

*Preliminary*

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

**Ver 1.04**



**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
Rev. 1.03 Internal Spec Updates	Jan. 22, 1998
Rev. 1.04 Design Spec Updates	June 2, 1998

This publication, or any parts thereof, may not be reproduced in any form without the prior written permission of Mitsubishi Semiconductor America, Inc. (MSAI).

The product(s) described in this publication are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which failure of the product could create a situation where personal injury or death may occur. Should Buyer purchase or use this product for any such unintended or unauthorized application, Buyer shall indemnify and hold MSAI and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that MSAI was negligent regarding the design and manufacture of the part.

Information supplied by MSAI is believed to be accurate and reliable. MSAI assumes no responsibility for any errors that may appear in this publication. MSAI reserves the right, without notice, to make changes in device design or specifications. Product is subject to availability.

©1997 Mitsubishi Semiconductor America, Inc.

## 1 Overview

1.1 MCU Features .....	1-5
1.2 Pin Description and Layout .....	1-6

## 2 Functional Description

2.1 Central Processing Unit .....	2-3
2.1.1 Register Structure .....	2-3
2.1.2 Accumulator (A) .....	2-3
2.1.3 Index Registers X and Y .....	2-4
2.1.4 Stack Pointer .....	2-4
2.1.5 Program Counter .....	2-4
2.1.6 Processor Status Register .....	2-5
2.2 CPU Mode Registers .....	2-7
2.3 Oscillator Circuit .....	2-8
2.3.1 Description .....	2-8
2.3.2 Frequency Synthesizer Circuit .....	2-11
2.4 Memory Map .....	2-14
2.4.1 Zero page .....	2-15
2.4.2 Special Page .....	2-15
2.4.3 Special Function Registers .....	2-15
2.5 Processor Modes .....	2-17
2.5.1 Single Chip .....	2-17
2.5.2 Memory Expansion .....	2-18
2.5.3 Microprocessor .....	2-18
2.5.4 EPROM .....	2-18
2.5.5 Slow Memory Wait .....	2-19
2.5.6 Hold Function .....	2-23
2.5.7 Expanded Data Memory Access .....	2-23
2.6 Peripheral Interface .....	2-25
2.6.1 Chip Bus Timing .....	2-25
2.6.2 Peripheral Interface and Access Timing .....	2-26
2.7 Input and Output Ports .....	2-28
2.7.1 Ports .....	2-28
2.7.1.1 I/O Ports .....	2-29
2.7.1.2 Power and Ground Pins .....	2-40
2.7.1.3 CNV <sub>SS</sub> Pin .....	2-40
2.7.1.4 X <sub>in</sub> and X <sub>out</sub> Pins .....	2-40
2.7.1.5 X <sub>Cin</sub> and X <sub>Cout</sub> Pins .....	2-40
2.7.1.6 RESET Pin .....	2-40
2.7.1.7 RDY Pin .....	2-41
2.7.1.8 DMA <sub>out</sub> Pin .....	2-41
2.7.1.9 F <sub>out</sub> Pin .....	2-41
2.7.1.10 SYNC <sub>out</sub> Pin .....	2-41
2.7.1.11 RD and WR Pins .....	2-41
2.7.1.12 LPF Pin .....	2-41
2.7.1.13 USB D+/D- Pins .....	2-41
2.7.1.14 Ext. Cap Pin .....	2-41
2.7.2 Port Control Register .....	2-42
2.7.3 Port 2 Pull-up Control Register .....	2-42
2.8 Interrupt Control Unit .....	2-43
2.8.1 Interrupt Control .....	2-43
2.8.2 Interrupt Sequence and Timing .....	2-47
2.9 Universal Serial Bus .....	2-49
2.9.1 USB Function Control Unit (USB FCU) .....	2-50
2.9.1.1 Serial Interface Engine .....	2-50
2.9.1.2 Generic Function Interface .....	2-50
2.9.1.3 Serial Engine Interface Unit .....	2-50
2.9.1.4 Microcontroller Interface Unit .....	2-50
2.9.1.5 USB Transceiver .....	2-50
2.9.2 USB Interrupts .....	2-51
2.9.2.1 USB Function Interrupt .....	2-51
2.9.2.2 USB SOF Interrupt .....	2-52
2.9.3 USB Endpoint FIFOs .....	2-52
2.9.3.1 IN (Transmit) FIFOs .....	2-52
2.9.3.2 Out (Receive) FIFOs .....	2-53
2.9.4 USB Special Function Registers .....	2-54
2.10 Master CPU Bus Interface .....	2-65
2.10.1 Data Bus Buffer Status Registers (DBBS0, DBBS1) .....	2-68
2.10.2 Input Data Bus Buffer Registers (DBBIN0, DBBIN1) .....	2-68
2.10.3 Output Data Bus Buffer Registers (DBBOUT0, DBBOUT1) .....	2-68
2.11 Direct Memory Access Controller .....	2-69
2.11.1 Operation .....	2-70
2.11.1.1 Source, Destination, and Transfer Count Register Operation .....	2-71
2.11.1.2 DMAC Transfer Request Sources .....	2-71
2.11.1.3 Transfer Features for USB and MBI ..	2-72
2.11.1.4 DMAC Transfer Mode .....	2-74
2.11.1.5 DMAC Transfer Timing .....	2-74
2.11.2 Special Count Source Generator .....	2-79
2.11.2.1 SCSG Operation .....	2-79
2.11.2.2 SCSG Description .....	2-80
2.11.2.2.1 SCSG1 .....	2-80
2.11.2.2.2 SCSG2 .....	2-80
2.12 Timers .....	2-82
2.12.1 Timer X .....	2-82
2.12.1.1 Read and Write Method .....	2-82
2.12.1.2 Count Stop Control .....	2-83
2.12.1.3 Timer Mode .....	2-83
2.12.1.4 Pulse Output Mode .....	2-83
2.12.1.5 Event Counter Mode .....	2-84
2.12.1.6 Pulse Width Measurement Mode .....	2-84
2.12.2 Timer Y .....	2-84
2.12.2.1 Read and Write Method .....	2-85
2.12.2.2 Count Stop Control .....	2-85
2.12.2.3 Timer Mode .....	2-85
2.12.2.4 Pulse Period Measurement Mode .....	2-86
2.12.2.5 Event Counter Mode .....	2-86
2.12.2.6 HL Pulse-width Measurement Mode ..	2-86
2.12.3 Timer 1 .....	2-87
2.12.3.1 Timer Mode .....	2-87
2.12.3.2 Pulse Output Mode .....	2-87
2.12.4 Timer 2 .....	2-88
2.12.4.1 Timer Mode .....	2-88
2.12.4.2 Pulse Output Mode .....	2-88

2.13.5 Timer 3 .....	2-88
2.13.5.1 Timer Mode .....	2-88
2.14 UART .....	2-90
2.14.1 Baud Rate Selection.....	2-91
2.14.2 UART Mode Register.....	2-93
2.14.3 UART Control Register.....	2-94
2.14.4 UART Baud Rate Register .....	2-94
2.14.5 UART Status Register .....	2-94
2.14.6 Transmit/Receive Format .....	2-96
2.14.7 Interrupts.....	2-98
2.14.8 Clear-to Send (CTSx) and Request-to-Send (RTSx) Signals.....	2-99
2.14.9 UART Address Mode.....	2-100
2.15 Serial I/O .....	2-102
2.15.1 SIO Control Register .....	2-102
2.15.2 SIO Operation.....	2-102
2.16 Low Power Modes.....	2-105
2.16.1 Stop Mode.....	2-105
2.16.2 Wait Mode .....	2-106
2.17 Reset .....	2-107
2.18 Key-On Wake-Up.....	2-108

### **3 Electrical Characteristics**

3.1 Absolute Maximum Ratings.....	3-3
3.2 Recommended Operating conditions .....	3-4
3.3 Electrical Characteristics .....	3-6
3.4 Timing Requirements and Switching Characteristics .....	3-8

### **4 Application Notes**

4.1 DMAC .....	4-3
4.1.1 Application .....	4-3
4.2 UART .....	4-4
4.2.1 Application .....	4-4
4.3 Timer .....	4-5
4.3.1 Usage .....	4-5
4.4 Frequency Synthesizer Interface .....	4-6
4.5 USB Transceiver .....	4-7
4.6 Ports.....	4-8
4.7 Programming Notes.....	4-9

### **5 Register List**

---



**PRELIMINARY**

# **Chapter 1**

## **PRODUCT DESCRIPTION**

1 Overview . . . . .	1-3
1.1 MCU Features . . . . .	1-5
1.2 Pin Description and Layout . .	1-6





# 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. See Figure 1-1 for an application system diagram.

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.

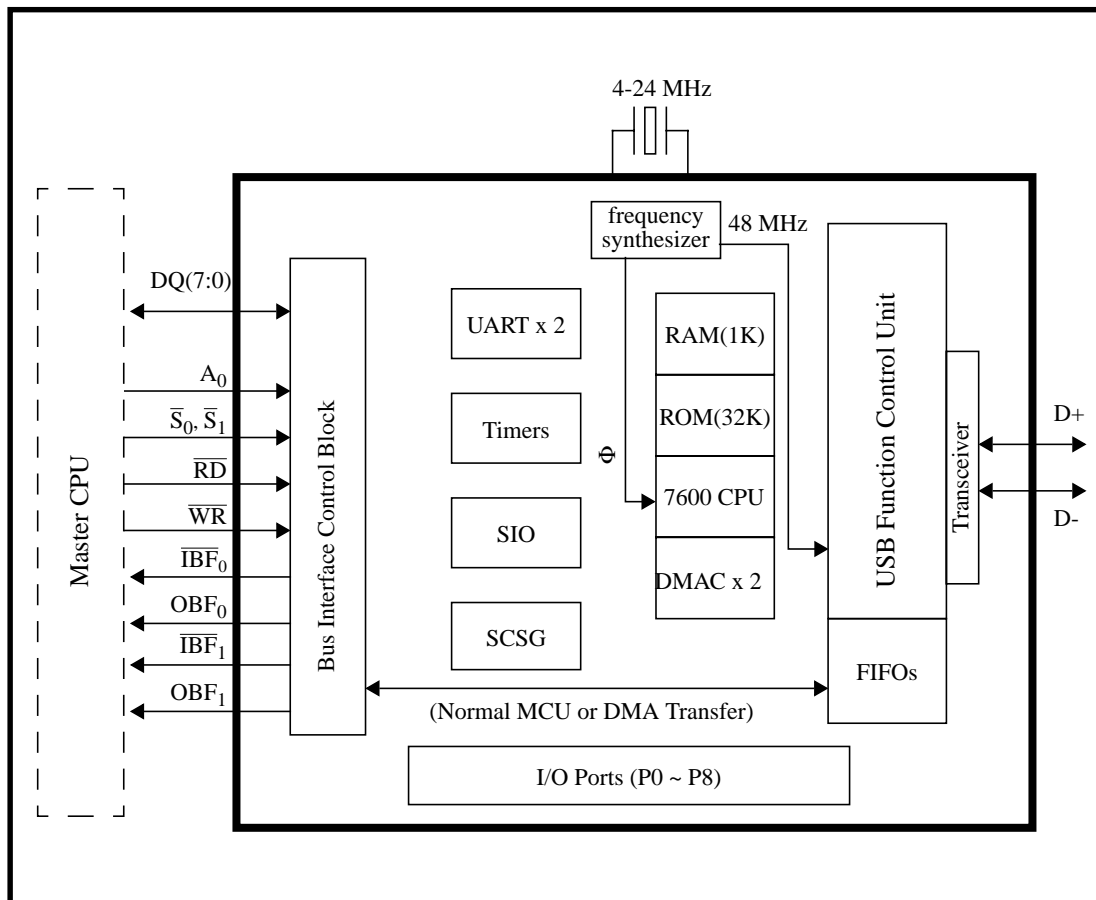


Figure 1-1. Application System Diagram

Table 1-1. Device Feature List

Parameter		Function Description	
Number of basic instructions		71	
Instruction execution time (minimum)		83ns at $\Phi = 12$ MHz (setting $\Phi$ to less than 5MHz is NOT recommended)	
Clock frequency (maximum)		$X_{in} = 48$ MHz, $XC_{in} = 5$ MHz (square wave), $\Phi = 12$ MHz	
Clock multiplier option		External clock $X_{in}$ and $XC_{in}$ can be selectively divided and multiplied by X to create system internal clock $\Phi$	
Memory size	ROM	32K bytes	
	RAM	1K bytes	
Input/Output ports	P0~P3, P5, P6, P8	I/O	8-bit X 7 (Port 2 has a key-on wake-up feature)
	P4, P7	I/O	5-bit X 2
USB Function Control		FIFO: 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	
Master CPU bus interface		DQ(7:0), $\bar{R}(E)$ , $\bar{W}(R/\bar{W})$ , $\bar{S}_0$ , $\bar{S}_1$ , $A_0$ , $\bar{IBF}_0$ , $OBF_0$ , $\bar{IBF}_1$ , $OBF_1$ ; total of 17 signals interface with master CPU (Intel 8042-like interface)	
Special Count Source Generator(SCSG)		Baud rate synthesizer	
UART X 2		7/8/9-bit character length, with $\bar{CTS}$ , $\bar{RTS}$ available	
Serial I/O		8-bit clock synchronous serial I/O, supports both master and slave modes	
Timers		8-bit X 3, 16-bit X 2	
DMAC		2 channels, 16 address lines, support single byte or burst transfer modes	
Software selectable slew rate control		Ports P0 ~ P8	
Interrupts		4 external, 19 internal, 1 software, 1 system interrupts	
Supply voltage		$V_{cc} = 4.15 \sim 5.25$ V	
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
- Minimum instruction execution time of 83ns (1-cycle instruction @  $\Phi = 12$  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 selectable 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 and two 16-bit timers available
- 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	Pin #
P0 <sub>0</sub> /AB0 ~ P1 <sub>7</sub> /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.	56-41
P2 <sub>0</sub> /DB0 ~ P2 <sub>7</sub> /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.	64-57
P3 <sub>0</sub> /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).	40
P3 <sub>1</sub>	I/O	CMOS I/O port.	39
P3 <sub>2</sub> /(VRFY)	I/O	CMOS I/O port. When the MCU is in EPROM program mode, the pin is used as VRFY (EPROM memory verify).	38
P3 <sub>3</sub> /DMA <sub>out</sub> /PGM	I/O	CMOS I/O port (DMA <sub>out</sub> ). When the MCU is in memory expansion or microprocessor mode, this pin is set to a "1" during a DMA transfer. When the MCU is in EPROM program mode, the pin is used as PGM (EPROM memory program).	37
P3 <sub>4</sub> /Φ <sub>out</sub>	I/O	CMOS I/O port (Φ). When the MCU is in memory expansion or microprocessor mode, this pin becomes Φ <sub>out</sub> pin.	36
P3 <sub>5</sub> /SYNC <sub>out</sub>	I/O	CMOS I/O port (SYNC output). When the MCU is in memory expansion or microprocessor mode, this pin becomes the SYNC <sub>out</sub> pin.	35
P3 <sub>6</sub> /WR/(CE)	I/O	CMOS I/O port. (WR output). When the MCU is in memory expansion or microprocessor mode, this pin becomes WR. When the MCU is in EPROM program mode, the pin is used as CE (EPROM memory chip enable).	34
P3 <sub>7</sub> /RD/(OE)	I/O	CMOS I/O port. (RD output). When the MCU is in memory expansion or microprocessor mode, this pin becomes RD. When the MCU is in EPROM program mode, the pin is used as OE (EPROM memory output enable).	33
P4 <sub>0</sub> /EDMA	I/O	CMOS I/O port (EDMA: Expanded Data Memory Access). When the MCU is in memory expansion or microprocessor mode, this pin can become the EDMA pin.	24
P4 <sub>1</sub> /INT0 ~ P4 <sub>2</sub> /INT1	I/O	CMOS I/O port or external interrupt ports INT0 and INT1. These external interrupts can be configured to be active high or low.	23-22
P4 <sub>3</sub> /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. The interrupt polarity is selected in the Timer X mode register.	21
P4 <sub>4</sub> /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. The interrupt polarity is selected in the Timer Y mode register.	20
P5 <sub>0</sub> /XC <sub>in</sub>	I/O	CMOS I/O port or XC <sub>in</sub> .	12
P5 <sub>1</sub> /T <sub>out</sub> / XC <sub>out</sub>	I/O	CMOS I/O port or Timer 1/2 pulse output pin (can be configured initially high or initially low), or XC <sub>out</sub> .	11
P5 <sub>2</sub> /OBF <sub>0</sub>	I/O	CMOS I/O port or OBF <sub>0</sub> output to master CPU for data bus buffer 0.	8
P5 <sub>3</sub> /IBF <sub>0</sub>	I/O	CMOS I/O port or IBF <sub>0</sub> output to master CPU for data bus buffer 0.	7
P5 <sub>4</sub> /S <sub>0</sub>	I/O	CMOS I/O port or S <sub>0</sub> input from master CPU for data bus buffer 0.	6
P5 <sub>5</sub> /A <sub>0</sub>	I/O	CMOS I/O port or A <sub>0</sub> input from master CPU.	5
P5 <sub>6</sub> /R(E)	I/O	CMOS I/O port or R(E) input from master CPU.	4
P5 <sub>7</sub> /W(R/W)	I/O	CMOS I/O port or W(R/W) input from master CPU.	3
P6 <sub>0</sub> /DQ0 ~ P6 <sub>7</sub> /DQ7	I/O	CMOS I/O port or master CPU data bus.	2-1, 80-75
USB D <sup>-</sup>	I/O	USB D <sup>-</sup> voltage line interface, a series resistor of 33 Ω should be connected to this pin. (see note)	71
USB D <sup>+</sup>	I/O	USB D <sup>+</sup> voltage line interface, a series resistor of 33 Ω should be connected to this pin. (see note)	70
P7 <sub>0</sub> /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.	69
P7 <sub>1</sub> /HOLD	I/O	CMOS I/O port or HOLD pin.	68
P7 <sub>2</sub> /S <sub>1</sub>	I/O	CMOS I/O port or S <sub>1</sub> input from master CPU for data bus buffer 1.	67

Table 1-2. Pin Description

Name	I/O	Description	Pin #
P7 <sub>3</sub> /IBF <sub>1</sub> / HLDA	I/O	CMOS I/O port or IBF <sub>1</sub> output to master CPU for data bus buffer 1, or HLDA pin. IBF <sub>1</sub> and HLDA are mutually exclusive. IBF <sub>1</sub> has priority over HLDA.	66
P7 <sub>4</sub> /OBF <sub>1</sub>	I/O	CMOS I/O port or OBF <sub>1</sub> output to master CPU for data bus buffer 1.	65
P8 <sub>0</sub> /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.	32
P8 <sub>1</sub> /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.	31
P8 <sub>2</sub> /CTS2/ SRXD	I/O	CMOS I/O port or UART2 pin CTS2 or SIO pin SRXD. UART2 and SIO are mutually exclusive, UART2 has priority over SIO.	30
P8 <sub>3</sub> /RTS2/ STXD	I/O	CMOS I/O port or UART2 pin RTS2 or SIO pin STXD. UART2 and SIO are mutually exclusive, UART2 has priority over SIO.	29
P8 <sub>4</sub> /UTXD1	I/O	CMOS I/O port or UART1 pin UTXD1.	28
P8 <sub>5</sub> /URXD1	I/O	CMOS I/O port or UART1 pin URXD1.	27
P8 <sub>6</sub> /CTS1	I/O	CMOS I/O port or UART1 pin CTS1.	26
P8 <sub>7</sub> /RTS1	I/O	CMOS I/O port or UART1 pin RTS1.	25
AV <sub>cc</sub> , AV <sub>ss</sub>	I	Power supply inputs for analog circuitry AV <sub>cc</sub> = 4.15~ 5.25V, AV <sub>ss</sub> = 0V	17,19
CNV <sub>ss</sub>	I	Controls the processor mode of the chip. Normally connected to V <sub>ss</sub> or V <sub>cc</sub> . When the MCU is in EPROM program mode, this pin supplies the programming voltage to the EPROM.	9
V <sub>cc</sub> , V <sub>ss</sub>	I	Power supply inputs: V <sub>cc</sub> = 4.15~ 5.25V, V <sub>ss</sub> = 0V	16/74, 13/73
RESET	I	To enter the reset state, this pin must be kept L for more that 2μs (20 Φ cycles under normal V <sub>cc</sub> conditions). If the crystal or ceramic resonator requires more time to stabilize, extend this L level time appropriately.	10
XC <sub>in</sub> XC <sub>out</sub>	I O	An external ceramic or quartz crystal oscillator can be connected between the XC <sub>in</sub> and XC <sub>out</sub> pins. If an external clock source is used, connect the clock source to the XC <sub>in</sub> pin and leave the XC <sub>out</sub> pin open.	12 11
X <sub>in</sub> X <sub>out</sub>	I O	Input and output signals to and from the internal clock generation circuit. Connect a ceramic resonator or quartz crystal between X <sub>in</sub> and X <sub>out</sub> pins to set the oscillation frequency. If an external clock is used, connect the clock source to the X <sub>in</sub> pin and leave the X <sub>out</sub> pin open.	14 15
LPF	O	Loop filter for the frequency synthesizer.	18
Ext. Cap	I	An external capacitor (Ext. Cap) pin. When the USB transceiver voltage converter is used, a 2μf or larger capacitor should connect between this pin and V <sub>ss</sub> to ensure proper operation of the USB line driver. The voltage converter is enabled by setting bit 4 of the USB control register (0013 <sub>16</sub> ) to a "1".	72

**D+/D- Line driver notes:** In order to match the USB cable impedance, a series resistor of 33Ω, 1%, 1/8 W should be connected to each USB line; i.e. on D+ (pin 70) and on D- (pin 71). Also, a coupling capacitor with the recommended value of 33pF should be connected between D+ and D- after the 33Ω series resistors. 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.

**VDD/VSS notes:** In order to reduce the effects of the inductance of the traces on the board, decoupling capacitors should be connected between pins 73(VSS) and 74(VDD), 13(VSS) and 16(VDD), 17(VSS) and 19(AVDD), and 17(AVDD) and 19(AVSS). Recommended values are a 4.7 μF in parallel with a 0.1 μF.

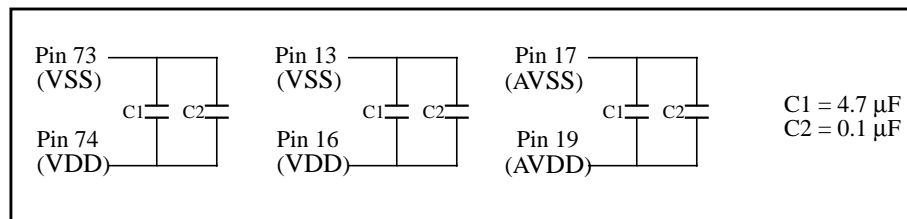


Figure 1-3. VDD/VSS decoupling capacitor connections



**PRELIMINARY**

# **Chapter 2**

## **Functional Description**

2.1 Central Processing Unit . . . . .	2-3
2.2 CPU Mode Registers . . . . .	2-7
2.3 Oscillator Circuit . . . . .	2-8
2.4 Memory Map . . . . .	2-14
2.5 Processor Modes . . . . .	2-17
2.6 Peripheral Interface . . . . .	2-25
2.7 Input and Output Ports . . . . .	2-28
2.8 Interrupt Control Unit . . . . .	2-43
2.9 Universal Serial Bus . . . . .	2-49
2.10 Master CPU Bus Interface . . . . .	2-65
2.11 Direct Memory Access Controller . . . . .	2-69
2.12 Special Count Source Generator . . . . .	2-79
2.13 Timers . . . . .	2-82
2.14 UART . . . . .	2-90
2.15 Serial I/O . . . . .	2-102
2.16 Low Power Modes . . . . .	2-105
2.17 Reset . . . . .	2-107
2.18 Key-On Wake-Up . . . . .	2-108





# 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

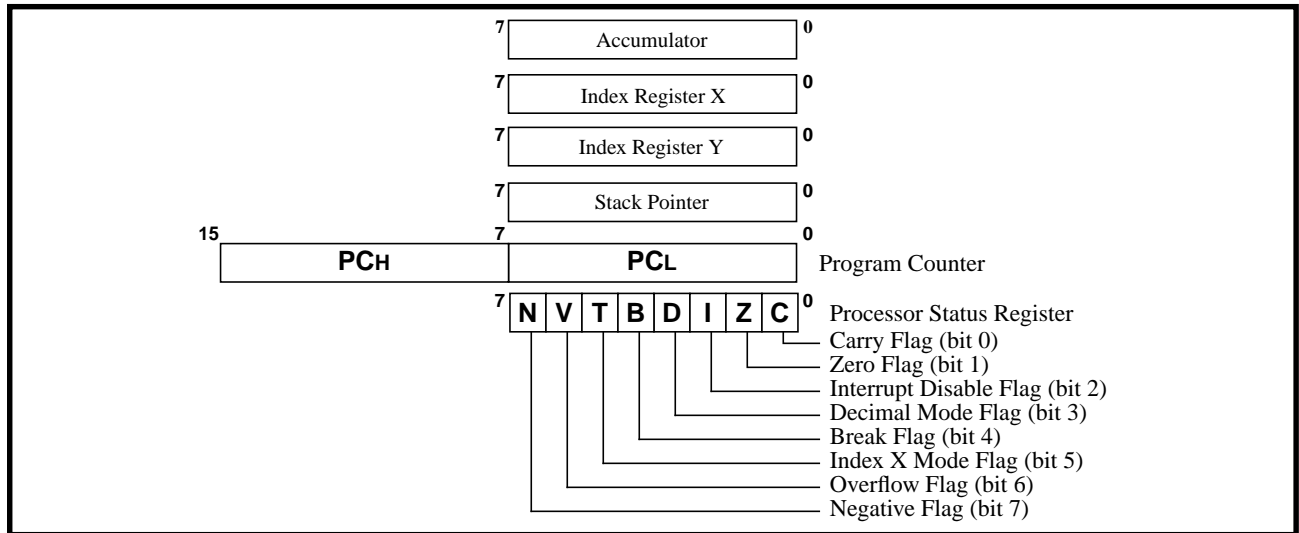


Figure 2-1. Register Structure

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

The Program counter (PC) is a 16-bit register consisting of two 8-bit registers (PCH and PCL) (see Figure 2-1.).

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

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 "1" (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 2-2.

### 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.



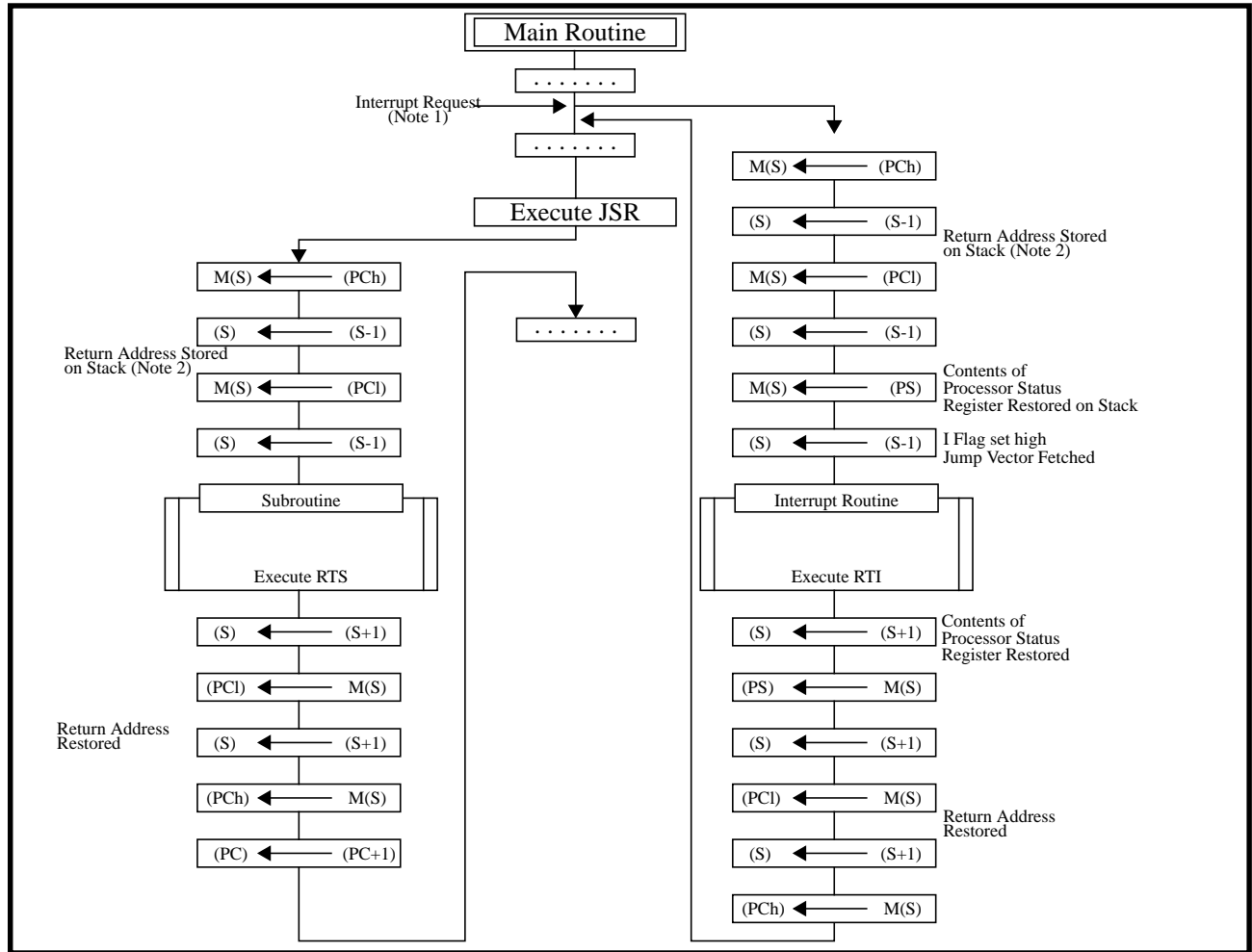


Figure 2-2. Register Push and Pop when Servicing Interrupts and Calling Subroutines

**Note 1.** The condition to enable an interrupt: Interrupt enable bit is set to a “1” and Interrupt inhibit flag (I flag) is a “0”.

**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.

## 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 to a “1”, 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 “1”. When an interrupt occurs, this flag is automatically set to a “1” 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 “0”; decimal arithmetic is executed when it is “1”. 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 “0”. When the BRK instruction is used to generate an interrupt, the processor status register is pushed onto the stack with the break flag set to a “1”. 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 “1”, 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 “1”). 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	Acronym and Value at Reset
0000 <sub>16</sub>	CPU mode register A	CPMA=0C
0001 <sub>16</sub>	CPU mode register B	CPMB=83

This device has two CPU mode registers: CPU Mode Register A (CPMA) and CPU Mode Register B (CPMB) 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	CPMA7	CPMA6	CPMA5	CPMA4	CPMA3	CPMA2	CPMA1	CPMA0	LSB 0	Address: 0000 <sub>16</sub> Access: R/W Reset: 0C <sub>16</sub>
	CPMA0,1 Processor Mode Bits (bits 1,0)									
				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 <sub>out</sub> Drive Capacity Selection Bit (bit 3)									
				0: Low						
				1: High						
	CPMA4 Clock X <sub>C<sub>in</sub></sub> -X <sub>C<sub>out</sub></sub> Stop Bit (bit 4)									
				0: Stop						
				1: Oscillator						
	CPMA5 Clock X <sub>in</sub> -X <sub>out</sub> Stop Bit (bit 5)									
				0: Oscillator						
				1: Stop						
	CPMA6 Internal Clock Selection Bit (bit 6)									
				0: External Clock						
				1: f <sub>syn</sub>						
	CPMA7 External Clock Selection Bit (bit 7)									
				0: X <sub>in</sub> -X <sub>out</sub>						
				1: X <sub>C<sub>in</sub></sub> -X <sub>C<sub>out</sub></sub>						

Figure 2-3. CPU Mode Register A

MSB 7	CPMB7	Reserved	CPMB5	CPMB4	CPMB3	CPMB2	CPMB1	CPMB0	LSB 0	Address: 0001 <sub>16</sub> Access: R/W Reset: 83 <sub>16</sub>
	CPMB0,1 Slow Memory Wait Bits (bits 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 Slow Memory Mode Bit (bits 3,2)									
				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: EDMA output disabled (64 Kbyte data access area)						
				1: 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 (Read/Write "0")									
	CPMB7 X <sub>out</sub> 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 Oscillator Circuit

### 2.3.1 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 flow diagram for the oscillator circuit is shown in Figure 2-6 and a block diagram of the oscillator circuit is shown in Figure 2-7. The following external clock inputs are supported:

- A quartz crystal oscillator of up to 24 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  $XC_{in}$ .

The two-phase non-overlapping system clock (CPU and peripherals) is derived from the source to the clock circuit and is 1/2 the frequency of the source. (i.e. Source = 24 MHz, system clock = 12 MHz) Any one of four clock signals can be chosen as the source for the system and peripheral clocks;  $f_{Xin}/2$ ,  $f_{XCin}$ ,  $f_{SYN}$ . 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  $f_{Xin} = 24\text{MHz}$ , then the CPU will be running at  $\Phi = 6\text{MHz}$  (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	CCR7	CCR6	CCR5	CCR4	Reserved	Reserved	Reserved	Reserved	LSB 0	Address: 001F <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
	Bits 0-3	Reserved (Read/Write “0”)								
	CCR4:	PLL Bypass Bit (bit 4) 0: $f_{USB} = f_{VCO}$ (Frequency synthesizer output) 1: $f_{USB} = f_{Xin}$								
	CCR5:	$XC_{out}$ Oscillation Drive Disable Bit (bit 5) 0: $XC_{out}$ oscillation drive is enabled (when $XC_{in}$ oscillation is enabled). 1: $XC_{out}$ oscillation drive is disabled.								
	CCR6:	$X_{out}$ Oscillation Drive Disable Bit (bit 6) 0: $X_{out}$ oscillation drive is enabled (when $X_{in}$ oscillation is enabled). 1: $X_{out}$ oscillation drive is disabled.								
	CCR7:	$X_{in}$ Divider Select Bit (bit 7) 0: $f_{Xin}/2$ is used for the system clock source when CMA7:6=00 1: $f_{Xin}$ is used for the system clock source when CMA7:6=00								

Figure 2-5. 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.

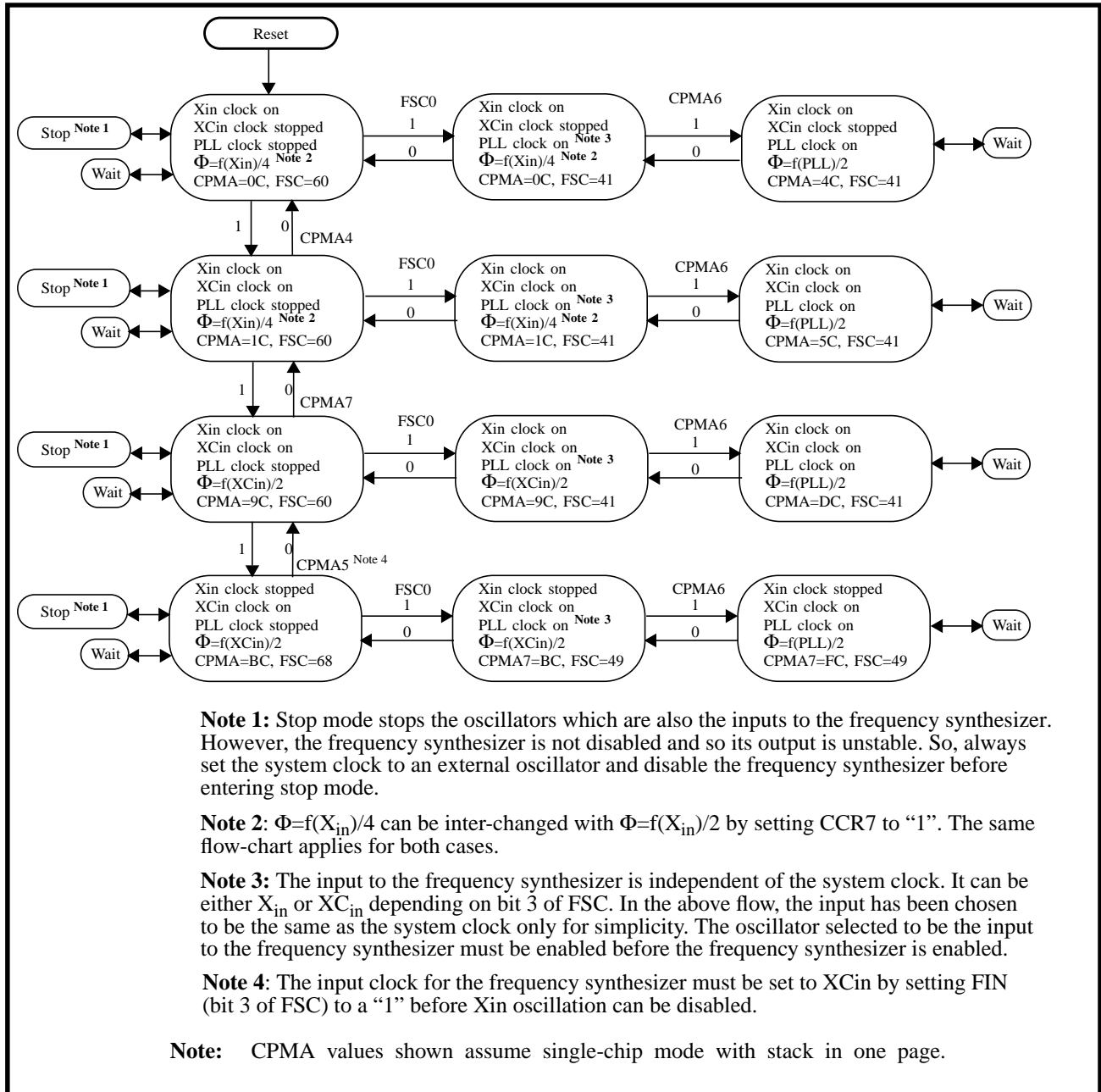


Figure 2-6. Clock Flow Diagram

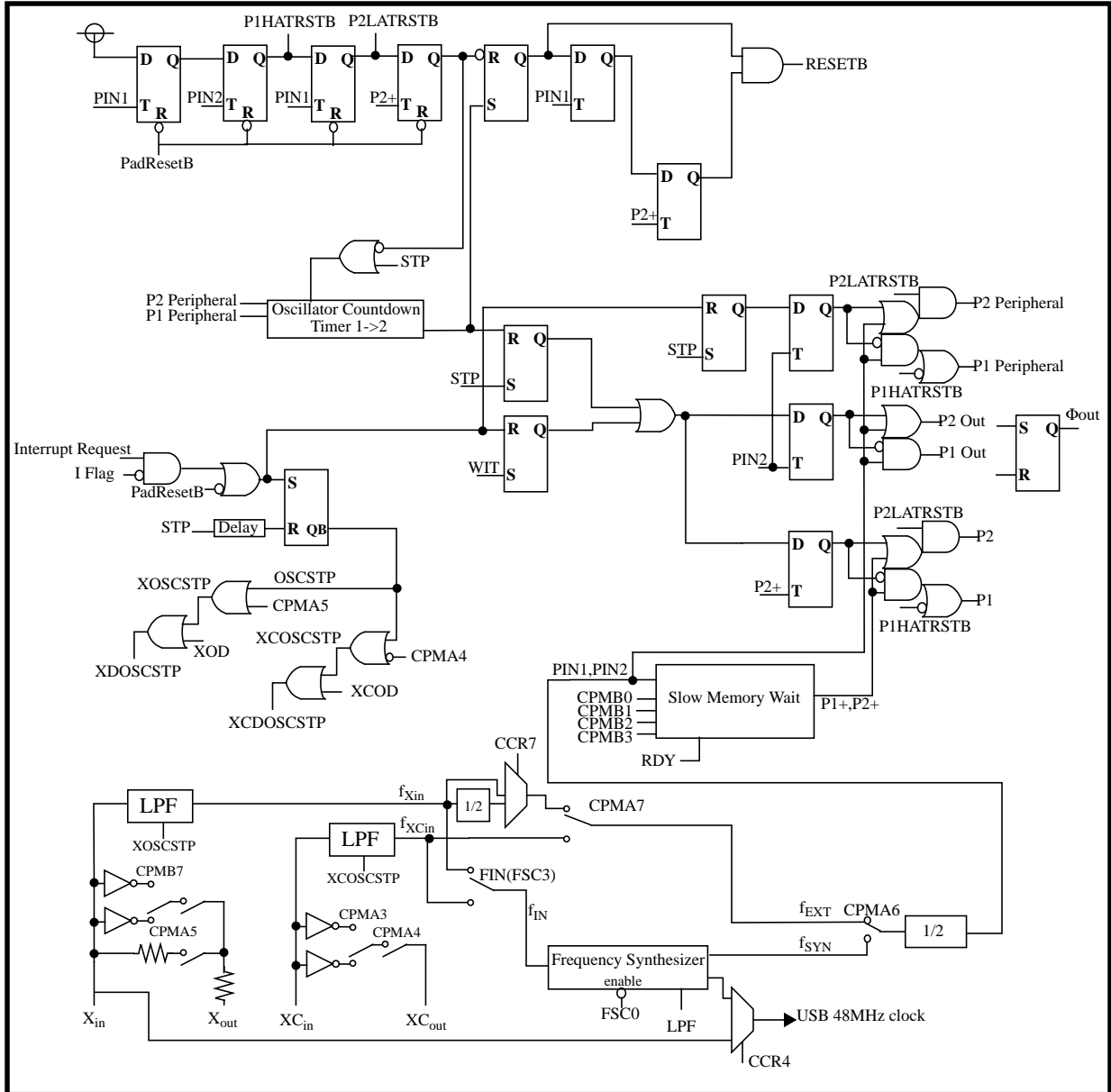


Figure 2-7. Clock Block Diagram



### 2.3.2 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 2-8.

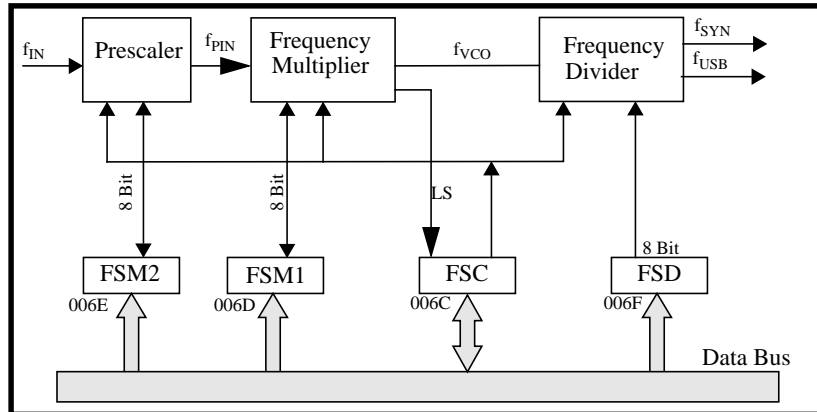


Figure 2-8. Frequency Synthesizer Circuit

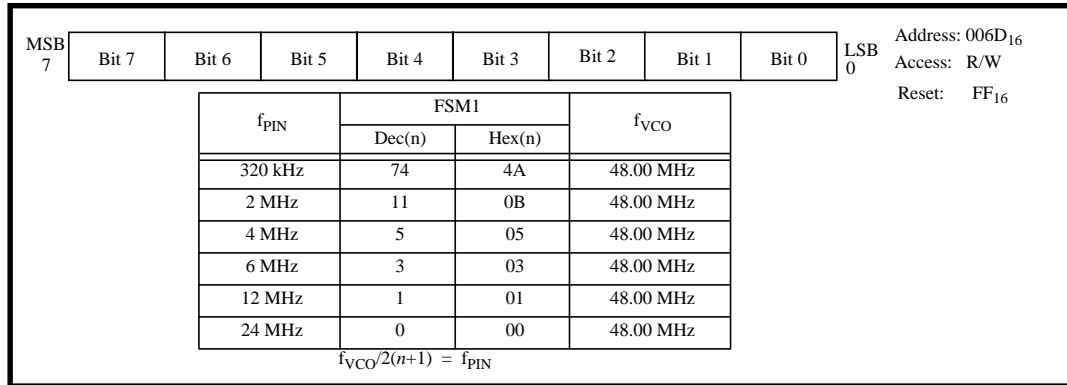
The frequency synthesizer consists of a prescaler, frequency multiplier macro, a frequency divider macro, and four registers, namely FSM1, FSM2, FSC and FSD. Two multiply registers (FSM1, FSM2) control the frequency multiply amount. Clock  $f_{IN}$  is prescaled using FSM2 to generate  $f_{PIN}$ .  $f_{PIN}$  is multiplied using FSM1 to generate an  $f_{VCO}$  clock which 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:

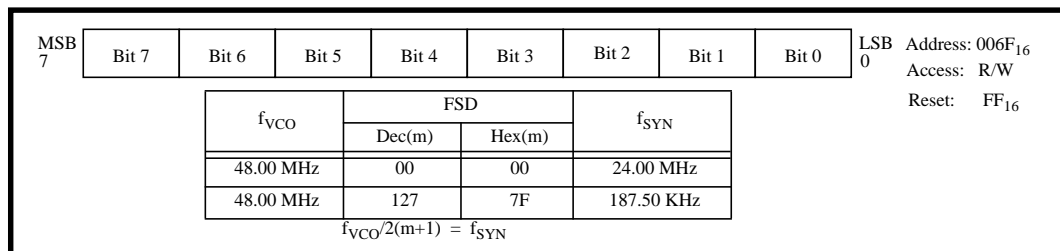
- $f_{PIN} = f_{IN} / 2(n+1)$  where n is a decimal number between 0 and 254. Setting FSM2 to 255 disables the prescaler and  $f_{PIN} = f_{IN}$ .

MSB 7	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	LSB 0	Address: 006E <sub>16</sub>
									Access: R/W	
									Reset: FF <sub>16</sub>	
	$f_{PIN}$	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 2-9. Frequency Synthesizer Multiply Control Register FSM2



**Figure 2-10. Frequency Synthesizer Multiply Control register FSM1**



**Figure 2-11. 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 disables  $f_{SYN}$ .

The FSC0 bit in the FSC Control Register enables the frequency synthesizer block. When disabled (FSC0 = "0"),  $f_{VCO}$  is held at either a high or low state. When the frequency synthesizer control bit is active (FSC0 = "1"), a lock status (LS = "1") indicates that  $f_{SYN}$  and  $f_{VCO}$  are the correct frequency. The LS and FSC0 control bits in the FSC Control register are shown in Figure 2-12.

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

Once the frequency synthesizer is enabled, a delay of 2-5ms is recommended before the output of the frequency synthesizer is used. This is done to allow the output to stabilize. It is also recommended that none of the registers be modified once the frequency synthesizer is enabled as it will cause the output to be temporarily (2-5ms) unstable.



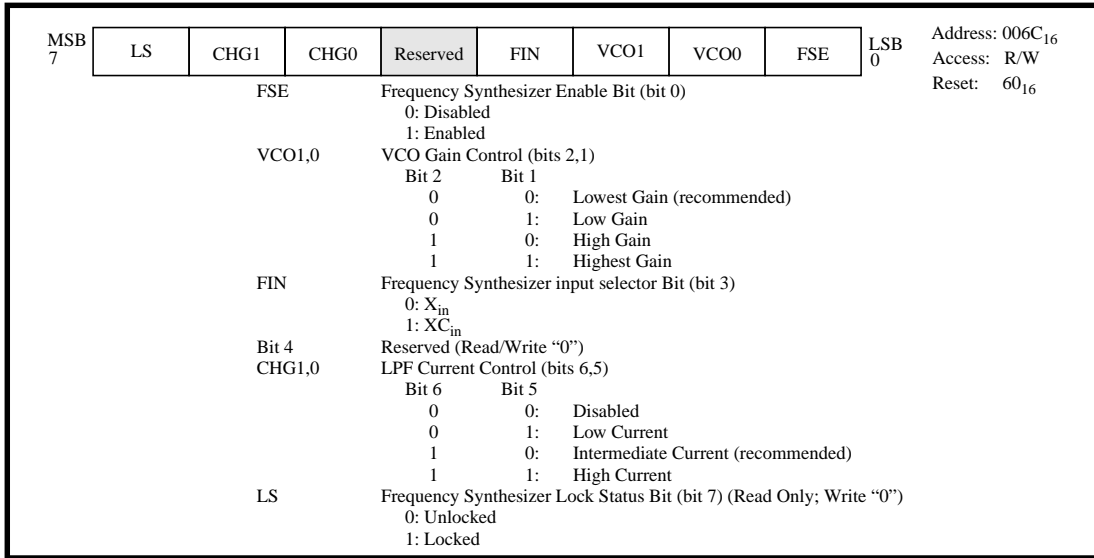


Figure 2-12. Frequency Synthesizer Control Register

## 2.4 Memory Map

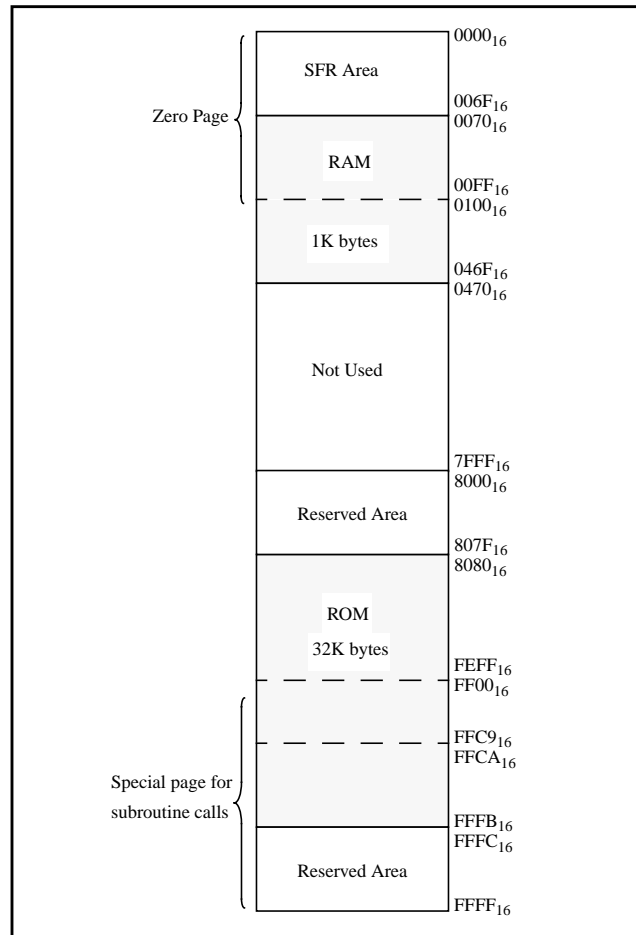


Figure 2-13. Memory Map

The first 112 bytes of memory from 0000<sub>16</sub> to 006F<sub>16</sub> is the special function register (SFR) area and contains the CPU mode registers, interrupt registers, and other registers to control peripheral functions (see Figure 2-13.).

The general purpose RAM resides from 0070<sub>16</sub> to 046F<sub>16</sub>. 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<sub>16</sub> to 7FFF<sub>16</sub> 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<sub>16</sub> to 807F<sub>16</sub> and from FFFC<sub>16</sub> to FFFF<sub>16</sub> 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<sub>16</sub> to FFFB<sub>16</sub>. 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.



### 2.4.1 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 2-14.). These dedicated addressing modes enable access to this area with fewer instruction cycles.

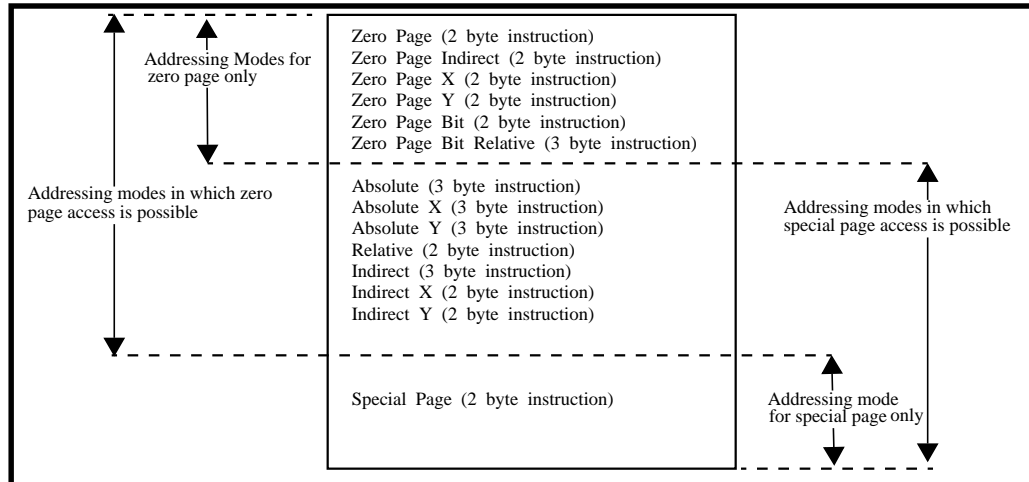


Figure 2-14. Zero Page and Special Page Addressing Modes

### 2.4.2 Special Page

The 256 bytes from address FF00<sub>16</sub> to FFFF<sub>16</sub> are called the special page area. In this area special page addressing can be used to specify memory addresses (see Figure 2-14.). This dedicated special page addressing mode enables access to this area with fewer instruction cycles. Frequently used subroutines are normally stored in this area.

### 2.4.3 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 2-1.). The reserved addresses should not be read or written to.

Table 2-1. SFR Addresses

Addr	Description	Acronym and Value at Reset	Addr	Description	Acronym and Value at Reset
0000 <sub>16</sub>	CPU Mode Register A	CPMA=0C	0038 <sub>16</sub>	UART2 Mode Register	U2MOD=00
0001 <sub>16</sub>	CPU Mode Register B	CPMB=03	0039 <sub>16</sub>	UART2 Baud Rate Generator	U2BRG=XX
0002 <sub>16</sub>	Interrupt Request Register A	IREQA=00	003A <sub>16</sub>	UART2 Status Register	U2STS=03
0003 <sub>16</sub>	Interrupt Request Register B	IREQB=00	003B <sub>16</sub>	UART2 Control Register	U2CON=00
0004 <sub>16</sub>	Interrupt Request Register C	IREQC=00	003C <sub>16</sub>	UART2 Transmit/Receiver Buffer 1	U2TRB1=XX
0005 <sub>16</sub>	Interrupt Control Register A	ICONA=00	003D <sub>16</sub>	UART2 Transmit/Receiver Buffer 2	U2TRB2=XX
0006 <sub>16</sub>	Interrupt Control Register B	ICONB=00	003E <sub>16</sub>	UART2 RTS Control Register	U2RTSC=80
0007 <sub>16</sub>	Interrupt Control Register C	ICONC=00	003F <sub>16</sub>	DMAC Index and Status Register	DMAIS=00
0008 <sub>16</sub>	Port P0	P0=00	0040 <sub>16</sub>	DMAC Channel x Mode Register 1	DMAxM1=00
0009 <sub>16</sub>	Port P0 Direction Register	POD=00	0041 <sub>16</sub>	DMAC Channel x Mode Register 2	DMAxM2=00
000A <sub>16</sub>	Port P1	P1=00	0042 <sub>16</sub>	DMAC Channel x Source Register Low	DMAxSL=00
000B <sub>16</sub>	Port P1 Direction Register	P1D=00	0043 <sub>16</sub>	DMAC Channel x Source Register High	DMAxSH=00
000C <sub>16</sub>	Port P2	P2=00	0044 <sub>16</sub>	DMAC Channel x Destination Register Low	DMAxDL=00
000D <sub>16</sub>	Port P2 Direction Register	P2D=00	0045 <sub>16</sub>	DMAC Channel x Destination Register High	DMAxDH=00
000E <sub>16</sub>	Port P3	P3=00	0046 <sub>16</sub>	DMAC Channel x Count Register Low	DMAxCL=00
000F <sub>16</sub>	Port P3 Direction Register	P3D=00	0047 <sub>16</sub>	DMAC Channel x Count Register High	DMAxCH=00
0010 <sub>16</sub>	Port Control Register	PTC=00	0048 <sub>16</sub>	Data Bus Buffer register 0	DBB0=00
0011 <sub>16</sub>	Interrupt Polarity Selection Register	IPOL=00	0049 <sub>16</sub>	Data Bus Buffer status register 0	DBBS0=00
0012 <sub>16</sub>	Port P2 pull-up Control Register	PUP2=00	004A <sub>16</sub>	Data Bus Buffer Control Register 0	DBBC0=00
0013 <sub>16</sub>	USB Control Register	USBC=00	004B <sub>16</sub>	Reserved	
0014 <sub>16</sub>	Port P6	P6=00	004C <sub>16</sub>	Data Bus Buffer register 1	DBB1=00
0015 <sub>16</sub>	Port P6 Direction Register	P6D=00	004D <sub>16</sub>	Data Bus Buffer Status Register 1	DBBS1=00
0016 <sub>16</sub>	Port P5	P5=00	004E <sub>16</sub>	Data Bus Buffer Control Register 1	DBBC1=00
0017 <sub>16</sub>	Port P5 Direction Register	P5D=00	004F <sub>16</sub>	Reserved	
0018 <sub>16</sub>	Port P4	P4=00	0050 <sub>16</sub>	USB Address Register	USBA=00
0019 <sub>16</sub>	Port P4 Direction Register	P4D=00	0051 <sub>16</sub>	USB Power Management Register	USBPM=00
001A <sub>16</sub>	Port P7	P7=00	0052 <sub>16</sub>	USB Interrupt Status Register 1	USBIS1=00
001B <sub>16</sub>	Port P7 Direction Register	P7D=00	0053 <sub>16</sub>	USB Interrupt Status Register 2	USBIS2=00
001C <sub>16</sub>	Port P8	P8=00	0054 <sub>16</sub>	USB Interrupt Enable Register 1	USBIE1=00
001D <sub>16</sub>	Port P8 Direction Register	P8D=00	0055 <sub>16</sub>	USB Interrupt Enable Register 2	USBIE2=00
001E <sub>16</sub>	Reserved		0056 <sub>16</sub>	USB Frame Number Register Low	USBSOFL=00
001F <sub>16</sub>	Clock Control Register	CCR=00	0057 <sub>16</sub>	USB Frame Number Register High	USBSOFH=00
0020 <sub>16</sub>	Timer XL	TXL=FF	0058 <sub>16</sub>	USB Endpoint Index	USBINDEX=00
0021 <sub>16</sub>	Timer XH	TXH=FF	0059 <sub>16</sub>	USB Endpoint x IN CSR	IN_CSR=00
0022 <sub>16</sub>	Timer YL	TYL=FF	005A <sub>16</sub>	USB Endpoint x OUT CSR	OUT_CSR=00
0023 <sub>16</sub>	Timer YH	TYH=FF	005B <sub>16</sub>	USB Endpoint x IN MAXP	IN_MAXP=00
0024 <sub>16</sub>	Timer 1	T1=FF	005C <sub>16</sub>	USB Endpoint x OUT MAXP	OUT_MAXP=00
0025 <sub>16</sub>	Timer 2	T2=01	005D <sub>16</sub>	USB Endpoint x OUT WRT_CNT Low	WRT_CNTL=00
0026 <sub>16</sub>	Timer 3	T3=FF	005E <sub>16</sub>	USB Endpoint x OUT WRT_CNT High	WRT_CNTH=00
0027 <sub>16</sub>	Timer X Mode Register	TXM=00	005F <sub>16</sub>	Reserved	
0028 <sub>16</sub>	Timer Y Mode Register	TYM=00	0060 <sub>16</sub>	USB Endpoint 0 FIFO	USBFIFO0=N/A
0029 <sub>16</sub>	Timer 123 Mode Register	T123M=00	0061 <sub>16</sub>	USB Endpoint 1 FIFO	USBFIFO1=N/A
002A <sub>16</sub>	SIO Shift Register	SIOSHT=XX	0062 <sub>16</sub>	USB Endpoint 2 FIFO	USBFIFO2=N/A
002B <sub>16</sub>	SIO Control Register 1	SIOCON1=00	0063 <sub>16</sub>	USB Endpoint 3 FIFO	USBFIFO3=N/A
002C <sub>16</sub>	SIO Control Register 2	SIOCON2=18	0064 <sub>16</sub>	USB Endpoint 4 FIFO	USBFIFO4=N/A
002D <sub>16</sub>	Special Count Source Generator1	SCSG1=FF	0065 <sub>16</sub>	Reserved	
002E <sub>16</sub>	Special Count Source Generator2	SCSG2=FF	0066 <sub>16</sub>	Reserved	
002F <sub>16</sub>	Special Count Source Mode Register	SCSM=00	0067 <sub>16</sub>	Reserved	
0030 <sub>16</sub>	UART1 Mode Register	U1MOD=00	0068 <sub>16</sub>	Reserved	
0031 <sub>16</sub>	UART1 Baud Rate Generator	U1BRG=XX	0069 <sub>16</sub>	Reserved	
0032 <sub>16</sub>	UART1 Status Register	U1STS=03	006A <sub>16</sub>	Reserved	
0033 <sub>16</sub>	UART1 Control Register	U1CON=00	006B <sub>16</sub>	Reserved	
0034 <sub>16</sub>	UART1 Transmit/Receiver Buffer 1	U1TRB1=XX	006C <sub>16</sub>	Freq Synthesizer Control	FSC=60
0035 <sub>16</sub>	UART1 Transmit/Receiver Buffer 2	U1TRB2=XX	006D <sub>16</sub>	Freq Synthesizer Multiply Register 1	FSM1=FF
0036 <sub>16</sub>	UART1 RTS Control Register	U1RTSC=80	006E <sub>16</sub>	Freq Synthesizer Multiply Register 2	FSM2=FF
0037 <sub>16</sub>	Reserved		006F <sub>16</sub>	Freq Synthesizer Divide Register	FSD=FF



## 2.5 Processor Modes

The operation modes are described below. The memory maps for the first three modes are shown in Figure 2-15.

Single chip mode is normally entered after reset. However, if the MCU has a CNV<sub>SS</sub> 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 CPMA. However, while CNV<sub>SS</sub> is high, bit 1 of CPMA is “1” and cannot be changed.

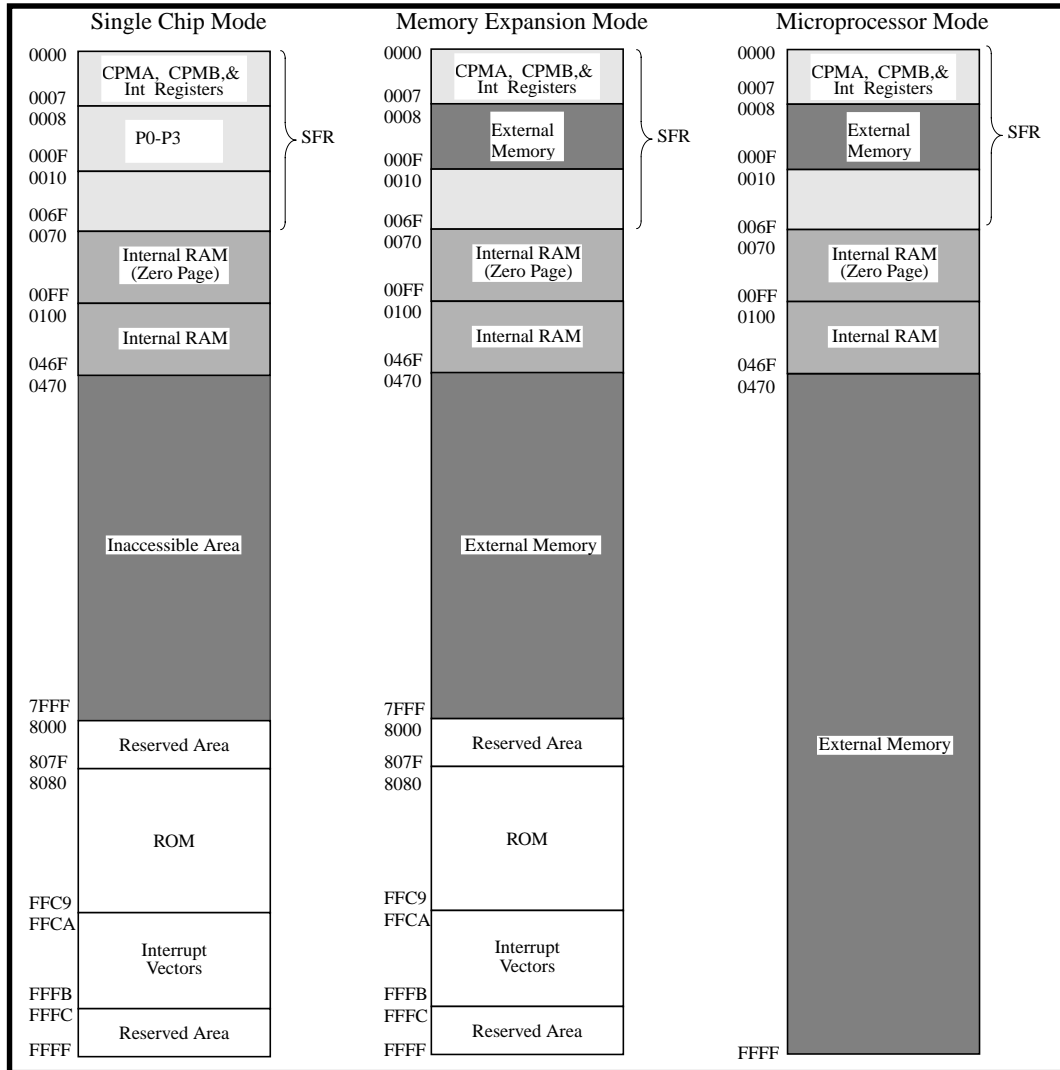


Figure 2-15. Operation Modes Memory Maps

### 2.5.1 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 EDMA are disabled in this mode.

## 2.5.2 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  $\overline{RD}$ ,  $\overline{WR}$ ,  $\text{SYNCout}$ ,  $\Phi_{out}$ , and  $\text{DMAout}$ , 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 2-15.) In this mode, slow memory wait and  $\overline{EDMA}$  can be enabled.

## 2.5.3 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.

## 2.5.4 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.

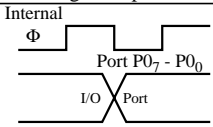
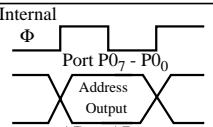
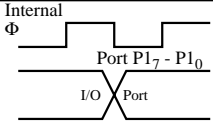
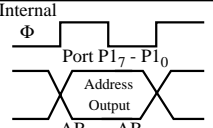
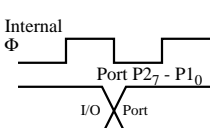
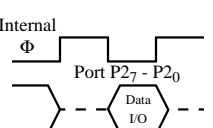
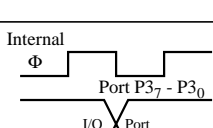
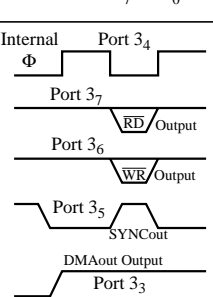
CPMA1	0	1	0
CPMA0	0	0	1
Port Mode	Single Chip Mode	Microprocessor Mode	Memory Expansion Mode
Port P0			Same as Microprocessor Mode
Port P1			Same as Microprocessor Mode
Port P2			Same as Microprocessor Mode
Port P3			Same as Microprocessor Mode

Figure 2-16. Function of Ports P<sub>0</sub>-P<sub>3</sub> in each Processor Mode



### 2.5.5 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 2-17.

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  $\Phi_{out}$  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  $\Phi_{out}$  going low at the beginning of the read/write cycle, the cycle is not extended. The timing for RDY wait is shown in Figure 2-18..

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  $\Phi_{out}$  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  $\Phi_{out}$  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 2-19.

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.

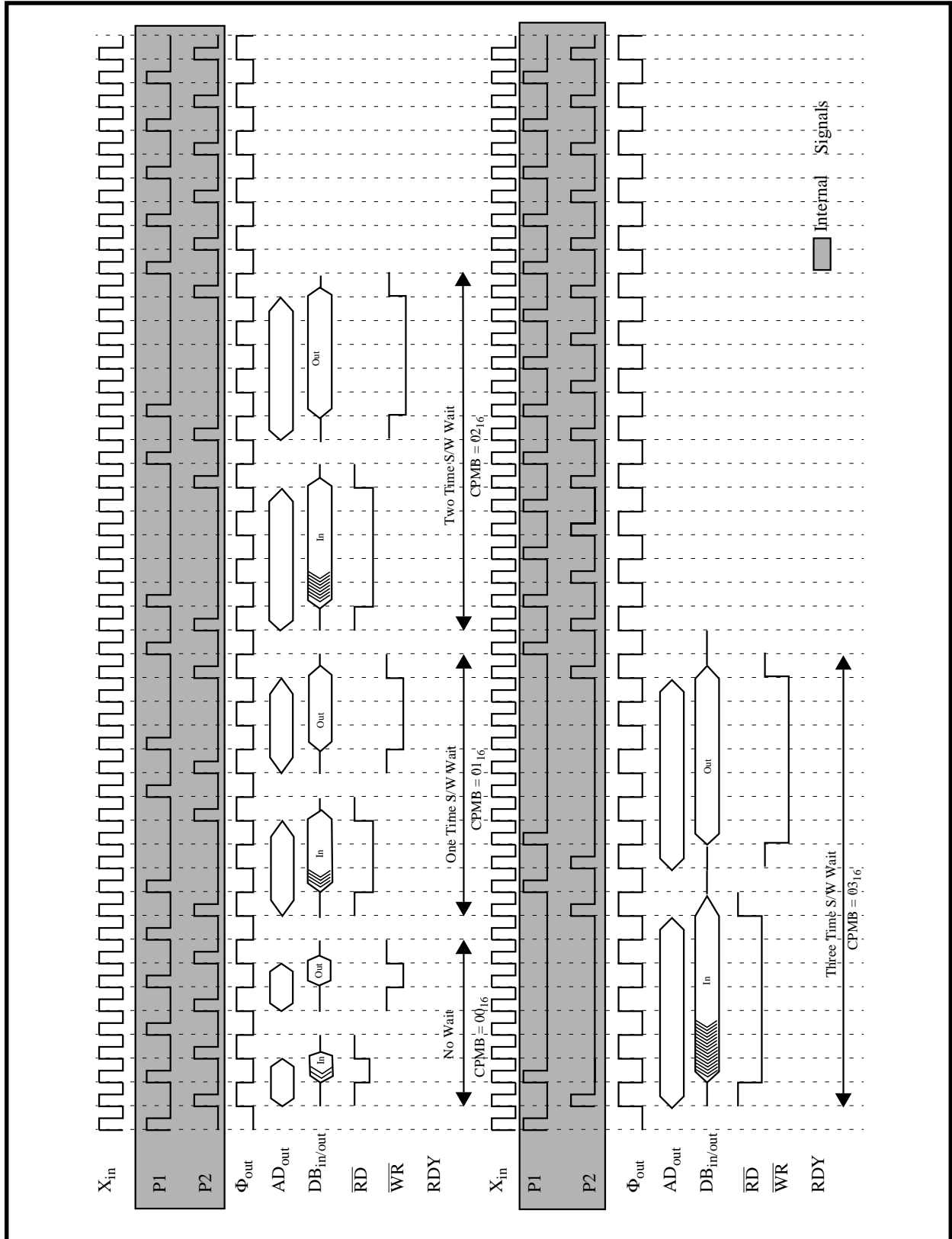


Figure 2-17. Software Wait Timing Diagram



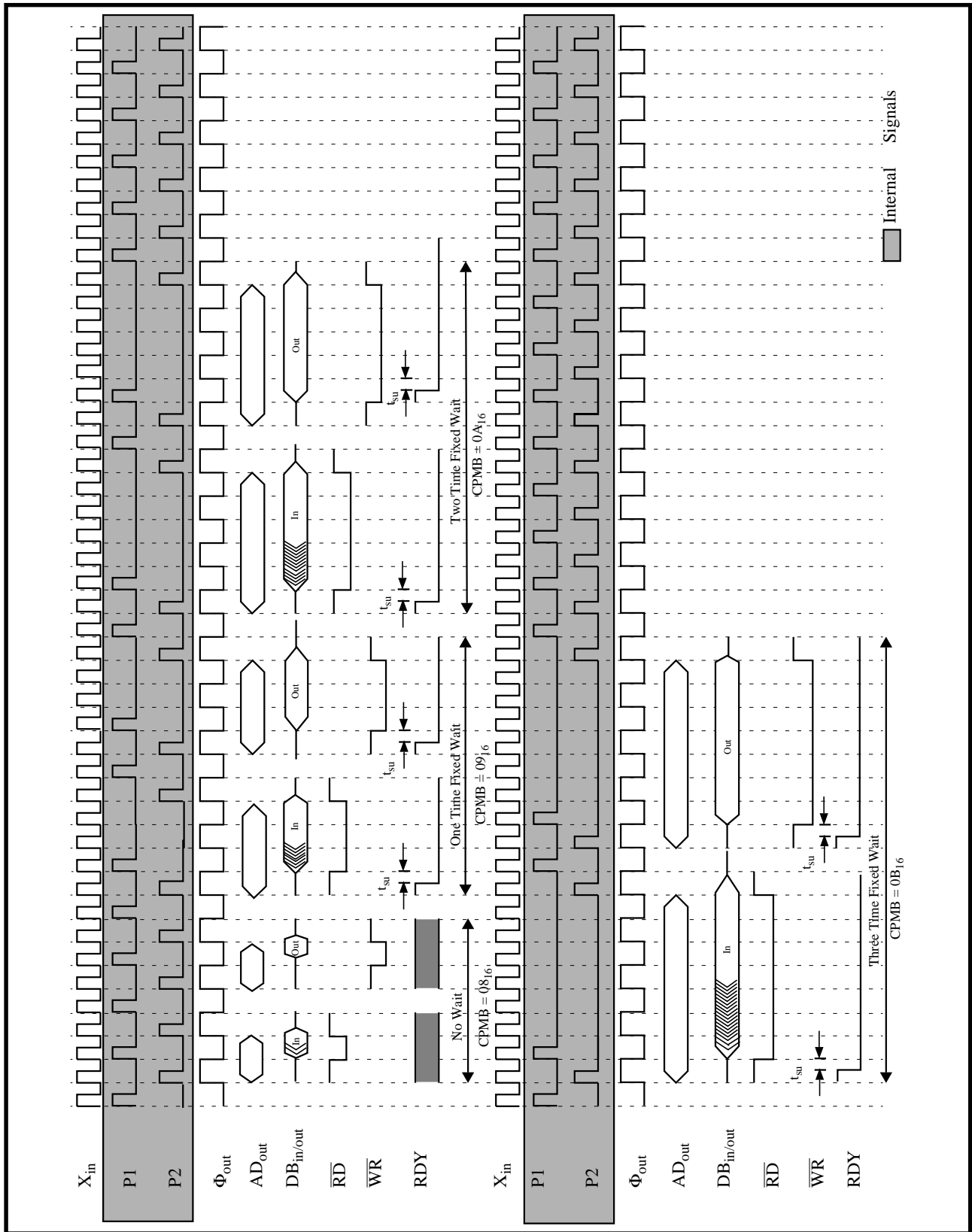


Figure 2-18. RDY Wait Timing Diagram

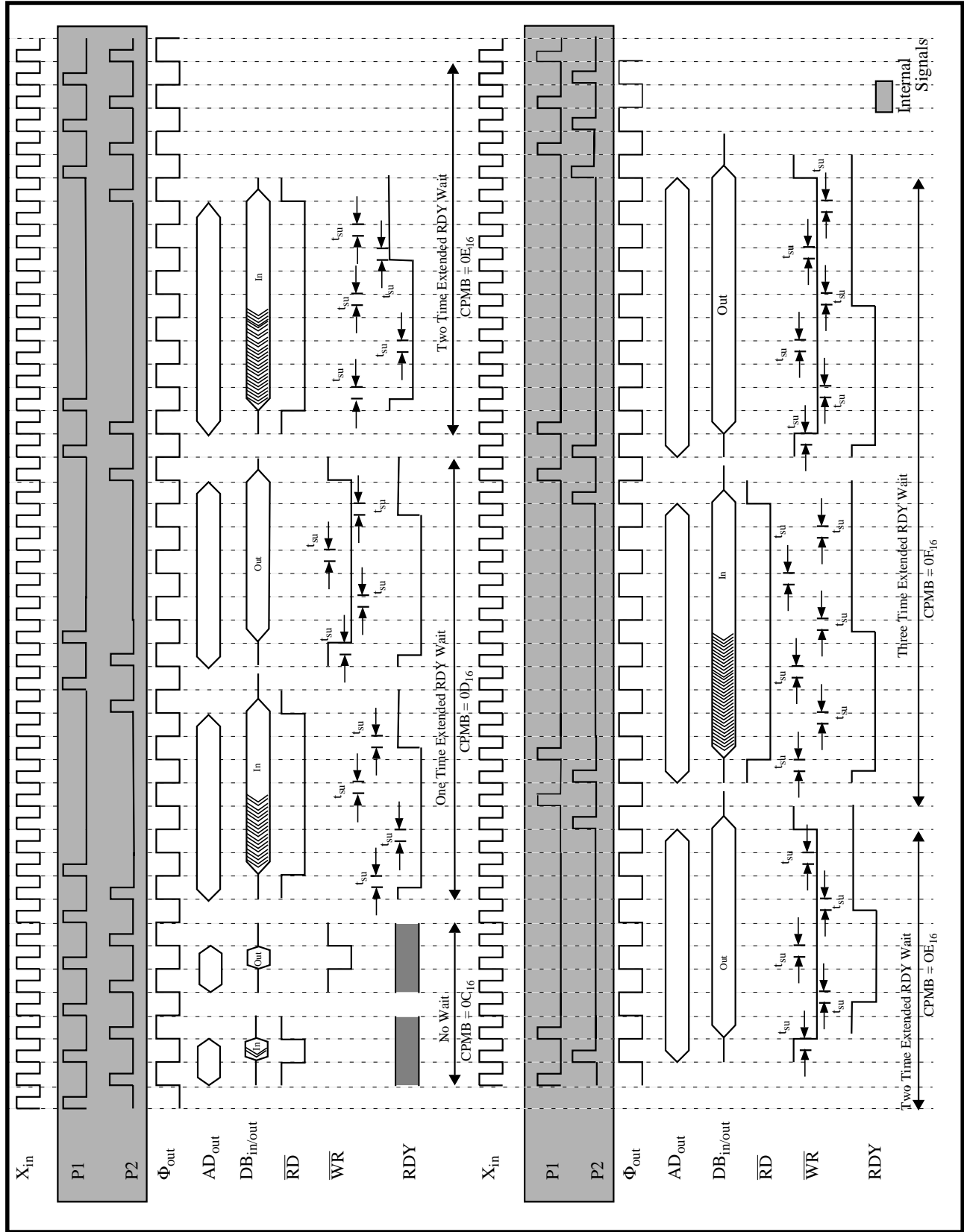


Figure 2-19. Extended RDY Wait Timing Diagram



## 2.5.6 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,  $\overline{\text{HOLD}}$  (P7<sub>1</sub>) and  $\overline{\text{HLDA}}$  (P7<sub>3</sub>).  $\overline{\text{HOLD}}$  is an input to the MCU and is brought low when an external device wants the MCU to relinquish the address and data buses.  $\overline{\text{HLDA}}$  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<sub>7</sub> ( $\overline{\text{RD}}$ ) and port P3<sub>6</sub> ( $\overline{\text{WR}}$ ) high. Ports P3<sub>7</sub> and P3<sub>6</sub> are held high to prevent any external device that is enabled by  $\overline{\text{RD}}$  or  $\overline{\text{WR}}$  from being falsely activated. The clocks to the CPU are stopped, but the peripheral clocks and port P3<sub>4</sub> ( $\Phi_{\text{out}}$ ) continue to oscillate.  $\overline{\text{HOLD}}$  is brought high to allow the MCU to regain the address and data buses. When this occurs,  $\overline{\text{HLDA}}$  will go high and ports P<sub>1</sub>, P<sub>2</sub>, P3<sub>7</sub> and P3<sub>6</sub> will begin to drive the external buses again. The timing for the hold function is shown in Figure 2-20. The hold function is only valid for memory expansion and microprocessor modes. Bit 5 of CPMB is used to enable the hold function.  $\overline{\text{HLDA}}$  will lose its function when the  $\overline{\text{IBF}}_1$  pin functionality is used.

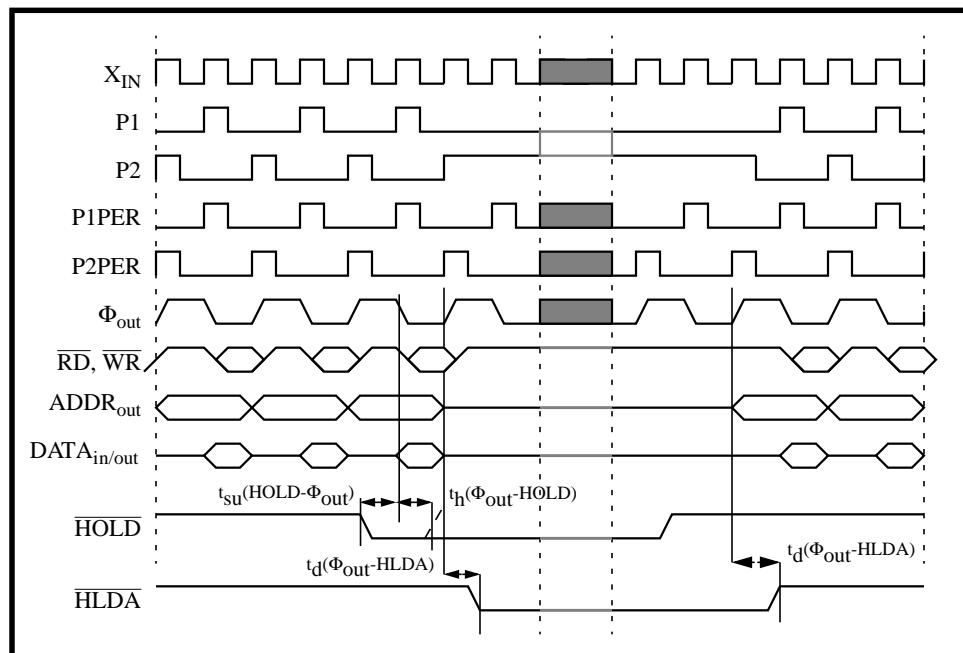


Figure 2-20. Hold Mode Timing Diagram

## 2.5.7 Expanded Data Memory Access

The Expanded Data Memory Access ( $\overline{\text{EDMA}}$ ) mode feature allows the user to access a greater than 64 Kbyte data area for instructions LDA (IndY) with T = "0" and T = "1", and STA (IndY). Bit 4 of CPMB is used to enable/disable the  $\overline{\text{EDMA}}$  function. If bit 4 of CPMB equals "1", then during the data read/write cycle of instructions LDA (IndY) and STA (IndY) Port 4<sub>0</sub> ( $\overline{\text{EDMA}}$ ) is driven low. The  $\overline{\text{EDMA}}$  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 2-21. and Figure 2-22.

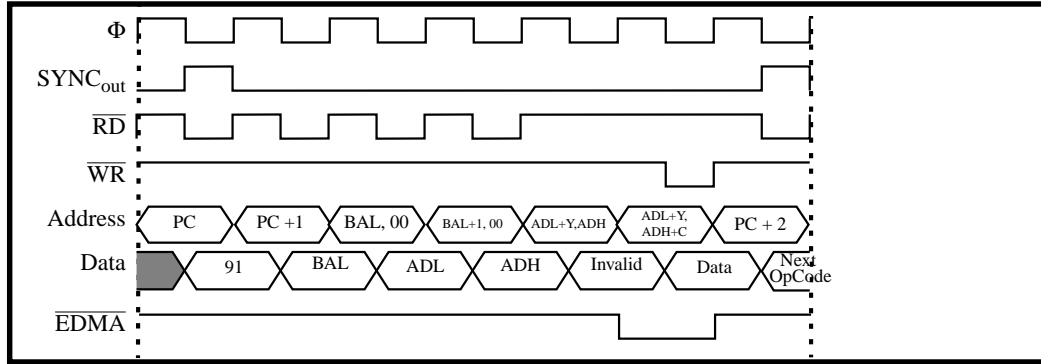


Figure 2-21. STA (\$zz),Y Instruction Sequence with  $\overline{\text{EDMA}}$  Enabled

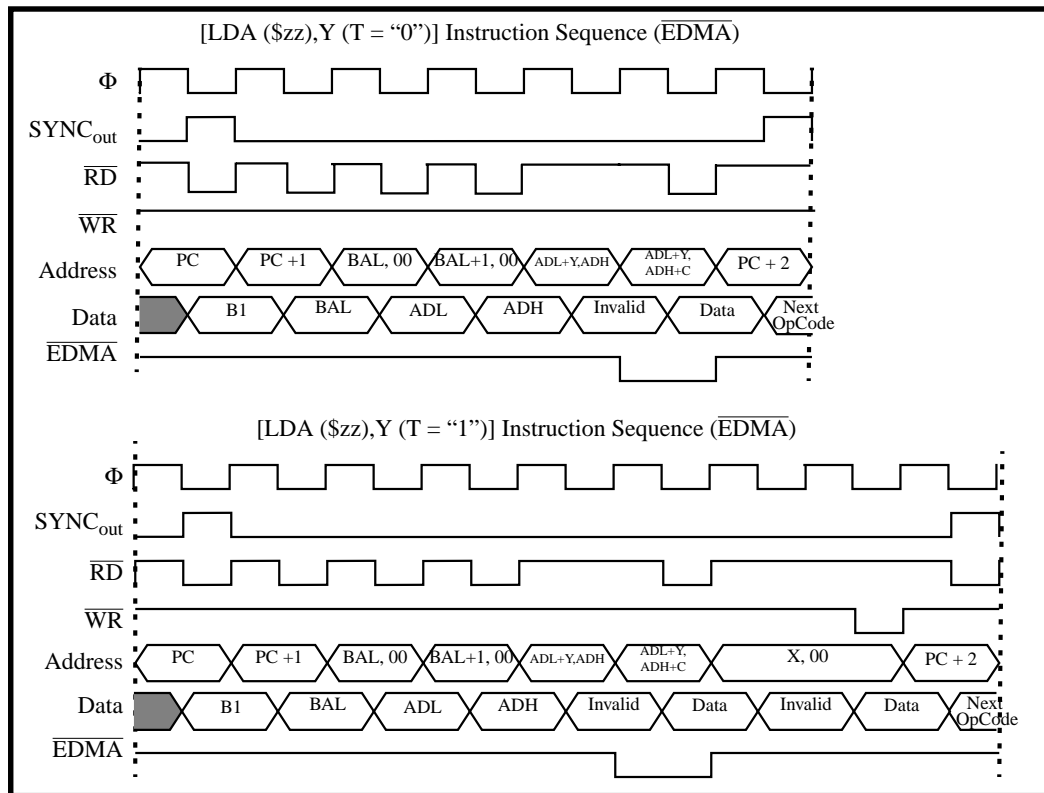


Figure 2-22. LDA (\$zz),Y Instruction Sequences with  $\overline{\text{EDMA}}$  Enabled



## 2.6 Peripheral Interface

### 2.6.1 Chip Bus Timing

The internal bus timing is described below for the CPU (or DMAC) writing to and reading from a peripheral (see Figure 2-23.)

- 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 PDnB signal (peripheral decode) is assigned to each peripheral and is low when reading from or writing to the peripheral. Each PDnB signal is clocked on P2 timing.

The address,  $R/\overline{W}$ , EB, and PDnB 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.

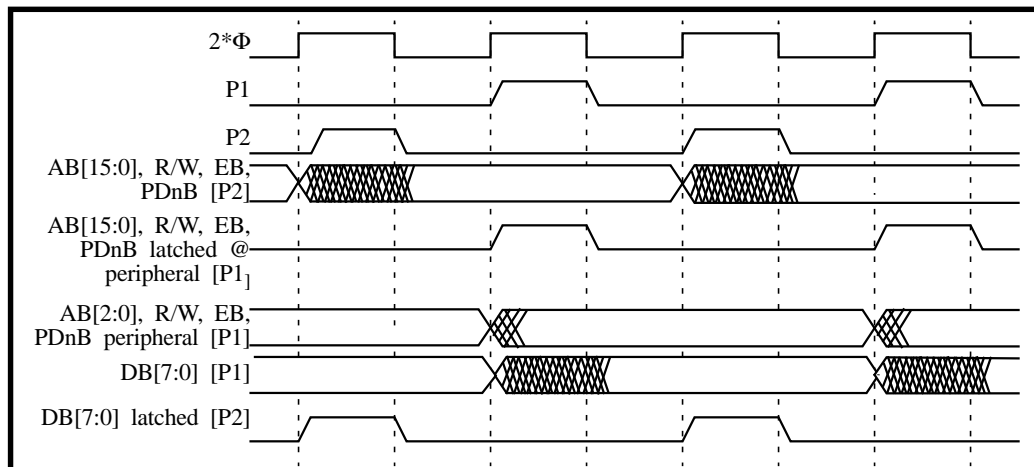


Figure 2-23. M37640 Internal Bus Timing

## 2.6.2 Peripheral Interface and Access Timing

The M37640 offers a wide variety of peripherals. These include RAM, EPROM, UARTs, SIOs, 8-bit and 16-bit timers, various I/O ports, clock generators, and USB core.

The interface between the CPU, the peripheral decode block, and peripheral blocks is shown in Figure 2-24. Signals DB7 to DB0, AB2 to AB0, R/W, EB, and at least one peripheral decode ( $PD_nB$ ) 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.

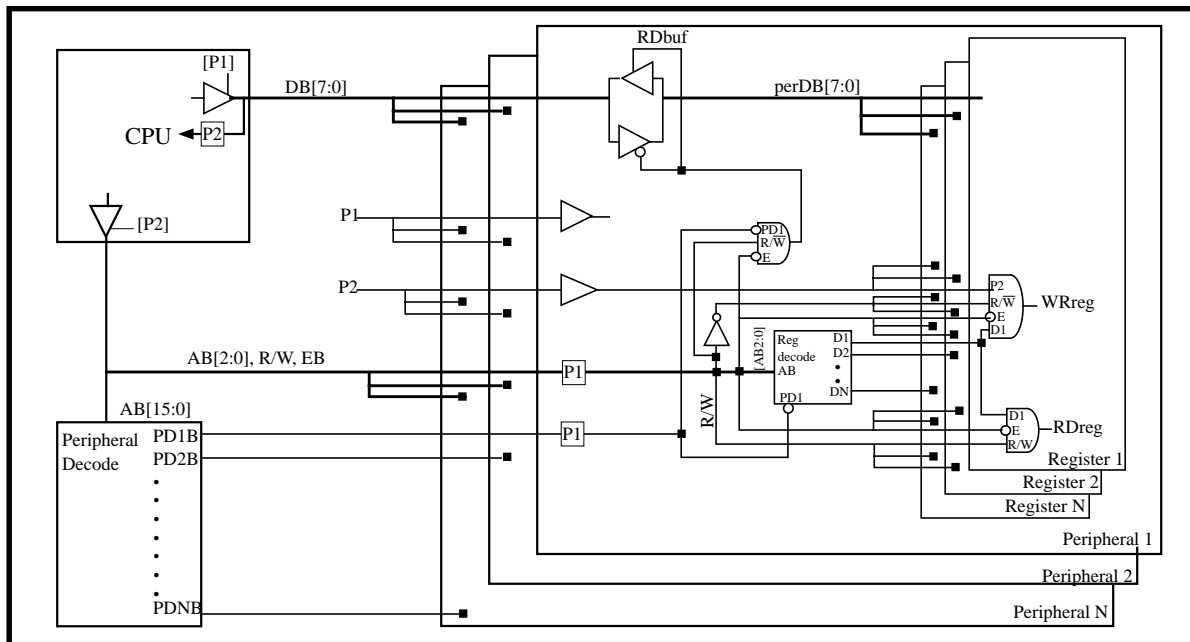


Figure 2-24. Internal Peripheral Interface



The bus timing for reading from and writing to a peripheral is shown in Figure 2-25.

- 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, EPROM, or USB core.

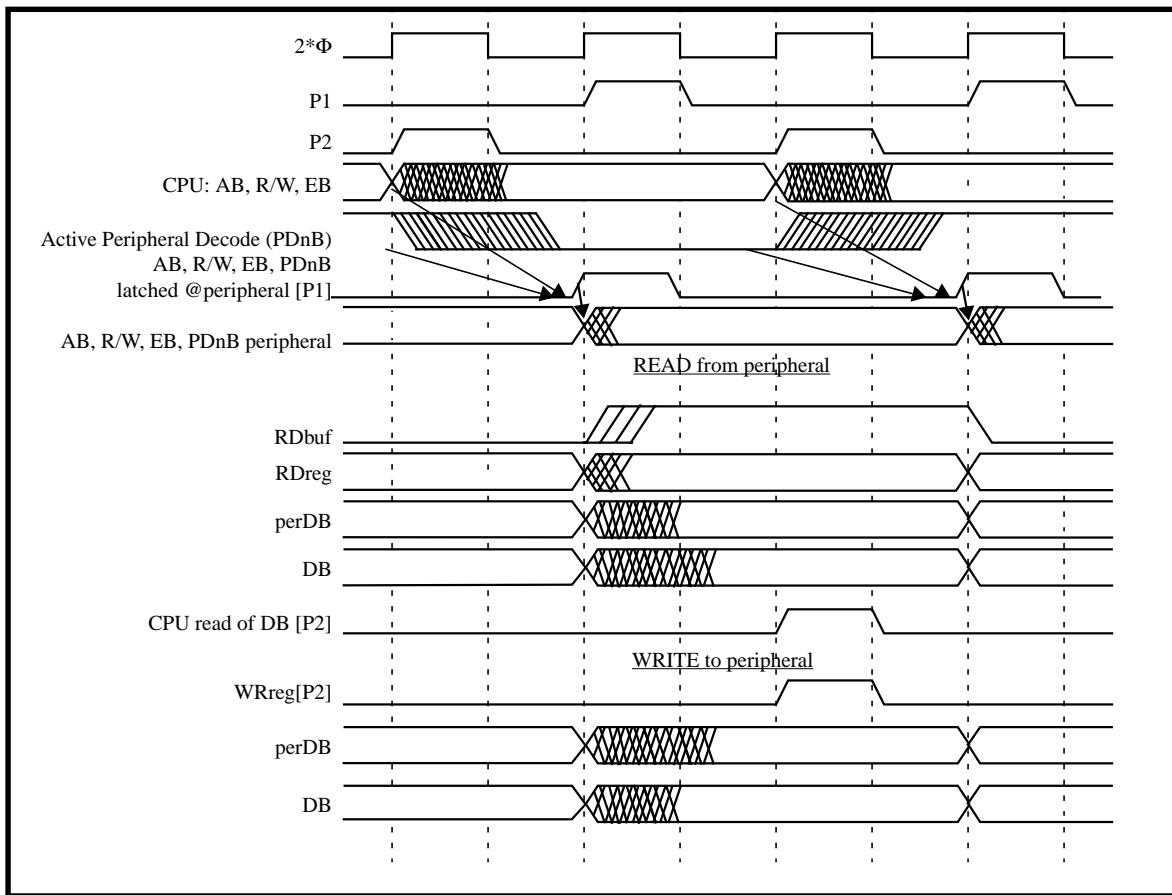


Figure 2-25. M37640 Peripheral Bus Timing

## 2.7 Input and Output Ports

Address	Description	Acronym and Value at Reset
0008 <sub>16</sub>	Port P0	P0=00
0009 <sub>16</sub>	Port P0 direction register	P0D=00
000A <sub>16</sub>	Port P1	P1=00
000B <sub>16</sub>	Port P1 direction register	P1D=00
000C <sub>16</sub>	Port P2	P2=00
000D <sub>16</sub>	Port P2 direction register	P2D=00
000E <sub>16</sub>	Port P3	P3=00
000F <sub>16</sub>	Port P3 direction register	P3D=00
0010 <sub>16</sub>	Port control register	PTC=00
0012 <sub>16</sub>	Port P2 pull-up Control Register	PUP2=00

Address	Description	Acronym and Value at Reset
0014 <sub>16</sub>	Port P6	P6=00
0015 <sub>16</sub>	Port P6 direction register	P6D=00
0016 <sub>16</sub>	Port P5	P5=00
0017 <sub>16</sub>	Port P5 direction register	P5D=00
0018 <sub>16</sub>	Port P4	P4=00
0019 <sub>16</sub>	Port P4 direction register	P4D=00
001A <sub>16</sub>	Port P7	P7=00
001B <sub>16</sub>	Port P7 direction register	P7D=00
001C <sub>16</sub>	Port P8	P8=00
001D <sub>16</sub>	Port P8 direction register	P8D=00

Port	2nd Function
Bit 4 <sub>0</sub>	multiplexed with $\overline{\text{EDMA}}$
Bits 4 <sub>1-42</sub>	multiplexed with external interrupts INT0, INT1
Bit 4 <sub>3</sub>	multiplexed with Timer X CNTR0 pin
Bit 4 <sub>4</sub>	multiplexed with Timer Y CNTR1 pin
Bit 5 <sub>0</sub>	multiplexed with $\text{XC}_{\text{in}}$
Bit 5 <sub>1</sub>	multiplexed with Timer 1/2 pulse output pin or $\text{XC}_{\text{out}}$
Bit 5 <sub>2</sub>	multiplexed with $\text{OBF}_0$ output to master CPU
Bit 5 <sub>3</sub>	multiplexed with $\overline{\text{IBF}}_0$ output to master CPU
Bit 5 <sub>4</sub>	multiplexed with $\overline{\text{S}}_0$ input from master CPU
Bit 5 <sub>5</sub>	multiplexed with $\overline{\text{A}}_0$ input from master CPU
Bit 5 <sub>6</sub>	multiplexed with $\overline{\text{R}}$ (E) input from master CPU
Bit 5 <sub>7</sub>	multiplexed with $\overline{\text{W}}$ (R/ $\overline{\text{W}}$ ) input from master CPU
Bits 6 <sub>0-67</sub>	multiplexed with Master CPU Bus I/F DQ0-DQ7 pins
Bit 7 <sub>0</sub>	multiplexed with $\overline{\text{SOF}}$
Bit 7 <sub>1</sub>	multiplexed with $\overline{\text{HOLD}}$
Bit 7 <sub>2</sub>	multiplexed with $\overline{\text{S}}_1$ input from master CPU
Bit 7 <sub>3</sub>	multiplexed with $\overline{\text{IBF}}_1$ or $\overline{\text{HLDA}}$
Bit 7 <sub>4</sub>	multiplexed with $\text{OBF}_1$ input from master CPU
Bits 8 <sub>0-83</sub>	multiplexed with the first alternate function UART2 pins or 2nd alternate function SIO pins
Bits 8 <sub>4-87</sub>	multiplexed with the UART1 pins

### 2.7.1 Ports

This device has 66 programmable I/O pins arranged as ports P0<sub>0</sub> to P8<sub>7</sub>. 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<sub>16</sub>, 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 GPI/O 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.

### 2.7.1.1 I/O Ports

#### Ports 0, 1, and 3

Ports 0 and 1 act as the address bus (AB<sub>0</sub>-AB<sub>15</sub>) in Microprocessor and Memory Expansion modes.

Bits 0 and 3-7 of Port 3 acts as control signals in Microprocessor and Memory Expansion modes.

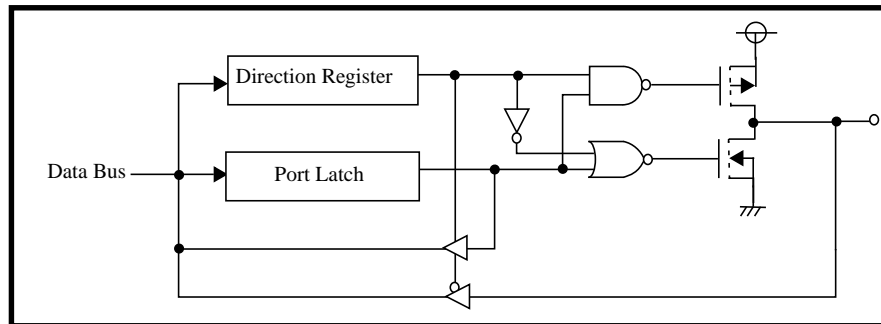


Figure 2-26. Port P0, P1, and P3 Block Diagram

#### Port 2

Port 2 is an 8-bit general purpose I/O port when in single chip mode. In this mode, the port has key-on wake up circuitry which can be used to restart the chip externally from a WIT or STP low power mode. This port also acts as the data bus during microprocessor and memory expansion modes. Port 2 input level can be set to reduced VIH level or CMOS level by bit 6 of the port control register (PTC).

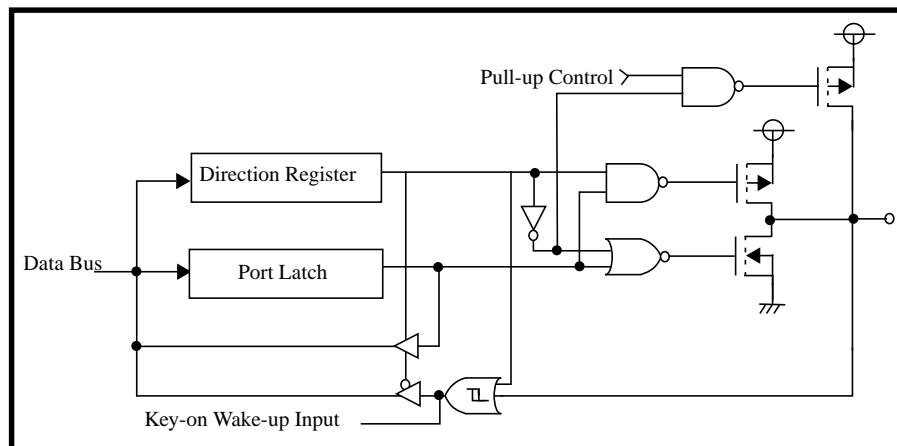


Figure 2-27. Port P2 Block Diagram

**Port 4**

Port 4 is a 5-bit general purpose I/O port that can be configured to access special second functions. The port can be set up in any configuration in all three processor modes.

**Port 4<sub>0</sub>**

This pin is multiplexed with the  $\overline{\text{EDMA}}$  (Extended Data Memory Access) function. When the MCU is in memory expansion or microprocessor mode and CPMB4 is set to "1", this pin operates as the EDMA output as described in section 2.5.7 on page 2-23.

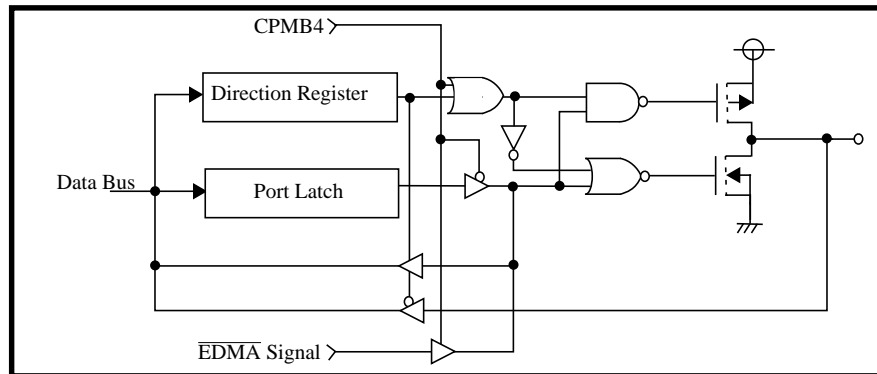


Figure 2-28. Port P4<sub>0</sub> Block Diagram

**Port 4<sub>1</sub>- 4<sub>2</sub>**

These pins are multiplexed with external interrupts 0 and 1 (INT0 and INT1). The external interrupt function is enabled by setting the bits to "1" in the interrupt control register that correspond to INT0 and INT1. The interrupt polarity register can be configured to define INT0 and INT1 as active high or low interrupts. See section 2.8.1 on page 2-43 for more information on configuring interrupts.

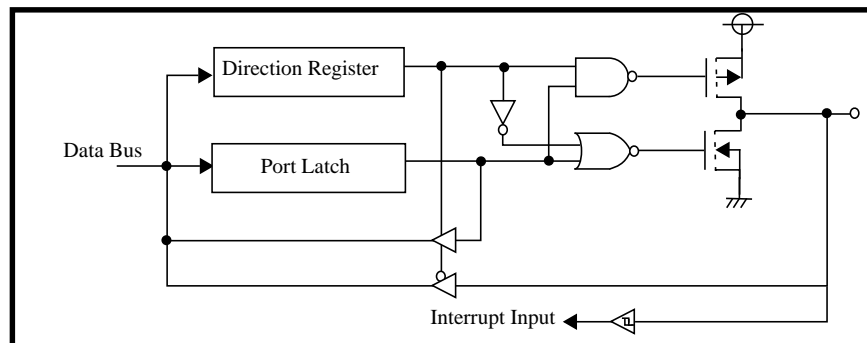


Figure 2-29. Port P4<sub>1</sub> and P4<sub>2</sub> Block Diagram

**Port 4<sub>3</sub>- 4<sub>4</sub>**

These pins are multiplexed with Timer X and Y functions for pin 4<sub>3</sub> and pin 4<sub>4</sub> respectively. The timer functions of the pins are independently defined by configuring the timer peripheral. Pin 4<sub>3</sub> acts as Timer X input pin for pulse width measurement mode and event counter mode or as Timer X output pin for pulse output mode. Pin 4<sub>3</sub> can also be used as an external interrupt (CNTR0) when Timer X is not in output mode. The polarity is selected in the Timer X mode register. The external interrupt function is enabled by setting the bit to "1" in the interrupt control register that



corresponds to CNTR0. See section 2.8.1 on page 2-43 for more information on configuring interrupts. Pin 4<sub>4</sub> acts as Timer Y input pin for pulse period measurement mode, pulse H-L measurement mode, and event counter mode or as Timer Y output pin for pulse output mode. Pin 4<sub>3</sub> can also be used as an external interrupt (CNTR1) when Timer Y is not in output mode. The polarity is selected in the Timer Y mode register. The external interrupt function is enabled by setting the bit to “1” in the interrupt control register that corresponds to CNTR1. See section 2.8.1 on page 2-43 for more information on configuring interrupts.

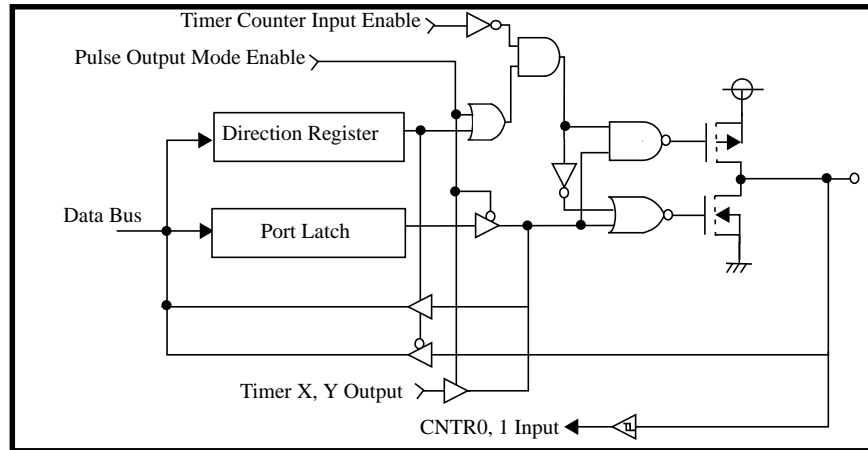


Figure 2-30. Port P4<sub>3</sub> and P4<sub>4</sub> Block Diagram

**Port 5**

Port 5 is an 8-bit general purpose I/O port that can be configured to access special second functions. The port can be set up in any configuration in all three processor modes.

**Port 5<sub>0</sub>**

This pin is multiplexed with the XC<sub>in</sub> clock input. When the XC<sub>in</sub> clock is activated, the pin's I/O is disabled.

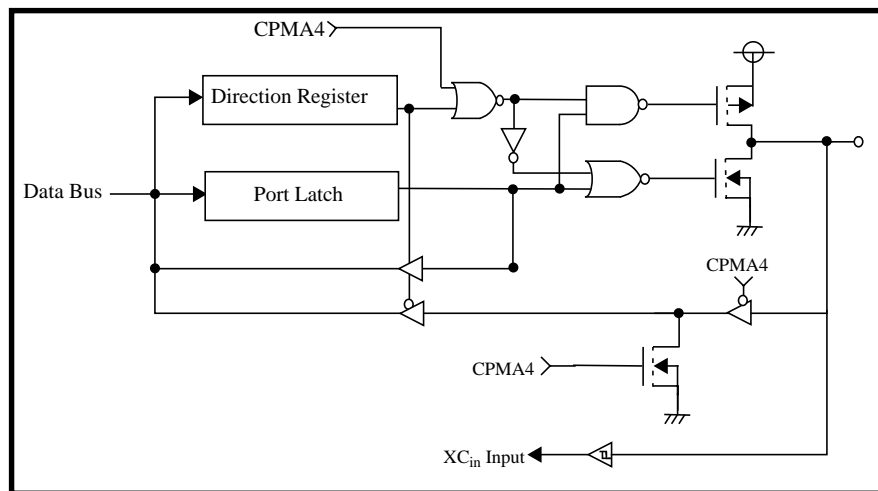


Figure 2-31. Port P5<sub>0</sub> Block Diagram

**Port 5<sub>1</sub>**

This pin is multiplexed with the XC<sub>out</sub> clock output and the Timer 1/2 pulse output. When the XC<sub>in</sub> clock is activated, the pin's I/O is disabled. If XC<sub>in</sub> is not being used as a system clock or XC<sub>out</sub> oscillation is disabled, the pin can be configured as the Timer 1/2 pulse output pin. This feature is configured in the Timer123 mode register as described in section 2.13.3 on page 2-87.

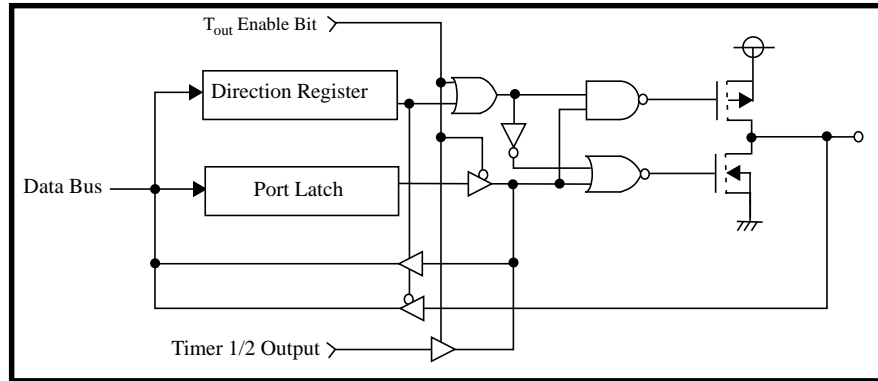


Figure 2-32. Port P5<sub>1</sub> Block Diagram

**Port 5<sub>2</sub>- 5<sub>7</sub>**

These pins are multiplexed with control pins for the bus interface control block. Pin 5<sub>2</sub> acts as OBF<sub>0</sub> output to a master cpu when DBBC00 is "1".

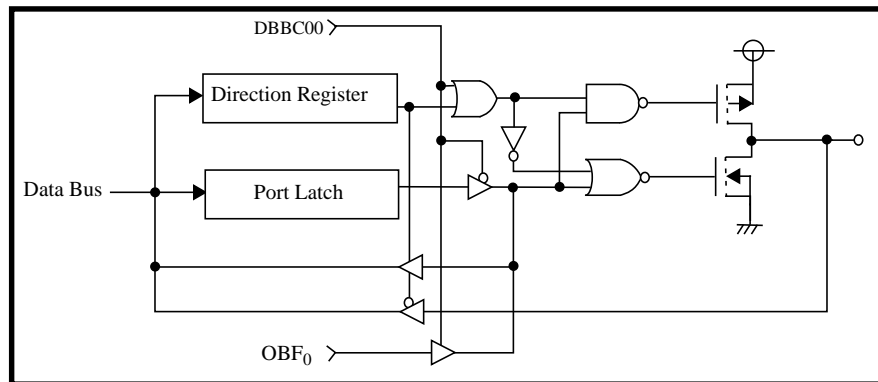


Figure 2-33. Port P5<sub>2</sub> Block Diagram



Pin 5<sub>3</sub> acts as  $\overline{IBF}_0$  output to a master cpu when DBBC01 is “1”.

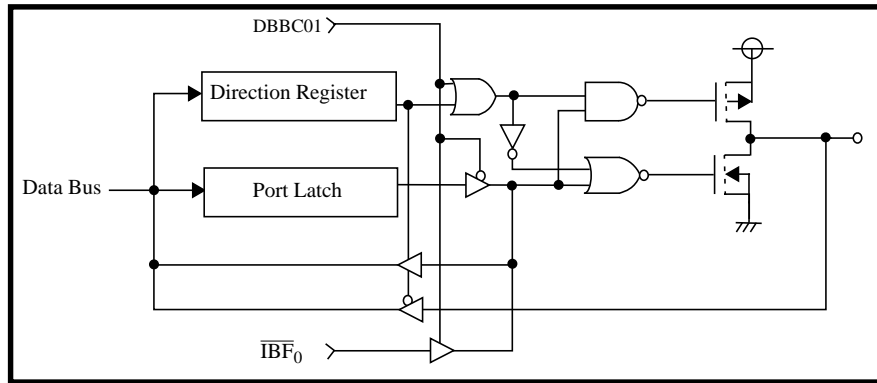


Figure 2-34. Port P5<sub>3</sub> Block Diagram

Pins 5<sub>4</sub>-5<sub>7</sub> act as input control signals from a master cpu when DBBC06 is “1”. Table 2-2 shows the bus interface control signal that corresponds to each pin.

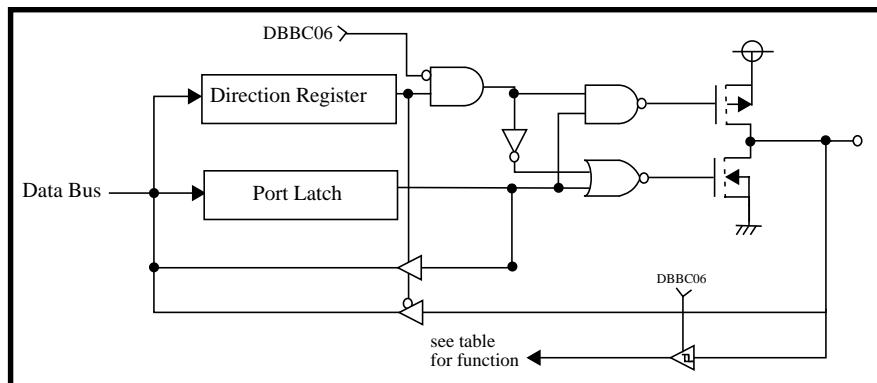


Figure 2-35. Port P5<sub>4</sub> ~ P5<sub>7</sub> Block Diagram

Table 2-2. Port P5<sub>4</sub> ~ P5<sub>7</sub> function

Pin	Function
5 <sub>4</sub>	$\overline{S}_0$
5 <sub>5</sub>	A <sub>0</sub>
5 <sub>6</sub>	$\overline{R}(E)$
5 <sub>7</sub>	$\overline{W}(R/\overline{W})$

### Port 6

Port 6 is an 8-bit general purpose I/O port that can be configured to access special second functions. The port acts as the data bus interface for the bus interface control block when DBBC06 is “1”. The port can be set up in any configuration in all three processor modes.

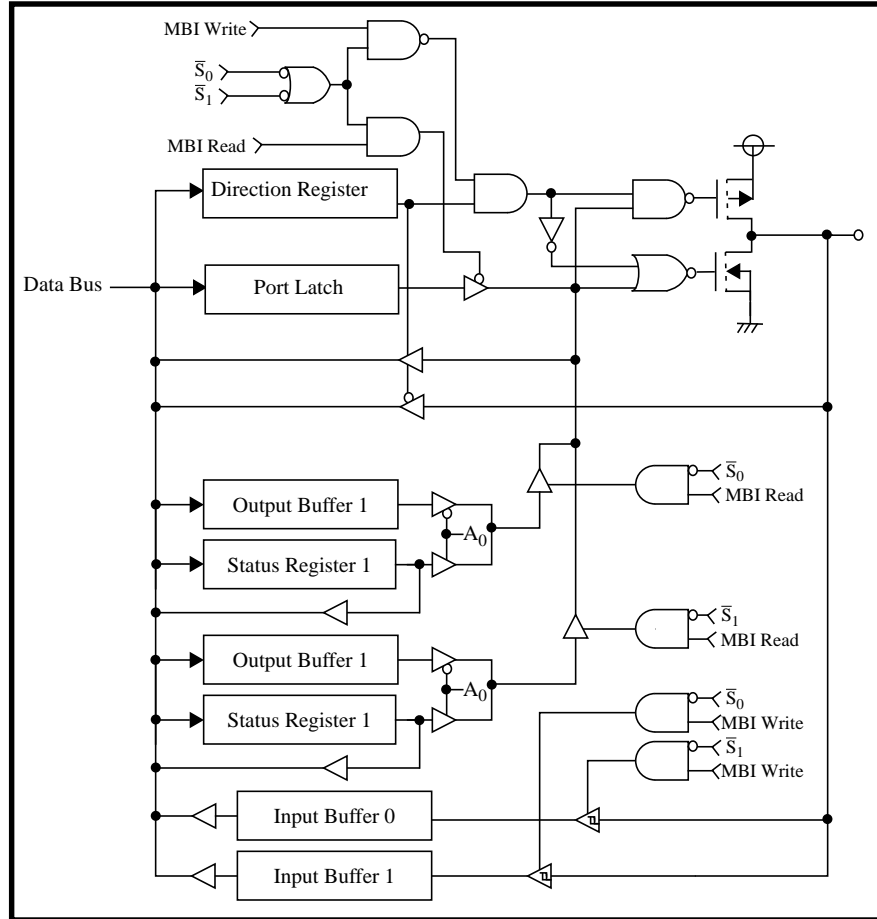


Figure 2-36. Port P6 Block Diagram

**Port 7**

Port 7 is a 5-bit general purpose I/O port that can be configured to access special second functions.

**Port 7<sub>0</sub>**

This pin is multiplexed with the USB start of frame pulse ( $\overline{\text{SOF}}$ ) output. When USBC6 is a “1”, this pin outputs the USB  $\overline{\text{SOF}}$ .

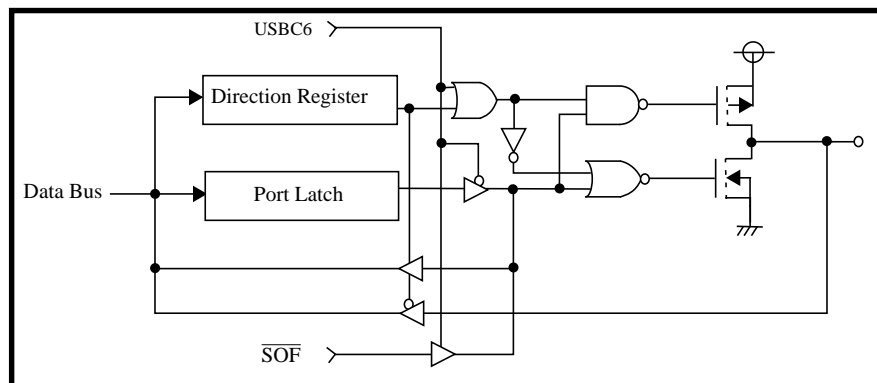


Figure 2-37. Port P7<sub>0</sub> Block Diagram



**Port 7<sub>1</sub>**

This pin is multiplexed with the  $\overline{\text{HOLD}}$  function. When the MCU is in memory expansion or microprocessor mode and CPMB5 is set to “1”, this pin operates as the  $\overline{\text{HOLD}}$  input as described in section 2.5.6 on page 2-23.

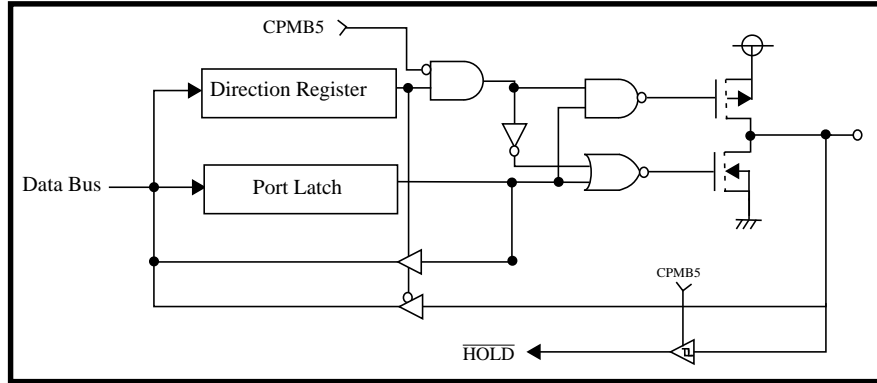


Figure 2-38. Port P7<sub>1</sub> Block Diagram

**Port 7<sub>2</sub>**

This pin is multiplexed with the  $\overline{\text{S}}_1$  input control signal from a master cpu. When DBBC17 is “1”, the pin takes on the function of the  $\overline{\text{S}}_1$  input control signal.

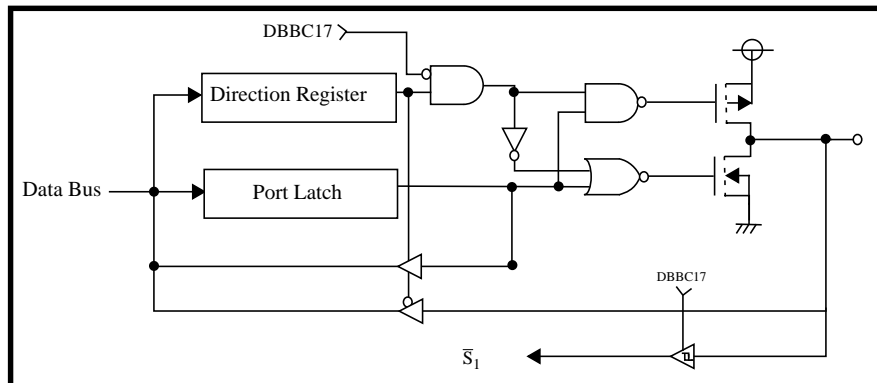


Figure 2-39. Port P7<sub>2</sub> Block Diagram

**Port 7<sub>3</sub>**

This pin is multiplexed with the  $\overline{\text{IBF}}_1$  output control signal for a master cpu and the  $\overline{\text{HLDA}}$  function. When DBBC11 and DBBC17 are “1”, the pin takes on the function of the  $\overline{\text{IBF}}_1$  output control signal. When the MCU is in memory expansion or microprocessor mode, CPMB5 is set to “1”, and the  $\overline{\text{IBF}}_1$  function is not enabled, this pin operates as the  $\overline{\text{HLDA}}$  output as described in section 2.5.6 on page 2-23.

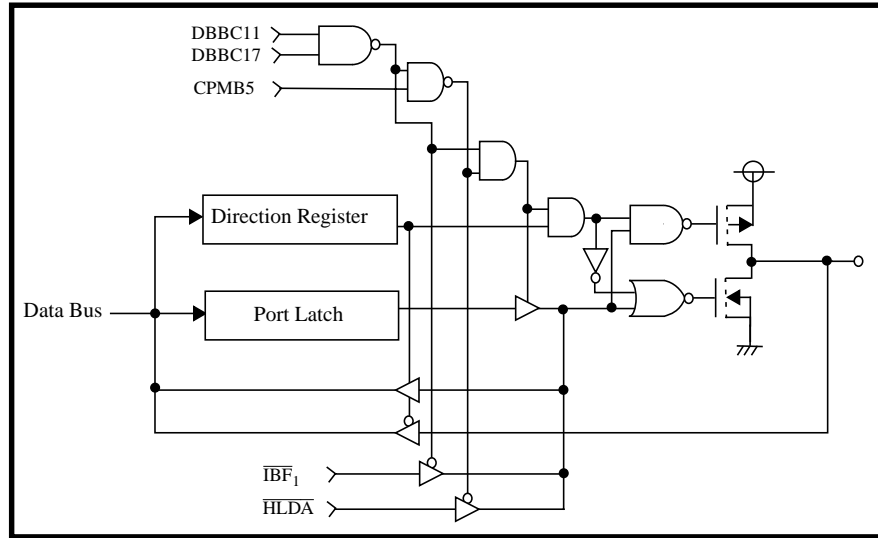


Figure 2-40. Port P7<sub>3</sub> Block Diagram

**Port 7<sub>4</sub>**

This pin is multiplexed with the  $\text{OBF}_1$  control pin for the bus interface control block. Pin 7<sub>4</sub> acts as  $\text{OBF}_1$  output to a master cpu when DBBC10 and DBBC17 are “1”.

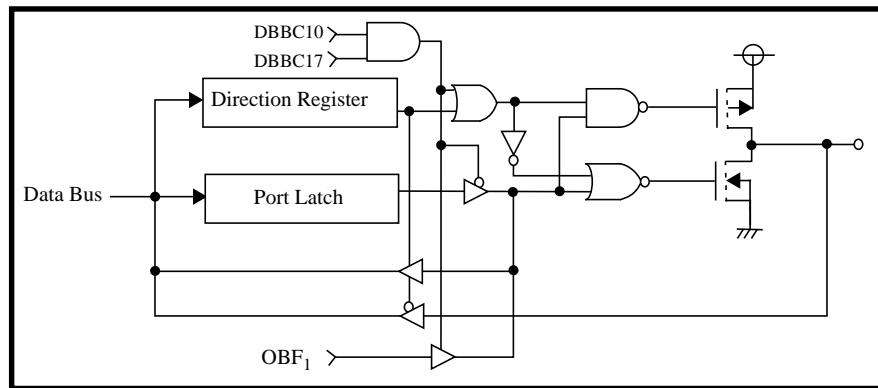


Figure 2-41. Port P7<sub>4</sub> Block Diagram

**Port 8**

Port 8 is an 8-bit general purpose I/O port that can be configured to access special second functions. The port can be set up in any configuration in all three processor modes.





**Port 8<sub>0</sub>**

This pin is multiplexed with the SIO  $\overline{\text{SRDY}}$  signal and the UART2 TxD signal. When UART2 is in transmit mode, the pin acts as the TxD output signal. When the pin is not being used as the UART2 TxD output and bit 4 of the SIO control register 1 (SIOCON1) is a “1”, the port acts as the SIO  $\overline{\text{SRDY}}$  output signal. If during this function, the SIO is configured in slave mode, this pin acts as a slave input from a master. See section 2.15.2 on page 2-102 for more SIO information.

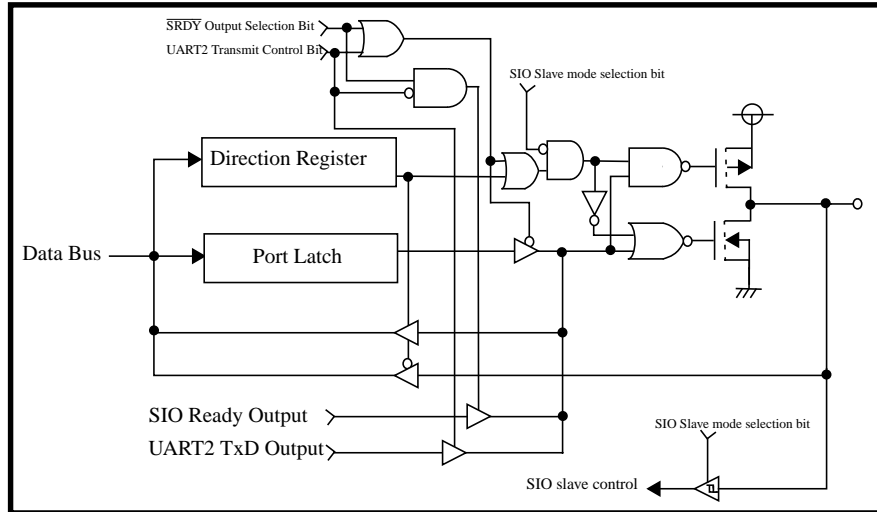


Figure 2-42. Port P8<sub>0</sub> Block Diagram

**Port 8<sub>1</sub>**

This pin is multiplexed with the SIO SCLK signal and the UART2 RxD signal. When UART2 is in receive mode, the pin acts as the RxD input signal. When the pin is not being used as the UART2 RxD input and bit 2 of the SIO control register 1 (SIOCON1) is a “1”, the port acts as the SIO SCLK signal. In this mode a “1” in bit 6 of SIOCON1 configures the pin to output SCLK whereas a “0” configures the pin to input SCLK.

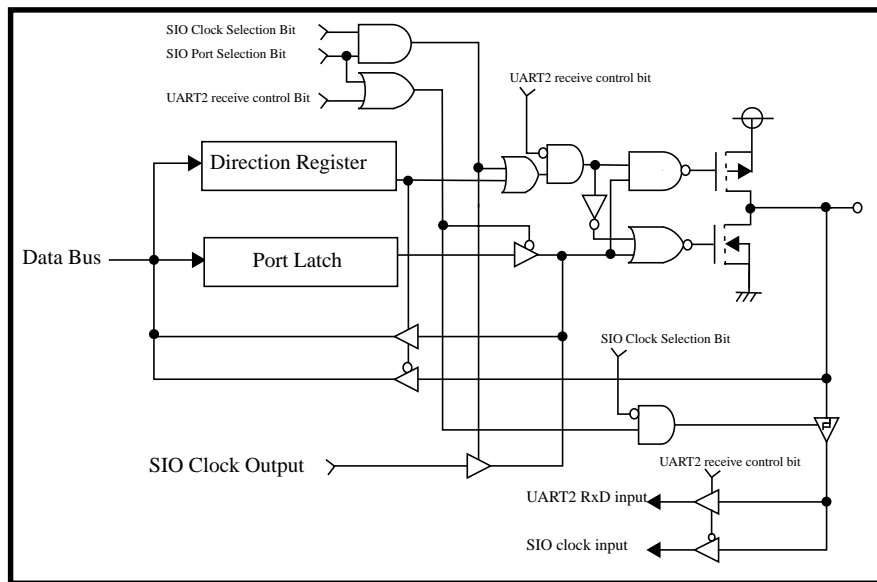


Figure 2-43. Port P8<sub>1</sub> Block Diagram

**Port 8<sub>2</sub>**

This pin is multiplexed with the SIO SRxD signal and the UART2  $\overline{\text{CTS}}$  signal. When bit 5 of the UART2 control register (U2CON) is a “1”, the port acts as the  $\overline{\text{CTS}}$  input signal. When the pin is not being used as the UART2  $\overline{\text{CTS}}$  input and bit 2 of the SIO control register 2 (SIOCON2) is a “1”, the port acts as the SIO SRxD input signal.

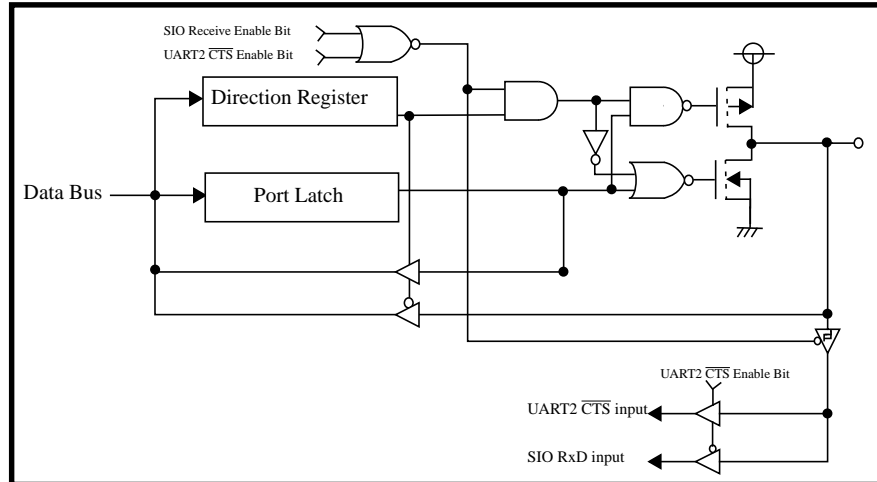


Figure 2-44. Port P8<sub>2</sub> Block Diagram

**Port 8<sub>3</sub>**

This pin is multiplexed with the SIO STxD signal and the UART2  $\overline{\text{RTS}}$  signal. When bit 6 of the UART2 control register (U2CON) is a “1”, the port acts as the  $\overline{\text{RTS}}$  output signal. When the pin is not being used as the UART2  $\overline{\text{RTS}}$  output and bit 3 of the SIO control register 1 (SIOCON1) is a “1”, the port acts as the SIO STxD output signal.

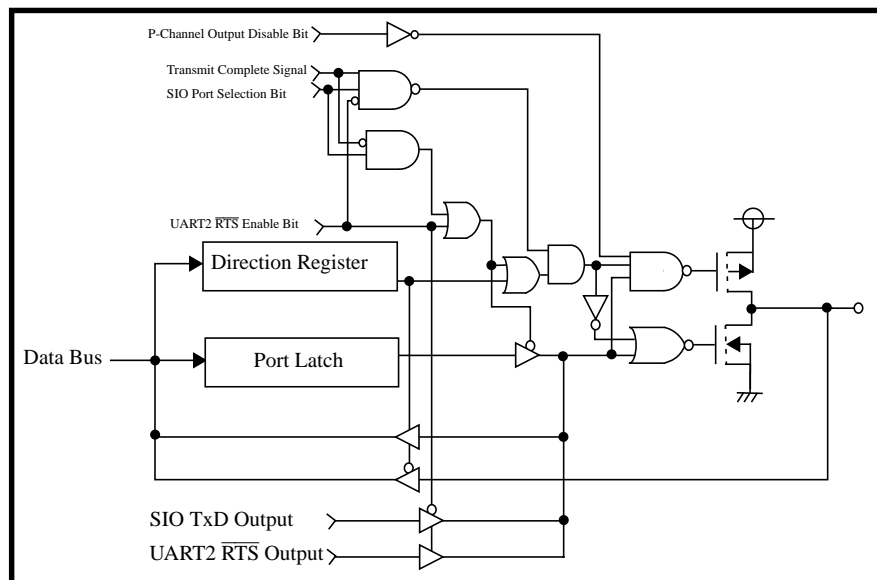


Figure 2-45. Port P8<sub>3</sub> Block Diagram



**Port 8<sub>4</sub>**

This pin is multiplexed with the UART1 Tx<sub>D</sub> signal. When UART1 is in transmit mode, the pin acts as the Tx<sub>D</sub> output signal.

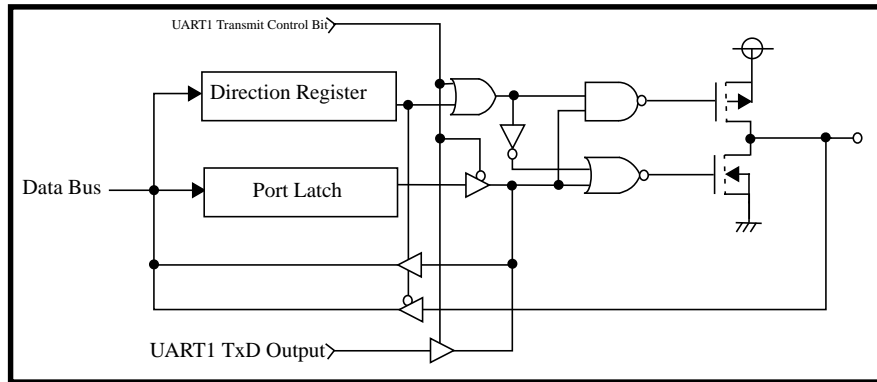


Figure 2-46. Port P8<sub>4</sub> Block Diagram

**Port 8<sub>5</sub>**

This pin is multiplexed with the UART1 Rx<sub>D</sub> signal. When UART1 is in receive mode, the pin acts as the Rx<sub>D</sub> input signal.

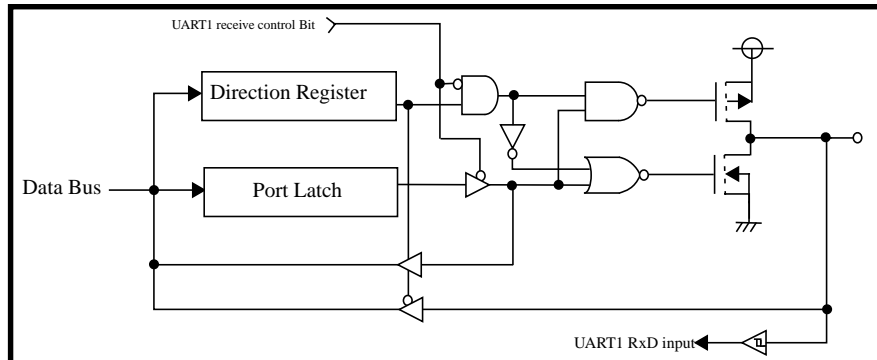


Figure 2-47. Port P8<sub>5</sub> Block Diagram

**Port 8<sub>6</sub>**

This pin is multiplexed with the UART1  $\overline{\text{CTS}}$  signal. When bit 5 of the UART1 control register (U1CON) is a “1”, the port acts as the  $\overline{\text{CTS}}$  input signal.

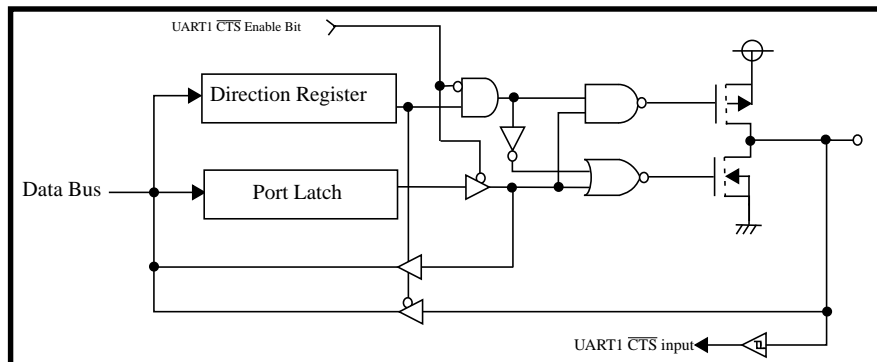


Figure 2-48. Port P8<sub>6</sub> Block Diagram

## Port 8<sub>7</sub>

This pin is multiplexed with the UART1  $\overline{\text{RTS}}$  signal. When bit 6 of the UART1 control register (U1CON) is a “1”, the port acts as the  $\overline{\text{RTS}}$  output signal.

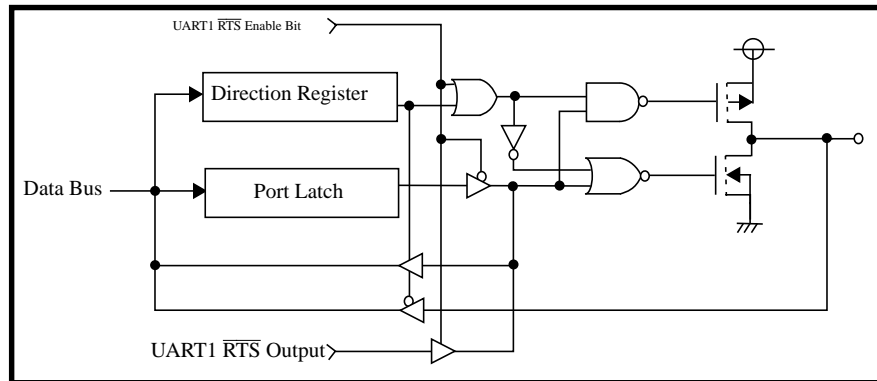


Figure 2-49. Port P8<sub>7</sub> Block Diagram

### 2.7.1.2 Power and Ground Pins

There are two  $V_{SS}$  and two  $V_{DD}$  pins that supply power to the MCU. There is also one analog  $V_{DD}$  ( $AV_{DD}$ ) and one analog  $V_{SS}$  ( $AV_{SS}$ ) pin for the analog circuits.

### 2.7.1.3 $CNV_{SS}$ 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 CPMA. However, while  $CNV_{SS}$  is connected  $V_{DD}$ , bit 1 of CPMA can not be overwritten. With  $CNV_{SS}$  connected to  $V_{SS}$ , the MCU enters single chip mode after a reset.

### 2.7.1.4 $X_{in}$ 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 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. The frequency of  $X_{in}$  can be  $\leq 48\text{MHz}$  with an external clock source and  $\leq 24\text{MHz}$  with a crystal.

### 2.7.1.5 $XC_{in}$ and $XC_{out}$ Pins

The  $P5_0/XC_{in}$  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. The frequency of  $XC_{in}$  can be  $\leq 5\text{MHz}$  with an external clock source or  $32\text{KHz}$  with a crystal.

### 2.7.1.6 $\overline{\text{RESET}}$ Pin

The MCU is reset by holding  $\overline{\text{RESET}}$  low for at least  $2\mu\text{s}$  before returning to high.

**2.7.1.7 RDY Pin**

The P3<sub>0</sub>/RDY pin is used to control the slow memory wait function of the MCU. For a detailed description of this pin, see section 2.5.5 on page 2-19.

**2.7.1.8 DMAout Pin**

When the chip is in microprocessor or memory expansion mode, the DMAout (P3<sub>3</sub>/DMAout) pin goes high during a DMA transfer.

**2.7.1.9  $\Phi_{out}$  Pin**

When the MCU is in microprocessor or memory expansion mode, pin P3<sub>4</sub> outputs the internal system clock  $\Phi_{out}$ . When the STP or WIT instructions are executed, the output of the  $\Phi_{out}$  pin stops at a high level.

**2.7.1.10 SYNC<sub>out</sub> Pin**

When the MCU is in microprocessor or memory expansion mode, the SYNC<sub>out</sub> pin outputs a signal that is high for one-half cycle of  $\Phi_{out}$  every time an OpCode is fetched.

**2.7.1.11  $\overline{RD}$  and  $\overline{WR}$  Pins**

When the MCU is in microprocessor or memory expansion mode, a read control signal is output from the  $\overline{RD}$  pin and write control signal is output from the  $\overline{WR}$  pin (P3<sub>6</sub>/ $\overline{WR}$  and P3<sub>7</sub>/ $\overline{RD}$ ). 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. These signals are active for both internal and external accesses.

**2.7.1.12 LPF Pin**

When the Frequency Synthesizer is active, the LPF pin is the loop filter for the Frequency Synthesizer.

**2.7.1.13 USB D+/D- Pins**

These two pins are used as the data transmission/reception lines for the USB core.

**2.7.1.14 Ext. Cap Pin**

When the USB transceiver voltage converter is used, an external capacitor must be connected to this pin.

### 2.7.2 Port Control Register

This device is equipped with a port control register to turn on and off the slew rate control and to control the input levels for port 2 and the MBI pins. (see Figure 2-50.).

MSB 7	PTC7	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0	LSB 0	Address: 0010 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
			PTC0	Slew Rate Control Bit Ports 0-3 (bit 0)							
				0: Disabled							
				1: Enabled							
			PTC1	Slew Rate Control Bit Port 4 (bit 1)							
				0: Disabled							
				1: Enabled							
			PTC2	Slew Rate Control Bit Port 5 (bit 2)							
				0: Disabled							
				1: Enabled							
			PTC3	Slew Rate Control Bit Port 6 (bit 3)							
				0: Disabled							
				1: Enabled							
			PTC4	Slew Rate Control Bit Port 7 (bit 4)							
				0: Disabled							
				1: Enabled							
			PTC5	Slew Rate Control Bit Port 8 (bit 5)							
				0: Disabled							
				1: Enabled							
			PTC6	Port 2 Input Level Select Bit (bit 6)							
				0: Reduced VIH <sub>L</sub> level input							
				1: CMOS level input							
			PTC7	Master Bus Input Level Select Bit (bit 7)							
				0: CMOS level input							
				1: TTL level input							

Figure 2-50. Port Control Register

### 2.7.3 Port 2 Pull-up Control Register

This device is equipped with internal pull-ups on Port 2 that can be enabled by software. Each bit of the pull-up control register controls a corresponding pin of Port 2. 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	PUP2 <sub>7</sub>	PUP2 <sub>6</sub>	PUP2 <sub>5</sub>	PUP2 <sub>4</sub>	PUP2 <sub>3</sub>	PUP2 <sub>2</sub>	PUP2 <sub>1</sub>	PUP2 <sub>0</sub>	LSB 0	Address: 0012 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
			PUP2 <sub>0</sub>	Pull-up Control for Port 2 (bit 0)							
				0: Disabled							
				1: Enabled							
			PUP2 <sub>1</sub>	Pull-up Control for Port 2 (bit 1)							
				0: Disabled							
				1: Enabled							
			PUP2 <sub>2</sub>	Pull-up Control for Port 2 (bit 2)							
				0: Disabled							
				1: Enabled							
			PUP2 <sub>3</sub>	Pull-up Control for Port 2 (bit 3)							
				0: Disabled							
				1: Enabled							
			PUP2 <sub>4</sub>	Pull-up Control for Port 2 (bit 4)							
				0: Disabled							
				1: Enabled							
			PUP2 <sub>5</sub>	Pull-up Control for Port 2 (bit 5)							
				0: Disabled							
				1: Enabled							
			PUP2 <sub>6</sub>	Pull-up Control for Port 2 (bit 6)							
				0: Disabled							
				1: Enabled							
			PUP2 <sub>7</sub>	Pull-up Control for Port 2 (bit 7)							
				0: Disabled							
				1: Enabled							

Figure 2-51. Pull-up Control Register



## 2.8 Interrupt Control Unit

Address	Description	Acronym and Value at Reset
0002 <sub>16</sub>	Interrupt request register A	IREQA=00
0003 <sub>16</sub>	Interrupt request register B	IREQB=00
0004 <sub>16</sub>	Interrupt request register C	IREQC=00
0005 <sub>16</sub>	Interrupt control register A	ICONA=00

Address	Description	Acronym and Value at Reset
0006 <sub>16</sub>	Interrupt control register B	ICONB=00
0007 <sub>16</sub>	Interrupt control register C	ICONC=00
0011 <sub>16</sub>	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 2-3 for the interrupt sources, jump destination addresses, interrupt priorities, and section references for the interrupt request sources.

### 2.8.1 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 to at “1” 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 a “0”, the interrupt request bit is not seen; and when the enable bit is a “1”, the interrupt request is seen.

The interrupt request register configurations for the 23 maskable interrupts are shown in Figure 2-52., Figure 2-53., and Figure 2-54. The interrupt control register configurations for the 23 maskable interrupts are shown in Figure 2-55., Figure 2-56., and Figure 2-57.

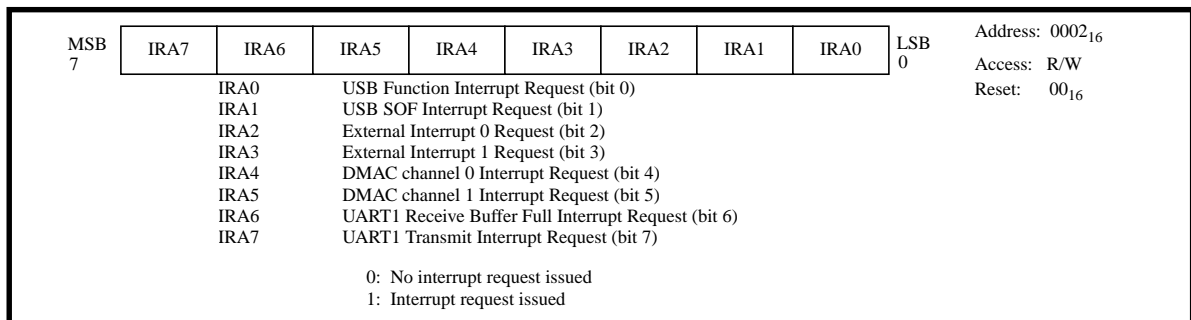


Figure 2-52. IREQA Configuration

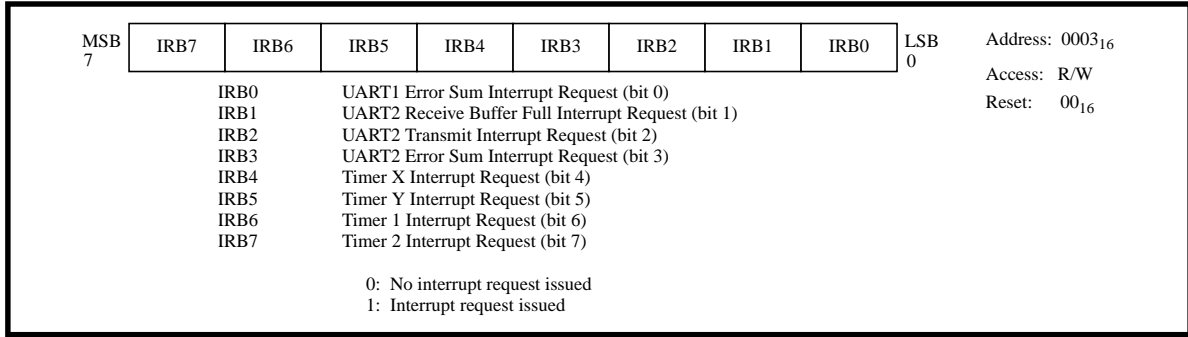


Figure 2-53. IREQB Configuration

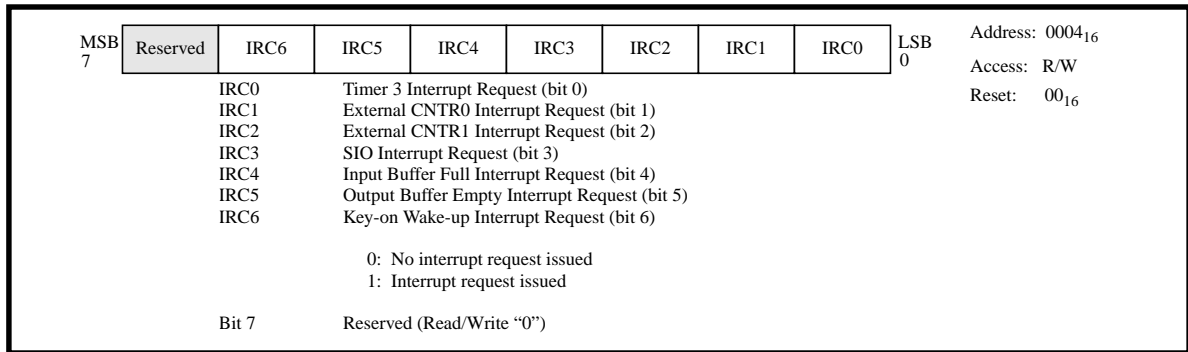


Figure 2-54. IREQC Configuration

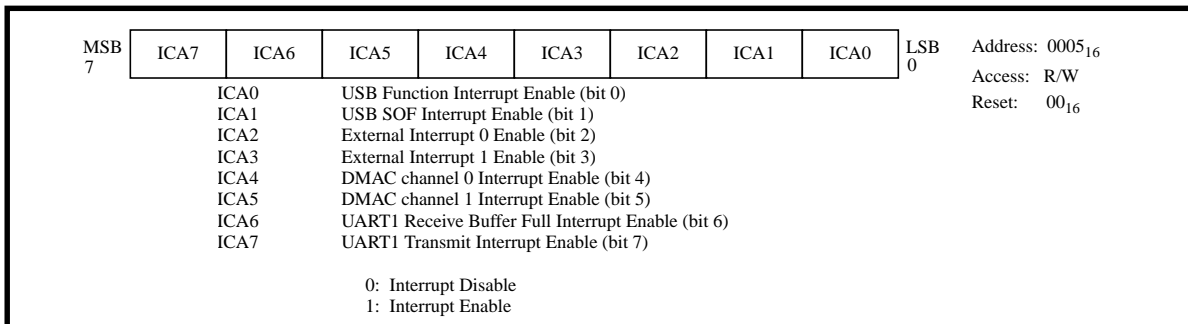


Figure 2-55. ICONA Configuration

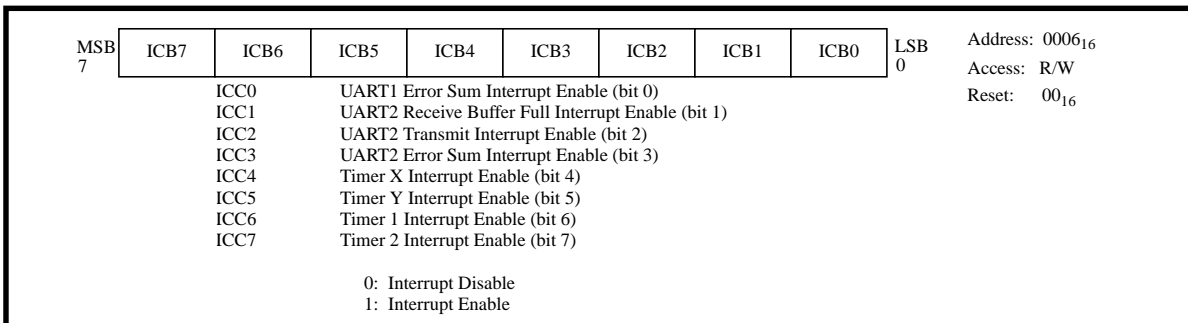


Figure 2-56. ICONB Configuration





MSB 7	Reserved	ICC6	ICC5	ICC4	ICC3	ICC2	ICC1	ICC0	LSB 0	Address: 0007 <sub>16</sub>
		ICC0	Timer 3 Interrupt Enable (bit 0)							
		ICC1	External CNTR0 Interrupt Enable (bit 1)							Reset: 00 <sub>16</sub>
		ICC2	External CNTR1 Interrupt Enable (bit 2)							
		ICC3	SIO Interrupt Enable (bit 3)							
		ICC4	Input Buffer Full Interrupt Enable (bit 4)							
		ICC5	Output Buffer Empty Interrupt Enable (bit 5)							
		ICC6	Key-on Wake-up Interrupt Enable (bit 6)							
			0: Interrupt disabled 1: Interrupt enabled							
	Bit 7		Reserved (Read/Write "0")							

Figure 2-57. ICONC Configuration

The interrupt polarity register allows the user to select the external interrupt edge which triggers the interrupt request. The configuration of the polarity register for the external interrupts is shown in Figure 2-58.

MSB 7	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	INT1 Pol	INT0 Pol	LSB 0	Address: 0011 <sub>16</sub>
							INT0 Pol	INT0 Interrupt Edge Selection Bit 0: Falling edge selected. 1: Rising edge selected.		
							INT1 Pol	INT1 Interrupt Edge Selection Bit 0: Falling edge selected. 1: Rising edge selected.		Reset: 00 <sub>16</sub>
							Bits 2-7	Reserved (Read/Write "0")		

Figure 2-58. IPOL Configuration

Table 2-3 Interrupt Vector Table

Priority	Interrupt	Jump Destination Storage Address (Vector Address)		Remarks				Reference
		High-order Byte	Low-order Byte					
1	RSRV1	FFFF	FFFE	Reserved for factory use				
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	Section 2.9.2.1
5	SOF	FFF7	FFF6	USB SOF Interrupt	1			Section 2.9.2.2
6	INT0	FFF5	FFF4	External Interrupt 0	2			Section 2.8.1
7	INT1	FFF3	FFF2	External Interrupt 1	3			Section 2.8.1
8	DMA1	FFF1	FFF0	DMAC Channel 0 Interrupt	4			Section 2.11
9	DMA2	FFEF	FFEE	DMAC Channel 1 Interrupt	5			Section 2.11
10	U1RBF	FFED	FFEC	UART1 Receiver Buffer Full	6			Section 2.14.7
11	U1TX	FFEB	FFEA	UART1 Transmit Interrupt	7	MSB		Section 2.14.7
12	U1ES	FFE9	FFE8	UART1 Error Sum Interrupt	0	LSB		Section 2.14.7
13	U2RBF	FFE7	FFE6	UART2 Receiver Buffer Full	1			Section 2.14.7
14	U2TX	FFE5	FFE4	UART2 Transmit Interrupt	2		Section 2.14.7	
15	U2ES	FFE3	FFE2	UART2 Error Sum Interrupt	3		Section 2.14.7	
16	TX	FFE1	FFE0	Timer X Interrupt	4		IREQB and ICONB	Section 2.13
17	TY	FFDF	FFDE	Timer Y Interrupt	5			Section 2.13
18	T1	FFDD	FFDC	Timer 1 Interrupt	6			Section 2.13
19	T2	FFDB	FFDA	Timer 2 Interrupt	7	MSB		Section 2.13
20	T3	FFD9	FFD8	Timer 3 Interrupt	0	LSB	IREQC and ICONC	Section 2.13
21	CNTR0	FFD7	FFD6	External CNTR0 Interrupt	1			Section 2.13.1.6
22	CNTR1	FFD5	FFD4	External CNTR1 Interrupt	2			Section 2.13.2
23	SIO	FFD3	FFD2	SIO Interrupt	3			Section 2.15
24	IBF	FFD1	FFD0	Input Buffer Full Interrupt	4			Section 2.10
25	OBE	FFCF	FFCE	Output Buffer Empty Interrupt	5			Section 2.10
26	KEY	FFCD	FFCC	Key-on Wake Up	6	MSB		Section 2.18
27	BRK	FFCB	FFCA	BRK Instruction (Non-Maskable)				



### 2.8.2 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 to a “1”, its interrupt enable bit is set to a “1”, and the interrupt inhibit flag I is set low. The I flag is used to disable all maskable interrupts. When this bit is set to a “1”, only a BRK instruction or a user Reset can cause an interrupt service request to be generated. Figure 2-59 is a simplified version of the logic that controls whether an interrupt service request is generated.

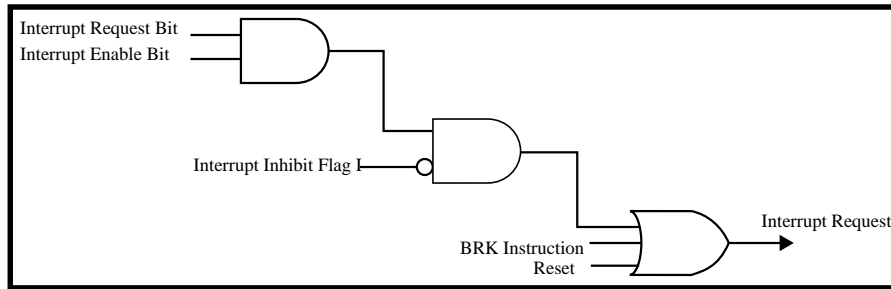


Figure 2-59. Interrupt Service Request Control Logic

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 2-60.)

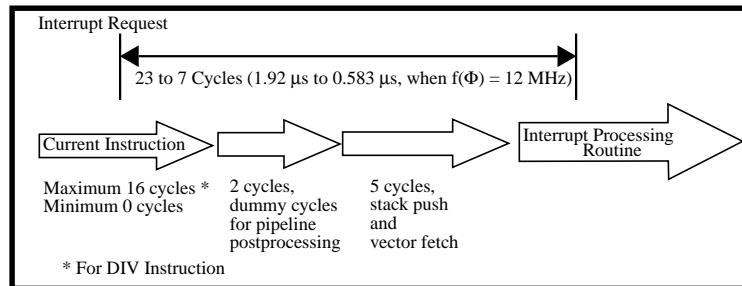


Figure 2-60. Execution Time Prior to Interrupt Service Routine

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 decrementing the stack pointer by one as illustrated in Figure 2-2. The I flag is set to a “1” 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 2-61.

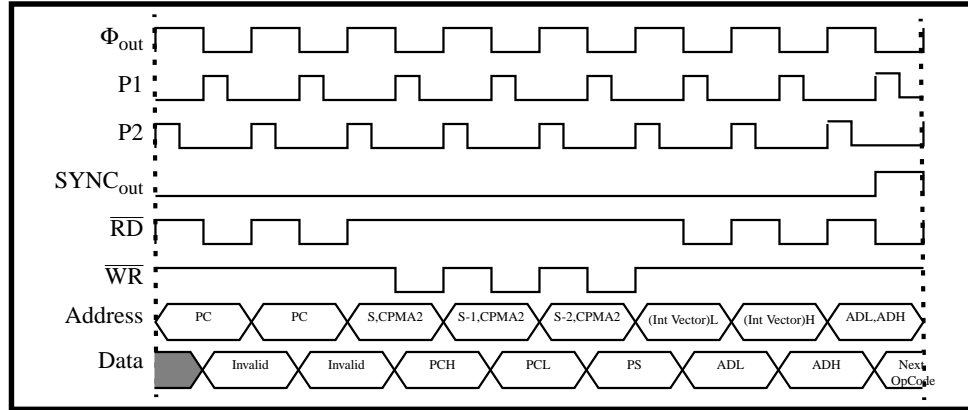


Figure 2-61. Interrupt Cycle Timing

See Figure 2-62. for the stack and program counter modifications that occur when an interrupt request is serviced.

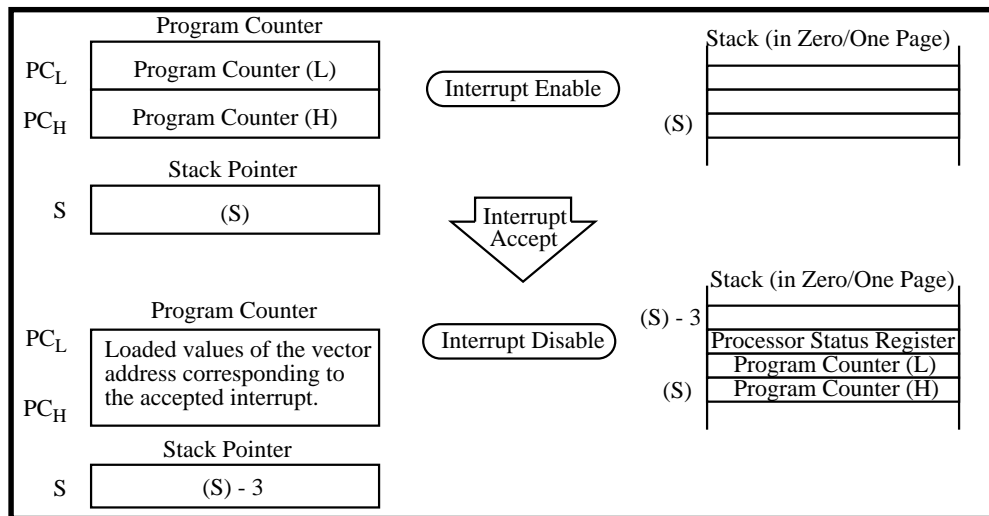


Figure 2-62. Stack Pointer and Program Counter Modifications During Interrupt Service Sequence

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.



## 2.9 Universal Serial Bus

Address	Description	Acronym and Value at Reset
0013 <sub>16</sub>	USB Control Register	USBC=00
0050 <sub>16</sub>	USB Address Register	USBA=00
0051 <sub>16</sub>	USB Power Management Register	USBPM=00
0052 <sub>16</sub>	USB Interrupt Status Register 1	USBIS1=00
0053 <sub>16</sub>	USB Interrupt Status Register 2	USBIS2=00
0054 <sub>16</sub>	USB Interrupt Enable Register 1	USBIE1=FF
0055 <sub>16</sub>	USB Interrupt Enable Register 2	USBIE2=33
0056 <sub>16</sub>	USB Frame Number Low Register	USBSOFL=00
0057 <sub>16</sub>	USB Frame Number High Register	USBSOFH=00
0058 <sub>16</sub>	USB Endpoint Index	USBINDEX=00
0059 <sub>16</sub>	USB Endpoint x IN CSR	IN_CSR=00

Address	Description	Acronym and Value at Reset
005A <sub>16</sub>	USB Endpoint x OUT CSR	OUT_CSR=00
005B <sub>16</sub>	USB Endpoint x IN MAXP	IN_MAXP
005C <sub>16</sub>	USB Endpoint x OUT MAXP	OUT_MAXP
005D <sub>16</sub>	USB Endpoint x OUT WRT CNT Low	WRT_CNTL=00
005E <sub>16</sub>	USB Endpoint x OUT WRT CNT High	WRT_CNTH=00
005F <sub>16</sub>	Reserved	
0060 <sub>16</sub>	USB Endpoint 0 FIFO	USBFIFO0=N/A
0061 <sub>16</sub>	USB Endpoint 1 FIFO	USBFIFO1=N/A
0062 <sub>16</sub>	USB Endpoint 2 FIFO	USBFIFO2=N/A
0063 <sub>16</sub>	USB Endpoint 3 FIFO	USBFIFO3=N/A
0064 <sub>16</sub>	USB Endpoint 4 FIFO	USBFIFO4=N/A

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

- Complete USB Specification (version 1.0) Compatibility
- Error Handling capabilities
- 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
  - Five independent IN and five independent OUT endpoints
- Complete Device Configuration
- Supports All Device Commands
- Supports Full-Speed Functions
- Support of All USB Transfer Types:
  - Isochronous
  - Bulk
  - Control
  - Interrupt
- Suspend/Resume Operation
- On-chip USB Transceiver with voltage converter
- Start-of-frame interrupt and output pin

## 2.9.1 USB Function Control Unit (USB FCU)

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 2-63.)

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 (MCI)
- USB Transceiver

### 2.9.1.1 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.

### 2.9.1.2 Generic Function Interface

The GFI handles the all USB standard requests from the host through the control endpoint (endpoint zero), handles Bulk, Isochronous and Interrupt transfers through endpoints 1-4. The GFI handles read pointer reversal for re-transmit the current data set; write pointer reversal for re-receive the last data set; data toggle synchronization.

### 2.9.1.3 Serial Engine Interface Unit

The SIU block decodes the Address and Endpoint fields from the USB host.

### 2.9.1.4 Microcontroller Interface Unit

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

### 2.9.1.5 USB Transceiver

The USB transceiver, designed to interface with the physical layer of the USB, is compliant with the USB Specification (version 1.0) for high speed devices. It consists of two 6-ohm drivers, a receiver, and schmitt triggers for single-ended receive signals.

The transceiver also includes a voltage converter. The voltage converter can supply 3.0 - 3.6V to the transmitter when the rest of the chip (CPU, USB FCU, etc...) operates at 4.15 - 5.25V. To enable the voltage converter, set bit 4 of the USB Control Register (USBC) to a "1". To disable the voltage converter, set bit 4 of the USBC to a "0". Refer to section 4.5 "USB Transceiver" for more detailed information.

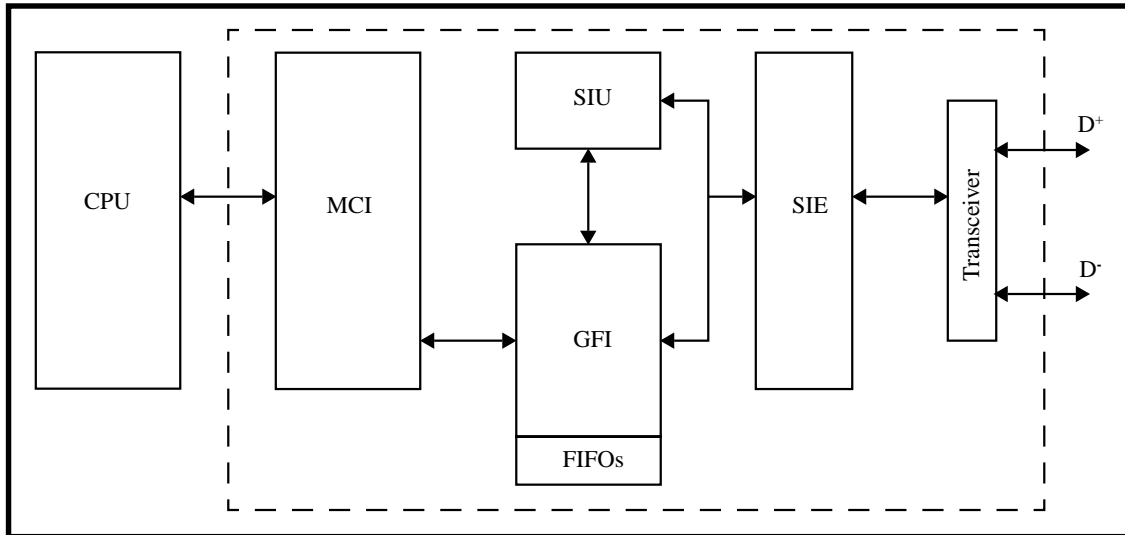


Figure 2-63. USB Function Control Unit Block Diagram

## 2.9.2 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 (ISO) data.

### 2.9.2.1 USB Function Interrupt

Endpoint 1-4, each has two interrupt status bits associated with it to control the data transfer or to report a STALL/UNDER\_RUN/OVER\_RUN condition. The EP<sub>x</sub>\_OUT\_INT bit is set when the USB FCU successfully receives a packet of data, or sets the FORCE\_STALL bit, or the OVER\_RUN bit of the Endpoint x OUT CSR. The EP<sub>x</sub>\_IN\_INT bit is set when the USB FCU successfully sends a packet of data, or sets the UNDER\_RUN bit of the Endpoint x IN CSR. Endpoint 0 - the control endpoint - has one interrupt status bit associated with it to control the data transfer or report a STALL condition. The EP0\_INT is set when the USB FCU successfully receives/sends a packet of data, or sets the SETUP\_END bit, the FORCE\_STALL bit, or clears the DATA\_END bit in the Endpoint 0 IN CSR. Each endpoint 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 USB FCU sets the interrupt status bits. The CPU writes a "1" to clear the corresponding status bit. By writing back the same value it read, the CPU will clear all the existing interrupts. The CPU must read then write both status registers, writing status register 1 first and status register 2 second to guarantee proper operation.

The suspend interrupt status bit is set if a USB suspend signaling is received. If the device is in suspend mode, the resume interrupt status bit is set when a USB resume signaling 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 USB reset signaling is received. When this bit is set, all USB internal registers will be reset to their default values 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 (applicable to endpoints used for isochronous data transfer), when an overrun condition occurs in an endpoint (CPU is too slow to unload the data from the FIFO), or when an underrun condition occurs in an endpoint (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.

### 2.9.2.2 USB SOF Interrupt

The USB SOF (Start-Of-Frame) interrupt is used to control the transfer of isochronous data. The USB FCU generates a start-of-frame interrupt when 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.

## 2.9.3 USB Endpoint FIFOs

The USB FCU has an IN (transmit) FIFO and an OUT (receive) FIFO for each endpoint. Both FIFOs support up to two separate data sets of variable size (except Endpoint 0), and provide the ability of back-to-back transmission and reception. Throughout this specification, the terms "IN FIFO" and "OUT FIFO" refer these FIFOs associated with the current endpoint as specified by the Endpoint Index Register.

In the event of a bad transmission/reception, the USB FCU handles all the read/write pointer reversal and data set management tasks when it is applicable.

### 2.9.3.1 IN (Transmit) FIFOs

The CPU/DMA writes data to the endpoint's IN FIFO location specified by the FIFO write pointer, which automatically increments by "1" after a write. The CPU/DMA should only write data to the IN FIFO if the IN\_PKT\_RDY bit of the IN CSR is a "0".

**Endpoint 0 IN FIFO Operation:** The CPU writes a "1" to the IN\_PKT\_RDY bit after it finishes writing a packet of data to the IN FIFO. The USB FCU clears the IN\_PKT\_RDY bit after the packet is successfully transmitted to the host (ACK is received from the host) or the SETUP\_END bit of the IN CSR is set to a "1".

**Endpoint 1-4 IN FIFO Operation when AUTO\_SET (bit 7 of IN CSR) = "0":**

MAXP > half of the IN FIFO size: The CPU writes a "1" to IN\_PKT\_RDY bit after the CPU/DMA finishes writing a packet of data to the IN FIFO. The USB FCU clears the IN\_PKT\_RDY bit after the packet is successfully transmitted to the host (ACK is received from the host).

MAXP <= half of the IN FIFO size: The CPU writes a "1" to the IN\_PKT\_RDY bit after the CPU/DMA finishes writing a packet of data to the IN FIFO. The USB FCU clears the IN\_PKT\_RDY bit as soon as the IN FIFO is ready to accept another data packet (The FIFO can hold up to two data packets at the same time in this configuration, for back-to-back transmission). Since the set and the clear operations could be as fast as 83ns (one 12MHz clock period) apart from each other, the set may be transparent to the user.

**Endpoint 1-4 IN FIFO Operation when AUTO\_SET (bit 7 of IN CSR) = "1":**

MAXP > half of the IN FIFO size: When the number of bytes of data equal to the MAXP (maximum packet size) is written to the IN FIFO by the CPU/DMA, the USB FCU sets the IN\_PKT\_RDY bit to a '1' automatically. The USB FCU clears the IN\_PKT\_RDY bit after the packet is successfully transmitted to the host (ACK is received from the host).





MAXP  $\leq$  half of the IN FIFO size: When the number of bytes of data equal to the MAXP (maximum packet size) is written to the IN FIFO by the CPU/DMAC, the USB FCU sets the IN\_PKT\_RDY bit to a '1' automatically. The USB FCU clears the IN\_PKT\_RDY bit as soon as the IN FIFO is ready to accept another data packet (The FIFO can hold up to two data packets at the same time in this configuration, for back-to-back transmission). Since the set and the clear operations could be as fast as 83ns (one 12MHz clock period) apart from each other, the set may be transparent to the user.

A software or a hardware flush acts as if a packet is being successfully transmitted out to the host. If there is one packet in the IN FIFO, a flush will cause the IN FIFO to be empty, if there are two packets in the IN FIFO, a flush will cause the older packet to be flushed out from the IN FIFO. Flush will update the IN FIFO status (IN\_PKT\_RDY and TX\_NOT\_EMPTY bits).

The status of the endpoint 1-4 IN FIFO for both of the above cases, could be obtained from the IN CSR as follows:

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 if MAXP $\leq$ half of the FIFO size.
1	0	Invalid
1	1	Two data packets in TX FIFO if MAXP $\leq$ half of the FIFO size or One data packet in TX FIFO if MAXP $>$ half of the FIFO size

**Interrupt Endpoints:** Any endpoint can be used for interrupt transfers. For normal interrupt transfers, the interrupt transactions behave the same as bulk transactions, i.e., no special setting is required. The IN endpoints may also be used to communicate rate feedback information for certain types of isochronous functions. This is done by setting the INTPT bit in the IN CSR register of the corresponding endpoint. When the INTPT bit is set, the data toggle bits will be changed after each packet is sent to the host without regard to the presence or type of handshake packet.

The following outlines the operation sequence for an IN endpoint used to communicate rate feedback information:

1. Set MAXP  $>$  1/2 of the endpoint's FIFO size;
2. Set INTPT bit of the IN CSR;
3. Flush the old data in the FIFO;
4. Load interrupt status information and set IN\_PKT\_RDY bit in the IN CSR;
5. Repeat steps 3 & 4 for all subsequent interrupt status updates.

### 2.9.3.2 Out (Receive) FIFOs

The USB FCU writes data to the endpoint's OUT FIFO location specified by the FIFO write pointer, which automatically increments by one after a write. When the USB FCU has successfully received a data packet, it sets the OUT\_PKT\_RDY bit to a "1" in the OUT CSR. The CPU/DMAC should only read data from the OUT FIFO if the OUT\_PKT\_RDY bit of the OUT CSR is a "1", with the exception of endpoint 1 (see detail description below).

**Endpoint 0 OUT FIFO Operation:** The USB FCU sets the OUT\_PKT\_RDY bit to a '1' after it has successfully received a packet of data from the host. The CPU writes a "0" to the OUT\_PKT\_RDY bit after the packet of data is unloaded from the OUT FIFO by the CPU.

**Endpoint 1-4 OUT FIFO Operation when AUTO\_CLR (bit 7 of OUT CSR) = "0":**



MAXP > half of the OUT FIFO size: The USB FCU sets the OUT\_PKT\_RDY bit to a “1” after it has successfully received a packet of data from the host. The CPU writes a “0” to the OUT\_PKT\_RDY bit after the packet of data is unloaded from the OUT FIFO by the CPU/DMAC.

MAXP <= half of the OUT FIFO size: The USB FCU sets the OUT\_PKT\_RDY bit to a “1” after it has successfully received a packet of data from the host. The CPU writes a “0” to the OUT\_PKT\_RDY bit after the packet of data is unloaded from the OUT FIFO by the CPU/DMAC. In this configuration, the FIFO can hold upto two data packets at the same time, for back-to-back reception. Therefore, the OUT\_PKT\_RDY bit may remain set after the CPU writes a “0” to it if there is another packet in the OUT FIFO.

**Endpoint 1-4 OUT FIFO Operation when AUTO\_CLR (bit 7 of OUT CSR) = “1”:**

MAXP > half of the OUT FIFO size: The USB FCU sets the OUT\_PKT\_RDY bit to a “1” after it has successfully received a packet of data from the host. The USB FCU clears the OUT\_PKT\_RDY bit to a ‘0’ automatically when the number of bytes of data equal to the MAXP (maximum packet size) is unloaded from the OUT FIFO by the CPU/DMAC.

MAXP <= half of the OUT FIFO size: The USB FCU sets the OUT\_PKT\_RDY bit to a “1” after it has successfully received a packet of data from the host. The USB FCU clears the OUT\_PKT\_RDY bit to a “0” automatically when the number of bytes of data equal to the MAXP (maximum packet size) is unloaded from the OUT FIFO by the CPU/DMAC. In this configuration, the FIFO can hold up to two data packets at the same time, for back-to-back reception. Therefore, the OUT\_PKT\_RDY bit may remain set after one packet (size equal to MAXP) of data is unloaded if there is another packet in the OUT FIFO.

A software flush acts as if a packet is being unloaded from the OUT FIFO. If there is one packet in the OUT FIFO, a flush will cause the OUT FIFO to be empty, if there are two packets in the OUT FIFO, a flush will cause the older packet to be flushed out from the OUT FIFO.

**Special case for OUT endpoint 1:** In addition to the OUT FIFO operations described above, the DMAC can also start unloading the OUT FIFO as soon as there is data in it (byte-by-byte transfer). This feature should only be used with ISO transfers. See section 2.11 "Direct Memory Access Controller" on page 2-69 for details.

## 2.9.4 USB Special Function Registers

The MCU controls 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: 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 endpoint-indexed SFR, the target endpoint number should be written to the Endpoint Index Register first. The lower 3 bits (EPINDX2:0) of the Endpoint Index Register are used for endpoint selection.

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

Some USB special function registers have a mix of read/write, read only, and write only register bits. Additionally, the bits may be configured to allow the user to write only a “0” or a “1” to individual bits. When accessing these registers, writing a “0” to a register that can only be set to a “1” by the CPU will have no affect on that register bit. Each figure and description of the special function registers will detail this operation.

The **USB Control Register**, shown in Figure 2-64, is used to control the USB FCU. This register is not reset by a USB reset signaling. After the USB is enabled (USBC7 set to “1”), a minimum delay of 250 ns (three 12Mhz clock periods) is needed before performing any other USB register read/write operations.



MSB 7	USBC7	USBC6	USBC5	USBC4	USBC3	Reserved	USBC1	Reserved	LSB 0	Address: 0013 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
	Bit 0		Reserved (Read/Write "0")							
	USBC1		USB Default State Selection Bit (bit 1) 0: In default state after powerup/reset 1: In default state after received the USB reset signaling							
	Bit 2		Reserved (Read/Write "0")							
	USBC3		Transceiver Voltage Converter High/Low Current Mode Selection Bit (bit 3) 0: High current mode 1: Low current mode							
	USBC4		USB Transceiver Voltage Converter Enable Bit (bit 4) 0: USB transceiver voltage converter disabled 1: USB transceiver voltage converter enabled							
	USBC5		USB Clock Enable Bit (bit 5) 0: 48 MHz clock to the USB block is disabled. 1: 48 MHz clock to the USB block is enabled.							
	USBC6		USB $\overline{\text{SOF}}$ Port Select Bit (bit 6) 0: USB $\overline{\text{SOF}}$ output is disabled. P7 <sub>0</sub> is used as GPIO pin. 1: USB $\overline{\text{SOF}}$ output is enabled							
	USBC7		USB Enable Bit (bit 7) 0: USB block is disabled, all USB internal registers are held at their default values. 1: USB block is enabled							

Figure 2-64. USB Control Register

The **USB Function Address Register**, shown in Figure 2-65, maintains the 7-bit USB address assigned by the host. The USB FCU uses this register value to decode USB token packet addresses. At reset, when the device is not yet configured, the value is 00<sub>16</sub>.

MSB 7	Reserved	FUNAD6	FUNAD5	FUNAD4	FUNAD3	FUNAD2	FUNAD1	FUNAD0	LSB 0	Address: 0050 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
	FUNAD6:0		7-bit programmable Function Address (bits 6-0)							
	Bit 7		Reserved (Read/Write "0")							

Figure 2-65. USB Function Address Register

The **USB Power Management Register**, shown in Figure 2-66, is used for power management in the USB FCU.

#### USB Suspend Detection Flag

When the USB FCU receives a USB suspend signaling, it sets the SUSPEND bit and generates an interrupt. The CPU writes a "0" to clear this bit when the device is resumed by the host (resume interrupt is generated and Resume Detection Flag is set) or remote wake-up by itself (The CPU writes a "1" to Remote Wake-up Bit).

#### USB Resume Detection Flag

When the USB FCU is in suspend mode and receives a USB resume signaling, it sets the RESUME bit, and generates an interrupt. The CPU writes a "0" to clear this bit.

#### USB Remote Wake-up Bit

The CPU writes a "1" to the WAKEUP bit for remote wake-up. While this bit is set, and the USB FCU is in suspend mode, it will generate a resume signaling to the host. The CPU must keep this bit set for a minimum of 10ms and a maximum of 15ms before writing a "0" to this bit.

MSB 7	Reserved	Reserved	Reserved	Reserved	Reserved	WAKEUP	RESUME	SUSPEND	LSB 0	Address: 0051 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
		SUSPEND	USB Suspend Detection Flag (bit 0) (Write "0" only or Read)							
			0: No USB suspend signal detected							
			1: USB suspend signal detected							
		RESUME	USB Resume Detection Flag (bit 1) (Write "0" only or Read)							
			0: No USB resume signal detected							
			1: USB resume signal detected							
		WAKEUP	USB Remote Wake-up Bit (bit 2)							
			0: End remote resume signaling							
			1: Remote resume signaling (If SUSPEND = "1")							
		Bit7:3	Reserved (Read/Write "0")							

Figure 2-66. USB Power Management Register

The USB FCU is able to generate a USB function interrupt as discussed in section 2.9.2.1. The **USB Interrupt Status Registers**, shown in Figure 2-67 and Figure 2-68, are used to indicate the condition that caused a USB function interrupt to the CPU. A "1" indicates the corresponding condition caused a USB function interrupt. The USB Interrupt Status Registers can be cleared by writing back to the register the same value that was read. To ensure proper operation, the CPU should read both USB interrupt status registers, then write back the same values it read to these two registers for clearing the status bits. The CPU must write the USB Interrupt Status Register 1 first, then the USB Interrupt Status Register 2. The registers cannot be cleared by writing a "0" to the bits that are a "1".

The **USB Interrupt Enable Registers**, shown in Figure 2-69 and Figure 2-70, are used to enable the corresponding interrupt status conditions, which can generate a USB function interrupt. If the bit to a corresponding interrupt condition is "0", that condition will not generate a USB function interrupt. If the bit is a "1", that condition can generate a USB function interrupt. Upon reset, all USB interrupt status conditions are enabled except bit 7 of USB Interrupt Enable Register 2 - i.e., suspend and resume interrupt is disabled.

MSB 7	INTST7	INTST6	INTST5	INTST4	INTST3	INTST2	Reserved	INTST0	LSB 0	Address: 0052 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
		INTST0	USB Endpoint 0 Interrupt Status Flag (bit 0)							
		Bit 1	Reserved (Read/Write "0")							
		INTST2	USB Endpoint 1 IN Interrupt Status Flag (bit 2)							
		INTST3	USB Endpoint 1 OUT Interrupt Status Flag (bit 3)							
		INTST4	USB Endpoint 2 IN Interrupt Status Flag (bit 4)							
		INTST5	USB Endpoint 2 OUT Interrupt Status Flag (bit 5)							
		INTST6	USB Endpoint 3 IN Interrupt Status Flag (bit 6)							
		INTST7	USB Endpoint 3 OUT Interrupt Status Flag (bit 7)							
			0: No interrupt request issued							
			1: Interrupt request issued							

Figure 2-67. USB Interrupt Status Register 1

**INTST0** is set to a "1" by the USB FCU if (in Endpoint 0 IN CSR):

- Successfully receives a packet of data
- Successfully sends a packet of data
- IN0CSR3 (DATA\_END) bit is cleared
- IN0CSR4 (FORCE\_STALL) bit is set
- IN0CSR5 (SETUP\_END) bit is set

**INTST2**, **INTST4**, **INTST6** or **INTST8** is set to a "1" by the USB FCU if (in Endpoint x IN CSR):

- Successfully sends a packet of data
- INXCSR1 (UNDER\_RUN) bit is set



INTST3, INTST5, INTST7 or INTST9 is set to a “1” by the USB FCU if (in Endpoint xOUT CSR):

- Successfully receives a packet of data
- OUTXCSR1 (OVER\_RUN) bit is set
- OUTXCSR4 (FORCE\_STALL) bit is set

MSB 7	INTST15	INTST14	INTST13	INTST12	Reserved	Reserved	INTST9	INTST8	LSB 0	Address: 0053 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
		INTST8	USB Endpoint 4 In Interrupt Status Flag (bit 0)								
		INTST9	USB Endpoint 4 Out Interrupt Status Flag (bit 1)								
		Bit 3:2	Reserved (Read/Write “0”)								
		INTST12	USB Overrun/Underrun Interrupt Status Flag (bit 4)								
		INTST13	USB Reset Interrupt Status Flag (bit 5)								
		INTST14	USB Resume Signaling Interrupt Status Flag (bit 6)								
		INTST15	USB Suspend Signaling Interrupt Status Flag (bit 7)								
			0: No interrupt request issued 1: Interrupt request issued								

Figure 2-68. USB Interrupt Status Register 2

INTST12 is set to a “1” by the USB FCU if an overrun or underrun condition occurs in any of the endpoints.

INTST13 is set to a “1” by the USB FCU if a USB reset signaling from the host is received. All USB internal registers will be reset to their default values except this bit.

INTST14 is set to a “1” by the USB FCU if a USB resume signaling is received from the host.

INTST15 is set to a “1” by the USB FCU if a USB suspend signaling is received from the host.

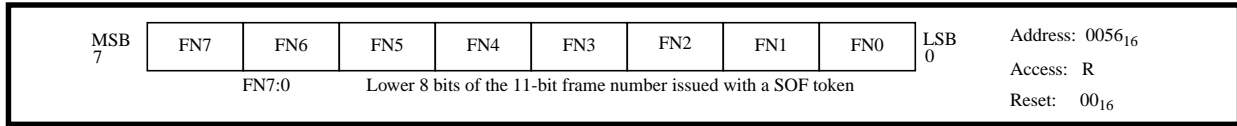
MSB 7	INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	Reserved	INTEN0	LSB 0	Address: 0054 <sub>16</sub> Access: R/W Reset: FF <sub>16</sub>	
		INTEN0	USB Endpoint 0 In Interrupt Enable Bit (bit 0)								
		Bit 1	Reserved (Read/Write “0”)								
		INTEN2	USB Endpoint 1 IN Interrupt Enable Bit (bit 2)								
		INTEN3	USB Endpoint 1 OUT Interrupt Enable Bit (bit 3)								
		INTEN4	USB Endpoint 2 IN Interrupt Enable Bit (bit 4)								
		INTEN5	USB Endpoint 2 OUT Interrupt Enable Bit (bit 5)								
		INTEN6	USB Endpoint 3 IN Interrupt Enable Bit (bit 6)								
		INTEN7	USB Endpoint 3 OUT Interrupt Enable Bit (bit 7)								
			0: Interrupt disabled 1: Interrupt enabled								

Figure 2-69. USB Interrupt Enable Register 1

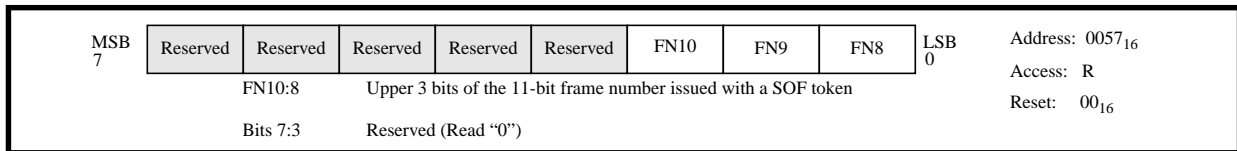
MSB 7	INTEN15	Reserved	INTEN13	INTEN12	Reserved	Reserved	INTEN9	INTEN8	LSB 0	Address: 0055 <sub>16</sub> Access: R/W Reset: 33 <sub>16</sub>	
		INTEN8	USB Endpoint 4 IN Interrupt Enable Bit (bit 0)								
		INTEN9	USB Endpoint 4 OUT Interrupt Enable Bit (bit 1)								
		Bit 3:2	Reserved (Read/Write “0”)								
		INTEN12	USB Overrun/Underrun Interrupt Enable Bit (bit 4)								
		INTEN13	USB Reset Interrupt Enable Bit (bit 5)								
		Bit 6	Reserved (Read/Write “0”)								
		INTEN15	USB Suspend/Resume Signaling Interrupt Enable Bit (bit 7)								
			0: Interrupt disabled 1: Interrupt enabled								

Figure 2-70. USB Interrupt Enable Register 2

The **USB Frame Number Low Register**, shown in Figure 2-71, contains the lower 8 bits of the 11-bit frame number received from the host. The **USB Frame Number High Register**, shown in Figure 2-72 contains the upper 3 bits of the 11-bit frame number received from the host.



**Figure 2-71. USB Frame Number Low Register**



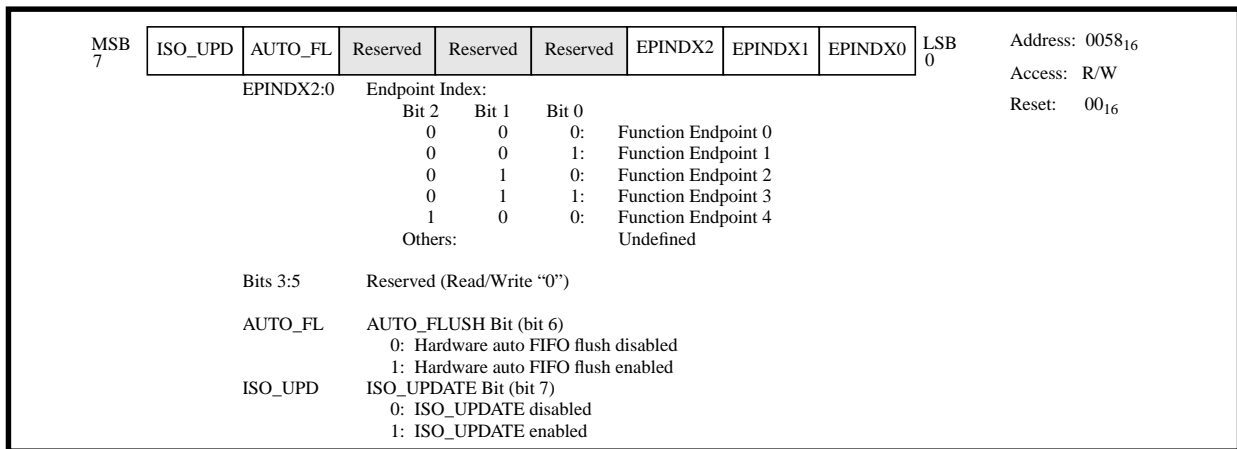
**Figure 2-72. USB Frame Number High Register**

The **USB Endpoint Index Register**, shown in Figure 2-73, identifies the endpoint pair. It serves as an index to endpoint-specific IN CSR, OUT CSR, IN MAXP, OUT MAXP and OUT WRT CNT registers.

This register also contains two global bits, ISO\_UPD and AUTO\_FL for endpoints 1-4 regarding the isochronous data transfer.

If ISO\_UPD = "0", a data packet in an endpoint's IN FIFO is always 'ready to transmit' upon receiving the next IN\_TOKEN from the host (with matched address & endpoint number). If ISO\_UPD = "1" and the ISO bit of the corresponding endpoint's IN CSR is set, then the internal 'ready to transmit' signal to the transmit control logic is delayed until the next SOF. In this way the data loaded in frame n will be transmitted out in frame n+1. The ISO\_UPD bit is a global bit for endpoints 1 to 4, and works with isochronous pipes only.

If AUTO\_FL = "1", ISO\_UPD = "1", and a particular IN endpoint's ISO bit is set, then at the time the USB FCU detects a SOF packet, if the corresponding IN endpoint's IN\_PKT\_RDY = "1", the USB FCU automatically flushes the oldest packet from the IN FIFO. In this case, IN\_PKT\_RDY = "1" indicates that two data packet are in the IN FIFO. Since, for ISO transfer, double buffering is a requirement, MAXP must set to be less than or equal to 1/2 of the FIFO size.



**Figure 2-73. USB Endpoint Index Register**

The **Endpoint 0 IN CSR** (Control & Status Register), shown in Figure 2-74, contains the control and status information of Endpoint 0.



**IN0CSR0** (OUT\_PKT\_RDY): The USB FCU sets this bit to a “1” upon receiving a valid SETUP/OUT token from the host. The CPU clears this bit after unloading the FIFO, by way of writing a “1” to IN0CSR6. The CPU should not clear the OUT\_PKT\_RDY bit before finishes decoding the host request. If IN0CSR2 (SEND\_STALL) needs to be set - the CPU decodes an invalid or unsupported request - the setting IN0CSR6 = “1” & IN0CSR2 = “1” should be done in a same CPU write.

**IN0CSR1** (IN\_PKT\_RDY): The CPU writes a “1” to this bit after finishes writing a packet of data to the endpoint 0 FIFO. The USB FCU clears this bit after the packet is successful transmitted to the host, or the IN0CSR5 (SETUP\_END) bit is set.

**IN0CSR2** (SEND\_STALL): The CPU writes a “1” to this bit if it decodes an invalid or unsupported standard device request from the host. The USB FCU returns a STALL handshake for all subsequent IN/OUT transactions (during control transfer data or status stages) while this bit is set. The CPU writes a “0” to clear this bit.

**IN0CSR3** (DATA\_END): For control transfers, the CPU writes a “1” to 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 USB FCU that the specific amount of data in the setup phase is transferred. The USB FCU will advance to the status phase once this bit is set. When the status phase completes, the USB FCU clears this bit. When this bit is set to a “1”, and the host again requests or sends more data, the USB FCU will return a STALL handshake.

**IN0CSR4** (FORCE\_STALL): The USB FCU sets this bit to a “1” if the host sends out a larger data packet than the MAXP size, or if during a data stage a command pipe is sent more data or is requested to return more data than was indicated in the setup stage (also see description for IN0CSR3). The USB FCU returns a STALL handshake for all subsequent IN/OUT transactions (during data or status stages) while this bit is set. The CPU writes a “0” to clear this bit.

**IN0CSR5** (SETUP\_END): The USB FCU sets this bit to a “1” if a control transfer has ended before the specific length of data is transferred during the data phase. The CPU clears this bit by way of writing a “1” to IN0CSR7. Once the CPU sees the SETUP\_END bit set, it should stop accessing 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 a new SETUP token is in the FIFO.

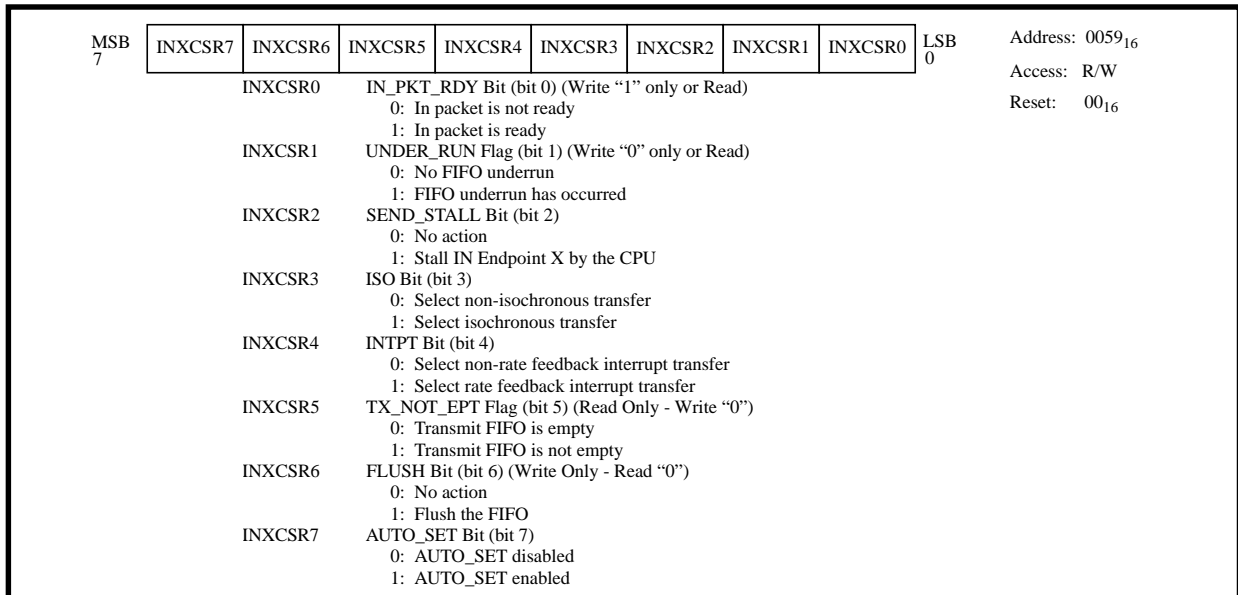
**IN0CSR6** and **IN0CSR7**: These bits are used to clear IN0CSR0 and IN0CSR5 respectively. Writing a “1” to these bits will clear the corresponding register bit.





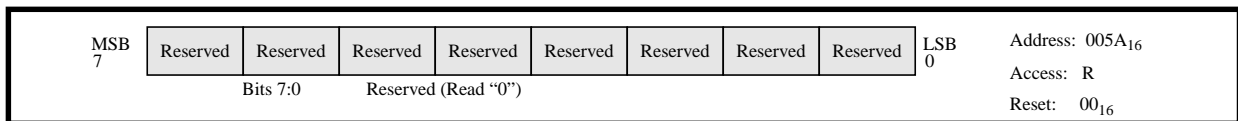


**INXCSR7 (AUTO\_SET):** If the CPU sets this bit to a “1”, the IN\_PKT\_RDY bit is set automatically by the USB FCU after the number of bytes of data equal to the maximum packet size (MAXP) is written into the IN FIFO (see Chapter 2.9.3.1. IN (Transmit) FIFOs for details).



**Figure 2-75. USB Endpoints x IN CSR**

All bits in **USB Endpoint 0 OUT CSR** (Control & Status Register), shown in Figure 2-76, are reserved (all control and status info is in Endpoint 0 IN CSR)



**Figure 2-76. USB Endpoint 0 OUT CSR**

The **USB Endpoint x OUT CSR** (Control & Status Register), shown in Figure 2-77, contains control and status information of the respective OUT endpoint 1-4. The specific endpoint is selected by the USB Endpoint Index Register.

**OUTXCSR0 (OUT\_PKT\_RDY):** The USB FCU sets the this bit to a “1” after it successfully receives a packet of data from the host. This bit is cleared by the CPU or by the USB FCU after a packet of data is unloaded from the FIFO (See Chapter 2.9.3.2. Out (Receive) FIFOs for details).

**OUTXCSR1 (OVER\_RUN):** This bit is used in ISO mode only to indicate to the CPU that a FIFO overrun has occurred. The USB FCU sets this bit to a “1” at the beginning of an OUT token if the OUTXCSR0 (OUT\_PKT\_RDY) bit is not cleared. Setting this bit will cause the INST12 bit of the Interrupt Status Register 2 to set. The CPU writes a “0” to clear this bit.

**OUTXCSR2 (SEND\_STALL):** The CPU writes a “1” to this bit when the endpoint is stalled (receiver halt). The USB FCU returns a STALL handshake while this bit is set. The CPU writes a “0” to clear this bit.

**OUTXCSR3 (ISO):** The CPU sets this bit to a “1” to initialize the respective endpoint as an Isochronous endpoint for OUT transactions.



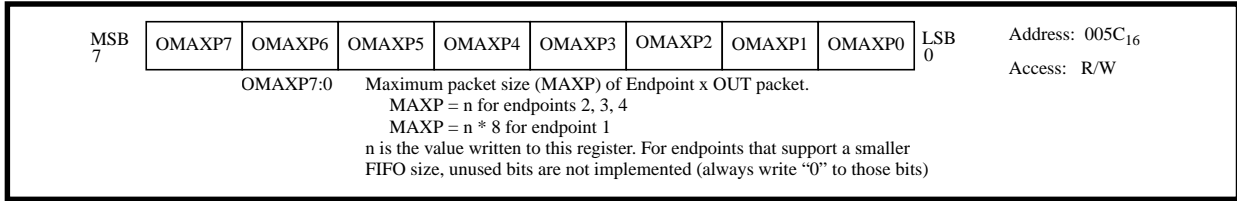


Figure 2-79. USB Endpoint x OUT MAXP

The **USB Endpoint x OUT WRT CNT Low** & the **USB Endpoint x OUT WRT CNT High** registers, shown in Figure 2-80 and Figure 2-81, contain the number of bytes in the Endpoint x OUT FIFO. The USB FCU sets the values in these two Write Count Registers after having successfully received a packet of data 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 WRT CNT Low first then WRT CNT High.

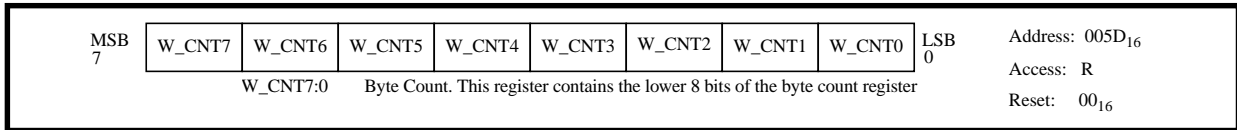


Figure 2-80. USB Endpoint x OUT WRT CNT Low

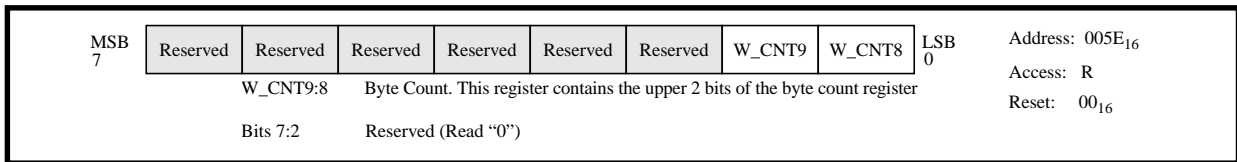


Figure 2-81. USB Endpoint x OUT WRT CNT High

The **USB Endpoint x FIFO Registers**, shown in Figure 2-82 through Figure 2-86, are the USB IN (transmit) and OUT (receive) FIFO data registers. The CPU writes data to these registers for the corresponding Endpoint IN FIFO and reads data from these registers for the corresponding Endpoint OUT FIFO.

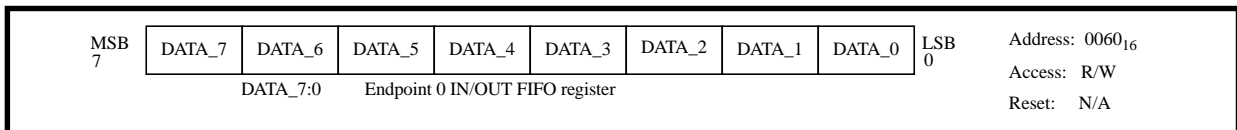


Figure 2-82. USB Endpoint 0 FIFO Register

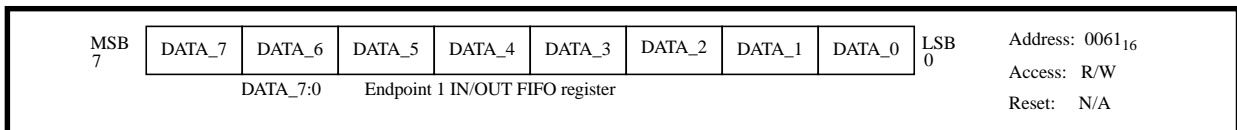


Figure 2-83. USB Endpoint 1 FIFO Register

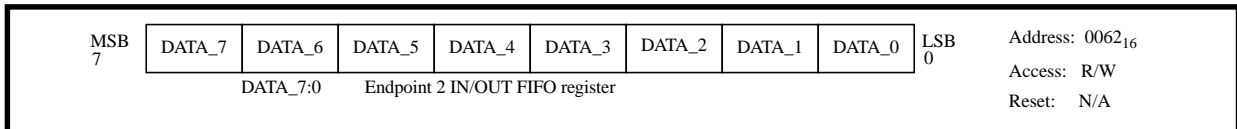


Figure 2-84. USB Endpoint 2 FIFO Register

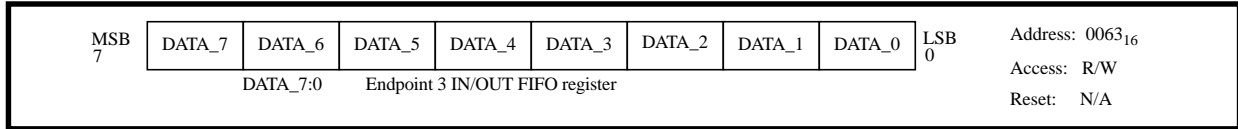


Figure 2-85. USB Endpoint 3 FIFO Register

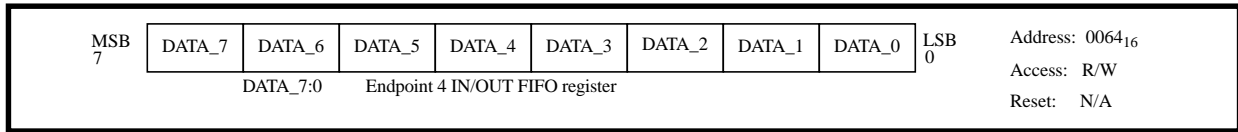


Figure 2-86. USB Endpoint 4 FIFO Register



## 2.10 Master CPU Bus Interface

Address	Description	Acronym and Value at Reset
0048 <sub>16</sub>	Data bus buffer register 0	DBB0=00
0049 <sub>16</sub>	Data bus buffer status register 0	DBBS0=00
004A <sub>16</sub>	Data bus buffer control register 0	DBBC0=00
004C <sub>16</sub>	Data bus buffer register 1	DBB1=00
004D <sub>16</sub>	Data bus buffer status register 1	DBBS1=00
004E <sub>16</sub>	Data bus buffer control register 1	DBBC1=00

Pin	Description
P6 <sub>0</sub> -P6 <sub>7</sub>	are multiplexed with DQ0-DQ7
P5 <sub>2</sub>	is multiplexed with OBF <sub>0</sub>
P5 <sub>3</sub>	is multiplexed with $\overline{\text{IBF}}_0$
P5 <sub>4</sub>	is multiplexed with $\overline{\text{S}}_0$
P5 <sub>5</sub>	is multiplexed with A <sub>0</sub>

Pin	Description
P5 <sub>6</sub>	is multiplexed with $\overline{\text{R}}$ or E
P5 <sub>7</sub>	is multiplexed with $\overline{\text{W}}$ or R/ $\overline{\text{W}}$
P7 <sub>2</sub>	is multiplexed with $\overline{\text{S}}_1$
P7 <sub>3</sub>	is multiplexed with $\overline{\text{IBF}}_1$
P7 <sub>4</sub>	is multiplexed with OBF <sub>1</sub>

This device has a bus interface function with 2 I/O buffers that can be operated in slave mode by control signals from the master CPU (see Figure 2-87. Bus Interface Circuit). The bus interface can be connected directly to either a R/ $\overline{\text{W}}$  type of CPU or a CPU with RD and 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 bit 7 of the data bus buffer control register 1. When selecting the double data bus buffer mode, port P7<sub>2</sub> becomes  $\overline{\text{S}}_1$  input. Prior to enabling the MBI, port 6 must be placed in input mode by writing 00<sub>16</sub> to P6D (0015<sub>16</sub>).

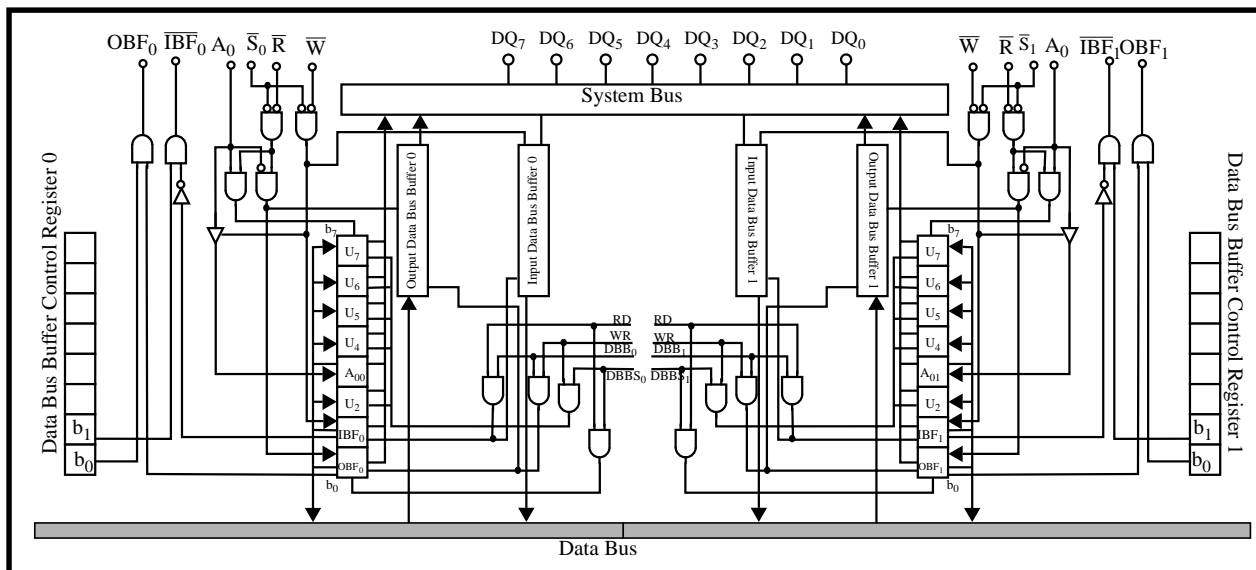


Figure 2-87. Bus Interface Circuit

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<sub>0</sub>-DQ<sub>7</sub> 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<sub>0</sub> become host CPU control signal input pins.

The two input buffer full interrupt requests and two output buffer full requests are multiplexed as shown in Figure 2-88.

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.

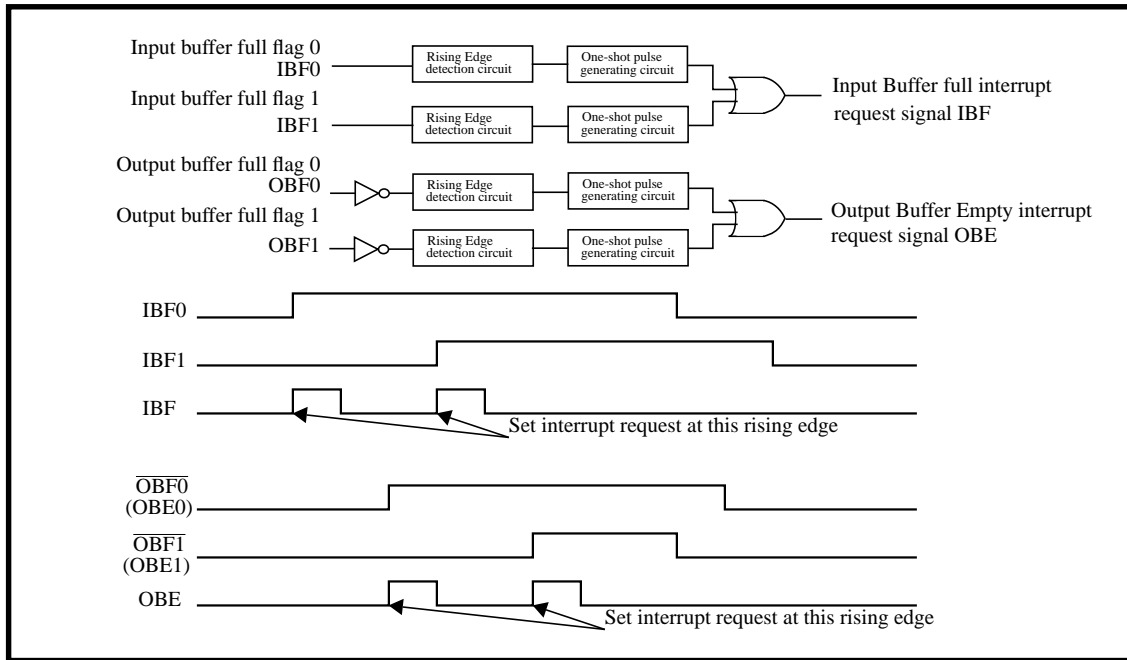


Figure 2-88. Data Bus Buffer Interrupt Request Circuit

MSB 7	DBBS07	DBBS06	DBBS05	DBBS04	DBBS03	DBBS02	DBBS01	DBBS00	LSB 0	Address: 0049 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
								DBBS00		Output Buffer Full (OBF <sub>0</sub> ) Flag (bit 0) 0: Output buffer empty. 1: Output buffer full.
								DBBS01		Input Buffer Full (IBF <sub>0</sub> ) Flag (bit 1) 0: Input buffer empty. 1: Input buffer full.
								DBBS02		User Definable (U2) Flag (bit 2)
								DBBS03		A <sub>0</sub> (A <sub>00</sub> ) Flag (bit 3) Indicates the A <sub>0</sub> status when IBF flag is set
								DBBS04		User Definable (U4) Flag (bit 4)
								DBBS05		User Definable (U5) Flag (bit 5)
								DBBS06		User Definable (U6) Flag (bit 6)
								DBBS07		User Definable (U7) Flag (bit 7)

Figure 2-89. Data Bus Buffer Status Register 0



### 2.10.1 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.

### 2.10.2 Input Data Bus Buffer Registers (DBBIN<sub>0</sub>, DBBIN<sub>1</sub>)

The data on the data bus is latched into DBBIN<sub>0</sub> or DBBIN<sub>1</sub> by a write request from the master CPU. The data in DBBIN<sub>0</sub> or DBBIN<sub>1</sub> can be read from the data bus buffer register in the SFR area.

### 2.10.3 Output Data Bus Buffer Registers (DBBOUT<sub>0</sub>, DBBOUT<sub>1</sub>)

Data is set in DBBOUT<sub>0</sub> or DBBOUT<sub>1</sub> 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.





## 2.11 Direct Memory Access Controller

Address	Description	Acronym and Value at Reset
003F <sub>16</sub>	DMAC index and status register	DMAIS=00
0040 <sub>16</sub>	DMAC channel x mode register 1	DMAxM1=00
0041 <sub>16</sub>	DMAC channel x mode register 2	DMAxM2=00
0042 <sub>16</sub>	DMAC channel x source register Low	DMAxSL=00
0043 <sub>16</sub>	DMAC channel x source register High	DMAxSH=00
0044 <sub>16</sub>	DMAC channel x destination register Low	DMAxDL=00
0045 <sub>16</sub>	DMAC channel x destination register High	DMAxDH=00
0046 <sub>16</sub>	DMAC channel x transfer count register Low	DMAxCL=00
0047 <sub>16</sub>	DMAC channel x transfer count register High	DMAxCH=00

This device contains a two-channel Direct Memory Access Controller (DMAC). Each channel 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), MBI (4), external interrupts (4), UART1 (2), UART2 (2), SIO (1), TimerX (1), TimerY (1), Timer1 (1), and software triggers
- Closely coupled with USB and MBI 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  $\Phi$  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 2-93.

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 "1", channel 1 registers are accessible. The configuration of DMAIS and the mode registers are shown in Figure 2-94, Figure 2-95, Figure 2-96, and Figure 2-97.

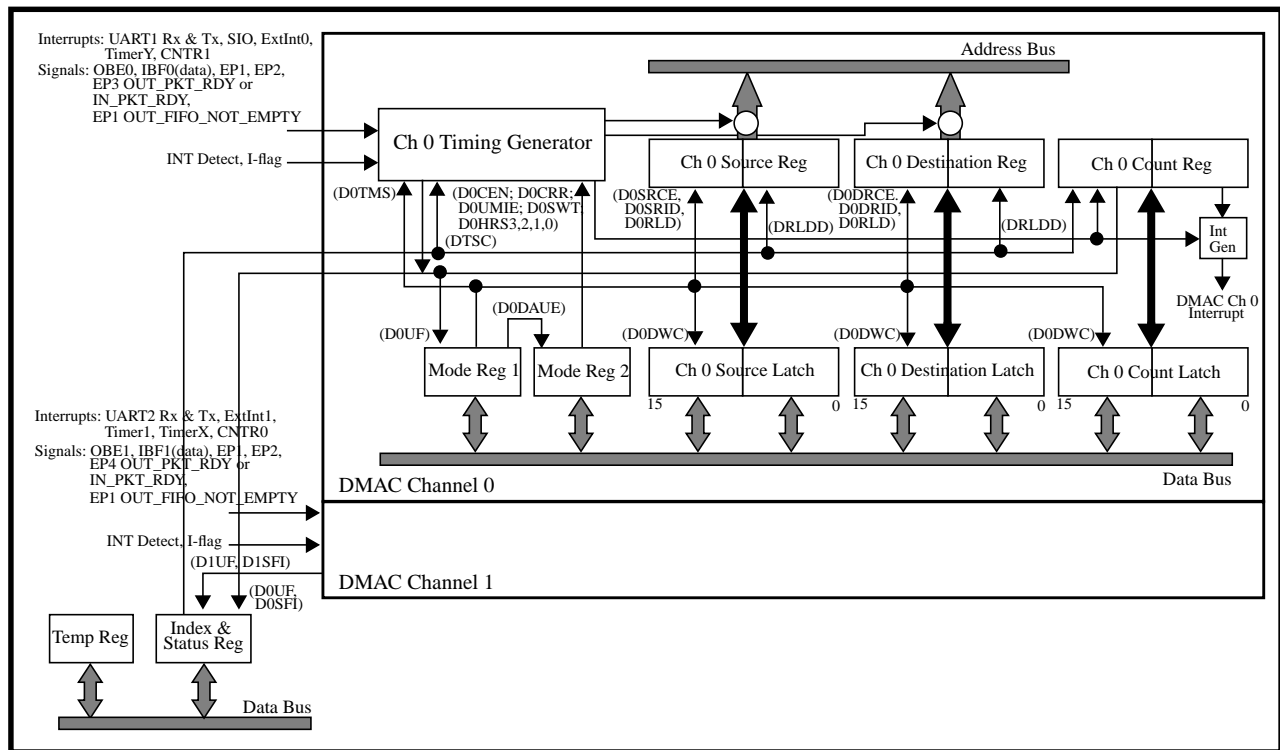


Figure 2-93. DMAC Block Diagram

### 2.11.1 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  $\Phi$  and causes the CPU and possibly the other DMAC channel to stall during this time. At least one cycle of  $\Phi$  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. The DMAC does not operate during WIT, STP, or Hold states.



### 2.11.1.1 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 “1” 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 to a “1”, 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 “1” 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 “1”, 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 to a “1”.

##### 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.

### 2.11.1.2 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 OUT\_PKT\_RDY signals, one of three USB endpoint IN\_PKT\_RDY signals, the USB endpoint 1 OUT\_FIFO\_NOT\_EMPTY signal, the OBE0 and IBF0 (data) signals from the MBI, 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 OUT\_PKT\_RDY signals, one of three USB endpoint IN\_PKT\_RDY signals, the USB endpoint 1 OUT\_FIFO\_NOT\_EMPTY signal, the OBE1 and IBF1 (data) signals from the MBI, 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  $\Phi$ . 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.

### 2.11.1.3 Transfer Features for USB and MBI

In order to make the transfer of data between the USB endpoint FIFOs and the input and output buffers of the MBI 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 MBI Enable Bit (DxUMIE) is set to a “1”. These features are only intended to be used with single-byte transfer mode.

#### USB OUT FIFO to MBI Output Buffer Transfers

The special features provided by both DMAC channels for transfer of data from a USB OUT FIFO to one of the MBI output buffers help facilitate either packet-by-packet transfers or byte-by-byte transfers.

#### Packet-by-Packet Transfers

When a USB endpoint OUT\_PKT\_RDY signal is selected as the hardware transfer request source for a DMAC channel and the DxUMIE bit of the same channel is set to a “1”, a transfer request is generated for that DMAC channel when the OUT\_PKT\_RDY signal for the chosen USB endpoint is high and output buffer x (where x is “0” for DMAC channel 0 and “1” for DMAC channel 1) of the MBI is empty. The OUT\_PKT\_RDY signal remains high until all bytes of the packet have been read from the OUT FIFO corresponding to that endpoint. Thus, the first transfer request is generated when the OUT\_PKT\_RDY signal goes high 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 OUT\_PKT\_RDY signal automatically goes low (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 to a “1”) 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 OUT FIFO to the master CPU (via the MBI) 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.



In order for this mode to function correctly, the time from the DMAC writing data to MBI output buffer x (which causes pin OBFx to go high) until the end of a master read of MBI output buffer x) must be greater than  $104.167 \cdot (3 - (12E6/\Phi))$  ns. For example, if  $\Phi = 12$  MHz, the delay must be greater than 208.334 ns, and if  $\Phi = 6$  MHz, the delay must be greater than 104.167 ns. When  $\Phi$  is less than or equal to 4 MHz, no delay is required.

### Byte-by-Byte Transfers

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 to a “1”, a transfer request is generated for the DMAC channel if the endpoint 1 OUT FIFO is not empty and output buffer x of the MBI is empty. Thus, a transfer request is generated as soon as new data is received in the endpoint 1 OUT FIFO and the master CPU has read the data previously placed in MBI output buffer x. As is the case when the packet-by-packet method is used, the OUT\_PKT\_RDY signal goes high once a complete packet has been received. It remains high until all bytes of the packet have been read from the OUT FIFO. When the final byte has been read from the OUT FIFO, the OUT\_PKT\_RDY signal goes low (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 to a “1”) 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 OUT FIFO before the end\_of\_packet signal is received by the USB block, the OUT\_PKT\_RDY signal will still go high and then low a short period of time later (if this option is enabled in the USB block).

This feature allows a channel of the DMAC in single-byte transfer mode to automatically transfer data received for endpoint 1 from the endpoint 1 OUT FIFO to the master CPU (via the MBI) prior to reception of the complete packet.

### MBI Input Buffer to USB IN FIFO Transfers

When a USB endpoint IN\_PKT\_RDY signal is selected as the hardware transfer request source for a DMAC channel and the DxUMIE bit of the same channel is set to a “1”, a transfer request is generated when the IN FIFO associated with the endpoint is not full (with respect to the programmed packet size) and input buffer x of the MBI contains data. The transfer request is not generated if input buffer x contains a command. The IN FIFO associated with an endpoint is not full when IN\_PKT\_RDY is low. The IN\_PKT\_RDY signal remains low until a full packet has been written to the IN FIFO. Thus, the first transfer request is generated when the IN\_PKT\_RDY goes low 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 IN FIFO, the IN\_PKT\_RDY signal is automatically set to a “1” (assuming 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 the registers once the last byte of the data is transferred from input buffer x to the endpoint’s IN 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 MBI) to the endpoint’s IN FIFO without any intervention by the on-chip CPU. Additionally, since the IN\_PKT\_RDY signal associated with the endpoint is automatically set (assuming 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.

#### 2.11.1.4 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<sub>16</sub>, 21 transfers will occur before control of the address bus and data bus is given back to the CPU.

#### 2.11.1.5 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  $\Phi$  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  $\Phi$ . 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 "0" and an interrupt with its interrupt control bit set to a "1" 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 to a "1" at this time. When the I flag value (which was automatically set to a "1" 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 to a "1", 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 2-98, Figure 2-99, and Figure 2-100. 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	DCI	Reserved	DRLDD	DTSC	D1SFI	D1UF	D0SFI	D0UF	LSB 0	Address: 003F <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
			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		DCI	Reserved (Read/Write "0") 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 2-94. DMAIS Configuration

MSB 7	DxTMS	DxRLD	DxDAUE	DxDWC	DxDRCE	DxDRID	DxSRCE	DxSRID	LSB 0	Address: 0040 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
			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 "0") 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 2-95. DMAxM1 Configuration

MSB 7	DOCEN	D0CRR	D0UMIE	D0SWT	D0HRS3	D0HRS2	D0HRS1	D0HRS0	LSB 0	Address: 0041 <sub>16</sub>
D0HRS3,2,1,0 DMAC Channel 0 Hardware Transfer Request Source Bits (bits 3, 2, 1, 0)										Access: R/W
0000: Disabled										Reset: 00 <sub>16</sub>
0001: UART1 receive interrupt										
0010: UART1 transmit interrupt										
0011: TimerY interrupt										
0100: External Interrupt 0										
0101: USB EndPoint 1 IN_PKT_RDY signal (falling edge active)										
0110: USB EndPoint 2 IN_PKT_RDY signal (falling edge active)										
0111: USB EndPoint 3 IN_PKT_RDY signal (falling edge active)										
1000: USB EndPoint 1 OUT_PKT_RDY signal (rising edge active)										
1001: USB EndPoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active)										
1010: USB EndPoint 2 OUT_PKT_RDY signal (rising edge active)										
1011: USB EndPoint 3 OUT_PKT_RDY signal (rising edge active)										
1100: MBI OBE <sub>0</sub> signal (rising edge active)										
1101: MBI IBF <sub>0</sub> (data) signal (rising edge active)										
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 "0")										
1: Writing "1" requests a channel 0 transfer										
D0UMIE	DMAC Channel 0 USB and MBI 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 "0")										
1: Setting to "1" causes reset of the channel 0 capture register										
DOCEN	DMAC Channel 0 Enable Bit (bit 7)									
0: Channel 0 disabled										
1: Channel 0 enabled										

Figure 2-96. DMA0M2 Configuration

MSB 7	D1CEN	D1CRR	D1UMIE	D1SWT	D1HRS3	D1HRS2	D1HRS1	D1HRS0	LSB 0	Address: 0041 <sub>16</sub>
D1HRS3,2,1,0 DMAC Channel 1 Hardware Transfer Request Source Bits (bits 3, 2, 1, 0)										Access: R/W
0000: Disabled										Reset: 00 <sub>16</sub>
0001: UART2 receive interrupt										
0010: UART2 transmit interrupt										
0011: TimerX interrupt										
0100: External Interrupt 1										
0101: USB EndPoint 1 IN_PKT_RDY signal (falling edge active)										
0110: USB EndPoint 2 IN_PKT_RDY signal (falling edge active)										
0111: USB EndPoint 4 IN_PKT_RDY signal (falling edge active)										
1000: USB EndPoint 1 OUT_PKT_RDY signal (rising edge active)										
1001: USB EndPoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active)										
1010: USB EndPoint 2 OUT_PKT_RDY signal (rising edge active)										
1011: USB EndPoint 4 OUT_PKT_RDY signal (rising edge active)										
1100: MBI OBE <sub>1</sub> signal (rising edge active)										
1101: MBI IBF <sub>1</sub> (data) signal (rising edge active)										
1110: Timer1 interrupt										
1111: CNTR0 interrupt										
D1SWT	DMAC Channel 1 Software Transfer Trigger (bit 4)									
0: No action (Bit is always read as "0")										
1: Writing "1" requests a channel 0 transfer										
D1UMIE	DMAC Channel 1 USB and MBI 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 "0")										
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 2-97. DMA1M2 Configuration



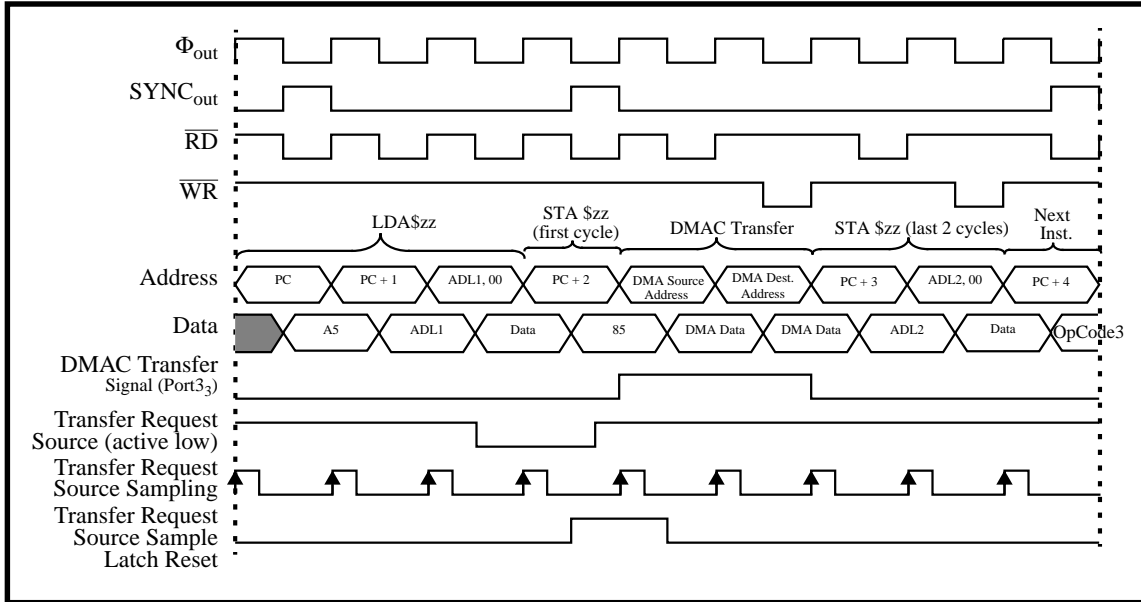


Figure 2-98. DMA Transfer - Hardware Source Initiated

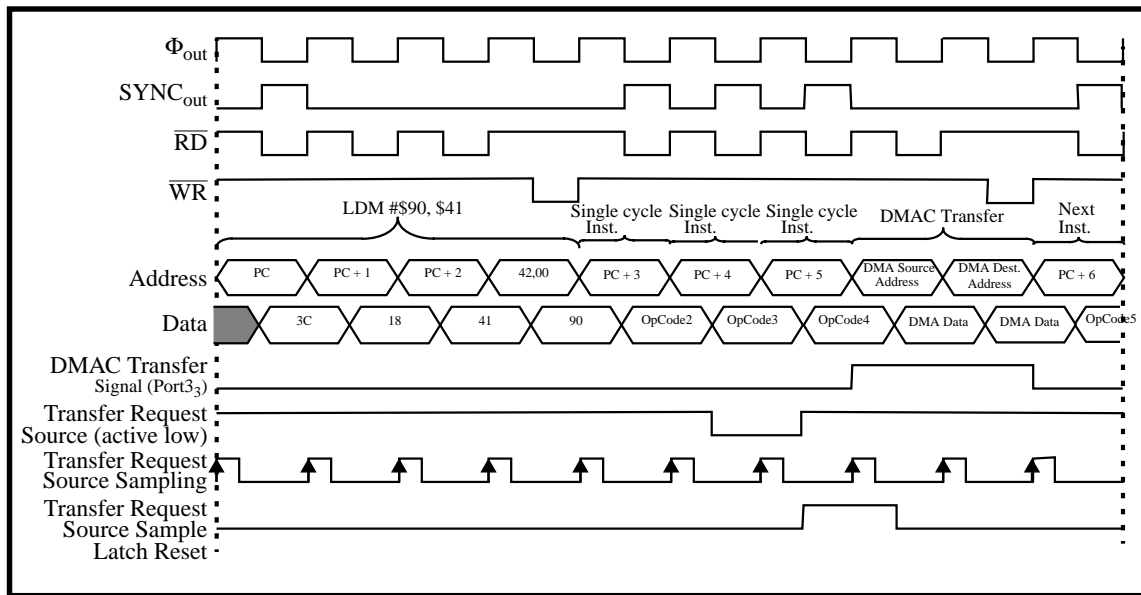


Figure 2-99. DMA Transfer - Software Trigger Initiated

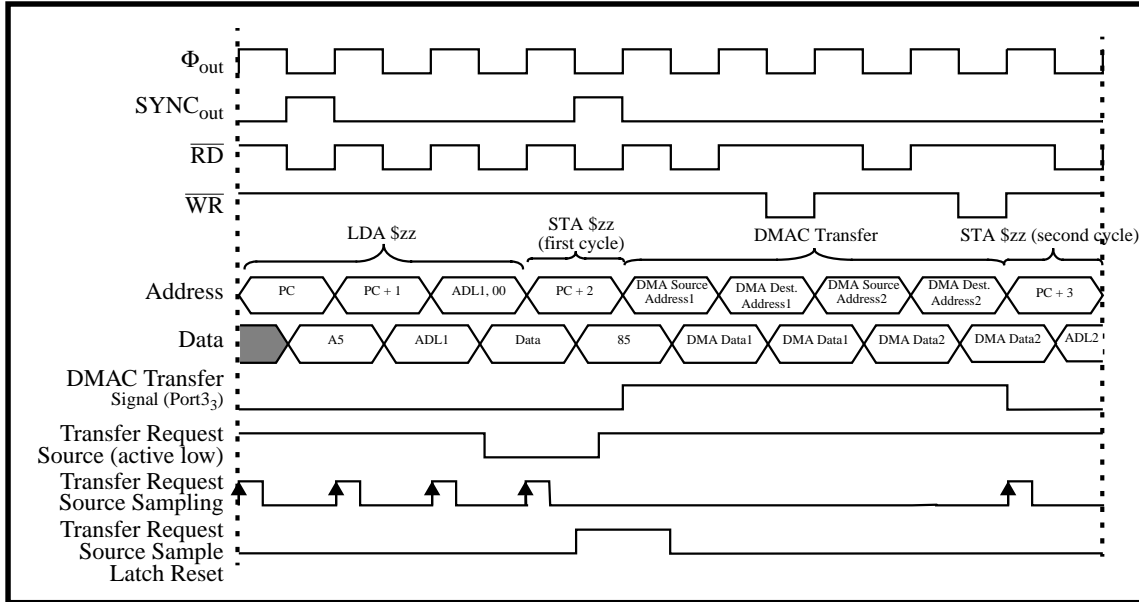


Figure 2-100. DMAC Transfer - Burst Transfer Mode



## 2.12 Special Count Source Generator

Address	Description	Acronym and Value at Reset
002D <sub>16</sub>	Special Count Source Generator1	SCSG1=FF
002E <sub>16</sub>	Special Count Source Generator2	SCSG2=FF
002F <sub>16</sub>	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 2-101.) 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.

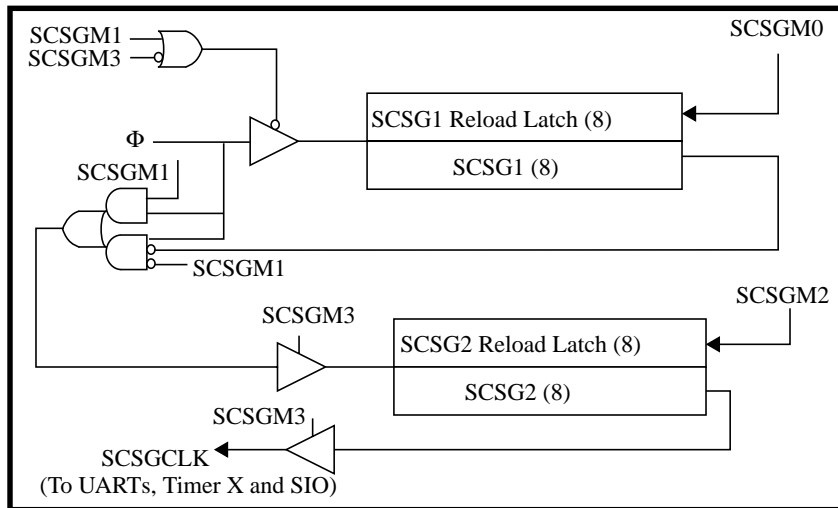


Figure 2-101. SCSG Block Diagram

### 2.12.1 SCSG Operation

The SCSG1 and SCSG2 are both down count timers. When the count of a timer reaches 00<sub>16</sub>, 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 2-102.).

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 ( $\Phi$ ) 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:

$$SCSGCLK = \Phi \cdot \frac{SCSG1}{SCSG1 + 1} \cdot \frac{1}{SCSG2 + 1}$$

where SCSG1 is the value written to SCSG1 and SCSG2 is the value written to SCSG2.

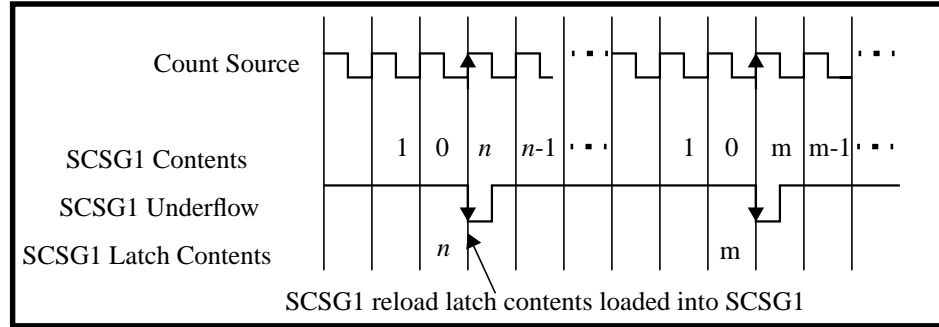


Figure 2-102. Timer Count Operation for SCSG1

## 2.12.2 SCSG Description

### 2.12.2.1 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 to a “1”, SCSG1 stops counting. This allows  $\Phi$  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 2-103.).

### 2.12.2.2 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	Reserved	Reserved	Reserved	Reserved	SCSGM3	SCSGM2	SCSGM1	SCSGM0	LSB 0	Address: 002F <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
					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.				
	Bits 4-7				Reserved (Read/Write “0”)					

Figure 2-103. 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.

## 2.13 Timers

Address	Description	Acronym and Value at Reset	Address	Description	Acronym and Value at Reset
0020 <sub>16</sub>	Timer XL	TXL=FF	0025 <sub>16</sub>	Timer 2	T2=01
0021 <sub>16</sub>	Timer XH	TXH=FF	0026 <sub>16</sub>	Timer 3	T3=FF
0022 <sub>16</sub>	Timer YL	TYL=FF	0027 <sub>16</sub>	Timer X mode register	TXM=00
0023 <sub>16</sub>	Timer YH	TYH=FF	0028 <sub>16</sub>	Timer Y mode register	TYM=00
0024 <sub>16</sub>	Timer 1	T1=FF	0029 <sub>16</sub>	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<sub>16</sub> (0000<sub>16</sub> 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 to a “1”.

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 or  $\overline{\text{RESET}}$  is asserted, 01<sub>16</sub> is loaded into Timer 2 and the Timer 2 reload latch, and FF<sub>16</sub> is loaded into Timer 1 and the Timer 1 reload latch.

Figure 2-107. is a block diagram of the five timers.

### 2.13.1 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 2-104.

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

#### 2.13.1.1 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	
		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 2,1)							
			Bit 2	Bit 1						
			0	0:	$\Phi$ divided by 8					
			0	1:	$\Phi$ divided by 16					
			1	0:	$\Phi$ divided by 32					
			1	1:	$\Phi$ divided by 64					
		TXM3	Timer X Internal Clock Select (bit 3)							
			0: $\Phi/n$							
			1: SCSGCLK (from chip special count source generator)							
		TXM5,4	Timer X Mode Bits (bits 5,4)							
			Bit 5	Bit 4						
			0	0:	Timer Mode					
			0	1:	Pulse output mode					
			1	0:	Event counter mode					
			1	1:	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 2-104. 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.

#### 2.13.1.2 Count Stop Control

If the Timer X Count Stop Bit (TXM7) (bit 7 of the TXM) is set to a “1”, Timer X stops counting in all four modes.

#### 2.13.1.3 Timer Mode

**Count Source:**  $\Phi/n$  (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 to a “1”, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

#### 2.13.1.4 Pulse Output Mode

**Count Source:**  $\Phi/n$  (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 to a “1”. 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.

### 2.13.1.5 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 to a “1”, 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).

### 2.13.1.6 Pulse Width Measurement Mode

**Count Source:**  $\Phi/n$  (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 to a “1”, 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.

## 2.13.2 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 2-105.). 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 <sub>16</sub>	
										Access: R/W	
										Reset: 00 <sub>16</sub>	
		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 3,2)								
			Bit 2	Bit 1							
			0	0:	Φ divided by 8						
			0	1:	Φ divided by 16						
			1	0:	Φ divided by 32						
			1	1:	Φ divided by 64						
		TYM5,4	Timer Y Mode Bits (bits 5,4)								
			Bit 2	Bit 1							
			0	0:	Timer mode						
			0	1:	Pulse period measurement mode						
			1	0:	Event counter mode						
			1	1:	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 2-105. TYM Register

### 2.13.2.1 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 “1”, 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.

### 2.13.2.2 Count Stop Control

If the Timer Y Count Stop Bit (TYM7) (bit 7) is set to a “1”, Timer Y stops counting in all four modes.

### 2.13.2.3 Timer Mode

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

In this mode, each time the timer underflows, the corresponding timer interrupt request bit is set to a “1”, 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 to a “1”. 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.

#### 2.13.2.4 Pulse Period Measurement Mode

**Count Source:**  $\Phi/n$  (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) = “0”**

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) = “1”**

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 to a “1”, the contents of the timer reload latch are loaded into the timer, and the countdown sequence begins again.

#### 2.13.2.5 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 to a “1”, 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

#### 2.13.2.6 HL Pulse-width Measurement Mode

**Count Source:**  $\Phi/n$  (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 to a “1”, the contents of the timer reload latch are loaded into the timer, and the countdown sequence begins again.

### 2.13.3 Timer 1

MSB 7	T123M7	T123M6	T123M5	T123M4	T123M3	T123M2	T123M1	T123M0	LSB 0	Address: 0029 <sub>16</sub>
										Access: R/W
										Reset: 00 <sub>16</sub>
		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: $\Phi$ divided by 8							
			1: XCin divided by 2							
		T123M3	Timer 2 Count Source Select Bit (bit 3)							
			0: Timer 1 underflow signal							
			1: $\Phi$							
		T123M4	Timer 3 Count Source Select Bit (bit 4)							
			0: Timer 1 underflow signal							
			1: $\Phi$ 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 2-106. 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 to a “0”.

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.

#### 2.13.3.1 Timer Mode

**Count Source:**  $\Phi/8$  or XCin/2

In Timer mode, each time the timer underflows, the corresponding timer interrupt request bit is set to a “1”, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

#### 2.13.3.2 Pulse Output Mode

**Count Source:**  $\Phi/8$  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 to a “1”. 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.

## 2.13.4 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 to a “0”.

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.

### 2.13.4.1 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  $\Phi$ .

In Timer mode, each time the timer underflows, the corresponding timer interrupt request bit is set to a “1”, the contents of the timer latch are loaded into the timer, and the count down sequence begins again.

### 2.13.4.2 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  $\Phi$ .

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 to a “1”. 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.

## 2.13.5 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.

### 2.13.5.1 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  $\Phi/8$

In Timer mode, each time the timer underflows, the corresponding timer interrupt request bit is set to a “1”, 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.

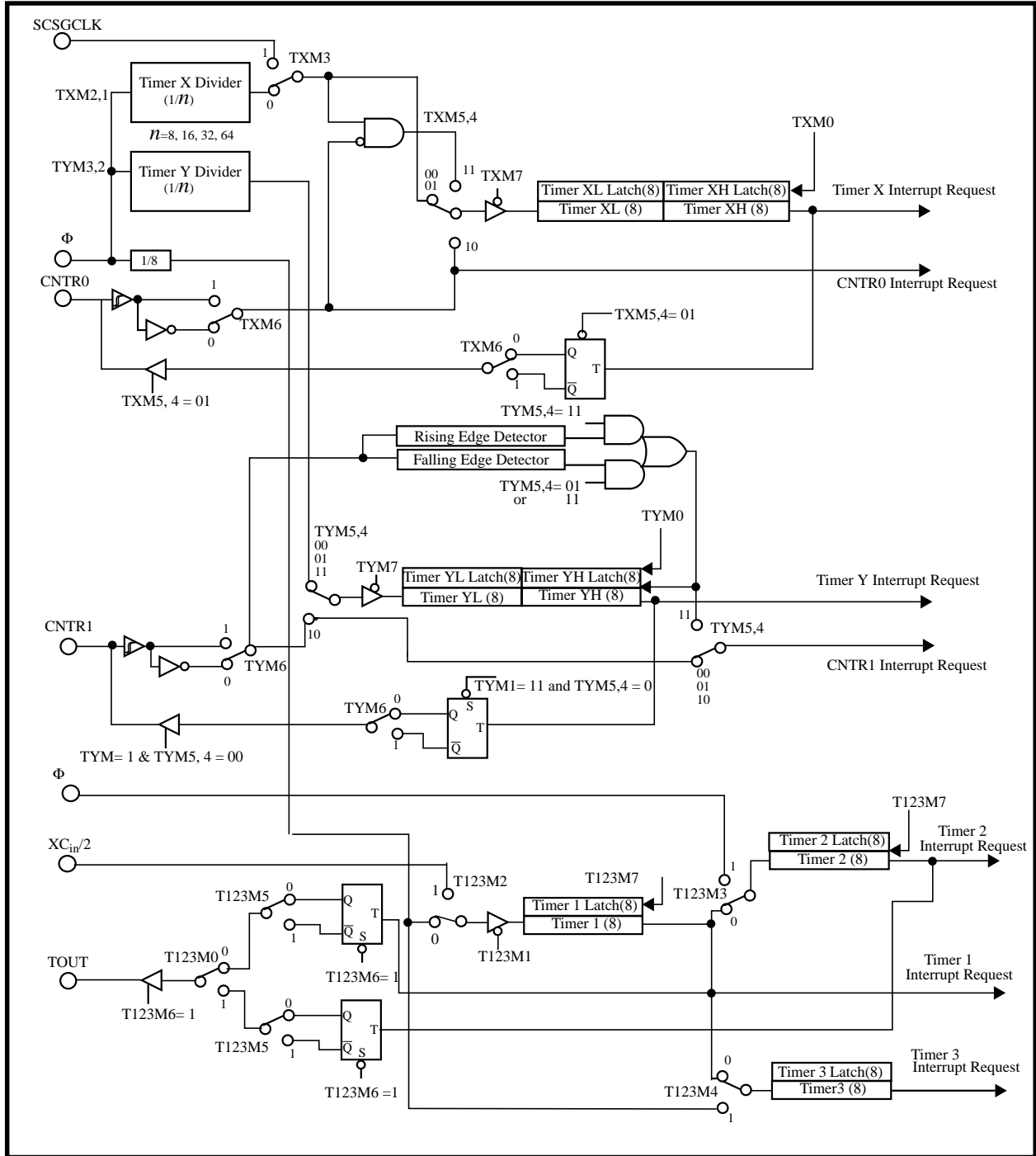


Figure 2-107. Block Diagram of Timers X, Y, 1, 2, and 3

## 2.14 UART

Address	UART1 Description	Acronym and Value at Reset
0030 <sub>16</sub>	UART1 mode register	U1MOD=00
0031 <sub>16</sub>	UART1 baud rate generator	U1BRG=XX
0032 <sub>16</sub>	UART1 status register	U1STS=03
0033 <sub>16</sub>	UART1 control register	U1CON=00
0034 <sub>16</sub>	UART1 transmit/receiver buffer 1	U1TRB1=XX
0035 <sub>16</sub>	UART1 transmit/receiver buffer 2	U1TRB2=XX
0036 <sub>16</sub>	UART1 RTS control register	U1RTSC=00

Address	UART2 Description	Acronym and Value at Reset
0038 <sub>16</sub>	UART2 mode register	U2MOD=00
0039 <sub>16</sub>	UART2 baud rate generator	U2BRG=XX
003A <sub>16</sub>	UART2 status register	U2STS=03
003B <sub>16</sub>	UART2 control register	U2CON=00
003C <sub>16</sub>	UART2 transmit/receiver buffer 1	U2TRB1=XX
003D <sub>16</sub>	UART2 transmit/receiver buffer 2	U2TRB2=XX
003E <sub>16</sub>	UART2 RTS control register	U2RTSC=00

Pin	Description
UTXD1	UART1 transmit pin is multiplexed with P8 <sub>4</sub>
URXD1	UART1 receive pin is multiplexed with P8 <sub>5</sub>
$\overline{\text{CTS1}}$	UART1 $\overline{\text{CTS1}}$ pin is multiplexed with P8 <sub>6</sub>
$\overline{\text{RTS1}}$	UART1 $\overline{\text{RTS1}}$ pin is multiplexed with P8 <sub>7</sub>

Pin	Description
UTXD2	UART2 transmit pin is multiplexed with P8 <sub>0</sub>
URXD2	UART2 receive pin is multiplexed with P8 <sub>1</sub>
$\overline{\text{CTS2}}$	UART2 $\overline{\text{CTS2}}$ pin is multiplexed with P8 <sub>2</sub>
$\overline{\text{RTS2}}$	UART2 $\overline{\text{RTS2}}$ pin is multiplexed with P8 <sub>3</sub>

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

- Clock selection:  $\Phi$  or SCSGCLK
- Prescaler selection: x1/x8/x32/x256 divisions (both  $\Phi$  and SCSGCLK)
- Baud rate: 11.4 bits/second - 750 Kbytes/second (at  $\Phi = 12\text{MHz}$ )
- 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)
- Handshaking: Clear-to-Send (CTS) and Request-to-Send (RTS)
- Interrupt generation conditions: Transmit Buffer Empty or Transmit Complete, Receive 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 DMAC, converts it into serial data, and transmits the results to the send data output terminal UTXD<sub>x</sub>. The UART receives serial data from an external source through the receive data input URXD<sub>x</sub>, converts it into parallel data, and makes it available to the core or DMAC. 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 Flag (TBE) or the Transmit Complete Flag (TCM) are set, depending on the state of the Transmit Interrupt Source Selection Bit (TIS, bit 4 of UxCON). The UART generates the Receive Buffer Full interrupt when receiving and the Receive Buffer Full Flag is set to a "1". The Receive Error Sum interrupt is generated instead of a Receive Buffer Full interrupt if the UART detects an error when receiving. Enabling a transmit or receive operation by setting the TEN or the REN (bits 0 and 1 of UxCON) automatically forces the corresponding 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 2-108 is a block diagram of the UART. It is valid for both UART1 and UART2.

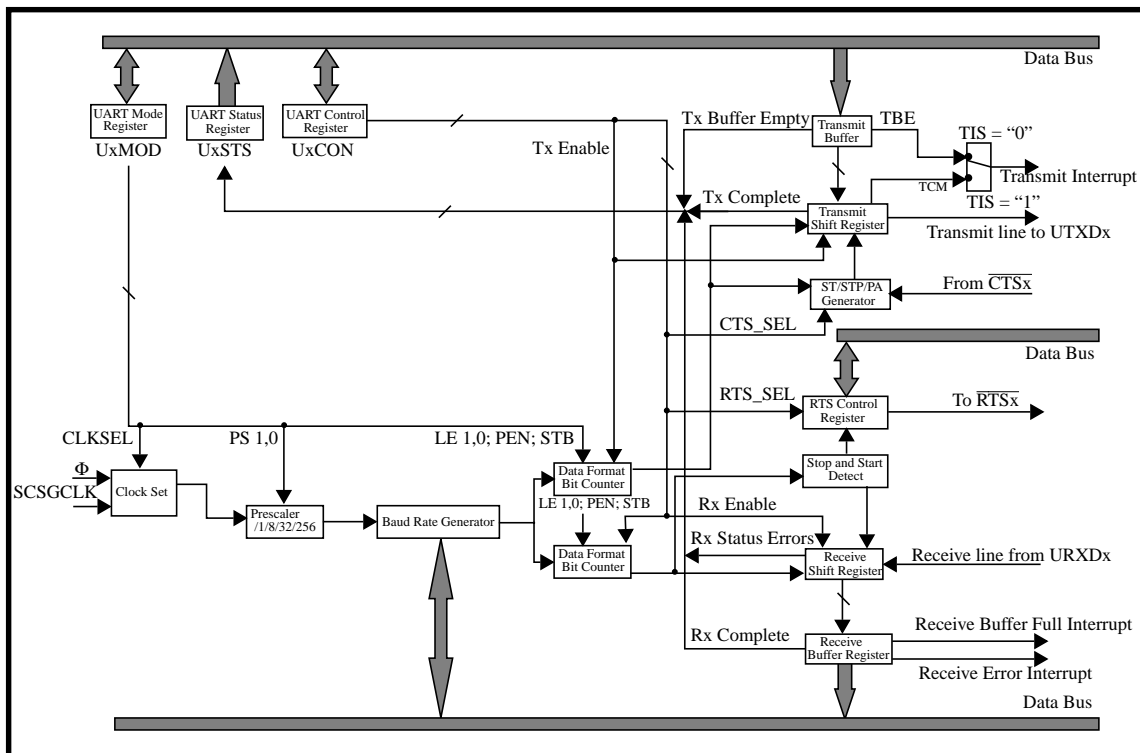


Figure 2-108. UART Block Diagram

### 2.14.1 Baud Rate Selection

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 and the prescale value chosen.

Either an internal clock  $\Phi$  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)). Bits PS0 and PS1 of the UxMOD are used to select a prescaling factor for the clock.

When the internal clock  $\Phi$  is selected,  $f$  is the prescaled value of the internal clock  $\Phi$ .

$$f = \Phi / 1, \Phi / 8, \Phi / 32, \text{ or } \Phi / 256$$

The correspondence between prescale values and baud rate for  $\Phi = 12\text{MHz}$  is given as an example in Table 2-4.

When SCSGCLK is selected,  $f$  is the prescaled value of SCSGCLK.

$$f = \text{SCSGCLK}/1, \text{SCSGCLK}/8, \text{SCSGCLK}/32 \text{ or } \text{SCSGCLK}/256$$

Example settings for SCSGCLK, which is controlled by Special Count Source Generator registers 1 and 2 (SCSG1,2), the Prescaler and UxBRG for several common baud rates when  $\Phi = 12\text{MHz}$  are given in Table 2-5.

**Table 2-4. Prescale Value and Baud Rate Table  $\Phi = 12\text{MHz}$**

$\Phi/1$		$\Phi/8$		$\Phi/32$		$\Phi/256$	
n	Baud Rate	n	Baud Rate	n	Baud Rate	n	Baud Rate
0	750,000.0						
1	375,000.0						
2	250,000.0						
3	187,500.0						
4	150,000.0						
5	125,000.0						
6	107,142.9						
7	93,750.0	0	93,750.0				
8	83,333.3	1	46,875.0				
9	75,000.0	2	31,250.0				
10	.	3	23,437.5	0	23,437.5		
11	.	4	18,750.0	1	11,718.8		
12	.	5	15,625.0	2	7,812.5		
13	.	6	13,392.9	3	5,859.4		
.	.	7	.	4	4,687.5		
.	.	8	.	5	3,906.3		
.	.	.	.	6	3,348.2		
.	.	.	.	7	2,929.7	0	2,929.7
.	.	.	.	8	2,604.2	1	1,464.8
.	.	.	.	.	.	2	976.6
.	.	.	.	.	.	3	732.4
.	.	.	.	.	.	4	585.9
.	.	.	.	.	.	5	488.3
.	.	.	.	.	.	6	418.5
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
255	2,929.7	255	366.2	255	91.6	255	11.44



Table 2-5. SCSGCLK, Prescaler, and UxBRG settings for common baud rates when  $\Phi = 12\text{MHz}$

Baud Rate (Hz)	SCSG (Hex)	SCSG2 (Hex)	SCSGCLK Rate (Hz)	Prescaler	UxBRG (Hex)	Actual Baud Rate (Hz)
50	bypassed	95	80000.00	1/1	63	50.00
75	bypassed	63	120000.00	1/1	63	75.00
110	4E	43	174236.78	1/1	62	110.00
134.5	AC	37	213047.07	1/1	62	134.50
150	bypassed	31	240000.00	1/1	63	150.00
300	bypassed	18	480000.00	1/1	63	300.00
600	18	0B	960000.00	1/1	63	600.00
1200	bypassed	18	480000.00	1/1	18	1200.00
1800	18	13	576000.00	1/1	13	1800.00
2000	bypassed	18	480000.00	1/1	0E	2000.00
2400	18	13	576000.00	1/1	0E	2400.00
3600	18	13	576000.00	1/1	09	3600.00
4800	18	0E	768000.00	1/1	09	4800.00
7200	18	13	576000.00	1/1	04	7200.00
9600	18	0E	768000.00	1/1	04	9600.00
14400	18	09	1152000.00	1/1	04	14400.00
19200	23	00	11666666.66	1/1	25	19188.60
28800	18	00	11520000.00	1/1	18	28800.00
31250	bypassed	bypassed	12000000.00	1/1	17	31250.00
38400	23	00	11666666.66	1/1	12	38377.19
57600	bypassed	bypassed	12000000.00	1/1	0C	57692.31
115200	0B	00	11000000.00	1/1	05	114583.33

2.14.2 UART Mode Register

UxMOD defines data formats and selects the clock to be used (see Figure 2-109).

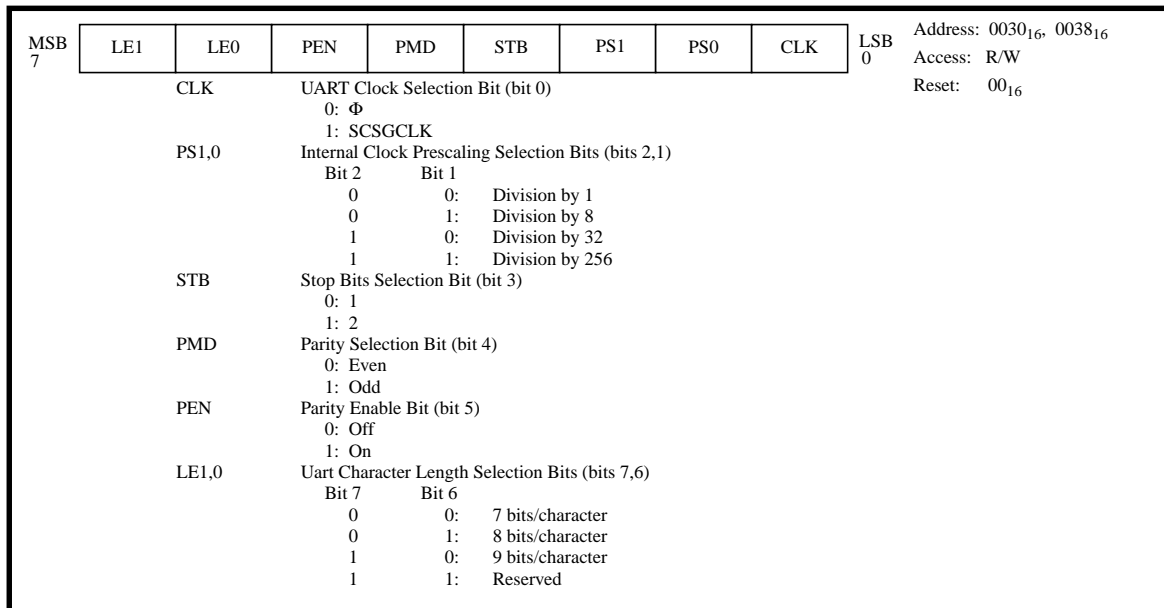


Figure 2-109. UxMOD Register

### 2.14.3 UART Control Register

The UxCON specifies the initialization and enabling of a transmit/receive process (see Figure 2-110). Data can be read from and written to the Control Register.

MSB 7	AME	RTS_SEL	CTS_SEL	TIS	RIN	TIN	REN	TEN	LSB 0	Address: 0033 <sub>16</sub> ,003B <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
	TEN	Transmission Enable Bit (bit 0) 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 (bit 1) 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 (bit 2) 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 (bit 3) 0: No action. 1: Clears the UART receive status flags 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 (bit 4) 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 ( $\overline{CTS}$ ) Enable Bit (bit 5) 0: CTS function is disabled, P8 <sub>6</sub> (or P8 <sub>2</sub> ) is used as GPIO pin. 1: CTS function is enabled, P8 <sub>6</sub> (or P8 <sub>2</sub> ) is used as $\overline{CTS}$ input.								
	RTS_SEL	Request-to-Send ( $\overline{RTS}$ ) Enable Bit (bit 6) 0: RTS function is disabled, P8 <sub>7</sub> (or P8 <sub>3</sub> ) is used as GPIO pin. 1: RTS function is enabled, P8 <sub>7</sub> (or P8 <sub>3</sub> ) is used as $\overline{RTS}$ output.								
	AME	UART Address Mode Enable Bit (bit 7) 0: Address Mode disabled. 1: Address Mode enabled.								

Figure 2-110. UxCON Register

### 2.14.4 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 state 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.

### 2.14.5 UART Status Register

The UART Status Register (UxSTS) reflects both the transmit and receive status (see Figure 2-111). 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 section “2.14.9 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 the receive operation completes while 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 data in the low-order byte of the receive buffer (UxTRB1) 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 the receive operation completes while 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 of UxMOD) is set, the flag is set if either of the two stop bits is a "0". This flag is reset when the status register is read, the hardware reset is asserted, or the receiver is initialized by RIN. If the receive operation completes while 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 of UxMOD) are different. It is enabled only if the Parity Enable Bit (PEN, bit 5 of UxMOD) 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 the receive operation completes while 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 low-order byte of the receive buffer (UxTRB1) is read, the hardware reset is asserted, or the receive process is initialized by RIN. If the receive operation completes while 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 where the transmit buffer does contain data, the TCM flag is set when the last bit in the transmit shift register is transmitted if TBE is a "0" or CTS handshaking is enabled and  $\overline{CTSx}$  is "1". 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 of UxCON). 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 to the low-order byte of the transmit buffer (UxTRB1).

MSB 7	Reserved	SER	OER	FER	PER	RBF	TBE	TCM	LSB 0	Address: 0032 <sub>16</sub> , 003A <sub>16</sub> Access: R only Reset: 03 <sub>16</sub>
		TCM	Transmit-Complete (Transmission Register Empty) Flag (bit 0) 0: Data in the transmission register. 1: No data in the transmission register.							
		TBE	TX Buffer Empty Flag (bit 1) 0: Data in the TX Buffer. 1: No data in the TX Buffer.							
		RBF	RX Buffer Full Flag (bit 2) 0: No data in the RX Buffer. 1: Data in the RX Buffer.							
		PER	Receive Parity Error Flag (bit 3) 0: No receive parity error. 1: Receive parity error.							
		FER	Receive Framing Error Flag (bit 4) 0: No receive framing error. 1: Receive framing error.							
		OER	Receive Overrun Flag (bit 5) 0: No receive overrun. 1: Receive overrun.							
		SER	Receive Error Sum Flag (bit 6) 0: No receive error. 1: Receive error.							
		Bit 7	Reserved (Read "0")							

Figure 2-111. UxSTS Register

## 2.14.6 Transmit/Receive Format

### Transmit Method

(See Figure 2-112.)

#### Setup

- Define the baud rate by writing a value from 0-255 into the UxBRG.
- Set the Transmission Initialization Bit (TIN, bit 2 of UxCON), to "1". This will reset the transmit status to a value of 03<sub>16</sub>.
- Select the interrupt source to be either TBE or TCM by clearing or setting the Transmit Interrupt Source Selection Bit (TIS, bit 4 of UxCON).
- Configure the data format and clock selection by writing the appropriate value to UxMOD.
- Set the Clear-To-Send Enable Bit (CTS\_SEL, bit 5 of UxCON), if CTS handshaking will be used.
- Set the Transmit Enable Bit (TEN, bit 0 of UxCON), to "1".

#### Operation

- When data is written to the low-order byte of the transmit buffer (UxTRB1), TBE is cleared to "0". If 9-bit character length has been selected, the high-order byte of the transmit buffer (UxTRB2) should be written before the low-order byte (UxTRB1).
- If no data is being shifted out of the transmit shift register and CTS handshaking is disabled, the data written to the transmit buffer is transferred to the transmit shift register and the TCM flag in UxSTS is cleared to a "0". In addition, the TBE flag is set to a "1", signalling that the next byte of data can be written to the transmit buffer. If CTS handshaking is enabled, the operation described above does not take place until  $\overline{CTSx}$  is brought low.
- 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 to a "0" while data is still being transmitted, the transmitter will continue until the last bit is sent. This is also the case when CTS handshaking is enabled and  $\overline{CTSx}$  is brought back high during transmission.



- When the last bit is transmitted, the TCM flag is set to a “1” if the transmit buffer is empty, TEN is a “0”, or CTS handshaking is enabled and  $\overline{CTSx}$  is “1”. If the transmit buffer is not empty, TEN is a “1”, and CTS handshaking is disabled or CTS handshaking is enabled and  $\overline{CTSx}$  is low, the TCM flag is not set because transfer of the contents of the transmit buffer to the transmit shift register occurs immediately.

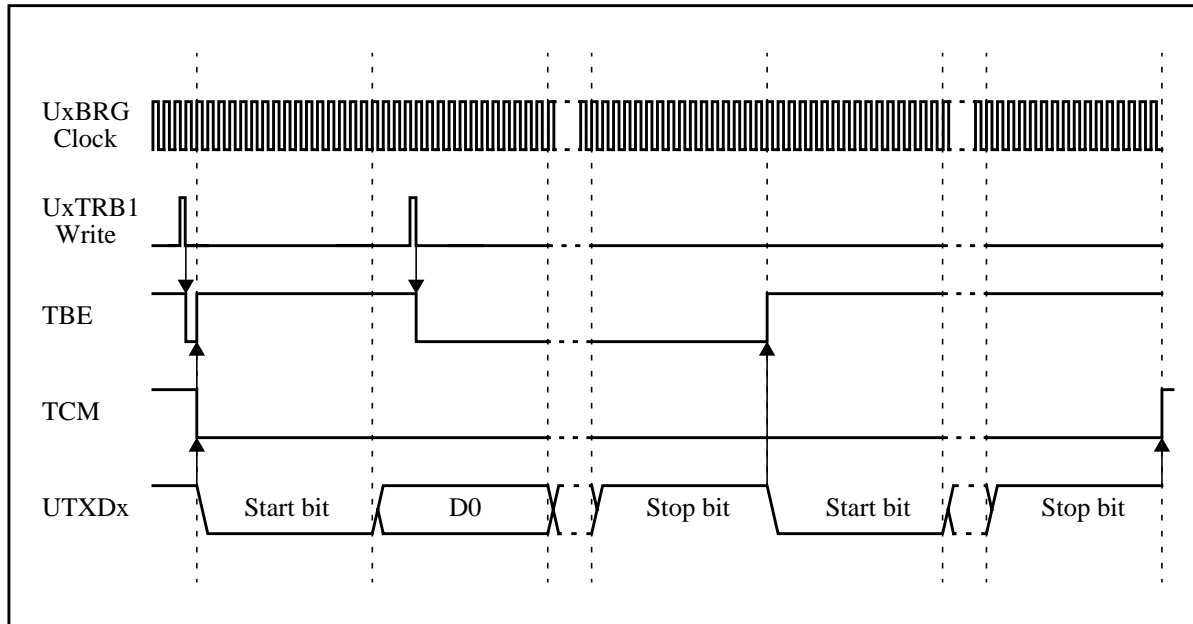


Figure 2-112. UART Transmit Operation Waveforms

### Receive Method

(See Figure 2-113.)

#### Set up

- Define the baud rate by writing a value from 0-255 into UxBRG.
- Set the Receive Initialization Bit (RIN, bit 3 in the UxCON), to “1”.
- Configure the data format and the clock selection by writing the appropriate value to UxMOD.
- Set the Request-To-Send Enable Bit (RTS\_SEL, bit 6 of UxCON), if RTS handshaking will be used.
- Set the Receive Enable Bit (REN, bit 1 in the UxCON), to “1”.

#### Operation

- When a falling edge is detected on the URXD<sub>x</sub> pin, the value on the pin is sampled at the basic clock rate, which is 16 times faster than the baud rate. If the pin is low for at least two cycles of the basic clock, the start bit is detected. Sampling is again performed three times in the approximate middle of the start bit. If two or more of the samples are low, the start bit is deemed valid. If two or more of the samples are not low, the start bit is invalidated and the UART again begins waiting for a falling edge on the URXD<sub>x</sub> pin.
- Once a valid start bit has been detected, input data received through the URXD<sub>x</sub> pin is read one bit at a time, LSB first, into the receive shift register. As is the case with the start bit, three samples are taken in the approximate middle of each data bit, the parity bit, and the stop bit(s). If two or more of the samples are low, a “0” is latched, and if two or more of the samples are high, a “1” is latched.
- When the number of bits specified by the data format has been received and the last stop bit is

detected, the contents of the receive shift register are transferred to the receive buffer and the Receive Buffer Full Flag in the UxSTS is set to a “1”, if a receive error has not occurred. The RBF interrupt request is also generated at this time if a receive error has not occurred. However, if a receive error did occur, the appropriate error flags are set and the Receive Error Sum (SER) interrupt request is generated at this time.

- When the low-order byte of the receive buffer (UxTRB1) is read, the Receive Buffer Full Flag is cleared, and the receive buffer is now ready for the next byte. If 9-bit character length has been selected, the high-order byte of the receive buffer (UxTRB2) should be read before reading the low-order byte (UxTRB1).

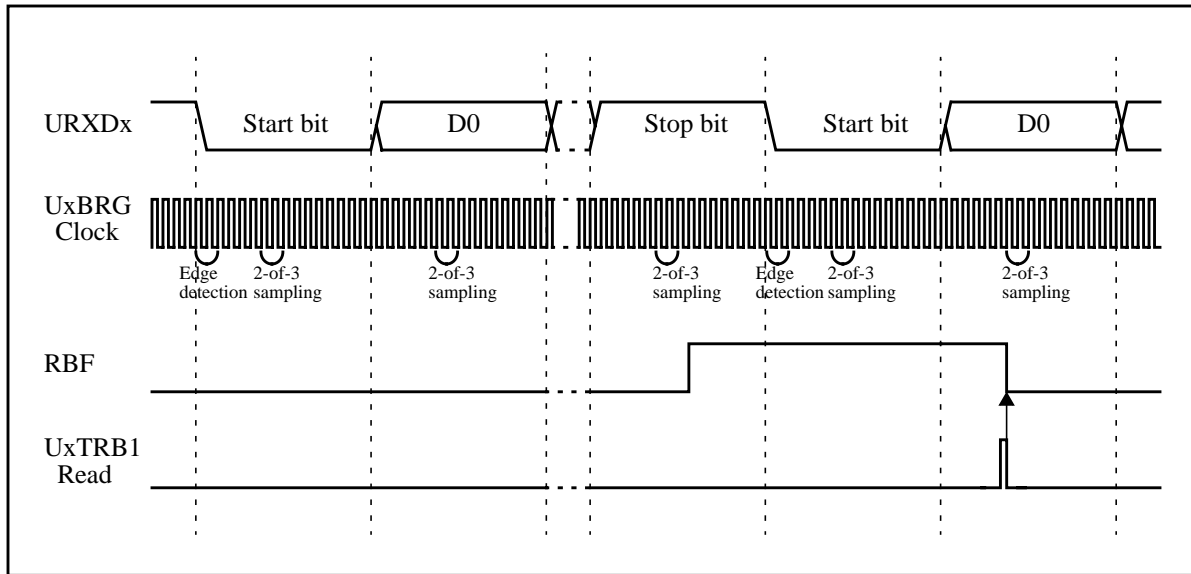


Figure 2-113. UART Receive Operation Waveforms

## 2.14.7 Interrupts

The transmit and receive interrupts are generated under the conditions described below. The generation of the receive interrupts when UART Address mode is enabled. The differences are described in section “2.14.9 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 or CTS handshaking is enabled and  $\overline{\text{CTSx}}$  is high (that is, when TCM flag set).

### Receive Interrupts

The UART generates the Receive Buffer Full (RBF) and Receive Error Sum (SER) 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 SER interrupt is generated when an overrun, framing or parity error occurs.



## 2.14.8 Clear-to Send ( $\overline{\text{CTSx}}$ ) and Request-to-Send ( $\overline{\text{RTSx}}$ ) Signals

The UART, as a transmitter, can be configured to recognize the Clear-to-Send ( $\overline{\text{CTSx}}$ ) input as a handshaking signal. As a receiver, the UART can be configured to generate the Request-to-Send ( $\overline{\text{RTSx}}$ ) handshaking signal.

### Clear-to-Send ( $\overline{\text{CTSx}}$ ) Input

CTS handshaking is enabled by setting the Clear-to-Send Enable Bit (CTS\_SEL, bit 5 of UxCON) to a "1". If CTS handshaking is enabled, when TEN is a "1" and the low-order byte of the transmit buffer (UxTRB1) is loaded, the UART begins the transmission process when the  $\overline{\text{CTSx}}$  pin is asserted (low input). After beginning a send operation, the UART does not stop sending until the transmission is completed, even if  $\overline{\text{CTSx}}$  is deasserted (high input). If TEN is cleared to "0", the UART will not stop transmitting and the port pins will remain under the control of the UART until the end of the transmission. If CTS handshaking is disabled and TEN is a "1", the UART begins the transmission process as soon as data is available in the low-order byte of the transmit buffer (UxTRB1). Figure 2-115 shows a timing example for  $\overline{\text{CTSx}}$ .

### Request-to-Send ( $\overline{\text{RTSx}}$ ) Output

RTS handshaking is enabled by setting the Request-to-Send Enable Bit (RTS\_SEL, bit 6 of UxCON) to a "1". When RTS handshaking is enabled, the UART drives the  $\overline{\text{RTSx}}$  output low or high based on the following conditions:

Assertion conditions (driven low):

- The Receive Enable Bit (REN) is set to a "1".
- Receive operation has completed with the reception of the last stop bit, REN is still a "1", and the programmable assertion delay has expired.

De-assertion conditions (driven high):

- A valid start bit is detected and REN is a "1".
- REN is cleared to a "0" before a receive operation is in progress.
- Receive operation has completed and REN is a "0".
- UART Receiver is initialized (RIN is set to a "1").

The delay time from the reception of the last stop bit to the re-assertion of  $\overline{\text{RTSx}}$  is programmable. The amount of delay is selected by setting the RTS Assertion Delay Count Bits (RTS3~0, bits 3 to 0 of UxRTSC) (see Figure 2-114). The time can be from no delay to 120 bit-times, with the delay beginning from the middle of the last stop bit. If a start bit is detected before the assertion delay has expired, the delay countdown is stopped and the  $\overline{\text{RTSx}}$  pin remains high. A full assertion delay countdown will begin again once the last stop bit of the incoming data has been received. Figure 2-115 shows a timing example for  $\overline{\text{RTSx}}$ .

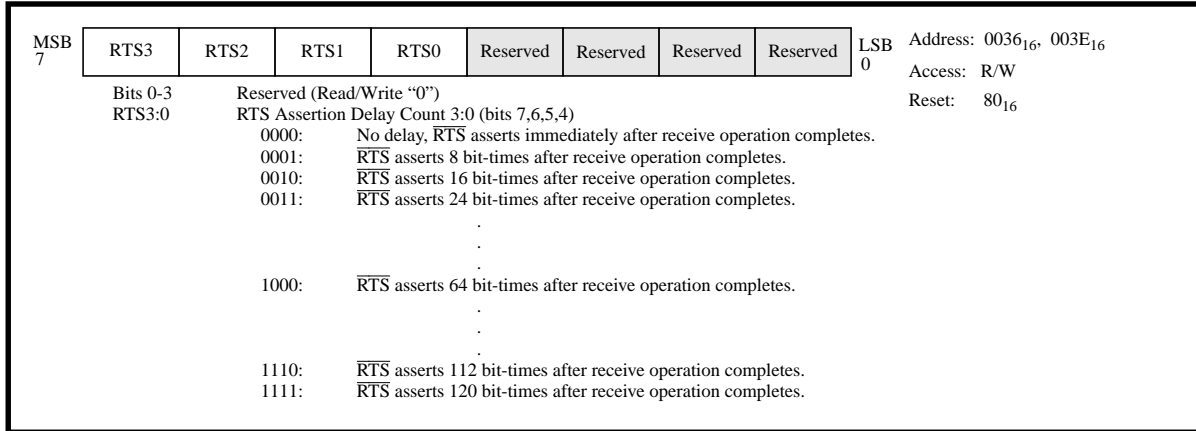


Figure 2-114. UxRTSC Register

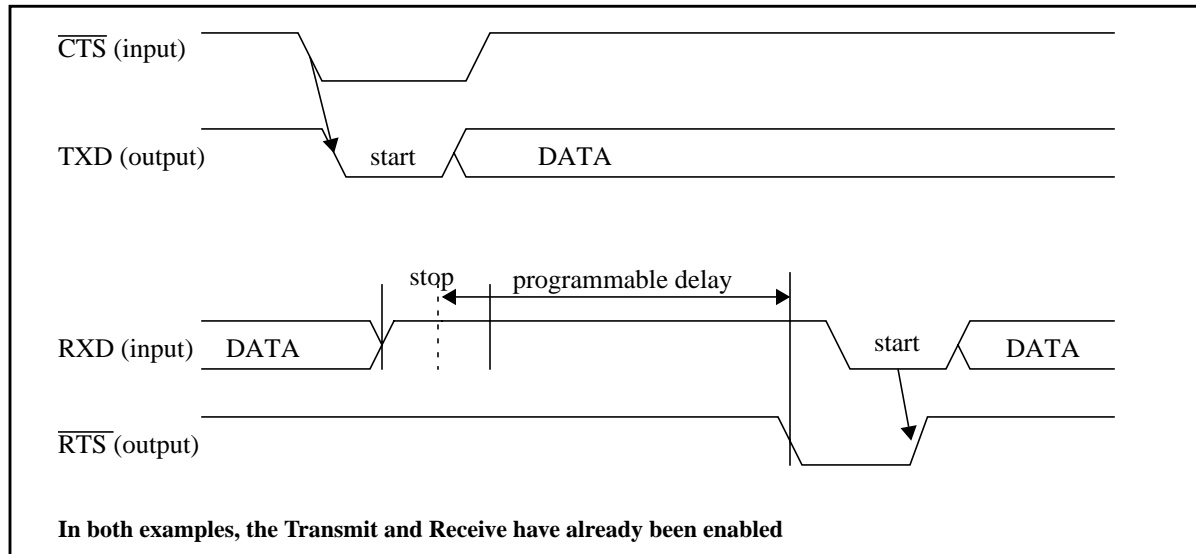


Figure 2-115.  $\overline{\text{CTSx}}$  and  $\overline{\text{RTSx}}$  Timing Examples

### 2.14.9 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 (LE1,0).

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.

## 2.15 Serial I/O

Address	Description	Acronym and Value at Reset
002A <sub>16</sub>	SIO shift register	SIOSHT=XX
002B <sub>16</sub>	SIO control register 1	SIOCON1=00
002C <sub>16</sub>	SIO control register 2	SIOCON2=00

Name	Pin
$\overline{\text{SRDY}}$	is multiplexed with P8 <sub>0</sub>
SCLK	is multiplexed with P8 <sub>1</sub>
SRXD	is multiplexed with P8 <sub>2</sub>
STXD	is multiplexed with P8 <sub>3</sub>

The Serial I/O has the following main features:

- Synchronous transmission or reception
- Handshaking via  $\overline{\text{SRDY}}$  output signal
- 8-bit character length
- Interrupt after transmission or reception
- 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 =  $\Phi$ ; if bit 1 of SIO Control Register2 is “1”, internal clock source = SCSGCLK.)
- External Clock (When SIO synchronous clock select bit is “1”, an external clock input from the SCLK pin is selected).

A block diagram of the clock synchronous SIO is shown in Figure 2-116.

### 2.15.1 SIO Control Register

The Serial I/O Control Register controls the various SIO functions (see Figure 2-117.). All of this register's bits can be read from and written to by software. At reset, this register is cleared to 00<sub>16</sub>.

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.

### 2.15.2 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. The start of a transfer is initiated by a write signal to the SIO shift register (address 002A<sub>16</sub>). The  $\overline{\text{SRDY}}$  signal then drops active low. On the negative edge of the transfer clock  $\overline{\text{SRDY}}$  returns high and the data is transmitted out the STXD pin. Data is latched in from the SRXD pin on the rising edge of the transfer clock. If an internal clock is selected, the STXD pin enters a high-impedance state after an 8-bit transfer is completed. 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 STXD high-impedance function after data is transferred.



Regardless of whether an internal or external clock is selected, after an 8-bit transfer, the interrupt request bit is set. Figure 2-119 shows the timing for the serial I/O with the LSB-first option selected.

The SIO can be operated in slave mode. In slave mode the  $\overline{\text{SRDY}}$  pin becomes an input from a master. If  $\overline{\text{SRDY}}$  is held high, the shift clock is inhibited,  $\text{STXD}$  is tri-stated, and shift count is reset. If  $\overline{\text{SRDY}}$  is held low, then the normal shift operation is performed.

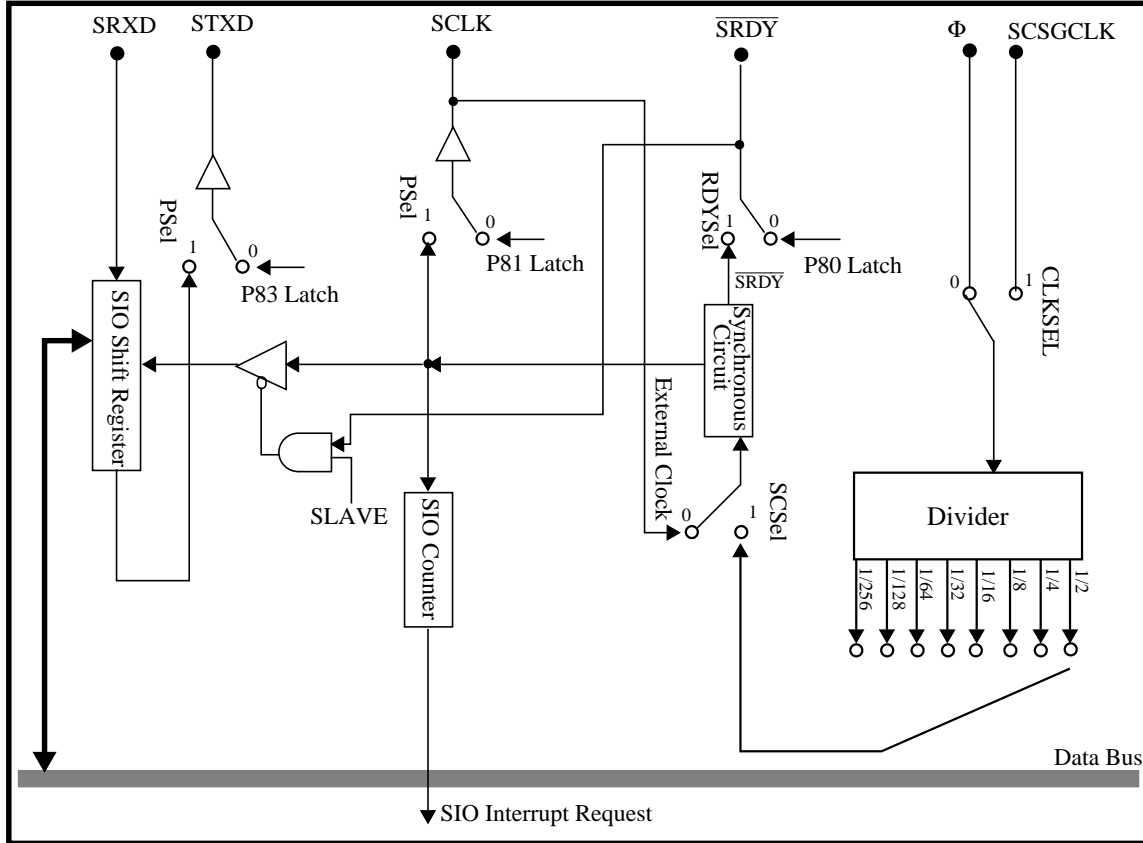


Figure 2-116. Clock Synchronous SIO Block Diagram

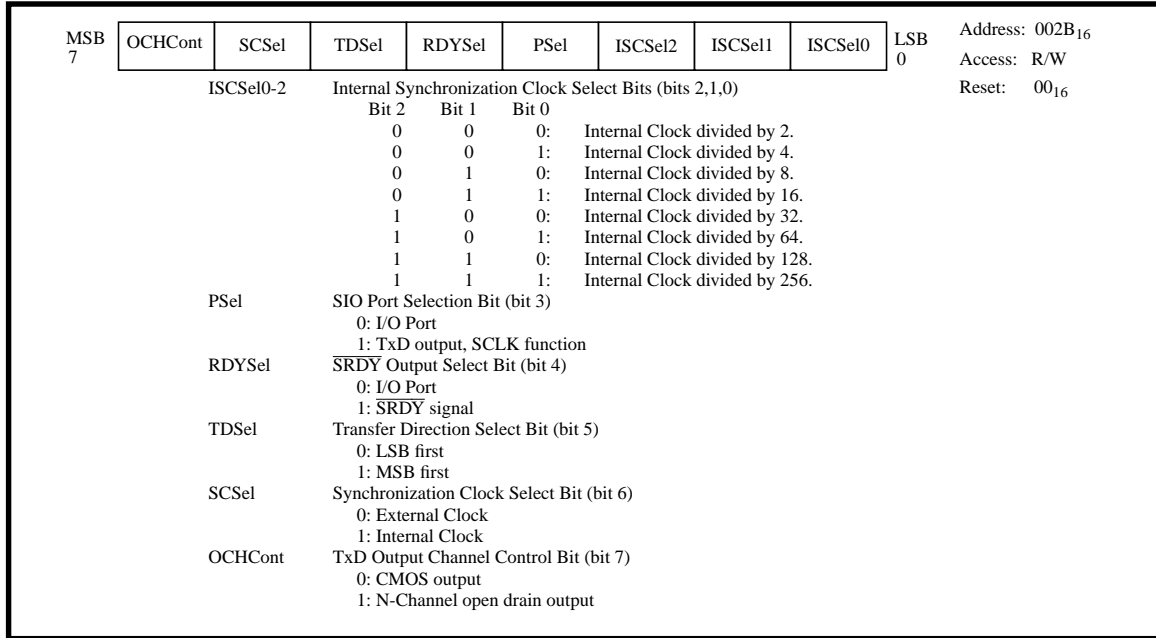


Figure 2-117. SIO Control Register 1

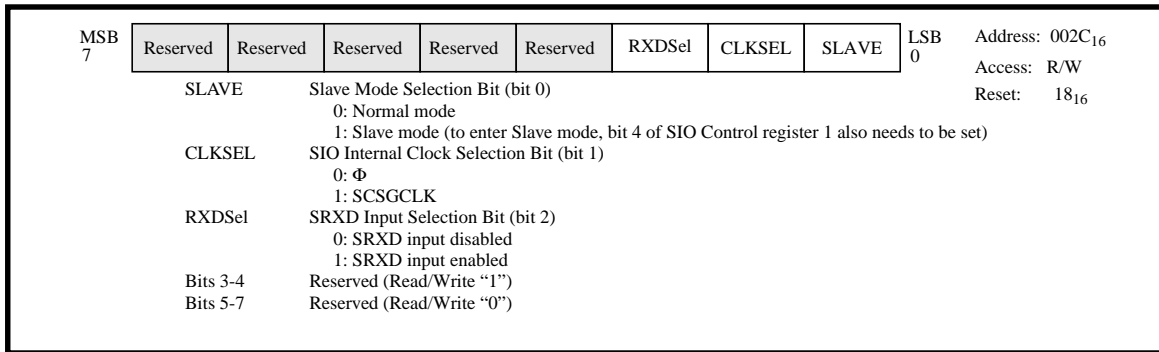


Figure 2-118. SIO Control Register 2

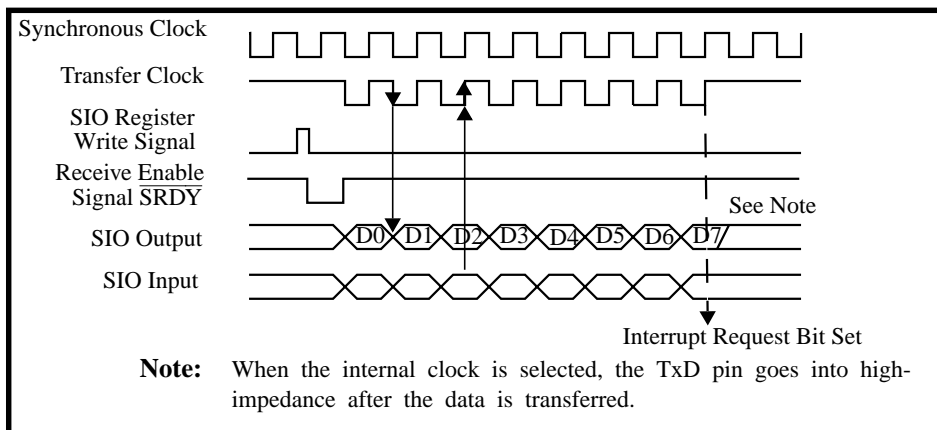


Figure 2-119. Normal Mode SIO Function Timing (with LSB-First selected)



## 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 mcu to be placed in a state where no internal excitation of the circuitry is taking place, thus resulting in extremely low power dissipation. The mcu enters the stop mode when the STP instruction is executed. The internal state of the mcu 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  $\Phi/8$  and the count source for Timer 2 is set to Timer 1 underflow.

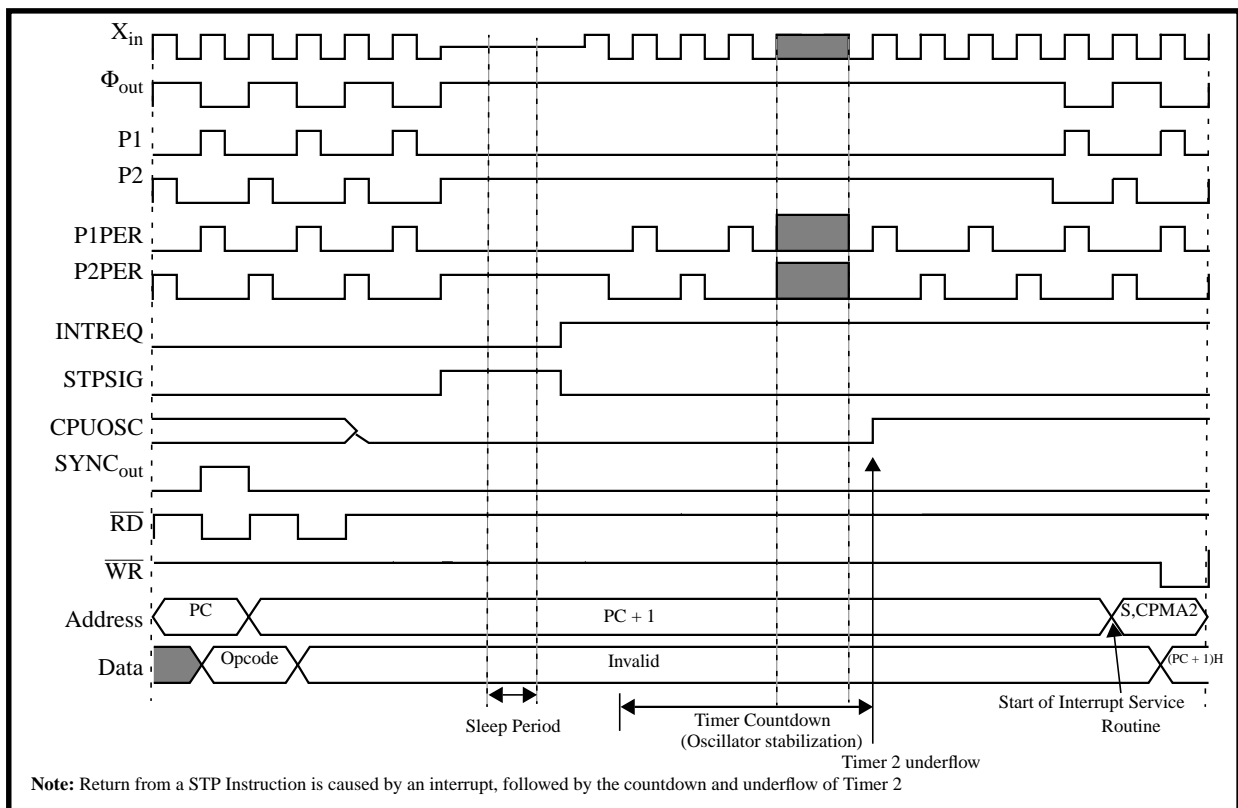


Figure 2-120. STP Cycle Timing Diagram

Oscillation is restarted (that is, 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 to a "1" and the I flag set to a "0" 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 mcu 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 2-120.

### 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 2-121.

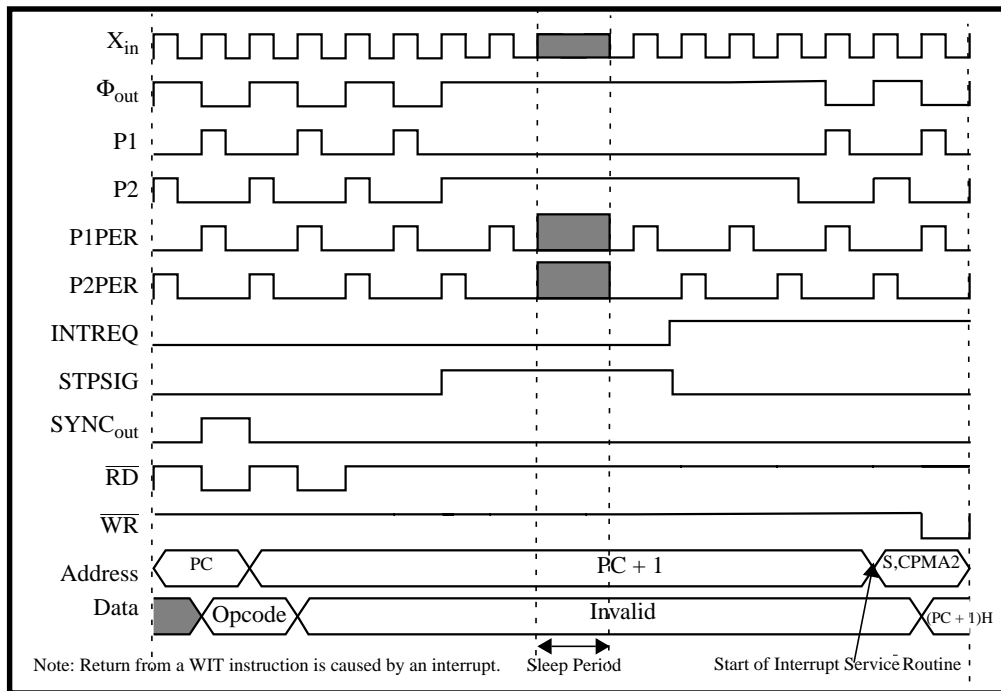


Figure 2-121. WIT Cycle Timing Diagram



## 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 4.75 and 5.25Volts. When the RESET pin returns high, the reset sequence commences (see Figure 2-122). 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<sub>16</sub> loaded in Timer 1 and 01<sub>16</sub> 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<sub>16</sub> and whose low-order byte is the contents of address FFFB<sub>16</sub>.

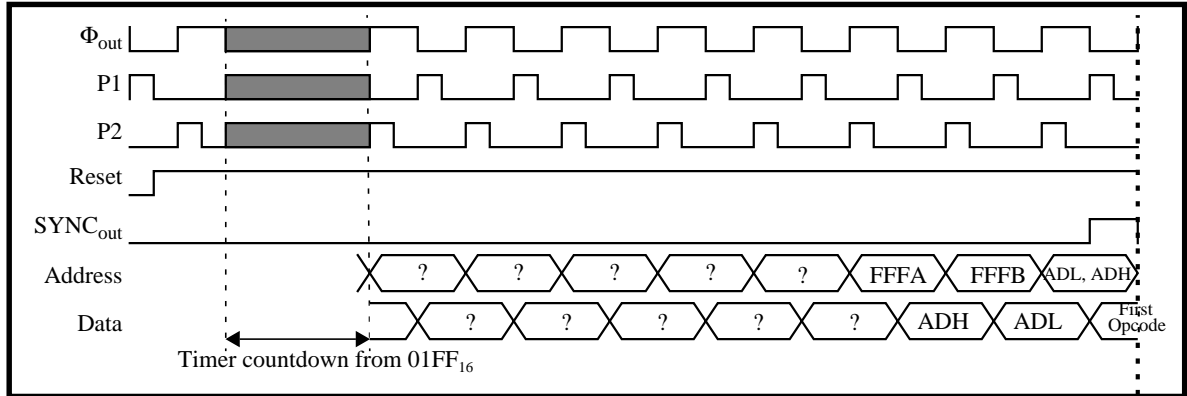


Figure 2-122. Internal Processing Sequence after RESET

## 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 2-123, the microcomputer can be returned to a normal state by pressing any one of the keys. Key-on wake-up is enabled in single-chip mode only.

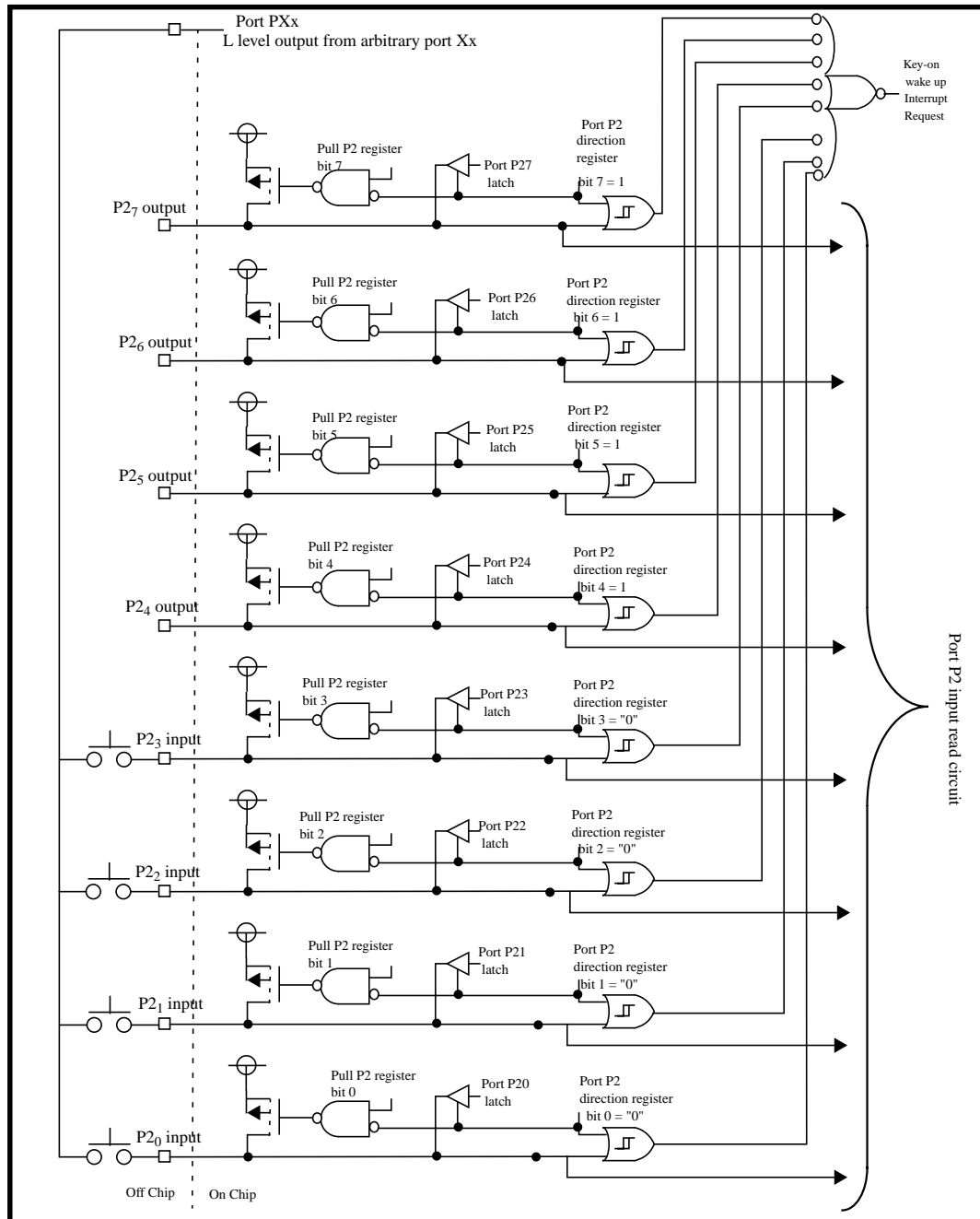


Figure 2-123. Port 2 with Key-on Wake-up Function





**PRELIMINARY**

# **Chapter 3**

## **Electrical Characteristics**

3.1 Absolute Maximum Ratings . . .	3-3
3.2 Recommended Operating conditions . . . . .	3-4
3.3 Electrical Characteristics . . . . .	3-6
3.4 Timing Requirements and Switching Characteristics . . . . .	3-8





## 3 Electrical Characteristics

### 3.1 Absolute Maximum Ratings

Table 3-1. Absolute Maximum Ratings

Symbol	Parameter	Conditions	Limits	Unit
$V_{CC}$	Power supply		-0.3 to 7.0	V
$AV_{CC}$	Analog power supply		-0.3 to $V_{CC} + 0.3$	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 $V_{CC} + 0.3$	V
$V_I$	Input voltage $\overline{RESET}$ , $X_{in}$ , $XC_{in}$		-0.3 to $V_{CC} + 0.3$	V
$V_I$	Input voltage $CNV_{SS}$		-0.3 to 13	V
$V_I$	Input voltage USB D+, D-		-0.5 to 3.8V	V
$V_O$	Output voltage P0, P1, P2, P3, P4, P5, P6, P7, P8, $X_{out}$ , $XC_{out}$		-0.3 to $V_{CC} + 0.3$	V
$P_D$	Power dissipation <sup>Note 1</sup>	$T_a = 25^\circ\text{C}$	750	mW
$T_{OPR}$	Operating temperature		-20 to +85	$^\circ\text{C}$
$T_{STG}$	Storage temperature		-40 to +125	$^\circ\text{C}$

**Note 1.** Maximum power dissipation is based on package heat dissipation characteristics, not chip power consumption.

## 3.2 Recommended Operating conditions

**Table 3-2. Recommended Operating Conditions**  
( $V_{CC} = 4.15$  to  $5.25V$ ,  $V_{SS} = 0V$ ,  $T_a = -20$  to  $85^\circ C$ , unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$V_{CC}$	Supply voltage	4.15	5	5.25	V
$AV_{CC}$	Analog supply voltage	4.15	5	$V_{CC}$	V
$V_{SS}$	Supply voltage		0		V
$AV_{SS}$	Analog supply voltage		0		V
$V_{IH}$	H input voltage	$\overline{RESET}$ , $X_{in}$ , $XC_{in}$ , $CNV_{SS}$	$0.8V_{CC}$	$V_{CC}$	V
$V_{IH}$	H input voltage	P0, P1, P2, P3, P4, P5, P6, P7, P8	$0.8V_{CC}$	$V_{CC}$	V
$V_{IH}$	H input voltage	P2 (When PTC6 = "0")	$0.5V_{CC}$	$V_{CC}$	V
$V_{IH}$	H input voltage	P5 <sub>7</sub> -P5 <sub>4</sub> , P6, P7 <sub>2</sub> (When MBI inputs and PTC7 = "1")	2.0	$V_{CC}$	V
$V_{IL}$	L input voltage	$\overline{RESET}$ , $X_{in}$ , $XC_{in}$ , $CNV_{SS}$	0	$0.2V_{CC}$	V
$V_{IL}$	L input voltage	P0, P1, P2, P3, P4, P5, P6, P7, P8	0	$0.2V_{CC}$	V
$V_{IL}$	L input voltage	P2 (When PTC6 = "0")	0	$0.16V_{CC}$	V
$V_{IL}$	L input voltage	P5 <sub>7</sub> -P5 <sub>4</sub> , P6, P7 <sub>2</sub> (When MBI inputs and PTC7 = "1")	0	0.8	V
$I_{OL}$ (peak)	L peak output current <sup>Note 1</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		10	mA
$I_{OL}$ (avg)	L average output current <sup>Note 2</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		5	mA
$I_{OH}$ (peak)	H peak output current <sup>Note 1</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		-10	mA
$I_{OH}$ (avg)	H average output current <sup>Note 2</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		-5	mA
$\Sigma I_{OL}$ (peak)	L total peak output current <sup>Note 3</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		80	mA
$\Sigma I_{OL}$ (avg)	L total average output current <sup>Note 4</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		40	mA
$\Sigma I_{OH}$ (peak)	H total peak output current <sup>Note 3</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		-80	mA
$\Sigma I_{OH}$ (avg)	H total average output current <sup>Note 4</sup>	P0, P1, P2, P3, P4, P5, P6, P7, P8		-40	mA
f(CNTR0)	TimerX - input frequency <sup>Note 5</sup>			5	MHz
f(CNTR1)	TimerY - input frequency <sup>Note 5</sup>			5	MHz
f( $X_{in}$ )	Clock frequency <sup>Note 5</sup>			24	MHz
f( $XC_{in}$ )	Clock frequency <sup>Note 5,6</sup>		32.768	50/5.0	KHz/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 current value measured over 100ms.

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

**Note 6.** The maximum oscillation frequency of 50KHz is for a crystal oscillator connected between  $XC_{in}$  and  $XC_{out}$ . An external clock signal having a maximum frequency of 5MHz can be input to  $XC_{in}$ .

### 3.3 Electrical Characteristics

**Table 3-3. Electrical Characteristics**  
( $V_{CC} = 4.15$  to  $5.25V$ ,  $V_{SS} = 0V$ ,  $T_a = -20$  to  $85^\circ C$ , unless otherwise noted)

Symbol	Parameters		Test Conditions	Limits			Unit
				Min	Typ.	Max	
$V_{OH}$	H output current	P0, P1, P2, P3, P4, P5, P6, P7, P8	$I_{oh} = -10mA$	$V_{CC} - 2.0$			V
$V_{OL}$	L output current	P0, P1, P2, P3, P4, P5, P6, P7, P8	$I_{ol} = 10mA$			2.0	V
$V_{T+}$ $\sim V_{T-}$	Hysteresis	CNTR0, CNTR1, INT0, INT1, Key-on wakeup (P2), RDY, $\overline{HOLD}$			0.5		V
		URXD1, URXD2 (SCLK), $\overline{CTS2}$ (SRXD), SRDY, CTS1			0.5		V
		$\overline{RESET}$			0.5		V
$I_{IH}$	H input current	P0, P1, P2, P3, P4, P5, P6, P7, P8	$V_i = V_{CC}$			5	$\mu A$
		$\overline{RESET}$ , $CNV_{SS}$				5	$\mu A$
		$X_{in}$			9	20	$\mu A$
		$XC_{in}$				5	$\mu A$
$I_{IL}$	L input current	P0, P1, P3, P4, P5, P6, P7, P8	$V_i = V_{SS}$			-5	$\mu A$
		P2	$V_i = V_{SS}$ (Pullups off)			-5	$\mu A$
			$V_{CC} = 5V$ , $V_i = V_{SS}$ (Pullups on)	-30	-70	-140	$\mu A$
		$\overline{RESET}$	$V_i = V_{SS}$			-5	$\mu A$
		$CNV_{SS}$				-20	
		$X_{in}$			-9	-20	$\mu A$
		$XC_{in}$				-5	$\mu A$
$V_{RAM}$	RAM retention voltage		Clocks stopped	2.0			V
$I_{CC}$	Supply current (Output transistors are isolated)	Normal Mode	$f(X_{in}) = 24MHz$ , $\Phi = 6MHz$ , USB operating, frequency synthesizer on <sup>Note 1</sup>		55	70	mA
			$f(X_{in}) = 24MHz$ , $\Phi = 12MHz$ , USB operating, frequency synthesizer on <sup>Note 1</sup>		70	90	mA
			$f(X_{in}) = 24MHz$ , $\Phi = 12MHz$ , USB suspended, frequency synthesizer on, USB clock disabled <sup>Note 2</sup>		35	45	mA
		Wait Mode	$f(X_{in}) = 24MHz$ , $\Phi = 12MHz$ , USB suspended, frequency synthesizer on, USB clock disabled <sup>Note 3</sup>		7.5	10	mA
			$f(XC_{in}) = 32KHz$ , $\Phi = 16KHz$ , USB disabled, frequency synthesizer off, transceiver voltage converter off <sup>Note 4</sup>		6	10	$\mu A$
		Stop Mode	Transceiver voltage converter on with USBC3 = "1" (low current mode)		200	250	$\mu A$
			$T_a = 25^\circ C$ , transceiver voltage converter off		0.1	1	$\mu A$
$T_a = 85^\circ C$ , transceiver voltage converter off				10	$\mu A$		

**Note 1.** Icc test conditions:

- Single chip mode (run state)
- Square wave clock input on  $X_{in}$  ( $X_{out}$  drive disabled)
- I/O pins isolated
- Frequency synthesizer running
- USB operating with transceiver voltage converter enabled
- CPU and DMAC running
- Timers and SCSG running
- Both UARTs transmitting
- MBI and SIO disabled

**Note 2.** Icc test conditions same as Note 1 except for the following:

- USB in suspend state with USB clock disabled

**Note 3.** Icc test conditions:

- Single chip mode (wait state)
- Square wave clock input on  $X_{in}$  ( $X_{out}$  drive disabled)
- I/O pins isolated
- Frequency synthesizer running
- USB in suspend state with USB clock disabled
- Transceiver voltage converter enabled
- Timers and SCSG running
- CPU and DMAC not running
- Both UARTs, SIO, and MBI disabled

**Note 4.** Icc test conditions:

- Single chip mode (wait state)
- $X_{in}/X_{out}$  oscillation disabled
- Square wave clock input on  $XC_{in}$  ( $XC_{out}$  drive disabled)
- I/O pins isolated
- Frequency synthesizer disabled
- USB and USB clock disabled
- Transceiver voltage converter disabled
- Timers and SCSG running
- CPU and DMAC not running
- Both UARTs, SIO, and MBI disabled

### 3.4 Timing Requirements and Switching Characteristics

**Table 3-4. Timing Requirements and Switching Characteristics**  
(V<sub>CC</sub> = 4.15 to 5.25V, V<sub>SS</sub> = 0V, Ta = -20 to 85°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
<b>INPUTS</b>					
tw(RESET)	RESET input "Low" pulse width	2			μs
tc(X <sub>in</sub> )	Clock input cycle time	41.66			ns
twh(X <sub>in</sub> )	Clock input "High" pulse width	0.4*tc(X <sub>in</sub> )			ns
twl(X <sub>in</sub> )	Clock input "Low" pulse width	0.4*tc(X <sub>in</sub> )			ns
tc(XC <sub>in</sub> )	Clock input cycle time	200			ns
twh(XC <sub>in</sub> )	Clock input "High" pulse width	0.4*tc(XC <sub>in</sub> )			ns
twl(XC <sub>in</sub> )	Clock input "Low" pulse width	0.4*tc(XC <sub>in</sub> )			ns
<b>INTERRUPTS</b>					
tc(INT)	INT0, INT1 input cycle time	140			ns
twh(INT)	INT0, INT1 input "High" pulse width	55			ns
twl(INT)	INT0, INT1 input "Low" pulse width	55			ns
tc(CNTR1)	CNTR0, CNTR1 input cycle time	200			ns
twh(CNTR1)	CNTR0, CNTR1 input "High" pulse width	80			ns
twl(CNTR1)	CNTR0, CNTR1 input "Low" pulse width	80			ns
<b>TIMERS</b>					
td(Φ-TOUT)	TIMER TOUT delay time <sup>Note 1</sup>			15	ns
td(Φ-CNTR0)	TIMER CNTR0 delay time (pulse output mode) <sup>Note 1</sup>			15	ns
tc(CNTRE0)	TIMER CNTR0 input cycle time (event counter mode)	200			ns
twh(CNTRE0)	TIMER CNTR0 input "High" pulse width (event counter mode)	0.4*tc(CNTRE0)			ns
twl(CNTRE0)	TIMER CNTR0 input "Low" pulse width (event counter mode)	0.4*tc(CNTRE0)			ns
td(Φ-CNTR1)	TIMER CNTR1 delay time (pulse output mode) <sup>Note 1</sup>			15	ns
tc(CNTRE1)	TIMER CNTR1 input cycle time (event counter mode)	200			ns
twh(CNTRE1)	TIMER CNTR1 input "High" pulse width (event counter mode)	0.4*tc(CNTRE1)			ns
twl(CNTRE1)	TIMER CNTR1 input "Low" pulse width (event counter mode)	0.4*tc(CNTRE1)			ns
<b>SIO</b>					
tc(SCLKE)	SIO external clock input cycle time	400			ns
twh(SCLKE)	SIO external clock input "High" pulse width	190			ns
twl(SCLKE)	SIO external clock input "Low" pulse width	180			ns
tsu(SRXD-SCLKE)	SIO receive setup time (external clock)	15			ns
th(SCLKE-SRXD)	SIO receive hold time (external clock)	10			ns
td(SCLKE-STXD)	SIO transmit delay time (external clock)			25	ns
tv(SCLKE-SRDY)	SIO $\overline{\text{SRDY}}$ valid time (external clock)			26	ns
tc(SCLKI)	SIO internal clock output cycle time	166.66			ns
twh(SCLKI)	SIO internal clock output "High" pulse width	0.5*tc(SCLKI)-5			ns
twl(SCLKI)	SIO internal clock output "Low" pulse width	0.5*tc(SCLKI)-5			ns
tsu(SRXD-SCLKI)	SIO receive setup time (internal clock)	20			ns
th(SCLKI-SRXD)	SIO receive hold time (internal clock)	5			ns
td(SCLKI-STXD)	SIO transmit delay time (internal clock)			5	ns





Table 3-4. Timing Requirements and Switching Characteristics

(V<sub>CC</sub> = 4.15 to 5.25V, V<sub>SS</sub> = 0V, Ta = -20 to 85°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
<b>MBI (Separate R and W Type Mode)</b>					
tsu(S-R)	$\overline{S0}, \overline{S1}$ setup time for read	0			ns
tsu(S-W)	$\overline{S0}, \overline{S1}$ setup time for write	0			ns
th(R-S)	$\overline{S0}, \overline{S1}$ hold time for read	0			ns
th(W-S)	$\overline{S0}, \overline{S1}$ hold time for write	0			ns
tsu(A-R)	A0 setup time for read	10			ns
tsu(A-W)	A0 setup time for write	10			ns
th(R-A)	A0 hold time for read	0			ns
th(W-A)	A0 hold time for write	0			ns
tw(R)	Read pulse width	50			ns
tw(W)	Write pulse width	50			ns
tsu(D-W)	Data input setup time before write	25			ns
th(W-D)	Data input hold time after write	0			ns
ta(R-D)	Data output enable time after read			40	ns
tv(R-D)	Data output disable time after read	10			ns
tv(R-OBF)	OBF output transmission time after read			40	ns
td(W-IBF)	$\overline{IBF}$ output transmission time after write			40	ns
<b>MBI (R/W Type Mode)</b>					
tsu(S-E)	$\overline{S0}, \overline{S1}$ setup time	0			ns
th(E-S)	$\overline{S0}, \overline{S1}$ hold time	0			ns
tsu(A-E)	A0 setup time	10			ns
th(E-A)	A0 hold time	0			ns
tsu(RW-E)	R/ $\overline{W}$ setup time	10			ns
th(E-RW)	R/ $\overline{W}$ hold time	10			ns
tw(E)	Enable pulse width	50			ns
tw(E-E)	Enable pulse interval	50			ns
tsu(D-E)	Data input setup time before write	25			ns
th(E-D)	Data input hold time after write	0			ns
ta(E-D)	Data output enable time after read			40	ns
tv(E-D)	Data output disable time after read	10			ns
tv(E-OBF)	OBF output transmission time after E inactive			40	ns
td(E-IBF)	$\overline{IBF}$ output transmission time after E inactive			40	ns

**Note 1.** Timer clock is  $\Phi$  or a derivative of  $\Phi$ .

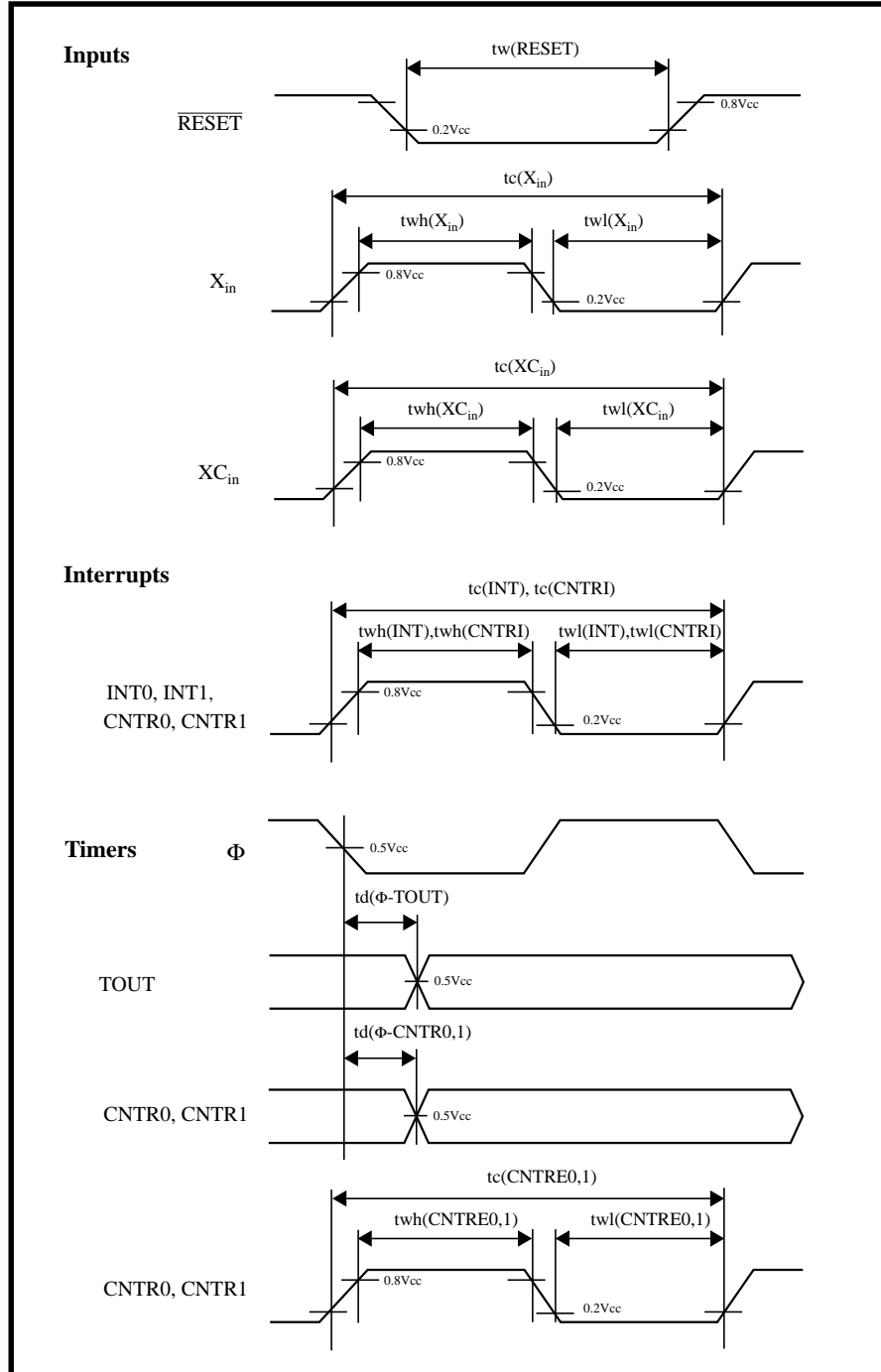


Figure 3-1. Reset, Clock, Interrupts and Timers Timing Diagram

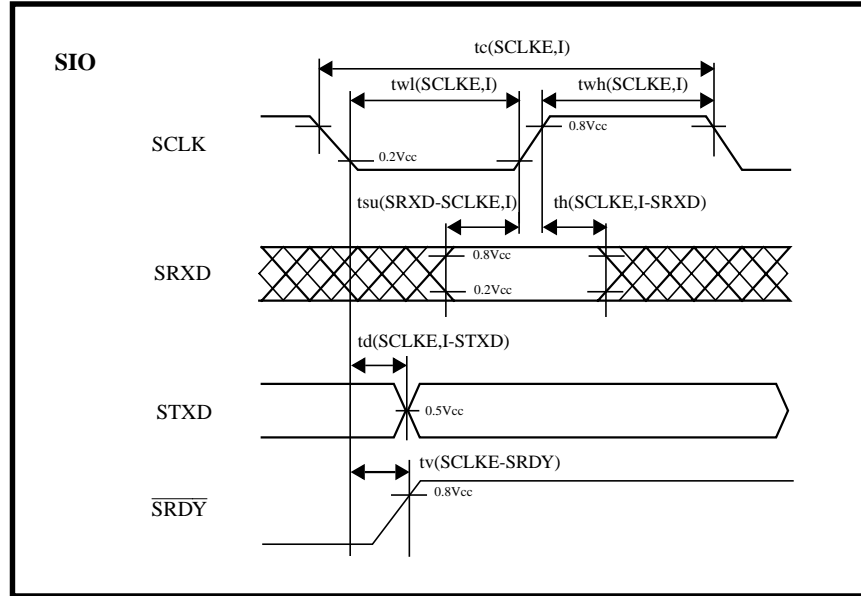


Figure 3-2. SIO Timing Diagram

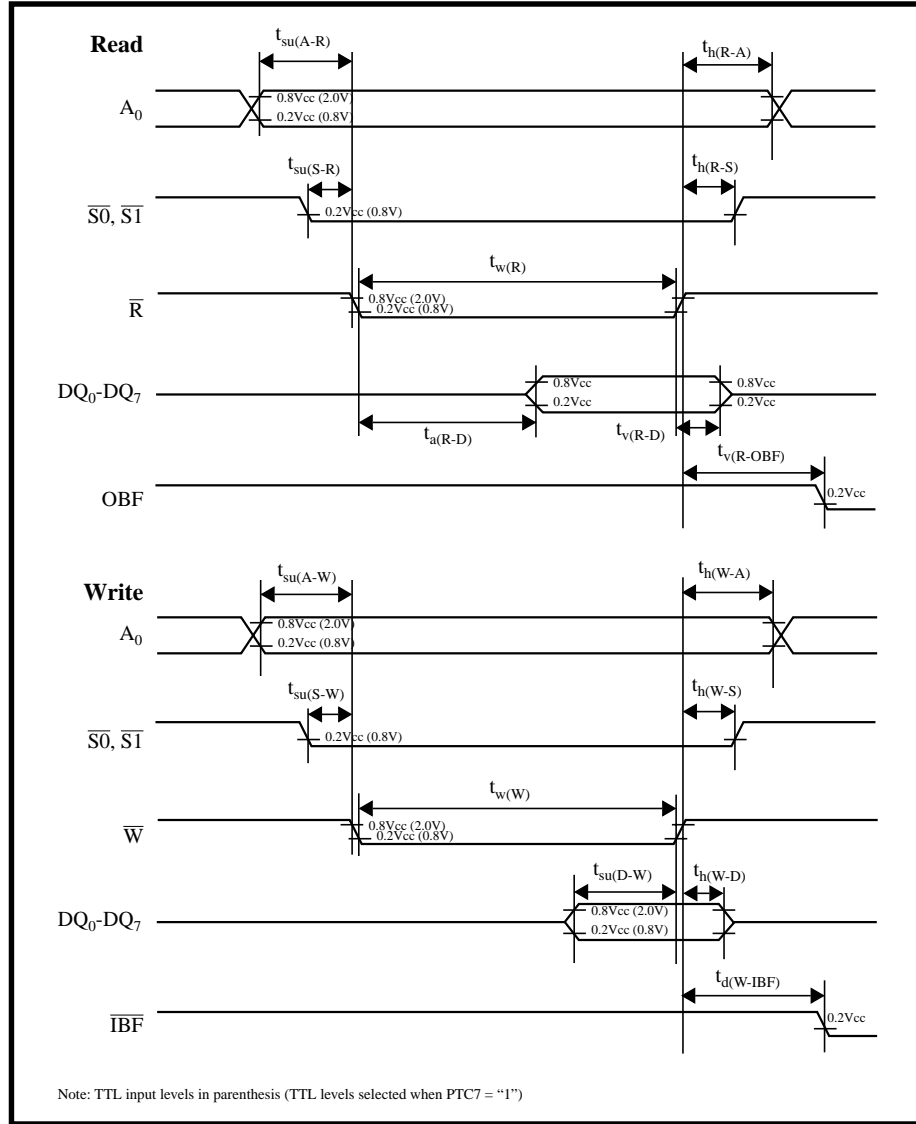


Figure 3-3. MBI Timing Diagram (Separate  $\overline{R}$  and  $\overline{W}$  Type Mode)

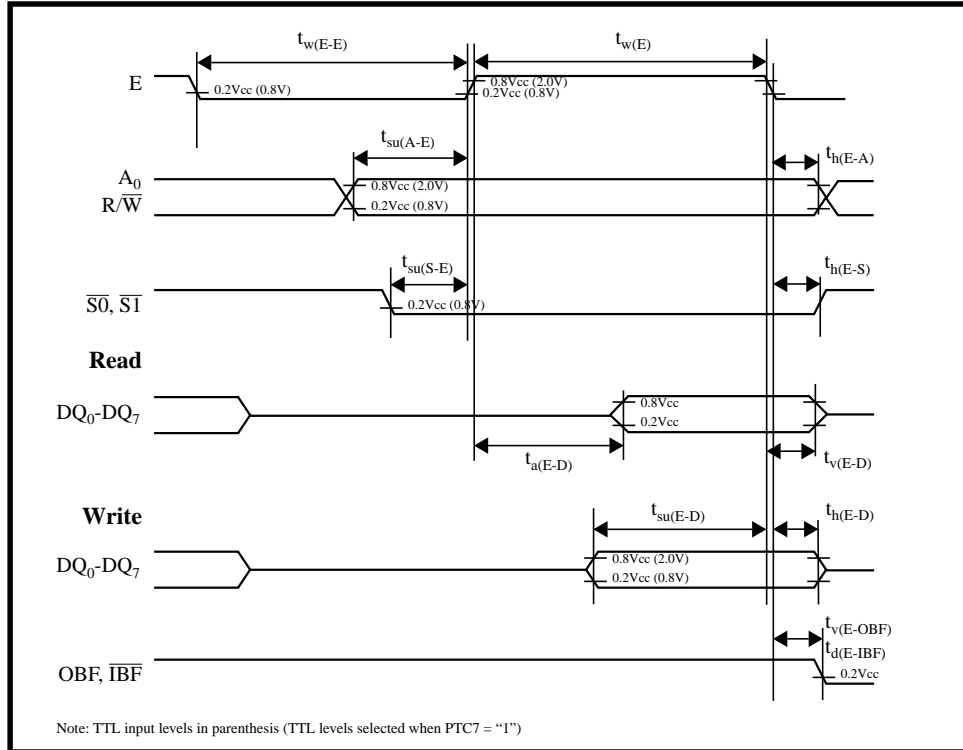


Figure 3-4. MBI Timing Diagram (R/ $\bar{W}$  Type Mode)

**Table 3-5. Memory Expansion Mode and Microprocessor Mode Timing**  
( $V_{CC} = 4.15$  to  $5.25V$ ,  $V_{SS} = 0V$ ,  $T_a = -20$  to  $85^{\circ}C$ , unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
tc( $\Phi$ )	$\Phi$ clock cycle time	83.33			ns
twh( $\Phi$ )	$\Phi$ clock "H" pulse width	$0.5 * t_c(\Phi) - 5$			ns
twl( $\Phi$ )	$\Phi$ clock "L" pulse width	$0.5 * t_c(\Phi) - 5$			ns
td( $\Phi$ -AH)	Address bus AB15-AB8 delay time with respect to $\Phi$			31	ns
tv( $\Phi$ -AH)	Address bus AB15-AB8 valid time with respect to $\Phi$	5			ns
td( $\Phi$ -AL)	Address bus AB7-AB0 delay time with respect to $\Phi$			33	ns
tv( $\Phi$ -AL)	Address bus AB7-AB0 valid time with respect to $\Phi$	5			ns
td( $\Phi$ -WR)	$\overline{WR}$ delay time			6	ns
tv( $\Phi$ -WR)	$\overline{WR}$ valid time	3			ns
td( $\Phi$ -RD)	$\overline{RD}$ delay time			6	ns
tv( $\Phi$ -RD)	$\overline{RD}$ valid time	3			ns
td( $\Phi$ -SYNC)	SYNC <sub>OUT</sub> delay time			6	ns
tv( $\Phi$ -SYNC)	SYNC <sub>OUT</sub> valid time	4			ns
td( $\Phi$ -DMA)	DMA <sub>OUT</sub> delay time			25	ns
tv( $\Phi$ -DMA)	DMA <sub>OUT</sub> valid time	5			ns
tsu(RDY- $\Phi$ )	RDY setup time with respect to $\Phi$	21			ns
th( $\Phi$ -RDY)	RDY hold time with respect to $\Phi$	0			ns
tsu(HOLD- $\Phi$ )	$\overline{HOLD}$ setup time	21			ns
th( $\Phi$ -HOLD)	$\overline{HOLD}$ hold time	0			ns
td( $\Phi$ -HLDA)	$\overline{HLDA}$ delay time			25	ns
tv( $\Phi$ -HLDA)	$\overline{HLDA}$ valid time			25	ns
tsu(DB- $\Phi$ )	Data bus setup time with respect to $\Phi$	7			ns
th( $\Phi$ -DB)	Data bus hold time with respect to $\Phi$	0			ns
td( $\Phi$ -DB)	Data bus delay time with respect to $\Phi$			22	ns
tv( $\Phi$ -DB)	Data bus valid time with respect to $\Phi$ <sup>Note 1</sup>	13			ns
twl(WR)	$\overline{WR}$ pulse width	$0.5 * t_c(\Phi) - 5$			ns
twl(RD)	$\overline{RD}$ pulse width	$0.5 * t_c(\Phi) - 5$			ns
td(AH-WR)	$\overline{WR}$ delay time after stable address AB15-AB8	$0.5 * t_c(\Phi) - 28$			ns
td(AL-WR)	$\overline{WR}$ delay time after stable address AB7-AB0	$0.5 * t_c(\Phi) - 30$			ns
tv(WR-AH)	Address bus AB15-AB8 valid time with respect to $\overline{WR}$	0			ns
tv(WR-AL)	Address bus AB7-AB0 valid time with respect to $\overline{WR}$	0			ns
td(AH-RD)	$\overline{RD}$ delay time after stable address AB15-AB8	$0.5 * t_c(\Phi) - 28$			ns
td(AL-RD)	$\overline{RD}$ delay time after stable address AB7-AB0	$0.5 * t_c(\Phi) - 30$			ns
tv(RD-AH)	Address bus AB15-AB8 valid time with respect to $\overline{RD}$	0			ns
tv(RD-AL)	Address bus AB7-AB0 valid time with respect to $\overline{RD}$	0			ns
tsu(RDY-WR)	RDY setup time with respect to $\overline{WR}$	27			ns
th(WR-RDY)	RDY hold time with respect to $\overline{WR}$	0			ns
tsu(RDY-RD)	RDY setup time with respect to $\overline{RD}$	27			ns
th(RD-RDY)	RDY hold time with respect to $\overline{RD}$	0			ns
tsu(DB-RD)	Data bus setup time with respect to $\overline{RD}$	13			ns
th(RD-DB)	Data bus hold time with respect to $\overline{RD}$	0			ns
td(WR-DB)	Data bus delay time with respect to $\overline{WR}$			20	ns
tv(WR-DB)	Data bus valid time with respect to $\overline{WR}$ <sup>Note 1</sup>	10			ns

**Note 1.** . Measurement conditions:  $I_{ohl} = \pm 5ma$ ,  $C_L = 50pF$

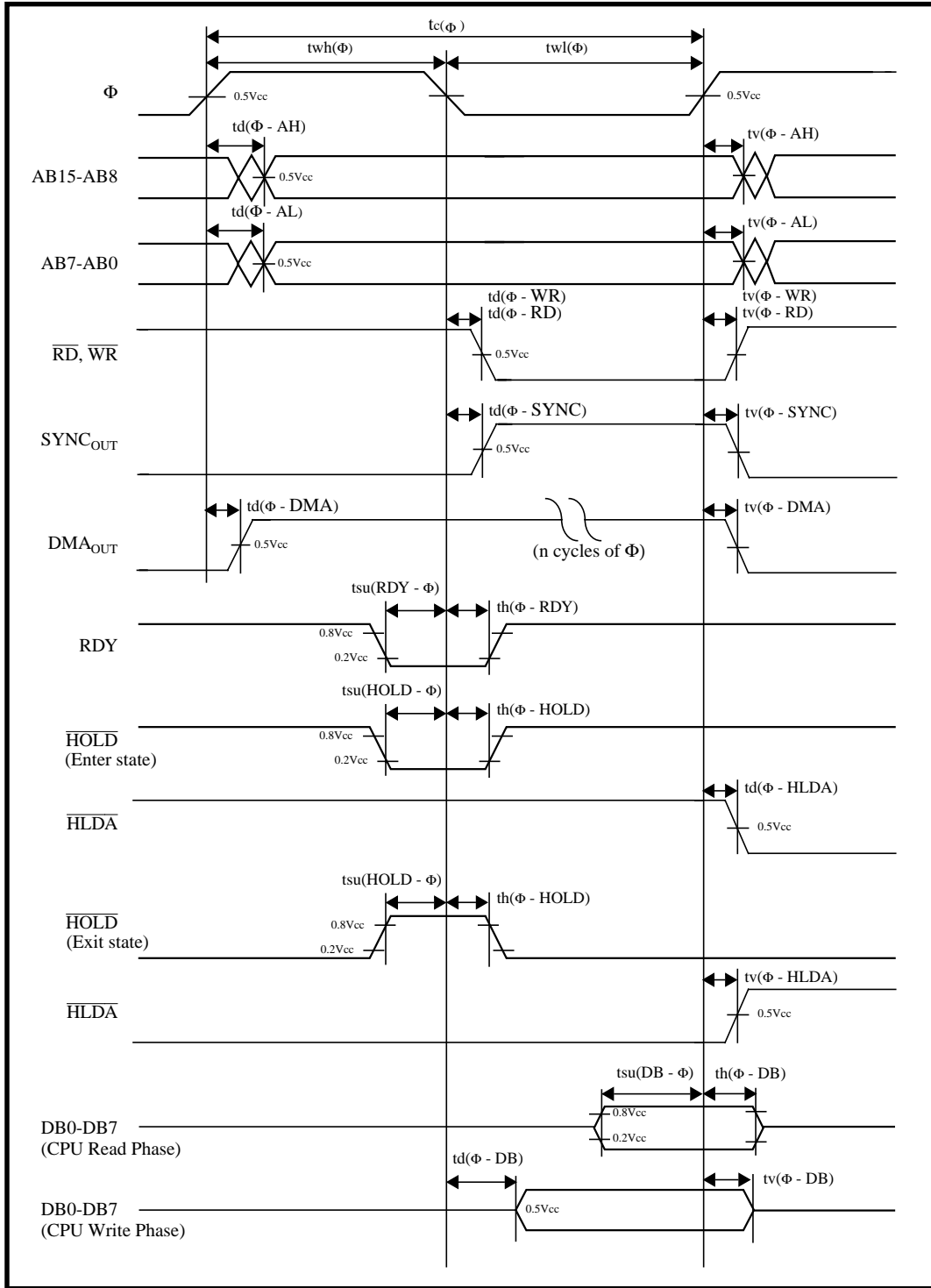


Figure 3-5. Microprocessor and Memory Expansion Mode Timing Diagram 1

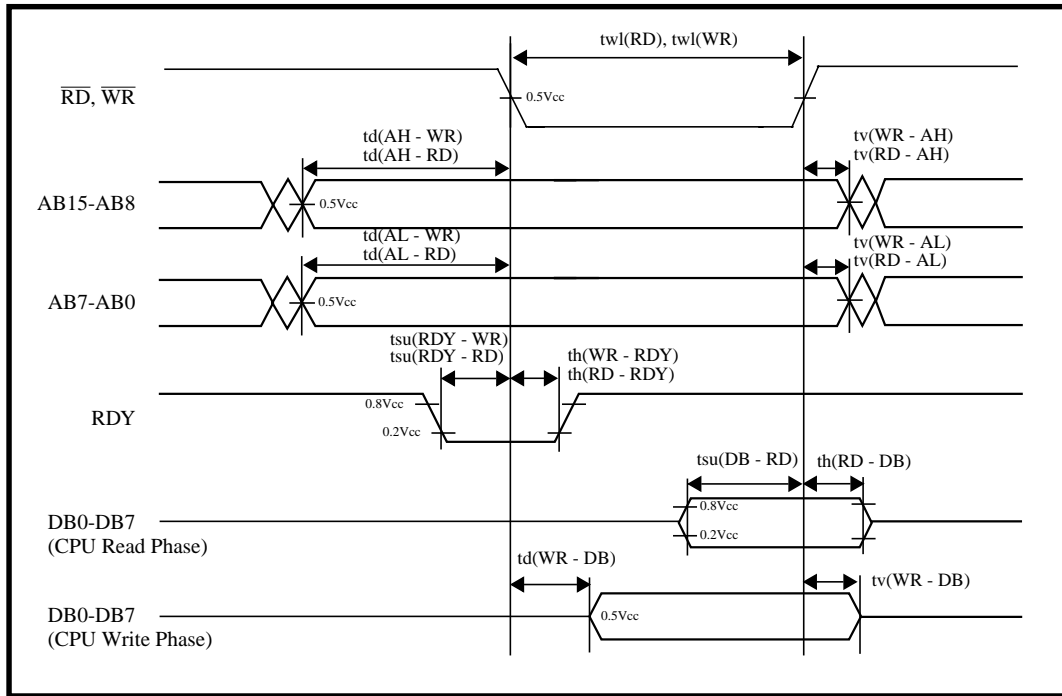


Figure 3-6. Microprocessor and Memory Expansion Mode Timing Diagram 2

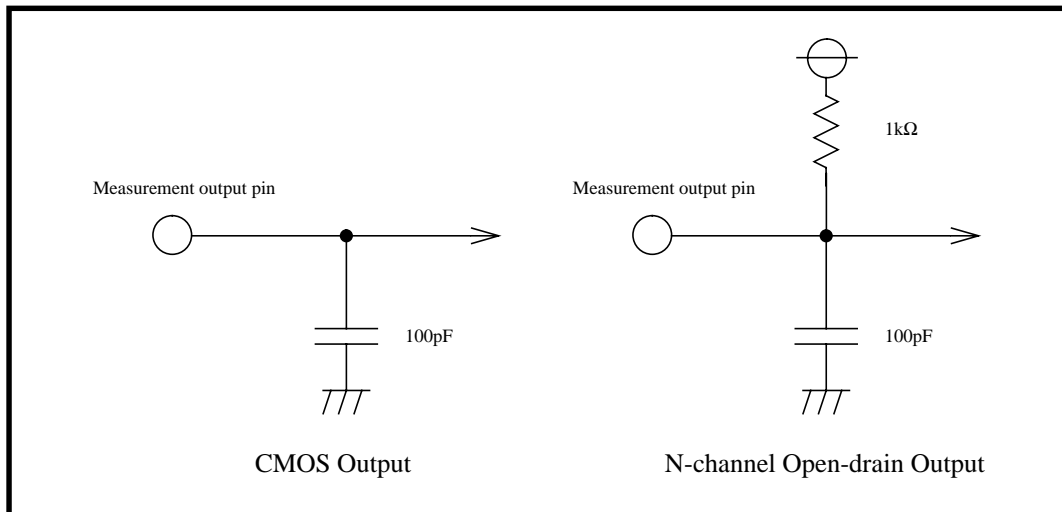


Figure 3-7. Output Switching Characteristics Measurement Circuits





**PRELIMINARY**

# **Chapter 4**

## **Application Notes**

4.1 DMAC .....	4-3
4.2 UART .....	4-4
4.3 Timer .....	4-5
4.4 Frequency Synthesizer Interface .....	4-6
4.5 USB Transceiver .....	4-7
4.6 Using the Frequency Synthesizer and DC-DC Converter .....	4-8
4.7 Ports .....	4-12
4.8 Programming Notes .....	4-13





---

# 4 Application Notes

---

## 4.1 DMAC

### 4.1.1 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 DMAC channel 0 from UART1 receive buffer to user RAM.

- Write  $08_{16}$  to DMA0M1 so that after each transfer the destination register will be decremented by one and the source register will remain unchanged.
- Write  $00_{16}$  to the low-order byte of the destination register (DMA0DL) and  $03_{16}$  to the high-order byte of the destination register (DMA0DH) so that the data received by the UART is placed in page three of the user RAM starting from address  $0300_{16}$ .
- Write  $34_{16}$  to the low-order byte of the source register (DMA0SL) and  $00_{16}$  to the high-order byte of the source register (DMA0SH) so that the DMAC reads from address  $0034_{16}$ , which is the low-order byte of the UART receive buffer.
- Write to the transfer count register (DMA0CL/H) with a 16-bit value that corresponds to the number of transfers to occur before flag CRUF and the DMAC channel 0 interrupt are set.
- Set the DMAC transfer initiating source to the UART receive interrupt by writing  $01_{16}$  to DMA0M2.
- Place the UART in the desired configuration for data reception by writing to the UART control (U1CON), UART mode (U1MOD), and UART baud rate (U1BRG) registers.
- Disable the UART receive interrupt from being serviced by the CPU by setting to a "0" bit 6 of interrupt control register A (ICONA).
- Enable the DMAC channel 0 interrupt by setting bit 4 of ICONA to "1".
- Enable DMAC channel 0 and reset the initiating source sample latch by writing  $C1_{16}$  to DMA0M2.

The DMA controller will transfer one byte of data from the UART receive buffer to third page user RAM each time that the UART1 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 decrements by one after each transfer. When it underflows, flag D0UF and the DMAC channel 0 interrupt are set. In the DMAC channel 0 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 (UxTRB2) 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 "0".
- The RTS Control Register (UxRTSC) is reset to  $80_{16}$  when the Receive Initialization Bit (RIN) is set to "1". When programming the RTS delay, ensure the RIN bit is set prior to programming the delay.

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



## 4.3 *Timer*

### 4.3.1 Usage

- If Port 4<sub>3</sub> is read when Timer X Pulse Output mode is being used, the value returned is the pulse output signal fed from the timer to the port.
- If Port 4<sub>4</sub> is read when Timer Y Pulse Output mode is being used, the value returned is the pulse output signal fed from the timer to the port.
- If Port 5<sub>1</sub> is read when Timer1/Timer2 Pulse Output mode is being used, the value returned is the pulse output signal fed from the timer to the port.

**Table 4-1. Initial Values of Timer Pulse Outputs**

<b>Timer</b>	<b>Selection Bit</b>	<b>Initial Output Value</b>
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/ Timer 2	T <sub>out</sub> Output Active Edge Selection Bit (T123M5)	0: Logic H 1: Logic L

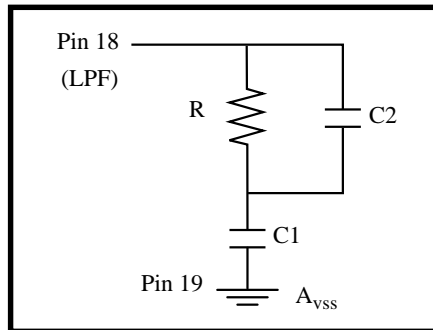
## 4.4 Frequency Synthesizer Interface

All passive components should be in close proximity to pin 18 (LPF). The recommended values are as follows:

**Table 4-2. Recommended Values**

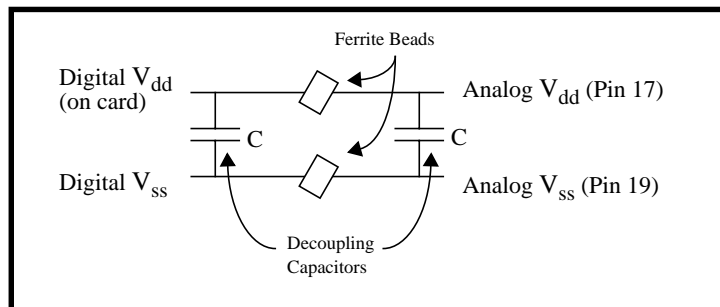
R 000 $\Omega$	1/8 watt	10%
C2=680 pf	5 V	10%
C1=0.1 $\mu$ f	5 V	10%

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



**Figure 4-1. LPF Filter Schematic**

Analog  $V_{SS}$  and Analog  $V_{DD}$ , pins 19 and 17 should have isolated connectors to the digital  $V_{SS}$  and  $V_{DD}$  ground planes. Figure 4-2 illustrates the power supply isolation.



**Figure 4-2. Power Supply Isolation**



## 4.5 USB Transceiver

When using the on-chip voltage converter to supply the necessary 3.3V to the driver circuit, a capacitor must be connected between Ext. Cap (pin 72) and  $V_{SS}$  (pin 73). The Capacitor spec is as follows: voltage: 5V, tolerance: 10%, type: mica, glass, polystyrene or low-loss ceramic. The recommended value of the capacitor on Ext. Cap is 2.2 $\mu$ F in parallel with a 0.1 $\mu$ F. The start-up time for this value of the capacitor is 3.2ms. The start-up time is approximately (1ms/ $\mu$ F) + 1ms.

After enabling the on-chip voltage converter, a certain amount of time must pass before a WIT or STP instruction is executed. The amount of time is given by (C+1)ms, when C is the value in  $\mu$ F of the external capacitance connected to the Ext. Cap pin. For example, if the external capacitance is 2.2 $\mu$ F, at least 3.2ms must elapse from the time that the on-chip voltage converter is enabled until a WIT or STP instruction is executed.

In order to meet the impedance matching requirements of the USB Specification, a 33 $\Omega$  resistor must be added to USB D+ (pin 70) and to USB D- (pin 71). In addition, a 33pF capacitor should be connected between USB D+ and USB D- after the 33 $\Omega$  resistors. The placement of external components is illustrated in Figure 4-3.

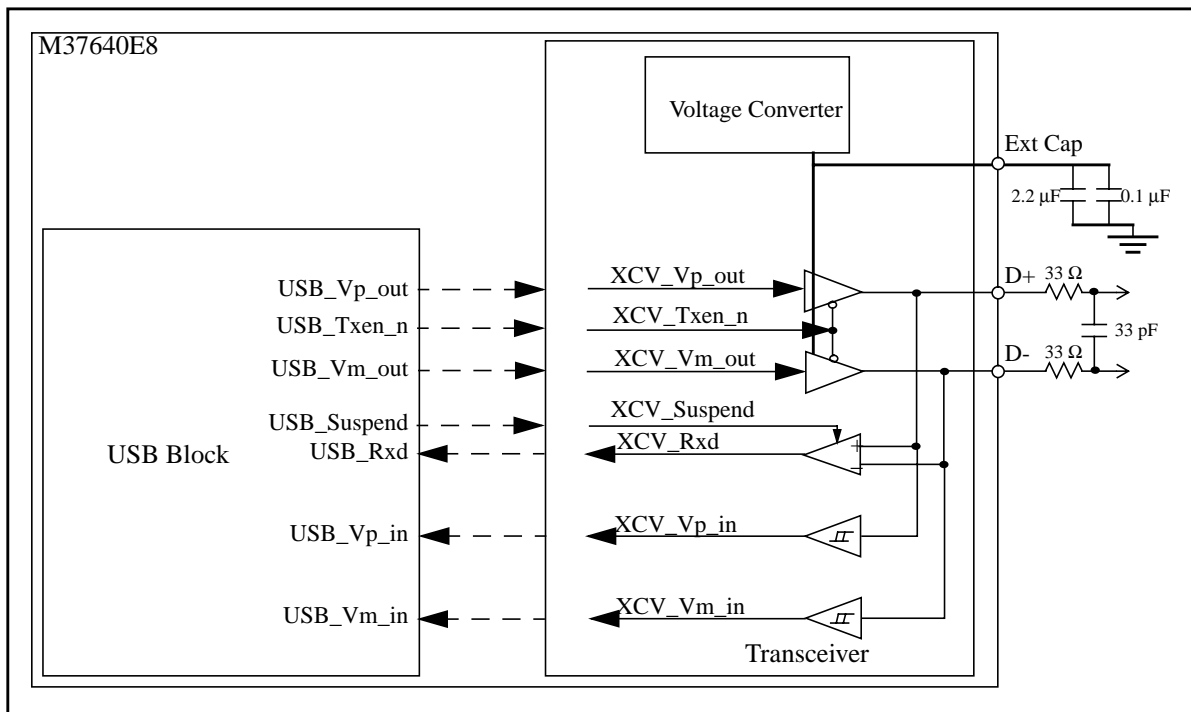


Figure 4-3. Configuration of External USB Components

## 4.6 Using the Frequency Synthesizer and DC-DC Converter

This section presents the recommended method of setting up and using the frequency synthesizer that generates the 48MHz clock needed by the USB FCU and the DC-DC converter that provides power to the D+/D- drivers.

### 4.6.1 Reset of USB Related Registers

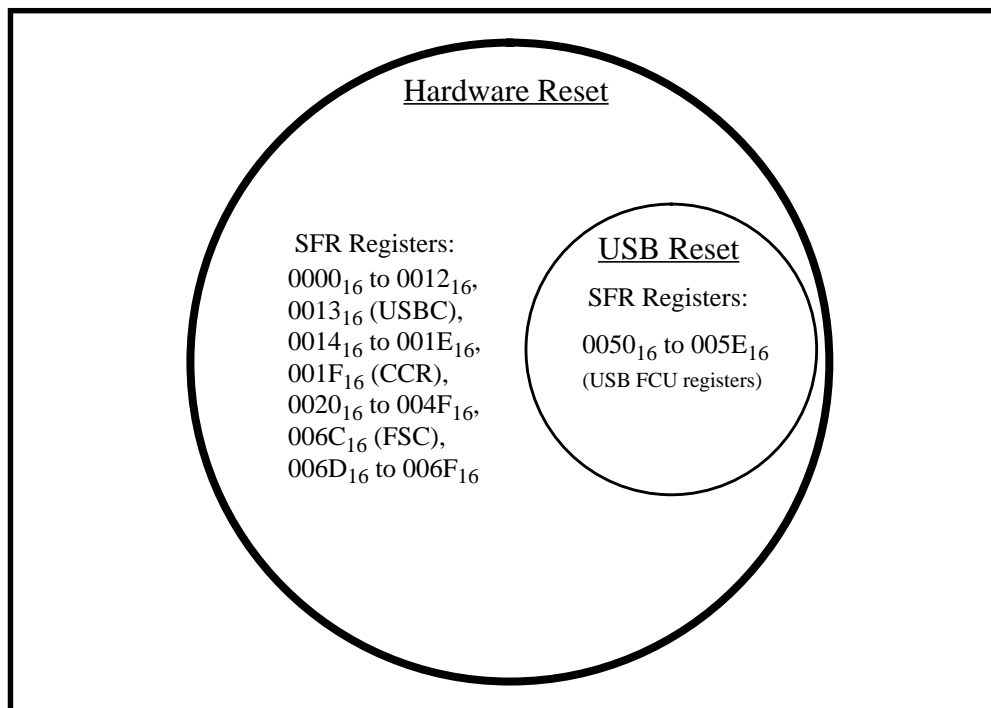


Figure 4-4. SFR Reset Venn Diagram

The special function registers (SFRs) that govern the operation of the frequency synthesizer, DC-DC converter and USB FCU are affected by one or more reset events. The addresses of the special function registers (SFRs) that are affected by Hardware Reset, USB Reset, or both are shown in Figure 4-4.

All resettable SFRs, including SFRs and other registers internal to the USB FCU, are affected by a Hardware Reset, which occurs when the  $\overline{\text{RESET}}$  pin is brought low or an undefined opcode is fetched. See Table 2.1 for a complete listing of SFRs and their reset values.

Only registers internal to the USB FCU are reset when a USB Reset sent by the Host/Hub is detected. These USB registers are reset to their default values except for bit 5 of USBIS2 (USB Reset Interrupt Status Flag), which is set to a “1”. USB FCU registers are registers from address 0050<sub>16</sub> to 005E<sub>16</sub> and all other registers within the USB FCU, many of which the MCU does not have direct access to (e.g. FIFO address pointers). The USB FIFO registers are not reset. Other SFRs such as USBC, FSC, and CCR are not affected by a USB Reset.





## 4.6.2 Set up of Frequency Synthesizer and DC-DC Converter

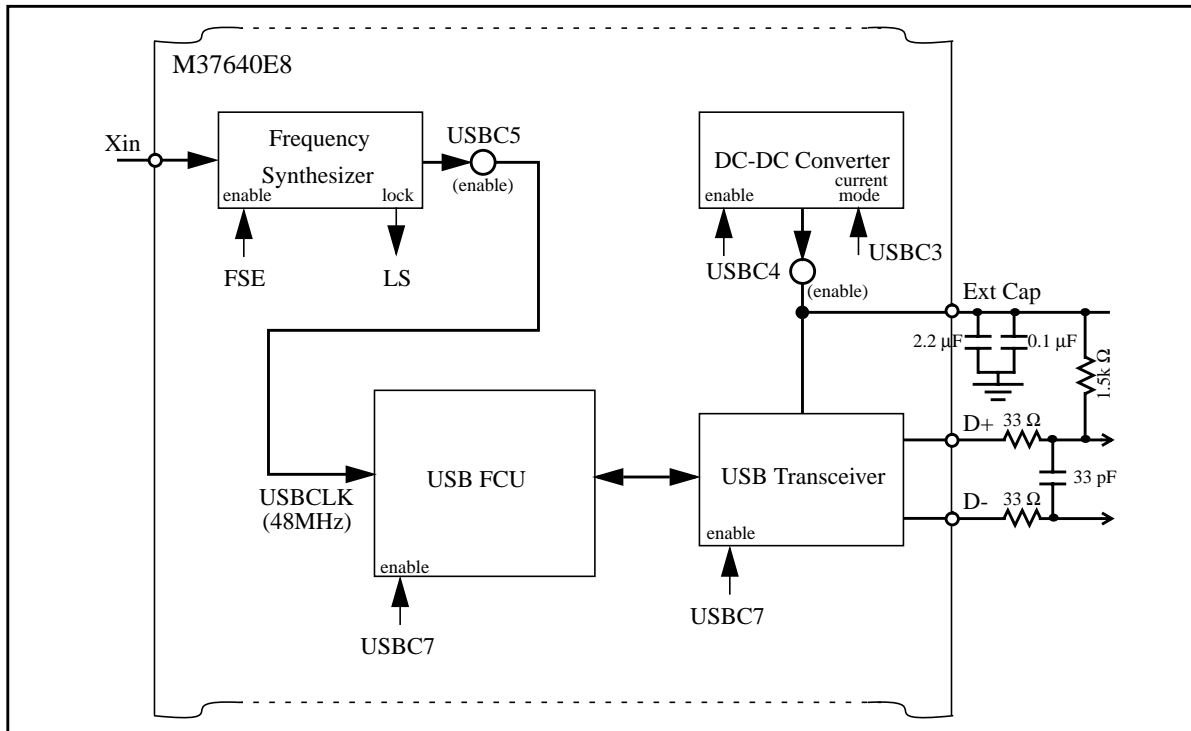


Figure 4-5. PLL, DC-DC Converter and USB Functional Block Diagram

A functional block diagram of the USB system on the M37640E8 which shows how the control signals affect operation is given in Figure 4-5.

### 4.6.2.1 Set up after Hardware Reset

A Hardware Reset occurs when either the  $\overline{\text{RESET}}$  pin is brought low for more than 2 $\mu\text{s}$  or an invalid opcode is fetched by the CPU. The frequency synthesizer (PLL) and DC-DC converter should be set up as follows in the Hardware Reset routine (see Figure 4-6):

- Power up the M37640E8 and other components on the peripheral device for less than 100mA operation. The current limit only applies for bus powered devices.
- Configure the PLL for 48MHz f(VCO) operation.
- Enable the PLL by setting FSE (bit 0 of the Frequency Synthesizer Control Register (FSC)) to a "1", then wait for 2ms.
- Check the lock status bit (LS, bit 7 of FSC).
  - If the bit is a "1", go on.
  - If the bit is a "0", wait 0.1ms longer and then re-check the bit.
- Enable the DC-DC converter in high current mode by setting USBC4 (bit 4 of the USB Control Register (USBC)) to a "1" and keeping USBC3 (bit 3 of USBC) a "0". High current mode should always be used during normal USB operation. Low current mode should only be used during a USB suspend.
- Wait (C + 1)ms (where C equals the external capacitance connected to the Ext Cap pin in  $\mu\text{F}$ ) for the voltage on Ext Cap to reach a steady state voltage of approximately 3.3V. (Since the D+ pullup is connected to the Ext Cap pin, the upstream hub will detect that the peripheral device has been plugged in once the voltage on D+ reaches approximately 2.0V.)

Example: A 2.3 $\mu$ F capacitor connected to Ext Cap requires 3.3ms for the voltage on Ext Cap to be stable.

- Enable the USB clock by setting USBC5 (bit 5 of USBC) to a "1". (If the USB clock and FCU are enabled before the voltage on Ext Cap is stable, a phantom USB Reset may be detected, or the actual USB Reset may not be detected.)
- Wait at least 4 cycles of  $\Phi$ , then enable the USB FCU by setting USBC7 (bit 7 of USBC) to a "1".
- Enable other blocks as necessary.

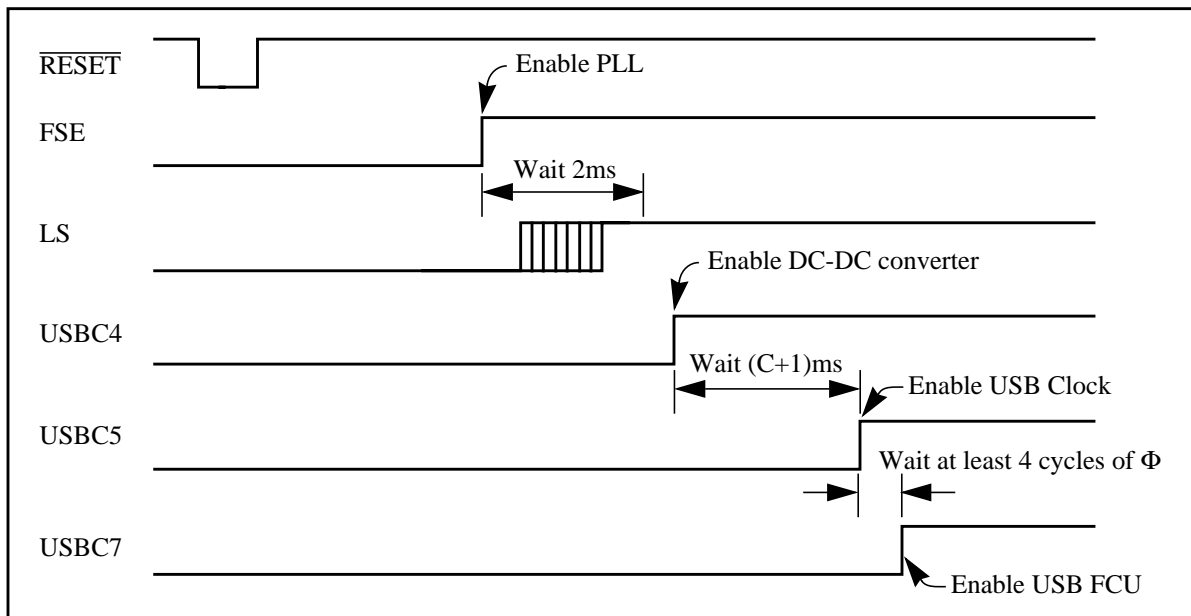


Figure 4-6. PLL and DC-DC Converter Set Up Timing after Hardware Reset

#### 4.6.2.2 Set up after USB Reset Signaling Detected

A USB Reset is detected by the USB FCU when an SE0 is present on D+/D- for at least 2.5 $\mu$ s. Detection of a USB Reset results in bit 5 of USB Interrupt Status Register 2 (USBIS2) being set to a "1" and the registers within the USB FCU being reset to their default values. Register USBC and the PLL registers are not affected by a USB Reset. A USB Function Interrupt request is also generated when the USB Reset is detected.

No modifications to the frequency synthesizer or DC-DC converter configuration should be made in the USB Function Interrupt routine. However, all USB FCU registers (addresses 0050<sub>16</sub> to 005F<sub>16</sub>) must be reconfigured to their pre-enumeration state.

#### 4.6.2.3 Set up after USB Suspend Detected

A USB Suspend occurs if the USB FCU does not detect any bus activity on D+/D- for at least 3ms. Detection of a suspend results in bit 7 of USBIS2 being set to a "1". If bit 7 of the USB Interrupt Enable Register 2 (USBIE2) is a "1", a USB Function Interrupt request is also generated.

The configuration of the frequency synthesizer and DC-DC converter should be changed as follows in the USB Function Interrupt routine (if the device is bus powered):

- Disable the USB clock by setting USBC5 (bit 5 of USBC) to a "0".
- Disable the PLL by setting FSE (bit 0 of FSC) to a "0".



- Change the DC-DC converter from high current mode to low current mode by setting USBC3 (bit 3 of the USBC) to a “1”.
- Perform other tasks to reduce total current to below 500uA.
- Execute the STP instruction. Make sure to enable the USB Suspend/Resume Signaling Interrupt Enable Bit (bit 7 of USBIE2 = “1”), the USB Function Interrupt (bit 0 of IREQA = “1”) and clear the I flag prior to executing the STP instruction so the MCU can wake up once resume signaling is detected.

Note that no action may be necessary if the device is self powered.

#### 4.6.2.4 Set up after USB Resume Signaling Detected

A resume occurs when the USB FCU is in the suspend state and detects non-idle signaling on D+/D-. Detection of a resume results in bit 6 of USBIS2 being set to a “1”. If bit 7 of USBIE2 is a “1”, a USB Function Interrupt request is also generated. If the MCU was in the stop state prior to the detection of the resume, the USB Function Interrupt request will cause the MCU to wake up from the stop state. See section 2.16.1 “Stop Mode” for details on waking up from the stop state.

The configuration of the frequency synthesizer and DC-DC converter should be changed as follows in the USB Function Interrupt routine (if the device is bus powered):

- Change the DC-DC converter from low current mode to high current mode by setting USBC3 (bit 3 of the USBC) to a “0”.
- Re-enable the PLL for 48MHz f(VCO) by setting FSE (bit 0 of the FSC) to a “1”, then wait for 2ms.
- Check the lock status bit (LS, bit 7 of FSC).
  - If the bit is a “1”, go on.
  - If the bit is a “0”, wait 0.1ms longer and then re-check the bit.
- Enable the USB clock by setting USBC5 (bit 5 of USBC) to a “1”.
- Enable other blocks as necessary.

Note that the configuration changes described above may not need to be made if the MCU was not placed in a suspend state as described in section 4.6.2.3 "Set up after USB Suspend Detected".

## 4.7 *Ports*

After reset, Port 2 input voltage characteristics are set to  $0.5V_{CC}$  for  $V_{IH}$  and  $0.16V_{CC}$  for  $V_{IL}$ . To change the input voltage characteristics to CMOS levels, set bit 6 of the port control register (PTC) to a "1".



## 4.8 *Programming Notes*

Always execute an SEI instruction immediately before executing a PLP instruction.





**MITSUBISHI SEMICONDUCTOR  
AMERICA, INC.**

**PRELIMINARY**

# **Chapter 5**

## **SFR Register List**

5 Register List . . . . . 5-3







# 5 Register List

MSB 7	CPMA7	CPMA6	CPMA5	CPMA4	CPMA3	CPMA2	CPMA1	CPMA0	LSB 0	Address: 0000 <sub>16</sub> Access: R/W Reset: 0C <sub>16</sub>	
	CPMA0,1		Processor Mode Bits (bits 1,0)								
			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 <sub>out</sub> Drive Capacity Selection Bit (bit 3)								
			0: Low								
			1: High								
	CPMA4		Clock XC <sub>in</sub> -XC <sub>out</sub> Stop Bit (bit 4)								
			0: Stop								
			1: Oscillator								
	CPMA5		Clock X <sub>in</sub> -X <sub>out</sub> Stop Bit (bit 5)								
			0: Oscillator								
			1: Stop								
	CPMA6		Internal Clock Selection Bit (bit 6)								
			0: External Clock								
			1: f <sub>syn</sub>								
	CPMA7		External Clock Selection Bit (bit 7)								
			0: X <sub>in</sub> -X <sub>out</sub>								
			1: XC <sub>in</sub> -XC <sub>out</sub>								

**Figure 5-1. CPU Mode Register A**

MSB 7	CPMB7	Reserved	CPMB5	CPMB4	CPMB3	CPMB2	CPMB1	CPMB0	LSB 0	Address: 0001 <sub>16</sub> Access: R/W Reset: 83 <sub>16</sub>	
	CPMB0,1		Slow Memory Wait Bits (bits 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		Slow Memory Mode Bit (bits 3,2)								
			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: EDMA output disabled (64 Kbyte data access area)								
			1: 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 (Read/Write "0")								
	CPMB7		X <sub>out</sub> Drive Capacity Selection Bit (bit 7)								
			0: Low								
			1: High (default state after reset and after STOP mode)								

**Figure 5-2. CPU Mode Register B**

MSB 7	IRA7	IRA6	IRA5	IRA4	IRA3	IRA2	IRA1	IRA0	LSB 0	Address: 0002 <sub>16</sub>
	<p>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)</p> <p>0: No interrupt request issued          1: Interrupt request issued</p>									Access: R/W Reset: 00 <sub>16</sub>

Figure 5-3. IREQA Configuration

MSB 7	IRB7	IRB6	IRB5	IRB4	IRB3	IRB2	IRB1	IRB0	LSB 0	Address: 0003 <sub>16</sub>
	<p>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 Interrupt Request (bit 5)          IRB6 Timer 1 Interrupt Request (bit 6)          IRB7 Timer 2 Interrupt Request (bit 7)</p> <p>0: No interrupt request issued          1: Interrupt request issued</p>									Access: R/W Reset: 00 <sub>16</sub>

Figure 5-4. IREQB Configuration

MSB 7	Reserved	IRC6	IRC5	IRC4	IRC3	IRC2	IRC1	IRC0	LSB 0	Address: 0004 <sub>16</sub>
	<p>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 Interrupt Request (bit 5)          IRC6 Key-on Wake-up Interrupt Request (bit 6)</p> <p>0: No interrupt request issued          1: Interrupt request issued</p> <p>Bit 7 Reserved (Read/Write "0")</p>									Access: R/W Reset: 00 <sub>16</sub>

Figure 5-5. IREQC Configuration

MSB 7	ICA7	ICA6	ICA5	ICA4	ICA3	ICA2	ICA1	ICA0	LSB 0	Address: 0005 <sub>16</sub>
	<p>ICA0 USB Function Interrupt Enable (bit 0)          ICA1 USB SOF Interrupt Enable (bit 1)          ICA2 External Interrupt 0 Enable (bit 2)          ICA3 External Interrupt 1 Enable (bit 3)          ICA4 DMAC channel 0 Interrupt Enable (bit 4)          ICA5 DMAC channel 1 Interrupt Enable (bit 5)          ICA6 UART1 Receive Buffer Full Interrupt Enable (bit 6)          ICA7 UART1 Transmit Interrupt Enable (bit 7)</p> <p>0: Interrupt Disable          1: Interrupt Enable</p>									Access: R/W Reset: 00 <sub>16</sub>

Figure 5-6. ICONA Configuration



MSB 7	ICB7	ICB6	ICB5	ICB4	ICB3	ICB2	ICB1	ICB0	LSB 0	Address: 0006 <sub>16</sub>
	<p>ICC0      UART1 Error Sum Interrupt Enable (bit 0)          ICC1      UART2 Receive Buffer Full Interrupt Enable (bit 1)          ICC2      UART2 Transmit Interrupt Enable (bit 2)          ICC3      UART2 Error Sum Interrupt Enable (bit 3)          ICC4      Timer X Interrupt Enable (bit 4)          ICC5      Timer Y Interrupt Enable (bit 5)          ICC6      Timer 1 Interrupt Enable (bit 6)          ICC7      Timer 2 Interrupt Enable (bit 7)</p> <p>0: Interrupt Disable          1: Interrupt Enable</p>									Access: R/W Reset: 00 <sub>16</sub>

Figure 5-7. ICONB Configuration

MSB 7	Reserved	ICC6	ICC5	ICC4	ICC3	ICC2	ICC1	ICC0	LSB 0	Address: 0007 <sub>16</sub>
	<p>ICC0      Timer 3 Interrupt Enable (bit 0)          ICC1      External CNTR0 Interrupt Enable (bit 1)          ICC2      External CNTR1 Interrupt Enable (bit 2)          ICC3      SIO Interrupt Enable (bit 3)          ICC4      Input Buffer Full Interrupt Enable (bit 4)          ICC5      Output Buffer Empty Interrupt Enable (bit 5)          ICC6      Key-on Wake-up Interrupt Enable (bit 6)</p> <p>0: Interrupt disabled          1: Interrupt enabled</p> <p>Bit 7      Reserved (Read/Write "0")</p>									Access: R/W Reset: 00 <sub>16</sub>

Figure 5-8. ICONC Configuration

MSB 7	PTC7	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0	LSB 0	Address: 0010 <sub>16</sub>
	<p>PTC0      Slew Rate Control Bit Ports 0-3 (bit 0)                    0: Disabled                    1: Enabled          PTC1      Slew Rate Control Bit Port 4 (bit 1)                    0: Disabled                    1: Enabled          PTC2      Slew Rate Control Bit Port 5 (bit 2)                    0: Disabled                    1: Enabled          PTC3      Slew Rate Control Bit Port 6 (bit 3)                    0: Disabled                    1: Enabled          PTC4      Slew Rate Control Bit Port 7 (bit 4)                    0: Disabled                    1: Enabled          PTC5      Slew Rate Control Bit Port 8 (bit 5)                    0: Disabled                    1: Enabled          PTC6      Port 2 Input Level Select Bit (bit 6)                    0: Reduced VIH level input                    1: CMOS level input          PTC7      Master Bus Input Level Select Bit (bit 7)                    0: CMOS level input                    1: TTL level input</p>									Access: R/W Reset: 00 <sub>16</sub>

Figure 5-9. Port Control Register

MSB 7	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	INT1 Pol	INT0 Pol	LSB 0	Address: 0011 <sub>16</sub>	
											Access: R/W	
											Reset: 00 <sub>16</sub>	
			INT0 Pol	INT0 Interrupt Edge Selection Bit 0: Falling edge selected. 1: Rising edge selected.								
		INT1 Pol	INT1 Interrupt Edge Selection Bit 0: Falling edge selected. 1: Rising edge selected.									
		Bits 2-7	Reserved (Read/Write "0")									

Figure 5-10. IPOL Configuration

MSB 7	PUP2 <sub>7</sub>	PUP2 <sub>6</sub>	PUP2 <sub>5</sub>	PUP2 <sub>4</sub>	PUP2 <sub>3</sub>	PUP2 <sub>2</sub>	PUP2 <sub>1</sub>	PUP2 <sub>0</sub>	LSB 0	Address: 0012 <sub>16</sub>			
											Access: R/W		
											Reset: 00 <sub>16</sub>		
			PUP2 <sub>0</sub>	Pull-up Control for Port 2 (bit 0) 0: Disabled 1: Enabled									
			PUP2 <sub>1</sub>	Pull-up Control for Port 2 (bit 1) 0: Disabled 1: Enabled									
			PUP2 <sub>2</sub>	Pull-up Control for Port 2 (bit 2) 0: Disabled 1: Enabled									
			PUP2 <sub>3</sub>	Pull-up Control for Port 2 (bit 3) 0: Disabled 1: Enabled									
			PUP2 <sub>4</sub>	Pull-up Control for Port 2 (bit 4) 0: Disabled 1: Enabled									
			PUP2 <sub>5</sub>	Pull-up Control for Port 2 (bit 5) 0: Disabled 1: Enabled									
		PUP2 <sub>6</sub>	Pull-up Control for Port 2 (bit 6) 0: Disabled 1: Enabled										
		PUP2 <sub>7</sub>	Pull-up Control for Port 2 (bit 7) 0: Disabled 1: Enabled										

Figure 5-11. Pull-up Control Register

MSB 7	USBC7	USBC6	USBC5	USBC4	USBC3	Reserved	USBC1	Reserved	LSB 0	Address: 0013 <sub>16</sub>			
											Access: R/W		
											Reset: 00 <sub>16</sub>		
			Bit 0	Reserved (Read/Write "0")									
			USBC1	USB Default State Selection Bit (bit 1) 0: In default state after powerup/reset 1: In default state after received the USB reset signaling									
			Bit 2	Reserved (Read/Write "0")									
			USBC3	Transceiver Voltage Converter High/Low Current Mode Selection Bit (bit 3) 0: High current mode 1: Low current mode									
			USBC4	USB Transceiver Voltage Converter Enable Bit (bit 4) 0: USB transceiver voltage converter disabled 1: USB transceiver voltage converter enabled									
			USBC5	USB Clock Enable Bit (bit 5) 0: 48 MHz clock to the USB block is disabled. 1: 48 MHz clock to the USB block is enabled.									
		USBC6	USB $\overline{\text{SOF}}$ Port Select Bit (bit 6) 0: USB $\overline{\text{SOF}}$ output is disabled. P7 <sub>0</sub> is used as GPIO pin. 1: USB $\overline{\text{SOF}}$ output is enabled										
		USBC7	USB Enable Bit (bit 7) 0: USB block is disabled, all USB internal registers are held at their default values. 1: USB block is enabled										

Figure 5-12. USB Control Register



MSB 7	CCR7	CCR6	CCR5	CCR4	Reserved	Reserved	Reserved	Reserved	LSB 0	Address: 001F <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
	Bits 0-3		Reserved (Read/Write "0")								
	CCR4:		PLL Bypass Bit (bit 4) 0: $f_{USB} = f_{VCO}$ (Frequency synthesizer output) 1: $f_{USB} = f_{Xin}$								
	CCR5:		XC <sub>out</sub> Oscillation Drive Disable Bit (bit 5) 0: XC <sub>out</sub> oscillation drive is enabled (when XC <sub>in</sub> oscillation is enabled). 1: XC <sub>out</sub> oscillation drive is disabled.								
	CCR6:		X <sub>out</sub> Oscillation Drive Disable Bit (bit 6) 0: X <sub>out</sub> oscillation drive is enabled (when X <sub>in</sub> oscillation is enabled). 1: X <sub>out</sub> oscillation drive is disabled.								
	CCR7:		X <sub>in</sub> Divider Select Bit (bit 7) 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								

Figure 5-13. Clock Control Register

MSB 7	TXM7	TXM6	TXM5	TXM4	TXM3	TXM2	TXM1	TXM0	LSB 0	Address: 0027 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
	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 2,1) Bit 2      Bit 1 0      0: $\Phi$ divided by 8 0      1: $\Phi$ divided by 16 1      0: $\Phi$ divided by 32 1      1: $\Phi$ divided by 64								
	TXM3		Timer X Internal Clock Select (bit 3) 0: $\Phi/n$ 1: SCSGCLK (from chip special count source generator)								
	TXM5,4		Timer X Mode Bits (bits 5,4) Bit 5      Bit 4 0      0:    Timer Mode 0      1:    Pulse output mode 1      0:    Event counter mode 1      1:    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 5-14. TXM Register

MSB 7	TYM7	TYM6	TYM5	TYM4	TYM3	TYM2	TYM1	TYM0	LSB 0	Address: 0028 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
		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 3,2) Bit 2                  Bit 1 0                      0: $\Phi$ divided by 8 0                      1: $\Phi$ divided by 16 1                      0: $\Phi$ divided by 32 1                      1: $\Phi$ divided by 64								
		TYM5,4	Timer Y Mode Bits (bits 5,4) Bit 2                  Bit 1 0                      0: Timer mode 0                      1: Pulse period measurement mode 1                      0: Event counter mode 1                      1: 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 5-15. TYM Register

MSB 7	T123M7	T123M6	T123M5	T123M4	T123M3	T123M2	T123M1	T123M0	LSB 0	Address: 0029 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
		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: $\Phi$ divided by 8 1: XCin divided by 2								
		T123M3	Timer 2 Count Source Select Bit (bit 3) 0: Timer 1 underflow signal 1: $\Phi$								
		T123M4	Timer 3 Count Source Select Bit (bit 4) 0: Timer 1 underflow signal 1: $\Phi$ 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 5-16. T123M Register



MSB 7	OCHCont	SCSel	TDSel	RDYSel	PSel	ISCSel2	ISCSel1	ISCSel0	LSB 0	Address: 002B <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>																																													
ISCSel0-2 Internal Synchronization Clock Select Bits (bits 2,1,0)																																																							
<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;"></td> <td style="width: 15%; text-align: center;">Bit 2</td> <td style="width: 15%; text-align: center;">Bit 1</td> <td style="width: 15%; text-align: center;">Bit 0</td> <td style="width: 45%;"></td> </tr> <tr> <td></td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0:</td> <td>Internal Clock divided by 2.</td> </tr> <tr> <td></td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1:</td> <td>Internal Clock divided by 4.</td> </tr> <tr> <td></td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0:</td> <td>Internal Clock divided by 8.</td> </tr> <tr> <td></td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1:</td> <td>Internal Clock divided by 16.</td> </tr> <tr> <td></td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0:</td> <td>Internal Clock divided by 32.</td> </tr> <tr> <td></td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1:</td> <td>Internal Clock divided by 64.</td> </tr> <tr> <td></td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0:</td> <td>Internal Clock divided by 128.</td> </tr> <tr> <td></td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1:</td> <td>Internal Clock divided by 256.</td> </tr> </table>												Bit 2	Bit 1	Bit 0			0	0	0:	Internal Clock divided by 2.		0	0	1:	Internal Clock divided by 4.		0	1	0:	Internal Clock divided by 8.		0	1	1:	Internal Clock divided by 16.		1	0	0:	Internal Clock divided by 32.		1	0	1:	Internal Clock divided by 64.		1	1	0:	Internal Clock divided by 128.		1	1	1:	Internal Clock divided by 256.
	Bit 2	Bit 1	Bit 0																																																				
	0	0	0:	Internal Clock divided by 2.																																																			
	0	0	1:	Internal Clock divided by 4.																																																			
	0	1	0:	Internal Clock divided by 8.																																																			
	0	1	1:	Internal Clock divided by 16.																																																			
	1	0	0:	Internal Clock divided by 32.																																																			
	1	0	1:	Internal Clock divided by 64.																																																			
	1	1	0:	Internal Clock divided by 128.																																																			
	1	1	1:	Internal Clock divided by 256.																																																			
PSel SIO Port Selection Bit (bit 3)																																																							
0: I/O Port																																																							
1: TxD output, SCLK function																																																							
RDYSel $\overline{\text{SRDY}}$ Output Select Bit (bit 4)																																																							
0: I/O Port																																																							
1: $\overline{\text{SRDY}}$ signal																																																							
TDSel Transfer Direction Select Bit (bit 5)																																																							
0: LSB first																																																							
1: MSB first																																																							
SCSel Synchronization Clock Select Bit (bit 6)																																																							
0: External Clock																																																							
1: Internal Clock																																																							
OCHCont TxD Output Channel Control Bit (bit 7)																																																							
0: CMOS output																																																							
1: N-Channel open drain output																																																							

Figure 5-17. SIO Control Register 1

MSB 7	Reserved	Reserved	Reserved	Reserved	Reserved	RXDSEL	CLKSEL	SLAVE	LSB 0	Address: 002C <sub>16</sub> Access: R/W Reset: 18 <sub>16</sub>
SLAVE Slave Mode Selection Bit (bit 0)										
0: Normal mode										
1: Slave mode (to enter Slave mode, bit 4 of SIO Control register 1 also needs to be set)										
CLKSEL SIO Internal Clock Selection Bit (bit 1)										
0: $\Phi$										
1: SCSGCLK										
RXDSEL SRXD Input Selection Bit (bit 2)										
0: SRXD input disabled										
1: SRXD input enabled										
Bits 3-4 Reserved (Read/Write "1")										
Bits 5-7 Reserved (Read/Write "0")										

Figure 5-18. SIO Control Register 2

MSB 7	Reserved	Reserved	Reserved	Reserved	SCSGM3	SCSGM2	SCSGM1	SCSGM0	LSB 0	Address: 002F <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
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.										
Bits 4-7 Reserved (Read/Write "0")										

Figure 5-19. SCSGM Register







MSB 7	AME	RTS_SEL	CTS_SEL	TIS	RIN	TIN	REN	TEN	LSB 0	Address: 0033 <sub>16</sub> , 003B <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
	TEN	Transmission Enable Bit (bit 0) 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 (bit 1) 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 (bit 2) 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 (bit 3) 0: No action. 1: Clears the UART receive status flags 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 (bit 4) 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) Enable Bit (bit 5) 0: CTS function is disabled, P8 <sub>6</sub> (or P8 <sub>2</sub> ) is used as GPIO pin. 1: CTS function is enabled, P8 <sub>6</sub> (or P8 <sub>2</sub> ) is used as $\overline{\text{CTS}}$ input.								
	RTS_SEL	Request-to-Send (RTS) Enable Bit (bit 6) 0: RTS function is disabled, P8 <sub>7</sub> (or P8 <sub>3</sub> ) is used as GPIO pin. 1: RTS function is enabled, P8 <sub>7</sub> (or P8 <sub>3</sub> ) is used as $\overline{\text{RTS}}$ output.								
	AME	UART Address Mode Enable Bit (bit 7) 0: Address Mode disabled. 1: Address Mode enabled.								

Figure 5-22. UxCON Register

MSB 7	RTS3	RTS2	RTS1	RTS0	Reserved	Reserved	Reserved	Reserved	LSB 0	Address: 0036 <sub>16</sub> , 003E <sub>16</sub> Access: R/W Reset: 80 <sub>16</sub>
	Bits 0-3	Reserved (Read/Write "0")								
	RTS3:0	RTS Assertion Delay Count 3:0 (bits 7,6,5,4)								
		0000:	No delay, $\overline{\text{RTS}}$ asserts immediately after receive operation completes.							
		0001:	$\overline{\text{RTS}}$ asserts 8 bit-times after receive operation completes.							
		0010:	$\overline{\text{RTS}}$ asserts 16 bit-times after receive operation completes.							
		0011:	$\overline{\text{RTS}}$ asserts 24 bit-times after receive operation completes.							
		.	.							
		.	.							
		1000:	$\overline{\text{RTS}}$ asserts 64 bit-times after receive operation completes.							
		.	.							
		.	.							
		1110:	$\overline{\text{RTS}}$ asserts 112 bit-times after receive operation completes.							
		1111:	$\overline{\text{RTS}}$ asserts 120 bit-times after receive operation completes.							

Figure 5-23. UxRTSC Register



MSB 7	DCI	Reserved	DRLDD	DTSC	DISFI	D1UF	D0SFI	D0UF	LSB 0	Address: 003F <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
			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							
			DISFI	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 DCI			Reserved (Read/Write "0") 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 5-24. DMAIS Configuration

MSB 7	DxTMS	DxRLD	DxDAUE	DxDWC	DxDRCE	DxDRID	DxSRCE	DxSRID	LSB 0	Address: 0040 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
			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 "0") 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 5-25. DMAxM1 Configuration



MSB 7	DOCEN	D0CRR	D0UMIE	D0SWT	D0HRS3	D0HRS2	D0HRS1	D0HRS0	LSB 0	Address: 0041 <sub>16</sub>
	D0HRS3,2,1,0 DMAC Channel 0 Hardware Transfer Request Source Bits (bits 3, 2, 1, 0)								Access: R/W	
	0000: Disabled								Reset: 00 <sub>16</sub>	
	0001: UART1 receive interrupt									
	0010: UART1 transmit interrupt									
	0011: TimerY interrupt									
	0100: External Interrupt 0									
	0101: USB EndPoint 1 IN_PKT_RDY signal (falling edge active)									
	0110: USB EndPoint 2 IN_PKT_RDY signal (falling edge active)									
	0111: USB EndPoint 3 IN_PKT_RDY signal (falling edge active)									
	1000: USB EndPoint 1 OUT_PKT_RDY signal (rising edge active)									
	1001: USB EndPoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active)									
	1010: USB EndPoint 2 OUT_PKT_RDY signal (rising edge active)									
	1011: USB EndPoint 3 OUT_PKT_RDY signal (rising edge active)									
	1100: MBI OBE <sub>0</sub> signal (rising edge active)									
	1101: MBI IBF <sub>0</sub> (data) signal (rising edge active)									
	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 "0")								
		1: Writing "1" requests a channel 0 transfer								
	D0UMIE	DMAC Channel 0 USB and MBI 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 "0")								
		1: Setting to "1" causes reset of the channel 0 capture register								
	DOCEN	DMAC Channel 0 Enable Bit (bit 7)								
		0: Channel 0 disabled								
		1: Channel 0 enabled								

Figure 5-26. DMA0M2 Configuration

MSB 7	D1CEN	D1CRR	D1UMIE	D1SWT	D1HRS3	D1HRS2	D1HRS1	D1HRS0	LSB 0	Address: 0041 <sub>16</sub>
	D1HRS3,2,1,0 DMAC Channel 1 Hardware Transfer Request Source Bits (bits 3, 2, 1, 0)								Access: R/W	
	0000: Disabled								Reset: 00 <sub>16</sub>	
	0001: UART2 receive interrupt									
	0010: UART2 transmit interrupt									
	0011: TimerX interrupt									
	0100: External Interrupt 1									
	0101: USB EndPoint 1 IN_PKT_RDY signal (falling edge active)									
	0110: USB EndPoint 2 IN_PKT_RDY signal (falling edge active)									
	0111: USB EndPoint 4 IN_PKT_RDY signal (falling edge active)									
	1000: USB EndPoint 1 OUT_PKT_RDY signal (rising edge active)									
	1001: USB EndPoint 1 OUT_FIFO_NOT_EMPTY signal(rising edge active)									
	1010: USB EndPoint 2 OUT_PKT_RDY signal (rising edge active)									
	1011: USB EndPoint 4 OUT_PKT_RDY signal (rising edge active)									
	1100: MBI OBE <sub>1</sub> signal (rising edge active)									
	1101: MBI IBF <sub>1</sub> (data) signal (rising edge active)									
	1110: Timer1 interrupt									
	1111: CNTR0 interrupt									
	D1SWT	DMAC Channel 1 Software Transfer Trigger (bit 4)								
		0: No action (Bit is always read as "0")								
		1: Writing "1" requests a channel 0 transfer								
	D1UMIE	DMAC Channel 1 USB and MBI 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 "0")								
		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 5-27. DMA1M2 Configuration

MSB 7	DBBS07	DBBS06	DBB05	DBBS04	DBBS03	DBBS02	DBBS01	DBBS00	LSB 0	Address: 0049 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
		DBBS00	Output Buffer Full (OBF <sub>0</sub> ) Flag (bit 0) 0: Output buffer empty. 1: Output buffer full.								
		DBBS01	Input Buffer Full (IBF <sub>0</sub> ) Flag (bit 1) 0: Input buffer empty. 1: Input buffer full.								
		DBBS02	User Definable (U2) Flag (bit 2)								
		DBBS03	A <sub>0</sub> (A <sub>00</sub> ) Flag (bit 3) Indicates the A <sub>0</sub> status when IBF flag is set								
		DBBS04	User Definable (U4) Flag (bit 4)								
		DBBS05	User Definable (U5) Flag (bit 5)								
		DBBS06	User Definable (U6) Flag (bit 6)								
		DBBS07	User Definable (U7) Flag (bit 7)								

Figure 5-28. Data Bus Buffer Status Register 0

MSB 7	DBBC07	DBBC06	Reserved	DBBC04	DBBC03	DBBC02	DBBC01	DBBC00	LSB 0	Address: 004A <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
		DBBC00	OBF Output Selection Bit (bit 0) 0: P5 <sub>2</sub> pin is operated as GPIO 1: P5 <sub>2</sub> pin is operated as OBF <sub>0</sub> output pin								
		DBBC01	IBF Output Selection Bit (bit 1) 0: P5 <sub>3</sub> pin is operated as GPIO 1: P5 <sub>3</sub> pin is operated as IBF <sub>0</sub> output pin								
		DBBC02	IBF <sub>0</sub> Interrupt Selection Bit (bit 2) 0: IBF <sub>0</sub> interrupt is generated by both write-data (A <sub>0</sub> = "0") and write-command (A <sub>0</sub> = "1") 1: IBF <sub>0</sub> interrupt is generated by write-command (A <sub>0</sub> = "1") only								
		DBBC03	Output buffer 0 empty interrupt disable Bit (bit 3) 0: Enabled 1: Disabled								
		DBBC04	Input buffer 0 full interrupt disable Bit (bit 4) 0: Enabled 1: Disabled								
		DBBC05	Reserved (Read/Write "0")								
		DBBC06	Master CPU Bus Interface Enable Bit (bit 6) 0: P6 <sub>0</sub> -P6 <sub>7</sub> , P5 <sub>4</sub> -P5 <sub>7</sub> are GPIO pins 1: P6 <sub>0</sub> -P6 <sub>7</sub> , P5 <sub>4</sub> -P5 <sub>7</sub> are bus interface signals DQ0-DQ7, $\bar{S}_0$ , A <sub>0</sub> , $\bar{R}$ , $\bar{W}$ respectively.								
		DBBC07	Bus Interface Type Selection Bit (bit 7) 0: RD, WR separate type bus 1: R/W type bus.								

Figure 5-29. Data Bus Buffer Control Register 0

MSB 7	DBBS17	DBBS16	DBB15	DBBS14	DBBS13	DBBS12	DBBS11	DBBS10	LSB 0	Address: 004D <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
		DBBS10	Output Buffer Full (OBF <sub>1</sub> ) Flag (bit 0) 0: Output buffer empty. 1: Output buffer full.								
		DBBS11	Input Buffer Full (IBF <sub>1</sub> ) Flag (bit 1) 0: Input buffer empty. 1: Input buffer full.								
		DBBS12	User Definable (U2) Flag (bit 3)								
		DBBS13	A <sub>0</sub> (A <sub>01</sub> ) Flag (bit 2) Indicates the A <sub>0</sub> status when IBF flag is set								
		DBBS14	User Definable (U4) Flag (bit 4)								
		DBBS15	User Definable (U5) Flag (bit 5)								
		DBBS16	User Definable (U6) Flag (bit 6)								
		DBBS17	User Definable (U7) Flag (bit 7)								

Figure 5-30. Data Bus Buffer Status Register 1



MSB 7	DBBC17	Reserved	Reserved	DBBC14	DBBC13	DBBC12	DBBC11	DBBC10	LSB 0	Address: 004E <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
				DBBC10	OBF <sub>1</sub> Output Selection Bit (bit 0) 0: P7 <sub>4</sub> pin is operated as GPIO 1: P7 <sub>4</sub> pin is operated as OBF <sub>1</sub> output pin if DBBC17 = "1"					
				DBBC11	IBF <sub>1</sub> Output Selection Bit (bit 1) 0: P7 <sub>3</sub> pin is operated as GPIO 1: P7 <sub>3</sub> pin is operated as IBF <sub>1</sub> output pin if DBBC17 = "1"					
				DBBC12	IBF <sub>1</sub> Interrupt Selection Bit (bit 2) 0: IBF <sub>1</sub> interrupt is generated by both write-data (A <sub>0</sub> = "0") and write-command (A <sub>0</sub> = "1") 1: IBF <sub>1</sub> interrupt is generated by write-command (A <sub>0</sub> = "1") only					
				DBBC13	Output Buffer 1 Empty interrupt disable Bit (bit 3) 0: Enabled 1: Disabled					
				DBBC14	Input Buffer 1 Full interrupt disable Bit (bit 4) 0: Enabled 1: Disabled					
				DBBC15	Reserved (Read/Write "0")					
				DBBC16	Reserved (Read/Write "0")					
				DBBC17	Data Bus Buffer Function Selection Bit (bit 7) 0: Single data bus buffer - P7 <sub>2</sub> is used as GPIO 1: Double data bus buffer - P7 <sub>2</sub> is used as S <sub>1</sub> input					

Figure 5-31. Data Bus Buffer Control Register 1

MSB 7	Reserved	FUNAD6	FUNAD5	FUNAD4	FUNAD3	FUNAD2	FUNAD1	FUNAD0	LSB 0	Address: 0050 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>	
		FUNAD6:0 7-bit programmable Function Address (bits 6-0)									
		Bit 7	Reserved (Read/Write "0")								

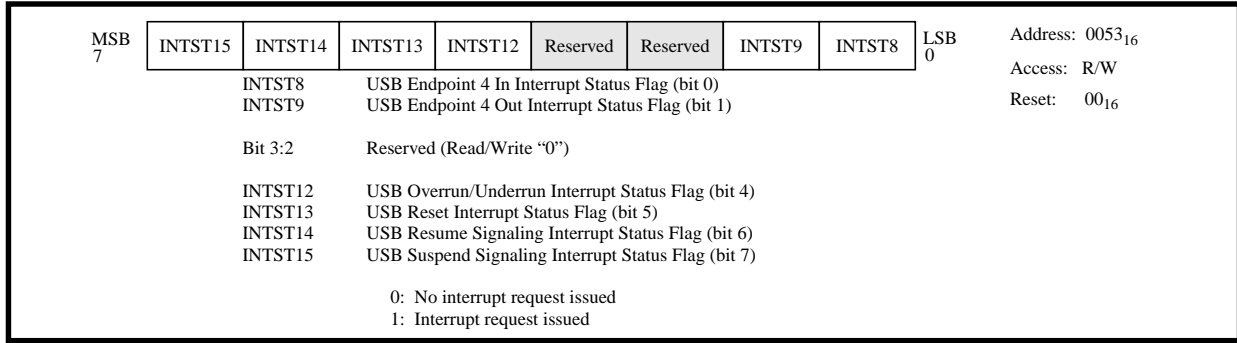
Figure 5-32. Function Address Register

MSB 7	Reserved	Reserved	Reserved	Reserved	Reserved	WAKEUP	RESUME	SUSPEND	LSB 0	Address: 0051 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
						SUSPEND	USB Suspend Detection Flag (bit 0) (Write "0" only or Read) 0: No USB suspend signal detected 1: USB suspend signal detected			
						RESUME	USB Resume Detection Flag (bit 1) (Write "0" only or Read) 0: No USB resume signal detected 1: USB resume signal detected			
						WAKEUP	USB Remote Wake-up Bit (bit 2) 0: End remote resume signaling 1: Remote resume signaling (If SUSPEND = "1")			
						Bit7:3	Reserved (Read/Write "0")			

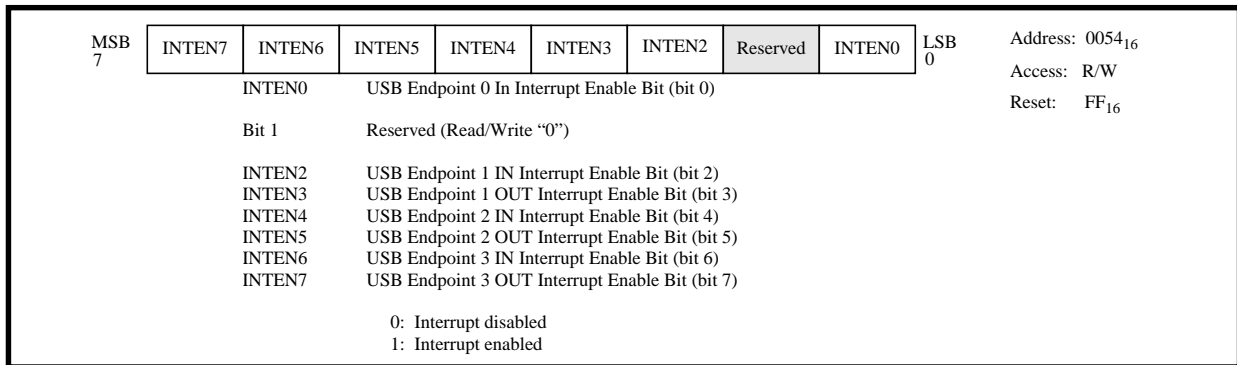
Figure 5-33. Power Management Register

MSB 7	INTST7	INTST6	INTST5	INTST4	INTST3	INTST2	Reserved	INTST0	LSB 0	Address: 0052 <sub>16</sub> Access: R/W Reset: 00 <sub>16</sub>
								INTST0	USB Endpoint 0 Interrupt Status Flag (bit 0)	
								Bit 1	Reserved (Read/Write "0")	
								INTST2	USB Endpoint 1 IN Interrupt Status Flag (bit 2)	
								INTST3	USB Endpoint 1 OUT Interrupt Status Flag (bit 3)	
								INTST4	USB Endpoint 2 IN Interrupt Status Flag (bit 4)	
								INTST5	USB Endpoint 2 OUT Interrupt Status Flag (bit 5)	
								INTST6	USB Endpoint 3 IN Interrupt Status Flag (bit 6)	
								INTST7	USB Endpoint 3 OUT Interrupt Status Flag (bit 7)	
									0: No interrupt request issued 1: Interrupt request issued	

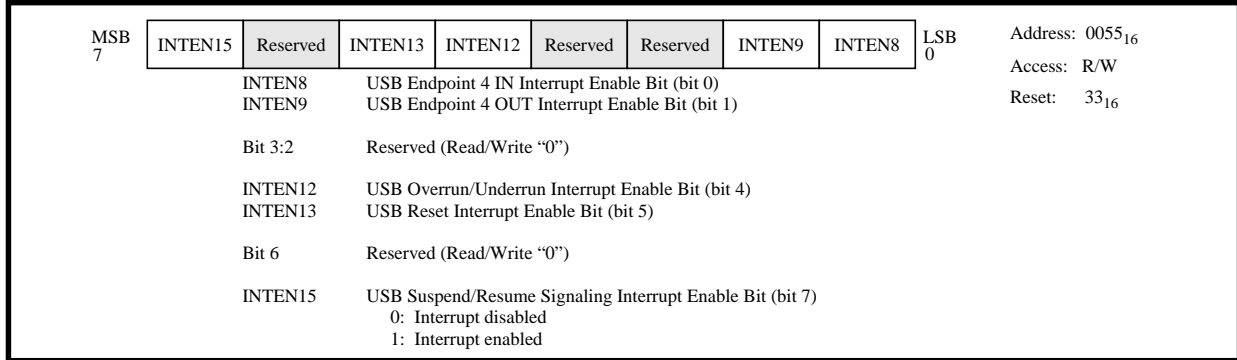
Figure 5-34. Interrupt Status Register 1



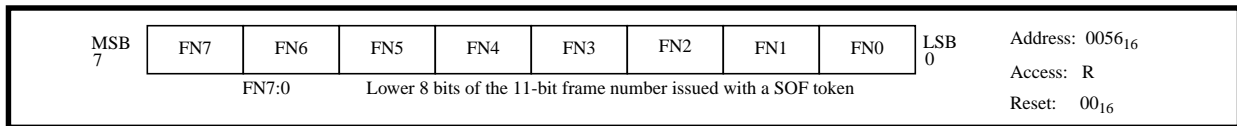
**Figure 5-35. Interrupt Status Register 2**



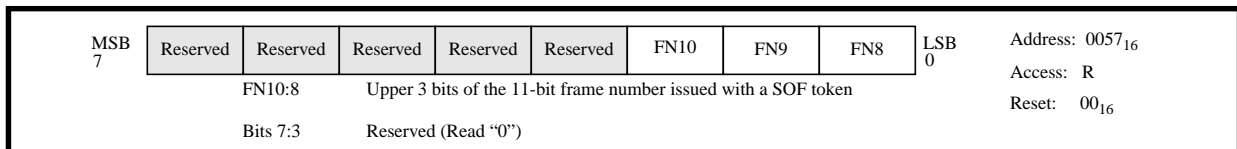
**Figure 5-36. Interrupt Enable Register 1**



**Figure 5-37. Interrupt Enable Register 2**



**Figure 5-38. Frame Number Register Low**



**Figure 5-39. Frame Number Register High**

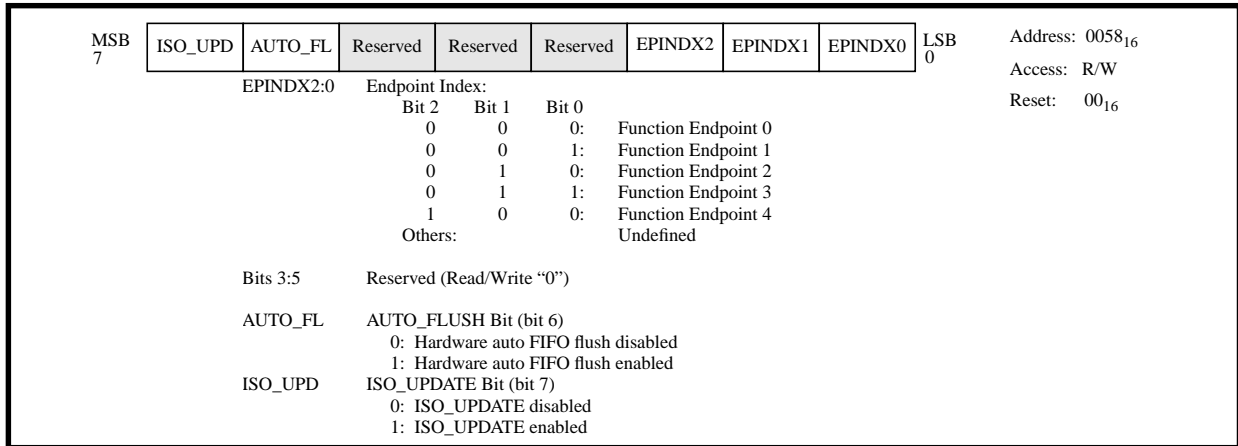


Figure 5-40. Endpoint Index Register

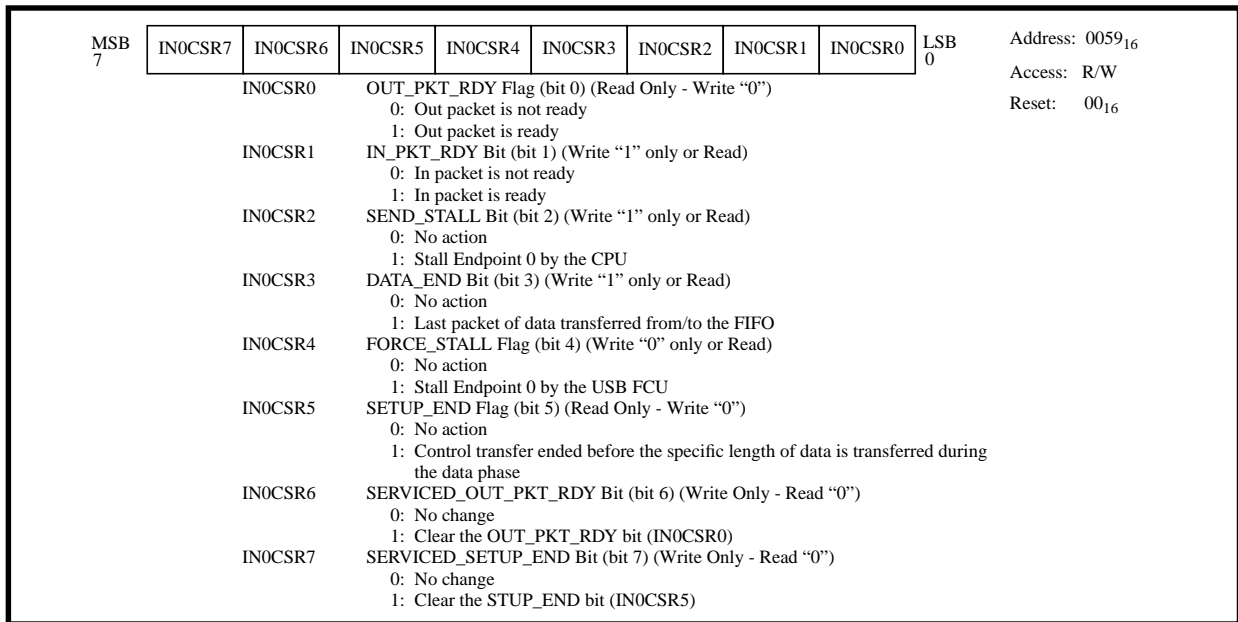


Figure 5-41. Endpoint 0 IN CSR

MSB 7	INXCSR7	INXCSR6	INXCSR5	INXCSR4	INXCSR3	INXCSR2	INXCSR1	INXCSR0	LSB 0	Address: 0059 <sub>16</sub>	
										Access: R/W	
										Reset: 00 <sub>16</sub>	
		INXCSR0	IN_PKT_RDY Bit (bit 0) (Write "1" only or Read)								
			0: In packet is not ready								
			1: In packet is ready								
		INXCSR1	UNDER_RUN Flag (bit 1) (Write "0" only or Read)								
			0: No FIFO underrun								
			1: FIFO underrun has occurred								
		INXCSR2	SEND_STALL Bit (bit 2)								
			0: No action								
			1: Stall IN Endpoint X by the CPU								
		INXCSR3	ISO Bit (bit 3)								
			0: Select non-isochronous transfer								
			1: Select isochronous transfer								
		INXCSR4	INTPT Bit (bit 4)								
			0: Select non-rate feedback interrupt transfer								
			1: Select rate feedback interrupt transfer								
		INXCSR5	TX_NOT_EPT Flag (bit 5) (Read Only - Write "0")								
			0: Transmit FIFO is empty								
			1: Transmit FIFO is not empty								
		INXCSR6	FLUSH Bit (bit 6) (Write Only - Read "0")								
			0: No action								
			1: Flush the FIFO								
		INXCSR7	AUTO_SET Bit (bit 7)								
			0: AUTO_SET disabled								
			1: AUTO_SET enabled								

Figure 5-42. Endpoints 1, 2, 3, 4 IN CSR

MSB 7	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	LSB 0	Address: 005A <sub>16</sub>
										Access: R
										Reset: 00 <sub>16</sub>
		Bits 7:0 Reserved (Read "0")								

Figure 5-43. Endpoint 0 OUT CSR

MSB 7	OUTXCSR7	OUTXCSR6	OUTXCSR5	OUTXCSR4	OUTXCSR3	OUTXCSR2	OUTXCSR1	OUTXCSR0	LSB 0	Address: 005A <sub>16</sub>	
										Access: R/W	
										Reset: 00 <sub>16</sub>	
		OUTXCSR0	OUT_PKT_RDY Flag (bit 0) (Write "0" only or Read)								
			0: Out packet is not ready								
			1: Out packet is ready								
		OUTXCSR1	OVER_RUN Flag (bit 1) (Write "0" only or Read)								
			0: No FIFO overrun								
			1: FIFO overrun occurred								
		OUTXCSR2	SEND_STALL Bit (bit 2)								
			0: No action								
			1: Stall OUT Endpoint X by the CPU								
		OUTXCSR3	ISO Bit (bit 3)								
			0: Select non-isochronous transfer								
			1: Select isochronous transfer								
		OUTXCSR4	FORCE_STALL Flag (bit 4) (Write "0" only or Read)								
			0: No action								
			1: Stall Endpoint X by the USB FCU								
		OUTXCSR5	DATA_ERR Flag (bit 5) (Write "0" only or Read)								
			0: No error								
			1: CRC or bit stuffing error received in an ISO packet								
		OUTXCSR6	FLUSH Bit (bit 6) (Write Only - Read "0")								
			0: No action								
			1: Flush the FIFO								
		OUTXCSR7	AUTO_CLR Bit (bit 7)								
			0: AUTO_CLR disabled								
			1: AUTO_CLR enabled								

Figure 5-44. Endpoint 1, 2, 3, 4 OUT CSR



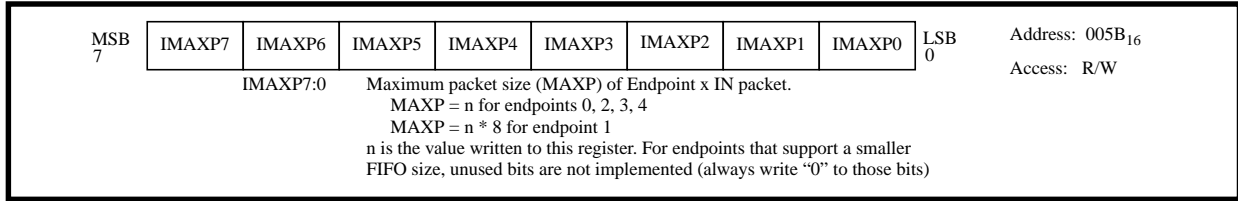


Figure 5-45. Endpoint x IN MAXP

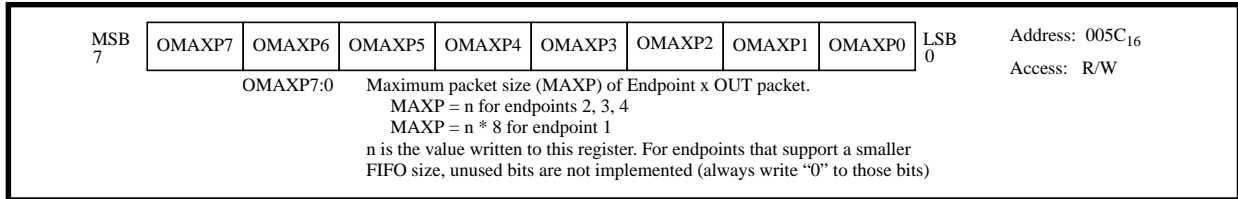


Figure 5-46. Endpoint x OUT MAXP

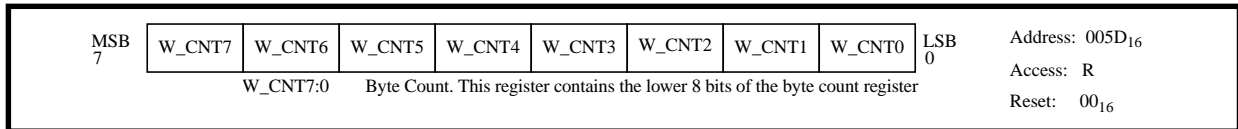


Figure 5-47. Endpoint 0, 1, 2, 3, 4 OUT Write Count Register Low

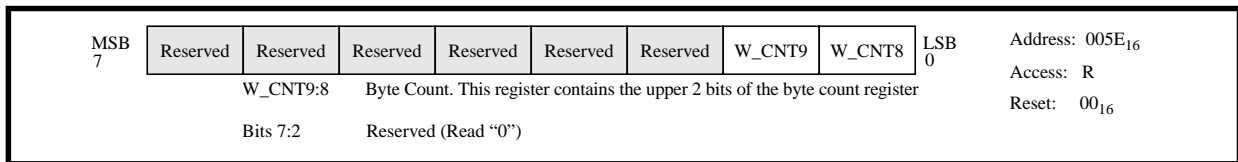


Figure 5-48. Endpoint 0, 1, 2, 3, 4 OUT Write Count Register High

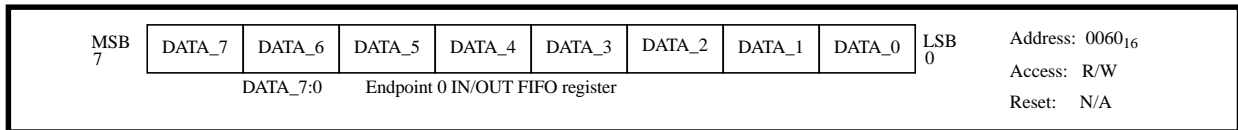


Figure 5-49. Endpoint 0 FIFO Register

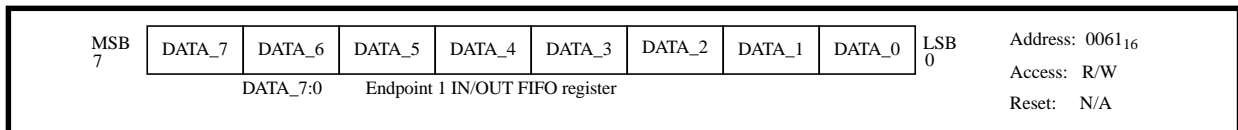


Figure 5-50. Endpoint 1 FIFO Register

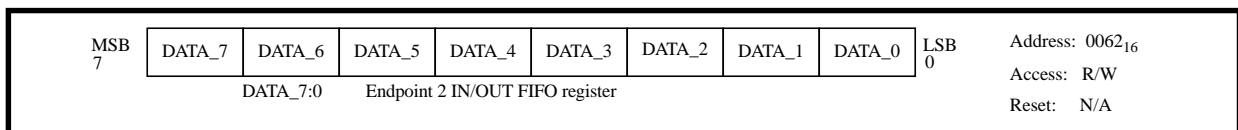


Figure 5-51. Endpoint 2 FIFO Register

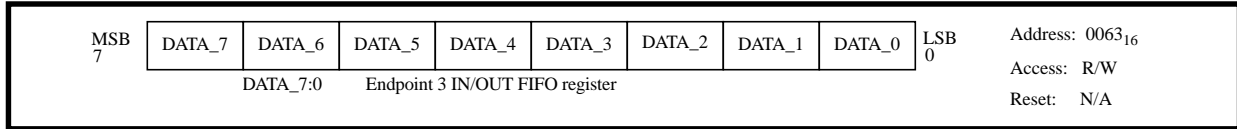


Figure 5-52. Endpoint 3 FIFO Register

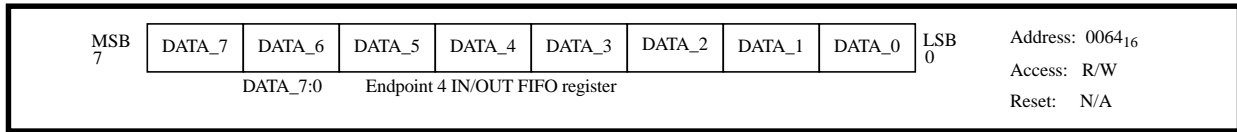


Figure 5-53. Endpoint 4 FIFO Register

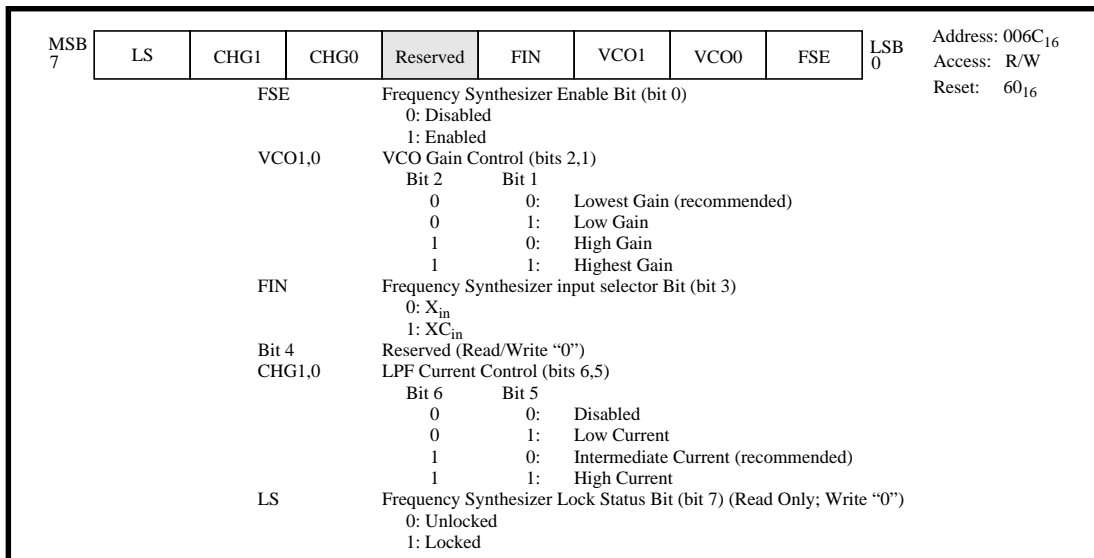


Figure 5-54. Frequency Synthesizer Control Register

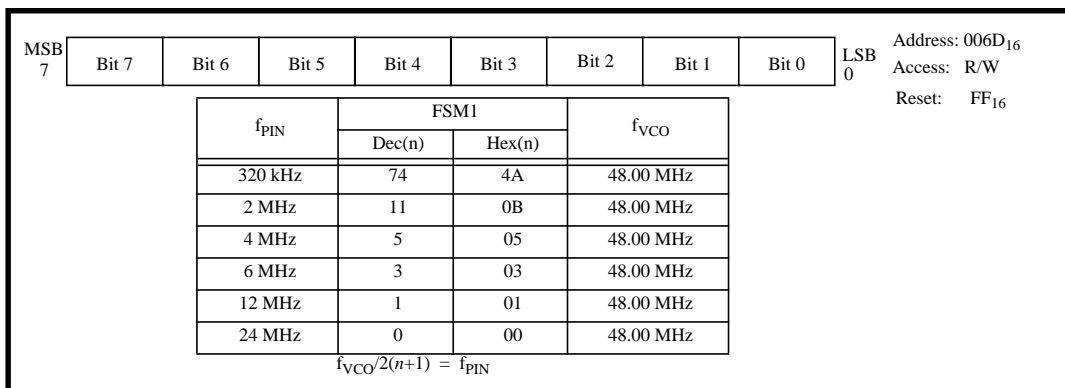


Figure 5-55. Frequency Synthesizer Multiply Control register FSM1



MSB 7	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	LSB 0	Address: 006E <sub>16</sub> Access: R/W Reset: FF <sub>16</sub>																											
	<table border="1"> <thead> <tr> <th rowspan="2">f<sub>PIN</sub></th> <th colspan="2">FSM2</th> <th rowspan="2">f<sub>IN</sub></th> </tr> <tr> <th>Dec(n)</th> <th>Hex(n)</th> </tr> </thead> <tbody> <tr> <td>24 MHz</td> <td>255</td> <td>FF</td> <td>24.00 MHz</td> </tr> <tr> <td>1 MHz</td> <td>11</td> <td>0C</td> <td>24.00 MHz</td> </tr> <tr> <td>2 MHz</td> <td>5</td> <td>05</td> <td>24.00 MHz</td> </tr> <tr> <td>3 MHz</td> <td>3</td> <td>03</td> <td>24.00 MHz</td> </tr> <tr> <td>6 MHz</td> <td>1</td> <td>01</td> <td>24.00 MHz</td> </tr> <tr> <td>12 MHz</td> <td>0</td> <td>00</td> <td>24.00 MHz</td> </tr> </tbody> </table>										f <sub>PIN</sub>	FSM2		f <sub>IN</sub>	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
f <sub>PIN</sub>	FSM2		f <sub>IN</sub>																																		
	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 5-56. 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: 006F <sub>16</sub> Access: R/W Reset: FF <sub>16</sub>											
	<table border="1"> <thead> <tr> <th rowspan="2">f<sub>VCO</sub></th> <th colspan="2">FSD</th> <th rowspan="2">f<sub>SYN</sub></th> </tr> <tr> <th>Dec(m)</th> <th>Hex(m)</th> </tr> </thead> <tbody> <tr> <td>48.00 MHz</td> <td>00</td> <td>00</td> <td>24.00 MHz</td> </tr> <tr> <td>48.00 MHz</td> <td>127</td> <td>7F</td> <td>187.50 KHz</td> </tr> </tbody> </table>										f <sub>VCO</sub>	FSD		f <sub>SYN</sub>	Dec(m)	Hex(m)	48.00 MHz	00	00	24.00 MHz	48.00 MHz
f <sub>VCO</sub>	FSD		f <sub>SYN</sub>																		
	Dec(m)	Hex(m)																			
48.00 MHz	00	00	24.00 MHz																		
48.00 MHz	127	7F	187.50 KHz																		

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

Figure 5-57. Frequency Synthesizer Divide Register

