Preliminary

Single-Chip 8-BIT CMOS Microcontroller
M37640E8-XXXFP Specification

Ver 1.02

MITSUBISHI SEMICONDUCTOR AMERICA, INC.

Rev. 1.0 Internal Release	April 2, 1997
Rev. 1.01 Design Spec Updates	July 1, 1997
Rev. 1.02 Design Spec Updates	August 28, 1997

Contents

1 Product Description

1.1 MCU Features 4
1.2 Pin Description and Layout 6

2 Functional Description

2.1 Central Processing Unit 11
2.1.1 Register Structure 11
2.1.2 Accumulator (A) 11
2.1.3 Index Registers X and Y 12
2.1.4 Stack Pointer 12
2.1.5 Program Counter 13
2.1.6 Processor Status Register 13
2.2 CPU Mode Registers 14
2.3 Memory Map 16
2.3.1 Zero page 17
2.3.2 Special Page 17
2.3.3 Special Function Registers 17
2.4 Processor Modes 19
2.4.1 Single Chip 19
2.4.2 Memory Expansion 20
2.4.3 Microprocessor 20
2.4.4 EPROM 20
2.4.5 Slow Memory Wait 20
2.4.6 Hold Function 23
2.4.7 Expanded Data Memory Access 24
2.5 Peripheral Interface 25
2.5.1 Chip Bus Timing 25
2.5.2 Peripheral Interface and Access Timing 26
2.6 Input and Output Ports 28
2.6.1 Ports 28
2.6.1.1 I/O Ports 29
2.6.1.2 Power and Ground Pins 31
2.6.1.3 CNV_{SS} Pin 31
2.6.1.4 X_{in} and Xout Pins 31
2.6.1.5 X_{Cin} and XCout Pins 31
2.6.1.6 RESET Pin 31
2.6.1.7 RDY Pin 31
2.6.1.8 DMAout Pin 31
2.6.1.9 F_{out} Pin 32
2.6.1.10 $SYNC_{out}$ PinC 32
2.6.1.11 RD and WR Pins 32
2.6.1.12 LPF Pin 32
2.6.2 Port Control Register 32
2.6.3 Port P2 Pull-up Control Register 33
2.7 Interrupt Control Unit 33
2.7.1 Interrupt Control 34
2.7.2 Interrupt Sequence and Timing 36
2.8 Direct Memory Access Controller 38
2.8.1 Operation 40
2.8.1.1 Source, Destination, and Transfer Count Register Operation 40
2.8.1.2 DMAC Transfer Request Sources 41
2.8.1.3 Transfer Features for USB and Master CPU Bus Interface 42
2.8.1.4 DMAC Transfer Mode 43
2.8.1.5 DMAC Transfer Timing 43
2.9 Timers 48
2.9.1 Timer X 48
2.9.1.1 Read and Write Method 48
2.9.1.2 Count Stop Control 49
2.9.1.3 Timer Mode 49
2.9.1.4 Pulse Output Mode 49
2.9.1.5 Event Counter Mode 50
2.9.1.6 Pulse Width Measurement Mode 50
2.9.2 Timer Y 50
2.9.2.1 Read and Write Method 51
2.9.2.2 Count Stop Control 51
2.9.2.3 Timer Mode 51
2.9.2.4 Pulse Period Measurement Mode 52
2.9.2.5 Event Counter Mode 52
2.9.2.6 HL Pulse-width Measurement Mode 52
2.9.3 Timer 1 53
2.9.3.1 Timer Mode 53
2.9.3.2 Pulse Output Mode 53
2.9.4 Timer 2 53
2.9.4.1 Timer Mode 54
2.9.4.2 Pulse Output Mode 54
2.9.5 Timer 3 54
2.9.5.1 Timer Mode 54
2.10 Universal Serial Bus 56
2.10.1 USB Function Control Unit 57
2.10.1.1 Serial Interface Engine 57
2.10.1.2 Generic Function Interface 57
2.10.1.3 Serial Engine Interface Unit 57
2.10.1.4 Microcontroller Interface Unit 57
2.10.1.5 USB Transceiver 57
2.10.2 USB Interrupts 58
2.10.2.1 USB Function Interrupt 58
2.10.2.2 USB SOF Interrupt 59
2.10.3 USB Endpoint FIFOs 59
2.10.3.1 Transmit FIFOs 59
2.10.3.2 Receive FIFOs 60
2.10.4 USB Special Function Registers 60
2.11 Master CPU Bus Interface 68
2.11.1 Data Bus Buffer Status Registers (DBBS0, DBBS1) 71
2.11.2 Input Data Bus Buffer Registers (DBBIN0, DBBIN1) 71
2.11.3 Output Data Bus Buffer Registers (DBBOUT0, DBBOUT1) 71
2.12 UART 72
2.12.1 Baud Rate Selection 73
2.12.2 UART Mode Register 74
2.12.3 UART Baud Rate Register 74

Contents

2.12.4 UART Control Register 75
2.12.5 UART Status Register 75
2.12.6 Transmit/Receive Format 78
2.12.7 Interrupts 78
2.12.8 Clear-to Send (CTS) and Request-to-Send (RTS) Signals 79
2.12.9 UART Address Mode 80
2.13 Serial I/O 81
2.13.1 SIO Control Register 81
2.13.2 SIO Operation 81
2.14 Special Count Source Generator 84
2.14.1 SCSG Operation 84
2.14.2 SCSG Description 85
2.14.2.1 SCSG1 85
2.14.2.2 SCSG2 85
2.15 Oscillator Circuit 86
2.15.1 Description 86
2.15.2 Frequency Synthesizer Circuit 89
2.16 Low Power Modes 91
2.16.1 Stop Mode 91
2.16.2 Wait Mode 92
2.17 Reset 93
2.18 Key-On Wake-Up 94

3 Electrical Characteristics

3.1 Absolute Maximum Rating 97
3.2 Recommended Operating conditions 97
3.3 Electrical Characteristics 99

4 Application Notes

4.1 DMAC 105
4.1.1 Programming 105
4.1.2 Application 105
4.2 UART 106
4.2.1 Application 106
4.3 Timer 106
4.3.1 Usage 106
4.4 Frequency Synthesizer Interface 107

MITSUBISHI SEMICONDUCTOR
AMERICA, INC.

PRELIMINARY

CHAPTER 1

PRODUCT DESCRIPTION

1 Overview 3

1.1 MCU Features 4

1.2 Pin Description and Layout . . . 6

1 Overview

The 7600 series, an enhanced family of CMOS 8-bit microcontrollers, offers high-speed operation at low voltage, large internal-memory options, and a wide variety of standard peripherals. The series is code compatible with the M38000, M37200, M37400, and the M37500 series, and provides many performance enhancements to the instruction set.

This device is a single chip PC peripheral microcontroller based on the Universal Serial Bus (USB) Version 1.0 specification. This device provides data exchange between a USB-equipped host computer and PC peripherals such as telephones, audio systems and digital cameras.

The USB function control unit can support all four data transfer types listed in the USB specification: Control, Isochronous, Interrupt, and Bulk. Each transfer type is used for controlling a different set of PC peripherals. Isochronous transfers provide guaranteed bus access, a constant data rate, and error tolerance for devices such as computer-telephone integration (CTI) and audio systems. Interrupt transfers are designed to support human input devices (HID) that communicate small amounts of data infrequently. Bulk transfers are necessary for devices such as digital cameras and scanners that communicate large amounts of data to the PC as bus bandwidth becomes free. Finally, control transfers are supported and are useful for bursty, host-initiated type communication where bus management is the primary concern.

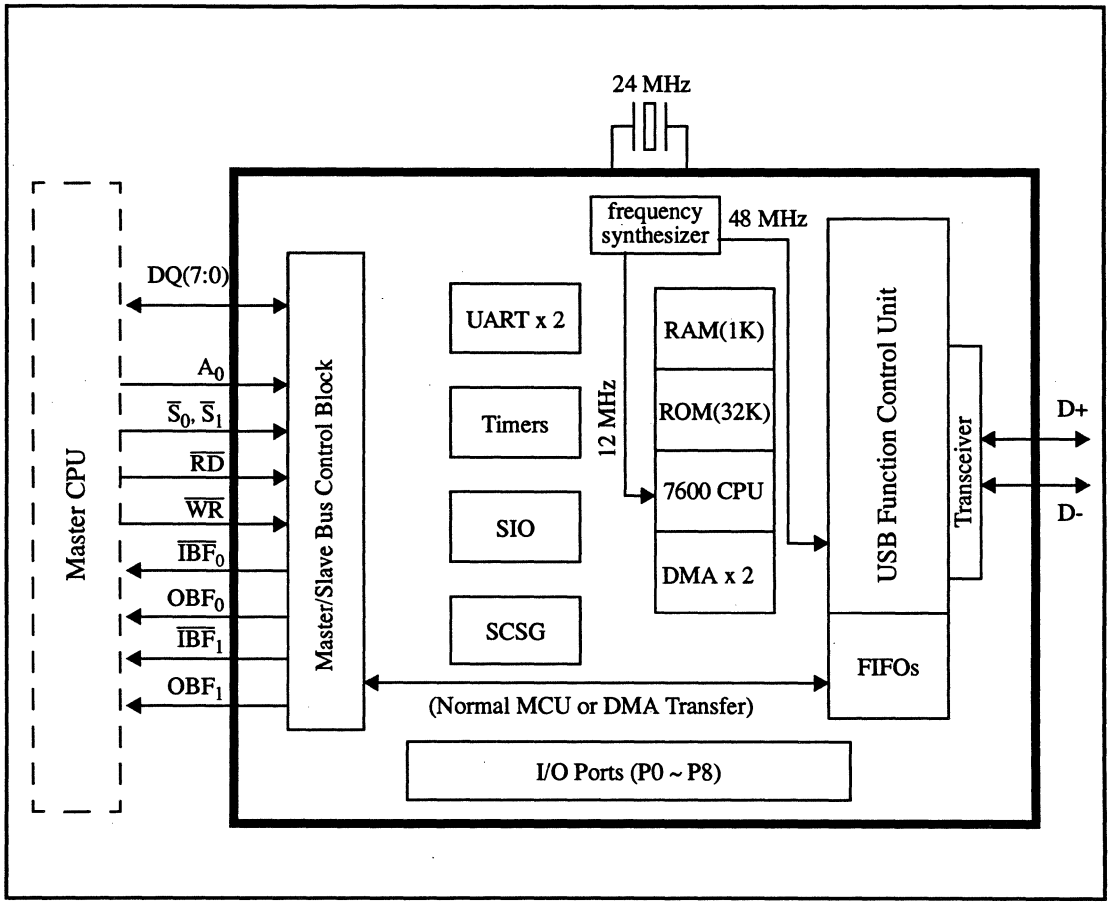


Figure 1-1. Application System Diagram

Table 1-1. Device Feature List

Parameter			Function Description
Number of basic instruction			71
Instruction execution time			83ns at f(X_{in}) = 24 MHz
Clock frequency (maximum)			Xin = 24 MHz, XC_{in} = 5 MHz (digital input), Φ= 12 MHz
Clock multiplier option			External clock X_{in} and XC_{in} can be selectively divided and multiplied by X to create system internal clock Φ
Memory size	ROM		32K bytes
	RAM		1K bytes
Input/Output ports	P0~P3, P5, P6, P8	I/O	8-bit X 7 (8 bits have Key-on Wake-up)
	P4	I/O	5-bit
	P7	I/O	5-bit
USB Function Control			FIFO: Endpoint 0: IN 16-byte OUT 16-byte Endpoint 1: IN 512-byteOUT 800-byte Endpoint 2: IN 32-byte OUT 32-byte Endpoint 3: IN 16-byte OUT 16-byte Endpoint 4: IN 16-byte OUT 16-byte
Master CPU bus interface			DQ(7:0), $\overline{R}$(E), $\overline{W}$(R/$\overline{W}$), $\overline{S}_0$, $\overline{S}_1$, A_0, $\overline{IBF}_0$, OBF_0, $\overline{IBF}_1$, OBF_1; total of 17 signals interface with master CPU (Intel 8042-like interface)
Special Count Source Generator(SCSG)			Baud rate synthesizer
UART 1			7/8/9-bit character length, with $\overline{CTS}$, $\overline{RTS}$ available
UART 2			7/8/9-bit character length, with $\overline{CTS}$, $\overline{RTS}$ available
Serial I/O			8-bit X 1 clock synchronous serial I/O, supports both master and slave modes
Timers			8-bit X 3, 16-bit X 2
DMA			2 channels, 16 address lines, support single byte or burst transfer modes
Software slew rate control			Ports P0 ~ P8
Interrupts			4 external, 19 internal, 1 software, 1 system interrupts
Supply voltage			V_{cc} = 4.5 ~ 5.5V
External memory expansion			Memory Expansion and Microprocessor mode
External Data Memory Access (EDMA)			Allows > 64 Kbyte data access for instruction LDA (indY) and STA (indY)
Device structure			CMOS
Package			80P6N
Operating temperature range			-20 to 85°C

1.1 MCU Features

- 7600 8-bit CPU core, CMOS process
- Instruction Execution Time of 83ns (1-cycle instruction @ Φ = 12 MHz)
- Efficient Software Support (C and/or Assembly)
- ROM: 32 KB On-chip
- RAM: 1 KB On-chip

- Built-in Microprocessor or Memory-expansion modes
- Three Slow Memory Wait modes: Software Wait, RDY Wait, and Extended RDY Wait
- Nine I/O Ports, Total 66 Programmable I/O Pins available
  - Programmable Direction Control on every I/O pin
  - Software Slew Rate Control on every I/O pin
- Master CPU Bus Interface:
  - MCU can be operated in Slave mode by control signals from the Host CPU
  - 8 Data lines (DQ7-DQ0) and $\overline{R}$(E), $\overline{W}$(R/$\overline{W}$), A_0, $\overline{S}_0$, $\overline{S}_1$, $\overline{IBF}_0$, OBF_0, $\overline{IBF}_1$, OBF_1 Signals Available
  - Master CPU Sends and Receives Data, Command and Status by means of DQ7-DQ0
- USB Function Control Unit
- USB Transceiver (conforms to USB V1.0 Specification)
- DMA Controller:
  - Two DMA channels available
  - 16 Address Lines for 64K byte Address Space
  - Single Byte or Burst Transfer modes
  - Transfer Request by external pins, Software Triggers or Built-in Peripherals
  - Maximum 6M byte/sec transfer Speed (in Burst mode)
- Timers: Three 8-bit Timers, Two 16-bit Timer available On-chip
- Two Full Duplex UARTs available
- One Master/Slave Clock Synchronous I/O (SIO), Internal or External Clock Selectable
- Built-in Special Count Source Generator (SCSG): can be a clock source for Timer X, UARTs, and SIO
- Power-saving Wait (IDLE) and Stop (Powerdown) modes.

1.2 Pin Description and Layout

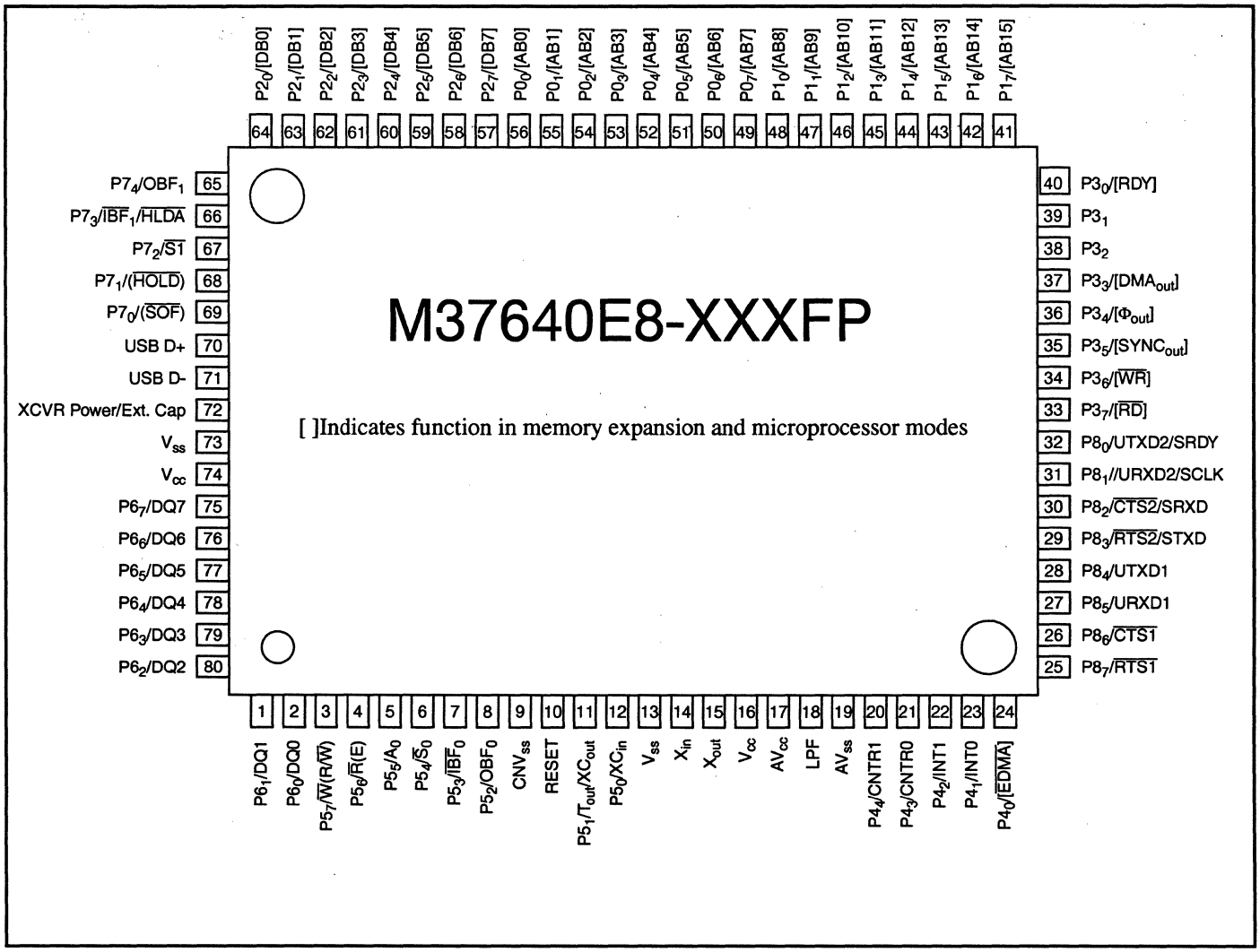


Figure 1-2. Pin Layout

Table 1-2. Pin Description

Name	I/O	Description
P0$_0$/AB0 ~ P1$_7$/AB15	I/O	CMOS I/O port (address bus). When the MCU is in memory expansion or microprocessor mode, these pins function as the address bus.
P2$_0$/DB0 ~ P2$_7$/DB7	I/O	CMOS I/O port (data bus). When the MCU is in memory expansion or microprocessor mode, these pins function as the data bus. These pins may also be used to implement the Key-on Wake up function.
P3$_0$/RDY	I/O	CMOS I/O port (Ready). When the MCU is in memory expansion or microprocessor mode, this pin functions as RDY (hardware wait cycle control).
P3$_1$	I/O	CMOS I/O port.
P3$_2$/(VRFY)	I/O	CMOS I/O port. When the MCU is in EPROM program mode, the pin is used as VRFY (EPROM memory verify).
P3$_3$/DMAout /PGM	I/O	CMOS I/O port (DMAout). When the MCU is in memory expansion or microprocessor mode, this pin goes high during a DMA transfer. When the MCU is in EPROM program mode, the pin is used as $\overline{PGM}$ (EPROM memory program).

Table 1-2. Pin Description

Name	I/O	Description
$P3_4/\Phi_{out}$	I/O	CMOS I/O port (Φ). When the MCU is in memory expansion or microprocessor mode, this pin becomes Φout pin.
$P3_5/SYNC_{out}$	I/O	CMOS I/O port (SYNC output). When the MCU is in memory expansion or microprocessor mode, this pin becomes the SYNCout pin.
$P3_6/\overline{WR}/(\overline{CE})$	I/O	CMOS I/O port. ($\overline{WR}$ output). When the MCU is in memory expansion or microprocessor mode, this pin becomes $\overline{WR}$. When the MCU is in EPROM program mode, the pin is used as $\overline{CE}$ (EPROM memory chip enable).
$P3_7/\overline{RD}/(\overline{OE})$	I/O	CMOS I/O port. ($\overline{RD}$ output). When the MCU is in memory expansion or microprocessor mode, this pin becomes $\overline{RD}$. When the MCU is in EPROM program mode, the pin is used as $\overline{OE}$ (EPROM memory output enable).
$P4_0/\overline{EDMA}$	I/O	CMOS I/O port ($\overline{EDMA}$: expanded Data Memory Access). When the MCU is in memory expansion or microprocessor mode, this pin can become the $\overline{EDMA}$ pin.
$P4_1$/INT0 ~ $P4_2$/INT1	I/O	CMOS I/O port or external interrupt ports INT0 and INT1. These external interrupts can be configured active high or low.
$P4_3$/CNTR0	I/O	CMOS I/O port or Timer X input pin for pulse width measurement mode and event counter mode or Timer X output pin for pulse output mode. This pin can also be used as an external interrupt when Timer X is not in output mode and the polarity is selected in the Timer X mode register.
$P4_4$/CNTR1	I/O	CMOS I/O port or Timer Y input pin for pulse period measurement mode, pulse H-L measurement mode and event counter mode or Timer Y output pin for pulse output mode. This pin can also be used as an external interrupt when Timer Y is not in output mode and the polarity is selected in the Timer Y mode register.
$P5_0/XC_{in}$	I/O	CMOS I/O port or XC_{in}.
$P5_1/T_{out}$/ XC_{out}	I/O	CMOS I/O port or timer 1 pulse output pin (can be configured initially high or initially low), or XC_{out}.
$P5_2/OBF_0$	I/O	CMOS I/O port or OBF_0 output to master CPU for data bus buffer 0.
$P5_3/\overline{IBF_0}$	I/O	CMOS I/O port or $\overline{IBF_0}$ output to master CPU for data bus buffer 0.
$P5_4/\overline{S_0}$	I/O	CMOS I/O port or $\overline{S_0}$ input from master CPU for data bus buffer 0.
$P5_5/A_0$	I/O	CMOS I/O port or A_0 input from master CPU.
$P5_6/\overline{R}$(E)	I/O	CMOS I/O port or $\overline{R}$(E) input from master CPU.
$P5_7/\overline{W}$(R/$\overline{W}$)	I/O	CMOS I/O port or $\overline{W}$(R/$\overline{W}$) input from master CPU.
$P6_0$/DQ0 ~ $P6_7$/DQ7	I/O	CMOS I/O port or master CPU data bus.
USB D^-	I/O	USB minus voltage line interface, a series resistor of 15-20 ohms should be connected to this pin.
USB D^+	I/O	USB plus voltage line interface, a series resistor of 15-20 ohms should be connected to this pin.
$P7_0/\overline{SOF}$	I/O	CMOS I/O port or USB start of frame pulse output, an 80 ns pulse outputs on this pin for every USB frame.
$P7_1/\overline{HOLD}$	I/O	CMOS I/O port or $\overline{HOLD}$ pin.
$P7_2/\overline{S_1}$	I/O	CMOS I/O port or $\overline{S_1}$ input from master CPU for data bus buffer 1.
$P7_3/\overline{IBF_1}$/ HLDA	I/O	CMOS I/O port or $\overline{IBF_1}$ output to master CPU for data bus buffer 1, or HLDA pin. $\overline{IBF_1}$ and HLDA are mutually exclusive. $\overline{IBF_1}$ has priority over HLDA.
$P7_4/OBF_1$	I/O	CMOS I/O port or OBF_1 output to master CPU for data bus buffer 1.
$P8_0$/UTXD2/ SRDY	I/O	CMOS I/O port or UART2 pin UTXD2 or SIO pin SRDY. UART2 and SIO are mutually exclusive, UART2 has priority over SIO.
$P8_1$/URXD2/ SCLK	I/O	CMOS I/O port or UART2 pin URXD2 or SIO pin SCLK. UART2 and SIO are mutually exclusive, UART2 has priority over SIO.
$P8_2/\overline{CTS2}$/ SRXD	I/O	CMOS I/O port or UART2 pin $\overline{CTS2}$ or SIO pin SRXD. UART2 and SIO are mutually exclusive, UART2 has priority over SIO.
$P8_3/\overline{RTS2}$/ STXD	I/O	CMOS I/O port or UART2 pin $\overline{RTS2}$ or SIO pin STXD. UART2 and SIO are mutually exclusive, UART2 has priority over SIO.
$P8_4$/UTXD1	I/O	CMOS I/O port or UART1 pin UTXD1.
$P8_5$/URXD1	I/O	CMOS I/O port or UART1 pin URXD1.
$P8_6/\overline{CTS1}$	I/O	CMOS I/O port or UART1 pin $\overline{CTS1}$.

Table 1-2. Pin Description

Name	I/O	Description
$P8_7$/RTS1	I/O	CMOS I/O port or UART1 pin RTS1.
AV_{cc},AV_{ss}	I	Power supply inputs for analog circuitry.
CNV_{ss}	I	Controls the processor mode of the chip. Normally connected to V_{ss} or V_{cc}.
V_{cc},V_{ss}	I	Power supply inputs: V_{cc} = 4.5~ 5.5V, V_{ss} = 0V.
$\overline{RESET}$	I	To enter the reset state, this pin must be kept L for more that 2µs (20 Φ cycles under normal V_{cc} conditions). If the crystal or ceramic resonator requires more time to stabilize, extend this L level time appropriately.
XC_{in} XC_{out}	I O	An external ceramic or quartz crystal oscillator is connected between the XC_{in} and XC_{out} pins. If an external clock source is used, connect the clock source to the XC_{in} pin and leave the XC_{out} pin open.
X_{in} X_{out}	I O	Input and output signals to and from the internal clock generation circuit. Connect a ceramic resonator or quartz crystal between X_{in} and X_{out} pins to set the oscillation frequency. If an external clock is used, connect the clock source to the X_{in} pin and leave the X_{out} pin open.
LPF	O	Low pass filter for the frequency synthesizer.
XCVR Power/ Ext. Cap	I	1) A 3.3V line driver power supply (default after reset) pin 2) An external capacitor (Ext. Cap) pin. If V_{dd} (AV_{dd}) =5V is used for the entire chip, (no separate 3.3V power supply), a 1µf or larger capacitor should connect between this pin and V_{ss} to ensure proper operation of the USB line driver. This option is enabled by setting bit 4 of the USB control register (0013_{16}) High.

D+/D- Line driver notes: In order to match the USB cable impedance, a series resistor of 15~20Ω should be connected to each USB line; i.e. on D+ (pin 70) and on D- (pin 71). If the USB line is improperly terminated or not matched, signal fidelity will suffer, resulting in excessive overshoot or undershoot. This will potentially introduce bit errors.

PRELIMINARY

Chapter 2

Functional Description

2.1 Central Processing Unit.....11
2.2 CPU Mode Registers.....14
2.3 Memory Map.....16
2.4 Processor Modes.....19
2.5 Peripheral Interface.....25
2.6 Input and Output Ports.....28
2.7 Interrupt Control Unit.....33
2.8 Direct Memory Access Controller. 38
2.9 Timers.....48
2.10 Universal Serial Bus.....56
2.11 Master CPU Bus Interface....68
2.12 UART.....72
2.13 Serial I/O.....81
2.14 Special Count Source Generator 84
2.15 Oscillator Circuit.....86
2.16 Low Power Modes.....91
2.17 Reset.....93
2.18 Key-On Wake-Up.....94

2 Functional Description

2.1 Central Processing Unit

The central processing unit (CPU) has six registers:

- Accumulator (A)
- Index Register X (X)
- Index Register Y (Y)
- Stack Pointer (S)
- Processor Status Register (PS)
- Program Counter (PC)

2.1.1 Register Structure

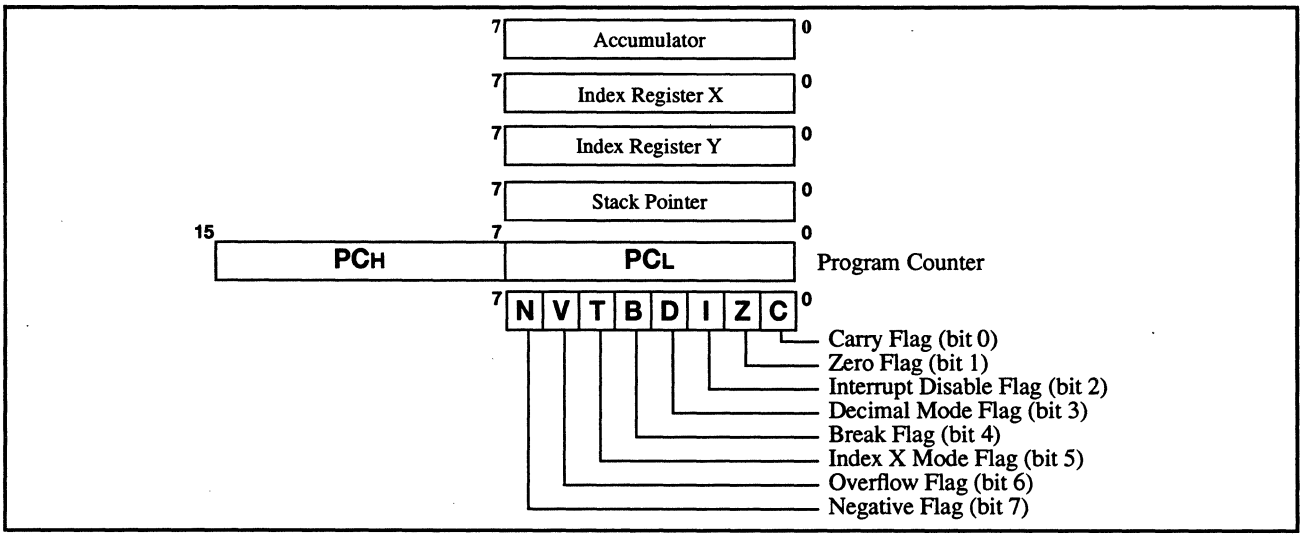


Figure 2-1. Register Structure

Five of the CPU registers are 8-bit registers, Accumulator (A), Index register X (X), Index register Y (Y), Stack pointer (S), and the Processor Status register (PS).

The PC is a 16-bit register consisting of two 8-bit registers (PCH and PCL) (see Figure 2-3.).

After a hardware reset, bit 2 (the I flag) of the PS is set High and the values at the addresses $FFFA_{16}$ and $FFFB_{16}$ are stored in the PC, but the values of the other bits of the PS and the other registers are undefined. Initialization of undefined registers may be necessary for some programs.

2.1.2 Accumulator (A)

The accumulator is the main register of the microcomputer. Data operations such as data transfer, input/output, and so forth, are executed mainly through the accumulator.

2.1.3 Index Registers X and Y

Both index registers X and Y are 8-bit registers. In the absolute addressing modes, the contents of these registers are added to the value of the OPERAND to specify the real address.

In the indirect X addressing mode, the value of the OPERAND is added to the contents of register X to specify the zero page basic address. The data at the basic address specifies the real address.

In the indirect Y addressing mode, the value of the operand specifies a zero page address. The data at this address is added to the contents of register Y to produce the real address. These addressing modes are useful for referencing subroutine tables and memory tables.

When the T flag in the processor status register is set High, the value contained in index register X points to a zero page memory location that replaces the accumulator for most accumulator based instructions.

2.1.4 Stack Pointer

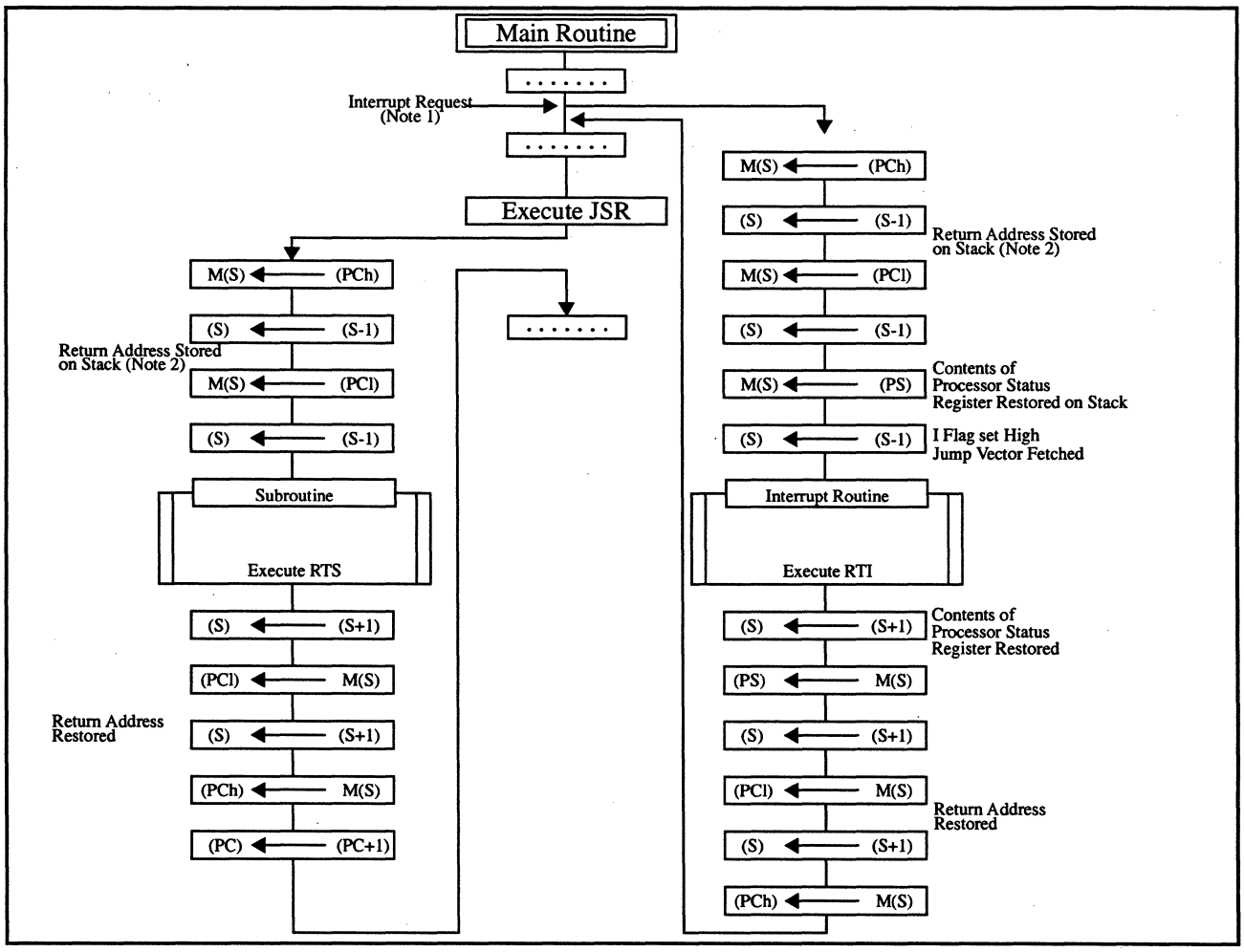


Figure 2-2. Register Push and Pop when Servicing Interrupts and Calling Subroutines

Note 1. The condition to enable an interrupt: Interrupt enable bit is High and Interrupt inhibit flag (I flag) is Low.

Note 2. When an interrupt occurs, the address of the next instruction to be executed is stored on the stack. When a subroutine is called, the address of (next instruction -1) to be executed is stored on the stack.

The stack pointer is an 8-bit register used during subroutine calls and interrupts. The stack is used to store the current address data and processor status when branching to subroutines or interrupt routines. The lower eight bits of the stack address are determined by the contents of the stack pointer. The upper eight bits of the stack address are determined by the Stack Page Select Bit, bit 2 of the CPU Mode Register A. If the Stack Page Select bit is "0", then the RAM in the zero page (addresses 0070_{16} to $00FF_{16}$) is used as the stack area. If the stack page select bit is High (the default value), then the RAM in one page (addresses 0100_{16} to $01FF_{16}$) is used as the stack area. The base of the stack must be set in software, and stack grows towards lower addresses from that point. The operations of pushing register contents onto the stack and popping them from the stack are shown in Figure 2-4.

2.1.5 Program Counter

The program counter (PC) is a 16-bit register consisting of two 8-bit sub-registers PCH and PCL. It is used to indicate the address of the next instruction to be executed.

2.1.6 Processor Status Register

The processor status (PS) register is an 8-bit register consisting of flags that indicate the status of the processor after an arithmetic operation. Branch operations can be performed by testing the Carry (C), Zero (Z), Overflow (V), or the Negative (N) flags.

After reset, the I flag is set High, but all other flags are undefined. Because the T and D flags directly affect arithmetic operations, they should be initialized in the beginning of a program.

Carry Flag (C)
The C flag contains a carry or borrow generated by the arithmetic logic unit (ALU) immediately after an arithmetic operation. It is also affected by shift and rotate instructions. The C flag can be set directly by the set carry (SEC) instruction and cleared by the clear carry (CLC) instruction.

Zero Flag (Z)
The Z flag is set if the result of an arithmetic operation or a data transfer is "0", and cleared if the result is anything other than "0".

Interrupt Disable Flag (I)
The I flag disables all interrupts except for the interrupt generated by the BRK instruction and any non-maskable interrupts, if available. Interrupts are disabled when the I flag is High. When an interrupt occurs, this flag is automatically set High to prevent other interrupts from interfering until the current interrupt service routine is completed. The I flag can be set by the set interrupt disable (SEI) instruction and cleared by the clear interrupt disable (CLI) instruction.

Decimal Mode Flag (D)
The D flag determines whether additions and subtractions are executed in binary or decimal. Binary arithmetic is executed when this flag is Low; decimal arithmetic is executed when it is High. Decimal correction is automatic in decimal mode. Only the ADC and SBC instructions can be used for decimal arithmetic. The D flag can be set by the set decimal mode (SED) instruction and cleared by the clear decimal mode (CLD) instruction.

Break Flag (B)
The B flag is used to indicate whether the current interrupt was generated by the BRK instruction. The BRK flag in the processor status register is nominally Low. When the BRK instruction is used to generate an interrupt, the processor status register is pushed onto the stack with the break flag set High. The saved processor status is the only place where the break flag is ever set.

Index X Mode Flag (T)
When the T flag is "0", arithmetic operations are performed between accumulator and memory, and the results are stored in the accumulator. When the T flag is High, direct arithmetic operations and direct data transfers are enabled between memory and memory, as well as between I/O and I/O. The result of an arithmetic operation performed on data in memory location 1 and memory location 2 is stored in memory location 1.

The address of memory location 1 is specified by index register X, and the address of memory location 2 is specified by normal addressing modes. The T flag can be set by the set T flag (SET) instruction and cleared by the clear T flag (CLT) instruction. Because the T flag directly affects calculations, it should be initialized after a reset.

Overflow Flag (V)
The V flag is used during the addition or subtraction of one byte of signed data. It is set if the result exceeds the range from +127 to -128. When the BIT instruction is executed, bit 6 of the memory location operated on by the BIT instruction is stored in the overflow flag. The V flag can be cleared by the CLV instruction, but there is no set instruction. In decimal mode, the V flag is invalid.

Negative Flag (N)
The N flag is set if the result of an arithmetic operation or data transfer is negative, that is (bit 7 is High). When the BIT instruction is executed, bit 7 of the memory location operated by the BIT instruction is stored in the negative flag. There are no instructions for directly setting or clearing the N flag.

2.2 CPU Mode Registers

Address	Description	Code
0000_{16}	CPU mode register A	CPUMA=0C
0001_{16}	CPU mode register B	CPUMB=83

This device has two CPU mode registers: CPU Mode Register A (CPUMA) and CPU Mode Register B (CPUMB) that control the processor mode, clock, slow memory wait and other CPU functions. The bit representation of each register is described in Figure 2-3 and Figure 2-4:

MSB 7								LSB 0
CPMA7	CPMA6	CPMA5	CPMA4	CPMA3	CPMA2	CPMA1	CPMA0	

Address: 0000_{16}
Access: R/W
Reset: $0C_{16}$

CPMA0,1 Processor Mode Bits (bits 0,1)
Bit 1 Bit 0
0 0: Single-Chip Mode
0 1: Memory Expansion Mode
1 0: Microprocessor Mode
1 1: Not used

CPMA2 Stack Page Selection Bit (bit 2)
0:In page 0 area
1:In page 1 area

CPMA3 X_{cout} Drive Capacity Selection Bit (bit 3)
0:Low
1:High

CPMA4 Clock XC_{in}-XC_{out} Stop Bit (bit 4)
0:Stop
1:Oscillator

CPMA5 Clock X_{in}-X_{out} Stop Bit (bit 5)
0:Oscillator
1:Stop

CPMA6 Internal Clock Selection Bit (bit 6)
0:External Clock
1:f_{syn}

CPMA7 External Clock Selection Bit (bit 7)
0:X_{in}-X_{out}
1:XC_{in}-XC_{out}

Figure 2-3. CPU Mode Register A

MSB 7								LSB 0
CPMB7	Reserved	CPMB5	CPMB4	CPMB3	CPMB2	CPMB1	CPMB0	

Address: 0001_{16}
Access: R/W
Reset: 83_{16}

CPMB0,1 Slow Memory Wait Bits (bit 1,0)
Bit 1 Bit 0
0 0: No wait
0 1: One time wait
1 0: Two time wait
1 1: Three time wait

CPMB2,3 Stack Page Selection Bit (bit 2)
Bit 3 Bit 2
0 0: Software wait
0 1: Not used
1 0: Fixed wait by RDY pin L
1 1: Extended RDY wait

CPMB4 Expanded Data Memory Access Bit (bit 4)
0:$\overline{EDMA}$ output disabled (64 Kbyte data access area)
1:$\overline{EDMA}$ output enabled (greater than 64 Kbytes data access area)

CPMB5 HOLD Function Enable Bit (bit 5)
0:HOLD Function Disabled
1:HOLD Function Enabled

CPMB6 Reserved

CPMB7 X_{out} Drive Capacity Selection Bit (bit 7)
0:Low
1:High (default state after reset and after STOP mode)

Figure 2-4. CPU Mode Register B

2.3 Memory Map

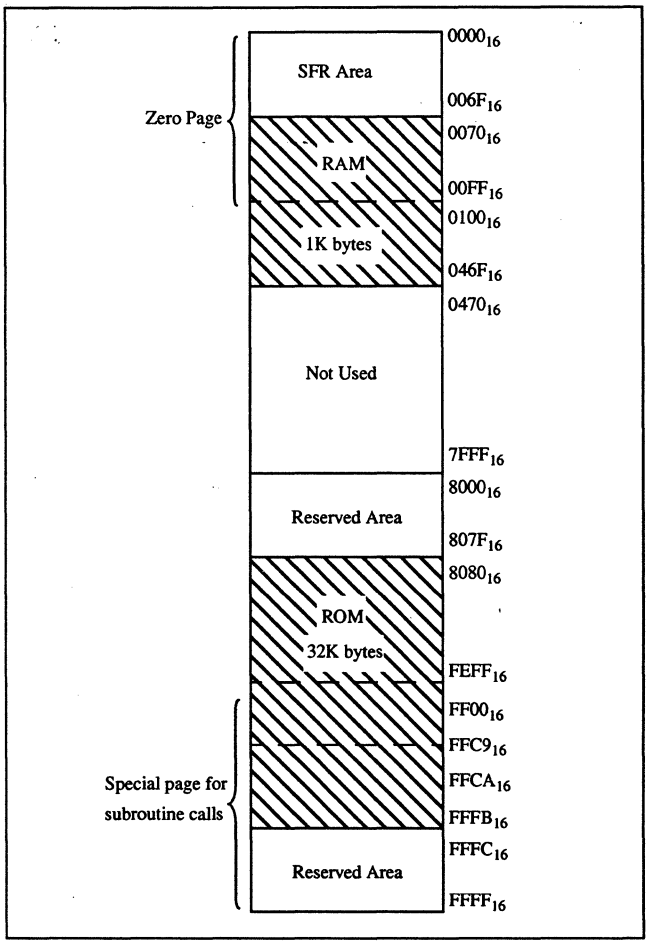


Figure 2-5. Memory Map

The first 112 bytes of memory from 0000_{16} to $006F_{16}$ are the special function register (SFR) area and contain the CPU mode registers, interrupt registers, and other registers to control peripheral functions (see Figure 2-7.).

The general purpose RAM resides from 0070_{16} to $046F_{16}$. When the MCU is in memory expansion or microprocessor mode and external memory is overlaid on the internal RAM, the CPU reads data from the internal RAM. However, the CPU writes data in both the internal and external memory. The area from 0470_{16} to $7FFF_{16}$ is not used in single-chip mode, but can be mapped for an external memory device when the MCU is in memory expansion or microprocessor mode.

The area from 8000_{16} to $807F_{16}$ and from $FFFC_{16}$ to $FFFF_{16}$ are factory reserved areas. Mitsubishi uses it for test and evaluation purposes. The user can not use this area in single-chip or memory expansion modes.

The user 32K byte ROM resides from 8080_{16} to $FFFB_{16}$. When the MCU is in microprocessor mode, the CPU accesses an external area rather than accessing the internal ROM.

Zero page and special page area can be accessed by 2-byte commands by using special addressing modes.

2.3.1 Zero page

The 256 bytes zero page area is where the SFR and part of the internal RAM are allocated. The zero page addressing modes can be used to specify memory and register addresses in this area (see Figure 2-6.). These dedicated addressing modes enable access to this area with fewer instruction cycles.

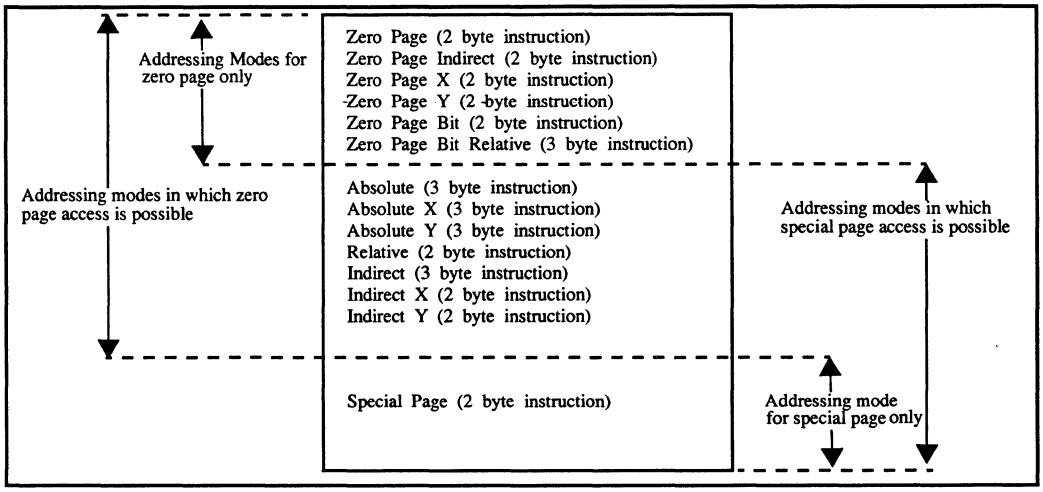


Figure 2-6. Zero Page and Special Page Addressing Modes

2.3.2 Special Page

The 256 bytes from address $FF00_{16}$ to $FFFF_{16}$ are called the special page area. In this area special page addressing can be used to specify memory addresses (see Figure 2-6.). This dedicated special page addressing mode enables access to this area with fewer instruction cycles. Frequently used subroutines are normally stored in this area.

2.3.3 Special Function Registers

The special function registers (SFR) are used for controlling the functional blocks, such as I/O ports, Timers, UART, and so forth (see Table 2-3.). The reserved addresses should not be read or written to.

Table 2-3. SFR Addresses

Addr	Description	Acronym and Value at Reset
0000_{16}	CPU Mode Register A	CPUMA=0C
0001_{16}	CPU Mode Register B	CPUMB=03
0002_{16}	Interrupt Request Register A	IREQA=00
0003_{16}	Interrupt Request Register B	IREQB=00
0004_{16}	Interrupt Request Register C	IREQC=00
0005_{16}	Interrupt Control Register A	ICONA=00
0006_{16}	Interrupt Control Register B	ICONB=00
0007_{16}	Interrupt Control Register C	ICONC=00
0008_{16}	Port P0	P0=00
0009_{16}	Port P0 Direction Register	P0D=00
$000A_{16}$	Port P1	P1=00
$000B_{16}$	Port P1 Direction Register	P1D=00
$000C_{16}$	Port P2	P2=00
$000D_{16}$	Port P2 Direction Register	P2D=00
$000E_{16}$	Port P3	P3=00
$000F_{16}$	Port P3 Direction Register	P3D=00
0010_{16}	Port Control Register	PTC=00
0011_{16}	Interrupt Polarity Selection Register	IPOL=00
0012_{16}	Port P2 pull-up Control Register	PUP2=00
0013_{16}	USB Control Register	USBC=00
0014_{16}	Port P6	P4=00
0015_{16}	Port P6 Direction Register	P4D=00
0016_{16}	Port P5	P5=00
0017_{16}	Port P5 Direction Register	P5D=00
0018_{16}	Port P4	P6=00
0019_{16}	Port P4 Direction Register	P6D=00
$001A_{16}$	Port P7	P7=00
$001B_{16}$	Port P7 Direction Register	P7D=00
$001C_{16}$	Port P8	P8=00
$001D_{16}$	Port P8 Direction Register	P8D=00
$001E_{16}$	Reserved	
$001F_{16}$	Clock Control Register	CCR=00
0020_{16}	Timer XL	TXL=FF
0021_{16}	Timer XH	TXH=FF
0022_{16}	Timer YL	TYL=FF
0023_{16}	Timer YH	TYH=FF
0024_{16}	Timer 1	T1=FF
0025_{16}	Timer 2	T2=01
0026_{16}	Timer 3	T3=FF
0027_{16}	Timer X Mode Register	TXM=00
0028_{16}	Timer Y Mode Register	TYM=00
0029_{16}	Timer 123 Mode Register	T123M=00
$002A_{16}$	SIO Shift Register	SIOSHT=XX
$002B_{16}$	SIO Control Register 1	SIOCON1=00
$002C_{16}$	SIO Control Register 2	SIOCON2=00
$002D_{16}$	Special Count Source Generator1	SCSG1=FF
$002E_{16}$	Special Count Source Generator2	SCSG2=FF
$002F_{16}$	Special Count Source Mode Register	SCSM=00
0030_{16}	UART1 Mode Register	U1MOD=00
0031_{16}	UART1 Baud Rate Generator	U1BRG=XX
0032_{16}	UART1 Status Register	U1STS=03
0033_{16}	UART1 Control Register	U1CON=00
0034_{16}	UART1 Transmit/Receiver Buffer 1	U1TRB1=XX
0035_{16}	UART1 Transmit/Receiver Buffer 2	U1TRB2=XX
0036_{16}	UART1 RTS Control Register	U1RTSC=00
0037_{16}	Reserved	

Addr	Description	Acronym and Value at Reset
0038_{16}	UART2 Mode Register	U2MOD=00
0039_{16}	UART2 Baud Rate Generator	U2BRG=XX
$003A_{16}$	UART2 Status Register	U2STS=03
$003B_{16}$	UART2 Control Register	U2CON=00
$003C_{16}$	UART2 Transmit/Receiver Buffer 1	U2TRB1=XX
$003D_{16}$	UART2 Transmit/Receiver Buffer 2	U2TRB2=XX
$003E_{16}$	UART2 RTS Control Register	U2RTSC=00
$003F_{16}$	DMAC Index and Status Register	DMAIS=00
0040_{16}	DMAC Channel x Mode Register 1	DMAxM1=00
0041_{16}	DMAC Channel x Mode Register 2	DMAxM2=00
0042_{16}	DMAC Channel x Source Register Low	DMAxSL=00
0043_{16}	DMAC Channel x Source Register High	DMAxSH=00
0044_{16}	DMAC Channel x Destination Register Low	DMAxDL=00
0045_{16}	DMAC Channel x Destination Register High	DMAxDH=00
0046_{16}	DMAC Channel x Count Register Low	DMAxCL=00
0047_{16}	DMAC Channel x Count Register High	DMAxCH=00
0048_{16}	Data Bus Buffer register 0	DBB0=00
0049_{16}	Data Bus Buffer status register 0	DBBS0=00
$004A_{16}$	Data Bus Buffer Control Register 0	DBBC0=00
$004B_{16}$	Reserved	
$004C_{16}$	Data Bus Buffer register 1	DBB1=00
$004D_{16}$	Data Bus Buffer Status Register 1	DBBS1=00
$004E_{16}$	Data Bus Buffer Control Register 1	DBBC1=00
$004F_{16}$	Reserved	
0050_{16}	USB Address Register	USBA=00
0051_{16}	USB Power Management Register	USBPM=00
0052_{16}	USB Interrupt Status Register 1	USBIS1=00
0053_{16}	USB Interrupt Status Register 2	USBIS2=00
0054_{16}	USB Interrupt Enable Register 1	USBIE1=00
0055_{16}	USB Interrupt Enable Register 2	USBIE2=00
0056_{16}	USB Frame Number Register Low	USBSOFL=00
0057_{16}	USB Frame Number Register High	USBSOFH=00
0058_{16}	USB Endpoint Index	USBINDEX=00
0059_{16}	USB Endpoint x IN CSR	IN_CSR=00
$005A_{16}$	USB Endpoint x OUT CSR	OUT_CSR=00
$005B_{16}$	USB Endpoint x IN MAXP	IN_MAXP=00
$005C_{16}$	USB Endpoint x OUT MAXP	OUT_MAXP=00
$005D_{16}$	USB Endpoint x OUT WRT_CNT Low	WRT_CNTL=00
$005E_{16}$	USB Endpoint x OUT WRT_CNT High	WRT_CNTH=00
$005F_{16}$	Reserved	
0060_{16}	USB Endpoint 0 FIFO	USBFIFO0=00
0061_{16}	USB Endpoint 1 FIFO	USBFIFO1=00
0062_{16}	USB Endpoint 2 FIFO	USBFIFO2=00
0063_{16}	USB Endpoint 3 FIFO	USBFIFO3=00
0064_{16}	USB Endpoint 4 FIFO	USBFIFO4=00
0065_{16}	Reserved	
0066_{16}	Reserved	
0067_{16}	Reserved	
0068_{16}	Reserved	
0069_{16}	Reserved	
$006A_{16}$	Reserved	
$006B_{16}$	Reserved	
$006C_{16}$	Freq Synthesizer Control	FSC=00
$006D_{16}$	Freq Synthesizer Multiply Register 1	FSM1=TBD
$006E_{16}$	Freq Synthesizer Multiply Register 2	FSM2=TBD
$006F_{16}$	Freq Synthesizer Divide Register	FSD=00

2.4 Processor Modes

The operation modes are described below. The memory maps for the first three modes are shown in Figure 2-7.

Single chip mode is normally entered after reset. However, if the MCU has a CNV_{SS} pin, holding this pin High will cause microprocessor mode to be entered after reset. After the reset sequence has completed, the mode can be changed with software by modifying the value of bits 0 and 1 of CPUMA. However, while CNV_{SS} is High, bit 1 of CPUMA is High and cannot be changed.

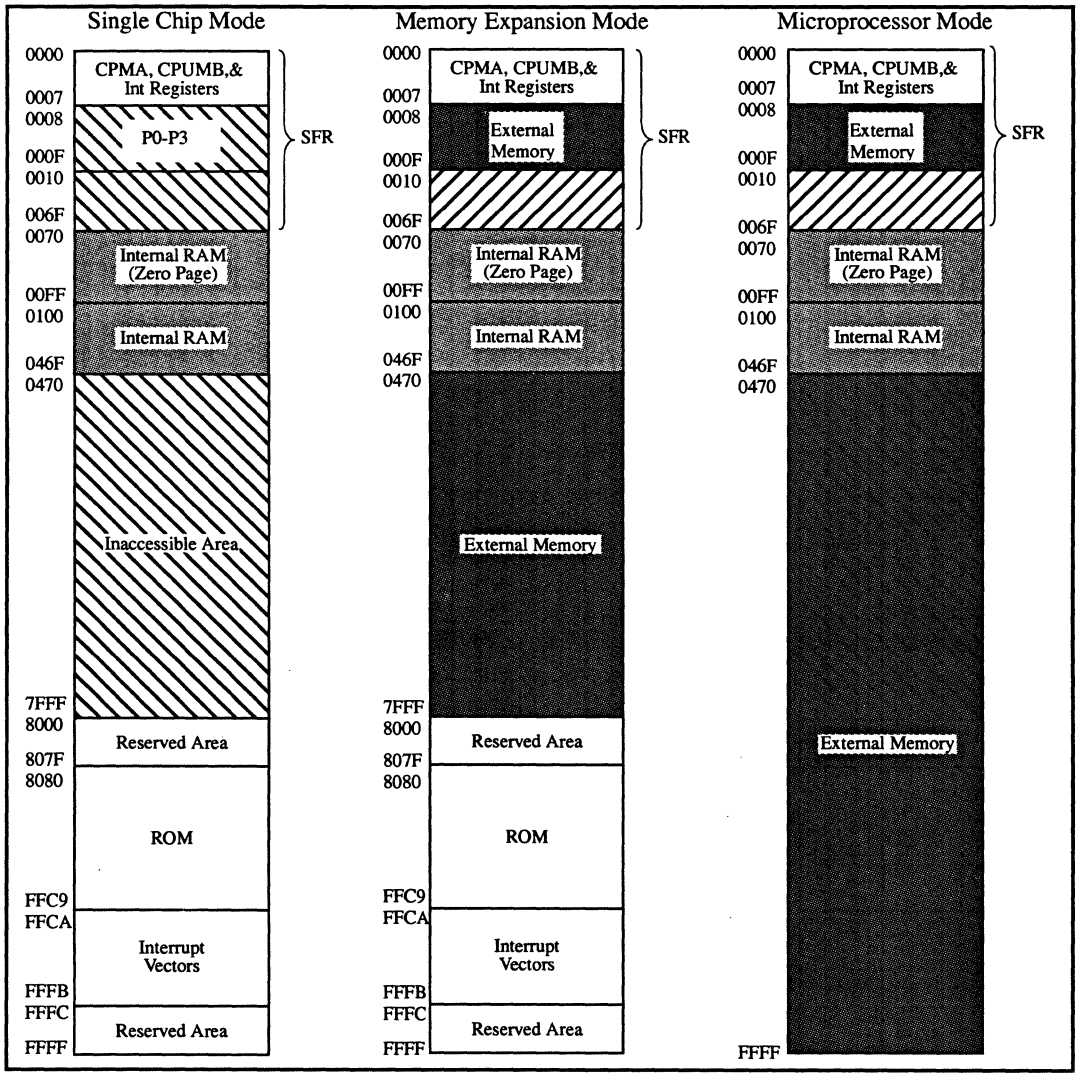


Figure 2-7. Operation Modes Memory Maps

2.4.1 Single Chip

In this mode, all ports take on their primary function and all internal memory is accessible. Those areas that are not in internal memory are not accessible. Also, slow memory wait and $\overline{EDMA}$ are disabled in this mode.

2.4.2 Memory Expansion

In this mode, Ports 0 and 1 output the address bus (AB_0-AB_{15}), port 2 acts as the data bus input and output, and port 3 bits 7 to 3 output $\overline{RD}$, $\overline{WR}$, SYNCout, Φout, and DMAout, respectively. All memory areas that are not internal memory or SFR area are accessed externally. Because ports 0 to 3 lose their normal function in this mode, the address area for the ports and their direction registers are treated as external memory (see Figure 2-8.) In this mode, slow memory wait and $\overline{EDMA}$ can be enabled.

2.4.3 Microprocessor

This mode is primarily the same as memory expansion mode. The difference is that the internal ROM / EPROM area can not be accessed and is instead treated as external memory. Slow memory wait and $\overline{EDMA}$ can be enabled in this mode.

2.4.4 EPROM

This mode is used for programming and testing the internal EPROM. In this mode, ports 0 and 1 input the address, port 2 acts as the data bus input and output, and port 3 bits 7, 6, 3 and 2 input OEB, CEB, PGMB, and VRFY, respectively.

2.4.5 Slow Memory Wait

The wait function is used when interfacing with external memories that are too slow to operate at the normal read/write speed of the MCU. When this is the case, a wait can be used to extend the read/write cycle. Three different wait modes are supported; software wait, RDY wait, and extended RDY wait. The appropriate mode is chosen by the setting of bits 0 to 3 of CPMB. The wait function is disabled for internal memory and is valid only for memory expansion and microprocessor modes.

Software wait is used to extend the read/write cycle by one, two, or three cycles. The cycle number is determined by the value of bits 0 and 1 of the CPMB. When software wait is selected, the value on the RDY pin is ignored. The timing for software wait is shown in Figure 2-9.

RDY wait is also used to extend the read/write cycle by one, two, or three cycles. In this case, the read/write cycle is extended if the RDY pin is Low a specific amount of time prior to Φ_{out} going Low at the beginning of the read/write cycle. The extension time is fixed by the value of bits 0 and 1 of CPMB and does not depend on the state of the RDY pin when the read/write cycle has begun. If the RDY pin is High at the specified time prior to Φout going Low at the beginning of the read/write cycle, the cycle is not extended. The timing for RDY wait is shown in Figure 2-10..

The extended RDY wait mode is used to extend the read/write cycle by one, two, or three cycles, and then by an additional amount, dependent on the state of the RDY pin. In this mode, initial extension is identical to that of the RDY wait. The state of the RDY pin is checked a specific amount of time prior to the completion of the first cycle of the read/write extension. If the RDY pin is Low, the extension is re-initiated from the time that Φout goes Low at the end of the first cycle of the read/write extension. The RDY pin continues to be checked until it is brought High. When the RDY pin is brought High, the wait is no longer re-initiated when Φout goes Low, and the read/write cycle completes in one, two, or three cycles, dependent on the value in bits 0 and 1 of CPMB. The timing for this mode is shown in Figure 2-11.

The wait function can only be enabled for external memory access in microprocessor or memory expansion modes. However, the wait function can not be enabled for accesses to addresses 0008_{16} to $000F_{16}$ (port 0 through port 3 direction registers) in these modes, even though the locations are mapped as external memory.

$CPUM_A$	0	1	0
$CPUM_B$	0	0	1
Port \ Mode	Single Chip Mode	Microprocessor Mode	Memory Expansion Mode
Port P0	Internal Φ Port $P0_7$ - $P0_0$ I/O Port	Internal Φ Port $P0_7$ - $P0_0$ Address Output AB_7 - AB_0	Same as Microprocessor Mode
Port P1	Internal Φ Port $P1_7$ - $P1_0$ I/O Port	Internal Φ Port $P1_7$ - $P1_0$ Address Output AB_{15} - AB_8	Same as Microprocessor Mode
Port P2	Internal Φ Port $P2_7$ - $P1_0$ I/O Port	Internal Φ Port $P2_7$ - $P2_0$ Data I/O DB_7 - DB_0	Same as Microprocessor Mode
Port P3	Internal Φ Port $P3_7$ - $P3_0$ I/O Port	Internal Φ Port 3_4 Port 3_7 RD Output Port 3_6 WR Output Port 3_5 SYNCout DMAout Output Port 3_3	Same as Microprocessor Mode

Figure 2-8. Function of Ports P_0-P_3 in each Processor Mode

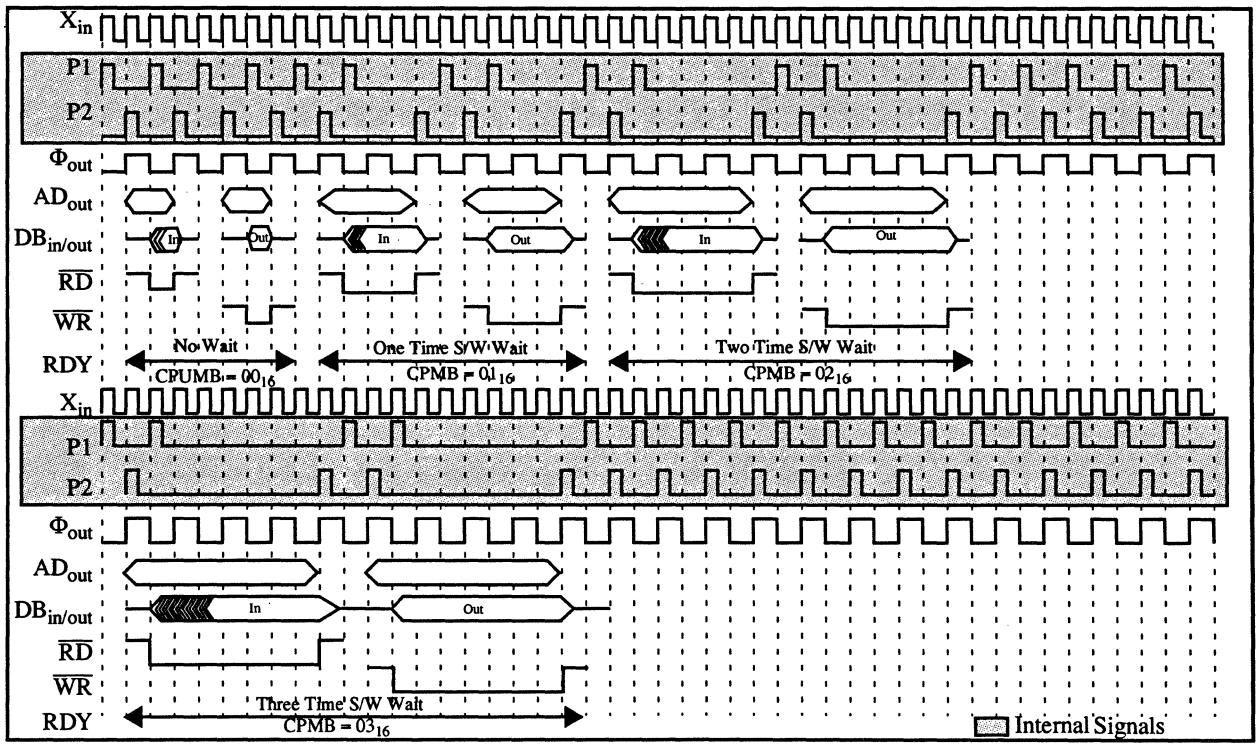


Figure 2-9. Software Wait Timing Diagram

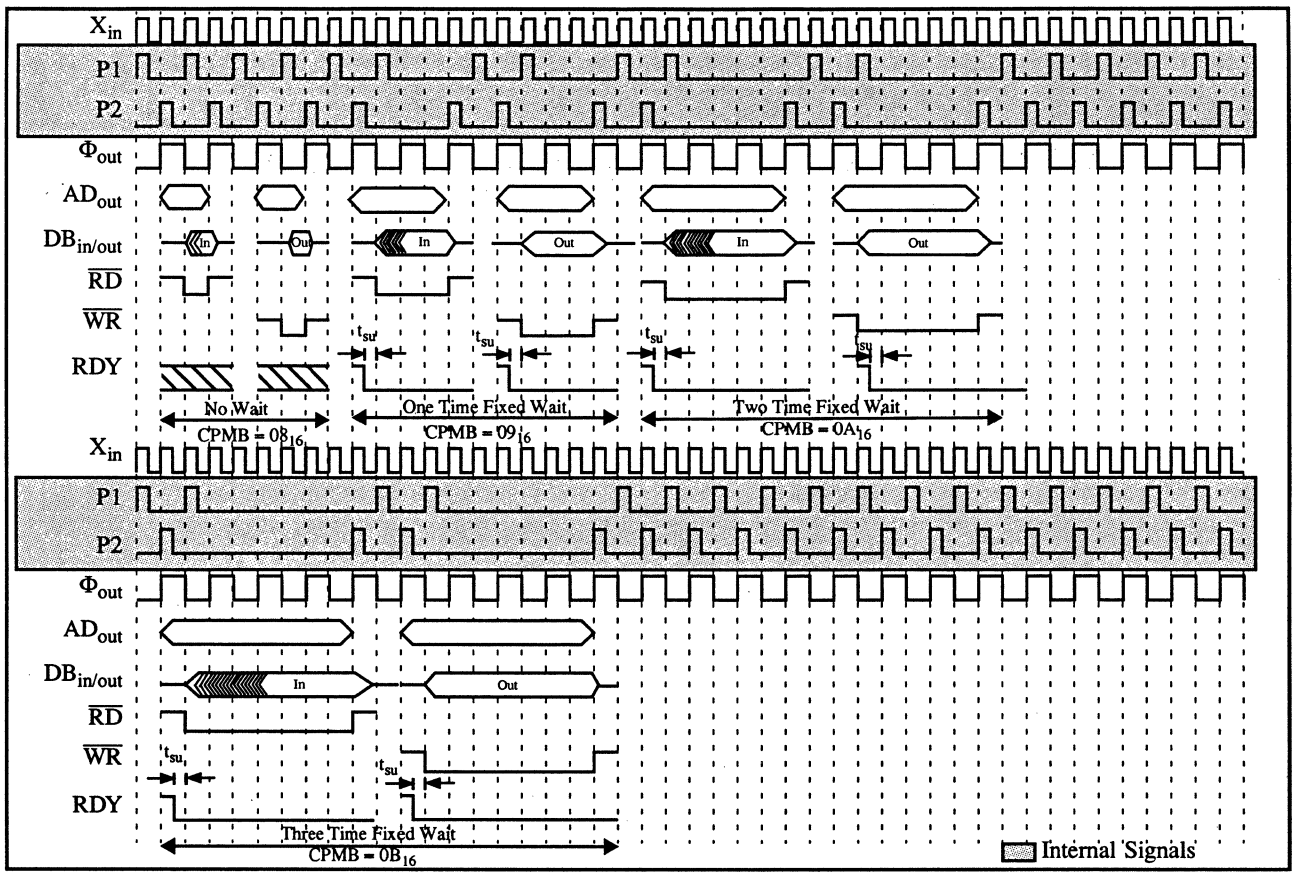


Figure 2-10. RDY Wait Timing Diagram

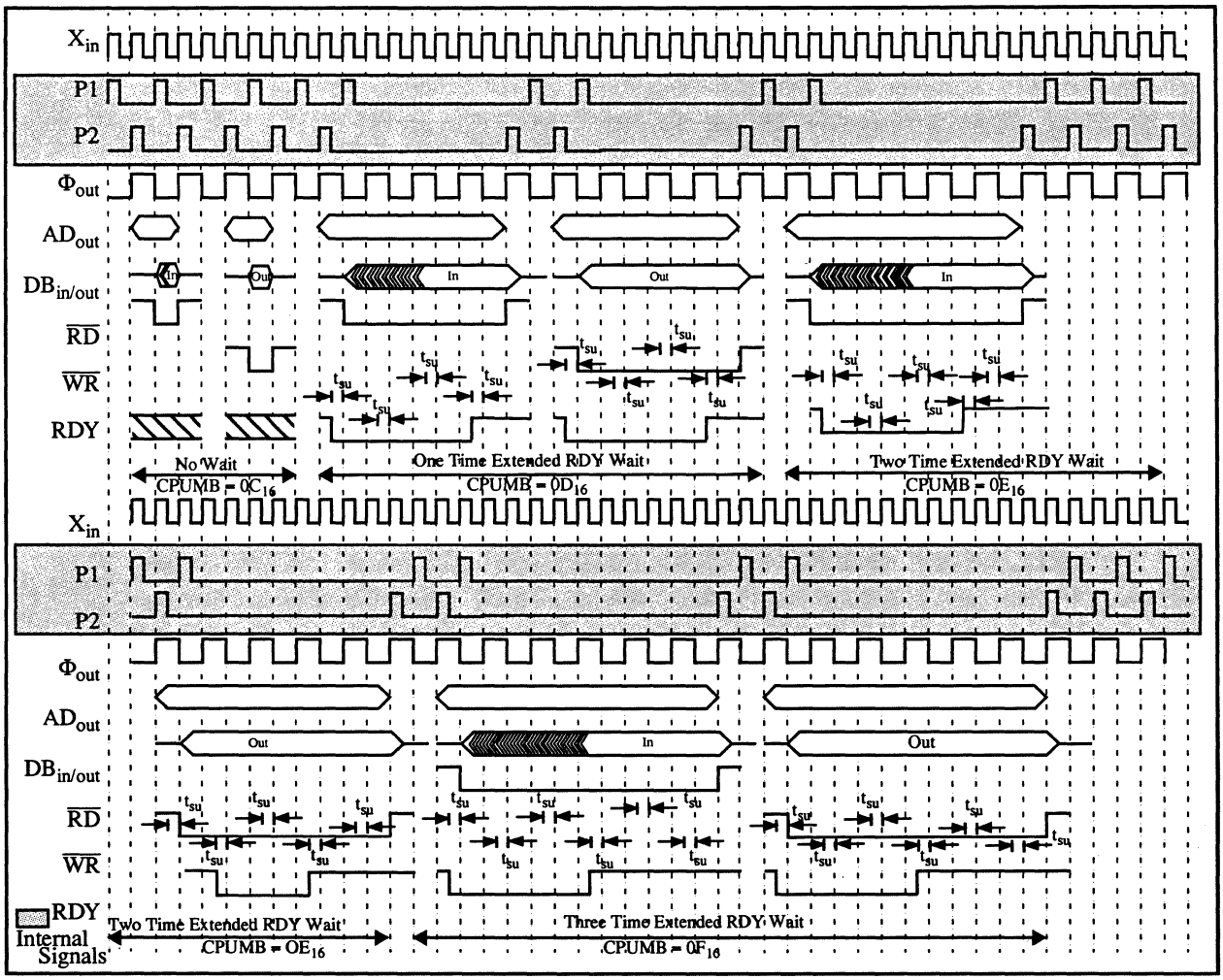


Figure 2-11. Extended RDY Wait Timing Diagram

2.4.6 Hold Function

The hold function is used when the MCU is put in a system where more than one device will need control of the external address and data buses. Two signals are used to implement this function, $\overline{HOLD}$ and $\overline{HLDA}$. $\overline{HOLD}$ is an input to the MCU and is brought Low when an external device wants the MCU to relinquish the address and data buses. $\overline{HLDA}$ is an output from the MCU that signals when the MCU has relinquished the buses. When this is the case, the MCU tri-state ports 0 and 1 (address bus) and port 2 (data bus), and holds port $P3_7$ ($\overline{RD}$) and port $P3_6$ ($\overline{WR}$) High. Ports $P3_7$ and $P3_6$ are held High to prevent any external device that is enabled by $\overline{RD}$ or $\overline{WR}$ from being falsely activated. The clocks to the CPU are stopped, but the peripheral clocks and port P34 (Φ_{out}) continue to oscillate. $\overline{HOLD}$ is brought High to allow the MCU to regain the address and data buses. When this occurs, $\overline{HLDA}$ will go High and ports P_1, P_2, $P3_7$ and $P3_6$ will begin to drive the external buses again. The timing for the hold function is shown in Figure 2-12. The hold function is only valid for memory expansion and microprocessor modes. Bit 5 of CPUMB is used to enable the hold function.

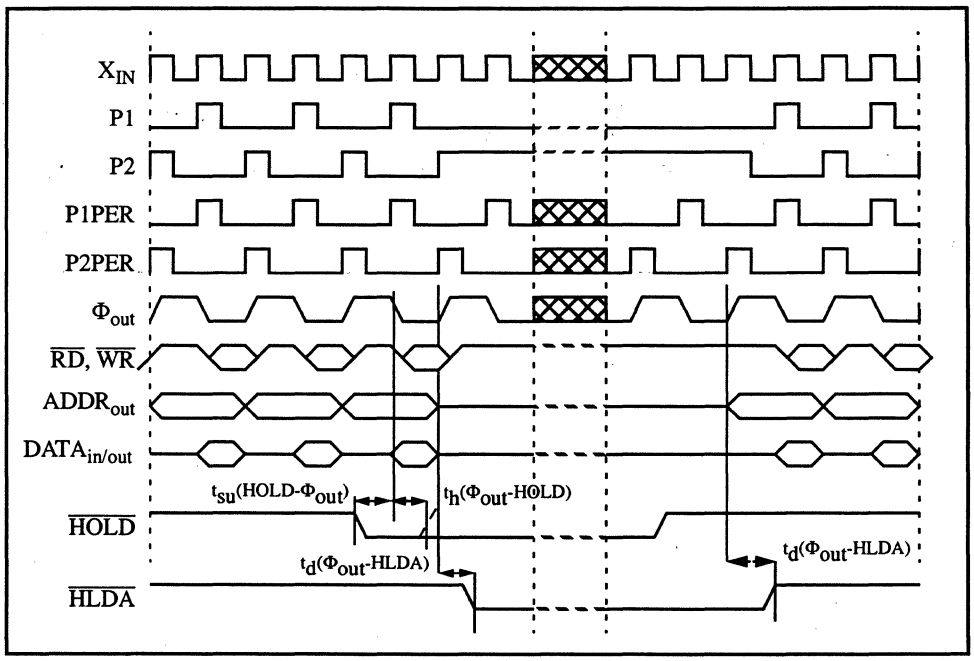


Figure 2-12. Hold Mode Timing Diagram

2.4.7 Expanded Data Memory Access

The Expanded Data Memory Access ($\overline{EDMA}$) mode feature allows the user to access a greater than 64 Kbyte data area for instructions LDA (IndY) with T = zero and T = one, and STA (IndY). Bit 4 of CPMB is used to enable/disable the $\overline{EDMA}$ function. If bit 4 of CPMB equals one, then during the data read/write cycle of instructions LDA (IndY) and STA (IndY) Port 40 ($\overline{EDMA}$) is driven Low. The $\overline{EDMA}$ signal output can be used by an external decoder to indicate when the read/write is to a different 64 Kbyte bank. The actual determination of which bank to access can be done by using a few bits of a port to represent the extended addresses above AB15. For example, if four banks are accessed, then two bits are needed to uniquely identify each bank. Two port bits can be used for this, one representing AB16 and the other AB17. The instruction sequences for STA (IndY) and LDA (IndY) are shown in Figure 2-13. and Figure 2-14.

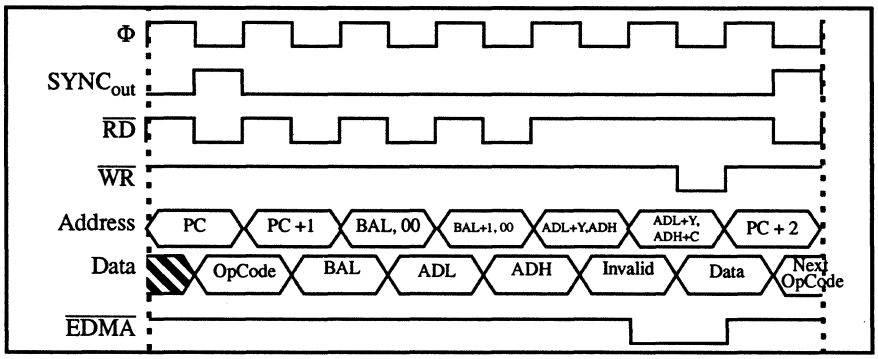


Figure 2-13. STA Y Instruction Sequence with $\overline{EDMA}$ Enabled

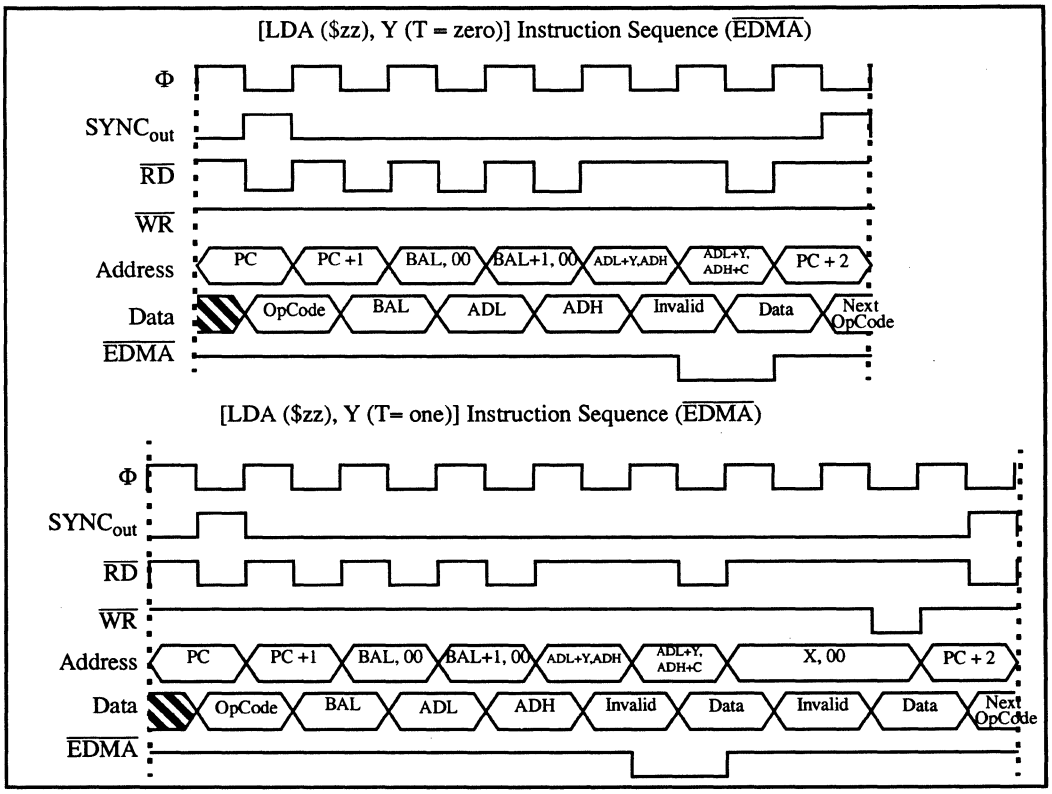


Figure 2-14. LDA Y Instruction Sequences with EDMA Enabled

2.5 *Peripheral Interface*

2.5.1 Chip Bus Timing

The internal bus timing is described below for the CPU (or DMAC) writing to and reading from a peripheral (see Figure 2-15.)

- The address (AB[15:0]) is output from the CPU on P2.
- The data bus (DB[7:0]) is driven by the CPU during a write, or by a peripheral during a read, on P1.
- The R/$\overline{W}$ signal is High for a read and Low for a write, and changes on P2.
- The EB signal is High when a read or write is not valid, and is Low for a valid read or write. It changes on P2.
- A PD*n*B signal (peripheral decode) is assigned to each peripheral and is Low when reading from or writing to the peripheral. Each PD*n*B signal is clocked on P2 timing.

The address, R/$\overline{W}$, EB, and PD*n*B signals are latched at the peripheral block on P1, so they must all be valid before this time. The data bus is latched by the CPU during a read, or by a peripheral during a write, on P2; so the value on the data bus must be valid before this time.

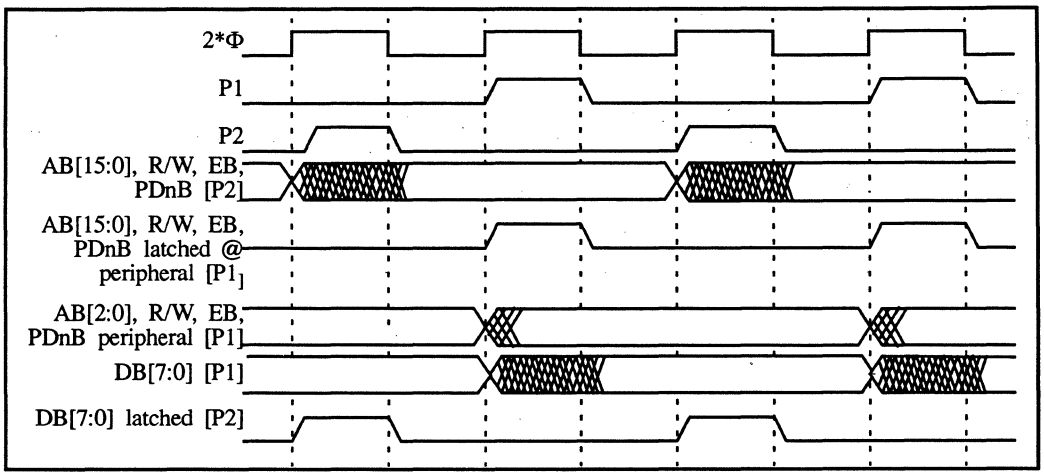


Figure 2-15. 7600 Series Internal Bus Timing

2.5.2 Peripheral Interface and Access Timing

The 7600 series offers a wide variety of peripherals. These include RAM, ROM, EPROM, UARTs, SIOs, 8-bit and 16-bit timers, A/D converters, D/A converters, various I/O ports, OpAmps, and clock generators, to name a few.

The interface between the CPU, the peripheral decode block, and peripheral blocks is shown in Figure 2-16. Signals DB7 to DB0, AB2 to AB0, R/W, EB, and at least one peripheral decode (PD*n*B) are routed to each peripheral. The address signals and peripheral decode signal are used in the peripheral block to create decode signals for each register. Because three address bits are available at the peripheral, a maximum of eight decode signals can be created for each peripheral decode signal. If the peripheral contains more than eight registers, additional peripheral decode signals are routed to the peripheral.

The bus timing for reading from and writing to a peripheral is shown in Figure 2-17.

- When P2 goes High, the address, R/W, and EB are output from the CPU. All address signals are routed to the peripheral decode block where a peripheral decode signal is generated asynchronously. Also, data read from a peripheral in the previous half cycle is latched in the CPU, and data written to a peripheral in the previous half cycle is latched in the desired register of the peripheral at this time.
- When P1 goes High, address AB[2:0], R/W, EB, and PDnB are latched at the peripherals. From these signals, the determination of which peripheral and register inside of that peripheral is to be written or read is made. Also, if the CPU is writing to a peripheral, it begins to drive the data bus at this time with the data to be written to the peripheral. If the CPU is reading from a peripheral, the peripheral begins to drive the data bus as soon as the decode is finished and the data is available from the register.

This timing does not apply for the RAM and ROM/EPROM.

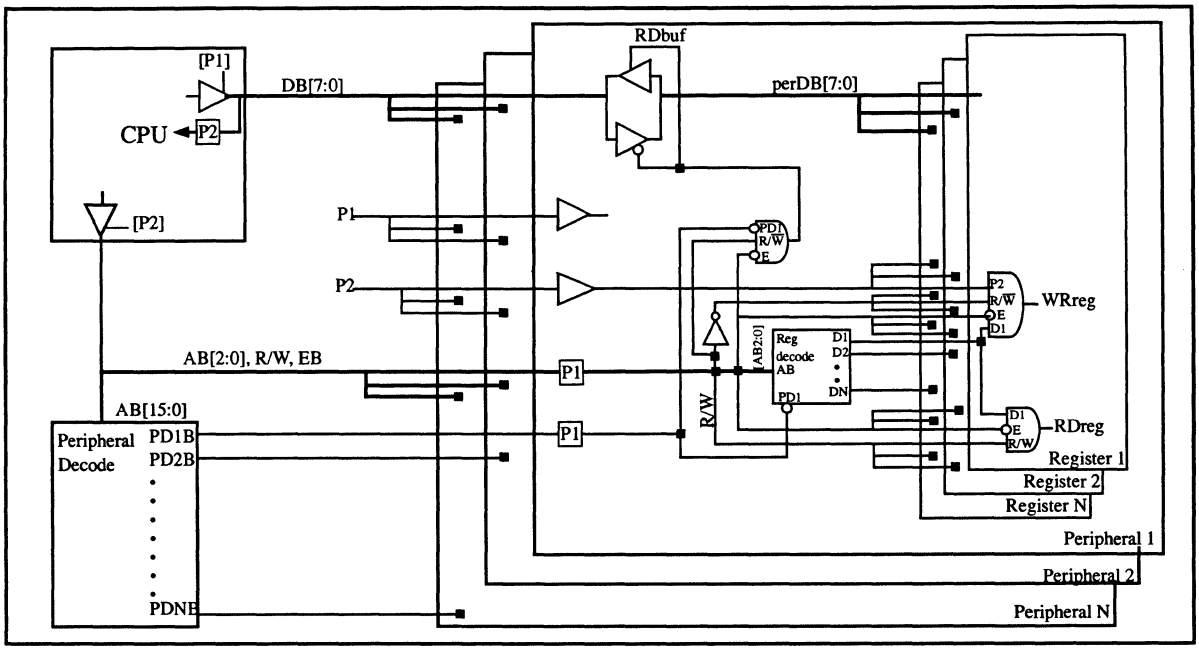


Figure 2-16. Internal Peripheral Interface

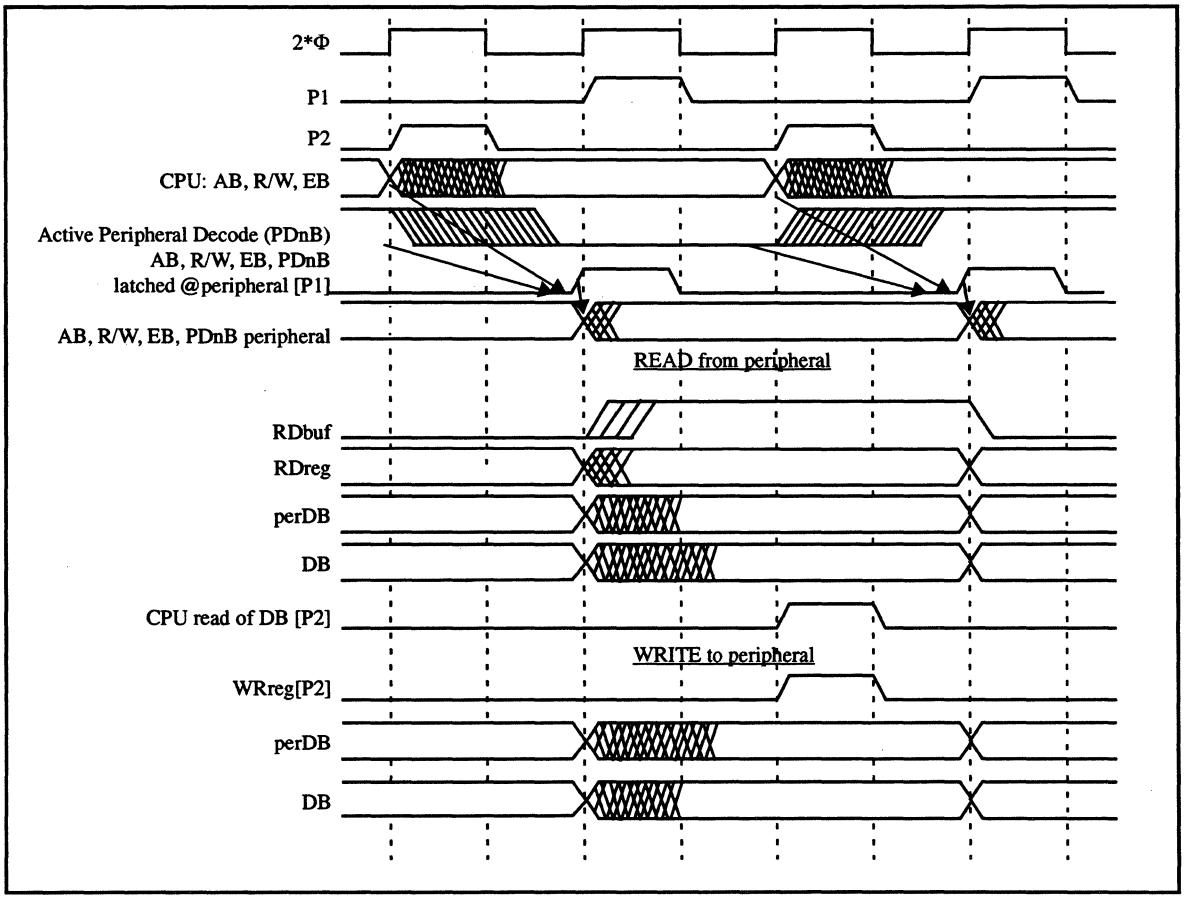


Figure 2-17. 7600 Series Peripheral Bus Timing

2.6 Input and Output Ports

Address	Description	Acronym and Value at Reset
0008_{16}	Port P0	P0=00
0009_{16}	Port P0 direction register	P0D=00
$000A_{16}$	Port P1	P1=00
$000B_{16}$	Port P1 direction register	P1D=00
$000C_{16}$	Port P2	P2=00
$000D_{16}$	Port P2 direction register	P2D=00
$000E_{16}$	Port P3	P3=00
$000F_{16}$	Port P3 direction register	P3D=00
0014_{16}	Port P4	P4=00
0015_{16}	Port P4 direction register	P4D=00

Address	Description	Acronym and Value at Reset
0012_{16}	Port P2 pull-up Control Register	PUP2=00
0016_{16}	Port P5	P5=00
0017_{16}	Port P5 direction register	P5D=00
0018_{16}	Port P6	P6=00
0019_{16}	Port P6 direction register	P6D=00
$001A_{16}$	Port P7	P7=00
$001B_{16}$	Port P7 direction register	P7D=00
$001C_{16}$	Port P8	P8=00
$001D_{16}$	Port P8 direction register	P8D=00
0010_{16}	Port control register	PTC=00

Bit 4_0	is multiplexed with EDMA
Bits 4_1-4_2	are multiplexed with external interrupts INT0, INT1
Bit 4_3	is multiplexed with Timer X CNTR0 pin
Bit 4_4	is multiplexed with Timer Y CNTR1 pin
Bit 5_0	is multiplexed with XC_{in}
Bit 5_1	is multiplexed with Timer 1 pulse output pin or XC_{out}
Bit 5_2	is multiplexed with OBF_0 output to master CPU
Bit 5_3	is multiplexed with $\overline{IBF}_0$ output to master CPU
Bit 5_4	is multiplexed with S_0 input to master CPU
Bit 5_5	is multiplexed with $\overline{A}_0$ input to master CPU
Bit 5_6	is multiplexed with $\overline{R}$ (E) input to master CPU
Bit 5_7	is multiplexed with $\overline{W}$ (R/$\overline{W}$) input to master CPU
Bits 6_0-6_7	are multiplexed with Master CPU Bus I/F DQ0-DQ7 pins
Bit 7_0	is multiplexed with $\overline{SOF}$
Bit 7_1	is multiplexed with $\overline{HOLD}$
Bit 7_2	is multiplexed with $\overline{S}_1$
Bit 7_3	is multiplexed with $\overline{IBF}_1$ or HLDA
Bit 7_4	is multiplexed with OBF_1
Bits 8_0-8_3	are multiplexed with the first alternate function UART2 pins or 2nd alternate function with SIO pins
Bits 8_4-8_7	are multiplexed with the UART1 pins

2.6.1 Ports

This device has 66 programmable I/O pins arranged as ports $P0_0$ to $P8_7$. Each port bit can be configured as input or output. To set the I/O port bit direction, write a "1" to the corresponding direction register bit to select output mode, or write a "0" to the direction register bit to select input mode.

At reset, all of the direction registers are initialized to 00_{16}, setting all of the I/O ports to input mode.

If data is written to a pin and then read from that pin while it is in output mode, the data read is the value of the port latch rather than the value of the pin itself. Therefore, if an external load changes the value of an output pin, the intended output value will still be read correctly. Pins set to input mode are floating (provided that the pull up resistors are not being used) to ensure that the value input to such a pin can be read accurately. In the case when data is written to a pin configured as an input, the data is written only to the port latch; the pin itself remains floating.

Most of the I/O Ports are multiplexed with secondary functions. When a GPIO is multiplexed with a second function, the control signal from the peripheral overrides the direction register. The multiplexing is briefly described below. The second function signals to and from the I/O ports are described in detail in their respective block's description.

2.6.1.1 I/O Ports

Ports 0 and 1

Ports 0 and 1 act as the address bus (AB_0-AB_{15}) in Microprocessor and Memory Expansion modes.

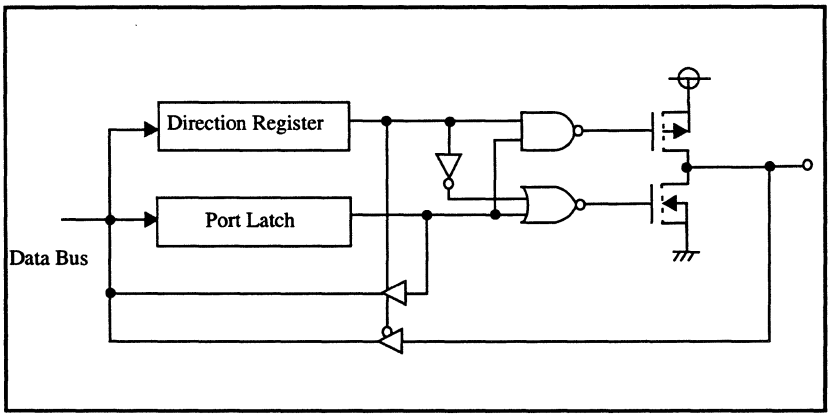


Figure 2-18. Port P0, P1, P3, and $P6_6$ Block Diagram

Port 2

Port 2 acts as the data bus during microprocessor and memory expansion modes and also contains key-on wake up circuitry (see Figure 2-19.).

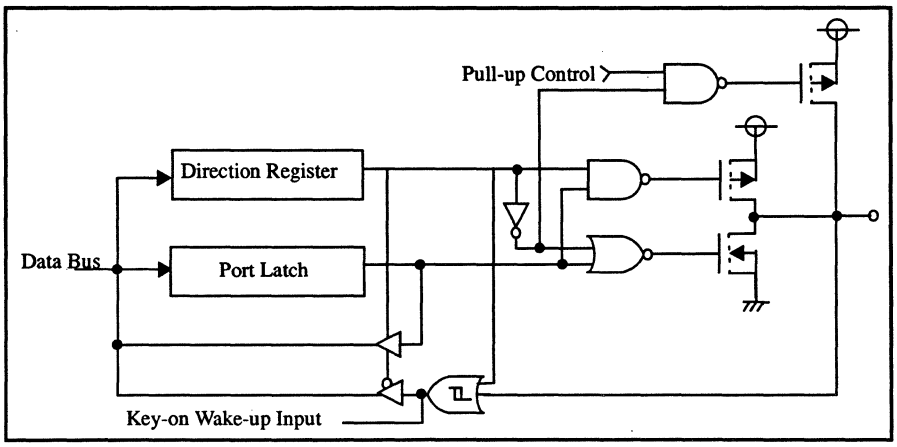


Figure 2-19. Port P2 Block Diagram

Port 3

Port 3 outputs the signals described in the pin description (Figure 1-2. Pin Description).

Port 4

See Figure 2-20., Figure 2-21., and Figure 2-22.

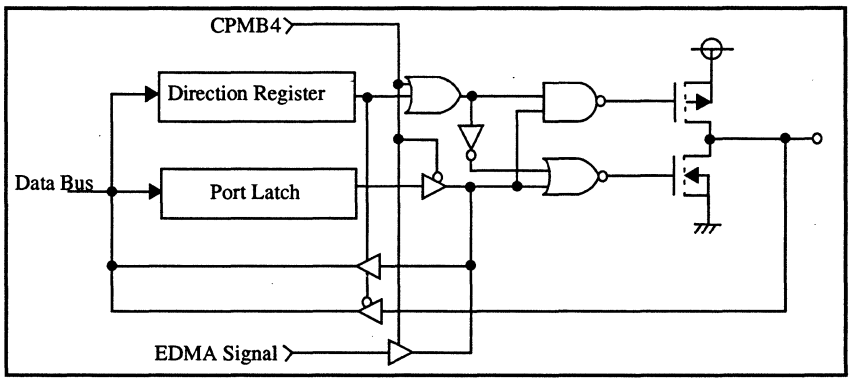


Figure 2-20. Port $P4_0$ Block Diagram

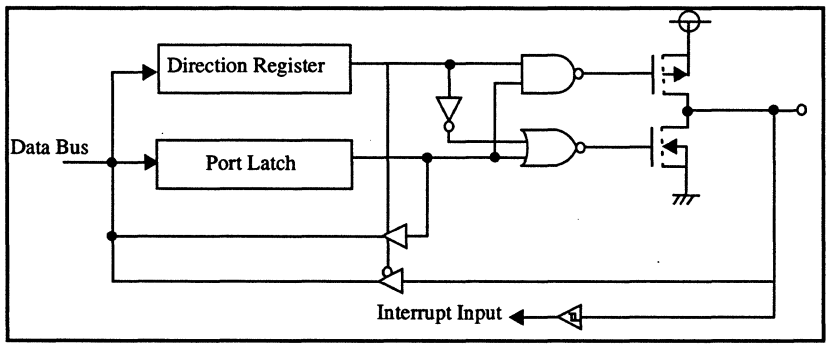


Figure 2-21. Port $P4_1$ and $P4_2$ Block Diagram

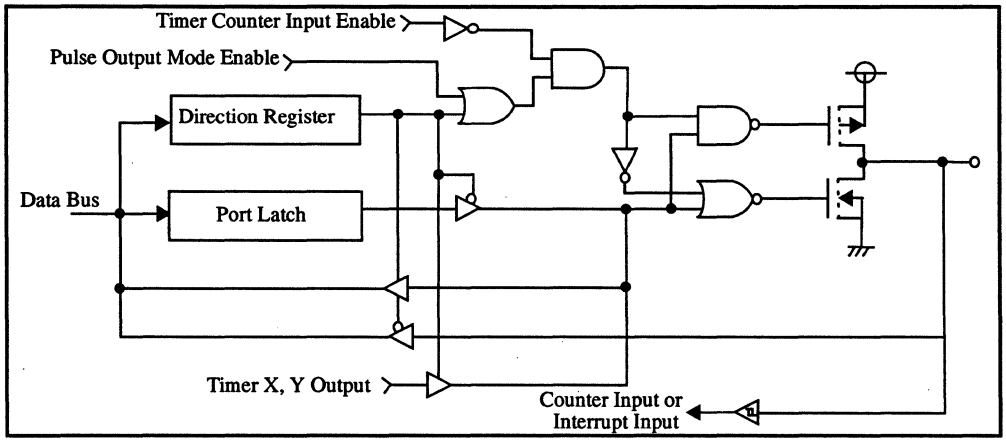


Figure 2-22. Port $P4_3$ and $P4_4$ Block Diagram

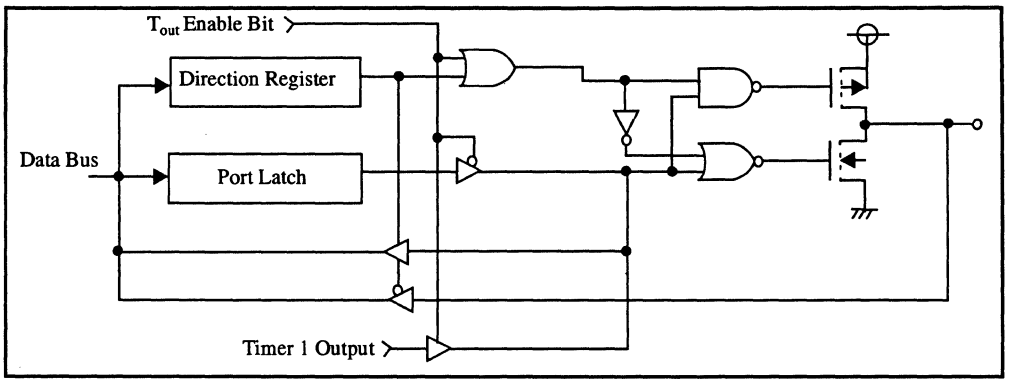


Figure 2-23. Port $P4_5$ Block Diagram

2.6.1.2 Power and Ground Pins

There are two V_{ss} and two V_{dd} pins that supply power to the MCU. There are also one analog V_{dd} (AV_{cc}) and one analog V_{ss} (AV_{ss}) pins for the analog circuits.

2.6.1.3 CNV_{ss} Pin

The level of the signal input to the CNV_{ss} pin at reset determines whether the chip enters single chip or microprocessor mode. With CNV_{ss} connected to V_{dd}, the MCU enters microprocessor mode after a reset. After the reset sequence has been completed, the mode can be changed by modifying the value of bits 0 and 1 of CPUMA. However, while CNV_{ss} is connected V_{dd}, bit 1 of CPUMA can not be overwritten. With CNV_{ss} connected to V_{ss}, the MCU enters single chip mode after a reset.

2.6.1.4 X_{in} and X_{out} Pins

The X_{in} and X_{out} pins are clock input and output pins. This device has a built-in clock generation circuit whose oscillation frequency is set by a ceramic or a quartz oscillator. Also, an external clock source can be used by connecting the X_{in} pin to a clock generator and leaving the X_{out} pin floating.

2.6.1.5 XC_{in} and XC_{out} Pins

The $P5_0/X_{cin}$ and $P5_1/T_{out}/XC_{out}$ pins are clock input and output pins. This device has a built-in clock generation circuit whose oscillation frequency is set by a ceramic or quartz oscillator. An external clock may also be used by connecting the XC_{in} pin to a clock generator and leaving the XC_{out} pin floating.

2.6.1.6 $\overline{\text{RESET}}$ Pin

The MCU is reset by holding $\overline{\text{RESET}}$ Low for at least 2μs before returning to High.

2.6.1.7 RDY Pin

For a detailed description of the $P3_0$/RDY pin see "2.4.5 Slow Memory Wait".

2.6.1.8 DMAout Pin

When the chip is in microprocessor or memory expansion mode, the DMAout ($P3_3$/DMAout) pin goes High during a DMA access.

2.6.1.9 Φ_{out} Pin

When the MCU is in microprocessor or memory expansion mode, pin $P3_4$ outputs the internal system clock Φ_{out}. When the STP or WIT instructions are executed, the output of the Φ_{out} pin stops at a High level.

2.6.1.10 $SYNC_{out}$ PinC

When the MCU is in microprocessor or memory expansion mode, the $SYNC_{out}$ pin outputs a signal that is High for one-half cycle of Φ_{out} every time an OpCode is fetched.

2.6.1.11 $\overline{RD}$ and $\overline{WR}$ Pins

A read control signal is output from the $\overline{RD}$ pin and write control signal is output from the $\overline{WR}$ pin ($P3_6/\overline{WR}$ and $P3_7/\overline{RD}$). A Low output from the $\overline{RD}$ pin indicates that the CPU is reading and a Low output from the $\overline{WR}$ pin indicates that the CPU is writing.

2.6.1.12 LPF Pin

When the Frequency Synthesizer is active, the X_{out}/LPF pin is the loop filter for the Frequency Synthesizer.

2.6.2 Port Control Register

This device is equipped with a port control register to control multiplexing of several pins and to turn on and off the slew rate control (SRC) (see Figure 2-24.).

MSB 7	PTC7	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0	LSB 0

Address: 0010_{16}
Access: R/W
Reset: 00_{16}

PTC0	Slew Rate Control Ports 0-3 0:Disabled 1:Enabled
PTC1	Slew Rate Control Port 4 0:Disabled 1:Enabled
PTC2	Slew Rate Control Port 5 0:Disabled 1:Enabled
PTC3	Slew Rate Control Port 6 0:Disabled 1:Enabled
PTC4	Slew Rate Control Port 7 0:Disabled 1:Enabled
PTC5	Slew Rate Control Port 8 0:Disabled 1:Enabled
PTC6	Port 2 Input Level Select 0:TTL level input 1:CMOS level input
PTC7	Master Bus Input Level Select 0:CMOS level input 1:TTL level input

Figure 2-24. Port Control Register

2.6.3 Port P2 Pull-up Control Register

This device is equipped with internal pull-ups on Port P2 that can be enabled by software. Each bit of that pull-up control register controls a corresponding pin of Port P2. The pull-up control register pulls up the port when the port is in input mode. The value of the pull-up control register has no effect when the port is in output mode.

MSB 7								LSB 0
$PUP2_7$	$PUP2_6$	$PUP2_5$	$PUP2_4$	$PUP2_3$	$PUP2_2$	$PUP2_1$	$PUP2_0$	

Address: 0012_{16}
Access: R/W
Reset: 00_{16}

$PUP2_0$ Pull-up Control for Port 2, Bit 0
0:Disabled
1:Enabled

$PUP2_1$ Pull-up Control for Port 2, Bit 1
0:Disabled
1:Enabled

$PUP2_2$ Pull-up Control for Port 2, Bit 2
0:Disabled
1:Enabled

$PUP2_3$ Pull-up Control for Port 2, Bit 3
0:Disabled
1:Enabled

$PUP2_4$ Pull-up Control for Port 2, Bit 4
0:Disabled
1:Enabled

$PUP2_5$ Pull-up Control for Port 2, Bit 5
0:Disabled
1:Enabled

PUP26 Pull-up Control for Port 2, Bit 6
0:Disabled
1:Enabled

PUP27 Pull-up Control for Port 2, Bit 7
0:Disabled
1:Enabled

Figure 2-25. Pull-up Control Register

2.7 *Interrupt Control Unit*

Address	Description	Code
0002_{16}	Interrupt request register A	IREQA=00
0003_{16}	Interrupt request register B	IREQB=00
0004_{16}	Interrupt request register C	IREQC=00
0005_{16}	Interrupt control register A	ICONA=00

Address	Description	Code
0006_{16}	Interrupt control register B	ICONB=00
0007_{16}	Interrupt control register C	ICONC=00
0037_{16}	Interrupt polarity selection register	IPOL=00

The interrupt control unit (ICU), a specialized peripheral, is described in detail in this section.

This series supports a maximum of 23 maskable interrupts, one software interrupt, and one reset vector that is treated as a non-maskable interrupt.

See Table 2-4 for the interrupt sources, jump destination addresses, and interrupt priorities. For example, User Reset has a jump destination storage address of FFFA and FFFB and a priority of 3.

2.7.1 Interrupt Control

Each maskable interrupt has associated with it an interrupt request bit and an interrupt enable bit. These bits, along with the I flag, determine whether interrupt events can cause an interrupt service request to be generated. An interrupt request bit is set High when its corresponding interrupt event is activated. The bit is cleared to a "0" when the interrupt is serviced or when a "0" is written to the bit. The bit can not be set High by writing "1" to it.

Each interrupt enable bit determines whether the interrupt request bit it is paired with is seen when the interrupts are polled. When the interrupt enable bit is Low, the interrupt request bit is not seen; and when the enable bit is High, the interrupt request is seen.

The interrupt request register configurations for the 23 maskable interrupts are shown in Figure 2-26., Figure 2-27., and Figure 2-28. The interrupt control register configurations for the 23 maskable interrupts are shown in Figure 2-29., Figure 2-30., and Figure 2-31. The configuration of the polarity register for the external interrupts is shown in Figure 2-32.

MSB 7	IRA7	IRA6	IRA5	IRA4	IRA3	IRA2	IRA1	IRA0	LSB 0

Address: 0002_{16}
Access: R/W
Reset: 00_{16}

IRA0	USB Function Interrupt Request Bit 0
IRA1	USB SOF Interrupt Request Bit 1
IRA2	External Interrupt 0 Request Bit 2
IRA3	External Interrupt 1 Request Bit 3
IRA4	DMAC channel 0 Interrupt Request Bit 4
IRA5	DMAC channel 1 Interrupt Request Bit 5
IRA6	UART1 Receive Buffer Full Interrupt Request Bit 6
IRA7	UART1 Transmit Interrupt Request Bit 7
	0: No interrupt request issued
	1: Interrupt request issued

Figure 2-26. IREQA Configuration

MSB 7	IRB7	IRB6	IRB5	IRB4	IRB3	IRB2	IRB1	IRB0	LSB 0

Address: 0003_{16}
Access: R/W
Reset: 00_{16}

IRB0	UART1 Error Sum Interrupt Request Bit 0
IRB1	UART2 Receive Buffer Full Interrupt Request Bit 1
IRB2	UART2 Transmit Interrupt Request Bit 2
IRB3	UART2 Error Sum Interrupt Request Bit 3
IRB4	Timer X Interrupt Request Bit 4
IRB5	Timer Y Transmit Interrupt Request Bit 5
IRB6	Timer 1 Interrupt Request Bit 6
IRB7	Timer 2 Interrupt Request Bit 7
	0: No interrupt request issued
	1: Interrupt request issued

Figure 2-27. IREQB Configuration

MSB 7	Reserved	IRC6	IRC5	IRC4	IRC3	IRC2	IRC1	IRC0	LSB 0

Address: 0004_{16}
Access: R/W
Reset: 00_{16}

IRC0	Timer 3 Interrupt Request Bit 0
IRC1	External CNTR0 Interrupt Request Bit 1
IRC2	External CNTR1 Interrupt Request Bit 2
IRC3	SIO Interrupt Request Bit 3
IRC4	Input Buffer Full Interrupt Request Bit 4
IRC5	Output Buffer Empty Request Bit 5
IRC6	Key-on Wake-up Interrupt Request Bit 6
	0: No interrupt request issued
	1: Interrupt request issued
Bit 7	BRK Instruction - Reserved (0 when read)

Figure 2-28. IREQC Configuration

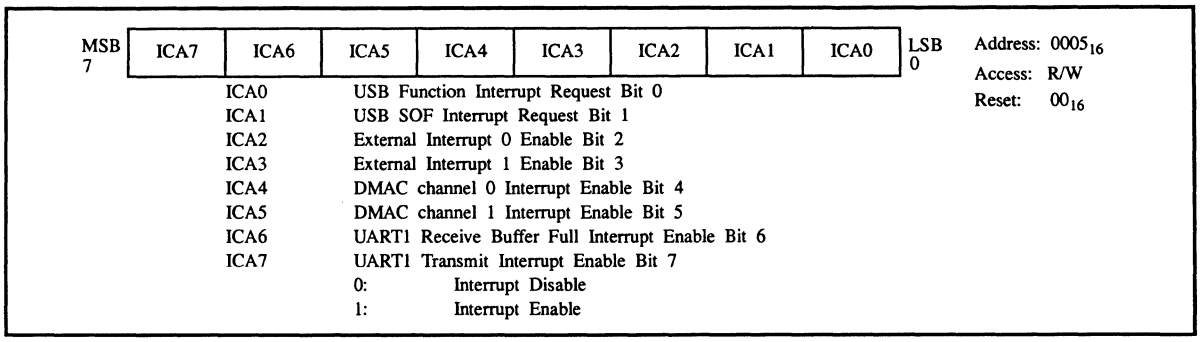


Figure 2-29. ICONA Configuration

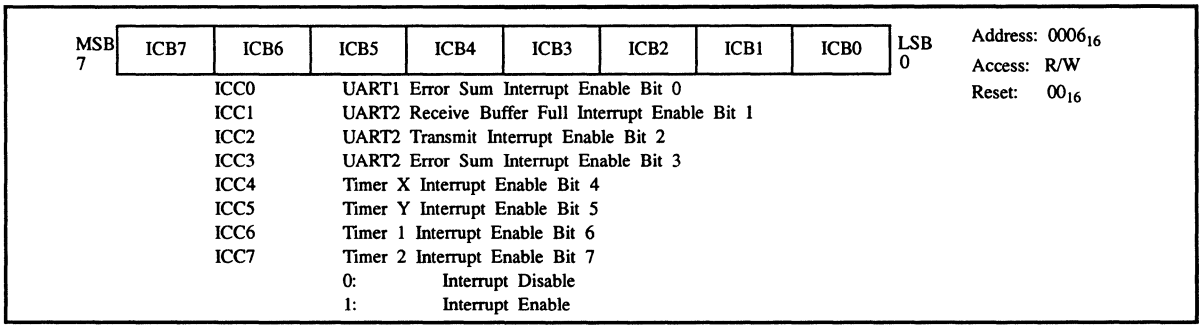


Figure 2-30. ICONB Configuration

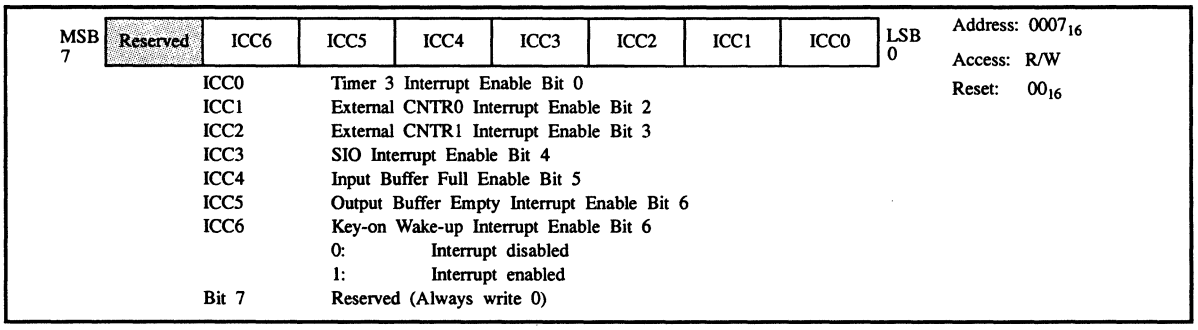


Figure 2-31. ICONC Configuration

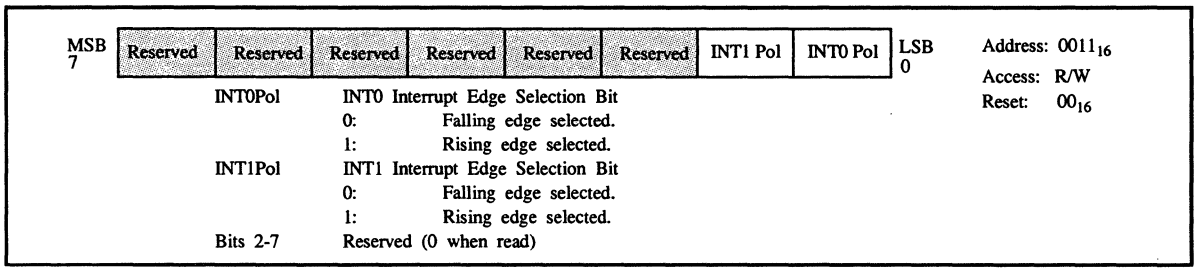


Figure 2-32. IPOL Configuration

Table 2-4 Interrupt Vector Table

Priority	Interrupt	Jump Destination Storage Address (Vector Address)		Remarks				
		High-order Byte	Low-order Byte					
1	RSRV1	FFFF	FFFE	Reserved for factory use				Corresponding Register Assignment
2	RSRV2	FFFD	FFFC	Reserved for factory use				
3	Reset	FFFB	FFFA	User-Reset (Non-Maskable)				
4	USB	FFF9	FFF8	USB Function Interrupt	0	LSB	IREQA and ICONA	
5	SOF	FFF7	FFF6	USB SOF Interrupt	1			
6	INT0	FFF5	FFF4	External Interrupt 0	2			
7	INT1	FFF3	FFF2	External Interrupt 1	3			
8	DMA1	FFF1	FFF0	DMAC Channel 0 Interrupt	4			
9	DMA2	FFEF	FFEE	DMAC Channel 1Interrupt	5			
10	U1RBF	FFED	FFEC	UART1 Receiver Buffer Full	6			
11	U1TX	FFEB	FFEA	UART1 Transmit Interrupt	7	MSB		
12	U1ES	FFE9	FFE8	UART1 Error Sum Interrupt	0	LSB	IREQB and ICONB	
13	U2RBF	FFE7	FFE6	UART2 Receiver Buffer Full	1			
14	U2TX	FFE5	FFE4	UART2 Transmit Interrupt	2			
15	U2ES	FFE3	FFE2	UART2 Error Sum Interrupt	3			
16	TX	FFE1	FFE0	Timer X	4			
17	TY	FFDF	FFDE	Timer Y	5			
18	T1	FFDD	FFDC	Timer 1	6			
19	T2	FFDB	FFDA	Timer 2	7	MSB		
20	T3	FFD9	FFD8	Timer 3	0	LSB	IREQC and ICONC	
21	CNTR0	FFD7	FFD6	External CNTR0 Interrupt	1			
22	CNTR1	FFD5	FFD4	External CNTR1 Interrupt	2			
23	SIO	FFD3	FFD2	SIO Interrupt	3			
24	IBF	FFD1	FFD0	Input Buffer Full Interrupt	4			
25	OBE	FFDF	FFCE	Output Buffer Empty Interrupt	5			
26	KEY	FFCD	FFCC	Key-on Wake Up	6	MSB		
27	BRK	FFCB	FFCA	BRK Instruction (Non-Maskable)				

2.7.2 Interrupt Sequence and Timing

The interrupts are polled prior to the beginning of each instruction. An interrupt service request is generated when an interrupt event has its interrupt request bit set High, its interrupt enable bit is set High, and the interrupt inhibit flag I is set Low. The I flag is used to disable all maskable interrupts. When this bit is set High, only a BRK instruction or a user Reset can cause an interrupt service request to be generated. Figure 2-33. is a simplified version of the logic that controls whether an interrupt service request is generated.

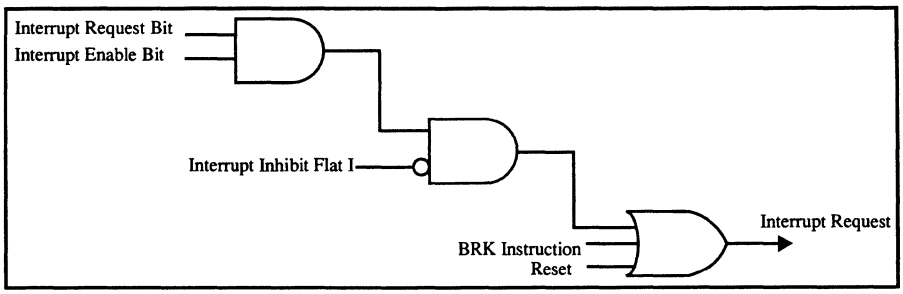


Figure 2-33. Interrupt Service Request Control Logic

The time elapsed from the occurrence of an interrupt event until execution of its service routine varies from 7 cycles to 23 cycles, depending on what instruction is executing when the interrupt event occurs (see Figure 2-34.)

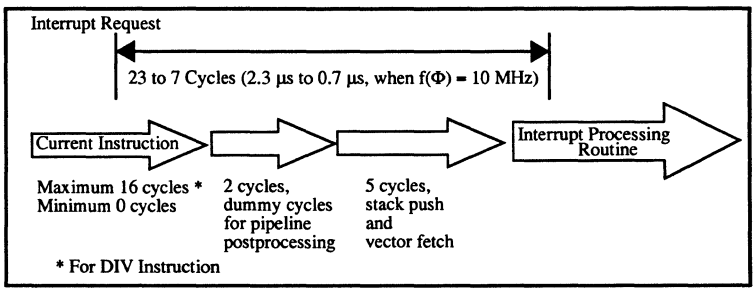


Figure 2-34. Execution Time Prior to Interrupt Service Routine

When an interrupt service request occurs, the current instruction stream is temporarily halted and the appropriate interrupt service routine is executed. After the interrupt service routine ends, the current instruction stream is resumed with the next instruction.

The interrupt service request causes the MCU to automatically push the high-order byte of the program counter, the low-order byte of the program counter, and the contents of the processor status register onto the stack. A push consists of storing data at the stack address and decrements the stack pointer by one. The I flag is set High to prevent other interrupts from being serviced during the interrupt service routine, and the request bit corresponding to the interrupting event is automatically cleared to "0". The program counter is set to the address specified in the vector table for the interrupt being serviced. This address contains the address for the first instruction of the interrupt service routine. The timing for the pushing of data onto the stack, and fetching the starting address of the interrupt routine is illustrated in Figure 2-35.

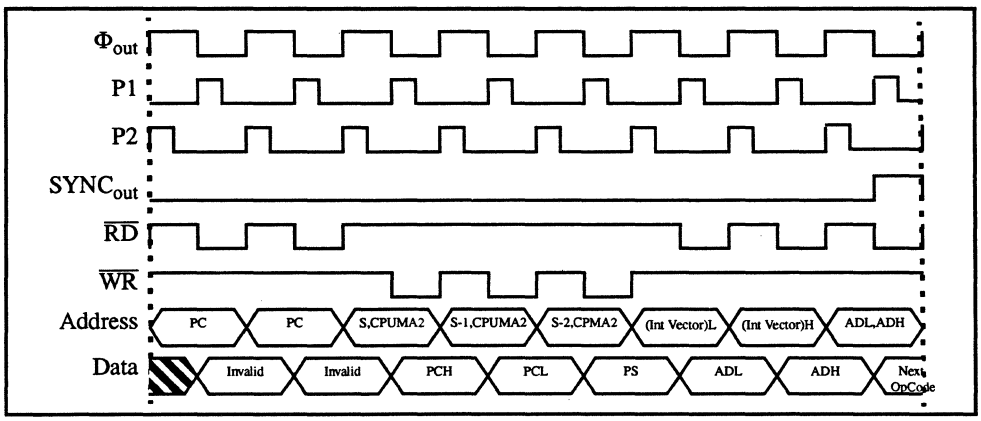


Figure 2-35. Interrupt Cycle Timing

See Figure 2-36. for the stack and program counter modifications that occur when an interrupt request is serviced.

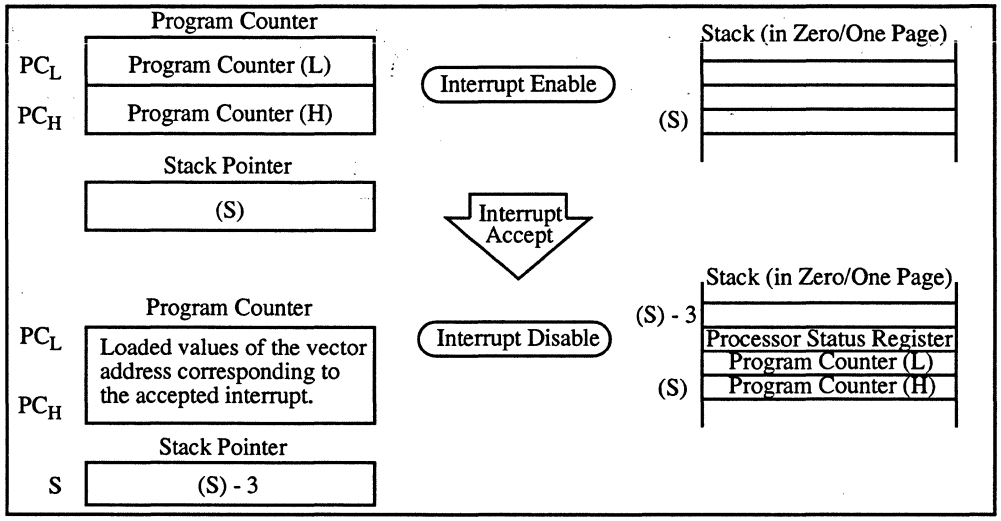


Figure 2-36. Stack Pointer and Program Counter Modifications During Interrupt Service Sequence

Returning from an interrupt is accomplished by executing an RTI instruction. This causes the MCU to pop the contents of the process status register and the low-order and high-order bytes of the program counter from the stack. The I flag is cleared to "0" when the process status value is restored from the stack.

2.8 *Direct Memory Access Controller*

Address	Description	Acronym and Value
$003F_{16}$	DMAC index and status register	DMAIS=00
0040_{16}	DMAC channel x mode register 1	DMAxM1=00
0041_{16}	DMAC channel x mode register 2	DMAxM2=00
0042_{16}	DMAC channel x source register Low	DMAxSL=00
0043_{16}	DMAC channel x source register High	DMAxSH=00
0044_{16}	DMAC channel x destination register Low	DMAxDL=00
0045_{16}	DMAC channel x destination register High	DMAxDH=00
0046_{16}	DMAC channel x transfer count register Low	DMAxCL=00
0047_{16}	DMAC channel x transfer count register High	DMAxCH=00

This device contains a two-channel Direct Memory Access Controller (DMAC). It performs fast data transfers between any two locations in the memory map initiated by specific peripheral events or software triggers.

The main features of the DMAC are as follows:

- Two independent channels
- Single-byte and burst transfer modes

- 16-bit source and destination address registers (for a 64K byte address space)
- 16-bit transfer count registers (for up to 64K bytes transferred before underflow)
- Source/Destination register automatic increment/decrement and no-change options
- Source/Destination/Transfer count register reload on write or after transfer count register underflow options
- Transfer requests from USB (9), Master CPU Bus Interface (4), external interrupts (4), UART1 (2), UART2 (2), SIO (1), TimerX (1), TimerY (1), Timer1 (1), and software triggers
- Closely coupled with USB and Master CPU Bus Interface for efficient data transfers
- Interrupt generated for each channel when their respective transfer count register underflows
- Fixed channel priority (channel 0 > channel 1)
- Two cycles of Φ required per byte transferred

Each channel of the DMAC is made up of the following:

- 16-bit source and destination registers
- A 16-bit transfer count register
- Two mode registers
- Status flags contained in a status register shared by the two channels
- Control and timing logic

The 16-bit source and destination registers allow accesses to any two locations in the 64K byte memory area. The 16-bit transfer count register decrements by one for each transfer performed and causes an interrupt and flag to be set when it underflows. The mode registers control the configuration and operation of the DMAC channel associated with the registers. A block diagram of the DMAC is shown in Figure 2-37.

The SFR addresses for the two mode, source, destination, and transfer count registers of a channel are the same for each channel. Which channel's registers are accessible is determined by the value of the DMAC Channel Index Bit (DCI) (bit 7 of the DMAC Index and Status Register (DMAIS)). When this bit is a "0", channel 0 registers are accessible, and when this bit is a one, channel 1 registers are accessible. The configuration of DMAIS and the mode registers are shown in Figure 2-38., Figure 2-39., Figure 2-40., and Figure 2-41.

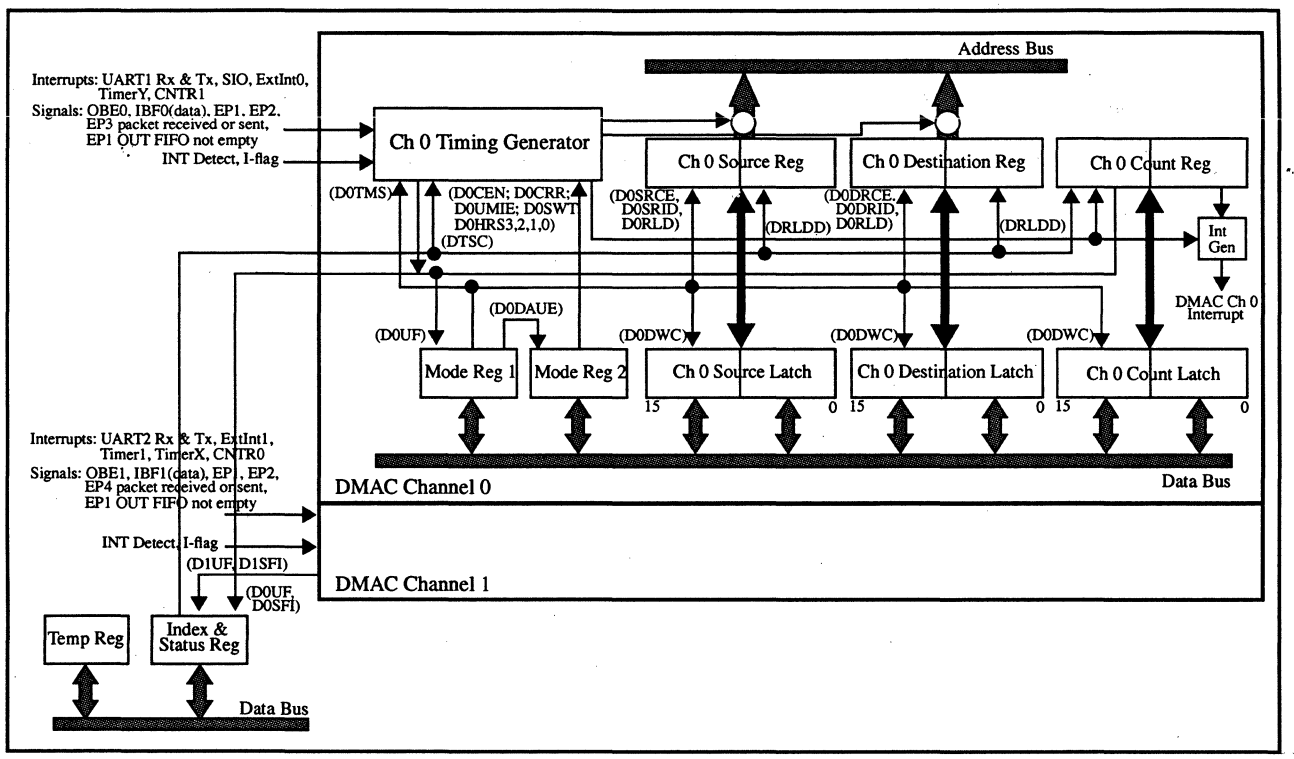


Figure 2-37. DMAC Block Diagram

2.8.1 Operation

Each channel of the DMAC transfers byte data from a source address to a destination address when a selected event occurs. If single-byte transfer mode is enabled, one byte of data is transferred per request. If burst transfer mode is enabled, several bytes can be transferred per request, one byte at a time. A temporary register internal to the DMAC stores the data read from the source address until it is written to the destination address on the next cycle. The transfer of one byte takes two cycles of Φ and causes the CPU and possibly the other DMAC channel to stall during this time. At least one cycle of Φ with the CPU operating must take place between transfers by the same DMAC channel caused by different events or between transfers by the two DMAC channels.

2.8.1.1 Source, Destination, and Transfer Count Register Operation

The user can choose whether the source and destination register for each channel will increment by one, decrement by one, or remain unchanged after each transfer by setting bits 0 through 3 (DxSRID, DxSRCE, DxDRID, and DxDRCE) of DMAC Channel x Mode Register 1 (DMAxM1) to the appropriate values. The values in the source and destination registers are updated with their reload latch values when the transfer count register underflows. The transfer count register is also reloaded when it underflows, and both a flag (DxUF) and the DMAC interrupt associated with the channel are set. Reload of the source and destination registers due to underflow of the transfer count register can be disabled by setting to one the DMAC Register Reload Disable Bit (DRLDD). This bit affects reload for both channel 0 and channel 1.

Because the transfer count register is 16-bits wide, up to 65,536 transfers can take place before an underflow and the resulting actions described above occur. If the Channel x Disable After Count Register Underflow Enable Bit (DxDAUE) is set High, then the Channel x Enable Bit (DxCEN) is cleared to a "0" when the transfer count register underflows, disabling channel x of the DMAC.

The source, destination, and transfer count registers of a DMAC channel can be updated with their reload latch value at any time by setting to a one the DMAC Channel x Register Reload Bit (DxRLD).

DMAC Source, Destination, and Transfer Count Register Read and Write Method

Read and write operations on the high and low-order bytes of the source, destination, and transfer count registers must be performed in a specific order.

Write Method
When writing to the source, destination, or transfer count register, the low-order byte is written first. Next, the high-order byte is written. When this is done, the data is placed in the reload latch of the high-order byte of the register and the previously written low-order byte data is placed in the reload latch of the low-order byte. At this point, if the DMAC Channel x Write Control Bit (DxDWC) is "0", the values in the reload latches are also loaded in the low and high-order bytes of the register. If DxDWC is one, the data in the reload latches are loaded in the register after the transfer count register of the DMAC channel underflows or the DxRLD bit of the DMAC channel is set High.

Read Method
When reading from the source, destination, or transfer count register, the high-order byte is read first. The low-order byte of the register is then read. The value read from the low-order byte of the register is its value when the high-order byte was read.

2.8.1.2 DMAC Transfer Request Sources

The hardware source for initiating a DMAC transfer for each channel is selectable by setting the DMAC Channel x Hardware Transfer Request Source Bits (DxHRS0, 1, 2, 3) to appropriate values.

The choices for channel 0 are the UART1 receive or transmit interrupts, the TimerY interrupt, external interrupt 0, one of three USB endpoint packet received signals, one of three USB endpoint packet sent signals, the USB endpoint 1 OUT FIFO not empty signal, the OBE0 and IBF0 (data) signals from the Master CPU Bus Interface, the SIO combined receive/transmit interrupt, and the CNTR1 interrupt.

The choices for channel 1 are the UART2 receive and transmit interrupts, the TimerX interrupt, external interrupt 1, one of three USB endpoint packet received signals, one of three USB endpoint packet sent signals, the USB endpoint 1 OUT FIFO not empty signal, the OBE1 and IBF1 (data) signals from the Master CPU Bus Interface, the Timer1 interrupt, and the CNTR0 interrupt.

In addition, each channel has a software trigger that can initiate a DMAC transfer. The software trigger is set by writing a "1" to the DMAC Software Transfer Trigger (DxSWT). The hardware transfer request source for each channel can be disabled by writing a "0" to DxHRS0, 1, 2, and 3. When these bits are all "0", which is the reset state, only the software trigger can be used to initiate a transfer.

The initiating source for each channel is latched by the DMAC asynchronously and sampled on the rising edge of Φ. Writing a "1" to the DMAC Channel x Transfer Initiation Source Capture Register Reset bit (DxCRR) causes the initiating source sample latch of the associated DMAC channel to be reset. The sample latch is reset automatically one cycle of PHI after a transfer request is detected. New transfer requests for a channel that occur during a DMAC transfer by that same channel are latched as long as they occur after the sample latch is reset. However, if multiple transfer requests occur during a transfer, only one transfer request will be registered.

If an interrupt is chosen as the initiating source for DMAC transfers, its interrupt control bit located in one of the three interrupt control registers of the ICU should be cleared to "0" if the user does not wish to have the interrupt serviced by the CPU.

2.8.1.3 Transfer Features for USB and Master CPU Bus Interface

In order to make the transfer of data between the USB endpoint FIFOs and the input and output buffers of the Master CPU Bus Interface more efficient, special features have been included in the transfer request logic of each DMAC channel. These features are enabled for a channel when one of the USB endpoint signals is selected as the hardware transfer request source and the DMAC Channel x USB and Master CPU Bus Interface Enable Bit (DxUMIE) is set High. These features are only intended to be used with single-byte transfer mode.

USB OUT FIFO to Master CPU Bus Interface Output Buffer Transfers

When a USB endpoint packet received signal is selected as the hardware transfer request source for a DMAC channel and the DxUMIE bit of the same channel is set High, a transfer request is generated for that DMAC channel when the packet received signal for the chosen USB endpoint is active and output buffer x (where x is "0" for DMAC channel 0 and "1" for DMAC channel 1) of the Master CPU Bus Interface is empty. The packet received signal remains active until all bytes of the packet have been read from the FIFO corresponding to that endpoint. Thus, the first transfer request is generated when the packet received signal goes active and subsequent transfer requests are generated each time output buffer x becomes empty. Once the final byte of the received packet has been read, the packet received signal automatically goes inactive (if this option is enabled in the USB block). This in turn causes the source, destination, and transfer count registers of the involved DMAC channel to be reloaded (unless the DRLDD bit is set High) and the DMAC interrupt for the involved channel to be set. In addition, if the DxDAUE bit associated with the channel is "1", the channel's DxCEN bit is automatically cleared to "0", disabling the channel.

This feature allows a channel of the DMAC in single-byte transfer mode to automatically transfer a received packet of an endpoint from the endpoint's FIFO to the master CPU (via the Master CPU Bus Interface) without any intervention by the on-chip CPU. Also, because the source, destination, and transfer count registers are automatically reloaded once the current packet has been completely transferred, on-chip CPU intervention is not needed to set up the DMAC channel for transfer of subsequently received packets, even in the case of reception of a short packet.

A second method for generating DMAC transfer requests based on a signal from USB endpoint 1 and the status of output buffer x of the Master CPU Bus Interface is supported. This method facilitates byte-by-byte transfers; whereas, the method described above facilitates packet-by-packet transfers. This method is enabled when the USB endpoint 1 OUT FIFO not empty signal is chosen as the hardware transfer request source for a DMAC channel and the DxUMIE bit of the same channel is set High. A transfer request is generated for the DMAC channel if the endpoint 1 FIFO is not empty and output buffer x of the Master CPU Bus Interface is empty. As is the case when the packet-by-packet method is used, the packet received signal goes active once a complete packet has been received. It remains active until all bytes of the packet have been read from the FIFO. When the final byte has been read from the FIFO, the packet received signal goes inactive (if this option is enabled in the USB block), which causes the source, destination, and transfer count registers of the involved DMAC channel to be reloaded (unless the DRLDD bit is set High) and the DMAC interrupt corresponding to the involved channel to be set. Also, if the DxDAUE bit associated with the channel is "1", the channel's DxCEN bit is automatically cleared to "0", disabling the channel. If the last byte of the packet has been read from the FIFO before the end_of_packet signal is received by the USB block, the packet received signal will still go to its active state and then go inactive a short period of time later (if this option is enabled in the USB block).

Master CPU Bus Interface Input Buffer to USB IN FIFO Transfers

When a USB endpoint packet sent signal is selected as the hardware transfer request source for a DMAC channel and the DxUMIE bit of the same channel is set High, a transfer request is generated when the FIFO associated with the endpoint is not full (with respect to the programmed packet size) and input buffer x of the Master CPU Bus Interface contains data. The transfer request is not generated if input buffer x contains a command. The FIFO not full signal of the endpoint remains active until a full packet has been written to the FIFO. Thus, the first transfer request is generated when the FIFO not full signal goes active and subsequent transfer requests are generated when data is written to input buffer x by an external device. Once the full packet has been written to the FIFO, the FIFO not full signal goes inactive and the in_packet_ready signal of the endpoint is automatically set (if this option is enabled in the USB block). In this case, the source, destination, and transfer count registers are not automatically reloaded. Instead, the packet size for the endpoint should be written to the transfer count register at initialization time so that it underflows and reloads once the last byte of the data is transferred from input buffer x to the endpoint's FIFO.

The feature described above allows a channel of the DMAC in single-byte transfer mode to automatically transfer data received from the master CPU (via the Master CPU Bus Interface) to the endpoint's FIFO without any intervention by the on-chip CPU. Additionally, since the in_packet_ready signal associated with the endpoint is automatically set (if this option is enabled in the USB block), multiple packets can be transferred by a channel of the DMAC without on-chip CPU intervention. Note however that short packets are not handled automatically and instead require intervention by the on-chip CPU.

2.8.1.4 DMAC Transfer Mode

Each channel of the DMAC can be operated in single-byte transfer mode or burst transfer mode. The choice is made by the setting of the Channel x DMAC Transfer Mode Selection Bit (DxTMS). When single-byte transfer mode is selected, one byte of data is transferred per transfer request. When burst transfer mode is selected, the value in the transfer count register determines how many single byte transfers occur per transfer request. For example, if the value in the transfer count register is 0014_{16}, 21 transfers will occur before control of the address bus and data bus is given back to the CPU.

2.8.1.5 DMAC Transfer Timing

A DMAC transfer can occur at any point during the execution of an instruction by the CPU. However, at least one cycle of Φ with the CPU operating takes place between transfers by the same DMAC channel caused by different events or between transfers by the two DMAC channels. Also, burst transfers and possibly single-byte transfers are prevented from occurring during interrupt service routines.

The transfer initiating sources for the two channels are latched by the DMAC asynchronously and polled on the rising edge of Φ. If a transfer request is seen for both channels, the channel 0 request will be serviced first followed by the channel 1 request.

If channel 1 is performing a burst transfer when channel 0 receives a transfer request, the channel 1 transfer is suspended at the end of the next source read/destination write operation. The channel 0 transfer is then serviced. Once the channel 0 transfer completes, the channel 1 transfer automatically continues where it left off. In order to prevent channel 0 from completely shutting out channel 1 transfers, one cycle of a suspended channel 1 transfer is allowed to occur after a channel 0 burst transfer even if another channel 0 transfer request is pending.

If the I flag value is Low and an interrupt with its interrupt control bit High occurs during a burst transfer by either channel, the transfer is suspended, allowing the interrupt service routine to be entered. The DMAC Channel x Suspend (due to interrupt service request) Flag (DxSFI) corresponding to the channel whose transfer was suspended is automatically set High at this time. When the I flag value

(which was automatically set High when the interrupt service routine was entered) becomes a "0" again, the DxSF flag is automatically cleared and the transfer continues where it left off. If a DMAC burst transfer request occurs during the servicing of an interrupt, the transfer does not take place until after the interrupt has been serviced, which is understood to have happened when the I flag becomes a "0".

If the DMAC Transfer Suspend Control Bit (DTSC) is set High, both single-byte and burst mode transfers are suspended by interrupts.

A suspended DMAC transfer can be re-started in the interrupt service routine by writing a "1" to the DxCEN bit of the suspended channel.

Sample timing diagrams are shown in Figure 2-42., Figure 2-43., and Figure 2-44. for a single-byte transfer initiated by a hardware source, a single-byte transfer initiated by the software trigger, and a burst transfer initiated by a hardware source, respectively.

MSB 7								LSB 0
DCI	Reserved	DRLDD	DTSC	D1SFI	D1UF	D0SFI	D0UF	

Address: $003F_{16}$
Access: R/W
Reset: 00_{16}

D0UF	DMAC Channel 0 Count Register Underflow Flag - Bit 0 0: Channel 0 transfer count register underflow has not occurred 1: Channel 0 transfer count register underflow has occurred
D0SFI	DMAC Channel 0 Suspend (due to interrupt service request) Flag - Bit 1 0: Channel 0 transfer has not been suspended 1: Channel 0 transfer has been suspended
D1UF	DMAC Channel 1 Count Register Underflow Flag - Bit 2 0: Channel 1 transfer count register underflow has not occurred 1: Channel 1 transfer count register underflow has occurred
D1SFI	DMAC Channel 1 Suspend (due to interrupt service request) Flag - Bit 3 0: Channel 1 transfer has not been suspended 1: Channel 1 transfer has been suspended
DTSC	DMAC Transfer Suspend Control Bit - Bit 4 0: Only burst transfers are suspended during interrupt servicing 1: Both burst and single-byte transfers are suspended during interrupt servicing
DRLDD	DMAC Register Reload Disable Bit - Bit 5 0: Reload of source and destination registers of both channels enabled 1: Reload of source and destination registers of both channels disabled
Bit 6	Reserved (write 0)
DCI	Channel Index Bit - Bit 7 0: Channel 0 mode, source, destination, and transfer count registers accessible 1: Channel 1 mode, source, destination, and transfer count registers accessible

Figure 2-38. DMAIS Configuration

MSB 7								LSB 0
DxTMS	DxRLD	DxDAUE	DxDWC	DxDRCE	DxDRID	DxSRCE	DxSRID	

Address: 0040_{16}
Access: R/W
Reset: 00_{16}

DxSRID DMAC Channel x Source Register Increment/Decrement Select Bit - Bit 0
0: Increment after transfer
1: Decrement after transfer

DxSRCE DMAC Channel x Source Register Increment/Decrement Enable Bit - Bit 1
0: Increment/Decrement disabled (No change after transfer)
1: Increment/Decrement enabled

DxDRID DMAC Channel x Destination Register Increment/Decrement Select Bit - Bit 2
0: Increment after transfer
1: Decrement after transfer

DxDRCE DMAC Channel x Destination Register Increment/Decrement Enable Bit - Bit 3
0: Increment/Decrement disabled (No change after transfer)
1: Increment/Decrement enabled

DxDWC DMAC Channel x Data Write Control Bit - Bit 4
0: Write data in reload latches and registers
1: Write data in reload latches only

DxDAUE DMAC Channel x Disable After Count Register Underflow Enable Bit - Bit 5
0: Channel x not disabled after count register underflow
1: Channel x disabled after count register underflow

DxRLD DMAC Channel x Register Reload Bit - Bit 6
0: No action (Bit is always read as zero)
1: Setting to "1" causes the source, destination, and transfer count registers of channel x to be reloaded

DxTMS DMAC Channel x Transfer Mode Selection Bit - Bit 7
0: Single-byte transfer mode
1: Burst transfer mode

Figure 2-39. DMAxM1 Configuration

MSB 7								LSB 0
D0CRR	D0CRR	D0UMIE	D0SWT	D0HRS3	D0HRS2	D0HRS1	D0HRS0	

Address: 0041_{16}
Access: R/W
Reset: 00_{16}

D0HRS3,2,1,0 DMAC Channel 0 Hardware Transfer Request Source Bits - Bits 3, 2, 1, 0
0000: Disabled
0001: UART1 receive interrupt
0010: UART1 transmit interrupt
0011: TimerY interrupt
0100: External Interrupt 0
0101: USB EndPoint 1 packet sent signal
0110: USB EndPoint 2 packet sent signal
0111: USB EndPoint 3 packet sent signal
1000: USB EndPoint 1 packet received signal
1001: USB EndPoint 1 OUT FIFO not empty signal
1010: USB EndPoint 2 packet received signal
1011: USB EndPoint 3 packet received signal
1100: Master CPU Bus Interface OBE0 signal
1101: Master CPU Bus Interface IBF0(data) signal
1110: SIO receive/transmit interrupt
1111: CNTR1 interrupt

D0SWT DMAC Channel 0 Software Transfer Trigger - Bit 4
0: No action (bit is always read as zero)
1: Writing "1" requests a channel 0 transfer

D0UMIE DMAC Channel 0 USB and Master CPU Bus Interface Enable Bit - Bit 5
0: Disabled
1: Enabled

D0CRR DMAC Channel 0 Transfer Initiation Source Capture Register Reset - Bit 6
0: No action (Bit is always read as zero)
1: Setting to "1" causes reset of the channel 0 capture register

D0CEN DMAC Channel 0 Enable Bit - Bit 7
0: Channel 0 disabled
1: Channel 0 enabled

Figure 2-40. DMA0M2 Configuration

MSB 7								LSB 0
D1CEN	D1CRR	D1UMIE	D1SWT	D1HRS3	D1HRS2	D1HRS1	D1HRS0	

Address: 0041_{16}
Access: R/W
Reset: 00_{16}

D1HRS3,2,1,0 DMAC Channel 1Hardware Transfer Request Source Bits - Bits 3, 2, 1, 0
0000: Disabled
0001: UART2 receive interrupt
0010: UART2 transmit interrupt
0011: TimerX interrupt
0100: External Interrupt 1
0101: USB EndPoint 1 packet sent signal
0110: USB EndPoint 2 packet sent signal
0111: USB EdnPoint 4 packet sent signal
1000: USB EndPoint 1 packet received signal
1001: USB EndPoint 1 OUT FIFO not empty signal
1010: USB EndPoint 2 packet received signal
1011: USB EndPoint 4 packet received signal
1100: Master CPU Bus Interface OBE1 signal
1101: Master CPU Bus Interface IBF1(data) signal
1110: Timer1 interrupt
1111: CNTR0 interrupt

D1SWT DMAC Channel 1 Software Transfer Trigger - Bit 4
0: No action (bit is always read as zero)
1: Writing "1" requests a channel 0 transfer

D1UMIE DMAC Channel 1 USB and Master CPU Bus Interface Enable Bit - Bit 5
0: Disabled
1: Enabled

D1CRR DMAC Channel 1 Transfer Initiation Source Capture Register Reset - Bit 6
0: No action (Bit is always read as zero)
1: Setting to "1" causes reset of the channel 1 capture register

D1CEN DMAC Channel 1 Enable Bit - Bit 7
0: Channel 1 disabled
1: Channel 1 enabled

Figure 2-41. DMA1M2 Configuration

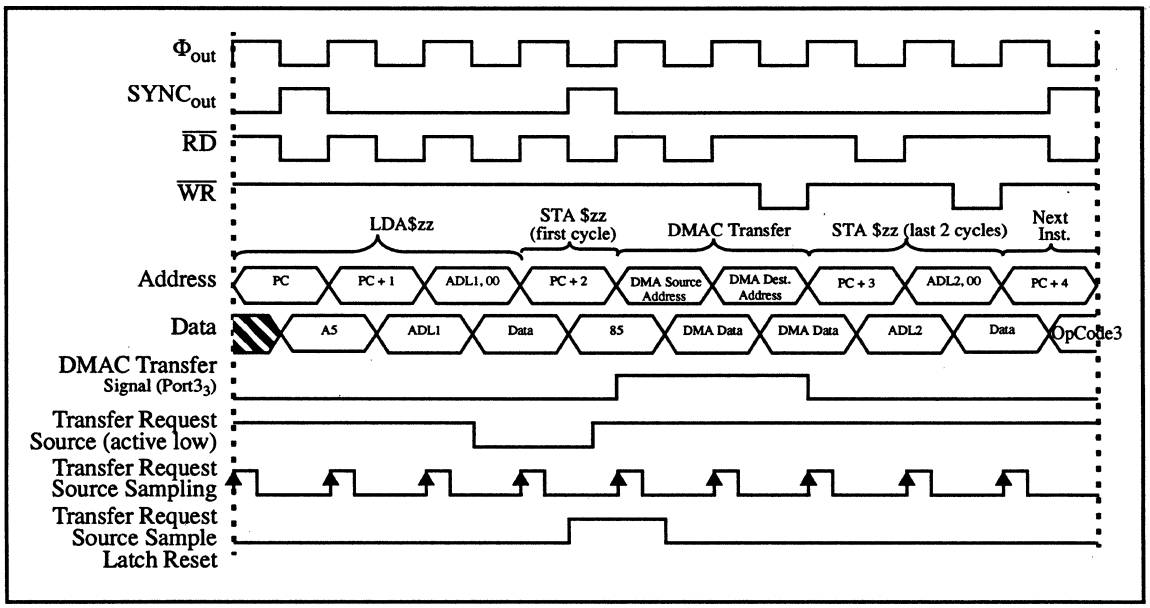


Figure 2-42. DMAC Transfer - Hardware Source Initiated

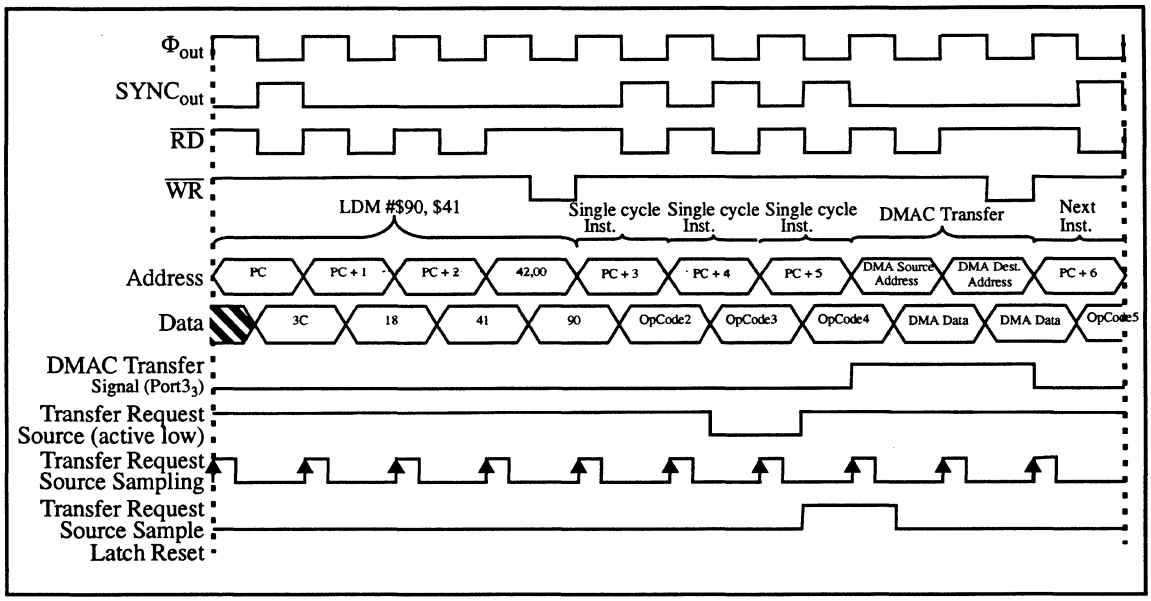


Figure 2-43. DMAC Transfer - Software Trigger Initiated

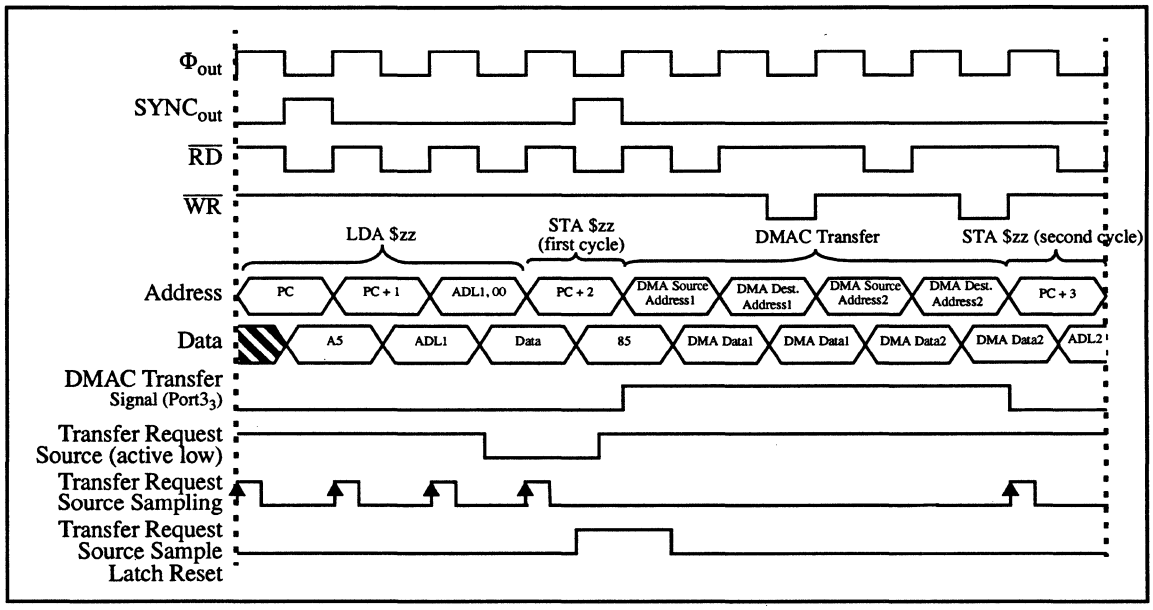


Figure 2-44. DMAC Transfer - Burst Transfer Mode

2.9 Timers

Address	Description	Acronym and Value at Reset
0020_{16}	Timer XL	TXL=FF
0021_{16}	Timer XH	TXH=FF
0022_{16}	Timer YL	TYL=FF
0023_{16}	Timer YH	TYH=FF
0024_{16}	Timer 1	T1=FF

Address	Description	Acronym and Value at Reset
0025_{16}	Timer 2	T2=01
0026_{16}	Timer 3	T3=FF
0027_{16}	Timer X mode register	TXM=00
0028_{16}	Timer Y mode register	TYM=00
0029_{16}	Timer 123 mode register	T123M=00

This device has five built-in timers: Timer X, Timer Y, Timer 1, Timer 2, and Timer 3.

The contents of the timer latch, corresponding to each timer, determine the divide ratio. The timers can be read or written at any time. However, the read and write operations on the high and low-order bytes of the 16-bit timers (Timer X and Y) must be performed in a specific order.

The timers are all down count timers; when the count of a timer reaches 00_{16} (0000_{16} for Timer X and Y), an underflow occurs at the next count pulse and the contents of the corresponding timer reload latch are reloaded into the timer. When a timer underflows, the interrupt request bit corresponding to that timer is set High.

The divide ratio of a timer is given by $1/(n + 1)$, where n is the value written to the timer. When the STP instruction is executed, 01_{16} is loaded into Timer 2 and the Timer 2 reload latch, and FF_{16} is loaded into Timer 1 and the Timer 1 reload latch.

Figure 2-48. is a block diagram of the five timers.

2.9.1 Timer X

Timer X is a 16-bit timer that has a 16-bit reload latch, and can be placed in one of four modes by setting bits TXM4 and TXM5 (bits 4 and 5 of the Mode Register, TXM). The bit assignment of the TXM is shown in Figure 2-45..

Bit5 -TXM5	Bit4 -TXM4	Timer X Mode
0	0	Timer mode
0	1	Pulse output mode
1	0	Event counter mode
1	1	Pulse width measurement mode

2.9.1.1 Read and Write Method

Read and write operations on the high and low-order bytes of Timer X must be performed in a specific order.

Write Method

When writing to the timer, the lower order byte is written first. This data is placed in a temporary register that is assigned the same address as Timer XL. Next, the higher order byte is written. When this is done, the data is placed in the Timer XH reload latch and the low-order byte is transferred from its temporary register to the Timer XL reload latch. At this point, if the Timer X Data Write Control Bit (TXM0) (bit 0) is "0", the value in the Timer X reload latch is also loaded in Timer X. If TXM0 is "1", the data in the Timer X reload latch is loaded in Timer X after Timer X underflows.

MSB 7	TXM7	TXM6	TXM5	TXM4	TXM3	TXM2	TXM1	TXM0	LSB 0

Address: 0027_{16}
Access: R/W
Reset: 00_{16}

TXM0 Timer X Data Write Control Bit - Bit 0
0:Write data in latch and timer
1:Write data in latch only

TXM2,1 Timer X Frequency Division Ratio Bits - Bits 2 and 1
00:Φ divided by 8
01:Φ divided by 16
10:Φ divided by 32
11:Φ divided by 64

TXM3 Timer X Internal Clock Select Bit Bit 3
0:ΦΦ/n
1:SCSGCLK (from chip special count source generator)

TXM5,4 Timer X Mode Bits - Bit 5 and 4
00:Timer Mode
01:Pulse output mode
10:Event counter mode
11:Pulse width measurement mode

TXM6 CNTR0 Polarity Select Bit - Bit 6
0:For event counter mode, clocked by rising edge
For pulse output mode, start from High level output
For CNTR0 interrupt request, falling edge active
For pulse width measurement mode, measure High period
1:For event counter mode, clocked on falling edge
For pulse output mode, start from Low level output
For CNTR0 interrupt request, rising edge active
For pulse width measurement mode, measure Low period

TXM7 Timer X Stop Bit - Bit 7
0:Count start
1:Count stop

Figure 2-45. TXM Register

Read Method

When reading Timer X, the high-order byte is read first. Reading the high-order byte causes the values of Timer XH and Timer XL to be placed in temporary registers assigned the same addresses as Timer XH and Timer XL. The low-order byte of Timer X is then read from its temporary register. This operation assures the correct reading of Timer X while it is counting.

2.9.1.2 Count Stop Control

If the Timer X Count Stop Bit (TXM7) (bit 7 of the TXM) is set High, Timer X stops counting in all four modes.

2.9.1.3 Timer Mode

Count Source: *Φ/n* (where *n* is 8, 16, 32, or 64) or SCSGCLK

In this mode, each time the timer underflows, the corresponding timer interrupt request bit is set High, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

2.9.1.4 Pulse Output Mode

Count Source: *Φ/n* (where *n* is 8, 16, 32, or 64) or SCSGCLK

Each time the timer X underflows, the output of the CNTR0 pin is inverted, and the corresponding Timer X interrupt request bit is set High. The repeated inversion of the CNTR0 pin output produces a rectangular waveform with a duty ratio of 50 percent. The initial level of the output is determined by the CNTR0 polarity select bit (bit 6). When this bit is Low, the output starts from a High level. When this bit is High, the output starts from a Low level.

2.9.1.5 Event Counter Mode

Count Source: CNTR0

Timer countdown is triggered by inputs to the CNTR0 pin. Each time a timer underflows, the corresponding timer interrupt request bit is set High, the contents of the timer reload latch are loaded into the timer, and the countdown sequence begins again.

The edge used to clock Timer X is determined by the CNTR0 polarity select bit (bit 6).

2.9.1.6 Pulse Width Measurement Mode

Count Source: Φ/*n* (where *n* is 8, 16, 32, or 64) or SCSGCLK

This mode measures either the high or low-pulse width of the signal on the CNTR0 pin. The pulse width measured is determined by the CNTR0 polarity select bit (bit 6). When this bit is "0", the High pulse is measured. When this bit is "1", the Low pulse is measured.

The timer counts down while the level on the CNTR0 pin is the polarity selected by the CNTR0 polarity select bit. When the timer underflows, the Timer X interrupt request bit is set High, the contents of the timer reload latch are reloaded into the timer, and the timer continues counting down. Each time the signal polarity switches to the inactive state, a CNTR0 interrupt occurs indicating that the pulse width has been measured. The width of the measured pulse can be found by reading Timer X during the CNTR0 interrupt service routine.

2.9.2 Timer Y

Timer Y is a 16-bit timer that has a 16-bit reload latch, and can be placed in any of four modes by setting TYM4 and TYM5 (bits 4 and 5) (see Figure 2-46.). The desired mode is selected by modifying the values of TYM4 and TYM5.

Bit5 - TYM5	Bit4 -TYM4	Timer Y Mode
0	0	Timer mode
0	1	Pulse period measurement mode
1	0	Event counter mode
1	1	HL Pulse width measurement mode

MSB 7	TYM7	TYM6	TYM5	TYM4	TYM3	TYM2	TYM1	TYM0	LSB 0

Address: 0028_{16}
Access: R/W
Reset: 00_{16}

TYM0 Timer Y Data Write Control Bit - Bit 0
0:Write data in latch and timer
1:Write data in latch only

TYM1 Timer Y Output Control Bit - Bit 1
0:TYOUT output disable
1:TYOUT output enable

TYM3,2 Timer Y Frequency Division Ratio Bits - Bit 3 and 2
00:Φ divided by 8
01:Φ divided by 16
10:Φ divided by 32
11:ΦΦ divided by 64

TYM5,4 Timer Y Mode Bits - Bit 5 and 4
00:Timer mode
01:Pulse period measurement mode
10:Event counter mode
11:HL pulse width measurement mode (continuously measures High period and Low period)

TYM6 CNTR1 Polarity Select Bit - Bit 6
0:For event counter mode, clocked by rising edge
For pulse period measurement mode, falling edge detection
For CNTR1 interrupt request, falling edge active
For TYOUT, start on High output
1:For event counter mode, clocked on falling edge
For pulse period measurement mode, rising edge detection
For CNTR1 interrupt request, rising edge active
For TYOUT, start on Low output

TYM7 Timer Y Stop Bit - Bit 7
0:Count start
1:Count stop

Figure 2-46. TYM Register

2.9.2.1 Read and Write Method

Read and write operations on the high and low-order bytes of Timer Y must be performed in a specific order.

Write Method

When writing to the timer, the lower order byte is written first. This data is placed in a temporary register that is assigned the same address as Timer YL. Next, the high-order byte is written. Then, the data is placed in the Timer YH reload latch and the low-order byte is transferred from its temporary register to the Timer YL reload latch. At this point, if the Timer Y Data Write Control Bit (TYM0) (bit 0) is Low, the value in the Timer Y reload latch is also loaded in Timer Y. If TYM0 is High, the data in the Timer Y reload latch is loaded in Timer Y after Timer Y underflows.

Read Method

When reading Timer Y, the high-order byte is read first. Reading the high-order byte causes the values of Timer YH and Timer YL to be placed in temporary registers that are assigned the same addresses as Timer YH and Timer YL. The low-order byte of Timer Y is then read from its temporary register. This operation assures the correct reading of Timer Y while it is counting.

2.9.2.2 Count Stop Control

If the Timer Y Count Stop Bit (TYM7) (bit 7) is set High, Timer Y stops counting in all four modes.

2.9.2.3 Timer Mode

Count Source: Φ/*n* (where *n* is 8, 16, 32, or 64)

In this mode, each time the timer underflows, the corresponding timer interrupt request bit is set High, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

In Timer mode, the signal TYOUT can also be brought out on the CNTR1 pin. This is controlled by TYM1 (bit1).

Each time the Timer Y underflows, the output of the CNTR1 pin is inverted, and the corresponding Timer Y interrupt request bit is set High. The repeated inversion of the CNTR1 pin output produces a rectangular waveform with a duty ratio of 50 percent. The initial level of the output is determined by the CNTR1 polarity select bit (bit 6). When this bit is Low, the output starts from a High level. When this bit is High, the output starts from a Low level.

2.9.2.4 Pulse Period Measurement Mode

Count Source: Φ/*n* (where *n* is 8, 16, 32, or 64).

This mode measures the period of the event waveform input to the CNTR1 pin.

CNTR1 Polarity Select Bit (TYM6) = zero
When the falling edge of an event waveform is detected on the CNTR1 pin, the contents of Timer Y are stored in the temporary register that is assigned the same address as Timer Y. Simultaneously, the value in the Timer Y reload latch is transferred to Timer Y, and Timer Y continues counting down. The falling edge of an event waveform also causes the CNTR1 interrupt request; therefore, the period of the event waveform from falling edge to falling edge is found by reading Timer Y in the CNTR1 interrupt routine. The data read from Timer Y is the data previously stored in its temporary register.

CNTR1 Polarity Select Bit (TYM6) = one
When the rising edge of an event waveform is detected on the CNTR1 pin, the contents of Timer Y are stored in the temporary register that is assigned the same address as Timer Y. Simultaneously, the value in the Timer Y reload latch is transferred to Timer Y, and Timer Y continues counting down.

The rising edge of an event waveform also causes the CNTR1 interrupt request; therefore, the period of the event waveform from rising edge to rising edge is found by reading Timer Y in the CNTR1 interrupt routine. The data read from Timer Y is the data previously stored in its temporary register.

Each time the timer underflows, the Timer Y interrupt request bit is set High, the contents of the timer reload latch are loaded into the timer, and the countdown sequence begins again.

2.9.2.5 Event Counter Mode

Count Source: CNTR1

Timer countdown is triggered by input to the CNTR1 pin. Each time a timer underflows, the corresponding timer interrupt request bit is set High, the contents of the timer reload latch are loaded into the timer, and the countdown sequence begins again.

The edge used to clock Timer Y is determined by the CNTR1 polarity select bit (bit 6). When these bits are "0"s, the timers are clocked on the rising edge. When these bits are "1"s, the timers are clocked on the falling edge

2.9.2.6 HL Pulse-width Measurement Mode

Count Source: Φ/*n* (where *n* is 8, 16, 32, or 64).

This mode continuously measures both the logical High pulse width and the logical Low pulse width of an event waveform input to the CNTR1 pin. When the falling (or rising) edge of the event waveform is detected on the CNTR1 pin, the contents of Timer Y are stored in the temporary register that is assigned the same address as Timer Y, regardless of the setting of the CNTR1 polarity select bit. Simultaneously, the value in the Timer Y reload latch is transferred to Timer Y, which continues counting down. The falling or rising edge of an event waveform causes the CNTR1 interrupt request; therefore, the width of the event waveform from the falling or rising edge to rising or falling edge is found by reading Timer Y in the CNTR1 interrupt routine. The data read from Timer Y is the data previously stored in its temporary register.

Each time the timer underflows, the Timer Y interrupt request bit is set High, the contents of the timer reload latch are loaded into the timer, and the countdown sequence begins again.

2.9.3 Timer 1

MSB 7	T123M7	T123M6	T123M5	T123M4	T123M3	T123M2	T123M1	T123M0	LSB 0

Address: 0029_{16}
Access: R/W
Reset: 00_{16}

T123M0 TOUT Source Selection Bit - Bit 0
0:TOUT = Timer 1 output.
1:TOUT = Timer 2 output.

T123M1 Timer 1 Stop Bit - Bit 1
0:Timer running
1:Timer stopped

T123M2 Timer 1 Count Source Select Bit - Bit 2
0:Φ divided by 8
1:XCin divided by 2

T123M3 Timer 2 Count Source Select Bit - Bit 3
0:Timer 1 underflow signal
1:Φ

T123M4 Timer 3 Count Source Select Bit - Bit 4
0:Timer 1 underflow signal
1:Φ divided by 8

T123M5 TOUT Output Active Edge Selection Bit - Bit 5
0:Start on High output
1:Start on Low output

T123M6 TOUT Output Control Bit - Bit 6
0:TOUT output disabled
1:TOUT output enabled

T123M7 Timer 1 and 2 Data Write Control Bit - Bit 7
0:Write data in latch and timer
1:Write data in latch only

Figure 2-47. T123M Register

Timer 1 is an 8-bit timer with an 8-bit reload latch and has a pulse output option.

T123M7 of Timer123 mode register (T123M) is the Timer 1 and 2 Data Write Control Bit. If T123M7 is "1", data written to Timer 1 is placed only in the Timer 1 reload latch. The latch value is loaded into Timer 1 after Timer 1 underflows. If T123M7 is "0", the value written to Timer 1 is placed in Timer 1 and the Timer 1 reload latch. At reset, T123M7 is set Low.

The output signal TOUT is controlled by T123M5 and T123M6. T123M5 controls the polarity of TOUT. Setting the bit T123M5 to "1" causes TOUT to start at a Low level, and clearing this bit to "0" causes TOUT to start at a High level. Setting T123M6 to "1" enables TOUT, and clearing T123M6 to "0" disables TOUT.

2.9.3.1 Timer Mode

Count Source: Φ/8 or XCin/2

In Timer mode, each time the timer underflows, the corresponding timer interrupt request bit is set High, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

2.9.3.2 Pulse Output Mode

Count Source: Φ/8 or XCin/2

Timer 1 Pulse Output mode is enabled by setting T123M6 to "1" and T123M0 to a "0". Each time the Timer 1 underflows, the output of the TOUT pin is inverted, and the corresponding Timer 1 interrupt request bit is set High. The repeated inversion of the TOUT pin output produces a rectangular waveform with a duty ratio of 50 percent. The initial level of the output is determined by the TOUT polarity select bit (T123M5). When this bit is Low, the output starts from a High level. When this bit is High, the output starts from a Low level.

2.9.4 Timer 2

Timer 2 is an 8-bit timer with an 8-bit reload latch.

T123M7 (bit 7 of T123M) is the Timer 1 and 2 Data Write Control Bit. If T123M7 is "1", data written to Timer 2 is placed only in the Timer 2 reload latch (see Figure 2-50). The latch value is loaded into Timer 2 after Timer 2 underflows. If the T123M7 is "0", the value written to Timer 2 is placed in Timer 2 and the Timer 2 reload latch. At reset, T123M2 is set Low.

The Timer 2 reload latch value is not affected by a change of the count source. However, because changing the count source may cause an inadvertent countdown of the timer, the timer should be rewritten when the count source is changed.

2.9.4.1 Timer Mode

Count Source: If T123M3 is "0", the Timer 2 count source is the Timer 1 underflow output.

If T123M3 is "1", the Timer 2 count source is Φ.

In Timer mode, each time the timer underflows, the corresponding timer interrupt request bit is set High, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

2.9.4.2 Pulse Output Mode

Count Source: If T123M3 is "0", the Timer 2 count source is the Timer 1 underflow output.

If T123M3 is "1", the Timer 2 count source is Φ.

Timer 2 Pulse Output mode is enabled by setting T123M6 to a "1" and T123M0 to a "1". Each time the Timer 2 underflows, the output of the TOUT pin is inverted, and the corresponding Timer 2 interrupt request bit is set High. The repeated inversion of the TOUT pin output produces a rectangular waveform with a duty ratio of 50 percent. The initial level of the output is determined by the TOUT polarity select bit (T123M5). When this bit is "0", the output starts from a High level. When this bit is "1", the output starts from a Low level.

2.9.5 Timer 3

Timer 3 is an 8-bit timer with an 8-bit reload latch. The Timer 3 reload latch value is not affected by a change of the count source. Because changing the count source may cause an inadvertent countdown of the timer, the timer should be rewritten whenever the count source is changed.

2.9.5.1 Timer Mode

Count Source: If T123M4 is "0", the Timer 3 count source is the Timer 1 underflow output.

If T123M4 is "1", the count source is Φ/8

In Timer mode, each time the timer underflows, the corresponding timer interrupt request bit is set High, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

Data written to Timer 3 is always placed in Timer 3 and the Timer 3 reload latch.

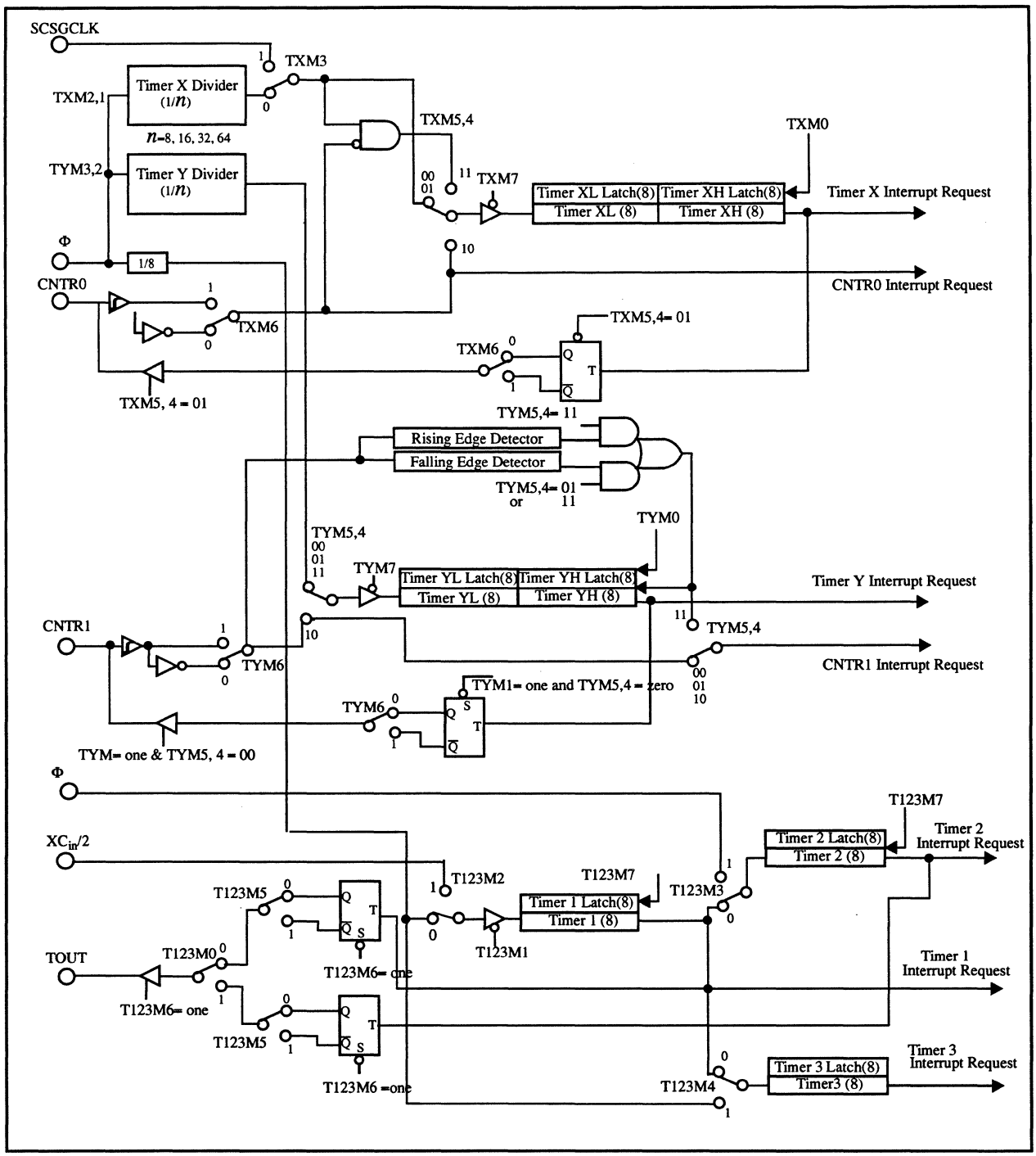


Figure 2-48. Block Diagram of Timers X, Y, 1, 2, and 3

2.10 Universal Serial Bus

Address	Description	Acronym and Value
0050_{16}	USB Address Register	USBA=00
0051_{16}	USB Power Management Register	USBPM=00
0052_{16}	USB Interrupt Status Register 1	USBIS1=00
0053_{16}	USB Interrupt Status Register 2	USBIS2=00
0054_{16}	USB Interrupt Enable Register 1	USBIE1=00
0055_{16}	USB Interrupt Enable Register 2	USBIE2=00
0056_{16}	USB Frame Number Low Register	USBSOFL=00
0057_{16}	USB Frame Number High Register	USBSOFH=00
0058_{16}	USB Endpoint Index	USBINDEX=00
0059_{16}	USB Endpoint x IN CSR	IN_CSR=00
$005A_{16}$	USB Endpoint x OUT CSR	OUT_CSR=00

Address	Description	Acronym and Value
$005B_{16}$	USB Endpoint x IN MAXP	IN_MAXP=00
$005C_{16}$	USB Endpoint x OUT MAXP	OUT_MAXP=00
$005D_{16}$	USB Endpoint x OUT WRT_CNT Low	WRT_CNTL=00
$005E_{16}$	USB Endpoint x OUT WRT_CNT High	WRT_CNTH=00
$005F_{16}$	Reserved	
0060_{16}	USB Endpoint 0 FIFO	USBFIFO0=00
0061_{16}	USB Endpoint 1 FIFO	USBFIFO1=00
0062_{16}	USB Endpoint 2 FIFO	USBFIFO2=00
0063_{16}	USB Endpoint 3 FIFO	USBFIFO3=00
0064_{16}	USB Endpoint 4 FIFO	USBFIFO4=00
0013_{16}	USB Control Register	USBCR=00

The Universal Serial Bus (USB) has the following features:

- Complete USB Specification (version 1.0) Compatibility
- Easy Device Emulation
- Error Handling capabilities:
  - CRC Errors
  - Data Retries
  - Response Time-Out
  - ID Error
- FIFOs:
  - Endpoint 0: IN 16-byte OUT 16-byte
  - Endpoint 1: IN 512-byte OUT 800-byte
  - Endpoint 2: IN 32-byte OUT 32-byte
  - Endpoint 3: IN 16-byte OUT 16-byte
  - Endpoint 4: IN 16-byte OUT 16-byte
  - Total Five Endpoints Available
- Complete Device Configuration
- Supports All Device Commands
- Supports Full-Speed Functions
- User Friendly Interface:
  - Easy to Write Control Software for USB
- Support of All USB Transfer Types:
  - Isochronous
  - Bulk
  - Control
  - Interrupt

- Suspend/Resume Operation
- On-chip USB Transceiver
- Frequency Synthesizer
  - 24 MHz External, 48 MHz Internal for USB core Operation

2.10.1 USB Function Control Unit

The implementation of the USB by this device is accomplished chiefly through the device's USB Function Control Unit. The Function Control Unit's overall purpose is to handle the USB packet protocol layer. The Function Control Unit notifies the MCU that a valid token has been received. When this occurs, the data portion of the token is routed to the appropriate FIFO. The MCU transfers the data to, or from, the host by interacting with that endpoint's FIFO and CSR register. (see Figure 2-49.)

The USB Function Control Unit is composed of five sections:

- Serial Interface Engine (SIE)
- Generic Function Interface (GFI)
- Serial Engine Interface Unit (SIU)
- Microcontroller Interface Unit (MCI)
- USB Transceiver

2.10.1.1 Serial Interface Engine

The SIE interfaces to the USB serial data and handles Deserialization/Serialization of data, NRZI encoding decoding, Clock extraction, CRC generation and checking, Bit Stuffing, and other specifications pertaining to the USB protocol such as handling inter-packet time-outs and PID decoding.

2.10.1.2 Generic Function Interface

The GFI handles the USB protocols for the Control zero endpoint, IN and OUT endpoints. It also handles other details such as the retry mechanisms and the data toggle synchronization. It handles retries for NAK handshake to the host without software intervention from the CPU.

2.10.1.3 Serial Engine Interface Unit

The SIU block decodes the Address and Endpoints from the USB bus.

2.10.1.4 Microcontroller Interface Unit

The MCI block handles the Microcontroller interface and does address decoding and synchronization of control signals.

2.10.1.5 USB Transceiver

The USB transceiver is designed to interface with the physical layer of the USB. It is capable of transmitting and receiving serial data at full speed.

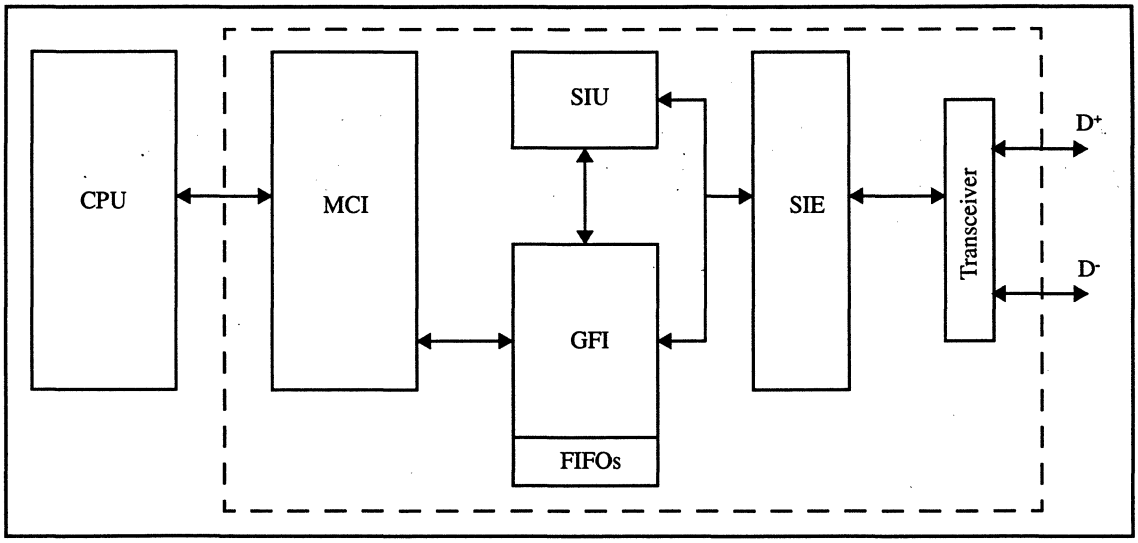


Figure 2-49. USB Function Control Unit Block Diagram

2.10.2 USB Interrupts

There are two types of USB interrupts in this device: the first type is the USB function (including overrun/underrun, reset, suspend and resume) interrupt, used to control the flow of data and USB power control; the second type is start-of-frame (SOF) interrupt, used to monitor the transfer of isochronous data.

2.10.2.1 USB Function Interrupt

The USB function has two interrupt status bits for each endpoint (except endpoint 0, which has one interrupt status bit, and it is set under different conditions than endpoints 1-4) to control the data transfer: the out packet ready (OUT_PKT_RDY) and the in packet ready (IN_PKT_RDY). The EPx_OUT_INT bit is set when GFI sets the OUT_PKT_RDY bit, the EPx_IN_INT bit is set when GFI clears the IN_PKT_RDY bit in the respective CSR register. Each individual USB function interrupt is enabled by setting the corresponding bit in the USB Interrupt Enable Register 1 and 2. The USB Interrupt Status Register 1 and 2 are used to indicate pending interrupts for a given endpoint.

The suspend interrupt status bit is set if a suspend signal is received. If the device is in suspend mode, the resume interrupt status bit is set when a resume signal is received. There is a single interrupt enable bit for both of suspend and resume interrupts (bit 7 of the interrupt enable register 2).

The USB reset interrupt status bit is set if a reset signal is received. When this bit is set, all USB internal registers will be reset to their default value except this bit itself. This bit is cleared by the CPU writing a "0" to it. When the CPU detects a USB reset interrupt, it needs to re-initialize the USB block in order to accept packets from the host.

The Over/Underrun status bit is set (any endpoint used for asynchronous data transfer), when an overrun condition occurred in an Endpoint (the CPU is too slow to unload the data from the FIFO), or when an underrun condition occurred in an Endpoint (the CPU is too slow to load the data to the FIFO).

The USB Function Interrupt (sum of all individual function interrupts) is enabled by setting the corresponding bit in the Interrupt Control Register of the Interrupt Control Unit. (see "2.7 Interrupt Control Unit" for detail).

2.10.2.2 USB SOF Interrupt

The USB SOF (start-of-frame) interrupt is used to control the transfer of isochronous data. This device generates a start-of-frame interrupt whenever a start-of-frame packet is received. The USB SOF interrupt is enabled by setting the corresponding bit in the Interrupt Control Register of the Interrupt Control Unit. (see "2.7 Interrupt Control Unit" for detail).

2.10.3 USB Endpoint FIFOs

This device has a transmit FIFO and a receive FIFO for each endpoint, both FIFOs support up to two separate data sets of variable size, and provide the ability of back-to-back transmission and reception.

2.10.3.1 Transmit FIFOs

The CPU writes data to the endpoint's transmit FIFO location specified by the write pointer, which automatically increments by "1" after a write. When a full packet of data is written to the FIFO, the IN_PKT_RDY bit in the corresponding Endpoint IN Control and Status Register (IN CSR) is set, signaling the USB GFI that it can transmit the data packet in the next IN token (for non-isochronous data transfer). Setting the IN_PKT_RDY bit can be done in either of two ways. If bit 7 of the IN_CSR is a "0", IN_PKT_RDY is set by the CPU writing a "1" to that bit in the IN_CSR. If bit 7 of the IN_CSR is a "1", IN_PKT_RDY is automatically set by the GFI when the number of bytes of data equal to the maximum packet size (MAXP) is written to the FIFO. In this case, if a short packet is going to transmit, the CPU still needs to write a "1" to set IN_PKT_RDY bit. Clearing the IN_PKT_RDY is done by the GFI. In the case where the FIFO is configured to hold a single data set (MAXP > half of the FIFO size), the IN_PKT_RDY is cleared upon successful transmission of the data packet. In the case where the FIFO is configured to hold two data sets (MAXP <= half of the FIFO size), every time there is one or two data sets available, the IN_PKT_RDY is cleared. In the later case, in order to determine if a data packet has been transmitted out, a read only bit, TX_NOT_EMPTY is available. The status of a TX FIFO status could be obtained by:

IN_PKT_RDY	TX_NOT_EMPTY	TX FIFO Status
0	0	No data packet in TX FIFO
0	1	One data packet in TX FIFO
1	0	Invalid
1	1	Two data packets in TX FIFO

For isochronous data transfer, an additional bit, named ISO_UPDATE is provided. The function of this bit is described below:

If ISO_UPDATE = "0", whenever the IN_PKT_RDY is set, it is sent to the Serial Interface Engine (SIE) immediately, available for the next IN_TOKEN from the host; if ISO_UPDATE = "1" and ISO bit of the corresponding endpoint's IN CSR is set, then the IN_PKT_RDY bit is delayed until the next SOF before being sent to the SIE. In this way the IN_PKT_RDY is synchronized with the SOF. The ISO_UPDATE bit is a global bit for endpoints 1 to 4, and works with isochronous pipe only.

Endpoints 2 to 4 (IN direction) can be used for interrupt transfer. This is done by setting the INTPT bit in the IN CSR register of the corresponding endpoint. The CPU sets this bit to initialize that endpoint as a status change endpoint, meaning that whenever the USB GFI receives an IN token addressed to that endpoint, the GFI will unconditionally transmit out the most recent updated data packet in the TX FIFO which the size is equal to the MAXP set for that endpoint. In order to avoid transmitting some unwanted data to the host while the CPU is in initialization stage, the following setup sequence is recommended:

1. Set MAXP;
2. Load interrupt status information with the number of bytes equal to MAXP to that endpoint's FIFO;
3. Set IN_PKT_RDY and INTPT bits of the IN CSR;
4. Load the FIFO with the number of bytes equal to MAXP for all the subsequent interrupt status updates.

Because all endpoints start up default to be bulk transfer, before Step 3 listed above is done, the GFI will send NAK to every IN token the host addresses to this endpoint, therefore unwanted data will not be transmitted out to the host.

The USB function control unit handles all the bad transmission retry and data set management tasks.

2.10.3.2 Receive FIFOs

The USB GFI writes data to the endpoint's receive FIFO location specified by the write pointer, which automatically increments by one after a write. When a good reception is complete, the GFI sets the OUT_PKT_RDY bit in the corresponding Endpoint OUT Control and Status Register (OUT CSR), signaling the CPU that it has successfully received a data packet. This bit is cleared upon unloading the packet from the FIFO. Clearing the OUT_PKT_RDY bit can be done in two ways. If bit 7 of the OUT_CSR is a "0", OUT_PKT_RDY is cleared by the CPU writing a "0" to that bit. If bit 7 of the OUT_CSR is a "1", OUT_PKT_RDY is automatically cleared by the GFI when the number of bytes of data equal to the value of the Write_Count register (WRT_CNT) is unloaded from the FIFO. In the case where the FIFO is configured to hold two data sets, every time there is one or two data sets available, the OUT_PKT_RDY bit is set.

The USB function control unit handles all the write pointer reversal for a bad reception and data set management tasks.

For endpoint 1, when transferring ISO data, the receive FIFO can be unloaded as soon as there is some data in the FIFO, which is indicated by DATA_NOT_EMPTY signal. The completion of unloading a packet of data is signified by OUT_PKT_RDY = one, and DATA_SET_NOT_EMPTY = zero. This feature is available only in DMA transfer mode, and is tightly coupled with the DMA controller, see the DMA section for detail.

2.10.4 USB Special Function Registers

This device controls the USB operation through the use of special function registers (SFR). This section describes in detail each USB related SFR. Certain USB SFRs are endpoint-indexed. Those are the Control & Status Registers (IN CSR and OUT CSR), the Maximum Packet Size Registers (IN MAXP and OUT MAXP) and the Write Count Registers (OUT WRT_CNT). To access each of the endpoint-indexed SFR, the target endpoint number should be written to the Endpoint Index Register first. The lower 3 bits (EPINDEX2:0) of the Endpoint Index Register are used for endpoint selection.

Note: Each endpoint's FIFO Register is NOT endpoint-indexed.

MSB 7								LSB 0
USBC7	USBC6	USBC5	USBC4	Reserved	Reserved	Reserved	Reserved	

Address: 0013_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
3:0	-	Reserved, write "0" to these bits.	R
4	USBC4	USB Line Driver Supply Select 0: Pin #72 is the USB line driver power supply pin. This should connect to a 3.3V supply. 1: Pin #72 is the USB line driver external capacitor pin. This should connect to a 1µf or larger capacitor between pin #72 and Vss for 5V only power supply applications.	W/R
5	USBC5	USB Clock Enable 0: 48 MHz clock to the USB block is disabled. 1: 48 MHz clock to the USB block is enabled.	W/R
6	USBC6	USB SOF Port Select 0: USB SOF output is disabled. P70 is used as GPIO pin. 1: USB SOF output is enabled.	W/R
7	USBC7	USB Enable 0: USB block is disabled, all USB internal registers are held at their default values. 1: USB block is enabled.	W/R

Figure 2-50. USB Control Register

MSB 7								LSB 0
Reserved	FUN_ADDR6	FUN_ADDR5	FUN_ADDR4	FUN_ADDR3	FUN_ADDR2	FUN_ADDR1	FUN_ADDR0	

Address: 0050_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
6:0	FUN_ADDR6:0	7-bit programmable Function Address	W/R
7	-	Reserved, write a "0" to this bit	R

Figure 2-51. Function Address Register

The Function Address register maintains the USB address assigned by the host. The function control unit uses this register value to decode USB token packet address. At reset, the device is not yet configured, the value is 00_{16}.

MSB 7								LSB 0
Reserved	Reserved	Reserved	Reserved	Reserved	S_RESUME	R_RESUME	SUSPEND	

Address: 0051_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
0	SUSPEND	When the function control unit receives suspend signaling, it sets this bit and suspends its clocks. This also generates an interrupt. Upon seeing this bit set, the CPU can save its internal register and then enter suspend mode, independent of the function control unit. The CPU clears this bit when any of the resume bit is set.	W/R
1	R_RESUME	When the function control unit is in suspend mode and receives resume signaling, its clocks are enabled, and this bit is set, An interrupt is also generated when this bit is set. Upon seeing this bit set, the CPU can start its wake up sequence. The CPU should clear this bit after the wake up sequence is done.	W/R
2	S_RESUME	When the CPU is waken up (by peripheral or timer), it starts its wake up sequence and set this bit. While this bit is set, and the function control unit was in suspend mode, it will generate resume signal to the host. The CPU should keep this bit set for a minimum of 10ms and a maximum of 15ms.	W/R
7:3	-	Reserved, write "0" to these bits	R

Figure 2-52. Power Management Register

The Power Management register is used for power management in the function control unit.

MSB 7								LSB 0
EP3_OUT_INT	EP3_IN_INT	EP2_OUT_INT	EP2_IN_INT	EP1_OUT_INT	EP1_IN_INT	Reserved	ENDPT0_INT	

Address: 0052_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
0	ENDPT0_INT	GFI sets this bit upon: 1. Setting OUT_PKT_RDY bit of the ENDPOINT 0 IN CSR register 2. Clearing IN_PKT_RDY bit of the ENDPOINT 0 IN CSR register 3. Setting SETUP_END bit of the ENDPOINT 0 IN CSR register 4. Clearing DATA_END bit of the ENDPOINT 0 IN CSR register 5. Setting FORCE_STALL bit of the ENDPOINT 0 IN CSR register	R/W
1	-	Reserved	R/W
2	EP1_IN_INT	GFI sets this bit upon clearing IN_PKT_RDY bit of the ENDPOINT 1 IN CSR register	R/W
3	EP1_OUT_INT	GFI sets this bit upon setting OUT_PKT_RDY bit of the ENDPOINT 1 OUT CSR register	R/W
4	EP2_IN_INT	GFI sets this bit upon clearing IN_PKT_RDY bit of the ENDPOINT 2 IN CSR register	R/W
5	EP2_OUT_INT	GFI sets this bit upon setting OUT_PKT_RDY bit of the ENDPOINT 2 OUT CSR register	R/W
6	EP3_IN_INT	GFI sets this bit upon clearing IN_PKT_RDY bit of the ENDPOINT 3 IN CSR register	R/W
7	EP3_OUT_INT	GFI sets this bit upon setting OUT_PKT_RDY bit of the ENDPOINT 3 OUT CSR register	R/W

Figure 2-53. Interrupt Status Register 1

The Interrupt Status Register 1 is used to indicate what condition caused an interrupt to the CPU. The CPU writes a "1" to clear the corresponding interrupt status bit.

MSB 7								LSB 0
SUSPEND_INT	RESUME_INT	USB_RST_INT	OVER/UNDER	Reserved	Reserved	EP4_OUT_INT	EP4_IN_INT	

Address: 0053_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
0	EP4_IN_INT	GFI sets this bit upon clearing IN_PKT_RDY bit of the ENDPOINT 0 IN CSR register	R/W
1	EP4_OUT_INT	GFI sets bit upon setting OUT_PKT_RDY bit of the ENDPOINT 0 OUT CSR register	R/W
3:2	-	Reserved	R/W
4	OVER/UNDER	GFI sets this bit if an overrun or underrun condition occurred in any of the isochronous endpoints	R/W
5	USB_RST_INT	GIF sets this bit upon receiving a reset signal from the host. All USB internal registers will be reset to their default values except this bit. This bit is cleared by the CPU writing a "0" to it. An interrupt will also generate if the corresponding enable bit is set.	R/W
6	RESUME_INT	GFI sets this bit upon receiving resume signaling from the host.	R/W
7	SUSPEND_INT	GFI sets this bit upon receiving suspend signaling from the host.	R/W

Figure 2-54. Interrupt Status Register 2

The Interrupt Status Register 2 register is used to indicate what condition caused an interrupt to the CPU. The CPU writes a "1" to clear the corresponding interrupt status bit.

MSB 7								LSB 0
E_EP3_O_INT	E_EP3_I_INT	E_EP2_O_INT	E_EP2_I_INT	E_EP1_O_INT	E_EP1_I_INT	Reserved	E_EP0_I_INT	

Address: 0054_{16}
Reset: FF_{16}

Bit	Name	Description	CPU
0	E_EP0_I_INT	Endpoint 0 IN interrupt is enabled when this bit is set High	W/R
1	-	Reserved	R
2	E_EP1_I_INT	Endpoint 1 IN interrupt is enabled when this bit is set High	W/R
3	E_EP1_O_INT	Endpoint 1 OUT interrupt is enabled when this bit is set High	W/R
4	E_EP2_I_INT	Endpoint 2 IN interrupt is enabled when this bit is set High	W/R
5	E_EP2_O_INT	Endpoint 2 OUT interrupt is enabled when this bit is set High	W/R
6	E_EP3_I_INT	Endpoint 3 IN interrupt is enabled when this bit is set High	W/R
7	E_EP3_O_INT	Endpoint 3 OUT interrupt is enabled when this bit is set High	W/R

Figure 2-55. Interrupt Enable Register 1

The Interrupt Enable Register 1 register is an interrupt enable register, if the corresponding bit is "0", the respective interrupt is disabled. Upon reset, all the interrupts are disabled.

MSB 7								LSB 0
E_SUS_INT	Reserved	E_RST_INT	E_OU_INT	Reserved	Reserved	E_EP0_O_INT	E_EP4_I_INT	

Address: 0055_{16}
Reset: 33_{16}

Bit	Name	Description	CPU
0	E_EP4_I_INT	Endpoint 4 IN interrupt is enabled when this bit is set High	W/R
1	E_EP4_O_INT	Endpoint 4 OUT interrupt is enabled when this bit is set High	W/R
3:2	-	Reserved	R
4	E_OU_INT	Overrun/Underrun interrupt is enabled when this bit is set High	W/R
5	E_RST_INT	USB Reset interrupt is enabled when this bit is set High	W/R
6	-	Reserved	R
7	E_SUS_INT	Suspend and Resume interrupts are enabled when this bit is set High	W/R

Figure 2-56. Interrupt Enable Register 2

The Interrupt Mask Register 2 is an interrupt enable register, if the corresponding bit is "0", the respective interrupt is disabled. Upon reset, all the interrupts are enabled except bit 7 - suspend and resume interrupt is disabled.

MSB 7								LSB 0
FN7	FN6	FN5	FN4	FN3	FN2	FN1	FN0	

Address: 0056_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
7:0	RN7:0	Lower 8 bits of the 11-bit frame number issued with a SOF token	R

Figure 2-57. Frame Number Register Low

The frame number low register contains the lower 8 bits of the 11-bit time stamp received from the host.

MSB 7								LSB 0
Reserved	Reserved	Reserved	Reserved	Reserved	FN10	FN9	FN8	

Address: 0057_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
2:0	FN10:8	Upper 3 bits of the 11-bit frame number issued with a SOF token	R
7:3	-	Reserved	R

Figure 2-58. Frame Number Register High

The frame number high register contains the upper 3 bits of the 11-bit time stamp received from the host.

MSB 7								LSB 0
ISO_UPDATE	Reserved	Reserved	Reserved	Reserved	EPINDEX2	EPINDEX1	EPINDEX0	

Address: 0058_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
2:0	EPINDEX2:0	Endpoint Index: 000 Function Endpoint 0 001 Function Endpoint 1 010 Function Endpoint 2 011 Function Endpoint 3 100 Function Endpoint 4 Others: Undefined	W/R
6:3	-	Reserved	R
7	ISO_UPDATE	If ISO_UPDATE = zero, whenever the IN_PKT_RDY bit is set, it is send to the Serial Interface Engine (SIE) immediately, available for the next IN_TOKEN from the host; if IOS_UPDATE = one and ISO bit of the corresponding endpoint's IN CSR is set, the in IN_PKT_RDY bit is delayed until the next SOF before being sent to the SIE. In this way, the IN_PKT_RDY is synchronized with the SOF. The ISO_UP_DATE bit is a global bit for endpoints 1 to 4, and works with isochronous pipe only.	W/R

Figure 2-59. Endpoint Index Register

The Endpoint Index Register identifies the endpoint pair. Its contents select the transmit and receive FIFO pair and serve as an index to endpoint-specific IN_CSR, OUT_CSR, IN_MAXP, OUTMAXP and OUT_WRT_CNT registers. This register also contains a global bit, ISO_UPDATE, for endpoints 1-4, to specify the way IN_PKT_RDY is sent by the SIE.

MSB 7								LSB 0
SERVICED SETUP_END	SERVICED OUT_PKT_RDY	SETUP_END	SENT_STALL	DATA_END	FORCE_STALL	IN_PKT_RDY	OUT_PKT_RDY	

Address: 0059_{16}
Reset: 00_{16}

Bit	Name	Description	CPU
0	OUT_PKT_RDY	GFI sets this bit upon receiving a valid SETUP/OUT token from the host. CPU clears this bit after unloading the FIFO, by way of writing a "1" to bit 6 of this register.	R
1	IN_PKT_RDY	CPU sets this bit upon writing a packet of data to endpoint 0 FIFO. GFI clears this bit upon successful transmission of IN packet.	W/R
2	SEND_STALL	CPU writes a "1" to this bit if it decodes an invalid SETUP token. GFI clears this bit upon sending a STALL handshake.	W/R
3	DATA_END	CPU sets this bit when it writes (IN data phase) or reads (OUT data phase) the last packet of data from/to the FIFO. This bit indicates to the GFI that the specific amount of data in the setup phase is transferred. The GFI will advance to status phase once this bit is set. When the status phase completes, the GFI clears this bit.	W/R
4	FORCE_STALL	GFI sets this bit if it detects a protocol violation, sends a STALL handshake, and also causes an interrupt. The CPU writes a "0" to clear this bit.	W/R
5	SETUP_END	GFI sets this bit if a control transfer has ended before the specific length of data is transferred during the data phase (for instance, if it gets a new setup phase or a status phase before data end is set). CPU clears this bit, after returning to idle state by way of writing a "1" to bit 7 of this register. Once the CPU sees SETUP_END bit set, it will stop access the FIFO to service the previous setup transaction. If OUT_PKT_RDY is set at the same time SETUP_END is set, it indicates the previous setup transaction ended, and new SETUP token is in the FIFO.	R
6	SERVICED_OUT_PKT_RDY	CPU writes a "1" to this bit to clear OUT_PKT_RDY bit (bit 0).	W
7	SERVICED_SETUP_END	CPU writes a "1" to this bit to clear SETUP_END bit (bit 5).	W

Figure 2-60. Endpoint 0 IN CSR

The Endpoint 0 IN register contains the control and status information of the respective endpoint. (Endpoint 0 uses this register for all control and status Information).

MSB 7								LSB 0	Address: 0059_{16} Reset: 00_{16}
AUTO_SET	FLUSH	TX_NOT_EPT	INTPT	ISO	FORCE_STALL	UNDER_RUN	IN_PKT_RDY		

Bit	Name	Description	CPU
0	IN_PKT_RDY	This bit is set when a packet of data is written to the FIFO. If bit 7 (AUTO_SET) is a "0", the IN_PKT_RDY bit is set by the CPU after a packet of data is loaded into the FIFO; If AUTO_SET bit is a "1", the IN_PKT_RDY is set automatically by the GFI after the number of bytes of data equal to the maximum packet size (MAXP) is written into the FIFO. This bit is cleared by the GFI. (see 'Transmit FIFOs' description for detail)	W/R
1	UNDER_RUN	This bit is used in ISO mode only to indicate to the CPU that FIFO underrun occurred. GFI sets this bit at the beginning of a IN token if IN_PKT_RDY bit is not set. Upon setting this bit, the OVER_UNDER bit of the Interrupt Status Register 2 will be set. This bit is cleared by the CPU writing a "0" to this bit.	W/R
2	SEND_STALL	CPU writes a "1" to this bit when it encounters a STALL condition. The GFI sends a STALL handshake when this bit is set. the CPU writes a "0" to clear this bit after the STALL condition is removed.	W/R
3	ISO	CPU sets this bit to initialize the respective endpoint as an Isochronous endpoint for IN transaction.	W/R
4	INTPT	CPU sets this bit to initialize this endpoint as a status change endpoint for IN transaction. This bit should be set if the corresponding endpoint is to be used as an interrupt endpoint. (see 'Transmit FIFOs' description for detail)	W/R
5	TX_NOT_EPT	Transmit FIFO not empty, the GFI sets this when there is data in the TX FIFO. This bit in conjunction with IN_PKT_RDY bit will provide the transmit FIFO status information (see 'Transmit FIFOs' description for detail)	R
6	FLUSH	CPU sets this bit to flush the FIFO	W
7	AUTO_SET	If this bit is a "1", the IN_PKT_RDY is set automatically by the GFI after the number of bytes of data equal to the maximum packet size (MAXP) is written into the FIFO.	W/R

Figure 2-61. Endpoints 1, 2, 3, 4 IN CSR

This register contains the control and status information of the respective IN endpoint. (For Endpoints 1, 2, 3, 4)

MSB 7								LSB 0	Address: $005A_{16}$ Reset: 00_{16}
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved		

Bit	Name	Description	CPU
7:0	-	Reserved	R

Figure 2-62. Endpoint 0 OUT CSR

For Endpoint 0, all bits in this register are reserved (all the control and status information is in Endpoint 0 IN CSR)

MSB 7								LSB 0	Address: $005A_{16}$ Reset: 00_{16}
AUTO_CLR	FLUSH	DATA_ERR	SENT_STALL	ISO	FORCE_STALL	OVER_RUN	OUT_PKT_RDY		

Bit	Name	Description	CPU
0	OUT_PKT_RDY	GFI sets this bit after it has written a packet of data to the FIFO. This bit is cleared when a packet of data is unloaded from the FIFO. If bit 7 (AUTO_CLR) is a "0", the OUT_PKT_RDY bit is cleared by the CPU after a packet of data is unloaded from the FIFO; If AUTO_CLR bit is a "1", the OUT_PKT_RDY is cleared automatically by the GFI after the packet of data in unloaded from the FIFO.	W/R
1	OVER_RUN	This bit is used in ISO mode only to indicate to the CPU that FIFO overrun occurred. If the GFI receives a OUT ISO token while OUT_PKT_RDY bit is not cleared, then it discards the data from the host, and sets this bit. This bit is cleared by the CPU writing a "0" to this bit.	W/R
2	SEND_STALL	CPU sets this bit when it encounters a STALL condition. CPU writes a "0" to clear this bit after the STALL condition is removed.	W/R
3	ISO	CPU sets this bit to initialize the respective endpoint as an Isochronous endpoint for OUT transaction.	W/R
4	FORCE_STALL	GFI sets this bit if it detects a protocol violation, and sends a STALL handshake. CPU writes a "0" to clear this bit after the STALL condition is removed.	R
5	DATA_ERR	GFI sets this bit to indicate a CRC error or bit stuffing error received in an ISO packet The CPU writes a "0" to clear this bit.	W/R
6	FLUSH	CPU sets this bit to flush the FIFO	W
7	AUTO_CLR	If this bit is a "1", the OUT_PKT_RDY is cleared automatically by the GFI after the packet of data in unloaded from the FIFO.	W/R

Figure 2-63. Endpoint 1, 2, 3, 4 OUT CSR

This register contains the control and status information of the respective endpoint. (For Endpoints 1, 2, 3, 4)

MSB 7								LSB 0	Address: $005B_{16}$
MAXP7	MAXP6	MAXP5	MAXP4	MAXP3	MAXP2	MAXP1	MAXP0		Reset: 00_{16}

Bit	Name	Description	CPU
7:0	MAXP7:0	MAXP = n for endpoints 0, 2, 3, 4 MAXP = n * 8 for endpoint 1 Where n is the value written to this register. For endpoints that support smaller FIFO size, unused bits are not implemented (always write "0" to those bits)	W/R

Figure 2-64. Endpoint x IN MAXP

This register indicates the maximum packet size (MAXP) of Endpoint x IN packet. The default value for endpoint 0 is 8-byte, the default values for endpoints 1-4 are 0-byte. The CPU can change this value, as negotiated with the host controller through the SET_DESCRIPTOR command.

MSB 7								LSB 0	Address: $005C_{16}$
MAXP7	MAXP6	MAXP5	MAXP4	MAXP3	MAXP2	MAXP1	MAXP0		Reset: 00_{16}

Bit	Name	Description	CPU
7:0	MAXP7:0	MAXP = n for endpoints 2, 3, 4 MAXP = n * 8 for endpoint 1 where n is the value written to this register. For endpoints that support smaller FIFO size, unused bits are not implemented (always write "0" to those bits)	W/R

Figure 2-65. Endpoint x OUT MAXP

This register indicates the maximum packet size (MAXP) of Endpoint x OUT packet. The default values for endpoints 1-4 are 0-byte. The CPU can change this value, as negotiated with the host controller through the SET_DESCRIPTOR command.

For endpoint 0, all bits in this register are reserved; endpoint 0 uses IN MAXP register for both IN and OUT transfers.

MSB 7								LSB 0	Address: $005D_{16}$
WRT_CNT7	WRT_CNT6	WRT_CNT5	WRT_CNT4	WRT_CNT3	WRT_CNT2	WRT_CNT1	WRT_CNT0		Reset: 00_{16}

Bit	Name	Description	CPU
7:0	WRT_CNT7:0	Byte Count. This register contains the lower 8 bits of the byte count register.	R

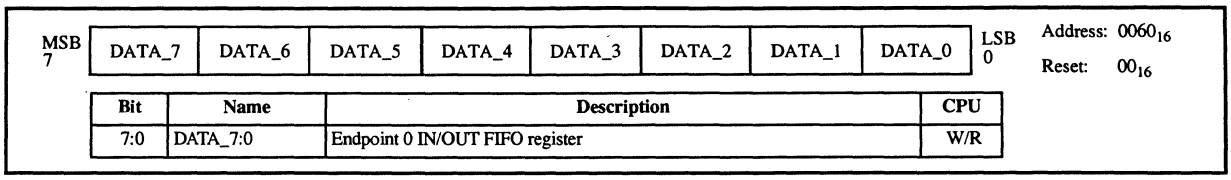
Figure 2-66. Endpoint 0, 1, 2, 3, 4 OUT Write Count Register Low

MSB 7								LSB 0	Address: $005E_{16}$
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	WRT_CNT9	WRT_CNT8		Reset: 00_{16}

Bit	Name	Description	CPU
1:0	WRT_CNT9:8	Byte Count. This register contains the upper 2 bits of the byte count register. The GFI sets the value in these two Write Count Registers after received end-of-packet (EOP) signal from the host. The CPU reads these two registers to determine the number of bytes to be read from the FIFO. The CPU should read Count Register Low first then Register High.	R
7:2	-	Reserved.	R

Figure 2-67. Endpoint 0, 1, 2, 3, 4 OUT Write Count Register High

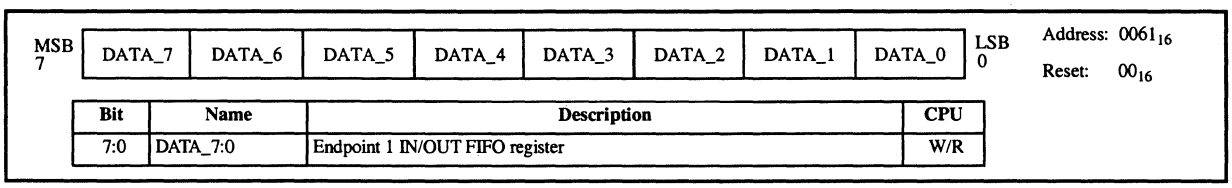
These two registers contains the number of bytes in Endpoint x OUT FIFO.



Bit	Name	Description	CPU
7:0	DATA_7:0	Endpoint 0 IN/OUT FIFO register	W/R

Figure 2-68. Endpoint 0 FIFO Register

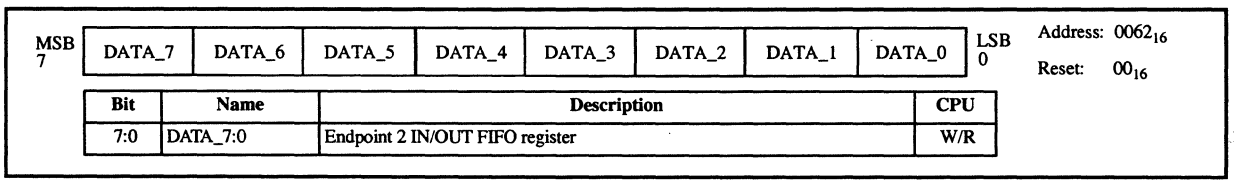
This register is the USB transmit (IN) and receive (OUT) FIFO data register. The CPU writes data to this register for Endpoint 0 IN FIFO. The CPU reads data from this register for Endpoint 0 OUT FIFO.



Bit	Name	Description	CPU
7:0	DATA_7:0	Endpoint 1 IN/OUT FIFO register	W/R

Figure 2-69. Endpoint 1 FIFO Register

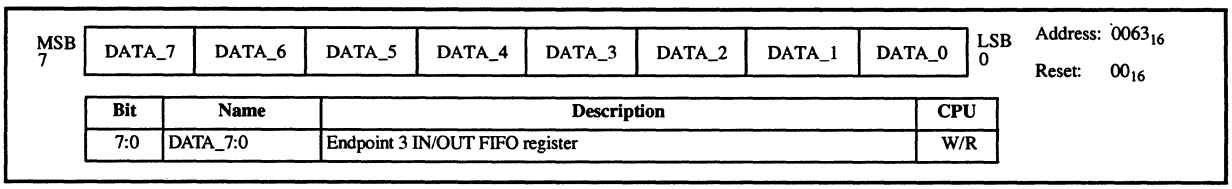
This register is the USB transmit (IN) and receive (OUT) FIFO data register. The CPU writes data to this register for Endpoint 1 IN FIFO. The CPU reads data from this register for Endpoint 1 OUT FIFO.



Bit	Name	Description	CPU
7:0	DATA_7:0	Endpoint 2 IN/OUT FIFO register	W/R

Figure 2-70. Endpoint 2 FIFO Register

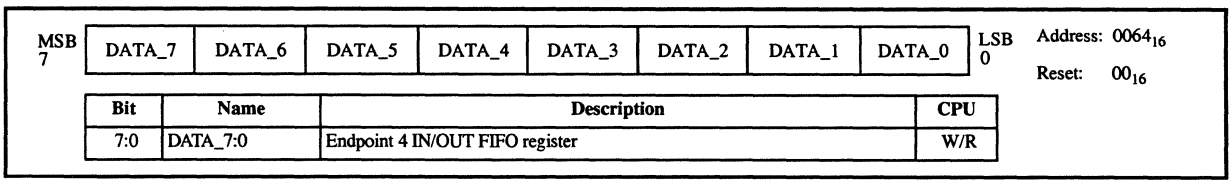
This register is the USB transmit (IN) and receive (OUT) FIFO data register. The CPU writes data to this register for Endpoint 2 IN FIFO. The CPU reads data from this register for Endpoint 2 OUT FIFO.



Bit	Name	Description	CPU
7:0	DATA_7:0	Endpoint 3 IN/OUT FIFO register	W/R

Figure 2-71. Endpoint 3 FIFO Register

This register is the USB transmit (IN) and receive (OUT) FIFO data register. The CPU writes data to this register for Endpoint 3 IN FIFO. The CPU reads data from this register for Endpoint 3 OUT FIFO.



Bit	Name	Description	CPU
7:0	DATA_7:0	Endpoint 4 IN/OUT FIFO register	W/R

Figure 2-72. Endpoint 4 FIFO Register

This register is the USB transmit (IN) and receive (OUT) FIFO data register. The CPU writes data to this register for Endpoint 4 IN FIFO. The CPU reads data from this register for Endpoint 4 OUT FIFO.

2.11 Master CPU Bus Interface

Address	Description	Code
0048_{16}	Data bus buffer register 0	DBB0=00
0049_{16}	Data bus buffer status register 0	DBBS0=00
$004A_{16}$	Data bus buffer control register 0	DBBC0=00
$004C_{16}$	Data bus buffer register 1	DBB1=00
$004D_{16}$	Data bus buffer status register 1	DBBS1=00
$004E_{16}$	Data bus buffer control register 1	DBBC1=00

Pin	Description
P60/P67	are multiplexed with DQ0-DQ7
P51	is multiplexed with OBF_0
P52	is multiplexed with $\overline{IBF}_0$
P53	is multiplexed with $\overline{S}_0$
P54	is multiplexed with A_0

Pin	Description
P55	is multiplexed with $\overline{R}$ or E
P56	is multiplexed with $\overline{W}$ or R/$\overline{W}$
P72	is multiplexed with $\overline{S}_1$
P73	is multiplexed with $\overline{IBF}_1$
P77	is multiplexed with OBF_1

This device has a 2-bit internal bus interface function that can be operated in slave mode by control signals from the master CPU (see Figure 2-73. Bus Interface Circuit). The bus interface can be connected directly to either a R/$\overline{W}$ type of CPU or a CPU with $\overline{RD}$ and $\overline{WR}$ separate signals. Slave mode is selected with the bit 7 of the data buffer control register 0. The single data bus buffer mode and the double data bus buffer mode are selected with the bit 7 of the data bus buffer control register 1. When selecting the double data bus buffer mode. Port P72 becomes $\overline{S}_1$ input.

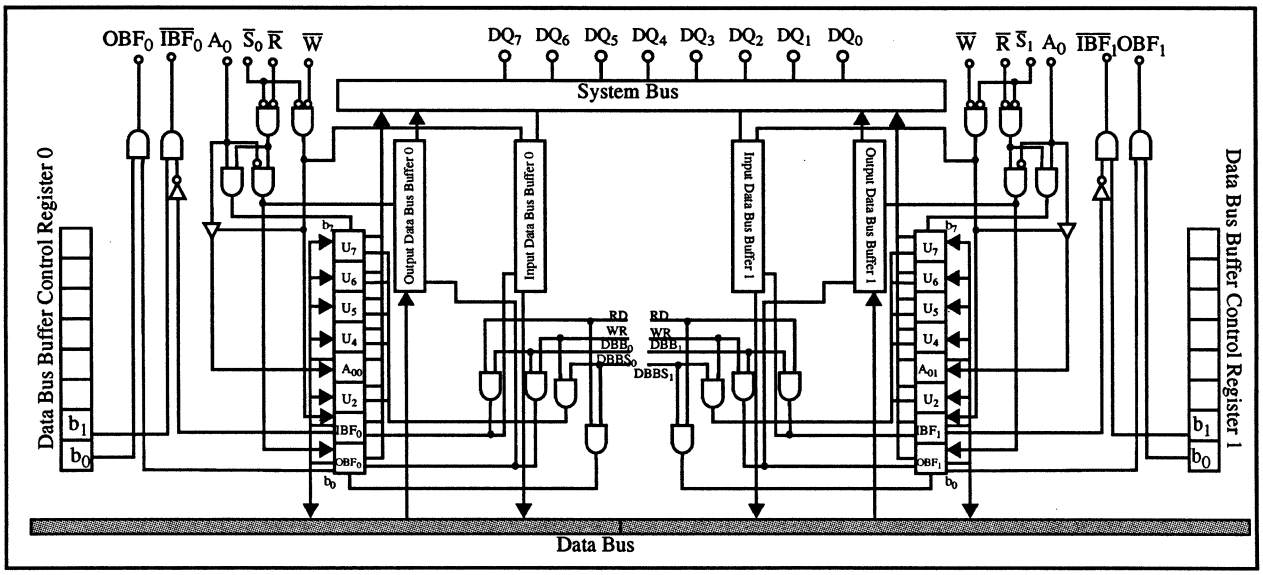


Figure 2-73. Bus Interface Circuit

When data is written to the MCU from the master CPU, an input buffer full interrupt request occurs. Similarly, when data is read from the master CPU, an output buffer empty interrupt request occurs.

When the bus interface is operating, DQ_0-DQ_7 become a 3-state data bus that sends and receives data, command, and status to and from the master CPU. At the same time, $\overline{W}$, $\overline{R}$, $\overline{S}_0$, $\overline{S}_1$, and A_0 become host CPU control signal input pins.

Two input buffer full interrupt requests and two output buffer full requests are used as shown in Figure 2-74..

The bus interface can be operated under normal MCU control or under on-chip DMA control for fast data transfer. If a master CPU has a large amount of data to be transferred, use of the on-chip DMA controller is highly recommended.

The bus interface signal input level can be programmed as CMOS level (default) or as TTL level. Bit 7 of the Port Control Register (PTC7) is used for the input level selection.

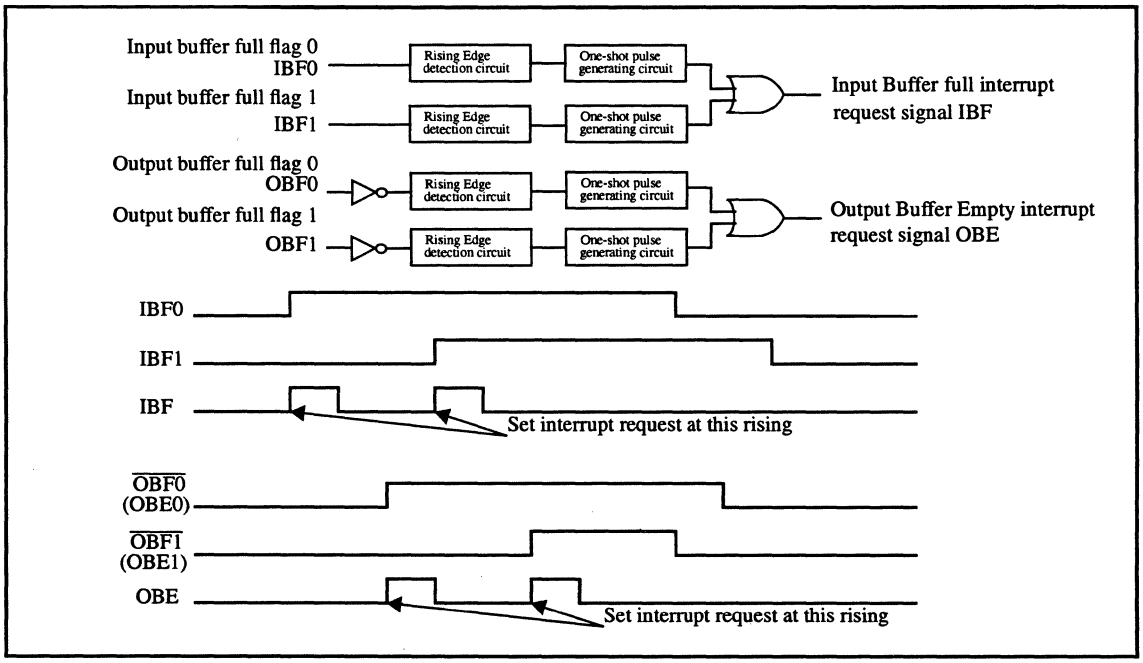


Figure 2-74. Data Bus Buffer Interrupt Request Circuit

MSB 7								LSB 0
DBBS07	DBBS06	DBB05	DBBS04	DBBS03	DBBS02	DBBS01	DBBS00	

Address: 0049_{16}
Access: R/W
Reset: 00_{16}

DBBS00 Output Buffer Full (OBF_0) Flag
0 Output buffer empty.
1 Output buffer full.

DBBS01 Input Buffer Full (IBF_0) Flag
0 Input buffer empty.
1 Input buffer full.

DBBS02 User Definable (U2) Flag

DBBS03 A_0 (A_{00}) Flag
Indicates the A_0 status when IBF flag is set

DBBS04 User Definable (U4) Flag

DBBS05 User Definable (U5) Flag

DBBS06 User Definable (U6) Flag

DBBS07 User Definable (U7) Flag

Figure 2-75. Data Bus Buffer Status Register 0

MSB 7								LSB 0
DBBC07	DBBC06	Reserved	DBBC04	DBBC03	DBBC02	DBBC01	DBBC00	

Address: $004A_{16}$
Access: R/W
Reset: 00_{16}

DBBC00 OBF Output Selection Bit
0 P52 pin is operated as normal I/O pin
1 P52 pin is operated as OBF_0 output pin

DBBC01 $\overline{IBF}$ Output Selection Bit
0 P53 pin is operated as normal I/O pin
1 P53 pin is operated as $\overline{IBF}_0$ output pin

DBBC02 IBF_0 Interrupt Selection Bit
0 IBF_0 interrupt is generated by both write-data (A_0 = zero) and write-command (A_0 = one)
1 IBF_0 interrupt is generated by write-command (A_0 = one) only

DBBC03 Output buffer 0 empty interrupt disable

DBBC04 Input buffer 0 full interrupt disable

DBBC05 Reserved

DBBC06 Master CPU Bus Interface Enable Bit
0 P60-P67, P54-P57 are GPIO pins
1 P60-P67, P54-P57 are bus interface signals DQ0-DQ7, $\overline{S}_0$, A_0, $\overline{R}$, $\overline{W}$ respectively.

DBBC07 Bus Interface Type Selection Bit
0 RD, WR separate type bus
1 R/W type bus.

Figure 2-76. Data Bus Buffer Control Register 0

MSB 7								LSB 0
DBBS17	DBBS16	DBB15	DBBS14	DBBS13	DBBS12	DBBS11	DBBS10	

Address: $004D_{16}$
Access: R/W
Reset: 00_{16}

DBBS10 Output Buffer Full (OBF_1) Flag
0 Output buffer empty.
1 Output buffer full.

DBBS11 Input Buffer Full (IBF_1) Flag
0 Input buffer empty.
1 Input buffer full.

DBBS12 User Definable (U2) Flag

DBBS13 A_0 (A_{01}) Flag
Indicates the A_0 status when IBF flag is set

DBBS14 User Definable (U4) Flag

DBBS15 User Definable (U5) Flag

DBBS16 User Definable (U6) Flag

DBBS17 User Definable (U7) Flag

Figure 2-77. Data Bus Buffer Status Register 1

MSB 7								LSB 0
DBBC17	Reserved	Reserved	DBBC14	DBBC13	DBBC12	DBBC11	DBBC10	

Address: $004E_{16}$
Access: R/W
Reset: 00_{16}

DBBC10 OBF_1 Output Selection Bit
0: P74 pin is operated as normal I/O pin
1: P74 pin is operated as OBF_1 output pin if DBBC17 = one

DBBC11 $\overline{IBF}_1$ Output Selection Bit
0: P73 pin is operated as normal I/O pin
1: P73 pin is operated as $\overline{IBF}_1$ output pin if DBBC17 = one

DBBC12 IBF_1 Interrupt Selection Bit
0: IBF_1 interrupt is generated by both write-data (A_0 = zero) and write-command (A_0 = one)
1: IBF_1 interrupt is generated by write-command (A_0 = one) only

DBBC13 Output Buffer 1 Empty interrupt disable
1: Disable
0: Enable (default)

DBBC14 Input Buffer 1 Full interrupt disable

DBBC15 Reserved

DBBC16 Reserved

DBBC17 Data Bus Buffer Function Selection bit
0: Single data bus buffer - P72 is used as GPIO
1: Double data bus buffer - P72 is used as $\overline{S}_1$ input

Figure 2-78. Data Bus Buffer Control Register 1

2.11.1 Data Bus Buffer Status Registers (DBBS0, DBBS1)

The data bus buffer status register is an 8-bit register that indicates the data bus status, with bits 0, 1, and 3 being dedicated read-only bits. Bits 2, 4, 5, 6, and 7 are user definable flags set by software, and can be read and write. When the A_0 pin is High, the master CPU can read the contents of this register.

Output Buffer Full Flag (OBF_0, OBF_1)

The OBF_0 and the OBF_1 flags are set High when data is written to the output data bus buffer by the slave CPU and is cleared to "0" when data is read by the master CPU.

Input Buffer Full Flag (IBF_0, IBF_1)

The IBF_0 and the IBF_1 flags are set High when data is written to the input data bus buffer by the master CPU and is cleared to "0" when data is read by the slave CPU.

A_0 Flag (A_{00}, A_{01})

The level of the A_0 pin is latched when data has been written from the host CPU to the input data bus buffer.

2.11.2 Input Data Bus Buffer Registers ($DBBIN_0$, $DBBIN_1$)

The data on the data bus is latched into $DBBIN_0$ or $DBBIN_1$ by a write request from the master CPU. The data in $DBBIN_0$ or $DBBIN_1$ can be read from the data bus buffer register in the SFR area.

2.11.3 Output Data Bus Buffer Registers ($DBBOUT_0$, $DBBOUT_1$)

Data is set in $DBBOUT_0$ or $DBBOUT_1$ by writing to the data bus buffer register in the SFR area. When the A_0 pin is Low, the data of this register is output by a read request from the host CPU.

2.12 UART

Address	UART1 Description	Acronym and Value at Reset
0030_{16}	UART1 mode register	U1MOD=00
0031_{16}	UART1 baud rate generator	U1BRG=XX
0032_{16}	UART1 status register	U1STS=03
0033_{16}	UART1 control register	U1CON=00
0034_{16}	UART1transmit/receiver buffer 1	U1TURB1=XX
0035_{16}	UART1transmit/receiver buffer 2	U1TURB2=XX
0036_{16}	UART1 RTS control register	U1RTSC=00

Address	UART2 Description	Acronym and Value at Reset
0038_{16}	UART2 mode register	U2MOD=00
0039_{16}	UART2 baud rate generator	U2BRG=XX
$003A_{16}$	UART2 status register	U2STS=03
$003B_{16}$	UART2 control register	U2CON=00
$003C_{16}$	UART2transmit/receiver buffer 1	U2TRB1=XX
$003D_{16}$	UART2transmit/receiver buffer 2	U2TRB2=XX
$003E_{16}$	UART2 RTS control register	U2RTSC=00

Pin	Description
UTXD1	UART1 transmit pin is multiplexed with P84.
URXD1	UART1 receive pin is multiplexed with P85.
$\overline{CTS1}$	UART1 $\overline{CTS1}$ pin is multiplexed with P86
RTS1	UART1 $\overline{RTS1}$ pin is multiplexed with P87

Pin	Description
UTXD2	UART2 transmit pin is multiplexed with P80.
URXD2	UART2 receive pin is multiplexed with P81.
CTS2	UART2 $\overline{CTS2}$ pin is multiplexed with P82
RTS2	UART2 $\overline{RTS2}$ pin is multiplexed with P83

This chip contains two identical UARTs. Each UART has the following main features:

- Clock selection: Φ or SCSGCLK
- Prescaler selection: x1/x8/x32/x256 divisions (both Φ and SCSGCLK)
- Baud rate: 9.5 bits/second - 625 Kbytes/second (at Φ = 10MHz)
- Error detection: parity/framing/overrun/error sum
- Parity: odd/even/none
- Stop bits: 1 or 2
- Character length: 7, 8, or 9 bits
- Transmit/receive buffer: 2 stages (double buffering)
- Interrupt generation conditions: Tx Buffer Empty or Transmit Complete, Rx Buffer full and Receive error sum.
- Address mode for multi-receiver environment

The following descriptions apply to both UARTs.

The UART receives parallel data from the core, or DMA converts it into serial data, and transmits the results to the send data output terminal UTXDx. The UART receives serial data from an external source through the receive data input, URXDx, converts it into parallel data and makes it available to the core or DMA. The UART can detect parity, overrun, and framing errors in the input stream and report the appropriate status information. A double buffering configuration is used for the UART's transmit and receive operations. This double buffering is accomplished by the use of a Transmit Buffer and Transmit Shift Register on the transmit side and the Receive Buffer and Receive Shift Register on the receive side.

The UART generates the Transmit interrupt when either the Transmit Buffer Empty (TBE) flag or the Transmit Complete (TCM) flag are set, depending on the state of the Transmit Interrupt Source Selection (bit 4 of the UxCON). The UART generates the Rx Buffer full interrupt when receiving and

the Rx Buffer Full flag goes High. The Receive Error interrupt is generated instead of a Rx Buffer full interrupt if the UART detects an error when receiving (see Figure 2-79.). Enabling a transmit or receive operation by setting the TEN or the REN (bits 0 and 1 of UxCON) automatically forces the corresponding the UART port pins in the appropriate direction.

The UART supports an address mode for use in a multi-receiver environment where an address is sent before each message to designate which UART or UARTs are to wake-up and receive the message.

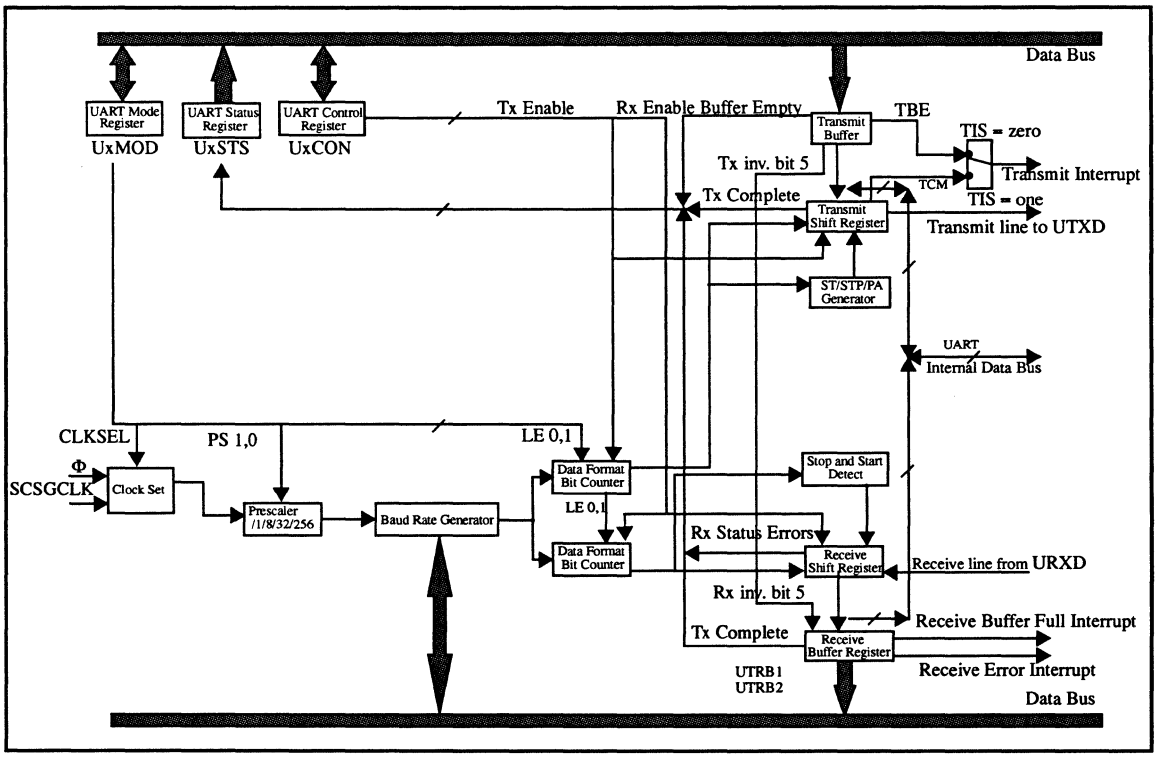


Figure 2-79. UART Block Diagram

2.12.1 Baud Rate Selection

Either an internal clock Φ or the output of the chip special count source generator (SCSGCLK) can be selected as the input clock source by means of the UART Clock Selection Bit (CLK) (bit 0 of the UART Mode Register (UxMOD)). UART rate selection is controlled by the UxBRG and is calculated as follows:

$$\text{Baud Rate} = f/[16 \times (n + 1)]$$

where n denotes the decimal value set in the UxBRG, and where f denotes the clock frequency that depends on the clock selection. When the internal clock Φ is selected, f is the prescaled value of the internal clock Φ.

f = Φ/1, Φ/8, Φ/32, or Φ/256

Bits PS0 and PS1 of the UxBRG are used to select a prescaling factor for the internal clock. For Φ = 10MHz, the correspondence between prescale values and baud rate is given as an example in Figure 2-79. When SCSGCLK is selected, f is the prescaled value of SCSGCLK, which is the output of the chip BRG.

f = SCSGCLK/1, SCSGCLK/8, SCSGCLK/32 or SCSGCLK/256

Φ/1		Φ/8		Φ/32		Φ/256	
n	Baud Rate	n	Baud Rate	n	Baud Rate	n	Baud Rate
0	625,000.0						
1	312,500.0						
2	208,333.3						
3	156,250.0						
4	125,000.0						
5	104,166.7						
6	89,285.7						
7	78,125.0						
8	69,444.4	0	78,125.0				
9	62,500.0	1	39,062,5				
10	.	2	26,041.7	0	19,531.3		
11	.	3	19,531.8	1	9,765.6		
12	.	4	15,625.0	2	6,510.4		
13	.	5	13,020.8	3	4,882.8		
.	.	6	11,160.7	4	3,906.3		
.	.	7	.	5	3,255.2		
.	.	8	.	6	2,790.2		
.	.		.	7	2,441.4	0	2,441.4
.	.		.	8	2,170.1	1	1,220.7
.	.		.		.	2	813.8
.	.		.		.	3	610.3
.	.		.		.	4	488.3
.	.		.		.	5	406.9
.	.		.		.	6	348.7
.	.		.		.		.
.	.		.		.		.
255	441.4	255	305.2	255	76.3	255	9.54

Figure 2-80. Prescale Value and Baud Rate Table

2.12.2 UART Mode Register

UxMOD defines data formats and selects the clock to be used (see Figure 2-82.)

2.12.3 UART Baud Rate Register

In the UART Baud Rate Register (UxBRG), any value can be specified to obtain the desired baud rate. This register remains in effect whether the UART is send-enabled, receive-enabled, transmit-in-progress, or receive-in-progress. The contents of this register can be modified only when the UART is not in any of these four states.

2.12.4 UART Control Register

The UxCON specifies the initialization and enabling of a transmit/receive process (see Figure 2-81.) Information can be read from and written to the Control Register. Bits 5 and 6 are always Low during read; writing to these bits is ignored.

2.12.5 UART Status Register

The UART Status Register (UxSTS) reflects both the transmit and receive status (see Figure 2-83.). The status register is read only. The MSB is always "0" during a read operation. Writing to this register has no effect. Status flags are set and reset under the conditions indicated below. The setting and resetting of the transmit and receive status are not affected by transmit and receive enable flags. The setting and resetting of the receive error flags and receive buffer full flag differs when UART address mode is enabled. These differences are described in "2.12.9 UART Address Mode".

Receive Error Sum Flag

The Receive Error Sum Flag (SER) is set when an overrun, framing, or parity error occurs after completion of a receive operation.

It is reset when the status register is read, the hardware reset is asserted, or the receiver is initialized by setting the Receive Initialization Bit (RIN). If a receive operation is completed when the status register is being read, the status information is updated upon completion of the status register read.

Receive Overrun Flag

The Receive Overrun Flag (OER) is set if the previous lower byte data is not read before the current receive operation is completed. It is also set if a receive error occurred for the previous data and the status register is not read before the current receive operation is completed. This flag is reset when the status register is read. This flag is also reset when the hardware reset is asserted or the receiver is initialized by RIN. If a receive operation is completed when the status register is being read, the status information is updated upon completion of the status register read.

Receive Framing Error Flag

The Receive Framing Error Flag (FER) is set when the stop bit of the received data is "0". If the Stop Bit Selection Bit (STB, bit 3) is set, the flag is set if either of the two stop bits is low. This flag is reset when the status register is read, the hardware reset is asserted, or the receiver is initialized by RIN. If a receive operation is completed when the status register is being read, the status information is updated upon completion of the status register read.

Receive Parity Error Flag

The Receive Parity Error Flag (PER) is set when the parity of received data and the Parity Selection Bit (PMD, bit 4) are different. It is enabled only if the Parity Enable Bit (PEN, bit 5) is set.

This flag is reset when the status register is read, the hardware reset is asserted, or the receiver is initialized by RIN. If a receive operation is completed when the status register is being read, the status information is updated upon completion of the status register read.

Receive Buffer Full Flag

The Receive Buffer Full flag (RBF) is set when the last stop bit of the data is received. It is not set when a receive error occurs. This flag is reset when the lower byte of the receive buffer is read, the hardware reset is asserted, or the receive process is initialized by RIN. If a receive operation is completed when the status register is being read, the status information is updated upon completion of the status register read.

Transmission Complete Flag

In the case where no data is contained in the transmit buffer, the Transmission Complete Flag (TCM) is set when the last bit in the transmit shift register is transmitted. In the case of disabling transmission when the transmission when the transmit buffer still contains data, the TCM flag is set when the last bit in the transmit shift register is transmitted. The TCM flag is also set when the hardware reset is asserted or when the transmitter is initialized by setting the Transmit Initialization Bit (TIN) (bit 2). It is reset when a transmission operation begins.

Transmission Buffer Empty Flag

The Transmission Buffer Empty Flag (TBE) is set when the contents of the transmit buffer are loaded into the transmit shift register. The TBE flag is also set when the hardware reset is asserted or when the transmitter is initialized by TIN. It is reset when a write operation is performed on the lower byte of the transmit buffer.

MSB 7								LSB 0
AME	RTS_SEL	CTS_SEL	TIS	RIN	TIN	REN	TEN	

Address: 0033_{16},$003B_{16}$
Access: R/W
Reset: 00_{16}

TEN — Transmission Enable Bit
0:Disable the transmit process
1:Enables the transmit process. If the transmit process is disabled (TEN cleared) during transmission, the transmit will not stop until completed.

REN — Receive Enable Bit
0:Disable the receive process
1:Enables the receive process. If the receive process is disabled (REN cleared) during reception, the receive will not stop until completed.

TIN — Transmission Initialization Bit
0:No action.
1:Resets the UART transmit status register bits as well as stopping the transmission operation. The TEN bit must be set and the transmit buffer reloaded in order to transmit again. The TIN is automatically reset one cycle after TIN is set.

RIN — Receive Initialization Bit
0:No action.
1:Clears the receive status flags for the UART and the REN bit. If RIN is set during receive in progress, receive operation is aborted. The RIN bit is automatically reset one cycle after RIN is set.

TIS — Transmit Interrupt Source Selection Bit
0:Transmit interrupt occurs when the Transmit Buffer Empty flag is set.
1:Transmit interrupt occurs when the Transmit Complete flag is set.

CTS_SEL — Clear To Send (CTS) Selection Bit
0:CTS function is disabled, P86 (or P82) is used as GPIO pin.
1:CTS function is enabled, P86 (or P82) is used as CTS input.

RTS_SEL — Request To Send (RTS) Selection Bit
0:RTS function is disabled, P87 (or P83) is used as GPIO pin.
1:RTS function is enabled, P83 (or P83) is used as RTS output.

AME — UART Address Mode Enable Bit
0:Address Mode disabled.
1:Address Mode enabled.

Figure 2-81. UxCON Register

MSB 7								LSB 0
LE1	LE0	PEN	PMD	STB	PS1	PS0	CLK	

Address: 0030_{16}, 0038_{16}
Access: R/W
Reset: 00_{16}

CLK — UART Clock Selection Bit
0:Φ
1:SCSGCLK

PS0-1 — Internal Clock Prescaling Selection Bits
00:Division by 1
01:Division by 8
10:Division by 32
11:Division by 256

STB — Stop Bits Selection Bit
0:1
1:2

PMD — Parity Selection Bit
0:Even
1:Odd

PEN — Parity Enable Bit
0:Off
1:On

LE0,1 — Uart Character Length Selection Bits
00:7 bits/character
01:8 bits/character
10:9 bits/character
11:Reserved

Figure 2-82. UxMOD Register

MSB 7								LSB 0
Reserved	SER	OER	FER	PER	RBF	TBE	TCM	

Address: 0032_{16}, $003A_{16}$
Access: R only
Reset: 03_{16}

TCM — Transmit-Complete (Transmission Register Empty) Flag
0:Data in the transmission register.
1:No data in the transmission register.

TBE — TX Buffer Empty Flag
0:Data in the TX Buffer.
1:No data in the TX Buffer.

RBF — RX Buffer Full Flag
0:No data in the RX Buffer.
1:Data in the RX Buffer.

PER — Receive Parity Error Flag
0:No receive parity error.
1:Receive parity error.

FER — Receive Framing Error Flag
0:No receive framing error.
1:Receive framing error.

OER — Receive Overrun Flag
0:No receive overrun.
1:Receive overrun.

SER — Receive Error Sum Flag
0:No receive error.
1:Receive error.

Bit 7 — Reserved (Read as zero)

Figure 2-83. UxSTS Register

MSB 7								LSB 0
RTS3	RTS2	RTS1	RTS0	Reserved	Reserved	Reserved	Reserved	

Address: 0036_{16}, $003E_{16}$
Access: R/W
Reset: 80_{16}

RTS3:0 — RTS Assertion Delay Count 3:0
0000: No delay, $\overline{RTS}$ asserts immediately upon receive operation completed.
0001: $\overline{RTS}$ asserts 8 bit-time upon receive operation completed.
0010: $\overline{RTS}$ asserts 16 bit-time upon receive operation completed.
0011: $\overline{RTS}$ asserts 24 bit-time upon receive operation completed.
.
.
1110: $\overline{RTS}$ asserts 112 bit-time upon receive operation completed.
1111: $\overline{RTS}$ asserts 120 bit-time upon receive operation completed.

Bit 3:0 — Reserved

Figure 2-84. UxRTSC Register

2.12.6 Transmit/Receive Format

Transmit Method

Setup

- Set the baud rate by writing a value from 0-255 into the UxBRG.
- Set the TIN to one, bit 2 of UxCON. This will reset the transmit status to a value of 03_{16}.
- Select the interrupt source to either the TBE or the TCM by clearing or setting the TIS bit, bit 4.
- Set the data format and clock selection by writing the appropriate value to UxMOD.
- Set the TEN to one, bit 0 in the UxCON.

Operation

- If no data is being shifted out of the Transmit Shift Register, the TCM Flag in UxSTS goes High and the data written to the Transmit Buffer Register is transferred to the Transmit Shift Register. The TBE flag is set High and TCM is signalling that the next byte of data can be written to the transmit buffer.
- Data from the Transmit Shift Register is transmitted one bit at a time beginning with the start bit and ending with the stop bit. Note that the LSB is transmitted first.
- If the TEN bit is cleared while data is still being transmitted, the transmitter will continue until the last bit is sent.
- When the last bit is transmitted, the TCM bit is set High. The transmitter is now ready for the next byte.

Receive Method

Set up

- Set the baud rate by writing a value from 0-255 into UxBGR.
- Set the RIN, bit 3 in the UxCON, to "1".
- Set the data format and the clock selection by using writing the appropriate value to UxMOD.
- Set the REN, bit 1 in the UxCON, to "1".

Operation

- Input data received through the URXDx pin is read one bit at a time, LSB first, into the Receive Shift Register when the start bit is detected and the receiver is enabled.
- When the number of bits specified by the data format has been received and the stop bit is detected, the contents of the Receive Shift Register are transferred to the Receive Buffer Register and the Rx Buffer Full Flag is set High in the UxSTS, if a receive error has not occurred. The receive interrupt request is also generated at this time if a receive interrupt has not occurred. Also, at this time the error flags are checked and the receive interrupt request is generated.
- When the Receive Buffer Register is read, the Rx Buffer Full Flag is cleared, and the Receive Buffer Register is now ready for the next byte.

2.12.7 Interrupts

The transmit and receive interrupts are generated under the conditions described below. The generation of the receive interrupts differs when UART Address mode is enabled. The differences are described in "2.12.9 UART Address Mode".

Transmit interrupts

The UART generates a Transmit Interrupt to the CPU core. The source of the Transmit Interrupt is selectable by setting TIS.

- If TIS = "0", the Transmit interrupt is generated when the transmit buffer register becomes empty (that is, when TBE flag set).
- If TIS = "1", The Transmit interrupt is generated after the last bit is sent out of the transmit shift register and no data has been written to the transmit buffer (that is, when TCM flag set).

Receive Interrupts

The UART generates the Receive Buffer Full (RBF) and Receive Error interrupts to the CPU core when receiving.

- The RBF interrupt is generated when a receive operation completes and a receive error is not generated.
- The Receive Error (SER) interrupt is generated when a receive error sum, overrun, framing or parity error occurs.

2.12.8 Clear-to Send ($\overline{CTS}$) and Request-to-Send ($\overline{RTS}$) Signals

The UART, as a transmitter, recognizes the Clear-to-Send ($\overline{CTS}$) input as a handshaking signal. As a receiver, the UART will generate the Request-to-Send ($\overline{RTS}$) handshaking signal.

Clear-to-Send ($\overline{CTS}$) Input

When TEN is set and Tx buffer is loaded, the UART begins the transmission process when a $\overline{CTS}$ is asserted (low input). After beginning a send operation, the UART does not stop sending until the transmission is completed, even if the $\overline{CTS}$ is negated (high input). If TEN is cleared, the UART will not stop transmitting and the port pins will remain under the control of the UART until the end of the transmission.

Request-to-Send ($\overline{RTS}$) Output

The UART controls the $\overline{RTS}$ output under the following conditions:

Assertion conditions (active low):

- Receive-enable (REN in Control Register) is set.
- Receive operation has completed during receiver being enabled.

The timing of the RTS assertion from the last frame's stop bit is programmable.(see 'RST control register' for detail)

Non-assertion conditions (inactive high):

- Falling edge of the start bit is detected during Receive-enable.
- Receive-enable (REN in Control Register) is cleared before receive operation is in progress.
- Hardware reset.
- Receiver of the UART is initialized (RIN is set).

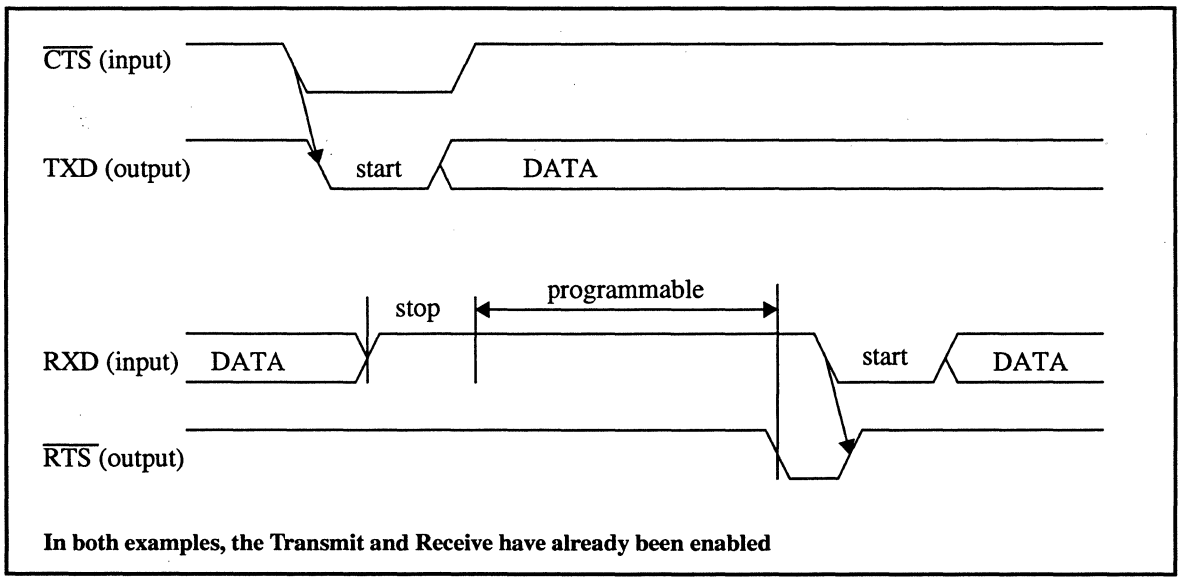


Figure 2-85. $\overline{CTS}$ and $\overline{RTS}$ Timing Example

2.12.9 UART Address Mode

The UART address mode is intended for use in a multi-receiver environment where an address is sent before each message to designate which UART or UARTs are to wake-up and receive the message. An address is identified by the MSB of the incoming data byte being a "1". The bit is "0" for non-address data. UART address mode can be used in either 8-bit or 9-bit character length mode. The character length is chosen by writing the appropriate values to the UART Character Length Selection Bits (LE0,1).

UART address mode is enabled by setting the UART Address Mode Enable Bit (AME) to "1". When UART address mode is enabled, the MSB of a newly received byte of data (that is either 8 or 9 bits in length) is examined if a valid stop bit is detected and a parity error has not occurred (if parity is enabled). If the MSB is "1", then the receive buffer full interrupt and flag are set and AME is automatically cleared, disabling UART address mode. If the MSB is "0", then the receive buffer full interrupt is not set. However, the RBF flag is still set for this case. If a valid stop bit is not detected or a parity error has occurred, neither the receive buffer full flag nor interrupt is set and the MSB of the data is not examined. Instead, either the framing error or parity error flag is set, the error sum flag is set, and the error sum interrupt is set.

While in UART address mode, the generation of overrun errors is disabled after the first byte of data is received. Therefore, when non-address data is received without errors while in the UART address mode, it is not necessary to read the UART receive buffer prior to the reception of the next byte of data. Also, if a framing or parity error occurs while in UART address mode, it is not necessary to read the UxSTS prior to the reception of the next byte of data. However, an overrun error will occur if an address byte is received and the UART receive buffer is not read before a new byte of data is received. This is the case because the UART address mode was automatically disabled when the address byte was received. Also, an overrun error will occur for the first byte received after UART address mode is enabled if the preceding byte received did not generate an error and the UART receive buffer was not read, or the preceding byte did generate an error and UxSTS was not read.

When the MSB is "1" and the UART address mode is automatically disabled, the UART reverts back to normal reception mode. In normal reception mode, the value of the MSB of each byte of received data has no effect on the setting of the receive buffer full interrupt or the determination of overrun errors.

2.13 Serial I/O

Address	Description	Acronym and Value at Reset
$002A_{16}$	SIO shift register	SIOSHT=XX
$002B_{16}$	SIO control register 1	SIOCON1=00
$002C_{16}$	SIO control register 2	SIOCON2=00

Name	Pin
SRDY	is multiplexed with P80
SCLK	is multiplexed with P81
SRXD	is multiplexed with P82
STXD	is multiplexed with P83

The SIO uses the clock synchronous method (see Figure 2-86.).

- Transfer method: Half Duplex data transfer is available.
- Synchronous Clock
- Internal Clock (when serial I/O synchronous clock select bit is "1", internal clock source divided by 2, 4, 8, 16, 32, 64, 128, 256 can be selected). If bit 1 of SIO Control Register2 is "0", internal clock source = Φ; if bit 1 of SIO Control Register2 is "1", internal clock source = SCS-GCLK
- External Clock (when SIO synchronous clock select bit is "1", an external clock input from the SCLK pin is selected).

2.13.1 SIO Control Register

The Serial I/O Control Register controls the various SIO functions (see Figure 2-87.). All of this register's bits can be read from and written to by software. At reset, this register is cleared to 00_{16}. The SIO Control Register determines whether the device's pins are used as ordinary I/O ports or as SIO function pins. This register also determines the transfer direction and transfer clock for serial data.

2.13.2 SIO Operation

An internal clock or an external clock can be selected as the synchronous clock. When the internal clock is chosen, dividers are built in to provide eight different clock selections. If an internal clock is selected, start of transfer is done by a write signal to the serial I/O register. After an 8-bit transfer is completed, the TxD pin enters a high-impedance state. If an external clock is selected, the contents of the serial I/O register continue to be shifted while the send/receive clock is being input. Therefore, the clock needs to be controlled by the external source. Also there is no TxD high impedance function after data is transferred.

Regardless of an internal or external clock, after an 8-bit transfer, the interrupt request bit is set. Figure 2-89. shows the timing for the serial I/O with the LSB-first option selected.

SIO can be operated in slave mode. In slave mode the SRDY pin becomes an input from a master. If SRDY is held High, the shift clock is inhibited, STXD is tri-stated, and shift count is reset. If SRDY is held Low, then the normal shift operation is performed.

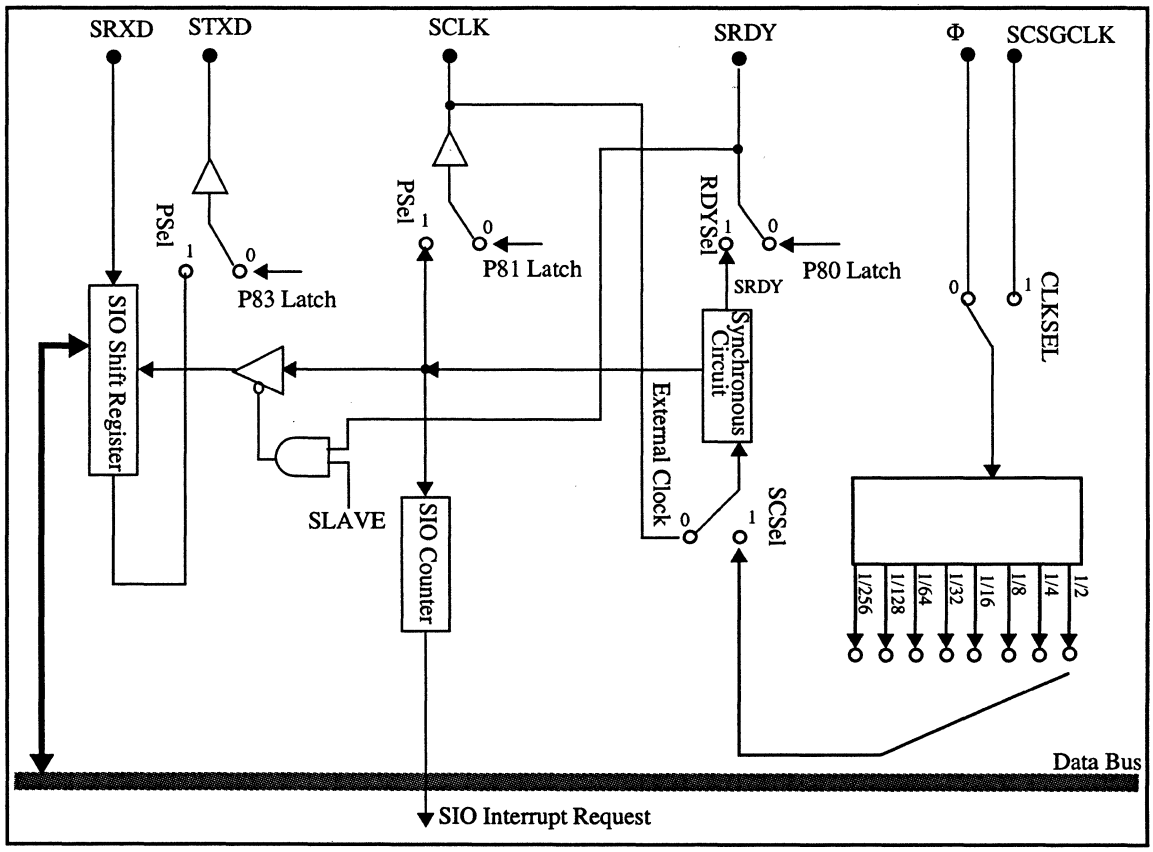


Figure 2-86. Clock Synchronous SIO Block Diagram

MSB 7								LSB 0
OCHCont	SCSel	TDSel	RDYSel	PSel	ISCSel2	ISCSel1	ISCSel0	

Address: $002B_{16}$
Access: R/W
Reset: 00_{16}

ISCSel0-2 Internal Synchronization Clock Select Bit
000: Internal Clock divided by 2.
001: Internal Clock divided by 4.
010: Internal Clock divided by 8.
011: Internal Clock divided by 16
100: Internal Clock divided by 32.
101: Internal Clock divided by 64
110: Internal Clock divided by 128.
111: Internal Clock divided by 256

PSel SIO Port Selection Bit
0: I/O Port
1: TxD output, SCLK function

RDYSel SRDY Output Select Bit
0: I/O Port.
1: SRDY signal

RDSel Transfer Direction Select Bit
0: LSB first.
1: MSB first.

SCSel Synchronization Clock Select Bit
0: External Clock.
1: Internal Clock.

OCHCont TxD Output Channel Control Bit
0: CMOS output.
1: N-Channel open drain output.

Figure 2-87. SIO Control Register 1

MSB 7								LSB 0
Reserved	Reserved	Reserved	Reserved	Reserved	RXDSel	CLKSEL	SLAVE	

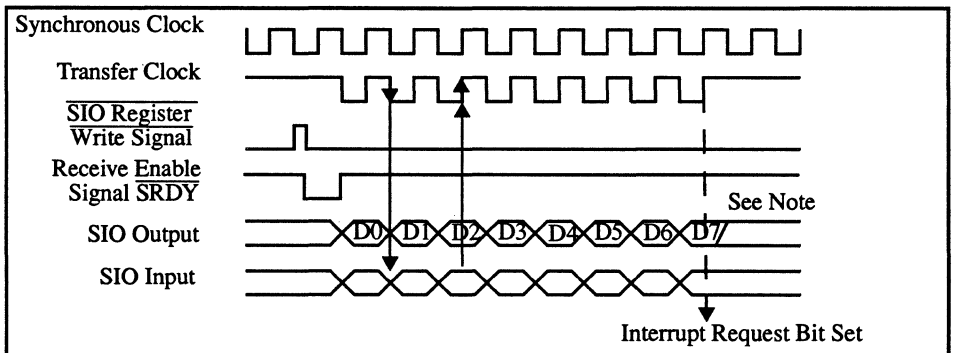
Address: $002C_{16}$
Access: R/W
Reset: 00_{16}

SLAVE Slave Mode Selection
0: Normal mode
1: Slave mode (to enter Slave mode, bit 4 of SIO Control register 1 also needs to set)

CLKSEL SIO Internal Clock Selection Bit
0: Φ
1: SCSGCLK

RXDSel SRXD Input Selection Bit
0: SRXD input disabled

Figure 2-88. SIO Control Register 2



Note: When the internal clock is selected, the TxD pin goes into high-impedance after the data is transferred.

Figure 2-89. Normal Mode SIO Function Timing (with LSB-First selected)

2.14 Special Count Source Generator

Address	Description	Code
$002D_{16}$	Special Count Source Generator1	SCSG1=FF
$002E_{16}$	Special Count Source Generator2	SCSG2=FF
$002F_{16}$	Special Count Source mode register	SCSM=00

This device has a built-in special count source generator. It consists of two 8-bit timers: SCSG1, and SCSG2 (see Figure 2-90.) The contents of the timer latch, corresponding to each timer, determine the divide ratio. The timers can be written to at any time. The output of the special count source generator can be a clock source for Timer X, SIO and the two UARTs.

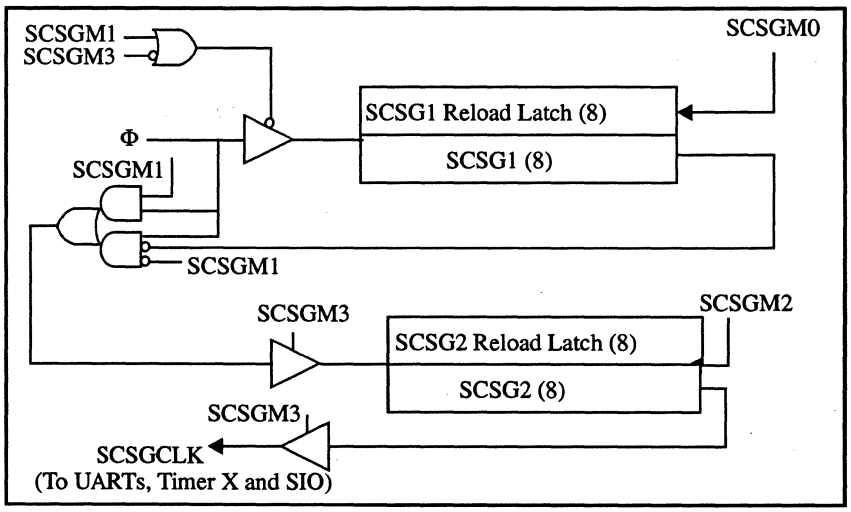


Figure 2-90. SCSG Block Diagram

2.14.1 SCSG Operation

The SCSG1 and SCSG2 are both down count timers. When the count of a timer reaches 00_{16}, an underflow occurs at the next count pulse and the contents of the corresponding timer reload latch are loaded into the timer. For the count operation for SCSG1 with the Data Write Mode set to write to the latch only see (Figure 2-91.).

A memory map and the initial values after reset of the timers and timer reload latches are detailed above. The divide ratio of each timer is given by $1/(n + 1)$, where n is the value written to the timer. The output of the first timer (SCSG1) is effectively ANDed with the original clock (Φ) to provide a count source for the second timer (SCSG2). This results in a count source of $n/(n + 1)$ being fed to SCSG2.

The output of the SCSG is a clock, SCSGCLK. The frequency is calculated as follows:

$SCSGCLK = \Phi \bullet \frac{SCSG1}{SCSG1 + 1} \bullet \frac{1}{SCSG2 + 1}$ where SCSG1 is the value written to SCSG1 and SCSG2 is the value written to SCSG2.

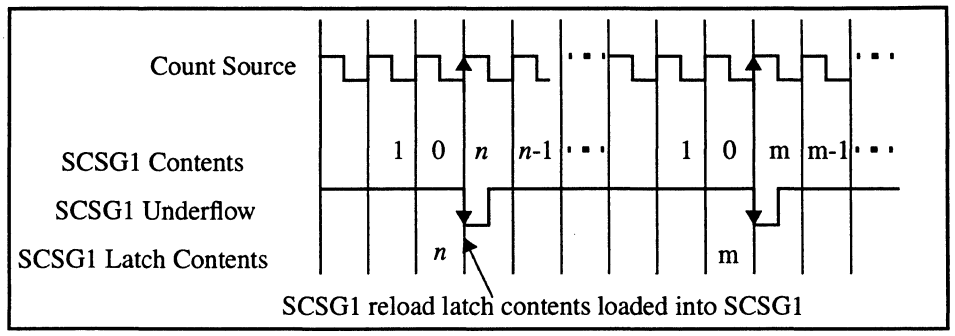


Figure 2-91. Timer Count Operation for SCSG1

2.14.2 SCSG Description

2.14.2.1 SCSG1

SCSG1 is an 8-bit timer that has an 8-bit reload latch, and is a normal count down timer.

Write Method

When writing to the timer, the data is placed in the SCSG1 reload latch. At this point, if the SCSG1 Data Write Control Bit (SCSGM0) is "0", the value in the SCSG1 reload latch is also loaded in SCSG1. If SCSGM0 is "1", the data in the SCSG1 reload latch is loaded in SCSG1 after SCSG1 underflows.

SCSG1 Count Stop Control

If the SCSG1 Count Stop Bit (SCSGM1) (bit 1 of the SCSGM Register) is set High, SCSG1 stops counting. This allows Φ to bypass SCSG1 and act as the clock source for SCSG2. If the SCSGCLK Output Control Bit (SCSGM3) is cleared to "0", SCSGCLK is disabled and SCSG1 stops counting (see Figure 2-92.).

2.14.2.2 SCSG2

SCSG2 is an 8-bit timer that has an 8-bit reload latch, and is a normal count down timer.

Write Method

When writing to the timer, the data is placed in the SCSG2 reload latch. At this point, if the SCSG2 Data Write Control Bit (SCSGM2) is Low, the value in the SCSG2 reload latch is also loaded in SCSG2. If SCSGM2 is High, the data in the SCSG2 reload latch is loaded in SCSG2 after SCSG2 underflows.

MSB 7								LSB 0	Address: $002F_{16}$
Reserved	Reserved	Reserved	Reserved	SCSGM3	SCSGM2	SCSGM1	SCSGM0		Access: R/W
									Reset: 00_{16}

SCSGM0	SCSG1 Data Write Control Bit (Bit 0) 0: Write data in latch and timer 1: Write data in latch only
SCSGM1	SCSG1 Count Stop Bit (Bit 1) 0: Count start 1: Count stop
SCSGM2	SCSG2 Data Write Control Bit (Bit 2) 0: Write data in latch and timer 1: Write data in latch only
SCSGM3	SCSGCLK Output Control Bit (Bit 3) 0: SCSGCLK output disabled (SCSG1 and SCSG2 off) 1: SCSGCLK output enabled.
SCSGM7-4	Not used (zero when read)

Figure 2-92. SCSGM Register

SCSG2 Count Stop Control

If the SCSGCLK Output Control Bit (SCSGM3) is cleared to "0", SCSGCLK is disabled and SCSG2 stops counting.

SCSG2 Output (SCSGCLK)

The output signal SCSGCLK (output to the UART and Timer blocks) is controlled by SCSGM3. When the SCSGCLK Output Control Bit (SCSGM3) is cleared to "0", SCSGCLK is disabled.

2.15 Oscillator Circuit

2.15.1 Description

An on-chip oscillator provides the system and peripheral clocks as well as the USB clock necessary for operation. This oscillator circuit is comprised of amplifiers that provide the gain necessary for oscillation, oscillation control logic, a frequency synthesizer, and buffering of the clock signals. A block diagram of the oscillator circuit is shown in Figure 2-94. The following external clock inputs are supported:

- A Ceramic resonator or quartz crystal oscillator of up to 48 MHz, connected to the X_{in} and X_{out} pins.
- An external clock signal of up to 48 MHz, connected to the X_{in} pin.
- A Ceramic resonator or quartz crystal oscillator of 32.768 kHz, connected to the XC_{in} and XC_{out} pins.
- An external clock signal of up to 5.12 MHz, connected to the XC_{in} pin.

The frequency synthesizer can be used to generate a 48MHz clock signal (f_{USB}) needed by the USB block and clock f_{SYN}, which can be chosen as the source for the system and peripheral clocks. Both f_{USB} and f_{SYN} are phase-locked frequency multiples of the frequency synthesizer input. The inputs to the frequency synthesizer can be either X_{in} or XC_{in}.

One of three clock signals can be chosen as the source for the system and peripheral clocks; $f_{XIN}/2$, f_{XIN}, f_{XCIN}, or f_{SYN}. The selection is based on the values of bits CPMA6, CPMA7 and CCR7. The default source after reset is $f_{XIN}/2$.

The default source for the system and peripheral clocks is $f_{XIN}/2$. If f_{XIN} = 24MHz, then the CPU will be running at Φ = 6MHz (low frequency mode.) For the CPU to run in high frequency mode, i.e., source of clock = f_{XIN}, write a "1" to bit 7 of the clock control register.

MSB 7								LSB 0
CCR7	CCR6	CCR5	CCR4	Reserved	Reserved	Reserved	Reserved	

Address: $001F_{16}$
Access: R/W
Reset: 00_{16}

CCR7: X_{IN} Divider Select
0: $f_{XIN}/2$ is used for the system clock source when CMPA7:6=00
1: f_{XIN} is used for the system clock source when CMPA7:6=00

CCR6: X_{OUT} Oscillation Drive Disable bit
0: X_{OUT} oscillation drive is enabled (when X_{in} oscillation is enabled).
1: X_{OUT} oscillation drive is disabled.

CCR5: X_{COUT} Oscillation Drive Disable Bit
0: X_{COUT} oscillation drive is enabled (when XC_{in} oscillation is enabled).
1: X_{COUT} oscillation drive is disabled.

CCR4: frequency synthesizer Bypass
0: 48MHz USB clock is from the Frequency Synthesizer.
1: 48MHz USB clock is from the X_{in}/X_{out} pins.

Figure 2-93. Clock Control Register

The drive strength of the X_{out} and XC_{out} inverting amplifier can be controlled by bits CPMB7 and CPMA3, respectively. High drive is the default at reset or after executing a STP instruction and must be chosen whenever restarting X_{in} or XC_{in} oscillation if a ceramic or crystal oscillator is used. When oscillation has been established, low drive can be selected to reduce power consumption. If an external clock signal is input to X_{in} or XC_{in}, the inverting amplifiers can be disabled by means of the CCR6 and CCR7 bits, respectively, in order to reduce power consumption.

There is an option to bypass the frequency synthesizer and use a 48MHz crystal on X_{in} - X_{out} to provide f_{USB}. This can be done by writing "1" to CCR4. Note that, in this mode, CCR7 must be set Low.

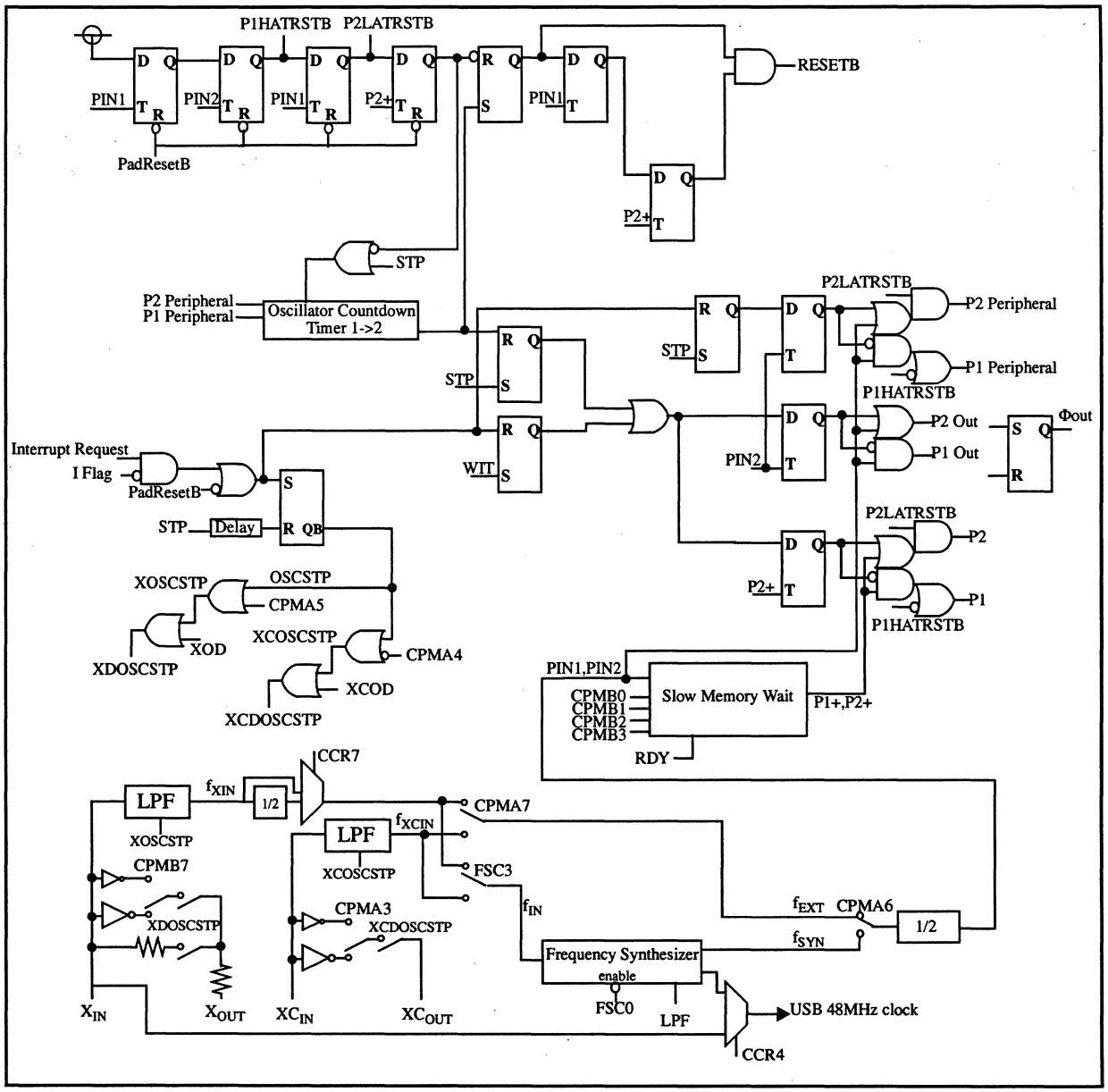


Figure 2-94. Clock Block Diagram

2.15.2 Frequency Synthesizer Circuit

The Frequency Synthesizer Circuit generates a 48MHz clock needed by the USB block and a clock f_{SYN} that are both a multiple of the external input reference clock f_{IN}. A block diagram of the circuit is shown in Figure 2-95..

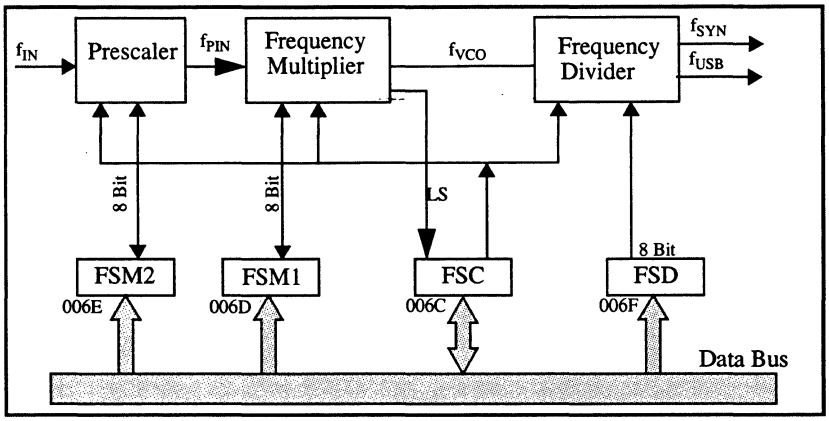


Figure 2-95. Frequency Synthesizer Circuit

The frequency synthesizer consists of a prescaler, frequency multiplier macro, a frequency divider macro, and four registers, namely FSM1, FSM2, FSC and FSD. Two multiply registers (FSM1, FSM2) control the frequency multiply amount. Clock f_{IN} is prescaled using FSM2 to generate f_{PIN}. f_{PIN} is multiplied using FSM1 to generate an f_{VCO} clock that is then divided using FSD to produce the clock f_{SYN}. The f_{VCO} clock is optimized for 48 MHz operation and is buffered and sent out of the frequency synthesizer block as signal f_{USB}. This signal is used by the USB block.

Clock f_{PIN} is a divided down version of clock f_{IN}, which can be either f_{XIN} or f_{XCIN}. The default clock after reset is f_{XIN}. The relationship between f_{PIN} and the clock input to the prescaler (f_{IN}) is as follows:

- $f_{PIN} = f_{IN} / 2(n+1)$ where n is a decimal number between 0 and 254 (TBD). Setting FSM2 to 255 disables the prescaler and $f_{PIN} = f_{IN}$.

MSB 7	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	LSB 0

Address: $006E_{16}$
Access: W/R
Reset: FF_{16}

f_{PIN}	FSM2		f_{IN}
	Dec(n)	Hex(n)	
24 MHz	255	FF	24.00 MHz
1 MHz	11	0C	24.00 MHz
2 MHz	5	05	24.00 MHz
3 MHz	3	03	24.00 MHz
6 MHz	1	01	24.00 MHz
12 MHz	0	00	24.00 MHz

$f_{IN}/2(n+1) = f_{PIN}$

Figure 2-96. Frequency Synthesizer Multiply Control Register FSM2

MSB 7	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	LSB 0

Address: $006D_{16}$
Access: W/R
Reset: FF_{16}

f_{PIN}	Decimal (*n*)	Hex		f_{VCO}
		FSM2	FSM1	
320 kHz	74	0	4A	48.00 MHz
2 MHz	11	0	0B	48.00 MHz
4 MHz	5	0	05	48.00 MHz
6 MHz	3	0	03	48.00 MHz
12 MHz	1	0	01	48.00 MHz
24 MHz	0	0	00	48.00 MHz

$f_{VCO}/2(n+1) = f_{PIN}$

Figure 2-97. Frequency Synthesizer Multiply Control register FSM1

MSB 7	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	LSB 0

Address: $006F_{16}$
Access: W/R
Reset: FF_{16}

f_{vco}	FSD (m)	Decimal	fsyn
48.00 MHz	00	00	24.00 MHz
48.00 MHz	7F	127	187.50 kHz

$f_{VCO}/2(m+1) = f_{SYN}$

Figure 2-98. Frequency Synthesizer Divide Register

The relationship between f_{PIN}, f_{VCO}, and f_{SYN} is as follows:

- $f_{VCO} = f_{PIN} \times 2(n+1)$ where *n* is the decimal equivalent of the value loaded in FSM2,FSM1.

Note: *n* must be chosen such that f_{VCO} equals 48 MHz.

- $f_{SYN} = f_{VCO} / 2(m+1)$ where m is the decimal equivalent of the value loaded in FSD

Note: Setting m = 255 disables the divider and negates f_{SYN}.

The FSC0 bit in the FSC Control Register enables the frequency synthesizer block. When disabled (FSC0 = zero), f_{VCO} is held at either a high or low state. When the frequency synthesizer control bit is active (FSC0 = one), a lock status (LS = one) indicates that f_{SYN} and f_{VCO} are the correct frequency. The LS and FSCO control bits in the FSC Control register are shown in Figure 2-99.

When using the frequency synthesizer, a low-pass filter must be connected to the LPF pin.

MSB 7								LSB 0
LS	Bit 6	Bit 5	Reserved	Bit 3	VCO1	VCO0	FSCO	

Address: $006C_{16}$
Access: W/R
Reset: 60_{16}

FSCO Frequency Synthesizer Enable Bit
0: Disabled
1: Enabled

VCO1,0 Frequency Gain Control Bits
00: TBD
01: TBD
10: TBD
11: TBD

Bit 3 Frequency Synthesizer input selector
0: X_{in}
1: XC_{in}

Bit 4> Not used (Always write 0)

Bits 6,5> LPF Gain Control Bits
00: DO NOT USE
01: Valid in-lock state
10: Valid in-lock state
11: During lock-in state

LS Lock Status Bit
0: Unlocked
1: Locked

Figure 2-99. Frequency Synthesizer Control Register

2.16 Low Power Modes

This device has two low-power dissipation modes:

- Stop
- Wait.

2.16.1 Stop Mode

Use of the stop mode allows the microcomputer to be placed in a state where no internal excitation of the circuitry is taking place, thus resulting in extremely low power dissipation. The microcomputer enters the stop mode when the STP instruction is executed. The internal state of the microcomputer after execution of the STP instruction is as follows:

- All internal oscillation stops with P2 and P2PER held High and P1 and P1PER held Low.
- Timer 1 and Timer 2 are loaded with FF_{16} and 01_{16} respectively.
- The count source for Timer 1 is set to Φ/8 and the count source for Timer 2 is set to Timer 1 underflow.

Oscillation is restarted (for example, all clocks other than P1 and P2 begin to oscillate) when a reset or an external interrupt is received. The interrupt control bit of the interrupt used to release the stop mode must be set High and the I flag set Low prior to the execution of the STP instruction. To allow the oscillation source time to stabilize, the oscillation source is connected as the clock source for the wake-up timer (Timer 1 and Timer 2 cascaded). When Timer 2 underflows, the system clocks P1 and P2 are restarted and the microcomputer services the interrupt that caused the return from the stop state. It then services any other enabled interrupts that occurred, in the order of their respective priorities, and returns to its state prior to the execution of the STP instruction. The timing for the STP instruction is shown in Figure 2-100.

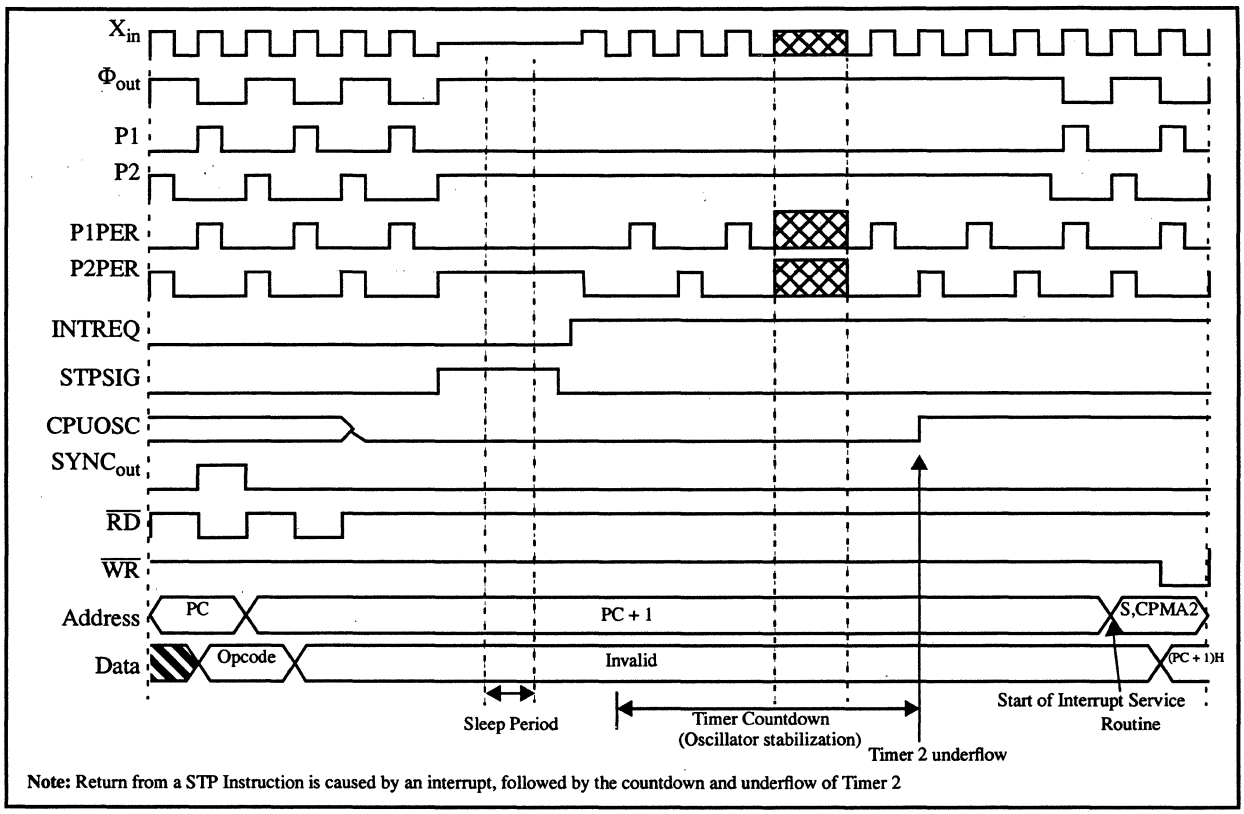


Figure 2-100. STP Cycle Timing Diagram

2.16.2 Wait Mode

Use of the wait mode allows the microcomputer to be placed in a state where excitation of the CPU is stopped, but the clocks to the peripherals continue to oscillate. This mode provides lower power dissipation during the idle periods and quick wake-up time. The microcomputer enters the wait mode when the WIT instruction is executed. After the instruction execution, P2 is held High and P1 is held Low.

Returning from wait mode is accomplished just as it is when returning from stop mode, with the exception that you need not provide time for the oscillator to stabilize, because the oscillation never stopped. Because P1PER and P2PER continue to oscillate in the wait mode, any peripheral interrupt can be used to bring the microcomputer out of the wait mode. The timing for the WIT instruction is shown in Figure 2-101.

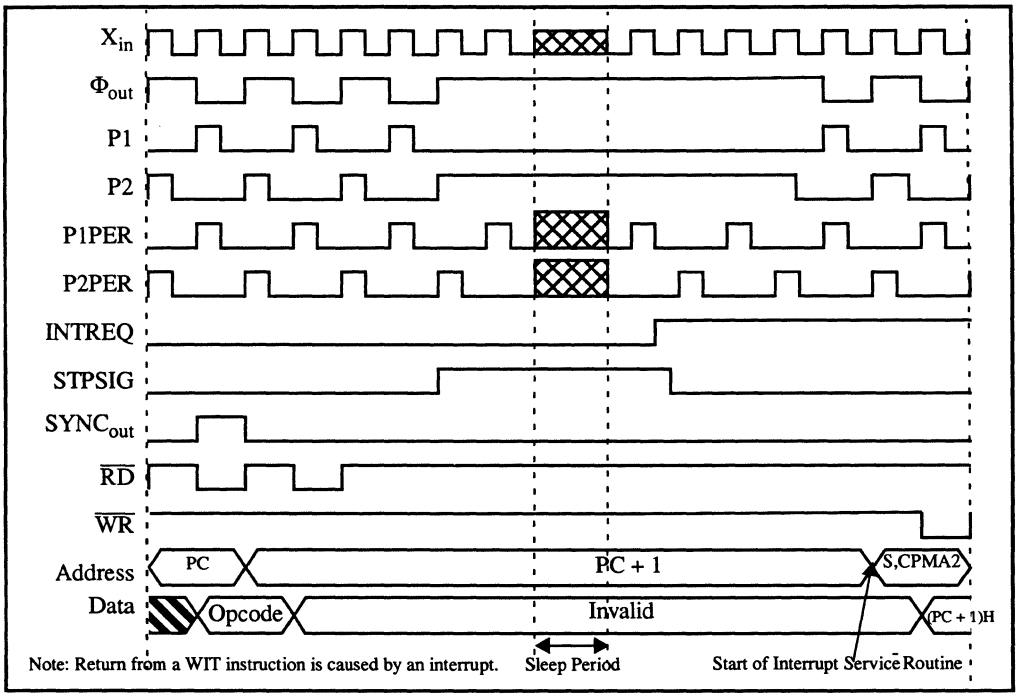


Figure 2-101. WIT Cycle Timing Diagram

2.17 Reset

This device is reset if the RESET pin is held Low for a minimum of 2μs while the supply voltage is between 4.5 and 5.5Volts. When the RESET pin returns High, the reset sequence commences (see Figure 2-102.) To allow the oscillation source the time to stabilize a delay is generated by the countdown of Timer 1 and Timer 2 cascaded with FF_{16} loaded in Timer 1 and 01_{16} loaded in Timer 2. After the reset sequence completes, program execution begins at the address whose high-order byte is the contents of address $FFFA_{16}$ and whose low-order byte is the contents of address $FFFB_{16}$.

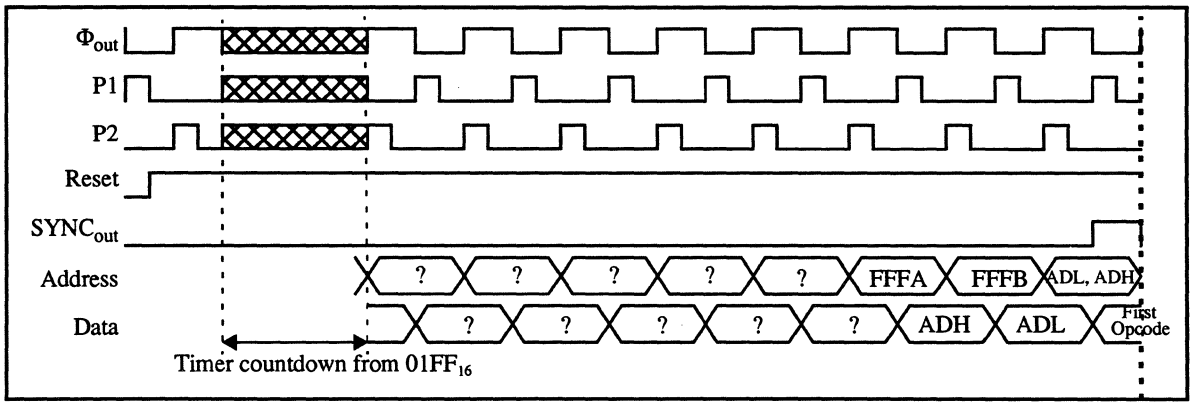


Figure 2-102. Internal Processing Sequence after RESET

2.18 Key-On Wake-Up

This device contains a key-on wake-up interrupt function. The key-on wake-up interrupt function is one way of returning from a power-down state caused by the STP or WIT instructions. This interrupt is generated by applying Low level to any pin of Port 2. If a key matrix is connected as shown in Figure 2-103., the microcomputer can be returned to a normal state by pressing any one of the keys.

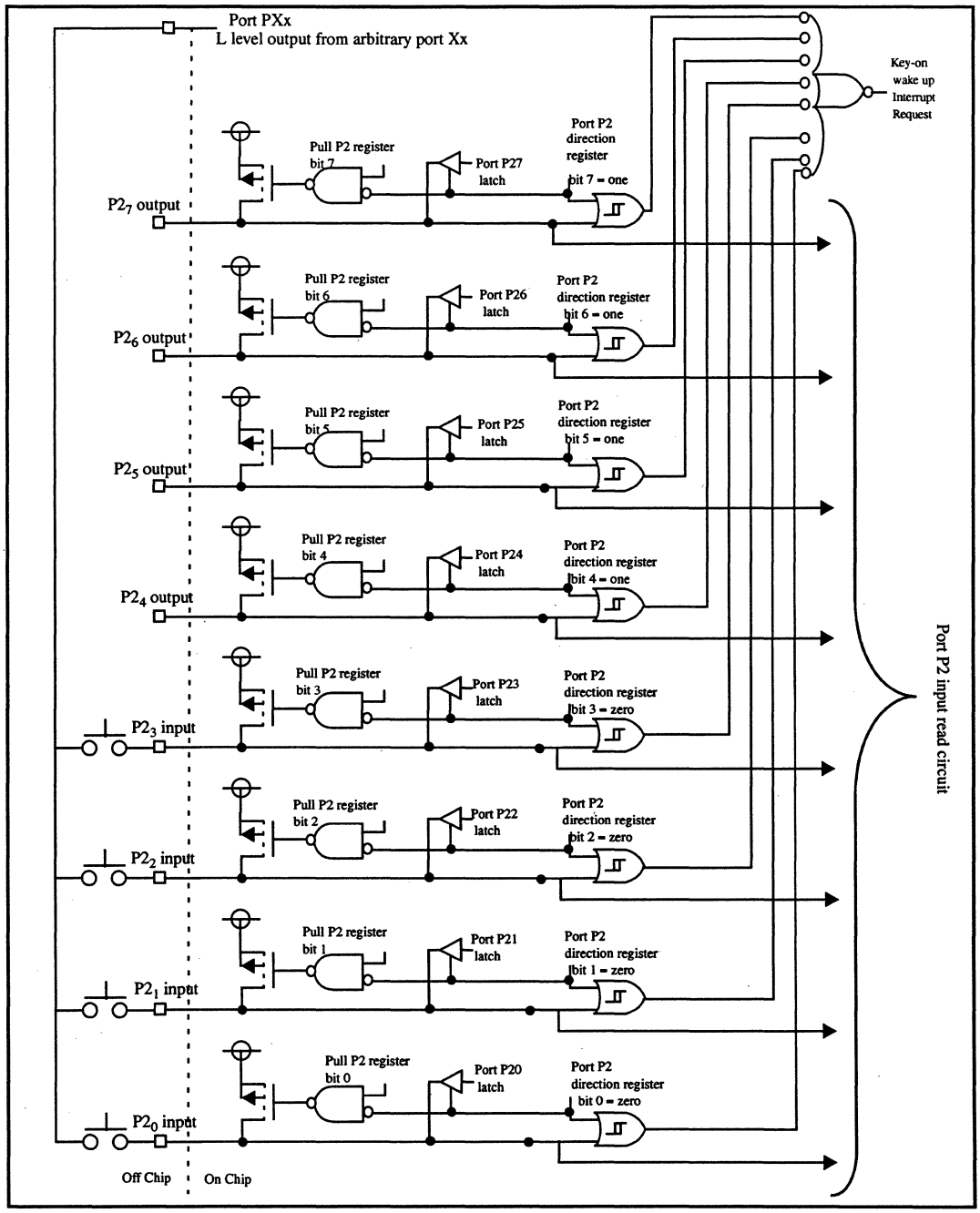


Figure 2-103. Port 2 with Key-ON Wake-Up Function

MITSUBISHI SEMICONDUCTOR
AMERICA, INC.

PRELIMINARY

Chapter 3

Electrical Characteristics

3.1 Absolute Maximum Rating. 97

3.2 Recommended Operating conditions 97

3.3 Electrical Characteristics 99

3 Electrical Characteristics

3.1 Absolute Maximum Rating

Table 3-1. Absolute Maximum Rating

Symbol	Parameter	Conditions	Limits	Unit
V_{cc}	Power Supply		-0.3 to 7.0	V
V_i	Input Voltage P0, P1, P2, P3, P4, P5, P6, P7, P8	Values are with respect to V_{ss}, output transistors are in off state.	-0.3 to Vcc + 0.3	V
V_i	Input Voltage RESET, X_{in}, XC_{in}		-0.3 to Vcc + 0.3	V
V_i	Input Voltage CNVSS		-0.3 to 13	V
V_o	Output Voltage P0, P1, P2, P3, P4, P5, P6, P7, P8, X_{out}, XC_{out}		-0.3 to Vcc + 0.3	SV
P_d	Power Dissipation	Ta = 25°C	750	mW
T_{opr}	Operating Temperature		-20 to +85°C	°c
T_{stg}	Storage Temperature		-40 to +125°C	°c

3.2 Recommended Operating conditions

Table 3-2. Recommended Operating Conditions (Vcc = 4.5 to 5.5V, Vss = 0V, Ta = -20 to 85°C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
V_{cc}	Supply voltage		4.5	5	5.5	V
V_{ss}	Supply voltage			0		V
V_{ih}	H input voltage	$\overline{RESET}$, X_{in}, XC_{in}, CNVSS	0.8Vcc		Vcc	V
V_{ih}	H input voltage	P0, P1, P2, P3, P4, P5, P6, P7, P8	0.8Vcc		Vcc	V
V_{il}	L input voltage	P0, P1, P2, P3,P 4, P5, P6, P7, P8	0		0.2Vcc	V
V_{il}	L input voltage	$\overline{RESET}$, X_{in}, XC_{in}, CNVSS	0		0.2Vcc	V
I_{ol} (peak)	L peak output current Note 1	P0,P1, P2, P3, P4, P5, P6, P7, P8			10	mA
I_{ol} (avg)	L average output current Note 2	P0, P1, P2, P3, P4, P5, P6, P7, P8			5	mA
I_{oh} (peak)	H peak output current Note 1	P0, P1, P2, P3, P4, P5, P6, P7, P8			-10	mA
I_{oh} (avg)	H average output current Note 2	P0, P1, P2, P3, P4, P5, P6, P7, P8			-5	mA

Table 3-2. Recommended Operating Conditions (Vcc = 4.5 to 5.5V, Vss = 0V, Ta = -20 to 85°C, unless otherwise noted)

ΣI_{ol} (peak	L total peak output current Note 3	P0,P 1, P2, P3, P4, P5, P6, P7, P8			80	mA
ΣI_{ol} (avg)	L total average output current Note 4	P0, P1, P2, P3, P4, P5, P6, P7, P8			40	mA
ΣI_{oh} (peak)	H total peak output current Note 3	P0, P1, P2, P3, P4, P5, P6, P7, P8			-80	mA
Symbol	**Parameter**		**Limits**			**Unit**
			Min.	**Typ.**	**Max.**	
ΣI_{oh} (avg)	H total average output current Note 4	P0, P1, P2, P3, P4, P5, P6, P7, P8			-40	mA
f(CNTR0)	TimerX - input frequency Note 5				2.6	MHz
f(CNTR1)	TimerY - input frequency Note 5				2.6	MHz
$f(X_{in})$	Clock frequency Note 5				48	MHz
$f(XC_{in})$	Clock frequency Note 5			TBD		MHz

Note 1. The peak output current is the peak current flowing through any pin of the listed ports.

Note 2. The average output current is an average current value measured over 100ms.

Note 3. The total peak output current is the peak current flowing through all pins of the listed ports.

Note 4. The total average output current is an average value measured over 100ms.

Note 5. The oscillation frequency has a 50% duty cycle.

3.3 Electrical Characteristics

Table 3-3. Electrical Characteristics (V_{cc} = 4.5 to 5.5V, V_{ss} = 0V, Ta = -20 to 85°C, unless otherwise noted)

Symbol	Parameter		Test Conditions	Limits			
				Min	Typ.	Max	Unit
Voh	H output current	P0, P1, P2,P3, P4,P5, P6, P7, P8	Ioh = -10mA	V_{cc}-2.0			V
Vol	L output current	P0, P1, P2, P3, P4, P5, P6, P7, P8	Iol = -10mA			2.0	V
Vt+ ~Vt-	Hysteresis	CNTR0, CNTR1, INT0, INT1			0.5		V
		URXD1, URXD2 (SCLK), $\overline{CTS2}$ (SRXD), SRDY, $\overline{CTS1}$, CNTR0, CNTR1, INT0, INT1			0.5		V
		$\overline{RESET}$			0.5		V
Iih	H input current	P0, P1, P2, P3, P4, P5, P6, P7, P8	$V_i = V_{cc}$			5	µA
		CNVSS, $\overline{RESET}$				5	µA
		X_{in}			4		µA
		XC_{in}			4		µA
Iil	L input current	P0, P1, P3, P4, P5, P6, P7, P8	$V_i = V_{ss}$			-5	µA
		P2	$V_i = V_{ss}$ (Pullups off)			-5	µA
			V_{cc} = 5V, $V_i = V_{ss}$ (Pullups on)		-70		µA
		$\overline{RESET}$, CNVSS	$V_i = V_{ss}$			-5	µA
		X_{in}			-4		µA
		XC_{in}			-4		µA
Vram	RAM retention voltage		with stopped clock	2.0		5.5	V
Icc	Supply current (Output transistors are in off state)	Normal Mode	Φ = 12MHz, V_{cc} = 5V, USB operating, frequency synthesizer on		150		mA
			Φ = 12MHz, V_{cc} = 5V, USB suspend, frequency synthesizer on		TBD		mA
			Φ = 12MHz, V_{cc} = 5V, USB suspend, frequency synthesizer off		TBD		mA
		Wait Mode	Φ = 12MHz, WIT mode, V_{cc} = 5V, USB operating, frequency synthesizer on		TBD		mA
			Φ = 12MHz, WIT mode, V_{cc} = 5V, USB suspend, frequency synthesizer on		TBD		mA
			Φ = 12MHz, WIT mode, V_{cc} = 5V, USB suspend, frequency synthesizer off		TBD		mA
		Stop Mode	T_a = 25°C, V_{cc} = 5V		0.1	1	µA
			T_a = 85°C, V_{cc} = 5V			10	µA

Table 3-4. Master CPU Bus Interface Timing (R and W Separation Type Mode)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_{su(S-R)}$	$\overline{S}$ Set-up Time		15		ns
$t_{su(S-W)}$	$\overline{S}$ Set-up Time		15		ns
$t_{h(R-S)}$	$\overline{S}$ Hold Time		10		ns
$t_{h(W-S)}$	$\overline{S}$ Hold Time		10		ns
$t_{su(A-R)}$	A0 Set-up Time		15		ns
$t_{su(A-W)}$	A0 Set-up Time		15		ns
$t_{h(R-A)}$	A0 Hold Time		10		ns
$t_{h(W-A)}$	A0 Hold Time		10		ns
$t_{w(R)}$	Read Pulse Width		60		ns
$t_{w(W)}$	Write Pulse Width		60		ns
$t_{su(D-W)}$	Date Input Set-up Time before Write		30		ns
$t_{h(W-D)}$	Date Input Hold Time after Write		10		ns
$t_{a(R-D)}$	Data Output Enable Time after Read		40		ns
$t_{v(R-D)}$	Data Output Disable Time after Read		30		ns
$t_{plh(R-OBF)}$	OBF Output Transmission Time after Read		50		ns
$t_{plh(W-IBF)}$	$\overline{IBF}$ Output Transmission Time after Write		50		ns

Table 3-5. Master CPU Bus Interface Timing (R/$\overline{W}$ Type Mode)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_{su(S-E)}$	$\overline{S}$ Set-up Time		15		ns
$t_{h(E-S)}$	$\overline{S}$ Hold Time		10		ns
$t_{su(A-E)}$	A0 Set-up Time		15		ns
$t_{h(E-A)}$	A0 Hold Time		10		ns
$t_{su(RW-E)}$	R/$\overline{W}$ Set-up Time		20		ns
$t_{h(E-RW)}$	R/$\overline{W}$ Hold Time		15		ns
$t_{w(E)}$	Enable pulse width		60		ns
$t_{w(E-E)}$	Enable pulse interval		60		ns
$t_{su(D-E)}$	Date Input Set-up Time before Write		60		ns
$t_{h(E-D)}$	Date Input Hold Time after Write		10		ns
$t_{a(E-D)}$	Data Output Enable Time after Read		40		ns
$t_{v(E-D)}$	Data Output Disable Time after Read		30		ns
$t_{plh(E-OBF)}$	OBF Output Transmission Time after $\overline{E}$ inactive		50		ns
$t_{plh(E-IBF)}$	$\overline{IBF}$ Output Transmission Time after $\overline{E}$ inactive		50		ns

Note: The timing information listed above is NOT derived from the device characterization, it is for reference only.

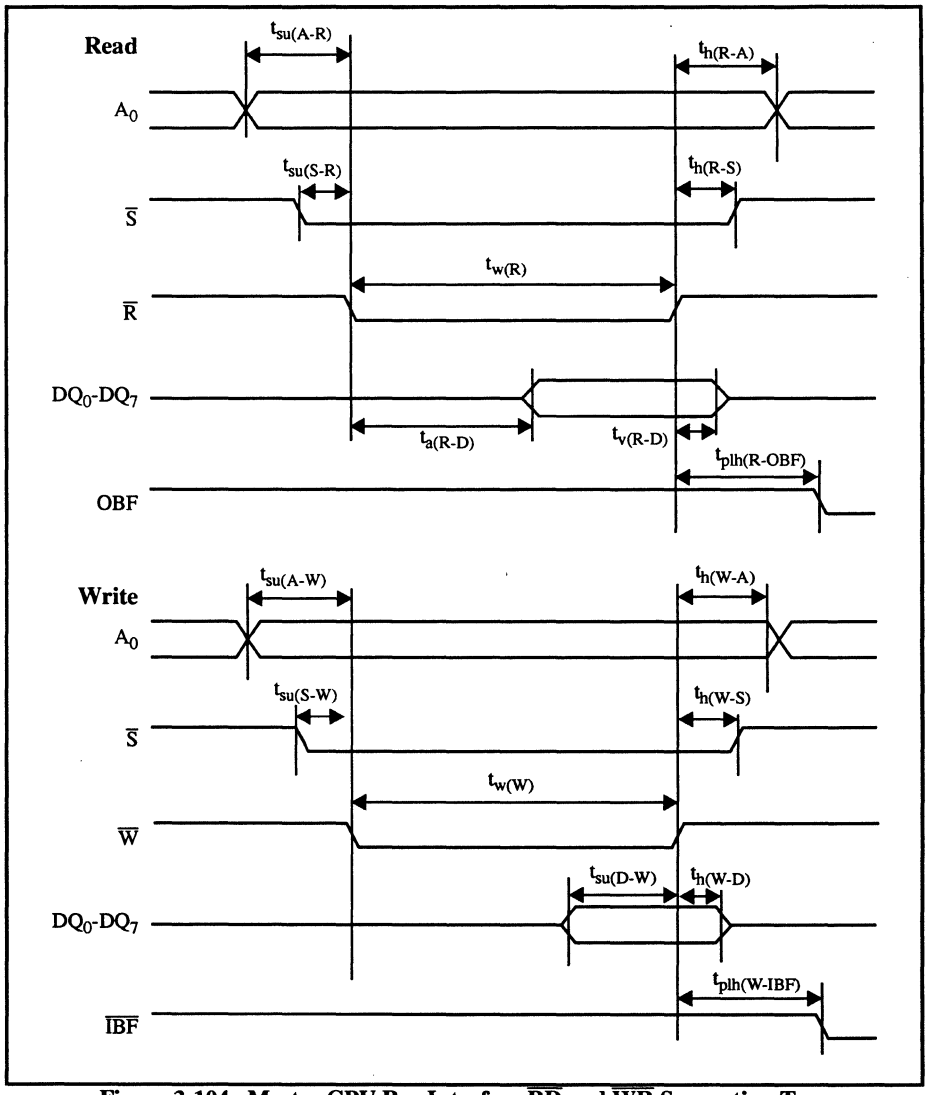


Figure 3-104. Master CPU Bus Interface $\overline{RD}$ and $\overline{WR}$ Separation Type

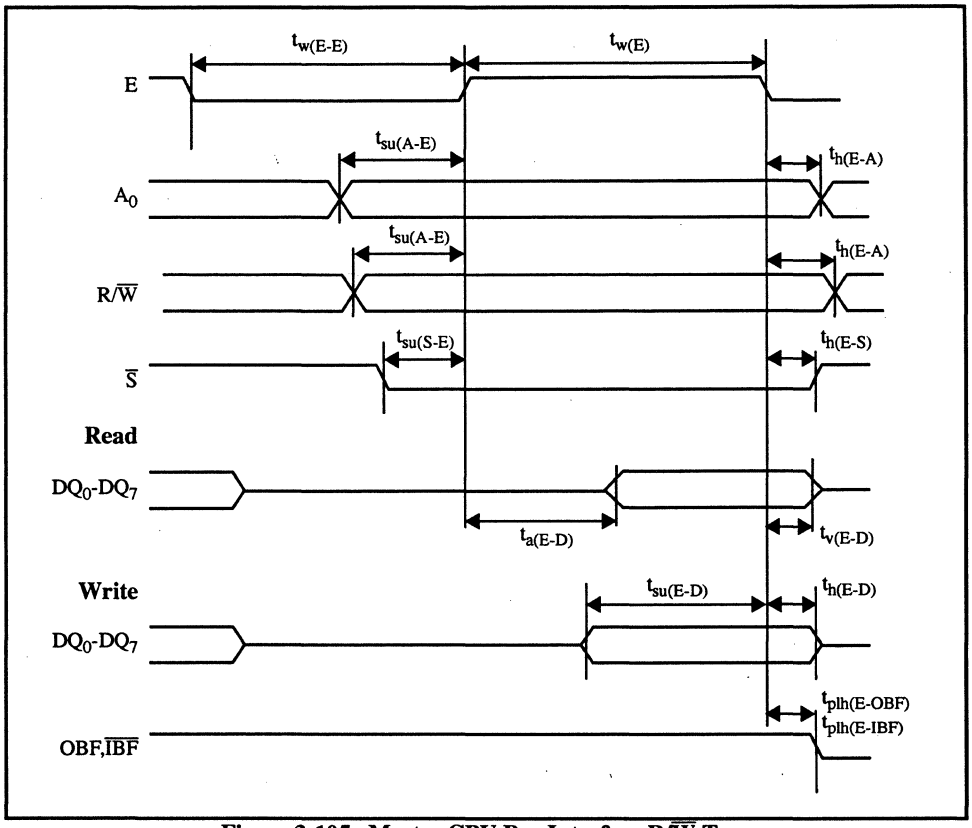


Figure 3-105. Master CPU Bus Interface R/$\overline{W}$ Type

MITSUBISHI SEMICONDUCTOR
AMERICA, INC.

PRELIMINARY

Chapter 4

Application Notes

4.1 DMAC 105

4.2 UART 106

4.3 Timer 106

4.4 Frequency Synthesizer Interface 107

4 Application Notes

4.1 DMAC

4.1.1 Programming

The following Programming notes should be adhered to for correct DMA operation:

- When using the software trigger to initiate a transfer request and the hardware requests are enabled (for example, RSD is zero), the user should follow the sequence given below:
- Disable the hardware requests by setting RSD to one.
- Trigger the software request and re-enable the hardware requests at the same time (for example, using the same assembly instruction) by writing a zero to RSD and a one to DSWT.

4.1.2 Application

The following is an example of how to set up the DMAC for interfacing with a peripheral block. In this case data is being transferred by the DMAC from the UART receive buffer to user RAM.

- Write 08_{16} to DMAM1 so that after each transfer the destination register will be decrements by one and the source register will remain unchanged.
- Write 00_{16} to the low-order byte of the destination register (DMADL) and 03_{16} to the high-order byte of the destination register (DMADH) so that the data received by the UART is placed in page three of the user RAM starting from address 0300_{16}.
- Write $3C_{16}$ to the low-order byte of the source register (DMASL) and 00_{16} to the high-order byte of the source register (DMASH) so that the DMA reads from address $003C_{16}$, which is the low-order byte of the UART receive buffer.
- Write to the transfer count register (DMAC) with an 8-bit value that corresponds to the number of transfers to occur before flag CRUF and the DMA interrupt are set.
- Set the DMA transfer initiating source to the UART receive interrupt by writing 00_{16} to DMAM2.
- Place the UART in the desired configuration for data reception by writing to the UART control (UCON), UART mode (UMOD), and UART baud rate (UBRG) registers.
- Disable the UART receive interrupt from being serviced by the CPU by clearing to a zero bit 4 of interrupt control register A (ICONA).
- Enable the DMA interrupt by setting bit 3 of ICONA to one.
- Enable the DMA controller and reset the initiating source sample latch by writing $C8_{16}$ to DMAM1.

The DMA controller will transfer one byte of data from the UART receive buffer to third page user RAM each time that the UART receive interrupt is set. Because the destination register is incremented by one after each transfer, third page user RAM is contiguously filled with received data.

The transfer count register is decrements by one after each transfer. When it underflows, flag CRUF and the DMA interrupt are set. In the DMA service routine, the user can either write new values to the source, destination, and transfer count registers, or leave these registers untouched. If they are left

untouched, then they contain the previously written values that were reloaded when the transfer count register underflowed. This would result in the previously transferred UART data in third page user RAM being overwritten with newly received UART data.

4.2 UART

4.2.1 Application

- 7 bit operation:
  When 7 bit data format is used, bit 7 of the transmit buffer register 1 is ignored. The transmit buffer register 2 does not affect the 7 or 8 bit format.
- 9 bit operation:
  The upper transmit/receive buffer register is a single bit register (bit 0). Writing to the upper bits in these registers has no affect. When reading the register the upper 7 bits are zero.

Note: The value in UBRG is not affected by a reset.

4.3 Timer

4.3.1 Usage

- For Timer X, if CNTR0 is read while Pulse Output mode is being used, the value returned is the pulse output signal fed from the timer to the port.
- For Timer Y, if CNTR1 is read while Pulse Output mode is being used, the value returned is the pulse output signal fed from the timer to the port.
- For Timer 1, if T_{out} is read while Pulse Output mode is being used, the value returned is the pulse output signal fed from the timer to the port.

Table 4-1.

Timer	Selection Bit	Initial Output Value
Timer Y	CNTR1 Polarity Select Bit (TXM6)	0: Logic H 1: Logic L
Timer X	CNTR0 Polarity Select Bit (TXM6)	0: Logic H 1: Logic L
Timer 1	T_{out} Output Active Edge Selection Bit (T123M5)	0: Logic H 1: Logic L

4.4 *Frequency Synthesizer Interface*

All passive components should be in close proximity to pin 86 (X_{out}/LPF). The recommended values are as follows:

Table 4-2. Recommended Values

40 MHz			20 MHz		
R=1000 Ω	1/8 watt	10%	R=1000 Ω	1/8 watt	10%
C2=640 pf	5 V	10%	C2=0.1µf	5 V	10%
C1=0.1 µf	5 V	10%	C1=1.0 µf	5 V	10%

See Figure 4-1 for a schematic of the LPF.

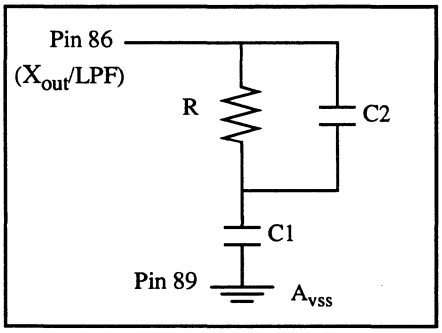


Figure 4-1. LPF Filter Schematic

Analog V_{ss} and Analog V_{dd}, pins 89 and 88 should have isolated connectors to the digital V_{ss} and V_{dd} ground planes. Figure 4-2 illustrates the power supply isolation. Depending on the card noise, the R=0 may have to change.

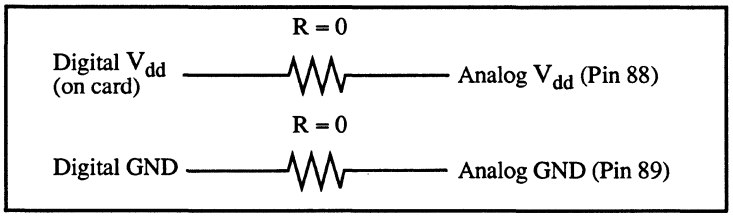


Figure 4-2. Power Supply Isolation

MITSUBISHI ELECTRONICS AMERICA, INC.
ELECTRONIC DEVICE GROUP

Direct Sales Offices

Northwest

Mitsubishi Electronics America, Inc.
1070 East Arques Avenue
Sunnyvale, CA 94086
Phone: (408) 730-5900
FAX: (408)737-1129

Mitsubishi Electronics America, Inc.
16101 SW 72nd Avenue, Suite 140
Portland, OR 97224
Phone: (503) 968-2727
FAX: (503) 968-2525

Southwest

Mitsubishi Electronics America, Inc.
1070 East Arques Avenue
Sunnyvale, CA 94086
Phone: (408) 730-5900
FAX: (408)737-1129

Rocky Mountain

Mitsubishi Electronics America, Inc.
1113 Spruce Street, Suite 503
Boulder, CO 80302
Phone: (303) 546-6300
FAX: (303)546-6242

North Central

Mitsubishi Electronics America, Inc.
9800 Bren Road East, Suite 243
Minnetonka, MN 55343-9671
Phone: (612) 938-7779
FAX: (612)938-5125

South Central

Mitsubishi Electronics America, Inc.
9000 Royal Lane
Irving, TX 75063
Phone: (214) 929-0046
FAX: (214)929-5363

Mitsubishi Electronics America, Inc.
8310 Capital of Texas Hwy North
Suite 260
Austin, TX 78731
Phone: (512) 346-4200
FAX:(512)346-4434

Mitsubishi Electronics America, Inc.
20405 SH249, Suite 170
Houston, TX 77070
Phone: 713-320-5959
FAX: 713-320-5960

Northeast

Mitsubishi Electronics America, Inc.
200 Unicorn Park Drive
Woburn, MA 01801
Phone: (617) 937-4300
FAX: (617) 938-1075

Mitsubishi Electronics America, Inc.
300 Westage Business Ctr, Suite 355
Fishkill, NY 12524
Phone: (914) 896-0896
FAX: (914) 896-8639

Mid-Atlantic

Mitsubishi Electronics America, Inc.
800 Cottontail Lane
Somerset, NJ 08873
Phone: (908) 469-8833
FAX: (908) 469-1909

South Atlantic

Mitsubishi Electronics America, Inc.
2500 Gateway Center Blvd., Suite 500
Cary, NC 27560
Phone: (919) 460-0404
FAX: (919)460-6205

Southeast

Mitsubishi Electronics America, Inc.
3100 Avalon Ridge, Suite 200
Norcross, GA 30071
Phone: (770) 613-5852
FAX: (770) 662-5208

Authorized Canadian Distributors

Toronto, Ottawa, Belleville and Eastern Canada

Mitsubishi Electronics America, Inc.
4299 14th Avenue
Markham, Ontario, Canada L3R 0J2
Phone: (905) 475-7728
FAX: (905)475-1918

Mitsubishi Electronics America, Inc.
340 March Road, Suite 300
Kanata, Ontario, Canada K2K 2B4
Phone: (613) 591-3348
FAX: (613)591-3948

Calgary and Western Canada

Mitsubishi Electronics America, Inc.
Airway Park Towers
2431 37th Ave. North East, Suite 230
Calgary, Alberta, Canada T2E 6Y7
Phone: (403) 291-2100
FAX: (403)291-1563

Mitsubishi Electronics America, Inc.
Electronic Device Group

Headquarters:
1050 East Arques Avenue
Sunnyvale, CA 940)16
Phone; (40)1)730-5900
FAX; (40)1) 732-9382

MITSUBISHI
ELECTRONIC DEVICE GROUP