MC68HC812A4

Data Sheet

M68HC12
Microcontrollers

MC68HC812A4
Rev. 7
05/2006

freescale.com

freescale™
semiconductor

MC68HC812A4

Data Sheet

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

http://freescale.com

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
August, 2001 (Continued on next page)	4	**Figure 1-3. Expanded Wide Mode SRAM Expansion Schematic** — Figure title changed from FLASH EEPROM to SRAM and address line designators corrected	40
		Figure 1-4. Expanded Narrow Mode SRAM Expansion Schematic — Figure title changed from FLASH EEPROM to SRAM and address line designators corrected	42
		Figure 8-16. Chip-Select Control Register 0 (CSCTL0) — Corrected reset value for CSPOE (bit 5)	138
		Figure 10-1. Clock Module Block Diagram — Corrected E- and P-clock generator options	156
		Figure 11-1. PLL Block Diagram — Revised diagram to show correct placement of divide-by-two block	170
		12.11.2 Timer Port Data Direction Register — Descriptive paragraph added for clarity	209

Revision History

Date	Revision Level	Description	Page Number(s)
August, 2001 (Continued)	4	**12.11.3 Data Direction Register for Timer Port** — Repetitive information removed. See **12.11.2 Timer Port Data Direction Register**	209
		18.12 Control Timing — Minimum values added for PW_{IRQ} and PW_{TIM}	329
		18.14 Non-Multiplexed Expansion Bus Timing — Table heading changed to reflect minimum and maximum values at 8 MHz	334
September, 2001	5	**Table 12-3. Prescaler Selection** — Added value column and updated prescale factors	197
		18.11 EEPROM Characteristics — Corrected minimum and maximum values for programming and erase times	328
August, 2002	6	**Figure 1-3. Expanded Wide Mode SRAM Expansion Schematic** — On sheet 1 of this schematic removed reference to resistor R2	40
		Figure 1-4. Expanded Narrow Mode SRAM Expansion Schematic — On sheet 1 of this schematic removed reference to resistor R2	42
		4.6.2 External Reset — Corrected reference to eight E-clock cycles to nine E-clock cycles	77
May, 2006	7	Updated to meet Freescale identity guidelines.	Throughout
		1.3 Ordering Information — Updated Table 1-1. Ordering Information and added Figure 1-1. Device Numbering System.	18
		Figure 1-4. Expanded Wide Mode SRAM Expansion Schematic (Sheet 1 of 3) — Updated sheet 1 and corrected title for sheets 2 and 3.	24
		Figure 1-5. Expanded Narrow Mode SRAM Expansion Schematic (Sheet 1 of 3) — Updated sheet 1 and corrected title for sheets 2 and 3.	26
		Figure 3-9. Condition Code Register (CCR) — Corrected reset state for bit 7.	46
		Table 4-1. Interrupt Vector Map — Corrected reference to clock monitor reset.	50
		4.5 Resets — Reworked paragraph for clarity.	52
		Figure 5-1. Mode Register (MODE) — Changed reset state designator from Peripheral to Special peripheral.	58
		Figure 10-3. Clock Function Register Map — Removed reference to Special Reset for the COP Control Register.	102
		Figure 10-9. COP Control Register (COPCTL) — Corrected reset states.	107
		12.4.1 Prescaler — Corrected number of prescaler divides.	122
		Figure 12-17. Timer Mask 2 Register (TMSK2) — Corrected reset state for bit 4.	131
		Table 16-5. ATD Interrupt Sources — Corrected table title.	207
		18.2 Functional Operating Range — Corrected operating temperature range entries.	222
		18.10 EEPROM Characteristics — Corrected minimum value for minimum programming clock frequency.	226
		18.11 Control Timing — Corrected maximum value for frequency of operation.	227
		18.12 Peripheral Port Timing — Corrected table heading.	231
		19.2 Package Dimensions — Replaced package dimension drawing with the latest available.	237

List of Chapters

Table of Contents

Chapter 5
Operating Modes and Resource Mapping

Chapter 6
Bus Control and Input/Output (I/O)

Chapter 7
EEPROM

Chapter 8
Memory Expansion and Chip-Select

Chapter 9
Key Wakeups

Chapter 10
Clock Module

Chapter 11
Phase-Lock Loop (PLL)

Chapter 12
Standard Timer Module

Chapter 13
Multiple Serial Interface (MSI)

Chapter 14
Serial Communications Interface Module (SCI)

Chapter 15
Serial Peripheral Interface (SPI)

Chapter 16
Analog-to-Digital Converter (ATD)

Chapter 17
Development Support

Chapter 18
Electrical Characteristics

Chapter 19
Mechanical Specifications

Chapter 1
General Description

1.1 Introduction

The MC68HC812A4 microcontroller unit (MCU) is a 16-bit device composed of standard on-chip peripheral modules connected by an intermodule bus. Modules include:

- 16-bit central processor unit (CPU12)
- Lite integration module (LIM)
- Two asynchronous serial communications interfaces (SCI0 and SCI1)
- Serial peripheral interface (SPI)
- Timer and pulse accumulator module
- 8-bit analog-to-digital converter (ATD)
- 1-Kbyte random-access memory (RAM)
- 4-Kbyte electrically erasable, programmable read-only memory (EEPROM)
- Memory expansion logic with chip selects, key wakeup ports, and a phase-locked loop (PLL)

1.2 Features

Features of the MC68HC812A4 include:

- Low-power, high-speed M68HC12 CPU
- Power-saving stop and wait modes
- Memory:
  - 1024-byte RAM
  - 4096-byte EEPROM
  - On-chip memory mapping allows expansion to more than 5-Mbyte address space
- Single-wire background debug mode
- Non-multiplexed address and data buses
- Seven programmable chip-selects with clock stretching (expanded modes)
- 8-channel, enhanced 16-bit timer with programmable prescaler:
  - All channels configurable as input capture or output compare
  - Flexible choice of clock source
- 16-bit pulse accumulator
- Real-time interrupt circuit
- Computer operating properly (COP) watchdog
- Clock monitor
- Phase-locked loop (PLL)

- Two enhanced asynchronous non-return-to-zero (NRZ) serial communication interfaces (SCI)
- Enhanced synchronous serial peripheral interface (SPI)
- 8-channel, 8-bit analog-to-digital converter (ATD)
- Up to 24 key wakeup lines with interrupt capability
- Available in 112-lead low-profile quad flat pack (LQFP) packaging

1.3 Ordering Information

The MC68HC812A4 is available in 112-lead low-profile quad flat pack (LQFP) packaging.

Operating temperature range and voltage requirements are specified when ordering the MC68HC812A4 device. Refer to Table 1-1 for part numbers and to Figure 1-1 for details of the device numbering system.

Table 1-1. Ordering Information

Order Number	Temperature		Voltage	Frequency (MHz)
	Range	Designator		
MC68HC812A4CPV8	–40 to +85°C	C	5.0	8.0
XC68HC812A4PV5	0 to +70°C	—	3.3	5.0

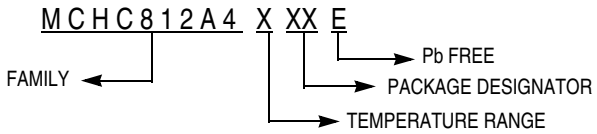


Figure 1-1. Device Numbering System

Evaluation boards, assemblers, compilers, and debuggers are available from Freescale and from third-party suppliers. An up-to-date list of products that support the M68HC12 Family of microcontrollers can be found on the World Wide Web at this URL:

http://freescale.com

Documents to assist in product selection are available from the Freescale Literature Distribution Center or local Freescale sales offices.

1.4 Block Diagram

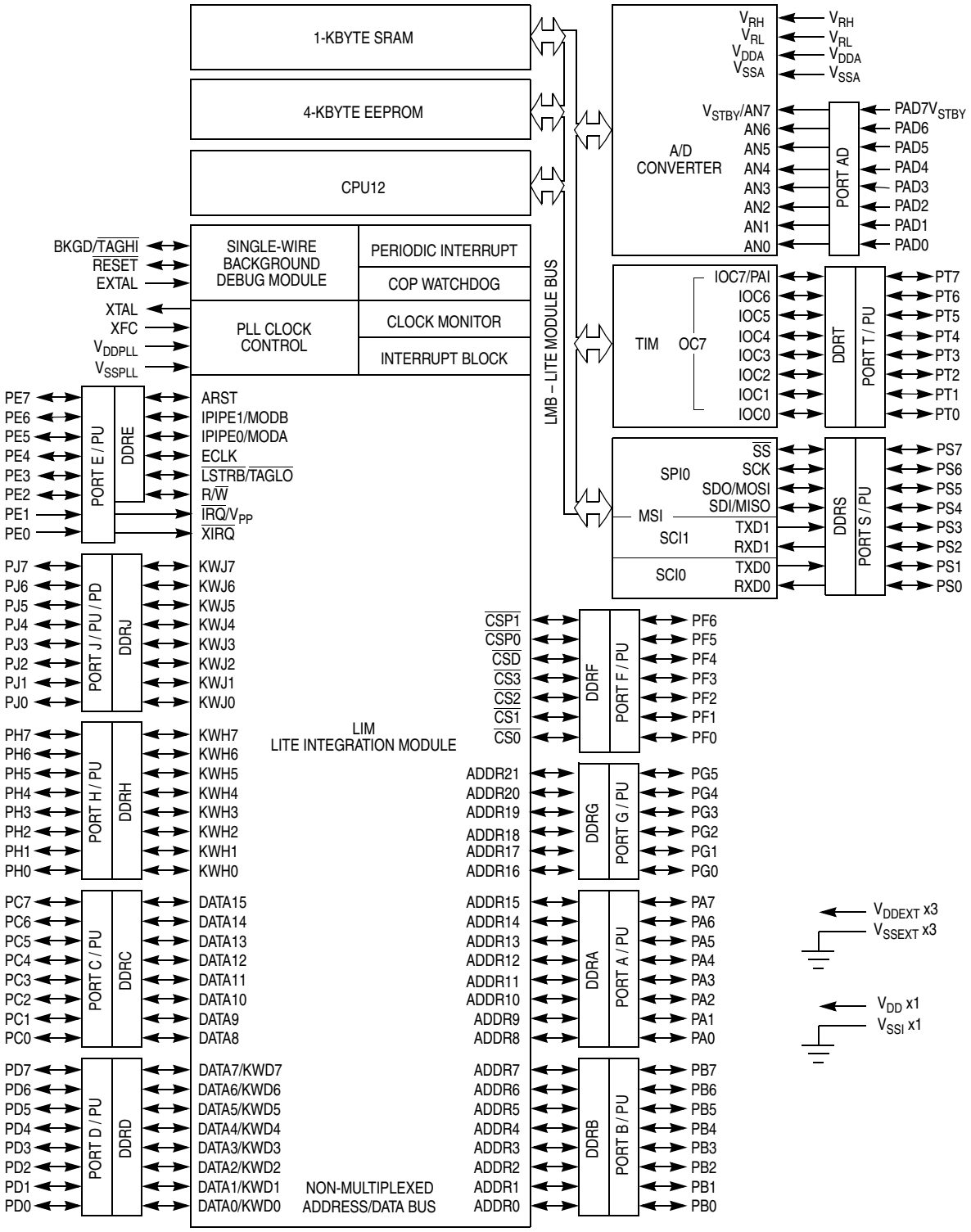


Figure 1-2. Block Diagram

1.5 Signal Descriptions

NOTE

A line over a signal name indicates an active low signal. For example, RESET is active high and $\overline{RESET}$ is active low.

The MC68HC812A4 is available in a 112-lead low-profile quad flat pack (LQFP). The pin assignments are shown in Figure 1-3. Most pins perform two or more functions, as described in Table 1-2. Individual ports are cross referenced in Table 1-3 and Table 1-4.

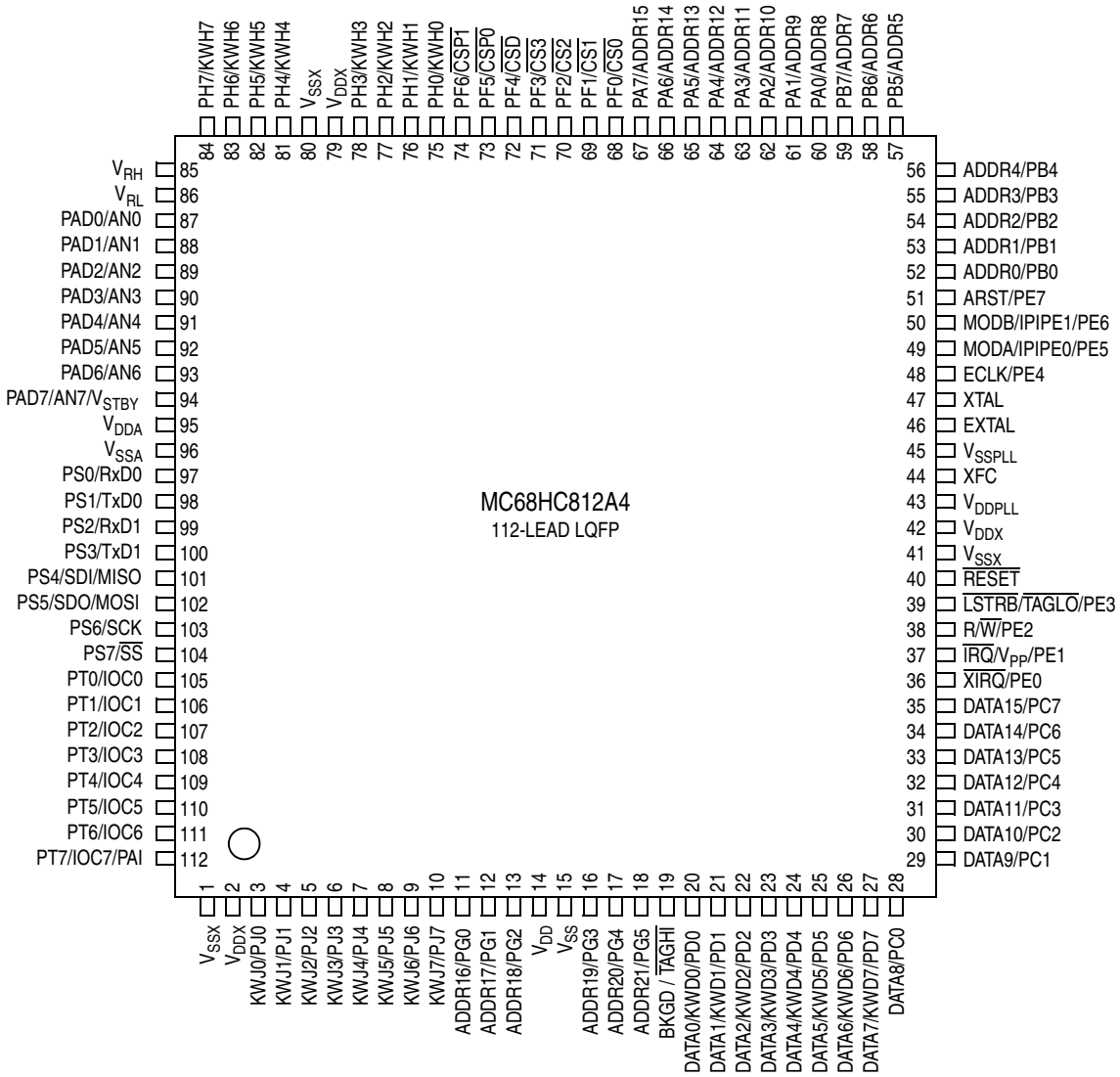


Figure 1-3. Pin Assignments

Table 1-2. Pin Descriptions

Pin	Port	Description
V_{DD}, V_{SS}	—	Operating voltage and ground for the MCU(1)
V_{RH}, V_{RL}	—	Reference voltages for the ADC
AV_{DD}, AV_{SS}	—	Operating voltage and ground for the ADC(2)
V_{DDPLL}, V_{SSPLL}	—	Power and ground for PLL clock control
V_{STBY}	Port AD	RAM standby power input
XTAL, EXTAL	—	Input pins for either a crystal or a CMOS compatible clock(3)
$\overline{XIRQ}$	PE0	Asynchronous, non-maskable external interrupt request input
$\overline{IRQ}$	PE1	Asynchronous, maskable external interrupt request input with selectable falling-edge triggering or low-level triggering
R/$\overline{W}$	PE2	Expansion bus data direction indicator General-purpose I/O; read/write in expanded modes
$\overline{LSTRB}$	PE3	Low byte strobe (0 = low byte valid)(4) General-purpose I/O
ECLK	PE4	Timing reference output for external bus clock (normally, half the crystal frequency) General-purpose I/O
BKGD	—	Mode-select pin determines initial operating mode of the MCU after reset
MODA	PE5	Mode-select input determines initial operating mode of the MCU after reset(5)
MODB	PE6	Mode-select input determines initial operating mode of the MCU after reset(5)
IPIPE0	PE5	Instruction queue tracking signals for development systems
IPIPE1	PE6	
ARST	PE7	Alternate active-high reset input General-purpose I/O
XFC	—	Loop filter pin for controlled damping of PLL VCO loop
$\overline{RESET}$	—	Active-low bidirectional control signal; input initializes MCU to known startup state; output when COP or clock monitor causes a reset
ADDR15–ADDR8	Port A	Single-chip modes: general-purpose I/O Expanded modes: external bus pins Port D in narrow data bus mode: general-purpose I/O or key wakeup port
ADDR7–ADDR0	Port B	
DATA15–DATA8	Port C	
DATA7–DATA0	Port D	
ADDR21–ADDR16	Port G	Memory expansion and general-purpose I/O
$\overline{CS3}$–$\overline{CS0}$,$\overline{CSD}$, $\overline{CSP1}$, $\overline{CSP0}$	Port F	Chip selects General-purpose I/O
BKGD	—	Single-wire background debug pin Mode-select pin that determines special or normal operating mode after reset
KWD7–KWD0	Port D	Key wakeup pins that can generate interrupt requests on high-to-low transitions General-purpose I/O
KWH7–KWH0	Port H	
KWJ7–KWJ0	Port J	Key wakeup pins that can generate interrupt requests on any transition General-purpose I/O
RxD0	PS0	Receive pin for SCI0
TxD0	PS1	Transmit pin for SCI0

Table 1-2. Pin Descriptions (Continued)

Pin	Port	Description
RxD1	PS2	Receive pin for SCI1
TxD1	PS3	Transmit pin for SCI1
SDI/MISO	PS4	Master in/slave out pin for SPI
SDO/MOSI	PS5	Master out/slave in pin for SPI
SCK	PS6	Serial clock for SPI
$\overline{SS}$	PS7	Slave select output for SPI in master mode; slave select input in slave mode
IOC7–IOC0	Port T	Input capture or output compare channels and pulse accumulator input

1. The MCU operates from a single power supply. Use the customary bypass techniques as very fast signal transitions occur on MCU pins.
2. Separate power supply pins allow the ADC power supply to be bypassed independently of the MCU power supply.
3. Out of reset the frequency applied to EXTAL is twice the desired E-clock rate. On reset all device clocks are derived from the EXTAL input frequency. XTAL is the crystal output.
4. $\overline{LSTRB}$ is the exclusive-NOR of A0 and the internal $\overline{SZ8}$ signal. $\overline{SZ8}$ indicates the size 16/8 access.
5. After reset, MODA and MODB can be configured as instruction queue tracking signals IPIPE0 and IPIPE1 or as general-purpose I/O pins.

Table 1-3. Port Descriptions

Port	Direction	Function
Port A	I/O	Single-chip modes: general-purpose I/O Expanded modes: external address bus ADDR15–ADDR8
Port B	I/O	Single-chip modes: general-purpose I/O Expanded modes: external address bus ADDR7–ADDR0
Port C	I/O	Single-chip modes: general-purpose I/O Expanded wide modes: external data bus DATA15–DATA8 Expanded narrow modes: external data bus DATA15–DATA8/DATA7–DATA0
Port D	I/O	Single-chip and expanded narrow modes: general-purpose I/O External data bus DATA7–DATA0 in expanded wide mode(1)
Port E	I/O and I(2)	External interrupt request inputs, mode select inputs, bus control signals General-purpose I/O
Port F	I/O	Chip select General-purpose I/O
Port G	I/O	Memory expansion General-purpose I/O
Port H	I/O	Key wakeup(3) General-purpose I/O
Port J	I/O	Key wakeup(4) General-purpose I/O
Port S	I/O	SCI and SPI ports General-purpose I/O
Port T	I/O	Timer port General-purpose I/O
Port AD	I	ADC port General-purpose input

1. Key wakeup interrupt request can occur when an input goes from high to low.
2. PE1 and PE0 are input-only pins.
3. Key wakeup interrupt request can occur when an input goes from high to low.
4. Key wakeup interrupt request can occur when an input goes from high to low or from low to high.

Table 1-4. Port Pullup, Pulldown, and Reduced Drive Summary

Port Name	Resistive Input Loads	Enable Bit			Reduced Drive Control Bit		
		Register (Address)	Bit Name	Reset State	Register (Address)	Bit Name	Reset State
Port A	Pullup	PUCR ($000C)	PUPA	Enabled	RDRIV ($000D)	RDPAB	Full drive
Port B	Pullup	PUCR ($000C)	PUPB	Enabled	RDRIV ($000D)	RDPAB	Full drive
Port C	Pullup	PUCR ($000C)	PUPC	Enabled	RDRIV ($000D)	RDPC	Full drive
Port D	Pullup	PUCR ($000C)	PUPD	Enabled	RDRIV ($000D)	RDPD	Full drive
Port E: PE7, PE3, PE2, PE0	Pullup	PUCR ($000C)	PUPE	Enabled	RDRIV ($000D)	RDPE	Full drive
Port E: PE1	Pullup	Always enabled			RDRIV ($000D)	RDPE	Full drive
Port E: PE4	None	—			RDRIV ($000D)	RDPE	Full drive
Port E: PE6 and PE5	Pulldown	Enabled during reset			—	—	—
Port F	Pullup	PUCR ($000C)	PUPF	Enabled	RDRIV ($000D)	RDPF	Full drive
Port G	Pullup	PUCR ($000C)	PUPG	Enabled	RDRIV ($000D)	RDPG	Full drive
Port H	Pullup	PUCR ($000C)	PUPH	Enabled	RDRIV ($000D)	RDPH	Full drive
Port J	Pullup/down(1)	PULEJ ($002E)	PULEJ[7:0]	Disabled	RDRIV ($000D)	RDPJ	Full drive
Port S	Pullup	SP0CR2 ($00D1)	PUPS	Enabled	SP0CR2 ($00D1)	RDS	Full drive
Port T	Pullup	TMSK2 ($008D)	PUPT	Enabled	TMSK2 ($008D)	RDPT	Full drive
Port AD	None	—			—		
BKGD	Pullup	—	—	Enabled	—	—	Full drive

1. Pullup or pulldown devices for each port J pin can be selected with the PUPSJ register ($002D). After reset, pulldowns are selected for all port J pins but must be enabled with PULEJ register.

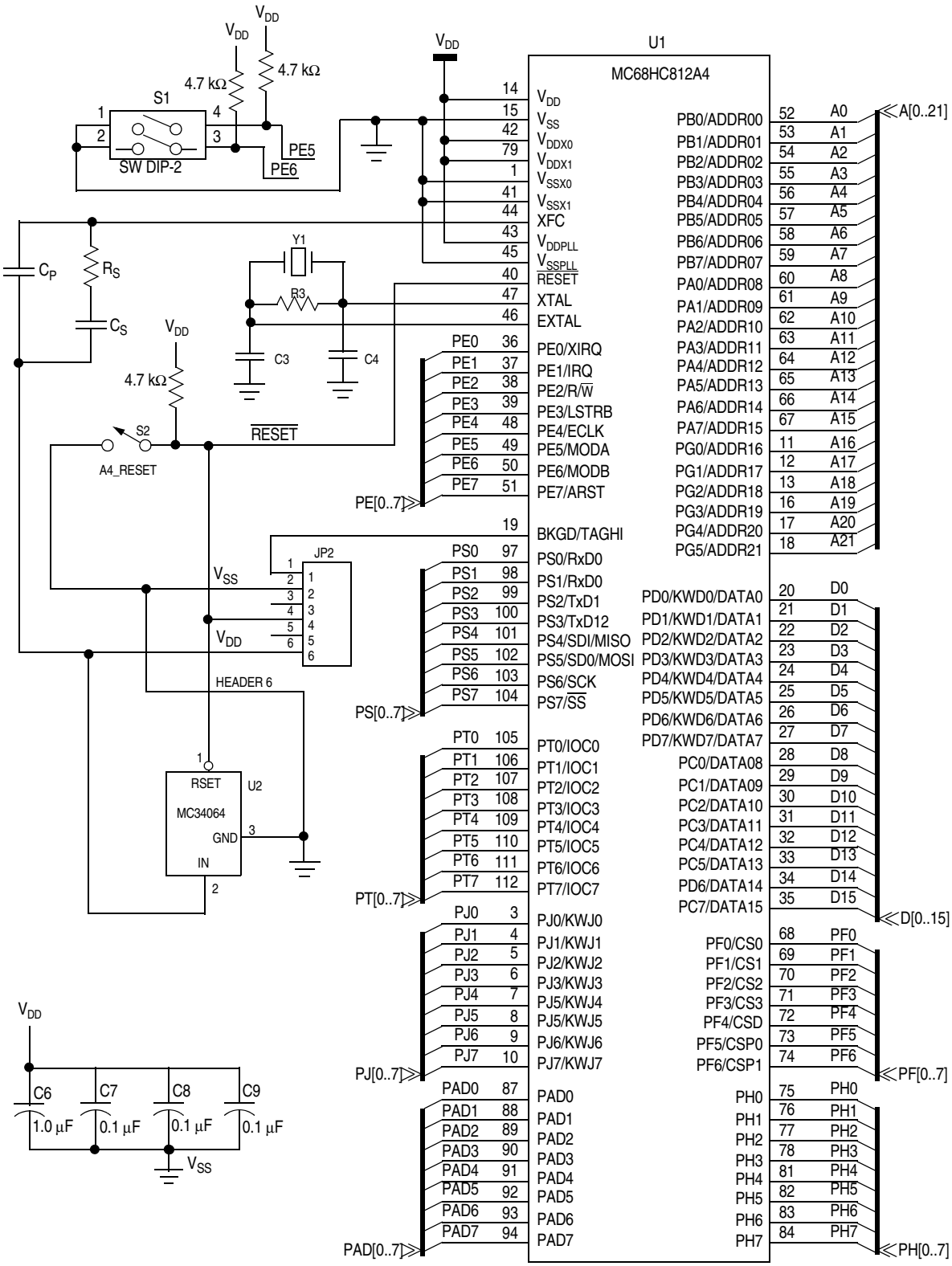


Figure 1-4. Expanded Wide Mode SRAM Expansion Schematic (Sheet 1 of 3)

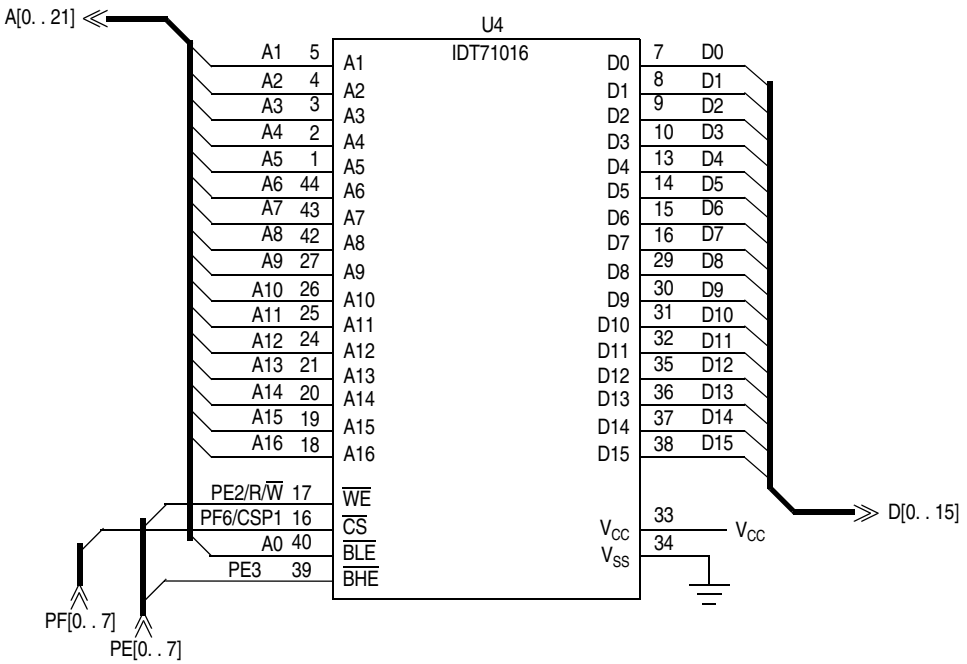


Figure 1-4. Expanded Wide Mode SRAM Expansion Schematic (Sheet 2 of 3)

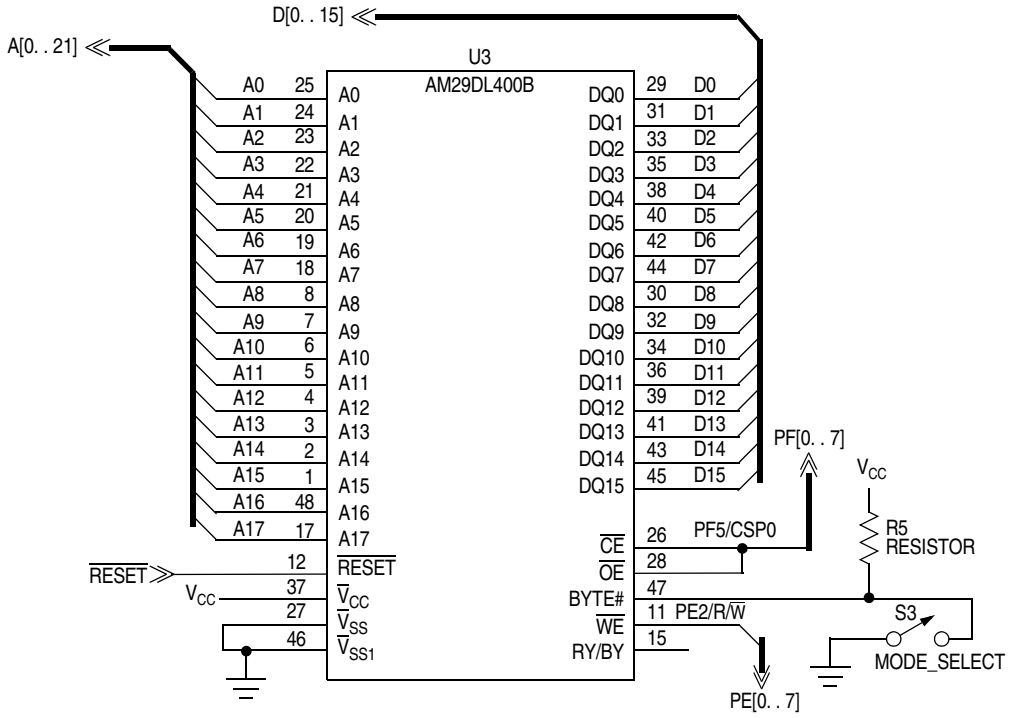


Figure 1-4. Expanded Wide Mode SRAM Expansion Schematic (Sheet 3 of 3)

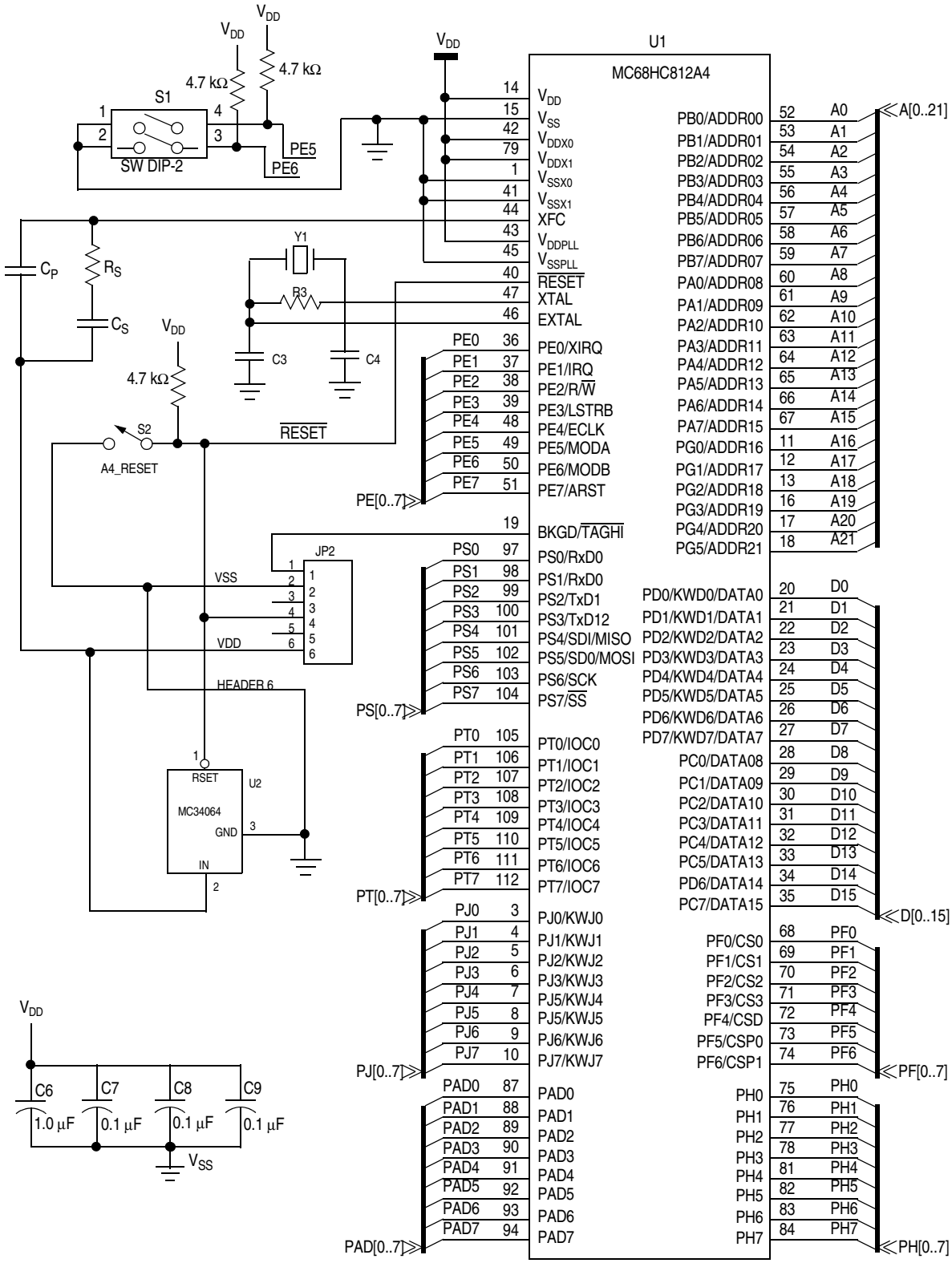


Figure 1-5. Expanded Narrow Mode SRAM Expansion Schematic (Sheet 1 of 3)

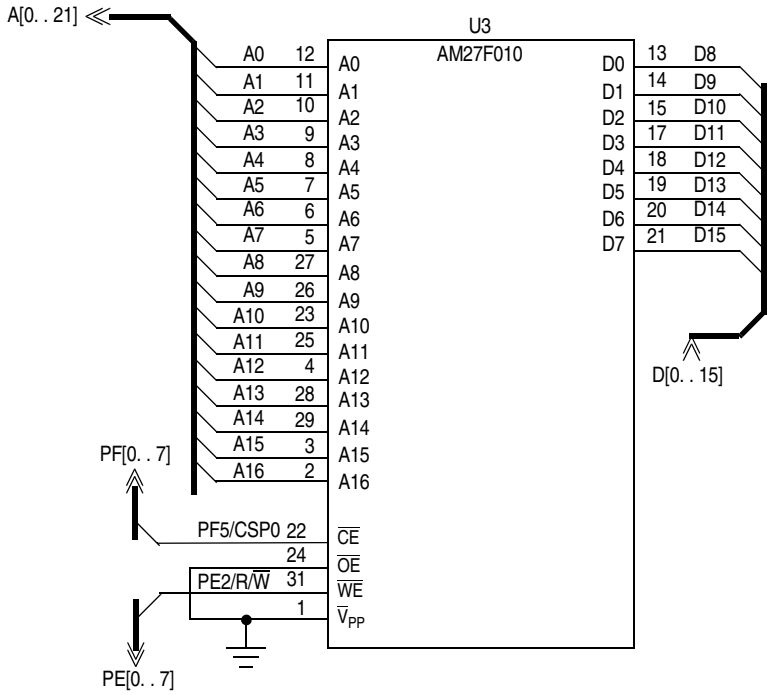


Figure 1-5. Expanded Narrow Mode SRAM Expansion Schematic (Sheet 2 of 3)

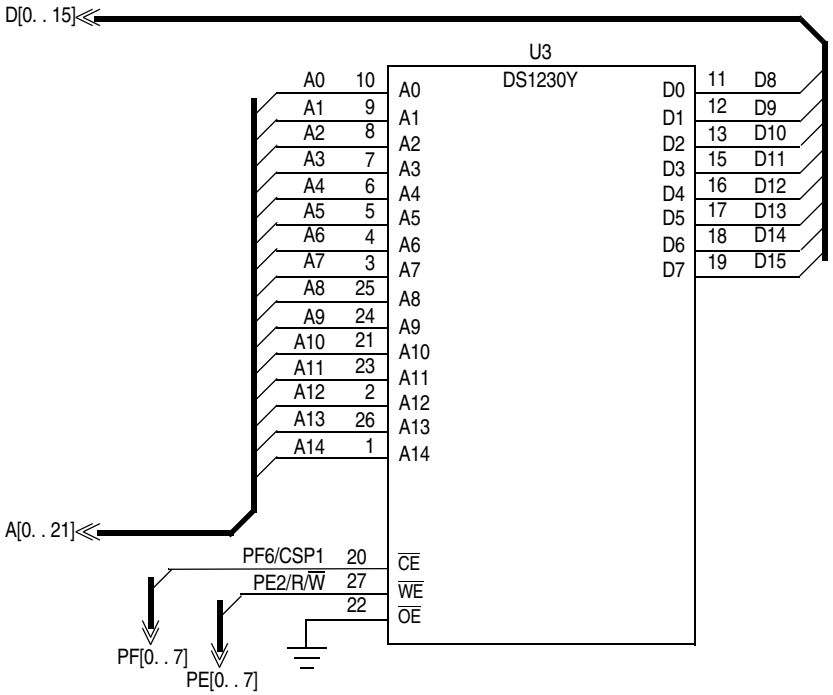


Figure 1-5. Expanded Narrow Mode SRAM Expansion Schematic (Sheet 3 of 3)

Chapter 2
Register Block

2.1 Overview

The register block can be mapped to any 2-Kbyte boundary within the standard 64-Kbyte address space by manipulating bits REG15–REG11 in the INITRG register. INITRG establishes the upper five bits of the register block's 16-bit address.

The register block occupies the first 512 bytes of the 2-Kbyte block. Figure 2-1 shows the default addressing.

2.2 Register Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PORTA) See page 64.	Read: Write:	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
		Reset:	0	0	0	0	0	0	0	0
$0001	Port B Data Register (PORTB) See page 65.	Read: Write:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
		Reset:	0	0	0	0	0	0	0	0
$0002	Port A Data Direction Register (DDRA) See page 64.	Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0
$0003	Port B Data Direction Register (DDRB) See page 65.	Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Reset:	0	0	0	0	0	0	0	0
$0004	Port C Data Register (PORTC) See page 66.	Read: Write:	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
		Reset:	0	0	0	0	0	0	0	0
$0005	Port D Data Register (PORTD) See page 67.	Read: Write:	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
		Reset:	0	0	0	0	0	0	0	0
$0006	Port C Data Direction Register (DDRC) See page 66.	Read: Write:	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 1 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0007	Port D Data Direction Register (DDRD) See page 67.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PORTE) See page 68.	Read: Write:	PE7	PE6	PE5	PD4	PD3	PD2	PD1	PD0
		Reset:	0	0	0	0	1	0	0	0
$0009	Port E Data Direction Register (DDRE) See page 68.	Read: Write:	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
		Reset:	0	0	0	0	0	0	1	1
$000A	Port E Assignment Register (PEAR) See page 69.	Read: Write:	ARSIE	PLLTE	PIPOE	NECLK	LSTRE	RDWE	0	0
		Reset:	0	0	1	0	1	1	0	0
$000B	Mode Register (MODE) See page 58.	Read: Write:	SMODN	MODB	MODA	ESTR	IVIS	0	EMD	EME
		Reset:	0	0	0	1	1	0	1	1
$000C	Pullup Control Register (PUCR) See page 71.	Read: Write:	PUPH	PUPG	PUPF	PUPE	PUPD	PUC	PUPB	PUPA
		Reset:	1	1	1	1	1	1	1	1
$000D	Reduced Drive Register (RDRIV) See page 72.	Read: Write:	RDPJ	RDPH	RDPG	RDPF	RDPE	PRPD	RDPC	RDPAB
		Reset:	0	0	0	0	0	0	0	0
$000E	Reserved		R	R	R	R	R	R	R	R
$000F	Reserved		R	R	R	R	R	R	R	R
$0010	RAM Initialization Register (INITRM) See page 60.	Read: Write:	RAM15	RAM14	RAM13	RAM12	RAM11	0	0	0
		Reset:	0	0	0	0	1	0	0	0
$0011	Register Initialization Register (INITRG) See page 59.	Read: Write:	REG15	REG14	REG13	REG12	REG11	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$0012	EEPROM Initialization Register (INITEE) See page 60.	Read: Write:	EE15	EE14	EE13	EE12	0	0	0	EEON
		Reset:	0	0	0	1	0	0	0	1
$0013	Miscellaneous Mapping Control Register (MISC) See page 61.	Read: Write:	EWDIR	NDRC	0	0	0	0	0	0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 2 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0014	Real-Tme Interrupt Control Register (RTICTL) See page 105.	Read:	RTIE	RSWAI	RSBCK	0	RTBYP	RTR2	RTR1	RTR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	Real-Time Interrupt Flag Register (RTIFLG) See page 107.	Read:	RTIF	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0016	COP Control Register (COPCTL) See page 107.	Read:	CME	FCME	FCM	FCOP	DISR	CR2	CR1	CR0
		Write:								
		Reset:	0	0	0	0	0	1	1	1
$0017	Arm/Reset COP Register (COPRST) See page 109.	Read:	0	0	0	0	0	0	0	0
		Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$0018	Reserved		R	R	R	R	R	R	R	R
↓	↓									
$001D	Reserved		R	R	R	R	R	R	R	R
$001E	Interrupt Control Register (INTCR) See page 51.	Read:	IRQE	IRQEN	DLY	0	0	0	0	0
		Write:								
		Reset:	0	1	1	0	0	0	0	0
$001F	Highest Priority I Interrupt Register (HPRIO) See page 51.	Read:	1	1	PSEL5	PSEL4	PSEL3	PSEL2	PSEL1	0
		Write:								
		Reset:	1	1	1	1	0	0	1	0
$0020	Port D Key Wakeup Interrupt Enable Register (KWIED) See page 94.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0021	Port D Key Wakeup Flag Register (KWIFD) See page 95.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0022	Reserved		R	R	R	R	R	R	R	R
$0023	Reserved		R	R	R	R	R	R	R	R
$0024	Port H Data Register (PORTH) See page 95.	Read:	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0025	Port H Data Direction Register (DDRH) See page 96.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 3 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0026	Port H Key Wakeup Interrupt Enable Register (KWIEH) See page 96.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$0027	Port H Key Wakeup Flag Register (KWIFH) See page 96.	Read: Write:	KWIFH7	KWIFH6	KWIFH5	KWIFH4	KWIFH3	KWIFH2	KWIFH1	KWIFH0
		Reset:	0	0	0	0	0	0	0	0
$0028	Port J Data Register (PORTJ) See page 97.	Read: Write:	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
		Reset:	0	0	0	0	0	0	0	0
$0029	Port J Data Direction Register (DDRJ) See page 97.	Read: Write:	DDRJ7	DDRJ6	DDRJ5	DDRJ4	DDRJ3	DDRJ2	DDRJ1	DDRJ0
		Reset:	0	0	0	0	0	0	0	0
$002A	Port J Key Wakeup Interrupt Enable Register (KWIEJ) See page 97.	Read: Write:	KWIEJ7	KWIEJ6	KWIEJ5	KWIEJ4	KWIEJ3	KWIEJ2	KWIEJ1	KWIEJ0
		Reset:	0	0	0	0	0	0	0	0
$002B	Port J Key Wakeup Flag Register (KWIFJ) See page 98.	Read: Write:	KWIFJ7	KWIFJ6	KWIFJ5	KWIFJ4	KWIFJ3	KWIFJ2	KWIFJ1	KWIFJ0
		Reset:	0	0	0	0	0	0	0	0
$002C	Port J Key Wakeup Polarity Register (KPOLJ) See page 98.	Read: Write:	KPOLJ7	KPOLJ6	KPOLJ5	KPOLJ4	KPOLJ3	KPOLJ2	KPOLJ1	KPOLJ0
		Reset:	0	0	0	0	0	0	0	0
$002D	Port J Key Wakeup Pullup/Pulldown Select Register (PUPSJ) See page 99.	Read: Write:	PUPSJ7	PUPSJ6	PUPSJ5	PUPSJ4	PUPSJ3	PUPSJ2	PUPSJ1	PUPSJ0
		Reset:	0	0	0	0	0	0	0	0
$002E	Port J Key Wakeup Pullup/Pulldown Enable Register (PULEJ) See page 99.	Read: Write:	PULEJ7	PULEJ6	PULEJ5	PULEJ4	PULEJ3	PULEJ2	PULEJ1	PULEJ0
		Reset:	0	0	0	0	0	0	0	0
$002F	Reserved		R	R	R	R	R	R	R	R
$0030	Port F Data Register (PORTF) See page 85.	Read: Write:	0	PF6	PF5	PF4	PF3	PF2	PF1	PF0
		Reset:	0	0	0	0	0	0	0	0
$0031	Port G Data Register (PORTG) See page 85.	Read: Write:	0	0	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$0032	Port F Data Direction Register (DDRF) See page 86.	Read: Write:	0	DDRF6	DDRF5	DDRF4	DDRF3	DDRF2	DDRF1	DDRF0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 4 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0033	Port G Data Direction Register (DDRG) See page 86.	Read:	0	0	DDRG5	DDRG4	DDRG3	DDRG2	DDRG1	DDRG0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0034	Data Page Register (DPAGE) See page 86.	Read:	PD19	PD18	PD17	PD16	PD15	PD14	PD13	PD12
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0035	Program Page Register (PPAGE) See page 87.	Read:	PPA21	PPA20	PPA19	PPA18	PPA17	PPA16	PPA15	PPA14
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0036	Extra Page Register (EPAGE) See page 87.	Read:	PEA17	PEA16	PEA15	PEA14	PEA13	PEA12	PEA11	PEA10
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0037	Window Definition Register (WINDEF) See page 87.	Read:	DWEN	PWEN	EWEN	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0038	Memory Expansion Assignment Register (MXAR) See page 88.	Read:	0	0	A21E	A20E	A19E	A18E	A17E	A16E
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0039	Reserved		R	R	R	R	R	R	R	R
$003A	Reserved		R	R	R	R	R	R	R	R
$003B	Reserved		R	R	R	R	R	R	R	R
$003C	Chip-Select Control Register 0 (CSCTL0) See page 89.	Read:	0	CSP1E	CSP0E	CSDE	CS3E	CS2E	CS1E	CS0E
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$003D	Chip-Select Control Register 1 (CSCTL1) See page 90.	Read:	0	CSP1FL	CSPA21	CSDHF	CS3EP	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$003E	Chip-Select Stretch Register 0 (CSSTR0) See page 91.	Read:	0	0	SRP1A	SRP1B	SRP0A	SRP0B	STRDA	STRDB
		Write:								
		Reset:	0	0	1	1	1	1	1	1
$003F	Chip-Select Stretch Register 1 (CSSTR1) See page 91.	Read:	STR3A	STR3B	STR2A	STR2B	STR1A	STR1B	STR0A	STR0B
		Write:								
		Reset:	0	0	1	1	1	1	1	1
$0040	Loop Divider Register High (LDVH) See page 113.	Read:	0	0	0	0	LDV11	LDV10	LDV9	LDV8
		Write:								
		Reset:	0	0	0	0	1	1	1	1

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 5 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0041	Loop Divider Register Low (LDVL) See page 113.	Read: Write:	LDV7	LDV6	LDV5	LDV4	LDV3	LDV2	LDV1	LDV0
		Reset:	1	1	1	1	1	1	1	1
$0042	Reference Divider Register High (RDVH) See page 114.	Read:	0	0	0	0	RDV11	RDV10	RDV9	RDV8
		Write:								
		Reset:	0	0	0	0	1	1	1	1
$0043	Reference Divider Register Low (RDVL) See page 114.	Read: Write:	RDV7	RDV6	RDV5	RDV4	RDV3	RDV2	RDV1	RDV0
		Reset:	1	1	1	1	1	1	1	1
$0044	Reserved		R	R	R	R	R	R	R	R
$0045	Reserved		R	R	R	R	R	R	R	R
$0046	Reserved		R	R	R	R	R	R	R	R
$0047	Clock Control Register (CLKCTL) See page 114.	Read:	LCKF	PLLON	PLLS	BCSC	BCSB	BCSA	MCSB	MCSA
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0048	Reserved		R	R	R	R	R	R	R	R
↓	↓									
$005F	Reserved		R	R	R	R	R	R	R	R
$0060	ATD Control Register 0 (ATDCTL0) See page 199.	Read: Write:	0	0	0	0	0	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$0061	ATD Control Register 1 (ATDCTL1) See page 199.	Read:	0	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0062	ATD Control Register 2 (ATDCTL2) See page 200.	Read:	ADPU	AFFC	AWAI	0	0	0	ASCIE	ASCIF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0063	ATD Control Register 3 (ATDCTL3) See page 201.	Read:	0	0	0	0	0	0	FRZ1	FRZ0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0064	ATD Control Register 4 (ATDCTL4) See page 201.	Read:	0	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
		Write:								
		Reset:	0	0	0	0	0	0	0	1
$0065	ATD Control Register 5 (ATDCTL5) See page 202.	Read:	0	S8CM	SCAN	MULT	CD	CC	CB	CA
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 6 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0066	ATD Status Register 1 (ATDSTAT1) See page 204.	Read:	SCF	0	0	0	0	CC2	CC1	CC0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0067	ATD Status Register 2 (ATDSTAT2) See page 204.	Read:	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0068	ATD Test Register 1 (ATDTEST1) See page 205.	Read: Write:	SAR9	SAR8	SAR7	SAR6	SAR5	SAR4	SAR3	SAR2
		Reset:	0	0	0	0	0	0	0	0
$0069	ATD Test Register 2 (ATDTEST2) See page 205.	Read: Write:	SAR1	SAR0	RST	TSTOUT	TST3	TST2	TST1	TST0
		Reset:	0	0	0	0	0	0	0	0
$006A	Reserved		R	R	R	R	R	R	R	R
↓	↓									
$006E	Reserved		R	R	R	R	R	R	R	R
$006F	Port AD Data Input Register (PORTAD) See page 207.	Read: Write:	PAD7	PAD6	PAD5	PAD4	PAD3	PAD2	PAD1	PAD0
		Reset:	0	0	0	0	0	0	0	0
$0070	ATD Result Register 0 (ADR0H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0071	Reserved		R	R	R	R	R	R	R	R
$0072	ATD Result Register 1 (ADR1H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0073	Reserved		R	R	R	R	R	R	R	R
$0074	ATD Result Register 2 (ADR2H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0075	Reserved		R	R	R	R	R	R	R	R
$0076	ATD Result Register 3 (ADR3H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0077	Reserved		R	R	R	R	R	R	R	R

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 7 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0078	ATD Result Register 4 (ADR4H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0079	Reserved		R	R	R	R	R	R	R	R
$007A	ATD Result Register 5 (ADR5H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$007B	Reserved		R	R	R	R	R	R	R	R
$007C	ATD Result Register 6 (ADR6H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$007D	Reserved		R	R	R	R	R	R	R	R
$007E	ATD Result Register 7 (ADR7H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$007F	Reserved		R	R	R	R	R	R	R	R
$0080	Timer IC/OC Select Register (TIOS) See page 125.	Read:	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0081	Timer Compare Force Register (CFORC) See page 125.	Read:	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0082	Timer Output Compare 7 Mask Register (OC7M) See page 126.	Read:	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0083	Timer Output Compare 7 Data Register (OC7D) See page 126.	Read:	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0084	Timer Counter Register High (TCNTH) See page 127.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0085	Timer Counter Register Low (TCNTL) See page 127.	Read:	Bit 7	BIt 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 8 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0086	Timer System Control Register (TSCR) See page 127.	Read:	TEN	TSWAI	TSBCK	TFFCA	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0087	Reserved		R	R	R	R	R	R	R	R
$0088	Timer Control Register 1 (TCTL1) See page 129.	Read:	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0089	Timer Control Register 2 (TCTL2) See page 129.	Read:	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$008A	Timer Control Register 3 (TCTL3) See page 130.	Read:	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$008B	Timer Control Register 4 (TCTL4) See page 130.	Read:	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$008C	Timer Mask Register 1 (TMSK1) See page 130.	Read:	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$008D	Timer Mask Register 2 (TMSK2) See page 131.	Read:	TOI	0	PUPT	RDPT	TCRE	PR2	PR1	PR0
		Write:								
		Reset:	0	0	1	1	0	0	0	0
$008E	Timer Flag Register 1 (TFLG1) See page 132.	Read:	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$008F	Timer Flag Register 2 (TFLG2) See page 132.	Read:	TOF	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0090	Timer Channel 0 Register High (TC0H) See page 133.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0091	Timer Channel 0 Register Low (TC0L) See page 133.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0092	Timer Channel 1 Register High (TC1H) See page 133.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 9 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0093	Timer Channel 1 Register Low (TC1L) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$0094	Timer Channel 2 Register High (TC2H) See page 133.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$0095	Timer Channel 2 Register Low (TC2L) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$0096	Timer Channel 3 Register High (TC3H) See page 133.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$0097	Timer Channel 3 Register Low (TC3L) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$0098	Timer Channel 4 Register High (TC4H) See page 133.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$0099	Timer Channel 4 Register Low (TC4L) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$009A	Timer Channel 5 Register High (TC5H) See page 133.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$009B	Timer Channel 5 Register Low (TC5L) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$009C	Timer Channel 6 Register High (TC6H) See page 133.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$009D	Timer Channel 6 Register Low (TC6L) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$009E	Timer Channel 7 Register High (TC7H) See page 133.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 10 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$009F	Timer Channel 7 Register Low (TC7L) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$00A0	Pulse Accumulator Control Register (PACTL) See page 134.	Read:	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00A1	Pulse Accumulator Flag Register (PAFLG) See page 135.	Read:	Bit 0	0	0	0	0	0	PAOVF	PAIF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00A2	Pulse Accumulator Counter Register High (PACNTH) See page 136.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$00A3	Pulse Accumulator Counter Register Low (PACNTL) See page 136.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$00A4	Reserved		R	R	R	R	R	R	R	R
↓	↓									
$00AC	Reserved		R	R	R	R	R	R	R	R
$00AD	Timer Test Register (TIMTST) See page 137.	Read:	0	0	0	0	0	0	TCBYP	PCBYP
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00AE	Timer Port Data Register (PORTT) See page 139.	Read: Write:	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
		Reset:	Unaffected by reset							
$00AF	Timer Port Data Direction Register (DDRT) See page 140.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$00B0	Reserved		R	R	R	R	R	R	R	R
↓	↓									
$00BF	Reserved		R	R	R	R	R	R	R	R
$00C0	SCI 0 Baud Rate Register High (SC0BDH) See page 168.	Read: Write:	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8
		Reset:	0	0	0	0	0	0	0	0
$00C1	SCI 0 Baud Rate Register Low (SC0BDL) See page 168.	Read: Write:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Reset:	0	0	0	0	0	1	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 11 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00C2	SCI 0 Control Register 1 (SC0CR1) See page 169.	Read:	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C3	SCI 0 Control Register 2 (SC0CR2) See page 171.	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C4	SCI 0 Status Register 1 (SC0SR1) See page 172.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$00C5	SCI 0 Status Register 2 (SC0SR2) See page 173.	Read:	0	0	0	0	0	0	0	RAF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C6	SCI 0 Data Register High (SC0DRH) See page 174.	Read:	R8	T8	0	0	0	0	0	0
		Write:								
		Reset:	Unaffected by reset							
$00C7	SCI 0 Data Register Low (SC0DRL) See page 174.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$00C8	SCI 1 Baud Rate Register High (SC1BDH) See page 168.	Read:	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C9	SCI 1 Baud Rate Register Low (SC1BDL) See page 168.	Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Write:								
		Reset:	0	0	0	0	0	1	0	0
$00CA	SCI 1 Control Register 1 (SC1CR1) See page 169.	Read:	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00CB	SCI 1 Control Register 2 (SC1CR2) See page 171.	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00CC	SCI 1 Status Register 1 (SC1SR1) See page 172.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$00CD	SCI 1 Status Register 2 (SC1SR2) See page 173.	Read:	0	0	0	0	0	0	0	RAF
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 12 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00CE	SCI 1 Data Register High (SC1DRH) See page 174.	Read:	R8	T8	0	0	0	0	0	0
		Write:								
		Reset:	Unaffected by reset							
$00CF	SCI 1 Data Register Low (SC1DRL) See page 174.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$00D0	SPI 0 Control Register 1 (SP0CR1) See page 186.	Read:	SPIE	SPE	SWOM	MSTR	CPOL	CPHA	SSOE	LSBF
		Write:								
		Reset:	0	0	0	0	0	1	0	0
$00D1	SPI 0 Control Register 2 (SP0CR2) See page 187.	Read:	0	0	0	0	PUPS	RDS	0	SPC0
		Write:								
		Reset:	0	0	0	0	1	0	0	0
$00D2	SPI Baud Rate Register (SP0BR) See page 188.	Read:	0	0	0	0	0	SPR2	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00D3	SPI Status Register (SP0SR) See page 189.	Read:	SPIF	WCOL	0	MODF	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00D4	Reserved		R	R	R	R	R	R	R	R
$00D5	SPI Data Register (SP0DR) See page 190.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Unaffected by reset							
$00D6	Port S Data Register (PORTS) See page 147.	Read:	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
		Write:								
		Reset:	Unaffected by reset							
$00D7	Port S Data Direction Register (DDRS) See page 148.	Read:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00D8	Reserved		R	R	R	R	R	R	R	R
↓	↓									
$00EF	Reserved		R	R	R	R	R	R	R	R
$00F0	EEPROM Configuration Register (EEMCR) See page 74.	Read:	1	1	1	1	1	EESWAI	PROTLCK	EERC
		Write:								
		Reset:	1	1	1	1	1	1	0	0
$00F1	EEPROM Block Protect Register (EEPROT) See page 75.	Read:	1	BPROT6	BPROT5	BPROT4	BPROT3	BPROT2	BPROT1	BPROT0
		Write:								
		Reset:	1	1	1	1	1	1	1	1

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 13 of 14)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00F2	EEPROM Test Register (EETST) See page 75.	Read:	EEODD	EEVEN	MARG	EECPD	EECPRD	0	EECPM	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00F3	EEPROM Programming Register (EEPROG) See page 76.	Read: Write:	BULKP	0	0	BYTE	ROW	ERASE	EELAT	EEPGM
		Reset:	1	0	0	0	0	0	0	0
$00F4	Reserved		R	R	R	R	R	R	R	R
↓	↓									
$01FF	Reserved		R	R	R	R	R	R	R	R

= Unimplemented R = Reserved U = Unaffected

Figure 2-1. Register Map (Sheet 14 of 14)

2.3 Modes of Operation

PORTA, PORTB, PORTC, and data direction registers DDRA, DDRB, and DDRC are not in the map in expanded and peripheral modes. PEAR, MODE, PUCR, and RDRIV are not in the map in peripheral mode.

When EMD is set:

- PORTD and DDRD are not the map in wide expanded modes, peripheral mode, and narrow special expanded mode.
- PORTE and DDRE are not in the map in peripheral mode and expanded modes.
- KWIED and KWIFD are not in the map in wide expanded modes and narrow special expanded mode.

Chapter 3
Central Processor Unit (CPU12)

3.1 Overview

The CPU12 is a high-speed, 16-bit processor unit. It has full 16-bit data paths and wider internal registers (up to 20 bits) for high-speed extended math instructions. The instruction set is a proper superset of the M68HC11instruction set. The CPU12 allows instructions with odd byte counts, including many single-byte instructions. This provides efficient use of ROM space. An instruction queue buffers program information so the CPU always has immediate access to at least three bytes of machine code at the start of every instruction. The CPU12 also offers an extensive set of indexed addressing capabilities.

3.2 Programming Model

CPU12 registers are an integral part of the CPU and are not addressed as if they were memory locations. See Figure 3-1.

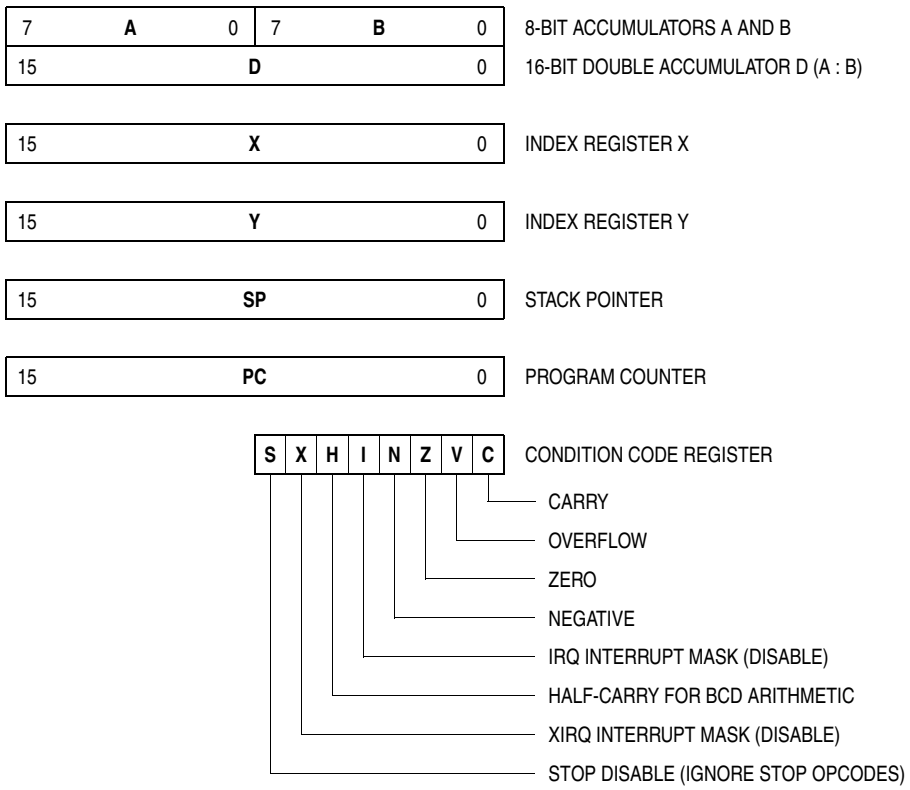


Figure 3-1. Programming Model

3.3 CPU Registers

This section describes the CPU registers.

3.3.1 Accumulators A and B

Accumulators A and B are general-purpose 8-bit accumulators that contain operands and results of arithmetic calculations or data manipulations.

	Bit 7	6	5	4	3	2	1	Bit 0
	A7	A6	A5	A4	A3	A2	A1	A0
Reset:	Unaffected by reset							

Figure 3-2. Accumulator A (A)

	Bit 7	6	5	4	3	2	1	Bit 0
	B7	B6	B5	B4	B3	B2	B1	B0
Reset:	Unaffected by reset							

Figure 3-3. Accumulator B (B)

3.3.2 Accumulator D

Accumulator D is the concatenation of accumulators A and B. Some instructions treat the combination of these two 8-bit accumulators as a 16-bit double accumulator.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	D15 (A7)	D14 (A6)	D13 (A5)	D12 (A4)	D11 (A3)	D10 (A2)	D9 (A1)	D8 (A0)	D7 (B7)	D6 (B6)	D5 (B5)	D4 (B4)	D3 (B3)	D2 (B2)	D1 (B1)	D0 (B0)
Reset:	Unaffected by reset															

Figure 3-4. Accumulator D (D)

3.3.3 Index Registers X and Y

Index registers X and Y are used for indexed addressing. Indexed addressing adds the value in an index register to a constant or to the value in an accumulator to form the effective address of the operand.

Index registers X and Y can also serve as temporary data storage locations.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	X15	X14	X13	X12	X11	X10	X9	X8	X7	X6	X5	X4	X3	X2	X1	X0
Reset:	Unaffected by reset															

Figure 3-5. Index Register X (X)

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	Y15	Y14	Y13	Y12	Y11	Y10	Y9	Y8	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
Reset:	Unaffected by reset															

Figure 3-6. Index Register Y (Y)

3.3.4 Stack Pointer

The stack pointer (SP) contains the last stack address used. The CPU12 supports an automatic program stack that is used to save system context during subroutine calls and interrupts.

The stack pointer can also serve as a temporary data storage location or as an index register for indexed addressing.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
Reset:	Unaffected by reset															

Figure 3-7. Stack Pointer (SP)

3.3.5 Program Counter

The program counter contains the address of the next instruction to be executed.

The program counter can also serve as an index register in all indexed addressing modes except autoincrement and autodecrement.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
Reset:	Unaffected by reset															

Figure 3-8. Program Counter (PC)

3.3.6 Condition Code Register

	Bit 7	6	5	4	3	2	1	Bit 0
	S	X	H	I	N	Z	V	C
Reset:	1	1	U	1	U	U	U	U

U = Unaffected

Figure 3-9. Condition Code Register (CCR)

S — Stop Disable Bit
Setting the S bit disables the STOP instruction.

X — XIRQ Interrupt Mask Bit
Setting the X bit masks interrupt requests from the $\overline{\text{XIRQ}}$ pin.

H — Half-Carry Flag
The H flag is used only for BCD arithmetic operations. It is set when an ABA, ADD, or ADC instruction produces a carry from bit 3 of accumulator A. The DAA instruction uses the H flag and the C flag to adjust the result to correct BCD format.

I — Interrupt Mask Bit
Setting the I bit disables maskable interrupt sources.

N — Negative Flag
The N flag is set when the result of an operation is less than 0.

Z — Zero Flag
The Z flag is set when the result of an operation is all 0s.

V — Two's Complement Overflow Flag
The V flag is set when a two's complement overflow occurs.

C — Carry/Borrow Flag
The C flag is set when an addition or subtraction operation produces a carry or borrow.

3.4 Data Types

The CPU12 supports four data types:

1. Bit data
2. 8-bit and 16-bit signed and unsigned integers
3. 16-bit unsigned fractions
4. 16-bit addresses

A byte is eight bits wide and can be accessed at any byte location. A word is composed of two consecutive bytes with the most significant byte at the lower value address. There are no special requirements for alignment of instructions or operands.

3.5 Addressing Modes

Addressing modes determine how the CPU accesses memory locations to be operated upon. The CPU12 includes all of the addressing modes of the M68HC11 CPU as well as several new forms of indexed addressing. Table 3-1 is a summary of the available addressing modes.

Table 3-1. Addressing Mode Summary

Addressing Mode	Source Format	Abbreviation	Description
Inherent	INST	INH	Operands (if any) are in CPU registers.
Immediate	INST #*opr8i* or INST #*opr16i*	IMM	Operand is included in instruction stream. 8- or 16-bit size implied by context
Direct	INST *opr8a*	DIR	Operand is the lower 8 bits of an address in the range $0000–$00FF.
Extended	INST *opr16a*	EXT	Operand is a 16-bit address
Relative	INST *rel8* or INST *rel16*	REL	An 8-bit or 16-bit relative offset from the current pc is supplied in the instruction.
Indexed (5-bit offset)	INST *oprx5,xysp*	IDX	5-bit signed constant offset from x, y, sp, or pc
Indexed (auto pre-decrement)	INST *oprx3,–xys*	IDX	Auto pre-decrement x, y, or sp by 1 ~ 8
Indexed (auto pre-increment)	INST *oprx3,+xys*	IDX	Auto pre-increment x, y, or sp by 1 ~ 8
Indexed (auto post-decrement)	INST *oprx3,xys–*	IDX	Auto post-decrement x, y, or sp by 1 ~ 8
Indexed (auto post-increment)	INST *oprx3,xys+*	IDX	Auto post-increment x, y, or sp by 1 ~ 8
Indexed (accumulator offset)	INST *abd,xysp*	IDX	Indexed with 8-bit (A or B) or 16-bit (D) accumulator offset from x, y, sp, or pc
Indexed (9-bit offset)	INST *oprx9,xysp*	IDX1	9-bit signed constant offset from x, y, sp, or pc (lower 8-bits of offset in one extension byte)
Indexed (16-bit offset)	INST *oprx16,xysp*	IDX2	16-bit constant offset from x, y, sp, or pc (16-bit offset in two extension bytes)
Indexed-indirect (16-bit offset)	INST [*oprx16,xysp*]	[IDX2]	Pointer to operand is found at... 16-bit constant offset from x, y, sp, or pc (16-bit offset in two extension bytes)
Indexed-indirect (D accumulator offset)	INST [D,*xysp*]	[D,IDX]	Pointer to operand is found at... x, y, sp, or pc plus the value in D

3.6 Indexed Addressing Modes

The CPU12 indexed modes reduce execution time and eliminate code size penalties for using the Y index register. CPU12 indexed addressing uses a postbyte plus zero, one, or two extension bytes after the instruction opcode.

The postbyte and extensions do these tasks:

- Specify which index register is used
- Determine whether a value in an accumulator is used as an offset
- Enable automatic pre- or post-increment or decrement
- Specify use of 5-bit, 9-bit, or 16-bit signed offsets

Table 3-2. Summary of Indexed Operations

Postbyte Code (xb)	Source Code Syntax	Comments rr: 00 = X, 01 = Y, 10 = SP, 11 = PC
rr0nnnnn	,r n,r -n,r	**5-bit constant offset** n = –16 to +15 r can specify x, y, sp, or pc
111rr0zs	n,r -n,r	**Constant offset** (9- or 16-bit signed) z:0 = 9-bit with sign in LSB of postbyte(s) 1 = 16-bit if z = s = 1, 16-bit offset indexed-indirect (see below) rr can specify x, y, sp, or pc
111rr011	[n,r]	**16-bit offset indexed-indirect** rr can specify x, y, sp, or pc
rr1pnnnn	n,-r n,+r n,r- n,r+	**Auto pre-decrement/increment** or **Auto post-decrement/increment**; p = pre-(0) or post-(1), n = –8 to –1, +1 to +8 rr can specify x, y, or sp (pc not a valid choice)
111rr1aa	A,r B,r D,r	**Accumulator offset** (unsigned 8-bit or 16-bit) aa:00 = A 01 = B 10 = D (16-bit) 11 = see accumulator D offset indexed-indirect rr can specify x, y, sp, or pc
111rr111	[D,r]	**Accumulator D offset indexed-indirect** rr can specify x, y, sp, or pc

3.7 Opcodes and Operands

The CPU12 uses 8-bit opcodes. Each opcode identifies a particular instruction and associated addressing mode to the CPU. Several opcodes are required to provide each instruction with a range of addressing capabilities.

Only 256 opcodes would be available if the range of values were restricted to the number that can be represented by 8-bit binary numbers. To expand the number of opcodes, a second page is added to the opcode map. Opcodes on the second page are preceded by an additional byte with the value $18.

To provide additional addressing flexibility, opcodes can also be followed by a postbyte or extension bytes. Postbytes implement certain forms of indexed addressing, transfers, exchanges, and loop primitives. Extension bytes contain additional program information such as addresses, offsets, and immediate data.

Chapter 4
Resets and Interrupts

4.1 Introduction

Resets and interrupts are exceptions. Each exception has a 16-bit vector that points to the memory location of the associated exception-handling routine. Vectors are stored in the upper 128 bytes of the standard 64-Kbyte address map.

The six highest vector addresses are used for resets and non-maskable interrupt sources. The remainder of the vectors are used for maskable interrupts, and all must be initialized to point to the address of the appropriate service routine.

4.2 Exception Priority

A hardware priority hierarchy determines which reset or interrupt is serviced first when simultaneous requests are made. Six sources are not maskable. The remaining sources are maskable and any one of them can be given priority over other maskable interrupts.

The priorities of the non-maskable sources are:

1. POR (power-on reset) or $\overline{\text{RESET}}$ pin
2. Clock monitor reset
3. COP (computer operating properly) watchdog reset
4. Unimplemented instruction trap
5. Software interrupt instruction (SWI)
6. $\overline{\text{XIRQ}}$ signal (if X bit in CCR = 0)

4.3 Maskable Interrupts

Maskable interrupt sources include on-chip peripheral systems and external interrupt service requests. Interrupts from these sources are recognized when the global interrupt mask bit (I) in the CCR is cleared. The default state of the I bit out of reset is 1, but it can be written at any time.

Interrupt sources are prioritized by default but any one maskable interrupt source may be assigned the highest priority by means of the HPRIO register. The relative priorities of the other sources remain the same.

An interrupt that is assigned highest priority is still subject to global masking by the I bit in the CCR or by any associated local bits. Interrupt vectors are not affected by priority assignment. HPRIO can only be written while the I bit is set (interrupts inhibited). Table 4-1 lists interrupt sources and vectors in default order of priority.

Table 4-1. Interrupt Vector Map

Vector Address	Exception Source	Flag	Local Enable	CCR Mask	HPRIO Value to Elevate
$FFFE, $FFFF	Power-on reset	—	None	None	—
$FFFC, $FFFD	Clock monitor reset	—	CME, FCME	None	—
$FFFA, $FFFB	COP reset	—	COP rate selected	None	—
$FFF8, $FFF9	Unimplemented instruction trap	—	None	None	—
$FFF6, $FFF7	SWI instruction	—	None	None	—
$FFF4, $FFF5	$\overline{\text{XIRQ}}$ pin	—	None	X bit	—
$FFF2, $FFF3	$\overline{\text{IRQ}}$ pin or key wakeup D	—	IRQEN, KWIED[7–0]	I bit	$F2
$FFF0, $FFF1	Real-time interrupt	RTIF	RTIE	I bit	$F0
$FFEE, $FFEF	Timer channel 0	C0F	C0I	I bit	$EE
$FFEC, $FFED	Timer channel 1	C1F	C1I	I bit	$EC
$FFEA, $FFEB	Timer channel 2	C2F	C2I	I bit	$EA
$FFE8, $FFE9	Timer channel 3	C3F	C3I	I bit	$E8
$FFE6, $FFE7	Timer channel 4	C4F	C4I	I bit	$E6
$FFE4, $FFE5	Timer channel 5	C5F	C5I	I bit	$E4
$FFE2, $FFE3	Timer channel 6	C6F	C6I	I bit	$E2
$FFE0, $FFE1	Timer channel 7	C7F	C7I	I bit	$E0
$FFDE, $FFDF	Timer overflow	TOF	TOI	I bit	$DE
$FFDC, $FFDD	Pulse accumulator overflow	PAOVF	PAOVI	I bit	$DC
$FFDA, $FFDB	Pulse accumulator input edge	PAIF	PAI	I bit	$DA
$FFD8, $FFD9	SPI serial transfer complete Mode fault	SPIF MODF	SPI0E	I bit	$D8
$FFD6, $FFD7	SCI0 transmit data register empty SCI0 transmission complete SCI0 receive data register full SCI0 receiver overrun SCI0 receiver idle	TDRE TC RDRF OR IDLE	TIE TCIE RIE RIE ILIE	I bit	$D6
$FFD4, $FFD5	SCI1 transmit data register empty SCI1 transmission complete SCI1 receive data register full SCI1 receiver overrun SCI1 receiver idle	TDRE TC RDRF OR IDLE	TIE TCIE RIE RIE ILIE	I bit	$D4
$FFD2, $FFD3	ATD	ASCIF	ASCIE	I bit	$D2
$FFD0, $FFD1	Key wakeup J (stop wakeup)	—	KWIEJ[7–0]	I bit	$D0
$FFCE, $FFCF	Key wakeup H (stop wakeup)	—	KWIEH[7–0]	I bit	$CE
$FF80–$FFCD	Reserved	—	—	I bit	$80–$CC

4.4 Interrupt Registers

This section describes the interrupt registers.

4.4.1 Interrupt Control Register

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	IRQE	IRQEN	DLY	0	0	0	0	0
Write:								
Reset:	0	1	1	0	0	0	0	0

= Unimplemented

Figure 4-1. Interrupt Control Register (INTCR)

Read: Anytime
Write: Varies from bit to bit

IRQE — $\overline{\text{IRQ}}$ Edge-Sensitive-Only Bit

IRQE can be written once in normal modes. In special modes, IRQE can be written anytime, but the first write is ignored.

1 = $\overline{\text{IRQ}}$ responds only to falling edges.
0 = $\overline{\text{IRQ}}$ pin responds to low levels.

IRQEN — $\overline{\text{IRQ}}$ Enable Bit

IRQEN can be written anytime in all modes. The $\overline{\text{IRQ}}$ pin has an internal pullup.

1 = $\overline{\text{IRQ}}$ pin and key wakeup D connected to interrupt logic
0 = $\overline{\text{IRQ}}$ pin and key wakeup D disconnected from interrupt logic

DLY — Oscillator Startup Delay on Exit from Stop Mode Bit

DLY can be written once in normal modes. In special modes, DLY can be written anytime.
The delay time of about 4096 cycles is based on the M-clock rate chosen.

1 = Stabilization delay on exit from stop mode
0 = No stabilization delay on exit from stop mode

4.4.2 Highest Priority I Interrupt Register

Address: $001F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	1	1	PSEL5	PSEL4	PSEL3	PSEL2	PSEL1	0
Write:								
Reset:	1	1	1	1	0	0	1	0

= Unimplemented

Figure 4-2. Highest Priority I Interrupt Register (HPRIO)

Read: Anytime
Write: Only if I mask in CCR = 1 (interrupts inhibited)

To give a maskable interrupt source highest priority, write the low byte of the vector address to the HPRIO register. For example, writing $F0 to HPRIO assigns highest maskable interrupt priority to the real-time

interrupt timer ($FFF0). If an unimplemented vector address or a non-I-masked vector address (a value higher than $F2) is written, then $\overline{\text{IRQ}}$ is the default highest priority interrupt.

4.5 Resets

There are five possible sources of reset:

1. Power-on reset (POR)
2. External reset on the $\overline{\text{RESET}}$ pin
3. Reset from the alternate reset pin, ARST
4. The computer operating properly (COP) reset
5. Clock monitor reset

NOTE

The first three reset sources all share the power-on reset vector and the last two have their own vector for a total of three possible reset vectors.

Entry into reset is asynchronous and does not require a clock but the MCU cannot sequence out of reset without a system clock.

4.5.1 Power-On Reset

A positive transition on V_{DD} causes a power-on reset (POR). An external voltage level detector, or other external reset circuits, are the usual source of reset in a system. The POR circuit only initializes internal circuitry during cold starts and cannot be used to force a reset as system voltage drops.

4.5.2 External Reset

The CPU distinguishes between internal and external reset conditions by sensing whether the reset pin rises to a logic 1 in less than nine E-clock cycles after an internal device releases reset. When a reset condition is sensed, the $\overline{\text{RESET}}$ pin is driven low by an internal device for about 16 E-clock cycles, then released. Nine E-clock cycles later, it is sampled. If the pin is still held low, the CPU assumes that an external reset has occurred. If the pin is high, it indicates that the reset was initiated internally by either the COP system or the clock monitor.

To prevent a COP or clock monitor reset from being detected during an external reset, hold the reset pin low for at least 32 cycles. An external RC power-up delay circuit on the reset pin is not recommended since circuit charge time can cause the MCU to misinterpret the type of reset that has occurred.

4.5.3 COP Reset

The MCU includes a computer operating properly (COP) system to help protect against software failures. When COP is enabled, software must write $55 and $AA (in this order) to the COPRST register to keep a watchdog timer from timing out. Other instructions may be executed between these writes. A write of any value other than $55 or $AA or software failing to execute the sequence properly causes a COP reset to occur.

4.5.4 Clock Monitor Reset

If clock frequency falls below a predetermined limit when the clock monitor is enabled, a reset occurs.

4.6 Effects of Reset

When a reset occurs, MCU registers and control bits are changed to known startup states, as follows.

4.6.1 Operating Mode and Memory Map

The states of the BGND, MODA, and MODB pins during reset determine the operating mode and default memory mapping. The SMODN, MODA, and MODB bits in the MODE register reflect the status of the mode-select inputs at the rising edge of reset. Operating mode and default maps can subsequently be changed according to strictly defined rules.

4.6.2 Clock and Watchdog Control Logic

Reset enables the COP watchdog with the CR2–CR0 bits set for the longest timeout period. The clock monitor is disabled. The RTIF flag is cleared and automatic hardware interrupts are masked. The rate control bits are cleared, and must be initialized before the RTI system is used. The DLY control bit is set to specify an oscillator startup delay upon recovery from stop mode.

4.6.3 Interrupts

Reset initializes the HPRIO register with the value $F2, causing the $\overline{\text{IRQ}}$ pin to have the highest I bit interrupt priority. The $\overline{\text{IRQ}}$ pin is configured for level-sensitive operation (for wired-OR systems). However, the I and X bits in the CCR are set, masking $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$ interrupt requests.

4.6.4 Parallel I/O

If the MCU comes out of reset in an expanded mode, port A and port B are the address bus. Port C and port D are the data bus. In narrow mode, port C alone is the data bus. Port E pins are normally used to control the external bus. The PEAR register affects port E pin operation.

If the MCU comes out of reset in a single-chip mode, all ports are configured as general-purpose, high-impedance inputs except in normal narrow expanded mode (NNE). In NNE, PE3 is configured as an output driven high.

In expanded modes, PF5 is an active chip-select.

4.6.5 Central Processor Unit

After reset, the CPU fetches a vector from the appropriate address and begins executing instructions. The stack pointer and other CPU registers are indeterminate immediately after reset. The CCR X and I interrupt mask bits are set to mask any interrupt requests. The S bit is also set to inhibit the STOP instruction.

4.6.6 Memory

After reset, the internal register block is located at $0000–$01FF and RAM is at $0800–$0BFF. EEPROM is located at $1000–$1FFF in expanded modes and at $F000–$FFFF in single-chip modes.

4.6.7 Other Resources

The timer, serial communications interface (SCI), serial peripheral interface (SPI), and analog-to-digital converter (ATD) are off after reset.

4.7 Interrupt Recognition

Once enabled, an interrupt request can be recognized at any time after the I bit in the CCR is cleared. When an interrupt request is recognized, the CPU responds at the completion of the instruction being executed. Interrupt latency varies according to the number of cycles required to complete the instruction. Some of the longer instructions can be interrupted and resume normally after servicing the interrupt.

When the CPU begins to service an interrupt request, it:

- Clears the instruction queue
- Calculates the return address
- Stacks the return address and the contents of the CPU registers as shown in Table 4-2

Table 4-2. Stacking Order on Entry to Interrupts

Memory Location	Stacked Values
SP – 2	RTN_H : RTN_L
SP – 4	Y_H : Y_L
SP – 6	X_H : X_L
SP – 8	B : A
SP – 9	CCR

After stacking the CCR, the CPU:

- Sets the I bit to prevent other interrupts from disrupting the interrupt service routine
- Sets the X bit if an $\overline{\text{XIRQ}}$ interrupt request is pending
- Fetches the interrupt vector for the highest-priority request that was pending at the beginning of the interrupt sequence
- Begins execution of the interrupt service routine at the location pointed to by the vector

If no other interrupt request is pending at the end of the interrupt service routine, an RTI instruction recovers the stacked values. Program execution resumes program at the return address.

If another interrupt request is pending at the end of an interrupt service routine, the RTI instruction recovers the stacked values. However, the CPU then:

- Adjusts the stack pointer to point again at the stacked CCR location, SP – 9
- Fetches the vector of the pending interrupt
- Begins execution of the interrupt service routine at the location pointed to by the vector

Chapter 5
Operating Modes and Resource Mapping

5.1 Introduction

The MCU can operate in eight different modes. Each mode has a different default memory map and external bus configuration. After reset, most system resources can be mapped to other addresses by writing to the appropriate control registers.

5.2 Operating Modes

The states of the BKGD, MODB, and MODA pins during reset determine the operating mode after reset.

The SMODN, MODB, and MODA bits in the MODE register show the current operating mode and provide limited mode switching during operation. The states of the BKGD, MODB, and MODA pins are latched into these bits on the rising edge of the reset signal.

Table 5-1. Mode Selection

BKGD	MODB	MODA	Mode	Port A Port B	Port C	Port D
0	0	0	Special single-chip	G.P.(1) I/O	G.P. I/O	G.P. I/O
0	0	1	Special expanded narrow	ADDR	DATA	G.P. I/O
0	1	0	Special peripheral	ADDR	DATA	DATA
0	1	1	Special expanded wide	ADDR	DATA	DATA
1	0	0	Normal single chip	G.P. I/O	G.P. I/O	G.P. I/O
1	0	1	Normal expanded narrow	ADDR	DATA	G.P. I/O
1	1	0	Reserved (forced to peripheral)	—	—	—
1	1	1	Normal expanded wide	ADDR	DATA	DATA

1. G.P. = General purpose

The two basic types of operating modes are:

- Normal modes — Some registers and bits are protected against accidental changes.
- Special modes — Protected control registers and bits are allowed greater access for special purposes such as testing and emulation.

A system development and debug feature, background debug mode (BDM), is available in all modes. In special single-chip mode, BDM is active immediately after reset.

5.2.1 Normal Operating Modes

These modes provide three operating configurations. Background debugging is available in all three modes, but must first be enabled for some operations by means of a BDM command. BDM can then be made active by another BDM command.

5.2.1.1 Normal Expanded Wide Mode

The 16-bit external address bus uses port A for the high byte and port B for the low byte. The 16-bit external data bus uses port C for the high byte and port D for the low byte.

5.2.1.2 Normal Expanded Narrow Mode

The 16-bit external address bus uses port A for the high byte and port B for the low byte. The 8-bit external data bus uses port C. In this mode, 16-bit data is presented high byte first, followed by the low byte. The address is automatically incremented on the second cycle.

5.2.1.3 Normal Single-Chip Mode

There are no external buses in normal single-chip mode. The MCU operates as a stand-alone device and all program and data resources are on-chip. Port pins can be used for general-purpose I/O (input/output).

5.2.2 Special Operating Modes

Special operating modes are commonly used in factory testing and system development.

5.2.2.1 Special Expanded Wide Mode

This mode is for emulation of normal expanded wide mode and emulation of normal single-chip mode with a 16-bit bus. The bus-control pins of port E are all configured for their bus-control output functions rather than general-purpose I/O.

5.2.2.2 Special Expanded Narrow Mode

This mode is for emulation of normal expanded narrow mode. External 16-bit data is handled as two back-to-back bus cycles, one for the high byte followed by one for the low byte. Internal operations continue to use full 16-bit data paths.

For development purposes, port D can be made available for visibility of 16-bit internal accesses by setting the EMD and IVIS control bits.

5.2.2.3 Special Single-Chip Mode

This mode can be used to force the MCU to active BDM mode to allow system debug through the BKGD pin. There are no external address and data buses in this mode. The MCU operates as a stand-alone device and all program and data space are on-chip. External port pins can be used for general-purpose I/O.

5.2.2.4 Special Peripheral Mode

The CPU is not active in this mode. An external master can control on-chip peripherals for testing purposes. It is not possible to change to or from this mode without going through reset. Background debugging should not be used while the MCU is in special peripheral mode as internal bus conflicts between BDM and the external master can cause improper operation of both modes.

5.2.3 Background Debug Mode

Background debug mode (BDM) is an auxiliary operating mode that is used for system development. BDM is implemented in on-chip hardware and provides a full set of debug operations. Some BDM

commands can be executed while the CPU is operating normally. Other BDM commands are firmware based and require the BDM firmware to be enabled and active for execution.

In special single-chip mode, BDM is enabled and active immediately out of reset. BDM is available in all other operating modes, but must be enabled before it can be activated. BDM should not be used in special peripheral mode because of potential bus conflicts.

Once enabled, background mode can be made active by a serial command sent via the BKGD pin or execution of a CPU12 BGND instruction. While background mode is active, the CPU can interpret special debugging commands, and read and write CPU registers, peripheral registers, and locations in memory.

While BDM is active, the CPU executes code located in a small on-chip ROM mapped to addresses $FF20 to $FFFF, and BDM control registers are accessible at addresses $FF00 to $FF06. The BDM ROM replaces the regular system vectors while BDM is active. While BDM is active, the user memory from $FF00 to $FFFF is not in the map except through serial BDM commands.

5.3 Internal Resource Mapping

The internal register block, RAM, and EEPROM have default locations within the 64-Kbyte standard address space but may be reassigned to other locations during program execution by setting bits in mapping registers INITRG, INITRM, and INITEE. During normal operating modes, these registers can be written once. It is advisable to explicitly establish these resource locations during the initialization phase of program execution, even if default values are chosen, to protect the registers from inadvertent modification later.

Writes to the mapping registers go into effect between the cycle that follows the write and the cycle after that. To assure that there are no unintended operations, a write to one of these registers should be followed with a NOP (no operation) instruction.

If conflicts occur when mapping resources, the register block takes precedence over the other resources; RAM or EEPROM addresses occupied by the register block are not available for storage. When active, BDM ROM takes precedence over other resources although a conflict between BDM ROM and register space is not possible. Table 5-2 shows resource mapping precedence.

Table 5-2. Mapping Precedence

Precedence	Resource
1	BDM ROM (if active)
2	Register space
3	RAM
4	EEPROM
5	External memory

All address space not used by internal resources is external memory by default.

The memory expansion module manages three memory overlay windows:

1. Program
2. Data
3. One extra page overlay

The sizes and locations of the program and data overlay windows are fixed. One of two locations can be selected for the extra page (EPAGE).

5.4 Mode and Resource Mapping Registers

This section describes the mode and resource mapping registers.

5.4.1 Mode Register

MODE controls the MCU operating mode and various configuration options. This register is not in the map in peripheral mode.

Address: $000B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SMODN	MODB	MODA	ESTR	IVIS	0	EMD	EME
Reset States								
Special single-chip:	0	0	0	1	1	0	1	1
Special expanded narrow:	0	0	1	1	1	0	1	1
Special peripheral:	0	1	0	1	1	0	1	1
Special expanded wide:	0	1	1	1	1	0	1	1
Normal single-chip:	1	0	0	1	0	0	0	0
Normal expanded narrow:	1	0	1	1	0	0	0	0
Normal expanded wide:	1	1	1	1	0	0	0	0

Figure 5-1. Mode Register (MODE)

Read: Anytime
Write: Varies from bit to bit

SMODN, MODB, and MODA — Mode Select Special, B, and A Bits

These bits show the current operating mode and reflect the status of the BKGD, MODB, and MODA input pins at the rising edge of reset.

SMODN can be written only if SMODN = 0 (in special modes) but the first write is ignored. MODB and MODA may be written once if SMODN = 1; anytime if SMODN = 0, except that special peripheral and reserved modes cannot be selected.

ESTR — E-Clock Stretch Enable Bit

ESTR determines if the E-clock behaves as a simple free-running clock or as a bus control signal that is active only for external bus cycles.

1 = E stretches high during external access cycles and low during non-visible internal accesses.
0 = E never stretches (always free running).

Normal modes: Write once
Special modes: Write anytime

IVIS — Internal Visibility Bit

IVIS determines whether internal ADDR, DATA, R/$\overline{W}$, and $\overline{LSTRB}$ signals can be seen on the external bus during accesses to internal locations. If this bit is set in special narrow mode and EMD = 1 when an internal access occurs, the data appears wide on port C and port D. This allows for emulation. Visibility is not available when the part is operating in a single-chip mode.

1 = Internal bus operations are visible on external bus.
0 = Internal bus operations are not visible on external bus.

Normal modes: Write once
Special modes: Write anytime except the first time

EMD — Emulate Port D Bit

This bit only has meaning in special expanded narrow mode.

In expanded wide modes and special peripheral mode, PORTD, DDRD, KWIED, and KWIFD are removed from the memory map regardless of the state of this bit.

In single-chip modes and normal expanded narrow mode, PORTD, DDRD, KWIED, and KWIFD are in the memory map regardless of the state of this bit.

1 = If in special expanded narrow mode, PORTD, DDRD, KWIED, and KWIFD are removed from the memory map. Removing the registers from the map allows the user to emulate the function of these registers externally.
0 = PORTD, DDRD, KWIED, and KWIFD are in the memory map.

Normal modes: Write once
Special modes: Write anytime except the first time

EME — Emulate Port E Bit

In single-chip mode, PORTE and DDRE are always in the map regardless of the state of this bit.

1 = If in an expanded mode, PORTE and DDRE are removed from the internal memory map. Removing the registers from the map allows the user to emulate the function of these registers externally.
0 = PORTE and DDRE in the memory map

Normal modes: Write once
Special modes: Write anytime except the first time

5.4.2 Register Initialization Register

After reset, the 512-byte register block resides at location $0000 but can be reassigned to any 2-Kbyte boundary within the standard 64-Kbyte address space. Mapping of internal registers is controlled by five bits in the INITRG register. The register block occupies the first 512 bytes of the 2-Kbyte block.

Address: $0011

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	REG15	REG14	REG13	REG12	REG11	0	0	0
Reset:	0	0	0	0	0	0	0	0

Figure 5-2. Register Initialization Register (INITRG)

Read: Anytime
Write: Once in normal modes; anytime in special modes

REG15–REG11 — Register Position Bits

These bits specify the upper five bits of the 16-bit register address.

5.4.3 RAM Initialization Register

After reset, addresses of the 1-Kbyte RAM array begin at location $0800 but can be assigned to any 2-Kbyte boundary within the standard 64-Kbyte address space. Mapping of internal RAM is controlled by five bits in the INITRM register. The RAM array occupies the last 1 Kbyte of the 2-Kbyte block.

Address: $0010

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	RAM15	RAM14	RAM13	RAM12	RAM11	0	0	0
Reset:	0	0	0	0	1	0	0	0

Figure 5-3. RAM Initialization Register (INITRM)

Read: Anytime
Write: Once in normal modes; anytime in special modes

RAM15–RAM11 — RAM Position Bits
These bits specify the upper five bits of the 16-bit RAM address.

5.4.4 EEPROM Initialization Register

The MCU has 4 Kbytes of EEPROM which is activated by the EEON bit in the INITEE register.

Mapping of internal EEPROM is controlled by four bits in the INITEE register. After reset, EEPROM address space begins at location $1000 but can be mapped to any 4-Kbyte boundary within the standard 64-Kbyte address space.

Address: $0012

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EE15	EE14	EE13	EE12	0	0	0	EEON
Reset:								
Expanded and peripheral:	0	0	0	1	0	0	0	1
Single-chip:	1	1	1	1	0	0	0	1

Figure 5-4. EEPROM Initialization Register (INITEE)

Read: Anytime
Write: Varies from bit to bit

EE15–EE12 — EEPROM Position Bits
These bits specify the upper four bits of the 16-bit EEPROM address.
Normal modes: Write once
Special modes: Write anytime

EEON — EEPROM On Bit
EEON enables the on-chip EEPROM. EEON is forced to 1 in single-chip modes.
Write anytime
1 = EEPROM at address selected by EE15–EE12
0 = EEPROM removed from memory map

5.4.5 Miscellaneous Mapping Control Register

Additional mapping controls are available that can be used in conjunction with memory expansion and chip selects.

To use memory expansion, the part must be operated in one of the expanded modes. Sections of the standard 64-Kbyte memory map have memory expansion windows which allow more than 64 Kbytes to be addressed externally. Memory expansion consists of three memory expansion windows and six address lines in addition to the existing standard 16 address lines. The memory expansion function reuses as many as six of the standard 16 address lines. Usage of chip selects identifies the source of the internal address.

All of the memory expansion windows have a fixed size and two of them have a fixed address location. The third has two selectable address locations.

Address: $0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EWDIR	NDRC	0	0	0	0	0	0
Reset:	0	0	0	0	0	0	0	0

Figure 5-5. Miscellaneous Mapping Control Register (MISC)

Read: Anytime
Write: Once in normal modes; anytime in special modes

EWDIR — Extra Window Positioned in Direct Space Bit

This bit is only valid in expanded modes. If the EWEN bit in the WINDEF register is cleared, then this bit has no meaning or effect.

1 = If EWEN is set, then a 1 in this bit places the EPAGE at $0000–$03FF.
0 = If EWEN is set, then a 0 in this bit places the EPAGE at $0400–$07FF.

NDRC — Narrow Data Bus for Register Chip-Select Space Bit

This function requires at least one of the chip selects CS3–CS0 to be enabled. It effects the external 512-byte memory space.

1 = Makes the register-following chip-selects (2, 1, 0, and sometimes 3) active space (512-byte block) act the same as an 8-bit only external data bus. Data only goes through port C externally. This allows 8-bit and 16-bit external memory devices to be mixed in a system.
0 = Makes the register-following chip-select active space act as a full 16-bit data bus. In the narrow (8-bit) mode, NDRC has no effect.

5.5 Memory Map

Figure 5-6 illustrates the memory map for each mode of operation immediately after reset.

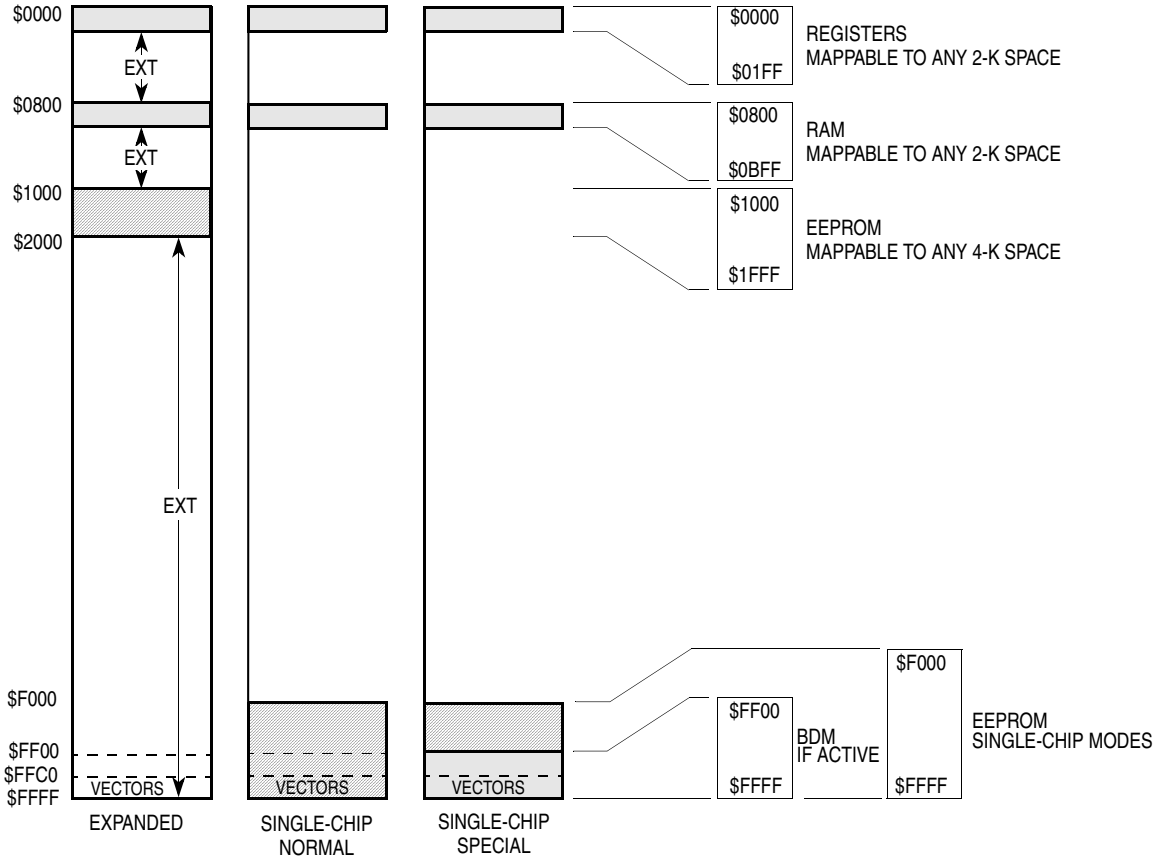


Figure 5-6. Memory Map

Chapter 6
Bus Control and Input/Output (I/O)

6.1 Introduction

Internally the MCU has full 16-bit data paths, but depending upon the operating mode and control registers, the external bus may be 8 or 16 bits. There are cases where 8-bit and 16-bit accesses can appear on adjacent cycles using the $\overline{\text{LSTRB}}$ signal to indicate 8-bit or 16-bit data.

6.2 Detecting Access Type from External Signals

The external signals $\overline{\text{LSTRB}}$, R/$\overline{\text{W}}$, and A0 can be used to determine the type of bus access that is taking place. Accesses to the internal RAM module are the only type of access that produce $\overline{\text{LSTRB}}$ = A0 = 1, because the internal RAM is specifically designed to allow misaligned 16-bit accesses in a single cycle. In these cases, the data for the address that was accessed is on the low half of the data bus and the data for address +1 is on the high half of the data bus.

Table 6-1. Access Type versus Bus Control Pins

$\overline{\text{LSTRB}}$	A0	R/$\overline{\text{W}}$	Type of Access
1	0	1	8-bit read of an even address
0	1	1	8-bit read of an odd address
1	0	0	8-bit write of an even address
0	1	0	8-bit write of an odd address
0	0	1	16-bit read of an even address
1	1	1	16-bit read of an odd address (low/high data swapped)
0	0	0	16-bit write to an even address
1	1	0	16-bit write to an even address (low/high data swapped)

6.3 Registers

Not all registers are visible in the memory map under certain conditions. In special peripheral mode, the first 16 registers associated with bus expansion are removed from the memory map.

In expanded modes, some or all of port A, port B, port C, port D, and port E are used for expansion buses and control signals. To allow emulation of the single-chip functions of these ports, some of these registers must be rebuilt in an external port replacement unit. In any expanded mode, port A, port B, and port C are used for address and data lines so registers for these ports, as well as the data direction registers for these ports, are removed from the on-chip memory map and become external accesses.

Port D and its associated data direction register may be removed from the on-chip map when port D is needed for 16-bit data transfers. If the MCU is in an expanded wide mode, port C and port D are used for 16-bit data and the associated port and data direction registers become external accesses. When the MCU is in expanded narrow mode, the external data bus is normally 8 bits. To allow full-speed operation while allowing visibility of internal 16-bit accesses, a 16-bit-wide data path is required. The emulate port D (EMD) control bit in the MODE register may be set to allow such 16-bit transfers. In this case of narrow special expanded mode and the EMD bit set, port D and data direction D registers are removed from the on-chip memory map and become external accesses so port D may be rebuilt externally.

In any expanded mode, port E pins may be needed for bus control (for instance, ECLK and R/$\overline{W}$). To regain the single-chip functions of port E, the emulate port E (EME) control bit in the MODE register may be set. In this special case of expanded mode and EME set, PORTE and DDRE registers are removed from the on-chip memory map and become external accesses so port E may be rebuilt externally.

6.3.1 Port A Data Register

Address: $0000

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Reset:	0	0	0	0	0	0	0	0
Expanded and peripheral:	ADDR15	ADDR14	ADDR13	ADDR12	ADDR11	ADDR10	ADDR9	ADDR8

Figure 6-1. Port A Data Register (PORTA)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

Bits PA7–PA0 are associated with addresses ADDR15–ADDR8 respectively. When this port is not used for external addresses such as in single-chip mode, these pins can be used as general-purpose I/O. DDRA determines the primary direction of each pin. This register is not in the on-chip map in expanded and peripheral modes.

6.3.2 Port A Data Direction Register

Address: $0002

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
Reset:	0	0	0	0	0	0	0	0

Figure 6-2. Port A Data Direction Register (DDRA)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

This register determines the primary direction for each port A pin when functioning as a general-purpose I/O port. DDRA is not in the on-chip map in expanded and peripheral modes.

1 = Associated pin is an output.
0 = Associated pin is a high-impedance input.

6.3.3 Port B Data Register

Address: $0001

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
Reset:	0	0	0	0	0	0	0	0
Expanded and peripheral:	ADDR7	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0

Figure 6-3. Port B Data Register (PORTB)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

Bits PB7–PB0 correspond to address lines ADDR7–ADDR0. When this port is not used for external addresses such as in single-chip mode, these pins can be used as general-purpose I/O. DDRB determines the primary direction of each pin. This register is not in the on-chip map in expanded and peripheral modes.

6.3.4 Port B Data Direction Register

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Reset:	0	0	0	0	0	0	0	0

Figure 6-4. Port B Data Direction Register (DDRB)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

This register determines the primary direction for each port B pin when functioning as a general-purpose I/O port. DDRB is not in the on-chip map in expanded and peripheral modes.

1 = Associated pin is an output.
0 = Associated pin is a high-impedance input.

6.3.5 Port C Data Register

Address: $0004

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Reset:	0	0	0	0	0	0	0	0
Expanded wide and peripheral:	DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8
Expanded narrow:	DATA15/7	DATA14/6	DATA13/5	DATA12/4	DATA11/3	DATA10/2	DATA9/1	DATA8/0

Figure 6-5. Port C Data Register (PORTC)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

Bits PC7–PC0 correspond to data lines DATA15–DATA8. When this port is not used for external data such as in single-chip mode, these pins can be used as general-purpose I/O. DDRC determines the primary direction for each pin. In narrow expanded modes, DATA15–DATA8 and DATA7–DATA0 are multiplexed into the MCU through port C pins on successive cycles. This register is not in the on-chip map in expanded and peripheral modes.

When the MCU is operating in special expanded narrow mode and port C and port D are being used for internal visibility, internal accesses produce full 16-bit information with DATA15–DATA8 on port C and DATA7–DATA0 on port D. This allows the MCU to operate at full speed while making 16-bit access information available to external development equipment in a single cycle. In this narrow mode, normal 16-bit accesses to external memory get split into two successive 8-bit accesses on port C alone.

6.3.6 Port C Data Direction Register

Address: $0006

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
Reset:	0	0	0	0	0	0	0	0

Figure 6-6. Port C Data Direction Register (DDRC)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

DDRC is not in the on-chip map in expanded and peripheral modes.

This register determines the primary direction for each port C pin when functioning as a general-purpose I/O port.

1 = Associated pin is an output.
0 = Associated pin is a high-impedance input.

6.3.7 Port D Data Register

Address: $0005

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
Reset:	0	0	0	0	0	0	0	0
Expanded wide and peripheral:	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
Alternate pin function:	KWD7	KWD6	KWD5	KWD4	KWD3	KWD2	KWD1	KWD0

Figure 6-7. Port D Data Register (PORTD)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

Bits PD7–PD0 correspond to data lines DATA7–DATA0. When port D is not used for external data, such as in single-chip mode, these pins can be used as general-purpose I/O or key wakeup signals. DDRD determines the primary direction of each port D pin.

In special expanded narrow mode, the external data bus is normally limited to eight bits on port C, but the emulate port D bit (EMD) in the MODE register can be set to allow port C and port D to be used together to provide single-cycle visibility of internal 16-bit accesses for debugging purposes. If the mode is special narrow expanded and EMD is set, port D is configured for DATA7–DATA0 of visible internal accesses and normal 16-bit external accesses are split into two adjacent 8-bit accesses through port C. This allows connection of a single 8-bit external program memory.

This register is not in the on-chip map in wide expanded and peripheral modes. Also, in special narrow expanded mode, the function of this port is determined by the EMD control bit. If EMD is set, this register is not in the on-chip map and port D is used for DATA7–DATA0 of visible internal accesses. If EMD is clear, this port serves as general-purpose I/O or key wakeup signals.

6.3.8 Port D Data Direction Register

Address: $0007

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 6-8. Port D Data Direction Register (DDRD)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

When port D is operating as a general-purpose I/O port, this register determines the primary direction for each port D pin.

1 = Associated pin is an output.
0 = Associated pin is a high-impedance input.

This register is not in the map in wide expanded and peripheral modes. Also, in special narrow expanded mode, the function of this port is determined by the EMD control bit. If EMD is set, this register is not in the on-chip map and port D is used for DATA7–DATA0 of visible internal accesses. If EMD is clear, this port serves as general-purpose I/O or key wakeup signals.

6.3.9 Port E Data Register

Address: $0008

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
Reset:	Unaffected by reset							
Normal narrow expanded:	0	0	0	0	1	0	0	0
All other modes:	0	0	0	0	0	0	0	0
Alternate pin function:	ARST	MODB or IPIPE1	MODA or IPIPE0	ECLK	LSTRB	R/$\overline{W}$	$\overline{IRQ}$	$\overline{XIRQ}$

Figure 6-9. Port E Data Register (PORTE)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

This register is associated with external bus control signals and interrupt inputs including auxiliary reset (ARST), mode select (MODB/IPIPE1, MODA/IPIPE0), E-clock, size ($\overline{LSTRB}$), read/write (R/$\overline{W}$), $\overline{IRQ}$, and $\overline{XIRQ}$. When the associated pin is not used for one of these specific functions, the pin can be used as general-purpose I/O. The port E assignment register (PEAR) selects the function of each pin. DDRE determines the primary direction of each port E pin when configured to be general-purpose I/O.

Some of these pins have software selectable pullups ($\overline{LSTRB}$, R/$\overline{W}$, and $\overline{XIRQ}$). A single control bit enables the pullups for all these pins which are configured as inputs. $\overline{IRQ}$ always has a pullup.

PE7 can be selected as a high-true auxiliary reset input.

This register is not in the map in peripheral mode or expanded modes when the EME bit is set.

6.3.10 Port E Data Direction Register

Address: $0009

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
Reset:	0	0	0	0	0	0	1	1
Normal narrow expanded:	0	0	0	0	1	0	0	0
All other modes:	0	0	0	0	0	0	0	0

Figure 6-10. Port E Data Direction Register (DDRE)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

This register determines the primary direction for each port E pin configured as general-purpose I/O.

1 = Associated pin is an output.
0 = Associated pin is a high-impedance input.

PE1 and PE0 are associated with $\overline{XIRQ}$ and $\overline{IRQ}$ and cannot be configured as outputs. These pins can be read regardless of whether the alternate interrupt functions are enabled.

This register is not in the map in peripheral mode and expanded modes while the EME control bit is set.

6.3.11 Port E Assignment Register

Address: $000A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	ARSIE	PLLTE	PIPOE	NECLK	LSTRE	RDWE	0	0
Reset:								
Special single-chip:	0	0	1	0	1	1	0	0
Special expanded narrow:	0	0	1	0	1	1	0	0
Peripheral:	0	1	0	1	0	0	0	0
Special expanded wide:	0	0	1	0	1	1	0	0
Normal single-chip	0	0	0	1	0	0	0	0
Normal expanded narrow:	0	0	0	0	0	0	0	0
Normal expanded wide:	0	0	0	0	0	0	0	0

Figure 6-11. Port E Assignment Register (PEAR)

Read: Anytime, if register is in the map
Write: Varies from bit to bit if register is in the map

The PEAR register selects between the general-purpose I/O functions and the alternate bus-control functions of port E. The alternate bus-control functions override the associated DDRE bits.

The reset condition of this register depends on the mode of operation.

- In normal single-chip mode, port E is general-purpose I/O.
- In special single-chip mode, the E-clock is enabled as a timing reference, and the rest of port E is general-purpose I/O.
- In normal expanded modes, the E-clock is configured for its alternate bus-control function, and the other bits of port E are general-purpose I/O. The reset vector is located in external memory and the E-clock may be required for this access. If R/$\overline{W}$ is needed for external writable resources, PEAR can be written during normal expanded modes.
- In special expanded modes, IPIPE1, IPIPE0, E, R/$\overline{W}$, and $\overline{LSTRB}$ are configured as bus-control signals.

In peripheral mode, the PEAR register is not accessible for reads or writes. However, the PLLTE control bit is cleared to configure PE6 as a test output from the PLL module.

ARSIE — Auxiliary Reset Input Enable Bit

Write anytime.

1 = PE7 is a high-true reset input; reset timing is the same as that of the low-true $\overline{RESET}$ pin.
0 = PE7 is general-purpose I/O.

PLLTE — PLL Testing Enable Bit

Normal modes: Write never
Special modes: Write anytime except the first time

1 = PE6 is a test signal output from the PLL module (no effect in single-chip or normal expanded modes); PIPOE = 1 overrides this function and forces PE6 to be a pipe status output signal.
0 = PE6 is general-purpose I/O or pipe output.

PIPOE — Pipe Status Signal Output Enable Bit

Normal modes: Write once
Special modes: Write anytime except the first time

1 = PE6 and PE5 are outputs and indicate the state of the instruction queue; no effect in single-chip modes.
0 = PE6 and PE5 are general-purpose I/O; if PLLTE = 1, PE6 is a test output signal from the PLL module.

NECLK — No External E Clock Bit

Normal modes: Write anytime
Special modes: Write never
In peripheral mode, E is an input; in all other modes, E is an output.

1 = PE4 is a general-purpose I/O pin.
0 = PE4 is the external E-clock pin. To get a free-running E-clock in single-chip modes, use NECLK = 0 and IVIS = 1. A 16-bit write to PEAR:MODE can configure these bits in one operation.

LSTRE — Low Strobe ($\overline{\text{LSTRB}}$) Enable Bit

Normal modes: Write once
Special modes: Write anytime except the first time
LSTRE has no effect in single-chip or normal expanded narrow modes.

1 = PE3 is configured as the $\overline{\text{LSTRB}}$ bus-control output, except in single-chip or normal expanded narrow modes.
0 = PE3 is a general-purpose I/O pin.

$\overline{\text{LSTRB}}$ is for external writes. After reset in normal expanded mode, $\overline{\text{LSTRB}}$ is disabled. If needed, it must be enabled before external writes. External reads do not normally need $\overline{\text{LSTRB}}$ because all 16 data bits can be driven even if the MCU only needs eight bits of data.
In normal expanded narrow mode, this pin is reset to an output driving high allowing the pin to be an output while in and immediately after reset.

RDWE — Read/Write Enable Bit

Normal modes: Write once
Special modes: Write anytime except the first time
RDWE has no effect in single-chip modes.

1 = PE2 is configured as the R/$\overline{\text{W}}$ pin. In single-chip modes, RDWE has no effect and PE2 is a general-purpose I/O pin.
0 = PE2 is a general-purpose I/O pin.

R/$\overline{\text{W}}$ is used for external writes. After reset in normal expanded mode, it is disabled. If needed, it must be enabled before any external writes.

6.3.12 Pullup Control Register

Address: $000C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PUPH	PUPG	PUPF	PUPE	PUPD	PUC	PUPB	PUPA
Reset:	1	1	1	1	1	1	1	1

Figure 6-12. Pullup Control Register (PUCR)

Read: Anytime, if register is in the map
Write: Anytime, if register is in the map

This register is not in the map in peripheral mode.

These bits select pullup resistors for any pin in the corresponding port that is currently configured as an input.

PUPH — Pullup Port H Enable Bit
1 = Enable pullup devices for all port H input pins
0 = Port H pullups disabled

PUPG — Pullup Port G Enable Bit
1 = Enable pullup devices for all port G input pins
0 = Port G pullups disabled

PUPF — Pullup Port F Enable Bit
1 = Enable pullup devices for all port F input pins
0 = Port F pullups disabled

PUPE — Pullup Port E Enable Bit
1 = Enable pullup devices for port E input pins PE3, PE2, and PE0
0 = Port E pullups on PE3, PE2, and PE0 disabled

PUPD — Pullup Port D Enable Bit
1 = Enable pullup devices for all port D input pins
0 = Port D pullups disabled
This bit has no effect if port D is being used as part of the data bus (the pullups are inactive).

PUPC — Pullup Port C Enable Bit
1 = Enable pullup devices for all port C input pins
0 = Port C pullups disabled
This bit has no effect if port C is being used as part of the data bus (the pullups are inactive).

PUPB — Pullup Port B Enable Bit
1 = Enable pullup devices for all port B input pins
0 = Port B pullups disabled
This bit has no effect if port B is being used as part of the address bus (the pullups are inactive).

PUPA — Pullup Port A Enable Bit
1 = Enable pullup devices for all port A input pins
0 = Port A pullups disabled
This bit has no effect if port A is being used as part of the address bus (the pullups are inactive).

6.3.13 Reduced Drive Register

Address: $000D

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	RDPJ	RDPH	RDPG	RDPF	RDPE	PRPD	RDPC	RDPAB
Reset:	0	0	0	0	0	0	0	0

Figure 6-13. Reduced Drive Register (RDRIV)

Read: Anytime, if register is in the map
Write: Anytime, in normal modes; never in special modes

This register is not in the map in peripheral mode.

These bits select reduced drive for the associated port pins. This gives reduced power consumption and reduced RFI with a slight increase in transition time (depending on loading). The reduced drive function is independent of which function is being used on a particular port.

RDPJ — Reduced Drive of Port J Bit
1 = Reduced drive for all port J output pins
0 = Full drive for all port J output pins

RDPH — Reduced Drive of Port H Bit
1 = Reduced drive for all port H output pins
0 = Full drive for all port H output pins

RDPG — Reduced Drive of Port G Bit
1 = Reduced drive for all port G output pins
0 = Full drive for all port G output pins

RDPF — Reduced Drive of Port F Bit
1 = Reduced drive for all port F output pins
0 = Full drive for all port F output pins

RDPE — Reduced Drive of Port E Bit
1 = Reduced drive for all port E output pins
0 = Full drive for all port E output pins

RDPD — Reduced Drive of Port D Bit
1 = Reduced drive for all port D output pins
0 = Full drive for all port D output pins

RDPC — Reduced Drive of Port C Bit
1 = Reduced drive for all port C output pins
0 = Full drive for all port C output pins

RDPAB — Reduced Drive of Port A and Port B Bit
1 = Reduced drive for all port A and port B output pins
0 = Full drive for all port A and port B output pins

Chapter 7
EEPROM

7.1 Introduction

The MC68HC812A4 EEPROM (electrically erasable, programmable, read-only memory) serves as a 4096-byte nonvolatile memory which can be used for frequently accessed static data or as fast access program code. Operating system kernels and standard subroutines would benefit from this feature.

The MC68HC812A4 EEPROM is arranged in a 16-bit configuration. The EEPROM array may be read as either bytes, aligned words, or misaligned words. Access times are one bus cycle for byte and aligned word access and two bus cycles for misaligned word operations.

Programming is by byte or aligned word. Attempts to program or erase misaligned words will fail. Only the lower byte will be latched and programmed or erased. Programming and erasing of the user EEPROM can be done in all modes.

Each EEPROM byte or aligned word must be erased before programming. The EEPROM module supports byte, aligned word, row (32 bytes), or bulk erase, all using the internal charge pump. Bulk erasure of odd and even rows is also possible in test modes; the erased state is $FF. The EEPROM module has hardware interlocks which protect stored data from corruption by accidentally enabling the program/erase voltage. Programming voltage is derived from the internal V_{DD} supply with an internal charge pump. The EEPROM has a minimum program/erase life of 10,000 cycles over the complete operating temperature range.

7.2 EEPROM Programmer's Model

The EEPROM module consists of two separately addressable sections. The first is a 4-byte memory mapped control register block used for control, testing and configuration of the EEPROM array. The second section is the EEPROM array itself.

At reset, the 4-byte register section starts at address $00F0 and the EEPROM array is located from addresses $1000 to $1FFF (see Figure 7-1). For information on remapping the register block and EEPROM address space, refer to Chapter 5 Operating Modes and Resource Mapping.

Read/write access to the memory array section can be enabled or disabled by the EEON control bit in the INITEE register. This feature allows the access of memory mapped resources that have lower priority than the EEPROM memory array. EEPROM control registers can be accessed and EEPROM locations may be programmed or erased regardless of the state of EEON.

Using the normal EEPROG control, it is possible to continue program/erase operations during wait. For lowest power consumption during wait, stop program/erase by turning off EEPGM.

If the stop mode is entered during programming or erasing, program/erase voltage is automatically turned off and the RC clock (if enabled) is stopped. However, the EEPGM control bit remains set. When stop mode is terminated, the program/erase voltage automatically turns back on if EEPGM is set.

At low bus frequencies, the RC clock must be turned on for program/erase.

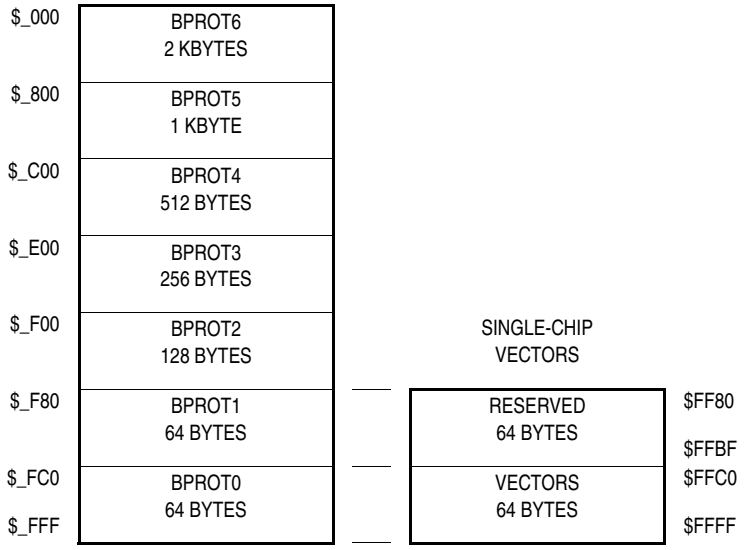


Figure 7-1. EEPROM Block Protect Mapping

7.3 EEPROM Control Registers

This section describes the EEPROM control registers.

7.3.1 EEPROM Module Configuration Register

Address: $00F0

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	1	1	1	1	1	EESWAI	PROTLCK	EERC
Write:								
Reset:	1	1	1	1	1	1	0	0

Figure 7-2. EEPROM Module Configuration Register (EEMCR)

Read: Anytime
Write: Varies from bit to bit

EESWAI — EEPROM Stops in Wait Mode Bit

0 = Module is not affected during wait mode.
1 = Module ceases to be clocked during wait mode.

This bit should be cleared if the wait mode vectors are mapped in the EEPROM array.

PROTLCK — Block Protect Write Lock Bit

0 = Block protect bits and bulk erase protection bit can be written.
1 = Block protect bits are locked.

Write once in normal modes (SMODN = 1). Set and clear anytime in special modes (SMODN = 0).

EERC — EEPROM Charge Pump Clock Bit

0 = System clock is used as clock source for the internal charge pump; internal RC oscillator is stopped.
1 = Internal RC oscillator drives the charge pump; RC oscillator is required when the system bus clock is lower than f_{PROG}.

Write: Anytime

7.3.2 EEPROM Block Protect Register

Address: $00F1

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	1	BPROT6	BPROT5	BPROT4	BPROT3	BPROT2	BPROT1	BPROT0
Reset:	1	1	1	1	1	1	1	1

Figure 7-3. EEPROM Block Protect Register (EEPROT)

Read: Anytime
Write: Anytime if EEPGM = 0 and PROTLCK = 0

This register prevents accidental writes to EEPROM.

BPROT6–BPROT0 — EEPROM Block Protection Bit

0 = Associated EEPROM block can be programmed and erased.
1 = Associated EEPROM block is protected from being programmed and erased.
These bits cannot be modified while programming is taking place (EEPGM = 1).

Table 7-1. 4-Kbyte EEPROM Block Protection

Bit Name	Block Protected	Block Size
BPROT6	$1000 to $17FF	2048 bytes
BPROT5	$1800 to $1BFF	1024 bytes
BPROT4	$1C00 to $1DFF	512 bytes
BPROT3	$1E00 to $1EFF	256 bytes
BPROT2	$1F00 to $1F7F	128 bytes
BPROT1	$1F80 to $1FBF	64 bytes
BPROT0	$1FC0 to $1FFF	64 bytes

7.3.3 EEPROM Test Register

Address: $00F2

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	EEODD	EEVEN	MARG	EECPD	EECPRD	0	EECPM	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 7-4. EEPROM Test Register (EEPROT)

Read: Anytime
Write: In special modes only (SMODN = 0)

These bits are used for test purposes only. In normal modes, the bits are forced to 0.

EEODD — Odd Row Programming Bit

1 = Bulk program/erase all odd rows
0 = Odd row bulk programming/erasing disabled

EEVEN — Even Row Programming Bit

1 = Bulk program/erase all even rows
0 = Even row bulk programming/erasing disabled

MARG — Program and Erase Voltage Margin Test Enable Bit

1 = Program and erase margin test
0 = Normal operation

This bit is used to evaluate the program/erase voltage margin.

EECPD — Charge Pump Disable Bit

1 = Disable charge pump
0 = Charge pump is turned on during program/erase

EECPRD — Charge Pump Ramp Disable Bit

1 = Disable charge pump controlled ramp up
0 = Charge pump is turned on progressively during program/erase

This bit is known to enhance write/erase endurance of EEPROM cells.

ECPM — Charge Pump Monitor Enable Bit

1 = Output the charge pump voltage on the $\overline{IRQ}/V_{PP}$ pin
0 = Normal operation

7.3.4 EEPROM Programming Register

Address: $00F3

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	BULKP	0	0	BYTE	ROW	ERASE	EELAT	EEPGM
Reset:	1	0	0	0	0	0	0	0

Figure 7-5. EEPROM Programming Register (EEPROG)

Read: Anytime
Write: Varies from bit to bit

BULKP — Bulk Erase Protection Bit

1 = EEPROM protected from bulk or row erase
0 = EEPROM can be bulk erased.

Write anytime, if EEPGM = 0 and PROTLCK = 0

BYTE — Byte and Aligned Word Erase Bit

1 = One byte or one aligned word erase only
0 = Bulk or row erase enabled

Write anytime, if EEPGM = 0

ROW — Row or Bulk Erase Bit (when BYTE = 0)

1 = Erase only one 32-byte row
0 = Erase entire EEPROM array

Write anytime, if EEPGM = 0

BYTE and ROW have no effect when ERASE = 0.

If BYTE = 1 and test mode is not enabled, only the location specified by the address written to the programming latches is erased. The operation is a byte or an aligned word erase depending on the size of written data.

Table 7-2. Erase Selection

Byte	Row	Block Size
0	0	Bulk erase entire EEPROM array
0	1	Row erase 32 bytes
1	0	Byte or aligned word erase
1	1	Byte or aligned word erase

ERASE — Erase Control Bit

1 = EEPROM configuration for erasure
0 = EEPROM configuration for programming

Write anytime, if EEPGM = 0

This bit configures the EEPROM for erasure or programming.

EELAT — EEPROM Latch Control Bit

1 = EEPROM address and data bus latches set up for programming or erasing
0 = EEPROM set up for normal reads

Write: Anytime, if EEPGM = 0

NOTE

When EELAT is set, the entire EEPROM is unavailable for reads; therefore, no program residing in the EEPROM can be executed while attempting to program unused EEPROM space. Care should be taken that no references to the EEPROM are used while programming. Interrupts should be turned off if the vectors are in the EEPROM. Timing and any serial communications must be done with polling during the programming process.

BYTE, ROW, ERASE, and EELAT bits can be written simultaneously or in any sequence.

EEPGM — Program and Erase Enable Bit

1 = Applies program/erase voltage to EEPROM
0 = Disables program/erase voltage to EEPROM

The EEPGM bit can be set only after EELAT has been set. When EELAT and EEPGM are set simultaneously, EEPGM remains clear but EELAT is set.

The BULKP, BYTE, ROW, ERASE, and EELAT bits cannot be changed when EEPGM is set. To complete a program or erase, two successive writes to clear EEPGM and EELAT bits are required before reading the programmed data. A write to an EEPROM location has no effect when EEPGM is set. Latched address and data cannot be modified during program or erase.

A program or erase operation should follow this sequence:

1. Write BYTE, ROW, and ERASE to the desired value; write EELAT = 1.
2. Write a byte or an aligned word to an EEPROM address.
3. Write EEPGM = 1.
4. Wait for programming (t_{PROG}) or erase (t_{Erase}) delay time.
5. Write EEPGM = 0.
6. Write EELAT = 0.

By jumping from step 5 to step 2, it is possible to program/erase more bytes or words without intermediate EEPROM reads.

Chapter 8
Memory Expansion and Chip-Select

8.1 Introduction

To use memory expansion, the MCU must be operated in one of the expanded modes. Sections of the standard 64-Kbyte address space have memory expansion windows which allow an external address space larger than 64 Kbytes. Memory expansion consists of three memory expansion windows and six address lines which are used in addition to the standard 16 address lines.

The memory expansion function reuses as many as six of the standard 16 address lines. To do this, some of the upper address lines of internal addresses falling in an active window are overridden. Consequently, the address viewed externally may not match the internal address. Usage of chip-selects identify the source of the internal address for debugging and selection of the proper external devices.

All memory expansion windows have a fixed size and two have a fixed address location. The third has two selectable address locations. When an internal address falls into one of these active windows, it is translated as shown in Table 8-1.

Addresses ADDR9–ADDR0 are not affected by memory expansion and are the same externally as they are internally. Addresses ADDR21–ADDR16 are generated only by memory expansion and are individually enabled by software-programmable control bits. If not enabled, they may be used as general-purpose I/O (input/output). Addresses ADDR15–ADDR10 can be the internal addresses or they can be modified by the memory expansion module. These are not available as general-purpose I/O in expanded modes.

Table 8-1. Memory Expansion Values[1]

Internal Address	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10
$0000–$03FF EWDIR[2]= 1, EWEN = 1	1	1	1	1	PEA17	PEA16	PEA15	PEA14	PEA13	PEA12	PEA11	PEA10
$0000–$03FF EWDIR or EWEN = 0	1	1	1	1	1	1	A15	A14	A13	A12	A11	A10
$0400–$07FF EWDIR = 0, EWEN = 1	1	1	1	1	PEA17	PEA16	PEA15	PEA14	PEA13	PEA12	PEA11	PEA10
$0400–$07FF EWDIR = 1, EWEN = x or EWDIR = x, EWEN = 0	1	1	1	1	1	1	A15	A14	A13	A12	A11	A10

Table 8-1. Memory Expansion Values[1] (Continued)

Internal Address	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10
$0800–$6FFF	1	1	1	1	1	1	A15	A14	A13	A12	A11	A10
$7000–$7FFF DWEN = 1	1	1	PDA19	PDA18	PDA17	PDA16	PDA15	PDA14	PDA13	PDA12	A11	A10
$7000–$7FFF DWEN = 0	1	1	1	1	1	1	A15	A14	A13	A12	A11	A10
$8000–$BFFF PWEN = 1	PPA21	PPA20	PPA19	PPA18	PPA17	PPA16	PPA15	PPA14	A13	A12	A11	A10
$8000–$BFFF PWEN = 0	1	1	1	1	1	1	A15	A14	A13	A12	A11	A10
$C000–$FFFF	1	1	1	1	1	1	A15	A14	A13	A12	A11	A10

1. All port G assigned to memory expansion
2. The EWDIR bit in the MISC register selects the E window address (1 = $0000–$03FF including direct space and 0 = $0400–$07FF).

8.2 Generation of Chip-Selects

To use chip-selects the MCU must be in one of the expanded modes. Each of the seven chip-selects has an address space for which it is active — that is, when the current CPU address is in the range of that chip-select, it becomes active. Chip-selects are generally used to reduce or eliminate external address decode logic. These active low signals usually are connected directly to the chip-select pin of an external device.

8.2.1 Chip-Selects Independent of Memory Expansion

Three types of chip-selects are program memory chip-selects, other memory chip-selects and peripheral chip-selects. Memory chip-selects cover a medium-to-large address space. Peripheral chip-selects (CS3–CS0) cover a small address space. The program memory chip-select includes the vector space and is generally used with non-volatile memory. To start the user's program, the program chip-select is designed to be active out of reset. This is the only chip-select which has a functional difference from the others, so a small memory could use a peripheral chip-select and a peripheral could use a memory chip-select.

Figure 8-1 shows peripheral chip-selects in an expanded portion of the memory map. Table 8-2 shows the register settings that correspond to the example. Chip-selects CS2–CS0 always map to the same 2-Kbyte block as the internal register space. The internal registers cover the first 512 bytes and these chip-selects cover all or part of the 512 bytes following the register space blocking out a full 1-Kbyte space. CS3 can map with these other chip-selects or be used in a 1-Kbyte space by itself which starts at either $0000 or $0400. CS3 can be used only for a 1-Kbyte space when it selects the E page of memory expansion and E page is active.

CS3 can be used with a 1-Kbyte space in systems not using memory expansion. However, it must be made to appear as if memory expansion is in use. One of many possible configurations is:

- Select the desired 1-Kbyte space for EPAGE (EWDIR in MISC in the MMI).
- Write the EPAGE register with $0000, if EWDIR is one or $0001 if EWDIR is 0.
- Designate all port G pins as I/O.
- Enable EPAGE and CS3.
- Make CS3 follow EPAGE.

8.2.2 Chip-Selects Used in Conjunction with Memory Expansion

Memory expansion and chip-select functions can work independently, but systems requiring memory expansion perform better when chip-selects are also used. For each memory expansion window there is a chip-select (or two) designed to function with it.

Figure 8-2 shows a memory expansion and chip-select example using three chip-selects. Table 8-3 shows the register settings that correspond to the example. The program space consists of 128 Kbytes of addressable memory in eight 16-Kbyte pages. Page 7 is always accessible in the space from $C000 to $FFFF. The data space consists of 64 Kbytes of addressable memory in 16, 4-Kbyte pages. Unless CSD is used to select the external RAM, pages 0 through 6 appear in the $0000 to $6FFF space wherever there is no higher priority resource. The extra space consists of four, 1-Kbyte pages making 4 Kbytes of addressable memory.

If memory is increased to the maximum in this example, the program space will consist of 4 Mbytes of addressable space with 256 16-Kbyte pages and page $FF always available. The data space will be 1 Mbyte of addressable space with 256 4-Kbyte pages and pages $F0 to $F6 mirrored to the $0000 to $6FFF space. The extra space will be 256 Kbytes of addressable space in 256 1-Kbyte pages.

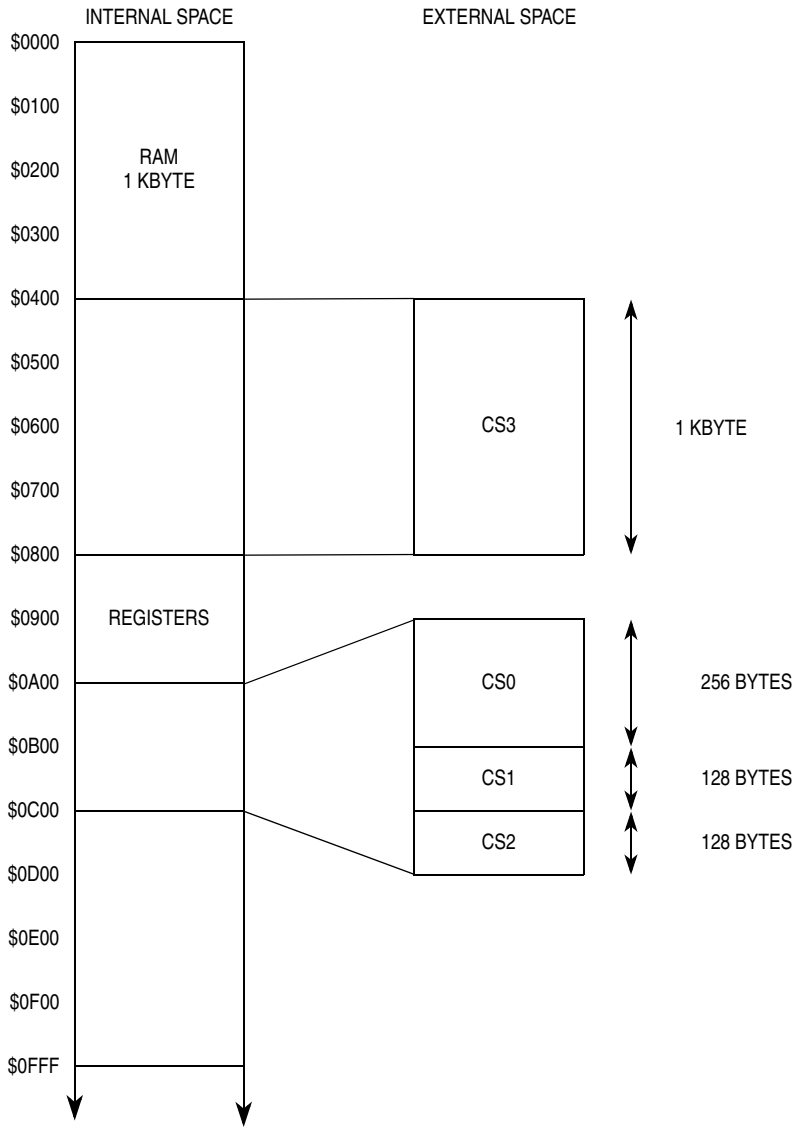


Figure 8-1. Chip-Selects CS3–CS0 Partial Memory Map

Table 8-2. Example Register Settings

Register	Value	Meaning
INITRM	$00	Assigns internal RAM to $0000–$0FFF
INITRG	$08	Assigns register block to $0800–$09FF and register-following chip-selects at $0A00–$0BFF
WINDEF	$20	Enable EPAGE
MXAR	$00	No port G lines assigned as extended address
CSCTL0	$xF	Enables CS3, CS2, CS1, and CS0
CSCTL1	$x8	Makes CS3 follow EPAGE
MISC	%0xxxxxxx	Puts EPAGE at $0400–$07FF
EPAGE	$01	Keeps the translated value of the upper addresses the same as it would have been before translation; not necessary if all external devices use chip-selects

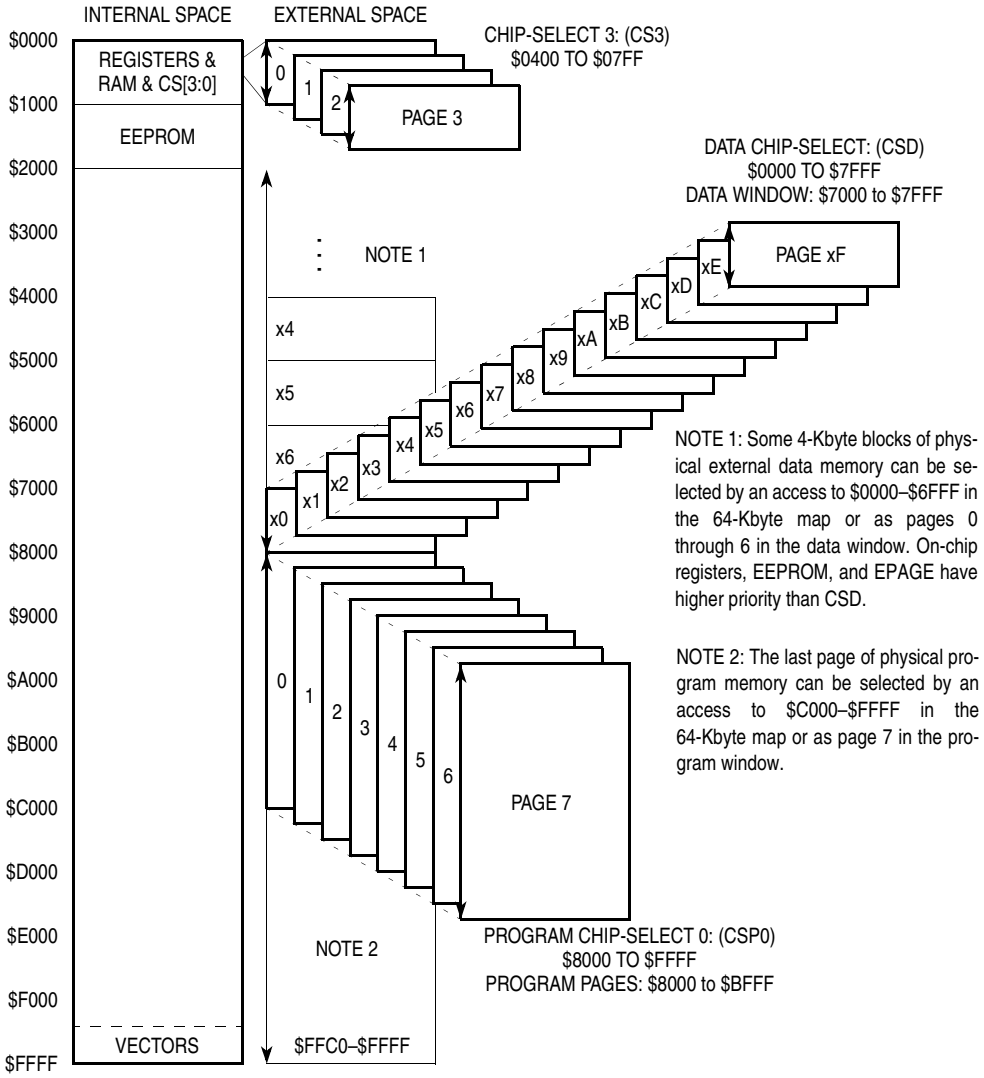


Figure 8-2. Memory Expansion and Chip-Select Example

Table 8-3. Example Register Settings

Register	Value	Meaning
WINDEF	$E0	Enable EPAGE, DPAGE, PPAGE
MXAR	$01	Port G bit 0 assigned as extended address ADDR16
CSCTL0	%00111xxx	Enables CSP0, CSD, and CS3
CSCTL1	$18	Makes CSD follow $0000–$7FFF and CS3 select EPAGE
MISC	%0xxxxxxx	Puts EPAGE at $0400–$09FF

8.3 Chip-Select Stretch

Each chip-select can be chosen to stretch bus cycles associated with it. Stretch can be zero, one, two, or three whole cycles added which allows interfacing to external devices which cannot meet full bus speed timing. Figure 8-3, Figure 8-4, Figure 8-5, and Figure 8-6 show the waveforms for zero to three cycles of stretch.

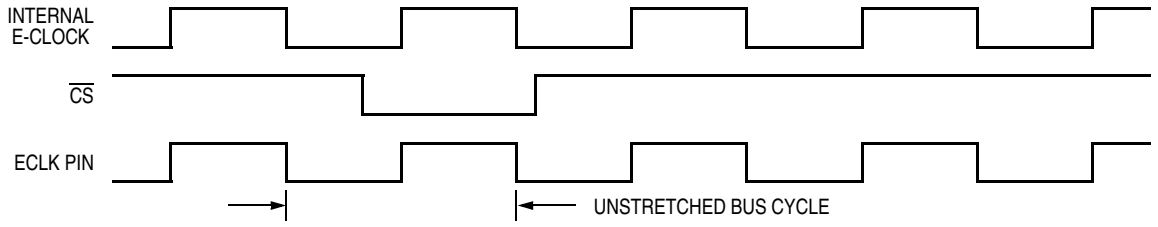


Figure 8-3. Chip-Select with No Stretch

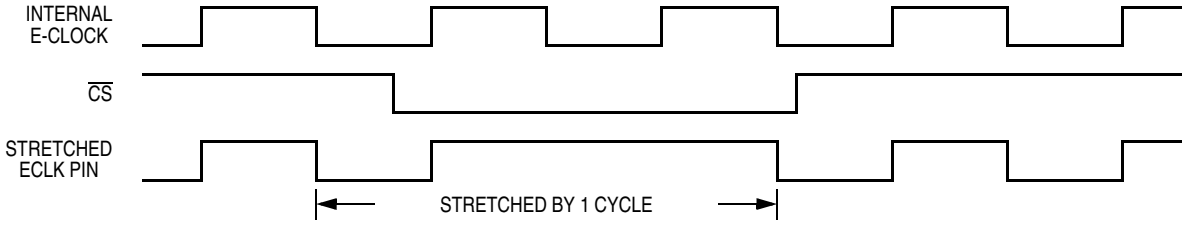


Figure 8-4. Chip-Select with 1-Cycle Stretch

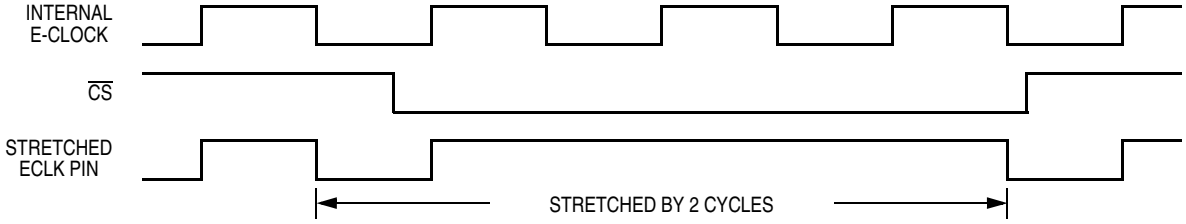


Figure 8-5. Chip-Select with 2-Cycle Stretch

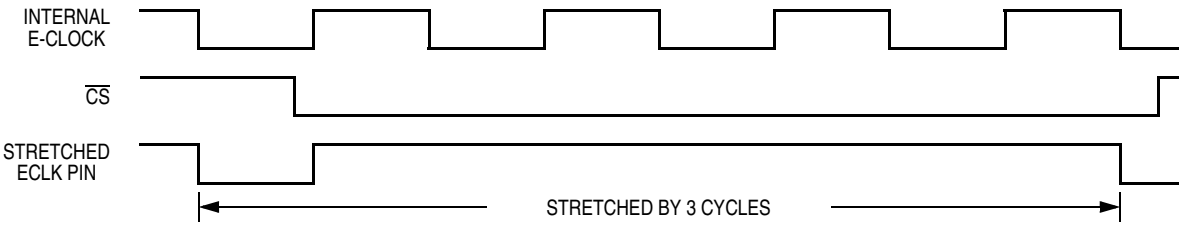


Figure 8-6. Chip-Select with 3-Cycle Stretch

The external E-clock may be the stretched E-clock, the E-clock, or no clock depending on the selection of control bits ESTR and IVIS in the MODE register and NECLK in the PEAR register.

8.4 Memory Expansion Registers

This section describes the memory expansion registers.

8.4.1 Port F Data Register

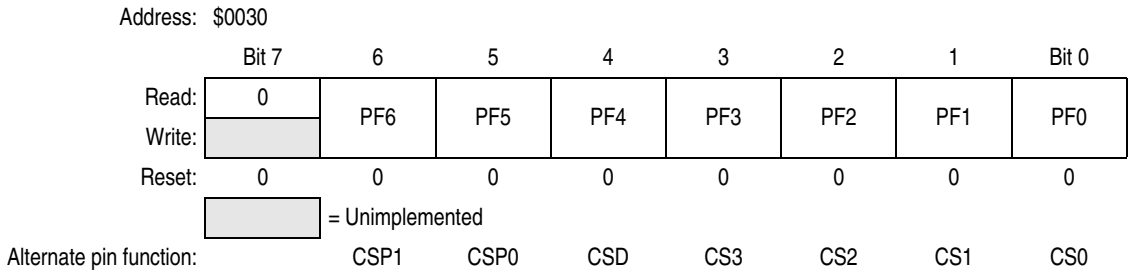


Figure 8-7. Port F Data Register (PORTF)

Read: Anytime
Write: Anytime

Seven port F pins are associated with chip-selects. Any pin not used as a chip-select can be used as general-purpose I/O. All pins are pulled up when inputs (if pullups are enabled). Enabling a chip-select overrides the associated data direction bit and port data bit.

8.4.2 Port G Data Register

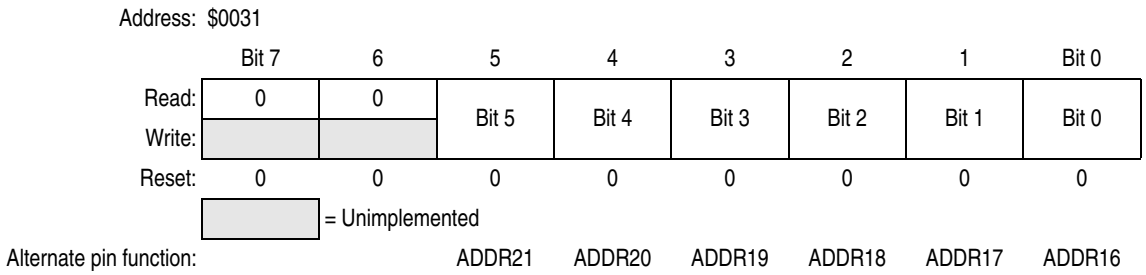


Figure 8-8. Port G Data Register (PORTG)

Read: Anytime
Write: Anytime

Six port G pins are associated with memory expansion. Any pin not used for memory expansion can be used as general-purpose I/O. All pins are pulled up when inputs (if pullups are enabled). Enabling a memory expansion address with the memory expansion assignment register overrides the associated data direction bit and port data bit.

8.4.3 Port F Data Direction Register

Address: $0032

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	DDRF6	DDRF5	DDRF4	DDRF3	DDRF2	DDRF1	DDRF0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 8-9. Port F Data Direction Register (DDRF)

Read: Anytime
Write: Anytime

When port F is active, DDRF determines pin direction.

1 = Associated bit is an output.
0 = Associated bit is an input.

8.4.4 Port G Data Direction Register

Address: $0033

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	DDRG5	DDRG4	DDRG3	DDRG2	DDRG1	DDRG0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 8-10. Port G Data Direction Register (DDRG)

Read: Anytime
Write: Anytime

When port G is active, DDRG determines pin direction.

1 = Associated bit is an output.
0 = Associated bit is an input.

8.4.5 Data Page Register

Address: $0034

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PD19	PD18	PD17	PD16	PD15	PD14	PD13	PD12
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 8-11. Data Page Register (DPAGE)

Read: Anytime
Write: Anytime

When enabled (DWEN = 1), the value in this register determines which of the 256 4-Kbyte pages is active in the data window. An access to the data page memory area ($7000 to $7FFF) forces the contents of DPAGE to address pins ADDR15–ADDR12 and expansion address pins ADDR19–ADDR16. Bits ADDR20 and ADDR21 are forced to 1 if enabled by MXAR. Data chip-select (CSD) must be used in conjunction with this memory expansion window.

8.4.6 Program Page Register

Address: $0035

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PPA21	PPA20	PPA19	PPA18	PPA17	PPA16	PPA15	PPA14
Reset:	0	0	0	0	0	0	0	0

Figure 8-12. Program Page Register (PPAGE)

Read: Anytime
Write: Anytime

When enabled (PWEN = 1), the value in this register determines which of the 256 16-Kbyte pages is active in the program window. An access to the program page memory area ($8000 to $BFFF) forces the contents of PPAGE to address pins ADDR15–ADDR14 and expansion address pins ADDR21–ADDR16. At least one of the program chip-selects (CSP0 or CSP1) must be used in conjunction with this memory expansion window. This register is used by the CALL and RTC instructions to facilitate automatic program flow changing between pages of program memory.

8.4.7 Extra Page Register

Address: $0036

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PEA17	PEA16	PEA15	PEA14	PEA13	PEA12	PEA11	PEA10
Reset:	0	0	0	0	0	0	0	0

Figure 8-13. Extra Page Register (EPAGE)

Read: Anytime
Write: Anytime

When enabled (EWEN = 1), the value in this register determines which of the 256 1-Kbyte pages is active in the extra window. An access to the extra page memory area forces the contents of EPAGE to address pins ADDR15–ADDR10 and expansion address pins ADDR16–ADDR17. Address bits ADDR21–ADDR18 are forced to one (if enabled by MXAR). Chip-select 3 set to follow the extra page window (CS3 with CS3EP = 1) must be used in conjunction with this memory expansion window.

8.4.8 Window Definition Register

Address: $0037

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DWEN	PWEN	EWEN	0	0	0	0	0
Reset:	0	0	0	0	0	0	0	0

Figure 8-14. Window Definition Register (WINDEF)

Read: Anytime
Write: Anytime

DWEN — Data Window Enable Bit

1 = Enables paging of the data space (4 Kbytes: $7000–$7FFF) via the DPAGE register
0 = Disables DPAGE

PWEN — Program Window Enable Bit

1 = Enables paging of the program space
(16 Kbytes: $8000–$BFFF) via the PPAGE register
0 = Disables PPAGE

EWEN — Extra Window Enable Bit

1 = Enables paging of the extra space (1 Kbyte) via the EPAGE register
0 = Disables EPAGE

8.4.9 Memory Expansion Assignment Register

Address: $0038

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	A21E	A20E	A19E	A18E	A17E	A16E
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 8-15. Memory Expansion Assignment Register (MXAR)

Read: Anytime
Write: Anytime

A21E, A20E, A19E, A18E, A17E, and A16E — These bits select the memory expansion pins ADDR21–ADDR16.

1 = Selects memory expansion for the associated bit function, overrides DDRG
0 = Selects general-purpose I/O for the associated bit function

In single-chip modes, these bits have no effect.

8.5 Chip-Selects

The chip-selects are all active low. All pins in the associated port are pulled up when they are inputs and the PUPF bit in PUCR is set.

If memory expansion is used, usually chip-selects should be used as well, since some translated addresses can be confused with untranslated addresses that are not in an expansion window.

In single-chip modes, enabling the chip-select function does not affect the associated pins.

The block of register-following chip-selects CS3–CS0 allows many combinations including:

- 512-byte CS0
- 256-byte CS0 and 256-byte CS1
- 256-byte CS0, 128-byte CS1, and 128-byte CS2
- 128-byte CS0, 128-byte CS1, 128-byte CS2, and 128-byte CS3

These register-following chip-selects are available in the 512-byte space next to and higher in address than the 512-byte space which includes the registers. For example, if the registers are located at $0800 to $09FF, then these register-following chip-selects are available in the space from $0A00 to $0BFF.

8.6 Chip-Select Registers

This section describes the chip-select registers.

8.6.1 Chip-Select Control Register 0

Address: $003C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	CSP1E	CSP0E	CSDE	CS3E	CS2E	CS1E	CS0E
Write:								
Reset:	0	0	1	0	0	0	0	0

= Unimplemented

Figure 8-16. Chip-Select Control Register 0 (CSCTL0)

Read: Anytime
Write: Anytime

Bits have no effect on the associated pin in single-chip modes.

CSP1E — Chip-Select Program 1 Enable Bit

This bit effectively selects the holes in the memory map. It can be used in conjunction with CSP0 to select between two 2-Mbyte devices based on address ADDR21.

1 = Enables this chip-select which covers the space $8000 to $FFFF or full map $0000 to $FFFF
0 = Disables this chip-select

CSP0E — Chip-Select Program 0 Enable Bit

1 = Enables this chip-select which covers the program space $8000 to $FFFF
0 = Disables this chip-select

CSDE — Chip-Select Data Enable Bit

1 = Enables this chip-select which covers either $0000 to $7FFF (CSDHF = 1) or $7000 to $7FFF (CSDHF = 0)
0 = Disables this chip-select

CS3E — Chip-Select 3 Enable Bit

1 = Enables this chip-select which covers a 128-byte space following the register space ($x280–$x2FF or $xA80–$xAFF) Alternately, it can be active for accesses within the extra page window.
0 = Disables this chip-select

CS2E — Chip-Select 2 Enable Bit

1 = Enables this chip-select which covers a 128-byte space following the register space ($x380–$x3FF or $xB80–$xBFF)
0 = Disables this chip-select

CS1E — Chip-Select 1 Enable Bit

CS2 and CS3 have a higher precedence and can override CS1 for a portion of this space.

1 = Enables this chip-select which covers a 256-byte space following the register space ($x300–$x3FF or $xB00–$xBFF)
0 = Disables this chip-select

CS0E — Chip-Select 0 Enable Bit

CS1, CS2, and CS3 have higher precedence and can override CS0 for portions of this space.

1 = Enables this chip-select which covers a 512-byte space following the register space ($x200–$x3FF or $xA00–$xBFF)

0 = Disables this chip-select

8.6.2 Chip-Select Control Register 1

Address: $003D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	CSP1FL	CSPA21	CSDHF	CS3EP	0	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 8-17. Chip-Select Control Register 1 (CSCTL1)

Read: Anytime
Write: Anytime

CSP1FL — Program Chip-Select 1 Covers Full Map

1 = If CSPA21 is cleared, chip-select program 1 covers the entire memory map. If CSPA21 is set, this bit has no meaning or effect.

0 = If CSPA21 is cleared, chip-select program 1 covers half the map, $8000 to $FFFF. If CSPA21 is set, this bit has no meaning or effect.

CSPA21 — Program Chip-Select Split Based on ADDR21

Setting this bit allows two 2-Mbyte memories to make up the 4-Mbyte addressable program space. Since ADDR21 is always one in the unpaged $C000 to $FFFF space, CSP0 is active in this space.

1 = Program chip-selects are both active (if enabled) for space $8000 to $FFFF; CSP0 if ADDR21 is set and CSP1 if ADDR21 is cleared.

0 = CSP0 and CSP1 do not rely on ADDR21.

CSDHF — Data Chip-Select Covers Half the Map

1 = Data chip-select covers half the memory map ($0000 to $7FFF) including the optional data page window ($7000 to $7FFF).

0 = Data chip-select covers only $7000 to $7FFF (the optional data page window).

CS3EP — Chip-Select 3 Follows Extra Page

1 = Chip-select 3 follows accesses to the 1-Kbyte extra page ($0400 to $07FF or $0000 to $03FF). Any accesses to this window cause the chip-select to go active. (EWEN must be set to 1.)

0 = Chip-select 3 includes only accesses to a 128-byte space following the register space.

8.6.3 Chip-Select Stretch Registers

Each of the seven chip-selects has a 2-bit field in this register which determines the amount of clock stretch for accesses in that chip-select space.

Read: Anytime
Write: Anytime

Address: $003E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	SRP1A	SRP1B	SRP0A	SRP0B	STRDA	STRDB
Write:								
Reset:	0	0	1	1	1	1	1	1

= Unimplemented

Figure 8-18. Chip-Select Stretch Register 0 (CSSTR0)

Address: $003F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	STR3A	STR3B	STR2A	STR2B	STR1A	STR1B	STR0A	STR0B
Write:								
Reset:	0	0	1	1	1	1	1	1

Figure 8-19. Chip-Select Stretch Register 1 (CSSTR1)

Table 8-4. Stretch Bit Definition

Stretch Bit SxxxA	Stretch Bit SxxxB	Number of E-Clocks Stretched
0	0	0
0	1	1
1	0	2
1	1	3

8.7 Priority

Only one module or chip-select may be selected at a time. If more than one module shares a space, only the highest priority module is selected.

Table 8-5. Module Priorities

Priority	Module or Space
Highest	On-chip register space — 512 bytes fully blocked for registers although some of this space is unused
	BDM space (internal) — When BDM is active, this 256-byte block of registers and ROM appear at $FFxx; cannot overlap RAM or registers
	On-chip RAM
	On-chip EEPROM (if enabled, EEON = 1)
	E space (external) (1) — 1 Kbyte at either $0000 to $03FF or $0400 to $07FF; may be used with “extra” memory expansion and CS3
	CS space (external) (1) — 512 bytes following the 512-byte register space; may be used with CS3–CS0
	P space (external) (1) — 16 Kbytes fixed at $8000 to $BFFF; may be used with program memory expansion and CSP0 and/or CSP1
	D space (external) (1) — 4 Kbytes fixed at $7000 to $7FFF; may be used with data memory expansion and CSD or CSP1 (if set for full memory space) or the entire half of memory space $0000–$7FFF
Lowest	Remaining external (1)

1. External spaces can be accessed only if the MCU is in expanded mode. Priorities of different external spaces affect chip-selects and memory expansion.

Only one chip-select is active at any address. In the event that two or more chip-selects cover the same address, only the highest priority chip-select is active.

Chip-selects have this order of priority:

Highest … Lowest

CS3	CS2	CS1	CS0	CSP0	CSD	CSP1

Chapter 9
Key Wakeups

9.1 Introduction

The key wakeup feature of the MC68HC812A4 issues an interrupt that wakes up the CPU when it is in stop or wait mode. Three ports are associated with the key wakeup function: port D, port H, and port J. Port D and port H wakeups are triggered with a falling signal edge. Port J key wakeups have a selectable falling or rising signal edge as the active edge. For each pin which has an interrupt enabled, there is a path to the interrupt request signal which has no clocked devices when the part is in stop mode. This allows an active edge to bring the part out of stop.

Default register addresses, as established after reset, are indicated in the following descriptions. For information on remapping the register block, refer to Chapter 5 Operating Modes and Resource Mapping.

9.2 Key Wakeup Registers

This section provides a summary of the key wakeup registers.

9.2.1 Port D Data Register

Address: $0005

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
Reset:	0	0	0	0	0	0	0	0
Alternate pin function:	KWD7	KWD6	KWD5	KWD4	KWD3	KWD2	KWD1	KWD0

Figure 9-1. Port D Data Register (PORTD)

This register is not in the map in wide expanded modes or in special expanded narrow mode with MODE register bit EMD set.

An interrupt is generated when a bit in the KWIFD register and its corresponding KWIED bit are both set. These bits correspond to the pins of port D. All eight bits/pins share the same interrupt vector and can wake the CPU when it is in stop or wait mode. Key wakeups can be used with the pins configured as inputs or outputs.

Key wakeup port D shares a vector and control bit with $\overline{\text{IRQ}}$. IRQEN must be set for key wakeup interrupts to signal the CPU.

9.2.2 Port D Data Direction Register

Address: $0007

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 9-2. Port D Data Direction Register (DDRD)

Read: Anytime
Write: Anytime

This register is not in the map in wide expanded modes or in special expanded narrow mode with MODE register bit EMD set.

Data direction register D is associated with port D and designates each pin as an input or output.

DDRD7–DDRD0 — Data Direction Port D Bits

1 = Associated pin is an output.
0 = Associated pin is an input.

9.2.3 Port D Key Wakeup Interrupt Enable Register

Address: $0020

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 9-3. Port D Key Wakeup Interrupt Enable Register (KWIED)

Read: Anytime
Write: Anytime

This register is not in the map in wide expanded modes and in special expanded narrow mode with MODE register bit EMD set.

KWIED7–KWIED0 — Key Wakeup Port D Interrupt Enable Bits

1 = Interrupt for the associated bit is enabled.
0 = Interrupt for the associated bit is disabled.

9.2.4 Port D Key Wakeup Flag Register

Address: $0021

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 9-4. Port D Key Wakeup Flag Register (KWIFD)

Read: Anytime
Write: Anytime

Each flag is set by a falling edge on its associated input pin. To clear the flag, write 1 to the corresponding bit in KWIFD.

This register is not in the map in wide expanded modes or in special expanded narrow mode with MODE register bit EMD set.

KWIFD7–KWIFD0 — Key Wakeup Port D Flags

1 = Falling edge on the associated bit has occurred. An interrupt occurs if the associated enable bit is set.
0 = Falling edge on the associated bit has not occurred.

9.2.5 Port H Data Register

Address: $0024

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
Reset:	0	0	0	0	0	0	0	0
Alternate pin function:	KWH7	KWH6	KWH5	KWH4	KWH3	KWH2	KWH1	KWH0

Figure 9-5. Port H Data Register (PORTH)

Read: Anytime
Write: Anytime

Port H is associated with key wakeup H. Key wakeups can be used with the pins designated as inputs or outputs. DDRH determines whether each pin is an input or output.

9.2.6 Port H Data Direction Register

Address: $0025

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRH7	DDRH6	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
Reset:	0	0	0	0	0	0	0	0

Figure 9-6. Port H Data Direction Register (DDRH)

Read: Anytime
Write: Anytime

Data direction register H is associated with port H and designates each pin as an input or output.

DDRH7–DDRH0 — Data Direction Port H Bits

1 = Associated pin is an output.
0 = Associated pin is an input.

9.2.7 Port H Key Wakeup Interrupt Enable Register

Address: $0026

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	KWIEH7	KWIEH6	KWIEH5	KWIEH4	KWIEH3	KWIEH2	KWIEH1	KWIEH0
Reset:	0	0	0	0	0	0	0	0

Figure 9-7. Port H Key Wakeup Interrupt Enable Register (KWIEH)

An interrupt is generated when a bit in the KWIFH register and its corresponding KWIEH bit are both set. These bits correspond to the pins of port H.

KWIEH7–KWIEH0 — Key Wakeup Port H Interrupt Enable Bits

1 = Interrupt for the associated bit is enabled.
0 = Interrupt for the associated bit is disabled.

9.2.8 Port H Key Wakeup Flag Register

Address: $0027

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	KWIFH7	KWIFH6	KWIFH5	KWIFH4	KWIFH3	KWIFH2	KWIFH1	KWIFH0
Reset:	0	0	0	0	0	0	0	0

Figure 9-8. Port H Key Wakeup Flag Register (KWIFH)

Read: Anytime
Write: Anytime

Each flag is set by a falling edge on its associated input pin. To clear the flag, write one to the corresponding bit in KWIFH.

KWIFH7–KWIFH0 — Key Wakeup Port H Flags

1 = Falling edge on the associated bit has occurred (an interrupt occurs if the associated enable bit is set)
0 = Falling edge on the associated bit has not occurred

9.2.9 Port J Data Register

Address: $0028

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
Reset:	0	0	0	0	0	0	0	0
Alternate pin function:	KWJ7	KWJ6	KWJ5	KWJ2	KWJ4	KWJ2	KWJ1	KWJ0

Figure 9-9. Port J Data Register (PORTJ)

Read: Anytime
Write: Anytime

Port J is associated with key wakeup J. Key wakeups can be used with the pins designated as inputs or outputs. DDRJ determines whether each pin is an input or output.

9.2.10 Port J Data Direction Register

Address: $0029

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRJ7	DDRJ6	DDRJ5	DDRJ4	DDRJ3	DDRJ2	DDRJ1	DDRJ0
Reset:	0	0	0	0	0	0	0	0

Figure 9-10. Port J Data Direction Register (DDRJ)

Determines direction of each port J pin.

DDRJ7–DDRJ0 — Data Direction Port J Bits

1 = Associated pin is an output.
0 = Associated pin is an input.

9.2.11 Port J Key Wakeup Interrupt Enable Register

Address: $002A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	KWIEJ7	KWIEJ6	KWIEJ5	KWIEJ4	KWIEJ3	KWIEJ2	KWIEJ1	KWIEJ0
Reset:	0	0	0	0	0	0	0	0

Figure 9-11. Port J Key Wakeup Interrupt Enable Register (KWIEJ)

Read: Anytime
Write: Anytime

An interrupt is generated when a bit in the KWIFJ register and its corresponding KWIEJ bit are both set. These bits correspond to the pins of port J. All eight bits/pins share the same interrupt vector.

KWIEJ7–KWIEF0 — Key Wakeup Port J Interrupt Enable Bits

1 = Interrupt for the associated bit is enabled.
0 = Interrupt for the associated bit is disabled.

9.2.12 Port J Key Wakeup Flag Register

Address: $002B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	KWIFJ7	KWIFJ6	KWIFJ5	KWIFJ4	KWIFJ3	KWIFJ2	KWIFJ1	KWIFJ0
Reset:	0	0	0	0	0	0	0	0

Figure 9-12. Port J Key Wakeup Flag Register (KWIFJ)

Read: Anytime
Write: Anytime

Each flag gets set by an active edge on the associated input pin. This could be a rising or falling edge based on the state of the KPOLJ register. To clear the flag, write 1 to the corresponding bit in KWIFJ.

Initialize this register after initializing KPOLJ so that illegal flags can be cleared.

KWIFJ7–KWIFJ0 — Key Wakeup Port J Flags

1 = An active edge on the associated bit has occurred. An interrupt occurs if the associated enable bit is set.
0 = An active edge on the associated bit has not occurred.

9.2.13 Port J Key Wakeup Polarity Register

Address: $002C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	KPOLJ7	KPOLJ6	KPOLJ5	KPOLJ4	KPOLJ3	KPOLJ2	KPOLJ1	KPOLJ0
Reset:	0	0	0	0	0	0	0	0

Figure 9-13. Port J Key Wakeup Polarity Register (KPOLJ)

Read: Anytime
Write: Anytime

It is best to clear the flags after initializing this register because changing the polarity of a bit can cause the associated flag to set.

KPOLJ7–KPOLJ0 — Key Wakeup Port J Polarity Select Bits

1 = Rising edge on the associated port J pin sets the associated flag bit in the KWIFJ register.
0 = Falling edge on the associated port J pin sets the associated flag bit in the KWIFJ register.

9.2.14 Port J Pullup/Pulldown Select Register

Address: $002D

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PUPSJ7	PUPSJ6	PUPSJ5	PUPSJ4	PUPSJ3	PUPSJ2	PUPSJ1	PUPSJ0
Reset:	0	0	0	0	0	0	0	0

Figure 9-14. Port J Pullup/Pulldown Select Register (PUPSJ)

Read: Anytime
Write: Anytime

Each bit in the register corresponds to a port J pin. Each bit selects a pullup or pulldown device for the associated port J pin. The pullup or pulldown is active only if enabled by the PULEJ register.

PUPSJ should be initialized before enabling the pullups/pulldowns (PUPEJ).

PUPSJ7–PUPSJ0 — Key Wakeup Port J Pullup/Pulldown Select Bits

1 = Pullup is selected for the associated port J pin.
0 = Pulldown is selected for the associated port J pin.

9.2.15 Port J Pullup/Pulldown Enable Register

Address: $002E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PULEJ7	PULEJ6	PULEJ5	PULEJ4	PULEJ3	PULEJ2	PULEJ1	PULEJ0
Reset:	0	0	0	0	0	0	0	0

Figure 9-15. Port J Pullup/Pulldown Enable Register (PULEJ)

Read: Anytime
Write: Anytime

Each bit in the register corresponds to a port J pin. If a pin is configured as an input, each bit enables an active pullup or pulldown device. PUPSJ selects whether a pullup or a pulldown is the active device.

PULEJ7–PULEJ0 — Key Wakeup Port J Pullup/Pulldown Enable Bits

1 = Selected pullup/pulldown device for the associated port J pin is enabled if it is an input.
0 = Associated port J pin has no pullup/pulldown device.

Chapter 10
Clock Module

10.1 Introduction

Clock generation circuitry generates the internal and external E-clock signals as well as internal clock signals used by the CPU and on-chip peripherals. A clock monitor circuit, a computer operating properly (COP) watchdog circuit, and a periodic interrupt circuit are also incorporated into the MCU.

10.2 Block Diagram

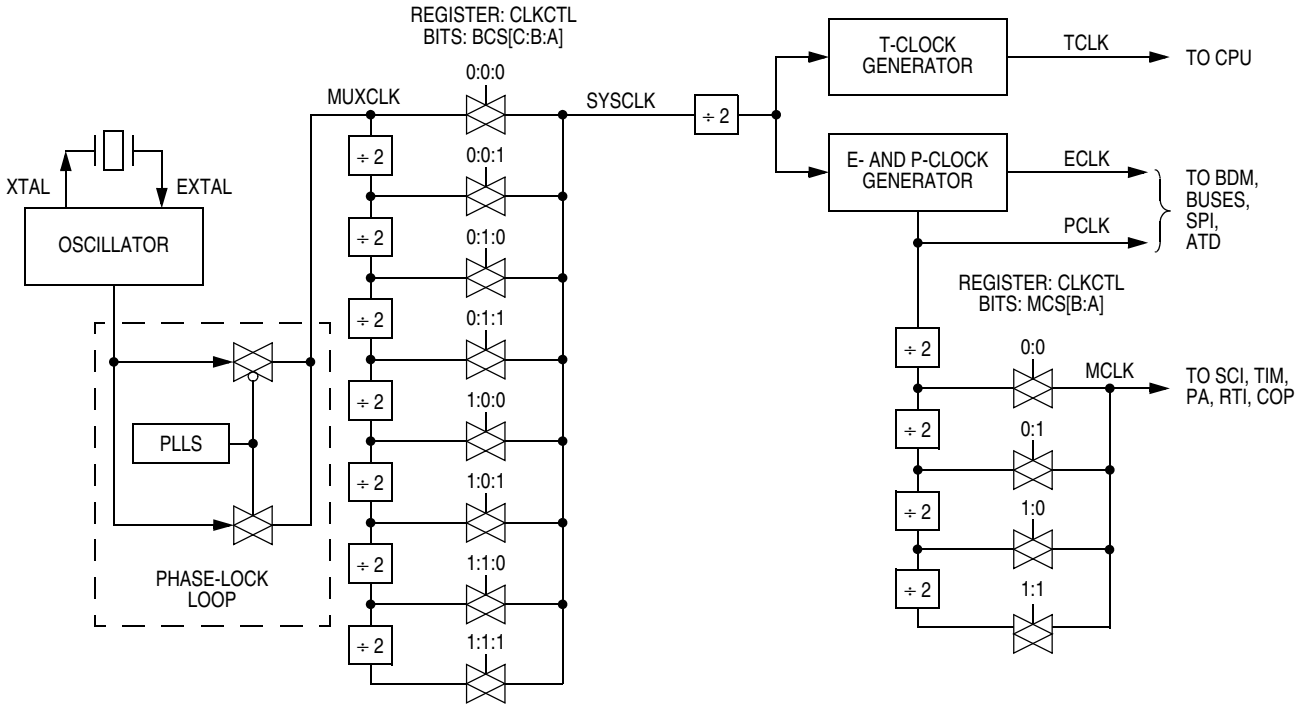


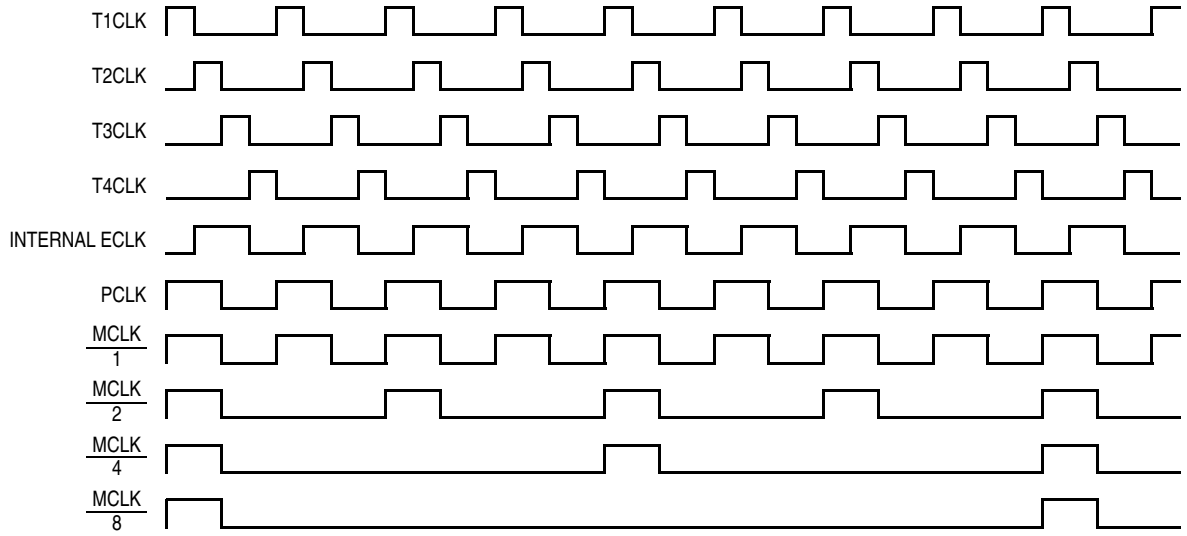
Figure 10-1. Clock Module Block Diagram

10.2.1 Clock Generators

The clock module generates four types of internal clock signals derived from the oscillator:

1. T-clocks — Drives the CPU
2. E-clock — Drives the bus interfaces, BDM, SPI, and ATD
3. P-clock — Drives the bus interfaces, BDM, SPI, and ATD
4. M-clock — Drives on-chip modules such as the timer, SCI, RTI, COP, and restart-from-stop delay time

Figure 10-2 shows clock timing relationships. Four bits in the CLKCTL register control the base clock and M-clock divide selection (÷1, ÷2, ÷4, and ÷8 are selectable).



Note: The MCLK depends on the chosen divider settings in the CLKCTL register.

Figure 10-2. Internal Clock Relationships

10.3 Register Map

NOTE

The register block can be mapped to any 2-Kbyte boundary within the standard 64-Kbyte address space. The register block occupies the first 512 bytes of the 2-Kbyte block. This register map shows default addressing after reset.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0014	Real-Time Interrupt Control Reg. (RTICTL) See page 105.	Read:	RTIE	RSWAI	RSBCK	0	RTBYP	RTR2	RTR1	RTR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	Real-Time Interrupt Flag Register (RTIFLG) See page 107.	Read:	RTIF	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0016	COP Control Register (COPCTL) See page 107.	Read:	CME	FCME	FCM	FCOP	DISR	CR2	CR1	CR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0017	Arm/Reset COP Register (COPRST) See page 109.	Read:	0	0	0	0	0	0	0	0
		Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-3. Clock Function Register Map

10.4 Functional Description

This section provides a functional description of the MC68HC812A4.

10.4.1 Computer Operating Properly (COP)

The COP or watchdog timer is an added check that a program is running and sequencing properly. When the COP is being used, software is responsible for keeping a free-running watchdog timer from timing out. If the watchdog timer times out, it is an indication that the software is no longer being executed in the intended sequence; thus, a system reset is initiated. Three control bits allow selection of seven COP timeout periods. When COP is enabled, sometime during the selected period the program must write $55 and $AA (in this order) to the COPRST register. If the program fails to do this, the part resets. If any value other than $55 or $AA is written, the part resets.

10.4.2 Real-Time Interrupt

There is a real-time (periodic) interrupt (RTI) available to the user. This interrupt occurs at one of seven selected rates. An interrupt flag and an interrupt enable bit are associated with this function. The rate select has three bits.

10.4.3 Clock Monitor

The clock monitor circuit is based on an internal resistor-capacitor (RC) time delay. If no MCU clock edges are detected within this RC time delay, the clock monitor can generate a system reset. The clock monitor function is enabled/disabled by the CME control bit in the COPCTL register. This timeout is based on an RC delay so that the clock monitor can operate without any MCU clocks.

CME enables clock monitor.

1 = Slow or stopped clocks (including the STOP instruction) cause a clock reset sequence.
0 = Clock monitor is disabled. Slow clocks and STOP instruction may be used.

Clock monitor timeouts are shown in Table 10-1.

Table 10-1. Clock Monitor Timeouts

Supply	Range
5 V ± 10%	2–20 µs
3 V ± 10%	5–100 µs

10.4.4 Peripheral Clock Divider Chains

Figure 10-4, Figure 10-5, and Figure 10-6 summarize the peripheral clock divider chains.

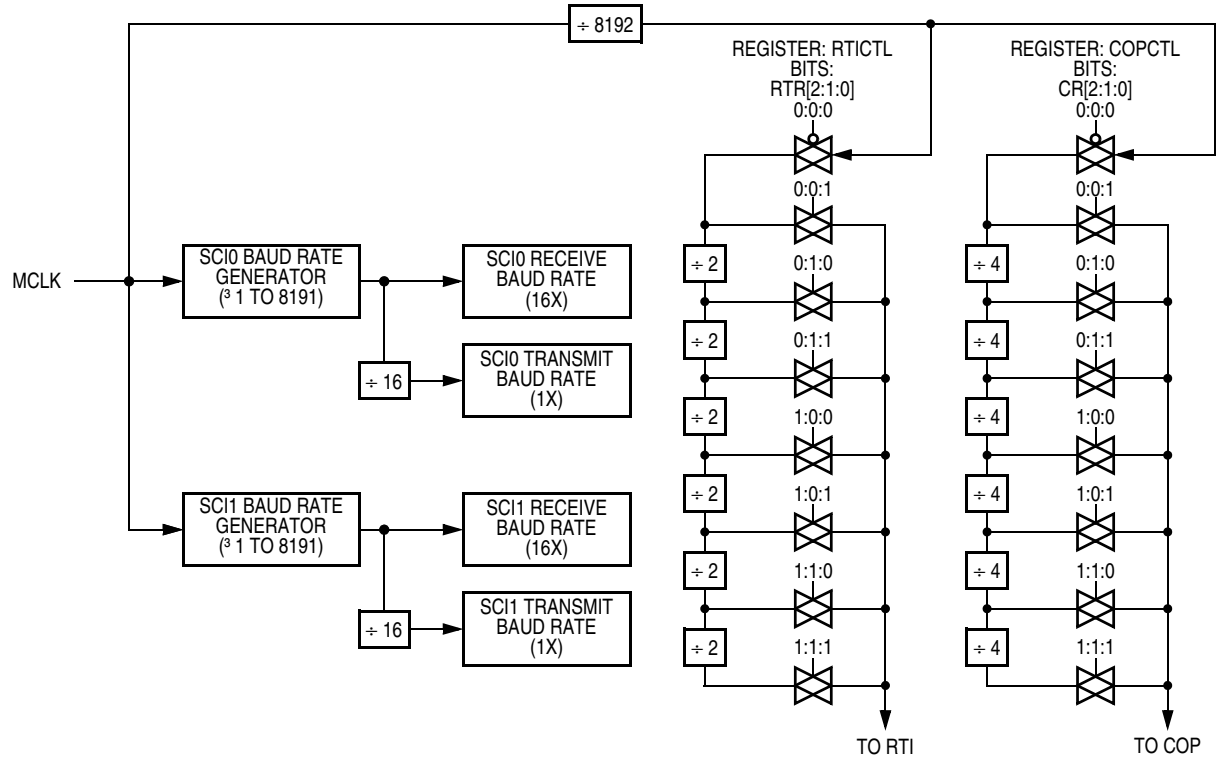


Figure 10-4. Clock Chain for SCI0, SCI1, RTI, and COP

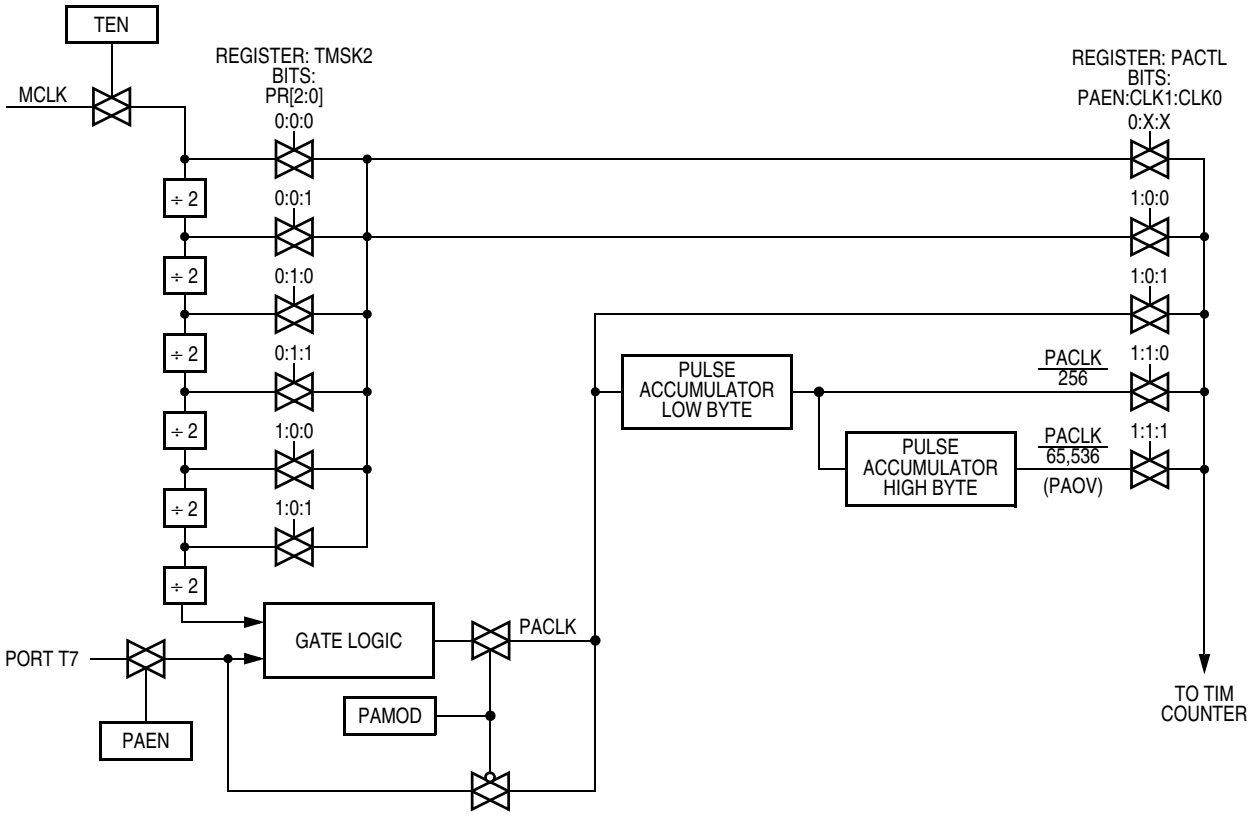


Figure 10-5. Clock Chain for TIM

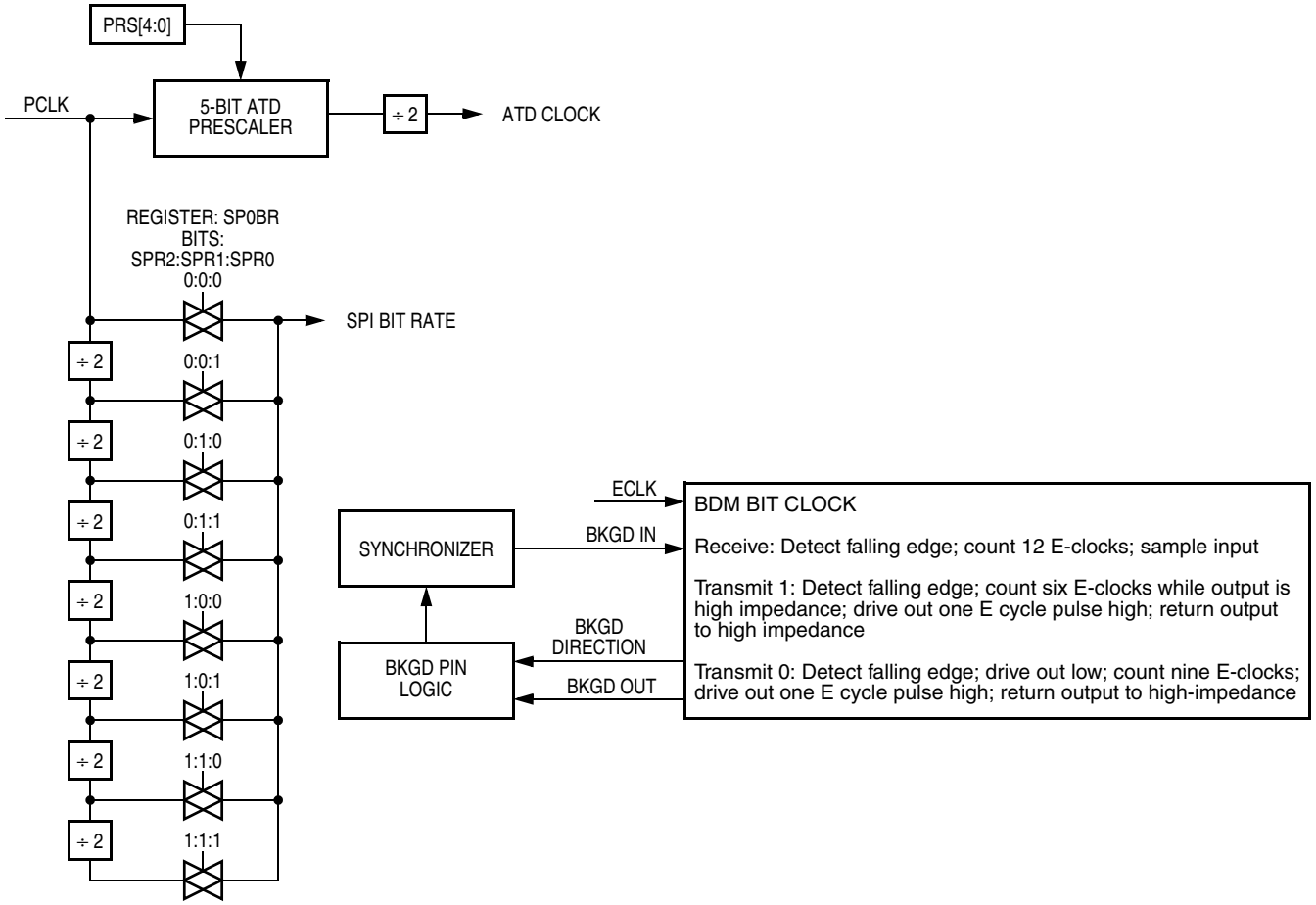


Figure 10-6. Clock Chain for SPI, ATD, and BDM

10.5 Registers and Reset Initialization

This section describes the registers and reset initialization.

10.5.1 Real-Time Interrupt Control Register

Address: $0014

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RTIE	RSWAI	RSBCK	0	RTBYP	RTR2	RTR1	RTR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-7. Real-Time Interrupt Control Register (RTICTL)

Read: Anytime
Write: Varies from bit to bit

RTIE — Real-Time Interrupt Enable Bit

Write: Anytime

RTIE enables interrupt requests generated by the RTIF flag.

1 = RTIF interrupt requests enabled
0 = RTIF interrupt requests disabled

RSWAI — RTI Stop in Wait Bit

Write: Once in normal modes, anytime in special modes

RSWAI disables the RTI and the COP during wait mode.

1 = RTI and COP disabled in wait mode
0 = RTI and COP enabled in wait mode

RSBCK — RTI Stop in Background Mode Bit

Write: Once in normal modes, anytime in special modes

RSBCK disables the RTI and the COP during background debug mode.

1 = RTI and COP disabled during background mode
0 = RTI and COP enabled during background mode

RTBYP — RTI Bypass Bit

Write: Never in normal modes, anytime in special modes

RTBYP allows faster testing by causing the divider chain to be bypassed. The divider chain normally divides M by 2^{13}. When RTBYP is set, the divider chain divides M by 4.

1 = Divider chain bypass
0 = No divider chain bypass

RTR2, RTR1, RTR0 — Real-Time Interrupt Rate Select Bits

Write: Anytime

Rate select for real-time interrupt. The clock used for this module is the module (M) clock.

Table 10-2. Real-Time Interrupt Rates

RTR[2:1:0]	M-Clock Divisor	Real-Time Timeout Period	
		M = 4.0 MHz	M = 8.0 MHz
000	Off	Off	Off
001	2^{13}	2.048 ms	1.024 ms
010	2^{14}	4.096 ms	2.048 ms
011	2^{15}	8.196 ms	4.096 ms
100	2^{16}	16.384 ms	8.196 ms
101	2^{17}	32.768 ms	16.384 ms
110	2^{18}	65.536 ms	32.768 ms
111	2^{19}	131.72 ms	65.536 ms

10.5.2 Real-Time Interrupt Flag Register

Address: $0015

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RTIF	0	0	0	0	0	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-8. Real-Time Interrupt Flag Register (RTIFLG)

RTIF — Real-Time Interrupt Flag

RTIF is set when the timeout period elapses. RTIF generates an interrupt request if the RTIE bit is set in the RTI control register. Clear RTIF by writing to the real-time interrupt flag register with RTIF set.

1 = Timeout period elapsed
0 = Timeout period not elapsed

10.5.3 COP Control Register

Address: $0016

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CME	FCME	FCM	FCOP	DISR	CR2	CR1	CR0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 10-9. COP Control Register (COPCTL)

Read: Anytime
Write: Varies from bit to bit

CME — Clock Monitor Enable Bit

Write: Anytime

CME enables the clock monitor. If the force clock monitor enable bit, FCME, is set, CME has no meaning or effect.

1 = Clock monitor enabled
0 = Clock monitor disabled

NOTE

Clear the CME bit before executing a STOP instruction and set the CME bit after exiting stop mode.

FCME — Force Clock Monitor Enable Bit

Write: Once in normal modes, anytime in special modes

FCME forces the clock monitor to be enabled until a reset occurs. When FCME is set, the CME bit has no effect.

1 = Clock monitor enabled
0 = CME bit enables or disables clock monitor

NOTE

Clear the FCME bit in applications that use the STOP instruction and the clock monitor.

FCM — Force Clock Monitor Reset Bit

Write: Never in normal modes, anytime in special modes

FCM forces a reset when the clock monitor is enabled and detects a slow or stopped clock.

1 = Clock monitor reset enabled
0 = Normal operation

NOTE

When the disable reset bit, DISR, is set, FCM has no effect.

FCOP — Force COP Reset Bit

Write: Never in normal modes; anytime in special modes

FCOP forces a reset when the COP is enabled and times out.

1 = COP reset enabled
0 = Normal operation

NOTE

When the disable reset bit, DISR, is set, FCOP has no effect.

DISR — Disable Reset Bit

Write: Never in normal modes; anytime in special modes

DISR disables clock monitor resets and COP resets.

1 = Clock monitor and COP resets disabled
0 = Normal operation

CR2, CR1, and CR0 — COP Watchdog Timer Rate Select Bits

Write: Once in normal modes, anytime in special modes

The COP system is driven by a constant frequency of $M/2^{13}$. These bits specify an additional division factor to arrive at the COP timeout rate. (The clock used for this module is the M-clock.)

Table 10-3. COP Watchdog Rates

CR[2:1:0]	M-Clock Divisor	COP Timeout Period	
		0/+2.048 ms	0/+1.024 ms
		M = 4.0 MHz	M = 8.0 MHz
000	Off	Off	Off
001	2^{13}	2.048 ms	1.024 ms
010	2^{15}	8.1920 ms	4.096 ms
011	2^{17}	32.768 ms	16.384 ms
100	2^{19}	131.072 ms	65.536 ms
101	2^{21}	524.288 ms	262.144 ms
110	2^{22}	1.048 s	524.288 ms
111	2^{23}	2.097 s	1.048576 s

10.5.4 Arm/Reset COP Timer Register

Address: $0017

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	0
Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 10-10. Arm/Reset COP Timer Register (COPRST)

To restart the COP timeout period and avoid a COP reset, write $55 and then $AA to this address before the end of the COP timeout period. Other instructions can be executed between these writes. Writing anything other than $55 or $AA causes a COP reset to occur.

Chapter 11
Phase-Lock Loop (PLL)

11.1 Introduction

The phase-lock loop (PLL) allows slight adjustments in the frequency of the MCU. The smallest increment of adjustment is ± 9.6 kHz to the output frequency (F_{Out}) rate assuming an input clock of 16.8 MHz (OSCXTAL) and a reference divider set to 1750. Figure 11-1 shows the PLL dividers and a portion of the clock module and Figure 11-2 provides a register map.

11.2 Block Diagram

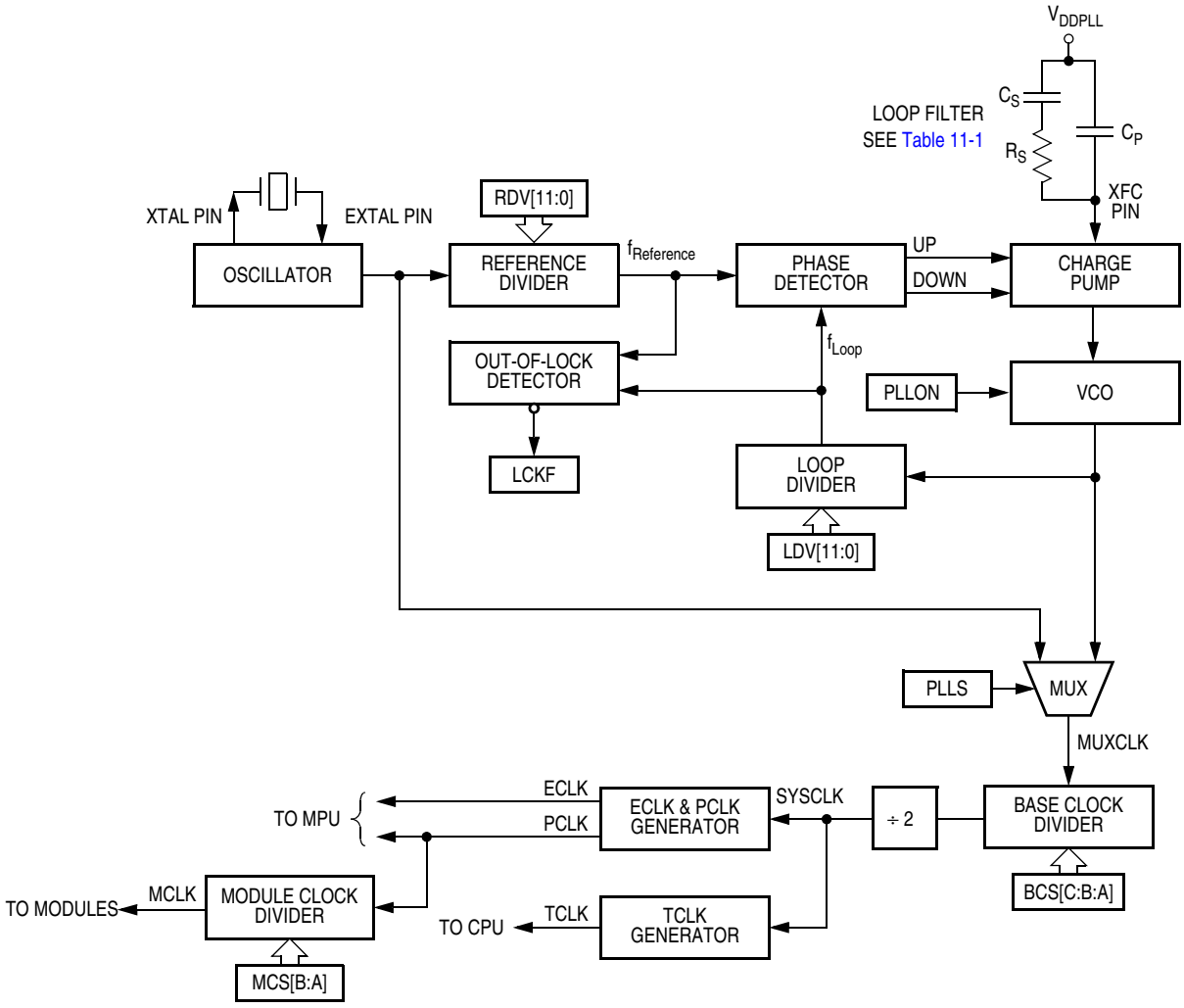


Figure 11-1. PLL Block Diagram

Table 11-1. PLL Filter Values

R_S	E Clock	C_S	Crystal
16,778.40524	8,000,000	0.000000033	32,000
11,864.12412	8,000,000	0.000000033	64,000
3,001.412373	8,000,000	0.000000033	1,000,000
2,450.642941	8,000,000	0.000000033	1,500,000
2,237.120698	8,000,000	0.000000033	1,800,000
2,122.319042	8,000,000	0.000000033	2,000,000
1,898.259859	8,000,000	0.000000033	2,500,000
1,732.866242	8,000,000	0.000000033	3,000,000
1,604.322397	8,000,000	0.000000033	3,500,000
1,500.706187	8,000,000	0.000000033	4,000,000
1,355.8999	8,000,000	0.000000033	4,900,000
1,342.272419	8,000,000	0.000000033	5,000,000

C_P = .0033 µF

11.3 Register Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0040	Loop Divider Register High (LDVH) See page 113.	Read:	0	0	0	0	LDV11	LDV10	LDV9	LDV8
		Write:								
		Reset:	0	0	0	0	1	1	1	1
$0041	Loop Divider Register Low (LDVL) See page 113.	Read:	LDV7	LDV6	LDV5	LDV4	LDV3	LDV2	LDV1	LDV0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0042	Reference Divider Register High (RDVH) See page 114.	Read:	0	0	0	0	RDV11	RDV10	RDV9	RDV8
		Write:								
		Reset:	0	0	0	0	1	1	1	1
$0043	Reference Divider Register Low (RDVL) See page 114.	Read:	RDV7	RDV6	RDV5	RDV4	RDV3	RDV2	RDV1	RDV0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0047	Clock Control Register (CLKCTL) See page 114.	Read:	LCKF	PLLON	PLLS	BCSC	BCSB	BCSA	MCSB	MCSA
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 11-2. PLL Register Map

11.4 Functional Description

The PLL may be used to run the MCU from a different timebase than the incoming crystal value. If the PLL is selected, it continues to run when it's in wait or stop mode which results in more power consumption than normal. To take full advantage of the reduced power consumption of stop mode, turn off the PLL before going into stop.

Although it is possible to set the divider to command a very high clock frequency, do not exceed the 16.8 MHz frequency limit for the MCU.

A passive external loop filter must be placed on the control line (XFC pad). The filter is a second-order, low-pass filter to eliminate the VCO input ripple.

11.5 Registers and Reset Initialization

This section describes the registers and reset initialization.

11.5.1 Loop Divider Registers

Address: $0040

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	LDV11	LDV10	LDV9	LDV8
Write:								
Reset:	0	0	0	0	1	1	1	1

= Unimplemented

Figure 11-3. Loop Divider Register High (LDVH)

Address: $0041

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LDV7	LDV6	LDV5	LDV4	LDV3	LDV2	LDV1	LDV0
Write:								
Reset:	1	1	1	1	1	1	1	1

Figure 11-4. Loop Divider Register Low (LDVL)

Read: Anytime
Write: Anytime

If the PLL is on, the count in the loop divider (LDV) 12-bit register effectively multiplies up from the PLL base frequency.

CAUTION

Do not exceed the maximum rated operating frequency for the CPU.

11.5.2 Reference Divider Registers

Address: $0042

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	RDV11	RDV10	RDV9	RDV8
Write:								
Reset:	0	0	0	0	1	1	1	1

= Unimplemented

Figure 11-5. Reference Divider Register High (RDVH)

Address: $0043

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RDV7	RDV6	RDV5	RDV4	RDV3	RDV2	RDV1	RDV0
Write:								
Reset:	1	1	1	1	1	1	1	1

Figure 11-6. Reference Divider Register Low (RDVL)

Read: Anytime
Write: Anytime

The count in the reference divider (RDV) 12-bit register divides the crystal oscillator clock input.

In the reset condition, both LDV and RDV are set to the maximum count which produces an internal frequency at the phase detector of 8.2 kHz and a final output frequency of 16.8 MHz with a 16.8 MHz input clock.

11.5.3 Clock Control Register

Address: $0047

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LCKF	PLLON	PLLS	BCSC	BCSB	BCSA	MCSB	MCSA
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 11-7. Clock Control Register (CLKCTL)

Read: Anytime
Write: Anytime

LCKF — Lock Flag

This read-only flag is set when the PLL frequency is at least half the target frequency and no more than twice the target frequency.

1 = PLL locked
0 = PLL not locked

PLLON — PLL On Bit

Setting PLLON turns on the PLL.

1 = PLL on
0 = PLL off

PLLS — PLL Select Bit (PLL output or crystal input frequency)

PLLS selects the PLL after the LCKF flag is set.

1 = PLL selected

0 = Crystal input selected

BCS[C:B:A] — Base Clock Select Bits

These bits determine the frequency of SYSCLK. SYSCLK is the source clock for the MCU, including the CPU and buses. See Table 11-2. SYSCLK and is twice the bus rate. MUXCLK is either the PLL output or the crystal input frequency as selected by the PLLS bit.

Table 11-2. Base Clock Selection

BCSC:BCSB:BCSA	SYSCLK
000	MUXCLK
001	$\frac{MUXCLK}{2}$
010	$\frac{MUXCLK}{4}$
011	$\frac{MUXCLK}{8}$
100	$\frac{MUXCLK}{16}$
101	$\frac{MUXCLK}{32}$
110	$\frac{MUXCLK}{64}$
111	$\frac{MUXCLK}{128}$

MCSA and MCSB — Module Clock Select Bits

These bits determine the clock used by some sections of some of the modules such as the baud rate generators of the SCIs, the timer counter, the RTI, and COP. See Table 11-3. MCLK is the module clock and PCLK is an internal bus rate clock.

Table 11-3. Module Clock Selection

MCS[B:A]	MCLK
00	PCLK
01	$\frac{PCLK}{2}$
10	$\frac{PCLK}{4}$
11	$\frac{PCLK}{8}$

The BCSx and MCSx bits can be changed with a single-write access. In combination, these bits can be used to “throttle” the CPU clock rate without affecting the MCLK rate; timing and baud rates can remain constant as the processor speed is changed to match system requirements. This can save overall system power.

Chapter 12
Standard Timer Module

12.1 Introduction

The standard timer module is a 16-bit, 8-channel timer with:

- Input capture
- Output compare
- Pulse accumulator functions

A block diagram is given in Figure 12-1.

12.2 Register Map

A summary of the input/oputput (I/O) registers is shown in Figure 12-2.

NOTE

The register block can be mapped to any 2-Kbyte boundary within the standard 64-Kbyte address space. The register block occupies the first 512 bytes of the 2-Kbyte block. This register map shows default addressing after reset.

In normal modes, writing to a reserved bit has no effect and reading returns logic 0.

In any mode, writing to an unimplemented bit has no effect and reading returns a logic 0.

12.3 Block Diagram

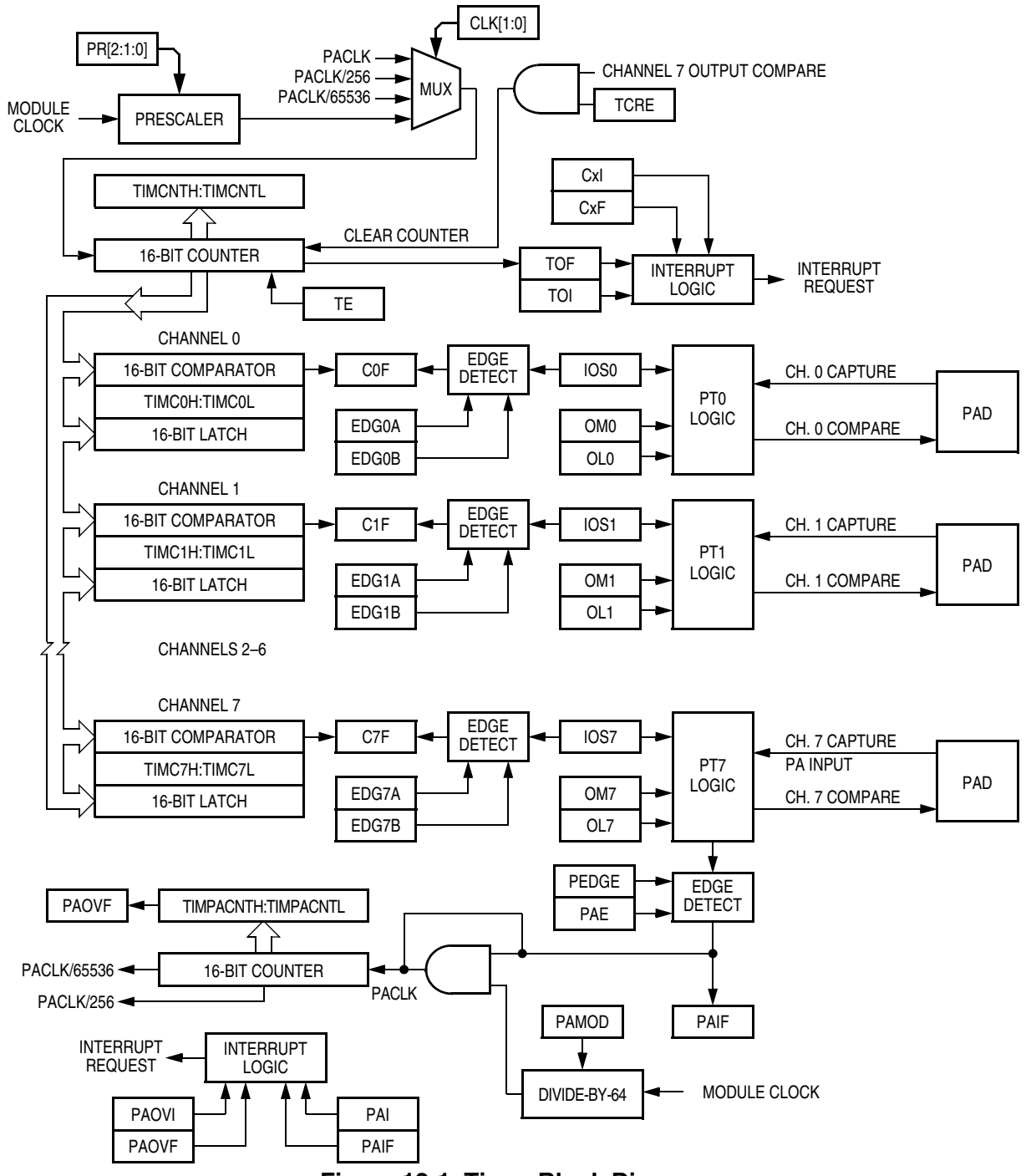


Figure 12-1. Timer Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0080	Timer IC/OC Select Register (TIOS) See page 125.	Read: Write:	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
		Reset:	0	0	0	0	0	0	0	0
$0081	Timer Compare Force Register (CFORC) See page 125.	Read: Write:	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
		Reset:	0	0	0	0	0	0	0	0
$0082	Timer Output Compare 7 Mask Register (OC7M) See page 126.	Read: Write:	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
		Reset:	0	0	0	0	0	0	0	0
$0083	Timer Output Compare 7 Data Register (OC7D) See page 126.	Read: Write:	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
		Reset:	0	0	0	0	0	0	0	0
$0084	Timer Counter Register High (TCNTH) See page 127.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0085	Timer Counter Register Low (TCNTL) See page 127.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0086	Timer System Control Register (TSCR) See page 127.	Read:	TEN	TSWAI	TSBCK	TFFCA	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0087	Reserved		R	R	R	R	R	R	R	R
$0088	Timer Control Register 1 (TCTL1) See page 129.	Read: Write:	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
		Reset:	0	0	0	0	0	0	0	0
$0089	Timer Control Register 2 (TCTL2) See page 129.	Read: Write:	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
		Reset:	0	0	0	0	0	0	0	0
$008A	Timer Control Register 3 (TCTL3) See page 130.	Read: Write:	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
		Reset:	0	0	0	0	0	0	0	0
$008B	Timer Control Register 4 (TCTL4) See page 130.	Read: Write:	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
		Reset:	0	0	0	0	0	0	0	0
$008C	Timer Mask Register 1 (TMSK1) See page 130.	Read: Write:	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-2. I/O Register Summary (Sheet 1 of 4)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$008D	Timer Mask Register 2 (TMSK2) See page 131.	Read:	TOI	0	PUPT	RDPT	TCRE	PR2	PR1	PR0
		Write:								
		Reset:	0	0	1	1	0	0	0	0
$008E	Timer Flag Register 1 (TFLG1) See page 132.	Read:	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$008F	Timer Flag Register 2 (TFLG2) See page 132.	Read:	TOF	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0090	Timer Channel 0 Register High (TC0H) See page 133.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0091	Timer Channel 0 Register Low (TC0L) See page 133.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0092	Timer Channel 1 Register High (TC1H) See page 133.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0093	Timer Channel 1 Register Low (TC1L) See page 133.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0094	Timer Channel 2 Register High (TC2H) See page 133.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0095	Timer Channel 2 Register Low (TC2L) See page 133.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0096	Timer Channel 3 Register High (TC3H) See page 133.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0097	Timer Channel 3 Register Low (TC3L) See page 133.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0098	Timer Channel 4 Register High (TC4H) See page 133.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-2. I/O Register Summary (Sheet 2 of 4)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0099	Timer Channel 4 Register Low (TC4L) See page 133.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$009A	Timer Channel 5 Register High (TC5H) See page 133.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$009B	Timer Channel 5 Register Low (TC5L) See page 133.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$009C	Timer Channel 6 Register High (TC6H) See page 133.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$009D	Timer Channel 6 Register Low (TC6L) See page 133.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$009E	Timer Channel 7 Register High (TC7H) See page 133.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$009F	Timer Channel 7 Register Low (TC7L) See page 133.	Read: Write:	Bit 7	6	5	4	3	2	1	0
		Reset:	0	0	0	0	0	0	0	0
$00A0	Pulse Accumulator Control Register (PACTL) See page 134.	Read:	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00A1	Pulse Accumulator Flag Register (PAFLG) See page 135.	Read:	0	0	0	0	0	0	PAOVF	PAIF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00A2	Pulse Accumulator Counter Register High (PACNTH) See page 136.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$00A3	Pulse Accumulator Counter Register Low (PACNTL) See page 136.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$00AD	Timer Test Register (TIMTST) See page 137.	Read:	0	0	0	0	0	0	TCBYP	PCBYP
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-2. I/O Register Summary (Sheet 3 of 4)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00AE	Timer Port Data Register (PORTT) See page 139.	Read: Write:	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
		Reset:	Unaffected by reset							
$00AF	Timer Port Data Direction Register (DDRT) See page 140.	Read: Write:	Bit 7	5	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-2. I/O Register Summary (Sheet 4 of 4)

12.4 Functional Description

This section provides a functional description of the standard timer.

12.4.1 Prescaler

The prescaler divides the module clock by 1, 2, 4, 8, 16, or 32. The prescaler select bits, PR2, PR1, and PR0, select the prescaler divisor. PR2, PR1, and PR0 are in the timer mask 2 register (TMSK2).

12.4.2 Input Capture

Clearing the I/O (input/output) select bit, IOSx, configures channel x as an input capture channel. The input capture function captures the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the timer transfers the value in the timer counter into the timer channel registers, TIMCxH and TIMCxL.

In 8-bit MCUs, the low byte of the timer channel register (TIMCxL) is held for one bus cycle after the high byte (TIMCxH) is read. This allows coherent reading of the timer channel such that an input capture does not occur between two back-to-back 8-bit reads. To read the 16-bit timer channel register, use a double-byte read instruction such as LDD or LDX.

The minimum pulse width for the input capture input is greater than two module clocks.

The input capture function does not force data direction. The timer port data direction register controls the data direction of an input capture pin. Pin conditions can trigger an input capture on a pin configured as an input. Software can trigger an input capture on an input capture pin configured as an output.

An input capture on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests.

12.4.3 Output Compare

Setting the I/O select bit, IOSx, configures channel x as an output compare channel. The output compare function can generate a periodic pulse with a programmable polarity, duration, and frequency. When the timer counter reaches the value in the channel registers of an output compare channel, the timer can set, clear, or toggle the channel pin. An output compare on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests.

The output mode and level bits, OMx and OLx, select set, clear, or toggle on output compare. Clearing both OMx and OLx disconnects the pin from the output logic.

Setting a force output compare bit, FOCx, causes an immediate output compare on channel x. A forced output compare does not set the channel flag.

An output compare on channel 7 overrides output compares on all other output compare channels. A channel 7 output compare causes any unmasked bits in the output compare 7 data register to transfer to the timer port data register. The output compare 7 mask register masks the bits in the output compare 7 data register. The timer counter reset enable bit, TCRE, enables channel 7 output compares to reset the timer counter. A channel 7 output compare can reset the timer counter even if the OC7/PAI pin is being used as the pulse accumulator input.

An output compare overrides the data direction bit of the output compare pin but does not change the state of the data direction bit.

Writing to the timer port bit of an output compare pin does not affect the pin state. The value written is stored in an internal latch. When the pin becomes available for general-purpose output, the last value written to the bit appears at the pin.

12.4.4 Pulse Accumulator

The pulse accumulator (PA) is a 16-bit counter that can operate in two modes:

- Event counter mode — Counting edges of selected polarity on the pulse accumulator input pin, PAI
- Gated time accumulation mode — Counting pulses from a divide-by-64 clock

The PA mode bit, PAMOD, selects the mode of operation.

The minimum pulse width for the PAI input is greater than two module clocks.

12.4.4.1 Event Counter Mode

Clearing the PAMOD bit configures the PA for event counter operation. An active edge on the PAI pin increments the PA. The PA edge bit, PEDGE, selects falling edges or rising edges to increment the PA.

An active edge on the PAI pin sets the PA input flag, PAIF. The PA input interrupt enable bit, PAI, enables the PAIF flag to generate interrupt requests.

NOTE

The PAI input and timer channel 7 use the same pin. To use the PAI input, disconnect it from the output logic by clearing the channel 7 output mode and output level bits, OM7 and OL7. Also clear the channel 7 output compare 7 mask bit, OC7M7.

The PA counter registers, TIMPACNTH/L, reflect the number of active input edges on the PAI pin since the last reset.

The PA overflow flag, PAOVF, is set when the PA rolls over from $FFFF to $0000. The PA overflow interrupt enable bit, PAOVI, enables the PAOVF flag to generate interrupt requests.

NOTE

The PA can operate in event counter mode even when the timer enable bit, TE, is clear.

12.4.4.2 Gated Time Accumulation Mode

Setting the PAMOD bit configures the PA for gated time accumulation operation. An active level on the PAI pin enables a divided-by-64 clock to drive the PA. The PA edge bit, PEDGE, selects low levels or high levels to enable the divided-by-64 clock.

The trailing edge of the active level at the PAI pin sets the PA input flag, PAIF. The PA input interrupt enable bit, PAI, enables the PAIF flag to generate interrupt requests.

NOTE

The PAI input and timer channel 7 use the same pin. To use the PAI input, disconnect it from the output logic by clearing the channel 7 output mode and output level bits, OM7 and OL7. Also clear the channel 7 output compare mask bit, OC7M7.

The PA counter registers, TIMPACNTH/L reflect the number of pulses from the divided-by-64 clock since the last reset.

NOTE

The timer prescaler generates the divided-by-64 clock. If the timer is not active, there is no divided-by-64 clock.

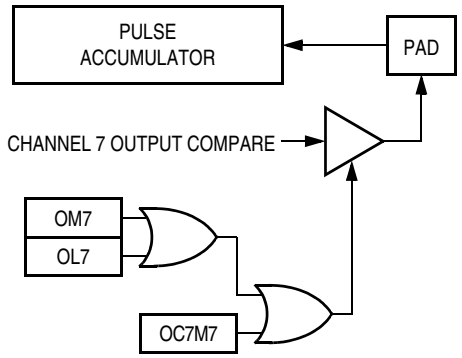


Figure 12-3. Channel 7 Output Compare/Pulse Accumulator Logic

12.5 Registers and Reset Initialization

This section describes the registers and reset initialization.

12.5.1 Timer IC/OC Select Register

Address: $0080

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
Reset:	0	0	0	0	0	0	0	0

Figure 12-4. Timer IC/OC Select Register (TIOS)

Read: Anytime
Write: Anytime

IOS7–IOS0 — Input Capture or Output Compare Select Bits

The IOSx bits enable input capture or output compare operation for the corresponding timer channel.

1 = Output compare enabled
0 = Input capture enabled

12.5.2 Timer Compare Force Register

Address: $0081

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
Reset:	0	0	0	0	0	0	0	0

Figure 12-5. Timer Compare Force Register (CFORC)

Read: Anytime; always read $00 (1 state is transient)
Write: Anytime

FOC7–FOC0 — Force Output Compare Bits

Setting an FOCx bit causes an immediate output compare on the corresponding channel. Forcing an output compare does not set the output compare flag.

1 = Force output compare
0 = No effect

12.5.3 Timer Output Compare 7 Mask Register

Address: $0082

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
Reset:	0	0	0	0	0	0	0	0

Figure 12-6. Timer Output Compare 7 Mask Register (OC7M)

Read: Anytime
Write: Anytime

OC7M7–OC7M0 — Output Compare 7 Mask Bits

Setting an OC7Mx bit configures the corresponding TIMPORT pin to be an output. OC7Mx makes the timer port pin an output regardless of the data direction bit when the pin is configured for output compare (IOSx = 1). The OC7Mx bits do not change the state of the TIMDDR bits.

1 = Corresponding TIMPORT pin output
0 = Corresponding TIMPORT pin input

12.5.4 Timer Output Compare 7 Data Register

Address: $0083

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
Reset:	0	0	0	0	0	0	0	0

Figure 12-7. Timer Output Compare 7 Data Register (OC7D)

Read: Anytime
Write: Anytime

OC7D7–OC7D0 — Output Compare Data Bits

When a successful channel 7 output compare occurs, these bits transfer to the timer port data register if the corresponding OC7Mx bits are set.

NOTE

A successful channel 7 output compare overrides any channel 0–6 compares. For each OC7M bit that is set, the output compare action reflects the corresponding OC7D bit.

12.5.5 Timer Counter Registers

Address: $0084

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-8. Timer Counter Register High (TCNTH)

Address: $0085

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-9. Timer Counter Register Low (TCNTL)

Read: Anytime
Write: Only in special modes; has no effect in normal modes

Use a double-byte read instruction to read the timer counter. Two single-byte reads return a different value than a double-byte read.

NOTE

The period of the first count after a write to the TCNT registers may be a different size because the write is not synchronized with the prescaler clock.

12.5.6 Timer System Control Register

Address: $0086

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TEN	TSWAI	TSBCK	TFFCA	0	0	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-10. Timer System Control Register (TSCR)

Read: Anytime
Write: Anytime

TEN — Timer Enable Bit

TEN enables the timer. Clearing TEN reduces power consumption.

1 = Timer enabled
0 = Timer and timer counter disabled

When the timer is disabled, there is no divided-by-64 clock for the PA since the prescaler generates the M ÷ 64 clock.

TSWAI — Timer Stop in Wait Mode Bit

TSWAI disables the timer and PA in wait mode.

1 = Timer and PA disabled in wait mode
0 = Timer and PA enabled in wait mode

NOTE

If timer and PA interrupt requests are needed to bring the MCU out of wait mode, clear TSWAI before executing the WAIT instruction.

TSBCK — Timer Stop in Background Mode Bit

TSBCK stops the timer during background mode.

1 = Timer disabled in background mode
0 = Timer enabled in background mode

NOTE

Setting TSBCK does not stop the PA when it is in event counter mode.

TFFCA — Timer Fast Flag Clear-All Bit

When TFFCA is set:

- An input capture read or a write to an output compare channel clears the corresponding channel flag, CnF.
- Any access of the timer counter registers, TCNTH/L, clears the TOF flag.
- Any access of the PA counter registers, PACNTH/L, clears both the PAOVF and PAIF flags in the PAFLG register.

When TFFCA is clear, writing logic 1s to the flags clears them.

1 = Fast flag clearing
0 = Normal flag clearing

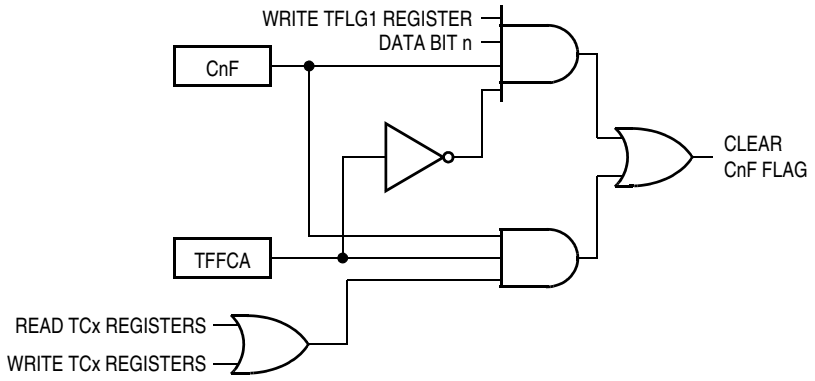


Figure 12-11. Fast Clear Flag Logic

12.5.7 Timer Control Registers 1 and 2

Address: $0088

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
Reset:	0	0	0	0	0	0	0	0

Figure 12-12. Timer Control Register 1 (TCTL1)

Address: $0089

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
Reset:	0	0	0	0	0	0	0	0

Figure 12-13. Timer Control Register 2 (TCTL2)

Read: Anytime
Write: Anytime

OMx/OLx — Output Mode/Output Level Bits

These bit pairs select the output action to be taken as a result of a successful output compare. When either OMx or OLx is set and the IOSx bit is set, the pin is an output regardless of the state of the corresponding DDRT bit.

Table 12-1. Selection of Output Compare Action

OMx:OLx	Action on Output Compare
00	Timer disconnected from output pin logic
01	Toggle OCn output line
10	Clear OCn output line
11	Set OCn output line

Channel 7 shares a pin with the pulse accumulator input pin. To use the PAI input, clear both the OM7 and OL7 bits and clear the OC7M7 bit in the output compare 7 mask register.

12.5.8 Timer Control Registers 3 and 4

Address: $008A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
Reset:	0	0	0	0	0	0	0	0

Figure 12-14. Timer Control Register 3 (TCTL3)

Address: $008B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
Reset:	0	0	0	0	0	0	0	0

Figure 12-15. Timer Control Register 4 (TCTL4)

Read: Anytime
Write: Anytime

EDGnB, EDGnA — Input Capture Edge Control Bits

These eight bit pairs configure the input capture edge detector circuits.

Table 12-2. Input Capture Edge Selection

EDGnB:EDGnA	Edge Selection
00	Input capture disabled
01	Input capture on rising edges only
10	Input capture on falling edges only
11	Input capture on any edge (rising or falling)

12.5.9 Timer Mask Register 1

Address: $008C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
Reset:	0	0	0	0	0	0	0	0

Figure 12-16. Timer Mask 1 Register (TMSK1)

Read: Anytime
Write: Anytime

C7I–C0I — Channel Interrupt Enable Bits

These bits enable the flags in timer flag register 1.

1 = Corresponding channel interrupt requests enabled
0 = Corresponding channel interrupt requests disabled

12.5.10 Timer Mask Register 2

Address: $008D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOI	0	PUPT	RDPT	TCRE	PR2	PR1	PR0
Write:								
Reset:	0	0	1	0	0	0	0	0

= Unimplemented

Figure 12-17. Timer Mask 2 Register (TMSK2)

Read: Anytime
Write: Anytime

TOI — Timer Overflow Interrupt Enable Bit

TOI enables interrupt requests generated by the TOF flag.

1 = TOF interrupt requests enabled
0 = TOF interrupt requests disabled

PUPT — Port T Pullup Enable Bit

PUPT enables pullup resistors on the timer port pins when the pins are configured as inputs.

1 = Pullup resistors enabled
0 = Pullup resistors disabled

RDPT — Port T Reduced Drive Bit

RDPT reduces the output driver size for lower current and less noise.

1 = Output drive reduction enabled
0 = Output drive reduction disabled

TCRE — Timer Counter Reset Enable Bit

TCRE allows the counter to be reset by a channel 7 output compare.

1 = Counter reset enabled
0 = Counter reset disabled

NOTE

When the timer channel 7 registers contain $0000 and TCRE is set, the timer counter registers remain at $0000 all the time.

When the timer channel 7 registers contain $FFFF and TCRE is set, TOF never gets set even though the timer counter registers go from $FFFF to $0000.

PR2, PR1, and PR0 — Timer Prescaler Select Bits

These bits select the prescaler divisor for the timer counter.

Table 12-3. Prescaler Selection

Value	PR[2:1:0]	Prescaler Divisor
0	000	1
1	001	2
2	010	4
3	011	8
4	100	16

Table 12-3. Prescaler Selection (Continued)

Value	PR[2:1:0]	Prescaler Divisor
5	101	32
6	110	32
7	111	32

NOTE

The newly selected prescale divisor does not take effect until the next synchronized edge when all prescale counter stages equal 0.

12.5.11 Timer Flag Register 1

Address: $008E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
Reset:	0	0	0	0	0	0	0	0

Figure 12-18. Timer Flag Register 1 (TFLG1)

Read: Anytime
Write: Anytime; writing 1 clears flag; writing 0 has no effect

C7F–C0F — Channel Flags

These flags are set when an input capture or output compare occurs on the corresponding channel. Clear a channel flag by writing a 1 to it.

NOTE

When the fast flag clear-all bit, TFFCA, is set, an input capture read or an output compare write clears the corresponding channel flag. TFFCA is in the timer system control register (TSCR).

12.5.12 Timer Flag Register 2

Address: $008F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	0	0	0	0	0	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-19. Timer Flag Register 2 (TFLG2)

Read: Anytime
Write: Anytime; writing 1 clears flag; writing 0 has no effect

TOF — Timer Overflow Flag

TOF is set when the timer counter rolls over from $FFFF to $0000. Clear TOF by writing a 1 to it.

1 = Timer overflow

0 = No timer overflow

NOTE

When the timer channel 7 registers contain $FFFF and the timer counter reset enable bit, TCRE, is set, TOF does not get set when the counter rolls over.

NOTE

When the fast flag clear-all bit, TFFCA, is set, any access to the timer counter registers clears TOF.

12.5.13 Timer Channel Registers

Address: TC0H/L: $0090/$0091
TC1H/L: $0092/$0093
TC2H/L: $0094/$0095
TC3H/L: $0096/$0097
TC4H/L: $0098/$0099
TC5H/L: $009A/$009B
TC6H/L: $009C/$009D
TC7H/L: $009E/$009F

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	14	13	12	11	10	9	8
Reset:	0	0	0	0	0	0	0	0

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 12-20. Timer Channel Registers (TCxH/L)

Read: Anytime
Write: Output compare channel, anytime; input capture channel, no effect

When a channel is configured for input capture (IOSx = 0), the timer channel registers latch the value of the free-running counter when a defined transition occurs on the corresponding input capture pin.

When a channel is configured for output compare (IOSx = 1), the timer channel registers contain the output compare value.

12.5.14 Pulse Accumulator Control Register

Address: $00A0

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-21. Pulse Accumulator Control Register (PACTL)

Read: Anytime
Write: Anytime

PAEN — Pulse Accumulator Enable Bit

PAEN enables the pulse accumulator.

1 = Pulse accumulator enabled
0 = Pulse accumulator disabled

NOTE

The pulse accumulator can operate even when the timer enable bit, TEN, is clear.

PAMOD — Pulse Accumulator Mode Bit

PAMOD selects event counter mode or gated time accumulation mode.

1 = Gated time accumulation mode
0 = Event counter mode

PEDGE — Pulse Accumulator Edge Bit

PEDGE selects falling or rising edges on the PAI pin to increment the counter.

In event counter mode (PAMOD = 0):

1 = Rising PAI edge increments counter
0 = Falling PAI edge increments counter

In gated time accumulation mode (PAMOD = 1):

1 = Low PAI input enables divided-by-64 clock to pulse accumulator and trailing rising edge on PAI sets PAIF flag
0 = High PAI input enables divided-by-64 clock to pulse accumulator and trailing falling edge on PAI sets PAIF flag

NOTE

The timer prescaler generates the divided-by-64 clock. If the timer is not active, there is no divided-by-64 clock.

To operate in gated time accumulation mode:

1. Apply logic 0 to the $\overline{\text{RESET}}$ pin.
2. Initialize registers for pulse accumulator mode test.
3. Apply appropriate level on PAI pin.
4. Enable the timer.

CLK1 and CLK0 — Clock Select Bits

CLK1 and CLK0 select the timer counter input clock as shown in Table 12-4.

Table 12-4. Clock Selection

CLK[1:0]	Timer Counter Clock(1)
00	Timer prescaler clock(2)
01	PACLK
10	PACLK / 256
11	PACLK / 65,536

1. Changing the CLKx bits causes an immediate change in the timer counter clock input.
2. When PAE = 0, the timer prescaler clock is always the timer counter clock.

PAOVI — Pulse Accumulator Overflow Interrupt Enable Bit

PAOVI enables the pulse accumulator overflow flag, PAOVF, to generate interrupt requests.

1 = PAOVF interrupt requests enabled
0 = PAOVF interrupt requests disabled

PAI — Pulse Accumulator Interrupt Enable Bit

PAI enables the pulse accumulator input flag, PAIF, to generate interrupt requests.

1 = PAIF interrupt requests enabled
0 = PAIF interrupt requests disabled

12.5.15 Pulse Accumulator Flag Register

Address: $00A1

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	PAOVF	PAIF
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-22. Pulse Accumulator Flag Register (PAFLG)

Read: Anytime
Write: Anytime; writing 1 clears the flag; writing 0 has no effect

PAOVF — Pulse Accumulator Overflow Flag

PAOVF is set when the 16-bit pulse accumulator overflows from $FFFF to $0000. Clear PAOVF by writing to the pulse accumulator flag register with PAOVF set.

1 = Pulse accumulator overflow
0 = No pulse accumulator overflow

PAIF — Pulse Accumulator Input Flag

PAIF is set when the selected edge is detected at the PAI pin. In event counter mode, the event edge sets PAIF. In gated time accumulation mode, the trailing edge of the gate signal at the PAI pin sets PAIF. Clear PAIF by writing to the pulse accumulator flag register with PAIF set.

1 = Active PAI input
0 = No active PAI input

NOTE

When the fast flag clear-all enable bit, TFFCA, is set, any access to the pulse accumulator counter registers clears all the flags in the PAFLG register.

12.5.16 Pulse Accumulator Counter Registers

Address: $00A2

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset:	0	0	0	0	0	0	0	0

Address: $00A3

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 12-23. Pulse Accumulator Counter Registers (PACNTH/L)

Read: Anytime
Write: Anytime

These registers contain the number of active input edges on the PAI pin since the last reset.

Use a double-byte read instruction to read the pulse accumulator counter. Two single-byte reads return a different value than a double-byte read.

NOTE

Reading the pulse accumulator counter registers immediately after an active edge on the PAI pin may miss the last count since the input has to be synchronized with the bus clock first.

12.5.17 Timer Test Register

Address: $00AD

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	TCBYP	PCBYP
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-24. Timer Test Register (TIMTST)

Read: Anytime
Write: Only in special mode (SMODN = 0)

TCBYP — Timer Divider Chain Bypass Bit

TCBYP divides the 16-bit free-running timer counter into two 8-bit halves. The clock drives both halves directly and bypasses the timer prescaler.

1 = Timer counter divided in half and prescaler bypassed
0 = Normal operation

PCBYP — Pulse Accumulator Divider Chain Bypass Bit

PCBYP divides the 16-bit PA counter into two 8-bit halves. The clock drives both halves directly and bypasses the timer prescaler.

1 = PA counter divided in half and prescaler bypassed
0 = Normal operation

12.6 External Pins

The timer has eight pins for input capture and output compare functions. One of the pins is also the pulse accumulator input. All eight pins are available for general-purpose I/O when not configured for timer functions.

12.6.1 Input Capture/Output Compare Pins

The IOSx bits in the timer IC/OC select register configure the timer port pins as either output compare or input capture pins.

The timer port data direction register controls the data direction of an input capture pin. External pin conditions trigger input captures on input capture pins configured as inputs. Software triggers input captures on input capture pins configured as outputs.

The timer port data direction register does not affect the data direction of an output compare pin. The output compare function overrides the data direction register but does not affect the state of the data direction register.

12.6.2 Pulse Accumulator Pin

Setting the PAE bit in the pulse accumulator control register enables the pulse accumulator input pin, PAI.

NOTE

The PAI input and timer channel 7 use the same pin. To use the PAI input, disconnect it from the output logic by clearing the channel 7 output mode and output level bits, OM7 and OL7. Also clear the channel 7 output compare mask bit, OC7M7.

12.7 Background Debug Mode

If the TSBCK bit is clear, background debug mode has no effect on the timer. If TSBCK is set, background debug mode disables the timer.

NOTE

Setting TSBCK does not stop the pulse accumulator when it is in event counter mode.

12.8 Low-Power Options

This section describes the three low-power modes:

- Run mode
- Wait mode
- Stop mode

12.8.1 Run Mode

Clearing the timer enable bit (TEN) or the pulse accumulator enable bit (PAEN) reduces power consumption in run mode. TEN is in the timer system control register (TSCR). PAEN is in the pulse accumulator control register (PACTL). Timer and pulse accumulator registers are still accessible, but clocks to the core of the timer are disabled.

12.8.2 Wait Mode

Timer and pulse accumulator operation in wait mode depend on the state of the TSWAI bit in the timer system control register TSCR).

- If TSWAI is clear, the timer and pulse accumulator operate normally when the CPU is in wait mode.
- If TSWAI is set, timer and pulse accumulator clock generation ceases and the TIM module enters a power-conservation state when the CPU is in wait mode. In this condition, timer and pulse accumulator registers are not accessible. Setting TSWAI does not affect the state of the timer enable bit, TEN, or the pulse accumulator enable bit, PAEN.

12.8.3 Stop Mode

The STOP instruction disables the timer for reduced power consumption.

12.9 Interrupt Sources

Table 12-5. Timer Interrupt Sources

Interrupt Source	Flag	Local Enable	CCR Mask	Vector Address
Timer channel 0	C0F	C0I	I bit	$FFEE, $FFEF
Timer channel 1	C1F	C1I	I bit	$FFEC, $FFED
Timer channel 2	C2F	C2I	I bit	$FFEA, $FFEB
Timer channel 3	C3F	C3I	I bit	$FFE8, $FFE9
Timer channel 4	C4F	C4I	I bit	$FFE6, $FFE7
Timer channel 5	C5F	C5I	I bit	$FFE4, $FFE5
Timer channel 6	C6F	C6I	I bit	$FFE2, $FFE3
Timer channel 7	C7F	C7I	I bit	$FFE0, $FFE1
Timer overflow	TOF	TOI	I bit	$FFDE, $FFDF
Pulse accumulator overflow	PAOVF	PAOVI	I bit	$FFDC, $FFDD
Pulse accumulator input	PAIF	PAI	I bit	$FFDA, $FFDB

12.10 General-Purpose I/O Ports

This section describes the general-purpose I/O ports.

12.10.1 Timer Port Data Register

An I/O pin used by the timer defaults to general-purpose I/O unless an internal function which uses that pin is enabled.

Address: $00AE

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
Reset:	Unaffected by reset							
Timer function:	IC/OC7	IC/OC6	IC/OC5	IC4OC4	IC/OC3	IC/OC2	IC/OC1	IC/OC0
PA function:	PAI							

Figure 12-25. Timer Port Data Register (PORTT)

Read: Anytime
Write: Anytime

PT7–PT0 — Timer Port Data Bits

Data written to PORTT is buffered and drives the pins only when they are configured as general-purpose outputs.

Reading an input (data direction bit = 0) reads the pin state; reading an output (data direction bit = 1) reads the latch.

Writing to a pin configured as a timer output does not change the pin state.

NOTE

Due to input synchronizer circuitry, the minimum pulse width for a pulse accumulator input or an input capture input should always be greater than the width of two module clocks.

Table 12-6. TIMPORT I/O Function

In		Out	
Data Direction Register	**Output Compare Action**	**Reading at Data Bus**	**Reading at Pin**
0	0	Pin	Pin
0	1	Pin	Output compare action
1	1	Port data register	Output compare action
1	0	Port data register	Port data register

12.10.2 Timer Port Data Direction Register

Address: $00AF

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 12-26. Timer Port Data Direction Register (DDRT)

Read: Anytime
Write: Anytime

Bits 7–0 — TIMPORT Data Direction Bits

These bits control the port logic of PORTT. Reset clears the timer port data direction register, configuring all timer port pins as inputs.

1 = Corresponding pin configured as output
0 = Corresponding pin configured as input

The timer forces the I/O state to be an output for each timer port pin associated with an enabled output compare. In these cases, the data direction bits do not change but have no effect on the direction of these pins. The DDRT reverts to controlling the I/O direction of a pin when the associated timer output compare is disabled. Input captures do not override the DDRT settings.

NOTE

By setting the IOSx bit input capture configuration no matter what the state of the data direction register is, the timer forces output compare pins to be outputs and input capture pins to be inputs.

12.11 Using the Output Compare Function to Generate a Square Wave

This timer exercise is intended to utilize the output compare function to generate a square wave of predetermined duty cycle and frequency.

Square wave frequency 1000 Hz, duty cycle 50%

The program generates a square wave, 50 percent duty cycle, on output compare 2 (OC2). The signal will be measured by the M68HC11 on the UDLP1 board. It assumes a 8.0 MHz operating frequency for the E clock. The control registers are initialized to disable interrupts, configure for proper pin control action and also the TC2H register for desired compare value. The appropiate count must be calculated to achieve the desired frequency and duty cycle. For example: for a 50 percent duty, 1 kHz signal each period must consist of 2048 counts or 1024 counts high and 1024 counts low. In essence a $0400 is added to generate a frequency of 1 kHz.

12.11.1 Sample Calculation to Obtain Period Counts

The sample calculation to obtain period counts is:

- For 1000 Hz frequency:
  - E-clock = 8 MHz
  - IC/OC resolution factor = 1/(E-clock/prescaler)
  - If the prescaler = 4, then output compare resolution is 0.5 µs
- For a 1 kHz, 50 percent duty cycle:
  - 1/F = T = 1/1000 = 1 ms
  - F for output compare = prescaler/E clock = 2 MHz

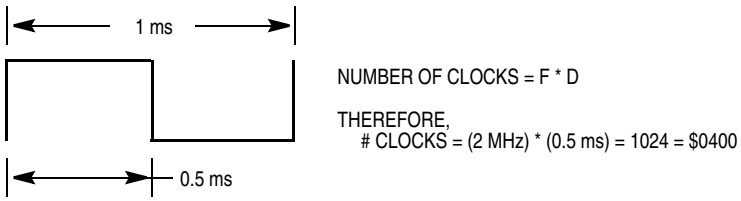


Figure 12-27. Example Waveform

12.11.2 Equipment

For this exercise, use the M68HC812A4EVB emulation board.

12.11.3 Code Listing

NOTE

A comment line is delimited by a semicolon. If there is no code before comment, a semicolon (;) must be placed in the first column to avoid assembly errors.

```
------------------------------------------------------------------------
;            MAIN PROGRAM
; ----------------------------------------------------------------------
         ORG     $7000          ; 16K On-Board RAM, User code data area,
;                               ; start main program at $7000
MAIN:
         BSR     TIMERINIT      ; Subroutine used to initialize the timer:
;                               ; Output compare channel, no interrupts
         BSR     SQWAVE         ; Subroutine to generate square wave
DONE:    BRA     DONE           ; Branch to itself, Convinient for Breakpoint

;* ---------------------------------------------------------------
;*  Subroutine TIMERINIT: Initialize Timer for Output Compare on OC2
;* ---------------------------------------------------------------
TIMERINIT:
         CLR     TMSK1          ; Disable All Interrupts

         MOVB    #$02,TMSK2     ; Disable overflow interrupt, disable pull-up
;                               ; resistor function with normal drive capability
;                               ;and free running counter, Prescaler = sys clock/4.

         MOVB    #$10,TCTL2     ; Initialize OC2 to toggle on successful compare.

         MOVB    #$04,TIOS      ; Select Channel 2 to act as output compare.

         MOVW    #$0400,TC2H    ; Load TC2 Reg with initial compare value.

         MOVB    #$80,TSCR      ; Enable Timer, Timer runs during wait state, and
;                               ; while in Background Mode, also clear flags
;                               ; normally.

         RTS                    ; Return from Subroutine

;* ------------------------------
;*      SUBROUTINE:  SQWAVE
;* ------------------------------
SQWAVE:

;* -------
CLEARFLG:
;* -------
;*  To clear the C2F flag: 1) read TFLG1 when
;*  C2F is set and then 2) write a logic "one" to C2F.

         LDAA    TFLG1          ; To clear OC2 Flag, first it must be read,
         ORAA    #$04           ; then a "1" must be written to it
         STAA    TFLG1

WTFLG:   BRCLR   TFLG1,#$04,WTFLG; Wait (Polling) for C2F Flag

         LDD     TC2H           ; Loads value of compare from TC2 Reg.
         ADDD    #$0400         ; Add hex value of 500us High Time
         STD     TC2H           ; Set-up next transition time in 500 us

         BRA     CLEARFLG       ; Continuously add 500 us, branch to CLEARFLAG

         RTS                    ; return from Subroutine

         END                    ; End of program
```

Chapter 13
Multiple Serial Interface (MSI)

13.1 Introduction

The multiple serial interface (MSI) module consists of three independent serial I/O interfaces:

- Two serial communication interfaces, SCI0 and SCI1
- One serial peripheral interface, SPI0

NOTE

Port S shares its pins with the multiple serial interface (MSI).
See 13.6 General-Purpose I/O Ports.

13.2 SCI Features

Serial comunication interface (SCI) features include:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 13-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Separate receiver and transmitter interrupt requests
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Five flags with interrupt-generation capability:
  - Transmitter empty
  - Transmission complete
  - Receiver full
  - Receiver overrun
  - Idle receiver input
- Receiver noise error detection
- Receiver framing error detection
- Receiver parity error detection

For additional information, refer to Chapter 14 Serial Communications Interface Module (SCI).

13.3 SPI Features

Serial preipheral interface (SPI) fetures include:

- Full-duplex operation
- Master mode and slave mode
- Programmable slave-select output option
- Programmable bidirectional data pin option
- Interrupt-driven operation with two flags:
  - Transmission complete
  - Mode fault
- Read data buffer
- Serial clock with programmable polarity and phase
- Reduced drive control for lower power consumption
- Programmable open-drain output option

For additional information, refer to Chapter 15 Serial Peripheral Interface (SPI)

13.4 MSI Block Diagram

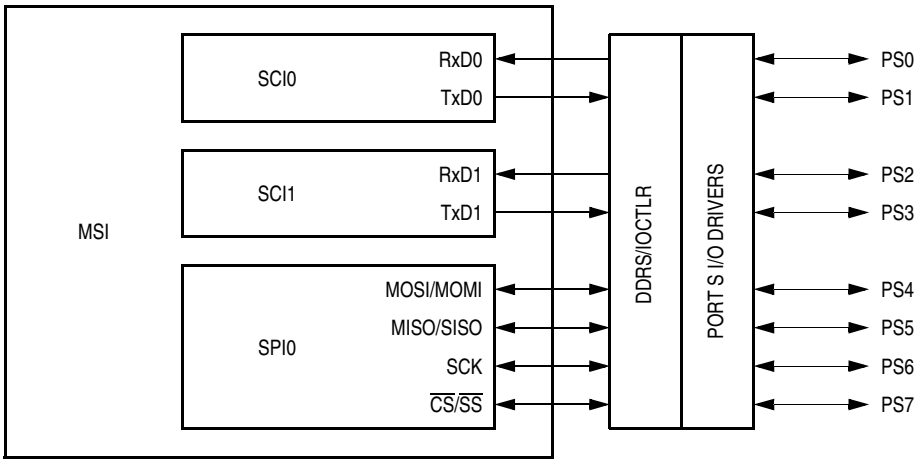


Figure 13-1. Multiple Serial Interface Block Diagram

13.5 MSI Register Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00C0	SCI 0 Baud Rate Register High (SC0BDH) See page 168.	Read: Write:	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8
		Reset:	0	0	0	0	0	0	0	0
$00C1	SCI 0 Baud Rate Register Low (SC0BDL) See page 168.	Read: Write:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Reset:	0	0	0	0	0	1	0	0
$00C2	SCI 0 Control Register 1 (SC0CR1) See page 169.	Read: Write:	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT
		Reset:	0	0	0	0	0	0	0	0
$00C3	SCI 0 Control Register 2 (SC0CR2) See page 171.	Read: Write:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Reset:	0	0	0	0	0	0	0	0
$00C4	SCI 0 Status Register 1 (SC0SR1) See page 172.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$00C5	SCI 0 Status Register 2 (SC0SR2) See page 173.	Read:	0	0	0	0	0	0	0	RAF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C6	SCI 0 Data Register High (SC0DRH) See page 174.	Read:	R8	T8	0	0	0	0	0	0
		Write:								
		Reset:	Unaffected by reset							
$00C7	SCI 0 Data Register Low (SC0DRL) See page 174.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$00C8	SCI 1 Baud Rate Register High (SC1BDH) See page 168.	Read: Write:	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8
		Reset:	0	0	0	0	0	0	0	0
$00C9	SCI 1 Baud Rate Register Low (SC1BDL) See page 168.	Read: Write:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Reset:	0	0	0	0	0	1	0	0
$00CA	SCI 1 Control Register 1 (SC1CR1) See page 169.	Read: Write:	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT
		Reset:	0	0	0	0	0	0	0	0
$00CB	SCI 1 Control Register 2 (SC1CR2) See page 171.	Read: Write:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 13-2. MSI Register Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00CC	SCI 1 Status Register 1 (SC1SR1) See page 172.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$00CD	SCI 1 Status Register 2 (SC1SR2) See page 173.	Read:	0	0	0	0	0	0	0	RAF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00CE	SCI 1 Data Register High (SC1DRH) See page 174.	Read:	R8	T8	0	0	0	0	0	0
		Write:								
		Reset:	Unaffected by reset							
$00CF	SCI 1 Data Register Low (SC1DRL) See page 174.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$00D0	SPI 0 Control Register 1 (SP0CR1) See page 186.	Read: Write:	SPIE	SPE	SWOM	MSTR	CPOL	CPHA	SSOE	LSBF
		Reset:	0	0	0	0	0	0	0	0
$00D1	SPI 0 Control Register 2 (SP0CR2) See page 187.	Read:	0	0	0	0	PUPS	RDS	0	SPC0
		Write:								
		Reset:	0	0	0	0	1	0	0	0
$00D2	SPI 0 Baud Rate Register (SP0BR) See page 188.	Read:	0	0	0	0	0	SPR2	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00D3	SPI 0 Status Register (SP0SR) See page 189.	Read:	SPIF	WCOL	0	MODF	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00D5	SPI 0 Data Register (SP0DR) See page 190.	Read: Write:	Bit 7	6	5	4	3	2	1	0
		Reset:	Unaffected by reset							
$00D6	Port S Data Register (PORTS) See page 147.	Read: Write:	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
		Reset:	Unaffected by reset							
$00D7	Port S Data Direction Register (DDRS) See page 148.	Read: Write:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 13-2. MSI Register Map (Continued)

Full register descriptions can be found in Chapter 14 Serial Communications Interface Module (SCI) and Chapter 15 Serial Peripheral Interface (SPI).

13.6 General-Purpose I/O Ports

Port S shares its pins with the multiple serial interface (MSI). In all modes, port S pins PS7–PS0 are available for either general-purpose I/O or for SCI and SPI functions.

13.6.1 Port S Data Register

Address: $00D6

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
Reset:	Unaffected by reset							
Pin function:	$\overline{SS}$	SCK	MOSI	MISO	TXD1	RXD1	TXD0	RXD0

Figure 13-3. Port S Data Register (PORTS)

Read: Anytime
Write: Anytime

PS7–PS4 — Port S Data Bits 7–4

Port S shares PS7–PS4 with SPI0.

$\overline{SS}$ is the SPI0 slave-select terminal.
SCK is the SPI0 serial clock terminal.
MOSI is the SPI0 master out, slave in terminal.
MISO is the SPI0 master in, slave out terminal.

PS3–PS0 — Port S Data Bits 3–0

Port S shares PS3–0 with SCI1 and SCI0.

TXD1 is the SCI1 transmit terminal.
RXD1 is the SCI1 receive terminal.
TXD0 is the SCI0 transmit terminal.
RXD0 is the SCI0 receive terminal.

NOTE

Reading a port S bit when its data direction bit is clear returns the level of the voltage on the pin. Reading a port S bit when its data direction bit is set returns the level of the voltage of the pin driver input.

A write to a port S bit is stored in an internal latch. The latch drives the pin only when the corresponding data direction bit is set.

Writes do not change the pin state when the pin is configured for SCI output.

13.6.2 Port S Data Direction Register

Address: $00D7

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
Reset:	0	0	0	0	0	0	0	0

Figure 13-4. Port S Data Direction Register (DDRS)

Read: Anytime
Write: Anytime

DDRS7–DDRS0 — Port S Data Direction Bits

These bits control the data direction of each port S pin. Setting a DDRS bit makes the pin an output; clearing a DDRS bit makes the pin an input. Reset clears the port S data direction register, configuring all port S pins as inputs.

1 = Corresponding port S pin configured as output
0 = Corresponding port S pin configured as input

NOTE

When the LOOPS bit is clear, the RX pins of SCI0 and SCI1 are inputs and the TX pins are outputs regardless of their DDRS bits.

When the SPI is enabled, an SPI input pin is an input regardless of its DDRS bit.

When the SPI is enabled, an SPI output is an output only if its DDRS bit is set. When the DDRS bit of an SPI output is clear, the pin is available for general-purpose I/O.

13.6.3 Port S Pullup and Reduced Drive Control

Address: $00D1

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	PUPS	RDS	0	SPC0
Write:								
Reset:	0	0	0	0	1	0	0	0

= Unimplemented

Figure 13-5. SPI Control Register 2 (SP0CR2)

Read: Anytime
Write: Anytime

PUPS — Pullup Port S Enable Bit

Setting PUPS enables internal pullup devices on all port S input pins. If a pin is programmed as output, the pullup device becomes inactive.

1 = Pullups enabled
0 = Pullups disabled

RDS — Reduced Drive Port S Bit

Setting RDS lowers the drive capability of all port S output pins for lower power consumption and less noise.

1 = Reduced drive
0 = Full drive

Table 13-1. Port S Pullup and Reduced Drive Enable

Register	Pullups			Reduced Drive		
	Control Bit	Pins Affected	Reset State	Control Bit	Pins Affected	Reset State
SPI control register 2 (SP0CR2)	PUPS	PS7–PS0	Enabled	RDS	PS7–PS0	Disabled

SPC0 — See Chapter 14 Serial Communications Interface Module (SCI).

13.6.4 Port S Wired-OR Mode Control

Table 13-2. Port S Wired-OR Mode Enable

Register	Control Bit	Pins Affected	Reset State
SPI control register 1 (SP0CR1)	SWOM	PS7–PS4	Disabled
SCI0 control register 1 (SC0CR1)	WOMS	PS3, PS2	Disabled
SCI1 control register 1 (SC1CR1)	WOMS	PS1, PS0	Disabled

Chapter 14
Serial Communications Interface Module (SCI)

14.1 Introduction

The serial communications interface (SCI) allows asynchronous serial communications with peripheral devices and other MCUs.

14.2 Features

Features of the SCI include:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 13-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Separate receiver and transmitter interrupt requests
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Five flags with interrupt-generation capability:
  - Transmitter empty
  - Transmission complete
  - Receiver full
  - Receiver overrun
  - Idle receiver input
- Receiver noise error detection
- Receiver framing error detection
- Receiver parity error detection

14.3 Block Diagram

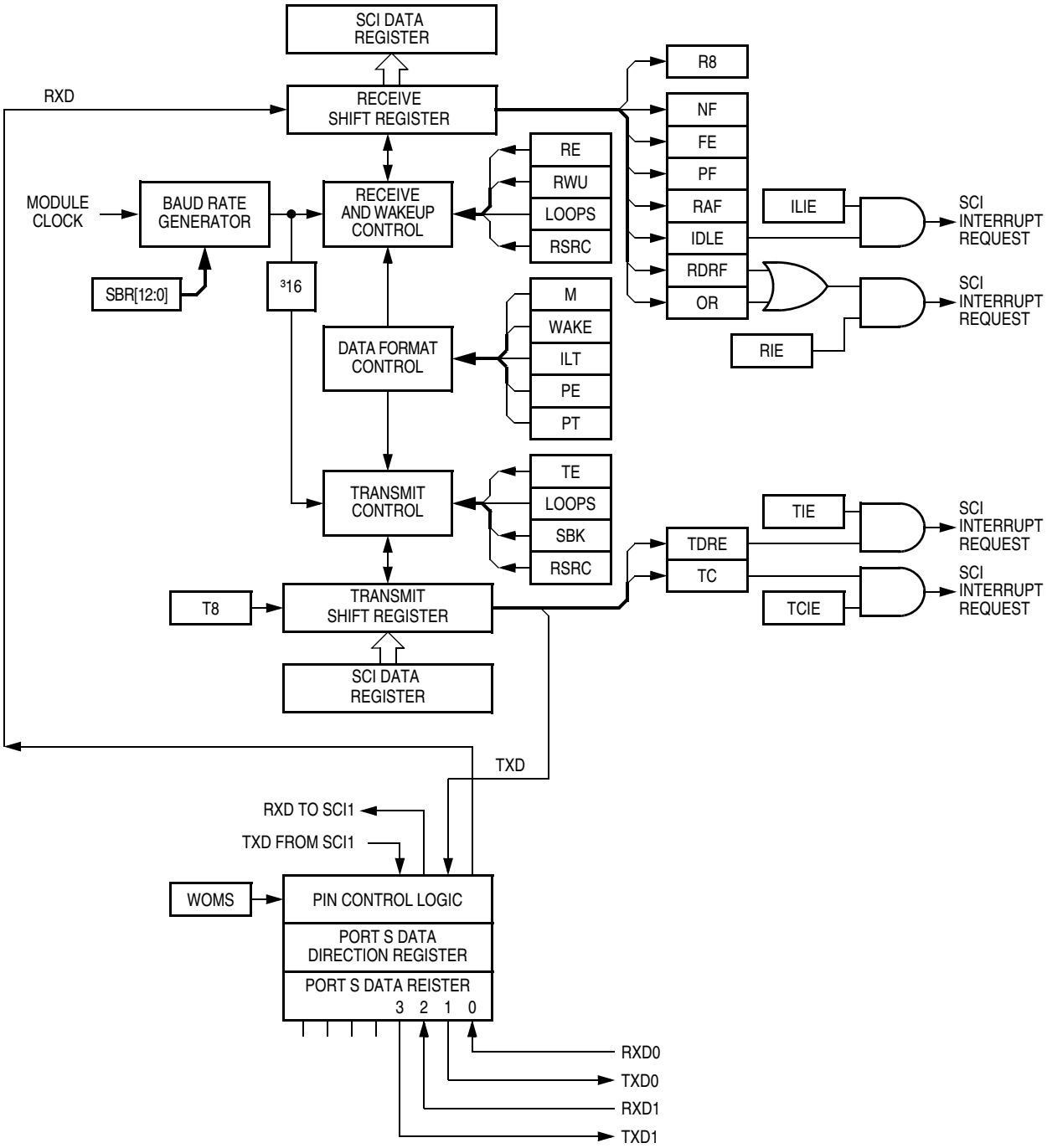


Figure 14-1. SCI Block Diagram

14.4 Register Map

NOTE

The register block can be mapped to any 2-Kbyte boundary within the standard 64-Kbyte address space. The register block occupies the first 512 bytes of the 2-Kbyte block. This register map shows default addressing after reset.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00C0	SCI 0 Baud Rate Register High (SC0BDH) See page 168.	Read:	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C1	SCI 0 Baud Rate Register Low (SC0BDL) See page 168.	Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Write:								
		Reset:	0	0	0	0	0	1	0	0
$00C2	SCI 0 Control Register 1 (SC0CR1) See page 169.	Read:	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C3	SCI 0 Control Register 2 (SC0CR2) See page 171.	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C4	SCI 0 Status Register 1 (SC0SR1) See page 172.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$00C5	SCI 0 Status Register 2 (SC0SR2) See page 173.	Read:	0	0	0	0	0	0	0	RAF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00C6	SCI 0 Data Register High (SC0DRH) See page 174.	Read:	R8	T8	0	0	0	0	0	0
		Write:								
		Reset:	Unaffected by reset							
$00C7	SCI 0 Data Register Low (SC0DRL) See page 174.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$00C8	SCI 1 Baud Rate Register High (SC1BDH) See page 168.	Read:	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 14-2. SCI Register Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00C9	SCI 1 Baud Rate Register Low (SC1BDL) See page 168.	Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Write:								
		Reset:	0	0	0	0	0	1	0	0
$00CA	SCI 1 Control Register 1 (SC1CR1) See page 169.	Read:	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00CB	SCI 1 Control Register 2 (SC1CR2) See page 171.	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00CC	SCI 1 Status Register 1 (SC1SR1) See page 172.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$00CD	SCI 1 Status Register 2 (SC1SR2) See page 173.	Read:	0	0	0	0	0	0	0	RAF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00CE	SCI 1 Data Register High (SC1DRH) See page 174.	Read:	R8	T8	0	0	0	0	0	0
		Write:								
		Reset:	Unaffected by reset							
$00CF	SCI 1 Data Register Low (SC1DRL) See page 174.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							

= Unimplemented

Figure 14-2. SCI Register Map (Continued)

14.5 Functional Description

The SCI allows full-duplex, asynchronous, NRZ serial communication between the MCU and remote devices, including other MCUs. The SCI transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

14.5.1 Data Format

The SCI uses the standard NRZ mark/space data format illustrated in Figure 14-3.

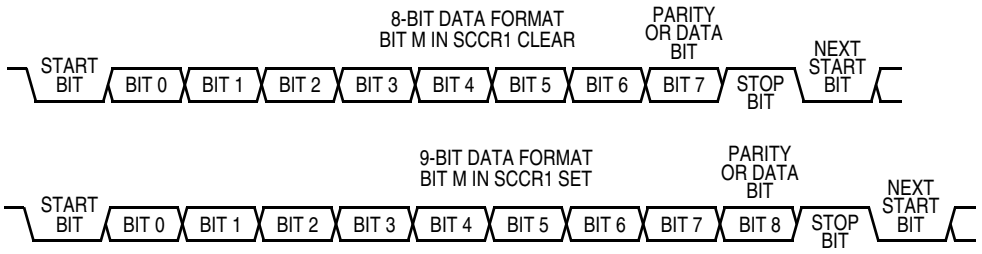


Figure 14-3. SCI Data Formats

Each data character is contained in a frame that includes a start bit, eight or nine data bits, and a stop bit. Clearing the M bit in SCI control register 1 configures the SCI for 8-bit data characters. A frame with eight data bits has a total of 10 bits. Setting the M bit configures the SCI for 9-bit data characters. A frame with nine data bits has a total of 11 bits.

Table 14-1. Example 8-Bit Data Formats

Start Bit	Data Bits	Address Bit	Parity Bit	Stop Bit
1	8	0	0	1
1	7	0		2
1	7	0	1	1
1	7	1[1]		1

1. The address bit identifies the frame as an address character. See 14.5.4.6 Receiver Wakeup.

Setting the M bit configures the SCI for 9-bit data characters. The ninth data bit is the T8 bit in SCI data register high (SCDRH). It remains unchanged after transmission and can be used repeatedly without rewriting it. A frame with nine data bits has a total of 11 bits.

Table 14-2. Example 9-Bit Data Formats

Start Bit	Data Bits	Address Bit	Parity Bit	Stop Bit
1	9	0	0	1
1	8	0		2
1	8	0	1	1
1	8	1[1]		1

1. The address bit identifies the frame as an address character. See 14.5.4.6 Receiver Wakeup.

14.5.2 Baud Rate Generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 0 to 8191 written to the SBR12–SBR0 bits determines the module clock divisor. The SBR bits are in the SCI baud rate registers (SCBDH and SCBDL). The baud rate clock is synchronized with the bus clock and drives the receiver. The baud rate clock divided by 16 drives the transmitter. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to two sources of error:

- Integer division of the module clock may not give the exact target frequency.
- Synchonization with the bus clock can cause phase shift.

Table 14-3 lists some examples of achieving target baud rates with a module clock frequency of 10.2 MHz.

Table 14-3. Baud Rates (Module Clock = 10.2 MHz)

Baud Rate Divisor(1)	Receiver Clock Rate (Hz)(2)	Transmitter Clock Rate (Hz)(3)	Target Baud Rate	Error (%)
17	600,000.0	37,500.0	38,400	2.3
33	309,090.9	19,318.2	19,200	0.62
66	154,545.5	9659.1	9600	0.62
133	76,691.7	4793.2	4800	0.14
266	38,345.9	2396.6	2400	0.14
531	19,209.0	1200.6	1200	0.11
1062	9604.5	600.3	600	0.05
2125	4800.0	300.0	300	0.00
4250	2400.0	150.0	150	0.00
5795	1760.1	110.0	110	0.00

1. The baud rate divisor is the value written to the SBR12–SBR0 bits.
2. The receiver clock frequency is the MCLK frequency divided by the baud rate divisor.
3. The transmitter clock frequency is the receiver clock frequency divided by 16.

14.5.3 Transmitter

A block diagram of the SCI transmitter is shown in Figure 14-4.

14.5.3.1 Character Length

The SCI transmitter can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCCR1) determines the length of data characters. When transmitting 9-bit data, bit T8 in SCI data register high (SCDRH) is the ninth bit (bit 8).

14.5.3.2 Character Transmission

During an SCI transmission, the transmit shift register shifts a frame out to the TXD pin. The SCI data registers (SCDRH and SCDRL) are the write-only buffers between the internal data bus and the transmit shift register.

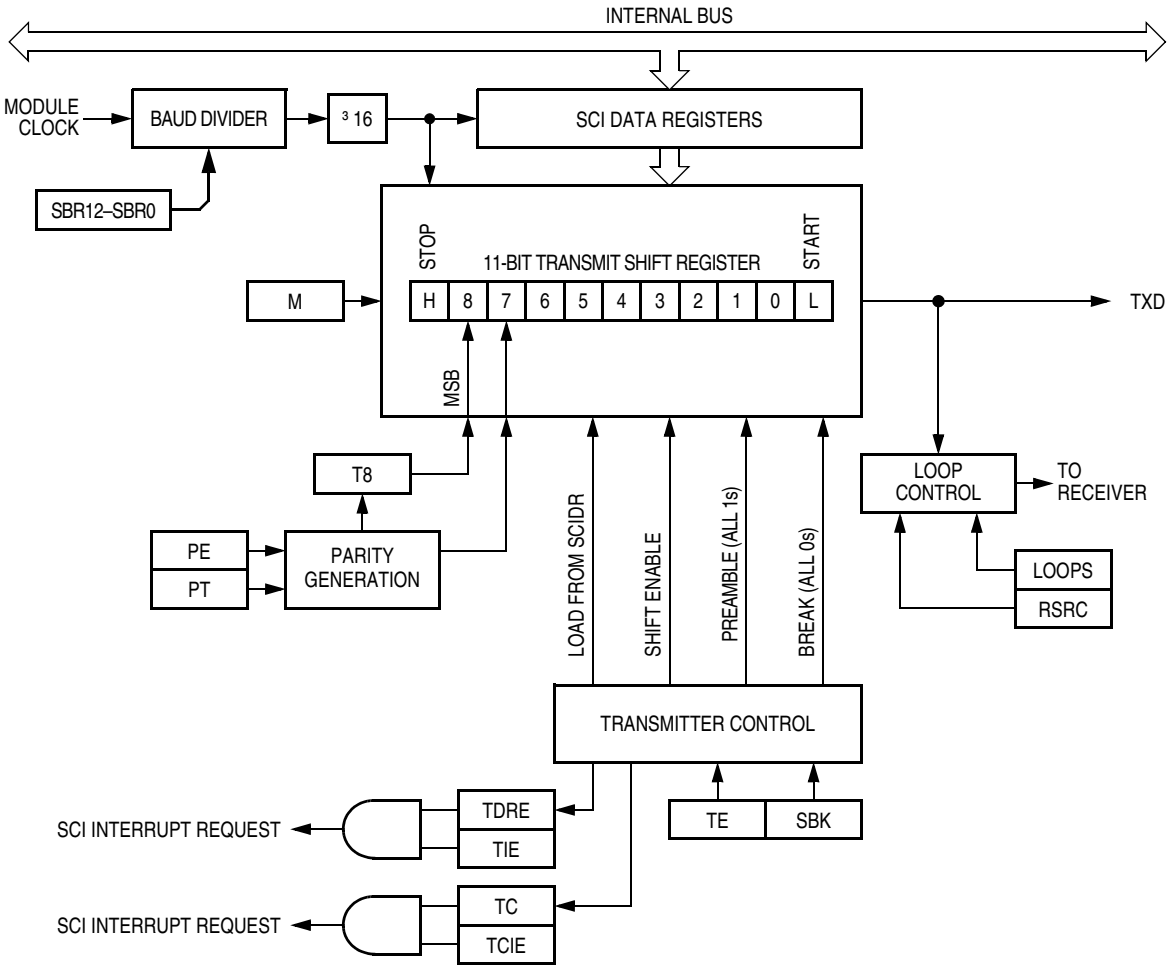


Figure 14-4. SCI Transmitter Block Diagram

To initiate an SCI transmission:

1. Enable the transmitter by writing a logic 1 to the transmitter enable bit, TE, in SCI control register 2 (SCCR2).
2. Clear the transmit data register empty flag, TDRE, by first reading SCI status register 1 (SCSR1) and then writing to SCI data register low (SCDRL). In 9-data-bit format, write the ninth bit to the T8 bit in SCI data register high (SCDRH).
3. Repeat step 2 for each subsequent transmission.

Writing the TE bit from 0 to 1 automatically loads the transmit shift register with a preamble of 10 logic 1s (if M = 0) or 11 logic 1s (if M = 1). After the preamble shifts out, control logic transfers the data from the SCI data register into the transmit shift register. A logic 0 start bit automatically goes into the least significant bit position of the transmit shift register. A logic 1 stop bit goes into the most significant bit position.

Hardware supports odd or even parity. When parity is enabled, the most significant bit (MSB) of the data character is the parity bit.

The transmit data register empty flag, TDRE, in SCI status register 1 (SCSR1) becomes set when the SCI data register transfers a byte to the transmit shift register. The TDRE flag indicates that the SCI data

register can accept new data from the internal data bus. If the transmit interrupt enable bit, TIE, in SCI control register 2 (SCCR2) is also set, the TDRE flag generates an SCI interrupt request.

When the transmit shift register is not transmitting a frame, the TXD pin goes to the idle condition, logic 1. If at any time software clears the TE bit in SCI control register 2 (SCCR2), the transmitter and receiver relinquish control of the port I/O pins.

If software clears TE while a transmission is in progress (TC = 0), the frame in the transmit shift register continues to shift out. Then the TXD pin reverts to being a general-purpose I/O pin even if there is data pending in the SCI data register. To avoid accidentally cutting off the last frame in a message, always wait for TDRE to go high after the last frame before clearing TE.

To separate messages with preambles with minimum idle line time, use this sequence between messages:

1. Write the last byte of the first message to SCDRH/L.
2. Wait for the TDRE flag to go high, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting the TE bit.
4. Write the first byte of the second message to SCDRH/L.

When the SCI relinquishes the TXD pin, the PORTS and DDRS registers control the TXD pin.

To force TXD high when turning off the transmitter, set bit 1 of the port S register (PORTS) and bit 1 of the port S data direction register (DDRS). The TXD pin goes high as soon as the SCI relinquishes it.

14.5.3.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCI control register 2 (SCCR2) loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCI control register 1 (SCCR1). As long as SBK is at logic 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next frame.

The SCI recognizes a break character when a start bit is followed by eight or nine logic 0 data bits and a logic 0 where the stop bit should be. Receiving a break character has these effects on SCI registers:

- Sets the framing error flag, FE
- Sets the receive data register full flag, RDRF
- Clears the SCI data registers, SCDRH/L
- May set the overrun flag, OR, noise flag, NF, parity error flag, PE, or the receiver active flag, RAF (see 14.6.4 SCI Status Register 1)

14.5.3.4 Idle Characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in SCI control register 1 (SCCR1). The preamble is a synchronizing idle character that begins the first transmission initiated after writing the TE bit from 0 to 1.

If the TE bit is cleared during a transmission, the TXD pin becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues an idle character to be sent after the frame currently being transmitted.

NOTE

When queueing an idle character, return the TE bit to logic 1 before the stop bit of the current frame shifts out to the TXD pin. Setting TE after the stop bit appears on TXD causes data previously written to the SCI data register to be lost.

Toggle the TE bit for a queued idle character when the TDRE flag becomes set and immediately before writing the next byte to the SCI data register.

14.5.4 Receiver

A block diagram of the SCI receiver is shown in Figure 14-5.

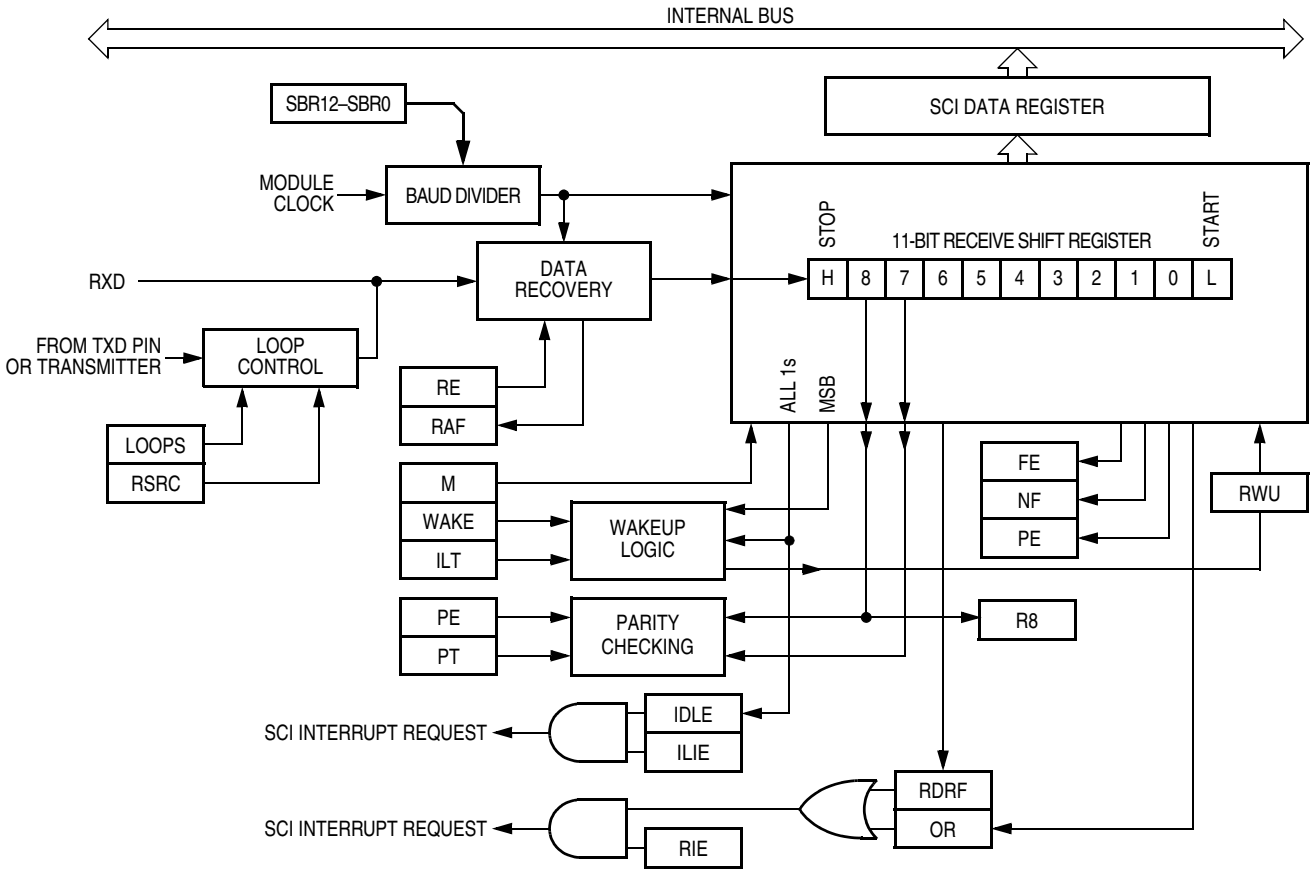


Figure 14-5. SCI Receiver Block Diagram

14.5.4.1 Character Length

The SCI receiver can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCCR1) determines the length of data characters. When receiving 9-bit data, bit R8 in SCI data register high (SCDRH) is the ninth bit (bit 8).

14.5.4.2 Character Reception

During an SCI reception, the receive shift register shifts a frame in from the RXD pin. The SCI data register is the read-only buffer between the internal data bus and the receive shift register.

After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the SCI data register. The receive data register full flag, RDRF, in SCI status register 1 (SCSR1) becomes set, indicating that the received byte can be read. If the receive interrupt enable bit, RIE, in SCI control register 2 (SCCR2) is also set, the RDRF flag generates an interrupt request.

14.5.4.3 Data Sampling

The receiver samples the RXD pin at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock (see Figure 14-6) is resynchronized:

- After every start bit
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0)

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

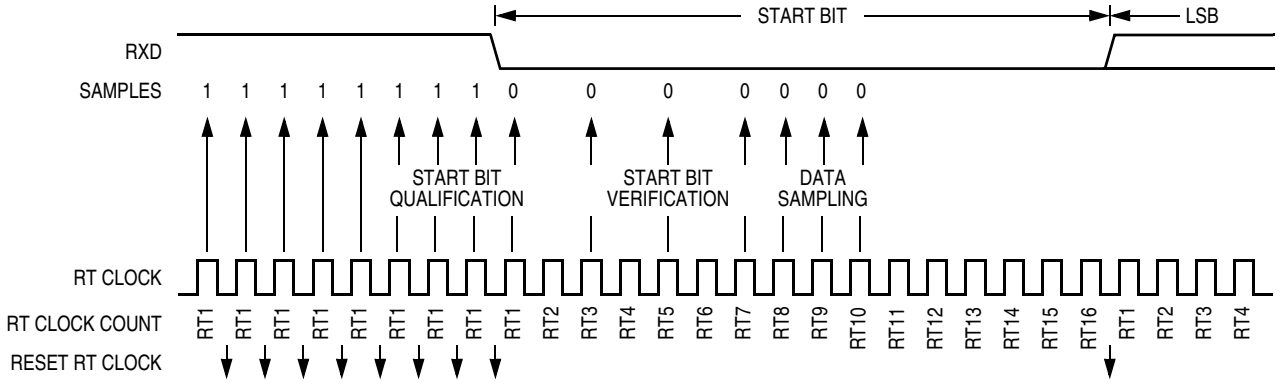


Figure 14-6. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Table 14-4 summarizes the results of the start bit verification samples.

Table 14-4. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 14-5 summarizes the results of the data bit samples.

Table 14-5. Data Bit Recovery

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

NOTE

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 14-6 summarizes the results of the stop bit samples.

Table 14-6. Stop Bit Recovery

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In Figure 14-7 the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

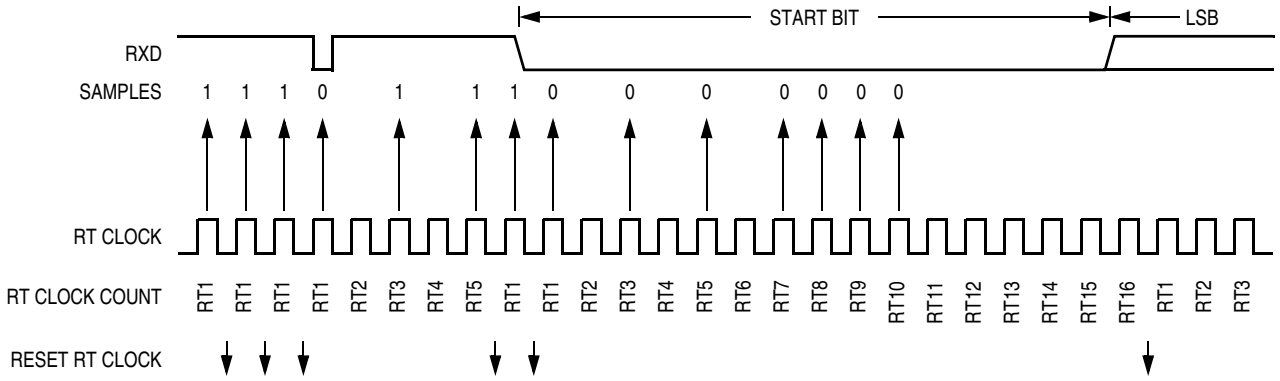


Figure 14-7. Start Bit Search Example 1

In Figure 14-8 noise is perceived as the beginning of a start bit although the verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

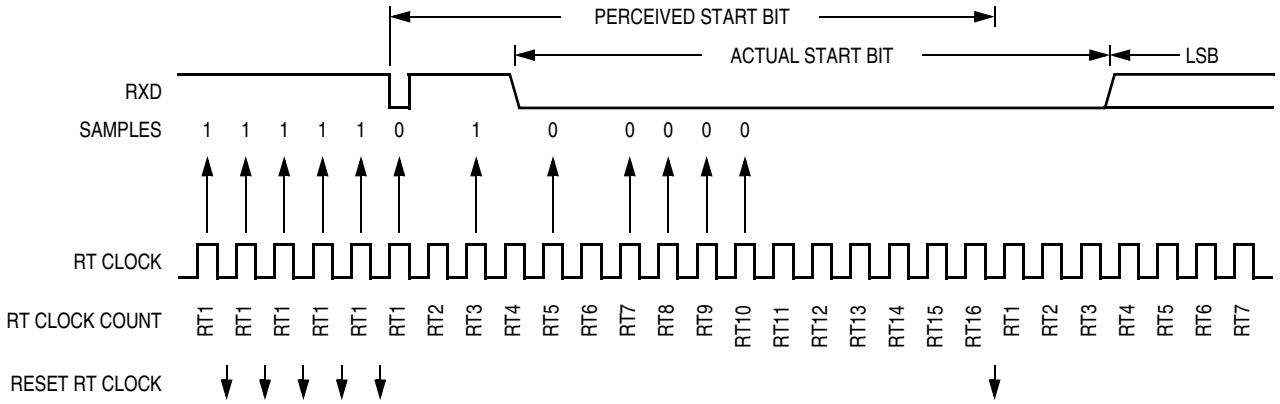


Figure 14-8. Start Bit Search Example 2

In Figure 14-9 a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

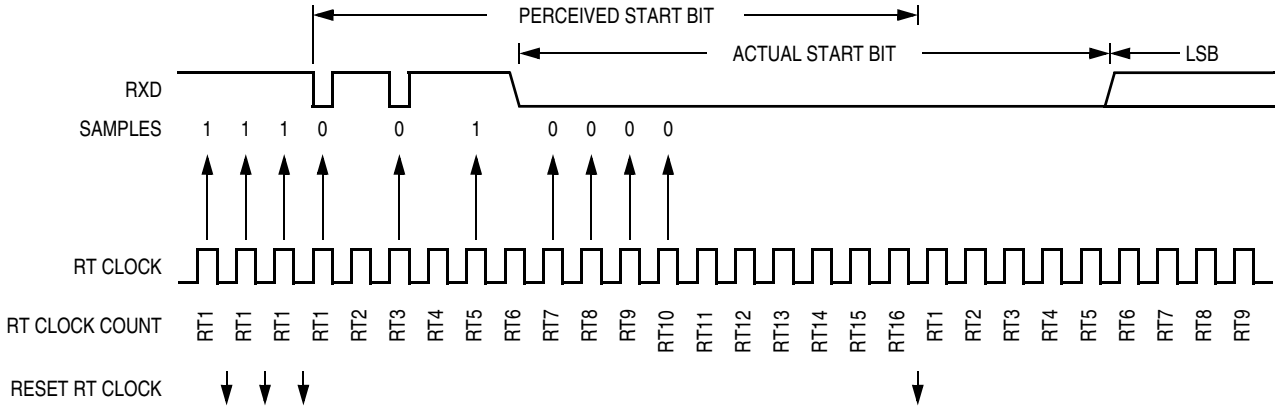


Figure 14-9. Start Bit Search Example 3

Figure 14-10 shows the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

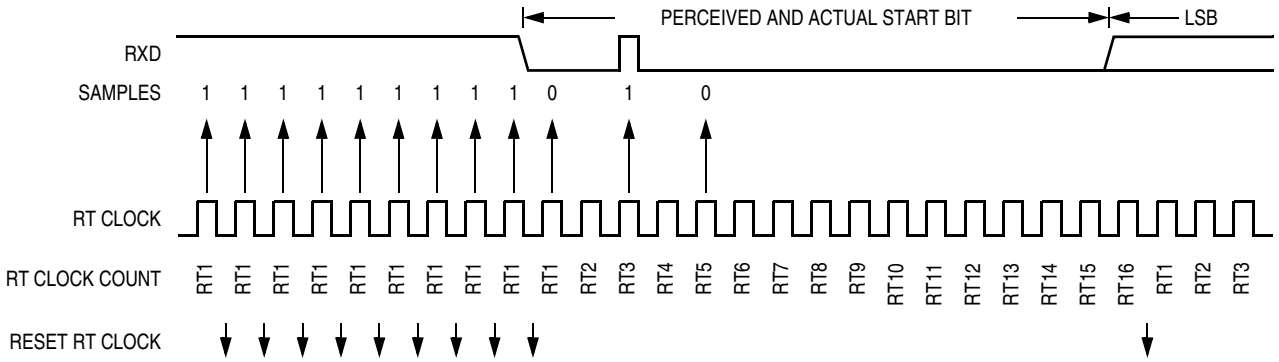


Figure 14-10. Start Bit Search Example 4

Figure 14-11 shows a burst of noise near the beginning of the start bit that resets the RT clock. The sample after the reset is low but is not preceded by three high samples that would qualify as a falling edge. Depending on the timing of the start bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

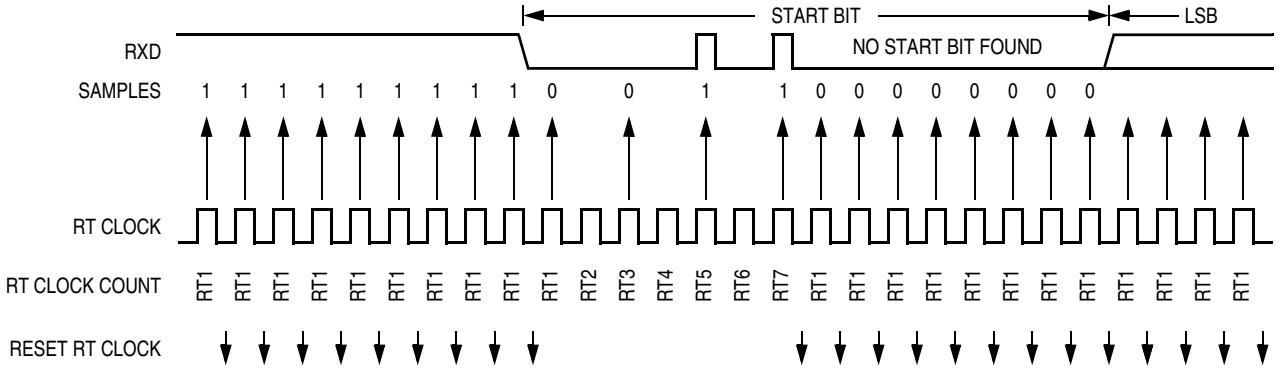


Figure 14-11. Start Bit Search Example 5

In Figure 14-12 a noise burst makes the majority of data samples RT8, RT9, and RT10 high. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored.

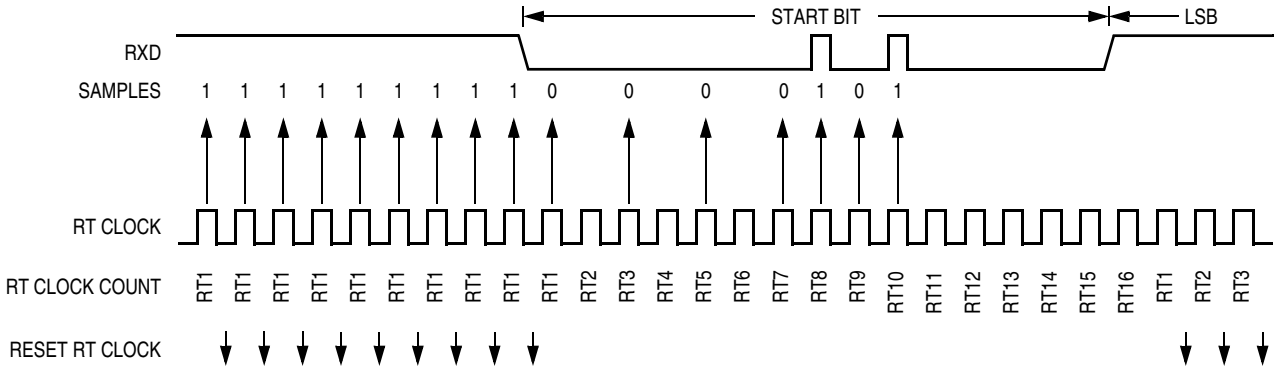


Figure 14-12. Start Bit Search Example 6

14.5.4.4 Framing Errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, FE, in SCI status register 1 (SCSR1). A break character also sets the FE flag because a break character has no stop bit. The FE flag is set at the same time that the RDRF flag is set.

14.5.4.5 Baud Rate Tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples to fall outside the actual stop bit. Then a noise error occurs. If more than one of the samples is outside the stop bit, a framing error occurs. In most applications, the baud rate tolerance is much more than the degree of misalignment that is likely to occur.

As the receiver samples an incoming frame, it resynchronizes the RT clock on any valid falling edge within the frame. Resynchronization within frames corrects misalignments between transmitter bit times and receiver bit times.

Slow Data Tolerance

Figure 14-13 shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

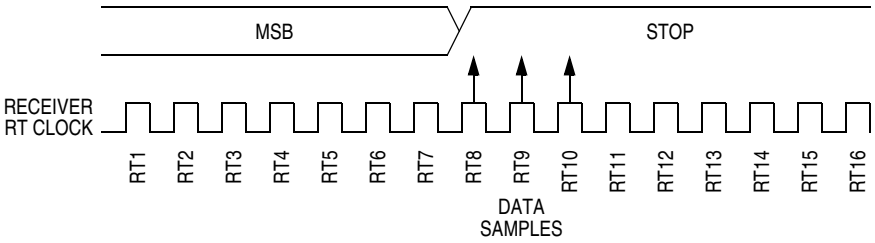


Figure 14-13. Slow Data

For an 8-bit data character, data sampling of the stop bit takes the receiver
9 bit times × 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in Figure 14-13, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 9 bit times × 16 RT cycles + 3 RT cycles = 147 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$\left|\frac{154 - 147}{154}\right| \times 100 = 4.54\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver
10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 14-13, the receiver counts 170 RT cycles at the point when the count of the transmitting device is: 10 bit times × 16 RT cycles + 3 RT cycles = 163 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$\left|\frac{170 - 163}{170}\right| \times 100 = 4.12\%$$

Fast Data Tolerance

Figure 14-14 shows how much a fast received frame can be misaligned without causing a noise error or a framing error. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

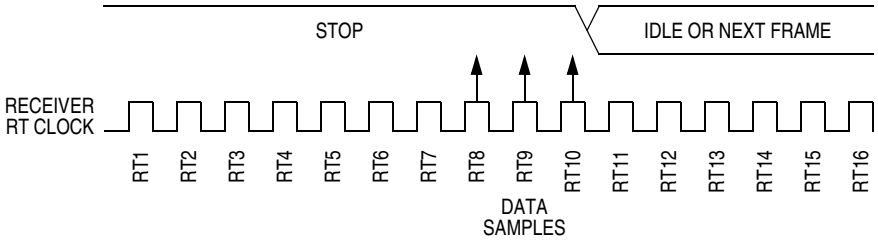


Figure 14-14. Fast Data

For an 8-bit data character, data sampling of the stop bit takes the receiver
9 bit times × 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in Figure 14-14, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 10 bit times × 16 RT cycles = 160 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$$\left|\frac{154 - 160}{154}\right| \times 100 = 3.90\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver
10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 14-14, the receiver counts 170 RT cycles at the point when the count of the transmitting device is: 11 bit times × 16 RT cycles = 176 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$$\left|\frac{170 - 176}{170}\right| \times 100 = 3.53\%$$

14.5.4.6 Receiver Wakeup

So that the SCI can ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in SCI control register 2 (SCCR2) puts the receiver into a standby state during which receiver interrupts are disabled.

The transmitting device can address messages to selected receivers by including addressing information in the initial frame or frames of each message.

The WAKE bit in SCI control register 1 (SCCR1) determines how the SCI is brought out of the standby state to process an incoming message. The WAKE bit enables either idle line wakeup or address mark wakeup:

- Idle input line wakeup (WAKE = 0) — In this wakeup method, an idle condition on the RXD pin clears the RWU bit and wakes up the SCI. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which

the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another idle character appears on the RXD pin.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

The idle character that wakes a receiver does not set the receiver idle flag, IDLE, or the receive data register full flag, RDRF.

The idle line type bit, ILT, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit. ILT is in SCI control register 1 (SCCR1).

- Address mark wakeup (WAKE = 1) — In this wakeup method, a logic 1 in the most significant bit (MSB) position of a frame clears the RWU bit and wakes up the SCI. The logic 1 in the MSB position marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another address frame appears on the RXD pin.

  The logic 1 MSB of an address frame clears the receiver's RWU bit before the stop bit is received and sets the RDRF flag.

  Address mark wakeup allows messages to contain idle characters but requires that the MSB be reserved for use in address frames.

NOTE

With the WAKE bit clear, setting the RWU bit after the RXD pin has been idle can cause the receiver to wake up immediately.

14.5.5 Single-Wire Operation

Normally, the SCI uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the SCI and is available as a general-purpose I/O pin. The SCI uses the TXD pin for both receiving and transmitting.

Setting the data direction bit for the TXD pin configures TXD as the output for transmitted data. Clearing the data direction bit configures TXD as the input for received data.

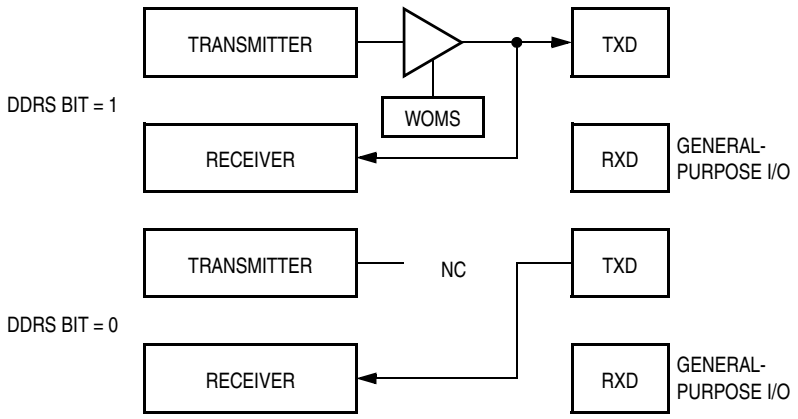


Figure 14-15. Single-Wire Operation (LOOPS = 1 and RSRC = 1)

Enable single-wire operation by setting the LOOPS bit and the receiver source bit, RSRC, in SCI control register 1 (SCCR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Setting the RSRC bit connects the receiver input to the output of the TXD pin driver. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).

The wired-OR mode select bit, WOMS, configures the TXD pin for full CMOS drive or for open-drain drive. WOMS controls the TXD pin in both normal operation and in single-wire operation. When WOMS is set, the data direction bit for the TXD pin does not have to be cleared for TXD to receive data.

14.5.6 Loop Operation

In loop operation, the transmitter output goes to the receiver input. The RXD pin is disconnected from the SCI and is available as a general-purpose I/O pin.

Setting the data direction bit for the TXD pin connects the transmitter output to the TXD pin. Clearing the data direction bit disconnects the transmitter output from the TXD pin.

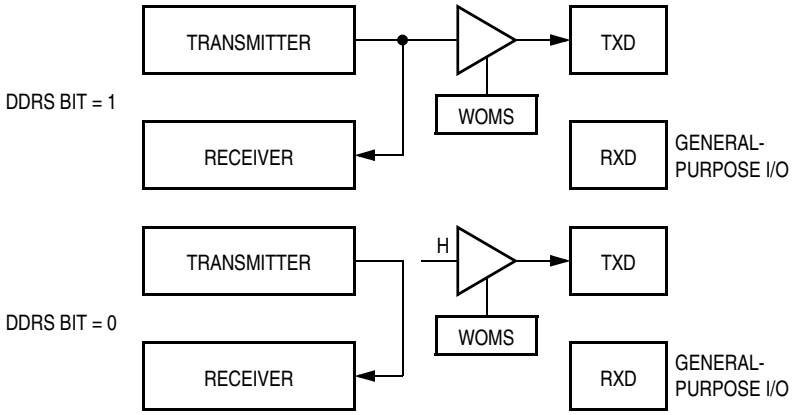


Figure 14-16. Loop Operation (LOOP = 1 and RSRC = 0)

Enable loop operation by setting the LOOPS bit and clearing the RSRC bit in SCI control register 1 (SCCR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Clearing the RSRC bit connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).

The wired-OR mode select bit, WOMS, configures the TXD pin for full CMOS drive or for open-drain drive. WOMS controls the TXD pin in both normal operation and in loop operation.

14.6 Register Descriptions and Reset Initialization

This section provides register descriptions and reset initialization.

14.6.1 SCI Baud Rate Registers

SCI0: $00C0
SCI1: $00C8

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8
Reset:	0	0	0	0	0	0	0	0

Figure 14-17. SCI Baud Rate Register High (SC0BDH or SC1BDH)

SCI0: $00C1
SCI1: $00C9

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
Reset:	0	0	0	0	0	1	0	0

Figure 14-18. SCI Baud Rate Register Low (SC0BDL or SC1BDL)

Read: Anytime
Write: SBR[12:0] anytime; BTST, BSPL, and BRLD only in special modes

BTST — Reserved for test function

BSPL — Reserved for test function

BRLD — Reserved for test function

SBR[12:0] — SCI Baud Rate Bits

The value written to SBR[12:0] determines the baud rate of the SCI. The new value takes effect when the low order byte is written. The formula for calculating baud rate is:

$$\text{SCI baud rate} = \frac{\text{MCLK}}{16 \times \text{BR}}$$

BR = value written to SBR[12:0], a value from 1 to 8191

NOTE

The baud rate generator is disabled until the TE bit or the RE bit is set for the first time after reset. The baud rate generator is disabled when BR = 0.

14.6.2 SCI Control Register 1

SCI0: $00C2
SCI1: $00CA

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT
Reset:	0	0	0	0	0	0	0	0

Figure 14-19. SCI Control Register 1 (SC0CR1 or SC1CR1)

Read: Anytime
Write: Anytime

LOOPS — Loop Select Bit

LOOPS enables loop operation. In loop operation the RXD pin is disconnected from the SCI, and the transmitter output goes into the receiver input. Both the transmitter and the receiver must be enabled to use the loop function.

1 = Loop operation enabled
0 = Normal operation enabled

The receiver input is determined by the RSRC bit. The transmitter output is controlled by the associated DDRS bit.

If the data direction bit for the TXD pin is set and LOOPS = 1, the transmitter output appears on the TXD pin. If the DDRS bit is clear and LOOPS = 1, the TXD pin is idle (high) if RSRC = 0 and high-impedance if RSRC = 1. See Table 14-7.

WOMS — Wired-OR Mode Select Bit

WOMS configures the TXD and RXD pins for open-drain operation. WOMS allows TXD pins to be tied together in a multiple-transmitter system. Then the TXD pins of non-active transmitters follow the logic level of an active 1. WOMS also affects the TXD and RXD pins when they are general-purpose outputs. External pullup resistors are necessary on open-drain outputs.

1 = TXD and RXD pins, open-drain when outputs
0 = TXD and RXD pins, full CMOS drive capability

RSRC — Receiver Source Bit

When LOOPS = 1, the RSRC bit determines the internal feedback path for the receiver.

1 = Receiver input connected to TXD pin
0 = Receiver input internally connected to transmitter output

Table 14-7. Loop Mode Functions

LOOPS	RSRC	DDRSx[1]	WOMS	Function of TXD Pin
0	X	X	X	Normal operation
1	0	0	X	Loop mode; transmitter output connected to receiver input TXD pin disconnected
1	0	1	0	Loop mode; transmitter output connected to receiver input TXD is CMOS output
1	0	1	1	Loop mode; transmitter output connected to receiver input TXD is open-drain output

Table 14-7. Loop Mode Functions (Continued)

LOOPS	RSRC	DDRSx[1]	WOMS	Function of TXD Pin
1	1	0	x	Single-wire mode; transmitter output disconnected TXD is high-impedance receiver input
1	1	1	0	Single-wire mode; TXD pin connected to receiver input
1	1	1	1	Single wire mode; TXD pin connected to receiver input TXD is open-drain for receiving and transmitting

1. DDRSx means the data direction bit of the TXD pin.

M — Mode Bit

M determines whether data characters are eight or nine bits long.

1 = One start bit, nine data bits, one stop bit
0 = One start bit, eight data bits, one stop bit

WAKE — Wakeup Bit

WAKE determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit position of a received data character or an idle condition on the RXD pin.

1 = Address mark wakeup
0 = Idle line wakeup

ILT — Idle Line Type Bit

ILT determines when the receiver starts counting logic 1s as idle character bits. The counting begins either after the start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit may cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions.

1 = Idle character bit count begins after stop bit.
0 = Idle character bit count begins after start bit.

PE — Parity Enable Bit

PE enables the parity function. When enabled, the parity function inserts a parity bit in the most significant bit position.

1 = Parity function enabled
0 = Parity function disabled

PT — Parity Type Bit

PT determines whether the SCI generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit.

1 = Odd parity
0 = Even parity

14.6.3 SCI Control Register 2

SCI0: $00C3

SCI1: $00CB

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset:	0	0	0	0	0	0	0	0

Figure 14-20. SCI Control Register 2 (SC0CR2 or SC1CR2)

Read: Anytime
Write: Anytime

TIE — Transmitter Interrupt Enable Bit

TIE enables the transmit data register empty flag, TDRE, to generate interrupt requests.

1 = TDRE interrupt requests enabled
0 = TDRE interrupt requests disabled

TCIE — Transmission Complete Interrupt Enable Bit

TCIE enables the transmission complete flag, TC, to generate interrupt requests.

1 = TC interrupt requests enabled
0 = TC interrupt requests disabled

RIE — Receiver Interrupt Enable Bit

RIE enables the receive data register full flag, RDRF, and the overrun flag, OR, to generate interrupt requests.

1 = RDRF and OR interrupt requests enabled
0 = RDRF and OR interrupt requests disabled

ILIE — Idle Line Interrupt Enable Bit

ILIE enables the idle line flag, IDLE, to generate interrupt requests.

1 = IDLE interrupt requests enabled
0 = IDLE interrupt requests disabled

TE — Transmitter Enable Bit

TE enables the SCI transmitter and configures the TXD pin as the SCI transmitter output. The TE bit can be used to queue an idle preamble.

1 = Transmitter enabled
0 = Transmitter disabled

RE — Receiver Enable Bit

RE enables the SCI receiver.

1 = Receiver enabled
0 = Receiver disabled

RWU — Receiver Wakeup Bit

1 = Standby state
0 = Normal operation

RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.

SBK — Send Break Bit

Toggling SBK sends one break character (10 or 11 logic 0s). As long as SBK is set, the transmitter sends logic 0s.

1 = Transmit break characters
0 = No break characters

14.6.4 SCI Status Register 1

SCI0: $00C4
SCI1: $00CC

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write:								
Reset:	1	1	0	0	0	0	0	0

= Unimplemented

Figure 14-21. SCI Status Register 1 (SC0SR1 or SC1SR1)

Read: Anytime
Write: Has no meaning or effect

TDRE — Transmit Data Register Empty Flag

TDRE is set when the transmit shift register receives a byte from the SCI data register. Clear TDRE by reading SCI status register 1 with TDRE set and then writing to the low byte of the SCI data register.

1 = Transmit data register empty
0 = Transmit date register not empty

TC — Transmission Complete Flag

TC is set when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 with TC set and then writing to the low byte of the SCI data register. TC clears automatically when a break, preamble, or data is queued and ready to be sent.

1 = Transmission complete
0 = Transmission in progress

RDRF — Receive Data Register Full Flag

RDRF is set when the data in the receive shift register transfers to the SCI data register. Clear RDRF by reading SCI status register 1 with RDRF set and then reading the low byte of the SCI data register.

1 = Receive data register full
0 = Data not available in SCI data register

IDLE — Idle Line Flag

IDLE is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M = 1) appear on the receiver input. Clear IDLE by reading SCI status register 1 with IDLE set and then writing to the low byte of the SCI data register. Once IDLE is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag.

1 = Receiver input has become idle
0 = Receiver input is either active now or has never become active since the IDLE flag was last cleared

NOTE

When the receiver wakeup bit (RWU) is set, an idle line condition does not set the IDLE flag.

OR — Overrun Flag

OR is set when software fails to read the SCI data register before the receive shift register receives the next frame. The data in the shift register is lost, but the data already in the SCI data registers is not affected. Clear OR by reading SCI status register 1 with OR set and then reading the low byte of the SCI data register.

1 = Overrun
0 = No overrun

NF — Noise Flag

NF is set when the SCI detects noise on the receiver input. NF is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear NF by reading SCI status register 1 and then reading the low byte of the SCI data register.

1 = Noise
0 = No noise

FE — Framing Error Flag

FE is set when a logic 0 is accepted as the stop bit. FE is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. FE inhibits further data reception until it is cleared. Clear FE by reading SCI status register 1 with FE set and then reading the low byte of the SCI data register.

1 = Framing error
0 = No framing error

PF — Parity Error Flag

PF is set when the parity enable bit, PE, is set and the parity of the received data does not match its parity bit. Clear PF by reading SCI status register 1 and then reading the low byte of the SCI data register.

1 = Parity error
0 = No parity error

14.6.5 SCI Status Register 2

SCI0: $00C5

SCI1: $00CD

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	RAF
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 14-22. SCI Status Register 2 (SC0SR2 or SC1SR2)

Read: Anytime
Write: Has no meaning or effect

RAF — Receiver Active Flag

RAF is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects false start bits (usually from noise or baud rate mismatch) or when the receiver detects an idle character.

1 = Reception in progress
0 = No reception in progress

14.6.6 SCI Data Registers

SCI0: $00C6
SCI1: $00CE

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R8	T8	0	0	0	0	0	0
Write:								
Reset:	Unaffected by reset							

= Unimplemented

Figure 14-23. SCI Data Register High (SC0DRH or SC1DRH)

SCI0: $00C7
SCI1: $00CF

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R7	R6	R5	R4	R3	R2	R1	R0
Write:	T7	T6	T5	T4	T3	T2	T1	T0
Reset:	Unaffected by reset							

Figure 14-24. SCI Data Register Low (SC0DRL or SC1DRL)

Read: Anytime; reading accesses receive data register
Write: Anytime; writing accesses transmit data register; writing to R8 has no effect

R8 — Received Bit 8

R8 is the ninth data bit received when the SCI is configured for 9-bit data format (M = 1).

T8 — Transmitted Bit 8

T8 is the ninth data bit transmitted when the SCI is configured for 9-bit data format (M = 1).

R7–R0 — Received Bits 7–0

T7–T0 — Transmitted Bits 7–0

NOTE

If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten. The same value is transmitted until T8 is rewritten.

In 8-bit data format, only SCI data register low (SCDRL) needs to be accessed.

When transmitting in 9-bit data format and using 8-bit write instructions, write first to SCI data register high (SCDRH).

14.7 External Pin Descriptions

This section provides a description of TXD and RXD, the SCI's two external pins.

14.7.1 TXD Pin

TXD is the SCI transmitter pin. TXD is available for general-purpose I/O when it is not configured for transmitter operation.

14.7.2 RXD Pin

RXD is the SCI receiver pin. RXD is available for general-purpose I/O when it is not configured for receiver operation.

14.8 Modes of Operation

The SCI functions the same in normal, special, and emulation modes.

14.9 Low-Power Options

This section provides a description of the three low-power modes:

- Run mode
- Wait mode
- Stop mode

14.9.1 Run Mode

Clearing the transmitter enable or receiver enable bits (TE or RE) in SCI control register 2 (SCCR2) reduces power consumption in run mode. SCI registers are still accessible when TE or RE is cleared, but clocks to the core of the SCI are disabled.

14.9.2 Wait Mode

The SCI remains active in wait mode. Any enabled interrupt request from the SCI can bring the MCU out of wait mode.

If SCI functions are not required during wait mode, reduce power consumption by disabling the SCI before executing the WAIT instruction.

14.9.3 Stop Mode

For reduced power consumption, the SCI is inactive in stop mode. The STOP instruction does not affect SCI register states. SCI operation resumes after an external interrupt.

Exiting stop mode by reset aborts any transmission or reception in progress and resets the SCI.

14.10 Interrupt Sources

Table 14-8. SCI Interrupt Sources

Interrupt Source	Flag	Local Enable	CCR Mask	Vector Address	
				SCI0	SCI1
Transmit data register empty	TDRE	TIE	I bit	$FFD6, $FFD7	$FFD4, $FFD5
Transmission complete	TC	TCIE	I bit		
Receive data register full	RDRF	RIE	I bit		
Receiver overrun	OR				
Receiver idle	IDLE	ILIE	I bit		

14.11 General-Purpose I/O Ports

Port S shares its pins with the multiple serial interface (MSI). In all modes, port S pins PS7–PS0 are available for either general-purpose I/O or for SCI and SPI functions. See Chapter 13 Multiple Serial Interface (MSI).

14.12 Serial Character Transmission Using the SCI

Code is intended to use SCI1 to serially transmit characters using polling to the LCD display on the UDLP1 board: when the transmission data register is empty a flag will get set, which is telling us that SC1DR is ready so we can write another byte. The transmission is performed at a baud rate of 9600. Since the SCI1 is only being used for transmit data, the data register will not be used bidirectionally for received data.

14.12.1 Equipment

For this exercise, use the M68HC812A4EVB emulation board.

14.12.2 Code Listing

NOTE

A comment line is delimited by a semicolon. If there is no code before comment, a semicolon (;) must be placed in the first column to avoid assembly errors.

```
INCLUDE 'EQUATES.ASM'      ; Equates for registers

; User Variables

; Bit Equates
; ---------------------------------------------------------------------
;           MAIN PROGRAM
; ---------------------------------------------------------------------
        ORG    $7000            ; 16K On-Board RAM, User code data area,
;                               ; start main program at $4000
MAIN:
        BSR    INIT             ; Subroutine to Initialize SCI0 registers
        BSR    TRANS            ; Subroutine to start transmission
DONE:   BRA    DONE             ; Always branch to DONE, convenient for breakpoint

; ---------------------------------------------------------------------
;         SUBROUTINE INIT:
; ---------------------------------------------------------------------
INIT:   TPA                     ; Transfer CCR to A accumulator
        ORAA   #$10             ; ORed A with #$10 to Set I bit
        TAP                     ; Transfer A to CCR

        MOVB   #$34,SC1BDL      ; Set BAUD =9600, in SCI1 Baud Rate Reg.

        MOVB   #$00,SC1CR1      ; Initialize for 8-bit Data format,
;                               ; Loop Mode  and parity disabled,(SC1CR1)

        MOVB   #$08,SC1CR2      ; Set for No Ints, and Transmitter enabled(SC1CR2)

        LDAA   SC1SR1           ; 1st step to clear TDRE flag: Read SC1SR1
        STD    SC1DRH           ; 2nd step to clear TDRE flag: Write SC1DR register

        LDX    #DATA            ; Use X as a pointer to DATA.

        RTS                     ; Return from subroutine

; ---------------------------------------------------------------------
;       TRANSMIT SUBROUTINE
; ---------------------------------------------------------------------
TRANS:  BRCLR  SC1SR1,#$80, TRANS ; Wait for TDRE flag
        MOVB   1,X+,SC1DRL      ; Transmit character, increment X pointer
        CPX    #EOT             ; Detect if last character has been transmitted
        BNE    TRANS            ; If last char. not equal to "eot", Branch to TRANS
        RTS                     ; else Transmission complete, Return from Subroutine

; ---------------------------------------------------------------------
;  TABLE : DATA TO BE TRANSMITTED
; ---------------------------------------------------------------------
DATA:   DC.B   'Freescale HC12 Banner - June, 1999'
        DC.B   $0D,$0A          ; Return (cr) ,Line Feed (LF)
        DC.B   'Scottsdale, Arizona'
        DC.B   $0D,$0A          ; Return (cr) ,Line Feed (LF)
EOT:    DC.B   $04              ; Byte used to test end of data = EOT

        END                     ; End of program
```

Chapter 15
Serial Peripheral Interface (SPI)

15.1 Introduction

The serial peripheral interface (SPI) allows full-duplex, synchronous, serial communications with peripheral devices.

15.2 Features

Features of the SPI include:

- Full-duplex operation
- Master mode and slave mode
- Programmable slave-select output option
- Programmable bidirectional data pin option
- Two flags with interrupt-generation capability:
  - Transmission complete
  - Mode fault
- Write collision detection
- Read data buffer
- Serial clock with programmable polarity and phase
- Reduced drive control for lower power consumption
- Programmable open-drain output option

15.3 Block Diagram

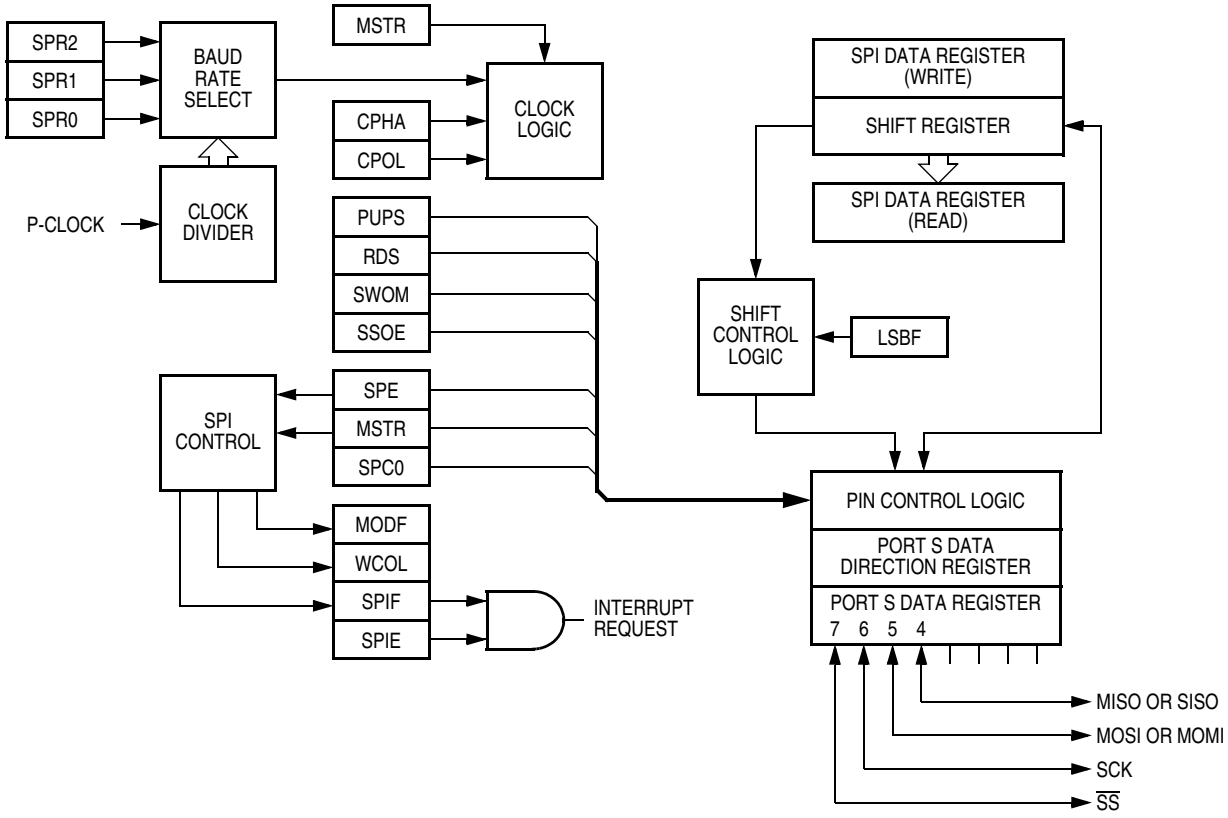


Figure 15-1. SPI Block Diagram

15.4 Register Map

NOTE

The register block can be mapped to any 2-Kbyte boundary within the standard 64-Kbyte address space. The register block occupies the first 512 bytes of the 2-Kbyte block. This register map shows default addressing after reset.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$00D0	SPI 0 Control Register 1 (SP0CR1) See page 186.	Read:	SPIE	SPE	SWOM	MSTR	CPOL	CPHA	SSOE	LSBF
		Write:								
		Reset:	0	0	0	0	0	1	0	0
$00D1	SPI 0 Control Register 2 (SP0CR2) See page 187.	Read:	0	0	0	0	PUPS	RDS	0	SPC0
		Write:								
		Reset:	0	0	0	0	1	0	0	0
$00D2	SPI Baud Rate Register (SP0BR) See page 188.	Read:	0	0	0	0	0	SPR2	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00D3	SPI Status Register (SP0SR) See page 189.	Read:	SPIF	WCOL	0	MODF	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00D5	SPI Data Register (SP0DR) See page 190.	Read:	Bit 7	6	5	4	3	2	1	0
		Write:								
		Reset:	Unaffected by reset							
$00D6	Port S Data Register (PORTS) See page 147.	Read:	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
		Write:								
		Reset:	Unaffected by reset							
$00D7	Port S Data Direction Register (DDRS) See page 148.	Read:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 15-2. SPI Register Map

15.5 Functional Description

The SPI allows full-duplex, synchronous, serial communication between the MCU and peripheral devices, including other MCUs. In master mode, the SPI generates the synchronizing clock and initiates transmissions. In slave mode, the SPI depends on a master peripheral to start and synchronize transmissions.

15.5.1 Master Mode

The SPI operates in master mode when the master mode bit, MSTR, is set.

NOTE

Configure SPI modules as master or slave before enabling them. Enable the master SPI before enabling the slave SPI. Disable the slave SPI before disabling the master SPI.

Only a master SPI module can initiate transmissions. Begin the transmission from a master SPI module by writing to the SPI data register. If the shift register is empty, the byte immediately transfers to the shift register. The byte begins shifting out on the master out, slave in pin (MOSI) under the control of the serial clock. See Figure 15-3.

As the byte shifts out on the MOSI pin, a byte shifts in from the slave on the master in, slave out pin (MISO) pin. On the eighth serial clock cycle, the transmission ends and sets the SPI flag, SPIF. At the same time that SPIF becomes set, the byte from the slave transfers from the shift register to the SPI data register. The byte remains in a read buffer until replaced by the next byte from the slave.

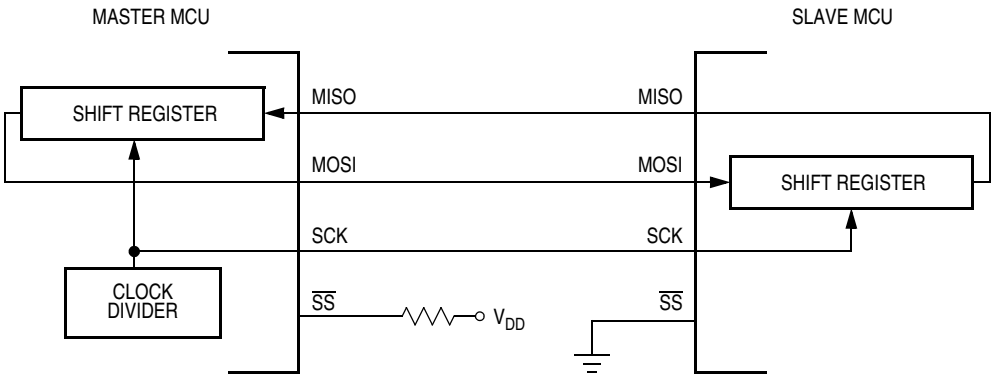


Figure 15-3. Full-Duplex Master/Slave Connections

15.5.2 Slave Mode

The SPI operates in slave mode when MSTR is clear. In slave mode, the SCK pin is the input for the serial clock from the master.

NOTE

Before a transmission occurs, the $\overline{SS}$ pin of the slave SPI must be at logic 0. The slave $\overline{SS}$ pin must remain low until the transmission is complete.

A transmission begins when initiated by a master SPI. The byte from the master SPI begins shifting in on the slave MOSI pin under the control of the master serial clock.

As the byte shifts in on the MOSI pin, a byte shifts out on the MISO pin to the master shift register. On the eighth serial clock cycle, the transmission ends and sets the SPI flag, SPIF. At the same time that SPIF

becomes set, the byte from the master transfers to the SPI data register. The byte remains in a read buffer until replaced by the next byte from the master.

15.5.3 Baud Rate Generation

A clock divider in the SPI produces eight divided P-clock signals. The P-clock divisors are 2, 4, 8, 16, 32, 64, 128, and 256. The SPR[2:1:0] bits select one of the divided P-clock signals to control the rate of the shift register. Through the SCK pin, the selected clock signal also controls the rate of the shift register of the slave SPI or other slave peripheral.

The clock divider is active only in master mode and only when a transmission is taking place. Otherwise, the divider is disabled to save power.

15.5.4 Clock Phase and Polarity

The clock phase and clock polarity bits, CPHA and CPOL, can select any of four combinations of serial clock phase and polarity. The CPHA bit determines whether a falling $\overline{SS}$ edge or the first SCK edge begins the transmission. The CPOL bit determines whether SCK is active-high or active-low.

NOTE

To transmit between SPI modules, both modules must have identical CPHA and CPOL values.

When CPHA = 0, a falling $\overline{SS}$ edge signals the slave to begin transmission. The capture strobe for the first bit occurs on the first serial clock edge. Therefore, the slave must begin driving its data before the first serial clock edge. After transmission of all eight bits, the slave $\overline{SS}$ pin must toggle from low to high to low again to begin another transmission. This format may be preferable in systems having more than one slave driving the master MISO line.

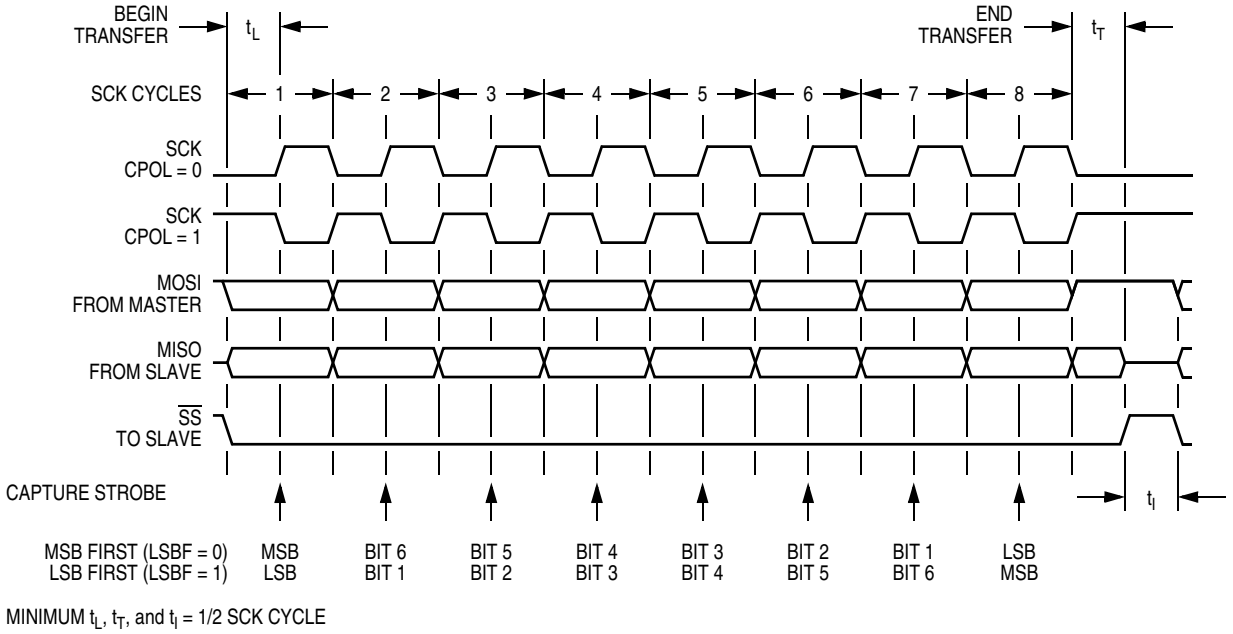


Figure 15-4. Transmission Format 0 (CPHA = 0)

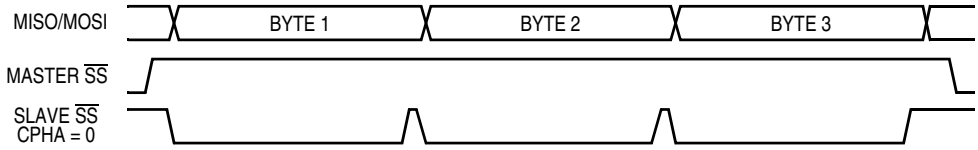


Figure 15-5. Slave SS Toggling When CPHA = 0

When CPHA = 1, the master begins driving its MOSI pin and the slave begins driving its MISO pin on the first serial clock edge. The SS pin can remain low between transmissions. This format may be preferable in systems having only one slave driving the master MISO line.

NOTE

The slave SCK pin must be in the proper idle state before the slave is enabled.

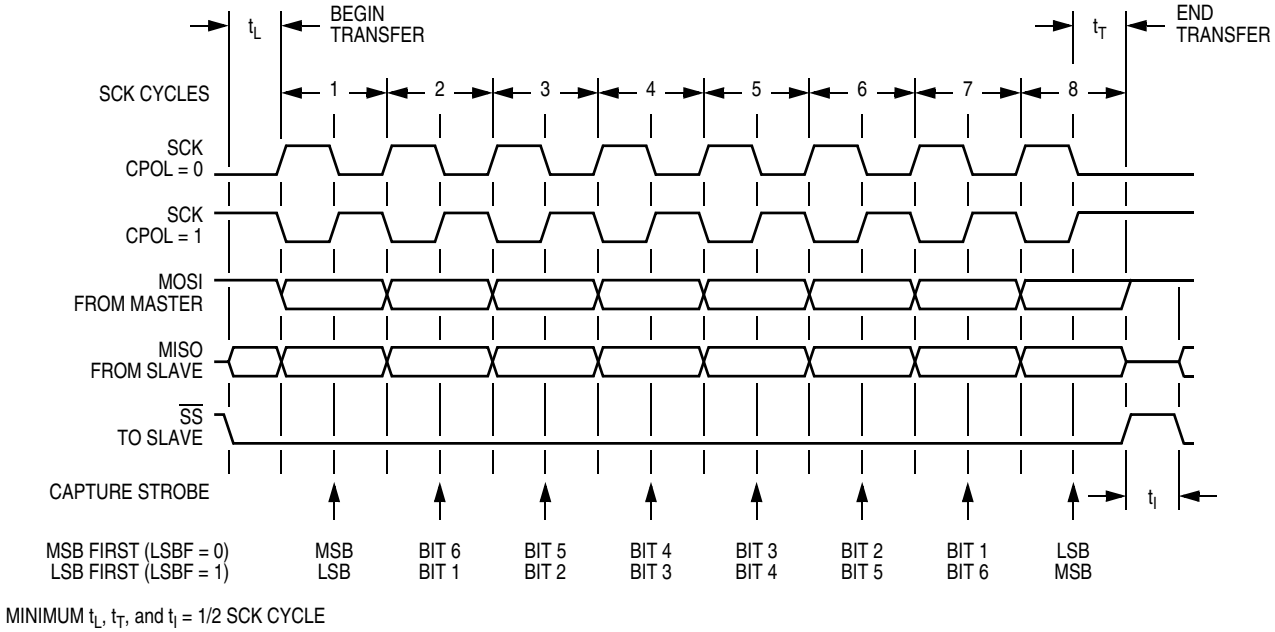


Figure 15-6. Transmission Format 1 (CPHA = 1)

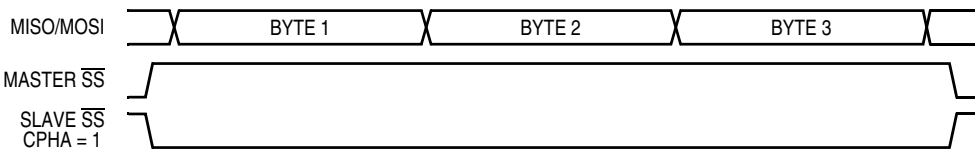


Figure 15-7. Slave SS When CPHA = 1

15.5.5 $\overline{SS}$ Output

In master mode only, the $\overline{SS}$ pin can function as a chip-select output for connection to the $\overline{SS}$ input of a slave. The master $\overline{SS}$ output automatically selects the slave by going low for each transmission and deselects the slave by going high during each idling state.

Enable the $\overline{SS}$ output by setting the master mode bit, MSTR, the slave-select output enable bit, SSOE, and the data direction bit of the $\overline{SS}$ pin. MSTR and SSOE are in SPI control register 1.

Table 15-1. $\overline{SS}$ Pin Configurations

Control Bits		$\overline{SS}$ Pin Function	
DDRS7	SSOE	Master Mode	Slave Mode
0	0	Slave-select input with mode-fault detection	Slave-select input
0	1	Reserved	
1	0	General-purpose output	
1	1	Slave-select output	

15.5.6 Single-Wire Operation

Normally, the SPI operates as a 2-wire interface; it uses its MOSI and MISO pins for transmitting and receiving.

In single-wire operation, a master SPI uses the MOSI pin for both receiving and transmitting. The MOSI pin becomes a master out, master in (MOMI) pin. The MISO pin is disconnected from the SPI and is available as a general-purpose port S I/O pin.

A slave SPI in single-wire operation uses the MISO pin for both receiving and transmitting. The MISO pin becomes a slave in, slave out (SISO) pin. The MOSI pin is disconnected from the SPI and is available as a general-purpose I/O pin.

Setting serial pin control bit 0, SPC0, configures the SPI for single-wire operation. The direction of the single-wire pin depends on its data direction bit.

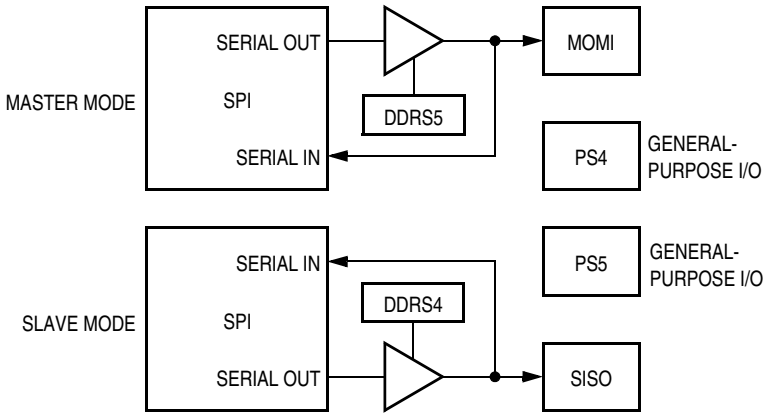


Figure 15-8. Single-Wire Operation (SPC0 = 1)

15.6 SPI Register Descriptions and Reset Initialization

This section describes the SPI registers and reset initialization.

15.6.1 SPI Control Register 1

Address: $00D0

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SPIE	SPE	SWOM	MSTR	CPOL	CPHA	SSOE	LSBF
Reset:	0	0	0	0	0	1	0	0

Figure 15-9. SPI Control Register 1 (SP0CR1)

Read: Anytime
Write: Anytime

SPIE — SPI Interrupt Enable Bit

SPIE enables the SPIF and MODF flags to generate interrupt requests.

1 = SPIF and MODF interrupt requests enabled
0 = SPIF and MODF interrupt requests disabled

SPE — SPI Enable Bit

Setting the SPE bit enables the SPI and configures port S pins 7–4 for SPI functions. Clearing SPE puts the SPI in a disabled, low-power state.

1 = SPI enabled
0 = SPI disabled

NOTE

When the MODF flag is set, SPE always reads as logic 0. Writing to SPI control register 1 is part of the mode fault recovery sequence.

SWOM — Port S Wired-OR Mode Bit

SWOM disables the pullup devices on port S pins 7–4 so that they become open-drain outputs.

1 = Open-drain port S pin 7–4 outputs
0 = Normal push-pull port S pin 7–4 outputs

MSTR — Master Mode Bit

MSTR selects master mode operation or slave mode operation.

1 = Master mode
0 = Slave mode

CPOL — Clock Polarity Bit

CPOL determines the logic state of the serial clock pin between transmissions. See Figure 15-4 and Figure 15-6.

1 = Active-high SCK
0 = Active-low SCK

CPHA — Clock Phase Bit

CPHA determines whether transmission begins on the falling edge of the $\overline{SS}$ pin or on the first edge of the serial clock. See Figure 15-4 and Figure 15-6.

1 = Transmission at first SCK edge
0 = Transmission at falling $\overline{SS}$ edge

SSOE — Slave Select Output Enable Bit

SSOE enables the output function of master $\overline{SS}$ pin when the DDRS7 bit is also set.

1 = $\overline{SS}$ output enabled
0 = $\overline{SS}$ output disabled

LSBF — LSB First Bit

LSBF enables least-significant-bit-first transmissions. It does not affect the position of data in the SPI data register; reads and writes of the SPI data register always have the MSB in bit 7.

1 = Least-significant-bit-first transmission
0 = Most-significant-bit-first transmission

15.6.2 SPI Control Register 2

Address: $00D1

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	PUPS	RDS	0	SPC0
Write:								
Reset:	0	0	0	0	1	0	0	0

= Unimplemented

Figure 15-10. SPI Control Register 2 (SP0CR2)

Read: Anytime
Write: Anytime

PUPS — Pullup Port S Bit

Setting PUPS enables internal pullup devices on all port S input pins. If a pin is programmed as output, the pullup device becomes inactive.

1 = Pullups enabled
0 = Pullups disabled

RDS — Reduced Drive Port S Bit

Setting RDS lowers the drive capability of all port S output pins for lower power consumption and less noise.

1 = Reduced drive
0 = Normal drive

SPC0 — Serial Pin Control Bit 0

SPC0 enables single-wire operation of the MOSI and MISO pins.

Table 15-2. Single-Wire Operation

Control Bits				Pins	
SPC0	MSTR	DDRS5	DDRS4	MOSI	MISO
1	1	0 1	—	Master input Master output	General-purpose I/O
	0	—	0 1	General-purpose I/O	Slave input Slave output

15.6.3 SPI Baud Rate Register

Address: $00D2

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	SPR2	SPR1	SPR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 15-11. SPI Baud Rate Register (SP0BR)

Read: Anytime
Write: Anytime

SPR2–SPR0 — SPI Clock Rate Select Bits

These bits select one of eight SPI baud rates as shown in Table 15-3. Reset clears SPR2–SPR0, selecting E-clock divided by two.

Table 15-3. SPI Clock Rate Selection

SPR[2:1:0]	E-Clock Divisor	Baud Rate (E-Clock = 4 MHz)	Baud Rate (E-Clock = 8 MHz)
000	2	2.0 MHz	4.0 MHz
001	4	1.0 MHz	2.0 MHz
010	8	500 kHz	1.0 MHz
011	16	250 kHz	500 kHz
100	32	125 kHz	250 kHz
101	64	62.5 kHz	125 kHz
110	128	31.3 kHz	62.5 kHz
111	256	15.6 kHz	31.3 kHz

15.6.4 SPI Status Register

Address: $00D3

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SPIF	WCOL	0	MODF	0	0	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 15-12. SPI Status Register (SP0SR)

Read: Anytime
Write: Has no meaning or effect

SPIF — SPI Flag

SPIF is set after the eighth serial clock cycle of a transmissson. SPIF generates an interrupt request if the SPIE bit in SPI control register 1 is set also. Clear SPIF by reading the SPI status register with SPIF set and then reading or writing to the SPI data register.

1 = Transfer complete
0 = Transfer not complete

WCOL — Write Collision Flag

WCOL is set when a write to the SPI data register occurs during a data transfer. The byte being transferred continues to shift out of the shift register, and the data written during the transfer is lost. WCOL does not generate an interrupt request. WCOL can be read when the transfer in progress is complete. Clear WCOL by reading the SPI status register with WCOL set and then reading or writing to the SPI data register.

1 = Write collision
0 = No write collision

MODF — Mode Fault Flag

MODF is set if the PS7 pin goes to logic 0 when it is configured as the $\overline{SS}$ input of a master SPI (MSTR = 1 and DDR7 = 0). Clear MODF by reading the SPI status register with MODF set and then writing to SPI control register 1.

1 = Mode fault
0 = No mode fault

NOTE

MODF is inhibited when the PS7 pin is configured as:

- *The $\overline{SS}$ output, DDRS7 = 1 and SSOE = 1, or*
- *A general-purpose output, DDRS7 = 1 and SSOE = 0*

15.6.5 SPI Data Register

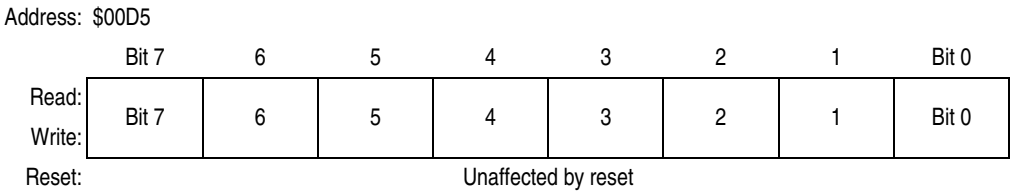


Figure 15-13. SPI Data Register (SP0DR)

Read: Anytime; normally, only after SPIF flag set
Write: Anytime a data transfer is not taking place

The SPI data register is both the input and output register for SPI data. Reads are double-buffered but writes cause data to be written directly into the SPI shift register. The data registers of two SPIs can be connected through their MOSI and MISO pins to form a distributed 16-bit register. A transmission between the SPIs shifts the data eight bit positions, exchanging the data between the master and the slave. The slave can also be another simpler device that only receives data from the master or that only sends data to the master.

15.7 External Pins

The SPI module has four I/O pins:

- MISO — Master data in, slave data out
- MOSI — Master data out, slave data in
- SCK — Serial clock
- $\overline{SS}$ — Slave select

The SPI has limited inter-integrated circuit (I^2C) capability (requiring software support) as a master in a single-master environment. To communicate with I^2C peripherals, MOSI becomes an open-drain output when the SWOM bit in the SPI control register is set. In I^2C communication, the MOSI and MISO pins are connected to a bidirectional pin from the I^2C peripheral and through a pullup resistor to V_{DD}.

15.7.1 MISO (Master In, Slave Out)

In a master SPI, MISO is the data input. In a slave SPI, MISO is the data output.

In a slave SPI, the MISO output pin is enabled only when its $\overline{SS}$ pin is at logic 0. To support a multiple-slave system, a logic 1 on the $\overline{SS}$ pin of a slave puts the MISO pin in a high-impedance state.

15.7.2 MOSI (Master Out, Slave In)

In a master SPI, MOSI is the data output. In a slave SPI, MOSI is the data input.

15.7.3 SCK (Serial Clock)

The serial clock synchronizes data transmission between master and slave devices. In a master SPI, the SCK pin is the clock output to the slave. In a slave MCU, the SCK pin is the clock input from the master.

15.7.4 $\overline{SS}$ (Slave Select)

The $\overline{SS}$ pin has multiple functions that depend on SPI configuration:

- The $\overline{SS}$ pin of a slave SPI is always configured as an input and allows the slave to be selected for transmission.
- When the CPHA bit is clear, the $\overline{SS}$ pin signals the start of a transmission.
- The $\overline{SS}$ pin of a master SPI can be configured as a mode-fault input, a slave-select output, or a general-purpose output.
  - As a mode-fault input (MSTR = 1, DDRS7 = 0, SSOE = 0), the $\overline{SS}$ pin can detect multiple masters driving MOSI and SPSCK.
  - As a slave-select output (MSTR = 1, DDRS7 = 1, SSOE = 1), the $\overline{SS}$ pin can select slaves for transmission.
  - When MSTR = 1, DDRS7 = 1, and SSOE = 0, the $\overline{SS}$ pin is available as a general-purpose output.

15.8 Low-Power Options

This section describes the three low-power modes:

- Run mode
- Wait mode
- Stop mode

15.8.1 Run Mode

Clearing the SPI enable bit, SPE, in SPI control register 1 reduces power consumption in run mode. SPI registers are still accessible when SPE is cleared, but clocks to the core of the SPI are disabled.

15.8.2 Wait Mode

The SPI remains active in wait mode. Any enabled interrupt request from the SPI can bring the MCU out of wait mode.

If SPI functions are not required during wait mode, reduce power consumption by disabling the SPI before executing the WAIT instruction.

15.8.3 Stop Mode

For reduced power consumption, the SPI is inactive in stop mode. The STOP instruction does not affect SPI register states. SPI operation resumes after an external interrupt.

Exiting stop mode by reset aborts any transmission in progress and resets the SPI. Entering stop mode during a transmission results in invalid data.

15.9 Interrupt Sources

Table 15-4. SPI Interrupt Sources

Interrupt Source	Flag	Local Enable	CCR Mask	Vector Address
Transmission complete	SPIF	SPIE	I bit	$FFD8, $FFD9
Mode fault	MODF			

15.10 General-Purpose I/O Ports

Port S shares its pins with the multiple serial interface (MSI). In all modes, port S pins PS7–PS0 are available for either general-purpose I/O or for SCI and SPI functions. See Chapter 13 Multiple Serial Interface (MSI).

15.11 Synchronous Character Transmission Using the SPI

This program is intended to communicate with the HC11 on the UDLP1 board. It utilizes the SPI to transmit synchronously characters in a string to be displayed on the LCD display. The program must configure the SPI as a master, and non-interrupt driven. The slave peripheral is chip-selected with the $\overline{SS}$ line at low voltage level. Between 8 bit transfers the $\overline{SS}$ line is held high. Also the clock idles low and takes data on the rising clock edges. The serial clock is set not to exceed 100 kHz baud rate.

15.11.1 Equipment

For this exercise, use the M68HC812A4EVB emulation board.

15.11.2 Code Listing

NOTE

A comment line is delimited by a semicolon. If there is no code before comment, a semicolon (;) must be placed in the first column to avoid assembly errors.

```
INCLUDE 'EQUATES.ASM'      ;Equates for all registers

; User Variables

; Bit Equates

; ---------------------------------------------------------------------
;           MAIN PROGRAM
; ---------------------------------------------------------------------
        ORG    $7000            ; 16K On-Board RAM, User code data area,
;                               ; start main program at $7000
MAIN:
        BSR    INIT             ; Subroutine to initialize SPI registers
        BSR    TRANSMIT         ; Subroutine to start transmission
FINISH:
        BRA    FINIS            ; Finished transmitting all DATA
```

```
; ---------------------------------------------------------------------
;*         SUBROUTINE INIT:
; ---------------------------------------------------------------------
INIT:
        BSET    PORTS,#$80        ; SET SS Line High to prevent glitch

        MOVB    #$E0,DDRS         ; Configure PORT S input/ouput levels
;                                 ; MOSI, SCK, SS* = ouput, MISO=Input

        MOVB    #$07,SP0BR        ; Select serial clock baud rate < 100 KHz

        MOVB    #$12,SP0CR1       ; Configure SPI(SP0CR1): No SPI interrupts,
;                                 ; MSTR=1, CPOL=0, CPHA=0

        MOVB    #$08,SP0CR2       ; Config. PORTS output drivers to operate normally,
;                                 ; and with active pull-up devices.

        LDX     #DATA             ; Use X register as pointer to first character

        LDAA    SP0SR             ; 1st step to clear SPIF Flag, Read SP0SR
        LDAA    SP0DR             ; 2nd step to clear SPIF Flag, Access SP0DR

        BSET    SP0CR1,#$40       ; Enable the SPI (SPE=1)

        RTS                       ; Return from subroutine

; ---------------------------------------------------------------------
;*        TRANSMIT SUBROUTINE
; ---------------------------------------------------------------------
TRANSMIT:
        LDAA    1,X+              ; Load Acc. with "NEW" character to send, Inc X
        BEQ     DONE              ; Detect if last character(0) has been transmitted
;                                 ; If last char. branch to DONE, else
        BCLR    PORTS,#$80        ; Assert SS Line to start X-misssion.
        STAA    SP0DR             ; Load Data into Data Reg.,X-mit.
;                                 ; it is also the 2nd step to clear SPIF flag.
FLAG:   BRCLR   SP0SR,#$80,FLAG   ;Wait for flag.
        BSET    PORTS,#$80        ; Disassert SS Line.
        BRA     TRANSMIT          ; Continue sending characters, Branch to TRANSMIT.

DONE:   RTS                       ; Return from subroutine

; ---------------------------------------------------------------------
;  TABLE OF DATA TO BE TRANSMITTED
; ---------------------------------------------------------------------
DATA:   DC.B    'Freescale'
        DC.B    $0D,$0A           ; Return (cr) ,Line Feed (LF)
EOT:    DC.B    $00               ; Byte used to test end of data = EOT

        END                       ; End of program
```

Chapter 16
Analog-to-Digital Converter (ATD)

16.1 Introduction

The analog-to-digital converter (ATD) is an 8-channel, 8-bit, multiplexed-input, successive approximation analog-to-digital converter, accurate to ±1 least significant bit (LSB). It does not require external sample and hold circuits because of the type of charge redistribution technique used. The ATD converter timing can be synchronized to the system P-clock. The ATD module consists of a 16-word (32-byte) memory-mapped control register block used for control, testing, and configuration.

16.2 Features

Features of the ATD module include:

- Eight multiplexed input channels
- Multiplexed-input successive approximation
- 8-bit resolution
- Single or continuous conversion
- Conversion complete flag with CPU interrupt request
- Selectable ATD clock

16.3 Block Diagram

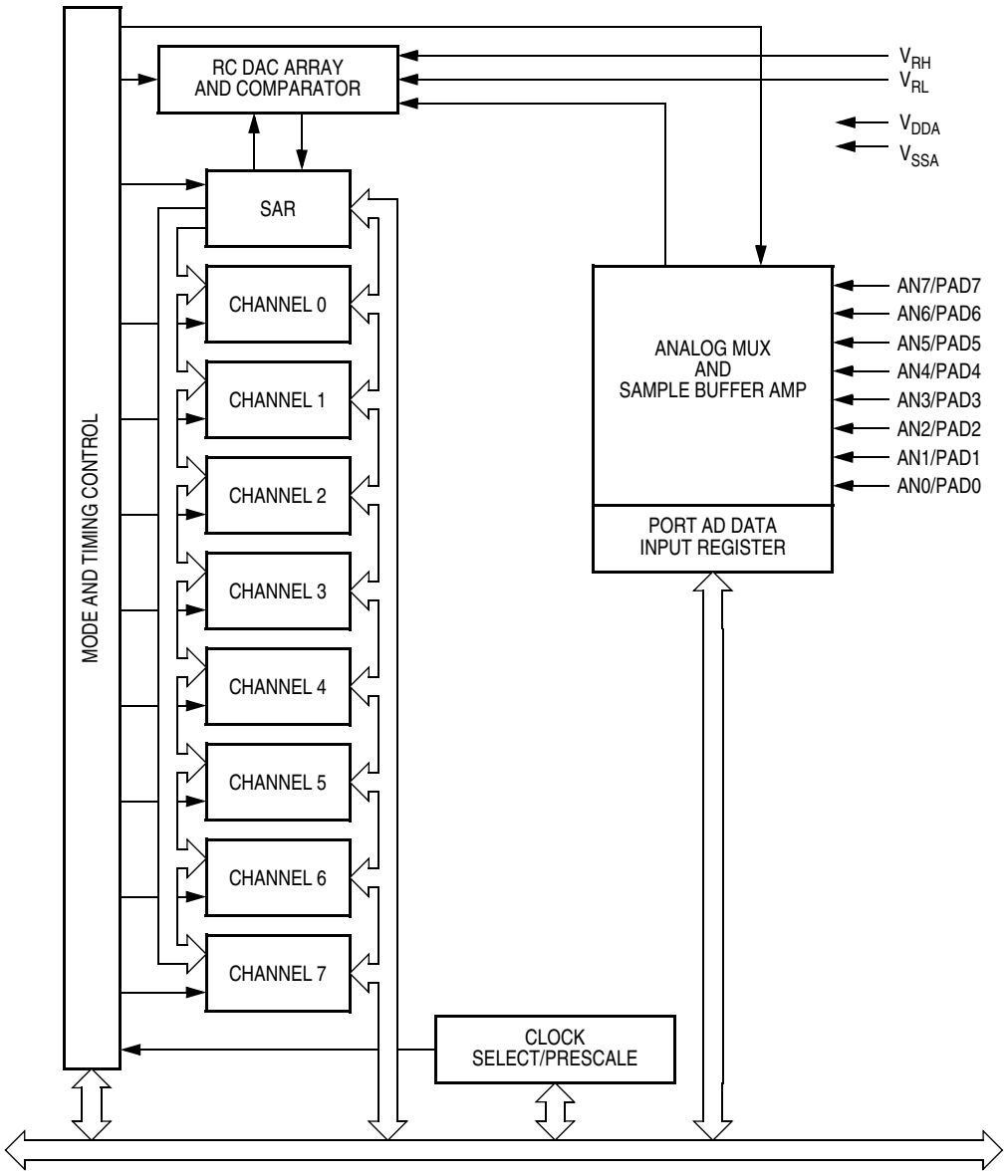


Figure 16-1. ATD Block Diagram

16.4 Register Map

NOTE

The register block can be mapped to any 2-Kbyte boundary within the standard 64-Kbyte address space. The register block occupies the first 512 bytes of the 2-Kbyte block. This register map shows default addressing after reset.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0060	ATD Control Register 0 (ATDCTL0) See page 199.	Read: Write:	0	0	0	0	0	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$0061	ATD Control Register 1 (ATDCTL1) See page 199.	Read:	0	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0062	ATD Control Register 2 (ATDCTL2) See page 200.	Read:	ADPU	AFFC	AWAI	0	0	0	ASCIE	ASCIF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0063	ATD Control Register 3 (ATDCTL3) See page 201.	Read:	0	0	0	0	0	0	FRZ1	FRZ0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0064	ATD Control Register 4 (ATDCTL4) See page 201.	Read:	0	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
		Write:								
		Reset:	0	0	0	0	0	0	0	1
$0065	ATD Control Register 5 (ATDCTL5) See page 202.	Read:	0	S8CM	SCAN	MULT	CD	CC	CB	CA
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0066	ATD Status Register 1 (ATDSTAT1) See page 204.	Read:	SCF	0	0	0	0	CC2	CC1	CC0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0067	ATD Status Register 2 (ATDSTAT2) See page 204.	Read:	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0068	ATD Test Register 1 (ATDTEST1) See page 205.	Read:	SAR9	SAR8	SAR7	SAR6	SAR5	SAR4	SAR3	SAR2
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0069	ATD Test Register 2 (ATDTEST2) See page 205.	Read:	SAR1	SAR0	RST	TSTOUT	TST3	TST2	TST1	TST0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-2. ATD I/O Register Summary

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$006F	Port AD Data Input Register (PORTAD) See page 207.	Read:	PAD7	PAD6	PAD5	PAD4	PAD3	PAD2	PAD1	PAD0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0070	ATD Result Register 0 (ADR0H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0072	ATD Result Register 1 (ADR1H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0074	ATD Result Register 2 (ADR2H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0076	ATD Result Register 3 (ADR3H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$0078	ATD Result Register 4 (ADR4H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$007A	ATD Result Register 5 (ADR5H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$007C	ATD Result Register 6 (ADR6H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							
$007E	ATD Result Register 7 (ADR7H) See page 206.	Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
		Write:								
		Reset:	Indeterminate							

= Unimplemented

Figure 16-2. ATD I/O Register Summary (Continued)

16.5 Functional Description

A single conversion sequence consists of four or eight conversions, depending on the state of the select 8-channel mode bit, S8CM, in ATD control register 5 (ATDCTL5). There are eight basic conversion modes.

In the non-scan modes, the sequence complete flag, SCF, is set after the sequence of four or eight conversions has been performed and the ATD module halts.

In the scan modes, the SCF flag is set after the first sequence of four or eight conversions has been performed, and the ATD module continues to restart the sequence.

In both modes, the CCF flag associated with each register is set when that register is loaded with the appropriate conversion result. That flag is cleared automatically when that result register is read. The conversions are started by writing to the control registers.

The ATD control register 4 selects the clock source and sets up the prescaler. Writes to the ATD control registers initiate a new conversion sequence. If a write occurs while a conversion is in progress, the conversion is aborted and ATD activity halts until a write to ATDCTL5 occurs.

The ATD control register 5 selects conversion modes and conversion channel(s) and initiates conversions.

A write to ATDCTL5 initiates a new conversion sequence. If a conversion sequence is in progress when a write occurs, the sequence is aborted and the SCF and CCF flags are cleared.

16.6 Registers and Reset Initialization

This section describes the registers and reset initialization.

16.6.1 ATD Control Register 0

Address: $0060

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	0	0	0	0	0	0	0	0
Reset:	0	0	0	0	0	0	0	0

Figure 16-3. ATD Control Register 0 (ATDCTL0)

NOTE

Writing to this register aborts the current conversion sequence.

16.6.2 ATD Control Register 1

Address: $0061

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-4. ATD Control Register 1 (ATDCTL1)

16.6.3 ATD Control Register 2

Address: $0062

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ADPU	AFFC	AWAI	0	0	0	ASCIE	ASCIF
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-5. ATD Control Register 2 (ATDCTL2)

Read: Anytime
Write: Anytime except ASCIF flag, which is read-only

NOTE

Writing to this register aborts the current conversion sequence.

ADPU — ATD Power-up Bit

ADPU enables the clock signal to the ATD and powers up its analog circuits.

1 = ATD enabled
0 = ATD disabled

NOTE

After ADPU is set, the ATD requires an analog circuit stabilization period.

AFFC — ATD Fast Flag Clear Bit

When AFFC is set, writing to a result register (ADR0H–ADR7H) clears the associated CCF flag if it is set. When AFFC is clear, clearing a CCF flag requires a read of the status register followed by a read of the result register.

1 = Fast CCF clearing enabled
0 = Fast CCF clearing disabled

AWAI — ATD Stop in Wait Mode Bit

ASWAI disables the ATD in wait mode for lower power consumption.

1 = ATD disabled in wait mode
0 = ATD enabled in wait mode

ASCIE — ATD Sequence Complete Interrupt Enable Bit

ASCIE enables interrupt requests generated by the ATD sequence complete interrupt flag, ASCIF.

1 = ASCIF interrupt requests enabled
0 = ASCIF interrupt requests disabled

ASCIF — ATD Sequence Complete Interrupt Flag

ASCIF is set when a conversion sequence is finished. If the ATD sequence complete interrupt enable bit, ASCIE, is also set, ASCIF generates an interrupt request.

1 = Conversion sequence complete
0 = Conversion sequence not complete

NOTE

The ASCIF flag is set only when a conversion sequence is completed and ASCIE = 1 or interrupts on the analog-to-digital converter (ATD) module are enabled.

16.6.4 ADT Control Register 3

Address: $0063

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	FRZ1	FRZ0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-6. ATD Control Register 3 (ATDCTL3)

FRZ1 and FRZ0 — Freeze Bits

The FRZ bits suspend ATD operation for background debugging. When debugging an application, it is useful in many cases to have the ATD pause when a breakpoint is encountered. These two bits determine how the ATD responds when background debug mode becomes active. See Table 16-1.

Table 16-1. ATD Response to Background Debug Enable

FRZ1:FRZ0	ATD Response
00	Continue conversions in active background mode
01	Reserved
10	Finish current conversion, then freeze
11	Freeze when BDM is active

16.6.5 ATD Control Register 4

Address: $0064

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
Write:								
Reset:	0	0	0	0	0	0	0	1

= Unimplemented

Figure 16-7. ATD Control Register 4 (ATDCTL4)

SMP1 and SMP0 — Sample Time Select Bits

These bits select one of four sample times after the buffered sample and transfer has occurred. Total conversion time depends on initial sample time (two ATD clocks), transfer time (four ATD clocks), final sample time (programmable, refer to Table 16-2), and resolution time (10 ATD clocks).

Table 16-2. Final Sample Time Selection

SMP[1:0]	Final Sample Time	Total 8-Bit Conversion Time
00	2 ATD clock periods	18 ATD clock periods
01	4 ATD clock periods	20 ATD clock periods
10	8 ATD clock periods	24 ATD clock periods
11	16 ATD clock periods	32 ATD clock periods

PRS[4:0] — Prescaler Select Bits

The prescaler divides the P-clock by the binary value written to PRS[4:0] plus one. To assure symmetry of the prescaler output, an additional divide-by-two circuit generates the ATD module clock. Clearing PRS[4:0] means the P-clock is divided only by the divide-by-two circuit.

The reset state of PRS[4:0] is 00001, giving a total P-clock divisor of four, which is appropriate for nominal operation at 2 MHz. Table 16-3 shows the appropriate range of system clock frequencies for each P clock divisor.

Table 16-3. Clock Prescaler Values

PRS[4:0]	P-Clock Divisor	Max P-Clock(1)	Min P-Clock(2)
00000	2	4 MHz	1 MHz
00001	4	8 MHz	2 MHz
00010	6	8 MHz	3 MHz
00011	8	8 MHz	4 MHz
00100	10	8 MHz	5 MHz
00101	12	8 MHz	6 MHz
00110	14	8 MHz	7 MHz
00111	16	8 MHz	8 MHz
01xxx	Do not use		
1xxxx			

1. Maximum conversion frequency is 2 MHz. Maximum P-clock divisor value becomes maximum conversion rate that can be used on this ATD module.
2. Minimum conversion frequency is 500 kHz. Minimum P-clock divisor value becomes minimum conversion rate that this ATD can perform.

16.6.6 ATD Control Register 5

Address: $0065

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	S8CM	SCAN	MULT	CD	CC	CB	CA
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-8. ATD Control Register 5 (ATDCTL5)

Read: Anytime
Write: Anytime

S8CM — Select Eight Conversions Mode Bit

S8CM selects conversion sequences of either eight or four conversions.

1 = Eight conversion sequences
0 = Four conversion sequences

SCAN — Continuous Channel Scan Bit

SCAN selects a single conversion sequence or continuous conversion sequences.

1 = Continuous conversion sequences (scan mode)
0 = Single conversion sequence

MULT — Multichannel Conversion Bit

Refer to Table 16-4.

1 = Conversions of sequential channels

0 = Conversions of a single input channel selected by the CD, CC, CB, and CA bits

CD, CC, CB, and CA — Channel Select Bits

The channel select bits select the input to convert. LT = 1, the ATD sequencer selects

Table 16-4. Multichannel Mode Result Register Assignment(1)

S8CM	CD	CC	CB	CA	Channel Input	Result in ADRxH
0	0	0	0*	0*	AN0	ADRxH0
			0*	1*	AN1	ADRxH1
			1*	0*	AN2	ADRxH2
			1*	1*	AN3	ADRxH3
		1	0*	0*	AN4	ADRxH0
			0*	1*	AN5	ADRxH1
			1*	0*	AN6	ADRxH2
			1*	1*	AN7	ADRxH3
0	1	0	0*	0*	Reserved	ADRxH0
			0*	1*	Reserved	ADRxH1
			1*	0*	Reserved	ADRxH2
			1*	1*	Reserved	ADRxH3
		1	0*	0*	V_{RH}	ADRxH0
			0*	1*	V_{RL}	ADRxH1
			1*	0*	$(V_{RH} + V_{RL})/2$	ADRxH2
			1*	1*	Test/Reserved	ADRxH3
1	0	0*	0*	0*	AN0	ADRxH0
			0*	1*	AN1	ADRxH1
			1*	0*	AN2	ADRxH2
			1*	1*	AN3	ADRxH3
		1*	0*	0*	AN4	ADRxH4
			0*	1*	AN5	ADRxH5
			1*	0*	AN6	ADRxH6
			1*	1*	AN7	ADRxH7
1	1	0*	0*	0*	Reserved	ADRxH0
			0*	1*	Reserved	ADRxH1
			1*	0*	Reserved	ADRxH2
			1*	1*	Reserved	ADRxH3
		1*	0*	0*	V_{RH}	ADRxH4
			0*	1*	V_{RL}	ADRxH5
			1*	0*	$(V_{RH} + V_{RL})/2$	ADRxH6
			1*	1*	Test/Reserved	ADRxH7

1. When MULT = 1, bits with asterisks are don't care bits. The 4-conversion sequence from AN0 to AN3 or the 8-conversion sequence from AN0 to AN7 is completed in the order shown.
When MULT = 0, the CD, CC, CB, and CA bits select one input channel. The conversion sequence is performed on this channel only.

16.6.7 ATD Status Registers

Address: $0066

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SCF	0	0	0	0	CC2	CC1	CC0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-9. ATD Status Register 1 (ATDSTAT1)

Address: $0067

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-10. ATD Status Register 2 (ATDSTAT2)

Read: Anytime
Write: Special mode only

SCF — Sequence Complete Flag

In single conversion sequence mode (SCAN = 0 in ATDCTL5), SCF is set at the end of the conversion sequence.

In continuous conversion mode (SCAN = 1 in ATDCTL5), SCF is set at the end of the first conversion sequence.

Clear SCF by writing to control register 5 (ATDCTL5) to initiate a new conversion sequence. When the fast flag clear enable bit, AFFC, is set, SCF is cleared after the first result register is read.

CC2–CC0 — Conversion Counter Bits

This 3-bit value reflects the value of the conversion counter pointer in either a 4-conversion or 8-conversion sequence. The pointer shows which channel is currently being converted and which result register will be written next.

CCF7–CCF0 — Conversion Complete Flags

Each ATD channel has a CCF flag. A CCF flag is set at the end of the conversion on that channel. Clear a CCF flag by reading status register 1 with the flag set and then reading the result register of that channel. When the fast flag clear enable bit, AFFC, is set, reading the result register clears the associated CCF flag even if the status register has not been read.

16.6.8 ATD Test Registers

Address: $0068

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SAR9	SAR8	SAR7	SAR6	SAR5	SAR4	SAR3	SAR2
Reset:	0	0	0	0	0	0	0	0

Figure 16-11. ATD Test Register 1 (ATDTEST1)

Address: $0069

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SAR1	SAR0	RST	TSTOUT	TST3	TST2	TST1	TST0
Reset:	0	0	0	0	0	0	0	0

Figure 16-12. ATD Test Register 2 (ATDTEST2)

Read: Special modes only
Write: Special modes only

The test registers control various special modes which are used during manufacturing. In the normal modes, reads of the test register return 0 and writes have no effect.

SAR9–SAR0 — SAR Data Bits

Reads of this byte return the current value in the SAR. Writes to this byte change the SAR to the value written. Bits SAR9–SAR2 reflect the eight SAR bits used during the resolution process for an 8-bit result. SAR1 and SAR0 are reserved to allow future derivatives to increase ATD resolution to 10 bits.

RST — Reset Bit

When set, this bit causes all registers and activity in the module to assume the same state as out of power-on reset (except for ADPU bit in ATDCTL2, which remains set, allowing the ATD module to remain enabled).

TSTOUT — Multiplex Output of TST3–TST0 (factory use)

TST3–TST0 — Test Bits 3 to 0 (reserved)

Selects one of 16 reserved factory testing modes

16.6.9 ATD Result Registers

Address: ADR0H: $0070
ADR1H: $0072
ADR2H: $0074
ADR3H: $0076
ADR4H: $0078
ADR5H: $007A
ADR6H: $007C
ADR7H: $007E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ADRxH7	ADRxH6	ADRxH5	ADRxH4	ADRxH3	ADRxH2	ADRxH1	ADRxH0
Write:								
Reset:	Indeterminate							

= Unimplemented

Figure 16-13. ATD Result Registers (ADR0H–ADR7H)

Read: Anytime
Write: Has no meaning or effect

ADRxH7–ADRxH0 — ATD Conversion Result Bits

These bits contain the left justified, unsigned result from the ATD conversion. The channel from which this result was obtained depends on the conversion mode selected. These registers are always read-only in normal mode.

16.7 Low-Power Options

This section describes the three low-power modes:

- Run mode
- Wait mode
- Stop mode

16.7.1 Run Mode

Clearing the ATD power-up bit, ADPU, in ATD control register 2 (ATDCTL2) reduces power consumption in run mode. ATD registers are still accessible, but the clock to the ATD is disabled and ATD analog circuits are powered down.

16.7.2 Wait Mode

ATD operation in wait mode depends on the state of the ATD stop in wait bit, AWAI, in ATD control register 2 (ATDCTL2).

- If AWAI is clear, the ATD operates normally when the CPU is in wait mode
- If AWAI is set, the ATD clock is disabled and conversion continues unless ASWAI bit in ATDCTL2 register is set.

16.7.3 Stop Mode

The ATD is inactive in stop mode for reduced power consumption. The STOP instruction aborts any conversion sequence in progress.

16.8 Interrupt Sources

Table 16-5. ATD Interrupt Sources

Interrupt Source	Flag	Local Enable	CCR Mask	Vector Address
Conversion sequence complete	ASCIF	ASCIE	I bit	$FFD2, $FFD3

NOTE

The ASCIF flag is set only when a conversion sequence is completed and ASCIE = 1 or interrupts on the analog-to-digital converter (ATD) module are enabled.

16.9 General-Purpose Ports

Port AD is an input-only port. When the ATD is enabled, port AD is the analog input port for the ATD. Setting the ATD power-up bit, ADPU, in ATD control register 2 enables the ATD.

Port AD is available for general-purpose input when the ATD is disabled. Clearing the ADPU bit disables the ATD.

16.10 Port AD Data Register

Address: $006F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PAD7	PAD6	PAD5	PAD4	PAD3	PAD2	PAD1	PAD0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-14. Port AD Data Input Register (PORTAD)

Read: Anytime; reads return logic levels on the PAD pins
Write: Has no meaning or effect

PAD7–PAD0 — Port AD Data Input Bits

16.11 Using the ATD to Measure a Potentiometer Signal

This exercise allows the student to utilize the ATD on the HC12 to measure a potentiometer signal output routed from the UDLP1 board to the HC12 ATD pin PAD6. First the ATDCTL registers are initialized. A delay loop of 100 μs is then executed. The resolution is set up followed by a conversion set up on channel 6. After waiting for the status bit to set, the result goes to the D accumulator. If the program is working properly, a different value should be found in the D accumulator as the left potentiometer is varied for each execution of the program.

16.11.1 Equipment

For this exercise, use the M68HC812A4EVB emulation board.

16.11.2 Code Listing

NOTE

A comment line is delimited by a semicolon. If there is no code before comment, a semicolon (;) must be placed in the first column to avoid assembly errors.

```
; ---------------------------------------------------------------------
;           MAIN PROGRAM
; ---------------------------------------------------------------------
          ORG      $7000          ; 16K On-Board RAM, User code data area,
;                                  start main program at $7000

MAIN:
          BSR      INIT           ; Branch to INIT subroutine to Initialize ATD
          BSR      CONVERT        ; Branch to CONVERT Subroutine for conversion
DONE:     BRA      DONE           ; Branch to Self, Convenient place for breakpoint

; ---------------------------------------------
;           Subroutine INIT: Initialize ATD    ;
; ---------------------------------------------
INIT:
          LDAA     #$80           ; Allow ATD to function normally,
          STAA     ATDCTL2        ; ATD Flags clear normally & disable interrupts

          BSR      DELAY          ; Delay (100 uS) for WAIT delay time.

          LDAA     #$00           ; Select continue conversion in BGND Mode
          STAA     ATDCTL3        ; Ignore FREEZE in ATDCTL3

          LDAA     #$01           ; Select Final Sample time = 2 A/D clocks
          STAA     ATDCTL4        ; Prescaler = Div by 4 (PRS4:0 = 1)

          RTS                     ; Return from subroutine
```

```
; ---------------------------------------------
;              Subroutine CONVERT:             ;
; ---------------------------------------------
; Set-up ATD, make single conversion and store the result to a memory location.
; Configure and start A/D conversion
; Analog Input Signal: On PORT AD6
; Convert: using single channel, non-continuous
; The result will be located in ADR2H

CONVERT:
         LDAA     #$06           ; Initializes ATD SCAN=0,MULT=0, PAD6,
;                                 ; Write Clears Flag
         STAA     ATDCTL5        ; 4 conversions on a Single Conversion
;                                 ; sequence,

WTCONV:  BRCLR    ATDSTATH,#$80,WTCONV; Wait for Sequence Complete Flag

         LDD      ADR2H          ; Loads conversion result(ADR2H)
;                                 ; into Accumulator

         BRA      CONVERT        ; Continuously updates results

         RTS                     ; Return from subroutine

;* -------------------------------
;*    Subroutine DELAY 100 uS    *
;* -------------------------------
; Delay Required for ATD converter to Stabilize (100 uSec)

         LDAA     #$C8           ; Load Accumulator with "100 uSec delay value"
DELAY:   DECA                    ; Decrement ACC
         BNE      DELAY          ; Branch if not equal to Zero
         RTS                     ; Return from subroutine

         END                     ; End of program
```

Chapter 17
Development Support

17.1 Introduction

This section describes:

- Instruction queue
- Queue tracking signals
- Background debug mode (BDM)
- Instruction tagging

17.2 Instruction Queue

The CPU12 instruction queue provides at least three bytes of program information to the CPU when instruction execution begins. The CPU12 always completely finishes executing an instruction before beginning to execute the next instruction. Status signals IPIPE[1:0] provide information about data movement in the queue and indicate when the CPU begins to execute instructions. This makes it possible to monitor CPU activity on a cycle-by-cycle basis for debugging. Information available on the IPIPE[1:0] pins is time multiplexed. External circuitry can latch data movement information on rising edges of the E-clock signal; execution start information can be latched on falling edges. Table 17-1 shows the meaning of data on the pins.

Table 17-1. IPIPE Decoding

Data Movement — IPIPE[1:0] Captured at Rising Edge of E Clock(1)

IPIPE[1:0]	Mnemonic	Meaning
0:0	—	No movement
0:1	LAT	Latch data from bus
1:0	ALD	Advance queue and load from bus
1:1	ALL	Advance queue and load from latch

Execution Start — IPIPE[1:0] Captured at Falling Edge of E Clock(2)

IPIPE[1:0]	Mnemonic	Meaning
0:0	—	No start
0:1	INT	Start interrupt sequence
1:0	SEV	Start even instruction
1:1	SOD	Start odd instruction

1. Refers to data that was on the bus at the previous E falling edge.
2. Refers to bus cycle starting at this E falling edge.

Program information is fetched a few cycles before it is used by the CPU. To monitor cycle-by-cycle CPU activity, it is necessary to externally reconstruct what is happening in the instruction queue. Internally, the MCU only needs to buffer the data from program fetches. For system debug, it is necessary to keep the data and its associated address in the reconstructed instruction queue. The raw signals required for reconstruction of the queue are ADDR, DATA, R/$\overline{W}$, ECLK, and status signals IPIPE[1:0].

The instruction queue consists of two 16-bit queue stages and a holding latch on the input of the first stage. To advance the queue means to move the word in the first stage to the second stage and move the word from either the holding latch or the data bus input buffer into the first stage. To start even (or odd) instruction means to execute the opcode in the high-order (or low-order) byte of the second stage of the instruction queue.

17.3 Background Debug Mode (BDM)

Background debug mode (BDM) is used for:

- System development
- In-circuit testing
- Field testing
- Programming

BDM is implemented in on-chip hardware and provides a full set of debug options.

Because BDM control logic does not reside in the CPU, BDM hardware commands can be executed while the CPU is operating normally. The control logic generally uses CPU dead cycles to execute these commands, but can steal cycles from the CPU when necessary. Other BDM commands are firmware based and require the CPU to be in active background mode for execution. While BDM is active, the CPU executes a firmware program located in a small on-chip ROM that is available in the standard 64-Kbyte memory map only while BDM is active.

The BDM control logic communicates with an external host development system serially, via the BKGD pin. This single-wire approach minimizes the number of pins needed for development support.

17.3.1 BDM Serial Interface

The BDM serial interface requires the external controller to generate a falling edge on the BKGD pin to indicate the start of each bit time. The external controller provides this falling edge whether data is transmitted or received.

BKGD is a pseudo-open-drain pin that can be driven either by an external controller or by the MCU. Data is transferred MSB first at 16 E-clock cycles per bit (nominal speed). The interface times out if 512 E-clock cycles occur between falling edges from the host. The hardware clears the command register when this timeout occurs.

The BKGD pin can receive a high or low level or transmit a high or low level. The following diagrams show timing for each of these cases. Interface timing is synchronous to MCU clocks but asynchronous to the external host. The internal clock signal is shown for reference in counting cycles.

Figure 17-1 shows an external host transmitting a logic 1 or 0 to the BKGD pin of a target M68HC12 MCU. The host is asynchronous to the target so there is a 0-to-1 cycle delay from the host-generated falling edge to where the target perceives the beginning of the bit time. Ten target E cycles later, the target senses the bit level on the BKGD pin. Typically, the host actively drives the pseudo-open-drain BKGD pin during host-to-target transmissions to speed up rising edges. Since the target does not drive the BKGD pin during this period, there is no need to treat the line as an open-drain signal during host-to-target transmissions.

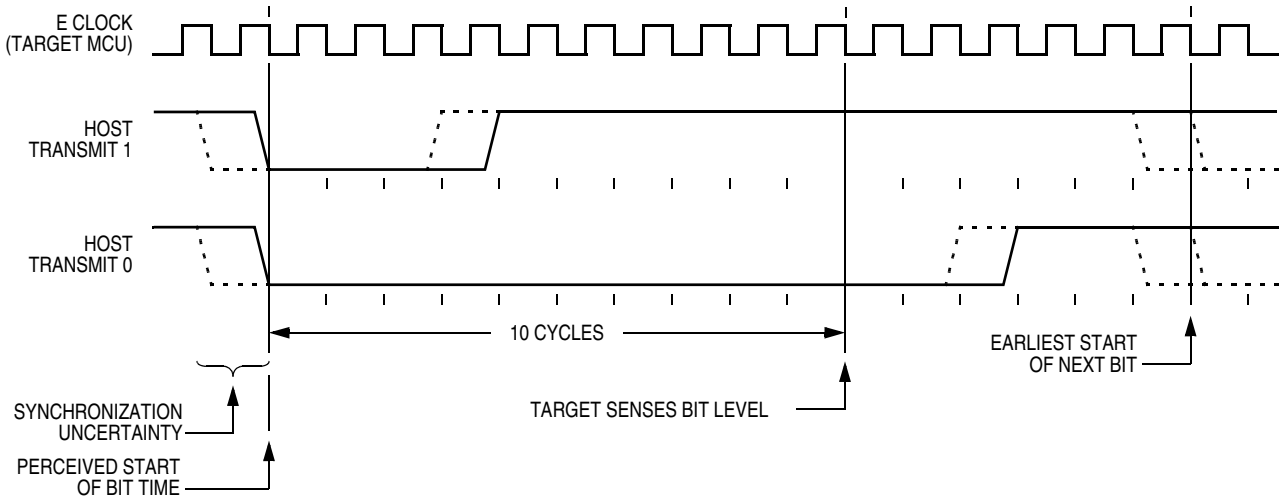


Figure 17-1. BDM Host-to-Target Serial Bit Timing

Figure 17-2 shows the host receiving a logic 1 from the target MC68HC812A4 MCU. Since the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target MCU. The host holds the BKGD pin low long enough for the target to recognize it (at least two target E cycles). The host must release the low drive before the target MCU drives a brief active-high speed-up pulse seven cycles after the perceived start of the bit time. The host should sample the bit level about 10 cycles after it started the bit time.

Figure 17-3 shows the host receiving a logic 0 from the target MC68HC812A4 MCU. Since the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target MCU. The host initiates the bit time but the target MC68HC812A4 finishes it. Since the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 E-clock cycles, then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 cycles after starting the bit time.

17.3.2 Enabling BDM Firmware Commands

BDM is available in all operating modes, but must be made active before firmware commands can be executed. BDM is enabled by setting the ENBDM bit in the BDM STATUS register via the single wire interface (using a hardware command; WRITE_BD_BYTE at $FF01). BDM must then be activated to map BDM registers and ROM to addresses $FF00 to $FFFF and to put the MCU in active background mode.

After the firmware is enabled, BDM can be activated by the hardware BACKGROUND command, by the BDM tagging mechanism, or by the CPU BGND instruction. An attempt to activate BDM before firmware has been enabled causes the MCU to resume normal instruction execution after a brief delay.

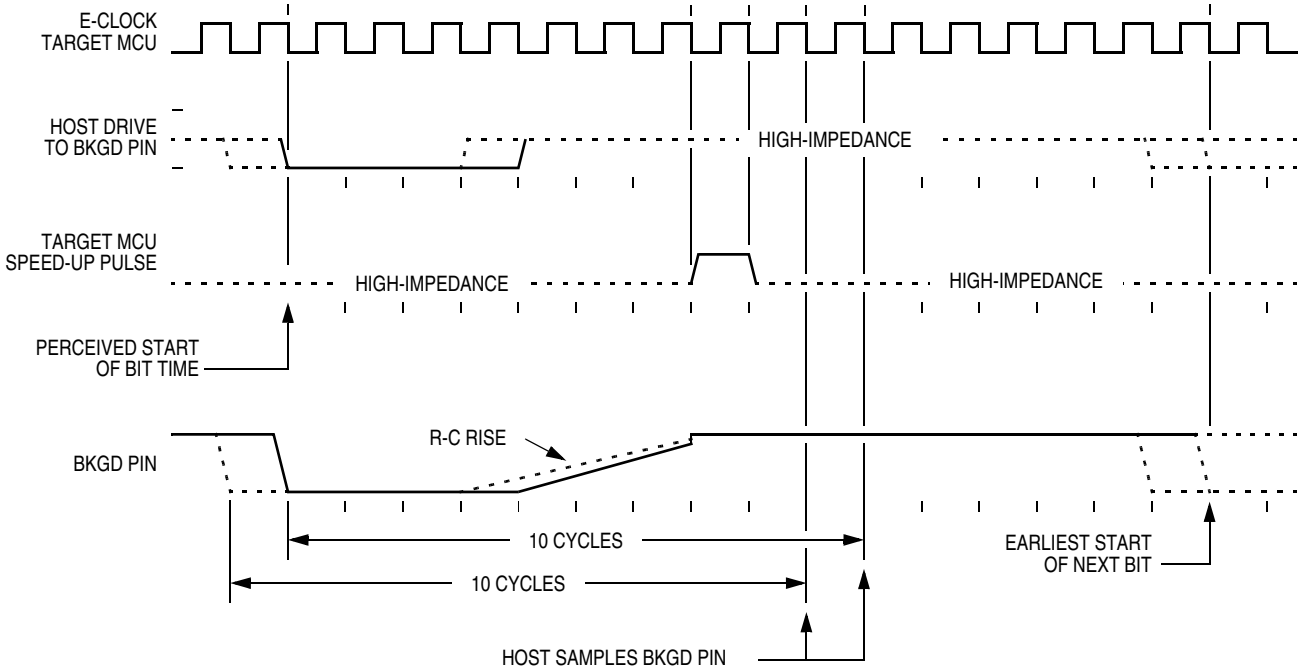


Figure 17-2. BDM Target-to-Host Serial Bit Timing (Logic 1)

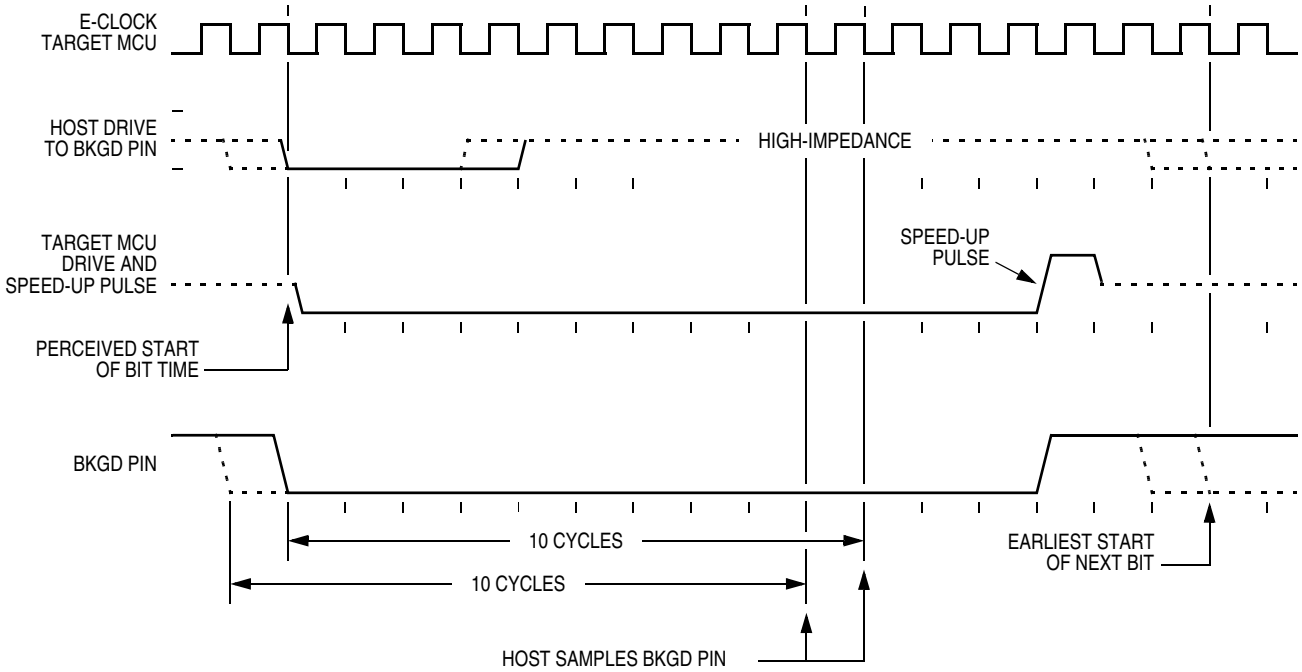


Figure 17-3. BDM Target-to-Host Serial Bit Timing (Logic 0)

BDM becomes active at the next instruction boundary following execution of the BDM BACKGROUND command, but tags activate BDM before a tagged instruction is executed.

In special single-chip mode, background operation is enabled and active immediately out of reset. This active case replaces the M68HC11 boot function and allows programming a system with blank memory.

While BDM is active, a set of BDM control registers are mapped to addresses $FF00 to $FF06. The BDM control logic uses these registers which can be read anytime by BDM logic, not user programs. Refer to 17.4 BDM Registers for detailed descriptions.

Some on-chip peripherals have a BDM control bit which allows suspending the peripheral function during BDM. For example, if the timer control is enabled, the timer counter is stopped while in BDM. Once normal program flow is continued, the timer counter is re-enabled to simulate real-time operations.

17.3.3 BDM Commands

All BDM command opcodes are eight bits long and can be followed by an address and/or data, as indicated by the instruction. These commands do not require the CPU to be in active BDM mode for execution.

The host controller must wait 150 cycles for a non-intrusive BDM command to execute before another command can be sent. This delay includes 128 cycles for the maximum delay for a dead cycle. For data read commands, the host must insert this delay between sending the address and attempting to read the data.

BDM logic retains control of the internal buses until a read or write is completed. If an operation can be completed in a single cycle, it does not intrude on normal CPU operation. However, if an operation requires multiple cycles, CPU clocks are frozen until the operation is complete.

The CPU must be in background mode to execute commands that are implemented in the BDM ROM. The BDM ROM is located at $FF20 to $FFFF while BDM is active. There are also seven bytes of BDM registers which are located at $FF00 to $FF06 while BDM is active. The CPU executes code from this ROM to perform the requested operation. These commands are shown in Table 17-2 and Table 17-3.

Table 17-2. BDM Commands Implemented in Hardware

Command	Opcode (Hex)	Data	Description
BACKGROUND	90	None	Enter background mode, if firmware is enabled.
READ_BD_BYTE	E4	16-bit address 16-bit data out	Read from memory with BDM in map (may steal cycles if external access); data for odd address on low byte, data for even address on high byte
STATUS(1)	E4	FF01, 0000 0000 (out)	READ_BD_BYTE $FF01. Running user code; BGND instruction is not allowed
		FF01, 1000 0000 (out)	READ_BD_BYTE $FF01. BGND instruction is allowed.
		FF01, 1100 0000 (out)	READ_BD_BYTE $FF01. Background mode active, waiting for single wire serial command
READ_BD_WORD	EC	16-bit address 16-bit data out	Read from memory with BDM in map (may steal cycles if external access); must be aligned access
READ_BYTE	E0	16-bit address 16-bit data out	Read from memory with BDM out of map (may steal cycles if external access); data for odd address on low byte, data for even address on high byte

Table 17-2. BDM Commands Implemented in Hardware (Continued)

Command	Opcode (Hex)	Data	Description
READ_WORD	E8	16-bit address 16-bit data out	Read from memory with BDM out of map (may steal cycles if external access); must be aligned access
WRITE_BD_BYTE	C4	16-bit address 16-bit data in	Write to memory with BDM in map (may steal cycles if external access); data for odd address on low byte, data for even address on high byte
ENABLE_ FIRMWARE(2)	C4	FF01, 1xxx xxxx (in)	Write byte $FF01, set the ENBDM bit. This allows execution of commands which are implemented in firmware. Typically, read STATUS, OR in the MSB, write the result back to STATUS.
WRITE_BD_WORD	CC	16-bit address 16-bit data in	Write to memory with BDM in map (may steal cycles if external access); must be aligned access
WRITE_BYTE	C0	16-bit address 16-bit data in	Write to memory with BDM out of map (may steal cycles if external access); data for odd address on low byte, data for even address on high byte
WRITE_WORD	C8	16-bit address 16-bit data in	Write to memory with BDM out of map (may steal cycles if external access); must be aligned access

1. STATUS command is a specific case of the READ_BD_BYTE command.
2. ENABLE_FIRMWARE is a specific case of the WRITE_BD_BYTE command.

Table 17-3. BDM Firmware Commands

Command	Opcode (Hex)	Data	Description
READ_NEXT	62	16-bit data out	X = X + 2; read next word pointed to by X
READ_PC	63	16-bit data out	Read program counter
READ_D	64	16-bit data out	Read D accumulator
READ_X	65	16-bit data out	Read X index register
READ_Y	66	16-bit data out	Read Y index register
READ_SP	67	16-bit data out	Read stack pointer
WRITE_NEXT	42	16-bit data in	X = X + 2; write next word pointed to by X
WRITE_PC	43	16-bit data in	Write program counter
WRITE_D	44	16-bit data in	Write D accumulator
WRITE_X	45	16-bit data in	Write X index register
WRITE_Y	46	16-bit data in	Write Y index register
WRITE_SP	47	16-bit data in	Write stack pointer
GO	08	None	Go to user program
TRACE1	10	None	Execute one user instruction, then return to BDM
TAGGO	18	None	Enable tagging and go to user program

17.4 BDM Registers

Seven BDM registers are mapped into the standard 64-Kbyte address space when BDM is active. The registers can be accessed with the hardware READ_BD and WRITE_BD commands, but must not be written during BDM operation. Most users are only interested in the STATUS register at $FF01; other registers are for use only by BDM firmware and logic.

The instruction register is discussed for two conditions:

- When a hardware command is executed
- When a firmware command is executed

17.4.1 BDM Instruction Register

This section describes the BDM instruction register under hardware command and firmware command.

17.4.1.1 Hardware Command

Address: $FF00

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	H/F	DATA	R/W	BKGND	W/B	BD/U	0	0
Reset:	0	0	0	0	0	0	0	0

Figure 17-4. BDM Instruction Register (INSTRUCTION)

The bits in the BDM instruction register have the following meanings when a hardware command is executed.

H/F — Hardware/Firmware Flag

1 = Hardware instruction
0 = Firmware instruction

DATA — Data Flag

1 = Data included in command
0 = No data

R/W — Read/Write Flag

0 = Write
1 = Read

BKGND — Hardware Request Bit to Enter Active Background Mode

1 = Hardware background command (INSTRUCTION = $90)
0 = Not a hardware background command

W/B — Word/Byte Transfer Flag

1 = Word transfer
0 = Byte transfer

BD/U — BDM Map/User Map Flag

Indicates whether BDM registers and ROM are mapped to addresses $FF00 to $FFFF in the standard 64-Kbyte address space. Used only by hardware read/write commands.

1 = BDM resources in map
0 = BDM resources not in map

17.4.1.2 Firmware Command

Address: $FF00

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	H/F	DATA	R/W	TTAGO		REGN		
Reset:	0	0	0	0	0	0	0	0

Figure 17-5. BDM Instruction Register (INSTRUCTION)

The bits in the BDM instruction register have the following meanings when a firmware command is executed.

H/F — Hardware/Firmware Flag

1 = Hardware control logic
0 = Firmware control logic

DATA — Data Flag

1 = Data included in command
0 = No data

R/W — Read/Write Flag

1 = Read
0 = Write

TTAGO — Trace, Tag, and Go Field

Table 17-4. TTAGO Decoding

TTAGO Value	Instruction
00	—
01	GO
10	TRACE1
11	TAGGO

REGN — Register/Next Field

Indicates which register is being affected by a command. In the case of a READ_NEXT or WRITE_NEXT command, index register X is pre-incremented by 2 and the word pointed to by X is then read or written.

Table 17-5. REGN Decoding

REGN Value	Instruction
000	—
001	—
010	READ/WRITE NEXT
011	PC
100	D
101	X
110	Y
111	SP

17.4.2 BDM Status Register

Address: $FF01

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	ENBDM	EDMACT	ENTAG	SDV	TRACE	0	0	0
Reset:	0	0	0	0	0	0	0	0
Single-chip peripheral:	1	0	0	0	0	0	0	0

Figure 17-6. BDM Status Register (STATUS)

This register can be read or written by BDM commands or firmware.

ENBDM — Enable BDM Bit (permit active background debug mode)

1 = BDM can be made active to allow firmware commands.
0 = BDM cannot be made active (hardware commands still allowed).

BDMACT — Background Mode Active Status Bit

1 = BDM active and waiting for serial commands
0 = BDM not active

ENTAG — Instruction Tagging Enable Bit

Set by the TAGGO instruction and cleared when BDM is entered.

1 = Tagging active (BDM cannot process serial commands while tagging is active.)
0 = Tagging not enabled or BDM active

SDV — Shifter Data Valid Bit

Shows that valid data is in the serial interface shift register. Used by firmware-based instructions.

1 = Valid data
0 = No valid data

TRACE — Asserted by the TRACE1 Instruction

17.4.3 BDM Shift Register

Address: $FF02

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	S15	S14	S13	S12	S11	S10	S9	S8
Reset:	0	0	0	0	0	0	0	0

Address: $FF03

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	S7	S6	S5	S4	S3	S2	S1	S0
Reset:	0	0	0	0	0	0	0	0

Figure 17-7. BDM Shift Register (SHIFTER)

This 16-bit register contains data being received or transmitted via the serial interface.

17.4.4 BDM Address Register

Address: $FF04

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	A15	A14	A13	A12	A11	A10	A9	A8
Reset:	0	0	0	0	0	0	0	0

Address: $FF05

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	A7	A6	A5	A4	A3	A2	A1	A0
Reset:	0	0	0	0	0	0	0	0

Figure 17-8. BDM Address Register (ADDRESS)

This 16-bit register is temporary storage for BDM hardware and firmware commands.

17.4.5 BDM CCR Holding Register

Address: $FF06

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	CCR7	CCR6	CCR5	CCR4	CCR3	CCR2	CCR1	CCR0
Reset:	0	0	0	0	0	0	0	0

Figure 17-9. BDM CCR Holding Register (CCRSAV)

This register preserves the content of the CPU12 CCR while BDM is active.

17.5 Instruction Tagging

The instruction queue and cycle-by-cycle CPU activity can be reconstructed in real time or from trace history that was captured by a logic analyzer. However, the reconstructed queue cannot be used to stop the CPU at a specific instruction, because execution has already begun by the time an operation is visible outside the MCU. A separate instruction tagging mechanism is provided for this purpose.

Executing the BDM TAGGO command configures two MCU pins for tagging. Tagging information is latched on the falling edge of ECLK along with program information as it is fetched. Tagging is allowed in all modes. Tagging is disabled when BDM becomes active and BDM serial commands cannot be processed while tagging is active.

$\overline{\text{TAGHI}}$ is a shared function of the BKGD pin.

$\overline{\text{TAGLO}}$ is a shared function of the PE3/$\overline{\text{LSTRB}}$ pin, a multiplexed I/O pin. For 1/4 cycle before and after the rising edge of the E-clock, this pin is the $\overline{\text{LSTRB}}$ driven output.

$\overline{\text{TAGLO}}$ and $\overline{\text{TAGHI}}$ inputs are captured at the falling edge of the E-clock. A logic 0 on $\overline{\text{TAGHI}}$ and/or $\overline{\text{TAGLO}}$ marks (tags) the instruction on the high and/or low byte of the program word that was on the data bus at the same falling edge of the E-clock.

The tag follows the information in the queue as the queue is advanced. When a tagged instruction reaches the head of the queue, the CPU enters active background debugging mode rather than executing the instruction. This is the mechanism by which a development system initiates hardware breakpoints.

Chapter 18
Electrical Characteristics

18.1 Maximum Ratings

Maximum ratings are the extreme limits to which the MCU can be exposed without permanently damaging it.

NOTE

This device is not guaranteed to operate properly at the maximum ratings. Refer to 18.4 DC Electrical Characteristics for guaranteed operating conditions.

Rating	Symbol	Value	Unit
Supply voltage	V_{DD} V_{DDA} V_{DDX}	–0.3 to +6.5	V
Input voltage	V_{In}	–0.3 to +6.5	V
Maximum current per pin excluding V_{DD} and V_{SS}	I_{In}	± 25	mA
Storage temperature	T_{STG}	–55 to +150	°C
V_{DD} differential voltage	V_{DD}–V_{DDX}	6.5	V

NOTE

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{In} and V_{Out} be constrained to the range $V_{SS} \le (V_{In}$ or $V_{Out}) \le V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (for example, either V_{SS} or V_{DD}).

18.2 Functional Operating Range

Rating	Symbol	Value	Unit
Operating temperature range(1) MC68HC812A4PV8 MC68HC812A4CPV8	T_A	T_L to T_H 0 to +70 –40 to +85	°C
Operating voltage range	V_{DD}	5.0 ± 10%	V

1. For additional information, refer to the technical supplement document for 3.3 volt specifications (MC68C812A4) . This supplement can be found at http://freescale.com

18.3 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Average junction temperature	T_J	$T_A + (P_D \times \Theta_{JA})$	°C
Ambient temperature	T_A	User-determined	°C
Package thermal resistance (junction-to-ambient) 112-pin thin quad flat pack (TQFP)	Θ_{JA}	39	°C/W
Total power dissipation(1)	P_D	$P_{INT} + P_{I/O}$ or $\frac{K}{T_J + 273°C}$	W
Device internal power dissipation	P_{INT}	$I_{DD} \times V_{DD}$	W
I/O pin power dissipation(2)	$P_{I/O}$	User-determined	W
A constant(3)	K	$P_D \times (T_A + 273°C) + \Theta_{JA} \times P_D^2$	W/°C

1. This is an approximate value, neglecting $P_{I/O}$.
2. For most applications, $P_{I/O} \ll P_{INT}$ and can be neglected.
3. K is a constant pertaining to the device. Solve for K with a known T_A and a measured P_D (at equilibrium). Use this value of K to solve for P_D and T_J iteratively for any value of T_A.

18.4 DC Electrical Characteristics

Characteristic[1]	Symbol	Min	Max	Unit
Input high voltage, all inputs	V_{IH}	$0.7 \times V_{DD}$	$V_{DD} + 0.3$	V
Input low voltage, all inputs	V_{IL}	V_{SS}–0.3	$0.2 \times V_{DD}$	V
Output high voltage, all I/O and output pins Normal drive strength I_{OH} = –10.0 µA I_{OH} = –0.8 mA Reduced drive strength I_{OH} = –4.0 µA I_{OH} = –0.3 mA	V_{OH}	 V_{DD} – 0.2 V_{DD} – 0.8 V_{DD} – 0.2 V_{DD} – 0.8	 — — — —	V
Output low voltage, all I/O and output pins, normal drive strength I_{OL} = 10.0 µA I_{OL} = 1.6 mA EXTAL, PAD[7:0], V_{RH}, V_{RL}, V_{FP}, $\overline{XIRQ}$, reduced drive strength I_{OL} = 3.6 µA I_{OL} = 0.6 mA	V_{OL}	 — — — —	 V_{SS} +0.2 V_{SS} +0.4 V_{SS} +0.2 V_{SS} +0.4	V
Input leakage current[2] all input pins V_{In} = V_{DD} or V_{SS} — V_{RL}, V_{RH}, PAD6–PAD0 V_{In} = V_{DD} or V_{SS} — $\overline{IRQ}$ V_{In} = V_{DD} or V_{SS} — PAD7	I_{In}	 — — —	 ±1 ±10 ±10	µA
Three-state leakage, I/O ports, BKGD, and $\overline{RESET}$	I_{OZ}	—	±2.5	µA
Input capacitance All input pins and ATD pins (non-sampling) ATD pins (sampling) All I/O pins	C_{In}	 — — —	 10 15 20	pF
Output load capacitance All outputs except PS7–PS4 PS7–PS4	C_L	 — —	 90 200	pF
Active pullup, pulldown current $\overline{IRQ}$, $\overline{XIRQ}$, $\overline{DBE}$, ECLK, $\overline{LSTRB}$, R/$\overline{W}$, and BKGD Ports A, B, C, D, F, G, H, J, P, S, and T	I_{APU}	50	500	µA
RAM standby voltage, power down	V_{SB}	1.5	—	V
RAM standby current	I_{SB}	—	10	µA

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted
2. Specification is for parts in the –40 to +85°C range. Higher temperature ranges will result in increased current leakage.

18.5 Supply Current

Characteristic[1]	Symbol	8 MHz Typical	2 MHz	4 MHz	8 MHz	Unit
Maximum total supply current						
Run	I_{DD}					
Single-chip mode		30	15	25	40	mA
Expanded mode		47	25	40	65	mA
Wait, all peripheral functions shut down	WI_{DD}					
Single-chip mode		7	1.5	4	8	mA
Expanded mode		8	4	5	10	mA
Stop, single-chip mode, no clocks	SI_{DD}					
–40 °C to +85 °C		< 1	10	10	10	µA
+85 °C to +105 °C		< 10	25	25	25	µA
+105 °C to +125 °C		< 25	50	50	50	µA
Maximum power dissipation[2]						
Single-chip mode	P_D	54	62	54	62	mW
Expanded mode		76	90	76	90	mW

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted
2. Includes I_{DD} and I_{DDA}

18.6 ATD Maximum Ratings

Characteristic	Symbol	Value	Units
ATD reference voltage $V_{RH} \le V_{DDA}$ $V_{RL} \ge V_{SSA}$	 V_{RH} V_{RL}	 –0.3 to +6.5 –0.3 to +6.5	V
V_{SS} differential voltage	$\lvert V_{SS}–V_{SSA}\rvert$	0.1	V
V_{DD} differential voltage	$\lvert V_{DD}–V_{DDA}\rvert$ $V_{DD}–V_{DDX}$	6.5 6.5	V
V_{REF} differential voltage	$\lvert V_{RH}–V_{RL}\rvert$	6.5	V
Reference to supply differential voltage	$\lvert V_{RH}–V_{DDA}\rvert$ $\lvert V_{RL}–V_{SSA}\rvert$	6.5 6.5	V

18.7 ATD DC Electrical Characteristcs

Characteristic[1]	Symbol	Min	Max	Unit
Analog supply voltage	V_{DDA}	4.5	5.5	V
Analog supply current, normal operation	I_{DDA}	—	1.0	mA
Reference voltage, low	V_{RL}	V_{SSA}	$V_{DDA}/2$	V
Reference voltage, high	V_{RH}	$V_{DDA}/2$	V_{DDA}	V
V_{REF} differential reference voltage[2]	$V_{RH}–V_{RL}$	4.5	5.5	V
Input voltage[3]	V_{INDC}	V_{SSA}	V_{DDA}	V
Input current, off channel[4]	I_{OFF}	—	100	nA
Reference supply current	I_{REF}	—	250	µA
Input capacitance Not sampling Sampling	 C_{INN} C_{INS}	 — —	 10 15	pF

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, ATD clock = 2 MHz, unless otherwise noted
2. Accuracy is guaranteed at $V_{RH} - V_{RL}$ = 5.0 Vdc ± 10%.
3. To obtain full-scale, full-range results, $V_{SSA} \le V_{RL} \le V_{INDC} \le V_{RH} \le V_{DDA}$.
4. Maximum leakage occurs at maximum operating temperature. Current decreases by approximately one-half for each 10°C decrease from maximum temperature.

18.8 Analog Converter Operating Characteristics

Characteristic[1]	Symbol	Min	Typical	Max	Unit
8-bit resolution[2]	2 counts	—	24	—	mV
Differential non-linearity[3]	DNL	–0.5	—	+0.5	Count
Integral non-linearity[3]	INL	–1	—	+1	Count
Absolute error[3],[4] 2, 4, 8, and 16 ATD sample clocks	AE	–2	—	+2	Count
Maximum source impedance	R_S	—	20	See note[5]	kΩ

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, ATD clock = 2 MHz, unless otherwise noted
2. $V_{RH}–V_{RL} \ge 5.12$ V; $V_{DDA}—V_{SSA}$ = 5.12 V
3. At V_{REF} = 5.12 V, one 8-bit count = 20 mV.
4. 8-bit absolute error of 1 count (20 mV) includes 1/2 count (10 mv) inherent quantization error and 1/2 count (10 mV) circuit (differential, integral, and offset) error.
5. Maximum source impedance is application-dependent. Error resulting from pin leakage depends on junction leakage into the pin and on leakage due to charge-sharing with internal capacitance.
   Error from junction leakage is a function of external source impedance and input leakage current. Expected error in result value due to junction leakage is expressed in voltage (V_{ERRJ}):

$$V_{ERRJ} = R_S \times I_{OFF}$$

   where I_{OFF} is a function of operating temperature. Charge-sharing effects with internal capacitors are a function of ATD clock speed, the number of channels being scanned, and source impedance. For 8-bit conversions, charge pump leakage is computed as follows:

$$V_{ERRJ} = .25\ pF \times V_{DDA} \times R_S \times ATDCLK/(8 \times \text{number of channels})$$

18.9 ATD AC Operating Characteristics

Characteristic[1]	Symbol	Min	Max	Unit
ATD operating clock frequency	f_{ATDCLK}	0.5	2.0	MHz
Conversion time per channel 0.5 MHz ≤ f_{ATDCLK} ≤ 2 MHz 18 ATD clocks 32 ATD clocks	t_{CONV}	 8.0 15.0	 32.0 60.0	µs
Stop recovery time V_{DDA} = 5.0 V	t_{SR}	—	50	µs

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, ATD clock = 2 MHz, unless otherwise noted

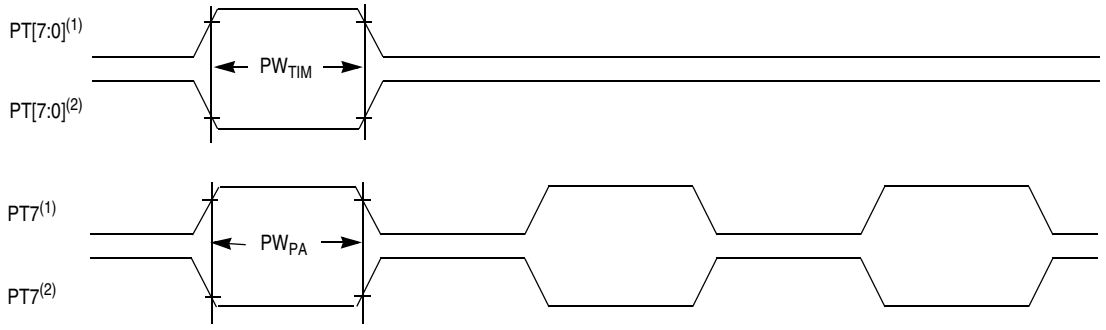
18.10 EEPROM Characteristics

Characteristic[1]	Symbol	Min	Typical	Max	Unit
Minimum programming clock frequency[2]	f_{PROG}	4.0	—	—	MHz
Programming time	t_{PROG}	10.0	—	10.5	ms
Clock recovery time following STOP, to continue programming	t_{CRSTOP}	—	—	t_{PROG}+ 1	ms
Erase time	t_{Erase}	10.0	—	10.5	ms
Write/erase endurance	—	10,000	30,000[3]	—	Cycles
Data retention	—	10	—	—	Years

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted
2. RC oscillator must be enabled if programming is desired and $f_{SYS} < f_{PROG}$.
3. If average T_H is below 85°C

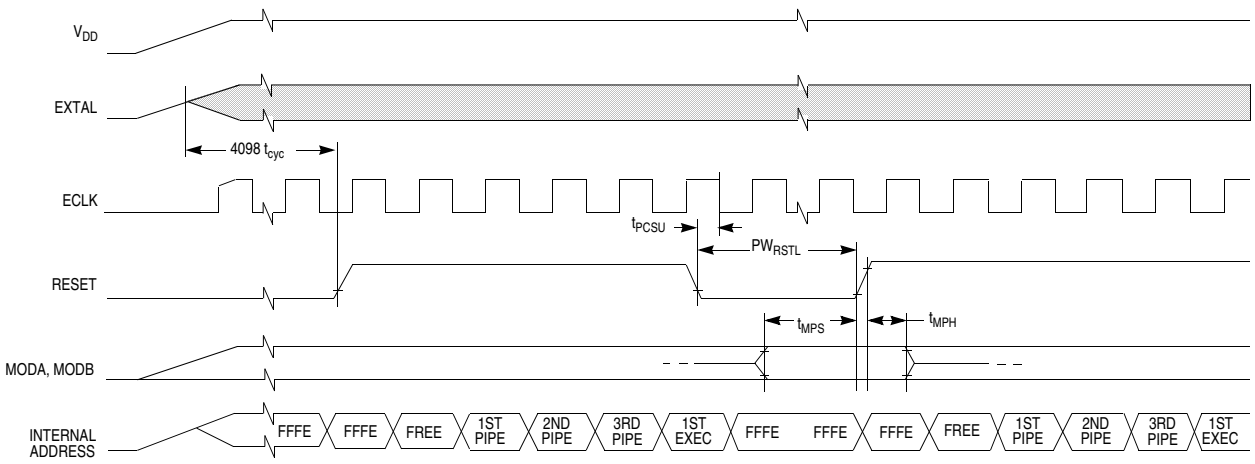
18.11 Control Timing

Characteristic	Symbol	8.0 MHz		Unit
		Min	Max	
Frequency of operation	f_o	dc	8.0	MHz
E-clock period	t_{cyc}	125	—	ns
Crystal frequency	f_{XTAL}	—	16.0	MHz
External oscillator frequency	2 f_o	dc	16.0	MHz
Processor control setup time $t_{PCSU} = t_{cyc}/2 + 30$	t_{PCSU}	82	—	ns
Reset input pulse width To guarantee external reset vector Minimum input time (can be pre-empted by internal reset)	PW_{RSTL}	 32 2	 — —	t_{cyc}
Mode programming setup time	t_{MPS}	4	—	t_{cyc}
Mode programming hold time	t_{MPH}	10	—	ns
Interrupt pulse width, $\overline{IRQ}$, edge-sensitive mode, KWU $PW_{IRQ} = 2\ t_{cyc} + 20$	PW_{IRQ}	270	—	ns
Wait recovery startup time	t_{WRS}	—	4	t_{cyc}
Timer pulse width, input capture pulse accumulator input $PW_{TIM} = 2\ t_{cyc} + 20$	PW_{TIM}	270	—	ns



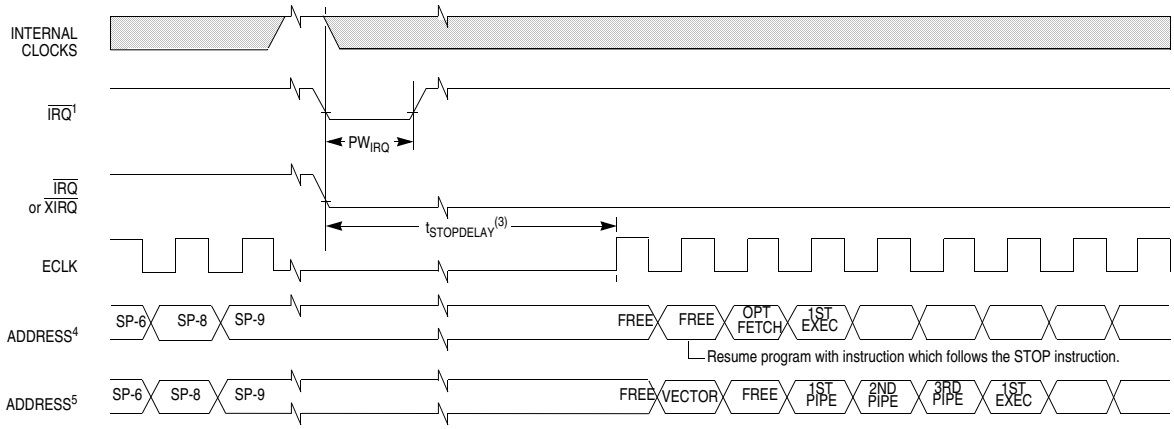
Notes:
1. Rising edge-sensitive input
2. Falling edge-sensitive input

Figure 18-1. Timer Inputs



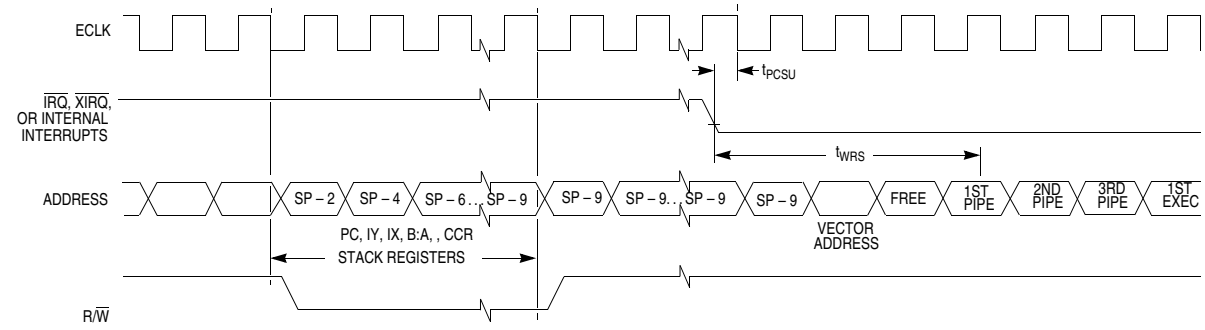
Note: Reset timing is subject to change.

Figure 18-2. POR and External Reset Timing Diagram



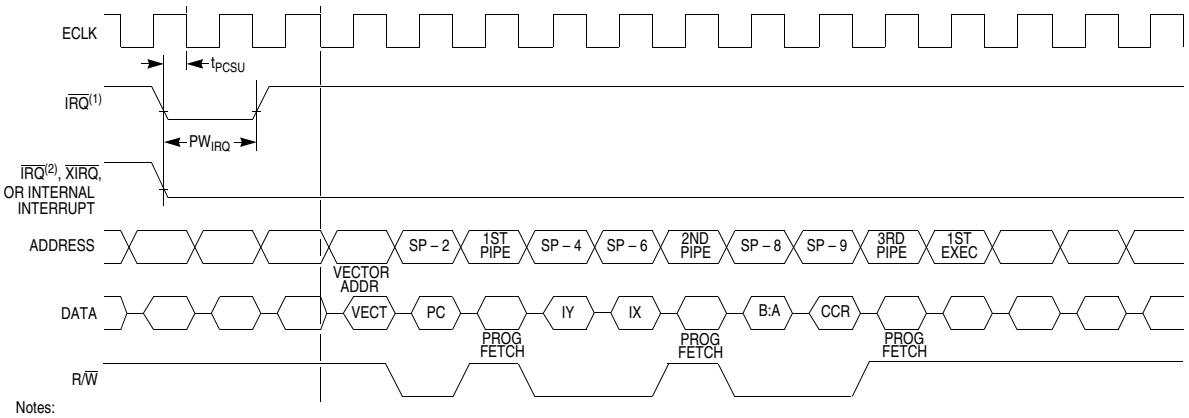
Notes:
1. Edge-sensitive $\overline{IRQ}$ pin (IRQE bit = 1)
2. Level-sensitive $\overline{IRQ}$ pin (IRQE bit = 0)
3. $t_{STOPDELAY}$ = 4098 t_{cyc} if DLY bit = 1 or 2 t_{cyc} if DLY = 0.
4. $\overline{XIRQ}$ with X bit in CCR = 1.
5. $\overline{IRQ}$ or ($\overline{XIRQ}$ with X bit in CCR = 0)

Figure 18-3. STOP Recovery Timing Diagram



Note: RESET also causes recovery from WAIT.

Figure 18-4. WAIT Recovery Timing Diagram



Notes:
1. Edge sensitive IRQ pin (IRQE bit = 1)
2. Level sensitive IRQ pin (IRQE bit = 0)

Figure 18-5. Interrupt Timing Diagram

18.12 Peripheral Port Timing

Characteristic	Symbol	8.0 MHz		Unit
		Min	Max	
Frequency of operation (E-clock frequency)	f_o	dc	8.0	MHz
E-clock period	t_{cyc}	125	—	ns
Peripheral data setup time, MCU read of ports $t_{PDSU} = t_{cyc}/2 + 30$	t_{PDSU}	102	—	ns
Peripheral data hold time, MCU read of ports	t_{PDH}	0	—	ns
Delay time, peripheral data write, MCU write to ports	t_{PWD}	—	40	ns

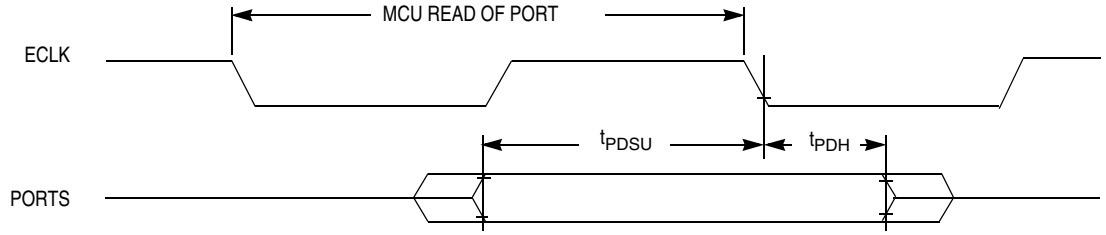


Figure 18-6. Port Read Timing Diagram

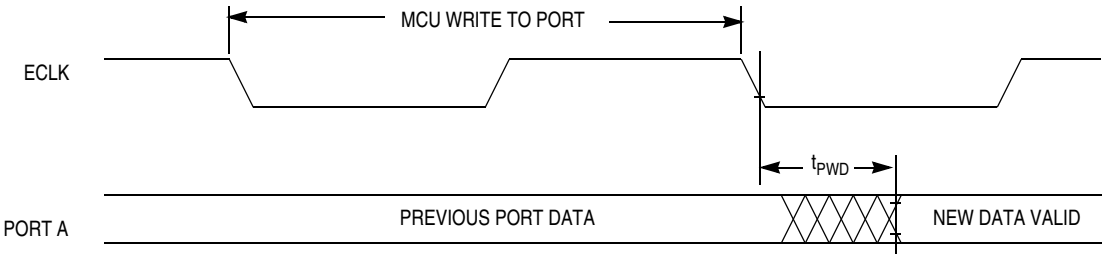
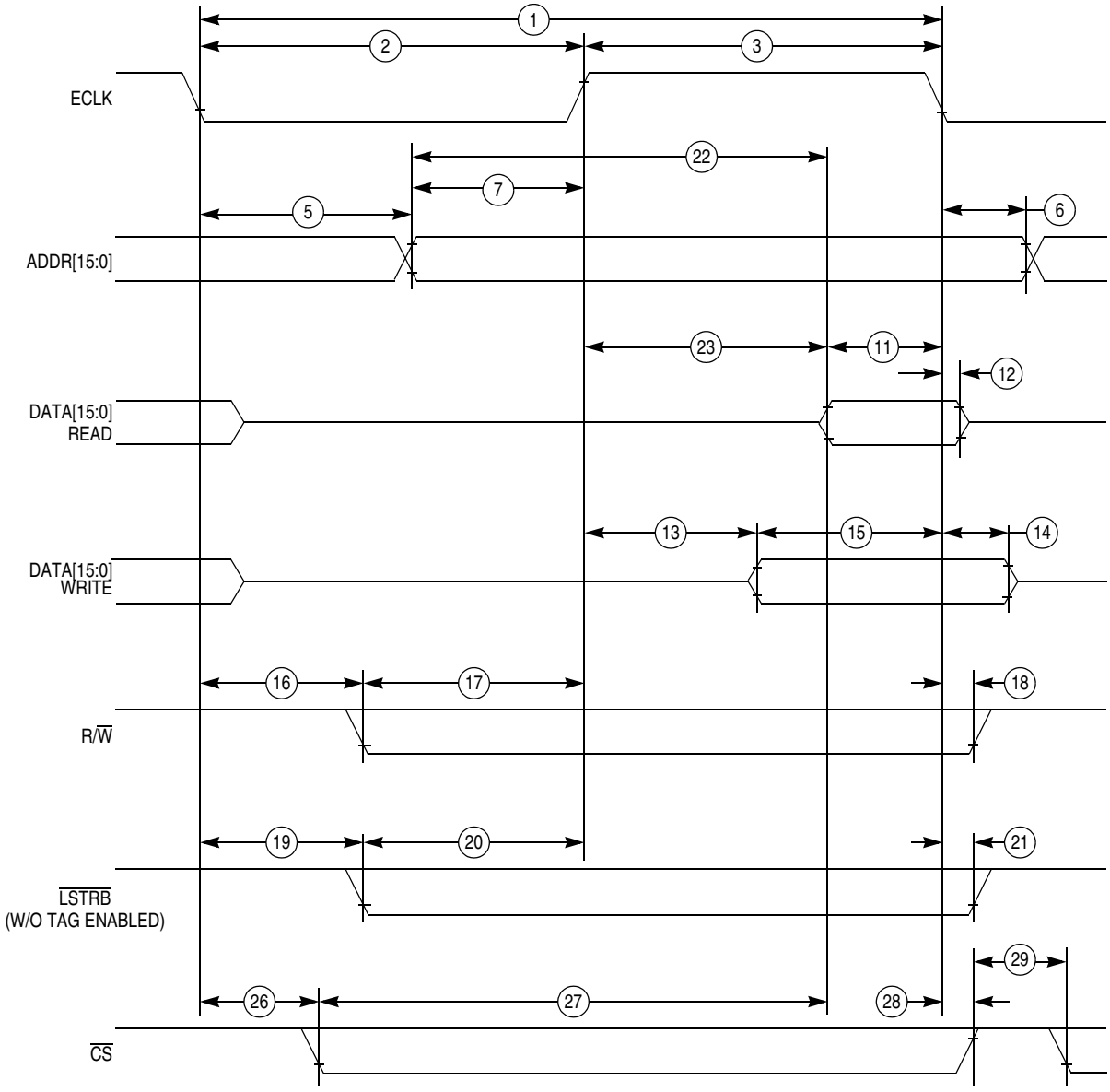


Figure 18-7. Port Write Timing Diagram

18.13 Non-Multiplexed Expansion Bus Timing

Num	Characteristic[1], [2]	Delay	Symbol	8 MHz		Unit
				Min	Max	
	Frequency of operation (E-clock frequency)	—	f_o	dc	8.0	MHz
1	Cycle time $t_{cyc} = 1/f_o$		t_{cyc}	125	—	ns
2	Pulse width, E low $PW_{EL} = t_{cyc}/2 + delay$	–2	PW_{EL}	60	—	ns
3	Pulse width, E high[3] $PW_{EH} = t_{cyc}/2 + delay$	–2	PW_{EH}	60	—	ns
5	Address delay time $t_{AD} = t_{cyc}/4 + delay$	29	t_{AD}	—	60	ns
6	Address hold time	—	t_{AH}	20	—	ns
7	Address valid time to E rise $t_{AV} = PW_{EL} - t_{AD}$	—	t_{AV}	0	—	ns
11	Read data setup time	—	t_{DSR}	30	—	ns
12	Read data hold time	—	t_{DHR}	0	—	ns
13	Write data delay time[4] $t_{DDW} = t_{cyc}/4 + delay$	25	t_{DDW}	—	46	ns
14	Write data hold time	—	t_{DHW}	20	—	ns
15	Write data setup time[3] $t_{DSW} = PW_{EH} - t_{DDW}$	—	t_{DSW}	30	—	ns
16	Read/write delay tim $wt_{RWD} = t_{cyc}/4 + delay$	18	t_{RWD}	—	49	ns
17	Read/write valid time to E rise $t_{RWV} = PW_{EL} - t_{RWD}$	—	t_{RWV}	20	—	ns
18	Read/write hold time	—	t_{RWH}	20	—	ns
19	Low strobe delay time $t_{LSD} = t_{cyc}/4 + delay$	18	t_{LSD}	—	49	ns
20	Low strobe valid time to E rise $t_{LSV} = PW_{EL} - t_{LSD}$	—	t_{LSV}	11	—	ns
21	Low strobe hold time	—	t_{LSH}	20	—	ns
22	Address access time[3] $t_{ACCA} = t_{cyc} - t_{AD} - t_{DSR}$	—	t_{ACCA}	—	35	ns
23	Access time from E rise[3] $t_{ACCE} = PW_{EH} - t_{DSR}$	—	t_{ACCE}	—	30	ns
26	Chip-select delay time $t_{CSD} = t_{cyc}/4 + delay$	29	t_{CSD}	—	60	ns
27	Chip-select access time[3] $t_{ACCS} = t_{cyc} - t_{CSD} - t_{DSR}$	—	t_{ACCS}	—	65	ns
28	Chip-select hold time	—	t_{CSH}	0	10	ns
29	Chip-select negated time $t_{CSN} = t_{cyc}/4 + delay$	5	t_{CSN}	36	—	ns

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted
2. All timings are calculated for normal port drives.
3. This characteristic is affected by clock stretch.
   Add N × t_{cyc} where N = 0, 1, 2, or 3, depending on the number of clock stretches.
4. Equation still under evaluation



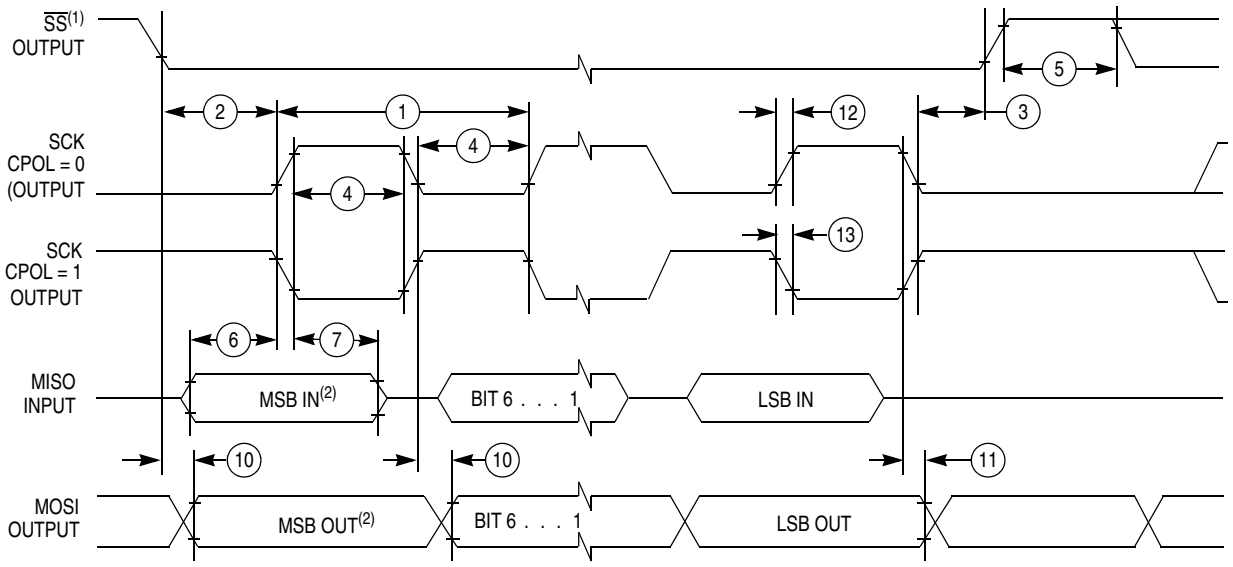
Note: Measurement points shown are 20% and 70% of V_{DD}.

Figure 18-8. Non-Multiplexed Expansion Bus Timing Diagram

18.14 SPI Timing

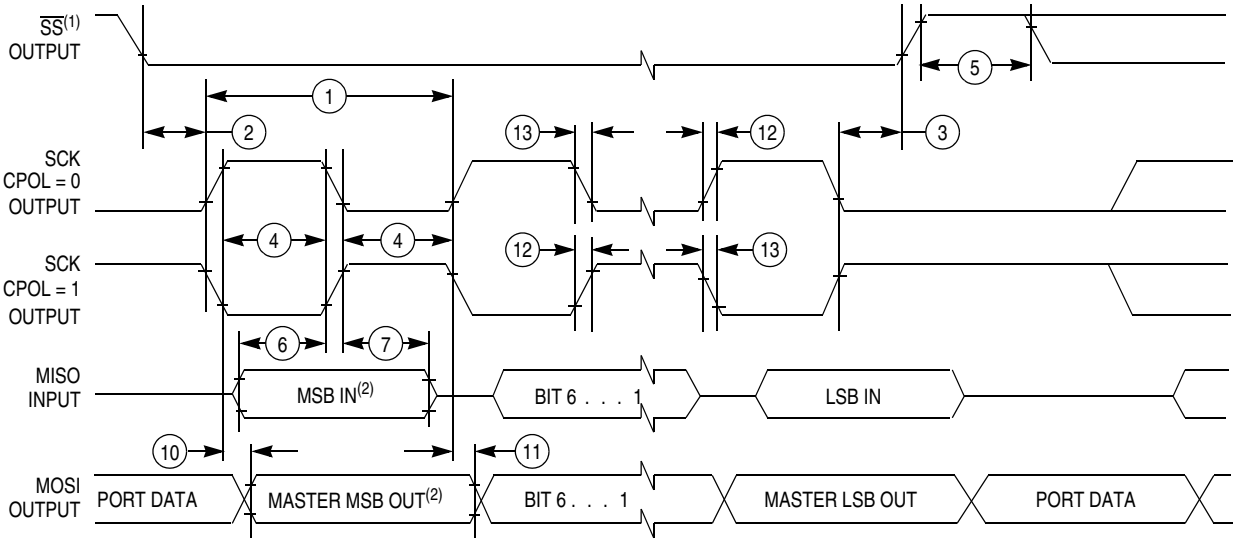
Num	Function[1], [2]	Symbol	Min	Max	Unit
	Operating frequency Master Slave	f_{op}	 dc dc	 1/2 1/2	E-clock frequency
1	SCK period Master Slave	t_{sck}	 2 2	 256 —	t_{cyc}
2	Enable lead time Master Slave	t_{Lead}	 1/2 1	 — —	 t_{sck} t_{cyc}
3	Enable lag time Master Slave	t_{Lag}	 1/2 1	 — —	 t_{sck} t_{cyc}
4	Clock (SCK) high or low time Master Slave	t_{wsck}	 t_{cyc} – 60 t_{cyc} – 30	 128 t_{cyc} —	ns
5	Sequential transfer delay Master Slave	t_{td}	 1/2 1	 — —	 t_{sck} t_{cyc}
6	Data setup time (inputs) Master Slave	t_{su}	 30 30	 — —	ns
7	Data hold time (inputs) Master Slave	t_{hi}	 0 30	 — —	ns
8	Slave access time	t_a	—	1	t_{cyc}
9	Slave MISO disable time	t_{dis}	—	1	t_{cyc}
10	Data valid (after SCK edge) Master Slave	t_v	 — —	 50 50	ns
11	Data hold time (outputs) Master Slave	t_{ho}	 0 0	 — —	ns
12	Rise time Input Output	 t_{ri} t_{ro}	 — —	 t_{cyc} – 30 30	 ns ns
13	Fall time Input Output	 t_{fi} t_{fo}	 — —	 t_{cyc} – 30 30	 ns ns

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, 200 pF load on all SPI pins
2. All ac timing is shown with respect to 20% V_{DD} and 70% V_{DD} levels, unless otherwise noted.



Notes:
1. $\overline{SS}$ output mode (DDS7 = 1, SSOE = 1)
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB

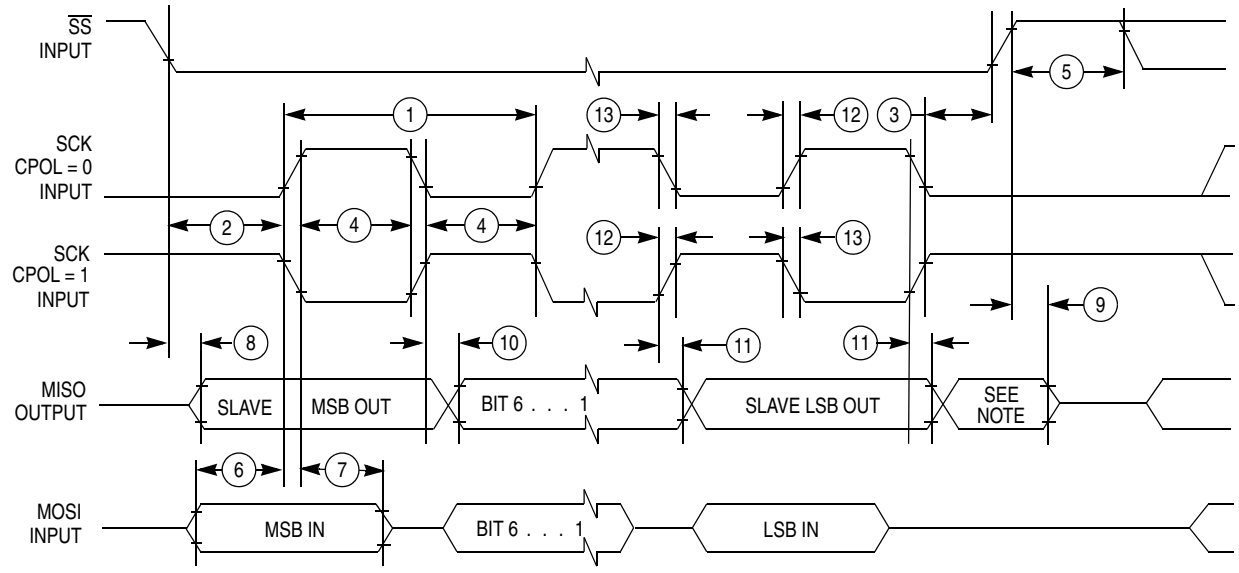
A) SPI Master Timing (CPHA = 0)



Notes:
1. $\overline{SS}$ output mode (DDS7 = 1, SSOE = 1)
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB

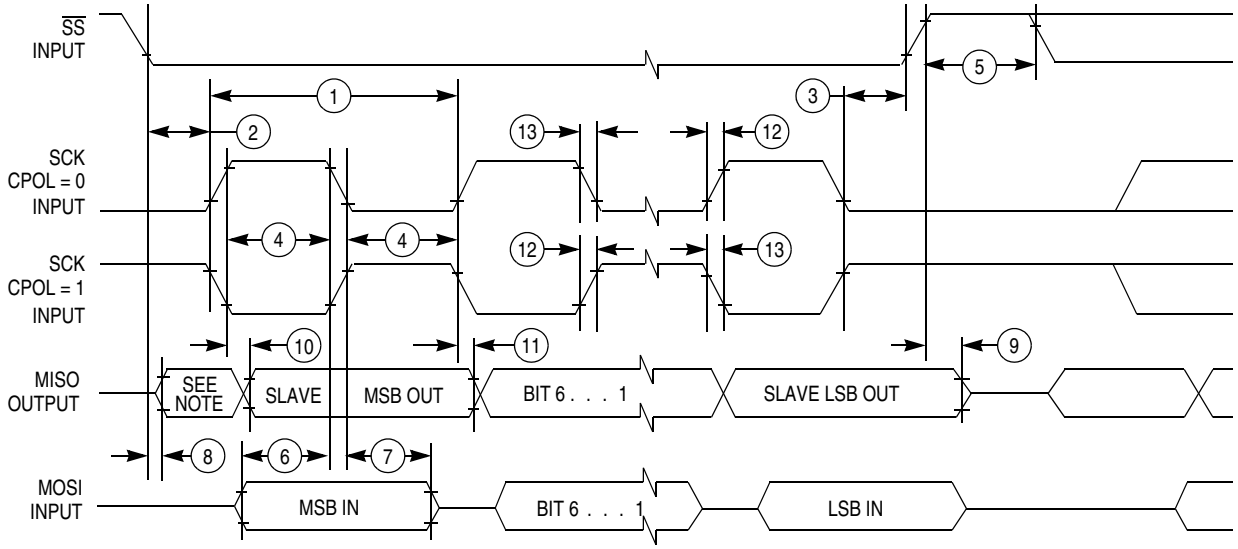
B) SPI Master Timing (CPHA = 1)

Figure 18-9. SPI Timing Diagram (Sheet 1 of 2)



Note: Not defined but normally MSB of character just received

A) SPI Slave Timing (CPHA = 0)



Note: Not defined but normally LSB of character just received

B) SPI Slave Timing (CPHA = 1)

Figure 18-9. SPI Timing Diagram (Sheet 2 of 2)

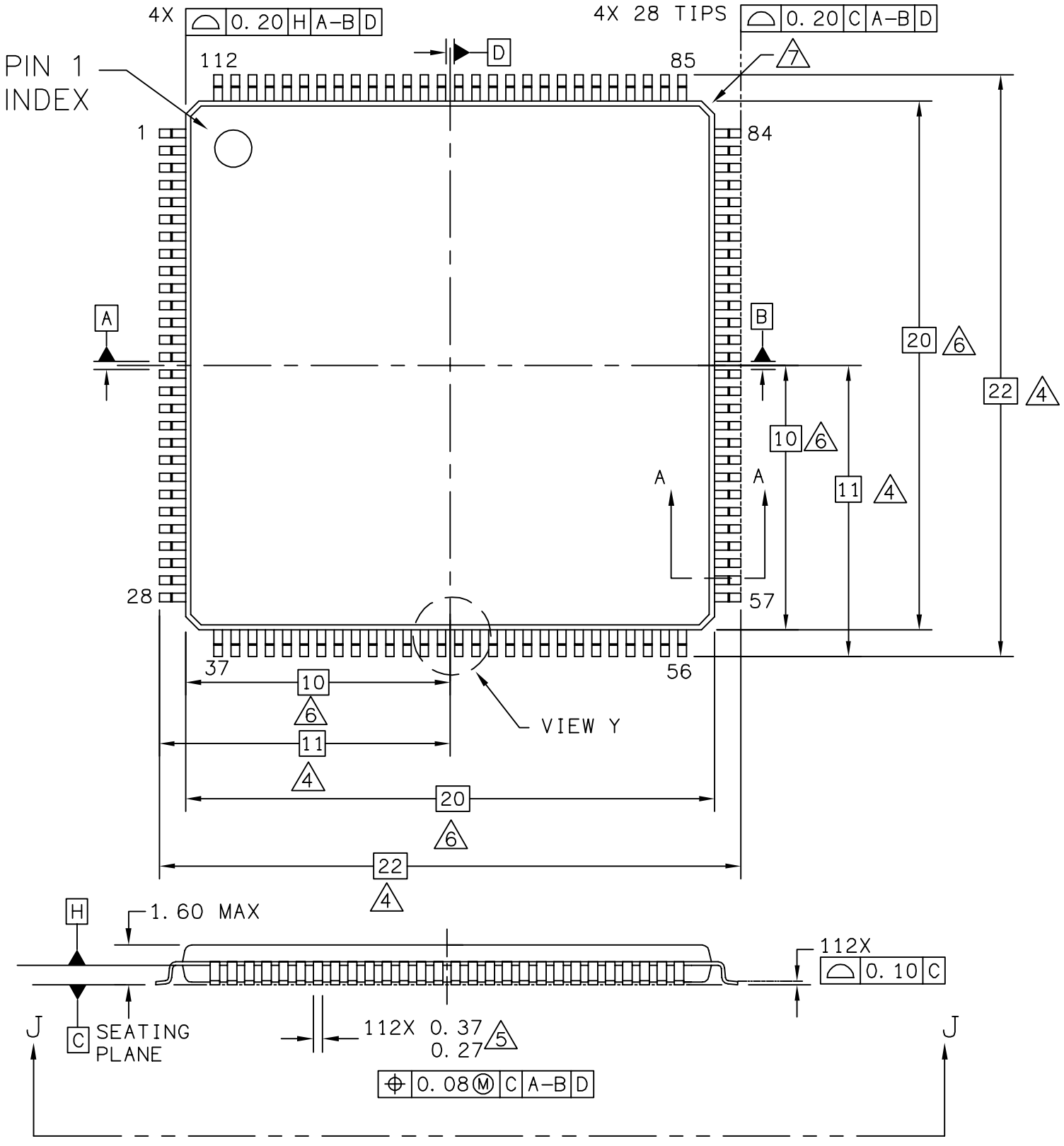
Chapter 19
Mechanical Specifications

19.1 Introduction

This section provides dimensions for the 112-lead low-profile quad flat pack (LQFP).

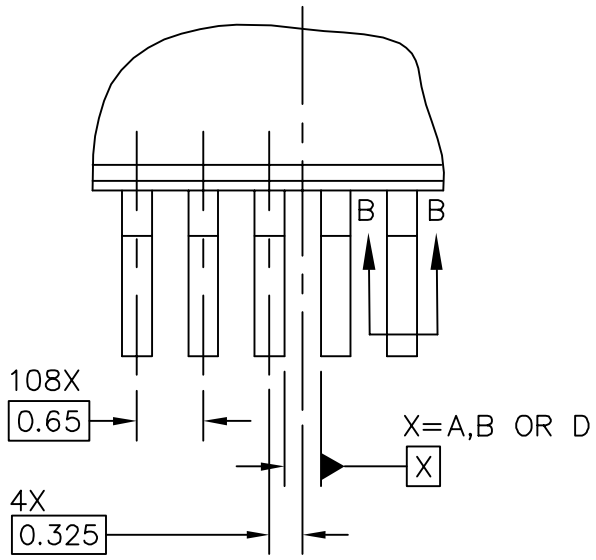
19.2 Package Dimensions

Refer to the following pages for detailed package dimensions.

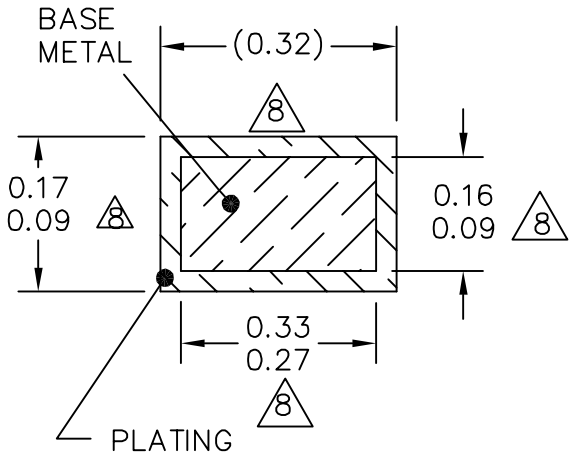


MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

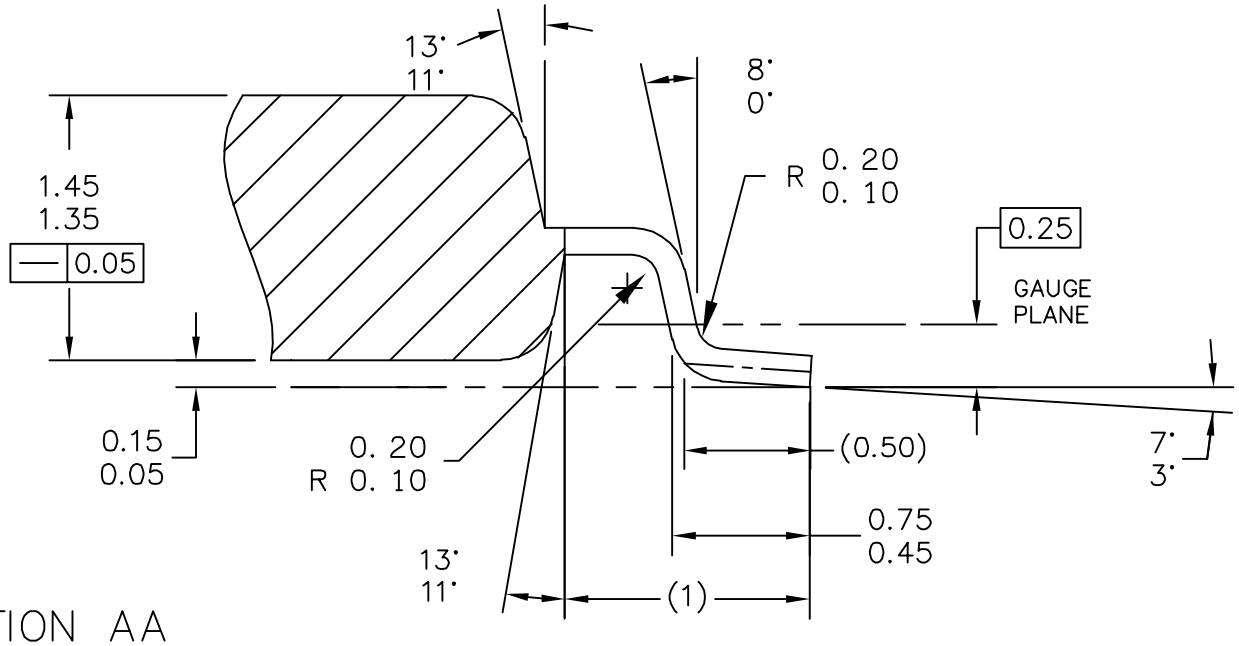
TITLE: 112LD LQFP 20 X 20 X 1.4 0.65 PITCH	DOCUMENT NO: 98ASS23330W	REV: E
	CASE NUMBER: 987-02	25 MAY 2005
	STANDARD: JEDEC MS-026 BFA	



VIEW Y



SECTION B–B



SECTION AA

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE

TITLE: 112LD LQFP 20 X 20 X 1.4 0.65 PITCH	DOCUMENT NO: 98ASS23330W	REV: E
	CASE NUMBER: 987-02	25 MAY 2005
	STANDARD: JEDEC MS-026 BFA	

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.
2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
3. DATUMS A, B AND D TO BE DETERMINED AT DATUM PLANE H.
4. DIMENSIONS TO BE DETERMINED AT SEATING PLANE C.
5. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE UPPER LIMIT BY MORE THAN 0.08 MM. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07 MM.
6. THIS DIMENSION DOES NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.254 MM PER SIDE. THIS DIMENSION IS MAXIMUM PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH.
7. EXACT SHAPE OF EACH CORNER IS OPTIONAL.
8. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.10 MM AND 0.25 MM FROM THE LEAD TIP.

MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE: 112LD LQFP, 20 X 20 X 1.4 PKG, 0.65 PITCH	DOCUMENT NO: 98ASS23330W	REV: E
	CASE NUMBER: 987–02	25 MAY 2005
	STANDARD: JEDEC MS–026 BFA	

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to http://www.freescale.com/epp.

MC68HC812A4
Rev. 7, 05/2006

freescale™ semiconductor