 11 sections. These sections are described below and are shown in Figure 5-4.

- MUX bus transceivers
- PMUX bus transceivers
- I/O test line synchronization
- MUX bus control signal generation
- PMUX bus control signal generation
- DMA word counter
- DMA address register/counter (DMAR)
- DMA sequencer
- Transceiver control PAL
- D to M bus enable PAL
- M to D bus enable PAL

5.3.1.1 Multiplexor (MUX) Bus Transceivers

The MUX bus bidirectional latches allow data to be latched from the M bus to the MUX bus or vice versa. The latching is always done in the halfword mode. When transferring data to a byte device, the control unit places the valid data on the least significant byte (LSB) on the M bus.

5.3.1.2 Private Multiplexor (PMUX) Bus Transceivers

When the ISELCH is in the idle mode, the PMUX bus latches work identical to the MUX bus latches. However, if the ISELCH is busy, the PMUX bus latches have the capability of converting two bytes to a halfword or vice versa. This is useful when the memory system (halfword) is communicating to a byte device.

5.3.1.3 Input/Output (I/O) Test Line Synchronization

The I/O test line synchronization synchronizes the asynchronous controller test lines of the MUX and PMUX buses to the synchronous operations of the control unit. This is done with the use of dual ranking circuitry. The test lines are clocked in the first stage at 150ns and in the second stage at 200ns of each period. This guarantees that all transitions occur at 200ns.

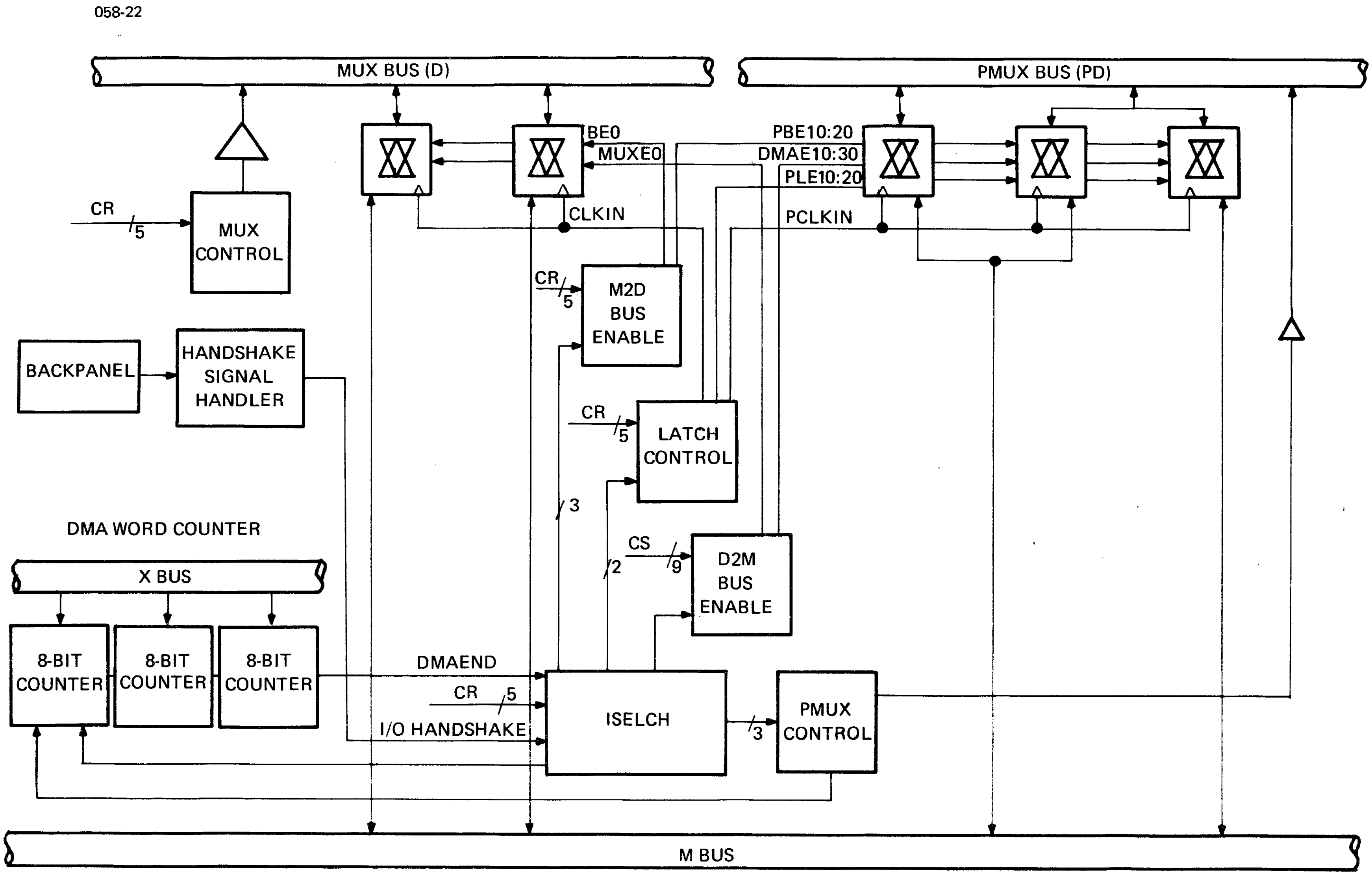


Figure 5-4 I/O Unit Hardware

5.3.1.4 Multiplexor (MUX) Bus Control Signal Generation PAL

The MUX bus control signal generation decodes the I/O commands coming from the control unit and activates the appropriate signal. These signals are outputs of a registered PAL circuit (see Table 5-3).

5.3.1.5 Private Multiplexor (PMUX) Bus Control Signal Generation PAL

The PMUX bus control signal generation works similarly to the MUX bus with the exception that it contains inputs from both the control store (CS) and the DMA sequencer. When the ISELCH is busy, the PMUX bus control signal generator responds to commands from the DMA sequencer (see Table 5-3).

5.3.1.6 Direct Memory Access (DMA) Word Counter

The DMA word counter controls the number of bytes transferred in a DMA block. This counter is loaded with the two's complement of the number of bytes or halfwords that are to be transferred. The counter increments each time data is transferred. The PMUX bus and the DMA sequencer stop transferring data when the DMA counters carry-bit goes active.

The DMA word counter is comprised of three 8-bit binary-up counters (Sheet 14), which form two registers. The DMA word counter-low (DCTL) is the least significant 16 bits and the DMA word counter-high (DCTH) is the most significant eight bits of the quantity of words to be transferred.

5.3.1.7 Direct Memory Access Address Register/Counter (DMAR)

The DMA address register points to the next memory location to be accessed. This register is part of the memory system but is used by the DMA logic. It is loaded with the starting address of the block to be transferred and increments with every transfer. For further information on DMAR control, see Section 6.3.

5.3.1.8 Direct Memory Access (DMA) Sequencer

The DMA sequencer is a state machine controlling the events of a DMA transfer and is implemented on a single programmable chip. When given a command GO (READ or WRITE), the sequencer steps through the states needed to complete the specified PMUX bus transfer. Information on the DMA sequencer states is in Table 5-6.

5.3.1.9 Transceiver Control PAL

The I/O unit latch control PAL generates the clock signals loading data from the M bus to MUX bus and PMUX bus transceivers. The PAL also generates the strobes latching data from the PMUX bus into the transceivers. Data is latched into the MUX bus transceivers by signal SYN0 (SYNC) returned from the device controller currently addressed when the data is valid.

Bytes are transferred from the PMUX to M bus by latching the first byte on the most significant M bus bits, then latching the second byte on the least significant M bus bits. Both bytes are then gated onto the M bus as a halfword. The PAL inputs are from the control register (CR) and the DMA sequencer.

5.3.1.10 D to M Bus Enable PAL

The D to M bus enable PAL generates the signals gating previously latched data from the transceivers to the M bus.

Signal DMAE30 is used only during a PMUX byte transfer and signal DMAE20 is used only during a PMUX halfword transfer. Inputs to this PAL are directly from the CS and the DMA sequencer.

5.3.1.11 M to D Bus Enable PAL

The M to D enable PAL generates the signals that gate previously loaded M bus data onto the PMUX or MUX bus. During a byte transfer, outgoing halfword data is gated onto the PMUX bus one byte at a time. Inputs to this PAL are from the DMA sequencer and CR.

5.3.2 Input/Output (I/O) Unit Control

The control of the I/O unit is performed by commands using CS bits 32:23 (see Table 5-3).

TABLE 5-3 I/O CONTROL FIELD

CONTROL REGISTER BITS					
32	31:28	27:24	23	COMMAND	DEFINITION
0	X 0 0 0	0 1 0 X	0	RST	Reset command line
0	X 0 0 0	0 1 0 X	1	SR	Status request
0	X 0 1 0	0 1 0 X	0	DA	Data available
0	X 0 1 0	0 1 0 X	1	DR	Data request
0	X 1 0 0	0 1 0 X	0	CMD	Command
0	X 1 0 0	0 1 0 X	1	TACK	Transmit acknowledge
0	X 1 1 0	0 1 0 X	0	ADRS	Address
0	X 1 1 0	0 1 0 X	1	RESERVED	PMUX status request
1	X 0 0 0	0 1 0 X	0	RESERVED	MAT write warning
1	X 0 0 0	0 1 0 X	1	M2Y-2	Y bus <= MUX
1	X 0 1 0	0 1 0 X	0	Y2M-2	MUX <= Y bus MUX bus <= MUX
1	X 0 1 0	0 1 0 X	1	RESERVED	CMND; bad status
1	X 1 0 0	0 1 0 X	0	GO WRT	
1	X 1 0 0	0 1 0 X	1	GO RD	
1	X 1 1 0	0 1 0 X	0	STOP	
1	X 1 0 1	0 1 0 X	0		M bus <= PMUX
1	X 1 0 1	0 1 0 X	1		PMUX <= M bus
0	X 0 0 1	0 1 0 X	0	RST PLE*	Reset latch enable
1	X 1 0 1	0 1 0 X	X	RST INT	Reset microinterrupt Level one

X = Don't-Care
* = Must be used following a DMA: M bus <= PMUX or PMUX bus <= PMUX.

5.3.3 Direct Memory Access (DMA) Sequencer Commands

The DMA sequencer generates an 8-bit command field that controls the ISELCH activity (see Tables 5-4 and 5-5). This sequencer uses inputs from the control unit, PMUX test lines and internal state counter to determine the next state of these outputs. The DMA sequencer is clocked every 100ns.

TABLE 5-4 DMA SEQUENCER COMMANDS

OUTPUT SIGNAL NAME	DEFINITION
DMAF1X0 DMAFX10	These two bits inform the PMUX control what operation to perform (see Table 5-5).
DMACNT0	Tells the PMUX transceiver PALS the type of operation in progress: 0 = first byte 1 = second byte or halfword
DMABST0	DMABDST0 indicates bad termination of a DMA transfer or that the control unit should now service the original termination, good or bad (see Section 3.7.1.5.)
DMAINT0	A microinterrupt that tells the control unit that the DMA state sequencer has come to a point where a memory operation (read or write) is necessary (see Section 3.5.1.6).
DMARD0	Informs the microinterrupt logic whether a DMAINT0 involves a read from or a write to memory (see Section 3.5.1.6).
DMABZY0	DMABZY0 is active whenever the DMA state sequencer leaves the idle mode.
ELFIN0	Indicates DMA transfer good termination (see Section 3.7.1.5).

TABLE 5-5 DMA CONTROL

DMAF1X0	DMAFX10	OPERATION
1	1	RESET
1	0	DA
0	1	DR
0	0	SR

5.4 INPUT/OUTPUT (I/O) BUS OPERATION

The following sections describe the I/O bus operations.

5.4.1 Input/Output (I/O) Writes (ADRS, DA, CMD)

In any of the I/O write operations the procedure is identical with the exception of which command line is activated. The steps for an I/O write are listed below.

1. The data is written into the I/O register.

   This is done with a MUX <= Y bus operation. This microcode command (MUX <= Y bus) latches data into the register that is the source of the I/O bus on all I/O writes. This command also enables the data on the backpanel. It should be noted that if the DMA is not busy, the identical data is driven out on the PMUX bus.

2. The I/O command line is activated.

   This is done by the control unit ADRS, DA or CMD commands. These microcode commands are decoded by PALs activating the appropriate I/O command line and this line remains active until a microcode RST CMND line is given. Once again, if the DMA is idle, the PMUX responds identically.

3. The data is available on the bus and waits for a SYNC.

   This waiting is done by sensing SYNC through the control units CCMUX. If the DMA is idle and SYNC is returned from the PMUX side, it is ORed into the CCMUX input. If DMA is busy, SYNC cannot be received from the PMUX side.

4. Once SYNC is acknowledged as active, a RST CMND line command is sent to the MUX.

   This command deactivates the command line (CMND, ADRS, DA) and removes data from the backpanel.

5.4.2 Input/Output (I/O) Reads (SR, DR, TACK)

The I/O read function is similar to the I/O write function with the exception of the command issued to activate the I/O command line. The steps necessary to perform an I/O read are listed below.

1. The CS issues the microcode commands necessary to perform the I/O read.

   If the DMA is idle, the same command line is activated on the PMUX bus.

2. The wait for SYNC, as in the I/O write, is done through the CCMUX.

   If the DMA is idle and the device returning SYNC is on the PMUX, it is ORed into the CCMUX.

3. Once SYNC is acknowledged, the data is read out of the I/O latches.

   This is performed by the microcode issuing a Y bus <= MUX command. If SYNC is acknowledged from the I/O bus, this command allows the data to be read from the I/O latches. If SYNC is not acknowledged on the I/O, but is on the P I/O, this command allows the data to be read from the PMUX bus.

4. Once the data has been read, a RST CMND line is issued.

   This command removes the I/O command line signal from the backpanel.

5.5 INTEGRATED SELECTOR CHANNEL (ISELCH)

The ISELCH has a private bus similar to the processor MUX bus. Controllers for the devices associated with the ISELCH are attached to this bus. When the ISELCH is idle, its private bus emulates the MUX bus. If this condition exists, the processor can address, command and accept interrupt requests from the devices attached to the ISELCH. When the ISELCH is busy, this connection is broken. All communication between the processor and devices on the ISELCH is cut off. Any attempt by the processor to address a device on this channel, when the channel is busy, results in instruction time-out.

The ISELCH has three processes: initialization, block transfer and termination.

5.5.1 Integrated Selector Channel (ISELCH) Initialization

During initialization, the ISELCH must be in the idle mode. All PMUX operations in this mode use the standard MUX bus protocol. Under program control, the ISELCH is addressed. The control unit recognizes the address (F0) and prepares to initialize the ISELCH. Under program control, the starting and final address of the block transfer is sent to the ISELCH. The control unit loads the starting address into the DMA address counter and the two's complement of the block size (block size = final address - starting address) in the DMA word counter. It should be noted that the program believes that the standard Series 3200 SELCH hardware is on the MUX bus. The control unit must decode all commands, data transfers and status requests to the virtual SELCH hardware and initiate the appropriate action in the ISELCH. At this time, a command GO READ or WRITE instruction is executed and the DMA sequencer takes control of the PMUX bus. The ISELCH is no longer idle and initialization is complete.

5.5.2 Integrated Selector Channel (ISELCH) Block Transfer

The operation of the DMA sequencer is illustrated in three ways. The state sequencer diagram (Figure 5-5) gives an overview of the various states and possible paths through them, six individual flowcharts depict all possible ISELCH operations; Table 5-6 describes each numbered state in detail. All three are cross-referenced to each other.

058-23

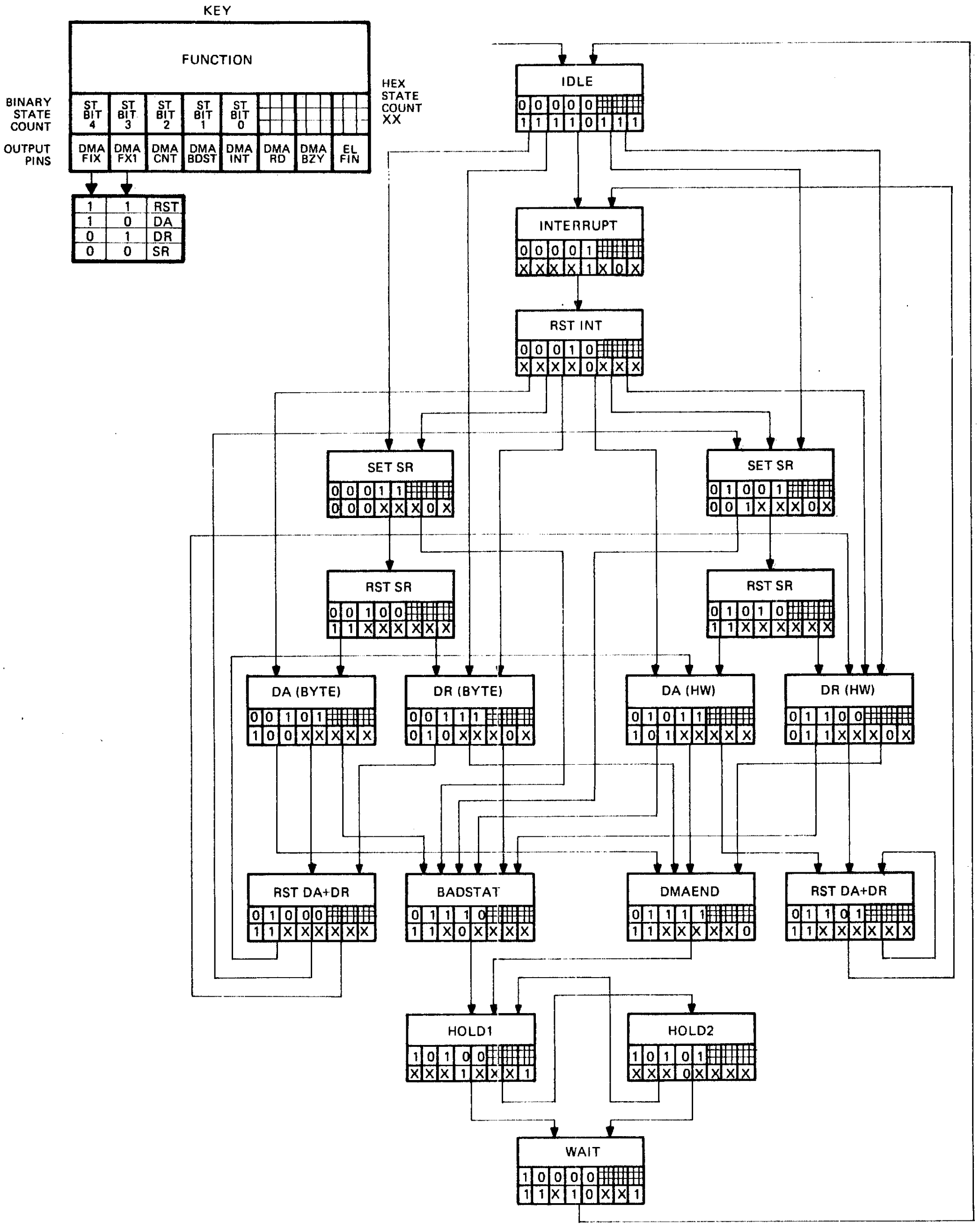
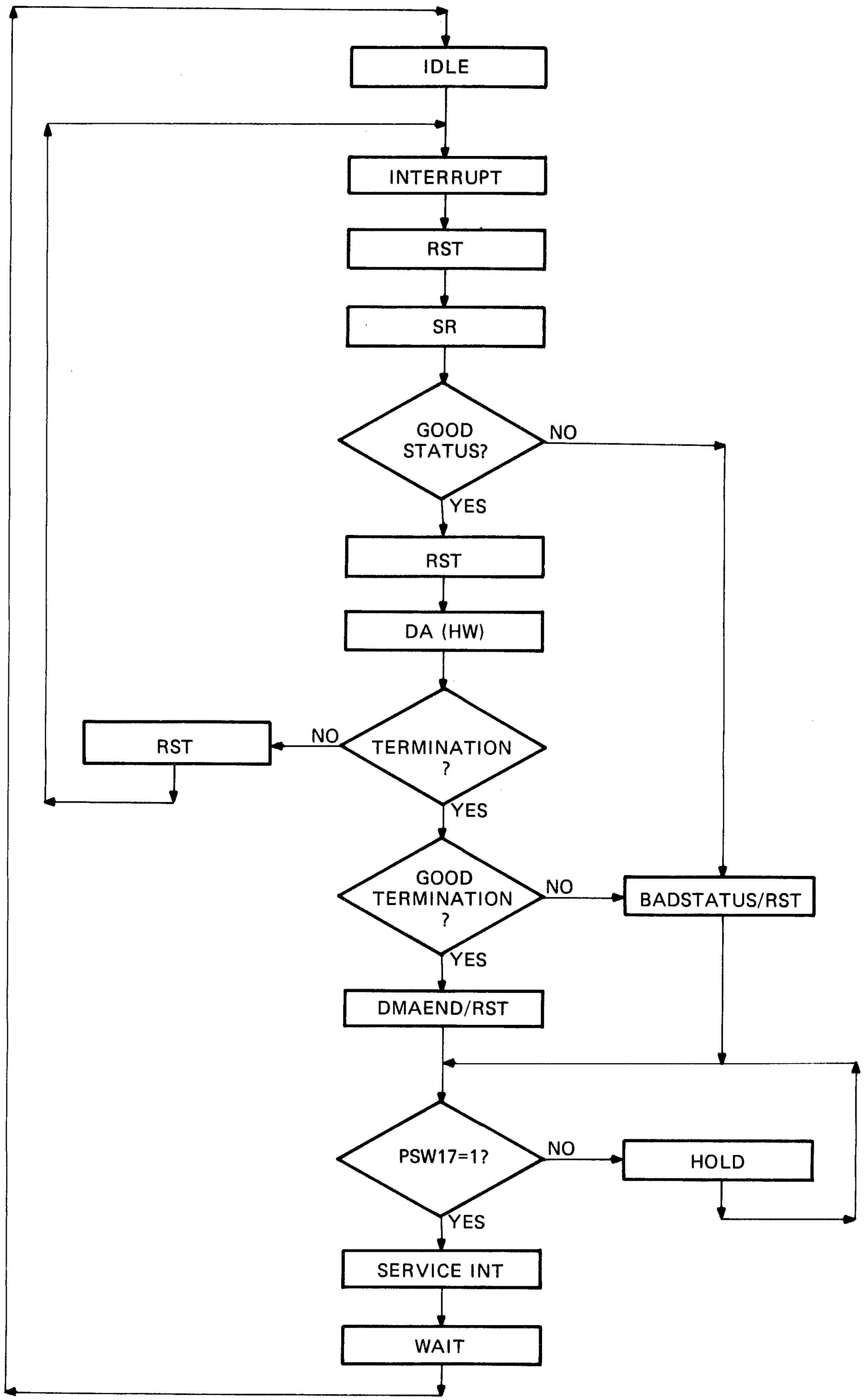


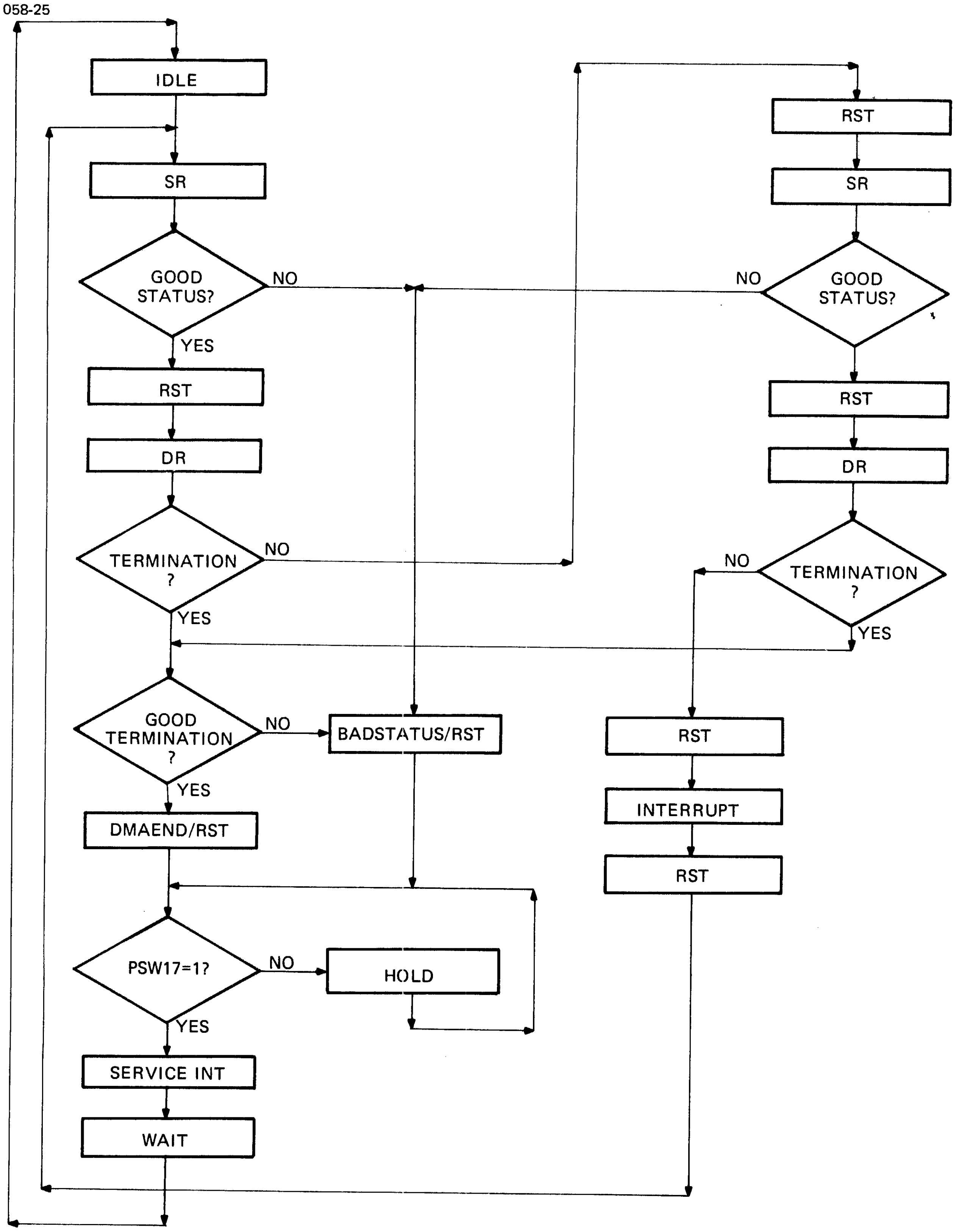
Figure 5-5 Private I/O State Diagram (DMA)

FLOWCHART 1 of 6

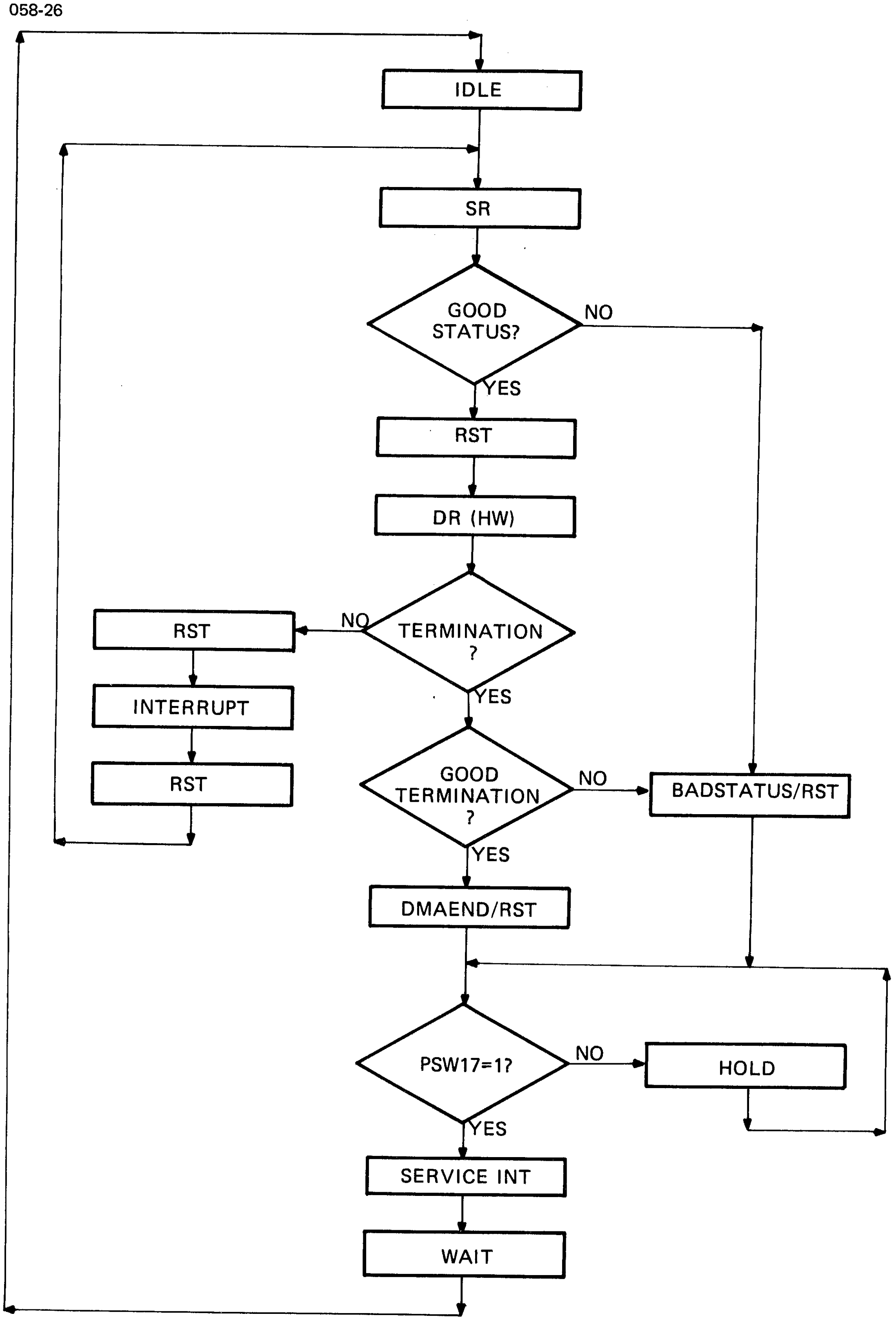
058-24



FLOWCHART 2 of 6

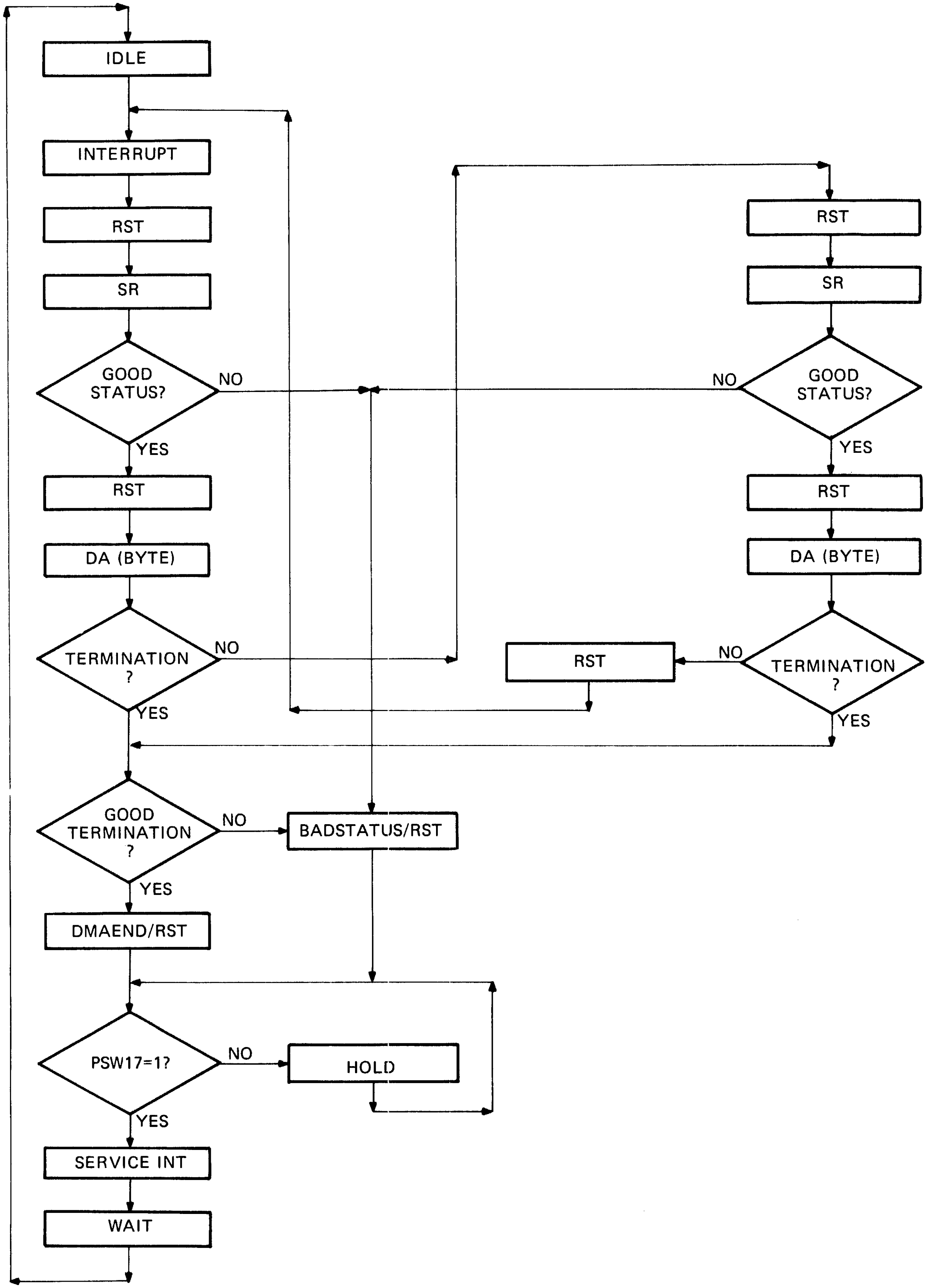


FLOWCHART 3 of 6

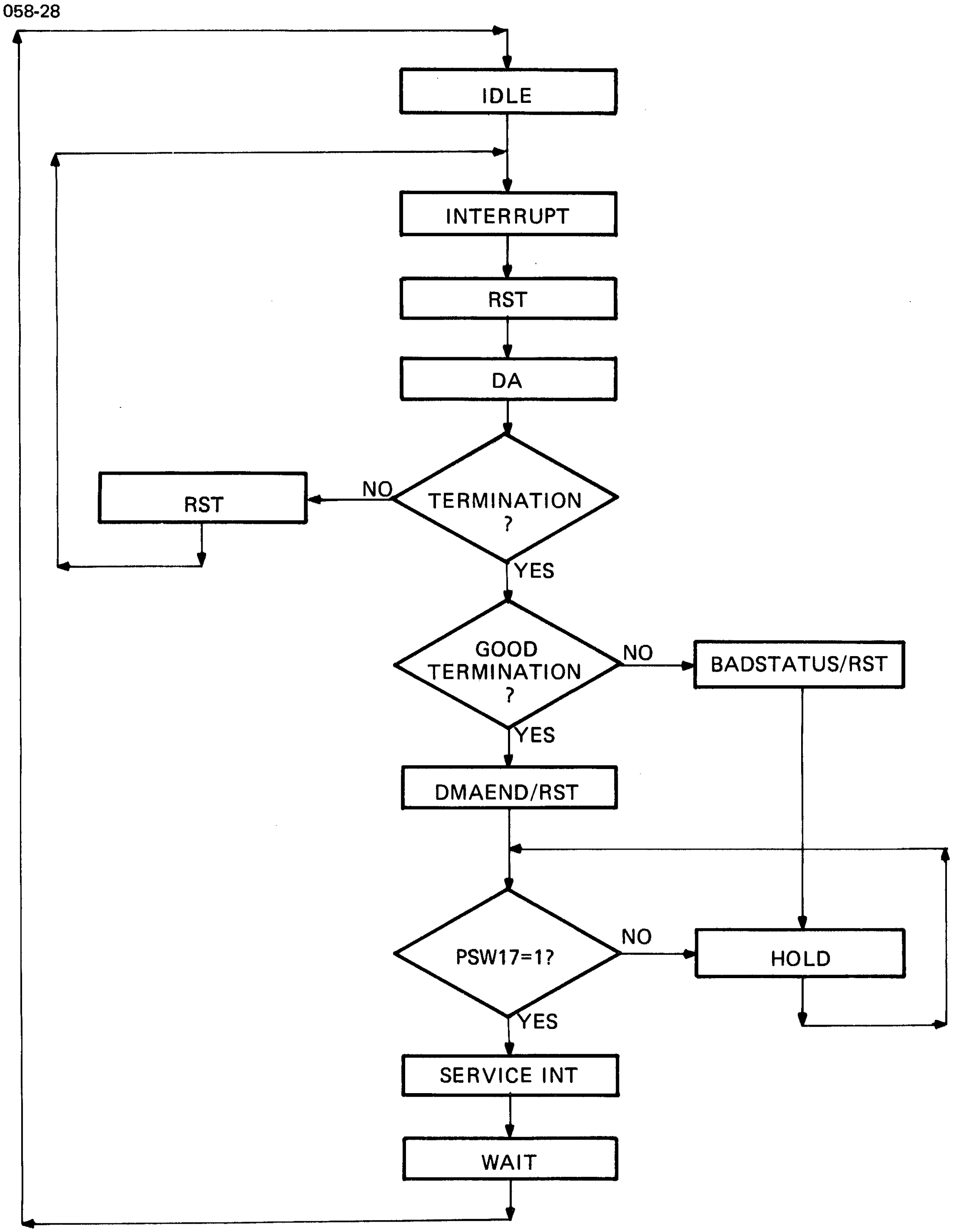


058-27

FLOWCHART 4 of 6



FLOWCHART 5 of 6



FLOWCHART 6 of 6

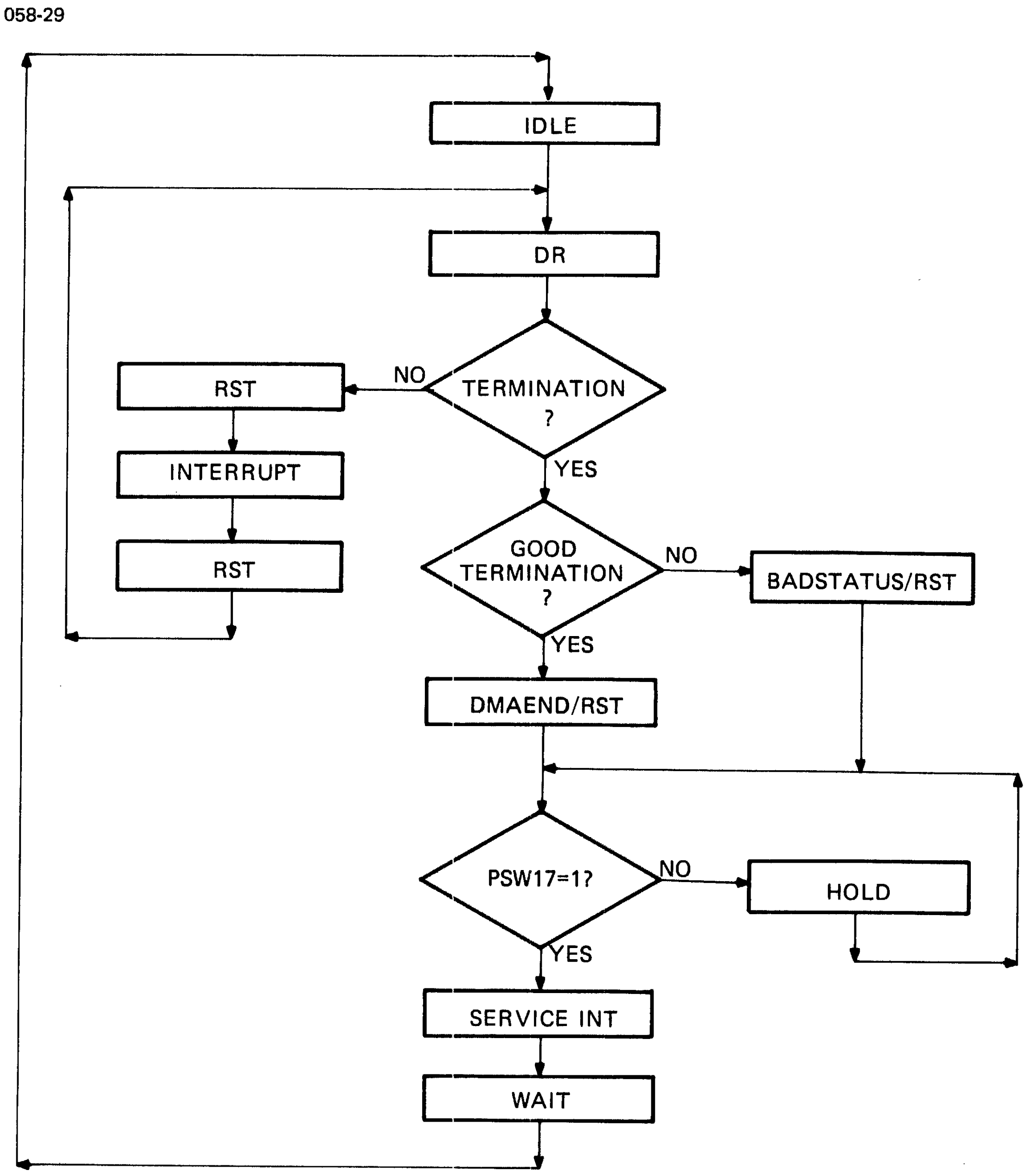


TABLE 5-6 DMA SEQUENCER STATES

STATE		POSSIBLE NEXT STATE	DEFINITION
NUMBER	OPERATION		
00	IDLE	1, 3, 9, C	The inactive state, state 00 on system clear.
01	INTERRUPT	2	Microinterrupt (DMAINTO) generated when memory operation is required.
02	RESET INTERRUPT	3, 9, B, C	Reset interrupt (DMAINTO).
03	SET STATUS REQUEST	4, E	Request status byte from I/O device, (PSRO) DMACNTO set low.
04	RESET STATUS REQUEST	5, 7	Reset status request (PSRO) upon return of SYNC.
05	DATA AVAILABLE (byte)	8, E, F	Sets signal PDAO. The bus is enabled. Tells device that data is on the bus.
06	DATA REQUEST (byte)	8, E, F	Sets signal PDRO. Data is latched in upon return of SYNC from I/O device.
08	RESET DA and DR	9	Resets signals PDAO or PDRO upon return of SYNC from I/O device.
09	SET STATUS REQUEST	A, E	Request status byte from I/O device. Second byte or halfword (PSRO) DMACNTO set high.
0A	RESET STATUS REQUEST	B, C	Reset status request upon return of SYNC. (PSRO)
0B	DATA AVAILABLE (halfword)	D, E, F	Sets signal PDAO. The bus is enabled. Tells device data is on the bus.
0C	DATA REQUEST (halfword)	D, E, F	Sets signal PDRO. Data latched in upon return of SYNC from I/O device.
0D	RESET DA and DR	D, 1	Resets signals PDAO and PDRO upon return of SYNC from I/O device.
0E	BADSTATUS	14	Set signal DMABSTO to indicate bad termination on a data transfer. Microcode bad termination.
0F	DMAEND	14	Set signal ELFINO to indicate good termination of a data transfer.
10	WAIT	00	WAIT state on microcode command STOP from every state except 00 or 10.
14	HOLD 1	10, 15	Hold state on DMA termination when interrupts are disabled (PSW171).
15	HOLD 2	10, 14	Sets DMABDSTO when interrupts are reenabled so that the microcode will service the DMA termination, good or bad.

5.5.3 Integrated Selector Channel (ISELCH) Termination

ISELCH termination can occur in one of two ways: good status (ELFIN0) or bad status (DMABST0).

Good termination occurs when carry-out from the DMA word counter (DMAEND0) informs the DMA sequencer that all words have been transferred. The DMA sequencer then sets signal ELFIN0 informing the macrointerrupt logic that good termination should be serviced.

Bad termination can occur in one of three ways:

- The microcode issues a command BADSTATUS after a noncorrectable error to inform the user that the transfer was invalid.
- The device controller involved in the transfer returns bad status on a STATUS REQUEST command from the DMA sequencer.
- Signal SCHK10 is activated by the device controller to indicate bad status to the DMA sequencer. This condition only occurs on a high-speed new protocol transfer.

DMA termination, good or bad, is handled in one of two ways:

- The microcode calculates the final address and then checks the interrupt mask bit (PSW171). Finding the interrupts enabled, the microcode clears the ISELCH, resets the ISELCH status flags in register external (REX) and jumps to execute the user interrupt handler.
- The microcode calculates the final address and then checks the user-controlled interrupt mask bit (PSW171). Finding the interrupts disabled, the microcode sets the interrupt pending flag and control is returned to the user. When the user reenables the interrupts (PSW171), the DMA sequencer sets signal DMABST0, used in this case not to indicate bad status, but to tell the interrupt logic to service the pending interrupt (good or bad). The microcode then resets the interrupt pending flag and moves to execute the user interrupt handler.

The final address is used to either check the length of the block transfer or to compare it with an expected last address known by the user. The microcode clears the ISELCH by issuing successive STOP commands.

CHAPTER 6
MEMORY THEORY OF OPERATION

6.1 INTRODUCTION

This chapter provides an analysis of the Model 3205 System memory unit (see Figure 6-1). The memory unit consists of the following sections:

- Bus structure
- Memory address registers (MAR, PMAR, PC and AMAR)
- Memory register (MR)
- Memory data register (MDR)
- Memory address translator (MAT)
- Error check and correction (ECC)
- Error logger
- Memory array
- Memory refresh

The function of the memory unit is to store the various programs and data needed by the user. Control of the memory unit is done primarily by using the control store (CS) MIC field (CS32:21).

The memory unit control register (CR) (Sheet 8) is used to store the memory control word and stores this data through a two-cycle memory operation. The CS bits and their memory related definitions are located in Table 6-1.

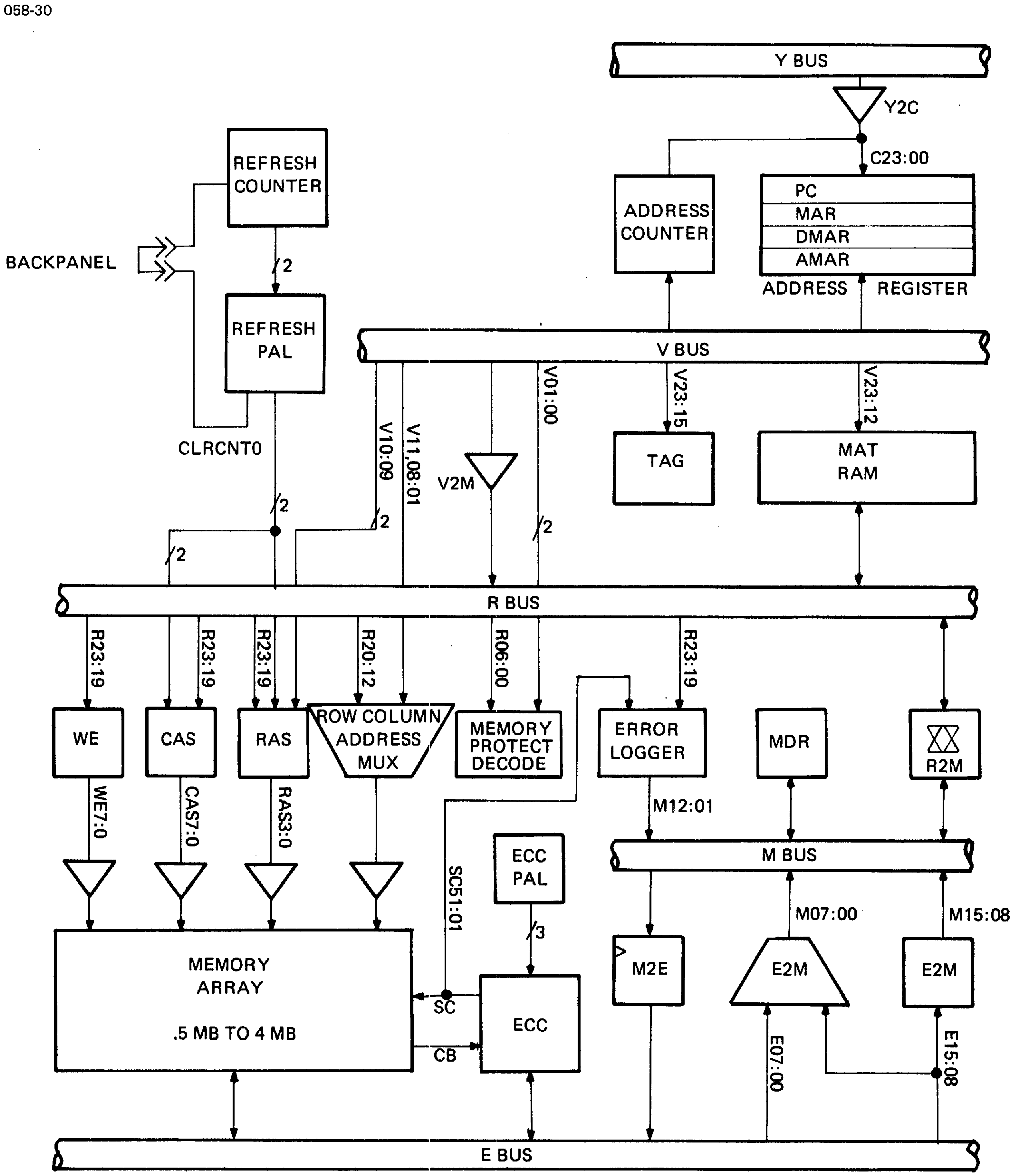


Figure 6-1 Memory Unit Block Diagram

TABLE 6-1 MEMORY CONTROL WORD

CS BIT	FUNCTION
21	MAT Enable Control 0 = disabled 1 = enabled
23	Read/Write Select 0 = write 1 = read
24	Halfword/Byte Select 0 = byte (24-bit bus, most significant byte) 1 = halfword (24-bit bus, least significant halfword)
27:25	Used to select sources and destinations on the M bus.
28	Memory Operation Select 0 = memory unit idle 1 = memory unit active
30:29	Address Register Select 0 0 = program counter (PC) 0 1 = memory address register (MAR) 1 0 = direct memory address register (DMAR) 1 1 = auxiliary memory address register (AMAR)
32:31	Memory Function Select 0 0 = MAT operation (read/write) 0 1 = address register (AR) operation (read/write) 1 0 = halfword/byte memory array access 1 1 = fullword memory array access

6.2 BUS STRUCTURE

The memory unit uses six buses for communications, four internal memory unit communications buses (V, R, A and C) and two external memory unit communications buses (M and Y).

The V, R, A and C buses are described below:

- V bus

  The V bus or virtual bus is driven by the address register output. It drives the address incrementer and addresses the MAT random access memory (RAM) along with the components mentioned below.

- R bus

  The R bus or real address bus carries the translated memory address when the MAT is enabled. The R bus drives the memory address parity generator and can be driven by the V bus when the MAT is disabled.

- A bus

  The A bus is comprised of 58 signal lines, a 9-bit memory array address field (MA, MX), a 16-bit data bus (E bus), 12 ECC parity lines (SC and CB), and 21 control lines (row address strobe (RAS), column address strobe (CAS), WE and DOEO).

- C bus

  The C bus or counter bus drives the AR. It is the output of the address incrementer. The C bus can be driven by the Y bus on an AR load operation.

The M and Y buses are described below.

- M bus

  The M bus links the memory unit to the execution and input/output (I/O) units.

- Y bus

  The Y bus is the main communicaton link of the execution unit. It interfaces the M bus through a transceiver and can drive the C bus when signal Y2CEO is active. The Y bus to C bus link is used to load the AR.

All six buses are described in detail in the following sections.

6.2.1 V Bus

The V bus is driven by the AR output and carries any of four addresses. V bus bits are routed to memory management, memory array control and the address incrementer. The most significant 12 bits (23:12) are used by the MAT and are translated when the MAT is enabled.

The 12 least significant bits (LSBs) (11:00) are used by the memory array control:

- 9 (V111,081:011) to the memory address multiplexor (MUX) (Sheet 15),
- 2 (V101:091) to the row address strobe (RAS) selection PAL (Sheet 7),
- 3 (V031:011) to the M register (Sheet 8), and
- 2 (V011:001) go to the memory decode protect PAL2 to check address alignment.

The memory address incrementer receives all 24 V bus bits.

6.2.2 R Bus

The R bus is used when executing the five operations listed in Table 6-2. R bus destinations and functions are listed in Table 6-3.

TABLE 6-2 OPERATIONS USING THE R BUS

OPERATION	SOURCE	DESTINATION
Read AR	V2R buffers	R2M transceivers
Read MAT	MAT RAM	R2M transceivers
Write MAT	R2M transceivers	MAT RAM
Memory array access (MAT enabled)	MAT RAM	Memory array control
Memory array access (MAT disabled)	V2R buffers	Memory array control

TABLE 6-3 R BUS DESTINATION AND FUNCTION

R BUS BITS	DESTINATION	FUNCTION
23:19	Error logger	To store the memory single in-line package (SIP) row being accessed when the error occurred.
	RAS PAL	Decoded to create the signal NOMEMO indicating nonpresent memory.
	CAS PAL	Decoded to create the correct CAS signal for the SIP row being accessed.
	WE PAL	Decoded to create the correct write enable signal for the SIP row being accessed.
20:12	Address MUX	Create the memory array column and row address.
6	Memory protect decode PAL 3	Used to create signal NMISSO (MAT miss).
05:00	Memory protect decode PAL 1	Used in creating signals DIRTYO and MATFLTO.

All R bus bits except R11:08 go to the R bus parity checker.

6.2.3 C Bus

The 24-bit C bus is used to load the AR and is driven by the address incrementer or the Y to C bus buffers (see Section 6.3).

6.2.4 A Bus

The A bus is the data and control lines for the memory unit.

- The E bus is the 16-bit data bus for the memory array and is driven by one of three sources.
  - The memory array drives the E bus on both data and instruction fetches.

  - The ECC drives the E bus when it has corrected a single-bit error on data or instruction fetch.

  - The M bus to E bus register drives the E bus on a write to the memory array.

- The three destinations from the E bus are:

  - The memory array receives E bus data on all memory writes.

  - The ECC receives data from the E bus on reads from and writes to the memory array. On a memory write, the ECC generates the six parity bits (SC51:01) stored with each memory word and on memory read checks the six parity bits (CB51:01) against the E bus data.

  - The E bus to M bus buffers are used to drive the M bus on all memory array reads.

- The syndrome check bits (SC51:01) are parity bits generated by the ECC and stored with each memory word.

- The check bits (CB51:01) are six parity bits used by the ECC to check for errors on a memory array read.

- Signal DOEO keeps the memory array from driving data onto the E bus when a single-bit error is present. This allows the ECC to drive the corrected data onto the E bus without interference.

- The 9-bit memory array address bus (MA80:00) is generated by a 2-to-1 MUX and carries the row and column address fields to the memory array.

- The RAS (RAS30:00) is used to clock the row address into the appropriate dynamic RAM.

- The CAS (CAS70:00) is used to clock the column address into the correct SIP column.

- The write enable (WE70:00) is used to store data from the E bus to the previously addressed memory array location.

6.2.5 M Bus

The M bus is the main communication link between the four system units. Table 6-4 lists the M bus sources, destinations and data valid time in nanoseconds (ns).

TABLE 6-4 M BUS SOURCES AND DESTINATION

SOURCE	DATA VALID	DESTINATION
Y2M transceivers	200ns	MUX bus transceivers
Y2M transceivers	200ns	Private multiplexor (PMUX) bus transceivers (integrated selector channel (ISELCH) idle)
Y2M transceivers	200ns	MDR
Y2M transceivers	200ns	M to E bus register
Y2M transceivers	200ns	R to M transceivers
MUX bus transceivers	100ns	Y2M transceivers
PMUX bus transceivers	200ns	M2E register (ISELCH busy)
PMUX bus transceivers	100ns	Y2M transceivers (ISELCH idle)
MDR (nibble)	100ns	Y2M transceivers
MDR (halfword)	100ns	Y2M transceivers
MDR	100ns	instruction register high (IRH)
MDR	200ns	M to E bus register
R2M transceivers	100ns	Y2M transceivers
R2M transceivers	200ns	MDR
E2M bus buffers	100ns	Y2M transceivers
E2M bus buffers	100ns	PMUX bus transceivers (ISELCH busy)
E2M buffers	100ns	MDR
E2M buffers	100ns	IR
Error logger	100ns	Y2M bus transceivers

6.2.6 Y Bus

The Y bus is internal to the execution unit and is used to load the AR. For further information on the Y bus, see Chapter 4.

6.3 ADDRESS REGISTERS (ARs) (PC, MAR, DMAR, AMAR)

The ARs provide a 24-bit address that points to a location in the memory array. The 24-bit virtual address from the AR is output onto the V bus and sent to the MAT, which translates it into the real address. The real address is the final address that accesses the appropriate location in the memory array.

The ARs consist of six 16 x 4 RAM (Sheet 11) chips, which form sixteen 24-bit registers. Only four of these registers are used. The input to the ARs is from the Y bus through transceivers onto the C bus and output onto the V bus.

The five operations listed below have an effect on the AR.

- Read byte
- Read halfword
- Write byte
- Write halfword
- Increment register

In these operations, CSR bits 30:29 point to the register (PC, MAR, DMAR and AMAR) that is affected. The MAT, in the enable or disable mode, does not have an effect on the AR.

- Read byte or halfword from AR

  The read AR instruction is used to load the PC, MAR, DMAR or AMAR address to the execution unit. This is done by accessing the correct register (CSR30:29), which is clocked in 25ns prior. This address is then loaded onto the R bus. This is accomplished by the address being loaded onto the V bus through buffers to the R bus, without memory translation, by enabling the R2M drivers signal HW1BYTE. This procedure allows the contents of the AR to be loaded onto the M bus in time to be used by the arithmetic logic unit (ALU) (execution unit) as an operand.

- Write byte or halfword to the AR

  The write to AR instruction loads the PC, MAR, DMAR and AMAR with an address. The appropriate register is selected at t-25ns by CS bits 30:29. The execution unit then loads the data (halfword or byte) onto the Y bus. This address is then loaded onto the C bus by the Y2C bus drivers and clocked into the AR at 175ns.

- Increment register

  The increment AR instruction is executed during the memory array access. This is a two-cycle operation; during the first cycle, the present 24-bit address is loaded into the counter; in the second cycle, this address is incremented by two and written back into the AR.

6.3.1 Program Counter (PC)

The contents of the PC point to the address of the next user instruction to be executed.

6.3.2 Memory Address Register (MAR)

The contents of the MAR are used for addressing a specific location in memory. If the program status word (PSW) bit 21 is set, the contents of the MAR are used by the MAT to generate a new (relocated) address.

6.3.3 Direct Memory Address Register (DMAR)

The contents of the DMAR are used for addressing a specific location in memory that is to be either read or written to during the DMA block transfer.

6.3.4 Auxiliary Memory Address Register (AMAR)

The AMAR is used in commercial instructions and MAT operations.

6.4 MEMORY REGISTER (MR)

The MR is used by the microcode to test certain system conditions. Table 6-5 describes the MR in full. Memory protection violations are listed in Table 6-6.

TABLE 6-5 MR BIT DEFINITION

M REGISTER		DEFINITION
OUTPUT	INPUT	
Y10:08	V03:01	These three bits indicate the address of the first halfword of the instruction being executed. In a case of having to reexecute an instruction due to a fault, PC is decremented until V bits match.
Y12:11	BCNT11 BCNT01	These two signals are derived from CS bits 30 and 29. They are used by the microcode to determine which of four AR locations was being used when a fault occurred.
Y14:13	FLT1 FLTO	These two bits are used by microcode to determine the type of MAT fault that occurred.

TABLE 6-5 MR BIT DEFINITION (Continued)

M REGISTER		DEFINITION
OUTPUT	INPUT	
Y15	NVM1	This signal is used by microcode on the start-up procedure to determine if memory power (P5U) was lost on power-down. If P5U was lost, then NVM1 (non-valid memory) is active and the microcode performs cold start to initialize memory.
Y07	DMACNT0	The signal DMACNT0 is generated by the DMA sequencer and is used by the microcode to determine which byte, first or second, is presently being transferred in an ISELCH byte operation (see Section 5.3.1.2 and Table 5-4).
Y06	DMASTS0	The signal DMASTS0 is generated by the microinterrupt logic when a double-bit error occurs while a DMA interrupt is being serviced (see Section 3.5). On a double-bit memory error, the microcode checks the signal DMASTS0. If a DMA transfer is in progress, the microcode halts the transfer and continues to execute the user program. The microcode does not service the double-bit error interrupt and does not issue a machine malfunction. On the next interruptible instruction fetch, the DMA transfer is terminated (see Section 5.5.3).

TABLE 6-6 MEMORY PROTECT VIOLATION

Y BIT 14:13		DEFINITION
0	0	Execute protect violation
0	1	Write protect violation
1	0	Read protect violation
1	1	Access level protect violation

6.5 MEMORY DATA REGISTER (MDR)

The MDR is loaded on a read from the memory array, the execution unit or the MAT. The least significant four bits (nibble) of the MDR have a special purpose. On all short form user instructions, the MDR least significant nibble is used by the execution unit as the second operand. On an MDR nibble read, the twelve most significant bits (MSBs) are forced to zero. M bus sources and destinations for MDR data can be found in Table 6-4.

6.6 MEMORY ADDRESS TRANSLATOR (MAT)

The MAT supports:

- 4Mb of physical memory addressing
- 16Mb of virtual memory addressing
- Read, write and execute protection
- Four levels of hardware controlled access to segments

The function of the MAT is to translate a virtual address (VA) or program address into a real address (RA) (physical memory address). For more information on MAT functions, see the Model 3205 System Instruction Set Reference Manual.

The mapping of VA space to RA space is accomplished by using information supplied in the segment table. The table must be aligned within a 128-byte boundary in physical memory and contain from 1 to 256 doubleword entries. Each doubleword entry and a segment table entry (STE) is indexed by the segment number field of the VA. Figure 6-2 is a simplified flowchart of the translation of a VA to an RA.

When the MAT is disabled, the program address lines directly access the memory address specified. With the MAT enabled, the memory address is the sum of the least significant 16 bits of the program address and a bias value contained in an entry in the process segment table (PST) in memory.

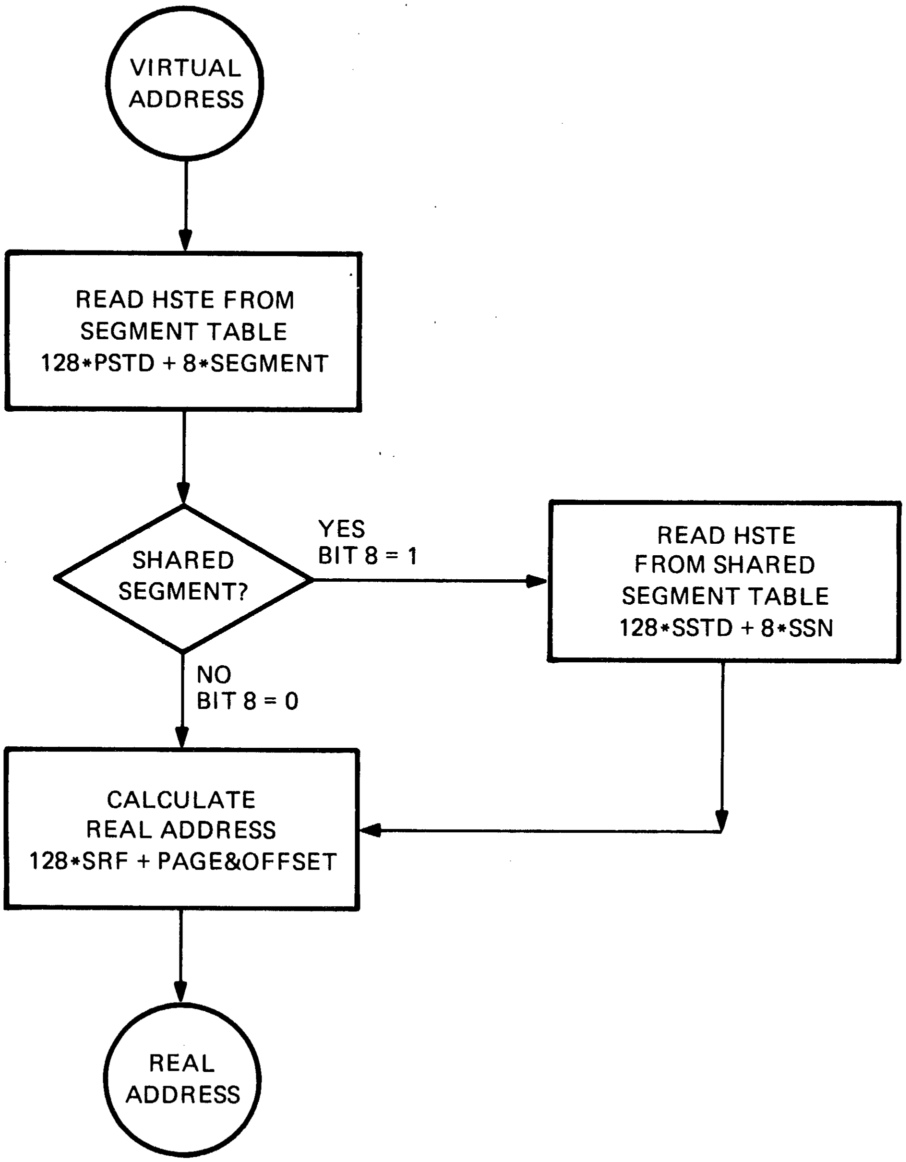

058-31
VIRTUAL ADDRESS
READ HSTE FROM SEGMENT TABLE 128*PSTD + 8*SEGMENT
SHARED SEGMENT?
YES BIT 8 = 1
READ HSTE FROM SHARED SEGMENT TABLE 128*SSTD + 8*SSN
NO BIT 8 = 0
CALCULATE REAL ADDRESS 128*SRF + PAGE&OFFSET
REAL ADDRESS

Figure 6-2 Flowchart of MAT Process

The MAT forms a functional part of the memory unit. The MAT functions are implemented through the use of the microcode. The MAT hardware is located on the processor board (see Figure 6-3).

The MAT translates the program or VA of memory used by the processor into the actual or real address of the memory system. The MAT allows the use of a full 16Mb address range by any process operating under its control. The MAT also decodes addressing faults such as a segment nonpresent in memory or a nonpermitted operation, and sends a fault code to the processor. These fault codes are explained in the Model 3205 System Instruction Set Reference Manual.

Figure 6-3 shows a portion of the memory system and the functional location of the MAT.

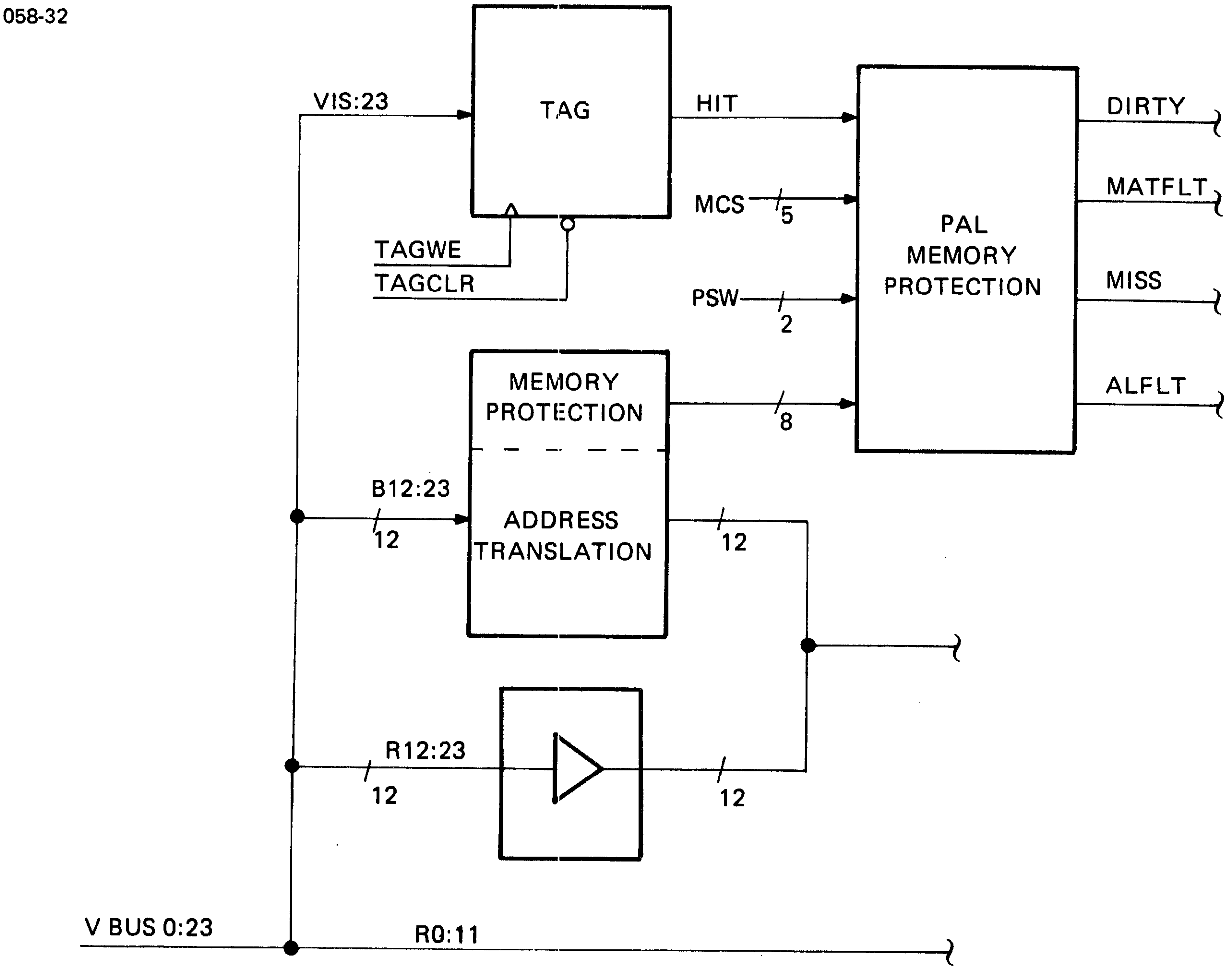


Figure 6-3 MAT

6.6.1 Memory Address Translator (MAT) Functions

When the MAT is disabled, the program address directly accesses the memory. With MAT enabled, the memory location is specified by the sum of the least significant 16 bits of the program address and a bias value contained in an entry of the PST in memory.

The PST for a task can consist of a maximum of 256 contiguous double fullwords or entries. Only the first 32 bits of an entry are of significance to the hardware. The starting address of the PST is specified by the process segment table descriptor (PSTD).

When the micocode loads a new entry in the MAT lookup table, the program address is considered to consist of two fields, the segment field and the offset field, as shown in Figure 6-6. With PSTD bits 31:15, PA15:08 are used as an index into the PST to select an entry. The accessed entry indicates if it is shared or private. If private, the STE contains the starting address of a segment in memory for which the offset field of the PA serves as an index. If PA15:08 are greater than the segment size specified by the PSTD, a MAT fault is generated.

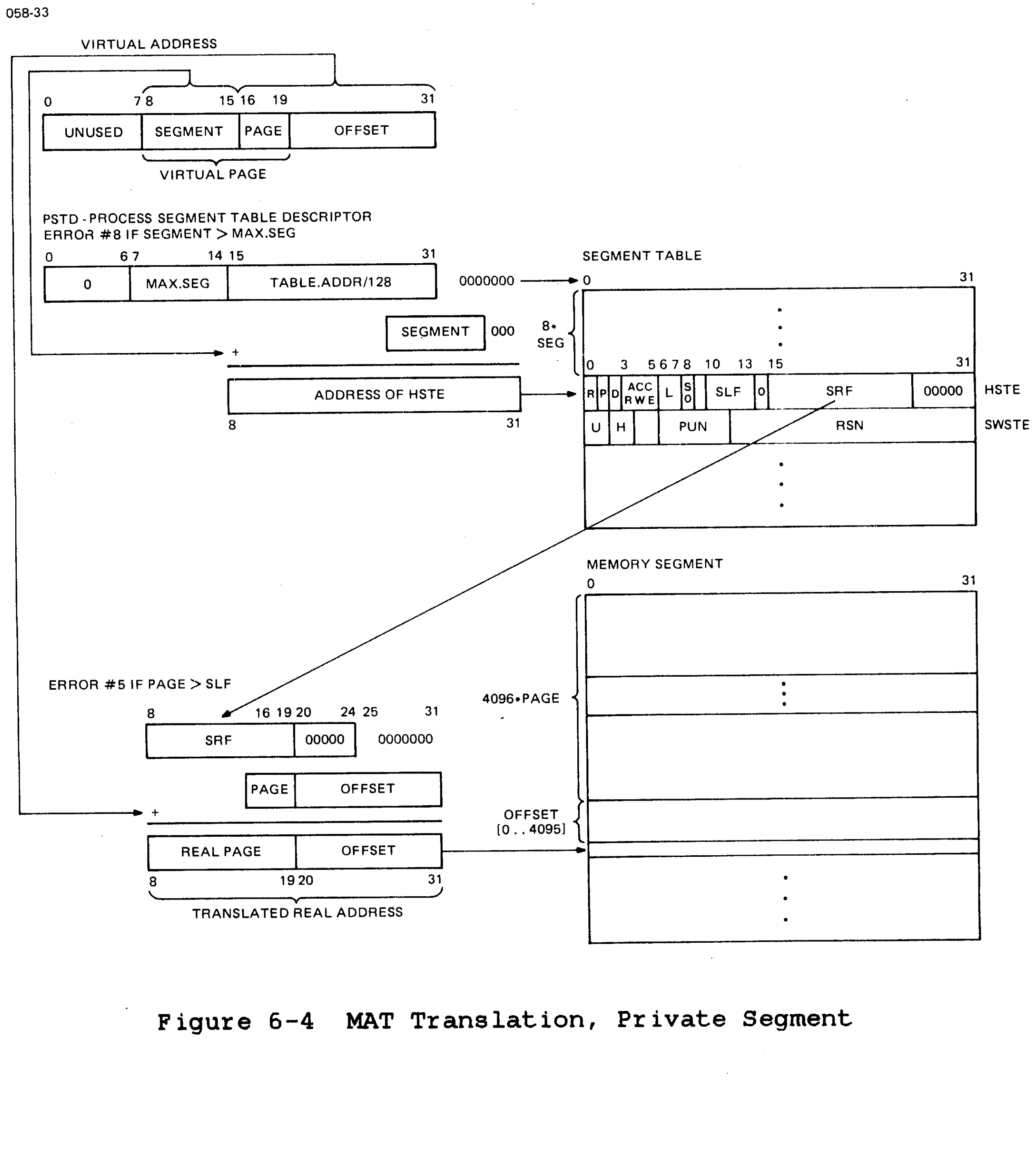


Figure 6-4 MAT Translation, Private Segment

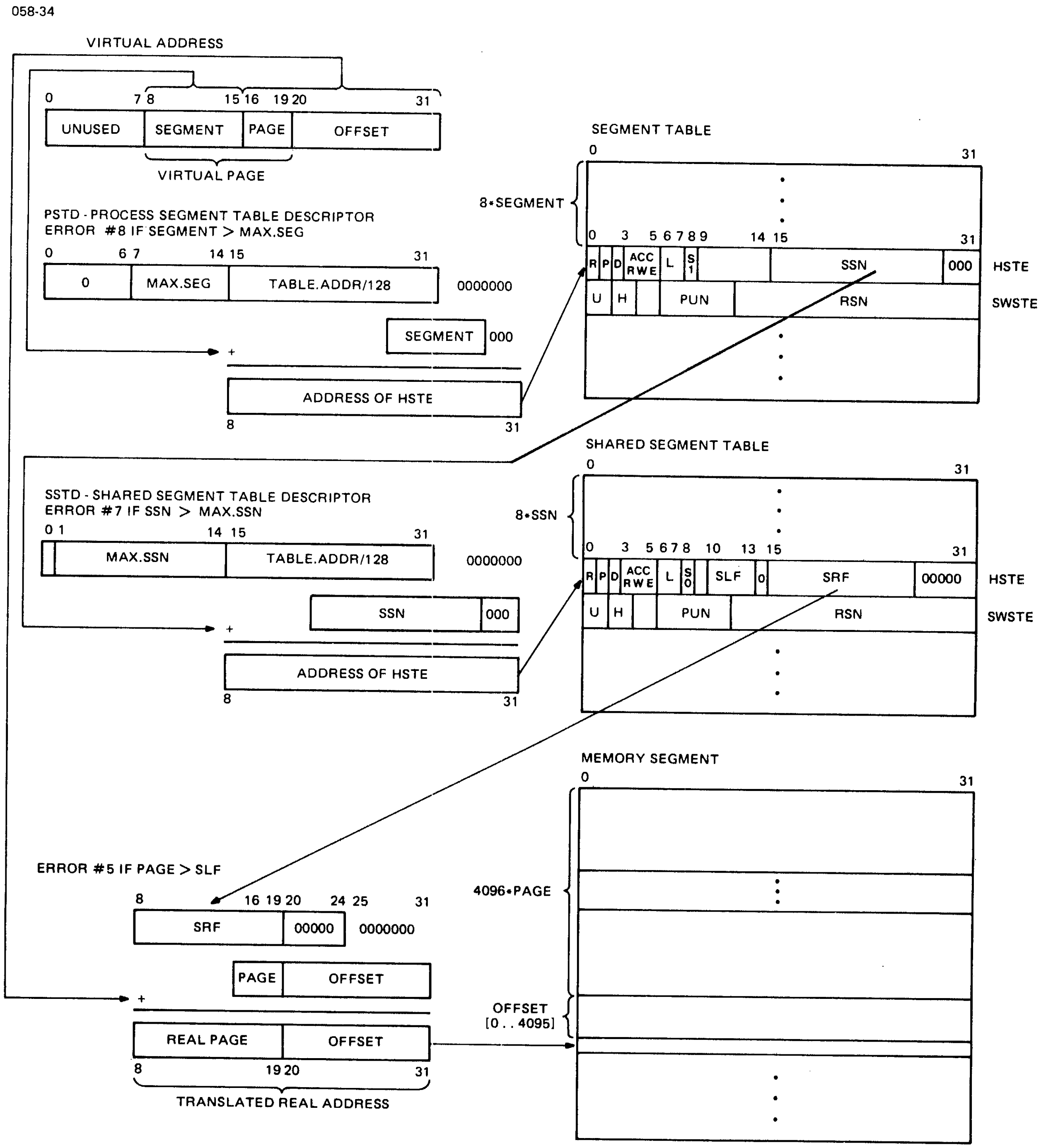


Figure 6-5 MAT Translation, Shared Segment

If the segment is shared, the shared segment table (SST) must be accessed to obtain the STE. The SST consists of 16K double fullwords or entries. The starting address in memory of the SST is contained in the shared segment table descriptor (SSTD). Before the SST is accessed, the shared segment size in the SSTD is compared with the PST STE. If SSTD bits 14:2 are less than the STE size, a MAT fault results; if not, the SST is accessed. SSTD bits 31:15 point to the first location in the SST and the STE serves as an index. The STE obtained from the SST points to the segment in memory and PA31:16 index into the segment. Only the first 32 bits of a STE relate to hardware.

Bits 31:0 of the STE contain the hardware segment table entry (HSTE), which is comprised of several fields (shown in Figure 6-6). There are eight bits in the HSTE for memory management. These bits allow the operating system to:

- specify which tasks have access to a segment,
- specify acceptable operations on a segment, and
- keep a record of whether a segment was accessed, written to or exists.

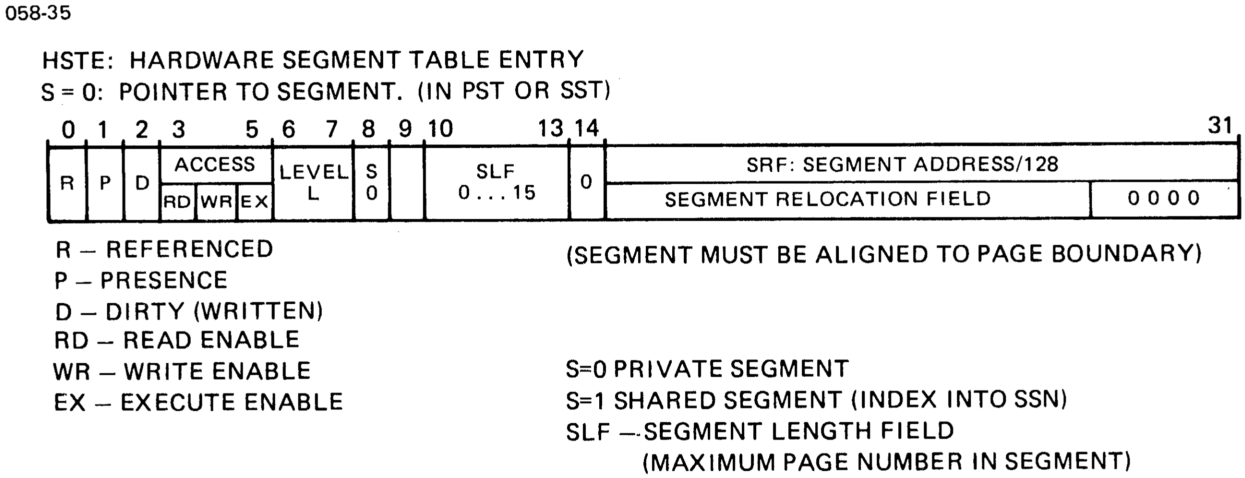


Figure 6-6 Bit Representation of the HSTE

Reserved fields of the HSTE must always contain zero. A list of the bits and their functions follows.

- Reference bit (R)

  Bit 0 of the HSTE is the reference (R) bit. This bit is set by the hardware when an attempt is made to read, write or execute the contents of the segment. DMA I/O does not set the R bit for the participating segment.

- Presence bit (P)

  Bit 1 of the HSTE is the presence (P) bit. The presence bit is set when the segment described by the STE is in memory; it is zero when the segment is not present in memory. When P=0, the hardware ignores the contents of the rest of the HSTE (bits 2 through 31), but the R bit is set as a result of the attempted access. Referencing a segment that is not present (P=0) causes a fault.

NOTE

> This bit should not be confused with the present bit used by the MAT lookup table.

- Dirty bit (D)

  Bit 2 of the HSTE is the dirty (D) bit. This bit is set by the hardware when a program modifies (writes into) a segment. This bit is not set by DMA I/O operations that modify the segment. If P=0, attempts to modify the segment do not affect the dirty bit.

- Access mode bits (A)

  Bits 3 through 5 of the HSTE are the access mode (A) bits. These bits determine the allowed modes of access (read, write or execute) to the segment. If an attempt is made to access a segment in a manner not permitted by the setting of the access mode bits, a MAT fault occurs. Table 6-7 defines the access field settings and types of access that are enabled.

TABLE 6-7 SEGMENT ACCESS FIELD SETTINGS

BIT	ACCESS MODE	FIELD SETTING
3	Read enabled	0 = Read protected
4	Write enable	0 = Write protected
5	Execute enable	0 = Execute protected

- Access level bits (L)

  Bits 6 to 7 of the HSTE are called the access level (L) bits. The L field is used in conjunction with bits 10 and 11 of the PSW to determine if a program can access a segment. If the value represented by the contents of PSW bits 10 and 11 is greater than or equal to the L bits, then access to the selected segment is allowed; otherwise, a fault occurs. The L bits are checked before the A bits.

- Shared segment bit (S)

  Bit 8 of the HSTE is called the shared segment (S) bit. If this bit is zero, the MAT performs protection and relocation functions as defined for a private segment. The S bit must be zero for all STEs in the SST.

  If the S bit is set in a PSTE, the selected segment is a shared segment. In this case, the segment relocation field (SRF) of the PST STE is used as a byte offset into the SST. The SST STE found at the resulting address is used by the MAT in performing protection and relocation functions as follows: the values of the A fields in the PST STE and the SST STE are ANDed to determine the allowed access modes; all other MAT protection and relocation functions are performed using data from the SST STE.

- Segment limit field (SLF)

  Bits 10 through 14 of the HSTE are called the SLF. The SLF indicates the size of a segment according to the following formula:

  SLF = (size of segment) divided by (4kb) - 1

  For example, for a segment size 4kb, the SLF would contain 0. When a VA is generated, the contents of bits 8:12 of the VA are compared to the SLF. If the SLF is less than this number, a MAT fault occurs.

  The granularity of memory allocation is 4kb (4,096). This means that memory must be allocated in 4kb units.

- Segment relocation field (SRF)

  Bits 15 through 31 of the HSTE are called the SRF. The interpretation of the SRF depends upon the setting of the S bit. If S is set in the PST, the PST SRF contains a byte offset into the SST, where the STE for the segment can be found. If the value contained in the PST SRF is greater than the size in bytes of the SST, a MAT fault occurs.

If S is 0 in the PST, the SRF contains the RA of the segment in memory, divided by 128. For example, if the segment starts at RA Y'146000', the SRF of the HSTE should contain X'28C0'.

NOTE

> For a shared segment, the least significant three bits of the SRF in a PST HSTE must be 0, as the indicated SST HSTE is aligned to a doubleword boundary. The least significant five bits must be 0 for the SRF in all other cases, as the SRF is the address of a segment aligned to a 4kb boundary divided by 28. If the MAT attempts to use an SRF that has a 1 in any of these trailing bits, the results are undefined.

6.6.2 Memory Address Translator (MAT) Implementation

Memory management on the Model 3205 System is implemented with the use of a 4K x 20-bit lookup table. This table acts like a cache. The data in the table is loaded under microcode control and is broken down into two fields: the memory translation field (R23:12) is the most significant 12-bits and the memory protection field (R7:0) identifies an invalid memory operation.

When the MAT is enabled, the 12 MSBs of the program address become the address in the lookup table. The 12 most significant outputs of the lookup table drive the 12 MSBs of the R bus. These bits become the RA. A TAG comparator chip determines if the correct translated address is in the lookup table. If the correct address has not been loaded, a microinterrupt (MISS0) is activated and the microcode loads eight lookup table entries. These eight entries allow the translation of any address within a 32K block. After the table has been loaded, the memory access that caused the miss is reexecuted.

6.6.3 TAG Circuitry

The TAG circuitry keeps track of the validity of the data in the lookup table. The data may be deemed invalid in one of three ways.

- TAG chip

  The TAG chip keeps track of 32K blocks of addresses. Each block is aligned on a 32K boundary with bits 23:15 determining the block. If a program enters a new 32K block that has not been previously addressed, the TAG chip generates a MISS microinterrupt, which results in the microcode loading eight lookup table entries that cover the 32K block.

- Valid bit entry

  Most user programs do not end on a 32K boundary. This results in some of the eight entries not having a valid RA. If the program tries to access this nonallocated memory, a MAT miss (NMISS0) microinterrupt is generated. During the miss microroutine, the program address is checked to see if it is within the limits of the allocated segment. If the address is not within the limit, a segment limit fault is generated.

- Parity

  Odd parity is generated when the MAT is written to. When the entry is used on a memory access, the parity is checked by memory protect PAL3. If an error is detected, a MAT MISS microinterrupt is generated. During the MISS interrupt routine, the correct data is generated, written to the MAT and read back. If the read MAT operation fails to produce the correct data, a MAT parity fault routine is executed.

TABLE 6-8 LOOKUP TABLE BIT DEFINITIONS

LOOKUP TABLE BIT		
NAME	NUMBER	DEFINITION
Access level	R1:0	Same as HSTE (R1:0≤ PSW11:10; access valid)
Execute enable	R2	Same as HSTE (R2 = 1; execute enable)
Write enable	R3	Same as HSTE (R3 = 1; write enable)
Read enable	R4	Same as HSTE (R4 = 1; read enable)
Dirty	R5	Same as HSTE (R5 = 1; segment has been written to)
Valid entry	R6	If R6 = 1, then the lookup table entry points to a valid real address. If R6 = 0, a segment limit fault occurs.
Parity	R7	Generates odd parity.
Translated address	R24:12	Contains the 12 MSBs of the real address.

6.7 ERROR CHECK AND CORRECTION (ECC)

The Model 3205 System ECC (Sheet 3) function is implemented on a single integrated circuit (IC) chip (see Figure 6-8) with a single PAL for data flow control. The ECC is connected to the E bus and can be a source or destination for E bus data. The purpose of the ECC circuit is to detect and correct errors in memory data. The ECC generates six parity bits (SC51:01) for every halfword written to memory to accomplish this task (see Figure 6-8 and Table 6-9). When the memory data is retrieved, the parity bits (CB51:00) are checked against the data halfword using a modified Hamming code (see Figure 6-7). If a single-bit error is detected, signal ERR0 is activated and the main system clock is stopped while the data is corrected. In this case, the normal two-microcycle memory operation is extended to three cycles, which allows time for the corrected data to appear on the bus. If a multiple-bit error is detected, signal MERR0 is generated, and in turn generates signal ECCD0, a level two microinterrupt. The microcode multiple-bit error handler issues a machine malfunction. Status is saved and control is returned to the user. On every error situation, the syndrome bits (SC15:01) are loaded into the error logger (see Section 6.8). The error logger is read by the user and the data used to determine which bit or bits were in error. Table 6-10 shows how the syndrome bits can lead to the faulty data bits.

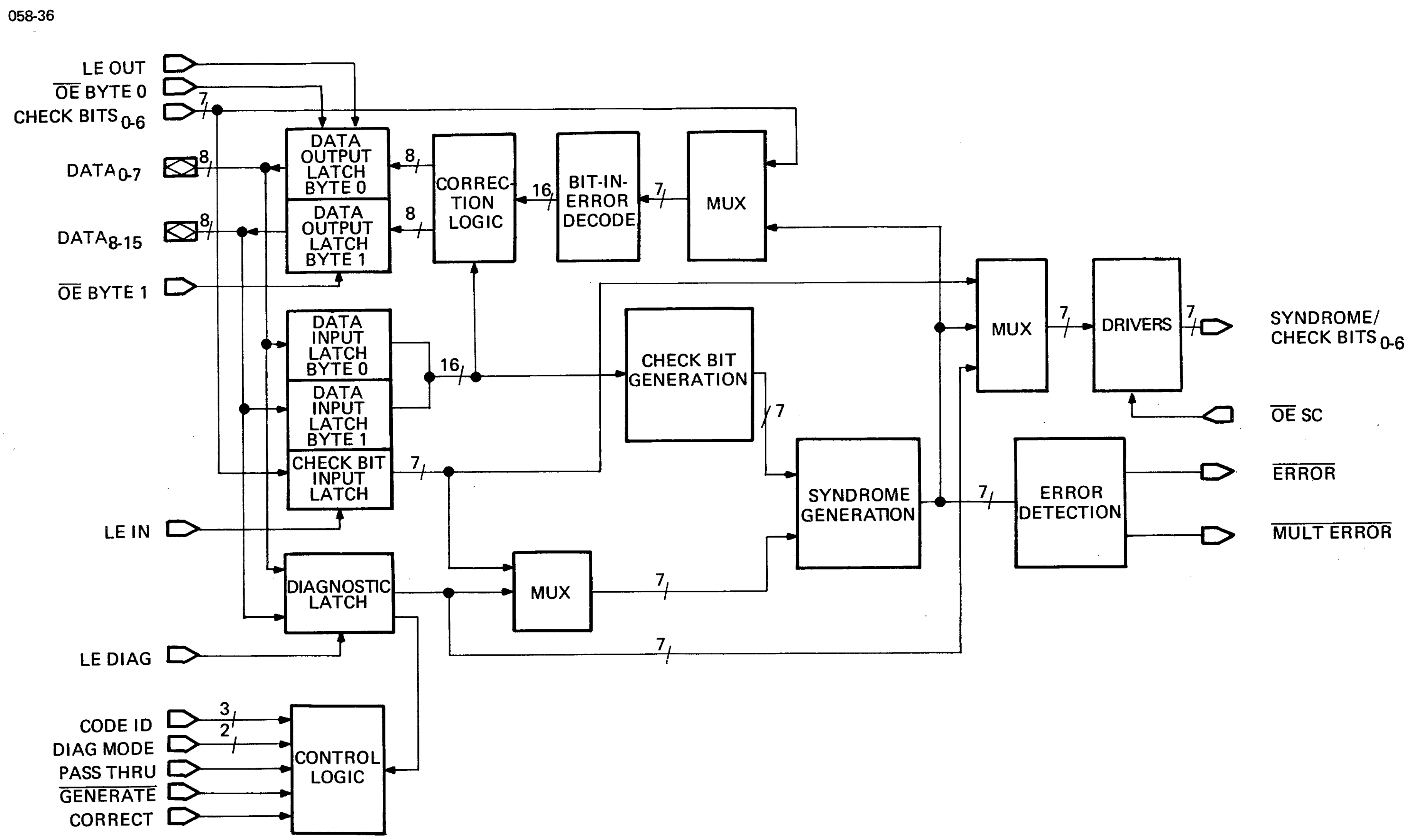


Figure 6-7 ECC Block Diagram

DATA		CHECK BITS					
BYTE 1	BYTE 2	CX	C0	C1	C2	C4	C8

Figure 6-8 16-Bit Data Format

TABLE 6-9 ECC CONTROL

CONTROL SIGNAL	DEFINITION
ECCOEO	Input ECCOEO enables the ECC output data latch after a single-bit error has been corrected. The ECC only drives the bus during the error correction cycle. 0 = Enable 1 = Tristate
ECCLOUTO	Input ECCLOUTO latches the ECC output data latch. 0 = Latch 1 = Transparent
GENO	Input GENO selects what mode the ECC is in, generate or correct. When GENO is low, the ECC generates the check bits to be stored with each memory word. When GENO is high, the ECC detects and corrects single-bit errors. Multiple-bit errors are also detected, but cannot be corrected.
PASS1	Input PASS1, when high, forces the contents of the check bit input latch (CB51:01) onto the syndrome check bit outputs (SC51:01) and the unmodified contents of the data input latch onto the inputs of the data output latch. This signal is used for XSTB instructions.
ERRO	Output ERRO is generated when the ECC detects a single-bit error in the current memory halfword. This signal is also generated on multiple errors.
MERRO	Output MERRO is generated when the ECC detects a multiple-bit error in the current memory halfword.

TABLE 6-10 CHECK BIT ENCODE CHART

GENERATED CHECK BITS SC(51:01)	PARITY	PARTICIPATING DATA BITS															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CX	Even (XOR)		X	X	X		X			X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X

The check bit is generated as either an XOR or XNOR of the eight data bits noted by an 'X' in the table.

6.8 ERROR LOGGER

The error logger (Sheet 8) is an 11-bit register that is loaded when an error has occurred on a memory read. The error logger inputs consist of five R bus bits (R23:19) indicating the present row in memory, and six parity bits (SC51:01) from the ECC indicating which bit or bits of memory data are incorrect. When a Read Error Logger instruction is executed, the data read reflects only the most recent error. Since the error logger is only one location deep, all previous errors have been overwritten.

058-38

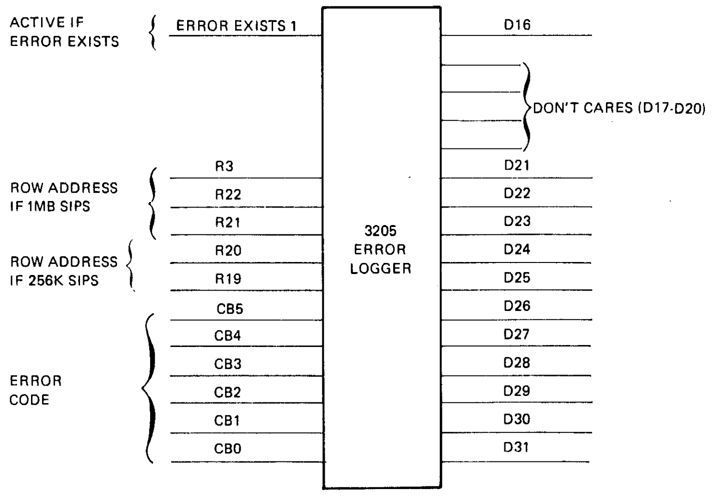


Figure 6-9 Error Logger

TABLE 6-11 ROW TO BIT-IN-ERROR

25	24	23	ROW
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Bits 31:26 are coded as shown in Table 6-12 and point to one of the 22 SIPs (16 data and 6 syndrome) where the error occurred.

TABLE 6-12 SYNDROME DECODE TO BIT-IN-ERROR

DATA BITS			S8	0	1	0	1	0	1	0	1
			S4	0	0	1	1	0	0	1	1
			S2	0	0	0	0	1	1	1	1
SX	S0	S1									
0	0	0		*	C8	C4	T	C2	T	T	M
0	0	1		C1	T	T	15	T	13	7	T
0	1	0		C0	T	T	M	T	12	6	T
0	1	1		T	10	4	T	0	T	T	M
1	0	0		CX	T	T	14	T	11	5	T
1	0	1		T	9	3	T	M	T	T	M
1	1	0		T	8	2	T	1	T	T	M
1	1	1		M	T	T	M	T	M	M	T

* = no errors detected
Number = the location of the single bit-in-error
T = two errors detected
M = three or more errors detected

6.9 MEMORY REFRESH

The dynamic nature of the RAM chips requires that memory refresh occur periodically. Refresh occurs every 14.4µs and is a two-cycle operation. A CAS before RAS refresh is used. CAS goes active at 50ns and RAS goes active at 150ns of the first cycle. Both strobes are removed at 150ns of the second cycle. Memory refresh functions are dependent on whether or not the processor is performing an operation that uses memory. There are three different states in which memory refresh takes place.

- When the processor is not performing a memory operation.

  The memory refresh PAL (A155)(16C9) produces REFCYCL0, which is sent to both the CAS PAL (A7)(7A2) and the RAS PAL (A9)(7A9).

- When the processor attempts to perform a memory operation during the second cycle of memory refresh.

  The memory refresh PAL (A155) produces REFFRZ0, which causes the CS and ALU clocks to freeze until completion of the memory refresh.

- When the processor is performing continuous memory operations.

  The memory refresh PAL (A155)(16C8) will wait eight cycles after the 14.4µs period for memory to become available. At the end of the eight cycles, if memory is still being accessed, the memory refresh PAL (A155)(16C8) produces REFRQ0. REFRQ0 is sent to the interrupt generator PAL (A158)(16J4), which produces the macrointerrupt REFINT0.

6.10 MEMORY ARRAY

The memory array is comprised of eight rows of SIPs (see Figure 6-10) with each row containing 22 SIPs. Each SIP contains a 64K x 1 RAM and all eight rows share common CAS, WE and address lines. The central processing unit (CPU) board contains the first two rows (1Mb) and the remaining six rows (3Mb) on the memory expansion board. During a memory access, every SIP receives one of the four possible RASs and then the appropriate CAS. The WE strobe is enabled for a write to memory operation.

058-37

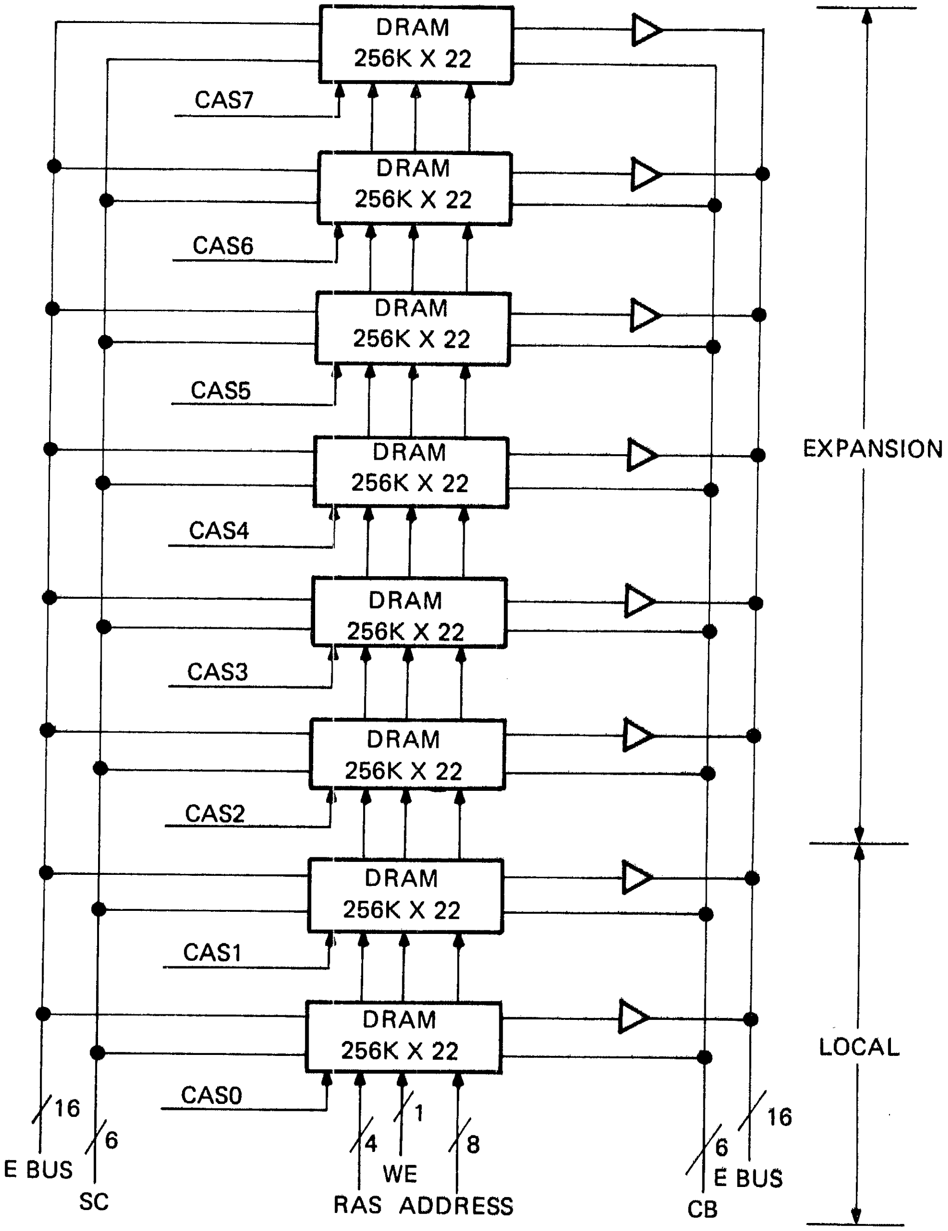


Figure 6-10 Memory Array

6.10.1 Memory Array Control

The memory array control generates the eight CAS lines, four RAS lines, eight WE lines and MUXs the eight row/column addresses for the memory array.

- The CAS generation PAL (A7) (3B1) decodes three 4-bit fields to generate the appropriate CAS. THe three most significant address bits (R21:19) informs the PAL which row of SIPs should receive the strobe. The memory CS field instucts the PAL if a memory array access is to be performed. The three clock signals (MST2:0) are used for timing. During a memory array access, CAS goes active at 150ns and for refresh, CAS is active at 50ns. This PAL monitors refresh signals and initiates a refresh cycle, when requested, if the memory array is not being accessed.

- The RAS PAL works similarly to the CAS PAL with two exceptions. This PAL uses R10:9 to determine which of the four strobes to activate. During a memory array access, RAS is active at 50ns and during a refresh cycle, RAS is active at 150ns.

- The write enable PAL decodes the identical fields that the CAS does, except that refresh does not have an affect on this PAL. The WE strobes are active at 100ns of the second cycle and deactivate at 150ns.

- The row/column MUX enables the row address on the bus from 0ns to 100ns and the column address on the bus from 100ns to 220ns.

CHAPTER 7
POWER UP/DOWN

7.1 INITIALIZE CONTROL

The initialize control allows the processor to power up, power down or initialize the processor logic in an orderly manner whenever power fail detect (PFDTO) is activated. Any one of the following activates PFDTO:

- Turning the consolette STANDBY-ON-LOCK switch to the STANDBY position.
- Operating the consolette initialize (INIT) switch.
- An external source such as the multiperipheral controller (MPC).
- An impending AC or DC voltage failure signaled by the power supply.
- A microprogrammed command power-down (POWO).

The initialize control logic is found on Sheet 31 of Functional Schematic 35-864 D08.

7.2 INITIALIZE SEQUENCE

If other than the power supply enables PFDTO, i.e., neither P5 nor P5U are going down, the processor is initialized. That is, when PFDTO (31C8) goes low, the processor goes through a power-down sequence as described in the power-down section. However, after SCLRO (31M6) is grounded and establishes the system components to predetermined states, the processor begins a power-up sequence, as described in the following sections. Note that an active SCLRO (31M6) overrides POWO, forces FEPF1 and FPPFO inactive and sets the BPFTD latch. Since P5U remains established, NVM1 (31L9) will not go active during the power-up sequence.

7.3 POWER-UP SEQUENCE

With all DC voltage down, the unenergized relay K1 (31L6) holds system clear (SCLR0) (31M6) at ground. When the consolette switch is turned from STANDBY to ON or LOCK, the power supply holds PFDT0 (31C8) low while the P5 and P5U voltage come up. Three forward biased diode drops serve as the voltage reference for the QUAD comparators (A32 (31E8), A63 (31E6)). A low on PFDT0 (31C8) holds pin 14 (A32) (31H8) low, which turns on transistor Q3 (31K6) to hold Q2 (31K6) cut-off. When P5 and P5U are fully established, the power supply drives PFDT0 (31C8) inactive and capacitor C74 (31G8) starts to charge P5U. Pin 14 (A32) is driven high when the voltage on C74 (31G8) reaches voltage reference level. Q3 (31K6) is cut-off and Q2 (31K6) starts conducting; K1 (31L6) is energized and SCLR0 (31M6) is pulled-up to P5 through a resistor. Cross-coupled power NAND gates (31L7) serve as a relay debounce circuit for SCLR0 (31M6).

During the power-up sequence, nonvalid memory (NVM1) (31L9) is tested by the microprogram. An active NVM1 (31L9) indicates that memory voltage has failed, and therefore, the data in memory is unreliable. Note that NVM1 (31L9) was latched high when P5U came up. With NVM1 (31L9) active, the microprogram goes into a cold start routine to initialize the first 512kb of memory and NVM1 (31L9) is driven low when the error logger is enabled (ERLOE0) (31K3). If NVM1 (31L9) is inactive on power-up, a cold start is not performed.

7.4 POWER-DOWN SEQUENCE

In normal operation, SCLR0 (31M6) is inactive. If the power supply detects that a voltage failure (power-down condition) is to occur, it forces PFDT0 (31C8) active, which causes C74 (31G8) to start discharging. Also, buffered power fail detect (BPFTD0) (31G9) is generated. BPFTD0 (31G9) sets a cross-coupled latch, begins a 1 millisecond time-out and forces CL70A0 (31F4) low. At the start of the time-out, the early power fail (FEPF1), flip-flop (A89)(31H1) is toggled active, which with PSW181 high, causes an early power fail interrupt. This interrupt alerts the user to perform a shut-down routine. Signal Cl70A0 (31F4) activates CL070 (clear) down the input/output (I/O) bus.

At the end of the 1 millisecond time-out, the primary power fail (FPPF0) flip-flop (A89)(31H3) is toggled active. This generates an unmasked interrupt to the microprogram to start a microcode shut-down. Upon completion of this shut-down, the microcode drives power-off (POW0) low, which resets a cross-coupled latch that inputs to pin 5 of a comparator (A63)(31F5). A low on this pin results in pin 14 on A63 (31H8) to be driven low to turn-on Q3 (31K6) and cut-off Q2 (31K6). The relay is deenergized and SCLR0 (31M6) goes low. If POW0 is not driven active, the voltage across C74 (31G8) eventually decays below voltage reference and causes pin 14 (A32)(31H8) low to deenergize the relay.

APPENDIX A
MNEMONICS

A.1 INTRODUCTION

The following is a list of the mnemonics used on the processor board. The meanings and 35-864 D08 schematic sources of each signal are provided.

MNEMONIC	MEANING	SCHEMATIC LOCATION
ACNT 01:11	Address register select	16
AFLT0	Memory alignment fault 1=Halfword alignment fault 2=Fullword alignment fault	16K4
ALUCT1	Arithmetic logic unit (ALU) conditional test	12D4
ARWEB1	Address register write enable byte	26D7
ARWEBE0	Address register write enable byte enable	16K2
ARWEH1	Address register write enable halfword	26D7
ARWEHE0	Address register write enable halfword enable	16K2
ATN0	Input/output (I/O) attention	29A3
BADSTAT1	Private multiplexor (PMUX) bad status	29E7
BE0	Multiplexor (MUX) bus enable	28E1
BLOCK0	Block register external (REX) write	16
BPFDT0	Power shut-down	31
SBUSY0	Direct memory access (DMA) busy	28K8

MNEMONIC	MEANING	SCHEMATIC LOCATION
C231:001	Address register input bus	15
CASOBO:1FO	Column address strobe	7
CASOO:70	Column address strobe	7C1
CB01:51	Error check and correction (ECC) check bits	6M9
CBEO	Error check and correction (ECC) check bit enable	3G9
CCATNO	Clear console attention	24
CCMO	Code compatible machine 0=CCM Mode 1=PE Mode	23
CCOEO	Condition code output enable	24
CCR00:30	Condition code register output	23
CCXH1 CCXL1	Condition code register and MUX control lines	23
CJVO	Condition jump vector	20
CLRTAGO	Clear TAG chip	8
CMDO	I/O command lines	28L5
CNT01:11	Address register select	11
CNT61:81	Memory refresh counter lines	25
CROE	Control register output enable	12K7
DAO	Data available	28L5
DBL1	Double precision floating point (DPFP)	24
DCTLLEO	DMA counter load low	24
DCTHLEO	DMA counter load high	24
DIRTO	Memory write bit	16
DMAO	A DMA request exists 1=DMA read 2=DMA write	

MNEMONIC	MEANING	SCHEMATIC LOCATION
DMABST0	DMA bad status	28E3
DMABZY0	DMA busy signal	28E3
DMACNT0	DMA count	28E1
DMAE10:30	DMA enable (PMUX)	28J1
DMAEND0	DMA good termination	14
DMAF1X0 DMAFX10	DMA function control lines	28E3
DMAINT1	DMA interrupt	28E3
DMARD0	DMA read/write select	28E3
DOE0	Data output enable	03E8
ECCD0	ECC multiple-bit memory read error	16
EME0	E bus to M bus enable	03E8
ERR0	Memory error flag	03M5
ERRB0	Memory error flag	03M7
FBMB0	Bomb function	18E8
FULL0	Microaddress sequencer stack full	21M4
GEN0	ECC generate/correct select	03E7
H/B0:1	Halfword/byte select	08D4
HIT1	MAT hit1	15K1
IRHLE0	Instruction register high latch enable	08D8
IRLLE0	Instruction register low latch enable	08D8
ISERV1	Interrupt service warning	22G8
IVEC0	Interrupt vector warning	22M3

MNEMONIC	MEANING	SCHEMATIC LOCATION
JACKO	R2M transceiver byte enable	16C6
LDCO	Load counter (AR)	16L1
MATE1	Memory address translator (MAT) enable	08D5
MATPRT1	MAT parity fault	18M8
MATWEBO	MAT write enable byte	16C7
MATWEHO	MAT write enable halfword	16C7
MEMOPO:1	Memory operation	08D2
MERRO	Multiple memory error	03L5
MFLTO	MAT fault microinterrupt	16K4
MIKEO	R2M transceiver halfword enable	16C6
MINTO	Macrointerrupt pending	20E1
MINTOEO	Macrointerrupt vector output enable	20E1
MISSO	MAT miss microinterrupt	16K3
MSKT1	Mask true (branching)	23J3
MSKF1	Mask false (branching)	23J3
MTNBLDO	MAT enabled	10E9
MTWEHO	MAT write enable halfword	10E9
NMEMO	Nonpresent memory microinterrupt	16K4
NOMEMO	Nonpresent microinterrupt	07D6
OPOEO	Opcode output enable	20E3
OUTEO1:11	Memory array output enable	07G1
PASS1	ECC pass mode	03K6
PRC1	P register clock	24M8

MNEMONIC	MEANING	SCHEMATIC LOCATION
PRODOEO	Multiple product output enable	24F2
PROMO	Constant PROM selected	24D5
PROMCEO	Constant PROM chip enable	24D5
RAS00:31B0	Row address strobe	Sheet 07
REFCCL20	Refresh cycle two	16K1
REFCYCLO	Refresh cycle pending	16D9
REFFRZO	Refresh master clock freeze	16D9
REFINTO	Refresh interrupt	16K3
REFRQ0	Refresh request	16C8
REPFO	Reset early power fail	24L5
REXPO	REX parity error	18E9
R/W0:1	Read/write select	8D5
SDHO	S bus driver select high	20E6
SDLO	S bus driver select low	20E6
SETSNGLO	Set single-step function	24H7
SRCIN1	Shift register carry-in	24M3
SRESETO	Shift register reset	25N4
STICKY1	4-bit shift right carry out = 1	20E5
TAGWEO	TAG chip write enable	10E4
WAITO	Wait light enable	14F4
WRTELG1	Write error logger	03L7

INDEX

PERKIN-ELMER

PUBLICATION COMMENT FORM

We try to make our publications easy to understand and free of errors. Our users are an integral source of information for improving future revisions. Please use this postage paid form to send us comments, corrections, suggestions, *etc.*

1. Publication number______________________________

2. Title of publication______________________________

3. Describe, providing page numbers, any technical errors you found. Attach additional sheet if neccessary.

____

____

____

4. Was the publication easy to understand? If no, why not?

____

5. Were illustrations adequate? ________________________

____

____

6. What additions or deletions would you suggest? _____________

____

____

7. Other comments: ______________________________

____

____

From ________________________ Date __________________

Position/Title ____________________

Company ______________________

Address ______________________

____

____

STAPLE STAPLE

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 22 OCEANPORT, N.J.

POSTAGE WILL BE PAID BY ADDRESSEE

PERKIN-ELMER

Data Systems Group
106 Apple Street
Tinton Falls, NJ 07724

ATTN:
TECHNICAL SYSTEMS PUBLICATIONS DEPT.

FOLD FOLD

STAPLE STAPLE