THE DATASHEET OF

MC68HC908JL8CFAE

www.win-source.net 0086-755-83957316

MC68HC908JL8
MC68HC908JK8
MC68HC908KL8
MC68HC08JL8
MC68HC08JK8

Data Sheet

M68HC08
Microcontrollers

MC68HC908JL8
Rev. 3.1
3/2005

freescale.com

freescale™
semiconductor

MC68HC908JL8
MC68HC908JK8
MC68HC908KL8
MC68HC08JL8
MC68HC08JK8

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://www.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.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc.
This product incorporates SuperFlash® technology licensed from SST.

© Freescale Semiconductor, Inc., 2005. All rights reserved.

Revision History

Date	Revision Level	Description	Page Number(s)
Mar 2005	3.1	Added IRQ timing to Table 17-5 . Control Timing (5V) and Table 17-8 . Control Timing (3V)	188, 190
Nov 2004	3	Chapter 9 Serial Communications Interface (SCI) — Corrected SCI module clock source from OSCCLK to Bus Clock throughout.	121–206
		Figure 13-2 . Keyboard Interrupt Block Diagram — Removed incorrect Schmitt trigger in block diagram.	168
		14.7.2 Stop Mode — STOP_ICLKDIS bit does not affect stop mode conditions for COP. Replaced section with new text.	176
		Added Appendix A MC68HC08JL8 — ROM parts.	201
		Added Appendix B MC68HC908KL8.	207
Nov 2002	2	First general release.	—

List of Chapters

Chapter 1 General Description 17
Chapter 2 Memory 25
Chapter 3 Configuration and Mask Option Registers (CONFIG & MOR) 41
Chapter 4 Central Processor Unit (CPU) 45
Chapter 5 System Integration Module (SIM) 61
Chapter 6 Oscillator (OSC) 79
Chapter 7 Monitor ROM (MON) 85
Chapter 8 Timer Interface Module (TIM) 105
Chapter 9 Serial Communications Interface (SCI) 121
Chapter 10 Analog-to-Digital Converter (ADC) 145
Chapter 11 Input/Output (I/O) Ports 151
Chapter 12 External Interrupt (IRQ) 163
Chapter 13 Keyboard Interrupt Module (KBI) 167
Chapter 14 Computer Operating Properly (COP) 173
Chapter 15 Low Voltage Inhibit (LVI) 177
Chapter 16 Break Module (BREAK) 179
Chapter 17 Electrical Specifications 185
Chapter 18 Mechanical Specifications 195
Chapter 19 Ordering Information 199
Appendix A MC68HC08JL8 201
Appendix B MC68HC908KL8 207

Table of Contents

Chapter 1
General Description

1.1 Introduction . . . 17
1.2 Features . . . 17
1.3 MCU Block Diagram . . . 18
1.4 Pin Assignments . . . 20
1.5 Pin Functions . . . 21

Chapter 2
Memory

2.1 Introduction . . . 25
2.2 I/O Section . . . 25
2.3 Monitor ROM . . . 25
2.4 Random-Access Memory (RAM) . . . 33
2.5 FLASH Memory . . . 33
2.6 Functional Description . . . 33
2.7 FLASH Control Register . . . 34
2.8 FLASH Page Erase Operation . . . 34
2.9 FLASH Mass Erase Operation . . . 35
2.10 FLASH Program Operation . . . 35
2.11 FLASH Block Protection . . . 38
2.12 FLASH Block Protect Register . . . 38

Chapter 3
Configuration and Mask Option Registers (CONFIG & MOR)

3.1 Introduction . . . 41
3.2 Functional Description . . . 41
3.3 Configuration Register 1 (CONFIG1) . . . 42
3.4 Configuration Register 2 (CONFIG2) . . . 43
3.5 Mask Option Register (MOR) . . . 43

Chapter 4
Central Processor Unit (CPU)

4.1 Introduction . . . 45
4.2 Features . . . 45
4.3 CPU Registers . . . 46
4.3.1 Accumulator . . . 46
4.3.2 Index Register . . . 46
4.3.3 Stack Pointer . . . 47

4.3.4 Program Counter . . . 47
4.3.5 Condition Code Register . . . 48
4.4 Arithmetic/Logic Unit (ALU) . . . 49
4.5 Low-Power Modes . . . 49
4.5.1 Wait Mode . . . 49
4.5.2 Stop Mode . . . 49
4.6 CPU During Break Interrupts . . . 50
4.7 Instruction Set Summary . . . 50
4.8 Opcode Map . . . 50

Chapter 5
System Integration Module (SIM)

5.1 Introduction . . . 61
5.2 SIM Bus Clock Control and Generation . . . 63
5.2.1 Bus Timing . . . 63
5.2.2 Clock Start-Up from POR or LVI Reset . . . 63
5.2.3 Clocks in Stop Mode and Wait Mode . . . 63
5.3 Reset and System Initialization . . . 64
5.3.1 External Pin Reset . . . 64
5.3.2 Active Resets from Internal Sources . . . 64
5.3.2.1 Power-On Reset . . . 65
5.3.2.2 Computer Operating Properly (COP) Reset . . . 66
5.3.2.3 Illegal Opcode Reset . . . 66
5.3.2.4 Illegal Address Reset . . . 66
5.3.2.5 Low-Voltage Inhibit (LVI) Reset . . . 66
5.4 SIM Counter . . . 67
5.4.1 SIM Counter During Power-On Reset . . . 67
5.4.2 SIM Counter During Stop Mode Recovery . . . 67
5.4.3 SIM Counter and Reset States . . . 67
5.5 Exception Control . . . 67
5.5.1 Interrupts . . . 67
5.5.1.1 Hardware Interrupts . . . 69
5.5.1.2 SWI Instruction . . . 70
5.5.2 Interrupt Status Registers . . . 70
5.5.2.1 Interrupt Status Register 1 . . . 71
5.5.2.2 Interrupt Status Register 2 . . . 72
5.5.2.3 Interrupt Status Register 3 . . . 72
5.5.3 Reset . . . 72
5.5.4 Break Interrupts . . . 72
5.5.5 Status Flag Protection in Break Mode . . . 73
5.6 Low-Power Modes . . . 73
5.6.1 Wait Mode . . . 73
5.6.2 Stop Mode . . . 74
5.7 SIM Registers . . . 75
5.7.1 Break Status Register (BSR) . . . 75
5.7.2 Reset Status Register (RSR) . . . 76
5.7.3 Break Flag Control Register (BFCR) . . . 77

Chapter 6
Oscillator (OSC)

6.1 Introduction . . . 79
6.2 Oscillator Selection . . . 79
6.2.1 XTAL Oscillator . . . 80
6.2.2 RC Oscillator . . . 81
6.3 Internal Oscillator . . . 81
6.4 I/O Signals . . . 82
6.4.1 Crystal Amplifier Input Pin (OSC1) . . . 82
6.4.2 Crystal Amplifier Output Pin (OSC2/RCCLK/PTA6/KBI6) . . . 82
6.4.3 Oscillator Enable Signal (SIMOSCEN) . . . 82
6.4.4 XTAL Oscillator Clock (XTALCLK) . . . 82
6.4.5 RC Oscillator Clock (RCCLK) . . . 82
6.4.6 Oscillator Out 2 (2OSCOUT) . . . 82
6.4.7 Oscillator Out (OSCOUT) . . . 83
6.4.8 Internal Oscillator Clock (ICLK) . . . 83
6.5 Low Power Modes . . . 83
6.5.1 Wait Mode . . . 83
6.5.2 Stop Mode . . . 83
6.6 Oscillator During Break Mode . . . 83

Chapter 7
Monitor ROM (MON)

7.1 Introduction . . . 85
7.2 Features . . . 85
7.3 Functional Description . . . 85
7.3.1 Entering Monitor Mode . . . 87
7.3.2 Baud Rate . . . 89
7.3.3 Data Format . . . 89
7.3.4 Echoing . . . 90
7.3.5 Break Signal . . . 90
7.3.6 Commands . . . 90
7.4 Security . . . 93
7.5 ROM-Resident Routines . . . 94
7.5.1 PRGRNGE . . . 96
7.5.2 ERARNGE . . . 97
7.5.3 LDRNGE . . . 98
7.5.4 MON_PRGRNGE . . . 99
7.5.5 MON_ERARNGE . . . 99
7.5.6 MON_LDRNGE . . . 100
7.5.7 EE_WRITE . . . 100
7.5.8 EE_READ . . . 103

Chapter 8
Timer Interface Module (TIM)

8.1 Introduction . . . 105
8.2 Features . . . 105
8.3 Pin Name Conventions . . . 105
8.4 Functional Description . . . 106
8.4.1 TIM Counter Prescaler . . . 108
8.4.2 Input Capture . . . 108
8.4.3 Output Compare . . . 108
8.4.3.1 Unbuffered Output Compare . . . 109
8.4.3.2 Buffered Output Compare . . . 109
8.4.4 Pulse Width Modulation (PWM) . . . 109
8.4.4.1 Unbuffered PWM Signal Generation . . . 110
8.4.4.2 Buffered PWM Signal Generation . . . 111
8.4.4.3 PWM Initialization . . . 111
8.5 Interrupts . . . 112
8.6 Low-Power Modes . . . 112
8.6.1 Wait Mode . . . 112
8.6.2 Stop Mode . . . 112
8.7 TIM During Break Interrupts . . . 113
8.8 I/O Signals . . . 113
8.8.1 TIM Clock Pin (ADC12/T2CLK) . . . 113
8.8.2 TIM Channel I/O Pins (PTD4/T1CH0, PTD5/T1CH1, PTE0/T2CH0, PTE1/T2CH1) . . . 113
8.9 I/O Registers . . . 114
8.9.1 TIM Status and Control Register . . . 114
8.9.2 TIM Counter Registers . . . 115
8.9.3 TIM Counter Modulo Registers . . . 116
8.9.4 TIM Channel Status and Control Registers . . . 117
8.9.5 TIM Channel Registers . . . 119

Chapter 9
Serial Communications Interface (SCI)

9.1 Introduction . . . 121
9.2 Features . . . 121
9.3 Pin Name Conventions . . . 121
9.4 Functional Description . . . 123
9.4.1 Data Format . . . 123
9.4.2 Transmitter . . . 124
9.4.2.1 Character Length . . . 124
9.4.2.2 Character Transmission . . . 125
9.4.2.3 Break Characters . . . 125
9.4.2.4 Idle Characters . . . 125
9.4.2.5 Inversion of Transmitted Output . . . 126
9.4.2.6 Transmitter Interrupts . . . 126
9.4.3 Receiver . . . 126
9.4.3.1 Character Length . . . 126

9.4.3.2 Character Reception . . . 126
9.4.3.3 Data Sampling . . . 128
9.4.3.4 Framing Errors . . . 129
9.4.3.5 Baud Rate Tolerance . . . 130
9.4.3.6 Receiver Wakeup . . . 131
9.4.3.7 Receiver Interrupts . . . 132
9.4.3.8 Error Interrupts . . . 132
9.5 Low-Power Modes . . . 133
9.5.1 Wait Mode . . . 133
9.5.2 Stop Mode . . . 133
9.6 SCI During Break Module Interrupts . . . 133
9.7 I/O Signals . . . 133
9.7.1 TxD (Transmit Data) . . . 133
9.7.2 RxD (Receive Data) . . . 133
9.8 I/O Registers . . . 134
9.8.1 SCI Control Register 1 . . . 134
9.8.2 SCI Control Register 2 . . . 136
9.8.3 SCI Control Register 3 . . . 138
9.8.4 SCI Status Register 1 . . . 139
9.8.5 SCI Status Register 2 . . . 142
9.8.6 SCI Data Register . . . 142
9.8.7 SCI Baud Rate Register . . . 143

Chapter 10
Analog-to-Digital Converter (ADC)

10.1 Introduction . . . 145
10.2 Features . . . 145
10.3 Functional Description . . . 145
10.3.1 ADC Port I/O Pins . . . 146
10.3.2 Voltage Conversion . . . 147
10.3.3 Conversion Time . . . 147
10.3.4 Continuous Conversion . . . 147
10.3.5 Accuracy and Precision . . . 147
10.4 Interrupts . . . 147
10.5 Low-Power Modes . . . 147
10.5.1 Wait Mode . . . 147
10.5.2 Stop Mode . . . 148
10.6 I/O Signals . . . 148
10.6.1 ADC Voltage In (ADCVIN) . . . 148
10.7 I/O Registers . . . 148
10.7.1 ADC Status and Control Register . . . 148
10.7.2 ADC Data Register . . . 150
10.7.3 ADC Input Clock Register . . . 150

Chapter 11
Input/Output (I/O) Ports

11.1 Introduction . . . 151
11.2 Port A . . . 153
11.2.1 Port A Data Register (PTA) . . . 153
11.2.2 Data Direction Register A (DDRA) . . . 153
11.2.3 Port A Input Pull-Up Enable Registers . . . 155
11.3 Port B . . . 156
11.3.1 Port B Data Register (PTB) . . . 156
11.3.2 Data Direction Register B (DDRB) . . . 156
11.4 Port D . . . 157
11.4.1 Port D Data Register (PTD) . . . 158
11.4.2 Data Direction Register D (DDRD) . . . 158
11.4.3 Port D Control Register (PDCR) . . . 160
11.5 Port E . . . 160
11.5.1 Port E Data Register (PTE) . . . 160
11.5.2 Data Direction Register E (DDRE) . . . 161

Chapter 12
External Interrupt (IRQ)

12.1 Introduction . . . 163
12.2 Features . . . 163
12.3 Functional Description . . . 163
12.3.1 $\overline{\text{IRQ}}$ Pin . . . 164
12.4 IRQ Module During Break Interrupts . . . 165
12.5 IRQ Status and Control Register (INTSCR) . . . 166

Chapter 13
Keyboard Interrupt Module (KBI)

13.1 Introduction . . . 167
13.2 Features . . . 167
13.3 I/O Pins . . . 167
13.4 Functional Description . . . 168
13.4.1 Keyboard Initialization . . . 169
13.5 Keyboard Interrupt Registers . . . 169
13.5.1 Keyboard Status and Control Register . . . 169
13.5.2 Keyboard Interrupt Enable Register . . . 170
13.6 Low-Power Modes . . . 171
13.6.1 Wait Mode . . . 171
13.6.2 Stop Mode . . . 171
13.7 Keyboard Module During Break Interrupts . . . 171

Chapter 14
Computer Operating Properly (COP)

14.1 Introduction . . . 173
14.2 Functional Description . . . 173

14.3 I/O Signals 174
14.3.1 ICLK 174
14.3.2 COPCTL Write 174
14.3.3 Power-On Reset 174
14.3.4 Internal Reset 174
14.3.5 Reset Vector Fetch 174
14.3.6 COPD (COP Disable) 174
14.3.7 COPRS (COP Rate Select) 175
14.4 COP Control Register 175
14.5 Interrupts 175
14.6 Monitor Mode 175
14.7 Low-Power Modes 175
14.7.1 Wait Mode 176
14.7.2 Stop Mode 176
14.8 COP Module During Break Mode 176

Chapter 15
Low Voltage Inhibit (LVI)

15.1 Introduction 177
15.2 Features 177
15.3 Functional Description 177
15.4 LVI Control Register (CONFIG2/CONFIG1) 178
15.5 Low-Power Modes 178
15.5.1 Wait Mode 178
15.5.2 Stop Mode 178

Chapter 16
Break Module (BREAK)

16.1 Introduction 179
16.2 Features 179
16.3 Functional Description 179
16.3.1 Flag Protection During Break Interrupts 180
16.3.2 CPU During Break Interrupts 180
16.3.3 TIM During Break Interrupts 180
16.3.4 COP During Break Interrupts 180
16.4 Break Module Registers 181
16.4.1 Break Status and Control Register (BRKSCR) 181
16.4.2 Break Address Registers 181
16.4.3 Break Status Register 182
16.4.4 Break Flag Control Register (BFCR) 183
16.5 Low-Power Modes 183
16.5.1 Wait Mode 183
16.5.2 Stop Mode 183

Chapter 17
Electrical Specifications

17.1 Introduction . . . 185
17.2 Absolute Maximum Ratings . . . 185
17.3 Functional Operating Range . . . 186
17.4 Thermal Characteristics . . . 186
17.5 5V DC Electrical Characteristics . . . 187
17.6 5V Control Timing . . . 188
17.7 5V Oscillator Characteristics . . . 188
17.8 3V DC Electrical Characteristics . . . 189
17.9 3V Control Timing . . . 190
17.10 3V Oscillator Characteristics . . . 191
17.11 Typical Supply Currents . . . 192
17.12 Timer Interface Module Characteristics . . . 193
17.13 ADC Characteristics . . . 193
17.14 Memory Characteristics . . . 194

Chapter 18
Mechanical Specifications

18.1 Introduction . . . 195
18.2 20-Pin Plastic Dual In-Line Package (PDIP) . . . 195
18.3 20-Pin Small Outline Integrated Circuit Package (SOIC) . . . 196
18.4 28-Pin Plastic Dual In-Line Package (PDIP) . . . 196
18.5 28-Pin Small Outline Integrated Circuit Package (SOIC) . . . 197
18.6 32-Pin Shrink Dual In-Line Package (SDIP) . . . 197
18.7 32-Pin Low-Profile Quad Flat Pack (LQFP) . . . 198

Chapter 19
Ordering Information

19.1 Introduction . . . 199
19.2 MC Order Numbers . . . 199

Appendix A
MC68HC08JL8

A.1 Introduction . . . 201
A.2 MCU Block Diagram . . . 201
A.3 Memory Map . . . 201
A.4 Reserved Registers . . . 202
A.5 Mask Option Register . . . 204
A.6 Monitor ROM . . . 204
A.7 Electrical Specifications . . . 204
A.7.1 DC Electrical Characteristics . . . 204
A.8 Memory Characteristics . . . 205
A.9 MC68HC08JL8 Order Numbers . . . 206

Appendix B
MC68HC908KL8

B.1 Introduction 207
B.2 MCU Block Diagram 207
B.3 Pin Assignments 207
B.4 Reserved Registers 210
B.5 Reserved Vectors 210
B.6 MC68HC908KL8 Order Numbers 210

Chapter 1
General Description

1.1 Introduction

The MC68HC908JL8 is a member of the low-cost, high-performance M68HC08 Family of 8-bit microcontroller units (MCUs). All MCUs in the family use the enhanced M68HC08 central processor unit (CPU08) and are available with a variety of modules, memory sizes and types, and package types.

Table 1-1. Summary of Devices

Generic Part	Description	Pin Count
MC68HC908JL8	FLASH part	28 or 32
MC68HC908JK8	FLASH part	20
MC68HC08JL8	ROM part for MC68HC908JL8	28 or 32
MC68HC08JK8	ROM part for MC68HC908JK8	20
MC68HC908KL8	ADC-less MC68HC908JL8	28 or 32

1.2 Features

Features of the MC68HC908JL8 include the following:

- High-performance M68HC08 architecture
- Fully upward-compatible object code with M6805, M146805, and M68HC05 Families
- Low-power design; fully static with stop and wait modes
- Maximum internal bus frequency:
  - 8-MHz at 5V operating voltage
  - 4-MHz at 3V operating voltage
- Oscillator options:
  - Crystal or resonator
  - RC oscillator
- 8,192 bytes user program FLASH memory with security[1] feature
- 256 bytes of on-chip RAM
- Two 16-bit, 2-channel timer interface modules (TIM1 and TIM2) with selectable input capture, output compare, and PWM capability on each channel; external clock input option on TIM2
- 13-channel, 8-bit analog-to-digital converter (ADC)
- Serial communications interface module (SCI)
- 26 general-purpose input/output (I/O) ports:
  - 8 keyboard interrupt with internal pull-up

1. No security feature is absolutely secure. However, Motorola's strategy is to make reading or copying the FLASH difficult for unauthorized users.

  - 11 LED drivers (sink)
  - 2 × 25mA open-drain I/O with pull-up
- Resident routines for in-circuit programming and EEPROM emulation
- System protection features:
  - Optional computer operating properly (COP) reset, driven by internal RC oscillator
  - Optional low-voltage detection with reset and selectable trip points for 3V and 5V operation
  - Illegal opcode detection with reset
  - Illegal address detection with reset
- Master reset pin with internal pull-up and power-on reset
- $\overline{\text{IRQ}}$ with schmitt-trigger input and programmable pull-up
- 20-pin dual in-line package (PDIP), 20-pin small outline integrated package (SOIC), 28-pin PDIP, 28-pin SOIC, 32-pin shrink dual in-line package (SDIP), and 32-pin low-profile quad flat pack (LQFP)
- Specific features of the MC68HC908JL8 in 28-pin packages are:
  - 23 general-purpose I/Os only
  - 7 keyboard interrupt with internal pull-up
  - 10 LED drivers (sink)
  - 12-channel ADC
  - Timer I/O pins on TIM1 only
- Specific features of the MC68HC908JL8 in 20-pin packages are:
  - 15 general-purpose I/Os only
  - 1 keyboard interrupt with internal pull-up
  - 4 LED drivers (sink)
  - 10-channel ADC
  - Timer I/O pins on TIM1 only

Features of the CPU08 include the following:

- Enhanced HC05 programming model
- Extensive loop control functions
- 16 addressing modes (eight more than the HC05)
- 16-bit index register and stack pointer
- Memory-to-memory data transfers
- Fast 8 × 8 multiply instruction
- Fast 16/8 divide instruction
- Binary-coded decimal (BCD) instructions
- Optimization for controller applications
- Efficient C language support

1.3 MCU Block Diagram

Figure 1-1 shows the structure of the MC68HC908JL8.

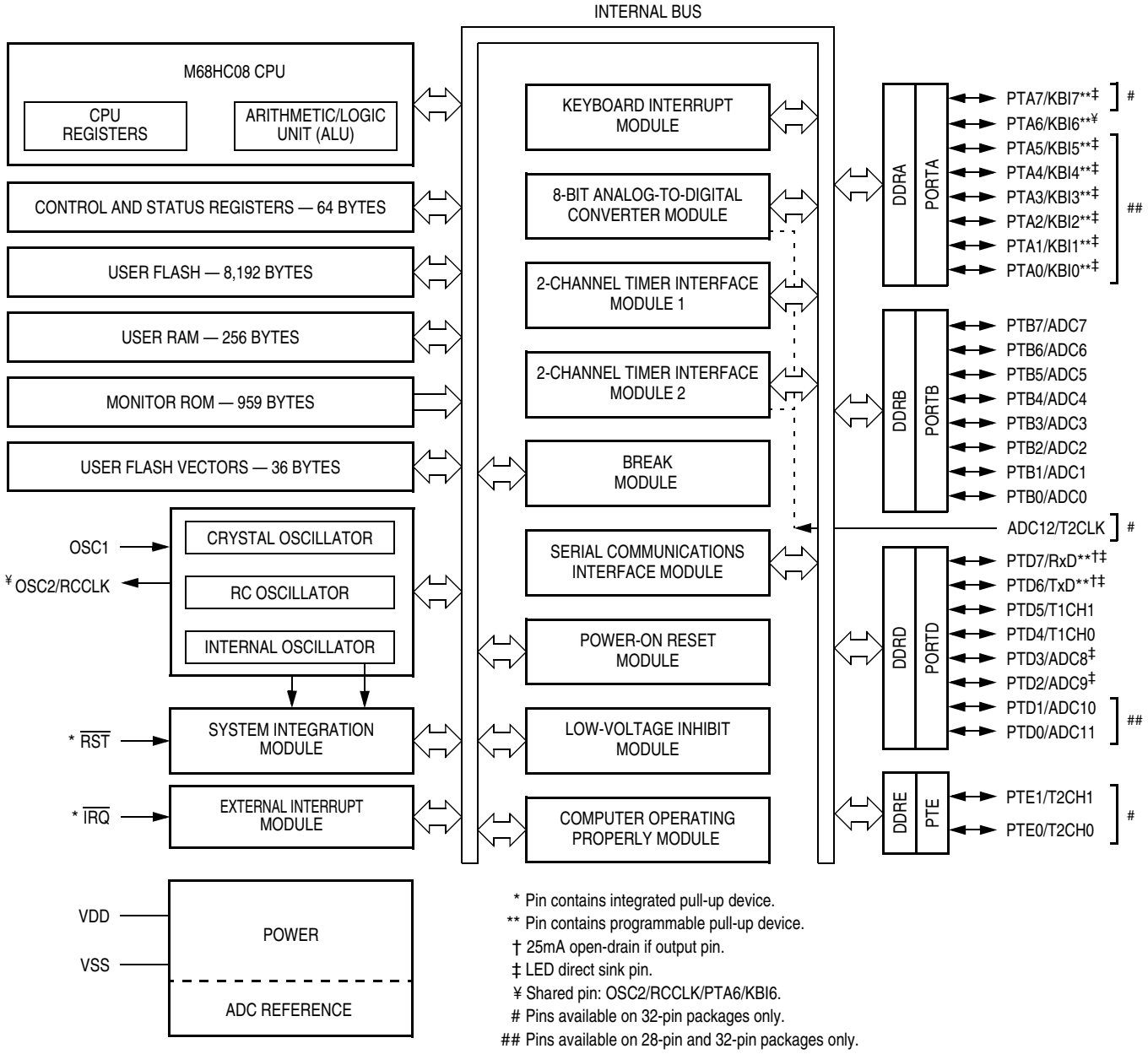


Figure 1-1. MC68HC908JL8 Block Diagram

1.4 Pin Assignments

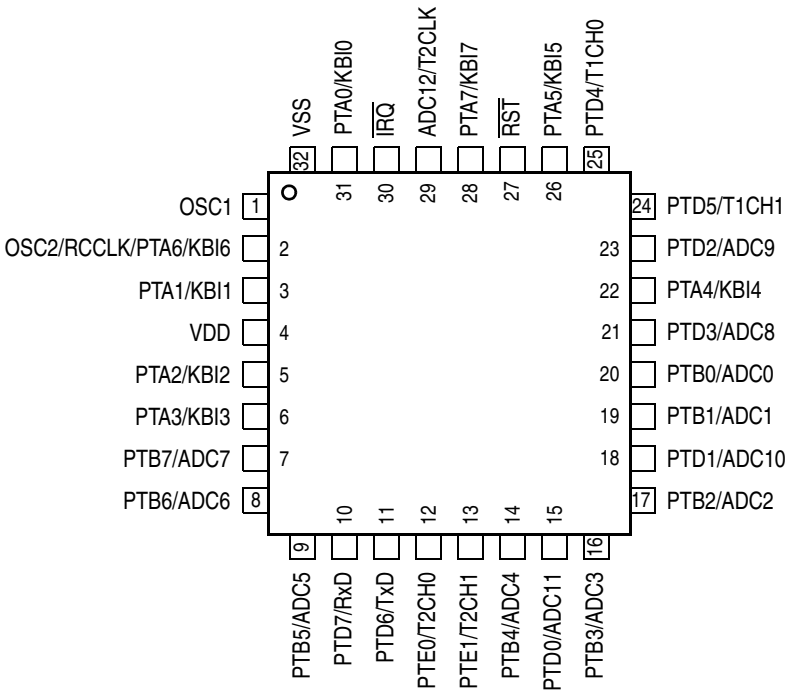


Figure 1-2. 32-Pin LQFP Pin Assignment

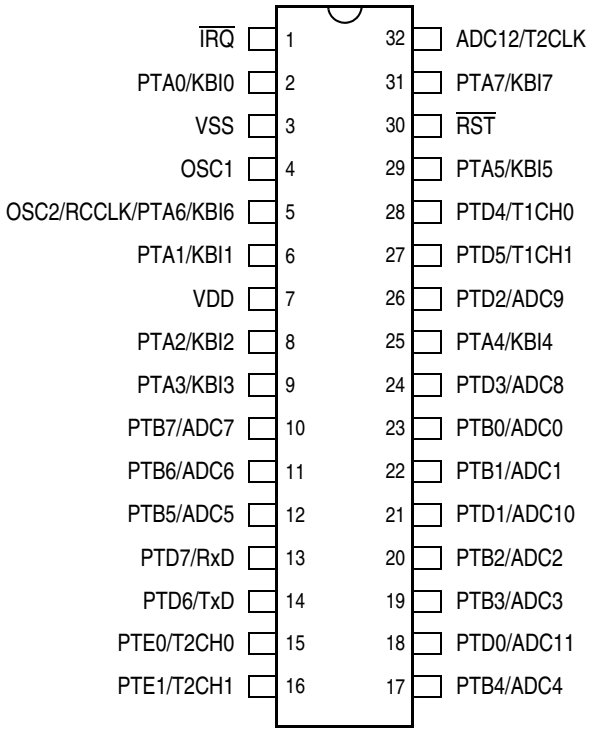
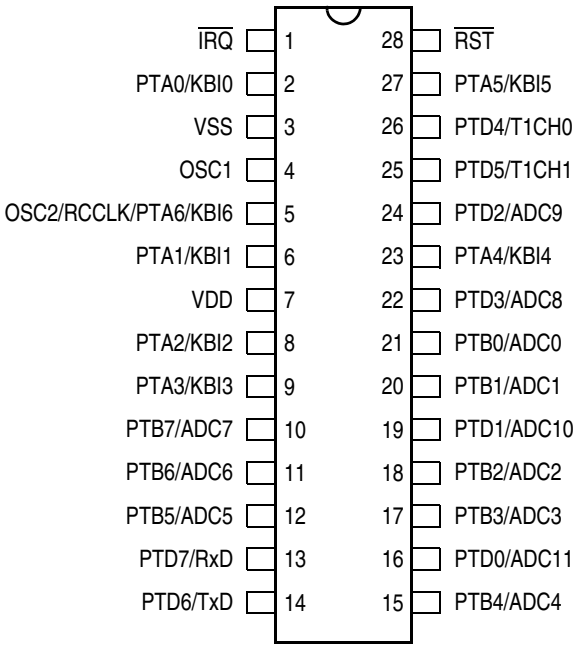


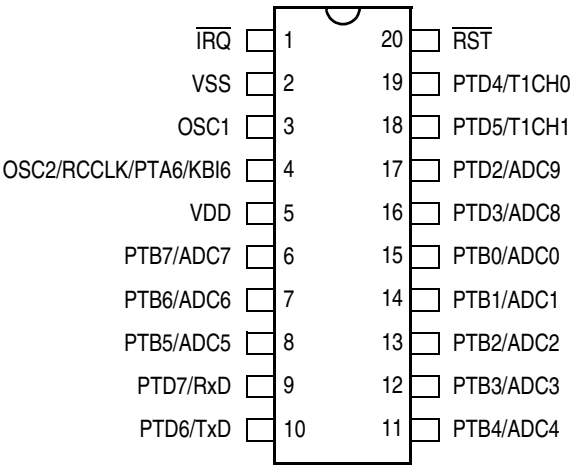
Figure 1-3. 32-Pin SDIP Pin Assignment



Pins not available on 28-pin packages	
	PTE0/T2CH0
	PTE1/T2CH1
	ADC12/T2CLK
PTA7/KBI7	

Internal pads are unconnected.
Set these unused port I/Os to output low.

Figure 1-4. 28-Pin PDIP/SOIC Pin Assignment



The 20-pin MC68HC908JL8 is designated MC68HC908JK8.

Pins not available on 20-pin packages	
PTA0/KBI0	PTD0/ADC11
PTA1/KBI1	PTD1/ADC10
PTA2/KBI2	
PTA3/KBI3	PTE0/T2CH0
PTA4/KBI4	PTE1/T2CH1
PTA5/KBI5	
	ADC12/T2CLK
PTA7/KBI7	

Internal pads are unconnected.
Set these unused port I/Os to output low.

Figure 1-5. 20-Pin PDIP/SOIC Pin Assignment

1.5 Pin Functions

Description of the pin functions are provided in Table 1-2.

Table 1-2. Pin Functions

PIN NAME	PIN DESCRIPTION	IN/OUT	VOLTAGE LEVEL
VDD	Power supply.	In	5V or 3V
VSS	Power supply ground.	Out	0V
$\overline{RST}$	Reset input, active low; with internal pull-up and schmitt trigger input.	In/Out	VDD
$\overline{IRQ}$	External IRQ pin; with programmable internal pull-up and schmitt trigger input.	In	VDD
	Used for monitor mode entry.	In	VDD to V_{TST}
OSC1	Crystal or RC oscillator input.	In	VDD
OSC2/RCCLK	OSC2: crystal oscillator output; inverted OSC1 signal.	Out	VDD
	RCCLK: RC oscillator clock output.	Out	VDD
	Pin as PTA6/KBI6 (see PTA0–PTA7).	In/Out	VDD
ADC12/T2CLK	ADC12: channel-12 input of ADC.	In	VSS to VDD
	T2CLK: external input clock for TIM2.	In	VDD
PTA0–PTA7	8-bit general purpose I/O port.	In/Out	VDD
	Each pin has programmable internal pull-up when configured as input.	In	VDD
	Pins as keyboard interrupts, KBI0–KBI7.	In	VDD
	PTA0–PTA5 and PTA7 have LED direct sink capability.	Out	VSS
	PTA6 as OSC2/RCCLK.	Out	VDD
PTB0–PTB7	8-bit general purpose I/O port.	In/Out	VDD
	Pins as ADC input channels, ADC0–ADC7.	In	VSS to VDD
PTD0–PTD7	8-bit general purpose I/O port; with programmable internal pull-ups on PTD6–PTD7.	In/Out	VDD
	PTD0–PTD3 as ADC input channels, ADC11–ADC8.	Input	VSS to VDD
	PTD2–PTD3 and PTD6–PTD7 have LED direct sink capability	Out	VSS
	PTD4 as T1CH0 of TIM1.	In/Out	VDD
	PTD5 as T1CH1 of TIM1.	In/Out	VDD
	PTD6–PTD7 have configurable 25mA open-drain output.	Out	VSS
	PTD6 as TxD of SCI.	Out	VDD
	PTD7 as RxD of SCI.	In	VDD

Table 1-2. Pin Functions (Continued)

PIN NAME	PIN DESCRIPTION	IN/OUT	VOLTAGE LEVEL
PTE0–PTE1	2-bit general purpose I/O port.	In/Out	VDD
	PTE0 as T2CH0 of TIM2.	In/Out	VDD
	PTE1 as T2CH1 of TIM2.	In/Out	VDD

NOTE

Devices in 28-pin packages, the following pins are not available: PTA7/KBI7, PTE0/T2CH0, PTE1/T2CH1, and ADC12/T2CLK.

Devices in 20-pin packages, the following pins are not available: PTA0/KBI0–PTA5/KBI5, PTD0/ADC11, PTD1/ADC10, PTA7/KBI7, PTE0/T2CH0, PTE1/T2CH1, and ADC12/T2CLK.

Chapter 2
Memory

2.1 Introduction

The CPU08 can address 64-kbytes of memory space. The memory map, shown in Figure 2-1, includes:

- 8,192 bytes of user FLASH memory
- 36 bytes of user-defined vectors
- 959 bytes of monitor ROM

2.2 I/O Section

Addresses $0000–$003F, shown in Figure 2-2, contain most of the control, status, and data registers. Additional I/O registers have the following addresses:

- $FE00; Break Status Register, BSR
- $FE01; Reset Status Register, RSR
- $FE02; Reserved
- $FE03; Break Flag Control Register, BFCR
- $FE04; Interrupt Status Register 1, INT1
- $FE05; Interrupt Status Register 2, INT2
- $FE06; Interrupt Status Register 3, INT3
- $FE07; Reserved
- $FE08; FLASH Control Register, FLCR
- $FE09; Reserved
- $FE0A; Reserved
- $FE0B; Reserved
- $FE0C; Break Address Register High, BRKH
- $FE0D; Break Address Register Low, BRKL
- $FE0E; Break Status and Control Register, BRKSCR
- $FE0F; Reserved
- $FFCF; FLASH Block Protect Register, FLBPR (FLASH register)
- $FFD0; Mask Option Register, MOR (FLASH register)
- $FFFF; COP Control Register, COPCTL

2.3 Monitor ROM

The 959 bytes at addresses $FC00–$FDFF and $FE10–$FFCE are reserved ROM addresses that contain the instructions for the monitor functions. (See Chapter 7 Monitor ROM (MON).)

$0000 ↓ $003F	I/O REGISTERS 64 BYTES
$0040 ↓ $005F	RESERVED 32 BYTES
$0060 ↓ $015F	RAM 256 BYTES
$0160 ↓ $DBFF	UNIMPLEMENTED 55,968 BYTES
$DC00 ↓ $FBFF	FLASH MEMORY 8,192 BYTES
$FC00 ↓ $FDFF	MONITOR ROM 512 BYTES
$FE00	BREAK STATUS REGISTER (BSR)
$FE01	RESET STATUS REGISTER (RSR)
$FE02	RESERVED
$FE03	BREAK FLAG CONTROL REGISTER (BFCR)
$FE04	INTERRUPT STATUS REGISTER 1 (INT1)
$FE05	INTERRUPT STATUS REGISTER 2 (INT2)
$FE06	INTERRUPT STATUS REGISTER 3 (INT3)
$FE07	RESERVED
$FE08	FLASH CONTROL REGISTER (FLCR)
$FE09 ↓ $FF0B	RESERVED
$FE0C	BREAK ADDRESS HIGH REGISTER (BRKH)
$FE0D	BREAK ADDRESS LOW REGISTER (BRKL)
$FE0E	BREAK STATUS AND CONTROL REGISTER (BRKSCR)
$FE0F	RESERVED
$FE10 ↓ $FFCE	MONITOR ROM 447 BYTES
$FFCF	FLASH BLOCK PROTECT REGISTER (FLBPR)
$FFD0	MASK OPTION REGISTER (MOR)
$FFD1 ↓ $FFDB	RESERVED 11 BYTES
$FFDC ↓ $FFFF	USER FLASH VECTORS 36 BYTES

Figure 2-1. Memory Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PTA)	Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
		Reset:	Unaffected by reset							
$0001	Port B Data Register (PTB)	Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Reset:	Unaffected by reset							
$0002	Unimplemented	Read: Write:								
$0003	Port D Data Register (PTD)	Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Reset:	Unaffected by reset							
$0004	Data Direction Register A (DDRA)	Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B (DDRB)	Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Reset:	0	0	0	0	0	0	0	0
$0006	Unimplemented	Read: Write:								
$0007	Data Direction Register D (DDRD)	Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PTE)	Read: Write:							PTE1	PTE0
		Reset:	Unaffected by reset							
$0009	Unimplemented	Read: Write:								
$000A	Port D Control Register (PDCR)	Read: Write:	0	0	0	0	SLOWD7	SLOWD6	PTDPU7	PTDPU6
		Reset:	0	0	0	0	0	0	0	0
$000B	Unimplemented	Read: Write:								
$000C	Data Direction Register E (DDRE)	Read: Write:							DDRE1	DDRE0
		Reset:	0	0	0	0	0	0	0	0
$000D	Port A Input Pull-up Enable Register (PTAPUE)	Read: Write:	PTA6EN	PTAPUE6	PTAPUE5	PTAPUE4	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
		Reset:	0	0	0	0	0	0	0	0
$000E	PTA7 Input Pull-up Enable Register (PTA7PUE)	Read: Write:	PTAPUE7							
		Reset:	0	0	0	0	0	0	0	0
$000F ↓ $0012	Unimplemented	Read: Write:								

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 1 of 5)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0013	SCI Control Register 1 (SCC1)	Read:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0014	SCI Control Register 2 (SCC2)	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	SCI Control Register 3 (SCC3)	Read:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
		Write:								
		Reset:	U	U	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1)	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2)	Read:							BKF	RPF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0018	SCI Data Register (SCDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0019	SCI Baud Rate Register (SCBR)	Read:			SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001A	Keyboard Status and Control Register (KBSCR)	Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
		Write:						ACKK		
		Reset:	0	0	0	0	0	0	0	0
$001B	Keyboard Interrupt Enable Register (KBIER)	Read:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001C	Unimplemented	Read:								
		Write:								
$001D	IRQ Status and Control Register (INTSCR)	Read:	0	0	0	0	IRQF	0	IMASK	MODE
		Write:						ACK		
		Reset:	0	0	0	0	0	0	0	0
$001E	Configuration Register 2 (CONFIG2)†	Read:	IRQPUD	R	R	LVIT1	LVIT0	R	R	STOP_ ICLKDIS
		Write:								
		Reset:	0	0	0	0*	0*	0	0	0
$001F	Configuration Register 1 (CONFIG1)†	Read:	COPRS	R	R	LVID	R	SSREC	STOP	COPD
		Write:								
		Reset:	0	0	0	0	0	0	0	0

† One-time writable register after each reset. * LVIT1 and LVIT0 reset to logic 0 by a power-on reset (POR) only.

$0020	TIM1 Status and Control Register (T1SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0021	TIM1 Counter Register High (T1CNTH)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 2 of 5)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0022	TIM1 Counter Register Low (T1CNTL)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0023	TIM Counter Modulo Register High (TMODH)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0024	TIM1 Counter Modulo Register Low (T1MODL)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0025	TIM1 Channel 0 Status and Control Register (T1SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0026	TIM1 Channel 0 Register High (T1CH0H)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	Indeterminate after reset							
$0027	TIM1 Channel 0 Register Low (T1CH0L)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	Indeterminate after reset							
$0028	TIM1 Channel 1 Status and Control Register (T1SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0029	TIM1 Channel 1 Register High (T1CH1H)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	Indeterminate after reset							
$002A	TIM1 Channel 1 Register Low (T1CH1L)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	Indeterminate after reset							
$002B ↓ $002F	Unimplemented	Read:								
		Write:								
$0030	TIM2 Status and Control Register (T2SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0031	TIM2 Counter Register High (T2CNTH)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0032	TIM2 Counter Register Low (T2CNTL)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0033	TIM2 Counter Modulo Register High (T2MODH)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0034	TIM2 Counter Modulo Register Low (T2MODL)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	1	1	1	1	1	1	1	1

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 3 of 5)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0035	TIM2 Channel 0 Status and Control Register (T2SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0036	TIM2 Channel 0 Register High (T2CH0H)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	Indeterminate after reset							
$0037	TIM2 Channel 0 Register Low (T2CH0L)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	Indeterminate after reset							
$0038	TIM2 Channel 1 Status and Control Register (T2SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0039	TIM2 Channel 1 Register High (T2CH1H)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	Indeterminate after reset							
$003A	TIM2 Channel 1 Register Low (T2CH1L)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	Indeterminate after reset							
$003B	Unimplemented	Read:								
		Write:								
$003C	ADC Status and Control Register (ADSCR)	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
$003D	ADC Data Register (ADR)	Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
		Write:								
		Reset:	Indeterminate after reset							
$003E	ADC Input Clock Register (ADICLK)	Read:	ADIV2	ADIV1	ADIV0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$003F	Unimplemented	Read:								
		Write:								
$FE00	Break Status Register (BSR)	Read:	R	R	R	R	R	R	SBSW	R
		Write:							See note	
		Reset:							0	
	Note: Writing a logic 0 clears SBSW.									
$FE01	Reset Status Register (RSR)	Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
		Write:								
		POR:	1	0	0	0	0	0	0	0
$FE02	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
$FE03	Break Flag Control Register (BFCR)	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 4 of 5)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE04	Interrupt Status Register 1 (INT1)	Read:	IF6	IF5	IF4	IF3	0	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE05	Interrupt Status Register 2 (INT2)	Read:	IF14	IF13	IF12	IF11	0	0	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE06	Interrupt Status Register 3 (INT3)	Read:	0	0	0	0	0	0	0	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE07	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
$FE08	FLASH Control Register (FLCR)	Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE09 ↓ $FE0B	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
$FE0C	Break Address High Register (BRKH)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0D	Break Address low Register (BRKL)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0E	Break Status and Control Register (BRKSCR)	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FFCF	FLASH Block Protect Register (FLBPR)#	Read:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
		Write:								
		Reset:	Unaffected by reset; $FF when blank							
$FFD0	Mask Option Register (MOR)#	Read:	OSCSEL	R	R	R	R	R	R	R
		Write:								
		Reset:	Unaffected by reset; $FF when blank							

Non-volatile FLASH registers; write by programming.

$FFFF	COP Control Register (COPCTL)	Read:	Low byte of reset vector
		Write:	Writing clears COP counter (any value)
		Reset:	Unaffected by reset

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 5 of 5)

Table 2-1. Vector Addresses

Vector Priority	INT Flag	Address	Vector
Lowest	—	$FFD0 ↓ $FFDD	Not Used
	IF15	$FFDE	ADC Conversion Complete Vector (High)
		$FFDF	ADC Conversion Complete Vector (Low)
	IF14	$FFE0	Keyboard Interrupt Vector (High)
		$FFE1	Keyboard Interrupt Vector (Low)
	IF13	$FFE2	SCI Transmit Vector (High)
		$FFE3	SCI Transmit Vector (Low)
	IF12	$FFE4	SCI Receive Vector (High)
		$FFE5	SCI Receive Vector (Low)
	IF11	$FFE6	SCI Error Vector (High)
		$FFE7	SCI Error Vector (Low)
	IF10 ↓ IF9	—	Not Used
	IF8	$FFEC	TIM2 Overflow Vector (High)
		$FFED	TIM2 Overflow Vector (Low)
	IF7	$FFEE	TIM2 Channel 1 Vector (High)
		$FFEF	TIM2 Channel 1 Vector (Low)
	IF6	$FFF0	TIM2 Channel 0 Vector (High)
		$FFF1	TIM2 Channel 0 Vector (Low)
	IF5	$FFF2	TIM1 Overflow Vector (High)
		$FFF3	TIM1 Overflow Vector (Low)
	IF4	$FFF4	TIM1 Channel 1 Vector (High)
		$FFF5	TIM1 Channel 1 Vector (Low)
	IF3	$FFF6	TIM1 Channel 0 Vector (High)
		$FFF7	TIM1 Channel 0 Vector (Low)
	IF2	—	Not Used
	IF1	$FFFA	$\overline{\text{IRQ}}$ Vector (High)
		$FFFB	$\overline{\text{IRQ}}$ Vector (Low)
	—	$FFFC	SWI Vector (High)
		$FFFD	SWI Vector (Low)
	—	$FFFE	Reset Vector (High)
Highest		$FFFF	Reset Vector (Low)

2.4 Random-Access Memory (RAM)

Addresses $0060 through $015F are RAM locations. The location of the stack RAM is programmable. The 16-bit stack pointer allows the stack to be anywhere in the 64-Kbyte memory space.

NOTE

For correct operation, the stack pointer must point only to RAM locations.

Within page zero are 160 bytes of RAM. Because the location of the stack RAM is programmable, all page zero RAM locations can be used for I/O control and user data or code. When the stack pointer is moved from its reset location at $00FF, direct addressing mode instructions can access efficiently all page zero RAM locations. Page zero RAM, therefore, provides ideal locations for frequently accessed global variables.

Before processing an interrupt, the CPU uses five bytes of the stack to save the contents of the CPU registers.

NOTE

For M6805 compatibility, the H register is not stacked.

During a subroutine call, the CPU uses two bytes of the stack to store the return address. The stack pointer decrements during pushes and increments during pulls.

NOTE

Be careful when using nested subroutines. The CPU may overwrite data in the RAM during a subroutine or during the interrupt stacking operation.

2.5 FLASH Memory

This sub-section describes the operation of the embedded FLASH memory. The FLASH memory can be read, programmed, and erased from a single external supply. The program and erase operations are enabled through the use of an internal charge pump.

2.6 Functional Description

The FLASH memory consists of an array of 8,192 bytes for user memory plus a block of 36 bytes for user interrupt vectors. *An erased bit reads as logic 1 and a programmed bit reads as a logic 0*. The FLASH memory page size is defined as 64 bytes, and is the minimum size that can be erased in a page erase operation. Program and erase operations are facilitated through control bits in FLASH control register (FLCR).

The address ranges for the FLASH memory are:

- $DC00–$FBFF; user memory; 12,288 bytes
- $FFDC–$FFFF; user interrupt vectors; 36 bytes

Programming tools are available from Freescale. Contact your local Freescale representative for more information.

NOTE

A security feature prevents viewing of the FLASH contents.[1]

1. No security feature is absolutely secure. However, Motorola's strategy is to make reading or copying the FLASH difficult for unauthorized users.

2.7 FLASH Control Register

The FLASH control register (FCLR) controls FLASH program and erase operations.

Address: $FE08

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 2-3. FLASH Control Register (FLCR)

HVEN — High Voltage Enable Bit

This read/write bit enables the charge pump to drive high voltages for program and erase operations in the array. HVEN can only be set if either PGM = 1 or ERASE = 1 and the proper sequence for program or erase is followed.

1 = High voltage enabled to array and charge pump on
0 = High voltage disabled to array and charge pump off

MASS — Mass Erase Control Bit

This read/write bit configures the memory for mass erase operation or page erase operation when the ERASE bit is set.

1 = Mass erase operation selected
0 = Page erase operation selected

ERASE — Erase Control Bit

This read/write bit configures the memory for erase operation. ERASE is interlocked with the PGM bit such that both bits cannot be equal to 1 or set to 1 at the same time.

1 = Erase operation selected
0 = Erase operation not selected

PGM — Program Control Bit

This read/write bit configures the memory for program operation. PGM is interlocked with the ERASE bit such that both bits cannot be equal to 1 or set to 1 at the same time.

1 = Program operation selected
0 = Program operation not selected

2.8 FLASH Page Erase Operation

Use the following procedure to erase a page of FLASH memory. A page consists of 64 consecutive bytes starting from addresses $XX00, $XX40, $XX80 or $XXC0. The 36-byte user interrupt vectors area also forms a page. Any page within the 8,192 bytes user memory area ($DC00–$FBFF) can be erased alone. *The 36-byte user interrupt vectors cannot be erased by the page erase operation because of security reasons. Mass erase is required to erase this page.*

1. Set the ERASE bit and clear the MASS bit in the FLASH control register.
2. Read the FLASH block protect register.
3. Write any data to any FLASH address within the page address range desired.
4. Wait for a time, t_{nvs} (10μs).
5. Set the HVEN bit.
6. Wait for a time t_{erase} (4ms).

7. Clear the ERASE bit.
8. Wait for a time, t_{nvh} (5µs).
9. Clear the HVEN bit.
10. After time, t_{rcv} (1µs), the memory can be accessed in read mode again.

NOTE

Programming and erasing of FLASH locations cannot be performed by code being executed from the FLASH memory. While these operations must be performed in the order as shown, but other unrelated operations may occur between the steps.

2.9 FLASH Mass Erase Operation

Use the following procedure to erase the entire FLASH memory:

1. Set both the ERASE bit and the MASS bit in the FLASH control register.
2. Read the FLASH block protect register.
3. Write any data to any FLASH location within the FLASH memory address range.
4. Wait for a time, t_{nvs} (10µs).
5. Set the HVEN bit.
6. Wait for a time t_{merase} (4ms).
7. Clear the ERASE bit.
8. Wait for a time, t_{nvh1} (100µs).
9. Clear the HVEN bit.
10. After time, t_{rcv} (1µs), the memory can be accessed in read mode again.

NOTE

Programming and erasing of FLASH locations cannot be performed by code being executed from the FLASH memory. While these operations must be performed in the order as shown, but other unrelated operations may occur between the steps.

2.10 FLASH Program Operation

Programming of the FLASH memory is done on a row basis. A row consists of 32 consecutive bytes starting from addresses $XX00, $XX20, $XX40, $XX60, $XX80, $XXA0, $XXC0 or $XXE0. Use this step-by-step procedure to program a row of FLASH memory:
(Figure 2-4 shows a flowchart of the programming algorithm.)

1. Set the PGM bit. This configures the memory for program operation and enables the latching of address and data for programming.
2. Read the FLASH block protect register.
3. Write any data to any FLASH location within the address range of the row to be programmed.
4. Wait for a time, t_{nvs} (10µs).
5. Set the HVEN bit.
6. Wait for a time, t_{pgs} (5µs).
7. Write data to the FLASH address to be programmed.

8. Wait for time, t_{prog} (30μs).
9. Repeat steps 7 and 8 until all bytes within the row are programmed.
10. Clear the PGM bit.
11. Wait for time, t_{nvh} (5μs).
12. Clear the HVEN bit.
13. After time, t_{rcv} (1μs), the memory can be accessed in read mode again.

This program sequence is repeated throughout the memory until all data is programmed.

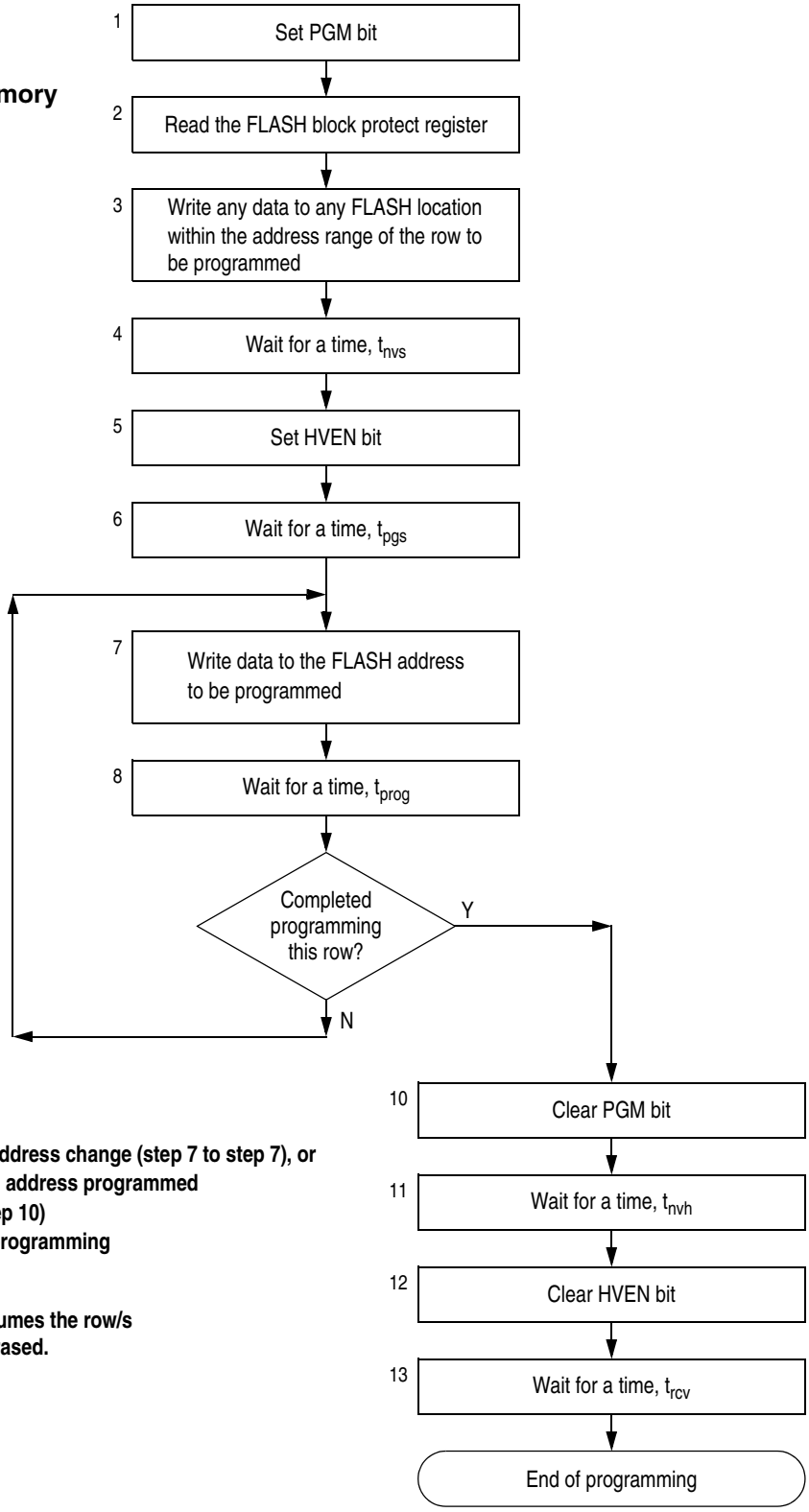
NOTE

The time between each FLASH address change (step 7 to step 7), or the time between the last FLASH addressed programmed to clearing the PGM bit (step 7 to step 10), must not exceed the maximum programming time, t_{prog} max.

NOTE

Programming and erasing of FLASH locations cannot be performed by code being executed from the FLASH memory. While these operations must be performed in the order shown, other unrelated operations may occur between the steps.

Algorithm for programming a row (32 bytes) of FLASH memory



NOTE:

The time between each FLASH address change (step 7 to step 7), or the time between the last FLASH address programmed to clearing PGM bit (step 7 to step 10) must not exceed the maximum programming time, t_{prog} max.

This row program algorithm assumes the row/s to be programmed are initially erased.

Figure 2-4. FLASH Programming Flowchart

2.11 FLASH Block Protection

Due to the ability of the on-board charge pump to erase and program the FLASH memory in the target application, provision is made to protect blocks of memory from unintentional erase or program operations due to system malfunction. This protection is done by use of a FLASH block protect register (FLBPR). The FLBPR determines the range of the FLASH memory which is to be protected. The range of the protected area starts from a location defined by FLBPR and ends to the bottom of the FLASH memory ($FFFF). When the memory is protected, the HVEN bit cannot be set in either erase or program operations.

NOTE

In performing a program or erase operation, the FLASH block protect register must be read after setting the PGM or ERASE bit and before asserting the HVEN bit

When the FLBPR is program with all 0's, the entire memory is protected from being programmed and erased. When all the bits are erased
(all 1's), the entire memory is accessible for program and erase.

When bits within the FLBPR are programmed, they lock a block of memory, address ranges as shown in 2.12 FLASH Block Protect Register. Once the FLBPR is programmed with a value other than $FF, any erase or program of the FLBPR or the protected block of FLASH memory is prohibited. The FLBPR itself can be erased or programmed only with an external voltage, V_{TST}, present on the $\overline{IRQ}$ pin. This voltage also allows entry from reset into the monitor mode.

2.12 FLASH Block Protect Register

The FLASH block protect register (FLBPR) is implemented as a byte within the FLASH memory, and therefore can only be written during a programming sequence of the FLASH memory. The value in this register determines the starting location of the protected range within the FLASH memory.

Address: $FFCF

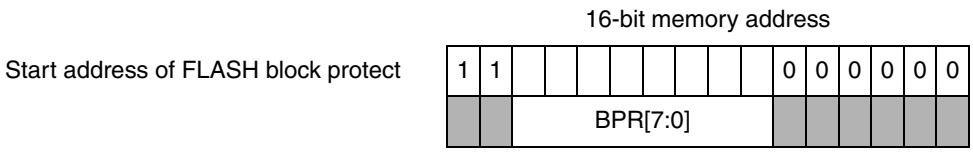
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
Reset:	Unaffected by reset; $FF when blank							

Non-volatile FLASH register; write by programming.

Figure 2-5. FLASH Block Protect Register (FLBPR)

BPR[7:0] — FLASH Block Protect Bits

BPR[7:0] represent bits [13:6] of a 16-bit memory address. Bits [15:14] are logic 1's and bits [5:0] are logic 0's.



The resultant 16-bit address is used for specifying the start address of the FLASH memory for block protection. The FLASH is protected from this start address to the end of FLASH memory, at $FFFF. With this mechanism, the protect start address can be XX00, XX40, XX80, or XXC0 (at page boundaries — 64 bytes) within the FLASH memory.

Examples of protect start address:

BPR[7:0]	Start of Address of Protect Range (1)
$00–$70	The entire FLASH memory is protected.
$71 (**0111 0001**)	$DC40 (11**01 1100 01**00 0000)
$72 (**0111 0010**)	$DC80 (11**01 1100 10**00 0000)
$73 (**0111 0011**)	$DCC0 (11**01 1100 11**00 0000)
and so on...	
$FD (**1111 1101**)	$FF40 (11**11 1111 01**00 0000)
$FE (**1111 1110**)	$FF80 (11**11 1111 10**00 0000)
$FF	The entire FLASH memory is not protected.

1. The end address of the protected range is always $FFFF.

Chapter 3
Configuration and Mask Option Registers (CONFIG & MOR)

3.1 Introduction

This section describes the configuration registers, CONFIG1 and CONFIG2; and the mask option register (MOR).

The configuration registers enable or disable these options:

- Computer operating properly module (COP)
- COP timeout period ($2^{13}–2^4$ or $2^{18}–2^4$ ICLK cycles)
- Internal oscillator during stop mode
- Low voltage inhibit (LVI) module
- LVI module voltage trip point selection
- STOP instruction
- Stop mode recovery time (32 or 4096 ICLK cycles)
- Pull-up on $\overline{\text{IRQ}}$ pin

The mask option register selects the oscillator option:

- Crystal or RC

3.2 Functional Description

The configuration registers are used in the initialization of various options. The configuration registers can be written once after each reset. All of the configuration register bits are cleared during reset. Since the various options affect the operation of the MCU, it is recommended that these registers be written immediately after reset. The configuration registers are located at $001E and $001F. The configuration registers may be read at anytime.

NOTE

The options except LVIT[1:0] are one-time writable by the user after each reset. The LVIT[1:0] bits are one-time writable by the user only after each POR (power-on reset). The CONFIG registers are not in the FLASH memory but are special registers containing one-time writable latches after each reset. Upon a reset, the CONFIG registers default to predetermined settings as shown in Figure 3-1 and Figure 3-2.

The mask option register (MOR) is used to select the oscillator option for the MCU: crystal oscillator or RC oscillator. The MOR is implemented as a byte in FLASH memory. Hence, writing to the MOR requires programming the byte.

3.3 Configuration Register 1 (CONFIG1)

Address:	$001F							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	COPRS	R	R	LVID	R	SSREC	STOP	COPD
Reset:	0	0	0	0	0	0	0	0
	R = Reserved							

Figure 3-1. Configuration Register 1 (CONFIG1)

COPRS — COP Rate Select Bit

COPRS selects the COP time-out period. Reset clears COPRS.
(See Chapter 14 Computer Operating Properly (COP).)

1 = COP timeout period is ($2^{13} – 2^4$) ICLK cycles
0 = COP timeout period is ($2^{18} – 2^4$) ICLK cycles

LVID — Low Voltage Inhibit Disable Bit

LVID disables the LVI module. Reset clears LVID.
(See Chapter 15 Low Voltage Inhibit (LVI).)

1 = Low voltage inhibit disabled
0 = Low voltage inhibit enabled

SSREC — Short Stop Recovery Bit

SSREC enables the CPU to exit stop mode with a delay of 32 ICLK cycles instead of a 4096 ICLK cycle delay.

1 = Stop mode recovery after 32 ICLK cycles
0 = Stop mode recovery after 4096 ICLK cycles

NOTE

Exiting stop mode by pulling reset will result in the long stop recovery. If using an external crystal, do not set the SSREC bit.

STOP — STOP Instruction Enable Bit

STOP enables the STOP instruction.

1 = STOP instruction enabled
0 = STOP instruction treated as illegal opcode

COPD — COP Disable Bit

COPD disables the COP module. Reset clears COPD.
(See Chapter 14 Computer Operating Properly (COP).)

1 = COP module disabled
0 = COP module enabled

3.4 Configuration Register 2 (CONFIG2)

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	IRQPUD	R	R	LVIT1	LVIT0	R	R	STOP_ ICLKDIS
Reset:	0	0	0	Not affected	Not affected	0	0	0
POR:	0	0	0	0	0	0	0	0

R = Reserved

One-time writable register after each reset. LVIT1 and LVIT0 reset to logic 0 by a power-on reset (POR) only.

Figure 3-2. Configuration Register 2 (CONFIG2)

IRQPUD — $\overline{\text{IRQ}}$ Pin Pull-Up Disable Bit

IRQPUD disconnects the internal pull-up on the $\overline{\text{IRQ}}$ pin.

1 = Internal pull-up is disconnected
0 = Internal pull-up is connected between $\overline{\text{IRQ}}$ pin and V_{DD}

LVIT1, LVIT0 — LVI Trip Voltage Selection Bits

Detail description of trip voltage selection is given in Chapter 15 Low Voltage Inhibit (LVI).

STOP_ICLKDIS — Internal Oscillator Stop Mode Disable Bit

Setting STOP_ICLKDIS disables the internal oscillator during stop mode. When this bit is cleared, the internal oscillator continues to operate in stop mode. Reset clears this bit.

1 = Internal oscillator disabled during stop mode
0 = Internal oscillator enabled during stop mode

3.5 Mask Option Register (MOR)

The mask option register (MOR) is implemented as a byte within the FLASH memory, and therefore can only be written during a programming sequence of the FLASH memory. This register is read after a power-on reset to determine the type of oscillator selected. (See Chapter 6 Oscillator (OSC).)

Address: $FFD0

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	OSCSEL	R	R	R	R	R	R	R
Erased:	1	1	1	1	1	1	1	1
Reset:	Unaffected by reset							

Non-volatile FLASH register; write by programming.

R = Reserved

Figure 3-3. Mask Option Register (MOR)

OSCSEL — Oscillator Select Bit

OSCSEL selects the oscillator type for the MCU. The erased or unprogrammed state of this bit is logic 1, selecting the crystal oscillator option. This bit is unaffected by reset.

1 = Crystal oscillator
0 = RC oscillator

Bits 6–0 — Should be left as logic 1’s.

NOTE

When Crystal oscillator is selected, the OSC2/RCCLK/PTA6/KBI6 pin is used as OSC2; other functions such as PTA6/KBI6 will not be available.

Chapter 4
Central Processor Unit (CPU)

4.1 Introduction

The M68HC08 CPU (central processor unit) is an enhanced and fully object-code-compatible version of the M68HC05 CPU. The *CPU08 Reference Manual* (Freescale document order number CPU08RM/AD) contains a description of the CPU instruction set, addressing modes, and architecture.

4.2 Features

- Object code fully upward-compatible with M68HC05 Family
- 16-bit stack pointer with stack manipulation instructions
- 16-Bit Index Register with X-Register Manipulation Instructions
- 8-MHz CPU Internal Bus Frequency
- 64-Kbyte Program/Data Memory Space
- 16 Addressing Modes
- Memory-to-Memory Data Moves without Using Accumulator
- Fast 8-Bit by 8-Bit Multiply and 16-Bit by 8-Bit Divide Instructions
- Enhanced Binary-Coded Decimal (BCD) Data Handling
- Modular Architecture with Expandable Internal Bus Definition for Extension of Addressing Range beyond 64 Kbytes
- Low-Power Stop and Wait Modes

4.3 CPU Registers

Figure 4-1 shows the five CPU registers. CPU registers are not part of the memory map.

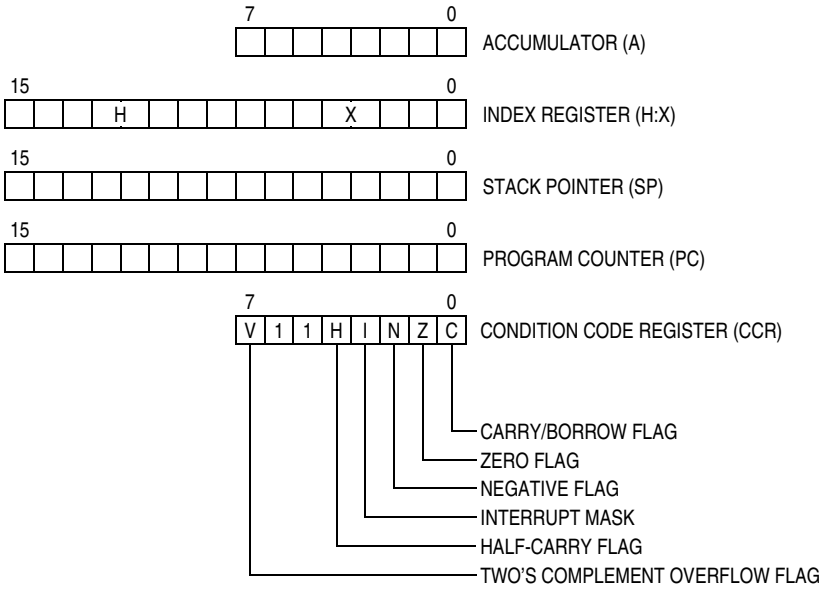


Figure 4-1. CPU Registers

4.3.1 Accumulator

The accumulator is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and the results of arithmetic/logic operations.

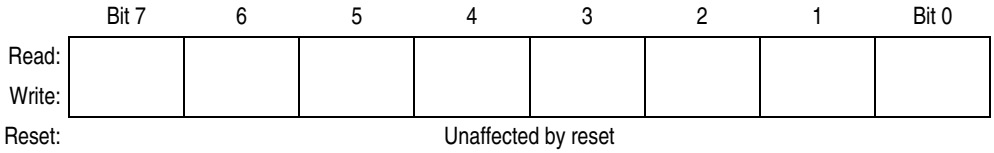


Figure 4-2. Accumulator (A)

4.3.2 Index Register

The 16-bit index register allows indexed addressing of a 64-Kbyte memory space. H is the upper byte of the index register, and X is the lower byte. H:X is the concatenated 16-bit index register.

In the indexed addressing modes, the CPU uses the contents of the index register to determine the conditional address of the operand.

The index register can serve also as a temporary data storage location.

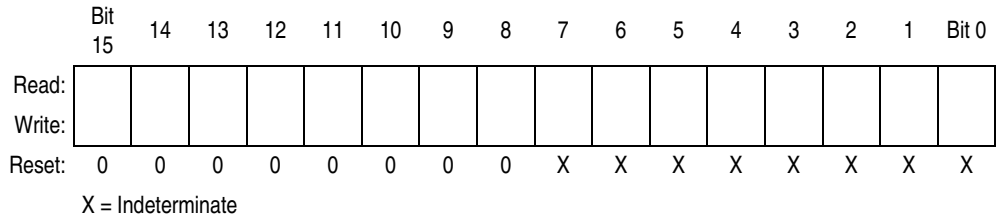


Figure 4-3. Index Register (H:X)

4.3.3 Stack Pointer

The stack pointer is a 16-bit register that contains the address of the next location on the stack. During a reset, the stack pointer is preset to $00FF. The reset stack pointer (RSP) instruction sets the least significant byte to $FF and does not affect the most significant byte. The stack pointer decrements as data is pushed onto the stack and increments as data is pulled from the stack.

In the stack pointer 8-bit offset and 16-bit offset addressing modes, the stack pointer can function as an index register to access data on the stack. The CPU uses the contents of the stack pointer to determine the conditional address of the operand.

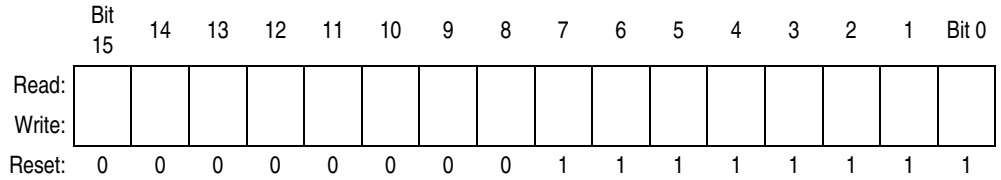


Figure 4-4. Stack Pointer (SP)

NOTE

The location of the stack is arbitrary and may be relocated anywhere in RAM. Moving the SP out of page 0 ($0000 to $00FF) frees direct address (page 0) space. For correct operation, the stack pointer must point only to RAM locations.

4.3.4 Program Counter

The program counter is a 16-bit register that contains the address of the next instruction or operand to be fetched.

Normally, the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, and interrupt operations load the program counter with an address other than that of the next sequential location.

During reset, the program counter is loaded with the reset vector address located at $FFFE and $FFFF. The vector address is the address of the first instruction to be executed after exiting the reset state.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
Read: Write:																
Reset:	Loaded with Vector from $FFFE and $FFFF															

Figure 4-5. Program Counter (PC)

4.3.5 Condition Code Register

The 8-bit condition code register contains the interrupt mask and five flags that indicate the results of the instruction just executed. Bits 6 and 5 are set permanently to logic 1. The following paragraphs describe the functions of the condition code register.

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

X = Indeterminate

Figure 4-6. Condition Code Register (CCR)

V — Overflow Flag

The CPU sets the overflow flag when a two's complement overflow occurs. The signed branch instructions BGT, BGE, BLE, and BLT use the overflow flag.

1 = Overflow
0 = No overflow

H — Half-Carry Flag

The CPU sets the half-carry flag when a carry occurs between accumulator bits 3 and 4 during an add-without-carry (ADD) or add-with-carry (ADC) operation. The half-carry flag is required for binary-coded decimal (BCD) arithmetic operations. The DAA instruction uses the states of the H and C flags to determine the appropriate correction factor.

1 = Carry between bits 3 and 4
0 = No carry between bits 3 and 4

I — Interrupt Mask

When the interrupt mask is set, all maskable CPU interrupts are disabled. CPU interrupts are enabled when the interrupt mask is cleared. When a CPU interrupt occurs, the interrupt mask is set automatically after the CPU registers are saved on the stack, but before the interrupt vector is fetched.

1 = Interrupts disabled
0 = Interrupts enabled

NOTE

To maintain M6805 Family compatibility, the upper byte of the index register (H) is not stacked automatically. If the interrupt service routine modifies H, then the user must stack and unstack H using the PSHH and PULH instructions.

After the I bit is cleared, the highest-priority interrupt request is serviced first.
A return-from-interrupt (RTI) instruction pulls the CPU registers from the stack and restores the interrupt mask from the stack. After any reset, the interrupt mask is set and can be cleared only by the clear interrupt mask software instruction (CLI).

N — Negative flag

The CPU sets the negative flag when an arithmetic operation, logic operation, or data manipulation produces a negative result, setting bit 7 of the result.

1 = Negative result
0 = Non-negative result

Z — Zero flag

The CPU sets the zero flag when an arithmetic operation, logic operation, or data manipulation produces a result of $00.

1 = Zero result
0 = Non-zero result

C — Carry/Borrow Flag

The CPU sets the carry/borrow flag when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some instructions — such as bit test and branch, shift, and rotate — also clear or set the carry/borrow flag.

1 = Carry out of bit 7
0 = No carry out of bit 7

4.4 Arithmetic/Logic Unit (ALU)

The ALU performs the arithmetic and logic operations defined by the instruction set.

Refer to the *CPU08 Reference Manual* (Freescale document order number CPU08RM/AD) for a description of the instructions and addressing modes and more detail about the architecture of the CPU.

4.5 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

4.5.1 Wait Mode

The WAIT instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling interrupts. After exit from wait mode by interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock

4.5.2 Stop Mode

The STOP instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling external interrupts. After exit from stop mode by external interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock

After exiting stop mode, the CPU clock begins running after the oscillator stabilization delay.

4.6 CPU During Break Interrupts

If a break module is present on the MCU, the CPU starts a break interrupt by:

- Loading the instruction register with the SWI instruction
- Loading the program counter with $FFFC:$FFFD or with $FEFC:$FEFD in monitor mode

The break interrupt begins after completion of the CPU instruction in progress. If the break address register match occurs on the last cycle of a CPU instruction, the break interrupt begins immediately.

A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the MCU to normal operation if the break interrupt has been deasserted.

4.7 Instruction Set Summary

Table 4-1 provides a summary of the M68HC08 instruction set.

4.8 Opcode Map

The opcode map is provided in Table 4-2.

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
ADC #*opr* ADC *opr* ADC *opr* ADC *opr*,X ADC *opr*,X ADC ,X ADC *opr*,SP ADC *opr*,SP	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A9 B9 C9 D9 E9 F9 9EE9 9ED9	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
ADD #*opr* ADD *opr* ADD *opr* ADD *opr*,X ADD *opr*,X ADD ,X ADD *opr*,SP ADD *opr*,SP	Add without Carry	A ← (A) + (M)	↕	↕	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	AB BB CB DB EB FB 9EEB 9EDB	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
AIS #*opr*	Add Immediate Value (Signed) to SP	SP ← (SP) + (16 « M)	–	–	–	–	–	–	IMM	A7	ii	2
AIX #*opr*	Add Immediate Value (Signed) to H:X	H:X ← (H:X) + (16 « M)	–	–	–	–	–	–	IMM	AF	ii	2

Table 4-1. Instruction Set Summary

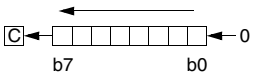
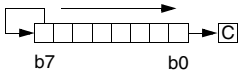
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
AND #*opr* AND *opr* AND *opr* AND *opr*,X AND *opr*,X AND ,X AND *opr*,SP AND *opr*,SP	Logical AND	A ← (A) & (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A4 B4 C4 D4 E4 F4 9EE4 9ED4	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
ASL *opr* ASLA ASLX ASL *opr*,X ASL ,X ASL *opr*,SP	Arithmetic Shift Left (Same as LSL)		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	4 1 1 4 3 5
ASR *opr* ASRA ASRX ASR *opr*,X ASR *opr*,X ASR *opr*,SP	Arithmetic Shift Right		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	37 47 57 67 77 9E67	dd ff ff	4 1 1 4 3 5
BCC *rel*	Branch if Carry Bit Clear	PC ← (PC) + 2 + rel ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BCLR *n*, *opr*	Clear Bit n in M	Mn ← 0	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	4 4 4 4 4 4 4 4
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + 2 + *rel* ? (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE *opr*	Branch if Greater Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGT *opr*	Branch if Greater Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) ǀ (N ⊕ V)=0	–	–	–	–	–	–	REL	92	rr	3
BHCC *rel*	Branch if Half Carry Bit Clear	PC ← (PC) + 2 + *rel* ? (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS *rel*	Branch if Half Carry Bit Set	PC ← (PC) + 2 + *rel* ? (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI *rel*	Branch if Higher	PC ← (PC) + 2 + *rel* ? (C) ǀ (Z) = 0	–	–	–	–	–	–	REL	22	rr	3
BHS *rel*	Branch if Higher or Same (Same as BCC)	PC ← (PC) + 2 + *rel* ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH *rel*	Branch if $\overline{\text{IRQ}}$ Pin High	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL *rel*	Branch if $\overline{\text{IRQ}}$ Pin Low	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 0	–	–	–	–	–	–	REL	2E	rr	3

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BIT #*opr* BIT *opr* BIT *opr* BIT *opr*,X BIT *opr*,X BIT ,X BIT *opr*,SP BIT *opr*,SP	Bit Test	(A) & (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A5 B5 C5 D5 E5 F5 9EE5 9ED5	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
BLE *opr*	Branch if Less Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) ∣ (N ⊕ V)=1	–	–	–	–	–	–	REL	93	rr	3
BLO *rel*	Branch if Lower (Same as BCS)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	PC ← (PC) + 2 + *rel* ? (C) ∣ (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT *opr*	Branch if Less Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 1	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel*	–	–	–	–	–	–	REL	20	rr	3
BRCLR *n*,*opr*,*rel*	Branch if Bit *n* in M Clear	PC ← (PC) + 3 + *rel* ? (Mn) = 0	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRN *rel*	Branch Never	PC ← (PC) + 2	–	–	–	–	–	–	REL	21	rr	3
BRSET *n*,*opr*,*rel*	Branch if Bit *n* in M Set	PC ← (PC) + 3 + *rel* ? (Mn) = 1	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BSET *n*,*opr*	Set Bit *n* in M	Mn ← 1	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	4 4 4 4 4 4 4 4

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BSR *rel*	Branch to Subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) – 1; push (PCH) SP ← (SP) – 1 PC ← (PC) + *rel*	–	–	–	–	–	–	REL	AD	rr	4
CBEQ *opr,rel* CBEQA #*opr,rel* CBEQX #*opr,rel* CBEQ *opr*,X+,*rel* CBEQ X+,*rel* CBEQ *opr*,SP,*rel*	Compare and Branch if Equal	PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (X) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 2 + rel ? (A) – (M) = $00 PC ← (PC) + 4 + rel ? (A) – (M) = $00	–	–	–	–	–	–	DIR IMM IMM IX1+ IX+ SP1	31 41 51 61 71 9E61	dd rr ii rr ii rr ff rr rr ff rr	5 4 4 5 4 6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask	I ← 0	–	–	0	–	–	–	INH	9A		2
CLR *opr* CLRA CLRX CLRH CLR *opr*,X CLR ,X CLR *opr*,SP	Clear	M ← $00 A ← $00 X ← $00 H ← $00 M ← $00 M ← $00 M ← $00	0	–	–	0	1	–	DIR INH INH INH IX1 IX SP1	3F 4F 5F 8C 6F 7F 9E6F	dd ff ff	3 1 1 1 3 2 4
CMP #*opr* CMP *opr* CMP *opr* CMP *opr*,X CMP *opr*,X CMP ,X CMP *opr*,SP CMP *opr*,SP	Compare A with M	(A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A1 B1 C1 D1 E1 F1 9EE1 9ED1	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
COM *opr* COMA COMX COM *opr*,X COM ,X COM *opr*,SP	Complement (One's Complement)	M ← ($\overline{M}$) = $FF – (M) A ← ($\overline{A}$) = $FF – (M) X ← ($\overline{X}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M)	0	–	–	↕	↕	1	DIR INH INH IX1 IX SP1	33 43 53 63 73 9E63	dd ff ff	4 1 1 4 3 5
CPHX #*opr* CPHX *opr*	Compare H:X with M	(H:X) – (M:M + 1)	↕	–	–	↕	↕	↕	IMM DIR	65 75	ii ii+1 dd	3 4
CPX #*opr* CPX *opr* CPX *opr* CPX ,X CPX *opr*,X CPX *opr*,X CPX *opr*,SP CPX *opr*,SP	Compare X with M	(X) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A3 B3 C3 D3 E3 F3 9EE3 9ED3	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
DAA	Decimal Adjust A	$(A)_{10}$	U	–	–	↕	↕	↕	INH	72		2

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
DBNZ *opr,rel* DBNZA *rel* DBNZX *rel* DBNZ *opr*,X,*rel* DBNZ X,*rel* DBNZ *opr*,SP,*rel*	Decrement and Branch if Not Zero	A ← (A)–1 or M ← (M)–1 or X ← (X)–1 PC ← (PC) + 3 + *rel* ? (result) ≠ 0 PC ← (PC) + 2 + *rel* ? (result) ≠ 0 PC ← (PC) + 2 + *rel* ? (result) ≠ 0 PC ← (PC) + 3 + *rel* ? (result) ≠ 0 PC ← (PC) + 2 + *rel* ? (result) ≠ 0 PC ← (PC) + 4 + *rel* ? (result) ≠ 0	–	–	–	–	–	–	DIR INH INH IX1 IX SP1	3B 4B 5B 6B 7B 9E6B	dd rr rr rr ff rr rr ff rr	5 3 3 5 4 6
DEC *opr* DECA DECX DEC *opr*,X DEC ,X DEC *opr*,SP	Decrement	M ← (M) – 1 A ← (A) – 1 X ← (X) – 1 M ← (M) – 1 M ← (M) – 1 M ← (M) – 1	↕	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3A 4A 5A 6A 7A 9E6A	dd ff ff	4 1 1 4 3 5
DIV	Divide	A ← (H:A)/(X) H ← Remainder	–	–	–	–	↕	↕	INH	52		7
EOR #*opr* EOR *opr* EOR *opr* EOR *opr*,X EOR *opr*,X EOR ,X EOR *opr*,SP EOR *opr*,SP	Exclusive OR M with A	A ← (A ⊕ M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A8 B8 C8 D8 E8 F8 9EE8 9ED8	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
INC *opr* INCA INCX INC *opr*,X INC ,X INC *opr*,SP	Increment	M ← (M) + 1 A ← (A) + 1 X ← (X) + 1 M ← (M) + 1 M ← (M) + 1 M ← (M) + 1	↕	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3C 4C 5C 6C 7C 9E6C	dd ff ff	4 1 1 4 3 5
JMP *opr* JMP *opr* JMP *opr*,X JMP *opr*,X JMP ,X	Jump	PC ← Jump Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff 	2 3 4 3 2
JSR *opr* JSR *opr* JSR *opr*,X JSR *opr*,X JSR ,X	Jump to Subroutine	PC ← (PC) + *n* (*n* = 1, 2, or 3) Push (PCL); SP ← (SP) – 1 Push (PCH); SP ← (SP) – 1 PC ← Unconditional Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff 	4 5 6 5 4
LDA #*opr* LDA *opr* LDA *opr* LDA *opr*,X LDA *opr*,X LDA ,X LDA *opr*,SP LDA *opr*,SP	Load A from M	A ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A6 B6 C6 D6 E6 F6 9EE6 9ED6	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
LDHX #*opr* LDHX *opr*	Load H:X from M	H:X ← (M:M + 1)	0	–	–	↕	↕	–	IMM DIR	45 55	ii jj dd	3 4

Table 4-1. Instruction Set Summary

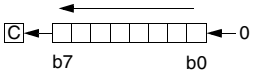
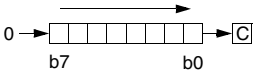
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
LDX #*opr* LDX *opr* LDX *opr* LDX *opr*,X LDX *opr*,X LDX ,X LDX *opr*,SP LDX *opr*,SP	Load X from M	X ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	AE BE CE DE EE FE 9EEE 9EDE	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
LSL *opr* LSLA LSLX LSL *opr*,X LSL ,X LSL *opr*,SP	Logical Shift Left (Same as ASL)		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	4 1 1 4 3 5
LSR *opr* LSRA LSR*X* LSR *opr*,X LSR ,X LSR *opr*,SP	Logical Shift Right		↕	–	–	0	↕	↕	DIR INH INH IX1 IX SP1	34 44 54 64 74 9E64	dd ff ff	4 1 1 4 3 5
MOV *opr*,*opr* MOV *opr*,X+ MOV #*opr*,*opr* MOV X+,*opr*	Move	$(M)_{Destination} \leftarrow (M)_{Source}$ H:X ← (H:X) + 1 (IX+D, DIX+)	0	–	–	↕	↕	–	DD DIX+ IMD IX+D	4E 5E 6E 7E	dd dd dd ii dd dd	5 4 4 4
MUL	Unsigned multiply	X:A ← (X) × (A)	–	0	–	–	–	0	INH	42		5
NEG *opr* NEGA NEGX NEG *opr*,X NEG ,X NEG *opr*,SP	Negate (Two's Complement)	M ← –(M) = $00 – (M) A ← –(A) = $00 – (A) X ← –(X) = $00 – (X) M ← –(M) = $00 – (M) M ← –(M) = $00 – (M)	↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	30 40 50 60 70 9E60	dd ff ff	4 1 1 4 3 5
NOP	No Operation	None	–	–	–	–	–	–	INH	9D		1
NSA	Nibble Swap A	A ← (A[3:0]:A[7:4])	–	–	–	–	–	–	INH	62		3
ORA #*opr* ORA *opr* ORA *opr* ORA *opr*,X ORA *opr*,X ORA ,X ORA *opr*,SP ORA *opr*,SP	Inclusive OR A and M	A ← (A) | (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	AA BA CA DA EA FA 9EEA 9EDA	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
PSHA	Push A onto Stack	Push (A); SP ← (SP) – 1	–	–	–	–	–	–	INH	87		2
PSHH	Push H onto Stack	Push (H); SP ← (SP) – 1	–	–	–	–	–	–	INH	8B		2
PSHX	Push X onto Stack	Push (X); SP ← (SP) – 1	–	–	–	–	–	–	INH	89		2
PULA	Pull A from Stack	SP ← (SP + 1); Pull (A)	–	–	–	–	–	–	INH	86		2

Table 4-1. Instruction Set Summary

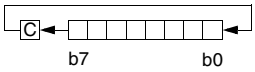
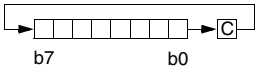
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
PULH	Pull H from Stack	SP ← (SP + 1); Pull (H)	–	–	–	–	–	–	INH	8A		2
PULX	Pull X from Stack	SP ← (SP + 1); Pull (X)	–	–	–	–	–	–	INH	88		2
ROL *opr* ROLA ROLX ROL *opr*,X ROL ,X ROL *opr*,SP	Rotate Left through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	39 49 59 69 79 9E69	dd ff ff	4 1 1 4 3 5
ROR *opr* RORA RORX ROR *opr*,X ROR ,X ROR *opr*,SP	Rotate Right through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	4 1 1 4 3 5
RSP	Reset Stack Pointer	SP ← $FF	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 1; Pull (CCR) SP ← (SP) + 1; Pull (A) SP ← (SP) + 1; Pull (X) SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	↕	↕	↕	↕	↕	↕	INH	80		7
RTS	Return from Subroutine	SP ← SP + 1; Pull (PCH) SP ← SP + 1; Pull (PCL)	–	–	–	–	–	–	INH	81		4
SBC #*opr* SBC *opr* SBC *opr* SBC *opr*,X SBC *opr*,X SBC ,X SBC *opr*,SP SBC *opr*,SP	Subtract with Carry	A ← (A) – (M) – (C)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A2 B2 C2 D2 E2 F2 9EE2 9ED2	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask	I ← 1	–	–	1	–	–	–	INH	9B		2
STA *opr* STA *opr* STA *opr*,X STA *opr*,X STA ,X STA *opr*,SP STA *opr*,SP	Store A in M	M ← (A)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	B7 C7 D7 E7 F7 9EE7 9ED7	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
STHX *opr*	Store H:X in M	(M:M + 1) ← (H:X)	0	–	–	↕	↕	–	DIR	35	dd	4
STOP	Enable $\overline{\text{IRQ}}$ Pin; Stop Oscillator	I ← 0; Stop Oscillator	–	–	0	–	–	–	INH	8E		1

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
STX *opr* STX *opr* STX *opr*,X STX *opr*,X STX ,X STX *opr*,SP STX *opr*,SP	Store X in M	M ← (X)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	BF CF DF EF FF 9EEF 9EDF	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
SUB #*opr* SUB *opr* SUB *opr* SUB *opr*,X SUB *opr*,X SUB ,X SUB *opr*,SP SUB *opr*,SP	Subtract	A ← (A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A0 B0 C0 D0 E0 F0 9EE0 9ED0	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
SWI	Software Interrupt	PC ← (PC) + 1; Push (PCL) SP ← (SP) – 1; Push (PCH) SP ← (SP) – 1; Push (X) SP ← (SP) – 1; Push (A) SP ← (SP) – 1; Push (CCR) SP ← (SP) – 1; I ← 1 PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		9
TAP	Transfer A to CCR	CCR ← (A)	↕	↕	↕	↕	↕	↕	INH	84		2
TAX	Transfer A to X	X ← (A)	–	–	–	–	–	–	INH	97		1
TPA	Transfer CCR to A	A ← (CCR)	–	–	–	–	–	–	INH	85		1
TST *opr* TSTA TSTX TST *opr*,X TST ,X TST *opr*,SP	Test for Negative or Zero	(A) – $00 or (X) – $00 or (M) – $00	0	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3D 4D 5D 6D 7D 9E6D	dd ff ff	3 1 1 3 2 4
TSX	Transfer SP to H:X	H:X ← (SP) + 1	–	–	–	–	–	–	INH	95		2
TXA	Transfer X to A	A ← (X)	–	–	–	–	–	–	INH	9F		1
TXS	Transfer H:X to SP	(SP) ← (H:X) – 1	–	–	–	–	–	–	INH	94		2

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				

A	Accumulator	*n*	Any bit
C	Carry/borrow bit	*opr*	Operand (one or two bytes)
CCR	Condition code register	PC	Program counter
dd	Direct address of operand	PCH	Program counter high byte
dd rr	Direct address of operand and relative offset of branch instruction	PCL	Program counter low byte
DD	Direct to direct addressing mode	REL	Relative addressing mode
DIR	Direct addressing mode	*rel*	Relative program counter offset byte
DIX+	Direct to indexed with post increment addressing mode	rr	Relative program counter offset byte
ee ff	High and low bytes of offset in indexed, 16-bit offset addressing	SP1	Stack pointer, 8-bit offset addressing mode
EXT	Extended addressing mode	SP2	Stack pointer 16-bit offset addressing mode
ff	Offset byte in indexed, 8-bit offset addressing	SP	Stack pointer
H	Half-carry bit	U	Undefined
H	Index register high byte	V	Overflow bit
hh ll	High and low bytes of operand address in extended addressing	X	Index register low byte
I	Interrupt mask	Z	Zero bit
ii	Immediate operand byte	&	Logical AND
IMD	Immediate source to direct destination addressing mode	|	Logical OR
IMM	Immediate addressing mode	⊕	Logical EXCLUSIVE OR
INH	Inherent addressing mode	( )	Contents of
IX	Indexed, no offset addressing mode	–( )	Negation (two's complement)
IX+	Indexed, no offset, post increment addressing mode	#	Immediate value
IX+D	Indexed with post increment to direct addressing mode	«	Sign extend
IX1	Indexed, 8-bit offset addressing mode	←	Loaded with
IX1+	Indexed, 8-bit offset, post increment addressing mode	?	If
IX2	Indexed, 16-bit offset addressing mode	:	Concatenated with
M	Memory location	↕	Set or cleared
N	Negative bit	—	Not affected

Table 4-2. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write						Control		Register/Mem				
	DIR	DIR	REL	DIR	INH	INH	IX1	SP1	IX	INH	INH	IMM	DIR	EXT	IX2	SP
MSB / LSB	0	1	2	3	4	5	6	9E6	7	8	9	A	B	C	D	9E
0	5 BRSET0 3 DIR	4 BSET0 2 DIR	3 BRA 2 REL	4 NEG 2 DIR	1 NEGA 1 INH	1 NEGX 1 INH	4 NEG 2 IX1	5 NEG 3 SP1	3 NEG 1 IX	7 RTI 1 INH	3 BGE 2 REL	2 SUB 2 IMM	3 SUB 2 DIR	4 SUB 3 EXT	4 SUB 3 IX2	SU 4
1	5 BRCLR0 3 DIR	4 BCLR0 2 DIR	3 BRN 2 REL	5 CBEQ 3 DIR	4 CBEQA 3 IMM	4 CBEQX 3 IMM	5 CBEQ 3 IX1+	6 CBEQ 4 SP1	4 CBEQ 2 IX+	4 RTS 1 INH	3 BLT 2 REL	2 CMP 2 IMM	3 CMP 2 DIR	4 CMP 3 EXT	4 CMP 3 IX2	CM 4
2	5 BRSET1 3 DIR	4 BSET1 2 DIR	3 BHI 2 REL		5 MUL 1 INH	7 DIV 1 INH	3 NSA 1 INH		2 DAA 1 INH		3 BGT 2 REL	2 SBC 2 IMM	3 SBC 2 DIR	4 SBC 3 EXT	4 SBC 3 IX2	SB 4
3	5 BRCLR1 3 DIR	4 BCLR1 2 DIR	3 BLS 2 REL	4 COM 2 DIR	1 COMA 1 INH	1 COMX 1 INH	4 COM 2 IX1	5 COM 3 SP1	3 COM 1 IX	9 SWI 1 INH	3 BLE 2 REL	2 CPX 2 IMM	3 CPX 2 DIR	4 CPX 3 EXT	4 CPX 3 IX2	CP 4
4	5 BRSET2 3 DIR	4 BSET2 2 DIR	3 BCC 2 REL	4 LSR 2 DIR	1 LSRA 1 INH	1 LSRX 1 INH	4 LSR 2 IX1	5 LSR 3 SP1	3 LSR 1 IX	2 TAP 1 INH	2 TXS 1 INH	2 AND 2 IMM	3 AND 2 DIR	4 AND 3 EXT	4 AND 3 IX2	AN 4
5	5 BRCLR2 3 DIR	4 BCLR2 2 DIR	3 BCS 2 REL	4 STHX 2 DIR	3 LDHX 3 IMM	4 LDHX 2 DIR	3 CPHX 3 IMM		4 CPHX 2 DIR	1 TPA 1 INH	2 TSX 1 INH	2 BIT 2 IMM	3 BIT 2 DIR	4 BIT 3 EXT	4 BIT 3 IX2	BI 4
6	5 BRSET3 3 DIR	4 BSET3 2 DIR	3 BNE 2 REL	4 ROR 2 DIR	1 RORA 1 INH	1 RORX 1 INH	4 ROR 2 IX1	5 ROR 3 SP1	3 ROR 1 IX	2 PULA 1 INH		2 LDA 2 IMM	3 LDA 2 DIR	4 LDA 3 EXT	4 LDA 3 IX2	LD 4
7	5 BRCLR3 3 DIR	4 BCLR3 2 DIR	3 BEQ 2 REL	4 ASR 2 DIR	1 ASRA 1 INH	1 ASRX 1 INH	4 ASR 2 IX1	5 ASR 3 SP1	3 ASR 1 IX	2 PSHA 1 INH	1 TAX 1 INH	2 AIS 2 IMM	3 STA 2 DIR	4 STA 3 EXT	4 STA 3 IX2	ST 4
8	5 BRSET4 3 DIR	4 BSET4 2 DIR	3 BHCC 2 REL	4 LSL 2 DIR	1 LSLA 1 INH	1 LSLX 1 INH	4 LSL 2 IX1	5 LSL 3 SP1	3 LSL 1 IX	2 PULX 1 INH	1 CLC 1 INH	2 EOR 2 IMM	3 EOR 2 DIR	4 EOR 3 EXT	4 EOR 3 IX2	EO 4
9	5 BRCLR4 3 DIR	4 BCLR4 2 DIR	3 BHCS 2 REL	4 ROL 2 DIR	1 ROLA 1 INH	1 ROLX 1 INH	4 ROL 2 IX1	5 ROL 3 SP1	3 ROL 1 IX	2 PSHX 1 INH	1 SEC 1 INH	2 ADC 2 IMM	3 ADC 2 DIR	4 ADC 3 EXT	4 ADC 3 IX2	AD 4
A	5 BRSET5 3 DIR	4 BSET5 2 DIR	3 BPL 2 REL	4 DEC 2 DIR	1 DECA 1 INH	1 DECX 1 INH	4 DEC 2 IX1	5 DEC 3 SP1	3 DEC 1 IX	2 PULH 1 INH	2 CLI 1 INH	2 ORA 2 IMM	3 ORA 2 DIR	4 ORA 3 EXT	4 ORA 3 IX2	OR 4
B	5 BRCLR5 3 DIR	4 BCLR5 2 DIR	3 BMI 2 REL	5 DBNZ 3 DIR	3 DBNZA 2 INH	3 DBNZX 2 INH	5 DBNZ 3 IX1	6 DBNZ 4 SP1	4 DBNZ 2 IX	2 PSHH 1 INH	2 SEI 1 INH	2 ADD 2 IMM	3 ADD 2 DIR	4 ADD 3 EXT	4 ADD 3 IX2	AD 4
C	5 BRSET6 3 DIR	4 BSET6 2 DIR	3 BMC 2 REL	4 INC 2 DIR	1 INCA 1 INH	1 INCX 1 INH	4 INC 2 IX1	5 INC 3 SP1	3 INC 1 IX	1 CLRH 1 INH	1 RSP 1 INH		2 JMP 2 DIR	3 JMP 3 EXT	4 JMP 3 IX2	
D	5 BRCLR6 3 DIR	4 BCLR6 2 DIR	3 BMS 2 REL	3 TST 2 DIR	1 TSTA 1 INH	1 TSTX 1 INH	3 TST 2 IX1	4 TST 3 SP1	2 TST 1 IX		1 NOP 1 INH	4 BSR 2 REL	4 JSR 2 DIR	5 JSR 3 EXT	6 JSR 3 IX2	
E	5 BRSET7 3 DIR	4 BSET7 2 DIR	3 BIL 2 REL		5 MOV 3 DD	4 MOV 2 DIX+	4 MOV 3 IMD		4 MOV 2 IX+D	1 STOP 1 INH	*	2 LDX 2 IMM	3 LDX 2 DIR	4 LDX 3 EXT	4 LDX 3 IX2	LD 4
F	5 BRCLR7 3 DIR	4 BCLR7 2 DIR	3 BIH 2 REL	3 CLR 2 DIR	1 CLRA 1 INH	1 CLRX 1 INH	3 CLR 2 IX1	4 CLR 3 SP1	2 CLR 1 IX	1 WAIT 1 INH	1 TXA 1 INH	2 AIX 2 IMM	3 STX 2 DIR	4 STX 3 EXT	4 STX 3 IX2	ST 4

INH Inherent
IMM Immediate
DIR Direct
EXT Extended
DD Direct-Direct
IX+D Indexed-Direct
REL Relative
IX Indexed, No Offset
IX1 Indexed, 8-Bit Offset
IX2 Indexed, 16-Bit Offset
IMD Immediate-Direct
DIX+ Direct-Indexed
SP1 Stack Pointer, 8-Bit Offset
SP2 Stack Pointer, 16-Bit Offset
IX+ Indexed, No Offset with Post Increment
IX1+ Indexed, 1-Byte Offset with Post Increment

*Pre-byte for stack pointer indexed instructions

MSB / LSB	0	High Byte of O
0	5 BRSET0 3 DIR	Cycles Opcode Mnem Number of Byt

Low Byte of Opcode in Hexadecimal

Chapter 5
System Integration Module (SIM)

5.1 Introduction

This section describes the system integration module (SIM), which supports up to 24 external and/or internal interrupts. Together with the CPU, the SIM controls all MCU activities. A block diagram of the SIM is shown in Figure 5-1. Figure 5-2 is a summary of the SIM I/O registers. The SIM is a system state controller that coordinates CPU and exception timing.

The SIM is responsible for:

- Bus clock generation and control for CPU and peripherals
  - Stop/wait/reset/break entry and recovery
  - Internal clock control
- Master reset control, including power-on reset (POR) and COP timeout
- Interrupt control:
  - Acknowledge timing
  - Arbitration control timing
  - Vector address generation
- CPU enable/disable timing
- Modular architecture expandable to 128 interrupt sources

Table 5-1 shows the internal signal names used in this section.

Table 5-1. Signal Name Conventions

Signal Name	Description
ICLK	Internal oscillator clock
OSCOUT	The XTAL or RC frequency divided by two. This signal is again divided by two in the SIM to generate the internal bus clocks. (Bus clock = OSCOUT ÷ 2)
IAB	Internal address bus
IDB	Internal data bus
PORRST	Signal from the power-on reset module to the SIM
IRST	Internal reset signal
R/$\overline{W}$	Read/write signal

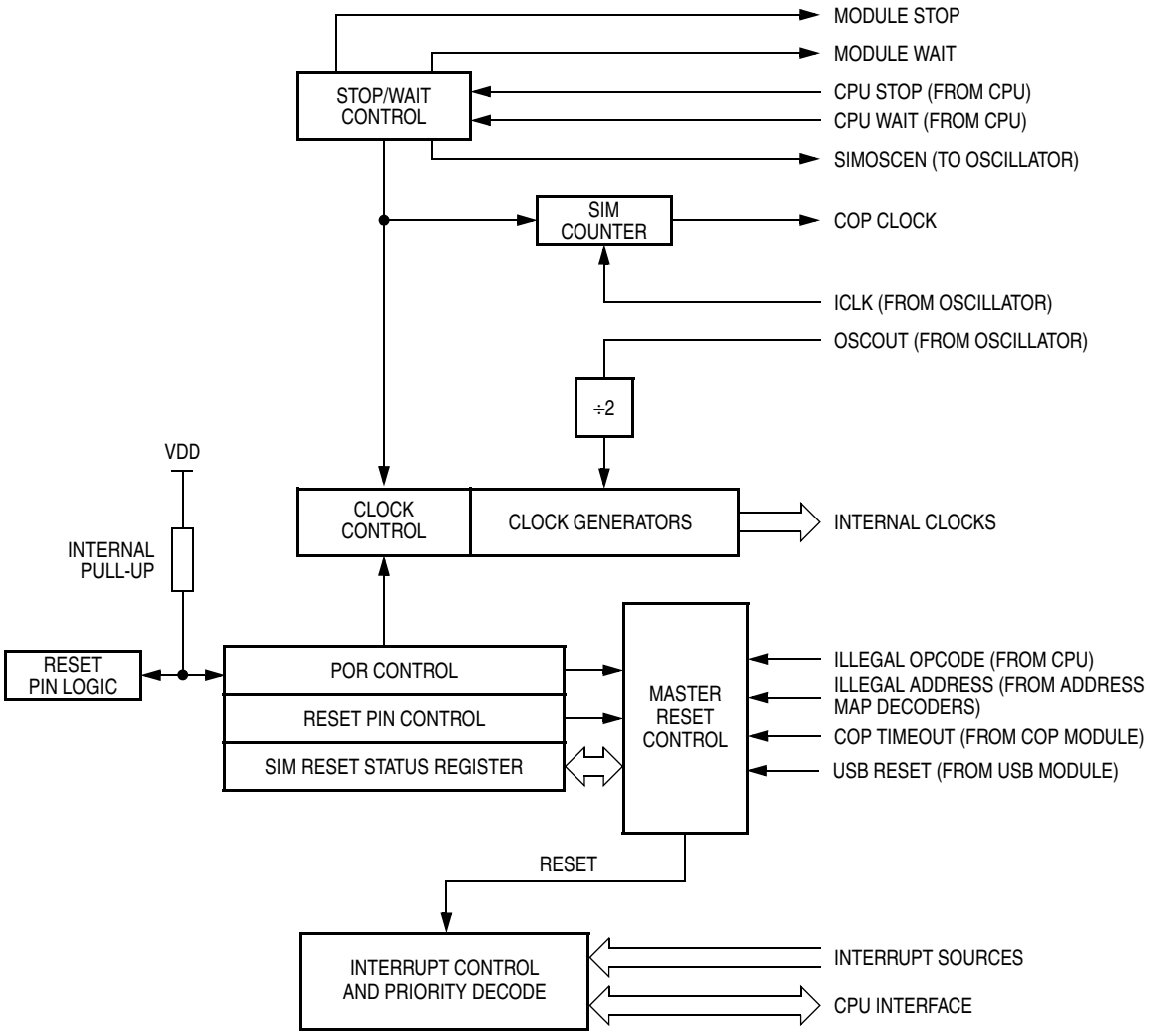


Figure 5-1. SIM Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	Break Status Register (BSR)	Read:	R	R	R	R	R	R	SBSW	R
		Write:							NOTE	
		Reset:	0	0	0	0	0	0	0	0
	Note: Writing a logic 0 clears SBSW.									
$FE01	Reset Status Register (RSR)	Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
		Write:								
		POR:	1	0	0	0	0	0	0	0
$FE02	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$FE03	Break Flag Control Register (BFCR)	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							

Figure 5-2. SIM I/O Register Summary

$FE04	Interrupt Status Register 1 (INT1)	Read:	IF6	IF5	IF4	IF3	0	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE05	Interrupt Status Register 2 (INT2)	Read:	IF14	IF13	IF12	IF11	0	0	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE06	Interrupt Status Register 3 (INT3)	Read:	0	0	0	0	0	0	0	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 5-2. SIM I/O Register Summary

5.2 SIM Bus Clock Control and Generation

The bus clock generator provides system clock signals for the CPU and peripherals on the MCU. The system clocks are generated from an incoming clock, OSCOUT, as shown in Figure 5-3.

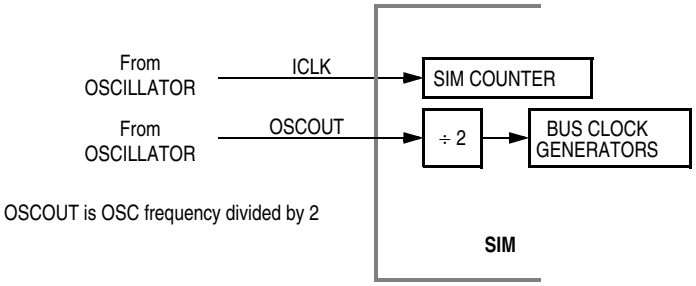


Figure 5-3. SIM Clock Signals

5.2.1 Bus Timing

In user mode, the internal bus frequency is the oscillator frequency divided by four.

5.2.2 Clock Start-Up from POR or LVI Reset

When the power-on reset module or the low-voltage inhibit module generates a reset, the clocks to the CPU and peripherals are inactive and held in an inactive phase until after the 4096 ICLK cycle POR timeout has completed. The $\overline{RST}$ pin is driven low by the SIM during this entire period. The IBUS clocks start upon completion of the timeout.

5.2.3 Clocks in Stop Mode and Wait Mode

Upon exit from stop mode by an interrupt, break, or reset, the SIM allows ICLK to clock the SIM counter. The CPU and peripheral clocks do not become active until after the stop delay time-out. This time-out is selectable as 4096 or 32 ICLK cycles. (See 5.6.2 Stop Mode.)

In wait mode, the CPU clocks are inactive. The SIM also produces two sets of clocks for other modules. Refer to the wait mode subsection of each module to see if the module is active or inactive in wait mode. Some modules can be programmed to be active in wait mode.

5.3 Reset and System Initialization

The MCU has these reset sources:

- Power-on reset module (POR)
- External reset pin ($\overline{RST}$)
- Computer operating properly module (COP)
- Low-voltage inhibit module (LVI)
- Illegal opcode
- Illegal address

All of these resets produce the vector $FFFE–$FFFF ($FEFE–$FEFF in Monitor mode) and assert the internal reset signal (IRST). IRST causes all registers to be returned to their default values and all modules to be returned to their reset states.

An internal reset clears the SIM counter (see 5.4 SIM Counter), but an external reset does not. Each of the resets sets a corresponding bit in the reset status register (RSR). (See 5.7 SIM Registers.)

5.3.1 External Pin Reset

The $\overline{RST}$ pin circuits include an internal pull-up device. Pulling the asynchronous $\overline{RST}$ pin low halts all processing. The PIN bit of the reset status register (RSR) is set as long as $\overline{RST}$ is held low for a minimum of 67 ICLK cycles, assuming that the POR was not the source of the reset. See Table 5-2 for details. Figure 5-4 shows the relative timing.

Table 5-2. PIN Bit Set Timing

Reset Type	Number of Cycles Required to Set PIN
POR	4163 (4096 + 64 + 3)
All others	67 (64 + 3)

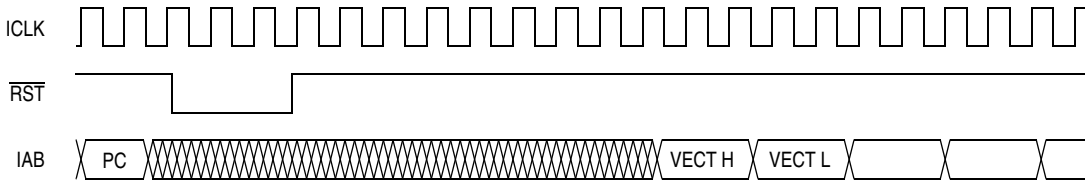


Figure 5-4. External Reset Timing

5.3.2 Active Resets from Internal Sources

All internal reset sources actively pull the $\overline{RST}$ pin low for 32 ICLK cycles to allow resetting of external peripherals. The internal reset signal IRST continues to be asserted for an additional 32 cycles (Figure 5-5). An internal reset can be caused by an illegal address, illegal opcode, COP time-out, or POR. (See Figure 5-6 . Sources of Internal Reset.) Note that for POR resets, the SIM cycles through 4096 ICLK cycles during which the SIM forces the $\overline{RST}$ pin low. The internal reset signal then follows the sequence from the falling edge of $\overline{RST}$ shown in Figure 5-5.

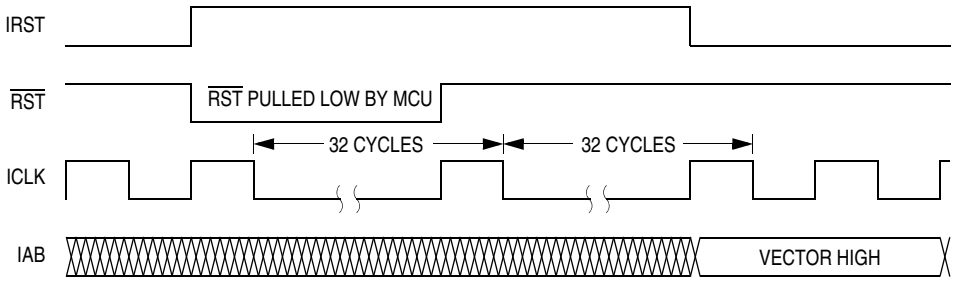


Figure 5-5. Internal Reset Timing

The COP reset is asynchronous to the bus clock.

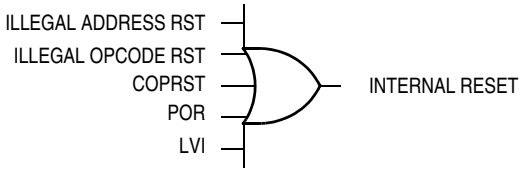


Figure 5-6. Sources of Internal Reset

The active reset feature allows the part to issue a reset to peripherals and other chips within a system built around the MCU.

5.3.2.1 Power-On Reset

When power is first applied to the MCU, the power-on reset module (POR) generates a pulse to indicate that power-on has occurred. The external reset pin ($\overline{RST}$) is held low while the SIM counter counts out 4096 ICLK cycles. Sixty-four ICLK cycles later, the CPU and memories are released from reset to allow the reset vector sequence to occur.

At power-on, the following events occur:

- A POR pulse is generated.
- The internal reset signal is asserted.
- The SIM enables OSCOUT.
- Internal clocks to the CPU and modules are held inactive for 4096 ICLK cycles to allow stabilization of the oscillator.
- The $\overline{RST}$ pin is driven low during the oscillator stabilization time.
- The POR bit of the reset status register (RSR) is set and all other bits in the register are cleared.

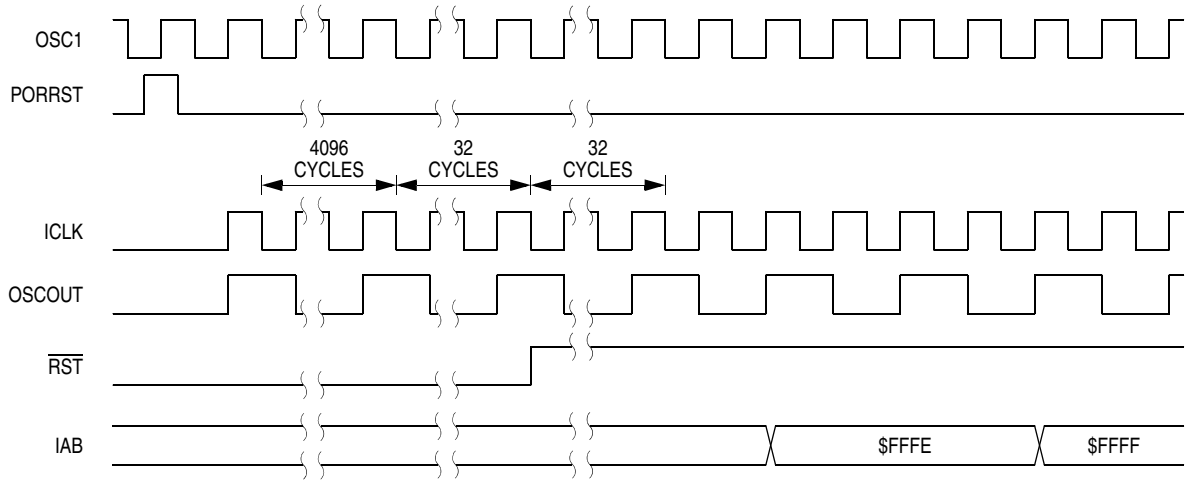


Figure 5-7. POR Recovery

5.3.2.2 Computer Operating Properly (COP) Reset

An input to the SIM is reserved for the COP reset signal. The overflow of the COP counter causes an internal reset and sets the COP bit in the reset status register (RSR). The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

To prevent a COP module time-out, write any value to location $FFFF. Writing to location $FFFF clears the COP counter and stages 12 through 5 of the SIM counter. The SIM counter output, which occurs at least every ($2^{12} - 2^4$) ICLK cycles, drives the COP counter. The COP should be serviced as soon as possible out of reset to guarantee the maximum amount of time before the first time-out.

The COP module is disabled if the $\overline{\text{RST}}$ pin or the $\overline{\text{IRQ}}$ pin is held at V_{TST} while the MCU is in monitor mode. The COP module can be disabled only through combinational logic conditioned with the high voltage signal on the $\overline{\text{RST}}$ or the $\overline{\text{IRQ}}$ pin. This prevents the COP from becoming disabled as a result of external noise. During a break state, V_{TST} on the $\overline{\text{RST}}$ pin disables the COP module.

5.3.2.3 Illegal Opcode Reset

The SIM decodes signals from the CPU to detect illegal instructions. An illegal instruction sets the ILOP bit in the reset status register (RSR) and causes a reset.

If the stop enable bit, STOP, in the mask option register is logic zero, the SIM treats the STOP instruction as an illegal opcode and causes an illegal opcode reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

5.3.2.4 Illegal Address Reset

An opcode fetch from an unmapped address generates an illegal address reset. The SIM verifies that the CPU is fetching an opcode prior to asserting the ILAD bit in the reset status register (RSR) and resetting the MCU. A data fetch from an unmapped address does not generate a reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

5.3.2.5 Low-Voltage Inhibit (LVI) Reset

The low-voltage inhibit module (LVI) asserts its output to the SIM when the V_{DD} voltage falls to the LVI trip voltage V_{TRIP}. The LVI bit in the reset status register (RSR) is set, and the external reset pin ($\overline{\text{RST}}$) is

held low while the SIM counter counts out 4096 ICLK cycles. Sixty-four ICLK cycles later, the CPU and memories are released from reset to allow the reset vector sequence to occur. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

5.4 SIM Counter

The SIM counter is used by the power-on reset module (POR) and in stop mode recovery to allow the oscillator time to stabilize before enabling the internal bus (IBUS) clocks. The SIM counter also serves as a prescaler for the computer operating properly module (COP). The SIM counter uses 12 stages for counting, followed by a 13th stage that triggers a reset of SIM counters and supplies the clock for the COP module. The SIM counter is clocked by the falling edge of ICLK.

5.4.1 SIM Counter During Power-On Reset

The power-on reset module (POR) detects power applied to the MCU. At power-on, the POR circuit asserts the signal PORRST. Once the SIM is initialized, it enables the oscillator to drive the bus clock state machine.

5.4.2 SIM Counter During Stop Mode Recovery

The SIM counter also is used for stop mode recovery. The STOP instruction clears the SIM counter. After an interrupt, break, or reset, the SIM senses the state of the short stop recovery bit, SSREC, in the mask option register. If the SSREC bit is a logic one, then the stop recovery is reduced from the normal delay of 4096 ICLK cycles down to 32 ICLK cycles. This is ideal for applications using canned oscillators that do not require long start-up times from stop mode. External crystal applications should use the full stop recovery time, that is, with SSREC cleared in the configuration register 1 (CONFIG1).

5.4.3 SIM Counter and Reset States

External reset has no effect on the SIM counter. (See 5.6.2 Stop Mode for details.) The SIM counter is free-running after all reset states. (See 5.3.2 Active Resets from Internal Sources for counter control and internal reset recovery sequences.)

5.5 Exception Control

Normal, sequential program execution can be changed in three different ways:

- Interrupts
  - Maskable hardware CPU interrupts
  - Non-maskable software interrupt instruction (SWI)
- Reset
- Break interrupts

5.5.1 Interrupts

An interrupt temporarily changes the sequence of program execution to respond to a particular event. Figure 5-8 flow charts the handling of system interrupts.

Interrupts are latched, and arbitration is performed in the SIM at the start of interrupt processing. The arbitration result is a constant that the CPU uses to determine which vector to fetch. Once an interrupt is latched by the SIM, no other interrupt can take precedence, regardless of priority, until the latched interrupt is serviced (or the I bit is cleared).

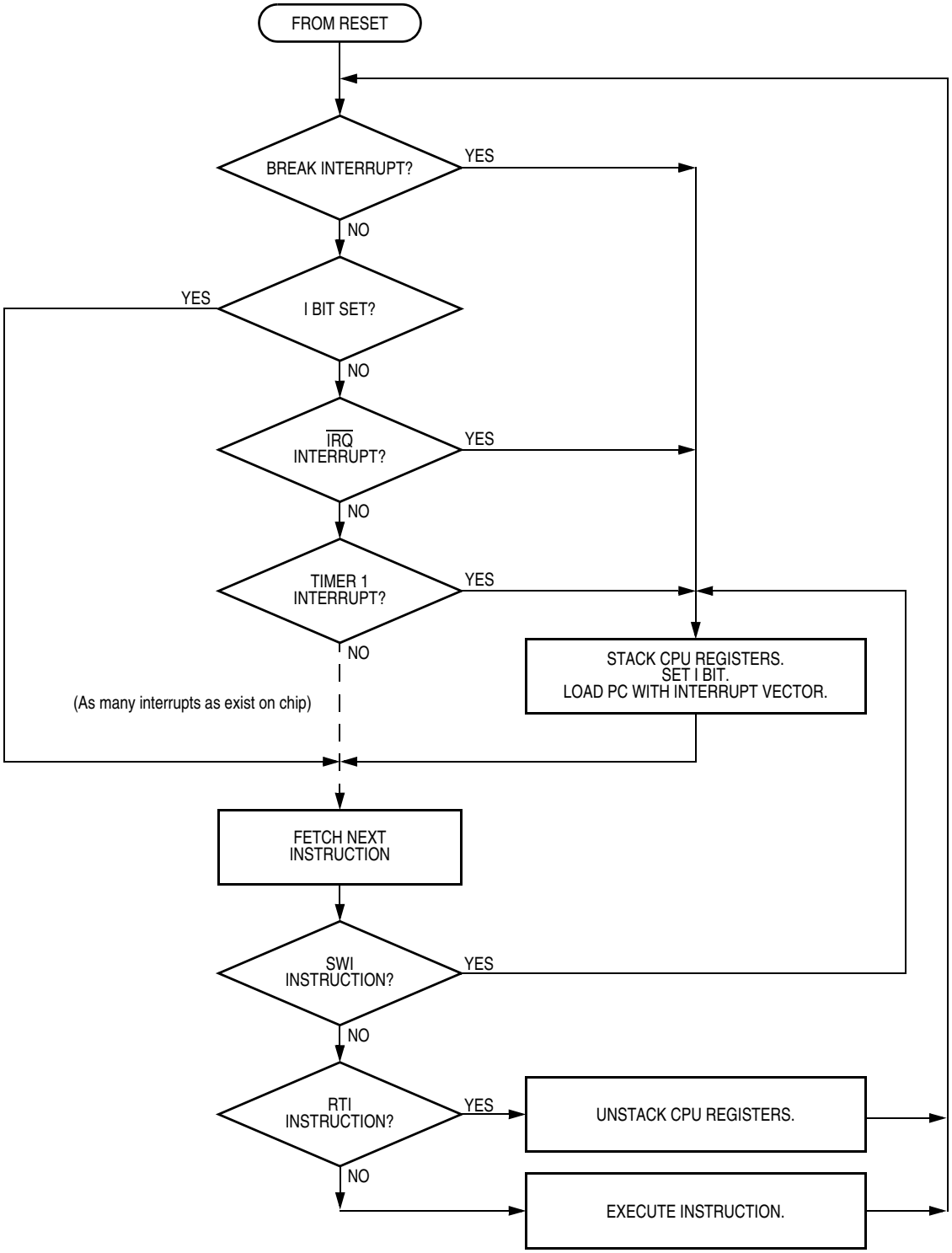


Figure 5-8. Interrupt Processing

At the beginning of an interrupt, the CPU saves the CPU register contents on the stack and sets the interrupt mask (I bit) to prevent additional interrupts. At the end of an interrupt, the RTI instruction recovers the CPU register contents from the stack so that normal processing can resume. Figure 5-9 shows interrupt entry timing.
Figure 5-10 shows interrupt recovery timing.

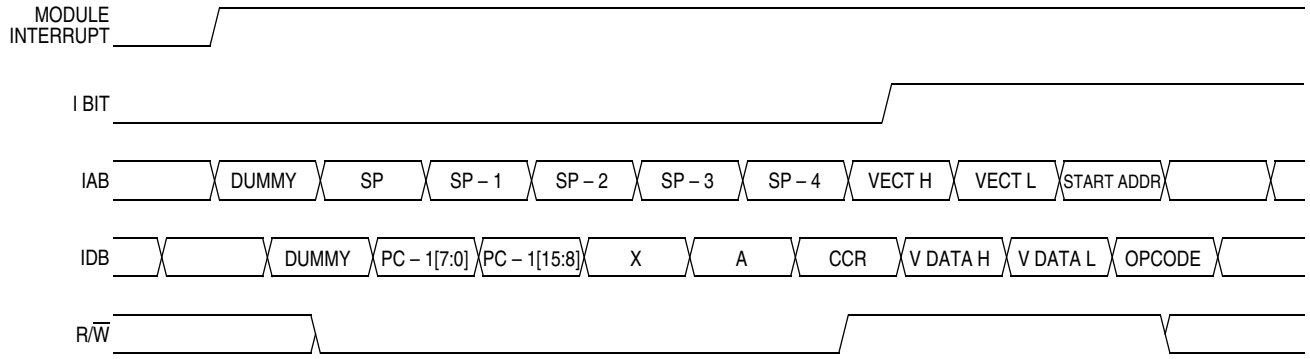


Figure 5-9. Interrupt Entry

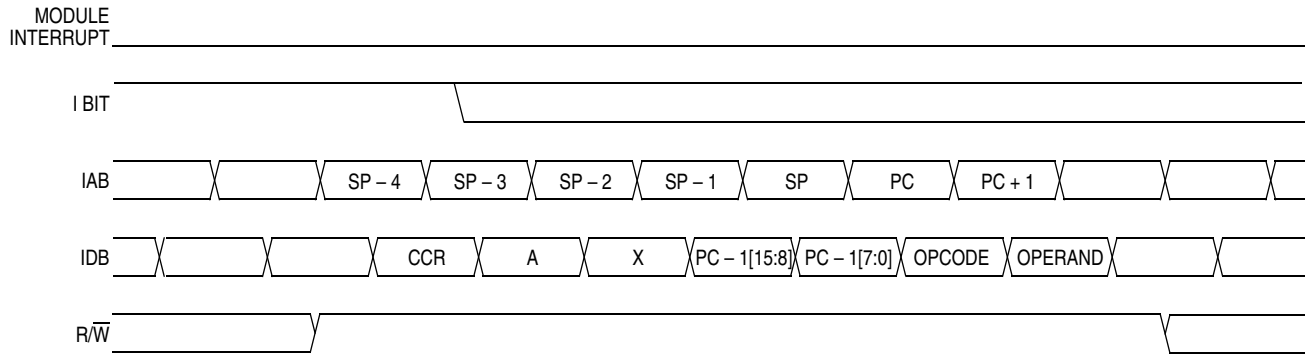


Figure 5-10. Interrupt Recovery

5.5.1.1 Hardware Interrupts

A hardware interrupt does not stop the current instruction. Processing of a hardware interrupt begins after completion of the current instruction. When the current instruction is complete, the SIM checks all pending hardware interrupts. If interrupts are not masked (I bit clear in the condition code register), and if the corresponding interrupt enable bit is set, the SIM proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If more than one interrupt is pending at the end of an instruction execution, the highest priority interrupt is serviced first. Figure 5-11 demonstrates what happens when two interrupts are pending. If an interrupt is pending upon exit from the original interrupt service routine, the pending interrupt is serviced before the LDA instruction is executed.

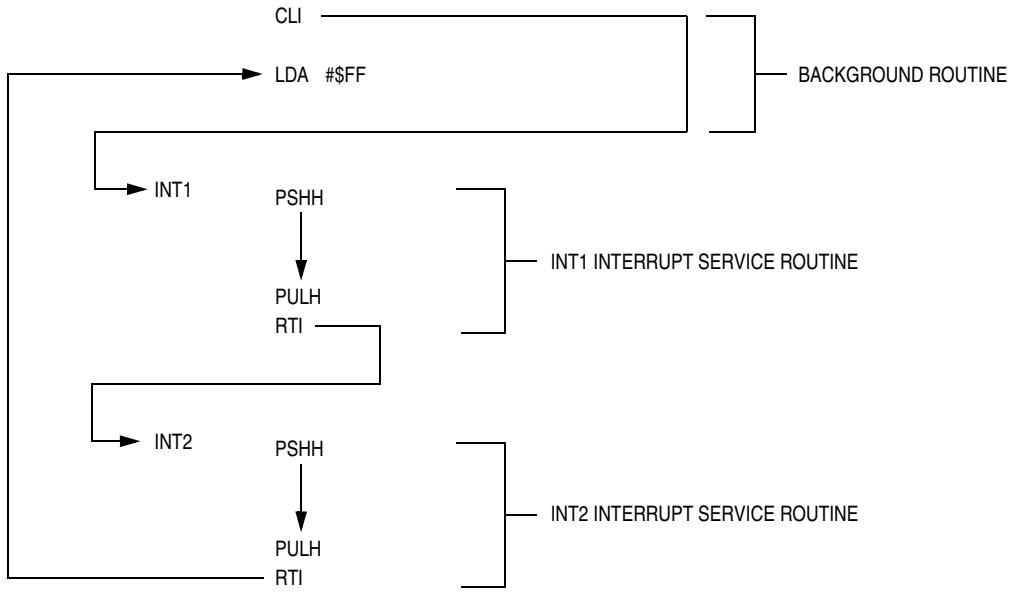


Figure 5-11. Interrupt Recognition Example

The LDA opcode is prefetched by both the INT1 and INT2 RTI instructions. However, in the case of the INT1 RTI prefetch, this is a redundant operation.

NOTE

To maintain compatibility with the M6805 Family, the H register is not pushed on the stack during interrupt entry. If the interrupt service routine modifies the H register or uses the indexed addressing mode, software should save the H register and then restore it prior to exiting the routine.

5.5.1.2 SWI Instruction

The SWI instruction is a non-maskable instruction that causes an interrupt regardless of the state of the interrupt mask (I bit) in the condition code register.

NOTE

A software interrupt pushes PC onto the stack. A software interrupt does not push PC – 1, as a hardware interrupt does.

5.5.2 Interrupt Status Registers

The flags in the interrupt status registers identify maskable interrupt sources. Table 5-3 summarizes the interrupt sources and the interrupt status register flags that they set. The interrupt status registers can be useful for debugging.

Table 5-3. Interrupt Sources

Priority	Source	Flag	Mask (1)	INT Flag	Vector Address
Highest	Reset	—	—	—	$FFFE–$FFFF
	SWI Instruction	—	—	—	$FFFC–$FFFD
	$\overline{\text{IRQ}}$ Pin	IRQF	IMASK	IF1	$FFFA–$FFFB
	Timer 1 Channel 0 Interrupt	CH0F	CH0IE	IF3	$FFF6–$FFF7
	Timer 1 Channel 1 Interrupt	CH1F	CH1IE	IF4	$FFF4–$FFF5
	Timer 1 Overflow Interrupt	TOF	TOIE	IF5	$FFF2–$FFF3
	Timer 2 Channel 0 Interrupt	CH0F	CH0IE	IF6	$FFF0–$FFF1
	Timer 2 Channel 1 Interrupt	CH1F	CH1IE	IF7	$FFEE–$FFEF
	Timer 2 Overflow Interrupt	TOF	TOIE	IF8	$FFEC–$FFED
	SCI Error	OR NF FE PE	ORIE NEIE FEIE PEIE	IF11	$FFE6–$FFE7
	SCI Receive	SCRF IDLE	SCRIE ILIE	IF12	$FFE4–$FFE5
	SCI Transmit	SCTE TC	SCTIE TCIE	IF13	$FFE2–$FFE3
	Keyboard Interrupt	KEYF	IMASKK	IF14	$FFE0–$FFE1
Lowest	ADC Conversion Complete Interrupt	COCO	AIEN	IF15	$FFDE–$FFDF

1. The I bit in the condition code register is a global mask for all interrupts sources except the SWI instruction.

5.5.2.1 Interrupt Status Register 1

Address: $FE04

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	IF6	IF5	IF4	IF3	0	IF1	0	0
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 5-12. Interrupt Status Register 1 (INT1)

IF1, IF3 to IF6 — Interrupt Flags

These flags indicate the presence of interrupt requests from the sources shown in Table 5-3.

1 = Interrupt request present

0 = No interrupt request present

Bit 0, 1, and 3 — Always read 0

5.5.2.2 Interrupt Status Register 2

Address: $FE05

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	IF14	IF13	IF12	IF11	0	0	IF8	IF7
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 5-13. Interrupt Status Register 2 (INT2)

IF7, IF8, IF11 to F14 — Interrupt Flags

This flag indicates the presence of interrupt requests from the sources shown in Table 5-3.

1 = Interrupt request present
0 = No interrupt request present

Bit 2 and 3 — Always read 0

5.5.2.3 Interrupt Status Register 3

Address: $FE06

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	IF15
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 5-14. Interrupt Status Register 3 (INT3)

IF15 — Interrupt Flags

These flags indicate the presence of interrupt requests from the sources shown in Table 5-3.

1 = Interrupt request present
0 = No interrupt request present

Bit 1 to 7 — Always read 0

5.5.3 Reset

All reset sources always have equal and highest priority and cannot be arbitrated.

5.5.4 Break Interrupts

The break module can stop normal program flow at a software-programmable break point by asserting its break interrupt output. (See Chapter 16 Break Module (BREAK).) The SIM puts the CPU into the break state by forcing it to the SWI vector location. Refer to the break interrupt subsection of each module to see how each module is affected by the break state.

5.5.5 Status Flag Protection in Break Mode

The SIM controls whether status flags contained in other modules can be cleared during break mode. The user can select whether flags are protected from being cleared by properly initializing the break clear flag enable bit (BCFE) in the break flag control register (BFCR).

Protecting flags in break mode ensures that set flags will not be cleared while in break mode. This protection allows registers to be freely read and written during break mode without losing status flag information.

Setting the BCFE bit enables the clearing mechanisms. Once cleared in break mode, a flag remains cleared even when break mode is exited. Status flags with a two-step clearing mechanism — for example, a read of one register followed by the read or write of another — are protected, even when the first step is accomplished prior to entering break mode. Upon leaving break mode, execution of the second step will clear the flag as normal.

5.6 Low-Power Modes

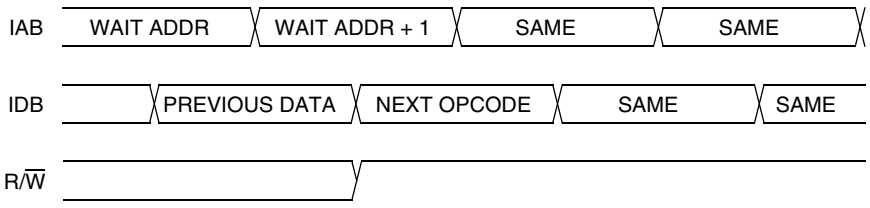
Executing the WAIT or STOP instruction puts the MCU in a low-power-consumption mode for standby situations. The SIM holds the CPU in a non-clocked state. The operation of each of these modes is described below. Both STOP and WAIT clear the interrupt mask (I) in the condition code register, allowing interrupts to occur.

5.6.1 Wait Mode

In wait mode, the CPU clocks are inactive while the peripheral clocks continue to run. Figure 5-15 shows the timing for wait mode entry.

A module that is active during wait mode can wake up the CPU with an interrupt if the interrupt is enabled. Stacking for the interrupt begins one cycle after the WAIT instruction during which the interrupt occurred. In wait mode, the CPU clocks are inactive. Refer to the wait mode subsection of each module to see if the module is active or inactive in wait mode. Some modules can be programmed to be active in wait mode.

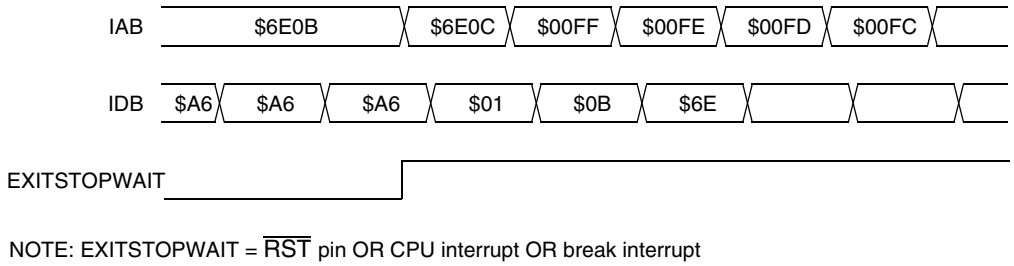
Wait mode can also be exited by a reset or break. A break interrupt during wait mode sets the SIM break stop/wait bit, SBSW, in the break status register (BSR). If the COP disable bit, COPD, in the mask option register is logic zero, then the computer operating properly module (COP) is enabled and remains active in wait mode.



NOTE: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

Figure 5-15. Wait Mode Entry Timing

Figure 5-16 and Figure 5-17 show the timing for WAIT recovery.



NOTE: EXITSTOPWAIT = $\overline{\text{RST}}$ pin OR CPU interrupt OR break interrupt

Figure 5-16. Wait Recovery from Interrupt or Break

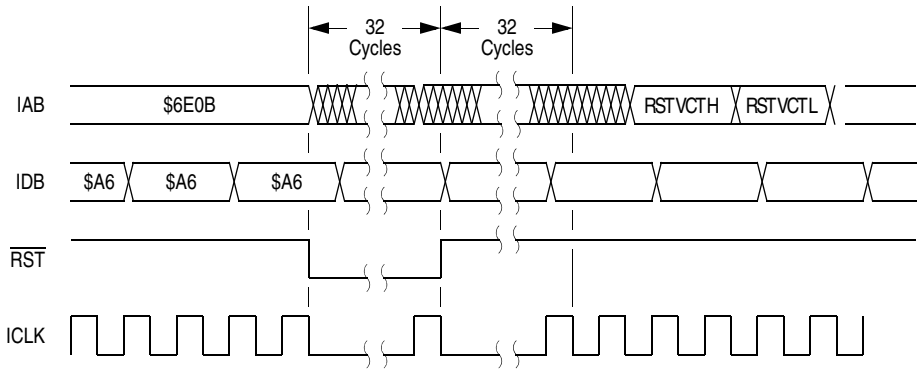


Figure 5-17. Wait Recovery from Internal Reset

5.6.2 Stop Mode

In stop mode, the SIM counter is reset and the system clocks are disabled. An interrupt request from a module can cause an exit from stop mode. Stacking for interrupts begins after the selected stop recovery time has elapsed. Reset or break also causes an exit from stop mode.

The SIM disables the oscillator signals (OSCOUT) in stop mode, stopping the CPU and peripherals. Stop recovery time is selectable using the SSREC bit in the configuration register 1 (CONFIG1). If SSREC is set, stop recovery is reduced from the normal delay of 4096 ICLK cycles down to 32. This is ideal for applications using canned oscillators that do not require long start-up times from stop mode.

NOTE

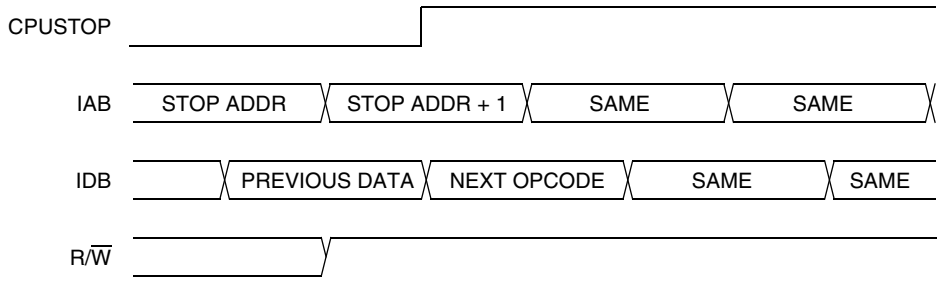
External crystal applications should use the full stop recovery time by clearing the SSREC bit.

A break interrupt during stop mode sets the SIM break stop/wait bit (SBSW) in the break status register (BSR).

The SIM counter is held in reset from the execution of the STOP instruction until the beginning of stop recovery. It is then used to time the recovery period. Figure 5-18 shows stop mode entry timing.

NOTE

To minimize stop current, all pins configured as inputs should be driven to a logic 1 or logic 0.



NOTE: Previous data can be operand data or the STOP opcode, depending on the last instruction.

Figure 5-18. Stop Mode Entry Timing

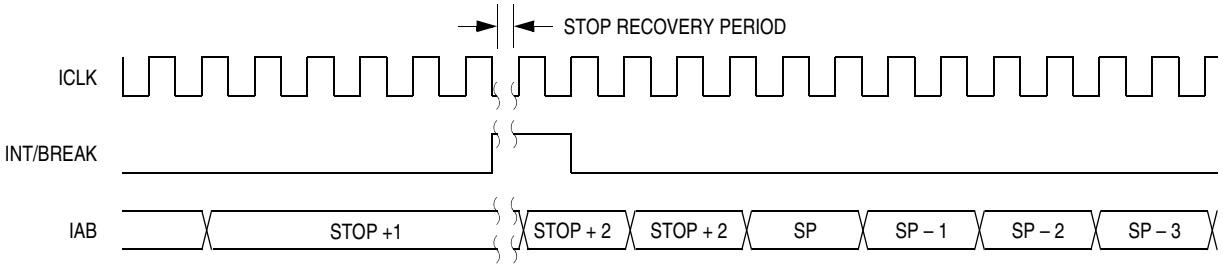


Figure 5-19. Stop Mode Recovery from Interrupt or Break

5.7 SIM Registers

The SIM has three memory mapped registers.

- Break Status Register (BSR)
- Reset Status Register (RSR)
- Break Flag Control Register (BFCR)

5.7.1 Break Status Register (BSR)

The break status register contains a flag to indicate that a break caused an exit from stop or wait mode.

Address: $FE00

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R	R	R	R	R	R	SBSW	R
Write:							Note(1)	
Reset:	0	0	0	0	0	0	0	0

R = Reserved

1. Writing a logic zero clears SBSW.

Figure 5-20. Break Status Register (BSR)

SBSW — SIM Break Stop/Wait

This status bit is useful in applications requiring a return to wait or stop mode after exiting from a break interrupt. Clear SBSW by writing a logic zero to it. Reset clears SBSW.

1 = Stop mode or wait mode was exited by break interrupt
0 = Stop mode or wait mode was not exited by break interrupt

SBSW can be read within the break state SWI routine. The user can modify the return address on the stack by subtracting one from it. The following code is an example of this. Writing zero to the SBSW bit clears it.

```
; This code works if the H register has been pushed onto the stack in the break
; service routine software. This code should be executed at the end of the
; break service routine software.

  HIBYTE   EQU    5

  LOBYTE   EQU    6

;          If not SBSW, do RTI

           BRCLR  SBSW,BSR, RETURN   ; See if wait mode or stop mode was exited
                                     ; by break.

           TST    LOBYTE,SP          ; If RETURNLO is not zero,

           BNE    DOLO               ; then just decrement low byte.

           DEC    HIBYTE,SP          ; Else deal with high byte, too.

  DOLO     DEC    LOBYTE,SP          ; Point to WAIT/STOP opcode.

  RETURN   PULH                      ; Restore H register.
           RTI
```

5.7.2 Reset Status Register (RSR)

This register contains six flags that show the source of the last reset. Clear the SIM reset status register by reading it. A power-on reset sets the POR bit and clears all other bits in the register.

Address: $FE01

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
Write:								
POR:	1	0	0	0	0	0	0	0

= Unimplemented

Figure 5-21. Reset Status Register (RSR)

POR — Power-On Reset Bit

1 = Last reset caused by POR circuit
0 = Read of RSR

PIN — External Reset Bit

1 = Last reset caused by external reset pin ($\overline{RST}$)
0 = POR or read of RSR

COP — Computer Operating Properly Reset Bit

1 = Last reset caused by COP counter
0 = POR or read of RSR

ILOP — Illegal Opcode Reset Bit

1 = Last reset caused by an illegal opcode
0 = POR or read of RSR

ILAD — Illegal Address Reset Bit (opcode fetches only)

1 = Last reset caused by an opcode fetch from an illegal address
0 = POR or read of RSR

MODRST — Monitor Mode Entry Module Reset bit

1 = Last reset caused by monitor mode entry when vector locations $FFFE and $FFFF are $FF after POR while $\overline{IRQ}$ = V_{DD}
0 = POR or read of RSR

LVI — Low Voltage Inhibit Reset bit

1 = Last reset caused by LVI circuit
0 = POR or read of RSR

5.7.3 Break Flag Control Register (BFCR)

The break control register contains a bit that enables software to clear status bits while the MCU is in a break state.

Address: $FE03

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

R = Reserved

Figure 5-22. Break Flag Control Register (BFCR)

BCFE — Break Clear Flag Enable Bit

This read/write bit enables software to clear status bits by accessing status registers while the MCU is in a break state. To clear status bits during the break state, the BCFE bit must be set.

1 = Status bits clearable during break
0 = Status bits not clearable during break

Chapter 6
Oscillator (OSC)

6.1 Introduction

The oscillator module provides the reference clocks for the MCU system and bus. Two oscillators are running on the device:

Selectable oscillator — for bus clock

- Crystal oscillator (XTAL) — built-in oscillator that requires an external crystal or ceramic-resonator. This option also allows an external clock that can be driven directly into OSC1.
- RC oscillator (RC) — built-in oscillator that requires an external resistor-capacitor connection only.

The selected oscillator is used to drive the bus clock, the SIM, and other modules on the MCU. The oscillator type is selected by programming a bit FLASH memory. The RC and crystal oscillator cannot run concurrently; one is disabled while the other is selected; because the RC and XTAL circuits share the same OSC1 pin.

Non-selectable oscillator — for COP

- Internal oscillator — built-in RC oscillator that requires no external components.

This internal oscillator is used to drive the computer operating properly (COP) module and the SIM. The internal oscillator runs continuously after a POR or reset, and is always available.

6.2 Oscillator Selection

The oscillator type is selected by programming a bit in a FLASH memory location; the mask option register (MOR), at $FFD0.
(See 3.5 Mask Option Register (MOR).)

NOTE

On the ROM device, the oscillator is selected by a ROM-mask layer at factory.

Address: $FFD0

	Bit 7	6	5	4	3	2	1	Bit 0
Read: / Write:	OSCSEL	R	R	R	R	R	R	R
Erased:	1	1	1	1	1	1	1	1
Reset:	Unaffected by reset							

Non-volatile FLASH register; write by programming.

R = Reserved

Figure 6-1. Mask Option Register (MOR)

OSCSEL — Oscillator Select Bit

OSCSEL selects the oscillator type for the MCU. The erased or unprogrammed state of this bit is logic 1, selecting the crystal oscillator option. This bit is unaffected by reset.

1 = Crystal oscillator
0 = RC oscillator

Bits 6–0 — Should be left as logic 1's.

NOTE

When Crystal oscillator is selected, the OSC2/RCCLK/PTA6/KBI6 pin is used as OSC2; other functions such as PTA6/KBI6 will not be available.

6.2.1 XTAL Oscillator

The XTAL oscillator circuit is designed for use with an external crystal or ceramic resonator to provide accurate clock source.

In its typical configuration, the XTAL oscillator is connected in a Pierce oscillator configuration, as shown in Figure 6-2. This figure shows only the logical representation of the internal components and may not represent actual circuitry. The oscillator configuration uses five components:

- Crystal, X_1
- Fixed capacitor, C_1
- Tuning capacitor, C_2 (can also be a fixed capacitor)
- Feedback resistor, R_B
- Series resistor, R_S (optional)

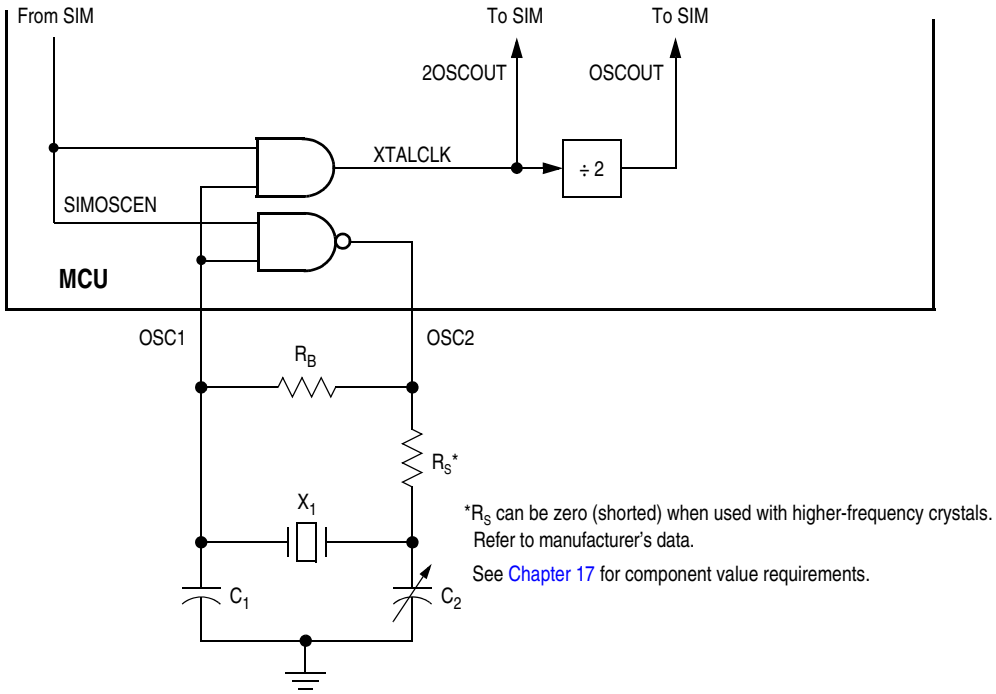


Figure 6-2. XTAL Oscillator External Connections

The series resistor (R_S) is included in the diagram to follow strict Pierce oscillator guidelines and may not be required for all ranges of operation, especially with high frequency crystals. Refer to the crystal manufacturer's data for more information.

6.2.2 RC Oscillator

The RC oscillator circuit is designed for use with external resistor and capacitor to provide a clock source with tolerance less than 10%.

In its typical configuration, the RC oscillator requires two external components, one R and one C. Component values should have a tolerance of 1% or less, to obtain a clock source with less than 10% tolerance. The oscillator configuration uses two components:

- C_{EXT}
- R_{EXT}

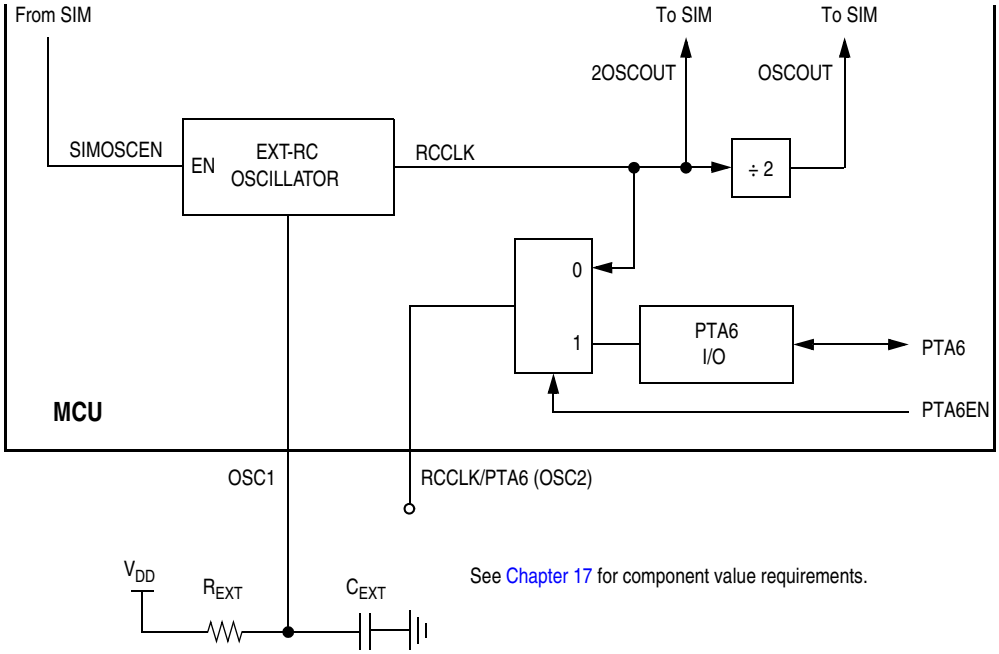


Figure 6-3. RC Oscillator External Connections

6.3 Internal Oscillator

The internal oscillator clock (ICLK) is a free running 50-kHz clock that requires no external components. It is used as the reference clock input to the computer operating properly (COP) module and the SIM.

The internal oscillator by default is always available and is free running after POR or reset. It can be stopped in stop mode by setting the STOP_ICLKDIS bit before executing the STOP instruction.

Figure 6-4 shows the logical representation of components of the internal oscillator circuitry.

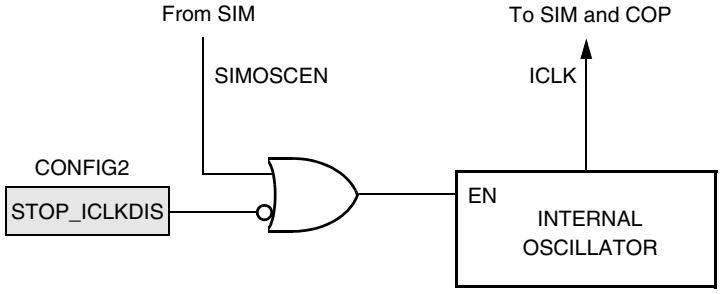


Figure 6-4. Internal Oscillator

NOTE

The internal oscillator is a free running oscillator and is available after each POR or reset. It is turned-off in stop mode by setting the STOP_ICLKDIS bit in CONFIG2 (see 3.4 Configuration Register 2 (CONFIG2)).

6.4 I/O Signals

The following paragraphs describe the oscillator I/O signals.

6.4.1 Crystal Amplifier Input Pin (OSC1)

OSC1 pin is an input to the crystal oscillator amplifier or the input to the RC oscillator circuit.

6.4.2 Crystal Amplifier Output Pin (OSC2/RCCLK/PTA6/KBI6)

For the XTAL oscillator**,** OSC2 pin is the output of the crystal oscillator inverting amplifier.

For the RC oscillator, OSC2 pin can be configured as a general purpose I/O pin PTA6, or the output of the RC oscillator, RCCLK.

Oscillator	OSC2 pin function
XTAL	Inverting OSC1
RC	Controlled by PTA6EN bit in PTAPUE ($000D) PTA6EN = 0: RCCLK output PTA6EN = 1: PTA6/KBI6

6.4.3 Oscillator Enable Signal (SIMOSCEN)

The SIMOSCEN signal comes from the system integration module (SIM) and enables/disables the XTAL oscillator circuit or the RC-oscillator.

6.4.4 XTAL Oscillator Clock (XTALCLK)

XTALCLK is the XTAL oscillator output signal. It runs at the full speed of the crystal (f_{XCLK}) and comes directly from the crystal oscillator circuit. Figure 6-2 shows only the logical relation of XTALCLK to OSC1 and OSC2 and may not represent the actual circuitry. The duty cycle of XTALCLK is unknown and may depend on the crystal and other external factors. Also, the frequency and amplitude of XTALCLK can be unstable at start-up.

6.4.5 RC Oscillator Clock (RCCLK)

RCCLK is the RC oscillator output signal. Its frequency is directly proportional to the time constant of the external R and C. Figure 6-3 shows only the logical relation of RCCLK to OSC1 and may not represent the actual circuitry.

6.4.6 Oscillator Out 2 (2OSCOUT)

2OSCOUT is same as the input clock (XTALCLK or RCCLK). This signal is driven to the SIM module.

6.4.7 Oscillator Out (OSCOUT)

The frequency of this signal is equal to half of the 2OSCOUT, this signal is driven to the SIM for generation of the bus clocks used by the CPU and other modules on the MCU. OSCOUT will be divided again in the SIM and results in the internal bus frequency being one fourth of the XTALCLK or RCCLK frequency.

6.4.8 Internal Oscillator Clock (ICLK)

ICLK is the internal oscillator output signal (typically 50-kHz), for the COP module and the SIM. Its frequency depends on the V_{DD} voltage. (See Chapter 17 Electrical Specifications for ICLK parameters.)

6.5 Low Power Modes

The WAIT and STOP instructions put the MCU in low-power consumption standby modes.

6.5.1 Wait Mode

The WAIT instruction has no effect on the oscillator logic. OSCOUT, 2OSCOUT, and ICLK continues to drive to the SIM module.

6.5.2 Stop Mode

The STOP instruction disables the XTALCLK or the RCCLK output, hence, OSCOUT and 2OSCOUT are disabled.

The STOP instruction also turns off the ICLK input to the COP module if the STOP_ICLKDIS bit is set in configuration register 2 (CONFIG2). After reset, the STOP_ICLKDIS bit is clear by default and ICLK is enabled during stop mode.

6.6 Oscillator During Break Mode

The OSCOUT, 2OSCOUT, and ICLK clocks continue to be driven out when the device enters the break state.

Chapter 7
Monitor ROM (MON)

7.1 Introduction

This section describes the monitor ROM (MON) and the monitor mode entry methods. The monitor ROM allows complete testing of the MCU through a single-wire interface with a host computer. This mode is also used for programming and erasing of FLASH memory in the MCU. Monitor mode entry can be achieved without use of the higher test voltage, V_{TST}, as long as vector addresses $FFFE and $FFFF are blank, thus reducing the hardware requirements for in-circuit programming.

7.2 Features

Features of the monitor ROM include the following:

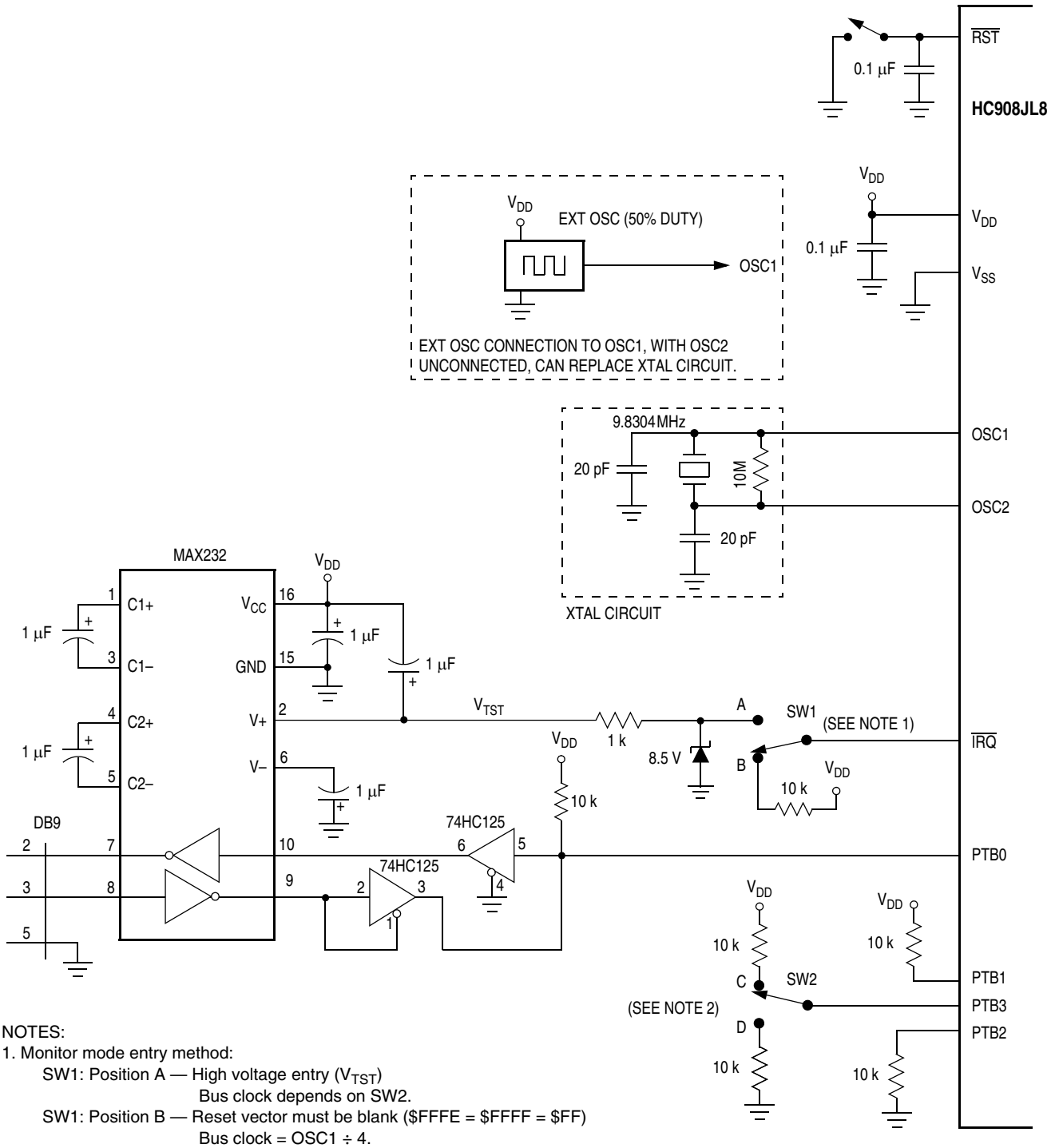
- Normal user-mode pin functionality
- One pin dedicated to serial communication between monitor ROM and host computer
- Standard mark/space non-return-to-zero (NRZ) communication with host computer
- Execution of code in RAM or FLASH
- FLASH memory security feature(1)
- FLASH memory programming interface
- 959 bytes monitor ROM code size
- Monitor mode entry without high voltage, V_{TST}, if reset vector is blank ($FFFE and $FFFF contain $FF)
- Standard monitor mode entry if high voltage, V_{TST}, is applied to $\overline{IRQ}$
- Resident routines for FLASH programming and EEPROM emulation

7.3 Functional Description

The monitor ROM receives and executes commands from a host computer. Figure 7-1 shows a example circuit used to enter monitor mode and communicate with a host computer via a standard RS-232 interface.

Simple monitor commands can access any memory address. In monitor mode, the MCU can execute host-computer code in RAM while most MCU pins retain normal operating mode functions. All communication between the host computer and the MCU is through the PTB0 pin. A level-shifting and multiplexing interface is required between PTB0 and the host computer. PTB0 is used in a wired-OR configuration and requires a pull-up resistor.

1. No security feature is absolutely secure. However, Motorola's strategy is to make reading or copying the FLASH difficult for unauthorized users.



NOTES:
1. Monitor mode entry method:
   SW1: Position A — High voltage entry (V_{TST})
   Bus clock depends on SW2.
   SW1: Position B — Reset vector must be blank ($FFFE = $FFFF = $FF)
   Bus clock = OSC1 ÷ 4.
2. Affects high voltage entry to monitor mode only (SW1 at position A):
   SW2: Position C — Bus clock = OSC1 ÷ 4
   SW2: Position D — Bus clock = OSC1 ÷ 2
5. See Table 17-4 for V_{TST} voltage level requirements.

Figure 7-1. Monitor Mode Circuit

7.3.1 Entering Monitor Mode

Table 7-1 shows the pin conditions for entering monitor mode. As specified in the table, monitor mode may be entered after a POR.

Communication at 9600 baud will be established provided one of the following sets of conditions is met:

1. If $\overline{IRQ}$ = V_{TST}:
   - Clock on OSC1 is 4.9125MHz
   - PTB3 = low
2. If $\overline{IRQ}$ = V_{TST}:
   - Clock on OSC1 is 9.8304MHz
   - PTB3 = high
3. If $FFFE and $FFFF are blank (contain $FF):
   - Clock on OSC1 is 9.8304MHz
   - $\overline{IRQ}$ = V_{DD}

Table 7-1. Monitor Mode Entry Requirements and Options

$\overline{IRQ}$	$FFFE and $FFFF	PTB3	PTB2	PTB1	PTB0	OSC1 Clock(1)	Bus Frequency	Comments
V_{TST}(2)	X	0	0	1	1	4.9152MHz	2.4576MHz	High voltage entry to monitor mode. 9600 baud communication on PTB0. COP disabled.
V_{TST}(1)	X	1	0	1	1	9.8304MHz	2.4576MHz	
V_{DD}	BLANK (contain $FF)	X	X	X	1	9.8304MHz	2.4576MHz	Blank reset vector (low-voltage) entry to monitor mode. 9600 baud communication on PTB0. COP disabled.
V_{DD}	NOT BLANK	X	X	X	X	X	OSC1 ÷ 4	Enters User mode.

1. RC oscillator cannot be used for monitor mode; must use either external oscillator or XTAL oscillator circuit.
2. See Table 17-4 for V_{TST} voltage level requirements.

If V_{TST} is applied to $\overline{IRQ}$ and PTB3 is low upon monitor mode entry (Table 7-1 condition set 1), the bus frequency is a divide-by-two of the clock input to OSC1. If PTB3 is high with V_{TST} applied to $\overline{IRQ}$ upon monitor mode entry (Table 7-1 condition set 2), the bus frequency is a divide-by-four of the clock input to OSC1. Holding the PTB3 pin low when entering monitor mode causes a bypass of a divide-by-two stage at the oscillator *only if V_{TST} is applied to $\overline{IRQ}$.* In this event, the OSCOUT frequency is equal to the 2OSCOUT frequency, and OSC1 input directly generates internal bus clocks. In this case, the OSC1 signal must have a 50% duty cycle at maximum bus frequency.

Entering monitor mode with V_{TST} on $\overline{IRQ}$, the COP is disabled as long as V_{TST} is applied to either $\overline{IRQ}$ or $\overline{RST}$. (See Chapter 5 System Integration Module (SIM) for more information on modes of operation.)

If entering monitor mode without high voltage on $\overline{IRQ}$ and reset vector being blank ($FFFE and $FFFF) (Table 7-1 condition set 3, where applied voltage is V_{DD}), then all port B pin requirements and conditions,

including the PTB3 frequency divisor selection, are not in effect. This is to reduce circuit requirements when performing in-circuit programming.

Entering monitor mode with the reset vector being blank, the COP is always disabled regardless of the state of $\overline{IRQ}$ or the $\overline{RST}$.

Figure 7-2. shows a simplified diagram of the monitor mode entry when the reset vector is blank and $\overline{IRQ}$ = V_{DD}. An OSC1 frequency of 9.8304MHz is required for a baud rate of 9600.

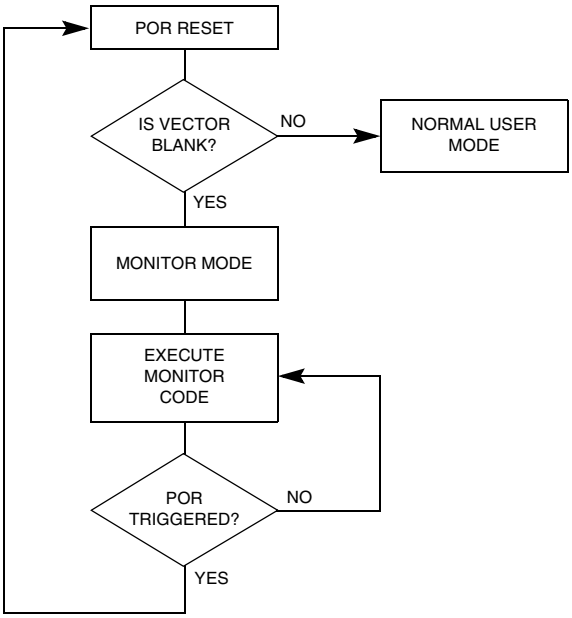


Figure 7-2. Low-Voltage Monitor Mode Entry Flowchart

Enter monitor mode with the pin configuration shown above by pulling $\overline{RST}$ low and then high. The rising edge of $\overline{RST}$ latches monitor mode. Once monitor mode is latched, the values on the specified pins can change.

Once out of reset, the MCU waits for the host to send eight security bytes. (See **7.4 Security**.) After the security bytes, the MCU sends a break signal (10 consecutive logic zeros) to the host, indicating that it is ready to receive a command. The break signal also provides a timing reference to allow the host to determine the necessary baud rate.

In monitor mode, the MCU uses different vectors for reset, SWI, and break interrupt. The alternate vectors are in the $FE page instead of the $FF page and allow code execution from the internal monitor firmware instead of user code.

Table 7-2 is a summary of the vector differences between user mode and monitor mode.

Table 7-2. Monitor Mode Vector Differences

Modes	Functions						
	COP	Reset Vector High	Reset Vector Low	Break Vector High	Break Vector Low	SWI Vector High	SWI Vector Low
User	Enabled	$FFFE	$FFFF	$FFFC	$FFFD	$FFFC	$FFFD
Monitor	Disabled(1)	$FEFE	$FEFF	$FEFC	$FEFD	$FEFC	$FEFD

Notes:
1. If the high voltage (V_{TST}) is removed from the $\overline{IRQ}$ pin or the $\overline{RST}$ pin, the SIM asserts its COP enable output. The COP is a mask option enabled or disabled by the COPD bit in the configuration register.

When the host computer has completed downloading code into the MCU RAM, the host then sends a RUN command, which executes an RTI, which sends control to the address on the stack pointer.

7.3.2 Baud Rate

The communication baud rate is dependant on oscillator frequency. The state of PTB3 also affects baud rate if entry to monitor mode is by $\overline{IRQ}$ = V_{TST}. When PTB3 is high, the divide by ratio is 1024. If the PTB3 pin is at logic zero upon entry into monitor mode, the divide by ratio is 512.

Table 7-3. Monitor Baud Rate Selection

Monitor Mode Entry By:	OSC1 Clock Frequency	PTB3	Baud Rate
$\overline{IRQ}$ = V_{TST}	4.9152 MHz	0	9600 bps
	9.8304 MHz	1	9600 bps
	4.9152 MHz	1	4800 bps
Blank reset vector, $\overline{IRQ}$ = V_{DD}	9.8304 MHz	X	9600 bps
	4.9152 MHz	X	4800 bps

7.3.3 Data Format

Communication with the monitor ROM is in standard non-return-to-zero (NRZ) mark/space data format. (See Figure 7-3 and Figure 7-4.)

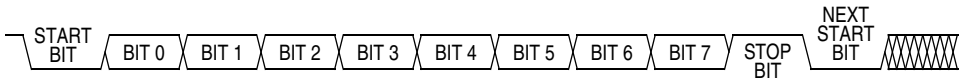


Figure 7-3. Monitor Data Format

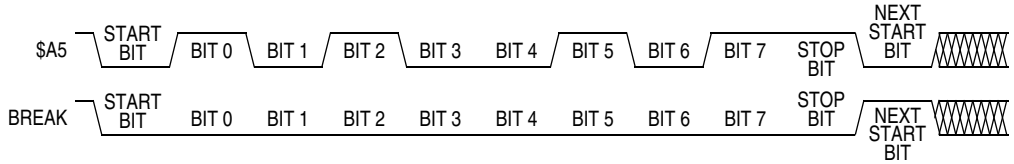


Figure 7-4. Sample Monitor Waveforms

The data transmit and receive rate can be anywhere from 4800 baud to 28.8 k-baud. Transmit and receive baud rates must be identical.

7.3.4 Echoing

As shown in Figure 7-5, the monitor ROM immediately echoes each received byte back to the PTB0 pin for error checking.

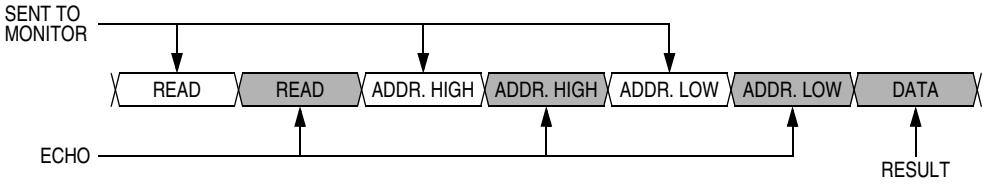


Figure 7-5. Read Transaction

Any result of a command appears after the echo of the last byte of the command.

7.3.5 Break Signal

A start bit followed by nine low bits is a break signal. (See **Figure 7-6**.) When the monitor receives a break signal, it drives the PTB0 pin high for the duration of two bits before echoing the break signal.

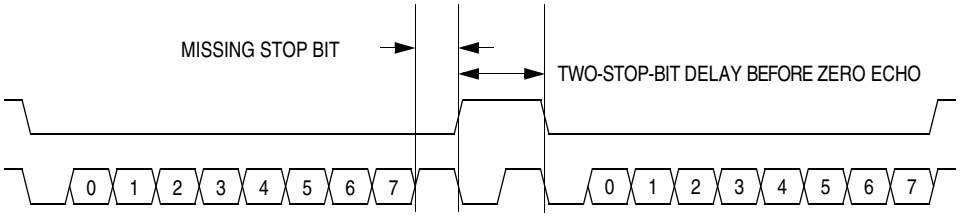


Figure 7-6. Break Transaction

7.3.6 Commands

The monitor ROM uses the following commands:

- READ (read memory)
- WRITE (write memory)
- IREAD (indexed read)
- IWRITE (indexed write)
- READSP (read stack pointer)
- RUN (run user program)

Table 7-4. READ (Read Memory) Command

Description	Read byte from memory
Operand	Specifies 2-byte address in high byte:low byte order
Data Returned	Returns contents of specified address
Opcode	$4A
Command Sequence	

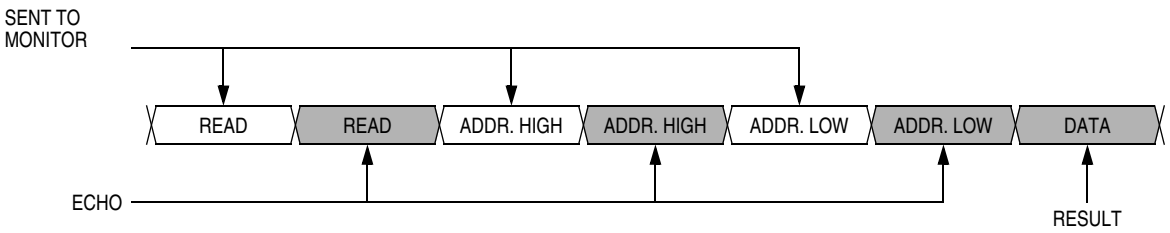


Table 7-5. WRITE (Write Memory) Command

Description	Write byte to memory
Operand	Specifies 2-byte address in high byte:low byte order; low byte followed by data byte
Data Returned	None
Opcode	$49
Command Sequence	

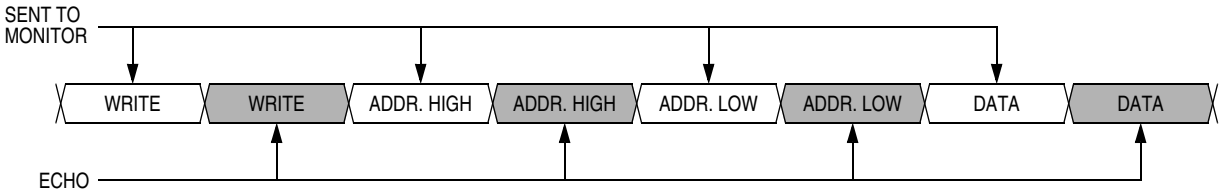


Table 7-6. IREAD (Indexed Read) Command

Description	Read next 2 bytes in memory from last address accessed
Operand	Specifies 2-byte address in high byte:low byte order
Data Returned	Returns contents of next two addresses
Opcode	$1A
Command Sequence	

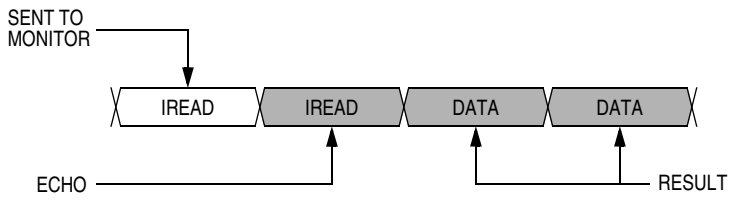
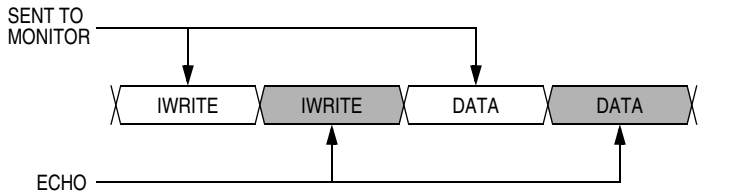


Table 7-7. IWRITE (Indexed Write) Command

Description	Write to last address accessed + 1
Operand	Specifies single data byte
Data Returned	None
Opcode	$19
Command Sequence	



NOTE

A sequence of IREAD or IWRITE commands can sequentially access a block of memory over the full 64-Kbyte memory map.

Table 7-8. READSP (Read Stack Pointer) Command

Description	Reads stack pointer
Operand	None
Data Returned	Returns stack pointer in high byte:low byte order
Opcode	$0C
Command Sequence	

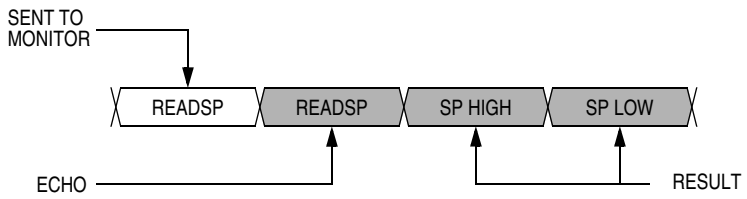
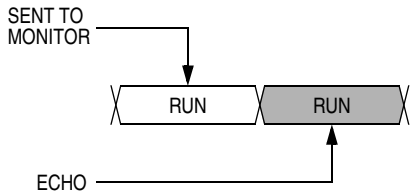


Table 7-9. RUN (Run User Program) Command

Description	Executes RTI instruction
Operand	None
Data Returned	None
Opcode	$28
Command Sequence	



7.4 Security

A security feature discourages unauthorized reading of FLASH locations while in monitor mode. The host can bypass the security feature at monitor mode entry by sending eight security bytes that match the bytes at locations $FFF6–$FFFD. Locations $FFF6–$FFFD contain user-defined data.

NOTE

Do not leave locations $FFF6–$FFFD blank. For security reasons, program locations $FFF6–$FFFD even if they are not used for vectors.

During monitor mode entry, the MCU waits after the power-on reset for the host to send the eight security bytes on pin PTB0. If the received bytes match those at locations $FFF6–$FFFD, the host bypasses the security feature and can read all FLASH locations and execute code from FLASH. Security remains bypassed until a power-on reset occurs. If the reset was not a power-on reset, security remains bypassed and security code entry is not required. (See Figure 7-7.)

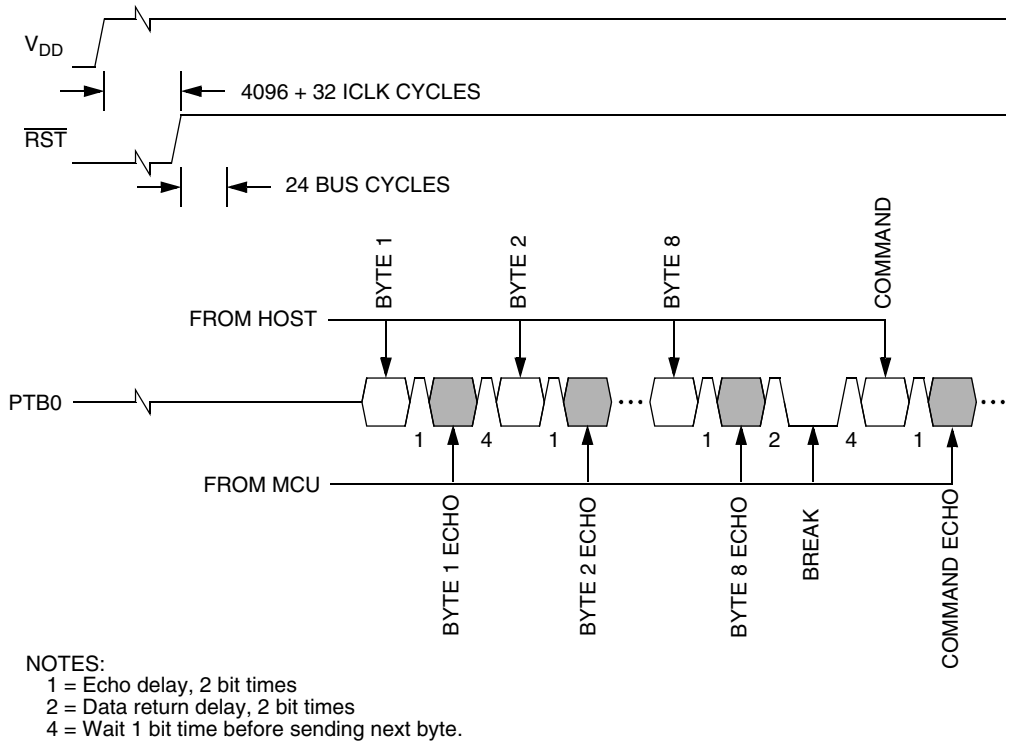


Figure 7-7. Monitor Mode Entry Timing

Upon power-on reset, if the received bytes of the security code do not match the data at locations $FFF6–$FFFD, the host fails to bypass the security feature. The MCU remains in monitor mode, but reading a FLASH location returns an invalid value and trying to execute code from FLASH causes an illegal address reset. After receiving the eight security bytes from the host, the MCU transmits a break character, signifying that it is ready to receive a command.

NOTE

The MCU does not transmit a break character until after the host sends the eight security bytes.

To determine whether the security code entered is correct, check to see if bit 6 of RAM address $60 is set. If it is, then the correct security code has been entered and FLASH can be accessed.

If the security sequence fails, the device should be reset by a power-on reset and brought up in monitor mode to attempt another entry. After failing the security sequence, the FLASH module can also be mass erased by executing an erase routine that was downloaded into internal RAM. The mass erase operation clears the security code locations so that all eight security bytes become $FF (blank).

7.5 ROM-Resident Routines

Eight routines stored in the monitor ROM area (thus ROM-resident) are provided for FLASH memory manipulation. Six of the eight routines are intended to simplify FLASH program, erase, and load operations. The other two routines are intended to simplify the use of the FLASH memory as EEPROM. **Table 7-10** shows a summary of the ROM-resident routines.

Table 7-10. Summary of ROM-Resident Routines

Routine Name	Routine Description	Call Address	Stack Used[1] (bytes)
PRGRNGE	Program a range of locations	$FC06	15
ERARNGE	Erase a page or the entire array	$FCBE	9
LDRNGE	Loads data from a range of locations	$FF30	9
MON_PRGRNGE	Program a range of locations in monitor mode	$FF28	17
MON_ERARNGE	Erase a page or the entire array in monitor mode	$FF2C	11
MON_LDRNGE	Loads data from a range of locations in monitor mode	$FF24	11
EE_WRITE	Emulated EEPROM write. Data size ranges from 2 to 15 bytes at a time.	$FD3F	24
EE_READ	Emulated EEPROM read. Data size ranges from 2 to 15 bytes at a time.	$FDD0	16

1. The listed stack size excludes the 2 bytes used by the calling instruction, JSR.

The routines are designed to be called as stand-alone subroutines in the user program or monitor mode. The parameters that are passed to a routine are in the form of a contiguous data block, stored in RAM. The index register (H:X) is loaded with the address of the first byte of the data block (acting as a pointer), and the subroutine is called (JSR). Using the start address as a pointer, multiple data blocks can be used, any area of RAM be used. A data block has the control and data bytes in a defined order, as shown in Figure 7-8.

During the software execution, it does not consume any dedicated RAM location, the run-time heap will extend the system stack, all other RAM location will not be affected.

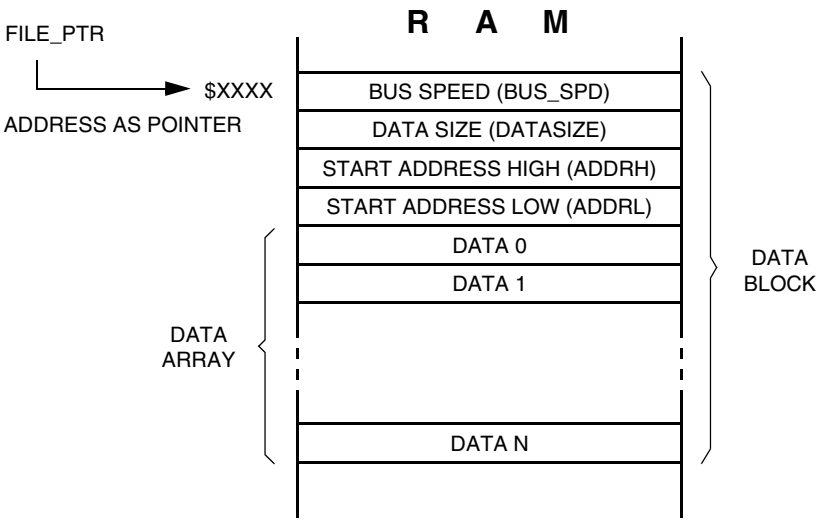


Figure 7-8. Data Block Format for ROM-Resident Routines

The control and data bytes are described below.

- **Bus speed** — This one byte indicates the operating bus speed of the MCU. The value of this byte should be equal to 4 times the bus speed, and should not be set to less than 4 (i.e. minimum bus speed is 1MHz).
- **Data size** — This one byte indicates the number of bytes in the data array that are to be manipulated. The maximum data array size is 128. Routines EE_WRITE and EE_READ are restricted to manipulate a data array between 2 to 15 bytes. Whereas routines ERARNGE and MON_ERARNGE do not manipulate a data array, thus, this data size byte has no meaning.
- **Start address** — These two bytes, high byte followed by low byte, indicate the start address of the FLASH memory to be manipulated.
- **Data array** — This data array contains data that are to be manipulated. Data in this array are programmed to FLASH memory by the programming routines: PRGRNGE, MON_PRGRNGE, EE_WRITE. For the read routines: LDRNGE, MON_LDRNGE, and EE_READ, data is read from FLASH and stored in this array.

7.5.1 PRGRNGE

PRGRNGE is used to program a range of FLASH locations with data loaded into the data array.

Table 7-11. PRGRNGE Routine

Routine Name	PRGRNGE
Routine Description	Program a range of locations
Calling Address	$FC06
Stack Used	15 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Start address high (ADDRH) Start address (ADDRL) Data 1 (DATA1) : Data N (DATAN)

The start location of the FLASH to be programmed is specified by the address ADDRH:ADDRL and the number of bytes from this location is specified by DATASIZE. The maximum number of bytes that can be programmed in one routine call is 128 bytes (max. DATASIZE is 128).

ADDRH:ADDRL do not need to be at a page boundary, the routine handles any boundary misalignment during programming. A check to see that all bytes in the specified range are erased is not performed by this routine prior programming. Nor does this routine do a verification after programming, so there is no return confirmation that programming was successful. User must assure that the range specified is first erased.

The coding example below is to program 32 bytes of data starting at FLASH location $EF00, with a bus speed of 4.9152 MHz. The coding assumes the data block is already loaded in RAM, with the address pointer, FILE_PTR, pointing to the first byte of the data block.

```
              ORG     RAM
       :
FILE_PTR:
BUS_SPD       DS.B    1; Indicates 4x bus frequency
DATASIZE      DS.B    1; Data size to be programmed
START_ADDR    DS.W    1; FLASH start address
DATAARRAY     DS.B    32; Reserved data array

PRGRNGE       EQU     $FC06
FLASH_START   EQU     $EF00

              ORG     FLASH
INITIALISATION:
       MOV    #20,    BUS_SPD
       MOV    #32,    DATASIZE
       LDHX   #FLASH_START
       STHX   START_ADDR
       RTS
MAIN:
       BSR    INITIALISATION
       :
       :
       LDHX   #FILE_PTR
       JSR    PRGRNGE
```

7.5.2 ERARNGE

ERARNGE is used to erase a range of locations in FLASH.

Table 7-12. ERARNGE Routine

Routine Name	ERARNGE
Routine Description	Erase a page or the entire array
Calling Address	$FCBE
Stack Used	9 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Starting address (ADDRH) Starting address (ADDRL)

There are two sizes of erase ranges: a page or the entire array. The ERARNGE will erase the page (64 consecutive bytes) in FLASH specified by the address ADDRH:ADDRL. This address can be any address within the page. Calling ERARNGE with ADDRH:ADDRL equal to $FFFF will erase the entire FLASH array (mass erase). Therefore, care must be taken when calling this routine to prevent an accidental mass erase. To avoid undesirable routine return addresses after a mass erase, the ERARNGE routine should not be called from code executed from FLASH memory. Load the code into an area of RAM before calling the ERARNGE routine.

The ERARNGE routine do not use a data array. The DATASIZE byte is a dummy byte that is also not used.

The coding example below is to perform a page erase, from $EF00–$EF3F. The Initialization subroutine is the same as the coding example for PRGRNGE (see 7.5.1 PRGRNGE).

```
ERARNGE         EQU     $FCBE
MAIN:
        BSR     INITIALISATION
        :
        :
        LDHX    #FILE_PTR
        JSR     ERARNGE
        :
```

7.5.3 LDRNGE

LDRNGE is used to load the data array in RAM with data from a range of FLASH locations.

Table 7-13. LDRNGE Routine

Routine Name	LDRNGE
Routine Description	Loads data from a range of locations
Calling Address	$FF30
Stack Used	9 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Starting address (ADDRH) Starting address (ADDRL) Data 1 : Data N

The start location of FLASH from where data is retrieved is specified by the address ADDRH:ADDRL and the number of bytes from this location is specified by DATASIZE. The maximum number of bytes that can be retrieved in one routine call is 128 bytes. The data retrieved from FLASH is loaded into the data array in RAM. Previous data in the data array will be overwritten. User can use this routine to retrieve data from FLASH that was previously programmed.

The coding example below is to retrieve 32 bytes of data starting from $EF00 in FLASH. The Initialization subroutine is the same as the coding example for PRGRNGE (see 7.5.1 PRGRNGE).

```
LDRNGE          EQU     $FF30
MAIN:
        BSR     INITIALIZATION
        :
        :
        LDHX    #FILE_PTR
        JSR     LDRNGE
        :
```

7.5.4 MON_PRGRNGE

In monitor mode, MON_PRGRNGE is used to program a range of FLASH locations with data loaded into the data array.

Table 7-14. MON_PRGRNGE Routine

Routine Name	MON_PRGRNGE
Routine Description	Program a range of locations, in monitor mode
Calling Address	$FC28
Stack Used	17 bytes
Data Block Format	Bus speed Data size Starting address (high byte) Starting address (low byte) Data 1 : Data N

The MON_PRGRNGE routine is designed to be used in monitor mode. It performs the same function as the PRGRNGE routine (see 7.5.1 PRGRNGE), except that MON_PRGRNGE returns to the main program via an SWI instruction. After a MON_PRGRNGE call, the SWI instruction will return the control back to the monitor code.

7.5.5 MON_ERARNGE

In monitor mode, ERARNGE is used to erase a range of locations in FLASH.

Table 7-15. MON_ERARNGE Routine

Routine Name	MON_ERARNGE
Routine Description	Erase a page or the entire array, in monitor mode
Calling Address	$FF2C
Stack Used	11 bytes
Data Block Format	Bus speed Data size Starting address (high byte) Starting address (low byte)

The MON_ERARNGE routine is designed to be used in monitor mode. It performs the same function as the ERARNGE routine (see 7.5.2 ERARNGE), except that MON_ERARNGE returns to the main program via an SWI instruction. After a MON_ERARNGE call, the SWI instruction will return the control back to the monitor code.

7.5.6 MON_LDRNGE

In monitor mode, LDRNGE is used to load the data array in RAM with data from a range of FLASH locations.

Table 7-16. ICP_LDRNGE Routine

Routine Name	MON_LDRNGE
Routine Description	Loads data from a range of locations, in monitor mode
Calling Address	$FF24
Stack Used	11 bytes
Data Block Format	Bus speed Data size Starting address (high byte) Starting address (low byte) Data 1 : Data N

The MON_LDRNGE routine is designed to be used in monitor mode. It performs the same function as the LDRNGE routine (see 7.5.3 LDRNGE), except that MON_LDRNGE returns to the main program via an SWI instruction. After a MON_LDRNGE call, the SWI instruction will return the control back to the monitor code.

7.5.7 EE_WRITE

EE_WRITE is used to write a set of data from the data array to FLASH.

Table 7-17. EE_WRITE Routine

Routine Name	EE_WRITE
Routine Description	Emulated EEPROM write. Data size ranges from 2 to 15 bytes at a time.
Calling Address	$FD3F
Stack Used	24 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE)(1) Starting address (ADDRH)(2) Starting address (ADDRL)(1) Data 1 : Data N

1. The minimum data size is 2 bytes. The maximum data size is 15 bytes.
2. The start address must be a page boundary start address: $xx00, $xx40, $xx80, or $00C0.

The start location of the FLASH to be programmed is specified by the address ADDRH:ADDRL and the number of bytes in the data array is specified by DATASIZE. The minimum number of bytes that can be

programmed in one routine call is 2 bytes, the maximum is 15 bytes. ADDRH:ADDRL must always be the start of boundary address (the page start address: $XX00, $XX40, $XX80, or $00C0) and DATASIZE must be the same size when accessing the same page.

In some applications, the user may want to repeatedly store and read a set of data from an area of non-volatile memory. This is easily possible when using an EEPROM array. As the write and erase operations can be executed on a byte basis. For FLASH memory, the minimum erase size is the page — 64 bytes per page for MC68HC908JL8. If the data array size is less than the page size, writing and erasing to the same page cannot fully utilize the page. Unused locations in the page will be wasted. The EE_WRITE routine is designed to emulate the properties similar to the EEPROM. Allowing a more efficient use of the FLASH page for data storage.

When the user dedicates a page of FLASH for data storage, and the size of the data array defined, each call of the EE_WRTIE routine will automatically transfer the data in the data array (in RAM) to the next blank block of locations in the FLASH page. Once a page is filled up, the EE_WRITE routine automatically erases the page, and starts to reuse the page again. In the 64-byte page, an 4-byte control block is used by the routine to monitor the utilization of the page. In effect, only 60 bytes are used for data storage. (see Figure 7-9). The page control operations are transparent to the user.

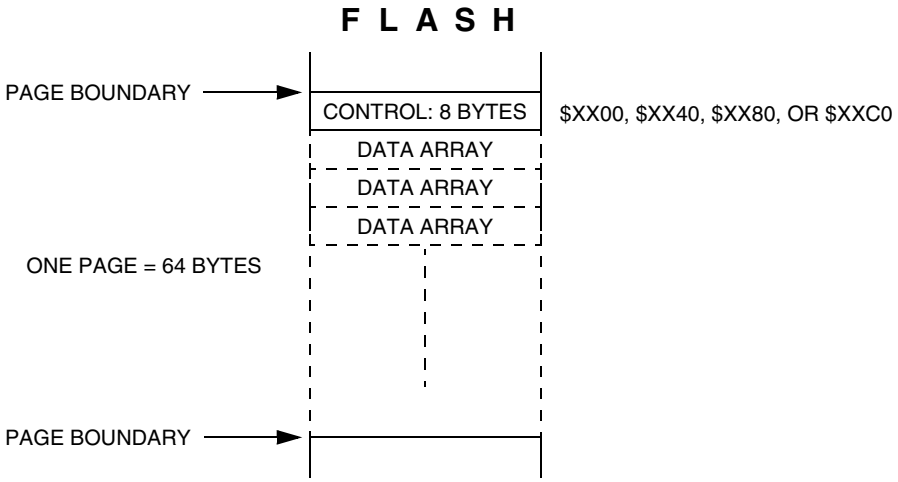


Figure 7-9. EE_WRITE FLASH Memory Usage

When using this routine to store a 3-byte data array, the FLASH page can be programmed 20 times before the an erase is required. In effect, the write/erase endurance is increased by 20 times. When a 15-byte data array is used, the write/erase endurance is increased by 5 times. Due to the FLASH page size limitation, the data array is limited from 2 bytes to 15 bytes.

The coding example below uses the $EF00–$EE3F page for data storage. The data array size is 15 bytes, and the bus speed is 4.9152 MHz. The coding assumes the data block is already loaded in RAM, with the address pointer, FILE_PTR, pointing to the first byte of the data block.

```
              ORG     RAM
       :
FILE_PTR:
BUS_SPD       DS.B    1; Indicates 4x bus frequency
DATASIZE      DS.B    1; Data size to be programmed
START_ADDR    DS.W    1; FLASH starting address
DATAARRAY     DS.B    15; Reserved data array

EE_WRITE      EQU     $FD3F
FLASH_START   EQU     $EF00

              ORG     FLASH
INITIALISATION:
       MOV    #20,    BUS_SPD
       MOV    #15,    DATASIZE
       LDHX   #FLASH_START
       STHX   START_ADDR
       RTS
MAIN:
       BSR    INITIALISATION
       :
       :
       LHDX   #FILE_PTR
       JSR    EE_WRITE
```

NOTE

The EE_WRITE routine is unable to check for incorrect data blocks, such as the FLASH page boundary address and data size. It is the responsibility of the user to ensure the starting address indicated in the data block is at the FLASH page boundary and the data size is 2 to 15. If the FLASH page is already programmed with a data array with a different size, the EE_WRITE call will be ignored.

7.5.8 EE_READ

EE_READ is used to load the data array in RAM with a set of data from FLASH.

Table 7-18. EE_READ Routine

Routine Name	EE_READ
Routine Description	Emulated EEPROM read. Data size ranges from 2 to 15 bytes at a time.
Calling Address	$FDD0
Stack Used	16 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Starting address (ADDRH)(1) Starting address (ADDRL)(1) Data 1 : Data N

1. The start address must be a page boundary start address: $xx00, $xx40, $xx80, or $00C0.

The EE_READ routine reads data stored by the EE_WRITE routine. An EE_READ call will retrieve the last data written to a FLASH page and loaded into the data array in RAM. Same as EE_WRITE, the data size indicated by DATASIZE is 2 to 15, and the start address ADDRH:ADDRL must the FLASH page boundary address.

The coding example below uses the data stored by the EE_WRITE coding example (see 7.5.7 EE_WRITE). It loads the 15-byte data set stored in the $EF00–$EE7F page to the data array in RAM. The initialization subroutine is the same as the coding example for EE_WRITE (see 7.5.7 EE_WRITE).

```
EE_READ         EQU     $FDD0

MAIN:
        BSR     INITIALIZATION
        :
        :
        LDHX    FILE_PTR
        JSR     EE_READ
        :
```

NOTE

The EE_READ routine is unable to check for incorrect data blocks, such as the FLASH page boundary address and data size. It is the responsibility of the user to ensure the starting address indicated in the data block is at the FLASH page boundary and the data size is 2 to 15. If the FLASH page is programmed with a data array with a different size, the EE_READ call will be ignored.

Chapter 8
Timer Interface Module (TIM)

8.1 Introduction

This section describes the timer interface (TIM) module. The TIM is a two-channel timer that provides a timing reference with Input capture, output compare, and pulse-width-modulation functions. Figure 8-1 is a block diagram of the TIM.

This particular MCU has two timer interface modules which are denoted as TIM1 and TIM2.

8.2 Features

Features of the TIM include:

- Two input capture/output compare channels:
  - Rising-edge, falling-edge, or any-edge input capture trigger
  - Set, clear, or toggle output compare action
- Buffered and unbuffered pulse-width-modulation (PWM) signal generation
- Programmable TIM clock input
  - 7-frequency internal bus clock prescaler selection
  - External clock input on timer 2 (bus frequency ÷2 maximum)
- Free-running or modulo up-count operation
- Toggle any channel pin on overflow
- TIM counter stop and reset bits

8.3 Pin Name Conventions

The text that follows describes both timers, TIM1 and TIM2. The TIM input/output (I/O) pin names are T[1,2]CH0 (timer channel 0) and T[1,2]CH1 (timer channel 1), where “1” is used to indicate TIM1 and “2” is used to indicate TIM2. The two TIMs share four I/O pins with four I/O port pins. The external clock input for TIM2 is shared with the an ADC channel pin. The full names of the TIM I/O pins are listed in Table 8-1. The generic pin names appear in the text that follows.

Table 8-1. Pin Name Conventions

TIM Generic Pin Names:		T[1,2]CH0	T[1,2]CH1	T2CLK
Full TIM Pin Names:	TIM1	PTD4/T1CH0	PTD5/T1CH1	—
	TIM2	PTE0/T2CH0	PTE1/T2CH1	ADC12/T2CLK

NOTE

References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TCH0 may refer generically to T1CH0 and T2CH0, and TCH1 may refer to T1CH1 and T2CH1.

8.4 Functional Description

Figure 8-1 shows the structure of the TIM. The central component of the TIM is the 16-bit TIM counter that can operate as a free-running counter or a modulo up-counter. The TIM counter provides the timing reference for the input capture and output compare functions. The TIM counter modulo registers, TMODH:TMODL, control the modulo value of the TIM counter. Software can read the TIM counter value at any time without affecting the counting sequence.

The two TIM channels (per timer) are programmable independently as input capture or output compare channels.

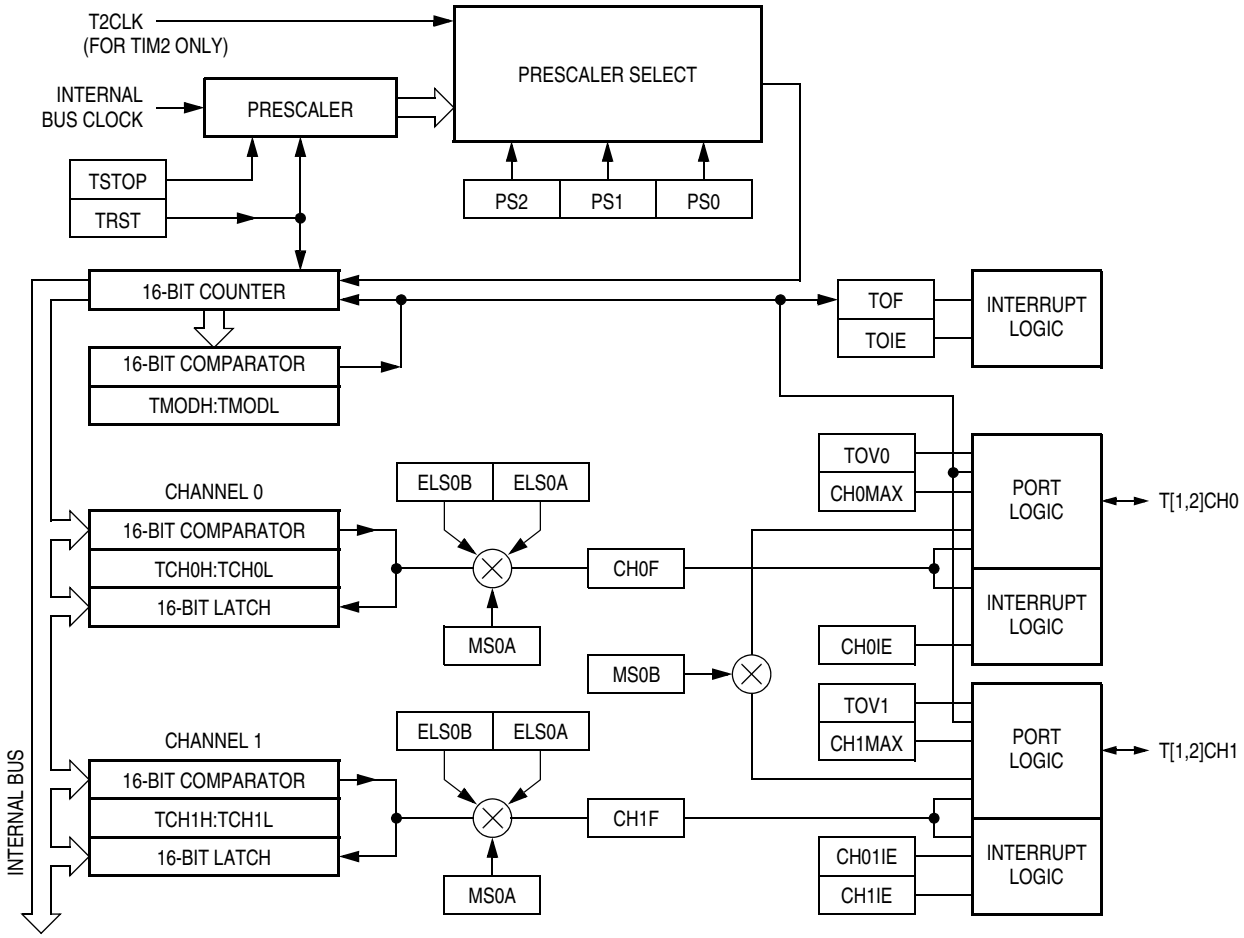


Figure 8-1. TIM Block Diagram

Figure 8-2 summarizes the timer registers.

NOTE

References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TSC may generically refer to both T1SC and T2SC.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0020	TIM1 Status and Control Register (T1SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0021	TIM1 Counter Register High (T1CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0022	TIM1 Counter Register Low (T1CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0023	TIM Counter Modulo Register High (TMODH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0024	TIM1 Counter Modulo Register Low (T1MODL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0025	TIM1 Channel 0 Status and Control Register (T1SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0026	TIM1 Channel 0 Register High (T1CH0H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0027	TIM1 Channel 0 Register Low (T1CH0L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0028	TIM1 Channel 1 Status and Control Register (T1SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0029	TIM1 Channel 1 Register High (T1CH1H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$002A	TIM1 Channel 1 Register Low (T1CH1L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0030	TIM2 Status and Control Register (T2SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0031	TIM2 Counter Register High (T2CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0032	TIM2 Counter Register Low (T2CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0033	TIM2 Counter Modulo Register High (T2MODH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1

= Unimplemented

Figure 8-2. TIM I/O Register Summary (Sheet 1 of 2)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0034	TIM2 Counter Modulo Register Low (T2MODL)	Read: / Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$0035	TIM2 Channel 0 Status and Control Register (T2SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0036	TIM2 Channel 0 Register High (T2CH0H)	Read: / Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0037	TIM2 Channel 0 Register Low (T2CH0L)	Read: / Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$0038	TIM2 Channel 1 Status and Control Register (T2SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0039	TIM2 Channel 1 Register High (T2CH1H)	Read: / Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$003A	TIM2 Channel 1 Register Low (T2CH1L)	Read: / Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							

= Unimplemented

Figure 8-2. TIM I/O Register Summary (Sheet 2 of 2)

8.4.1 TIM Counter Prescaler

The TIM1 clock source can be one of the seven prescaler outputs; TIM2 clock source can be one of the seven prescaler outputs or the TIM2 clock pin, T2CLK. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIM status and control register select the TIM clock source.

8.4.2 Input Capture

With the input capture function, the TIM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TIM latches the contents of the TIM counter into the TIM channel registers, TCHxH:TCHxL. The polarity of the active edge is programmable. Input captures can generate TIM CPU interrupt requests.

8.4.3 Output Compare

With the output compare function, the TIM can generate a periodic pulse with a programmable polarity, duration, and frequency. When the counter reaches the value in the registers of an output compare channel, the TIM can set, clear, or toggle the channel pin. Output compares can generate TIM CPU interrupt requests.

8.4.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in 8.4.3 Output Compare. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIM channel registers.

An unsynchronized write to the TIM channel registers to change an output compare value could cause incorrect operation for up to two counter overflow periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that counter overflow period. Also, using a TIM overflow interrupt routine to write a new, smaller output compare value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the output compare value on channel x:

- When changing to a smaller value, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current output compare pulse. The interrupt routine has until the end of the counter overflow period to write the new value.
- When changing to a larger output compare value, enable TIM overflow interrupts and write the new value in the TIM overflow interrupt routine. The TIM overflow interrupt occurs at the end of the current counter overflow period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same counter overflow period.

8.4.3.2 Buffered Output Compare

Channels 0 and 1 can be linked to form a buffered output compare channel whose output appears on the TCH0 pin. The TIM channel registers of the linked pair alternately control the output.

Setting the MS0B bit in TIM channel 0 status and control register (TSC0) links channel 0 and channel 1. The output compare value in the TIM channel 0 registers initially controls the output on the TCH0 pin. Writing to the TIM channel 1 registers enables the TIM channel 1 registers to synchronously control the output after the TIM overflows. At each subsequent overflow, the TIM channel registers (0 or 1) that control the output are the ones written to last. TSC0 controls and monitors the buffered output compare function, and TIM channel 1 status and control register (TSC1) is unused. While the MS0B bit is set, the channel 1 pin, TCH1, is available as a general-purpose I/O pin.

NOTE

In buffered output compare operation, do not write new output compare values to the currently active channel registers. User software should track the currently active channel to prevent writing a new value to the active channel. Writing to the active channel registers is the same as generating unbuffered output compares.

8.4.4 Pulse Width Modulation (PWM)

By using the toggle-on-overflow feature with an output compare channel, the TIM can generate a PWM signal. The value in the TIM counter modulo registers determines the period of the PWM signal. The channel pin toggles when the counter reaches the value in the TIM counter modulo registers. The time between overflows is the period of the PWM signal.

As Figure 8-3 shows, the output compare value in the TIM channel registers determines the pulse width of the PWM signal. The time between overflow and output compare is the pulse width. Program the TIM

to clear the channel pin on output compare if the state of the PWM pulse is logic 1. Program the TIM to set the pin if the state of the PWM pulse is logic 0.

The value in the TIM counter modulo registers and the selected prescaler output determines the frequency of the PWM output. The frequency of an 8-bit PWM signal is variable in 256 increments. Writing $00FF (255) to the TIM counter modulo registers produces a PWM period of 256 times the internal bus clock period if the prescaler select value is $000. See 8.9.1 TIM Status and Control Register.

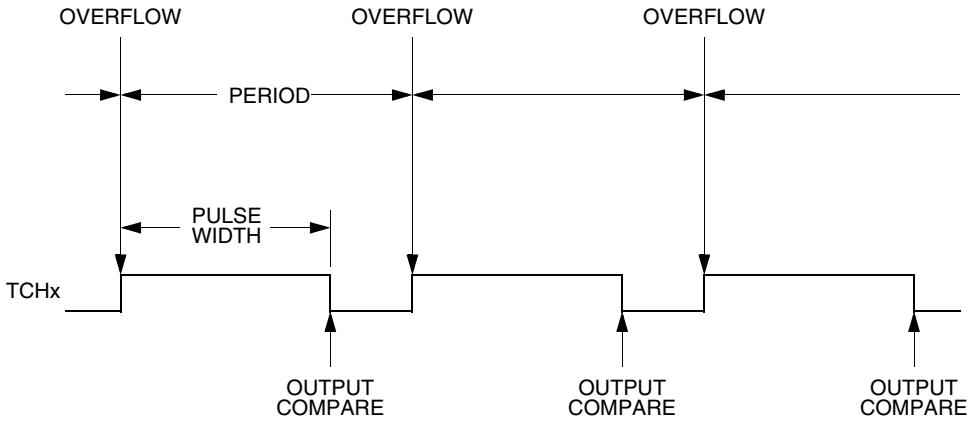


Figure 8-3. PWM Period and Pulse Width

The value in the TIM channel registers determines the pulse width of the PWM output. The pulse width of an 8-bit PWM signal is variable in 256 increments. Writing $0080 (128) to the TIM channel registers produces a duty cycle of 128/256 or 50%.

8.4.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in 8.4.4 Pulse Width Modulation (PWM). The pulses are unbuffered because changing the pulse width requires writing the new pulse width value over the old value currently in the TIM channel registers.

An unsynchronized write to the TIM channel registers to change a pulse width value could cause incorrect operation for up to two PWM periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that PWM period. Also, using a TIM overflow interrupt routine to write a new, smaller pulse width value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the PWM pulse width on channel x:

- When changing to a shorter pulse width, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current pulse. The interrupt routine has until the end of the PWM period to write the new value.
- When changing to a longer pulse width, enable TIM overflow interrupts and write the new value in the TIM overflow interrupt routine. The TIM overflow interrupt occurs at the end of the current PWM period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same PWM period.

NOTE

In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0% duty cycle generation and removes the ability of the channel to self-correct in the event of software error or noise. Toggling on output compare also can cause incorrect PWM signal generation when changing the PWM pulse width to a new, much larger value.

8.4.4.2 Buffered PWM Signal Generation

Channels 0 and 1 can be linked to form a buffered PWM channel whose output appears on the TCH0 pin. The TIM channel registers of the linked pair alternately control the pulse width of the output.

Setting the MS0B bit in TIM channel 0 status and control register (TSC0) links channel 0 and channel 1. The TIM channel 0 registers initially control the pulse width on the TCH0 pin. Writing to the TIM channel 1 registers enables the TIM channel 1 registers to synchronously control the pulse width at the beginning of the next PWM period. At each subsequent overflow, the TIM channel registers (0 or 1) that control the pulse width are the ones written to last. TSC0 controls and monitors the buffered PWM function, and TIM channel 1 status and control register (TSC1) is unused. While the MS0B bit is set, the channel 1 pin, TCH1, is available as a general-purpose I/O pin.

NOTE

In buffered PWM signal generation, do not write new pulse width values to the currently active channel registers. User software should track the currently active channel to prevent writing a new value to the active channel. Writing to the active channel registers is the same as generating unbuffered PWM signals.

8.4.4.3 PWM Initialization

To ensure correct operation when generating unbuffered or buffered PWM signals, use the following initialization procedure:

1. In the TIM status and control register (TSC):
   a. Stop the TIM counter by setting the TIM stop bit, TSTOP.
   b. Reset the TIM counter and prescaler by setting the TIM reset bit, TRST.
2. In the TIM counter modulo registers (TMODH:TMODL), write the value for the required PWM period.
3. In the TIM channel x registers (TCHxH:TCHxL), write the value for the required pulse width.
4. In TIM channel x status and control register (TSCx):
   a. Write 0:1 (for unbuffered output compare or PWM signals) or 1:0 (for buffered output compare or PWM signals) to the mode select bits, MSxB:MSxA. (See Table 8-3.)
   b. Write 1 to the toggle-on-overflow bit, TOVx.
   c. Write 1:0 (to clear output on compare) or 1:1 (to set output on compare) to the edge/level select bits, ELSxB:ELSxA. The output action on compare must force the output to the complement of the pulse width level. (See Table 8-3.)

NOTE

In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0% duty

cycle generation and removes the ability of the channel to self-correct in the event of software error or noise. Toggling on output compare can also cause incorrect PWM signal generation when changing the PWM pulse width to a new, much larger value.

5. In the TIM status control register (TSC), clear the TIM stop bit, TSTOP.

Setting MS0B links channels 0 and 1 and configures them for buffered PWM operation. The TIM channel 0 registers (TCH0H:TCH0L) initially control the buffered PWM output. TIM status control register 0 (TSCR0) controls and monitors the PWM signal from the linked channels.

Clearing the toggle-on-overflow bit, TOVx, inhibits output toggles on TIM overflows. Subsequent output compares try to force the output to a state it is already in and have no effect. The result is a 0% duty cycle output.

Setting the channel x maximum duty cycle bit (CHxMAX) and setting the TOVx bit generates a 100% duty cycle output. (See 8.9.4 TIM Channel Status and Control Registers.)

8.5 Interrupts

The following TIM sources can generate interrupt requests:

- TIM overflow flag (TOF) — The TOF bit is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. The TIM overflow interrupt enable bit, TOIE, enables TIM overflow CPU interrupt requests. TOF and TOIE are in the TIM status and control register.
- TIM channel flags (CH1F:CH0F) — The CHxF bit is set when an input capture or output compare occurs on channel x. Channel x TIM CPU interrupt requests are controlled by the channel x interrupt enable bit, CHxIE. Channel x TIM CPU interrupt requests are enabled when CHxIE = 1. CHxF and CHxIE are in the TIM channel x status and control register.

8.6 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power- consumption standby modes.

8.6.1 Wait Mode

The TIM remains active after the execution of a WAIT instruction. In wait mode, the TIM registers are not accessible by the CPU. Any enabled CPU interrupt request from the TIM can bring the MCU out of wait mode.

If TIM functions are not required during wait mode, reduce power consumption by stopping the TIM before executing the WAIT instruction.

8.6.2 Stop Mode

The TIM is inactive after the execution of a STOP instruction. The STOP instruction does not affect register conditions or the state of the TIM counter. TIM operation resumes when the MCU exits stop mode after an external interrupt.

8.7 TIM During Break Interrupts

A break interrupt stops the TIM counter.

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the break flag control register (BFCR) enables software to clear status bits during the break state. (See 5.7.3 Break Flag Control Register (BFCR).)

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at logic 0. After the break, doing the second step clears the status bit.

8.8 I/O Signals

Port D shares two of its pins with TIM1 and port E shares two of its pins with TIM2. The ADC12/T2CLK pin is an external clock input to TIM2. The four TIM channel I/O pins are T1CH0, T1CH1, T2CH0, and T2CH1.

8.8.1 TIM Clock Pin (ADC12/T2CLK)

ADC12/T2CLK is an external clock input that can be the clock source for the TIM2 counter instead of the prescaled internal bus clock. Select the ADC12/T2CLK input by writing logic 1's to the three prescaler select bits, PS[2:0]. (See 8.9.1 TIM Status and Control Register.) The minimum T2CLK pulse width, $T2CLK_{LMIN}$ or $T2CLK_{HMIN}$, is:

$$\frac{1}{\text{bus frequency}} + t_{SU}$$

The maximum T2CLK frequency is:

$$\text{bus frequency} \div 2$$

ADC12/T2CLK is available as a ADC input channel pin when not used as the TIM2 clock input.

8.8.2 TIM Channel I/O Pins (PTD4/T1CH0, PTD5/T1CH1, PTE0/T2CH0, PTE1/T2CH1)

Each channel I/O pin is programmable independently as an input capture pin or an output compare pin. T1CH0 and T2CH0 can be configured as buffered output compare or buffered PWM pins.

8.9 I/O Registers

NOTE

References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TSC may generically refer to both T1SC AND T2SC.

These I/O registers control and monitor operation of the TIM:

- TIM status and control register (TSC)
- TIM counter registers (TCNTH:TCNTL)
- TIM counter modulo registers (TMODH:TMODL)
- TIM channel status and control registers (TSC0, TSC1)
- TIM channel registers (TCH0H:TCH0L, TCH1H:TCH1L)

8.9.1 TIM Status and Control Register

The TIM status and control register (TSC):

- Enables TIM overflow interrupts
- Flags TIM overflows
- Stops the TIM counter
- Resets the TIM counter
- Prescales the TIM counter clock

Address: T1SC, $0020 and T2SC, $0030

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
Write:	0			TRST				
Reset:	0	0	1	0	0	0	0	0

= Unimplemented

Figure 8-4. TIM Status and Control Register (TSC)

TOF — TIM Overflow Flag Bit

This read/write flag is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. Clear TOF by reading the TIM status and control register when TOF is set and then writing a logic 0 to TOF. If another TIM overflow occurs before the clearing sequence is complete, then writing logic 0 to TOF has no effect. Therefore, a TOF interrupt request cannot be lost due to inadvertent clearing of TOF. Reset clears the TOF bit. Writing a logic 1 to TOF has no effect.

1 = TIM counter has reached modulo value
0 = TIM counter has not reached modulo value

TOIE — TIM Overflow Interrupt Enable Bit

This read/write bit enables TIM overflow interrupts when the TOF bit becomes set. Reset clears the TOIE bit.

1 = TIM overflow interrupts enabled
0 = TIM overflow interrupts disabled

TSTOP — TIM Stop Bit

This read/write bit stops the TIM counter. Counting resumes when TSTOP is cleared. Reset sets the TSTOP bit, stopping the TIM counter until software clears the TSTOP bit.

1 = TIM counter stopped
0 = TIM counter active

NOTE

Do not set the TSTOP bit before entering wait mode if the TIM is required to exit wait mode.

TRST — TIM Reset Bit

Setting this write-only bit resets the TIM counter and the TIM prescaler. Setting TRST has no effect on any other registers. Counting resumes from $0000. TRST is cleared automatically after the TIM counter is reset and always reads as logic 0. Reset clears the TRST bit.

1 = Prescaler and TIM counter cleared
0 = No effect

NOTE

Setting the TSTOP and TRST bits simultaneously stops the TIM counter at a value of $0000.

PS[2:0] — Prescaler Select Bits

These read/write bits select one of the seven prescaler outputs as the input to the TIM counter as Table 8-2 shows. Reset clears the PS[2:0] bits.

Table 8-2. Prescaler Selection

PS2	PS1	PS0	TIM Clock Source
0	0	0	Internal bus clock ÷ 1
0	0	1	Internal bus clock ÷ 2
0	1	0	Internal bus clock ÷ 4
0	1	1	Internal bus clock ÷ 8
1	0	0	Internal bus clock ÷ 16
1	0	1	Internal bus clock ÷ 32
1	1	0	Internal bus clock ÷ 64
1	1	1	T2CLK (for TIM2 only)

8.9.2 TIM Counter Registers

The two read-only TIM counter registers contain the high and low bytes of the value in the TIM counter. Reading the high byte (TCNTH) latches the contents of the low byte (TCNTL) into a buffer. Subsequent reads of TCNTH do not affect the latched TCNTL value until TCNTL is read. Reset clears the TIM counter registers. Setting the TIM reset bit (TRST) also clears the TIM counter registers.

NOTE

If you read TCNTH during a break interrupt, be sure to unlatch TCNTL by reading TCNTL before exiting the break interrupt. Otherwise, TCNTL retains the value latched during the break.

Address: T1CNTH, $0021 and T2CNTH, $0031

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

= Unimplemented

Figure 8-5. TIM Counter Registers High (TCNTH)

Address: T1CNTL, $0022 and T2CNTL, $0032

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

= Unimplemented

Figure 8-6. TIM Counter Registers Low (TCNTL)

8.9.3 TIM Counter Modulo Registers

The read/write TIM modulo registers contain the modulo value for the TIM counter. When the TIM counter reaches the modulo value, the overflow flag (TOF) becomes set, and the TIM counter resumes counting from $0000 at the next timer clock. Writing to the high byte (TMODH) inhibits the TOF bit and overflow interrupts until the low byte (TMODL) is written. Reset sets the TIM counter modulo registers.

Address: T1MODH, $0023 and T2MODH, $0033

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

Figure 8-7. TIM Counter Modulo Register High (TMODH)

Address: T1MODL, $0024 and T2MODL, $0034

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

Figure 8-8. TIM Counter Modulo Register Low (TMODL)

NOTE

Reset the TIM counter before writing to the TIM counter modulo registers.

8.9.4 TIM Channel Status and Control Registers

Each of the TIM channel status and control registers:

- Flags input captures and output compares
- Enables input capture and output compare interrupts
- Selects input capture, output compare, or PWM operation
- Selects high, low, or toggling output on output compare
- Selects rising edge, falling edge, or any edge as the active input capture trigger
- Selects output toggling on TIM overflow
- Selects 0% and 100% PWM duty cycle
- Selects buffered or unbuffered output compare/PWM operation

Address: T1SC0, $0025 and T2SC0, $0035

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
Write:	0							
Reset:	0	0	0	0	0	0	0	0

Figure 8-9. TIM Channel 0 Status and Control Register (TSC0)

Address: T1SC1, $0028 and T2SC1, $0038

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
Write:	0							
Reset:	0	0	0	0	0	0	0	0

Figure 8-10. TIM Channel 1 Status and Control Register (TSC1)

CHxF — Channel x Flag Bit

When channel x is an input capture channel, this read/write bit is set when an active edge occurs on the channel x pin. When channel x is an output compare channel, CHxF is set when the value in the TIM counter registers matches the value in the TIM channel x registers.

When TIM CPU interrupt requests are enabled (CHxIE = 1), clear CHxF by reading TIM channel x status and control register with CHxF set and then writing a logic 0 to CHxF. If another interrupt request occurs before the clearing sequence is complete, then writing logic 0 to CHxF has no effect. Therefore, an interrupt request cannot be lost due to inadvertent clearing of CHxF.

Reset clears the CHxF bit. Writing a logic 1 to CHxF has no effect.

1 = Input capture or output compare on channel x
0 = No input capture or output compare on channel x

CHxIE — Channel x Interrupt Enable Bit

This read/write bit enables TIM CPU interrupt service requests on channel x.

Reset clears the CHxIE bit.

1 = Channel x CPU interrupt requests enabled
0 = Channel x CPU interrupt requests disabled

MSxB — Mode Select Bit B

This read/write bit selects buffered output compare/PWM operation. MSxB exists only in the TIM1 channel 0 and TIM2 channel 0 status and control registers.

Setting MS0B disables the channel 1 status and control register and reverts TCH1 to general-purpose I/O.

Reset clears the MSxB bit.

1 = Buffered output compare/PWM operation enabled

0 = Buffered output compare/PWM operation disabled

MSxA — Mode Select Bit A

When ELSxB:ELSxA ≠ 0:0, this read/write bit selects either input capture operation or unbuffered output compare/PWM operation.

See Table 8-3.

1 = Unbuffered output compare/PWM operation

0 = Input capture operation

When ELSxB:ELSxA = 0:0, this read/write bit selects the initial output level of the TCHx pin. See Table 8-3. Reset clears the MSxA bit.

1 = Initial output level low

0 = Initial output level high

NOTE

Before changing a channel function by writing to the MSxB or MSxA bit, set the TSTOP and TRST bits in the TIM status and control register (TSC).

ELSxB and ELSxA — Edge/Level Select Bits

When channel x is an input capture channel, these read/write bits control the active edge-sensing logic on channel x.

When channel x is an output compare channel, ELSxB and ELSxA control the channel x output behavior when an output compare occurs.

When ELSxB and ELSxA are both clear, channel x is not connected to an I/O port, and pin TCHx is available as a general-purpose I/O pin. Table 8-3 shows how ELSxB and ELSxA work. Reset clears the ELSxB and ELSxA bits.

Table 8-3. Mode, Edge, and Level Selection

MSxB:MSxA	ELSxB:ELSxA	Mode	Configuration
X0	00	Output preset	Pin under port control; initial output level high
X1	00		Pin under port control; initial output level low
00	01	Input capture	Capture on rising edge only
00	10		Capture on falling edge only
00	11		Capture on rising or falling edge
01	01	Output compare or PWM	Toggle output on compare
01	10		Clear output on compare
01	11		Set output on compare
1X	01	Buffered output compare or buffered PWM	Toggle output on compare
1X	10		Clear output on compare
1X	11		Set output on compare

NOTE

Before enabling a TIM channel register for input capture operation, make sure that the TCHx pin is stable for at least two bus clocks.

TOVx — Toggle On Overflow Bit

When channel x is an output compare channel, this read/write bit controls the behavior of the channel x output when the TIM counter overflows. When channel x is an input capture channel, TOVx has no effect.

Reset clears the TOVx bit.

1 = Channel x pin toggles on TIM counter overflow

0 = Channel x pin does not toggle on TIM counter overflow

NOTE

When TOVx is set, a TIM counter overflow takes precedence over a channel x output compare if both occur at the same time.

CHxMAX — Channel x Maximum Duty Cycle Bit

When the TOVx bit is at logic 1, setting the CHxMAX bit forces the duty cycle of buffered and unbuffered PWM signals to 100%. As Figure 8-11 shows, the CHxMAX bit takes effect in the cycle after it is set or cleared. The output stays at the 100% duty cycle level until the cycle after CHxMAX is cleared.

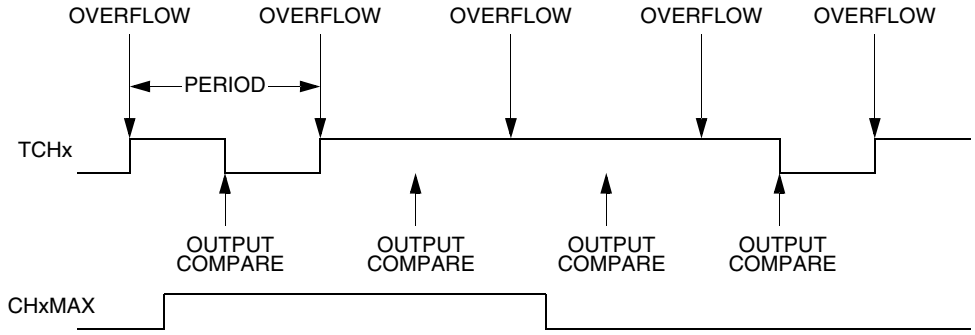


Figure 8-11. CHxMAX Latency

8.9.5 TIM Channel Registers

These read/write registers contain the captured TIM counter value of the input capture function or the output compare value of the output compare function. The state of the TIM channel registers after reset is unknown.

In input capture mode (MSxB:MSxA = 0:0), reading the high byte of the TIM channel x registers (TCHxH) inhibits input captures until the low byte (TCHxL) is read.

In output compare mode (MSxB:MSxA ≠ 0:0), writing to the high byte of the TIM channel x registers (TCHxH) inhibits output compares until the low byte (TCHxL) is written.

Address: T1CH0H, $0026 and T2CH0H, $0036

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

Figure 8-12. TIM Channel 0 Register High (TCH0H)

Address: T1CH0L, $0027 and T2CH0L $0037

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

Figure 8-13. TIM Channel 0 Register Low (TCH0L)

Address: T1CH1H, $0029 and T2CH1H, $0039

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

Figure 8-14. TIM Channel 1 Register High (TCH1H)

Address: T1CH1L, $002A and T2CH1L, $003A

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

Figure 8-15. TIM Channel 1 Register Low (TCH1L)

Chapter 9
Serial Communications Interface (SCI)

9.1 Introduction

This section describes the serial communications interface (SCI) module, which allows high-speed asynchronous communications with peripheral devices and other MCUs.

9.2 Features

Features of the SCI module include the following:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 32 programmable baud rates
- Programmable 8-bit or 9-bit character length
- Separately enabled transmitter and receiver
- Separate receiver and transmitter CPU interrupt requests
- Programmable transmitter output polarity
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Interrupt-driven operation with eight interrupt flags:
  - Transmitter empty
  - Transmission complete
  - Receiver full
  - Idle receiver input
  - Receiver overrun
  - Noise error
  - Framing error
  - Parity error
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection
- Bus clock as baud rate clock source

9.3 Pin Name Conventions

The generic names of the SCI I/O pins are:

- RxD (receive data)
- TxD (transmit data)

The SCI I/O (input/output) lines are dedicated pins for the SCI module. Table 9-1 shows the full names and the generic names of the SCI I/O pins.

The generic pin names appear in the text of this section.

Table 9-1. Pin Name Conventions

Generic Pin Names:	RxD	TxD
Full Pin Names:	PTD7/RxD	PTD6/TxD

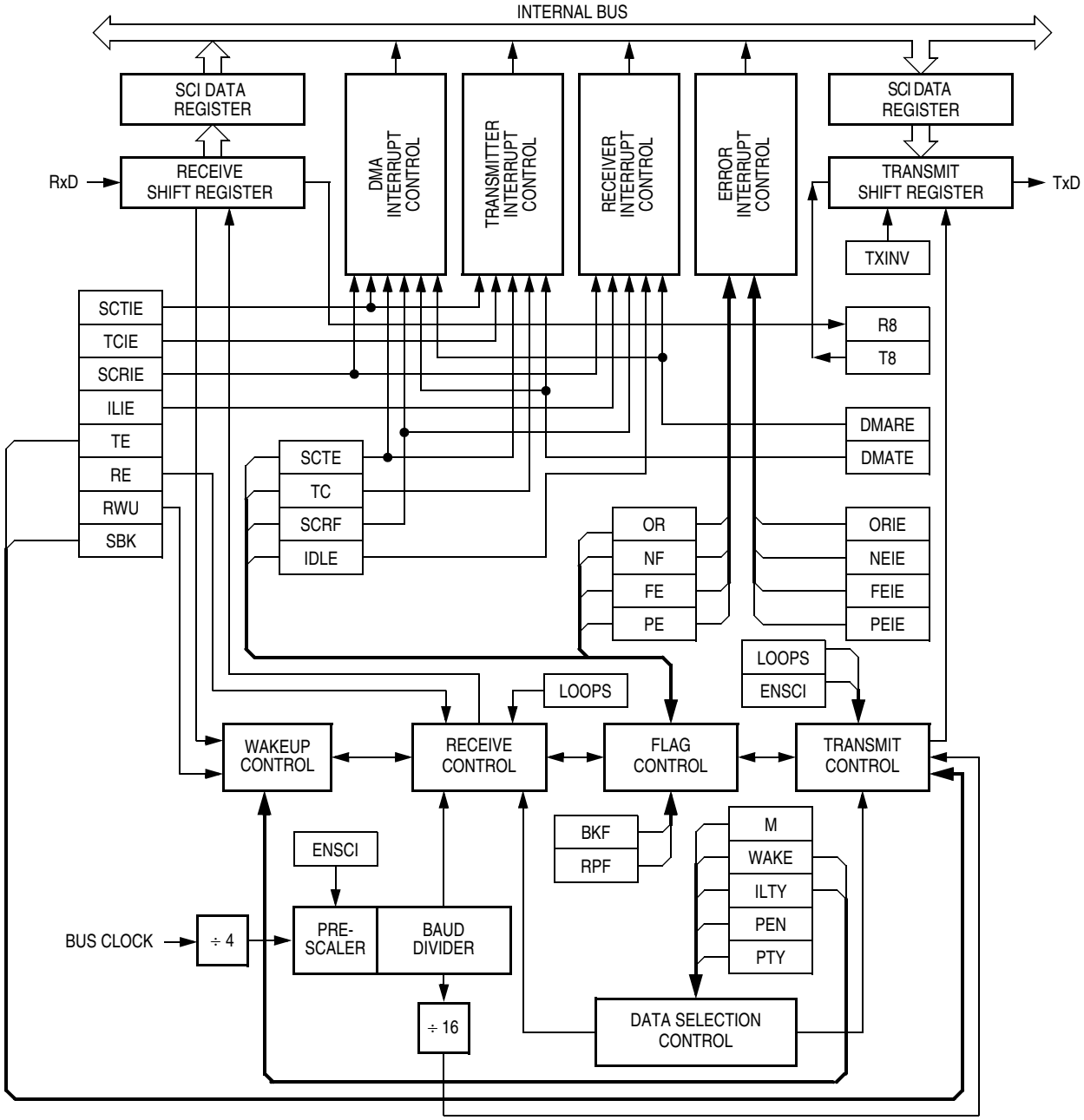


Figure 9-1. SCI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0013	SCI Control Register 1 (SCC1)	Read: Write:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Reset:	0	0	0	0	0	0	0	0
$0014	SCI Control Register 2 (SCC2)	Read: Write:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Reset:	0	0	0	0	0	0	0	0
$0015	SCI Control Register 3 (SCC3)	Read:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
		Write:								
		Reset:	U	U	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1)	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2)	Read:							BKF	RPF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0018	SCI Data Register (SCDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0019	SCI Baud Rate Register (SCBR)	Read:	0	0	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 9-2. SCI I/O Register Summary

9.4 Functional Description

Figure 9-1 shows the structure of the SCI module. The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The transmitter and receiver of the SCI operate independently, although they use the same baud rate generator. During normal operation, the CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data. The baud rate clock source for the SCI is the bus clock.

9.4.1 Data Format

The SCI uses the standard non-return-to-zero mark/space data format illustrated in Figure 9-3.

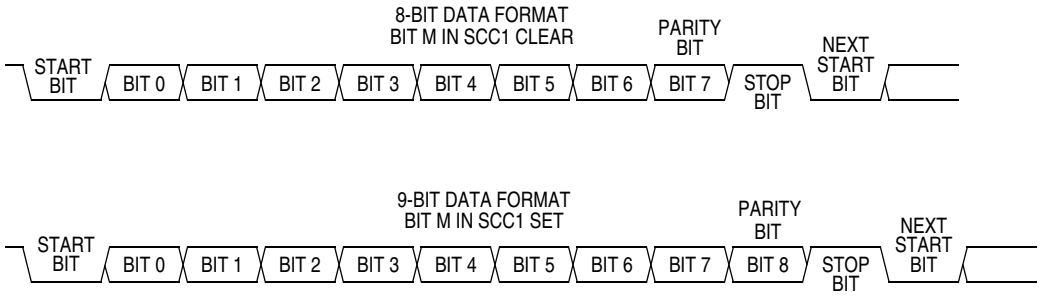


Figure 9-3. SCI Data Formats

9.4.2 Transmitter

Figure 9-4 shows the structure of the SCI transmitter.

The baud rate clock source for the SCI is the bus clock.

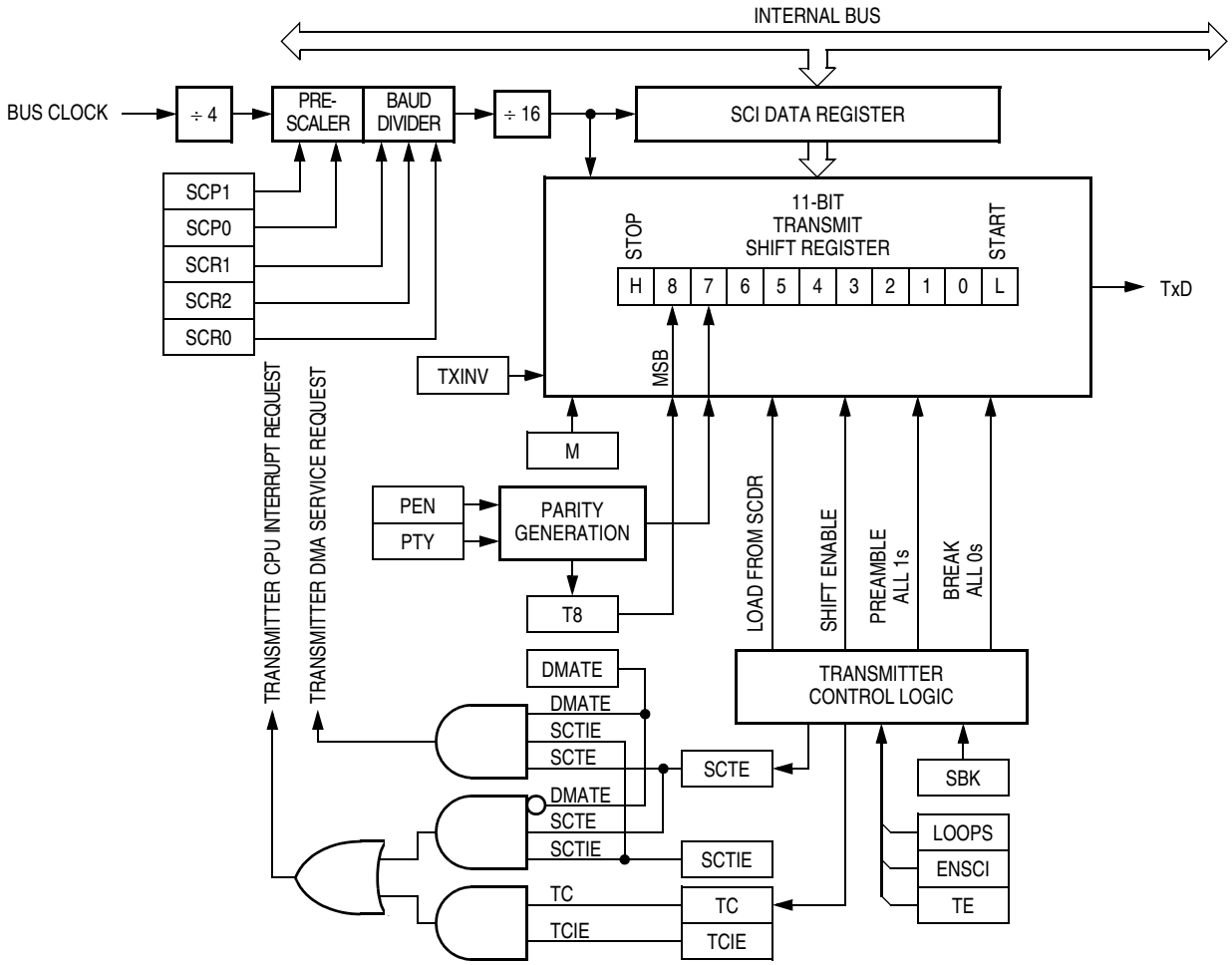


Figure 9-4. SCI Transmitter

9.4.2.1 Character Length

The transmitter can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCC1) determines character length. When transmitting 9-bit data, bit T8 in SCI control register 3 (SCC3) is the ninth bit (bit 8).

9.4.2.2 Character Transmission

During an SCI transmission, the transmit shift register shifts a character out to the TxD pin. The SCI data register (SCDR) is the write-only buffer between the internal data bus and the transmit shift register. To initiate an SCI transmission:

1. Enable the SCI by writing a logic 1 to the enable SCI bit (ENSCI) in SCI control register 1 (SCC1).
2. Enable the transmitter by writing a logic 1 to the transmitter enable bit (TE) in SCI control register 2 (SCC2).
3. Clear the SCI transmitter empty bit by first reading SCI status register 1 (SCS1) and then writing to the SCDR.
4. Repeat step 3 for each subsequent transmission.

At the start of a transmission, transmitter control logic automatically loads the transmit shift register with a preamble of logic 1s. After the preamble shifts out, control logic transfers the SCDR data 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.

The SCI transmitter empty bit, SCTE, in SCS1 becomes set when the SCDR transfers a byte to the transmit shift register. The SCTE bit indicates that the SCDR can accept new data from the internal data bus. If the SCI transmit interrupt enable bit, SCTIE, in SCC2 is also set, the SCTE bit generates a transmitter CPU interrupt request.

When the transmit shift register is not transmitting a character, the TxD pin goes to the idle condition, logic 1. If at any time software clears the ENSCI bit in SCI control register 1 (SCC1), the transmitter and receiver relinquish control of the port pin.

9.4.2.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCC2 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 SCC1. 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 character.

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 bit (FE) in SCS1
- Sets the SCI receiver full bit (SCRF) in SCS1
- Clears the SCI data register (SCDR)
- Clears the R8 bit in SCC3
- Sets the break flag bit (BKF) in SCS2
- May set the overrun (OR), noise flag (NF), parity error (PE), or reception in progress flag (RPF) bits

9.4.2.4 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 SCC1. The preamble is a synchronizing idle character that begins every transmission.

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 character currently being transmitted.

NOTE

When queueing an idle character, return the TE bit to logic 1 before the stop bit of the current character shifts out to the TxD pin. Setting TE after the stop bit appears on TxD causes data previously written to the SCDR to be lost.

Toggle the TE bit for a queued idle character when the SCTE bit becomes set and just before writing the next byte to the SCDR.

9.4.2.5 Inversion of Transmitted Output

The transmit inversion bit (TXINV) in SCI control register 1 (SCC1) reverses the polarity of transmitted data. All transmitted values, including idle, break, start, and stop bits, are inverted when TXINV is at logic 1. (See **9.8.1 SCI Control Register 1**.)

9.4.2.6 Transmitter Interrupts

These conditions can generate CPU interrupt requests from the SCI transmitter:

- SCI transmitter empty (SCTE) — The SCTE bit in SCS1 indicates that the SCDR has transferred a character to the transmit shift register. SCTE can generate a transmitter CPU interrupt request. Setting the SCI transmit interrupt enable bit, SCTIE, in SCC2 enables the SCTE bit to generate transmitter CPU interrupt requests.
- Transmission complete (TC) — The TC bit in SCS1 indicates that the transmit shift register and the SCDR are empty and that no break or idle character has been generated. The transmission complete interrupt enable bit, TCIE, in SCC2 enables the TC bit to generate transmitter CPU interrupt requests.

9.4.3 Receiver

Figure 9-5 shows the structure of the SCI receiver.

9.4.3.1 Character Length

The receiver can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCC1) determines character length. When receiving 9-bit data, bit R8 in SCI control register 2 (SCC2) is the ninth bit (bit 8). When receiving 8-bit data, bit R8 is a copy of the eighth bit (bit 7).

9.4.3.2 Character Reception

During an SCI reception, the receive shift register shifts characters in from the RxD pin. The SCI data register (SCDR) is the read-only buffer between the internal data bus and the receive shift register.

After a complete character shifts into the receive shift register, the data portion of the character transfers to the SCDR. The SCI receiver full bit, SCRF, in SCI status register 1 (SCS1) becomes set, indicating that the received byte can be read. If the SCI receive interrupt enable bit, SCRIE, in SCC2 is also set, the SCRF bit generates a receiver CPU interrupt request.

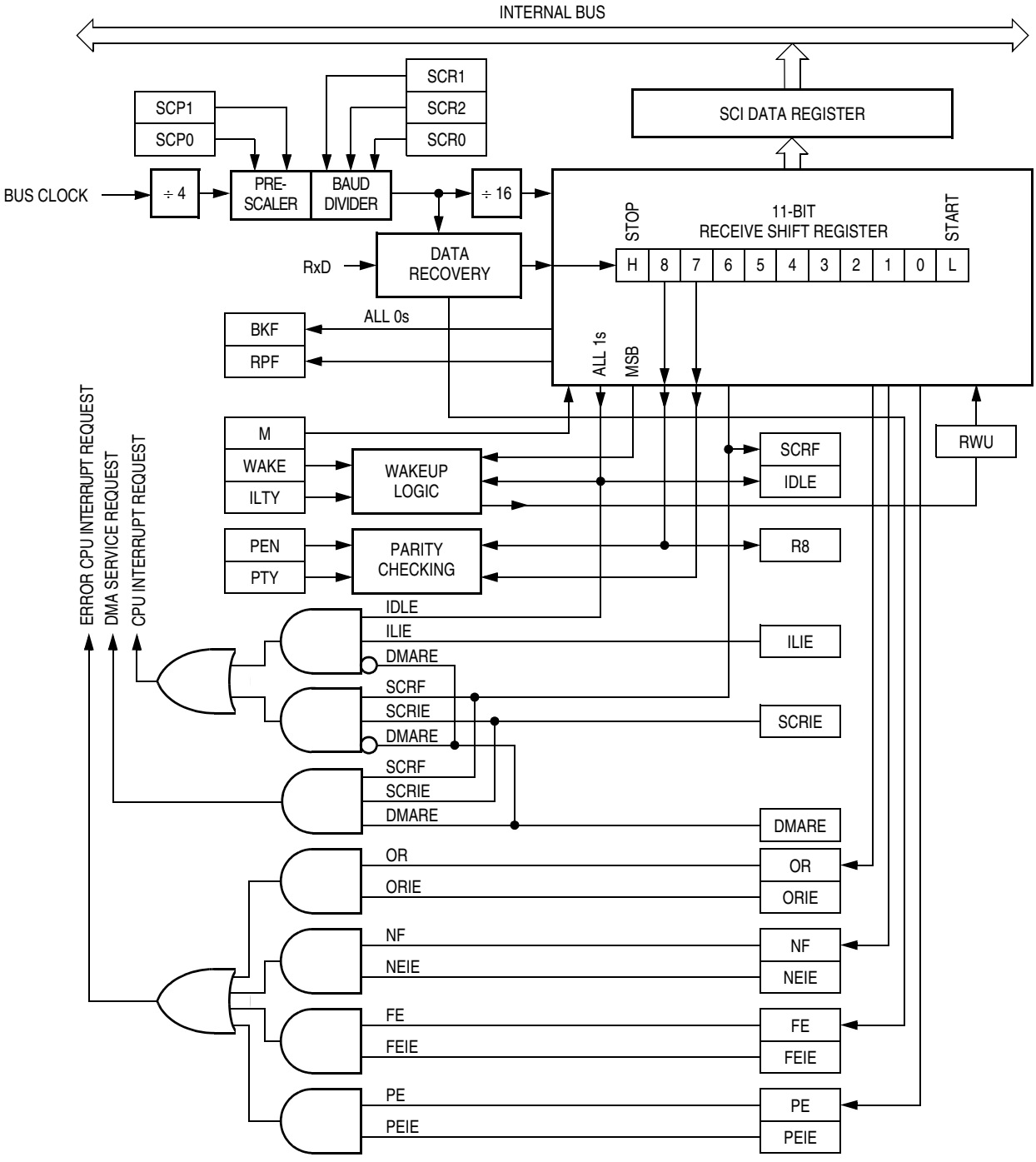


Figure 9-5. SCI Receiver Block Diagram

9.4.3.3 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 is resynchronized at the following times (see Figure 9-6):

- 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.

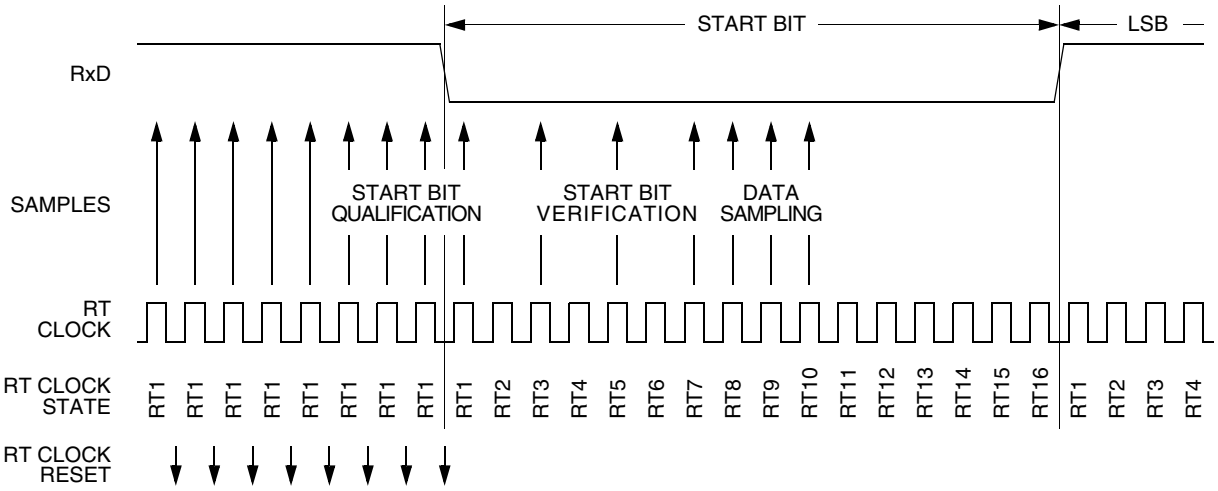


Figure 9-6. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Table 9-2 summarizes the results of the start bit verification samples.

Table 9-2. 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

Start bit verification is not successful if any two of the three verification samples are logic 1s. 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 9-3 summarizes the results of the data bit samples.

Table 9-3. 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.

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 9-4 summarizes the results of the stop bit samples.

Table 9-4. 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

9.4.3.4 Framing Errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming character, it sets the framing error bit, FE, in SCS1. A break character also sets the FE bit because a break character has no stop bit. The FE bit is set at the same time that the SCRF bit is set.

9.4.3.5 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 character, it resynchronizes the RT clock on any valid falling edge within the character. Resynchronization within characters corrects misalignments between transmitter bit times and receiver bit times.

Slow Data Tolerance

Figure 9-7 shows how much a slow received character 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.

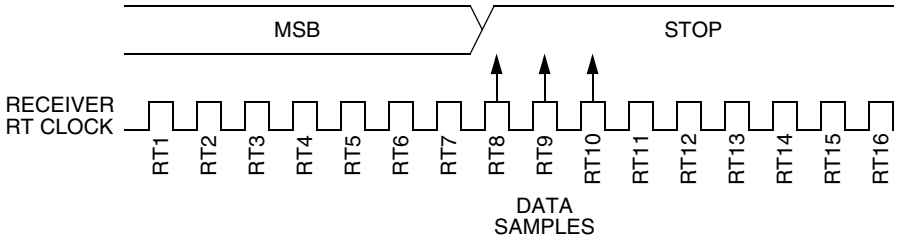


Figure 9-7. Slow Data

For an 8-bit 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 9-7, 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 character with no errors is

$$\left|\frac{154 - 147}{154}\right| \times 100 = 4.54\%$$

For a 9-bit 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 9-7, 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 9-8 shows how much a fast received character 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 there for the stop bit data samples at RT8, RT9, and RT10.

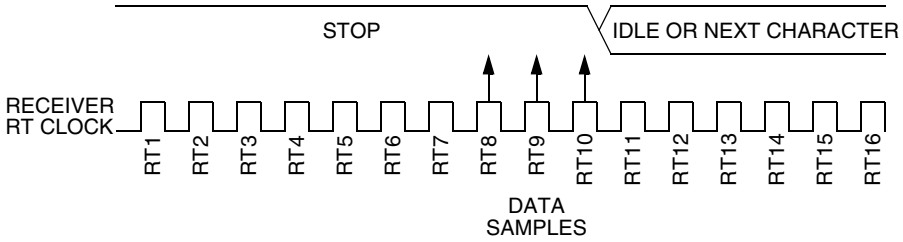


Figure 9-8. Fast Data

For an 8-bit 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 9-8, 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 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 9-8, 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

$$\left|\frac{170 - 176}{170}\right| \times 100 = 3.53\%$$

9.4.3.6 Receiver Wakeup

So that the MCU 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 SCC2 puts the receiver into a standby state during which receiver interrupts are disabled.

Depending on the state of the WAKE bit in SCC1, either of two conditions on the RxD pin can bring the receiver out of the standby state:

- Address mark — An address mark is a logic 1 in the most significant bit position of a received character. When the WAKE bit is set, an address mark wakes the receiver from the standby state by clearing the RWU bit. The address mark also sets the SCI receiver full bit, SCRF. Software can then compare the character containing the address mark to the user-defined address of the receiver. If they are the same, the receiver remains awake and processes the characters that follow. If they are not the same, software can set the RWU bit and put the receiver back into the standby state.
- Idle input line condition — When the WAKE bit is clear, an idle character on the RxD pin wakes the receiver from the standby state by clearing the RWU bit. The idle character that wakes the receiver does not set the receiver idle bit, IDLE, or the SCI receiver full bit, SCRF. The idle line type bit, ILTY, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit.

NOTE

With the WAKE bit clear, setting the RWU bit after the RxD pin has been idle may cause the receiver to wake up immediately.

9.4.3.7 Receiver Interrupts

The following sources can generate CPU interrupt requests from the SCI receiver:

- SCI receiver full (SCRF) — The SCRF bit in SCS1 indicates that the receive shift register has transferred a character to the SCDR. SCRF can generate a receiver CPU interrupt request. Setting the SCI receive interrupt enable bit, SCRIE, in SCC2 enables the SCRF bit to generate receiver CPU interrupts.
- Idle input (IDLE) — The IDLE bit in SCS1 indicates that 10 or 11 consecutive logic 1s shifted in from the RxD pin. The idle line interrupt enable bit, ILIE, in SCC2 enables the IDLE bit to generate CPU interrupt requests.

9.4.3.8 Error Interrupts

The following receiver error flags in SCS1 can generate CPU interrupt requests:

- Receiver overrun (OR) — The OR bit indicates that the receive shift register shifted in a new character before the previous character was read from the SCDR. The previous character remains in the SCDR, and the new character is lost. The overrun interrupt enable bit, ORIE, in SCC3 enables OR to generate SCI error CPU interrupt requests.
- Noise flag (NF) — The NF bit is set when the SCI detects noise on incoming data or break characters, including start, data, and stop bits. The noise error interrupt enable bit, NEIE, in SCC3 enables NF to generate SCI error CPU interrupt requests.
- Framing error (FE) — The FE bit in SCS1 is set when a logic 0 occurs where the receiver expects a stop bit. The framing error interrupt enable bit, FEIE, in SCC3 enables FE to generate SCI error CPU interrupt requests.
- Parity error (PE) — The PE bit in SCS1 is set when the SCI detects a parity error in incoming data. The parity error interrupt enable bit, PEIE, in SCC3 enables PE to generate SCI error CPU interrupt requests.

9.5 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power- consumption standby modes.

9.5.1 Wait Mode

The SCI module remains active after the execution of a WAIT instruction. In wait mode, the SCI module registers are not accessible by the CPU. Any enabled CPU interrupt request from the SCI module can bring the MCU out of wait mode.

If SCI module functions are not required during wait mode, reduce power consumption by disabling the module before executing the WAIT instruction.

Refer to 5.6 Low-Power Modes for information on exiting wait mode.

9.5.2 Stop Mode

The SCI module is inactive after the execution of a STOP instruction. The STOP instruction does not affect SCI register states. SCI module operation resumes after an external interrupt.

Because the internal clock is inactive during stop mode, entering stop mode during an SCI transmission or reception results in invalid data.

Refer to 5.6 Low-Power Modes for information on exiting stop mode.

9.6 SCI During Break Module Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the break flag control register (BFCR) enables software to clear status bits during the break state.

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at logic 0. After the break, doing the second step clears the status bit.

9.7 I/O Signals

The two SCI I/O pins are:

- PTD6/TxD — Transmit data
- PTD7/RxD — Receive data

9.7.1 TxD (Transmit Data)

The PTD6/TxD pin is the serial data output from the SCI transmitter.

9.7.2 RxD (Receive Data)

The PTD7/RxD pin is the serial data input to the SCI receiver.

9.8 I/O Registers

These I/O registers control and monitor SCI operation:

- SCI control register 1 (SCC1)
- SCI control register 2 (SCC2)
- SCI control register 3 (SCC3)
- SCI status register 1 (SCS1)
- SCI status register 2 (SCS2)
- SCI data register (SCDR)
- SCI baud rate register (SCBR)

9.8.1 SCI Control Register 1

SCI control register 1:

- Enables loop mode operation
- Enables the SCI
- Controls output polarity
- Controls character length
- Controls SCI wakeup method
- Controls idle character detection
- Enables parity function
- Controls parity type

Address: $0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
Reset:	0	0	0	0	0	0	0	0

Figure 9-9. SCI Control Register 1 (SCC1)

LOOPS — Loop Mode Select Bit

This read/write bit enables loop mode operation. In loop mode 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 loop mode. Reset clears the LOOPS bit.

1 = Loop mode enabled
0 = Normal operation enabled

ENSCI — Enable SCI Bit

This read/write bit enables the SCI and the SCI baud rate generator. Clearing ENSCI sets the SCTE and TC bits in SCI status register 1 and disables transmitter interrupts. Reset clears the ENSCI bit.

1 = SCI enabled
0 = SCI disabled

TXINV — Transmit Inversion Bit

This read/write bit reverses the polarity of transmitted data. Reset clears the TXINV bit.

1 = Transmitter output inverted
0 = Transmitter output not inverted

NOTE

Setting the TXINV bit inverts all transmitted values, including idle, break, start, and stop bits.

M — Mode (Character Length) Bit

This read/write bit determines whether SCI characters are eight or nine bits long. (See Table 9-5.) The ninth bit can serve as an extra stop bit, as a receiver wakeup signal, or as a parity bit. Reset clears the M bit.

1 = 9-bit SCI characters
0 = 8-bit SCI characters

WAKE — Wakeup Condition Bit

This read/write bit determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit position of a received character or an idle condition on the RxD pin. Reset clears the WAKE bit.

1 = Address mark wakeup
0 = Idle line wakeup

ILTY — Idle Line Type Bit

This read/write bit determines when the SCI 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. Reset clears the ILTY bit.

1 = Idle character bit count begins after stop bit
0 = Idle character bit count begins after start bit

PEN — Parity Enable Bit

This read/write bit enables the SCI parity function. (See Table 9-5.) When enabled, the parity function inserts a parity bit in the most significant bit position. (See Figure 9-3.) Reset clears the PEN bit.

1 = Parity function enabled
0 = Parity function disabled

PTY — Parity Bit

This read/write bit determines whether the SCI generates and checks for odd parity or even parity. (See Table 9-5.) Reset clears the PTY bit.

1 = Odd parity
0 = Even parity

NOTE

Changing the PTY bit in the middle of a transmission or reception can generate a parity error.

Table 9-5. Character Format Selection

Control Bits		Character Format				
M	PEN and PTY	Start Bits	Data Bits	Parity	Stop Bits	Character Length
0	0X	1	8	None	1	10 bits
1	0X	1	9	None	1	11 bits
0	10	1	7	Even	1	10 bits
0	11	1	7	Odd	1	10 bits
1	10	1	8	Even	1	11 bits
1	11	1	8	Odd	1	11 bits

9.8.2 SCI Control Register 2

SCI control register 2:

- Enables the following CPU interrupt requests:
  - Enables the SCTE bit to generate transmitter CPU interrupt requests
  - Enables the TC bit to generate transmitter CPU interrupt requests
  - Enables the SCRF bit to generate receiver CPU interrupt requests
  - Enables the IDLE bit to generate receiver CPU interrupt requests
- Enables the transmitter
- Enables the receiver
- Enables SCI wakeup
- Transmits SCI break characters

Address: $0014

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
Reset:	0	0	0	0	0	0	0	0

Figure 9-10. SCI Control Register 2 (SCC2)

SCTIE — SCI Transmit Interrupt Enable Bit

This read/write bit enables the SCTE bit to generate SCI transmitter CPU interrupt requests. Reset clears the SCTIE bit.

1 = SCTE enabled to generate CPU interrupt
0 = SCTE not enabled to generate CPU interrupt

TCIE — Transmission Complete Interrupt Enable Bit

This read/write bit enables the TC bit to generate SCI transmitter CPU interrupt requests. Reset clears the TCIE bit.

1 = TC enabled to generate CPU interrupt requests
0 = TC not enabled to generate CPU interrupt requests

SCRIE — SCI Receive Interrupt Enable Bit

This read/write bit enables the SCRF bit to generate SCI receiver CPU interrupt requests. Reset clears the SCRIE bit.

1 = SCRF enabled to generate CPU interrupt
0 = SCRF not enabled to generate CPU interrupt

ILIE — Idle Line Interrupt Enable Bit

This read/write bit enables the IDLE bit to generate SCI receiver CPU interrupt requests. Reset clears the ILIE bit.

1 = IDLE enabled to generate CPU interrupt requests
0 = IDLE not enabled to generate CPU interrupt requests

TE — Transmitter Enable Bit

Setting this read/write bit begins the transmission by sending a preamble of 10 or 11 logic 1s from the transmit shift register to the TxD pin. If software clears the TE bit, the transmitter completes any transmission in progress before the TxD returns to the idle condition (logic 1). Clearing and then setting TE during a transmission queues an idle character to be sent after the character currently being transmitted. Reset clears the TE bit.

1 = Transmitter enabled
0 = Transmitter disabled

NOTE

Writing to the TE bit is not allowed when the enable SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.

RE — Receiver Enable Bit

Setting this read/write bit enables the receiver. Clearing the RE bit disables the receiver but does not affect receiver interrupt flag bits. Reset clears the RE bit.

1 = Receiver enabled
0 = Receiver disabled

NOTE

Writing to the RE bit is not allowed when the enable SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.

RWU — Receiver Wakeup Bit

This read/write bit puts the receiver in a standby state during which receiver interrupts are disabled. The WAKE bit in SCC1 determines whether an idle input or an address mark brings the receiver out of the standby state and clears the RWU bit. Reset clears the RWU bit.

1 = Standby state
0 = Normal operation

SBK — Send Break Bit

Setting and then clearing this read/write bit transmits a break character followed by a logic 1. The logic 1 after the break character guarantees recognition of a valid start bit. If SBK remains set, the transmitter continuously transmits break characters with no logic 1s between them. Reset clears the SBK bit.

1 = Transmit break characters
0 = No break characters being transmitted

NOTE

Do not toggle the SBK bit immediately after setting the SCTE bit. Toggling SBK before the preamble begins causes the SCI to send a break character instead of a preamble.

9.8.3 SCI Control Register 3

SCI control register 3:

- Stores the ninth SCI data bit received and the ninth SCI data bit to be transmitted
- Enables these interrupts:
  - Receiver overrun interrupts
  - Noise error interrupts
  - Framing error interrupts
- Parity error interrupts

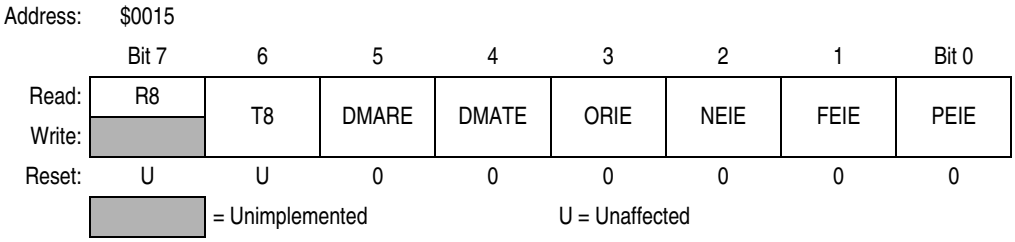


Figure 9-11. SCI Control Register 3 (SCC3)

R8 — Received Bit 8

When the SCI is receiving 9-bit characters, R8 is the read-only ninth bit (bit 8) of the received character. R8 is received at the same time that the SCDR receives the other 8 bits.

When the SCI is receiving 8-bit characters, R8 is a copy of the eighth bit (bit 7). Reset has no effect on the R8 bit.

T8 — Transmitted Bit 8

When the SCI is transmitting 9-bit characters, T8 is the read/write ninth bit (bit 8) of the transmitted character. T8 is loaded into the transmit shift register at the same time that the SCDR is loaded into the transmit shift register. Reset has no effect on the T8 bit.

DMARE — DMA Receive Enable Bit

CAUTION

The DMA module is not included on this MCU. Writing a logic 1 to DMARE or DMATE may adversely affect MCU performance.

1 = DMA not enabled to service SCI receiver DMA service requests generated by the SCRF bit (SCI receiver CPU interrupt requests enabled)

0 = DMA not enabled to service SCI receiver DMA service requests generated by the SCRF bit (SCI receiver CPU interrupt requests enabled)

DMATE — DMA Transfer Enable Bit

CAUTION

The DMA module is not included on this MCU. Writing a logic 1 to DMARE or DMATE may adversely affect MCU performance.

1 = SCTE DMA service requests enabled; SCTE CPU interrupt requests disabled

0 = SCTE DMA service requests disabled; SCTE CPU interrupt requests enabled

ORIE — Receiver Overrun Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the receiver overrun bit, OR.

1 = SCI error CPU interrupt requests from OR bit enabled

0 = SCI error CPU interrupt requests from OR bit disabled

NEIE — Receiver Noise Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the noise error bit, NE. Reset clears NEIE.

1 = SCI error CPU interrupt requests from NE bit enabled
0 = SCI error CPU interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the framing error bit, FE. Reset clears FEIE.

1 = SCI error CPU interrupt requests from FE bit enabled
0 = SCI error CPU interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the parity error bit, PE. (See **9.8.4 SCI Status Register 1**.) Reset clears PEIE.

1 = SCI error CPU interrupt requests from PE bit enabled
0 = SCI error CPU interrupt requests from PE bit disabled

9.8.4 SCI Status Register 1

SCI status register 1 (SCS1) contains flags to signal these conditions:

- Transfer of SCDR data to transmit shift register complete
- Transmission complete
- Transfer of receive shift register data to SCDR complete
- Receiver input idle
- Receiver overrun
- Noisy data
- Framing error
- Parity error

Address: $016

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
Write:								
Reset:	1	1	0	0	0	0	0	0

= Unimplemented

Figure 9-12. SCI Status Register 1 (SCS1)

SCTE — SCI Transmitter Empty Bit

This clearable, read-only bit is set when the SCDR transfers a character to the transmit shift register. SCTE can generate an SCI transmitter CPU interrupt request. When the SCTIE bit in SCC2 is set, SCTE generates an SCI transmitter CPU interrupt request. In normal operation, clear the SCTE bit by reading SCS1 with SCTE set and then writing to SCDR. Reset sets the SCTE bit.

1 = SCDR data transferred to transmit shift register
0 = SCDR data not transferred to transmit shift register

TC — Transmission Complete Bit

This read-only bit is set when the SCTE bit is set, and no data, preamble, or break character is being transmitted. TC generates an SCI transmitter CPU interrupt request if the TCIE bit in SCC2 is also set. TC is automatically cleared when data, preamble or break is queued and ready to be sent. There may be up to 1.5 transmitter clocks of latency between queueing data, preamble, and break and the transmission actually starting. Reset sets the TC bit.

1 = No transmission in progress
0 = Transmission in progress

SCRF — SCI Receiver Full Bit

This clearable, read-only bit is set when the data in the receive shift register transfers to the SCI data register. SCRF can generate an SCI receiver CPU interrupt request. When the SCRIE bit in SCC2 is set, SCRF generates a CPU interrupt request. In normal operation, clear the SCRF bit by reading SCS1 with SCRF set and then reading the SCDR. Reset clears SCRF.

1 = Received data available in SCDR
0 = Data not available in SCDR

IDLE — Receiver Idle Bit

This clearable, read-only bit is set when 10 or 11 consecutive logic 1s appear on the receiver input. IDLE generates an SCI receiver CPU interrupt request if the ILIE bit in SCC2 is also set. Clear the IDLE bit by reading SCS1 with IDLE set and then reading the SCDR. After the receiver is enabled, it must receive a valid character that sets the SCRF bit before an idle condition can set the IDLE bit. Also, after the IDLE bit has been cleared, a valid character must again set the SCRF bit before an idle condition can set the IDLE bit. Reset clears the IDLE bit.

1 = Receiver input idle
0 = Receiver input active (or idle since the IDLE bit was cleared)

OR — Receiver Overrun Bit

This clearable, read-only bit is set when software fails to read the SCDR before the receive shift register receives the next character. The OR bit generates an SCI error CPU interrupt request if the ORIE bit in SCC3 is also set. The data in the shift register is lost, but the data already in the SCDR is not affected. Clear the OR bit by reading SCS1 with OR set and then reading the SCDR. Reset clears the OR bit.

1 = Receive shift register full and SCRF = 1
0 = No receiver overrun

Software latency may allow an overrun to occur between reads of SCS1 and SCDR in the flag-clearing sequence. Figure 9-13 shows the normal flag-clearing sequence and an example of an overrun caused by a delayed flag-clearing sequence. The delayed read of SCDR does not clear the OR bit because OR was not set when SCS1 was read. Byte 2 caused the overrun and is lost. The next flag-clearing sequence reads byte 3 in the SCDR instead of byte 2.

In applications that are subject to software latency or in which it is important to know which byte is lost due to an overrun, the flag-clearing routine can check the OR bit in a second read of SCS1 after reading the data register.

NF — Receiver Noise Flag Bit

This clearable, read-only bit is set when the SCI detects noise on the RxD pin. NF generates an SCI error CPU interrupt request if the NEIE bit in SCC3 is also set. Clear the NF bit by reading SCS1 and then reading the SCDR. Reset clears the NF bit.

1 = Noise detected
0 = No noise detected

FE — Receiver Framing Error Bit

This clearable, read-only bit is set when a logic 0 is accepted as the stop bit. FE generates an SCI error CPU interrupt request if the FEIE bit in SCC3 also is set. Clear the FE bit by reading SCS1 with FE set and then reading the SCDR. Reset clears the FE bit.

1 = Framing error detected
0 = No framing error detected

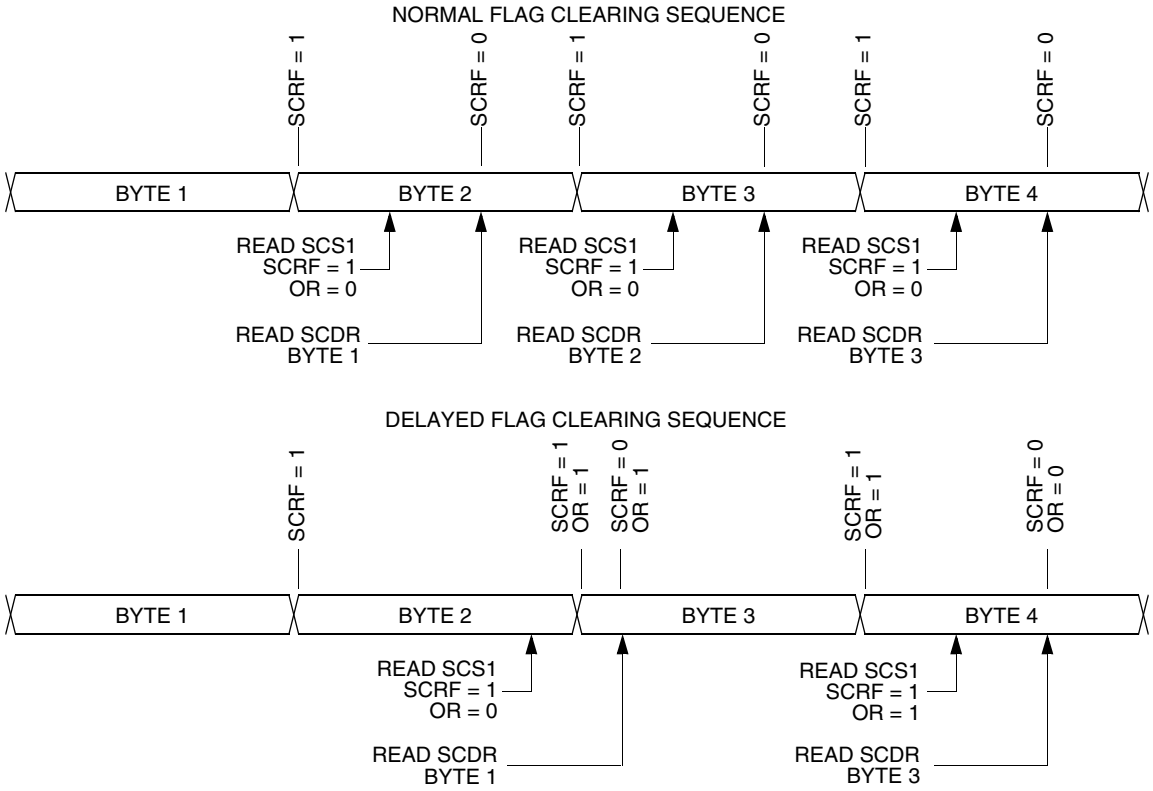


Figure 9-13. Flag Clearing Sequence

PE — Receiver Parity Error Bit

This clearable, read-only bit is set when the SCI detects a parity error in incoming data. PE generates an SCI error CPU interrupt request if the PEIE bit in SCC3 is also set. Clear the PE bit by reading SCS1 with PE set and then reading the SCDR. Reset clears the PE bit.

1 = Parity error detected
0 = No parity error detected

9.8.5 SCI Status Register 2

SCI status register 2 contains flags to signal the following conditions:

- Break character detected
- Incoming data

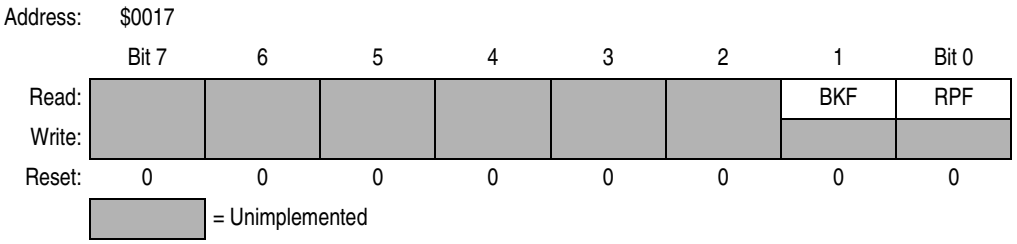


Figure 9-14. SCI Status Register 2 (SCS2)

BKF — Break Flag Bit

This clearable, read-only bit is set when the SCI detects a break character on the RxD pin. In SCS1, the FE and SCRF bits are also set. In 9-bit character transmissions, the R8 bit in SCC3 is cleared. BKF does not generate a CPU interrupt request. Clear BKF by reading SCS2 with BKF set and then reading the SCDR. Once cleared, BKF can become set again only after logic 1s again appear on the RxD pin followed by another break character. Reset clears the BKF bit.

1 = Break character detected
0 = No break character detected

RPF — Reception in Progress Flag Bit

This read-only bit is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RPF does not generate an interrupt request. RPF is reset after the receiver detects false start bits (usually from noise or a baud rate mismatch) or when the receiver detects an idle character. Polling RPF before disabling the SCI module or entering stop mode can show whether a reception is in progress.

1 = Reception in progress
0 = No reception in progress

9.8.6 SCI Data Register

The SCI data register (SCDR) is the buffer between the internal data bus and the receive and transmit shift registers. Reset has no effect on data in the SCI data register.

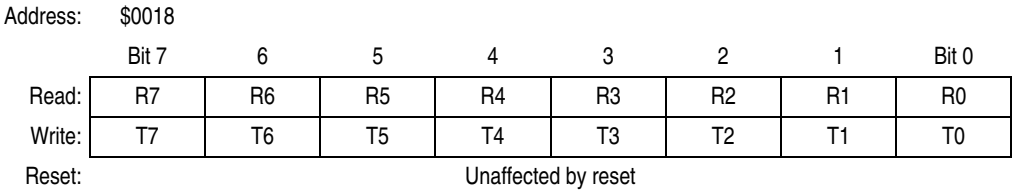


Figure 9-15. SCI Data Register (SCDR)

R7/T7–R0/T0 — Receive/Transmit Data Bits

Reading the SCDR accesses the read-only received data bits, R[7:0]. Writing to the SCDR writes the data to be transmitted, T[7:0]. Reset has no effect on the SCDR.

NOTE

Do not use read/modify/write instructions on the SCI data register.

9.8.7 SCI Baud Rate Register

The baud rate register (SCBR) selects the baud rate for both the receiver and the transmitter.

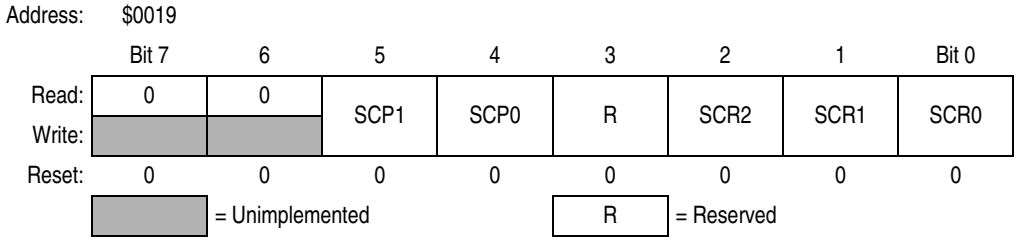


Figure 9-16. SCI Baud Rate Register (SCBR)

SCP1 and SCP0 — SCI Baud Rate Prescaler Bits

These read/write bits select the baud rate prescaler divisor as shown in Table 9-6. Reset clears SCP1 and SCP0.

Table 9-6. SCI Baud Rate Prescaling

SCP1 and SCP0	Prescaler Divisor (PD)
00	1
01	3
10	4
11	13

SCR2–SCR0 — SCI Baud Rate Select Bits

These read/write bits select the SCI baud rate divisor as shown in Table 9-7. Reset clears SCR2–SCR0.

Table 9-7. SCI Baud Rate Selection

SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

Use this formula to calculate the SCI baud rate:

$$\text{baud rate} = \frac{\text{SCI clock source}}{64 \times \text{PD} \times \text{BD}}$$

where:

SCI clock source = bus clock
PD = prescaler divisor
BD = baud rate divisor

Table 9-8 shows the SCI baud rates that can be generated with a 4.9152MHz bus clock.

Table 9-8. SCI Baud Rate Selection Examples

SCP1 and SCP0	Prescaler Divisor (PD)	SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)	Baud Rate (BUS CLOCK=4.9152MHz)
00	1	000	1	76,800
00	1	001	2	38,400
00	1	010	4	19,200
00	1	011	8	9,600
00	1	100	16	4,800
00	1	101	32	2,400
00	1	110	64	1,200
00	1	111	128	600
01	3	000	1	25,600
01	3	001	2	12,800
01	3	010	4	6,400
01	3	011	8	3,200
01	3	100	16	1,600
01	3	101	32	800
01	3	110	64	400
01	3	111	128	200
10	4	000	1	19,200
10	4	001	2	9,600
10	4	010	4	4,800
10	4	011	8	2,400
10	4	100	16	1,200
10	4	101	32	600
10	4	110	64	300
10	4	111	128	150
11	13	000	1	5,908
11	13	001	2	2,954
11	13	010	4	1,477
11	13	011	8	739
11	13	100	16	369
11	13	101	32	185
11	13	110	64	92
11	13	111	128	46

Chapter 10
Analog-to-Digital Converter (ADC)

10.1 Introduction

This section describes the 13-channel, 8-bit linear successive approximation analog-to-digital converter (ADC).

10.2 Features

Features of the ADC module include:

- 13 channels with multiplexed input
- Linear successive approximation with monotonicity
- 8-bit resolution
- Single or continuous conversion
- Conversion complete flag or conversion complete interrupt

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003C	ADC Status and Control Register (ADSCR)	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
$003D	ADC Data Register (ADR)	Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
		Write:								
		Reset:	Indeterminate after reset							
$003E	ADC Input Clock Register (ADICLK)	Read:	ADIV2	ADIV1	ADIV0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

Figure 10-1. ADC I/O Register Summary

10.3 Functional Description

Thirteen ADC channels are available for sampling external sources at pins PTB0–PTB7, PTD0–PTD3, and ADC12/T2CLK. An analog multiplexer allows the single ADC converter to select one of the 13 ADC channels as ADC voltage input (ADCVIN). ADCVIN is converted by the successive approximation register-based counters. The ADC resolution is 8 bits. When the conversion is completed, ADC puts the result in the ADC data register and sets a flag or generates an interrupt.
Figure 10-2 shows a block diagram of the ADC.

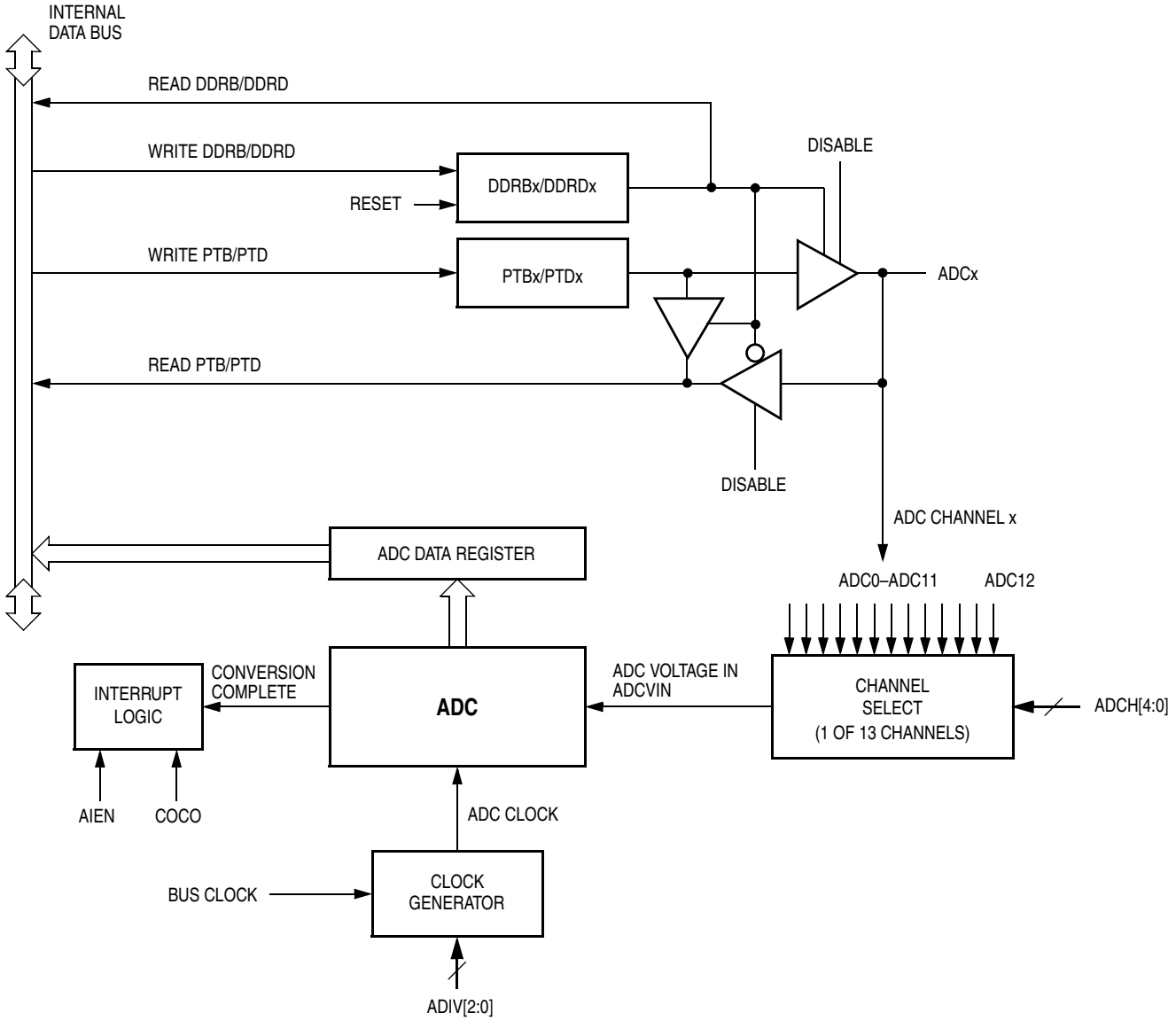


Figure 10-2. ADC Block Diagram

10.3.1 ADC Port I/O Pins

PTB0–PTB7 and PTD0–PTD3 are general-purpose I/O pins that are shared with the ADC channels. The channel select bits (ADC status and control register, $003C), define which ADC channel/port pin will be used as the input signal. The ADC overrides the port I/O logic by forcing that pin as input to the ADC. The remaining ADC channels/port pins are controlled by the port I/O logic and can be used as general-purpose I/O. Writes to the port register or DDR will not have any affect on the port pin that is selected by the ADC. Read of a port pin which is in use by the ADC will return a logic 0 if the corresponding DDR bit is at logic 0. If the DDR bit is at logic 1, the value in the port data latch is read.

10.3.2 Voltage Conversion

When the input voltage to the ADC equals V_{DD}, the ADC converts the signal to $FF (full scale). If the input voltage equals V_{SS}, the ADC converts it to $00. Input voltages between V_{DD} and V_{SS} are a straight-line linear conversion. All other input voltages will result in $FF if greater than V_{DD} and $00 if less than V_{SS}.

NOTE

Input voltage should not exceed the analog supply voltages.

10.3.3 Conversion Time

Fourteen ADC internal clocks are required to perform one conversion. The ADC starts a conversion on the first rising edge of the ADC internal clock immediately following a write to the ADSCR. If the ADC internal clock is selected to run at 1MHz, then one conversion will take 14μs to complete. With a 1MHz ADC internal clock the maximum sample rate is 71.43kHz.

$$\text{Conversion Time} = \frac{\text{14 ADC Clock Cycles}}{\text{ADC Clock Frequency}}$$

$$\text{Number of Bus Cycles} = \text{Conversion Time} \times \text{Bus Frequency}$$

10.3.4 Continuous Conversion

In the continuous conversion mode, the ADC continuously converts the selected channel filling the ADC data register with new data after each conversion. Data from the previous conversion will be overwritten whether that data has been read or not. Conversions will continue until the ADCO bit is cleared. The COCO bit (ADC status and control register, $003C) is set after each conversion and can be cleared by writing the ADC status and control register or reading of the ADC data register.

10.3.5 Accuracy and Precision

The conversion process is monotonic and has no missing codes.

10.4 Interrupts

When the AIEN bit is set, the ADC module is capable of generating a CPU interrupt after each ADC conversion. A CPU interrupt is generated if the COCO bit is at logic 0. The COCO bit is not used as a conversion complete flag when interrupts are enabled.

10.5 Low-Power Modes

The following subsections describe the ADC in low-power modes.

10.5.1 Wait Mode

The ADC continues normal operation during wait mode. Any enabled CPU interrupt request from the ADC can bring the MCU out of wait mode. If the ADC is not required to bring the MCU out of wait mode, power down the ADC by setting the ADCH[4:0] bits in the ADC status and control register to logic 1’s before executing the WAIT instruction.

10.5.2 Stop Mode

The ADC module is inactive after the execution of a STOP instruction. Any pending conversion is aborted. ADC conversions resume when the MCU exits stop mode. Allow one conversion cycle to stabilize the analog circuitry before attempting a new ADC conversion after exiting stop mode.

10.6 I/O Signals

The ADC module has 12 channels that are shared with I/O port B and port D, and one channel on ADC12/T2CLK pin.

10.6.1 ADC Voltage In (ADCVIN)

ADCVIN is the input voltage signal from one of the 13 ADC channels to the ADC module.

10.7 I/O Registers

These I/O registers control and monitor ADC operation:

- ADC status and control register (ADSCR)
- ADC data register (ADR)
- ADC clock register (ADICLK)

10.7.1 ADC Status and Control Register

The following paragraphs describe the function of the ADC status and control register.

Address: $003C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
Write:								
Reset:	0	0	0	1	1	1	1	1

= Unimplemented

Figure 10-3. ADC Status and Control Register (ADSCR)

COCO — Conversions Complete Bit

When the AIEN bit is a logic 0, the COCO is a read-only bit which is set each time a conversion is completed. This bit is cleared whenever the ADC status and control register is written or whenever the ADC data register is read. Reset clears this bit.

1 = Conversion completed (AIEN = 0)
0 = Conversion not completed (AIEN = 0)

When the AIEN bit is a logic 1 (CPU interrupt enabled), the COCO is a read-only bit, and will always be logic 0 when read.

AIEN — ADC Interrupt Enable Bit

When this bit is set, an interrupt is generated at the end of an ADC conversion. The interrupt signal is cleared when the data register is read or the status/control register is written. Reset clears the AIEN bit.

1 = ADC interrupt enabled
0 = ADC interrupt disabled

ADCO — ADC Continuous Conversion Bit

When set, the ADC will convert samples continuously and update the ADR register at the end of each conversion. Only one conversion is allowed when this bit is cleared. Reset clears the ADCO bit.

1 = Continuous ADC conversion
0 = One ADC conversion

ADCH[4:0] — ADC Channel Select Bits

ADCH[4:0] form a 5-bit field which is used to select one of the ADC channels. The five channel select bits are detailed in the following table. Care should be taken when using a port pin as both an analog and a digital input simultaneously to prevent switching noise from corrupting the analog signal. (See Table 10-1.)

The ADC subsystem is turned off when the channel select bits are all set to one. This feature allows for reduced power consumption for the MCU when the ADC is not used. Reset sets all of these bits to a logic 1.

NOTE

Recovery from the disabled state requires one conversion cycle to stabilize.

Table 10-1. MUX Channel Select

ADCH4	ADCH3	ADCH2	ADCH1	ADCH0	ADC Channel	Input Select
0	0	0	0	0	ADC0	PTB0
0	0	0	0	1	ADC1	PTB1
0	0	0	1	0	ADC2	PTB2
0	0	0	1	1	ADC3	PTB3
0	0	1	0	0	ADC4	PTB4
0	0	1	0	1	ADC5	PTB5
0	0	1	1	0	ADC6	PTB6
0	0	1	1	1	ADC7	PTB7
0	1	0	0	0	ADC8	PTD3
0	1	0	0	1	ADC9	PTD2
0	1	0	1	0	ADC10	PTD1
0	1	0	1	1	ADC11	PTD0
0	1	1	0	0	ADC12	ADC12/T2CLK
0 : 1	1 : 1	1 : 0	0 : 1	1 : 0	—	Unused(1)
1	1	0	1	1	—	Reserved
1	1	1	0	0	—	Reserved
1	1	1	0	1		V_{DD}(2)
1	1	1	1	0		V_{SS}(2)
1	1	1	1	1		ADC power off

1. If any unused channels are selected, the resulting ADC conversion will be unknown.
2. The voltage levels supplied from internal reference nodes as specified in the table are used to verify the operation of the ADC converter both in production test and for user applications.

10.7.2 ADC Data Register

One 8-bit result register is provided. This register is updated each time an ADC conversion completes.

Address:	$003D							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
Write:								
Reset:	Indeterminate after reset							

= Unimplemented

Figure 10-4. ADC Data Register (ADR)

10.7.3 ADC Input Clock Register

This register selects the clock frequency for the ADC.

Address:	$003E							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ADIV2	ADIV1	ADIV0	0	0	0	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-5. ADC Input Clock Register (ADICLK)

ADIV[2:0] — ADC Clock Prescaler Bits

ADIV[2:0] form a 3-bit field which selects the divide ratio used by the ADC to generate the internal ADC clock. Table 10-2 shows the available clock configurations. The ADC clock should be set to approximately 1 MHz.

Table 10-2. ADC Clock Divide Ratio

ADIV2	ADIV1	ADIV0	ADC Clock Rate
0	0	0	Bus Clock ÷ 1
0	0	1	Bus Clock ÷ 2
0	1	0	Bus Clock ÷ 4
0	1	1	Bus Clock ÷ 8
1	X	X	Bus Clock ÷ 16

X = don't care

Chapter 11
Input/Output (I/O) Ports

11.1 Introduction

Twenty six (26) bidirectional input-output (I/O) pins form four parallel ports. All I/O pins are programmable as inputs or outputs.

NOTE

Connect any unused I/O pins to an appropriate logic level, either V_{DD} or V_{SS}. Although the I/O ports do not require termination for proper operation, termination reduces excess current consumption and the possibility of electrostatic damage.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PTA)	Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
		Reset:	Unaffected by reset							
$0001	Port B Data Register (PTB)	Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Reset:	Unaffected by reset							
$0003	Port D Data Register (PTD)	Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Reset:	Unaffected by reset							
$0004	Data Direction Register A (DDRA)	Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B (DDRB)	Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Reset:	0	0	0	0	0	0	0	0
$0007	Data Direction Register D (DDRD)	Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PTE)	Read: Write:							PTE1	PTE0
		Reset:	Unaffected by reset							
$000A	Port D Control Register (PDCR)	Read: Write:	0	0	0	0	SLOWD7	SLOWD6	PTDPU7	PTDPU6
		Reset:	0	0	0	0	0	0	0	0
$000C	Data Direction Register E (DDRE)	Read: Write:							DDRE1	DDRE0
		Reset:	0	0	0	0	0	0	0	0

Figure 11-1. I/O Port Register Summary

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000D	Port A Input Pull-up Enable Register (PTAPUE)	Read: Write:	PTA6EN	PTAPUE6	PTAPUE5	PTAPUE4	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
		Reset:	0	0	0	0	0	0	0	0
$000E	PTA7 Input Pull-up Enable Register (PTA7PUE)	Read: Write:	PTAPUE7							
		Reset:	0	0	0	0	0	0	0	0

Figure 11-1. I/O Port Register Summary

Table 11-1. Port Control Register Bits Summary

Port	Bit	DDR	Module Control			Pin
			Module	Register	Control Bit	
A	0	DDRA0	KBI	KBIER ($001B)	KBIE0	PTA0/KBI0
	1	DDRA1			KBIE1	PTA1/KBI1
	2	DDRA2			KBIE2	PTA2/KBI2
	3	DDRA3			KBIE3	PTA3/KBI3
	4	DDRA4			KBIE4	PTA4/KBI4
	5	DDRA5			KBIE5	PTA5/KBI5
	6	DDRA6	OSC KBI	PTAPUE ($000D) KBIER ($001B)	PTA6EN KBIE6	RCCLK/PTA6/KBI6(1)
	7	DDRA7	KBI	KBIER ($001B)	KBIE7	PTA7/KBI7
B	0	DDRB0	ADC	ADSCR ($003C)	ADCH[4:0]	PTB0/ADC0
	1	DDRB1				PTB1/ADC1
	2	DDRB2				PTB2/ADC2
	3	DDRB3				PTB3/ADC3
	4	DDRB4				PTB4/ADC4
	5	DDRB5				PTB5/ADC5
	6	DDRB6				PTB6/ADC6
	7	DDRB7				PTB7/ADC7
D	0	DDRD0	ADC	ADSCR ($003C)	ADCH[4:0]	PTD0/ADC11
	1	DDRD1				PTD1/ADC10
	2	DDRD2				PTD2/ADC9
	3	DDRD3				PTD3/ADC8
	4	DDRD4	TIM1	T1SC0 ($0025)	ELS0B:ELS0A	PTD4/T1CH0
	5	DDRD5		T1SC1 ($0028)	ELS1B:ELS1A	PTD5/T1CH1
	6	DDRD6	SCI	SCC1 ($0013)	ENSCI	PTD6/TxD
	7	DDRD7				PTD7/RxD
E	0	DDRE0	TIM2	T2SC0 ($0035)	ELS0B:ELS0A	PTE0/T2CH0
	1	DDRE1		T2SC1 ($0038)	ELS1B:ELS1A	PTE1/T2CH1

1. RCCLK/PTA6/KBI6 pin is only available when OSCSEL=0 (RC option); PTAPUE register has priority control over the port pin. RCCLK/PTA6/KBI6 is the OSC2 pin when OSCSEL=1 (XTAL option).

11.2 Port A

Port A is an 8-bit special function port that shares all of its pins with the keyboard interrupt (KBI) module (see Chapter 13 Keyboard Interrupt Module (KBI)). Each port A pin also has software configurable pull-up device if the corresponding port pin is configured as input port. PTA0–PTA5 and PTA7 has direct LED drive capability.

NOTE

PTA0–PTA5 pins are available on 28-pin and 32-pin packages only. PTA7 pin is available on 32-pin packages only.

11.2.1 Port A Data Register (PTA)

The port A data register (PTA) contains a data latch for each of the eight port A pins.

Address:	$0000							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
Reset:	Unaffected by Reset							
Additional Functions:	LED (Sink)		LED (Sink)	LED (Sink)	LED (Sink)	LED (Sink)	LED (Sink)	LED (Sink)
	pull-up	pull-up	pull-up	pull-up	pull-up	pull-up	pull-up	pull-up
Alternative Functions:	Keyboard Interrupt	Keyboard Interrupt	Keyboard Interrupt	Keyboard Interrupt	Keyboard Interrupt	Keyboard Interrupt	Keyboard Interrupt	Keyboard Interrupt

Figure 11-2. Port A Data Register (PTA)

PTA[7:0] — Port A Data Bits

These read/write bits are software programmable. Data direction of each port A pin is under the control of the corresponding bit in data direction register A. Reset has no effect on port A data.

KBI7–KBI0 — Port A Keyboard Interrupts

The keyboard interrupt enable bits, KBIE[7:0], in the keyboard interrupt control register (KBIER) enable the port A pins as external interrupt pins, (see Chapter 13 Keyboard Interrupt Module (KBI)).

11.2.2 Data Direction Register A (DDRA)

Data direction register A determines whether each port A pin is an input or an output. Writing a logic 1 to a DDRA bit enables the output buffer for the corresponding port A pin; a logic 0 disables the output buffer.

NOTE

For those devices packaged in a 28-pin package, PTA7 is not connected. DDRA7 should be set to a 1 to configure PTA7 as an output.

For those devices packaged in a 20-pin package, PTA0–PTA5 and PTA7 are not connected. DDRA0–DDRA5 and DDRA7 should be set to a 1 to configure PTA0–PTA5 and PTA7 as outputs.

Address:	$0004							
	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 11-3. Data Direction Register A (DDRA)

DDRA[7:0] — Data Direction Register A Bits

These read/write bits control port A data direction. Reset clears DDRA[7:0], configuring all port A pins as inputs.

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

NOTE

Avoid glitches on port A pins by writing to the port A data register before changing data direction register A bits from 0 to 1.

Figure 11-4 shows the port A I/O logic.

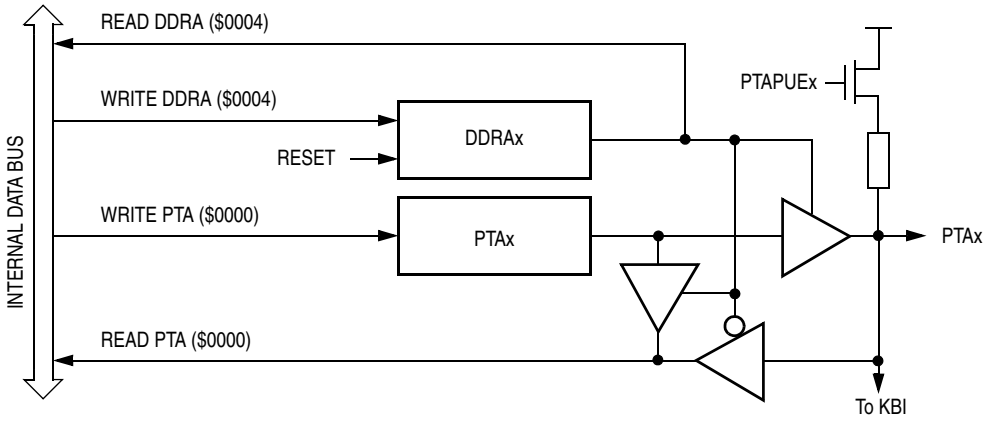


Figure 11-4. Port A I/O Circuit

When DDRAx is a logic 1, reading address $0000 reads the PTAx data latch. When DDRAx is a logic 0, reading address $0000 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit.

Table 11-2 summarizes the operation of the port A pins.

Table 11-2. Port A Pin Functions

PTAPUE Bit	DDRA Bit	PTA Bit	I/O Pin Mode	Accesses to DDRA	Accesses to PTA	
				Read/Write	Read	Write
1	0	X(1)	Input, V_{DD}(2)	DDRA[7:0]	Pin	PTA[7:0](3)
0	0	X	Input, Hi-Z(4)	DDRA[7:0]	Pin	PTA[7:0](3)
X	1	X	Output	DDRA[7:0]	PTA[7:0]	PTA[7:0]

1. X = Don't care.
2. Pin pulled to V_{DD} by internal pull-up.
3. Writing affects data register, but does not affect input.
4. Hi-Z = High impedance.

11.2.3 Port A Input Pull-Up Enable Registers

The port A input pull-up enable registers contain a software configurable pull-up device for each of the eight port A pins. Each bit is individually configurable and requires the corresponding data direction register, DDRAx be configured as input. Each pull-up device is automatically disabled when its corresponding DDRAx bit is configured as output.

Address: $000D

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTA6EN	PTAPUE6	PTAPUE5	PTAPUE4	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
Reset:	0	0	0	0	0	0	0	0

Figure 11-5. Port A Input Pull-up Enable Register (PTAPUE)

Address: $000E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTAPUE7							
Reset:	0	0	0	0	0	0	0	0

Figure 11-6. PTA7 Input Pull-up Enable Register (PTA7PUE)

PTA6EN — Enable PTA6 on OSC2

This read/write bit configures the OSC2 pin function when RC oscillator option is selected. This bit has no effect for XTAL oscillator option.

1 = OSC2 pin configured for PTA6 I/O, and has all the interrupt and pull-up functions
0 = OSC2 pin outputs the RC oscillator clock (RCCLK)

PTAPUE[7:0] — Port A Input Pull-up Enable Bits

These read/write bits are software programmable to enable pull-up devices on port A pins.

1 = Corresponding port A pin configured to have internal pull-up if its DDRA bit is set to 0
0 = Pull-up device is disconnected on the corresponding port A pin regardless of the state of its DDRA bit

11.3 Port B

Port B is an 8-bit special function port that shares all of its port pins with the analog-to-digital converter (ADC) module, see Chapter 10

11.3.1 Port B Data Register (PTB)

The port B data register contains a data latch for each of the eight port B pins.

Address:	$0001							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
Reset:	Unaffected by reset							
Alternative Functions:	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC2	ADC0

Figure 11-7. Port B Data Register (PTB)

PTB[7:0] — Port B Data Bits

These read/write bits are software programmable. Data direction of each port B pin is under the control of the corresponding bit in data direction register B. Reset has no effect on port B data.

ADC7–ADC0 — ADC channels 7 to 0

ADC7–ADC0 are pins used for the input channels to the analog-to-digital converter module. The channel select bits, ADCH[4:0], in the ADC status and control register define which port pin will be used as an ADC input and overrides any control from the port I/O logic. See Chapter 10 Analog-to-Digital Converter (ADC).

11.3.2 Data Direction Register B (DDRB)

Data direction register B determines whether each port B pin is an input or an output. Writing a logic 1 to a DDRB bit enables the output buffer for the corresponding port B pin; a logic 0 disables the output buffer.

Address:	$0005							
	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 11-8. Data Direction Register B (DDRB)

DDRB[7:0] — Data Direction Register B Bits

These read/write bits control port B data direction. Reset clears DDRB[7:0], configuring all port B pins as inputs.

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

NOTE

Avoid glitches on port B pins by writing to the port B data register before changing data direction register B bits from 0 to 1. Figure 11-9 shows the port B I/O logic.

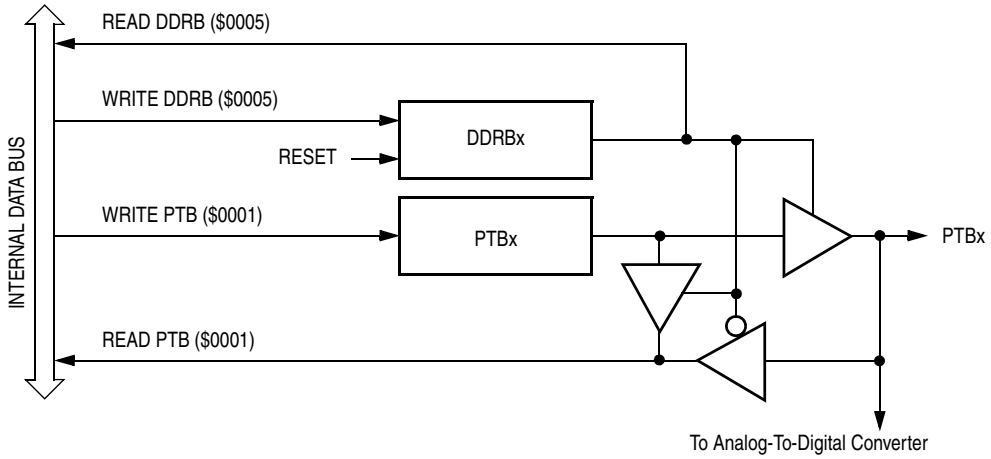


Figure 11-9. Port B I/O Circuit

When DDRBx is a logic 1, reading address $0001 reads the PTBx data latch. When DDRBx is a logic 0, reading address $0001 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 11-3 summarizes the operation of the port B pins.

Table 11-3. Port B Pin Functions

DDRB Bit	PTB Bit	I/O Pin Mode	Accesses to DDRB	Accesses to PTB	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRB[7:0]	Pin	PTB[7:0](3)
1	X	Output	DDRB[7:0]	PTB[7:0]	PTB[7:0]

1. X = don't care.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect the input.

11.4 Port D

Port D is an 8-bit special function port that shares two of its pins with the serial communications interface module (see Chapter 9), two of its pins with the timer 1 interface module, (see Chapter 8), and four of its pins with the analog-to-digital converter module (see Chapter 10). PTD6 and PTD7 each has high current sink (25mA) and programmable pull-up. PTD2, PTD3, PTD6 and PTD7 each has LED sink capability.

NOTE

PTD0–PTD1 are available on 28-pin and 32-pin packages only.

11.4.1 Port D Data Register (PTD)

The port D data register contains a data latch for each of the eight port D pins.

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
Reset:	Unaffected by reset							
Additional Functions	LED (Sink)	LED (Sink)			LED (Sink)	LED (Sink)		
	25mA sink (Slow Edge)	25mA sink (Slow Edge)						
	pull-up	pull-up						
Alternative Functions:	RxD	TxD	T1CH1	T1CH0	ADC8	ADC9	ADC10	ADC11

Figure 11-10. Port D Data Register (PTD)

PTD[7:0] — Port D Data Bits

These read/write bits are software programmable. Data direction of each port D pin is under the control of the corresponding bit in data direction register D. Reset has no effect on port D data.

ADC11–ADC8 — ADC channels 11 to 8

ADC[11:8] are pins used for the input channels to the analog-to-digital converter module. The channel select bits, ADCH[4:0], in the ADC status and control register define which port pin will be used as an ADC input and overrides any control from the port I/O logic. See Chapter 10 Analog-to-Digital Converter (ADC).

T1CH1, T1CH0 — Timer 1 Channel I/Os

The T1CH1 and T1CH0 pins are the TIM1 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTD4/T1CH0 and PTD5/T1CH1 pins are timer channel I/O pins or general-purpose I/O pins. See Chapter 8 Timer Interface Module (TIM).

TxD, RxD — SCI Data I/O Pins

The TxD and RxD pins are the transmit data output and receive data input for the SCI module. The enable SCI bit, ENSCI, in the SCI control register 1 enables the PTD6/TxD and PTD7/RxD pins as SCI TxD and RxD pins and overrides any control from the port I/O logic. See Chapter 9 Serial Communications Interface (SCI).

11.4.2 Data Direction Register D (DDRD)

Data direction register D determines whether each port D pin is an input or an output. Writing a logic 1 to a DDRD bit enables the output buffer for the corresponding port D pin; a logic 0 disables the output buffer.

NOTE

For those devices packaged in a 20-pin package, PTD0–PTD1 and are not connected. DDRD0–DDRD1 should be set to a 1 to configure PTD0–PTD1 as outputs.

Address:	$0007							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
Reset:	0	0	0	0	0	0	0	0

Figure 11-11. Data Direction Register D (DDRD)

DDRD[7:0] — Data Direction Register D Bits

These read/write bits control port D data direction. Reset clears DDRD[7:0], configuring all port D pins as inputs.

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

NOTE

Avoid glitches on port D pins by writing to the port D data register before changing data direction register D bits from 0 to 1. Figure 11-12 shows the port D I/O logic.

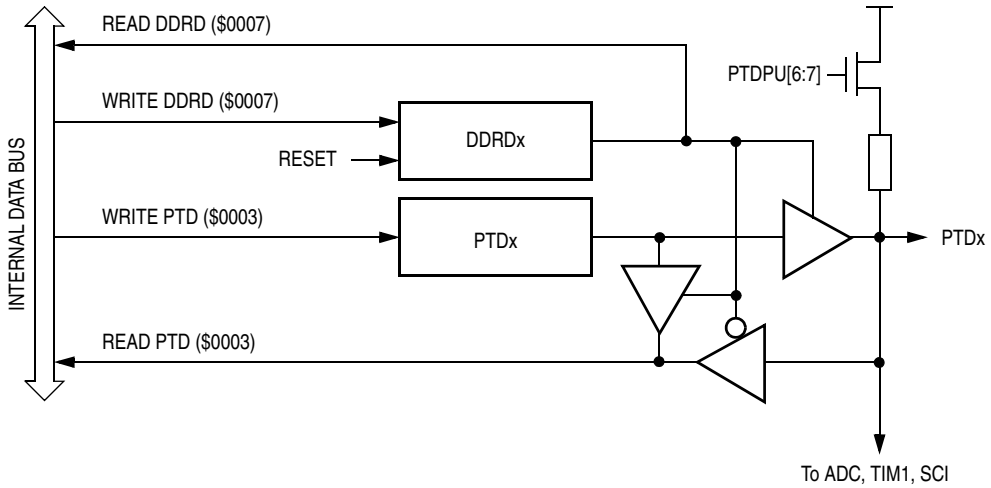


Figure 11-12. Port D I/O Circuit

When DDRDx is a logic 1, reading address $0003 reads the PTDx data latch. When DDRDx is a logic 0, reading address $0003 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 11-4 summarizes the operation of the port D pins.

Table 11-4. Port D Pin Functions

DDRD Bit	PTD Bit	I/O Pin Mode	Accesses to DDRD	Accesses to PTD	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRD[7:0]	Pin	PTD[7:0](3)
1	X	Output	DDRD[7:0]	PTD[7:0]	PTD[7:0]

1. X = don't care.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect the input.

11.4.3 Port D Control Register (PDCR)

The port D control register enables/disables the pull-up resistor and slow-edge high current capability of pins PTD6 and PTD7.

Address:	$000A							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	SLOWD7	SLOWD6	PTDPU7	PTDPU6
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 11-13. Port D Control Register (PDCR)

SLOWDx — Slow Edge Enable

The SLOWD6 and SLOWD7 bits enable the slow-edge, open-drain, high current output (25mA sink) of port pins PTD6 and PTD7 respectively. DDRDx bit is not affected by SLOWDx.

1 = Slow edge enabled; pin is open-drain output
0 = Slow edge disabled; pin is push-pull (standard I/O)

PTDPUx — Port D Pull-up Enable Bits

The PTDPU6 and PTDPU7 bits enable the pull-up device on PTD6 and PTD7 respectively, regardless the status of DDRDx bit.

1 = Enable pull-up device
0 = Disable pull-up device

11.5 Port E

Port E is a 2-bit special function port that shares its pins with the timer 2 interface module (see Chapter 8).

NOTE

PTE0–PTE1 are available on 32-pin packages only.

11.5.1 Port E Data Register (PTE)

The port E data register contains a data latch for each of the two port E pins.

Address:	$0008							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:							PTE1	PTE0
Write:								
Reset:	Unaffected by reset							
Alternative Functions:							T2CH1	T2CH0

Figure 11-14. Port E Data Register (PTE)

PTE[1:0] — Port E Data Bits

These read/write bits are software programmable. Data direction of each port E pin is under the control of the corresponding bit in data direction register E. Reset has no effect on port D data.

T2CH1, T2CH0 — Timer 2 Channel I/Os

The T2CH1 and T2CH0 pins are the TIM2 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTE0/T2CH0 and PTE1/T2CH1 pins are timer channel I/O pins or general-purpose I/O pins. See Chapter 8 Timer Interface Module (TIM).

11.5.2 Data Direction Register E (DDRE)

Data direction register E determines whether each port E pin is an input or an output. Writing a logic 1 to a DDRE bit enables the output buffer for the corresponding port E pin; a logic 0 disables the output buffer.

NOTE

For those devices packaged in a 20-pin package and 28-pin package, PTE0–PTE1 are not connected. DDRE0–DDRE1 should be set to a 1 to configure PTE0–PTE1 as outputs.

Address:	$000C							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:							DDRE1	DDRE0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 11-15. Data Direction Register E (DDRE)

DDRE[1:0] — Data Direction Register E Bits

These read/write bits control port E data direction. Reset clears DDRE[1:0], configuring all port E pins as inputs.

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

NOTE

Avoid glitches on port E pins by writing to the port E data register before changing data direction register E bits from 0 to 1. Figure 11-16 shows the port E I/O logic.

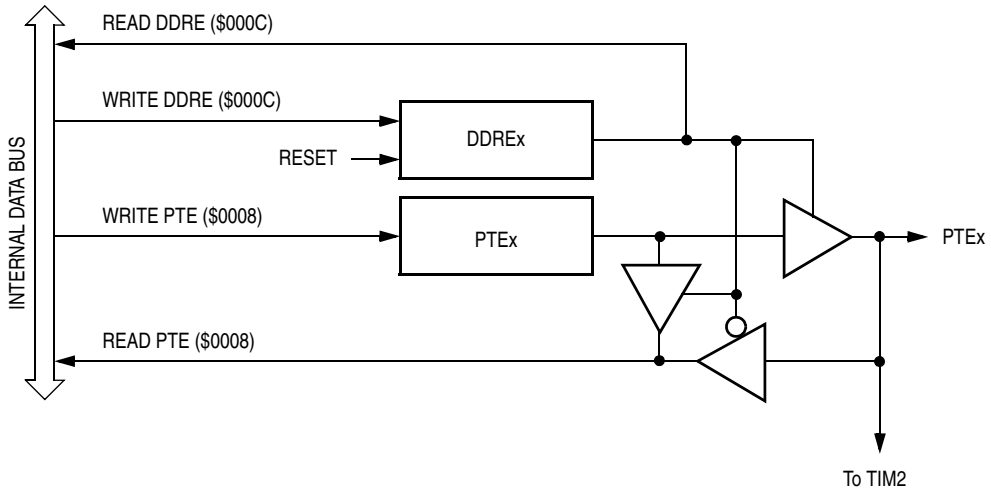


Figure 11-16. Port E I/O Circuit

When DDREx is a logic 1, reading address $0008 reads the PTEx data latch. When DDREx is a logic 0, reading address $0008 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 11-5 summarizes the operation of the port E pins.

Table 11-5. Port E Pin Functions

DDRE Bit	PTE Bit	I/O Pin Mode	Accesses to DDRE	Accesses to PTE	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRE[1:0]	Pin	PTE[1:0](3)
1	X	Output	DDRE[1:0]	PTE[1:0]	PTE[1:0]

1. X = don't care.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect the input.

Chapter 12
External Interrupt (IRQ)

12.1 Introduction

The external interrupt (IRQ) module provides a maskable interrupt input.

12.2 Features

Features of the IRQ module include the following:

- A dedicated external interrupt pin ($\overline{\text{IRQ}}$)
- IRQ interrupt control bits
- Hysteresis buffer
- Programmable edge-only or edge and level interrupt sensitivity
- Automatic interrupt acknowledge
- Selectable internal pullup resistor

12.3 Functional Description

A logic zero applied to the external interrupt pin can latch a CPU interrupt request. Figure 12-1 shows the structure of the IRQ module.

Interrupt signals on the $\overline{\text{IRQ}}$ pin are latched into the IRQ latch. An interrupt latch remains set until one of the following actions occurs:

- Vector fetch — A vector fetch automatically generates an interrupt acknowledge signal that clears the IRQ latch.
- Software clear — Software can clear the interrupt latch by writing to the acknowledge bit in the interrupt status and control register (INTSCR). Writing a logic one to the ACK bit clears the IRQ latch.
- Reset — A reset automatically clears the interrupt latch.

The external interrupt pin is falling-edge-triggered and is software-configurable to be either falling-edge or falling-edge and low-level-triggered. The MODE bit in the INTSCR controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin.

When the interrupt pin is edge-triggered only, the CPU interrupt request remains set until a vector fetch, software clear, or reset occurs.

When the interrupt pin is both falling-edge and low-level-triggered, the CPU interrupt request remains set until both of the following occur:

- Vector fetch or software clear
- Return of the interrupt pin to logic one

The vector fetch or software clear may occur before or after the interrupt pin returns to logic one. As long as the pin is low, the interrupt request remains pending. A reset will clear the latch and the MODE control bit, thereby clearing the interrupt even if the pin stays low.

When set, the IMASK bit in the INTSCR mask all external interrupt requests. A latched interrupt request is not presented to the interrupt priority logic unless the IMASK bit is clear.

NOTE

The interrupt mask (I) in the condition code register (CCR) masks all interrupt requests, including external interrupt requests. (See 5.5 Exception Control.)

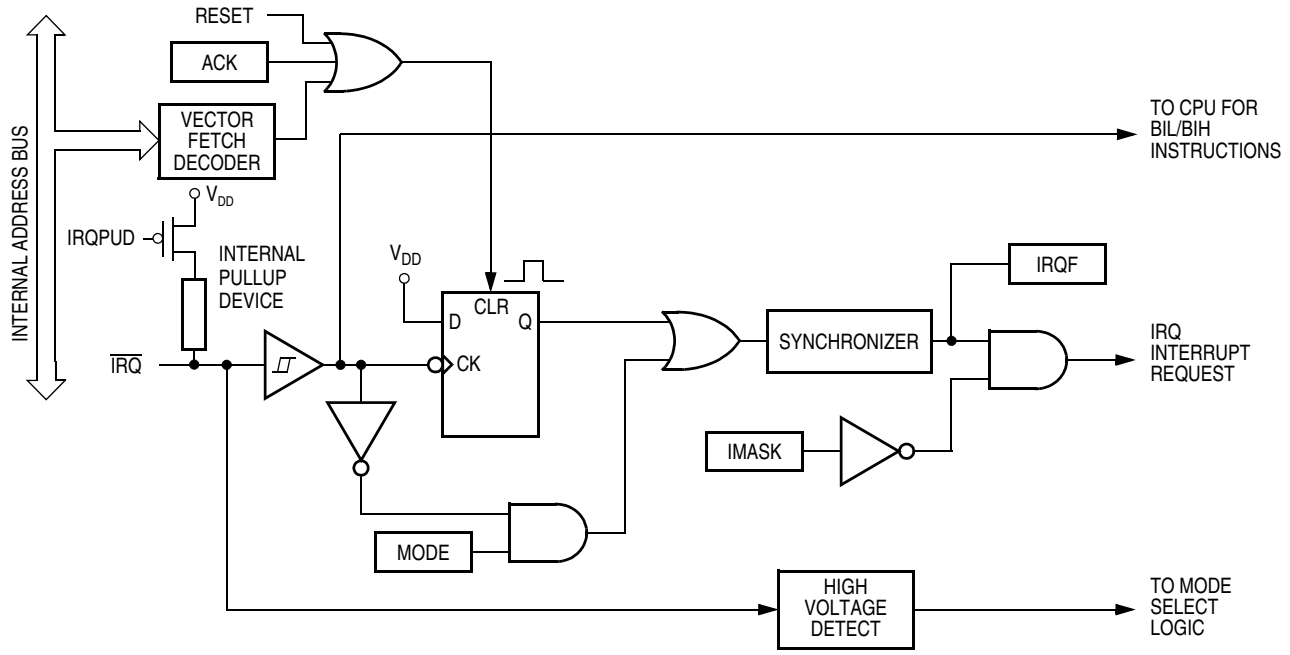


Figure 12-1. IRQ Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001D	IRQ Status and Control Register (INTSCR)	Read:	0	0	0	0	IRQF	0	IMASK	MODE
		Write:						ACK		
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-2. IRQ I/O Register Summary

12.3.1 $\overline{\text{IRQ}}$ Pin

A logic zero on the $\overline{\text{IRQ}}$ pin can latch an interrupt request into the IRQ latch. A vector fetch, software clear, or reset clears the IRQ latch.

If the MODE bit is set, the $\overline{\text{IRQ}}$ pin is both falling-edge-sensitive and low-level-sensitive. With MODE set, both of the following actions must occur to clear IRQ:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the latch. Software may generate the interrupt acknowledge signal by writing a logic one to the ACK bit in the interrupt status and control register (INTSCR). The ACK bit is useful in applications that poll the $\overline{IRQ}$ pin and require software to clear the IRQ latch. Writing to the ACK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACK does not affect subsequent transitions on the $\overline{IRQ}$ pin. A falling edge that occurs after writing to the ACK bit latches another interrupt request. If the IRQ mask bit, IMASK, is clear, the CPU loads the program counter with the vector address at locations $FFFA and $FFFB.
- Return of the $\overline{IRQ}$ pin to logic one — As long as the $\overline{IRQ}$ pin is at logic zero, IRQ remains active.

The vector fetch or software clear and the return of the $\overline{IRQ}$ pin to logic one may occur in any order. The interrupt request remains pending as long as the $\overline{IRQ}$ pin is at logic zero. A reset will clear the latch and the MODE control bit, thereby clearing the interrupt even if the pin stays low.

If the MODE bit is clear, the $\overline{IRQ}$ pin is falling-edge-sensitive only. With MODE clear, a vector fetch or software clear immediately clears the IRQ latch.

The IRQF bit in the INTSCR register can be used to check for pending interrupts. The IRQF bit is not affected by the IMASK bit, which makes it useful in applications where polling is preferred.

Use the BIH or BIL instruction to read the logic level on the $\overline{IRQ}$ pin.

NOTE

When using the level-sensitive interrupt trigger, avoid false interrupts by masking interrupt requests in the interrupt routine.

NOTE

An internal pull-up resistor to V_{DD} is connected to the $\overline{IRQ}$ pin; this can be disabled by setting the IRQPUD bit in the CONFIG2 register ($001E).

12.4 IRQ Module During Break Interrupts

The system integration module (SIM) controls whether the IRQ latch can be cleared during the break state. The BCFE bit in the break flag control register (BFCR) enables software to clear the latches during the break state. (See Chapter 5 System Integration Module (SIM).)

To allow software to clear the IRQ latch during a break interrupt, write a logic one to the BCFE bit. If a latch is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the latches during the break state, write a logic zero to the BCFE bit. With BCFE at logic zero (its default state), writing to the ACK bit in the IRQ status and control register during the break state has no effect on the IRQ latch.

12.5 IRQ Status and Control Register (INTSCR)

The IRQ status and control register (INTSCR) controls and monitors operation of the IRQ module. The INTSCR has the following functions:

- Shows the state of the IRQ flag
- Clears the IRQ latch
- Masks IRQ and interrupt request
- Controls triggering sensitivity of the $\overline{IRQ}$ interrupt pin

Address: $001D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	IRQF		IMASK	MODE
Write:						ACK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-3. IRQ Status and Control Register (INTSCR)

IRQF — IRQ Flag Bit

This read-only status bit is high when the IRQ interrupt is pending.

1 = $\overline{IRQ}$ interrupt pending
0 = $\overline{IRQ}$ interrupt not pending

ACK — IRQ Interrupt Request Acknowledge Bit

Writing a logic one to this write-only bit clears the IRQ latch. ACK always reads as logic zero. Reset clears ACK.

IMASK — IRQ Interrupt Mask Bit

Writing a logic one to this read/write bit disables IRQ interrupt requests. Reset clears IMASK.

1 = IRQ interrupt requests disabled
0 = IRQ interrupt requests enabled

MODE — IRQ Edge/Level Select Bit

This read/write bit controls the triggering sensitivity of the $\overline{IRQ}$ pin. Reset clears MODE.

1 = $\overline{IRQ}$ interrupt requests on falling edges and low levels
0 = $\overline{IRQ}$ interrupt requests on falling edges only

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	IRQPUD	R	R	LVIT1	LVIT0	R	R	R
Reset:	0	0	0	Not affected	Not affected	0	0	0
POR:	0	0	0	0	0	0	0	0

R = Reserved

Figure 12-4. Configuration Register 2 (CONFIG2)

IRQPUD — $\overline{IRQ}$ Pin Pull-Up Disable Bit

IRQPUD disconnects the internal pull-up on the $\overline{IRQ}$ pin.

1 = Internal pull-up is disconnected
0 = Internal pull-up is connected between $\overline{IRQ}$ pin and V_{DD}

Chapter 13
Keyboard Interrupt Module (KBI)

13.1 Introduction

The keyboard interrupt module (KBI) provides eight independently maskable external interrupts which are accessible via PTA0–PTA7. When a port pin is enabled for keyboard interrupt function, an internal pull-up device is also enabled on the pin.

13.2 Features

Features of the keyboard interrupt module include the following:

- Eight keyboard interrupt pins with pull-up devices
- Separate keyboard interrupt enable bits and one keyboard interrupt mask
- Programmable edge-only or edge- and level- interrupt sensitivity
- Exit from low-power modes

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001A	Keyboard Status and Control Register (KBSCR)	Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
		Write:						ACKK		
		Reset:	0	0	0	0	0	0	0	0
$001B	Keyboard Interrupt Enable Register (KBIER)	Read:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 13-1. KBI I/O Register Summary

13.3 I/O Pins

The eight keyboard interrupt pins are shared with standard port I/O pins. The full name of the KBI pins are listed in Table 13-1. The generic pin name appear in the text that follows.

Table 13-1. Pin Name Conventions

KBI Generic Pin Name	Full MCU Pin Name	Pin Selected for KBI Function by KBIEx Bit in KBIER
KBI0–KBI5	PTA0/KBI0–PTA5/KBI5	KBIE0–KBIE5
KBI6	OSC2/RCCLK/PTA6/KBI6(1)	KBIE6
KBI7	PTA7/KBI7	KBIE7

1. PTA6/KBI6 is only available when OSCSEL=0 at $FFD0 (RC option), and PTA6EN=1 at $000D.

13.4 Functional Description

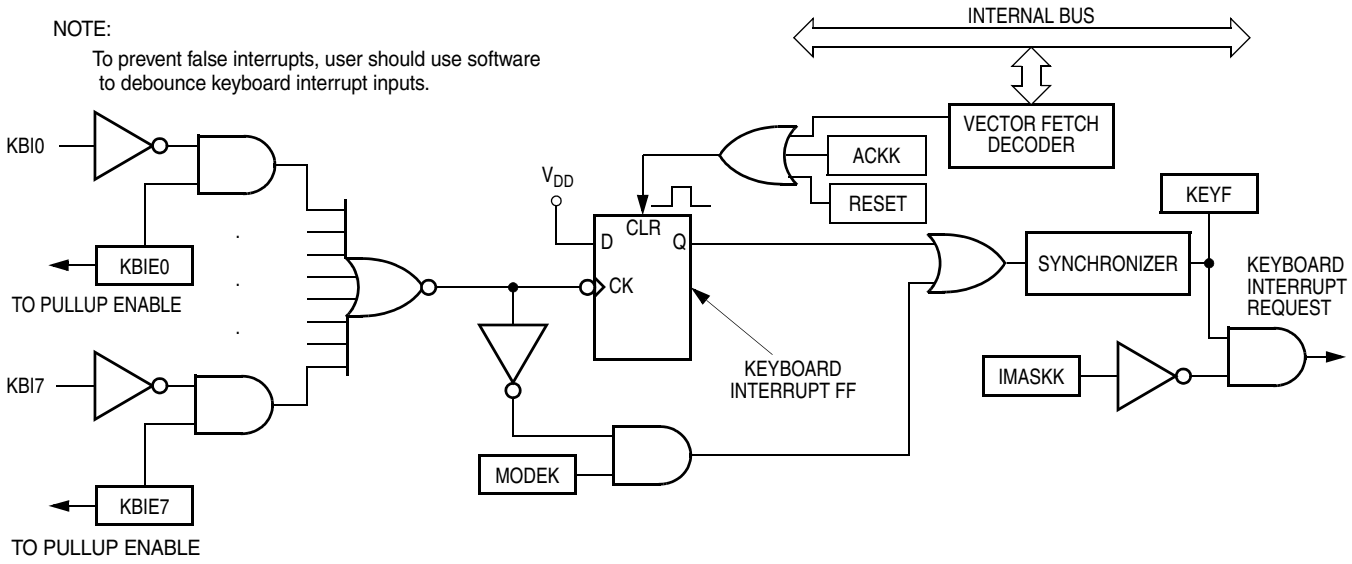


Figure 13-2. Keyboard Interrupt Block Diagram

Writing to the KBIE7–KBIE0 bits in the keyboard interrupt enable register independently enables or disables each port A pin as a keyboard interrupt pin. Enabling a keyboard interrupt pin in port A also enables its internal pull-up device regardless of PTAPUEx bits in the port A input pull-up enable register (see 11.2.3 Port A Input Pull-Up Enable Registers). A logic 0 applied to an enabled keyboard interrupt pin latches a keyboard interrupt request.

A keyboard interrupt is latched when one or more keyboard pins goes low after all were high. The MODEK bit in the keyboard status and control register controls the triggering mode of the keyboard interrupt.

- If the keyboard interrupt is edge-sensitive only, a falling edge on a keyboard pin does not latch an interrupt request if another keyboard pin is already low. To prevent losing an interrupt request on one pin because another pin is still low, software can disable the latter pin while it is low.
- If the keyboard interrupt is falling edge- and low level-sensitive, an interrupt request is present as long as any keyboard pin is low.

If the MODEK bit is set, the keyboard interrupt pins are both falling edge- and low level-sensitive, and both of the following actions must occur to clear a keyboard interrupt request:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the interrupt request. Software may generate the interrupt acknowledge signal by writing a logic 1 to the ACKK bit in the keyboard status and control register KBSCR. The ACKK bit is useful in applications that poll the keyboard interrupt pins and require software to clear the keyboard interrupt request. Writing to the ACKK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACKK does not affect subsequent transitions on the keyboard interrupt pins. A falling edge that occurs after writing to the ACKK bit latches another interrupt request. If the keyboard interrupt mask bit, IMASKK, is clear, the CPU loads the program counter with the vector address at locations $FFE0 and $FFE1.
- Return of all enabled keyboard interrupt pins to logic 1 — As long as any enabled keyboard interrupt pin is at logic 0, the keyboard interrupt remains set.

The vector fetch or software clear and the return of all enabled keyboard interrupt pins to logic 1 may occur in any order.

If the MODEK bit is clear, the keyboard interrupt pin is falling-edge-sensitive only. With MODEK clear, a vector fetch or software clear immediately clears the keyboard interrupt request.

Reset clears the keyboard interrupt request and the MODEK bit, clearing the interrupt request even if a keyboard interrupt pin stays at logic 0.

The keyboard flag bit (KEYF) in the keyboard status and control register can be used to see if a pending interrupt exists. The KEYF bit is not affected by the keyboard interrupt mask bit (IMASKK) which makes it useful in applications where polling is preferred.

To determine the logic level on a keyboard interrupt pin, disable the pull-up device, use the data direction register to configure the pin as an input and then read the data register.

NOTE

Setting a keyboard interrupt enable bit (KBIEx) forces the corresponding keyboard interrupt pin to be an input, overriding the data direction register. However, the data direction register bit must be a logic 0 for software to read the pin.

13.4.1 Keyboard Initialization

When a keyboard interrupt pin is enabled, it takes time for the internal pull-up to reach a logic 1. Therefore a false interrupt can occur as soon as the pin is enabled.

To prevent a false interrupt on keyboard initialization:

1. Mask keyboard interrupts by setting the IMASKK bit in the keyboard status and control register.
2. Enable the KBI pins by setting the appropriate KBIEx bits in the keyboard interrupt enable register.
3. Write to the ACKK bit in the keyboard status and control register to clear any false interrupts.
4. Clear the IMASKK bit.

An interrupt signal on an edge-triggered pin can be acknowledged immediately after enabling the pin. An interrupt signal on an edge- and level-triggered interrupt pin must be acknowledged after a delay that depends on the external load.

Another way to avoid a false interrupt:

1. Configure the keyboard pins as outputs by setting the appropriate DDRA bits in the data direction register A.
2. Write logic 1’s to the appropriate port A data register bits.
3. Enable the KBI pins by setting the appropriate KBIEx bits in the keyboard interrupt enable register.

13.5 Keyboard Interrupt Registers

Two registers control the operation of the keyboard interrupt module:

- Keyboard status and control register
- Keyboard interrupt enable register

13.5.1 Keyboard Status and Control Register

- Flags keyboard interrupt requests
- Acknowledges keyboard interrupt requests
- Masks keyboard interrupt requests
- Controls keyboard interrupt triggering sensitivity

Address: $001A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
Write:						ACKK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 13-3. Keyboard Status and Control Register (KBSCR)

KEYF — Keyboard Flag Bit

This read-only bit is set when a keyboard interrupt is pending on port A. Reset clears the KEYF bit.

1 = Keyboard interrupt pending
0 = No keyboard interrupt pending

ACKK — Keyboard Acknowledge Bit

Writing a logic 1 to this write-only bit clears the keyboard interrupt request on port A. ACKK always reads as logic 0. Reset clears ACKK.

IMASKK— Keyboard Interrupt Mask Bit

Writing a logic 1 to this read/write bit prevents the output of the keyboard interrupt mask from generating interrupt requests on port A. Reset clears the IMASKK bit.

1 = Keyboard interrupt requests masked
0 = Keyboard interrupt requests not masked

MODEK — Keyboard Triggering Sensitivity Bit

This read/write bit controls the triggering sensitivity of the keyboard interrupt pins on port A. Reset clears MODEK.

1 = Keyboard interrupt requests on falling edges and low levels
0 = Keyboard interrupt requests on falling edges only

13.5.2 Keyboard Interrupt Enable Register

The port-A keyboard interrupt enable register enables or disables each port-A pin to operate as a keyboard interrupt pin

Address: $001B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 13-4. Keyboard Interrupt Enable Register (KBIER)

KBIE7–KBIE0 — Port-A Keyboard Interrupt Enable Bits

Each of these read/write bits enables the corresponding keyboard interrupt pin on port-A to latch interrupt requests. Reset clears the keyboard interrupt enable register.

1 = KBIx pin enabled as keyboard interrupt pin
0 = KBIx pin not enabled as keyboard interrupt pin

13.6 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

13.6.1 Wait Mode

The keyboard modules remain active in wait mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of wait mode.

13.6.2 Stop Mode

The keyboard module remains active in stop mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of stop mode.

13.7 Keyboard Module During Break Interrupts

The system integration module (SIM) controls whether the keyboard interrupt latch can be cleared during the break state. The BCFE bit in the break flag control register (BFCR) enables software to clear status bits during the break state.

To allow software to clear the keyboard interrupt latch during a break interrupt, write a logic 1 to the BCFE bit. If a latch is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the latch during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), writing to the keyboard acknowledge bit (ACKK) in the keyboard status and control register during the break state has no effect.

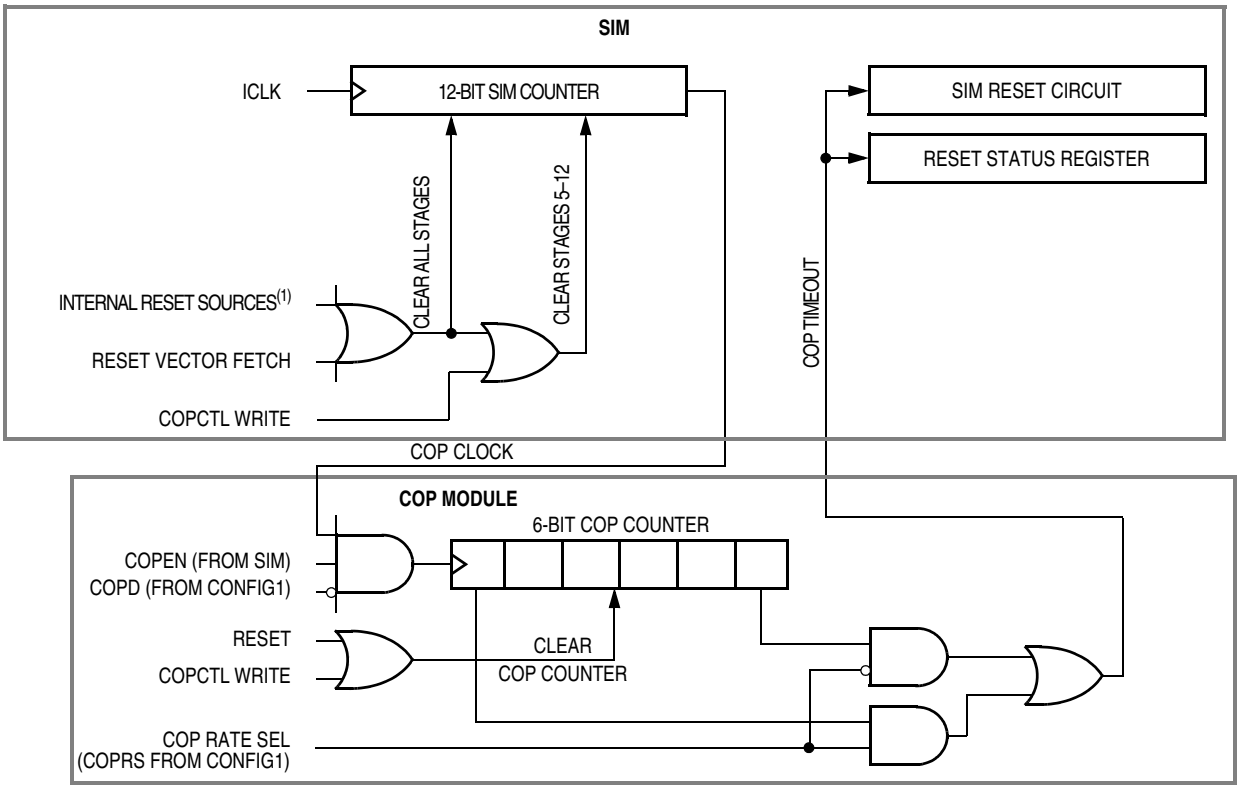
Chapter 14
Computer Operating Properly (COP)

14.1 Introduction

The computer operating properly (COP) module contains a free-running counter that generates a reset if allowed to overflow. The COP module helps software recover from runaway code. Prevent a COP reset by clearing the COP counter periodically. The COP module can be disabled through the COPD bit in the CONFIG1 register.

14.2 Functional Description

Figure 14-1 shows the structure of the COP module.



NOTE: See SIM section for more details.

Figure 14-1. COP Block Diagram

The COP counter is a free-running 6-bit counter preceded by the 12-bit system integration module (SIM) counter. If not cleared by software, the COP counter overflows and generates an asynchronous reset after $2^{18} - 2^4$ or $2^{13} - 2^4$ ICLK cycles; depending on the state of the COP rate select bit, COPRS, in configuration register 1. Writing any value to location $FFFF before an overflow occurs prevents a COP reset by clearing the COP counter and stages 12 through 5 of the SIM counter.

NOTE

Service the COP immediately after reset and before entering or after exiting stop mode to guarantee the maximum time before the first COP counter overflow.

A COP reset pulls the $\overline{\text{RST}}$ pin low for 32 × ICLK cycles and sets the COP bit in the reset status register (RSR). (See 5.7.2 Reset Status Register (RSR).).

NOTE

Place COP clearing instructions in the main program and not in an interrupt subroutine. Such an interrupt subroutine could keep the COP from generating a reset even while the main program is not working properly.

14.3 I/O Signals

The following paragraphs describe the signals shown in Figure 14-1.

14.3.1 ICLK

ICLK is the internal oscillator output signal, typically 50-kHz. The ICLK frequency varies depending on the supply voltage. See **Chapter 17 Electrical Specifications** for ICLK parameters.

14.3.2 COPCTL Write

Writing any value to the COP control register (COPCTL) (see **14.4 COP Control Register**) clears the COP counter and clears bits 12 through 5 of the SIM counter. Reading the COP control register returns the low byte of the reset vector.

14.3.3 Power-On Reset

The power-on reset (POR) circuit in the SIM clears the SIM counter 4096 × ICLK cycles after power-up.

14.3.4 Internal Reset

An internal reset clears the SIM counter and the COP counter.

14.3.5 Reset Vector Fetch

A reset vector fetch occurs when the vector address appears on the data bus. A reset vector fetch clears the SIM counter.

14.3.6 COPD (COP Disable)

The COPD signal reflects the state of the COP disable bit (COPD) in the configuration register 1 (CONFIG1). (See Chapter 3 Configuration and Mask Option Registers (CONFIG & MOR).)

14.3.7 COPRS (COP Rate Select)

The COPRS signal reflects the state of the COP rate select bit (COPRS) in the configuration register 1.

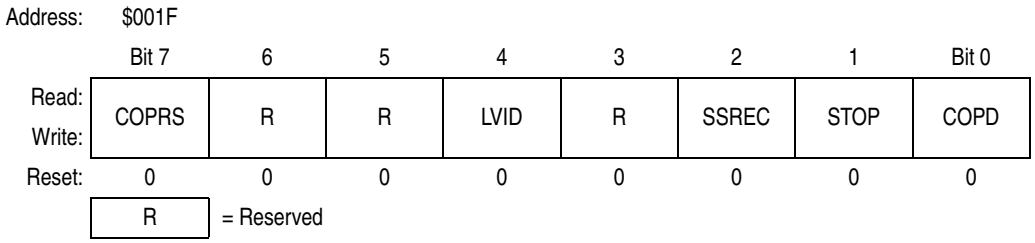


Figure 14-2. Configuration Register 1 (CONFIG1)

COPRS — COP Rate Select Bit

COPRS selects the COP timeout period. Reset clears COPRS.

1 = COP timeout period is ($2^{13} – 2^4$) ICLK cycles
0 = COP timeout period is ($2^{18} – 2^4$) ICLK cycles

COPD — COP Disable Bit

COPD disables the COP module.

1 = COP module disabled
0 = COP module enabled

14.4 COP Control Register

The COP control register is located at address $FFFF and overlaps the reset vector. Writing any value to $FFFF clears the COP counter and starts a new timeout period. Reading location $FFFF returns the low byte of the reset vector.

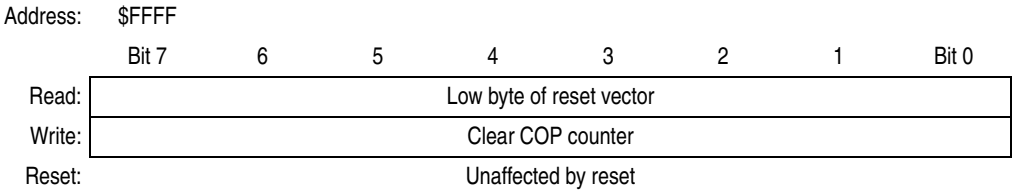


Figure 14-3. COP Control Register (COPCTL)

14.5 Interrupts

The COP does not generate CPU interrupt requests.

14.6 Monitor Mode

The COP is disabled in monitor mode when V_{TST} is present on the $\overline{IRQ}$ pin or on the $\overline{RST}$ pin.

14.7 Low-Power Modes

The WAIT and STOP instructions put the MCU in low-power consumption standby modes.

14.7.1 Wait Mode

The COP continues to operate during wait mode. To prevent a COP reset during wait mode, periodically clear the COP counter in a CPU interrupt routine.

14.7.2 Stop Mode

Stop mode turns off the ICLK input to the COP and clears the COP prescaler. Service the COP immediately before entering or after exiting stop mode to ensure a full COP timeout period after entering or exiting stop mode.

To prevent inadvertently turning off the COP with a STOP instruction, a configuration option is available that disables the STOP instruction. When the STOP bit in the configuration register has the STOP instruction is disabled, execution of a STOP instruction results in an illegal opcode reset.

14.8 COP Module During Break Mode

The COP is disabled during a break interrupt when V_{TST} is present on the $\overline{RST}$ pin.

Chapter 15
Low Voltage Inhibit (LVI)

15.1 Introduction

This section describes the low-voltage inhibit module (LVI), which monitors the voltage on the V_{DD} pin and generates a reset when the V_{DD} voltage falls to the LVI trip (LVI_{TRIP}) voltage.

15.2 Features

Features of the LVI module include the following:

- Selectable LVI trip voltage
- Selectable LVI circuit disable

15.3 Functional Description

Figure 15-1 shows the structure of the LVI module. The LVI is enabled after a reset. The LVI module contains a bandgap reference circuit and comparator. Setting LVI disable bit (LVID) disables the LVI to monitor V_{DD} voltage. The LVI trip voltage selection bits (LVIT1, LVIT0) determine at which V_{DD} level the LVI module should take actions.

The LVI module generates one output signal:

LVI Reset — an reset signal will be generated to reset the CPU when V_{DD} drops to below the set trip point.

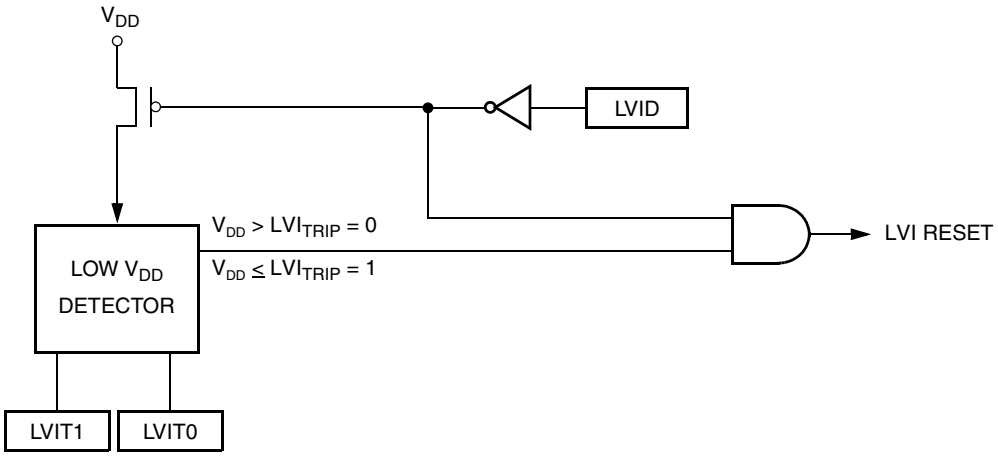


Figure 15-1. LVI Module Block Diagram

15.4 LVI Control Register (CONFIG2/CONFIG1)

The LVI module is controlled by three bits in the configuration registers, CONFIG1 and CONFIG2.

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	IRQPUD	R	R	LVIT1	LVIT0	R	R	STOP_ ICLKDIS
Reset:	0	0	0	Cleared by POR only		0	0	0

Figure 15-2. Configuration Register 2 (CONFIG2)

Address: $001F

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	COPRS	R	R	LVID	R	SSREC	STOP	COPD
Reset:	0	0	0	0	0	0	0	0

Figure 15-3. Configuration Register 1 (CONFIG1)

LVID — Low Voltage Inhibit Disable Bit

LVID disables the LVI module. Reset clears LVID.

1 = Low voltage inhibit disabled
0 = Low voltage inhibit enabled

LVIT1, LVIT0 — LVI Trip Voltage Selection Bits

These two bits determine at which level of V_{DD} the LVI module will come into action. LVIT1 and LVIT0 are cleared by a power-on reset only.

Table 15-1. Trip Voltage Selection

LVIT1	LVIT0	Trip Voltage[1]	Comments
0	0	V_{LVR3} (2.49V)	For V_{DD}=3V operation
0	1	V_{LVR3} (2.49V)	For V_{DD}=3V operation
1	0	V_{LVR5} (4.25V)	For V_{DD}=5V operation
1	1	Reserved	

1. See Chapter 17 Electrical Specifications for full parameters.

15.5 Low-Power Modes

The STOP and WAIT instructions put the MCU in low-power-consumption standby modes.

15.5.1 Wait Mode

The LVI module, when enabled, will continue to operate in wait mode.

15.5.2 Stop Mode

The LVI module, when enabled, will continue to operate in stop mode.

Chapter 16
Break Module (BREAK)

16.1 Introduction

This section describes the break module. The break module can generate a break interrupt that stops normal program flow at a defined address to enter a background program.

16.2 Features

Features of the break module include the following:

- Accessible I/O registers during the break Interrupt
- CPU-generated break interrupts
- Software-generated break interrupts
- COP disabling during break interrupts

16.3 Functional Description

When the internal address bus matches the value written in the break address registers, the break module issues a breakpoint signal ($\overline{\text{BKPT}}$) to the SIM. The SIM then causes the CPU to load the instruction register with a software interrupt instruction (SWI) after completion of the current CPU instruction. The program counter vectors to $FFFC and $FFFD ($FEFC and $FEFD in monitor mode).

The following events can cause a break interrupt to occur:

- A CPU-generated address (the address in the program counter) matches the contents of the break address registers.
- Software writes a logic one to the BRKA bit in the break status and control register.

When a CPU generated address matches the contents of the break address registers, the break interrupt begins after the CPU completes its current instruction. A return from interrupt instruction (RTI) in the break routine ends the break interrupt and returns the MCU to normal operation. Figure 16-1 shows the structure of the break module.

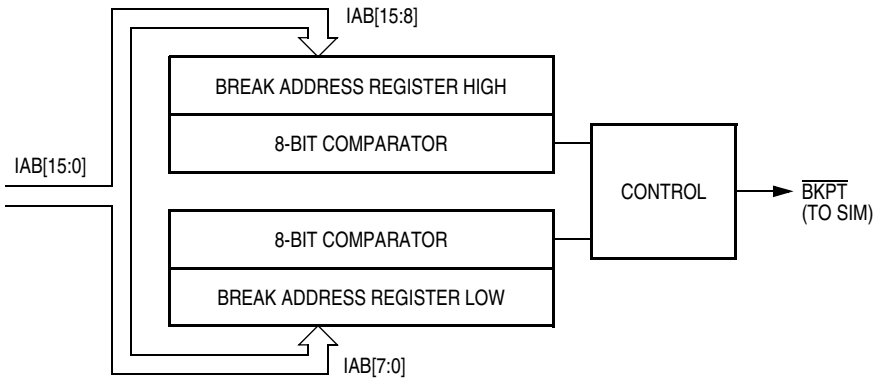


Figure 16-1. Break Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	Break Status Register (BSR)	Read:	R	R	R	R	R	R	SBSW	R
		Write:							See note	
		Reset:							0	
$FE03	Break Flag Control Register (BFCR)	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
$FE0C	Break Address High Register (BRKH)	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0D	Break Address low Register (BRKL)	Read:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0E	Break Status and Control Register (BRKSCR)	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

Note: Writing a logic 0 clears SBSW.

= Unimplemented R = Reserved

Figure 16-2. Break I/O Register Summary

16.3.1 Flag Protection During Break Interrupts

The system integration module (SIM) controls whether or not module status bits can be cleared during the break state. The BCFE bit in the break flag control register (BFCR) enables software to clear status bits during the break state. (See 5.7.3 Break Flag Control Register (BFCR) and see the Break Interrupts subsection for each module.)

16.3.2 CPU During Break Interrupts

The CPU starts a break interrupt by:

- Loading the instruction register with the SWI instruction
- Loading the program counter with $FFFC:$FFFD ($FEFC:$FEFD in monitor mode)

The break interrupt begins after completion of the CPU instruction in progress. If the break address register match occurs on the last cycle of a CPU instruction, the break interrupt begins immediately.

16.3.3 TIM During Break Interrupts

A break interrupt stops the timer counter.

16.3.4 COP During Break Interrupts

The COP is disabled during a break interrupt when V_{TST} is present on the $\overline{RST}$ pin.

16.4 Break Module Registers

These registers control and monitor operation of the break module:

- Break status and control register (BRKSCR)
- Break address register high (BRKH)
- Break address register low (BRKL)
- Break status register (BSR)
- Break flag control register (BFCR)

16.4.1 Break Status and Control Register (BRKSCR)

The break status and control register contains break module enable and status bits.

Address: $FE0E

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

= Unimplemented

Figure 16-3. Break Status and Control Register (BRKSCR)

BRKE — Break Enable Bit

This read/write bit enables breaks on break address register matches. Clear BRKE by writing a logic zero to bit 7. Reset clears the BRKE bit.

1 = Breaks enabled on 16-bit address match
0 = Breaks disabled

BRKA — Break Active Bit

This read/write status and control bit is set when a break address match occurs. Writing a logic one to BRKA generates a break interrupt. Clear BRKA by writing a logic zero to it before exiting the break routine. Reset clears the BRKA bit.

1 = Break address match
0 = No break address match

16.4.2 Break Address Registers

The break address registers contain the high and low bytes of the desired breakpoint address. Reset clears the break address registers.

Address: $FE0C

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

Figure 16-4. Break Address Register High (BRKH)

Address:	$FE0D							
	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 16-5. Break Address Register Low (BRKL)

16.4.3 Break Status Register

The break status register contains a flag to indicate that a break caused an exit from stop or wait mode.

Address:	$FE00							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R	R	R	R	R	R	SBSW	R
Write:							Note(1)	
Reset:							0	

R = Reserved

1. Writing a logic zero clears SBSW.

Figure 16-6. Break Status Register (BSR)

SBSW — SIM Break Stop/Wait

This status bit is useful in applications requiring a return to wait or stop mode after exiting from a break interrupt. Clear SBSW by writing a logic zero to it. Reset clears SBSW.

1 = Stop mode or wait mode was exited by break interrupt
0 = Stop mode or wait mode was not exited by break interrupt

SBSW can be read within the break state SWI routine. The user can modify the return address on the stack by subtracting one from it. The following code is an example of this.

```
; This code works if the H register has been pushed onto the stack in the break
; service routine software. This code should be executed at the end of the
; break service routine software.

  HIBYTE   EQU    5

  LOBYTE   EQU    6

;          If not SBSW, do RTI

           BRCLR  SBSW,BSR, RETURN   ; See if wait mode or stop mode was exited
                                     ; by break.

           TST    LOBYTE,SP          ; If RETURNLO is not zero,

           BNE    DOLO               ; then just decrement low byte.

           DEC    HIBYTE,SP          ; Else deal with high byte, too.

  DOLO     DEC    LOBYTE,SP          ; Point to WAIT/STOP opcode.

  RETURN   PULH                      ; Restore H register.
           RTI
```

16.4.4 Break Flag Control Register (BFCR)

The break control register contains a bit that enables software to clear status bits while the MCU is in a break state.

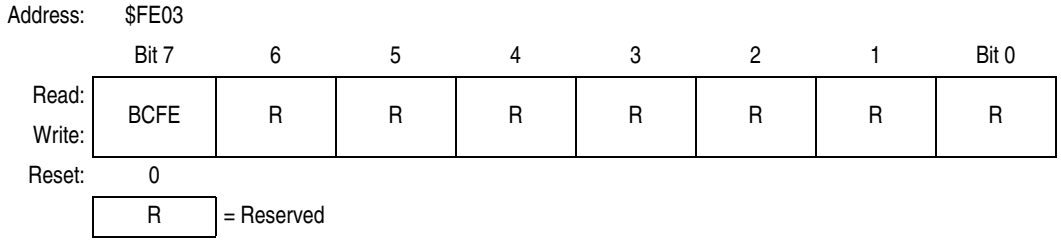


Figure 16-7. Break Flag Control Register (BFCR)

BCFE — Break Clear Flag Enable Bit

This read/write bit enables software to clear status bits by accessing status registers while the MCU is in a break state. To clear status bits during the break state, the BCFE bit must be set.

1 = Status bits clearable during break
0 = Status bits not clearable during break

16.5 Low-Power Modes

The WAIT and STOP instructions put the MCU in low-power-consumption standby modes.

16.5.1 Wait Mode

If enabled, the break module is active in wait mode. In the break routine, the user can subtract one from the return address on the stack if SBSW is set (see 5.6 Low-Power Modes). Clear the SBSW bit by writing logic zero to it.

16.5.2 Stop Mode

A break interrupt causes exit from stop mode and sets the SBSW bit in the break status register. See 5.7 SIM Registers.

Chapter 17
Electrical Specifications

17.1 Introduction

This section contains electrical and timing specifications.

17.2 Absolute 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 Sections 17.5 and 17.8 for guaranteed operating conditions.

Table 17-1. Absolute Maximum Ratings

Characteristic[1]	Symbol	Value	Unit
Supply voltage	V_{DD}	−0.3 to +6.0	V
Input voltage	V_{IN}	V_{SS}−0.3 to V_{DD} +0.3	V
Mode entry voltage, $\overline{IRQ}$ pin	V_{TST}	V_{SS}−0.3 to +8.5	V
Maximum current per pin excluding V_{DD} and V_{SS}	I	±25	mA
Storage temperature	T_{STG}	−55 to +150	°C
Maximum current out of V_{SS}	I_{MVSS}	100	mA
Maximum current into V_{DD}	I_{MVDD}	100	mA

1. Voltages referenced to V_{SS}.

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}.)

17.3 Functional Operating Range

Table 17-2. Operating Range

Characteristic	Symbol	Value		Unit
Operating temperature range	T_A	−40 to +125	−40 to +85	°C
Operating voltage range	V_{DD}	— 5 ±10%	3 ±10% 5 ±10%	V

17.4 Thermal Characteristics

Table 17-3. Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance 20-pin PDIP 20-pin SOIC 28-pin PDIP 28-pin SOIC 32-pin SDIP 32-pin LQFP	θ_{JA}	 70 70 70 70 70 95	°C/W
I/O pin power dissipation	$P_{I/O}$	User determined	W
Power dissipation(1)	P_D	$P_D = (I_{DD} \times V_{DD}) + P_{I/O} = K/(T_J + 273\ °C)$	W
Constant(2)	K	$P_D \times (T_A + 273\ °C) + P_D^2 \times \theta_{JA}$	W/°C
Average junction temperature	T_J	$T_A + (P_D \times \theta_{JA})$	°C

1. Power dissipation is a function of temperature.
2. K constant unique to the device. K can be determined for a known T_A and measured P_D. With this value of K, P_D and T_J can be determined for any value of T_A.

17.5 5V DC Electrical Characteristics

Table 17-4. DC Electrical Characteristics (5V)

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Output high voltage (I_{LOAD} = –2.0mA) PTA0–PTA7, PTB0–PTB7, PTD0–PTD7, PTE0–PTE1	V_{OH}	V_{DD}–0.8	—	—	V
Output low voltage (I_{LOAD} = 1.6mA) PTA6, PTB0–PTB7, PTD0, PTD1, PTD4, PTD5, PTE0–PTE1	V_{OL}	—	—	0.4	V
Output low voltage (I_{LOAD} = 25mA) PTD6, PTD7	V_{OL}	—	—	0.5	V
LED drives (V_{OL} = 3V) PTA0–PTA5, PTA7, PTD2, PTD3, PTD6, PTD7	I_{OL}	10	16	25	mA
Input high voltage PTA0–PTA7, PTB0–PTB7, PTD0–PTD7, PTE0–PTE1, $\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input low voltage PTA0–PTA7, PTB0–PTB7, PTD0–PTD7, PTE0–PTE1, $\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IL}	V_{SS}	—	0.3 × V_{DD}	V
V_{DD} supply current, f_{OP} = 8MHz Run[3] XTAL oscillator option RC oscillator option Wait[4] XTAL oscillator option RC oscillator option Stop[5] (–40°C to 125°C) XTAL oscillator option RC oscillator option	I_{DD}	 — — — — — —	 7.5 11 3 3.5 1.5 0.5	 10 13 5.5 6 8 3	 mA mA mA mA µA µA
Digital I/O ports Hi-Z leakage current	I_{IL}	—	—	± 10	µA
Input current	I_{IN}	—	—	± 1	µA
Capacitance Ports (as input or output)	C_{OUT} C_{IN}	— —	— —	12 8	pF
POR rearm voltage[6]	V_{POR}	0	—	100	mV
POR rise time ramp rate[7]	R_{POR}	0.035	—	—	V/ms
Monitor mode entry voltage	V_{TST}	1.5 × V_{DD}	—	8.5	V
Pullup resistors[8] PTD6, PTD7 $\overline{RST}$, $\overline{IRQ}$, PTA0–PTA7	 R_{PU1} R_{PU2}	 1.8 16	 3.3 26	 4.8 36	 kΩ kΩ
Low-voltage inhibit, trip falling voltage	V_{TRIPF}	3.60	4.25	4.48	V
Low-voltage inhibit, trip rising voltage	V_{TRIPR}	3.75	4.40	4.63	V

1. V_{DD} = 4.5 to 5.5 Vdc, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted.
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. Run (operating) I_{DD} measured using external square wave clock source (f_{OP} = 8MHz). All inputs 0.2V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects run I_{DD}. Measured with all modules enabled.
4. Wait I_{DD} measured using external square wave clock source (f_{OP} = 8MHz). All inputs 0.2V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects wait I_{DD}.
5. Stop I_{DD} measured with OSC1 grounded; no port pins sourcing current. LVI is disabled.
6. Maximum is highest voltage that POR is guaranteed.
7. If minimum V_{DD} is not reached before the internal POR reset is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached.
8. R_{PU1} and R_{PU2} are measured at V_{DD} = 5.0V.

17.6 5V Control Timing

Table 17-5. Control Timing (5V)

Characteristic[1]	Symbol	Min	Max	Unit
Internal operating frequency	f_{OP}	—	8	MHz
$\overline{RST}$ input pulse width low[2]	t_{RL}	750	—	ns
TIM2 external clock input	f_{T2CLK}	—	4	MHz
$\overline{IRQ}$ interrupt pulse width low (edge-triggered)[3]	t_{ILIH}	100	—	ns
$\overline{IRQ}$ interrupt pulse period[3]	t_{ILIL}	Note[4]	—	t_{CYC}

1. V_{DD} = 4.5 to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H; timing shown with respect to 20% V_{DD} and 70% V_{SS}, unless otherwise noted.
2. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.
3. Values are based on characterization results, not tested in production.
4. The minimum period is the number of cycles it takes to execute the interrupt service routine plus 1 t_{CYC}.

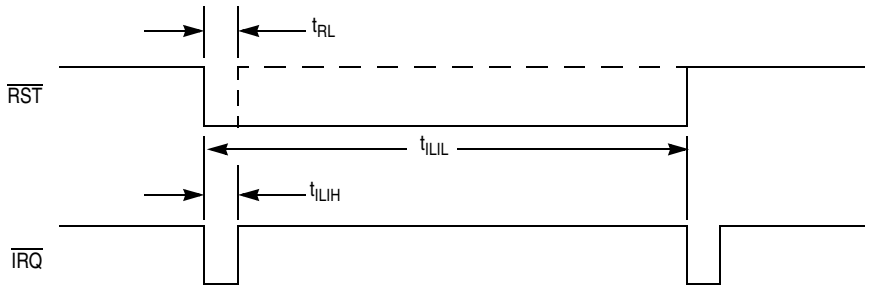


Figure 17-1. $\overline{RST}$ and $\overline{IRQ}$ Timing

17.7 5V Oscillator Characteristics

Table 17-6. Oscillator Specifications (5V)

Characteristic	Symbol	Min	Typ	Max	Unit
Internal oscillator clock frequency	f_{ICLK}		50k[1]		Hz
External reference clock to OSC1 [2]	f_{OSC}	dc	—	32M	Hz
Crystal reference frequency [3]	$f_{XTALCLK}$		—	32M	Hz
Crystal load capacitance [4]	C_L	—	—	—	
Crystal fixed capacitance [3]	C_1	—	$2 \times C_L$	—	
Crystal tuning capacitance [3]	C_2	—	$2 \times C_L$	—	
Feedback bias resistor	R_B	—	10 MΩ	—	
Series resistor [3], [5]	R_S	—	—	—	
External RC clock frequency	f_{RCCLK}	2M	—	12M	Hz
RC oscillator external R	R_{EXT}	See Figure 17-2			Ω
RC oscillator external C	C_{EXT}	—	10	—	pF

1. Typical value reflect average measurements at midpoint of voltage range, 25 °C only. See Figure 17-5 for plot.
2. No more than 10% duty cycle deviation from 50%.
3. Fundamental mode crystals only.
4. Consult crystal vendor data sheet.
5. Not required for high frequency crystals.

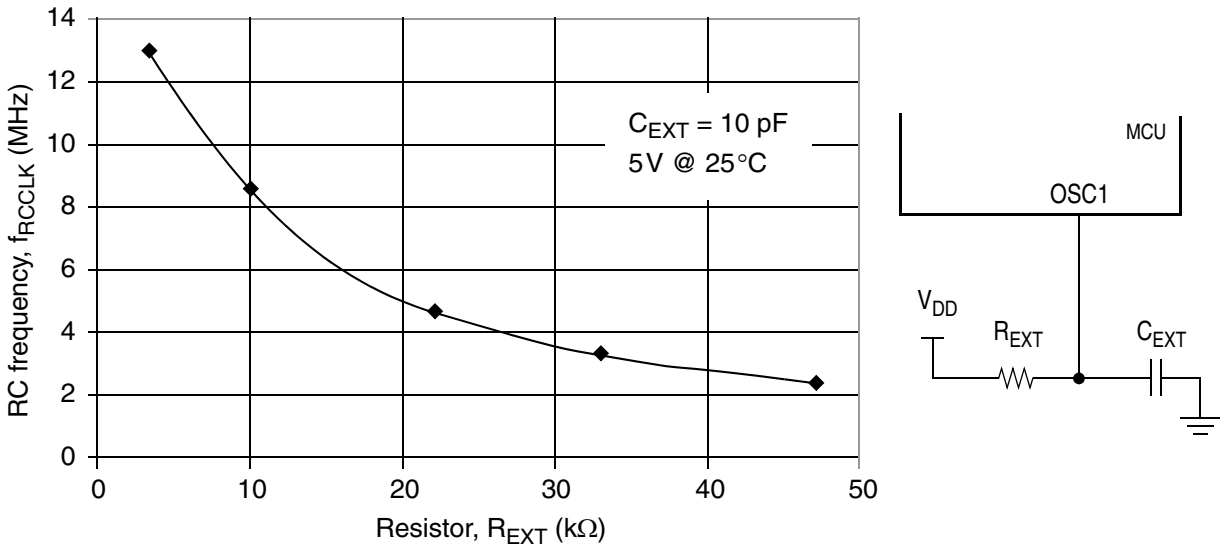


Figure 17-2. RC vs. Frequency (5V @25°C)

17.8 3V DC Electrical Characteristics

Table 17-7. DC Electrical Characteristics (3V)

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Output high voltage (I_{LOAD} = –1.0mA) PTA0–PTA7, PTB0–PTB7, PTD0–PTD7, PTE0–PTE1	V_{OH}	V_{DD}–0.4	—	—	V
Output low voltage (I_{LOAD} = 0.8mA) PTA6, PTB0–PTB7, PTD0, PTD1, PTD4, PTD5, PTE0–PTE1	V_{OL}	—	—	0.4	V
Output low voltage (I_{LOAD} = 20mA) PTD6, PTD7	V_{OL}	—	—	0.5	V
LED drives (V_{OL} = 1.8V) PTA0–PTA5, PTA7, PTD2, PTD3, PTD6, PTD7	I_{OL}	3	8	12	mA
Input high voltage PTA0–PTA7, PTB0–PTB7, PTD0–PTD7, PTE0–PTE1, $\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input low voltage PTA0–PTA7, PTB0–PTB7, PTD0–PTD7, PTE0–PTE1,$\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IL}	V_{SS}	—	0.3 × V_{DD}	V
V_{DD} supply current, f_{OP} = 4MHz Run[3] XTAL oscillator option RC oscillator option Wait[4] XTAL oscillator option RC oscillator option Stop[5] (–40°C to 85°C) XTAL oscillator option RC oscillator option	I_{DD}	 — — — — — —	 3 4 1 2 0.5 0.3	 8 10 4.5 6 5 2	 mA mA mA mA µA µA
Digital I/O ports Hi-Z leakage current	I_{IL}	—	—	± 10	µA
Input current	I_{IN}	—	—	± 1	µA

Table 17-7. DC Electrical Characteristics (3 V)

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Capacitance Ports (as input or output)	C_{OUT} C_{IN}	— —	— —	12 8	pF
POR rearm voltage[6]	V_{POR}	0	—	100	mV
POR rise time ramp rate[7]	R_{POR}	0.035	—	—	V/ms
Monitor mode entry voltage	V_{TST}	$1.5 \times V_{DD}$	—	8.5	V
Pullup resistors[8] PTD6, PTD7 $\overline{RST}$, $\overline{IRQ}$, PTA0–PTA7	 R_{PU1} R_{PU2}	 1.8 16	 3.3 26	 4.8 36	 kΩ kΩ
Low-voltage inhibit, trip voltage (No hysteresis implemented for 3V LVI)	V_{LVI3}	2.18	2.49	2.68	V

1. V_{DD} = 2.7 to 3.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted.
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. Run (operating) I_{DD} measured using external square wave clock source (f_{OP} = 4 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects run I_{DD}. Measured with all modules enabled.
4. Wait I_{DD} measured using external square wave clock source (f_{OP} = 4 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects wait I_{DD}.
5. Stop I_{DD} measured with OSC1 grounded; no port pins sourcing current. LVI is disabled.
6. Maximum is highest voltage that POR is guaranteed.
7. If minimum V_{DD} is not reached before the internal POR reset is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached.
8. R_{PU1} and R_{PU2} are measured at V_{DD} = 5.0 V.

17.9 3V Control Timing

Table 17-8. Control Timing (3 V)

Characteristic[1]	Symbol	Min	Max	Unit
Internal operating frequency	f_{OP}	—	4	MHz
$\overline{RST}$ input pulse width low[2]	t_{RL}	1.5	—	µs
TIM2 external clock input	f_{T2CLK}	—	2	MHz
$\overline{IRQ}$ interrupt pulse width low (edge-triggered)[3]	t_{ILIH}	200	—	ns
$\overline{IRQ}$ interrupt pulse period[3]	t_{ILIL}	Note[4]	—	t_{CYC}

1. V_{DD} = 2.7 to 3.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H; timing shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.
2. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.
3. Values are based on characterization results, not tested in production.
4. The minimum period is the number of cycles it takes to execute the interrupt service routine plus 1 t_{CYC}.

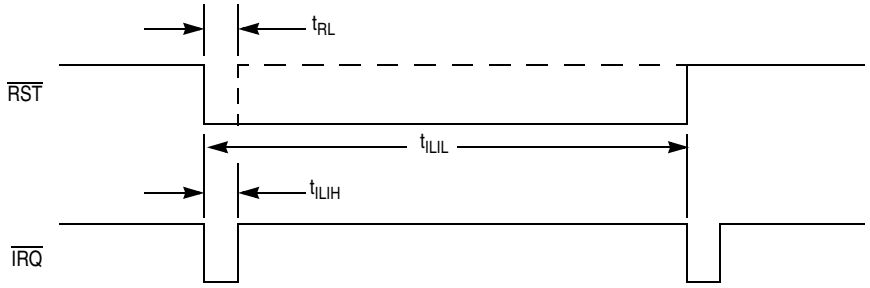


Figure 17-3. $\overline{RST}$ and $\overline{IRQ}$ Timing

17.10 3V Oscillator Characteristics

Table 17-9. Oscillator Specifications (3V)

Characteristic	Symbol	Min	Typ	Max	Unit
Internal oscillator clock frequency	f_{ICLK}		45k[1]		Hz
External reference clock to OSC1 [2]	f_{OSC}	dc	—	16M	Hz
Crystal reference frequency [3]	$f_{XTALCLK}$		—	16M	Hz
Crystal load capacitance [4]	C_L	—	—	—	
Crystal fixed capacitance [3]	C_1	—	$2 \times C_L$	—	
Crystal tuning capacitance [3]	C_2	—	$2 \times C_L$	—	
Feedback bias resistor	R_B	—	10 MΩ	—	
Series resistor [3], [5]	R_S	—	—	—	
External RC clock frequency	f_{RCCLK}	2M	—	10M	Hz
RC oscillator external R	R_{EXT}	See Figure 17-4			Ω
RC oscillator external C	C_{EXT}	—	10	—	pF

1. Typical value reflect average measurements at midpoint of voltage range, 25 °C only. See Figure 17-5 for plot.
2. No more than 10% duty cycle deviation from 50%.
3. Fundamental mode crystals only.
4. Consult crystal vendor data sheet.
5. Not required for high frequency crystals.

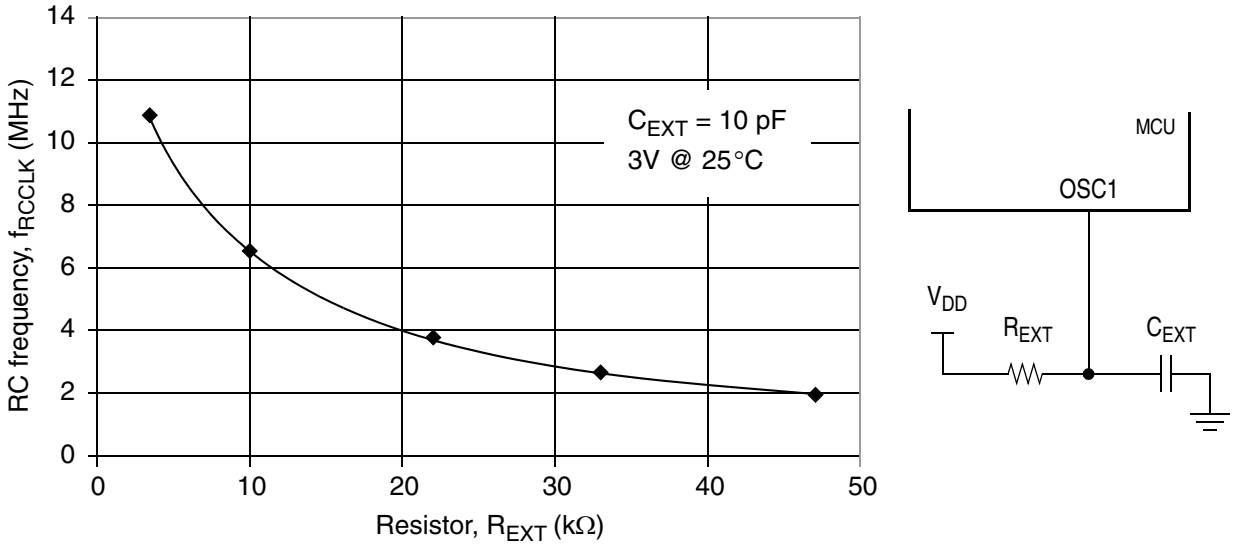


Figure 17-4. RC vs. Frequency (3V @25°C)

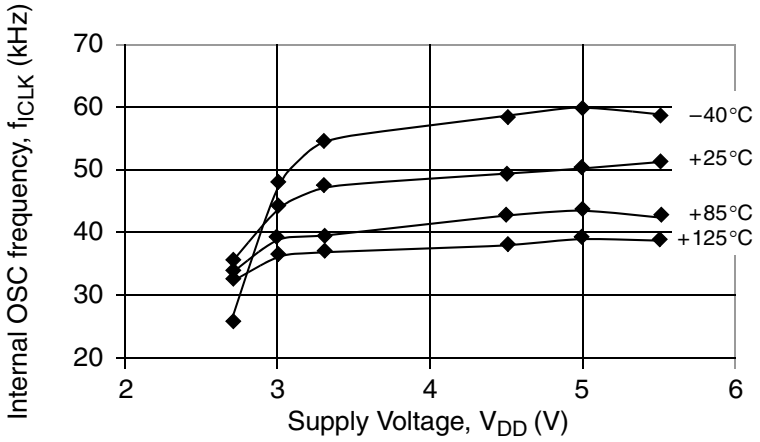


Figure 17-5. Internal Oscillator Frequency

17.11 Typical Supply Currents

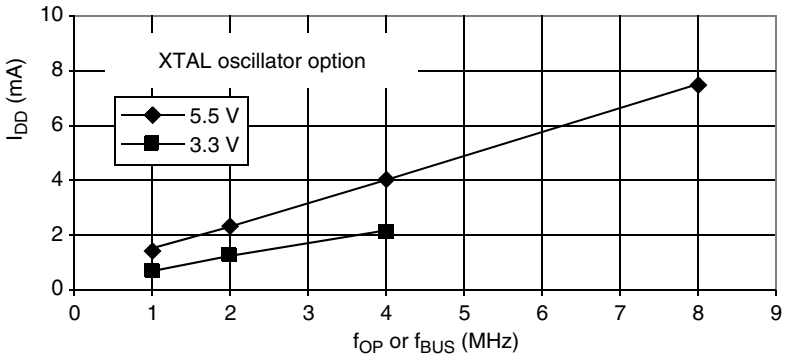


Figure 17-6. Typical Operating I_{DD} (XTAL osc), with All Modules Turned On (25 °C)

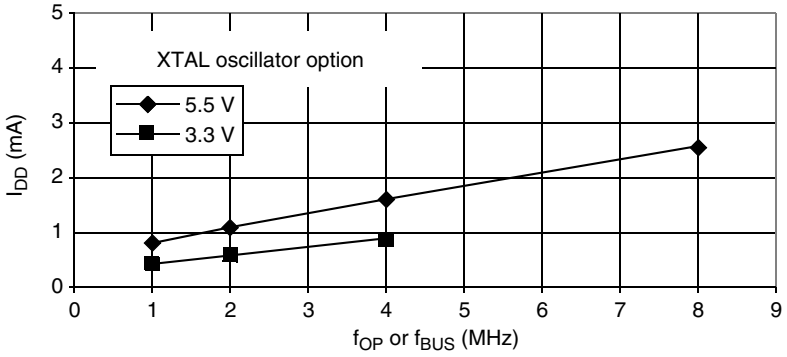


Figure 17-7. Typical Wait Mode I_{DD} (XTAL osc), with All Modules Turned Off (25 °C)

17.12 Timer Interface Module Characteristics

Table 17-10. Timer Interface Module Characteristics (5V and 3V)

Characteristic	Symbol	Min	Max	Unit
Input capture pulse width	t_{TIH}, t_{TIL}	$1/f_{OP}$	—	
Input clock pulse width (T2CLK pulse width)	t_{LMIN}, t_{HMIN}	$(1/f_{OP}) + 5$ns	—	

17.13 ADC Characteristics

Table 17-11. ADC Characteristics (5V and 3V)

Characteristic	Symbol	Min	Max	Unit	Comments
Supply voltage	V_{DDAD}	2.7 (V_{DD} min)	5.5 (V_{DD} max)	V	
Input voltages	V_{ADIN}	V_{SS}	V_{DD}	V	
Resolution	B_{AD}	8	8	Bits	
Absolute accuracy	A_{AD}	± 0.5	± 1.5	LSB	Includes quantization
ADC internal clock	f_{ADIC}	0.5	1.048	MHz	$t_{AIC} = 1/f_{ADIC}$, tested only at 1 MHz
Conversion range	R_{AD}	V_{SS}	V_{DD}	V	
Power-up time	t_{ADPU}	16		t_{AIC} cycles	
Conversion time	t_{ADC}	14	15	t_{AIC} cycles	
Sample time[1]	t_{ADS}	5	—	t_{AIC} cycles	
Zero input reading[2]	Z_{ADI}	00	01	Hex	$V_{IN} = V_{SS}$
Full-scale reading[3]	F_{ADI}	FE	FF	Hex	$V_{IN} = V_{DD}$
Input capacitance	C_{ADI}	—	(20) 8	pF	Not tested
Input leakage[3] Port B/port D	—	—	± 1	µA	

1. Source impedances greater than 10 kΩ adversely affect internal RC charging time during input sampling.
2. Zero-input/full-scale reading requires sufficient decoupling measures for accurate conversions.
3. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

17.14 Memory Characteristics

Table 17-12. Memory Characteristics

Characteristic	Symbol	Min	Max	Unit
RAM data retention voltage	V_{RDR}	1.3	—	V
FLASH program bus clock frequency	—	1	—	MHz
FLASH read bus clock frequency	f_{read} (1)	32k	8M	Hz
FLASH page erase time	t_{erase} (2)	4	—	ms
FLASH mass erase time	t_{merase} (3)	4	—	ms
FLASH PGM/ERASE to HVEN set up time	t_{nvs}	10	—	µs
FLASH high-voltage hold time	t_{nvh}	5	—	µs
FLASH high-voltage hold time (mass erase)	t_{nvhl}	100	—	µs
FLASH program hold time	t_{pgs}	5	—	µs
FLASH program time	t_{prog}	30	40	µs
FLASH return to read time	t_{rcv} (4)	1	—	µs
FLASH cumulative program hv period	t_{HV} (5)	—	4	ms
FLASH row erase endurance (6)	—	10k	—	cycles
FLASH row program endurance (7)	—	10k	—	cycles
FLASH data retention time (8)	—	10	—	years

1. f_{read} is defined as the frequency range for which the FLASH memory can be read.
2. If the page erase time is longer than t_{erase} (Min), there is no erase-disturb, but it reduces the endurance of the FLASH memory.
3. If the mass erase time is longer than t_{merase} (Min), there is no erase-disturb, but it reduces the endurance of the FLASH memory.
4. t_{rcv} is defined as the time it needs before the FLASH can be read after turning off the high voltage charge pump, by clearing HVEN to logic 0.
5. t_{HV} is defined as the cumulative high voltage programming time to the same row before next erase. t_{HV} must satisfy this condition: $t_{nvs} + t_{nvh} + t_{pgs} + (t_{prog} \times 32) \leq t_{HV}$ max.
6. The minimum row endurance value specifies each row of the FLASH memory is guaranteed to work for at least this many erase / program cycles.
7. The minimum row endurance value specifies each row of the FLASH memory is guaranteed to work for at least this many erase / program cycles.
8. The FLASH is guaranteed to retain data over the entire operating temperature range for at least the minimum time specified.

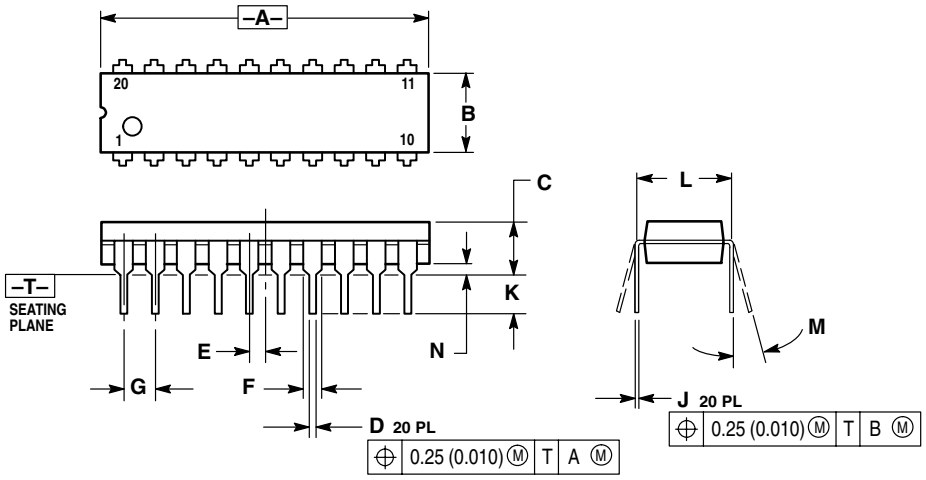
Chapter 18
Mechanical Specifications

18.1 Introduction

This section gives the dimensions for:

- 20-pin plastic dual in-line package (case #738)
- 20-pin small outline integrated circuit package (case #751D)
- 28-pin plastic dual in-line package (case #710)
- 28-pin small outline integrated circuit package (case #751F)
- 32-pin shrink dual in-line package (case #1376)
- 32-pin low-profile quad flat pack (case #873A)

18.2 20-Pin Plastic Dual In-Line Package (PDIP)

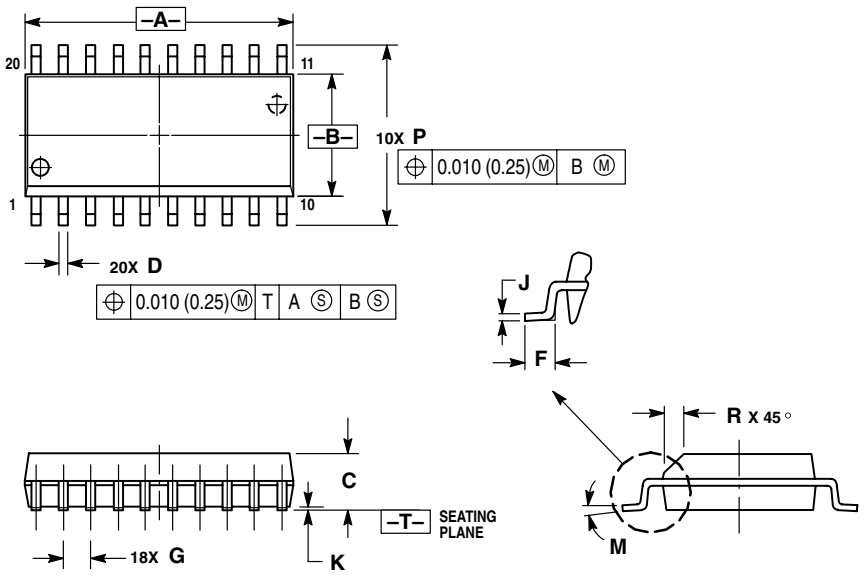


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION L TO CENTER OF LEAD WHEN FORMED PARALLEL.
4. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

	INCHES		MILLIMETERS	
DIM	MIN	MAX	MIN	MAX
A	1.010	1.070	25.66	27.17
B	0.240	0.260	6.10	6.60
C	0.150	0.180	3.81	4.57
D	0.015	0.022	0.39	0.55
E	0.050 BSC		1.27 BSC	
F	0.050	0.070	1.27	1.77
G	0.100 BSC		2.54 BSC	
J	0.008	0.015	0.21	0.38
K	0.110	0.140	2.80	3.55
L	0.300 BSC		7.62 BSC	
M	0°	15°	0°	15°
N	0.020	0.040	0.51	1.01

Figure 18-1. 20-Pin PDIP (Case #738)

18.3 20-Pin Small Outline Integrated Circuit Package (SOIC)

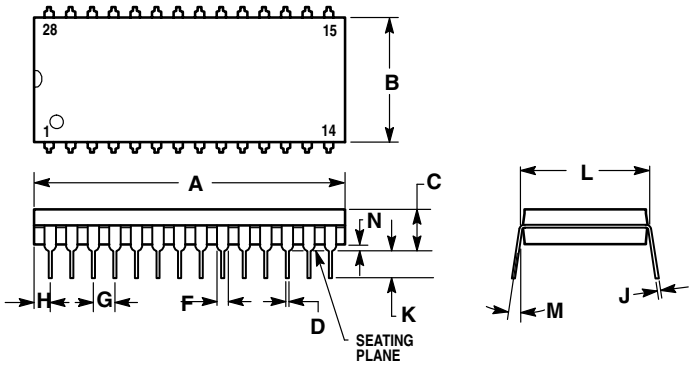


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION.
4. MAXIMUM MOLD PROTRUSION 0.150 (0.006) PER SIDE.
5. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.13 (0.005) TOTAL IN EXCESS OF D DIMENSION AT MAXIMUM MATERIAL CONDITION.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	12.65	12.95	0.499	0.510
B	7.40	7.60	0.292	0.299
C	2.35	2.65	0.093	0.104
D	0.35	0.49	0.014	0.019
F	0.50	0.90	0.020	0.035
G	1.27 BSC		0.050 BSC	
J	0.25	0.32	0.010	0.012
K	0.10	0.25	0.004	0.009
M	0°	7°	0°	7°
P	10.05	10.55	0.395	0.415
R	0.25	0.75	0.010	0.029

Figure 18-2. 20-Pin SOIC (Case #751D)

18.4 28-Pin Plastic Dual In-Line Package (PDIP)

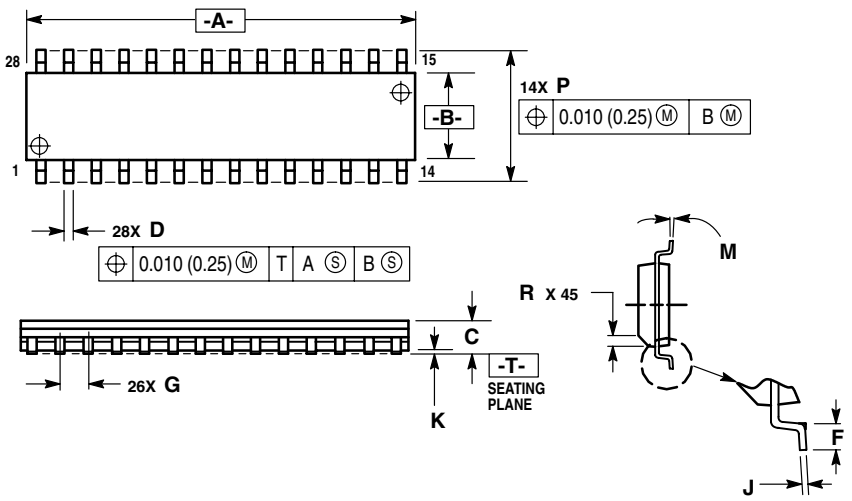


NOTES:
1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25 (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	36.45	37.21	1.435	1.465
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

Figure 18-3. 28-Pin PDIP (Case #710)

18.5 28-Pin Small Outline Integrated Circuit Package (SOIC)

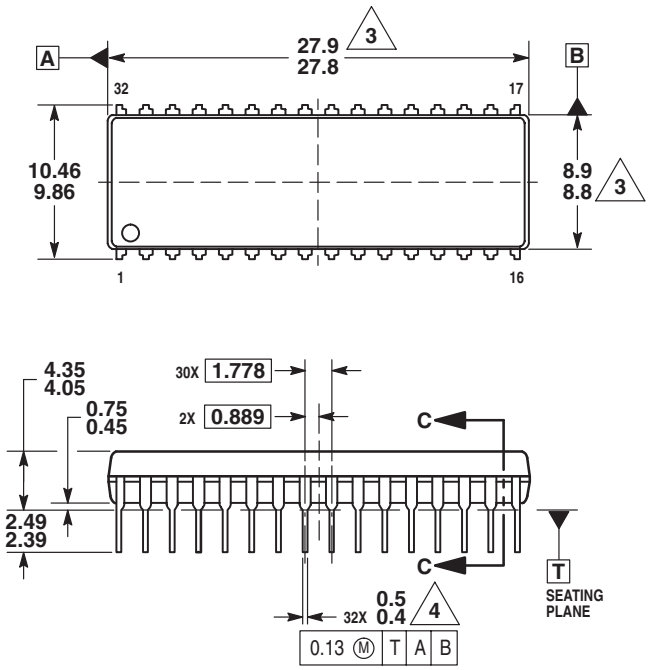


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSION A AND B DO NOT INCLUDE MOLD PROTRUSION.
4. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.
5. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.13 (0.005) TOTAL IN EXCESS OF D DIMENSION AT MAXIMUM MATERIAL CONDITION.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	17.80	18.05	0.701	0.711
B	7.40	7.60	0.292	0.299
C	2.35	2.65	0.093	0.104
D	0.35	0.49	0.014	0.019
F	0.41	0.90	0.016	0.035
G	1.27 BSC		0.050 BSC	
J	0.23	0.32	0.009	0.013
K	0.13	0.29	0.005	0.011
M	0°	8°	0°	8°
P	10.01	10.55	0.395	0.415
R	0.25	0.75	0.010	0.029

Figure 18-4. 28-Pin SOIC (Case #751F)

18.6 32-Pin Shrink Dual In-Line Package (SDIP)



NOTES:
1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5, 1994.
3. DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS.
4. DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION.

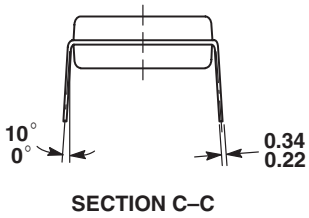
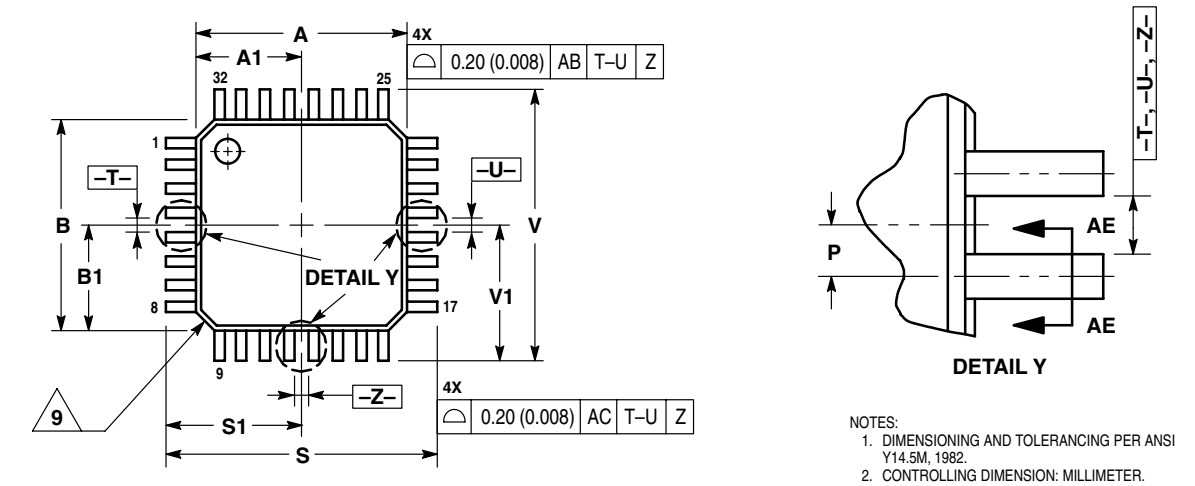
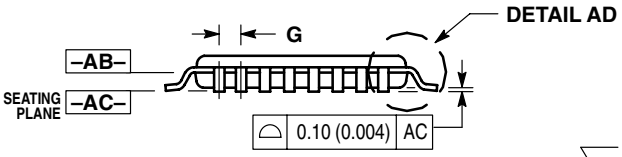
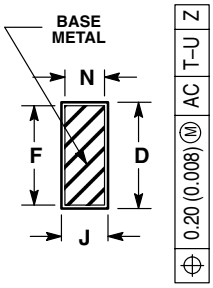


Figure 18-5. 32-Pin SDIP (Case #1376)

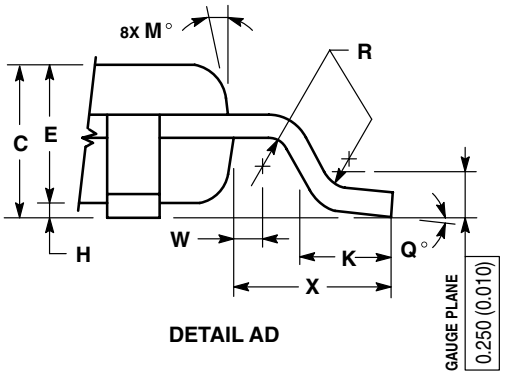
18.7 32-Pin Low-Profile Quad Flat Pack (LQFP)







SECTION AE–AE



DETAIL AD

NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE –AB– IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS –T–, –U–, AND –Z– TO BE DETERMINED AT DATUM PLANE –AB–.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE –AC–.
6. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.250 (0.010) PER SIDE. DIMENSIONS A AND B DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE –AB–.
7. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. DAMBAR PROTRUSION SHALL NOT CAUSE THE D DIMENSION TO EXCEED 0.520 (0.020).
8. MINIMUM SOLDER PLATE THICKNESS SHALL BE 0.0076 (0.0003).
9. EXACT SHAPE OF EACH CORNER MAY VARY FROM DEPICTION.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	7.000 BSC		0.276 BSC	
A1	3.500 BSC		0.138 BSC	
B	7.000 BSC		0.276 BSC	
B1	3.500 BSC		0.138 BSC	
C	1.400	1.600	0.055	0.063
D	0.300	0.450	0.012	0.018
E	1.350	1.450	0.053	0.057
F	0.300	0.400	0.012	0.016
G	0.800 BSC		0.031 BSC	
H	0.050	0.150	0.002	0.006
J	0.090	0.200	0.004	0.008
K	0.500	0.700	0.020	0.028
M	12° REF		12° REF	
N	0.090	0.160	0.004	0.006
P	0.400 BSC		0.016 BSC	
Q	1°	5°	1°	5°
R	0.150	0.250	0.006	0.010
S	9.000 BSC		0.354 BSC	
S1	4.500 BSC		0.177 BSC	
V	9.000 BSC		0.354 BSC	
V1	4.500 BSC		0.177 BSC	
W	0.200 REF		0.008 REF	
X	1.000 REF		0.039 REF	

Figure 18-6. 32-Pin LQFP (Case #873A)

Chapter 19
Ordering Information

19.1 Introduction

This section contains ordering numbers for the MC68HC908JL8.

19.2 MC Order Numbers

Table 19-1. MC Order Numbers

MC Order Number	Operating Temperature Range	Package
MC68HC908JK8CP	–40 °C to +85 °C	20-pin PDIP
MC68HC908JK8MP	–40 °C to +125 °C	
MC68HC908JK8CDW	–40 °C to +85 °C	20-pin SOIC
MC68HC908JK8MDW	–40 °C to +125 °C	
MC68HC908JL8CP	–40 °C to +85 °C	28-pin PDIP
MC68HC908JL8MP	–40 °C to +125 °C	
MC68HC908JL8CDW	–40 °C to +85 °C	28-pin SOIC
MC68HC908JL8MDW	–40 °C to +125 °C	
MC68HC908JL8CSP	–40 °C to +85 °C	32-pin SDIP
MC68HC908JL8MSP	–40 °C to +125 °C	
MC68HC908JL8CFA	–40 °C to +85 °C	32-pin LQFP
MC68HC908JL8MFA	–40 °C to +125 °C	

NOTE: Temperature grade "M" is available for V_{DD} = 5V only.

Appendix A
MC68HC08JL8

A.1 Introduction

This section introduces the MC68HC08JL8, the ROM part equivalent to the MC68HC908JL8/JK8. The entire data book applies to this ROM device, with exceptions outlined in this appendix.

Table A-1. Summary of MC68HC08JL8 and MC68HC908JL8 Differences

	MC68HC08JL8	MC68HC908JL8
Memory ($DC00–$FBFF)	8,192 bytes ROM	8,192 bytes FLASH
User vectors ($FFDC–$FFFF)	36 bytes ROM	36 bytes FLASH
Registers at $FE08 and $FFCF	Not used; locations are reserved.	FLASH related registers. $FE08 — FLCR $FFCF — FLBPR
Mask option register ($FFD0)	Defined by mask; read only.	Read/write FLASH register.
Monitor ROM ($FC00–$FDFF and $FE10–$FFCE)	$FC00–$FDFF: Not used. $FE10–$FFCE: Used for testing purposes only.	Used for testing and FLASH programming/erasing.
Available Packages	20-pin PDIP (MC68HC08JK8) 20-pin SOIC (MC68HC08JK8) 28-pin PDIP 28-pin SOIC 32-pin SDIP 32-pin LQFP	20-pin PDIP (MC68HC908JK8) 20-pin SOIC (MC68HC908JK8) 28-pin PDIP 28-pin SOIC 32-pin SDIP 32-pin LQFP

A.2 MCU Block Diagram

Figure A-1 shows the block diagram of the MC68HC08JL8.

A.3 Memory Map

The MC68HC08JL8 has 8,192 bytes of user ROM from $DC00 to $FBFF, and 36 bytes of user ROM vectors from $FFDC to $FFFF. On the MC68HC908JL8, these memory locations are FLASH memory.

Figure A-2 shows the memory map of the MC68HC08JL8.

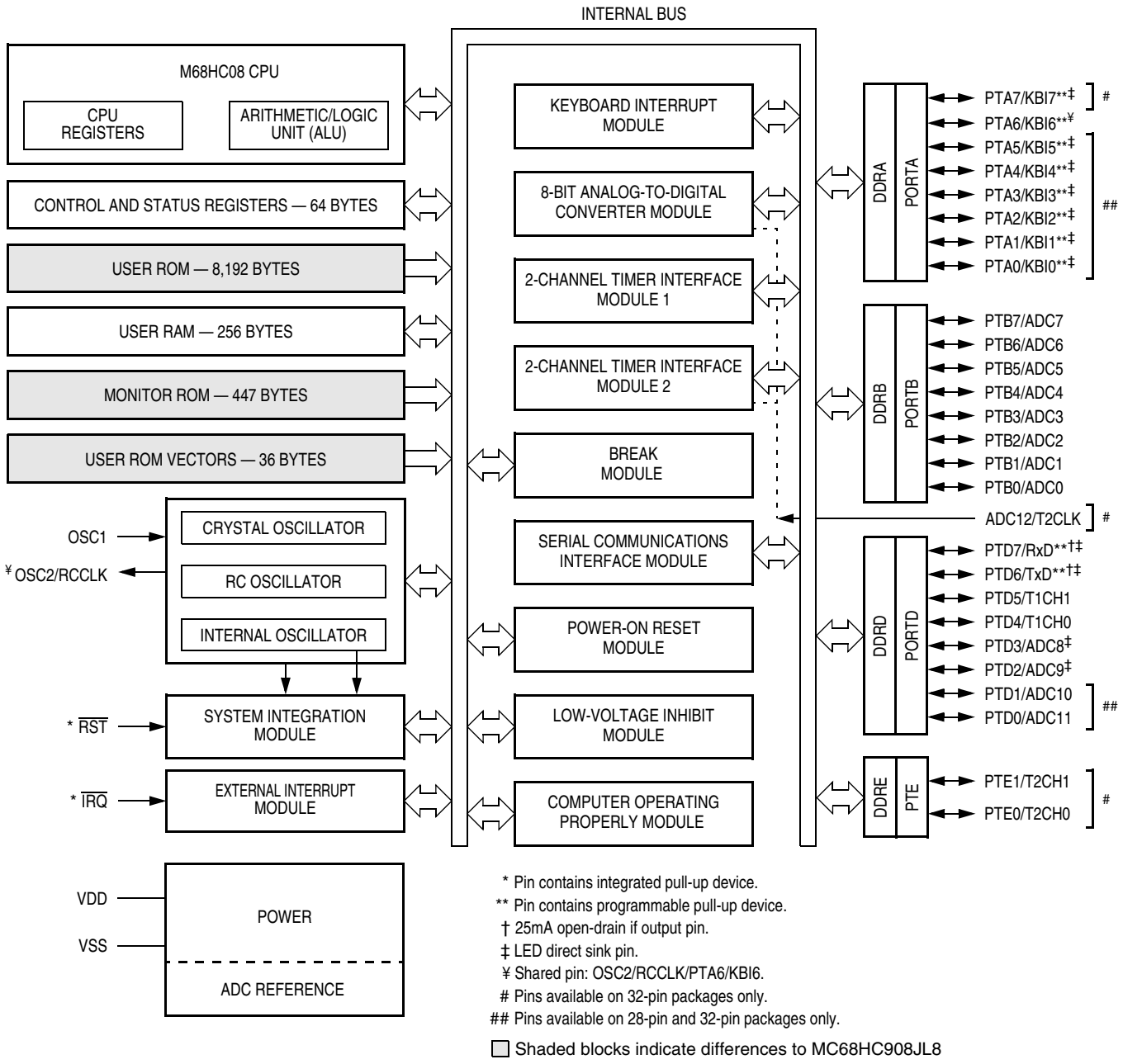


Figure A-1. MC68HC08JL8 Block Diagram

A.4 Reserved Registers

The two registers at $FE08 and $FFCF are reserved locations on the MC68HC08JL8.

On the MC68HC908JL8, these two locations are the FLASH control register and the FLASH block protect register respectively.

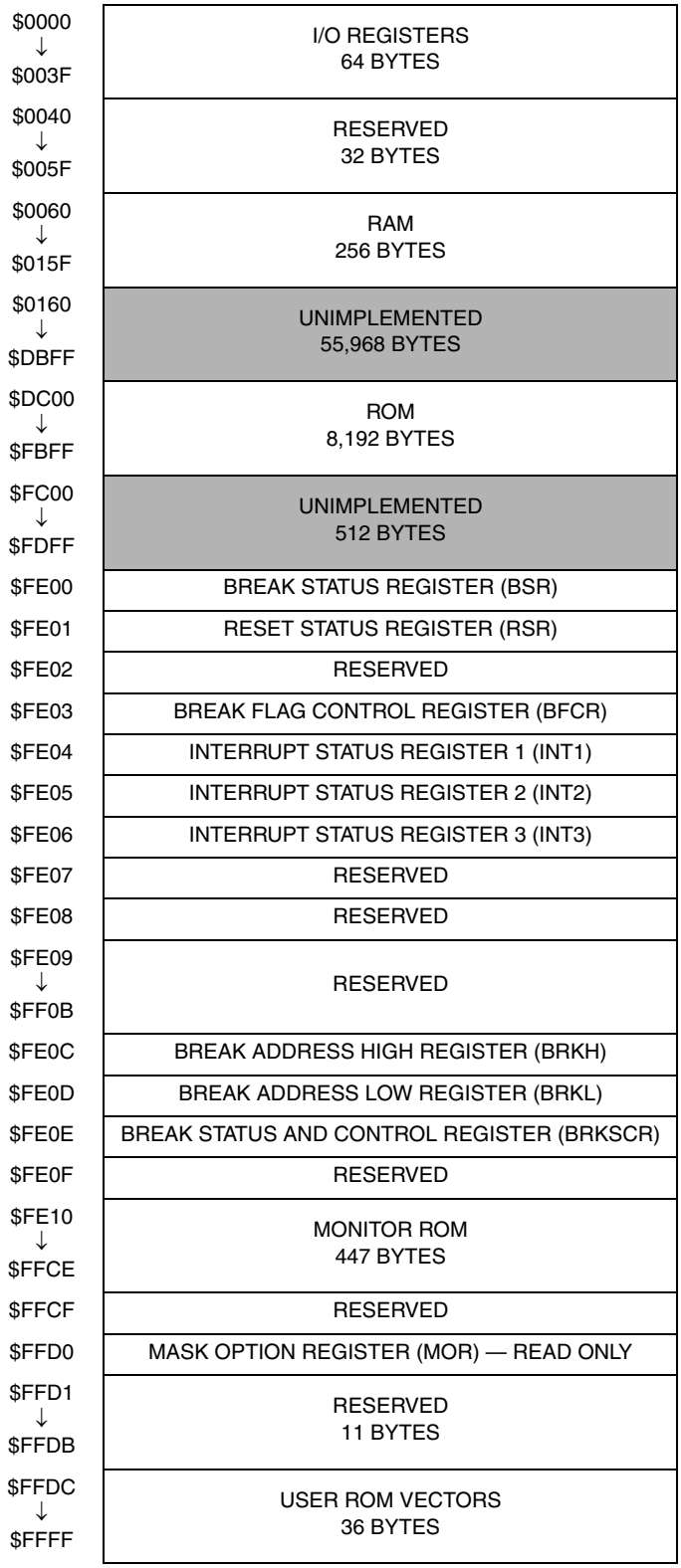


Figure A-2. MC68HC08JL8 Memory Map

A.5 Mask Option Register

The mask option register at $FFD0 is read only. The value is defined by mask option (hard-wired connections) specified at the time as the ROM code submission.

On the MC68HC908JL8, the MOR is implemented as a FLASH, which can be programmed, erased, and read.

A.6 Monitor ROM

The monitor program (monitor ROM: $FE10–$FFCE) on the MC68HC08JL8 is for device testing only. $FC00–$FDFF are unused.

A.7 Electrical Specifications

Electrical specifications for the MC68HC908JL8 apply to the MC68HC08JL8, except for the parameters indicated below.

A.7.1 DC Electrical Characteristics

Table A-2. DC Electrical Characteristics (5V)

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
V_{DD} supply current, f_{OP} = 8MHz RC oscillator option	I_{DD}	Values same as, and characterized from MC68HC908JL8, but not tested.			
Low-voltage inhibit, trip falling voltage	V_{TRIPF}	3.55 (3.60)[3]	4.02 (4.25)	4.48 (4.48)	V
Low-voltage inhibit, trip rising voltage	V_{TRIPR}	3.66 (3.75)	4.13 (4.40)	4.59 (4.63)	V

1. V_{DD} = 4.5 to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted.
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. The numbers in parenthesis are MC68HC908JL8 values.

Table A-3. DC Electrical Characteristics (3V)

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
V_{DD} supply current, f_{OP} = 4MHz RC oscillator option	I_{DD}	Values same as, and characterized from MC68HC908JL8, but not tested.			
Low-voltage inhibit, trip voltage (No hysteresis implemented for 3V LVI)	V_{LVI3}	2.1 (2.18)[3]	2.4 (2.49)	2.69 (2.68)	V

1. V_{DD} = 2.7 to 3.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted.
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. The numbers in parenthesis are MC68HC908JL8 values.

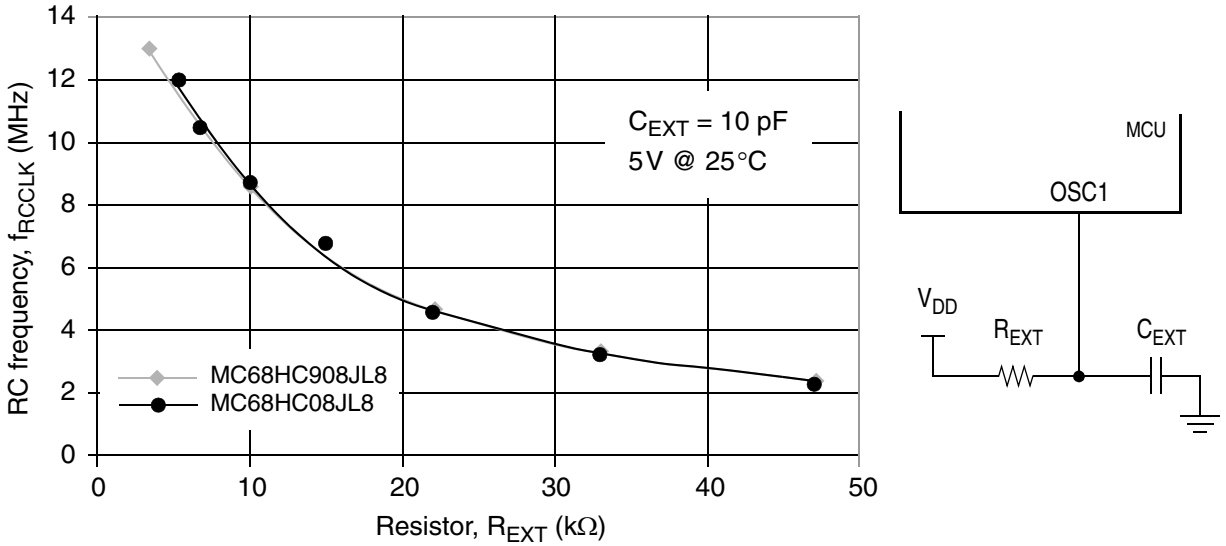


Figure A-3. RC vs. Frequency (5V @25°C)

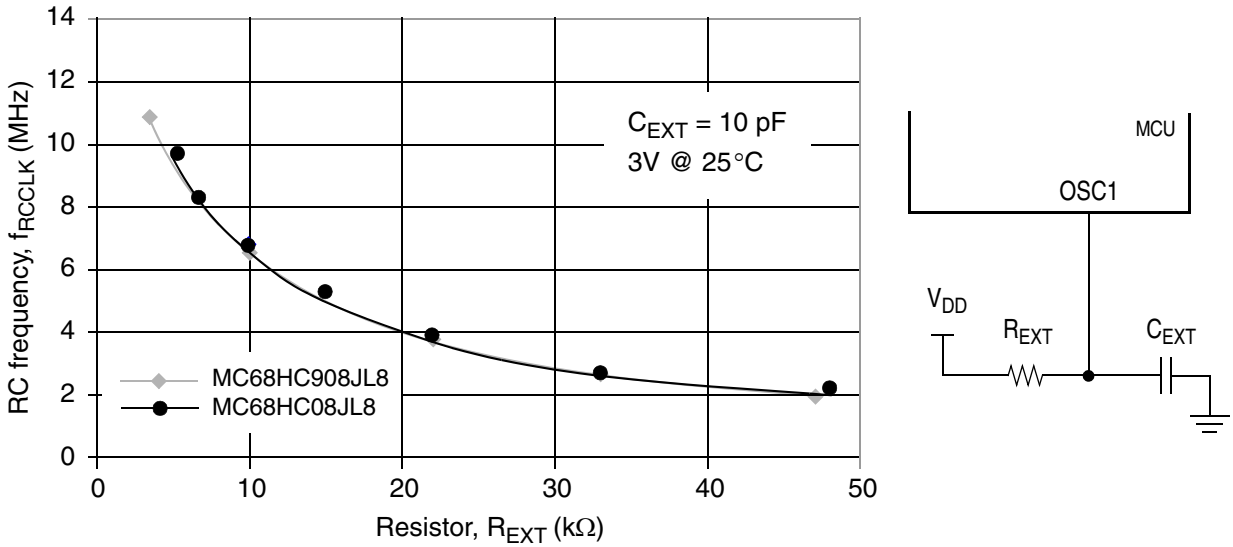


Figure A-4. RC vs. Frequency (3V @25°C)

A.8 Memory Characteristics

Table A-4. Memory Characteristics

Characteristic	Symbol	Min	Max	Unit
RAM data retention voltage	V_{RDR}	1.3	—	V

Notes:
Since MC68HC08JL8 is a ROM device, FLASH memory electrical characteristics do not apply.

A.9 MC68HC08JL8 Order Numbers

These part numbers are generic numbers only. To place an order, ROM code must be submitted to the ROM Processing Center (RPC).

Table A-5. MC68HC08JL8 Order Numbers

MC Order Number	Operating Temperature Range	Package
MC68HC08JK8CP	–40 °C to +85 °C	20-pin PDIP
MC68HC08JK8MP	–40 °C to +125 °C	
MC68HC08JK8CDW	–40 °C to +85 °C	20-pin SOIC
MC68HC08JK8MDW	–40 °C to +125 °C	
MC68HC08JL8CP	–40 °C to +85 °C	28-pin PDIP
MC68HC08JL8MP	–40 °C to +125 °C	
MC68HC08JL8CDW	–40 °C to +85 °C	28-pin SOIC
MC68HC08JL8MDW	–40 °C to +125 °C	
MC68HC08JL8CSP	–40 °C to +85 °C	32-pin SDIP
MC68HC08JL8MSP	–40 °C to +125 °C	
MC68HC08JL8CFA	–40 °C to +85 °C	32-pin LQFP
MC68HC08JL8MFA	–40 °C to +125 °C	

NOTE: Temperature grade "M" is available for V_{DD} = 5V only.

Appendix B
MC68HC908KL8

B.1 Introduction

This appendix introduces the MC68HC908KL8, an ADC-less device of the MC68HC908JL8. The entire data book applies to this device, with exceptions outlined in this appendix.

Table B-1. Summary of MC68HC908KL8 and MC68HC908JL8 Differences

	MC68HC908KL8	MC68HC908JL8
Analog-to-Digital Converter (ADC)	—	13-channel, 8-bit.
Registers at: $003C, $003E, and $003E	Not used; locations are reserved.	ADC registers.
Interrupt Vector at: $FFDE and $FFDF	Not used.	ADC interrupt vector.
Available Packages	— — 28-pin PDIP 28-pin SOIC 32-pin SDIP —	20-pin PDIP (MC68HC908JK8) 20-pin SOIC (MC68HC908JK8) 28-pin PDIP 28-pin SOIC 32-pin SDIP 32-pin LQFP

B.2 MCU Block Diagram

Figure B-1 shows the block diagram of the MC68HC908KL8.

B.3 Pin Assignments

Figure B-2 and Figure B-3 show the pin assignments for the MC68HC908KL8.

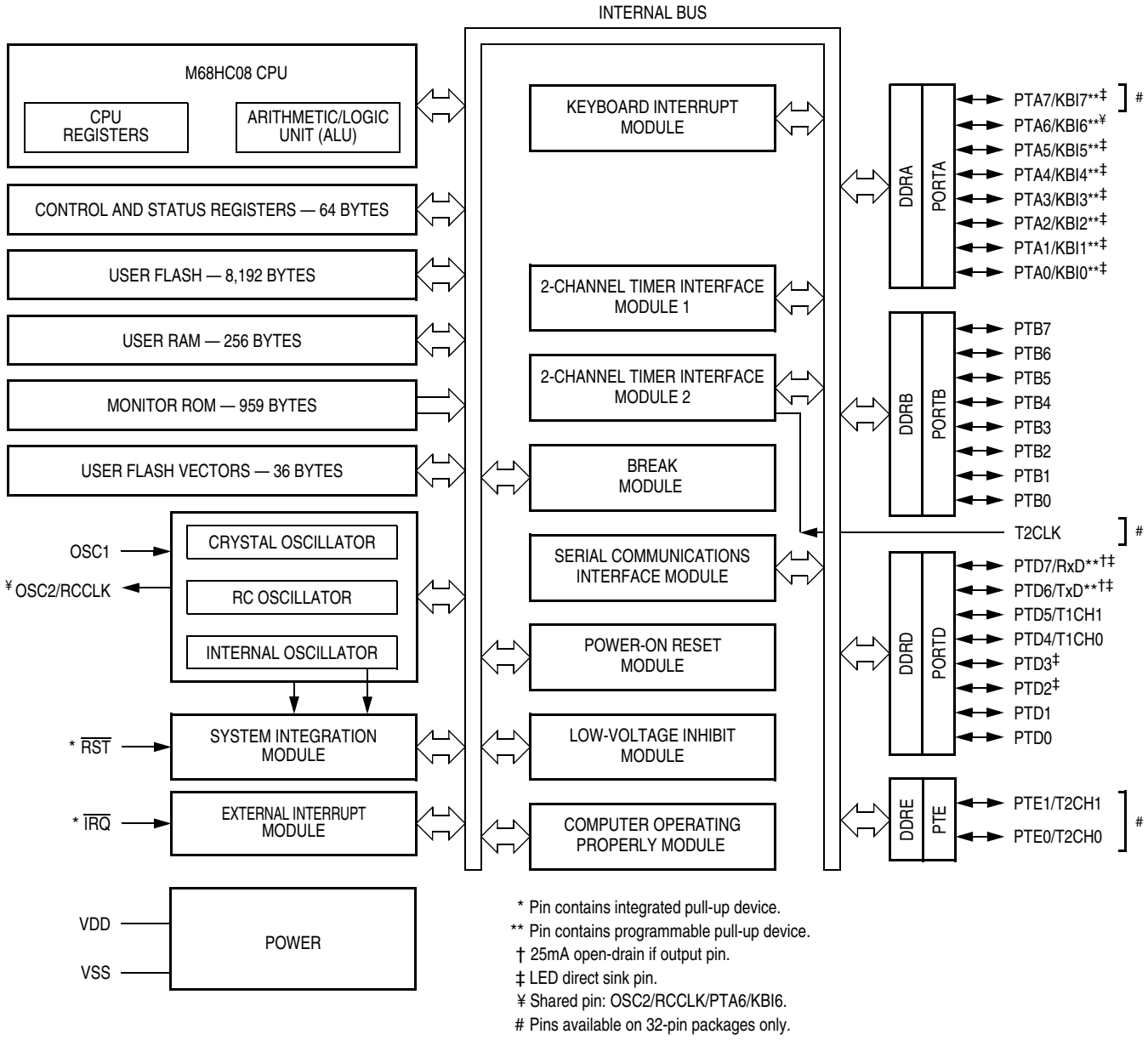


Figure B-1. MC68HC908KL8 Block Diagram

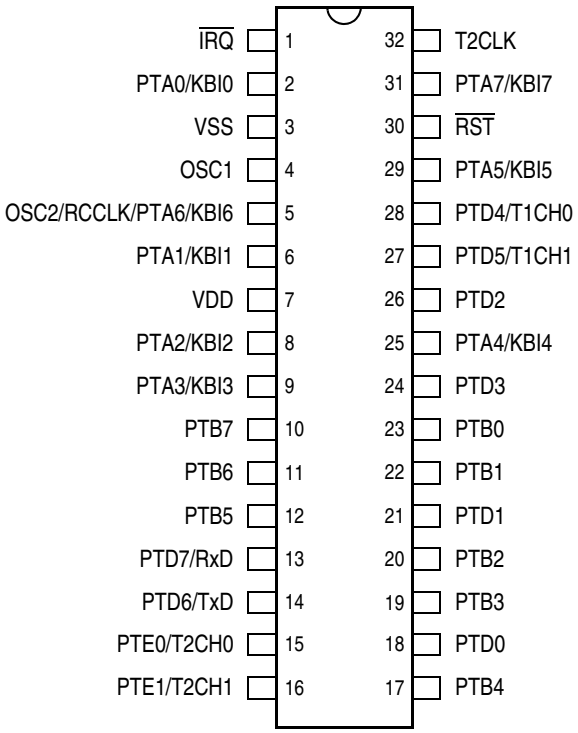
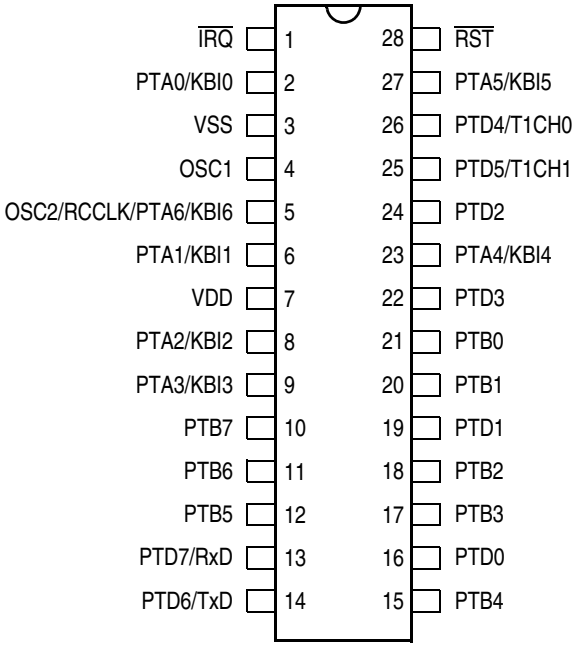


Figure B-2. 32-Pin SDIP Pin Assignment



Pins not available on 28-pin packages	
	PTE0/T2CH0
	PTE1/T2CH1
	T2CLK
PTA7/KBI7	

Internal pads are unconnected.
Set these unused port I/Os to output low.

Figure B-3. 28-Pin PDIP/SOIC Pin Assignment

B.4 Reserved Registers

The following registers are reserved location on the MC68HC908KL8.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003C	Reserved	Read: Write: Reset:	R	R	R	R	R	R	R	R
$003D	Reserved	Read: Write: Reset:	R	R	R	R	R	R	R	R
$003E	Reserved	Read: Write: Reset:	R	R	R	R	R	R	R	R

Figure B-4. Reserved Registers

B.5 Reserved Vectors

The following are reserved interrupt vectors on the MC68HC908KL8.

Table B-2. Reserved Vectors

Vector Priority	INT Flag	Address	Vector
—	IF15	$FFDE	Reserved
		$FFDF	Reserved

B.6 MC68HC908KL8 Order Numbers

Table B-3. MC68HC908KL8 Order Numbers

MC Order Number	Operating Temperature Range	Package
MC68HC908KL8CP	–40 °C to +85 °C	28-pin PDIP
MC68HC908KL8CDW	–40 °C to +85 °C	28-pin SOIC
MC68HC908KL8CSP	–40 °C to +85 °C	32-pin SDIP

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

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2005. All rights reserved.

MC68HC908JL8
Rev. 3.1, 3/2005

freescale™
semiconductor

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View MC68HC908JL8CFAE on WIN SOURCE
- Freescale Semiconductor - NXP Information

Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management