# THE DATASHEET OF

# MC68HC908QY2CDTE

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

# MC68HC908QY4
# MC68HC908QT4
# MC68HC908QY2
# MC68HC908QT2
# MC68HC908QY1
# MC68HC908QT1

Data Sheet

**M68HC08**
**Microcontrollers**

MC68HC908QY4/D
Rev. 6
03/2010

freescale.com

freescale™
semiconductor

# MC68HC908QY4
# MC68HC908QT4
# MC68HC908QY2
# MC68HC908QT2
# MC68HC908QY1
# MC68HC908QT1

**Data Sheet**

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

http://freescale.com/

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

© Freescale Semiconductor, Inc., 2005–2010. All rights reserved.

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

## Revision History (Sheet 1 of 3)

| Date | Revision Level | Description | Page Number(s) |
|---|---|---|---|
| September, 2002 | N/A | Initial release | N/A |
| December, 2002 | 0.1 | 1.2 Features — Added 8-pin dual flat no lead (DFN) packages to features list. | 19 |
| | | Figure 1-2. MCU Pin Assignments — Figure updated to include DFN packages. | 21 |
| | | Figure 2-1. Memory Map — Clarified illegal address and unimplemented memory. | 27 |
| | | Figure 2-2. Control, Status, and Data Registers — Corrected bit definitions for Port A Data Register (PTA) and Data Direction Register A (DDRA). | 27 |
| | | Table 13-3. Interrupt Sources — Corrected vector addresses for keyboard interrupt and ADC conversion complete interrupt. | 118 |
| | | Chapter 13 System Integration Module (SIM) — Removed reference to break status register as it is duplicated in break module. | 113 |
| | | 11.3.1 Internal Oscillator and 11.3.1.1 Internal Oscillator Trimming — Clarified oscillator trim option ordering information and what to expect with untrimmed device. | 92 |
| | | Figure 11-5. Oscillator Trim Register (OSCTRIM) — Bit 1 designation corrected. | 98 |
| | | Figure 15-13. Monitor Mode Circuit (Internal Clock, No High Voltage) — Diagram updated for clarity. | 150 |
| | | Figure 12-1. I/O Port Register Summary — Corrected bit definitions for PTA7, DDRA7, and DDRA6. | 99 |
| | | Figure 12-2. Port A Data Register (PTA) — Corrected bit definition for PTA7. | 100 |
| | | Figure 12-3. Data Direction Register A (DDRA) — Corrected bit definitions for DDRA7 and DDRA6. | 101 |
| | | Figure 12-6. Port B Data Register (PTB) — Corrected bit definition for PTB1 | 103 |
| | | Chapter 9 Keyboard Interrupt Module (KBI) — Section reworked after deletion of auto wakeup for clarity. | 83 |
| | | Chapter 4 Auto Wakeup Module (AWU) — New section added for clarity. | 49 |
| | | Figure 10-1. LVI Module Block Diagram — Corrected LVI stop representation. | 87 |
| | | Chapter 16 Electrical Specifications — Extensive changes made to electrical specifications. | 169 |
| | | 17.5 8-Pin Dual Flat No Lead (DFN) Package (Case #1452) — Added case outline drawing for DFN package. | 177 |
| | | Chapter 17 Ordering Information and Mechanical Specifications — Added ordering information for DFN package. | 185 |
| January, 2003 | 0.2 | 4.2 Features — Corrected third bulleted item. | 49 |

## Revision History (Sheet 2 of 3)

| Date | Revision Level | Description | Page Number(s) |
|---|---|---|---|
| August, 2003 | 1.0 | Reformatted to meet latest M68HC08 documentation standards | N/A |
| | | Figure 1-1. Block Diagram — Diagram redrawn to include keyboard interrupt module and TCLK pin designator. | 20 |
| | | Figure 1-2. MCU Pin Assignments — Added TCLK pin designator. | 21 |
| | | Table 1-2. Pin Functions — Added TCLK pin description. | 22 |
| | | Table 1-3. Function Priority in Shared Pins — Revised table for clarity and to add TCLK. | 23 |
| | | Figure 2-1. Memory Map — Corrected names for the IRQ status and control register (INTSCR) bits 3–0. | 26 |
| | | 3.7.3 ADC Input Clock Register — Clarified bit description for the ADC clock prescaler bits. | 47 |
| | | 4.3 Functional Description — Updated periodic wakeup request values. | 51 |
| | | Figure 6-1. COP Block Diagram — Reworked for clarity | 59 |
| | | Chapter 8 External Interrupt (IRQ) — Corrected bit names for MODE, IRQF, ACK, and IMASK | 77–79 |
| | | Chapter 14 Timer Interface Module (TIM) — Added TCLK function. | 131–139 |
| | | 15.3 Monitor Module (MON) — Updated with additional data. | 147 |
| | | Chapter 16 Electrical Specifications — Updated with additional data. | 169–173 |
| October, 2003 | 2.0 | Figure 2-2. Control, Status, and Data Registers — Deleted unimplemented areas from $FFB0–$FFBD and $FFC2–$FFCF as they are actually available. Also corrected $FFBF designation from unimplemented to reserved. | 27 |
| | | Figure 6-1. COP Block Diagram — Reworked for clarity | 59 |
| | | 6.3.2 STOP Instruction — Added subsection | 60 |
| | | 13.4.2 Active Resets from Internal Sources — Reworked notes for clarity. | 111 |
| | | Table 13-2. Reset Recovery Timing — Replaced previous table with new information. | 112 |
| | | Chapter 14 Timer Interface Module (TIM) — Updated with additional data. | 131 |
| | | Figure 15-3. Break I/O Register Summary — Corrected bit designators for the BRKAR register | 143 |
| | | 15.3 Monitor Module (MON) — Clarified seventh bullet. | 147 |
| | | Table 17-1. MC Order Numbers — Corrected temperature and package designators. | 175 |
| January, 2004 | 3.0 | Figure 2-2. Control, Status, and Data Registers — Corrected reset state for the FLASH Block Protect Register at address location $FFBE and the Internal Oscillator Trim Value at $FFC0. | 32 |
| | | Figure 2-5. FLASH Block Protect Register (FLBPR) — Restated reset state for clarity. | 38 |

# Revision History (Sheet 3 of 3)

| Date | Revision Level | Description | Page Number(s) |
|---|---|---|---|
| November, 2004 | 4.0 | Reformatted to meet current documentation standards | Throughout |
| | | 6.3.1 BUSCLKX4 — Clarified description of BUSCLKX4 | 58 |
| | | Chapter 7 Central Processor Unit (CPU) — In 7.7 Instruction Set Summary:<br>Reworked definitions for STOP instruction<br>Added WAIT instruction | <br>70<br>71 |
| | | 13.8.1 SIM Reset Status Register — Clarified SRSR flag setting | 117 |
| | | 14.9.1 TIM Status and Control Register — Added information to TSTOP note | 127 |
| | | 16.8 5-V Oscillator Characteristics — Added values for deviation from trimmed inernal oscillator | 155 |
| | | 16.12 3-V Oscillator Characteristics — Added values for deviation from trimmed inernal oscillator | 158 |
| July, 2005 | 5.0 | Figure 5-2. Configuration Register 1 (CONFIG1) — Clarified bit definitions for COPRS. | 54 |
| | | Chapter 8 External Interrupt (IRQ) — Reworked for clarification. | 73 |
| | | 11.3.4 RC Oscillator — Improved RC oscillator wording. | 93 |
| | | 12.1 Introduction — Added note pertaining to non-bonded port pins. | 97 |
| | | 17.3 Package Dimensions — Updated package information. | 165 |
| March, 2010 | 6.0 | Clarify internal oscillator trim register information. | 26, 27, 31, 34, 35, 38, 91, 96 |

# List of Chapters


Chapter 1 General Description . . . . . 17

Chapter 2 Memory . . . . . 25

Chapter 3 Analog-to-Digital Converter (ADC) . . . . . 41

Chapter 4 Auto Wakeup Module (AWU) . . . . . 49

Chapter 5 Configuration Register (CONFIG) . . . . . 53

Chapter 6 Computer Operating Properly (COP) . . . . . 57

Chapter 7 Central Processor Unit (CPU) . . . . . 61

Chapter 8 External Interrupt (IRQ) . . . . . 73

Chapter 9 Keyboard Interrupt Module (KBI) . . . . . 79

Chapter 10 Low-Voltage Inhibit (LVI) . . . . . 85

Chapter 11 Oscillator Module (OSC) . . . . . 89

Chapter 12 Input/Output Ports (PORTS) . . . . . 97

Chapter 13 System Integration Module (SIM) . . . . . 103

Chapter 14 Timer Interface Module (TIM) . . . . . 119

Chapter 15 Development Support . . . . . 133

Chapter 16 Electrical Specifications . . . . . 149

Chapter 17 Ordering Information and Mechanical Specifications . . . . . 165

# Table of Contents


**Chapter 1**
**General Description**

1.1 Introduction . . . 17
1.2 Features . . . 17
1.3 MCU Block Diagram . . . 19
1.4 Pin Assignments . . . 19
1.5 Pin Functions . . . 22
1.6 Pin Function Priority . . . 23

**Chapter 2**
**Memory**

2.1 Introduction . . . 25
2.2 Unimplemented Memory Locations . . . 25
2.3 Reserved Memory Locations . . . 25
2.4 Input/Output (I/O) Section . . . 27
2.5 Random-Access Memory (RAM) . . . 32
2.6 FLASH Memory (FLASH) . . . 33
2.6.1 FLASH Control Register . . . 33
2.6.2 FLASH Page Erase Operation . . . 34
2.6.3 FLASH Mass Erase Operation . . . 35
2.6.4 FLASH Program Operation . . . 35
2.6.5 FLASH Protection . . . 36
2.6.6 FLASH Block Protect Register . . . 38
2.6.7 Wait Mode . . . 39
2.6.8 Stop Mode . . . 39

**Chapter 3**
**Analog-to-Digital Converter (ADC)**

3.1 Introduction . . . 41
3.2 Features . . . 41
3.3 Functional Description . . . 41
3.3.1 ADC Port I/O Pins . . . 41
3.3.2 Voltage Conversion . . . 44
3.3.3 Conversion Time . . . 44
3.3.4 Continuous Conversion . . . 44
3.3.5 Accuracy and Precision . . . 44
3.4 Interrupts . . . 44
3.5 Low-Power Modes . . . 44
3.5.1 Wait Mode . . . 44
3.5.2 Stop Mode . . . 45

3.6 Input/Output Signals . . . 45
3.7 Input/Output Registers . . . 45
3.7.1 ADC Status and Control Register . . . 45
3.7.2 ADC Data Register . . . 47
3.7.3 ADC Input Clock Register . . . 47

## Chapter 4
## Auto Wakeup Module (AWU)

4.1 Introduction . . . 49
4.2 Features . . . 49
4.3 Functional Description . . . 49
4.4 Wait Mode . . . 50
4.5 Stop Mode . . . 50
4.6 Input/Output Registers . . . 51
4.6.1 Port A I/O Register . . . 51
4.6.2 Keyboard Status and Control Register . . . 51
4.6.3 Keyboard Interrupt Enable Register . . . 52

## Chapter 5
## Configuration Register (CONFIG)

5.1 Introduction . . . 53
5.2 Functional Description . . . 53

## Chapter 6
## Computer Operating Properly (COP)

6.1 Introduction . . . 57
6.2 Functional Description . . . 57
6.3 I/O Signals . . . 58
6.3.1 BUSCLKX4 . . . 58
6.3.2 STOP Instruction . . . 58
6.3.3 COPCTL Write . . . 58
6.3.4 Power-On Reset . . . 58
6.3.5 Internal Reset . . . 58
6.3.6 COPD (COP Disable) . . . 58
6.3.7 COPRS (COP Rate Select) . . . 59
6.4 COP Control Register . . . 59
6.5 Interrupts . . . 59
6.6 Monitor Mode . . . 59
6.7 Low-Power Modes . . . 59
6.7.1 Wait Mode . . . 59
6.7.2 Stop Mode . . . 59
6.8 COP Module During Break Mode . . . 59

## Chapter 7
## Central Processor Unit (CPU)


7.1 Introduction . . . 61
7.2 Features . . . 61
7.3 CPU Registers . . . 61
7.3.1 Accumulator . . . 62
7.3.2 Index Register . . . 62
7.3.3 Stack Pointer . . . 63
7.3.4 Program Counter . . . 63
7.3.5 Condition Code Register . . . 64
7.4 Arithmetic/Logic Unit (ALU) . . . 65
7.5 Low-Power Modes . . . 65
7.5.1 Wait Mode . . . 65
7.5.2 Stop Mode . . . 65
7.6 CPU During Break Interrupts . . . 65
7.7 Instruction Set Summary . . . 66
7.8 Opcode Map . . . 71


## Chapter 8
## External Interrupt (IRQ)


8.1 Introduction . . . 73
8.2 Features . . . 73
8.3 Functional Description . . . 73
8.3.1 MODE = 1 . . . 75
8.3.2 MODE = 0 . . . 75
8.4 Interrupts . . . 76
8.5 Low-Power Modes . . . 76
8.5.1 Wait Mode . . . 76
8.5.2 Stop Mode . . . 76
8.6 IRQ Module During Break Interrupts . . . 76
8.7 I/O Signals . . . 76
8.7.1 IRQ Input Pins ($\overline{\text{IRQ}}$) . . . 77
8.8 Registers . . . 77


## Chapter 9
## Keyboard Interrupt Module (KBI)


9.1 Introduction . . . 79
9.2 Features . . . 79
9.3 Functional Description . . . 79
9.3.1 Keyboard Operation . . . 79
9.3.2 Keyboard Initialization . . . 82
9.4 Wait Mode . . . 82
9.5 Stop Mode . . . 82
9.6 Keyboard Module During Break Interrupts . . . 82

9.7 Input/Output Registers . . . 83
9.7.1 Keyboard Status and Control Register . . . 83
9.7.2 Keyboard Interrupt Enable Register . . . 84

## Chapter 10
## Low-Voltage Inhibit (LVI)

10.1 Introduction . . . 85
10.2 Features . . . 85
10.3 Functional Description . . . 85
10.3.1 Polled LVI Operation . . . 86
10.3.2 Forced Reset Operation . . . 86
10.3.3 Voltage Hysteresis Protection . . . 86
10.3.4 LVI Trip Selection . . . 86
10.4 LVI Status Register . . . 87
10.5 LVI Interrupts . . . 87
10.6 Low-Power Modes . . . 87
10.6.1 Wait Mode . . . 87
10.6.2 Stop Mode . . . 87

## Chapter 11
## Oscillator Module (OSC)

11.1 Introduction . . . 89
11.2 Features . . . 89
11.3 Functional Description . . . 89
11.3.1 Internal Oscillator . . . 90
11.3.1.1 Internal Oscillator Trimming . . . 91
11.3.1.2 Internal to External Clock Switching . . . 91
11.3.2 External Oscillator . . . 91
11.3.3 XTAL Oscillator . . . 92
11.3.4 RC Oscillator . . . 93
11.4 Oscillator Module Signals . . . 93
11.4.1 Crystal Amplifier Input Pin (OSC1) . . . 93
11.4.2 Crystal Amplifier Output Pin (OSC2/PTA4/BUSCLKX4) . . . 94
11.4.3 Oscillator Enable Signal (SIMOSCEN) . . . 94
11.4.4 XTAL Oscillator Clock (XTALCLK) . . . 94
11.4.5 RC Oscillator Clock (RCCLK) . . . 94
11.4.6 Internal Oscillator Clock (INTCLK) . . . 94
11.4.7 Oscillator Out 2 (BUSCLKX4) . . . 94
11.4.8 Oscillator Out (BUSCLKX2) . . . 94
11.5 Low Power Modes . . . 95
11.5.1 Wait Mode . . . 95
11.5.2 Stop Mode . . . 95
11.6 Oscillator During Break Mode . . . 95
11.7 CONFIG2 Options . . . 95
11.8 Input/Output (I/O) Registers . . . 95
11.8.1 Oscillator Status Register . . . 96
11.8.2 Oscillator Trim Register (OSCTRIM) . . . 96

## Chapter 12
## Input/Output Ports (PORTS)


12.1 Introduction . . . 97
12.2 Port A . . . 97
12.2.1 Port A Data Register . . . 98
12.2.2 Data Direction Register A . . . 98
12.2.3 Port A Input Pullup Enable Register . . . 99
12.3 Port B . . . 100
12.3.1 Port B Data Register . . . 100
12.3.2 Data Direction Register B . . . 101
12.3.3 Port B Input Pullup Enable Register . . . 102


## Chapter 13
## System Integration Module (SIM)


13.1 Introduction . . . 103
13.2 $\overline{\text{RST}}$ and $\overline{\text{IRQ}}$ Pins Initialization . . . 104
13.3 SIM Bus Clock Control and Generation . . . 104
13.3.1 Bus Timing . . . 105
13.3.2 Clock Start-Up from POR . . . 105
13.3.3 Clocks in Stop Mode and Wait Mode . . . 105
13.4 Reset and System Initialization . . . 105
13.4.1 External Pin Reset . . . 106
13.4.2 Active Resets from Internal Sources . . . 106
13.4.2.1 Power-On Reset . . . 107
13.4.2.2 Computer Operating Properly (COP) Reset . . . 108
13.4.2.3 Illegal Opcode Reset . . . 108
13.4.2.4 Illegal Address Reset . . . 108
13.4.2.5 Low-Voltage Inhibit (LVI) Reset . . . 108
13.5 SIM Counter . . . 108
13.5.1 SIM Counter During Power-On Reset . . . 108
13.5.2 SIM Counter During Stop Mode Recovery . . . 109
13.5.3 SIM Counter and Reset States . . . 109
13.6 Exception Control . . . 109
13.6.1 Interrupts . . . 109
13.6.1.1 Hardware Interrupts . . . 111
13.6.1.2 SWI Instruction . . . 112
13.6.2 Interrupt Status Registers . . . 112
13.6.2.1 Interrupt Status Register 1 . . . 113
13.6.2.2 Interrupt Status Register 2 . . . 113
13.6.2.3 Interrupt Status Register 3 . . . 113
13.6.3 Reset . . . 114
13.6.4 Break Interrupts . . . 114
13.6.5 Status Flag Protection in Break Mode . . . 114
13.7 Low-Power Modes . . . 114
13.7.1 Wait Mode . . . 114
13.7.2 Stop Mode . . . 115

13.8 SIM Registers . . . 116
13.8.1 SIM Reset Status Register . . . 117
13.8.2 Break Flag Control Register . . . 118

## Chapter 14
## Timer Interface Module (TIM)

14.1 Introduction . . . 119
14.2 Features . . . 119
14.3 Pin Name Conventions . . . 119
14.4 Functional Description . . . 121
14.4.1 TIM Counter Prescaler . . . 122
14.4.2 Input Capture . . . 122
14.4.3 Output Compare . . . 122
14.4.3.1 Unbuffered Output Compare . . . 122
14.4.3.2 Buffered Output Compare . . . 122
14.4.4 Pulse Width Modulation (PWM) . . . 123
14.4.4.1 Unbuffered PWM Signal Generation . . . 124
14.4.4.2 Buffered PWM Signal Generation . . . 124
14.4.4.3 PWM Initialization . . . 125
14.5 Interrupts . . . 125
14.6 Wait Mode . . . 126
14.7 TIM During Break Interrupts . . . 126
14.8 Input/Output Signals . . . 126
14.8.1 TIM Clock Pin (PTA2/TCLK) . . . 126
14.8.2 TIM Channel I/O Pins (PTA0/TCH0 and PTA1/TCH1) . . . 126
14.9 Input/Output Registers . . . 126
14.9.1 TIM Status and Control Register . . . 127
14.9.2 TIM Counter Registers . . . 128
14.9.3 TIM Counter Modulo Registers . . . 129
14.9.4 TIM Channel Status and Control Registers . . . 129
14.9.5 TIM Channel Registers . . . 132

## Chapter 15
## Development Support

15.1 Introduction . . . 133
15.2 Break Module (BRK) . . . 133
15.2.1 Functional Description . . . 133
15.2.1.1 Flag Protection During Break Interrupts . . . 135
15.2.1.2 TIM During Break Interrupts . . . 135
15.2.1.3 COP During Break Interrupts . . . 135
15.2.2 Break Module Registers . . . 135
15.2.2.1 Break Status and Control Register . . . 136
15.2.2.2 Break Address Registers . . . 136
15.2.2.3 Break Auxiliary Register . . . 137
15.2.2.4 Break Status Register . . . 137
15.2.2.5 Break Flag Control Register . . . 138
15.2.3 Low-Power Modes . . . 138

15.3 Monitor Module (MON) . . . . . 138
15.3.1 Functional Description . . . . . 139
15.3.1.1 Normal Monitor Mode . . . . . 142
15.3.1.2 Forced Monitor Mode . . . . . 143
15.3.1.3 Monitor Vectors . . . . . 143
15.3.1.4 Data Format . . . . . 144
15.3.1.5 Break Signal . . . . . 144
15.3.1.6 Baud Rate . . . . . 144
15.3.1.7 Commands . . . . . 144
15.3.2 Security . . . . . 148

## Chapter 16
## Electrical Specifications

16.1 Introduction . . . . . 149
16.2 Absolute Maximum Ratings . . . . . 149
16.3 Functional Operating Range . . . . . 150
16.4 Thermal Characteristics . . . . . 150
16.5 5-V DC Electrical Characteristics . . . . . 151
16.6 Typical 5-V Output Drive Characteristics . . . . . 152
16.7 5-V Control Timing . . . . . 153
16.8 5-V Oscillator Characteristics . . . . . 154
16.9 3-V DC Electrical Characteristics . . . . . 155
16.10 Typical 3.0-V Output Drive Characteristics . . . . . 156
16.11 3-V Control Timing . . . . . 157
16.12 3-V Oscillator Characteristics . . . . . 158
16.13 Supply Current Characteristics . . . . . 159
16.14 Analog-to-Digital Converter Characteristics . . . . . 161
16.15 Timer Interface Module Characteristics . . . . . 162
16.16 Memory Characteristics . . . . . 163

## Chapter 17
## Ordering Information and Mechanical Specifications

17.1 Introduction . . . . . 165
17.2 MC Order Numbers . . . . . 165
17.3 Package Dimensions . . . . . 165

# Chapter 1
# General Description

## 1.1 Introduction

The MC68HC908QY4 is a member of the low-cost, high-performance M68HC08 Family of 8-bit microcontroller units (MCUs). The M68HC08 Family is a Complex Instruction Set Computer (CISC) with a Von Neumann architecture. 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 Device Variations**

| Device | FLASH Memory Size | Analog-to-Digital Converter | Pin Count |
|---|---|---|---|
| MC68HC908QT1 | 1536 bytes | — | 8 pins |
| MC68HC908QT2 | 1536 bytes | 4 ch, 8 bit | 8 pins |
| MC68HC908QT4 | 4096 bytes | 4 ch, 8 bit | 8 pins |
| MC68HC908QY1 | 1536 bytes | — | 16 pins |
| MC68HC908QY2 | 1536 bytes | 4 ch, 8 bit | 16 pins |
| MC68HC908QY4 | 4096 bytes | 4 ch, 8 bit | 16 pins |

## 1.2 Features

Features include:

- High-performance M68HC08 CPU core
- Fully upward-compatible object code with M68HC05 Family
- 5-V and 3-V operating voltages ($V_{DD}$)
- 8-MHz internal bus operation at 5 V, 4-MHz at 3 V
- Trimmable internal oscillator
  - 3.2 MHz internal bus operation
  - 8-bit trim capability allows 0.4% accuracy(1)
  - ± 25% untrimmed
- Auto wakeup from STOP capability
- Configuration (CONFIG) register for MCU configuration options, including:
  - Low-voltage inhibit (LVI) trip point
- In-system FLASH programming
- FLASH security(2)

1. The oscillator frequency is guaranteed to ±5% over temperature and voltage range after trimming.
2. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

- On-chip in-application programmable FLASH memory (with internal program/erase voltage generation)
  - MC68HC908QY4 and MC68HC908QT4 — 4096 bytes
  - MC68HC908QY2, MC68HC908QY1, MC68HC908QT2, and MC68HC908QT1 — 1536 bytes
- 128 bytes of on-chip random-access memory (RAM)
- 2-channel, 16-bit timer interface module (TIM)
- 4-channel, 8-bit analog-to-digital converter (ADC) on MC68HC908QY2, MC68HC908QY4, MC68HC908QT2, and MC68HC908QT4
- 5 or 13 bidirectional input/output (I/O) lines and one input only:
  - Six shared with keyboard interrupt function and ADC
  - Two shared with timer channels
  - One shared with external interrupt (IRQ)
  - Eight extra I/O lines on 16-pin package only
  - High current sink/source capability on all port pins
  - Selectable pullups on all ports, selectable on an individual bit basis
  - Three-state ability on all port pins
- 6-bit keyboard interrupt with wakeup feature (KBI)
- Low-voltage inhibit (LVI) module features:
  - Software selectable trip point in CONFIG register
- System protection features:
  - Computer operating properly (COP) watchdog
  - Low-voltage detection with reset
  - Illegal opcode detection with reset
  - Illegal address detection with reset
- External asynchronous interrupt pin with internal pullup ($\overline{\text{IRQ}}$) shared with general-purpose input pin
- Master asynchronous reset pin ($\overline{\text{RST}}$) shared with general-purpose input/output (I/O) pin
- Power-on reset
- Internal pullups on $\overline{\text{IRQ}}$ and $\overline{\text{RST}}$ to reduce external components
- Memory mapped I/O registers
- Power saving stop and wait modes
- MC68HC908QY4, MC68HC908QY2, and MC68HC908QY1 are available in these packages:
  - 16-pin plastic dual in-line package (PDIP)
  - 16-pin small outline integrated circuit (SOIC) package
  - 16-pin thin shrink small outline package (TSSOP)
- MC68HC908QT4, MC68HC908QT2, and MC68HC908QT1 are available in these packages:
  - 8-pin PDIP
  - 8-pin SOIC
  - 8-pin dual flat no lead (DFN) package

Features of the CPU08 include the following:

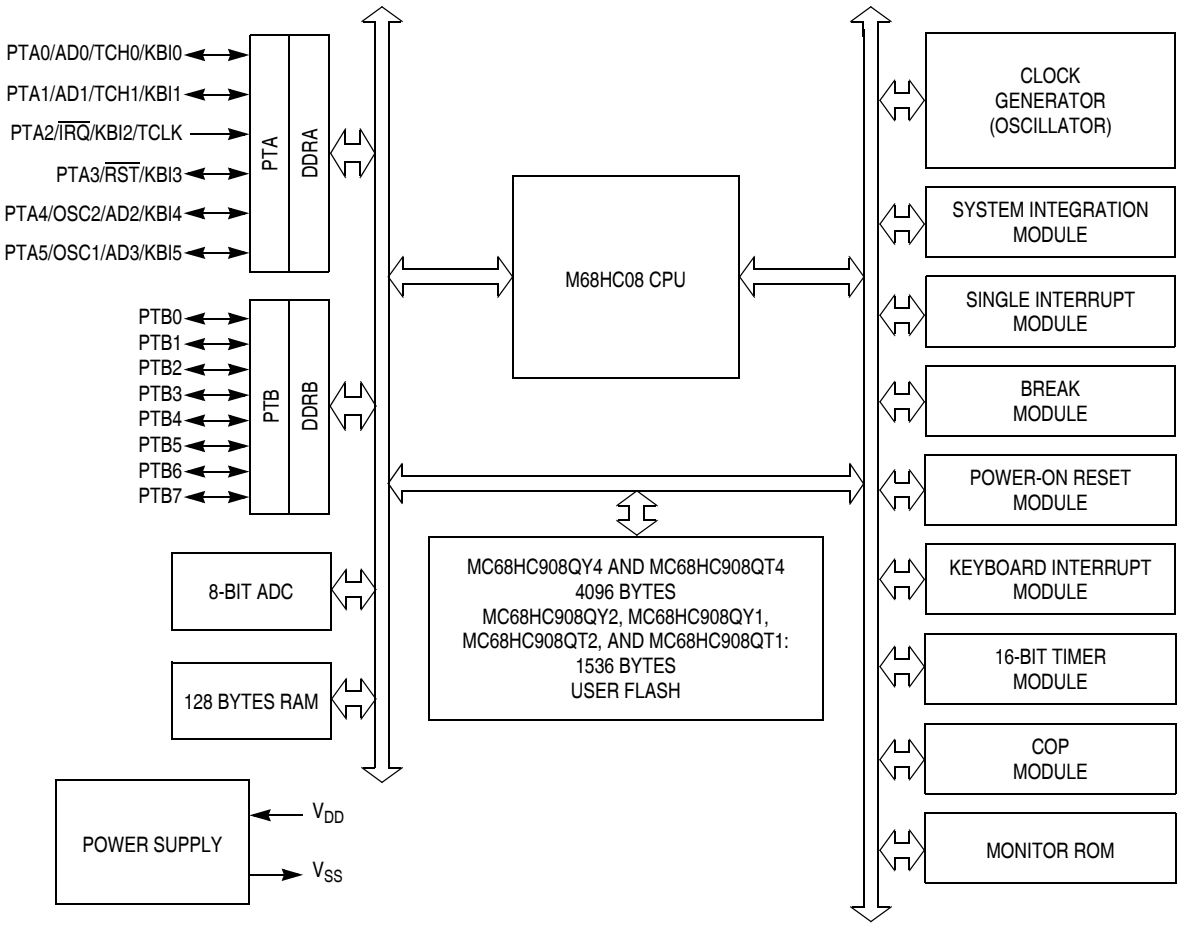
- Enhanced HC05 programming model
- Extensive loop control functions
- 16 addressing modes (eight more than the HC05)
- 16-bit index register and stack pointer
- Memory-to-memory data transfers
- Fast 8 × 8 multiply instruction
- Fast 16/8 divide instruction
- Binary-coded decimal (BCD) instructions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

## 1.3 MCU Block Diagram

Figure 1-1 shows the structure of the MC68HC908QY4.

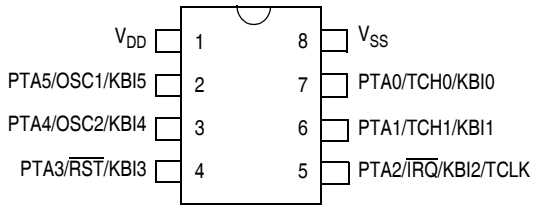
## 1.4 Pin Assignments

The MC68HC908QT4, MC68HC908QT2, and MC68HC908QT1 are available in 8-pin packages and the MC68HC908QY4, MC68HC908QY2, and MC68HC908QY1 in 16-pin packages. Figure 1-2 shows the pin assignment for these packages.

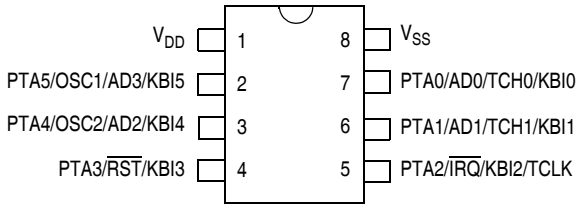



RST, IRQ: Pins have internal (about 30K Ohms) pull up
PTA[0:5]: High current sink and source capability
PTA[0:5]: Pins have programmable keyboard interrupt and pull up
PTB[0:7]: Not available on 8-pin devices – MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 (see note in 12.1 Introduction)
ADC: Not available on the MC68HC908QY1 and MC68HC908QT1

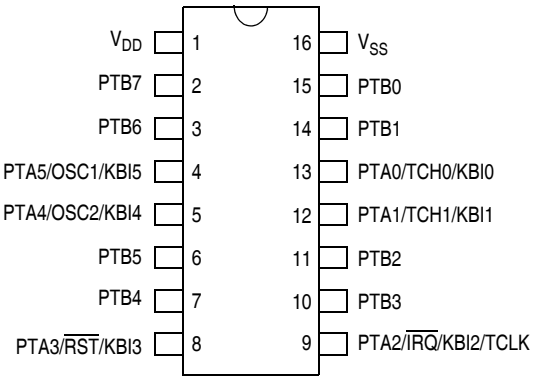
**Figure 1-1. Block Diagram**



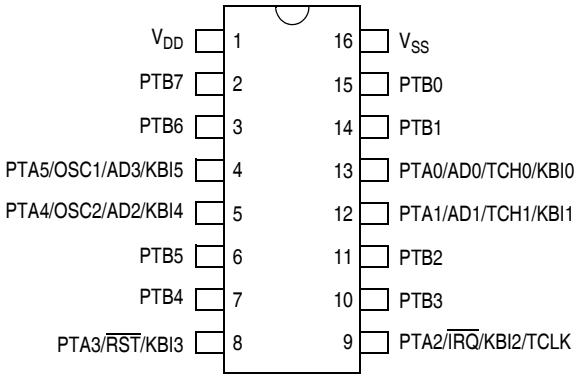

**8-PIN ASSIGNMENT**
**MC68HC908QT1 PDIP/SOIC**



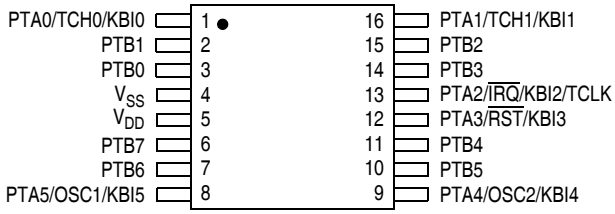

**8-PIN ASSIGNMENT**
**MC68HC908QT2 AND MC68HC908QT4 PDIP/SOIC**



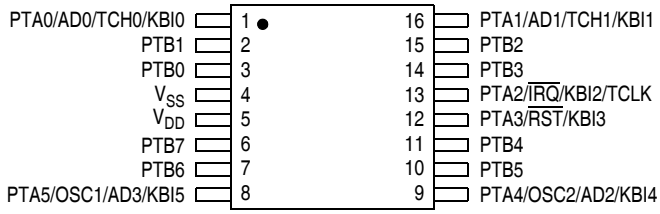

**16-PIN ASSIGNMENT**
**MC68HC908QY1 PDIP/SOIC**



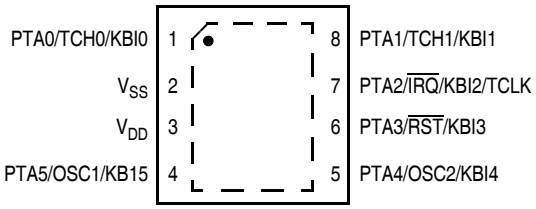

**16-PIN ASSIGNMENT**
**MC68HC908QY2 AND MC68HC908QY4 PDIP/SOIC**



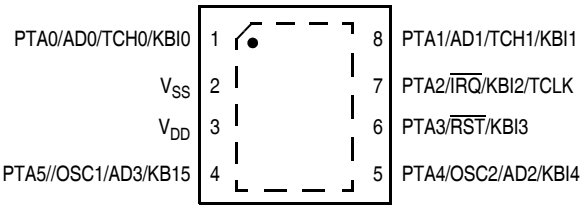

**16-PIN ASSIGNMENT**
**MC68HC908QY1 TSSOP**




**16-PIN ASSIGNMENT**
**MC68HC908QY2 AND MC68HC908QY4 TSSOP**




**8-PIN ASSIGNMENT**
**MC68HC908QT1 DFN**




**8-PIN ASSIGNMENT**
**MC68HC908QT2 AND MC68HC908QT4 DFN**

**Figure 1-2. MCU Pin Assignments**

## 1.5 Pin Functions

Table 1-2 provides a description of the pin functions.

**Table 1-2. Pin Functions**

| Pin Name | Description | Input/Output |
|---|---|---|
| $V_{DD}$ | Power supply | Power |
| $V_{SS}$ | Power supply ground | Power |
| PTA0 | PTA0 — General purpose I/O port | Input/Output |
| | AD0 — A/D channel 0 input | Input |
| | TCH0 — Timer Channel 0 I/O | Input/Output |
| | KBI0 — Keyboard interrupt input 0 | Input |
| PTA1 | PTA1 — General purpose I/O port | Input/Output |
| | AD1 — A/D channel 1 input | Input |
| | TCH1 — Timer Channel 1 I/O | Input/Output |
| | KBI1 — Keyboard interrupt input 1 | Input |
| PTA2 | PTA2 — General purpose input-only port | Input |
| | $\overline{IRQ}$ — External interrupt with programmable pullup and Schmitt trigger input | Input |
| | KBI2 — Keyboard interrupt input 2 | Input |
| | TCLK — Timer clock input | Input |
| PTA3 | PTA3 — General purpose I/O port | Input/Output |
| | $\overline{RST}$ — Reset input, active low with internal pullup and Schmitt trigger | Input |
| | KBI3 — Keyboard interrupt input 3 | Input |
| PTA4 | PTA4 — General purpose I/O port | Input/Output |
| | OSC2 —XTAL oscillator output (XTAL option only)<br>RC or internal oscillator output (OSC2EN = 1 in PTAPUE register) | Output<br>Output |
| | AD2 — A/D channel 2 input | Input |
| | KBI4 — Keyboard interrupt input 4 | Input |
| PTA5 | PTA5 — General purpose I/O port | Input/Output |
| | OSC1 — XTAL, RC, or external oscillator input | Input |
| | AD3 — A/D channel 3 input | Input |
| | KBI5 — Keyboard interrupt input 5 | Input |
| PTB[0:7](1) | 8 general-purpose I/O ports | Input/Output |

1. The PTB pins are not available on the 8-pin packages (see note in 12.1 Introduction).

## 1.6 Pin Function Priority

Table 1-3 is meant to resolve the priority if multiple functions are enabled on a single pin.

***NOTE***

*Upon reset all pins come up as input ports regardless of the priority table.*

**Table 1-3. Function Priority in Shared Pins**

| Pin Name | Highest-to-Lowest Priority Sequence |
|---|---|
| PTA0 | AD0 → TCH0 → KBI0 → PTA0 |
| PTA1 | AD1 →TCH1 → KBI1 → PTA1 |
| PTA2 | $\overline{\text{IRQ}}$ → KBI2 → TCLK → PTA2 |
| PTA3 | $\overline{\text{RST}}$ → KBI3 → PTA3 |
| PTA4 | OSC2 → AD2 → KBI4 → PTA4 |
| PTA5 | OSC1 → AD3 → KBI5 → PTA5 |

# Chapter 2
# Memory

## 2.1 Introduction

The central processor unit (CPU08) can address 64 Kbytes of memory space. The memory map, shown in Figure 2-1, includes:

- 4096 bytes of user FLASH for MC68HC908QT4 and MC68HC908QY4
- 1536 bytes of user FLASH for MC68HC908QT2, MC68HC908QT1, MC68HC908QY2, and MC68HC908QY1
- 128 bytes of random access memory (RAM)
- 48 bytes of user-defined vectors, located in FLASH
- 416 bytes of monitor read-only memory (ROM)
- 1536 bytes of FLASH program and erase routines, located in ROM

## 2.2 Unimplemented Memory Locations

Accessing an unimplemented location can have unpredictable effects on MCU operation. In Figure 2-1 and in register figures in this document, unimplemented locations are shaded.

## 2.3 Reserved Memory Locations

Accessing a reserved location can have unpredictable effects on MCU operation. In Figure 2-1 and in register figures in this document, reserved locations are marked with the word Reserved or with the letter R.

| | |
|---|---|
| $0000 ↓ $003F | I/O REGISTERS 64 BYTES |
| $0040 ↓ $007F | RESERVED(1) 64 BYTES |
| $0080 ↓ $00FF | RAM 128 BYTES |
| $0100 ↓ $27FF | UNIMPLEMENTED(1) 9984 BYTES |
| $2800 ↓ $2DFF | AUXILIARY ROM 1536 BYTES |
| $2E00 ↓ $EDFF | UNIMPLEMENTED(1) 49152 BYTES |
| $EE00 ↓ $FDFF | FLASH MEMORY MC68HC908QT4 AND MC68HC908QY4 4096 BYTES |
| $FE00 | BREAK STATUS REGISTER (BSR) |
| $FE01 | RESET STATUS REGISTER (SRSR) |
| $FE02 | BREAK AUXILIARY REGISTER (BRKAR) |
| $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 FOR FLASH TEST CONTROL REGISTER (FLTCR) |
| $FE08 | FLASH CONTROL REGISTER (FLCR) |
| $FE09 | BREAK ADDRESS HIGH REGISTER (BRKH) |
| $FE0A | BREAK ADDRESS LOW REGISTER (BRKL) |
| $FE0B | BREAK STATUS AND CONTROL REGISTER (BRKSCR) |
| $FE0C | LVISR |
| $FE0D ↓ $FE0F | RESERVED FOR FLASH TEST 3 BYTES |
| $FE10 ↓ $FFAF | MONITOR ROM 416 BYTES |
| $FFB0 ↓ $FFBD | FLASH 14 BYTES |
| $FFBE | FLASH BLOCK PROTECT REGISTER (FLBPR) |
| $FFBF | RESERVED FLASH |
| $FFC0 | INTERNAL OSCILLATOR TRIM VALUE (VDD = 5.0 V) |
| $FFC1 | INTERNAL OSCILLATOR TRIM VALUE (VDD = 3.0 V) |
| $FFC2 ↓ $FFCF | FLASH 14 BYTES |
| $FFD0 ↓ $FFFF | USER VECTORS 48 BYTES |

Note 1.
Attempts to execute code from addresses in this range will generate an illegal address reset.

| | |
|---|---|
| $2E00 ↓ $F7FF | UNIMPLEMENTED 51712 BYTES |
| $F800 ↓ $FDFF | FLASH MEMORY 1536 BYTES |

**MC68HC908QT1, MC68HC908QT2, MC68HC908QY1, and MC68HC908QY2 Memory Map**

**Figure 2-1. Memory Map**

## 2.4 Input/Output (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 these addresses:

- $FE00 — Break status register, BSR
- $FE01 — Reset status register, SRSR
- $FE02 — Break auxiliary register, BRKAR
- $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 — Break address register high, BRKH
- $FE0A — Break address register low, BRKL
- $FE0B — Break status and control register, BRKSCR
- $FE0C — LVI status register, LVISR
- $FE0D — Reserved
- $FFBE — FLASH block protect register, FLBPR
- $FFC0 — Internal OSC trim value (factory programmed, VDD = 5.0 V)
- $FFC1 — Internal OSC trim value (factory programmed, VDD = 3.0 V)
- $FFFF — COP control register, COPCTL

| Addr. | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| $0000 | Port A Data Register (PTA) See page 98. | Read: | R | AWUL | PTA5 | PTA4 | PTA3 | PTA2 | PTA1 | PTA0 |
| | | Write: | | | | | | | | |
| | | Reset: | Unaffected by reset | | | | | | | |
| $0001 | Port B Data Register (PTB) See page 100. | Read: | PTB7 | PTB6 | PTB5 | PTB4 | PTB3 | PTB2 | PTB1 | PTB0 |
| | | Write: | | | | | | | | |
| | | Reset: | Unaffected by reset | | | | | | | |
| $0002 | Unimplemented | | | | | | | | | |
| $0003 | Unimplemented | | | | | | | | | |
| $0004 | Data Direction Register A (DDRA) See page 98. | Read: | R | R | DDRA5 | DDRA4 | DDRA3 | 0 | DDRA1 | DDRA0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $0005 | Data Direction Register B (DDRB) See page 101. | Read: | DDRB7 | DDRB6 | DDRB5 | DDRB4 | DDRB3 | DDRB2 | DDRB1 | DDRB0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented R = Reserved U = Unaffected

**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 |
|---|---|---|---|---|---|---|---|---|---|---|
| $0006 ↓ $000A | Unimplemented | | | | | | | | | |
| $000B | Port A Input Pullup Enable Register (PTAPUE) See page 99. | Read: | OSC2EN | 0 | PTAPUE5 | PTAPUE4 | PTAPUE3 | PTAPUE2 | PTAPUE1 | PTAPUE0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $000C | Port B Input Pullup Enable Register (PTBPUE) See page 102. | Read: | PTBPUE7 | PTBPUE6 | PTBPUE5 | PTBPUE4 | PTBPUE3 | PTBPUE2 | PTBPUE1 | PTBPUE0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $000D ↓ $0019 | Unimplemented | | | | | | | | | |
| $001A | Keyboard Status and Control Register (KBSCR) See page 83. | 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) See page 84. | Read: | 0 | AWUIE | KBIE5 | KBIE4 | KBIE3 | KBIE2 | KBIE1 | KBIE0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $001C | Unimplemented | | | | | | | | | |
| $001D | IRQ Status and Control Register (INTSCR) See page 77. | 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)(1) See page 53. | Read: | IRQPUD | IRQEN | R | OSCOPT1 | OSCOPT0 | R | R | RSTEN |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0(2) |

1. One-time writable register after each reset.
2. RSTEN reset to 0 by a power-on reset (POR) only.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $001F | Configuration Register 1 (CONFIG1)(1) See page 54. | Read: | COPRS | LVISTOP | LVIRSTD | LVIPWRD | LVI5OR3 | SSREC | STOP | COPD |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0(2) | 0 | 0 | 0 |

1. One-time writable register after each reset.
2. LVI5OR3 reset to 0 by a power-on reset (POR) only.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $0020 | TIM Status and Control Register (TSC) See page 127. | Read: | TOF | TOIE | TSTOP | 0 | 0 | PS2 | PS1 | PS0 |
| | | Write: | 0 | | | TRST | | | | |
| | | Reset: | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $0021 | TIM Counter Register High (TCNTH) See page 128. | Read: | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented R = Reserved U = Unaffected

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

| Addr. | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| $0022 | TIM Counter Register Low (TCNTL) See page 128. | Read: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $0023 | TIM Counter Modulo Register High (TMODH) See page 129. | Read: | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | Write: | | | | | | | | |
| | | Reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $0024 | TIM Counter Modulo Register Low (TMODL) See page 129. | Read: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | Write: | | | | | | | | |
| | | Reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $0025 | TIM Channel 0 Status and Control Register (TSC0) See page 130. | Read: | CH0F | CH0IE | MS0B | MS0A | ELS0B | ELS0A | TOV0 | CH0MAX |
| | | Write: | 0 | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $0026 | TIM Channel 0 Register High (TCH0H) See page 132. | Read: | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | Write: | | | | | | | | |
| | | Reset: | Indeterminate after reset | | | | | | | |
| $0027 | TIM Channel 0 Register Low (TCH0L) See page 132. | Read: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | Write: | | | | | | | | |
| | | Reset: | Indeterminate after reset | | | | | | | |
| $0028 | TIM Channel 1 Status and Control Register (TSC1) See page 130. | Read: | CH1F | CH1IE | 0 | MS1A | ELS1B | ELS1A | TOV1 | CH1MAX |
| | | Write: | 0 | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $0029 | TIM Channel 1 Register High (TCH1H) See page 132. | Read: | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | Write: | | | | | | | | |
| | | Reset: | Indeterminate after reset | | | | | | | |
| $002A | TIM Channel 1 Register Low (TCH1L) See page 132. | Read: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | Write: | | | | | | | | |
| | | Reset: | Indeterminate after reset | | | | | | | |
| $002B ↓ $0035 | Unimplemented | | | | | | | | | |
| $0036 | Oscillator Status Register (OSCSTAT) See page 96. | Read: | R | R | R | R | R | R | ECGON | ECGST |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $0037 | Unimplemented | Read: | | | | | | | | |
| $0038 | Oscillator Trim Register (OSCTRIM) See page 96. | Read: | TRIM7 | TRIM6 | TRIM5 | TRIM4 | TRIM3 | TRIM2 | TRIM1 | TRIM0 |
| | | Write: | | | | | | | | |
| | | Reset: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented R = Reserved U = Unaffected

**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 |
|---|---|---|---|---|---|---|---|---|---|---|
| $0039 ↓ $003B | Unimplemented | | | | | | | | | |
| $003C | ADC Status and Control Register (ADSCR) See page 45. | Read: | COCO | AIEN | ADCO | CH4 | CH3 | CH2 | CH1 | CH0 |
| | | Write: | R | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| $003D | Unimplemented | | | | | | | | | |
| $003E | ADC Data Register (ADR) See page 47. | Read: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | Write: | | | | | | | | |
| | | Reset: | Indeterminate after reset | | | | | | | |
| $003F | ADC Input Clock Register (ADICLK) See page 47. | Read: | ADIV2 | ADIV1 | ADIV0 | 0 | 0 | 0 | 0 | 0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE00 | Break Status Register (BSR) See page 137. | Read: | R | R | R | R | R | R | SBSW | R |
| | | Write: | | | | | | | See note 1 | |
| | | Reset: | | | | | | | 0 | |
| $FE01 | SIM Reset Status Register (SRSR) See page 117. | Read: | POR | PIN | COP | ILOP | ILAD | MODRST | LVI | 0 |
| | | Write: | | | | | | | | |
| | | POR: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE02 | Break Auxiliary Register (BRKAR) See page 137. | Read: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | BDCOP |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE03 | Break Flag Control Register (BFCR) See page 138. | Read: | BCFE | R | R | R | R | R | R | R |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | | | | | | | |
| $FE04 | Interrupt Status Register 1 (INT1) See page 77. | Read: | 0 | 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) See page 77. | Read: | IF14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | Write: | R | R | R | R | R | R | R | R |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE06 | Interrupt Status Register 3 (INT3) See page 77. | 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 | | R | R | R | R | R | R | R | R |

1. Writing a 0 clears SBSW.

= Unimplemented  R = Reserved  U = Unaffected

**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 |
|---|---|---|---|---|---|---|---|---|---|---|
| $FE08 | FLASH Control Register (FLCR) See page 34. | Read: | 0 | 0 | 0 | 0 | HVEN | MASS | ERASE | PGM |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE09 | Break Address High Register (BRKH) See page 136. | Read: | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE0A | Break Address low Register (BRKL) See page 136. | Read: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE0B | Break Status and Control Register (BRKSCR) See page 136. | Read: | BRKE | BRKA | 0 | 0 | 0 | 0 | 0 | 0 |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE0C | LVI Status Register (LVISR) See page 87. | Read: | LVIOUT | 0 | 0 | 0 | 0 | 0 | 0 | R |
| | | Write: | | | | | | | | |
| | | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $FE0D ↓ $FE0F | Reserved for FLASH Test | | R | R | R | R | R | R | R | R |
| $FFBE | FLASH Block Protect Register (FLBPR) See page 39. | Read: | BPR7 | BPR6 | BPR5 | BPR4 | BPR3 | BPR2 | BPR1 | BPR0 |
| | | Write: | | | | | | | | |
| | | Reset: | Unaffected by reset | | | | | | | |
| $FFBF | Reserved | | R | R | R | R | R | R | R | R |
| $FFC0 | Internal Oscillator Trim (Factory Programmed, VDD = 5.0 V) | Read: | TRIM7 | TRIM6 | TRIM5 | TRIM4 | TRIM3 | TRIM2 | TRIM1 | TRIM0 |
| | | Write: | | | | | | | | |
| | | Reset: | Unaffected by reset | | | | | | | |
| $FFC1 | Internal Oscillator Trim (Factory Programmed, VDD = 3.0 V) | Read: | TRIM7 | TRIM6 | TRIM5 | TRIM4 | TRIM3 | TRIM2 | TRIM1 | TRIM0 |
| | | Write: | | | | | | | | |
| | | Reset: | Unaffected by reset | | | | | | | |
| $FFFF | COP Control Register (COPCTL) See page 59. | Read: | LOW BYTE OF RESET VECTOR | | | | | | | |
| | | Write: | WRITING CLEARS COP COUNTER (ANY VALUE) | | | | | | | |
| | | Reset: | Unaffected by reset | | | | | | | |

= Unimplemented  R = Reserved  U = Unaffected

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

**Table 2-1. Vector Addresses**

| Vector Priority | Vector | Address | Vector |
|---|---|---|---|
| Lowest ↑ | IF15 | $FFDE | ADC conversion complete vector (high) |
| | | $FFDF | ADC conversion complete vector (low) |
| | IF14 | $FFE0 | Keyboard vector (high) |
| | | $FFE1 | Keyboard vector (low) |
| | IF13 ↓ IF6 | — | Not used |
| | IF5 | $FFF2 | TIM overflow vector (high) |
| | | $FFF3 | TIM overflow vector (low) |
| | IF4 | $FFF4 | TIM Channel 1 vector (high) |
| | | $FFF5 | TIM Channel 1 vector (low) |
| | IF3 | $FFF6 | TIM Channel 0 vector (high) |
| | | $FFF7 | TIM 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) |

## 2.5 Random-Access Memory (RAM)

Addresses $0080–$00FF 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.*

Before processing an interrupt, the central processor unit (CPU) uses five bytes of the stack to save the contents of the CPU registers.

***NOTE***

*For M6805, M146805, and M68HC05 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.*

## 2.6 FLASH Memory (FLASH)

This subsection 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.

The FLASH memory consists of an array of 4096 or 1536 bytes with an additional 48 bytes for user vectors. The minimum size of FLASH memory that can be erased is 64 bytes; and the maximum size of FLASH memory that can be programmed in a program cycle is 32 bytes (a row). Program and erase operations are facilitated through control bits in the FLASH control register (FLCR). Details for these operations appear later in this section. The address ranges for the user memory and vectors are:

- $EE00 – $FDFF; user memory, 4096 bytes: MC68HC908QY4 and MC68HC908QT4
- $F800 – $FDFF; user memory, 1536 bytes: MC68HC908QY2, MC68HC908QT2, MC68HC908QY1 and MC68HC908QT1
- $FFD0 – $FFFF; user interrupt vectors, 48 bytes.

***NOTE***

*An erased bit reads as a 1 and a programmed bit reads as a 0.*
*A security feature prevents viewing of the FLASH contents.*[1]

### 2.6.1 FLASH Control Register

The FLASH control register (FLCR) 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 |

= Unimplemented

**Figure 2-3. FLASH Control Register (FLCR)**

**HVEN — High Voltage Enable Bit**

This read/write bit enables high voltage from the charge pump to the memory for either program or erase operation. It 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.

1 = Mass erase operation selected
0 = Mass erase operation unselected

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

**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 unselected

**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 unselected

### 2.6.2 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 48-byte user interrupt vectors area also forms a page. Any FLASH memory page can be erased alone.

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 location within the address range of the block to be erased.
4. Wait for a time, $t_{NVS}$ (minimum 10 μs).
5. Set the HVEN bit.
6. Wait for a time, $t_{Erase}$ (minimum 1 ms or 4 ms).
7. Clear the ERASE bit.
8. Wait for a time, $t_{NVH}$ (minimum 5 μs).
9. Clear the HVEN bit.
10. After time, $t_{RCV}$ (typical 1 μs), 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.*

*CAUTION*

*A page erase of the vector page will erase the internal oscillator trim values at $FFC0 and $FFC1.*

In applications that require more than 1000 program/erase cycles, use the 4 ms page erase specification to get improved long-term reliability. Any application can use this 4 ms page erase specification. However, in applications where a FLASH location will be erased and reprogrammed less than 1000 times, and speed is important, use the 1 ms page erase specification to get a shorter cycle time.

### 2.6.3 FLASH Mass Erase Operation

Use the following procedure to erase the entire FLASH memory to read as a 1:

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 address(1) within the FLASH memory address range.
4. Wait for a time, $t_{NVS}$ (minimum 10 μs).
5. Set the HVEN bit.
6. Wait for a time, $t_{MErase}$ (minimum 4 ms).
7. Clear the ERASE and MASS bits.

**NOTE**

*Mass erase is disabled whenever any block is protected (FLBPR does not equal $FF).*

8. Wait for a time, $t_{NVHL}$ (minimum 100 μs).
9. Clear the HVEN bit.
10. After time, $t_{RCV}$ (typical 1 μs), 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.*

*CAUTION*

*A mass erase will erase the internal oscillator trim values at $FFC0 and $FFC1.*

### 2.6.4 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 the following step-by-step procedure to program a row of FLASH memory

Figure 2-4 shows a flowchart of the programming algorithm.

**NOTE**

*Only bytes which are currently $FF may be programmed.*

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 desired.
4. Wait for a time, $t_{NVS}$ (minimum 10 μs).
5. Set the HVEN bit.
6. Wait for a time, $t_{PGS}$ (minimum 5 μs).
7. Write data to the FLASH address being programmed(2).

1. When in monitor mode, with security sequence failed (see 15.3.2 Security), write to the FLASH block protect register instead of any FLASH address.

8. Wait for time, $t_{PROG}$ (minimum 30 μs).
9. Repeat step 7 and 8 until all desired bytes within the row are programmed.
10. Clear the PGM bit(1).
11. Wait for time, $t_{NVH}$ (minimum 5 μs).
12. Clear the HVEN bit.
13. After time, $t_{RCV}$ (typical 1 μs), the memory can be accessed in read mode again.

***NOTE***

*The COP register at location $FFFF should not be written between steps 5–12, when the HVEN bit is set. Since this register is located at a valid FLASH address, unpredictable behavior may occur if this location is written while HVEN is set.*

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

***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. Do not exceed $t_{PROG}$ maximum, see 16.16 Memory Characteristics.*

### 2.6.5 FLASH 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***

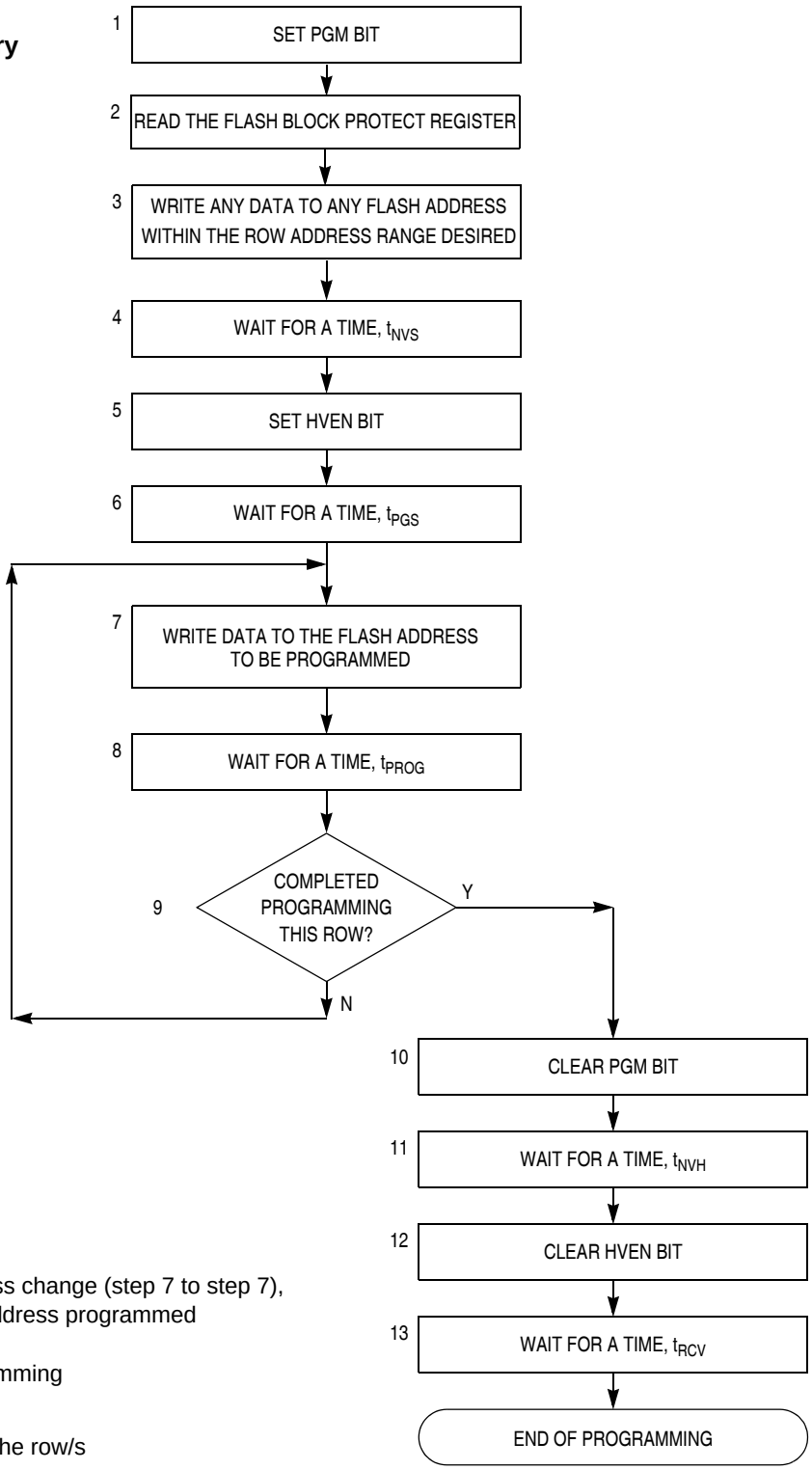
*In performing a program or erase operation, the FLASH block protect register must be read after setting the PGM or ERASE bit and before asserting the HVEN bit.*

When the FLBPR is programmed 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. The address ranges are shown in 2.6.6 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. Mass erase is disabled whenever any block is protected (FLBPR does not equal $FF). 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. The time between each FLASH address change, or the time between the last FLASH address programmed to clearing PGM bit, must not exceed the maximum programming time, $t_{PROG}$ maximum.

### Algorithm for Programming a Row (32 Bytes) of FLASH Memory




NOTES:

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**

## 2.6.6 FLASH Block Protect Register

The FLASH block protect register 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 address of the protected range within the FLASH memory.

Address: $FFBE

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read:<br>Write: | BPR7 | BPR6 | BPR5 | BPR4 | BPR3 | BPR2 | BPR1 | BPR0 |
| Reset: | Unaffected by reset. Initial value from factory is 1. | | | | | | | |

Write to this register is by a programming sequence to the FLASH memory.

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

**BPR[7:0] — FLASH Protection Register Bits [7:0]**

These eight bits in FLBPR represent bits [13:6] of a 16-bit memory address. Bits [15:14] are 1s and bits [5:0] are 0s.

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 within the FLASH memory. See Figure 2-6 and Table 2-2.

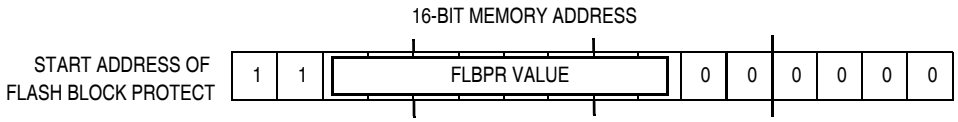



**Figure 2-6. FLASH Block Protect Start Address**

**Table 2-2. Examples of Protect Start Address**

| BPR[7:0] | Start of Address of Protect Range |
|---|---|
| $00–$B8 | The entire FLASH memory is protected. |
| $B9 (**1011 1001**) | $EE40 (11**10 1110 01**00 0000) |
| $BA (**1011 1010**) | $EE80 (11**10 1110 10**00 0000) |
| $BB (**1011 1011**) | $EEC0 (11**10 1110 11**00 0000) |
| $BC (**1011 1100**) | $EF00 (11**10 1111 00**00 0000) |
| and so on... | |
| $DE (**1101 1110**) | $F780 (11**11 0111 10**00 0000) |
| $DF (**1101 1111**) | $F7C0 (11**11 0111 11**00 0000) |
| $FE (**1111 1110**) | $FF80 (11**11 1111 10**00 0000)<br>FLBPR, internal oscillator trim values, and vectors are protected |
| $FF | The entire FLASH memory is not protected. |

### 2.6.7 Wait Mode

Putting the MCU into wait mode while the FLASH is in read mode does not affect the operation of the FLASH memory directly, but there will not be any memory activity since the CPU is inactive.

The WAIT instruction should not be executed while performing a program or erase operation on the FLASH, or the operation will discontinue and the FLASH will be on standby mode.

### 2.6.8 Stop Mode

Putting the MCU into stop mode while the FLASH is in read mode does not affect the operation of the FLASH memory directly, but there will not be any memory activity since the CPU is inactive.

The STOP instruction should not be executed while performing a program or erase operation on the FLASH, or the operation will discontinue and the FLASH will be on standby mode

***NOTE***

*Standby mode is the power-saving mode of the FLASH module in which all internal control signals to the FLASH are inactive and the current consumption of the FLASH is at a minimum.*

# Chapter 3
# Analog-to-Digital Converter (ADC)

## 3.1 Introduction

This section describes the analog-to-digital converter (ADC). The ADC is an 8-bit, 4-channel analog-to-digital converter. The ADC module is only available on the MC68HC908QY2, MC68HC908QT2, MC68HC908QY4, and MC68HC908QT4.

## 3.2 Features

Features of the ADC module include:

- 4 channels with multiplexed input
- Linear successive approximation with monotonicity
- 8-bit resolution
- Single or continuous conversion
- Conversion complete flag or conversion complete interrupt
- Selectable ADC clock frequency

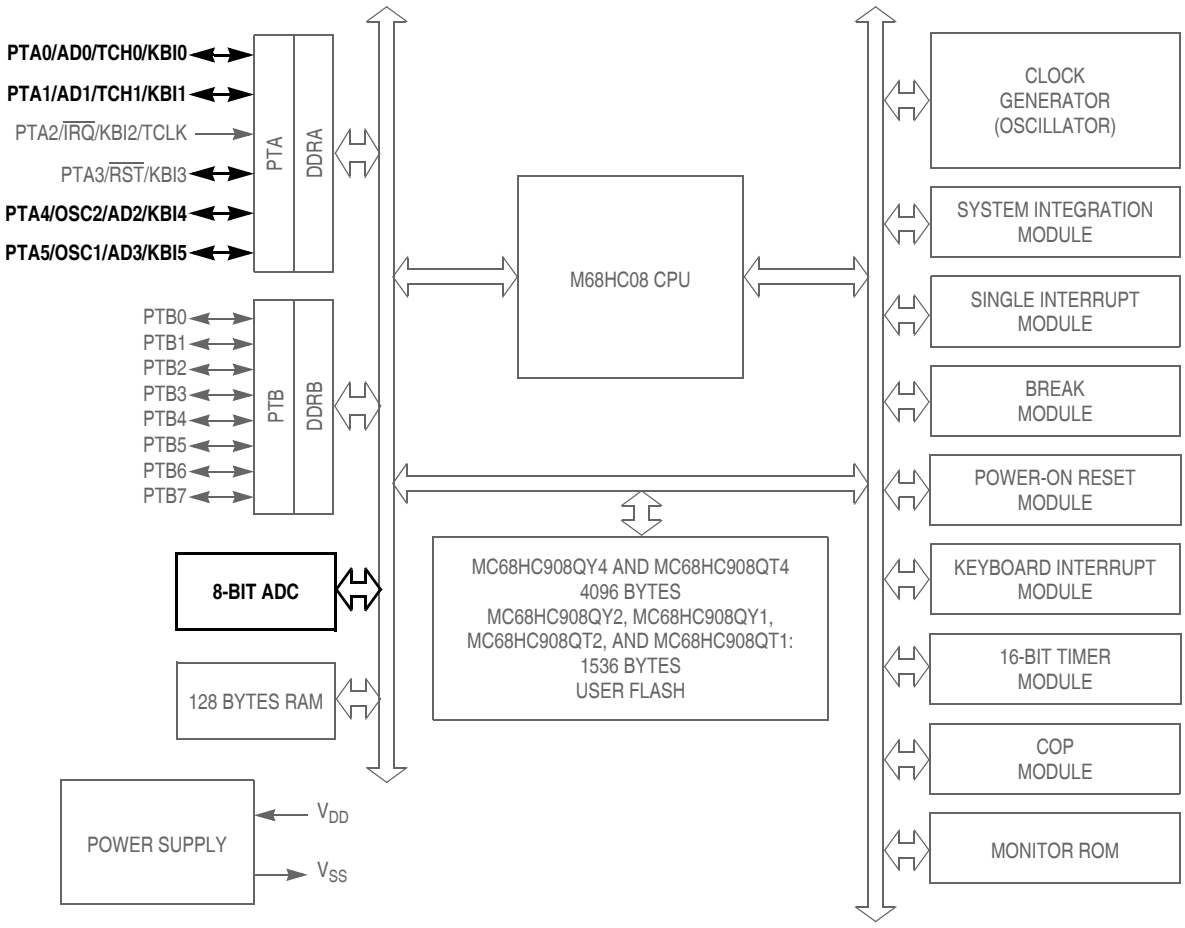
## 3.3 Functional Description

Four ADC channels are available for sampling external sources at pins PTA0, PTA1, PTA4, and PTA5. An analog multiplexer allows the single ADC converter to select one of the four ADC channels as an ADC voltage input (ADCVIN). ADCVIN is converted by the successive approximation register-based counters. The ADC resolution is eight bits. When the conversion is completed, ADC puts the result in the ADC data register and sets a flag or generates an interrupt.

Figure 3-2 shows a block diagram of the ADC.

### 3.3.1 ADC Port I/O Pins

PTA0, PTA1, PTA4, and PTA5 are general-purpose I/O pins that are shared with the ADC channels. The channel select bits (ADC status and control register (ADSCR), $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 data direction register (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 0 if the corresponding DDR bit is at 0. If the DDR bit is at 1, the value in the port data latch is read.




RST, IRQ: Pins have internal (about 30K Ohms) pull up
**PTA[0:5]: High current sink and source capability**
**PTA[0:5]: Pins have programmable keyboard interrupt and pull up**
PTB[0:7]: Not available on 8-pin devices – MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 (see note in 12.1 Introduction)
**ADC: Not available on the MC68HC908QY1 and MC68HC908QT1**

**Figure 3-1. Block Diagram Highlighting ADC Block and Pins**

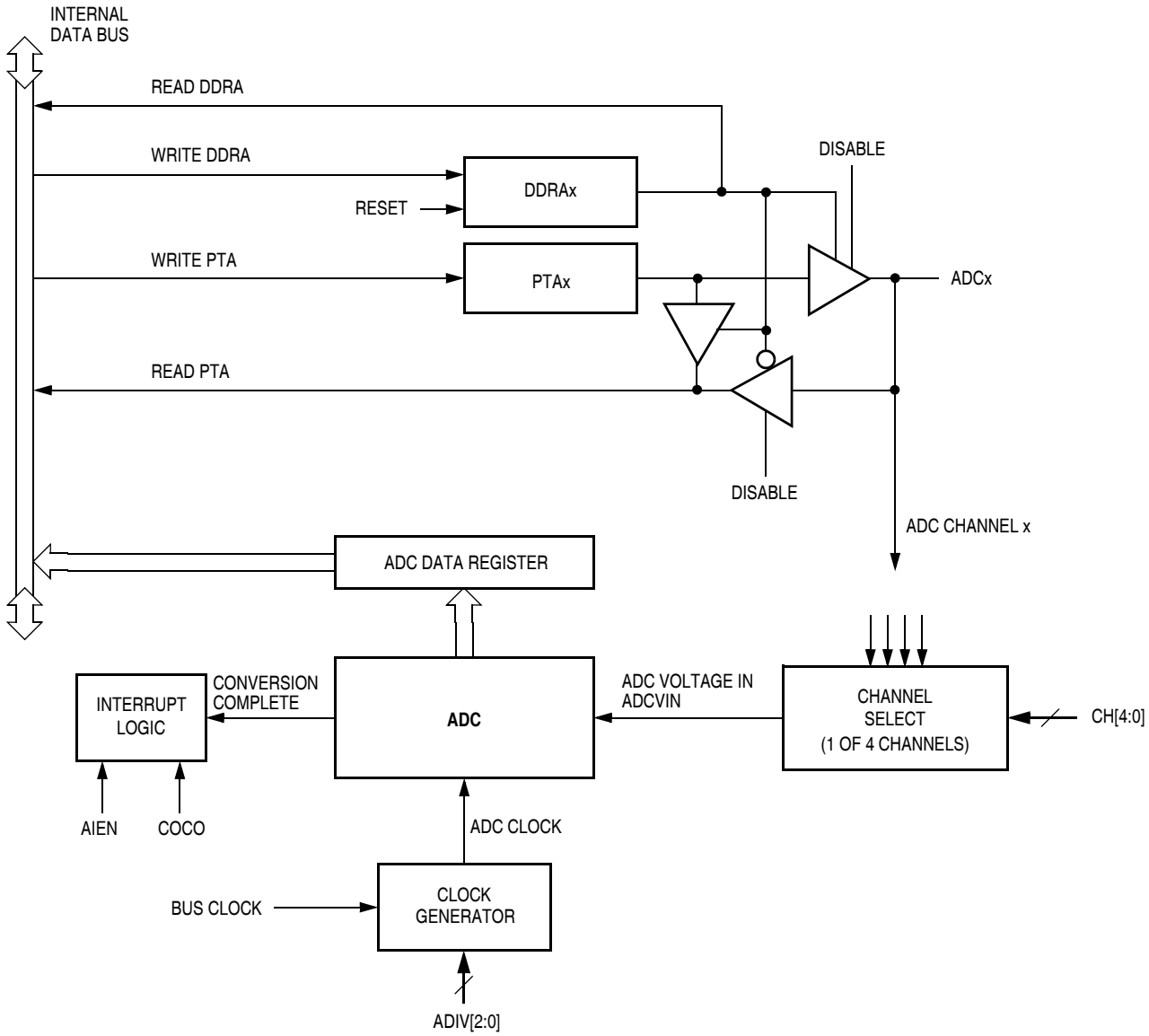



**Figure 3-2. ADC Block Diagram**

### 3.3.2 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.*

### 3.3.3 Conversion Time

Sixteen 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 1 MHz, then one conversion will take 16 μs to complete. With a 1-MHz ADC internal clock the maximum sample rate is 62.5 kHz.

$$\text{Conversion Time} = \frac{\text{16 ADC Clock Cycles}}{\text{ADC Clock Frequency}}$$

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

### 3.3.4 Continuous Conversion

In the continuous conversion mode (ADCO = 1), the ADC continuously converts the selected channel filling the ADC data register (ADR) 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 (ADSCR, $003C) is set after each conversion and will stay set until the next read of the ADC data register.

When a conversion is in process and the ADSCR is written, the current conversion data should be discarded to prevent an incorrect reading.

### 3.3.5 Accuracy and Precision

The conversion process is monotonic and has no missing codes.

## 3.4 Interrupts

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

## 3.5 Low-Power Modes

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

### 3.5.1 Wait Mode

The ADC continues normal operation during wait mode. Any enabled CPU interrupt request from the ADC can bring the microcontroller unit (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 CH[4:0] bits in ADSCR to 1s before executing the WAIT instruction.

### 3.5.2 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 using ADC data after exiting stop mode.

## 3.6 Input/Output Signals

The ADC module has four channels that are shared with I/O port A.

ADC voltage in (ADCVIN) is the input voltage signal from one of the four ADC channels to the ADC module.

## 3.7 Input/Output Registers

These I/O registers control and monitor ADC operation:

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

### 3.7.1 ADC Status and Control Register

The following paragraphs describe the function of the ADC status and control register (ADSCR). When a conversion is in process and the ADSCR is written, the current conversion data should be discarded to prevent an incorrect reading.

Address: $003C

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: | COCO | AIEN | ADCO | CH4 | CH3 | CH2 | CH1 | CH0 |
| Write: | R | | | | | | | |
| Reset: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

R = Reserved

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

**COCO — Conversions Complete Bit**

In non-interrupt mode (AIEN = 0), COCO is a read-only bit that is set at the end of each conversion. COCO will stay set until cleared by a read of the ADC data register. Reset clears this bit.

In interrupt mode (AIEN = 1), COCO is a read-only bit that is not set at the end of a conversion. It always reads as a 0.

1 = Conversion completed (AIEN = 0)
0 = Conversion not completed (AIEN = 0) or CPU interrupt enabled (AIEN = 1)

***NOTE***

*The write function of the COCO bit is reserved. When writing to the ADSCR register, always have a 0 in the COCO bit position.*

### 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 ADR is read or ADSCR 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 ADR 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

### CH[4:0] — ADC Channel Select Bits

CH4, CH3, CH2, CH1, and CH0 form a 5-bit field which is used to select one of the four ADC channels. The five select bits are detailed in Table 3-1. 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. The ADC subsystem is turned off when the channel select bits are all set to 1. This feature allows for reduced power consumption for the MCU when the ADC is not used. Reset sets all of these bits to 1.

***NOTE***

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

**Table 3-1. MUX Channel Select**

| CH4 | CH3 | CH2 | CH1 | CH0 | ADC Channel | Input Select |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | ADC0 | PTA0 |
| 0 | 0 | 0 | 0 | 1 | ADC1 | PTA1 |
| 0 | 0 | 0 | 1 | 0 | ADC2 | PTA4 |
| 0 | 0 | 0 | 1 | 1 | ADC3 | PTA5 |
| 0 | 0 | 1 | 0 | 0 | — | Unused(1) |
| ↓ | ↓ | ↓ | ↓ | ↓ | — | |
| 1 | 1 | 0 | 1 | 0 | — | |
| 1 | 1 | 0 | 1 | 1 | — | Reserved |
| 1 | 1 | 1 | 0 | 0 | — | Unused |
| 1 | 1 | 1 | 0 | 1 | — | $V_{DDA}$(2) |
| 1 | 1 | 1 | 1 | 0 | — | $V_{SSA}$(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.

### 3.7.2 ADC Data Register

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

Address: $003E

| | 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 3-4. ADC Data Register (ADR)**

### 3.7.3 ADC Input Clock Register

This register selects the clock frequency for the ADC.

Address: $003F

| | 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 3-5. ADC Input Clock Register (ADICLK)**

**ADIV2–ADIV0 — ADC Clock Prescaler Bits**

ADIV2, ADIV1, and ADIV0 form a 3-bit field which selects the divide ratio used by the ADC to generate the internal ADC clock. Table 3-2 shows the available clock configurations. The ADC clock frequency should be set between $f_{ADIC(MIN)}$ and $f_{ADIC(MAX)}$. The analog input level should remain stable for the entire conversion time (maximum = 17 ADC clock cycles).

**Table 3-2. 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 4
# Auto Wakeup Module (AWU)

## 4.1 Introduction

This section describes the auto wakeup module (AWU). The AWU generates a periodic interrupt during stop mode to wake the part up without requiring an external signal. Figure 4-1 is a block diagram of the AWU.

## 4.2 Features

Features of the auto wakeup module include:

- One internal interrupt with separate interrupt enable bit, sharing the same keyboard interrupt vector and keyboard interrupt mask bit
- Exit from low-power stop mode without external signals
- Selectable timeout periods
- Dedicated low-power internal oscillator separate from the main system clock sources

## 4.3 Functional Description

The function of the auto wakeup logic is to generate periodic wakeup requests to bring the microcontroller unit (MCU) out of stop mode. The wakeup requests are treated as regular keyboard interrupt requests, with the difference that instead of a pin, the interrupt signal is generated by an internal logic.

Writing the AWUIE bit in the keyboard interrupt enable register enables or disables the auto wakeup interrupt input (see Figure 4-1). A logic 1 applied to the AWUIREQ input with auto wakeup interrupt request enabled, latches an auto wakeup interrupt request.

Auto wakeup latch, AWUL, can be read directly from the bit 6 position of port A data register (PTA). This is a read-only bit which is occupying an empty bit position on PTA. No PTA associated registers, such as PTA6 data direction or PTA6 pullup exist for this bit.

Entering stop mode will enable the auto wakeup generation logic. An internal RC oscillator (exclusive for the auto wakeup feature) drives the wakeup request generator. Once the overflow count is reached in the generator counter, a wakeup request, AWUIREQ, is latched and sent to the KBI logic. See Figure 4-1.

Wakeup interrupt requests will only be serviced if the associated interrupt enable bit, AWUIE, in KBIER is set. The AWU shares the keyboard interrupt vector.

The overflow count can be selected from two options defined by the COPRS bit in CONFIG1. This bit was “borrowed” from the computer operating properly (COP) using the fact that the COP feature is idle (no MCU clock available) in stop mode. The typical values of the periodic wakeup request are (at room temperature):

- COPRS = 0: 650 ms @ 5 V, 875 ms @ 3 V
- COPRS = 1: 16 ms @ 5 V, 22 ms @ 3 V

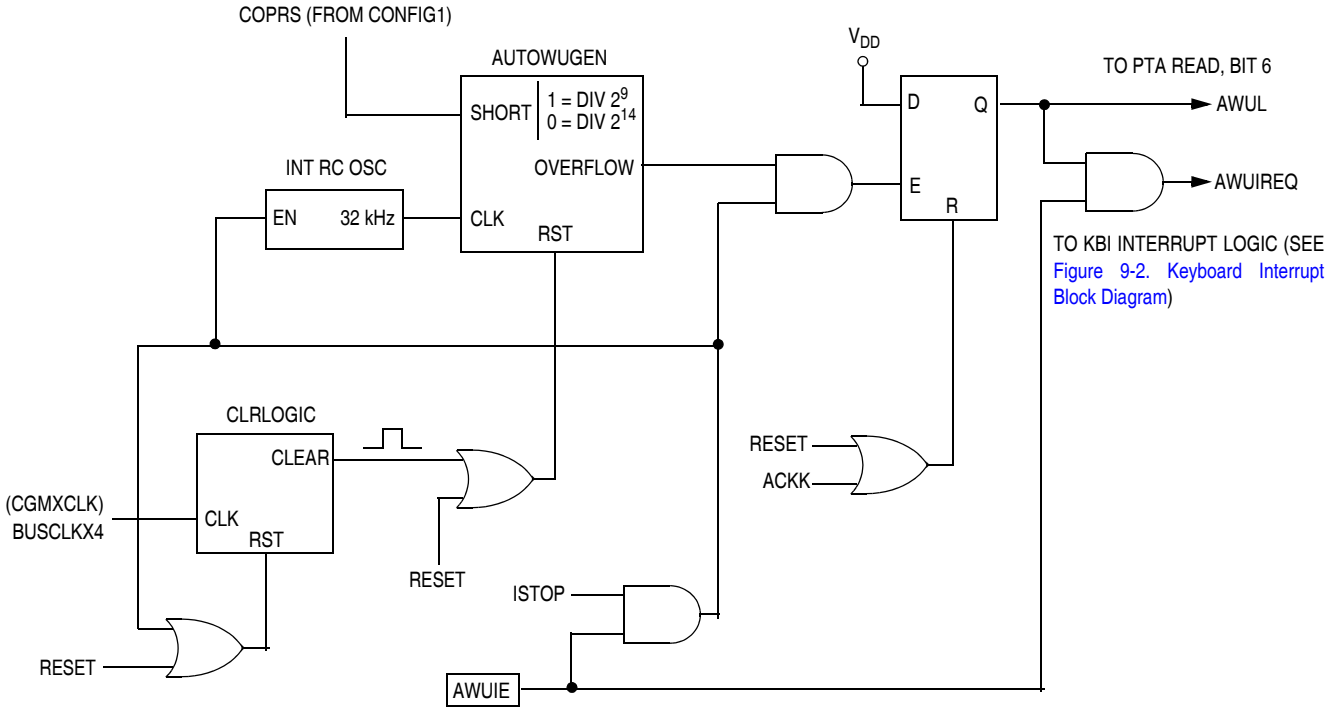



**Figure 4-1. Auto Wakeup Interrupt Request Generation Logic**

The auto wakeup RC oscillator is highly dependent on operating voltage and temperature. This feature is not recommended for use as a time-keeping function.

The wakeup request is latched to allow the interrupt source identification. The latched value, AWUL, can be read directly from the bit 6 position of PTA data register. This is a read-only bit which is occupying an empty bit position on PTA. No PTA associated registers, such as PTA6 data, PTA6 direction, and PTA6 pullup exist for this bit. The latch can be cleared by writing to the ACKK bit in the KBSCR register. Reset also clears the latch. AWUIE bit in KBI interrupt enable register (see Figure 4-1) has no effect on AWUL reading.

The AWU oscillator and counters are inactive in normal operating mode and become active only upon entering stop mode.

## 4.4 Wait Mode

The AWU module remains inactive in wait mode.

## 4.5 Stop Mode

When the AWU module is enabled (AWUIE = 1 in the keyboard interrupt enable register) it is activated automatically upon entering 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. The AWU counters start from ‘0’ each time stop mode is entered.

## 4.6 Input/Output Registers

The AWU shares registers with the keyboard interrupt (KBI) module and the port A I/O module. The following I/O registers control and monitor operation of the AWU:

- Port A data register (PTA)
- Keyboard interrupt status and control register (KBSCR)
- Keyboard interrupt enable register (KBIER)

### 4.6.1 Port A I/O Register

The port A data register (PTA) contains a data latch for the state of the AWU interrupt request, in addition to the data latches for port A.

Address: $0000

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: | 0 | AWUL | PTA5 | PTA4 | PTA3 | PTA2 | PTA1 | PTA0 |
| Write: | | | | | | | | |
| Reset: | 0 | 0 | Unaffected by reset | | | | | |

= Unimplemented

**Figure 4-2. Port A Data Register (PTA)**

**AWUL — Auto Wakeup Latch**

This is a read-only bit which has the value of the auto wakeup interrupt request latch. The wakeup request signal is generated internally. There is no PTA6 port or any of the associated bits such as PTA6 data direction or pullup bits.

1 = Auto wakeup interrupt request is pending
0 = Auto wakeup interrupt request is not pending

***NOTE***

*PTA5–PTA0 bits are not used in conjuction with the auto wakeup feature. To see a description of these bits, see 12.2.1 Port A Data Register.*

### 4.6.2 Keyboard Status and Control Register

The keyboard status and control register (KBSCR):

- Flags keyboard/auto wakeup interrupt requests
- Acknowledges keyboard/auto wakeup interrupt requests
- Masks keyboard/auto wakeup interrupt requests

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 4-3. Keyboard Status and Control Register (KBSCR)**

**Bits 7–4 — Not used**

These read-only bits always read as 0s.

**KEYF — Keyboard Flag Bit**

This read-only bit is set when a keyboard interrupt is pending on port A or auto wakeup. Reset clears the KEYF bit.

1 = Keyboard/auto wakeup interrupt pending
0 = No keyboard/auto wakeup interrupt pending

**ACKK — Keyboard Acknowledge Bit**

Writing a 1 to this write-only bit clears the keyboard/auto wakeup interrupt request on port A and auto wakeup logic. ACKK always reads as 0.Reset clears ACKK.

**IMASKK— Keyboard Interrupt Mask Bit**

Writing a 1 to this read/write bit prevents the output of the keyboard interrupt mask from generating interrupt requests on port A or auto wakeup. Reset clears the IMASKK bit.

1 = Keyboard/auto wakeup interrupt requests masked
0 = Keyboard/auto wakeup interrupt requests not masked

***NOTE***

*MODEK is not used in conjuction with the auto wakeup feature. To see a description of this bit, see 9.7.1 Keyboard Status and Control Register.*

### 4.6.3 Keyboard Interrupt Enable Register

The keyboard interrupt enable register (KBIER) enables or disables the auto wakeup to operate as a keyboard/auto wakeup interrupt input.

Address: $001B

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: | 0 | AWUIE | KBIE5 | KBIE4 | KBIE3 | KBIE2 | KBIE1 | KBIE0 |
| Write: | | | | | | | | |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 4-4. Keyboard Interrupt Enable Register (KBIER)**

**AWUIE — Auto Wakeup Interrupt Enable Bit**

This read/write bit enables the auto wakeup interrupt input to latch interrupt requests. Reset clears AWUIE.

1 = Auto wakeup enabled as interrupt input
0 = Auto wakeup not enabled as interrupt input

***NOTE***

*KBIE5–KBIE0 bits are not used in conjuction with the auto wakeup feature. To see a description of these bits, see 9.7.2 Keyboard Interrupt Enable Register.*

# Chapter 5
# Configuration Register (CONFIG)

## 5.1 Introduction

This section describes the configuration registers (CONFIG1 and CONFIG2). The configuration registers enable or disable the following options:

- Stop mode recovery time (32 × BUSCLKX4 cycles or 4096 × BUSCLKX4 cycles)
- STOP instruction
- Computer operating properly module (COP)
- COP reset period (COPRS): 8176 × BUSCLKX4 or 262,128 × BUSCLKX4
- Low-voltage inhibit (LVI) enable and trip voltage selection
- OSC option selection
- $\overline{\text{IRQ}}$ pin
- $\overline{\text{RST}}$ pin
- Auto wakeup timeout period

## 5.2 Functional Description

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

***NOTE***

*The CONFIG registers are one-time writable by the user after each reset. Upon a reset, the CONFIG registers default to predetermined settings as shown in Figure 5-1 and Figure 5-2.*

Address: $001E

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read:<br>Write: | IRQPUD | IRQEN | R | OSCOPT1 | OSCOPT0 | R | R | RSTEN |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | U |
| POR: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R = Reserved U = Unaffected

**Figure 5-1. Configuration Register 2 (CONFIG2)**

**IRQPUD — $\overline{\text{IRQ}}$ Pin Pullup Control Bit**

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

**IRQEN — $\overline{\text{IRQ}}$ Pin Function Selection Bit**

1 = Interrupt request function active in pin
0 = Interrupt request function inactive in pin

**OSCOPT1 and OSCOPT0 — Selection Bits for Oscillator Option**

(0, 0) Internal oscillator
(0, 1) External oscillator
(1, 0) External RC oscillator
(1, 1) External XTAL oscillator

**RSTEN — $\overline{\text{RST}}$ Pin Function Selection**

1 = Reset function active in pin
0 = Reset function inactive in pin

***NOTE***

*The RSTEN bit is cleared by a power-on reset (POR) only. Other resets will leave this bit unaffected.*

Address: $001F

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read:<br>Write: | COPRS | LVISTOP | LVIRSTD | LVIPWRD | LVI5OR3 | SSREC | STOP | COPD |
| Reset: | 0 | 0 | 0 | 0 | U | 0 | 0 | 0 |
| POR: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

U = Unaffected

**Figure 5-2. Configuration Register 1 (CONFIG1)**

**COPRS (Out of STOP Mode) — COP Reset Period Selection Bit**

1 = COP reset short cycle = 8176 × BUSCLKX4
0 = COP reset long cycle = 262,128 × BUSCLKX4

**COPRS (In STOP Mode) — Auto Wakeup Period Selection Bit**

1 = Auto wakeup short cycle = 512 × INTRCOSC
0 = Auto wakeup long cycle = 16,384 × INTRCOSC

**LVISTOP — LVI Enable in Stop Mode Bit**

When the LVIPWRD bit is clear, setting the LVISTOP bit enables the LVI to operate during stop mode. Reset clears LVISTOP.

1 = LVI enabled during stop mode
0 = LVI disabled during stop mode

**LVIRSTD — LVI Reset Disable Bit**

LVIRSTD disables the reset signal from the LVI module.

1 = LVI module resets disabled
0 = LVI module resets enabled

**LVIPWRD — LVI Power Disable Bit**

LVIPWRD disables the LVI module.

1 = LVI module power disabled
0 = LVI module power enabled

**LVI5OR3 — LVI 5-V or 3-V Operating Mode Bit**

LVI5OR3 selects the voltage operating mode of the LVI module. The voltage mode selected for the LVI should match the operating $V_{DD}$ for the LVI's voltage trip points for each of the modes.

1 = LVI operates in 5-V mode
0 = LVI operates in 3-V mode

***NOTE***

*The LVI5OR3 bit is cleared by a power-on reset (POR) only. Other resets will leave this bit unaffected.*

**SSREC — Short Stop Recovery Bit**

SSREC enables the CPU to exit stop mode with a delay of 32 BUSCLKX4 cycles instead of a 4096 BUSCLKX4 cycle delay.

1 = Stop mode recovery after 32 BUSCLKX4 cycles
0 = Stop mode recovery after 4096 BUSCLKX4 cycles

***NOTE***

*Exiting stop mode by an LVI reset will result in the long stop recovery.*

The system stabilization time for power-on reset and long stop recovery (both 4096 BUSCLKX4 cycles) gives a delay longer than the LVI enable time for these startup scenarios. There is no period where the MCU is not protected from a low-power condition. However, when using the short stop recovery configuration option, the 32 BUSCLKX4 delay must be greater than the LVI's turn on time to avoid a period in startup where the LVI is not protecting the MCU.

**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.

1 = COP module disabled
0 = COP module enabled

# Chapter 6
# Computer Operating Properly (COP)

## 6.1 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 configuration 1 (CONFIG1) register.

## 6.2 Functional Description

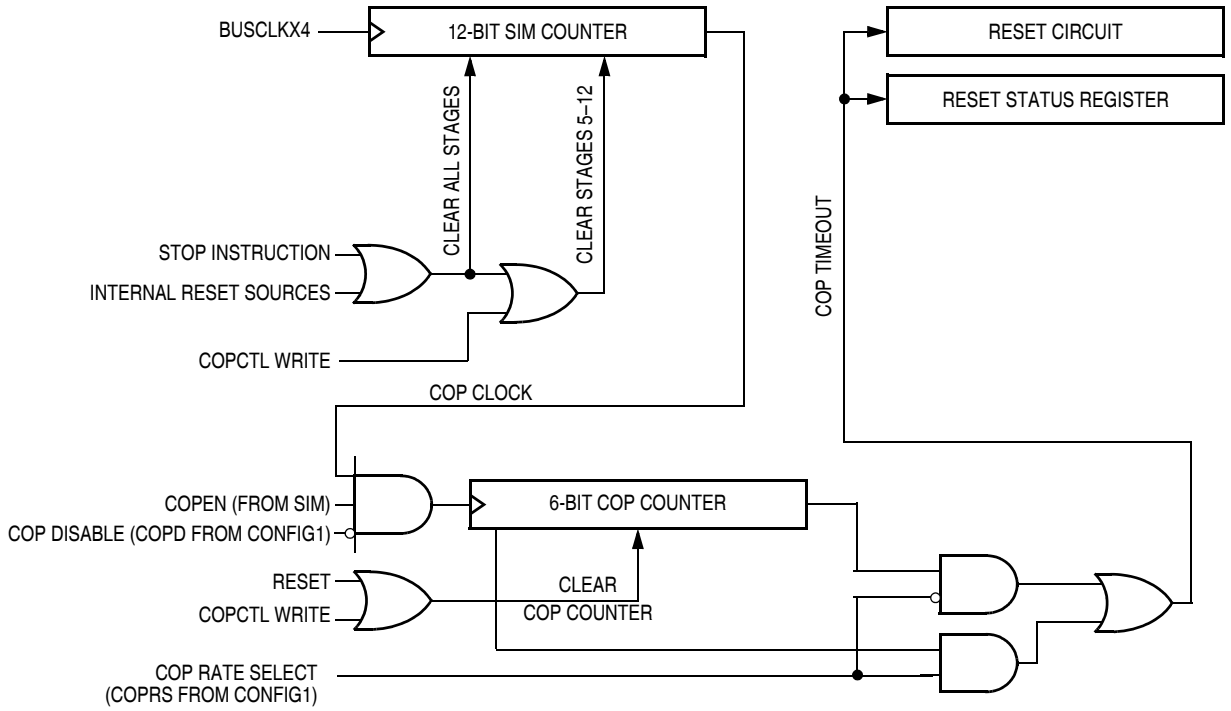



**Figure 6-1. COP Block Diagram**

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 262,128 or 8176 BUSCLKX4 cycles; depending on the state of the COP rate select bit, COPRS, in configuration register 1. With a 262,128 BUSCLKX4 cycle overflow option, the internal 12.8-MHz oscillator gives a COP timeout period of 20.48 ms. Writing any value to location $FFFF before an overflow occurs prevents a COP reset by clearing the COP counter and stages 12–5 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 (if the RSTEN bit is set in the CONFIG1 register) for 32 × BUSCLKX4 cycles and sets the COP bit in the reset status register (RSR). See 13.8.1 SIM Reset Status Register.

***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.*

## 6.3 I/O Signals

The following paragraphs describe the signals shown in Figure 6-1.

### 6.3.1 BUSCLKX4

BUSCLKX4 is the oscillator output signal. BUSCLKX4 frequency is equal to the internal oscillator frequency, the crystal frequency, or the RC-oscillator frequency.

### 6.3.2 STOP Instruction

The STOP instruction clears the SIM counter.

### 6.3.3 COPCTL Write

Writing any value to the COP control register (COPCTL) (see 6.4 COP Control Register) clears the COP counter and clears stages 12–5 of the SIM counter. Reading the COP control register returns the low byte of the reset vector.

### 6.3.4 Power-On Reset

The power-on reset (POR) circuit in the SIM clears the SIM counter 4096 × BUSCLKX4 cycles after power up.

### 6.3.5 Internal Reset

An internal reset clears the SIM counter and the COP counter.

### 6.3.6 COPD (COP Disable)

The COPD signal reflects the state of the COP disable bit (COPD) in the configuration register 1 (CONFIG1). See Chapter 5 Configuration Register (CONFIG).

### 6.3.7 COPRS (COP Rate Select)

The COPRS signal reflects the state of the COP rate select bit (COPRS) in the configuration register 1 (CONFIG1). See Chapter 5 Configuration Register (CONFIG).

## 6.4 COP Control Register

The COP control register (COPCTL) 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.

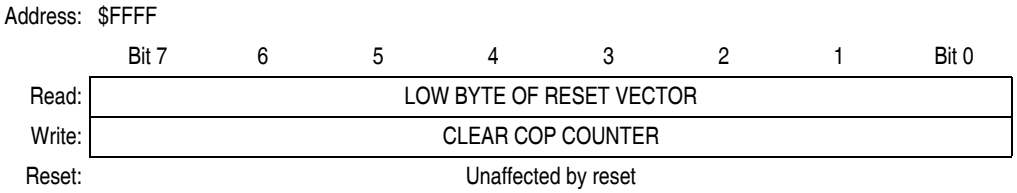



**Figure 6-2. COP Control Register (COPCTL)**

## 6.5 Interrupts

The COP does not generate CPU interrupt requests.

## 6.6 Monitor Mode

The COP is disabled in monitor mode when $V_{TST}$ is present on the $\overline{IRQ}$ pin.

## 6.7 Low-Power Modes

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

### 6.7.1 Wait Mode

The COP continues to operate during wait mode. To prevent a COP reset during wait mode, periodically clear the COP counter.

### 6.7.2 Stop Mode

Stop mode turns off the BUSCLKX4 input to the COP and clears the SIM counter. Service the COP immediately before entering or after exiting stop mode to ensure a full COP timeout period after entering or exiting stop mode.

## 6.8 COP Module During Break Mode

The COP is disabled during a break interrupt with monitor mode when BDCOP bit is set in break auxiliary register (BRKAR).

# Chapter 7
# Central Processor Unit (CPU)

## 7.1 Introduction

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

## 7.2 Features

Features of the CPU include:

- 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

## 7.3 CPU Registers

Figure 7-1 shows the five CPU registers. CPU registers are not part of the memory map.

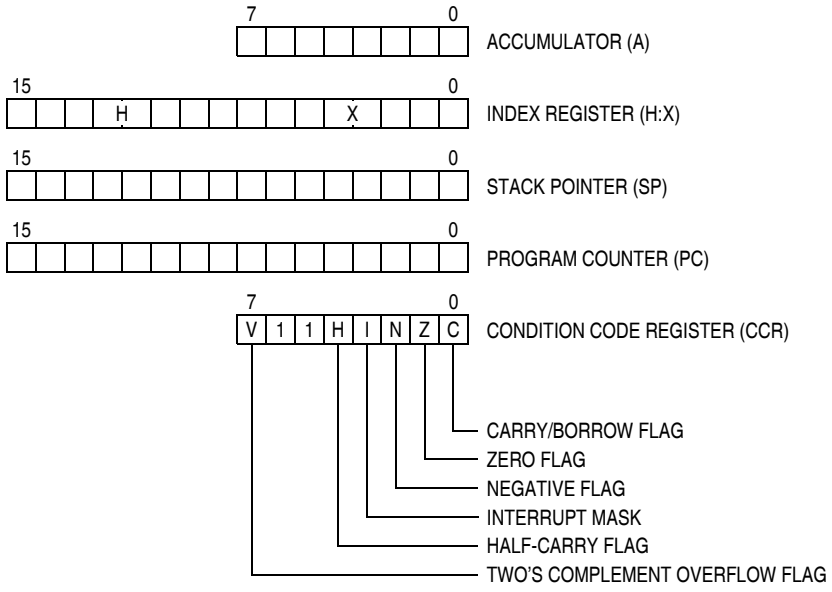



**Figure 7-1. CPU Registers**

### 7.3.1 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.

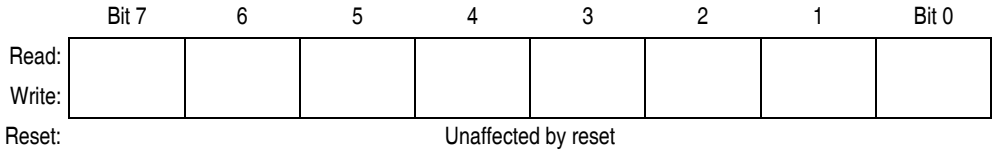



**Figure 7-2. Accumulator (A)**

### 7.3.2 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.

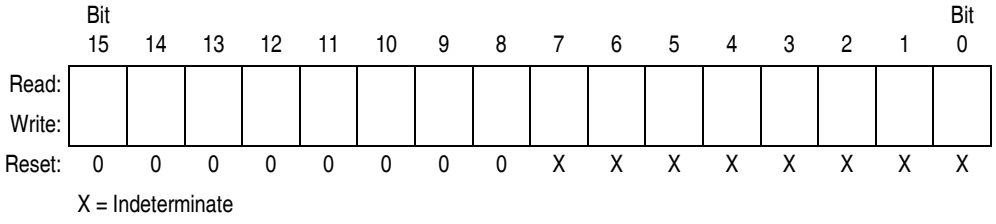



**Figure 7-3. Index Register (H:X)**

### 7.3.3 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.

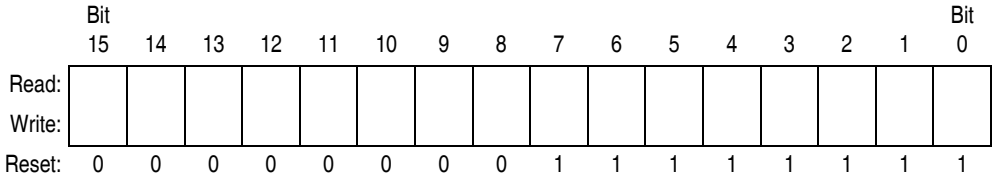



**Figure 7-4. Stack Pointer (SP)**

***NOTE***

*The location of the stack is arbitrary and may be relocated anywhere in random-access memory (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.*

### 7.3.4 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.

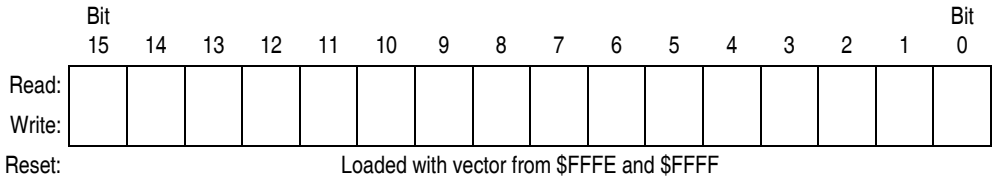



**Figure 7-5. Program Counter (PC)**

### 7.3.5 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 1. The following paragraphs describe the functions of the condition code register.

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

X = Indeterminate

**Figure 7-6. 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

## 7.4 Arithmetic/Logic Unit (ALU)

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

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

## 7.5 Low-Power Modes

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

### 7.5.1 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

### 7.5.2 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.

## 7.6 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

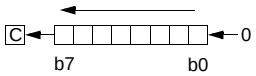
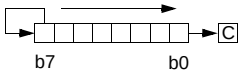
The break interrupt begins after completion of the CPU instruction in progress. If the break address register match occurs on the last cycle of a CPU instruction, the break interrupt begins immediately.

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.

## 7.7 Instruction Set Summary

Table 7-1 provides a summary of the M68HC08 instruction set.

**Table 7-1. Instruction Set Summary (Sheet 1 of 6)**

| Source Form | Operation | Description | Effect on CCR | | | | | | Address Mode | Opcode | Operand | Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | V | H | I | N | Z | C | | | | |
| ADC #*opr*<br>ADC *opr*<br>ADC *opr*<br>ADC *opr*,X<br>ADC *opr*,X<br>ADC ,X<br>ADC *opr*,SP<br>ADC *opr*,SP | Add with Carry | A ← (A) + (M) + (C) | ↕ | ↕ | – | ↕ | ↕ | ↕ | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A9<br>B9<br>C9<br>D9<br>E9<br>F9<br>9EE9<br>9ED9 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| ADD #*opr*<br>ADD *opr*<br>ADD *opr*<br>ADD *opr*,X<br>ADD *opr*,X<br>ADD ,X<br>ADD *opr*,SP<br>ADD *opr*,SP | Add without Carry | A ← (A) + (M) | ↕ | ↕ | – | ↕ | ↕ | ↕ | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | AB<br>BB<br>CB<br>DB<br>EB<br>FB<br>9EEB<br>9EDB | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>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 |
| AND #*opr*<br>AND *opr*<br>AND *opr*<br>AND *opr*,X<br>AND *opr*,X<br>AND ,X<br>AND *opr*,SP<br>AND *opr*,SP | Logical AND | A ← (A) & (M) | 0 | – | – | ↕ | ↕ | – | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A4<br>B4<br>C4<br>D4<br>E4<br>F4<br>9EE4<br>9ED4 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| ASL *opr*<br>ASLA<br>ASLX<br>ASL *opr*,X<br>ASL ,X<br>ASL *opr*,SP | Arithmetic Shift Left<br>(Same as LSL) |  | ↕ | – | – | ↕ | ↕ | ↕ | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 38<br>48<br>58<br>68<br>78<br>9E68 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| ASR *opr*<br>ASRA<br>ASRX<br>ASR *opr*,X<br>ASR *opr*,X<br>ASR *opr*,SP | Arithmetic Shift Right |  | ↕ | – | – | ↕ | ↕ | ↕ | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 37<br>47<br>57<br>67<br>77<br>9E67 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>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)<br>DIR (b1)<br>DIR (b2)<br>DIR (b3)<br>DIR (b4)<br>DIR (b5)<br>DIR (b6)<br>DIR (b7) | 11<br>13<br>15<br>17<br>19<br>1B<br>1D<br>1F | dd<br>dd<br>dd<br>dd<br>dd<br>dd<br>dd<br>dd | 4<br>4<br>4<br>4<br>4<br>4<br>4<br>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) \| (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) \| (Z) = 0 | – | – | – | – | – | – | REL | 22 | rr | 3 |

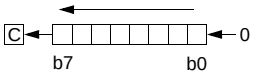
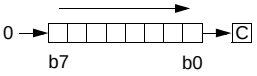
## Table 7-1. Instruction Set Summary (Sheet 2 of 6)

| Source Form | Operation | Description | Effect on CCR | | | | | | Address Mode | Opcode | Operand | Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | V | H | I | N | Z | C | | | | |
| 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 |
| BIT #*opr*<br>BIT *opr*<br>BIT *opr*<br>BIT *opr*,X<br>BIT *opr*,X<br>BIT ,X<br>BIT *opr*,SP<br>BIT *opr*,SP | Bit Test | (A) & (M) | 0 | – | – | ↕ | ↕ | – | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A5<br>B5<br>C5<br>D5<br>E5<br>F5<br>9EE5<br>9ED5 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>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)<br>DIR (b1)<br>DIR (b2)<br>DIR (b3)<br>DIR (b4)<br>DIR (b5)<br>DIR (b6)<br>DIR (b7) | 01<br>03<br>05<br>07<br>09<br>0B<br>0D<br>0F | dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr | 5<br>5<br>5<br>5<br>5<br>5<br>5<br>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)<br>DIR (b1)<br>DIR (b2)<br>DIR (b3)<br>DIR (b4)<br>DIR (b5)<br>DIR (b6)<br>DIR (b7) | 00<br>02<br>04<br>06<br>08<br>0A<br>0C<br>0E | dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr<br>dd rr | 5<br>5<br>5<br>5<br>5<br>5<br>5<br>5 |
| BSET *n*,*opr* | Set Bit *n* in M | Mn ← 1 | – | – | – | – | – | – | DIR (b0)<br>DIR (b1)<br>DIR (b2)<br>DIR (b3)<br>DIR (b4)<br>DIR (b5)<br>DIR (b6)<br>DIR (b7) | 10<br>12<br>14<br>16<br>18<br>1A<br>1C<br>1E | dd<br>dd<br>dd<br>dd<br>dd<br>dd<br>dd<br>dd | 4<br>4<br>4<br>4<br>4<br>4<br>4<br>4 |
| BSR *rel* | Branch to Subroutine | PC ← (PC) + 2; push (PCL)<br>SP ← (SP) – 1; push (PCH)<br>SP ← (SP) – 1<br>PC ← (PC) + *rel* | – | – | – | – | – | – | REL | AD | rr | 4 |
| CBEQ *opr*,*rel*<br>CBEQA #*opr*,*rel*<br>CBEQX #*opr*,*rel*<br>CBEQ *opr*,X+,*rel*<br>CBEQ X+,*rel*<br>CBEQ *opr*,SP,*rel* | Compare and Branch if Equal | PC ← (PC) + 3 + rel ? (A) – (M) = $00<br>PC ← (PC) + 3 + rel ? (A) – (M) = $00<br>PC ← (PC) + 3 + rel ? (X) – (M) = $00<br>PC ← (PC) + 3 + rel ? (A) – (M) = $00<br>PC ← (PC) + 2 + rel ? (A) – (M) = $00<br>PC ← (PC) + 4 + rel ? (A) – (M) = $00 | – | – | – | – | – | – | DIR<br>IMM<br>IMM<br>IX1+<br>IX+<br>SP1 | 31<br>41<br>51<br>61<br>71<br>9E61 | dd rr<br>ii rr<br>ii rr<br>ff rr<br>rr<br>ff rr | 5<br>4<br>4<br>5<br>4<br>6 |
| CLC | Clear Carry Bit | C ← 0 | – | – | – | – | – | 0 | INH | 98 | | 1 |

## Table 7-1. Instruction Set Summary (Sheet 3 of 6)

| Source Form | Operation | Description | Effect on CCR | | | | | | Address Mode | Opcode | Operand | Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | V | H | I | N | Z | C | | | | |
| CLI | Clear Interrupt Mask | I ← 0 | – | – | 0 | – | – | – | INH | 9A | | 2 |
| CLR *opr*<br>CLRA<br>CLRX<br>CLRH<br>CLR *opr*,X<br>CLR ,X<br>CLR *opr*,SP | Clear | M ← $00<br>A ← $00<br>X ← $00<br>H ← $00<br>M ← $00<br>M ← $00<br>M ← $00 | 0 | – | – | 0 | 1 | – | DIR<br>INH<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 3F<br>4F<br>5F<br>8C<br>6F<br>7F<br>9E6F | dd<br><br><br><br>ff<br><br>ff | 3<br>1<br>1<br>1<br>3<br>2<br>4 |
| CMP #*opr*<br>CMP *opr*<br>CMP *opr*<br>CMP *opr*,X<br>CMP *opr*,X<br>CMP ,X<br>CMP *opr*,SP<br>CMP *opr*,SP | Compare A with M | (A) – (M) | ↕ | – | – | ↕ | ↕ | ↕ | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A1<br>B1<br>C1<br>D1<br>E1<br>F1<br>9EE1<br>9ED1 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| COM *opr*<br>COMA<br>COMX<br>COM *opr*,X<br>COM ,X<br>COM *opr*,SP | Complement (One's Complement) | M ← ($\overline{M}$) = $FF – (M)<br>A ← ($\overline{A}$) = $FF – (M)<br>X ← ($\overline{X}$) = $FF – (M)<br>M ← ($\overline{M}$) = $FF – (M)<br>M ← ($\overline{M}$) = $FF – (M)<br>M ← ($\overline{M}$) = $FF – (M) | 0 | – | – | ↕ | ↕ | 1 | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 33<br>43<br>53<br>63<br>73<br>9E63 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| CPHX #*opr*<br>CPHX *opr* | Compare H:X with M | (H:X) – (M:M + 1) | ↕ | – | – | ↕ | ↕ | ↕ | IMM<br>DIR | 65<br>75 | ii ii+1<br>dd | 3<br>4 |
| CPX #*opr*<br>CPX *opr*<br>CPX *opr*<br>CPX ,X<br>CPX *opr*,X<br>CPX *opr*,X<br>CPX *opr*,SP<br>CPX *opr*,SP | Compare X with M | (X) – (M) | ↕ | – | – | ↕ | ↕ | ↕ | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A3<br>B3<br>C3<br>D3<br>E3<br>F3<br>9EE3<br>9ED3 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| DAA | Decimal Adjust A | $(A)_{10}$ | U | – | – | ↕ | ↕ | ↕ | INH | 72 | | 2 |
| DBNZ *opr,rel*<br>DBNZA *rel*<br>DBNZX *rel*<br>DBNZ *opr*,X,*rel*<br>DBNZ X,*rel*<br>DBNZ *opr*,SP,*rel* | Decrement and Branch if Not Zero | A ← (A) – 1 or M ← (M) – 1 or X ← (X) – 1<br>PC ← (PC) + 3 + *rel* ? (result) ≠ 0<br>PC ← (PC) + 2 + *rel* ? (result) ≠ 0<br>PC ← (PC) + 2 + *rel* ? (result) ≠ 0<br>PC ← (PC) + 3 + *rel* ? (result) ≠ 0<br>PC ← (PC) + 2 + *rel* ? (result) ≠ 0<br>PC ← (PC) + 4 + *rel* ? (result) ≠ 0 | – | – | – | – | – | – | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 3B<br>4B<br>5B<br>6B<br>7B<br>9E6B | dd rr<br>rr<br>rr<br>ff rr<br>rr<br>ff rr | 5<br>3<br>3<br>5<br>4<br>6 |
| DEC *opr*<br>DECA<br>DECX<br>DEC *opr*,X<br>DEC ,X<br>DEC *opr*,SP | Decrement | M ← (M) – 1<br>A ← (A) – 1<br>X ← (X) – 1<br>M ← (M) – 1<br>M ← (M) – 1<br>M ← (M) – 1 | ↕ | – | – | ↕ | ↕ | – | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 3A<br>4A<br>5A<br>6A<br>7A<br>9E6A | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| DIV | Divide | A ← (H:A)/(X)<br>H ← Remainder | – | – | – | – | ↕ | ↕ | INH | 52 | | 7 |
| EOR #*opr*<br>EOR *opr*<br>EOR *opr*<br>EOR *opr*,X<br>EOR *opr*,X<br>EOR ,X<br>EOR *opr*,SP<br>EOR *opr*,SP | Exclusive OR M with A | A ← (A ⊕ M) | 0 | – | – | ↕ | ↕ | – | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A8<br>B8<br>C8<br>D8<br>E8<br>F8<br>9EE8<br>9ED8 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| INC *opr*<br>INCA<br>INCX<br>INC *opr*,X<br>INC ,X<br>INC *opr*,SP | Increment | M ← (M) + 1<br>A ← (A) + 1<br>X ← (X) + 1<br>M ← (M) + 1<br>M ← (M) + 1<br>M ← (M) + 1 | ↕ | – | – | ↕ | ↕ | – | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 3C<br>4C<br>5C<br>6C<br>7C<br>9E6C | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |

## Table 7-1. Instruction Set Summary (Sheet 4 of 6)

| Source Form | Operation | Description | Effect on CCR | | | | | | Address Mode | Opcode | Operand | Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | V | H | I | N | Z | C | | | | |
| JMP *opr*<br>JMP *opr*<br>JMP *opr*,X<br>JMP *opr*,X<br>JMP ,X | Jump | PC ← Jump Address | – | – | – | – | – | – | DIR<br>EXT<br>IX2<br>IX1<br>IX | BC<br>CC<br>DC<br>EC<br>FC | dd<br>hh ll<br>ee ff<br>ff | 2<br>3<br>4<br>3<br>2 |
| JSR *opr*<br>JSR *opr*<br>JSR *opr*,X<br>JSR *opr*,X<br>JSR ,X | Jump to Subroutine | PC ← (PC) + *n* (*n* = 1, 2, or 3)<br>Push (PCL); SP ← (SP) – 1<br>Push (PCH); SP ← (SP) – 1<br>PC ← Unconditional Address | – | – | – | – | – | – | DIR<br>EXT<br>IX2<br>IX1<br>IX | BD<br>CD<br>DD<br>ED<br>FD | dd<br>hh ll<br>ee ff<br>ff | 4<br>5<br>6<br>5<br>4 |
| LDA #*opr*<br>LDA *opr*<br>LDA *opr*<br>LDA *opr*,X<br>LDA *opr*,X<br>LDA ,X<br>LDA *opr*,SP<br>LDA *opr*,SP | Load A from M | A ← (M) | 0 | – | – | ↕ | ↕ | – | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A6<br>B6<br>C6<br>D6<br>E6<br>F6<br>9EE6<br>9ED6 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| LDHX #*opr*<br>LDHX *opr* | Load H:X from M | H:X ← (M:M + 1) | 0 | – | – | ↕ | ↕ | – | IMM<br>DIR | 45<br>55 | ii jj<br>dd | 3<br>4 |
| LDX #*opr*<br>LDX *opr*<br>LDX *opr*<br>LDX *opr*,X<br>LDX *opr*,X<br>LDX ,X<br>LDX *opr*,SP<br>LDX *opr*,SP | Load X from M | X ← (M) | 0 | – | – | ↕ | ↕ | – | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | AE<br>BE<br>CE<br>DE<br>EE<br>FE<br>9EEE<br>9EDE | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| LSL *opr*<br>LSLA<br>LSLX<br>LSL *opr*,X<br>LSL ,X<br>LSL *opr*,SP | Logical Shift Left<br>(Same as ASL) |  | ↕ | – | – | ↕ | ↕ | ↕ | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 38<br>48<br>58<br>68<br>78<br>9E68 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| LSR *opr*<br>LSRA<br>LSR*X*<br>LSR *opr*,X<br>LSR ,X<br>LSR *opr*,SP | Logical Shift Right |  | ↕ | – | – | 0 | ↕ | ↕ | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 34<br>44<br>54<br>64<br>74<br>9E64 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| MOV *opr*,*opr*<br>MOV *opr*,X+<br>MOV #*opr*,*opr*<br>MOV X+,*opr* | Move | $(M)_{Destination} \leftarrow (M)_{Source}$<br><br>H:X ← (H:X) + 1 (IX+D, DIX+) | 0 | – | – | ↕ | ↕ | – | DD<br>DIX+<br>IMD<br>IX+D | 4E<br>5E<br>6E<br>7E | dd dd<br>dd<br>ii dd<br>dd | 5<br>4<br>4<br>4 |
| MUL | Unsigned multiply | X:A ← (X) × (A) | – | 0 | – | – | – | 0 | INH | 42 | | 5 |
| NEG *opr*<br>NEGA<br>NEGX<br>NEG *opr*,X<br>NEG ,X<br>NEG *opr*,SP | Negate (Two's Complement) | M ← –(M) = $00 – (M)<br>A ← –(A) = $00 – (A)<br>X ← –(X) = $00 – (X)<br>M ← –(M) = $00 – (M)<br>M ← –(M) = $00 – (M) | ↕ | – | – | ↕ | ↕ | ↕ | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 30<br>40<br>50<br>60<br>70<br>9E60 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| NOP | No Operation | None | – | – | – | – | – | – | INH | 9D | | 1 |
| NSA | Nibble Swap A | A ← (A[3:0]:A[7:4]) | – | – | – | – | – | – | INH | 62 | | 3 |
| ORA #*opr*<br>ORA *opr*<br>ORA *opr*<br>ORA *opr*,X<br>ORA *opr*,X<br>ORA ,X<br>ORA *opr*,SP<br>ORA *opr*,SP | Inclusive OR A and M | A ← (A) \| (M) | 0 | – | – | ↕ | ↕ | – | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | AA<br>BA<br>CA<br>DA<br>EA<br>FA<br>9EEA<br>9EDA | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>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 |

## Table 7-1. Instruction Set Summary (Sheet 5 of 6)

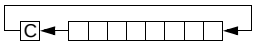
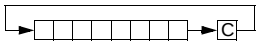
| Source Form | Operation | Description | Effect on CCR | | | | | | Address Mode | Opcode | Operand | Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | V | H | I | N | Z | C | | | | |
| PULA | Pull A from Stack | SP ← (SP + 1); Pull (A) | – | – | – | – | – | – | INH | 86 | | 2 |
| 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*<br>ROLA<br>ROLX<br>ROL *opr*,X<br>ROL ,X<br>ROL *opr*,SP | Rotate Left through Carry |  | ↕ | – | – | ↕ | ↕ | ↕ | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 39<br>49<br>59<br>69<br>79<br>9E69 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| ROR *opr*<br>RORA<br>RORX<br>ROR *opr*,X<br>ROR ,X<br>ROR *opr*,SP | Rotate Right through Carry |  | ↕ | – | – | ↕ | ↕ | ↕ | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 36<br>46<br>56<br>66<br>76<br>9E66 | dd<br><br><br>ff<br><br>ff | 4<br>1<br>1<br>4<br>3<br>5 |
| RSP | Reset Stack Pointer | SP ← $FF | – | – | – | – | – | – | INH | 9C | | 1 |
| RTI | Return from Interrupt | SP ← (SP) + 1; Pull (CCR)<br>SP ← (SP) + 1; Pull (A)<br>SP ← (SP) + 1; Pull (X)<br>SP ← (SP) + 1; Pull (PCH)<br>SP ← (SP) + 1; Pull (PCL) | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | INH | 80 | | 7 |
| RTS | Return from Subroutine | SP ← SP + 1; Pull (PCH)<br>SP ← SP + 1; Pull (PCL) | – | – | – | – | – | – | INH | 81 | | 4 |
| SBC #*opr*<br>SBC *opr*<br>SBC *opr*<br>SBC *opr*,X<br>SBC *opr*,X<br>SBC ,X<br>SBC *opr*,SP<br>SBC *opr*,SP | Subtract with Carry | A ← (A) – (M) – (C) | ↕ | – | – | ↕ | ↕ | ↕ | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A2<br>B2<br>C2<br>D2<br>E2<br>F2<br>9EE2<br>9ED2 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |
| SEC | Set Carry Bit | C ← 1 | – | – | – | – | – | 1 | INH | 99 | | 1 |
| SEI | Set Interrupt Mask | I ← 1 | – | – | 1 | – | – | – | INH | 9B | | 2 |
| STA *opr*<br>STA *opr*<br>STA *opr*,X<br>STA *opr*,X<br>STA ,X<br>STA *opr*,SP<br>STA *opr*,SP | Store A in M | M ← (A) | 0 | – | – | ↕ | ↕ | – | DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | B7<br>C7<br>D7<br>E7<br>F7<br>9EE7<br>9ED7 | dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 3<br>4<br>4<br>3<br>2<br>4<br>5 |
| STHX *opr* | Store H:X in M | (M:M + 1) ← (H:X) | 0 | – | – | ↕ | ↕ | – | DIR | 35 | dd | 4 |
| STOP | Enable Interrupts, Stop Processing, Refer to MCU Documentation | I ← 0; Stop Processing | – | – | 0 | – | – | – | INH | 8E | | 1 |
| STX *opr*<br>STX *opr*<br>STX *opr*,X<br>STX *opr*,X<br>STX ,X<br>STX *opr*,SP<br>STX *opr*,SP | Store X in M | M ← (X) | 0 | – | – | ↕ | ↕ | – | DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | BF<br>CF<br>DF<br>EF<br>FF<br>9EEF<br>9EDF | dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 3<br>4<br>4<br>3<br>2<br>4<br>5 |
| SUB #*opr*<br>SUB *opr*<br>SUB *opr*<br>SUB *opr*,X<br>SUB *opr*,X<br>SUB ,X<br>SUB *opr*,SP<br>SUB *opr*,SP | Subtract | A ← (A) – (M) | ↕ | – | – | ↕ | ↕ | ↕ | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP1<br>SP2 | A0<br>B0<br>C0<br>D0<br>E0<br>F0<br>9EE0<br>9ED0 | ii<br>dd<br>hh ll<br>ee ff<br>ff<br><br>ff<br>ee ff | 2<br>3<br>4<br>4<br>3<br>2<br>4<br>5 |

**Table 7-1. Instruction Set Summary (Sheet 6 of 6)**

| Source Form | Operation | Description | Effect on CCR | | | | | | Address Mode | Opcode | Operand | Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | V | H | I | N | Z | C | | | | |
| SWI | Software Interrupt | PC ← (PC) + 1; Push (PCL)<br>SP ← (SP) – 1; Push (PCH)<br>SP ← (SP) – 1; Push (X)<br>SP ← (SP) – 1; Push (A)<br>SP ← (SP) – 1; Push (CCR)<br>SP ← (SP) – 1; I ← 1<br>PCH ← Interrupt Vector High Byte<br>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*<br>TSTA<br>TSTX<br>TST *opr*,X<br>TST ,X<br>TST *opr*,SP | Test for Negative or Zero | (A) – $00 or (X) – $00 or (M) – $00 | 0 | – | – | ↕ | ↕ | – | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 3D<br>4D<br>5D<br>6D<br>7D<br>9E6D | dd<br><br><br>ff<br><br>ff | 3<br>1<br>1<br>3<br>2<br>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 |
| WAIT | Enable Interrupts; Wait for Interrupt | I bit ← 0; Inhibit CPU clocking until interrupted | – | – | 0 | – | – | – | INH | 8F | | 1 |

| | | | |
|---|---|---|---|
| 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 |

## 7.8 Opcode Map

See Table 7-2.

# Table 7-2. 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 | CI 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 | SI 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 | CI 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 | AI 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 | B 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 | LI 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 | OI 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 | LI 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

Low Byte of Opcode in Hexadecimal

| MSB / LSB | 0 |
|---|---|
| 0 | 5 BRSET0 3 DIR |

High Byte of O
Cycles
Opcode Mnem
Number of Byt

# Chapter 8
# External Interrupt (IRQ)

## 8.1 Introduction

The $\overline{\text{IRQ}}$ pin (external interrupt), shared with PTA2 (general purpose input) and keyboard interrupt (KBI), provides a maskable interrupt input

## 8.2 Features

Features of the IRQ module include the following:

- External interrupt pin, $\overline{\text{IRQ}}$
- $\overline{\text{IRQ}}$ interrupt control bits
- Programmable edge-only or edge and level interrupt sensitivity
- Automatic interrupt acknowledge
- Selectable internal pullup resistor

## 8.3 Functional Description

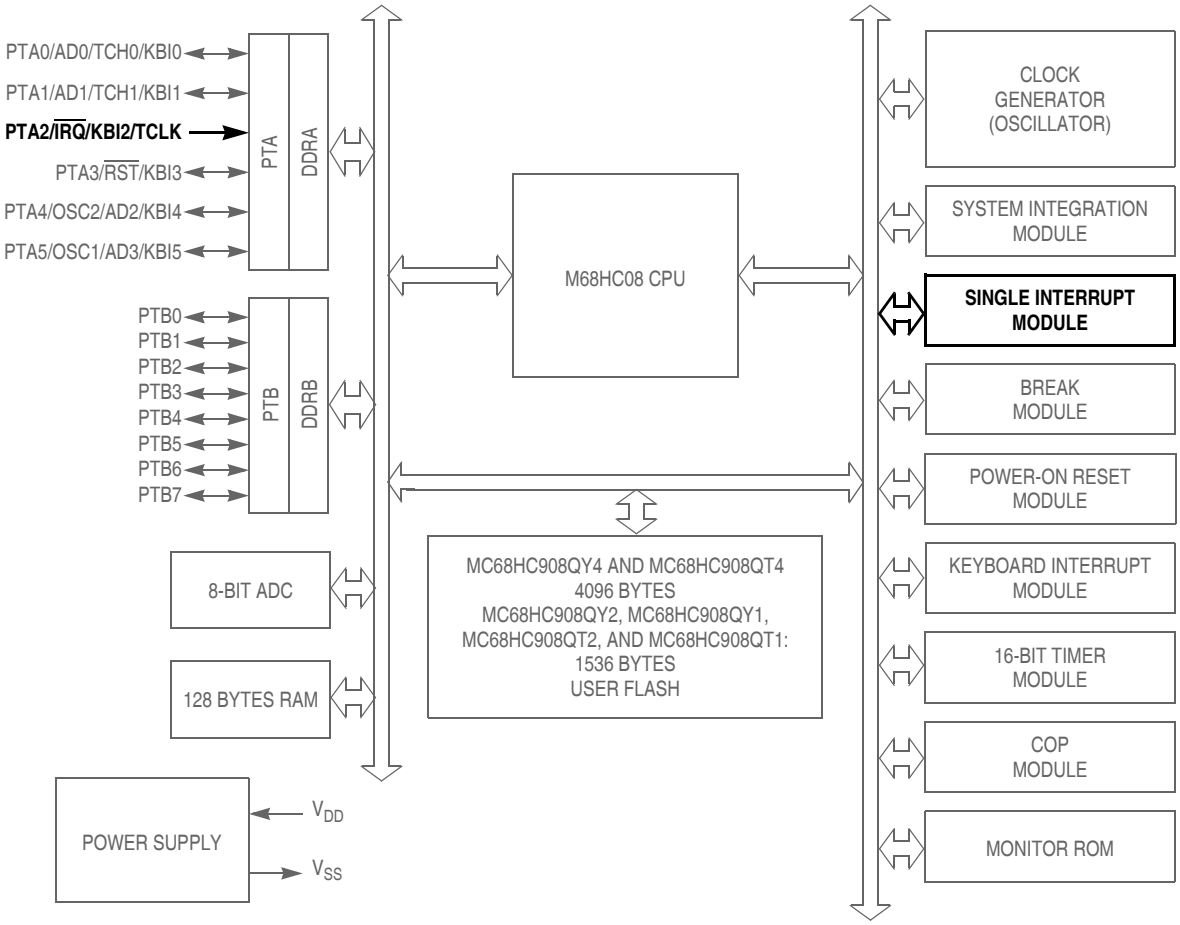
$\overline{\text{IRQ}}$ pin functionality is enabled by setting configuration register 2 (CONFIG2) IRQEN bit accordingly. A zero disables the IRQ function and PTA2 will assume the other shared functionalities. A one enables the IRQ function.

A low level applied to the external interrupt request ($\overline{\text{IRQ}}$) pin can latch a CPU interrupt request. Figure 8-2 shows the structure of the IRQ module.

Interrupt signals on the $\overline{\text{IRQ}}$ pin are latched into the IRQ latch. The IRQ latch remains set until one of the following actions occurs:

- IRQ vector fetch — An IRQ vector fetch automatically generates an interrupt acknowledge signal that clears the IRQ latch.
- Software clear — Software can clear the IRQ latch by writing a 1 to the ACK bit in the interrupt status and control register (INTSCR).
- Reset — A reset automatically clears the IRQ latch.

The external interrupt pin is falling-edge-triggered out of reset and is software-configurable to be either falling-edge or falling-edge and low-level triggered. The MODE bit in INTSCR controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin.




**RST, IRQ: Pins have internal (about 30K Ohms) pull up**
PTA[0:5]: High current sink and source capability
PTA[0:5]: Pins have programmable keyboard interrupt and pull up
PTB[0:7]: Not available on 8-pin devices – MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 (see note in 12.1 Introduction)
ADC: Not available on the MC68HC908QY1 and MC68HC908QT1

**Figure 8-1. Block Diagram Highlighting IRQ Block and Pins**

When set, the IMASK bit in INTSCR masks the IRQ interrupt request. A latched interrupt request is not presented to the interrupt priority logic unless IMASK is clear.

***NOTE***

*The interrupt mask (I) in the condition code register (CCR) masks all interrupt requests, including the IRQ interrupt request.*

A falling edge on the IRQ pin can latch an interrupt request into the IRQ latch. An IRQ vector fetch, software clear, or reset clears the IRQ latch.

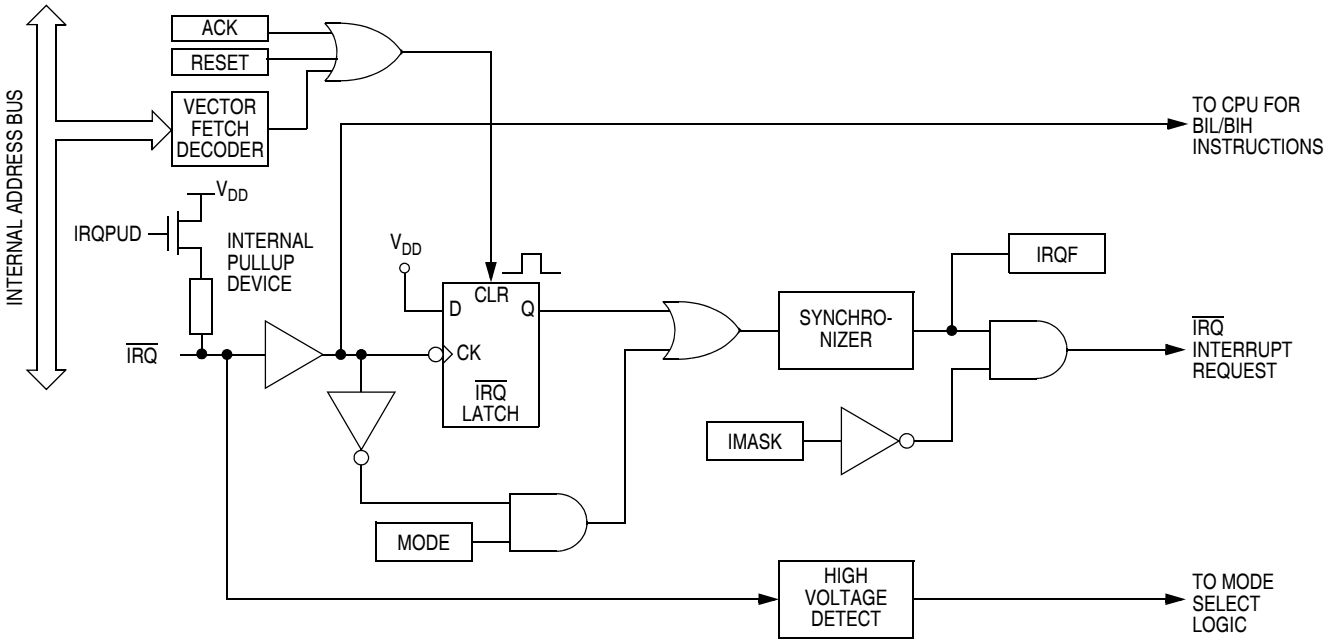



**Figure 8-2. IRQ Module Block Diagram**

### 8.3.1 MODE = 1

If the MODE bit is set, the $\overline{IRQ}$ pin is both falling edge sensitive and low level sensitive. With MODE set, both of the following actions must occur to clear the $\overline{IRQ}$ interrupt request:

- Return of the $\overline{IRQ}$ pin to a high level. As long as the $\overline{IRQ}$ pin is low, the IRQ request remains active.
- IRQ vector fetch or software clear. An IRQ vector fetch generates an interrupt acknowledge signal to clear the IRQ latch. Software generates the interrupt acknowledge signal by writing a 1 to ACK in INTSCR. The ACK bit is useful in applications that poll the $\overline{IRQ}$ pin and require software to clear the IRQ latch. Writing to ACK 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 ACK latches another interrupt request. If the IRQ mask bit, IMASK, is clear, the CPU loads the program counter with the IRQ vector address.

The IRQ vector fetch or software clear and the return of the $\overline{IRQ}$ pin to a high level may occur in any order. The interrupt request remains pending as long as the $\overline{IRQ}$ pin is low. A reset will clear the IRQ latch and the MODE control bit, thereby clearing the interrupt even if the pin stays low.

Use the BIH or BIL instruction to read the logic level on the $\overline{IRQ}$ pin.

### 8.3.2 MODE = 0

If the MODE bit is clear, the $\overline{IRQ}$ pin is falling edge sensitive only. With MODE clear, an IRQ vector fetch or software clear immediately clears the IRQ latch.

The IRQF bit in INTSCR can be read to check for pending interrupts. The IRQF bit is not affected by IMASK, which makes it useful in applications where polling is preferred.

***NOTE***

*When using the level-sensitive interrupt trigger, avoid false IRQ interrupts by masking interrupt requests in the interrupt routine.*

## 8.4 Interrupts

The following IRQ source can generate interrupt requests:

- Interrupt flag (IRQF) — The IRQF bit is set when the $\overline{\text{IRQ}}$ pin is asserted based on the IRQ mode. The IRQ interrupt mask bit, IMASK, is used to enable or disable IRQ interrupt requests.

## 8.5 Low-Power Modes

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

### 8.5.1 Wait Mode

The IRQ module remains active in wait mode. Clearing IMASK in INTSCR enables IRQ interrupt requests to bring the MCU out of wait mode.

### 8.5.2 Stop Mode

The IRQ module remains active in stop mode. Clearing IMASK in INTSCR enables IRQ interrupt requests to bring the MCU out of stop mode.

## 8.6 IRQ Module During Break 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. See Chapter 13 System Integration Module (SIM).

To allow software to clear status bits during a break interrupt, write a 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 0 to BCFE. With BCFE cleared (its default state), software can read and write registers during the break state without affecting status bits. Some status bits have a two-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 cleared. After the break, doing the second step clears the status bit.

## 8.7 I/O Signals

The IRQ module shares its pin with the keyboard interrupt, input/output ports, and timer interface modules.

***NOTE***

*When the $\overline{\text{IRQ}}$ function is enabled in the CONFIG2 register, the BIH and BIL instructions can be used to read the logic level on the $\overline{\text{IRQ}}$ pin. If the $\overline{\text{IRQ}}$ function is disabled, these instructions will behave as if the $\overline{\text{IRQ}}$ pin is a logic 1, regardless of the actual level on the pin. Conversely, when the $\overline{\text{IRQ}}$ function is enabled, bit 2 of the port A data register will always read a 0.*

*When using the level-sensitive interrupt trigger, avoid false interrupts by masking interrupt requests in the interrupt routine. An internal pullup resistor to $V_{DD}$ is connected to the $\overline{\text{IRQ}}$ pin; this can be disabled by setting the IRQPUD bit in the CONFIG2 register ($001E).*

### 8.7.1 IRQ Input Pins ($\overline{IRQ}$)

The $\overline{IRQ}$ pin provides a maskable external interrupt source. The $\overline{IRQ}$ pin contains an internal pullup device.

## 8.8 Registers

The IRQ status and control register (INTSCR) controls and monitors operation of the IRQ module. See Chapter 5 Configuration Register (CONFIG).

The INTSCR has the following functions:

- Shows the state of the IRQ flag
- Clears the IRQ latch
- Masks the IRQ 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 | 0 | IMASK | MODE |
| Write: | | | | | | ACK | | |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 8-3. IRQ Status and Control Register (INTSCR)**

**IRQF — IRQ Flag**

This read-only status bit is set 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 1 to this write-only bit clears the IRQ latch. ACK always reads as 0.

**IMASK — IRQ Interrupt Mask Bit**

Writing a 1 to this read/write bit disables the IRQ interrupt request.

1 = IRQ interrupt request disabled
0 = IRQ interrupt request enabled

**MODE — IRQ Edge/Level Select Bit**

This read/write bit controls the triggering sensitivity of the $\overline{IRQ}$ pin.

1 = $\overline{IRQ}$ interrupt request on falling edges and low levels
0 = $\overline{IRQ}$ interrupt request on falling edges only

# Chapter 9
# Keyboard Interrupt Module (KBI)

## 9.1 Introduction

The keyboard interrupt module (KBI) provides six independently maskable external interrupts, which are accessible via the PTA0–PTA5 pins.

## 9.2 Features

Features of the keyboard interrupt module include:

- Six keyboard interrupt pins with separate keyboard interrupt enable bits and one keyboard interrupt mask
- Software configurable pullup device if input pin is configured as input port bit
- Programmable edge-only or edge and level interrupt sensitivity
- Exit from low-power modes

## 9.3 Functional Description

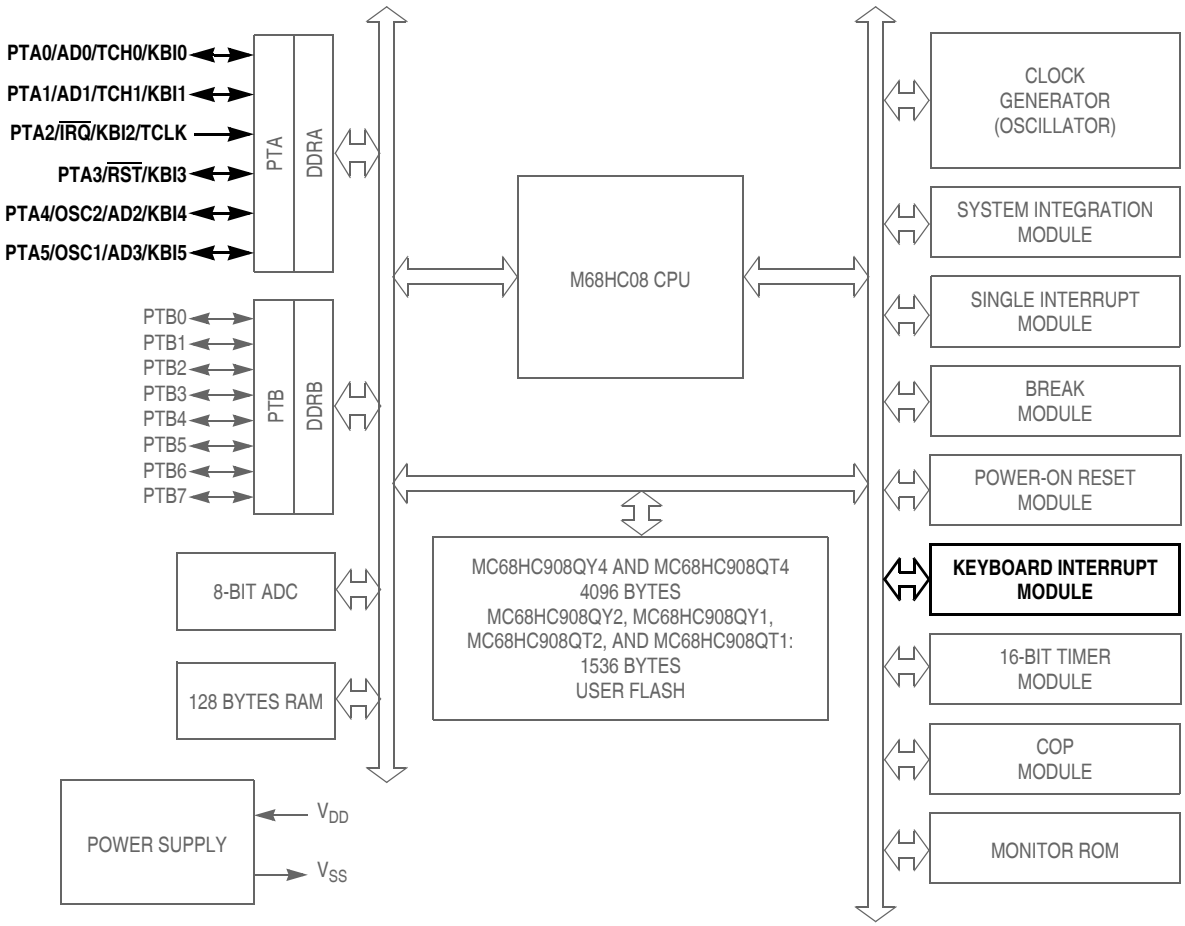
The keyboard interrupt module controls the enabling/disabling of interrupt functions on the six port A pins. These six pins can be enabled/disabled independently of each other. Refer to Figure 9-2.

### 9.3.1 Keyboard Operation

Writing to the KBIE0–KBIE5 bits in the keyboard interrupt enable register (KBIER) 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 pullup device irrespective of PTAPUEx bits in the port A input pullup enable register (see 12.2.3 Port A Input Pullup Enable Register). A logic 0 applied to an enabled keyboard interrupt pin latches a keyboard interrupt request.

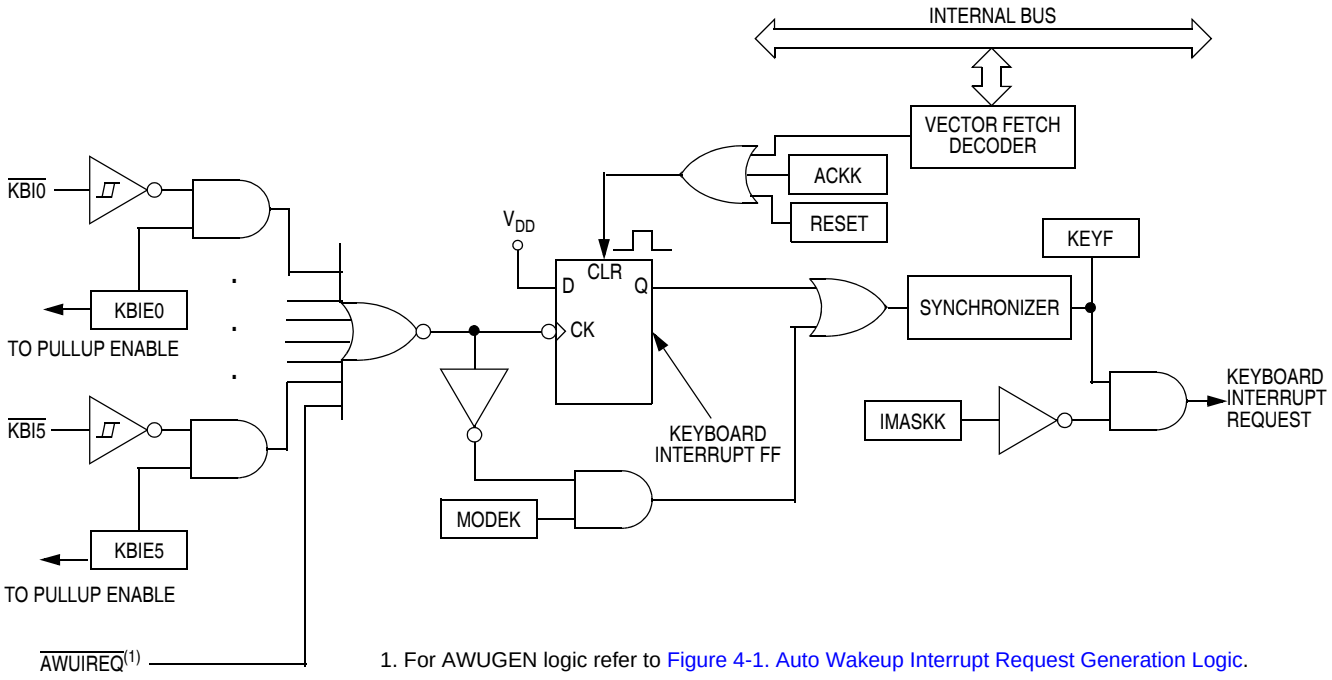
A keyboard interrupt is latched when one or more keyboard interrupt inputs goes low after all were high. The MODEK bit in the keyboard status and control register controls the triggering mode of the keyboard interrupt.

- If the keyboard interrupt is edge-sensitive only, a falling edge on a keyboard interrupt input does not latch an interrupt request if another keyboard pin is already low. To prevent losing an interrupt request on one input because another input is still low, software can disable the latter input 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 interrupt input is low.




**RST, IRQ: Pins have internal (about 30K Ohms) pull up**
**PTA[0:5]: High current sink and source capability**
**PTA[0:5]: Pins have programmable keyboard interrupt and pull up**
**PTB[0:7]: Not available on 8-pin devices – MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 (see note in 12.1 Introduction)**
ADC: Not available on the MC68HC908QY1 and MC68HC908QT1

**Figure 9-1. Block Diagram Highlighting KBI Block and Pins**




1. For AWUGEN logic refer to Figure 4-1. Auto Wakeup Interrupt Request Generation Logic.

**Figure 9-2. Keyboard Interrupt Block Diagram**

If the MODEK bit is set, the keyboard interrupt inputs 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 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 inputs 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 inputs. 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 central processor unit (CPU) loads the program counter with the vector address at locations $FFE0 and $FFE1.
- Return of all enabled keyboard interrupt inputs to logic 1 — As long as any enabled keyboard interrupt pin is at logic 0, the keyboard interrupt remains set. The auto wakeup interrupt input, AWUIREQ, will be cleared only by writing to ACKK bit in KBSCR or reset.

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 input 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, 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 0 for software to read the pin.*

### 9.3.2 Keyboard Initialization

When a keyboard interrupt pin is enabled, it takes time for the internal pullup 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 1s 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.

## 9.4 Wait Mode

The keyboard module remains 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.

## 9.5 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.

## 9.6 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 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 0 to the BCFE bit. With BCFE at 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.

## 9.7 Input/Output Registers

The following I/O registers control and monitor operation of the keyboard interrupt module:

- Keyboard interrupt status and control register (KBSCR)
- Keyboard interrupt enable register (KBIER)

### 9.7.1 Keyboard Status and Control Register

The keyboard status and control register (KBSCR):

- 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 9-3. Keyboard Status and Control Register (KBSCR)**

**Bits 7–4 — Not used**

These read-only bits always read as 0s.

**KEYF — Keyboard Flag Bit**

This read-only bit is set when a keyboard interrupt is pending on port A or auto wakeup. Reset clears the KEYF bit.

1 = Keyboard interrupt pending
0 = No keyboard interrupt pending

**ACKK — Keyboard Acknowledge Bit**

Writing a 1 to this write-only bit clears the keyboard interrupt request on port A and auto wakeup logic. ACKK always reads as 0. Reset clears ACKK.

**IMASKK— Keyboard Interrupt Mask Bit**

Writing a 1 to this read/write bit prevents the output of the keyboard interrupt mask from generating interrupt requests on port A or auto wakeup. 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 and auto wakeup. Reset clears MODEK.

1 = Keyboard interrupt requests on falling edges and low levels
0 = Keyboard interrupt requests on falling edges only

## 9.7.2 Keyboard Interrupt Enable Register

The port A keyboard interrupt enable register (KBIER) enables or disables each port A pin or auto wakeup to operate as a keyboard interrupt input.

Address: $001B

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: | 0 | AWUIE | KBIE5 | KBIE4 | KBIE3 | KBIE2 | KBIE1 | KBIE0 |
| Write: | | | | | | | | |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 9-4. Keyboard Interrupt Enable Register (KBIER)**

**KBIE5–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

***NOTE***

*AWUIE bit is not used in conjunction with the keyboard interrupt feature. To see a description of this bit, see Chapter 4 Auto Wakeup Module (AWU).*

# Chapter 10
# Low-Voltage Inhibit (LVI)

## 10.1 Introduction

This section describes the low-voltage inhibit (LVI) module, which monitors the voltage on the $V_{DD}$ pin and can force a reset when the $V_{DD}$ voltage falls below the LVI trip falling voltage, $V_{TRIPF}$.

## 10.2 Features

Features of the LVI module include:

- Programmable LVI reset
- Programmable power consumption
- Selectable LVI trip voltage
- Programmable stop mode operation

## 10.3 Functional Description

Figure 10-1 shows the structure of the LVI module. LVISTOP, LVIPWRD, LVI5OR3, and LVIRSTD are user selectable options found in the configuration register (CONFIG1). See Chapter 5 Configuration Register (CONFIG).

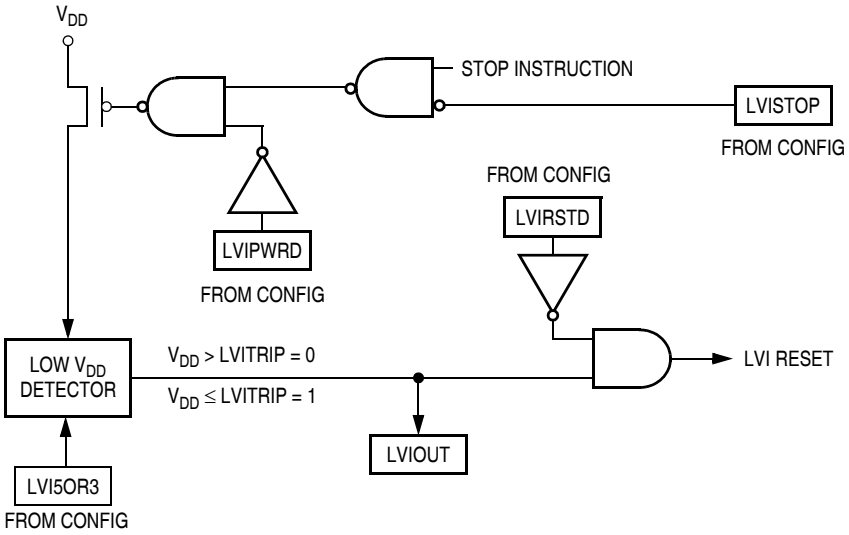



**Figure 10-1. LVI Module Block Diagram**

The LVI is enabled out of reset. The LVI module contains a bandgap reference circuit and comparator. Clearing the LVI power disable bit, LVIPWRD, enables the LVI to monitor $V_{DD}$ voltage. Clearing the LVI reset disable bit, LVIRSTD, enables the LVI module to generate a reset when $V_{DD}$ falls below a voltage,

$V_{TRIPF}$. Setting the LVI enable in stop mode bit, LVISTOP, enables the LVI to operate in stop mode. Setting the LVI 5-V or 3-V trip point bit, LVI5OR3, enables the trip point voltage, $V_{TRIPF}$, to be configured for 5-V operation. Clearing the LVI5OR3 bit enables the trip point voltage, $V_{TRIPF}$, to be configured for 3-V operation. The actual trip thresholds are specified in 16.5 5-V DC Electrical Characteristics and 16.9 3-V DC Electrical Characteristics.

***NOTE***

*After a power-on reset, the LVI's default mode of operation is 3 volts. If a 5-V system is used, the user must set the LVI5OR3 bit to raise the trip point to 5-V operation.*

*If the user requires 5-V mode and sets the LVI5OR3 bit after power-on reset while the $V_{DD}$ supply is not above the $V_{TRIPR}$ for 5-V mode, the microcontroller unit (MCU) will immediately go into reset. The next time the LVI releases the reset, the supply will be above the $V_{TRIPR}$ for 5-V mode.*

Once an LVI reset occurs, the MCU remains in reset until $V_{DD}$ rises above a voltage, $V_{TRIPR}$, which causes the MCU to exit reset. See Chapter 13 System Integration Module (SIM) for the reset recovery sequence.

The output of the comparator controls the state of the LVIOUT flag in the LVI status register (LVISR) and can be used for polling LVI operation when the LVI reset is disabled.

### 10.3.1 Polled LVI Operation

In applications that can operate at $V_{DD}$ levels below the $V_{TRIPF}$ level, software can monitor $V_{DD}$ by polling the LVIOUT bit. In the configuration register, the LVIPWRD bit must be cleared to enable the LVI module, and the LVIRSTD bit must be at set to disable LVI resets.

### 10.3.2 Forced Reset Operation

In applications that require $V_{DD}$ to remain above the $V_{TRIPF}$ level, enabling LVI resets allows the LVI module to reset the MCU when $V_{DD}$ falls below the $V_{TRIPF}$ level. In the configuration register, the LVIPWRD and LVIRSTD bits must be cleared to enable the LVI module and to enable LVI resets.

### 10.3.3 Voltage Hysteresis Protection

Once the LVI has triggered (by having $V_{DD}$ fall below $V_{TRIPF}$), the LVI will maintain a reset condition until $V_{DD}$ rises above the rising trip point voltage, $V_{TRIPR}$. This prevents a condition in which the MCU is continually entering and exiting reset if $V_{DD}$ is approximately equal to $V_{TRIPF}$. $V_{TRIPR}$ is greater than $V_{TRIPF}$ by the hysteresis voltage, $V_{HYS}$.

### 10.3.4 LVI Trip Selection

The LVI5OR3 bit in the configuration register selects whether the LVI is configured for 5-V or 3-V protection.

***NOTE***

*The microcontroller is guaranteed to operate at a minimum supply voltage. The trip point ($V_{TRIPF}$ [5 V] or $V_{TRIPF}$ [3 V]) may be lower than this. See 16.5 5-V DC Electrical Characteristics and 16.9 3-V DC Electrical Characteristics for the actual trip point voltages.*

## 10.4 LVI Status Register

The LVI status register (LVISR) indicates if the $V_{DD}$ voltage was detected below the $V_{TRIPF}$ level while LVI resets have been disabled.

Address: $FE0C

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: | LVIOUT | 0 | 0 | 0 | 0 | 0 | 0 | R |
| Write: | | | | | | | | |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented  R = Reserved

**Figure 10-2. LVI Status Register (LVISR)**

**LVIOUT — LVI Output Bit**

This read-only flag becomes set when the $V_{DD}$ voltage falls below the $V_{TRIPF}$ trip voltage and is cleared when $V_{DD}$ voltage rises above $V_{TRIPR}$. The difference in these threshold levels results in a hysteresis that prevents oscillation into and out of reset (see Table 10-1). Reset clears the LVIOUT bit.

**Table 10-1. LVIOUT Bit Indication**

| $V_{DD}$ | LVIOUT |
|---|---|
| $V_{DD} > V_{TRIPR}$ | 0 |
| $V_{DD} < V_{TRIPF}$ | 1 |
| $V_{TRIPF} < V_{DD} < V_{TRIPR}$ | Previous value |

## 10.5 LVI Interrupts

The LVI module does not generate interrupt requests.

## 10.6 Low-Power Modes

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

### 10.6.1 Wait Mode

If enabled, the LVI module remains active in wait mode. If enabled to generate resets, the LVI module can generate a reset and bring the MCU out of wait mode.

### 10.6.2 Stop Mode

When the LVIPWRD bit in the configuration register is cleared and the LVISTOP bit in the configuration register is set, the LVI module remains active in stop mode. If enabled to generate resets, the LVI module can generate a reset and bring the MCU out of stop mode.

# Chapter 11
# Oscillator Module (OSC)

## 11.1 Introduction

The oscillator module is used to provide a stable clock source for the microcontroller system and bus. The oscillator module generates two output clocks, BUSCLKX2 and BUSCLKX4. The BUSCLKX4 clock is used by the system integration module (SIM) and the computer operating properly module (COP). The BUSCLKX2 clock is divided by two in the SIM to be used as the bus clock for the microcontroller. Therefore the bus frequency will be one fourth of the BUSCLKX4 frequency.

## 11.2 Features

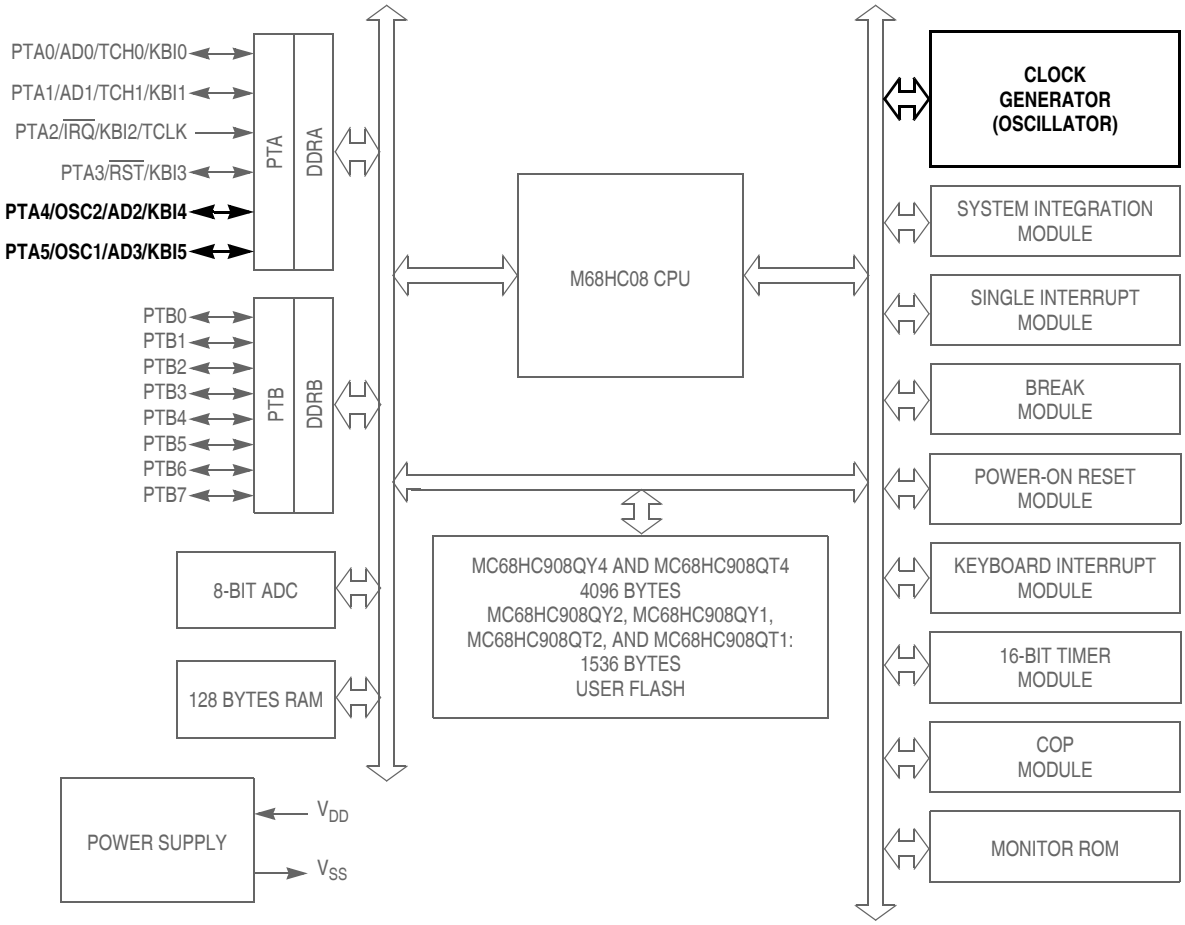
The oscillator has these four clock source options available:

1. Internal oscillator: An internally generated, fixed frequency clock, trimmable to ±5%.This is the default option out of reset.
2. External oscillator: An external clock that can be driven directly into OSC1.
3. External RC: A built-in oscillator module (RC oscillator) that requires an external R connection only. The capacitor is internal to the chip.
4. External crystal: A built-in oscillator module (XTAL oscillator) that requires an external crystal or ceramic-resonator.

## 11.3 Functional Description

The oscillator contains these major subsystems:

- Internal oscillator circuit
- Internal or external clock switch control
- External clock circuit
- External crystal circuit
- External RC clock circuit




RST, IRQ: Pins have internal (about 30K Ohms) pull up
**PTA[0:5]: High current sink and source capability**
**PTA[0:5]: Pins have programmable keyboard interrupt and pull up**
PTB[0:7]: Not available on 8-pin devices – MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 (see note in 12.1 Introduction)
ADC: Not available on the MC68HC908QY1 and MC68HC908QT1

**Figure 11-1. Block Diagram Highlighting OSC Block and Pins**

### 11.3.1 Internal Oscillator

The internal oscillator circuit is designed for use with no external components to provide a clock source with tolerance less than ±25% untrimmed.An 8-bit trimming register allows adjustment to a tolerance of less than ±5%.

The internal oscillator will generate a clock of 12.8 MHz typical (INTCLK) resulting in a bus speed (internal clock ÷ 4) of 3.2 MHz. 3.2 MHz came from the maximum bus speed guaranteed at 3 V which is 4 MHz.Since the internal oscillator will have a ±25% tolerance (pre-trim), then the +25% case should not allow a frequency higher than 4 MHz:

3.2 MHz + 25% = 4 MHz

Figure 11-3 shows how BUSCLKX4 is derived from INTCLK and, like the RC oscillator, OSC2 can output BUSCLKX4 by setting OSC2EN in PTAPUE register. See Chapter 12 Input/Output Ports (PORTS)

#### 11.3.1.1 Internal Oscillator Trimming

The 8-bit trimming register, OSCTRIM, allows a clock period adjust of +127 and –128 steps. Increasing OSCTRIM value increases the clock period. Trimming allows the internal clock frequency to be set to 12.8 MHz ± 5%.

All devices are factory programmed with trim values in reserved FLASH memory locations $FFC0 and $FFC1. The trim value is not automatically loaded into the OSCTRIM register. User software must copy the trim value from $FFC0 or $FFC1 into OSCTRIM if needed. The factory trim value provides the accuracy required for communication using forced monitor mode. Some production programmers erase the factory trim values, so confirm with your programmer vendor that the trim values at $FFC0 and $FFC1 are preserved, or are re-trimmed. Trimming the device in the user application board will provide the most accurate trim value.

#### 11.3.1.2 Internal to External Clock Switching

When external clock source (external OSC, RC, or XTAL) is desired, the user must perform the following steps:

1. For external crystal circuits only, OSCOPT[1:0] = 1:1: To help precharge an external crystal oscillator, set PTA4 (OSC2) as an output and drive high for several cycles. This may help the crystal circuit start more robustly.
2. Set CONFIG2 bits OSCOPT[1:0] according to . The oscillator module control logic will then set OSC1 as an external clock input and, if the external crystal option is selected, OSC2 will also be set as the clock output.
3. Create a software delay to wait the stabilization time needed for the selected clock source (crystal, resonator, RC) as recommended by the component manufacturer. A good rule of thumb for crystal oscillators is to wait 4096 cycles of the crystal frequency, i.e., for a 4-MHz crystal, wait approximately 1 msec.
4. After the manufacturer’s recommended delay has elapsed, the ECGON bit in the OSC status register (OSCSTAT) needs to be set by the user software.
5. After ECGON set is detected, the OSC module checks for oscillator activity by waiting two external clock rising edges.
6. The OSC module then switches to the external clock. Logic provides a glitch free transition.
7. The OSC module first sets the ECGST bit in the OSCSTAT register and then stops the internal oscillator.

**NOTE**

*Once transition to the external clock is done, the internal oscillator will only be reactivated with reset. No post-switch clock monitor feature is implemented (clock does not switch back to internal if external clock dies).*

### 11.3.2 External Oscillator

The external clock option is designed for use when a clock signal is available in the application to provide a clock source to the microcontroller. The OSC1 pin is enabled as an input by the oscillator module. The clock signal is used directly to create BUSCLKX4 and also divided by two to create BUSCLKX2.

In this configuration, the OSC2 pin cannot output BUSCLKX4.So the OSC2EN bit in the port A pullup enable register will be clear to enable PTA4 I/O functions on the pin

### 11.3.3 XTAL Oscillator

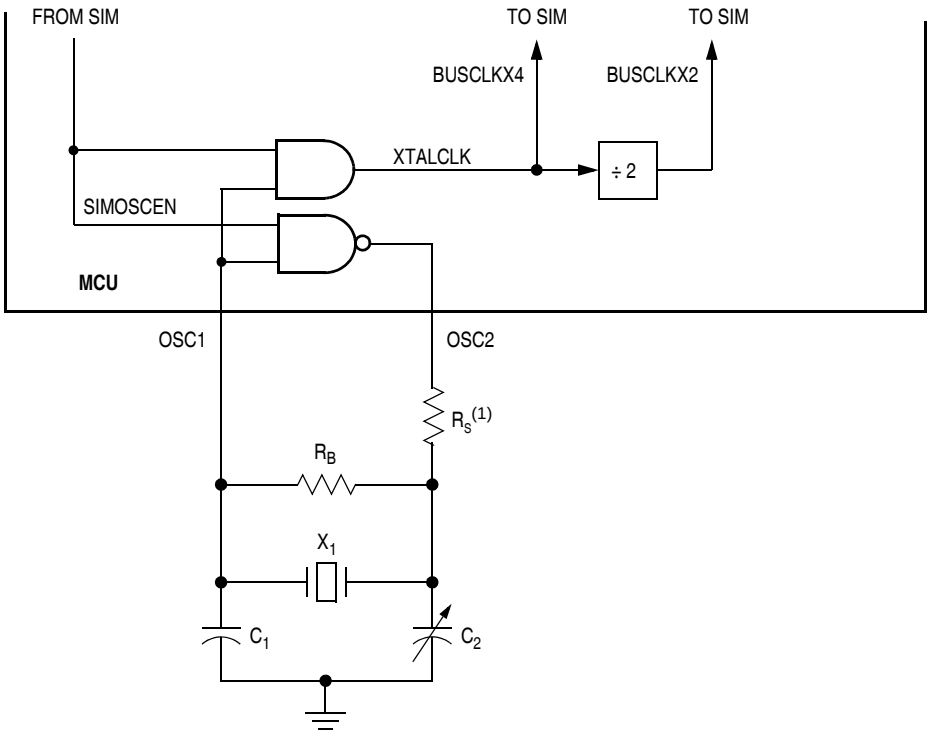
The XTAL oscillator circuit is designed for use with an external crystal or ceramic resonator to provide an accurate clock source. In this configuration, the OSC2 pin is dedicated to the external crystal circuit. The OSC2EN bit in the port A pullup enable register has no effect when this clock mode is selected.

In its typical configuration, the XTAL oscillator is connected in a Pierce oscillator configuration, as shown in Figure 11-2. 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)

**NOTE**

*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.*




Note 1.
$R_S$ can be zero (shorted) when used with higher-frequency crystals. Refer to manufacturer's data. See Chapter 16 Electrical Specifications for component value recommendations.

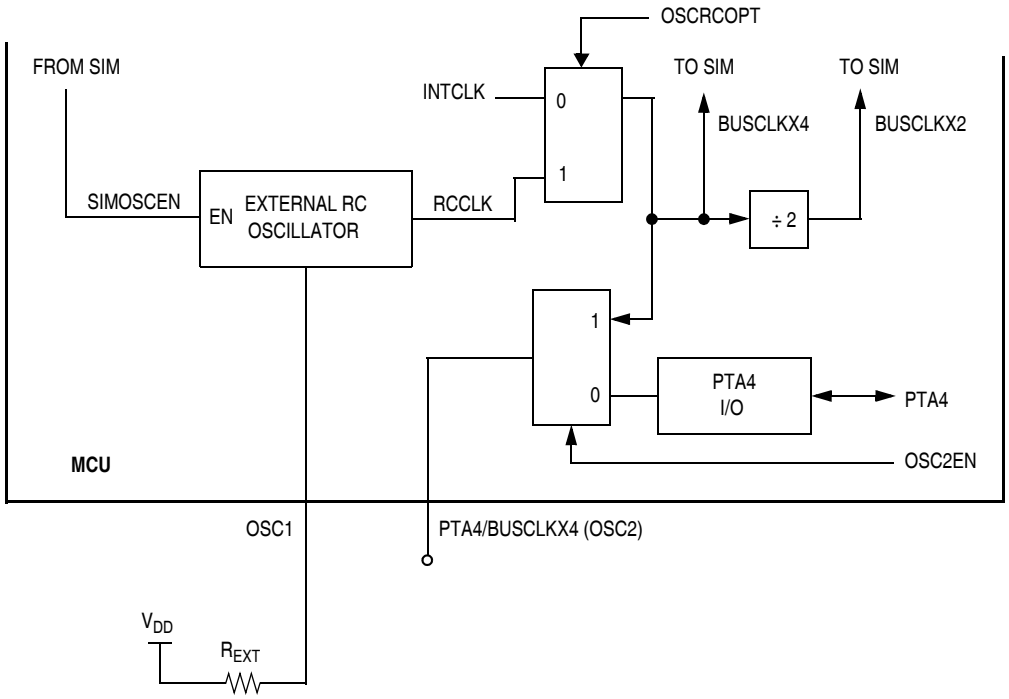
**Figure 11-2. XTAL Oscillator External Connections**

### 11.3.4 RC Oscillator

The RC oscillator circuit is designed for use with an external resistor ($R_{EXT}$) to provide a clock source with a tolerance within 25% of the expected frequency. See Figure 11-3.

The capacitor (C) for the RC oscillator is internal to the MCU. The $R_{EXT}$ value must have a tolerance of 1% or less to minimize its effect on the frequency.

In this configuration, the OSC2 pin can be left in the reset state as PTA4. Or, the OSC2EN bit in the port A pullup enable register can be set to enable the OSC2 output function on the pin. Enabling the OSC2 output slightly increases the external RC oscillator frequency, $f_{RCCLK}$.




See Chapter 16 Electrical Specifications for component value requirements.

**Figure 11-3. RC Oscillator External Connections**

## 11.4 Oscillator Module Signals

The following paragraphs describe the signals that are inputs to and outputs from the oscillator module.

### 11.4.1 Crystal Amplifier Input Pin (OSC1)

The OSC1 pin is either an input to the crystal oscillator amplifier, an input to the RC oscillator circuit, or an external clock source.

For the internal oscillator configuration, the OSC1 pin can assume other functions according to Table 1-3. Function Priority in Shared Pins.

### 11.4.2 Crystal Amplifier Output Pin (OSC2/PTA4/BUSCLKX4)

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

For the external clock option, the OSC2 pin is dedicated to the PTA4 I/O function. The OSC2EN bit has no effect.

For the internal oscillator or RC oscillator options, the OSC2 pin can assume other functions according to Table 1-3. Function Priority in Shared Pins, or the output of the oscillator clock (BUSCLKX4).

**Table 11-1. OSC2 Pin Function**

| Option | OSC2 Pin Function |
|---|---|
| XTAL oscillator | Inverting OSC1 |
| External clock | PTA4 I/O |
| Internal oscillator or RC oscillator | Controlled by OSC2EN bit in PTAPUE register<br>OSC2EN = 0: PTA4 I/O<br>OSC2EN = 1: BUSCLKX4 output |

### 11.4.3 Oscillator Enable Signal (SIMOSCEN)

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

### 11.4.4 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 11-2 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.

### 11.4.5 RC Oscillator Clock (RCCLK)

RCCLK is the RC oscillator output signal. Its frequency is directly proportional to the time constant of external R and internal C. Figure 11-3 shows only the logical relation of RCCLK to OSC1 and may not represent the actual circuitry.

### 11.4.6 Internal Oscillator Clock (INTCLK)

INTCLK is the internal oscillator output signal. Its nominal frequency is fixed to 12.8 MHz, but it can be also trimmed using the oscillator trimming feature of the OSCTRIM register (see 11.3.1.1 Internal Oscillator Trimming).

### 11.4.7 Oscillator Out 2 (BUSCLKX4)

BUSCLKX4 is the same as the input clock (XTALCLK, RCCLK, or INTCLK). This signal is driven to the SIM module and is used to determine the COP cycles.

### 11.4.8 Oscillator Out (BUSCLKX2)

The frequency of this signal is equal to half of the BUSCLKX4, this signal is driven to the SIM for generation of the bus clocks used by the CPU and other modules on the MCU. BUSCLKX2 will be divided

again in the SIM and results in the internal bus frequency being one fourth of either the XTALCLK, RCCLK, or INTCLK frequency.

## 11.5 Low Power Modes

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

### 11.5.1 Wait Mode

The WAIT instruction has no effect on the oscillator logic. BUSCLKX2 and BUSCLKX4 continue to drive to the SIM module.

### 11.5.2 Stop Mode

The STOP instruction disables either the XTALCLK, the RCCLK, or INTCLK output, hence BUSCLKX2 and BUSCLKX4.

## 11.6 Oscillator During Break Mode

The oscillator continues to drive BUSCLKX2 and BUSCLKX4 when the device enters the break state.

## 11.7 CONFIG2 Options

Two CONFIG2 register options affect the operation of the oscillator module: OSCOPT1 and OSCOPT0. All CONFIG2 register bits will have a default configuration. Refer to Chapter 5 Configuration Register (CONFIG) for more information on how the CONFIG2 register is used.

Table 11-2 shows how the OSCOPT bits are used to select the oscillator clock source.

**Table 11-2. Oscillator Modes**

| OSCOPT1 | OSCOPT0 | Oscillator Modes |
|---|---|---|
| 0 | 0 | Internal oscillator |
| 0 | 1 | External oscillator |
| 1 | 0 | External RC |
| 1 | 1 | External crystal |

## 11.8 Input/Output (I/O) Registers

The oscillator module contains these two registers:

1. Oscillator status register (OSCSTAT)
2. Oscillator trim register (OSCTRIM)

### 11.8.1 Oscillator Status Register

The oscillator status register (OSCSTAT) contains the bits for switching from internal to external clock sources.

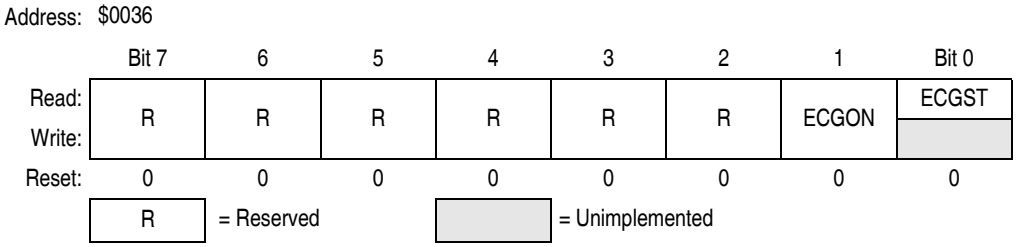



**Figure 11-4. Oscillator Status Register (OSCSTAT)**

**ECGON — External Clock Generator On Bit**

This read/write bit enables external clock generator, so that the switching process can be initiated. This bit is forced low during reset. This bit is ignored in monitor mode with the internal oscillator bypassed, PTM or CTM mode.

1 = External clock generator enabled
0 = External clock generator disabled

**ECGST — External Clock Status Bit**

This read-only bit indicates whether or not an external clock source is engaged to drive the system clock.

1 = An external clock source engaged
0 = An external clock source disengaged

### 11.8.2 Oscillator Trim Register (OSCTRIM)

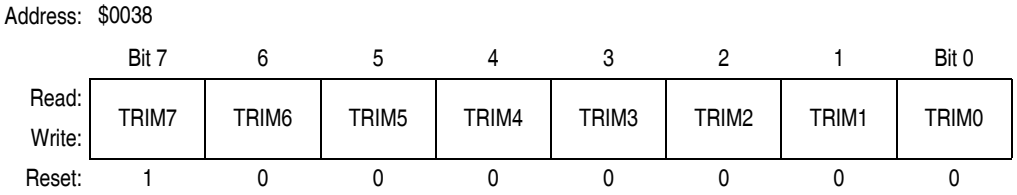



**Figure 11-5. Oscillator Trim Register (OSCTRIM)**

**TRIM7–TRIM0 — Internal Oscillator Trim Factor Bits**

These read/write bits change the size of the internal capacitor used by the internal oscillator. By measuring the period of the internal clock and adjusting this factor accordingly, the frequency of the internal clock can be fine tuned. Increasing (decreasing) this factor by one increases (decreases) the period by approximately 0.2% of the untrimmed period (the period for TRIM = $80). The trimmed frequency is guaranteed not to vary by more than ±5% over the full specified range of temperature and voltage. The reset value is $80, which sets the frequency to 12.8 MHz (3.2 MHz bus speed) ±25%.

Applications using the internal oscillator should copy the internal oscillator trim value at location $FFC0 or $FFC1 into this register to trim the clock source.

# Chapter 12
# Input/Output Ports (PORTS)

## 12.1 Introduction

The MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 have five bidirectional input-output (I/O) pins and one input only pin. The MC68HC908QY1, MC68HC908QY2, and MC68HC908QY4 have thirteen bidirectional pins and one input only pin. 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.*

*8-pin devices have non-bonded pins. These pins should be configured either as outputs driving low or high, or as inputs with internal pullups enabled. Configuring these non-bonded pins in this manner will prevent any excess current consumption caused by floating inputs.*

## 12.2 Port A

Port A is a 6-bit special function port that shares all six of its pins with the keyboard interrupt (KBI) module (see Chapter 9 Keyboard Interrupt Module (KBI)). Each port A pin also has a software configurable pullup device if the corresponding port pin is configured as an input port.

***NOTE***

*PTA2 is input only.*

*When the $\overline{IRQ}$ function is enabled in the configuration register 2 (CONFIG2), bit 2 of the port A data register (PTA) will always read a 0. In this case, the BIH and BIL instructions can be used to read the logic level on the PTA2 pin. When the $\overline{IRQ}$ function is disabled, these instructions will behave as if the PTA2 pin is a logic 1. However, reading bit 2 of PTA will read the actual logic level on the pin.*

### 12.2.1 Port A Data Register

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

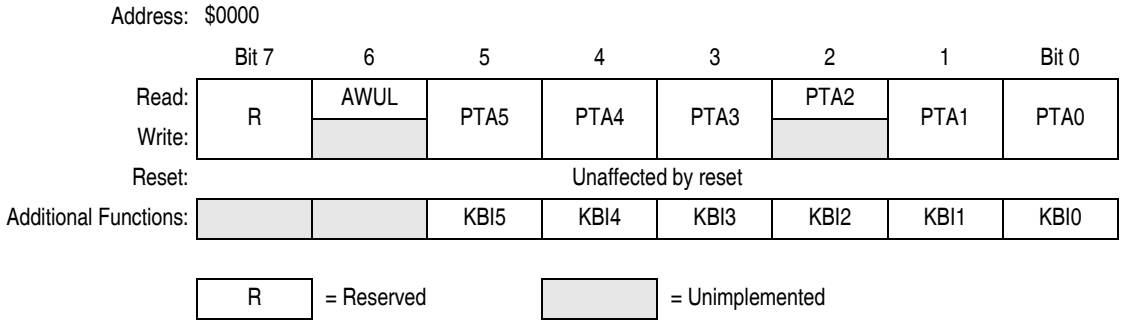



**Figure 12-1. Port A Data Register (PTA)**

**PTA[5:0] — 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.

**AWUL — Auto Wakeup Latch Data Bit**

This is a read-only bit which has the value of the auto wakeup interrupt request latch. The wakeup request signal is generated internally (see Chapter 4 Auto Wakeup Module (AWU)). There is no PTA6 port nor any of the associated bits such as PTA6 data register, pullup enable or direction.

**KBI[5:0] — Port A Keyboard Interrupts**

The keyboard interrupt enable bits, KBIE5–KBIE0, in the keyboard interrupt control enable register (KBIER) enable the port A pins as external interrupt pins (see Chapter 9 Keyboard Interrupt Module (KBI)).

### 12.2.2 Data Direction Register A

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

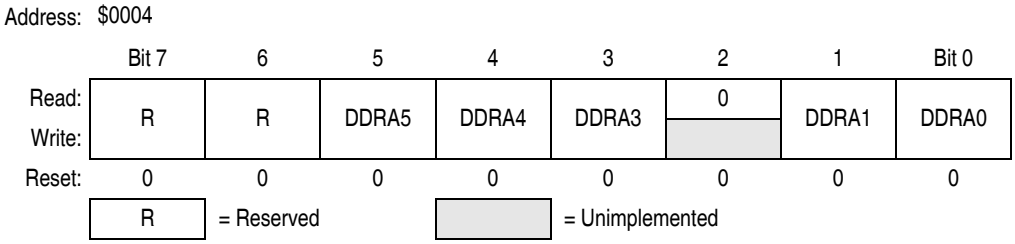



**Figure 12-2. Data Direction Register A (DDRA)**

**DDRA[5:0] — Data Direction Register A Bits**

These read/write bits control port A data direction. Reset clears DDRA[5:0], 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 12-3 shows the port A I/O logic.

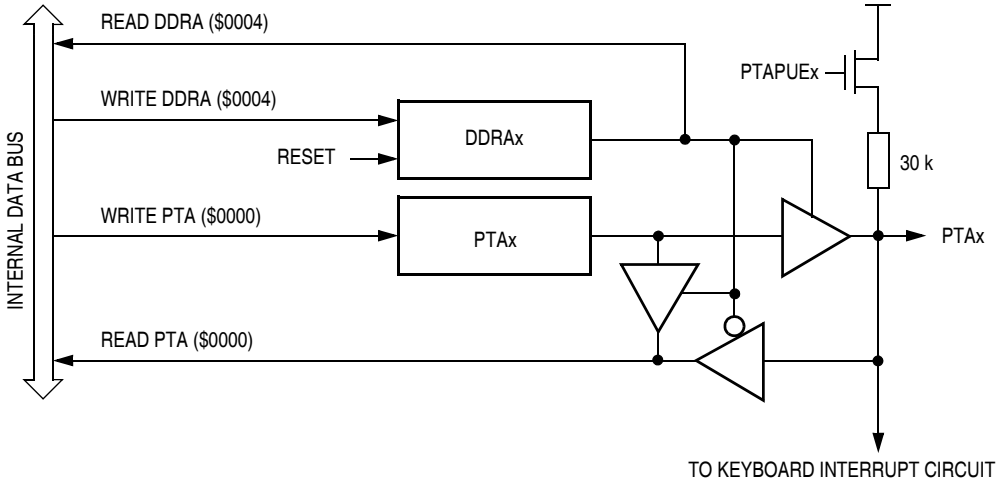



**Figure 12-3. Port A I/O Circuit**

***NOTE***

*Figure 12-3 does not apply to PTA2*

When DDRAx is a 1, reading address $0000 reads the PTAx data latch. When DDRAx is a 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.

### 12.2.3 Port A Input Pullup Enable Register

The port A input pullup enable register (PTAPUE) contains a software configurable pullup device for each if the six port A pins. Each bit is individually configurable and requires the corresponding data direction register, DDRAx, to be configured as input. Each pullup device is automatically and dynamically disabled when its corresponding DDRAx bit is configured as output.

Address: $000B

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read:<br>Write: | OSC2EN | | PTAPUE5 | PTAPUE4 | PTAPUE3 | PTAPUE2 | PTAPUE1 | PTAPUE0 |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 12-4. Port A Input Pullup Enable Register (PTAPUE)**

**OSC2EN — Enable PTA4 on OSC2 Pin**

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

1 = OSC2 pin outputs the internal or RC oscillator clock (BUSCLKX4)

0 = OSC2 pin configured for PTA4 I/O, having all the interrupt and pullup functions

### PTAPUE[5:0] — Port A Input Pullup Enable Bits

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

1 = Corresponding port A pin configured to have internal pull if its DDRA bit is set to 0

0 = Pullup device is disconnected on the corresponding port A pin regardless of the state of its DDRA bit

Table 12-1 summarizes the operation of the port A pins.

**Table 12-1. 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) | DDRA5–DDRA0 | Pin | PTA5–PTA0(3) |
| 0 | 0 | X | Input, Hi-Z(4) | DDRA5–DDRA0 | Pin | PTA5–PTA0(3) |
| X | 1 | X | Output | DDRA5–DDRA0 | PTA5–PTA0 | PTA5–PTA0(5) |

1. X = don't care
2. I/O pin pulled to $V_{DD}$ by internal pullup.
3. Writing affects data register, but does not affect input.
4. Hi-Z = high impedance
5. Output does not apply to PTA2

## 12.3 Port B

Port B is an 8-bit general purpose I/O port. Port B is only available on the MC68HC908QY1, MC68HC908QY2, and MC68HC908QY4.

### 12.3.1 Port B Data Register

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

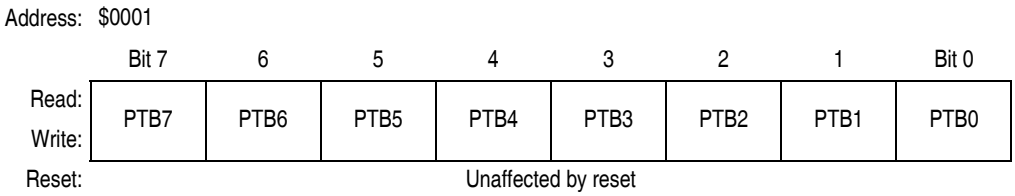



**Figure 12-5. Port B Data Register (PTB)**

### 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.

### 12.3.2 Data Direction Register B

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

Address: $0005

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

**Figure 12-6. 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 12-7 shows the port B I/O logic.*

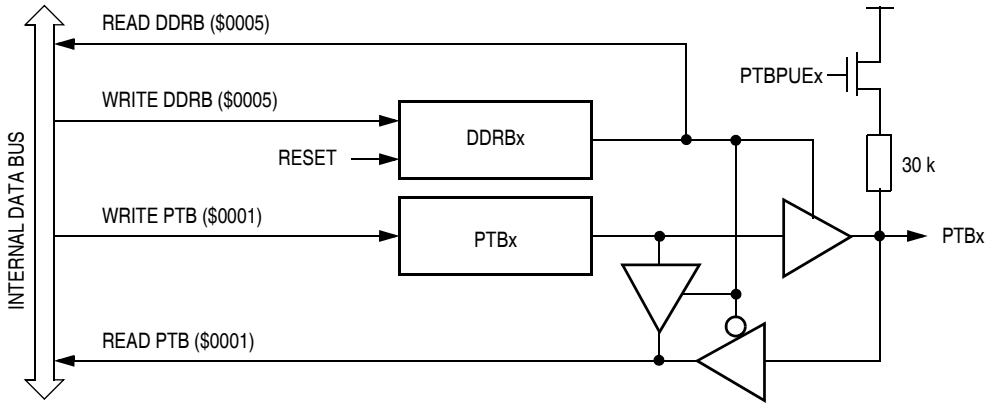



**Figure 12-7. Port B I/O Circuit**

When DDRBx is a 1, reading address $0001 reads the PTBx data latch. When DDRBx is a 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 12-2 summarizes the operation of the port B pins.

**Table 12-2. 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) | DDRB7–DDRB0 | Pin | PTB7–PTB0(3) |
| 1 | X | Output | DDRB7–DDRB0 | Pin | PTB7–PTB0 |

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

### 12.3.3 Port B Input Pullup Enable Register

The port B input pullup enable register (PTBPUE) contains a software configurable pullup device for each of the eight port B pins. Each bit is individually configurable and requires the corresponding data direction register, DDRBx, be configured as input. Each pullup device is automatically and dynamically disabled when its corresponding DDRBx bit is configured as output.

Address: $000C

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: Write: | PTBPUE7 | PTBPUE6 | PTBPUE5 | PTBPUE4 | PTBPUE3 | PTBPUE2 | PTBPUE2 | PTBPUE0 |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-8. Port B Input Pullup Enable Register (PTBPUE)**

**PTBPUE[7:0] — Port B Input Pullup Enable Bits**

These read/write bits are software programmable to enable pullup devices on port B pins

1 = Corresponding port B pin configured to have internal pull if its DDRB bit is set to 0

0 = Pullup device is disconnected on the corresponding port B pin regardless of the state of its DDRB bit.

Table 12-3 summarizes the operation of the port B pins.

**Table 12-3. Port B Pin Functions**

| PTBPUE Bit | DDRB Bit | PTB Bit | I/O Pin Mode | Accesses to DDRB | Accesses to PTB | |
|---|---|---|---|---|---|---|
| | | | | Read/Write | Read | Write |
| 1 | 0 | X(1) | Input, $V_{DD}$(2) | DDRB7–DDRB0 | Pin | PTB7–PTB0(3) |
| 0 | 0 | X | Input, Hi-Z(4) | DDRB7–DDRB0 | Pin | PTB7–PTB0(3) |
| X | 1 | X | Output | DDRB7–DDRB0 | PTB7–PTB0 | PTB7–PTB0 |

1. X = don't care
2. I/O pin pulled to $V_{DD}$ by internal pullup.
3. Writing affects data register, but does not affect input.
4. Hi-Z = high impedance

# Chapter 13
# System Integration Module (SIM)

## 13.1 Introduction

This section describes the system integration module (SIM), which supports up to 24 external and/or internal interrupts. Together with the central processor unit (CPU), the SIM controls all microcontroller unit (MCU) activities. A block diagram of the SIM is shown in Figure 13-1. 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 computer operating properly (COP) timeout
- Interrupt control:
  - Acknowledge timing
  - Arbitration control timing
  - Vector address generation
- CPU enable/disable timing

**Table 13-1. Signal Name Conventions**

| Signal Name | Description |
|---|---|
| BUSCLKX4 | Buffered clock from the internal, RC or XTAL oscillator circuit. |
| BUSCLKX2 | The BUSCLKX4 frequency divided by two. This signal is again divided by two in the SIM to generate the internal bus clocks (bus clock = BUSCLKX4 ÷ 4). |
| Address bus | Internal address bus |
| Data bus | Internal data bus |
| PORRST | Signal from the power-on reset module to the SIM |
| IRST | Internal reset signal |
| R/$\overline{W}$ | Read/write signal |

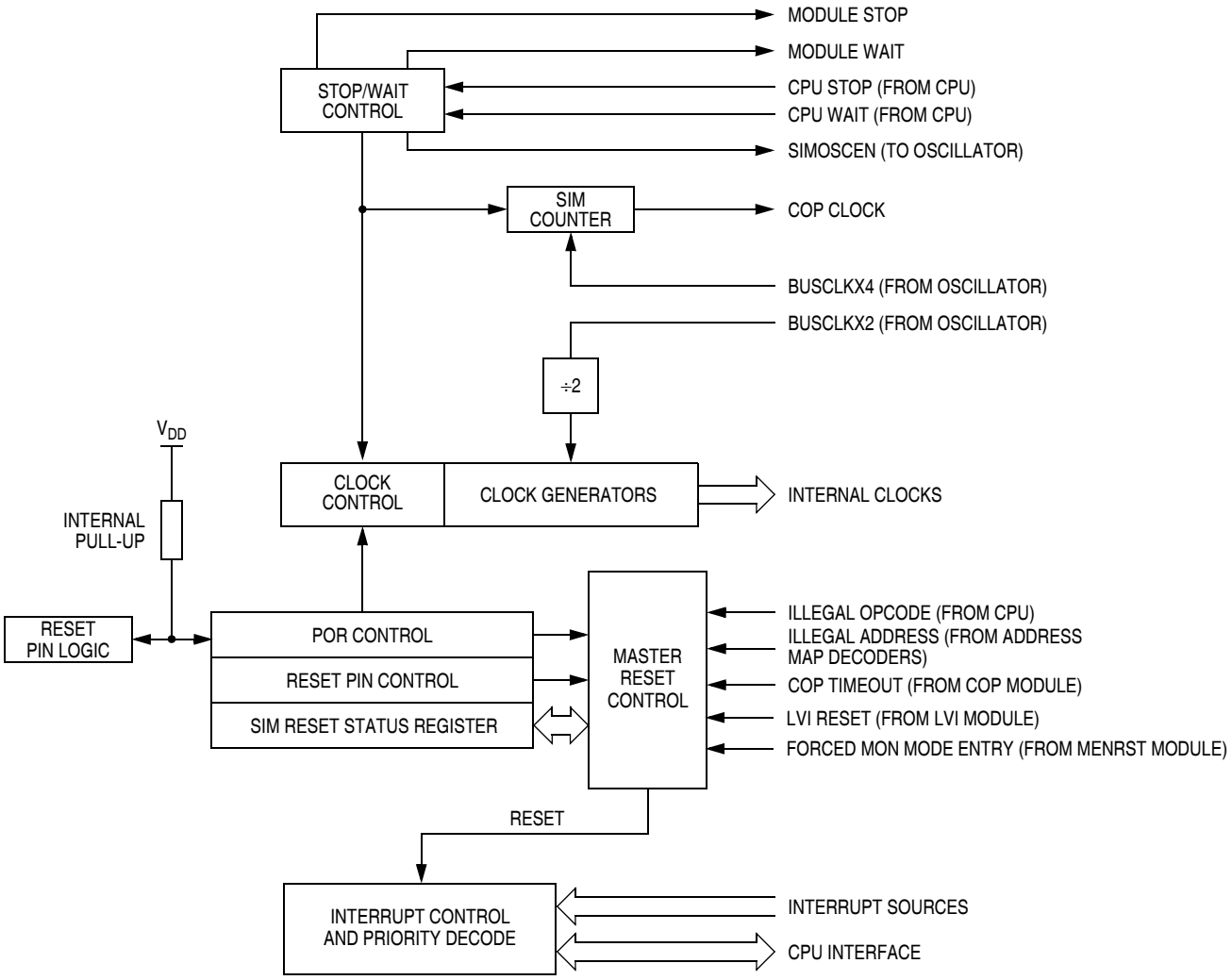



**Figure 13-1. SIM Block Diagram**

## 13.2 $\overline{\text{RST}}$ and $\overline{\text{IRQ}}$ Pins Initialization

$\overline{\text{RST}}$ and $\overline{\text{IRQ}}$ pins come out of reset as PTA3 and PTA2 respectively. $\overline{\text{RST}}$ and $\overline{\text{IRQ}}$ functions can be activated by programing CONFIG2 accordingly. Refer to Chapter 5 Configuration Register (CONFIG).

## 13.3 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, BUSCLKX2, as shown in Figure 13-2.

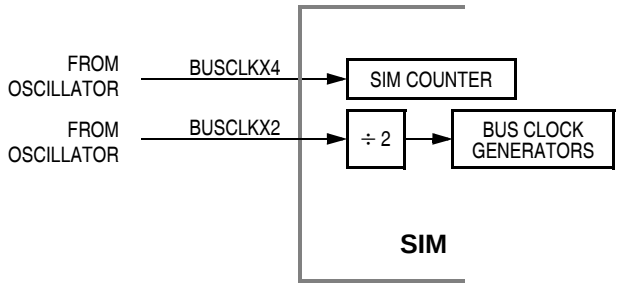



**Figure 13-2. SIM Clock Signals**

### 13.3.1 Bus Timing

In user mode**,** the internal bus frequency is the oscillator frequency (BUSCLKX4) divided by four.

### 13.3.2 Clock Start-Up from POR

When the power-on reset module generates a reset, the clocks to the CPU and peripherals are inactive and held in an inactive phase until after the 4096 BUSCLKX4 cycle POR time out has completed. The IBUS clocks start upon completion of the time out.

### 13.3.3 Clocks in Stop Mode and Wait Mode

Upon exit from stop mode by an interrupt or reset, the SIM allows BUSCLKX4 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 BUSCLKX4 cycles. See 13.7.2 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.

## 13.4 Reset and System Initialization

The MCU has these reset sources:

- Power-on reset module (POR)
- External reset pin ($\overline{\text{RST}}$)
- 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 13.5 SIM Counter), but an external reset does not. Each of the resets sets a corresponding bit in the SIM reset status register (SRSR). See 13.8 SIM Registers.

### 13.4.1 External Pin Reset

The $\overline{RST}$ pin circuits include an internal pullup device. Pulling the asynchronous $\overline{RST}$ pin low halts all processing. The PIN bit of the SIM reset status register (SRSR) is set as long as $\overline{RST}$ is held low for at least the minimum $t_{RL}$ time. Figure 13-3 shows the relative timing. The $\overline{RST}$ pin function is only available if the RSTEN bit is set in the CONFIG2 register.

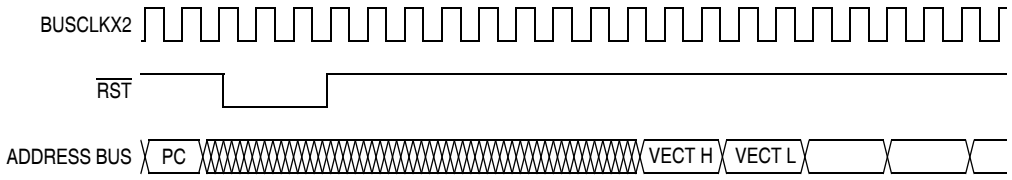



**Figure 13-3. External Reset Timing**

### 13.4.2 Active Resets from Internal Sources

The $\overline{RST}$ pin is initially setup as a general-purpose input after a POR. Setting the RSTEN bit in the CONFIG2 register enables the pin for the reset function. This section assumes the RSTEN bit is set when describing activity on the $\overline{RST}$ pin.

***NOTE***

*For POR and LVI resets, the SIM cycles through 4096 BUSCLKX4 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 13-4.*

*The COP reset is asynchronous to the bus clock.*

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

All internal reset sources actively pull the $\overline{RST}$ pin low for 32 BUSCLKX4 cycles to allow resetting of external peripherals. The internal reset signal IRST continues to be asserted for an additional 32 cycles (see Figure 13-4). An internal reset can be caused by an illegal address, illegal opcode, COP time out, LVI, or POR (see Figure 13-5).

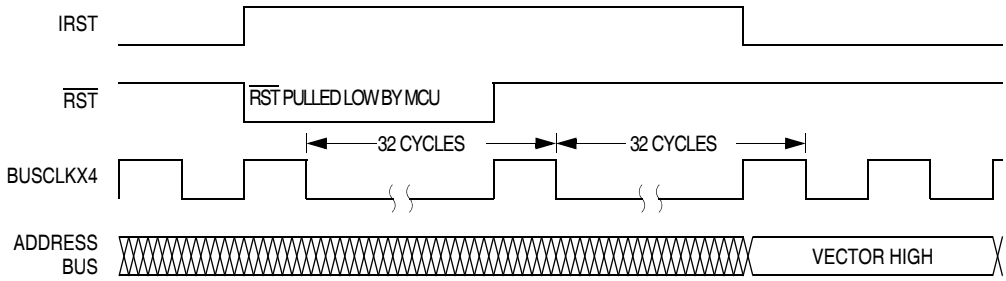



**Figure 13-4. Internal Reset Timing**

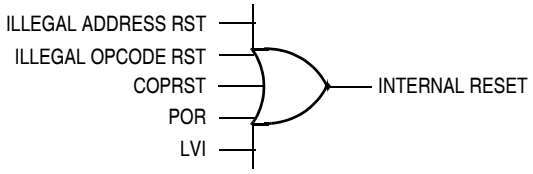



**Figure 13-5. Sources of Internal Reset**

**Table 13-2. Reset Recovery Timing**

| Reset Recovery Type | Actual Number of Cycles |
|---|---|
| POR/LVI | 4163 (4096 + 64 + 3) |
| All others | 67 (64 + 3) |

### *13.4.2.1 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 SIM counter counts out 4096 BUSCLKX4 cycles. Sixty-four BUSCLKX4 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 the oscillator to drive BUSCLKX4.
- Internal clocks to the CPU and modules are held inactive for 4096 BUSCLKX4 cycles to allow stabilization of the oscillator.
- The POR bit of the SIM reset status register (SRSR) is set

See Figure 13-6.

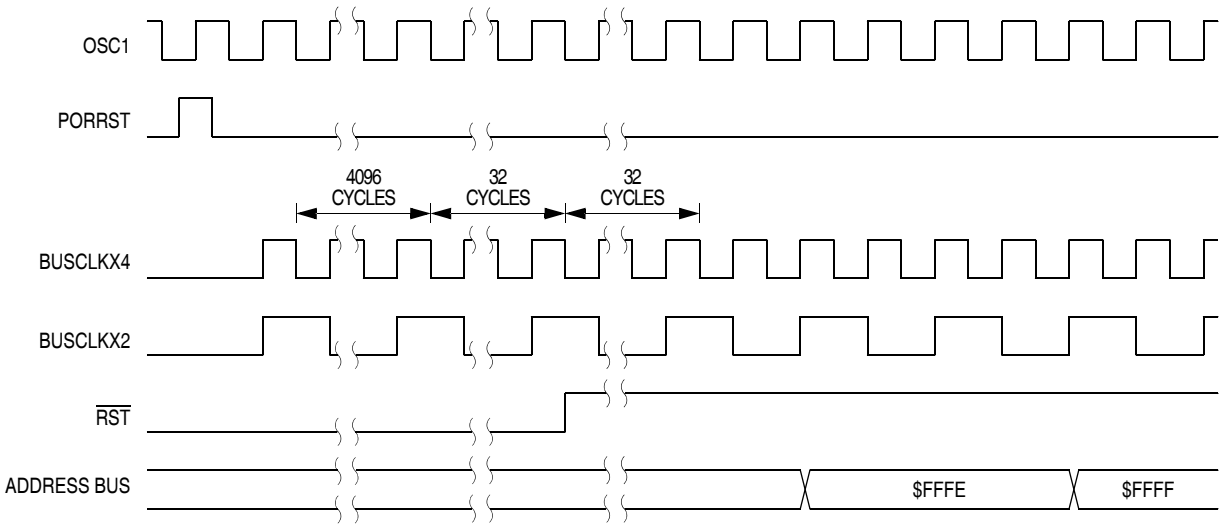



**Figure 13-6. POR Recovery**

#### 13.4.2.2 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 SIM reset status register (SRSR). 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 12–5 of the SIM counter. The SIM counter output, which occurs at least every 4080 BUSCLKX4 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 during a break interrupt with monitor mode when BDCOP bit is set in break auxiliary register (BRKAR).

#### 13.4.2.3 Illegal Opcode Reset

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

If the stop enable bit, STOP, in the mask option register is 0, 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.

#### 13.4.2.4 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 SIM reset status register (SRSR) 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. See Figure 2-1. Memory Map for memory ranges.

#### 13.4.2.5 Low-Voltage Inhibit (LVI) Reset

The LVI asserts its output to the SIM when the $V_{DD}$ voltage falls to the LVI trip voltage $V_{TRIPF}$. The LVI bit in the SIM reset status register (SRSR) is set, and the external reset pin ($\overline{\text{RST}}$) is held low while the SIM counter counts out 4096 BUSCLKX4 cycles after $V_{DD}$ rises above $V_{TRIPR}$. Sixty-four BUSCLKX4 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.

## 13.5 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 BUSCLKX4.

### 13.5.1 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.

### 13.5.2 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 configuration register 1 (CONFIG1). If the SSREC bit is a 1, then the stop recovery is reduced from the normal delay of 4096 BUSCLKX4 cycles down to 32 BUSCLKX4 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).

### 13.5.3 SIM Counter and Reset States

External reset has no effect on the SIM counter (see 13.7.2 Stop Mode for details.) The SIM counter is free-running after all reset states. See 13.4.2 Active Resets from Internal Sources for counter control and internal reset recovery sequences.

## 13.6 Exception Control

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

1. Interrupts
   a. Maskable hardware CPU interrupts
   b. Non-maskable software interrupt instruction (SWI)
2. Reset
3. Break interrupts

### 13.6.1 Interrupts

An interrupt temporarily changes the sequence of program execution to respond to a particular event. Figure 13-7 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).

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 13-8 shows interrupt entry timing. Figure 13-9 shows interrupt recovery timing.

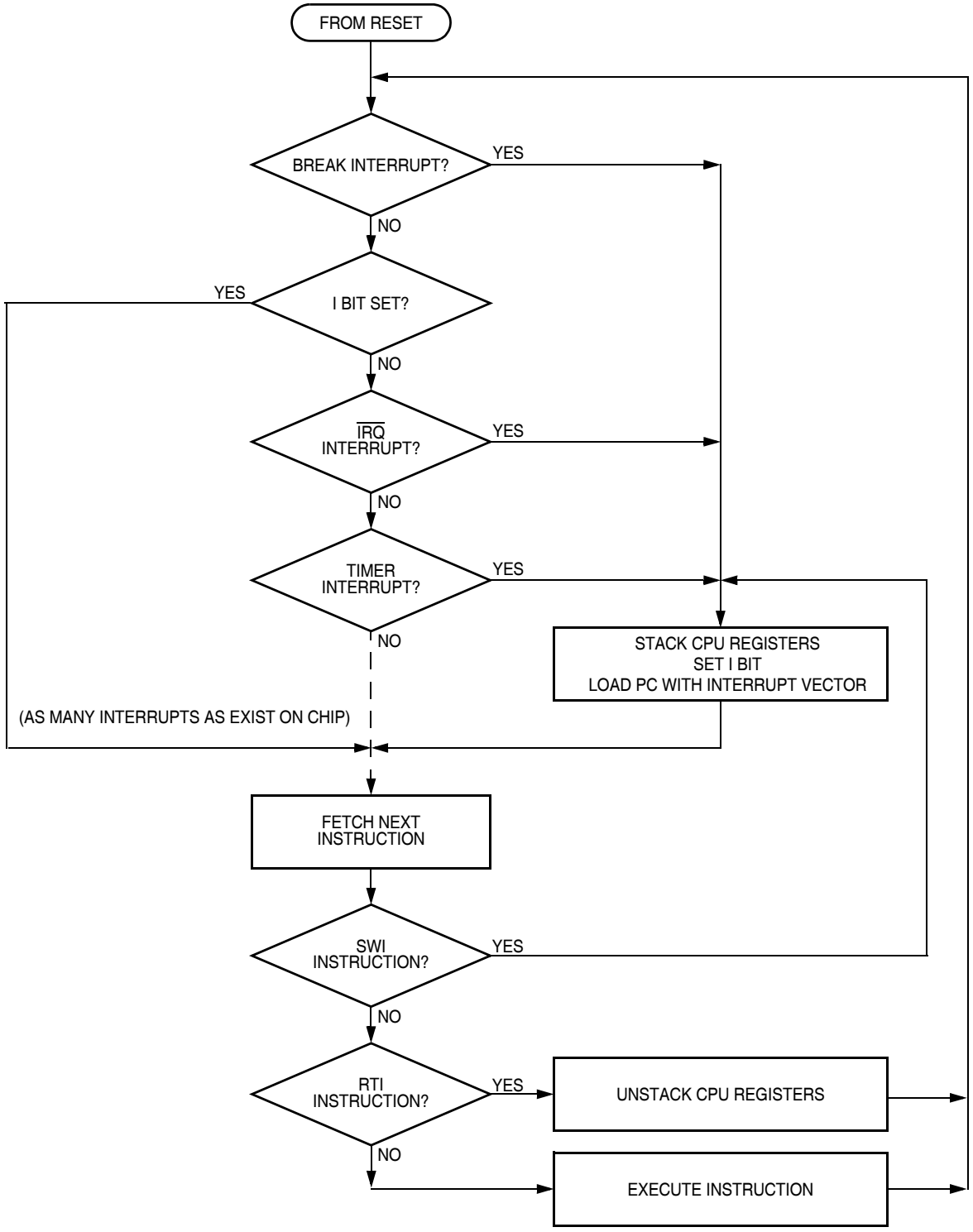



**Figure 13-7. Interrupt Processing**

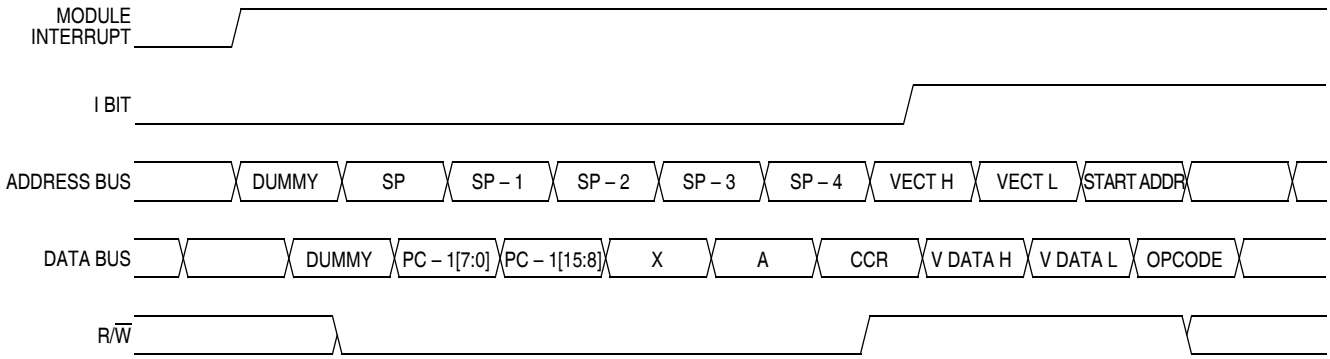



**Figure 13-8. Interrupt Entry**

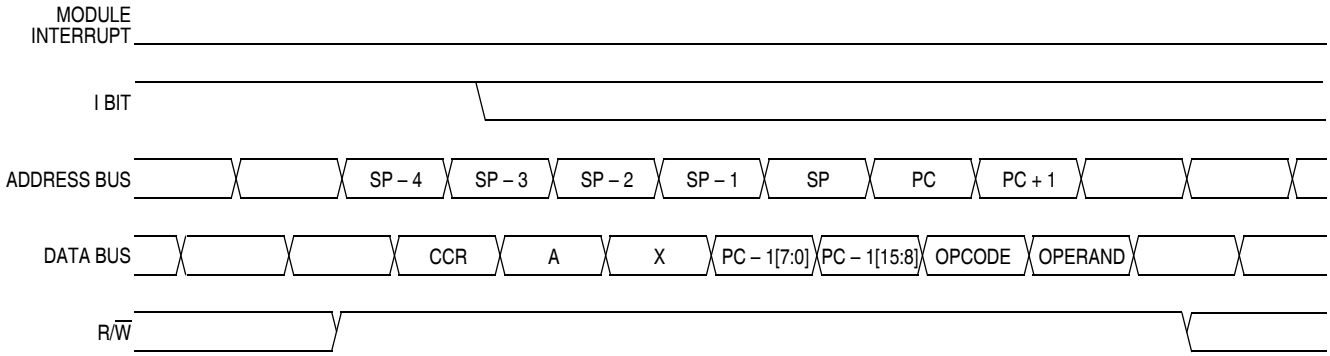



**Figure 13-9. Interrupt Recovery**

### *13.6.1.1 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 13-10 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.

The LDA opcode is prefetched by both the INT1 and INT2 return-from-interrupt (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.*

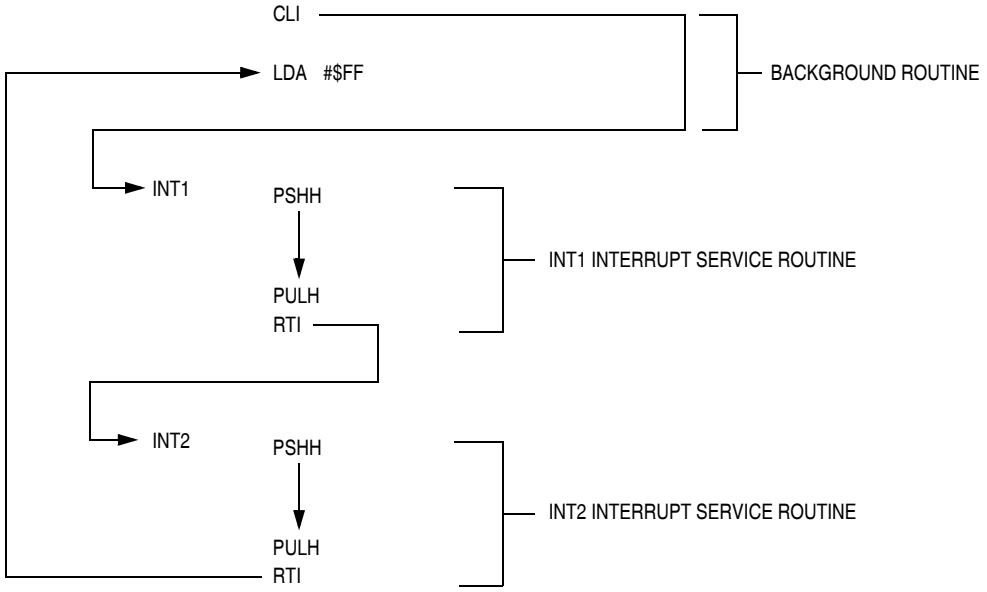



**Figure 13-10. Interrupt Recognition Example**

#### *13.6.1.2 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.*

### 13.6.2 Interrupt Status Registers

The flags in the interrupt status registers identify maskable interrupt sources. Table 13-3 summarizes the interrupt sources and the interrupt status register flags that they set. The interrupt status registers can be useful for debugging.

**Table 13-3. Interrupt Sources**

| Priority | Source | Flag | Mask[1] | INT Register Flag | Vector Address |
|---|---|---|---|---|---|
| Highest | Reset | — | — | — | $FFFE–$FFFF |
| | SWI instruction | — | — | — | $FFFC–$FFFD |
| | $\overline{\text{IRQ}}$ pin | IRQF | IMASK | IF1 | $FFFA–$FFFB |
| | Timer channel 0 interrupt | CH0F | CH0IE | IF3 | $FFF6–$FFF7 |
| | Timer channel 1 interrupt | CH1F | CH1IE | IF4 | $FFF4–$FFF5 |
| | Timer overflow interrupt | TOF | TOIE | IF5 | $FFF2–$FFF3 |
| | 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 interrupt sources except the SWI instruction.

### 13.6.2.1 Interrupt Status Register 1

Address: $FE04

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: | 0 | 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 13-11. Interrupt Status Register 1 (INT1)**

**IF1 and IF3–IF5 — Interrupt Flags**

These flags indicate the presence of interrupt requests from the sources shown in Table 13-3.

1 = Interrupt request present
0 = No interrupt request present

**Bit 0, 1, 3, and 7 — Always read 0**

### 13.6.2.2 Interrupt Status Register 2

Address: $FE05

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read: | IF14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Write: | R | R | R | R | R | R | R | R |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R = Reserved

**Figure 13-12. Interrupt Status Register 2 (INT2)**

**IF14 — Interrupt Flags**

This flag indicates the presence of interrupt requests from the sources shown in Table 13-3.

1 = Interrupt request present
0 = No interrupt request present

**Bit 0–6 — Always read 0**

### 13.6.2.3 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 13-13. Interrupt Status Register 3 (INT3)**

**IF15 — Interrupt Flags**

These flags indicate the presence of interrupt requests from the sources shown in Table 13-3.

1 = Interrupt request present
0 = No interrupt request present

**Bit 1–7 — Always read 0**

### 13.6.3 Reset

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

### 13.6.4 Break Interrupts

The break module can stop normal program flow at a software programmable break point by asserting its break interrupt output. (See Chapter 15 Development Support.) 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.

### 13.6.5 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.

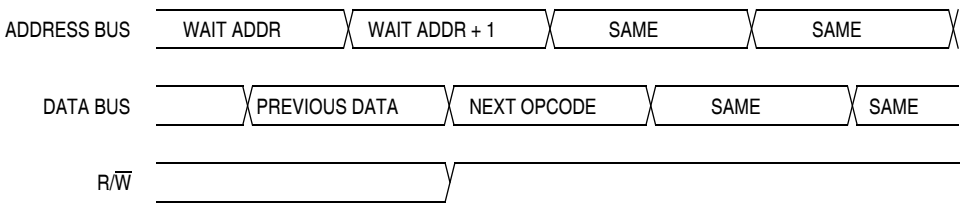
Setting the BCFE bit enables the clearing mechanisms. Once cleared in break mode, a flag remains cleared even when break mode is exited. Status flags with a two-step clearing mechanism — for example, a read of one register followed by the read or write of another — are protected, even when the first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

## 13.7 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.

### 13.7.1 Wait Mode

In wait mode, the CPU clocks are inactive while the peripheral clocks continue to run. Figure 13-14 shows the timing for wait mode entry.



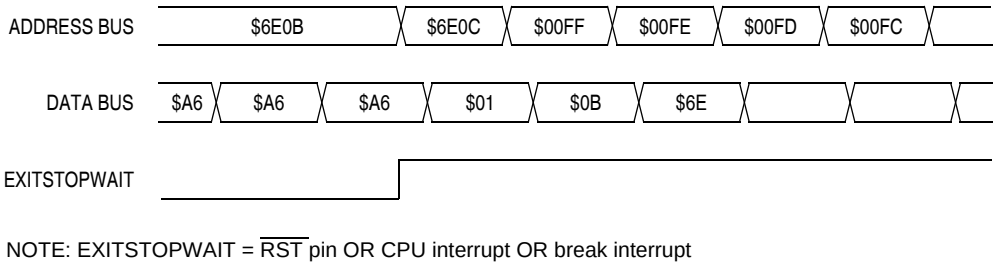

NOTE: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

**Figure 13-14. Wait Mode Entry Timing**

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 in emulation mode). 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 configuration register is 0, then the computer operating properly module (COP) is enabled and remains active in wait mode.

Figure 13-15 and Figure 13-16 show the timing for wait recovery.




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

**Figure 13-15. Wait Recovery from Interrupt**

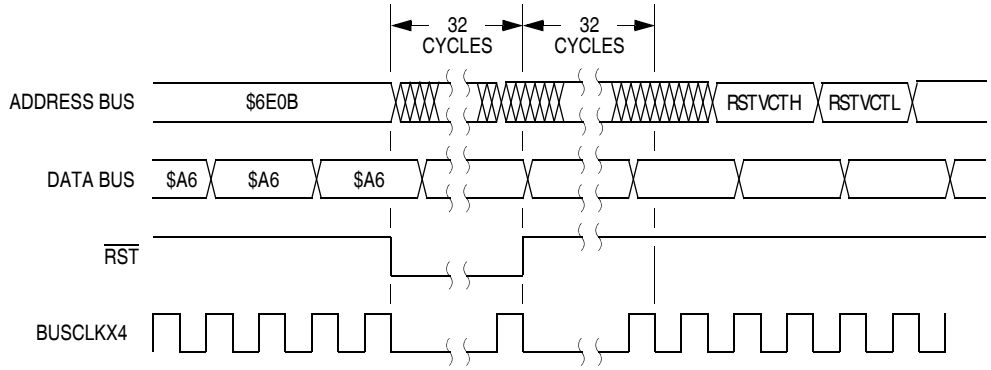



**Figure 13-16. Wait Recovery from Internal Reset**

### 13.7.2 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 (BUSCLKX2 and BUSCLKX4) 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 BUSCLKX4 cycles down to 32. This is ideal for the internal oscillator, RC oscillator, and external oscillator options which do not require long start-up times from stop mode.

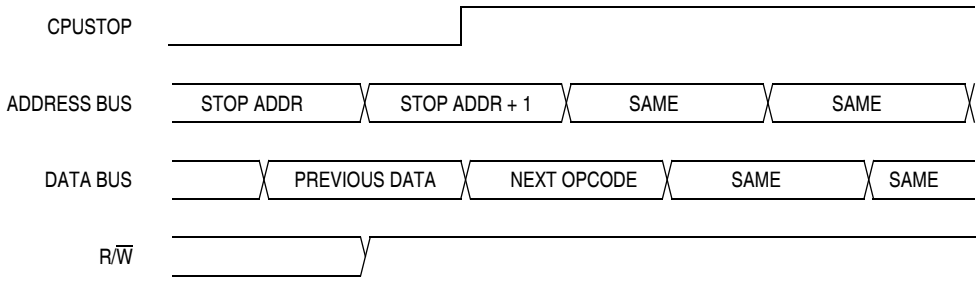
***NOTE***

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

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 13-17 shows stop mode entry timing and Figure 13-18 shows the stop mode recovery time from interrupt or break.

***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 13-17. Stop Mode Entry Timing**

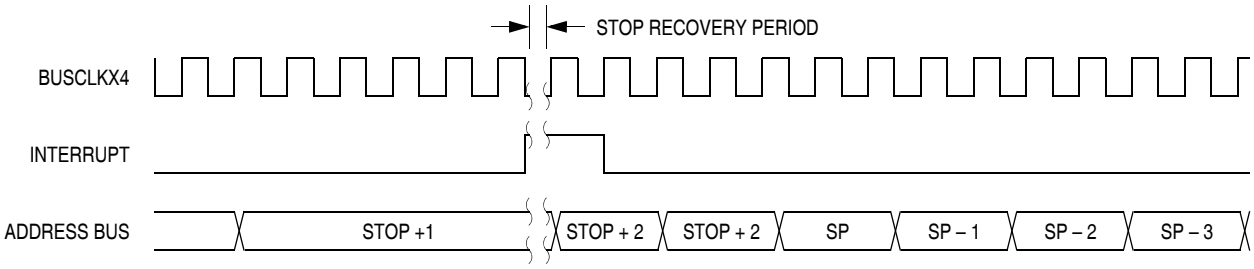



**Figure 13-18. Stop Mode Recovery from Interrupt**

## 13.8 SIM Registers

The SIM has three memory mapped registers. Table 13-4 shows the mapping of these registers.

**Table 13-4. SIM Registers**

| Address | Register | Access Mode |
|---|---|---|
| $FE00 | BSR | User |
| $FE01 | SRSR | User |
| $FE03 | BFCR | User |

### 13.8.1 SIM Reset Status Register

The SRSR register contains flags that show the source of the last reset. The status register will automatically clear after reading SRSR. A power-on reset sets the POR bit and clears all other bits in the register. All other reset sources set the individual flag bits but do not clear the register. More than one reset source can be flagged at any time depending on the conditions at the time of the internal or external reset. For example, the POR and LVI bit can both be set if the power supply has a slow rise time.

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 13-19. SIM Reset Status Register (SRSR)**

**POR — Power-On Reset Bit**

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

**PIN — External Reset Bit**

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

**COP — Computer Operating Properly Reset Bit**

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

**ILOP — Illegal Opcode Reset Bit**

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

**ILAD — Illegal Address Reset Bit (illegal attempt to fetch an opcode from an unimplemented address)**

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

**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 $\overline{IRQ} \neq V_{TST}$
0 = POR or read of SRSR

**LVI — Low Voltage Inhibit Reset Bit**

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

### 13.8.2 Break Flag Control Register

The break control register (BFCR) 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 13-20. 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 14
# Timer Interface Module (TIM)

## 14.1 Introduction

This section describes the timer interface module (TIM). The TIM is a two-channel timer that provides a timing reference with input capture, output compare, and pulse-width-modulation functions. Figure 14-2 is a block diagram of the TIM.

## 14.2 Features

Features of the TIM include the following:

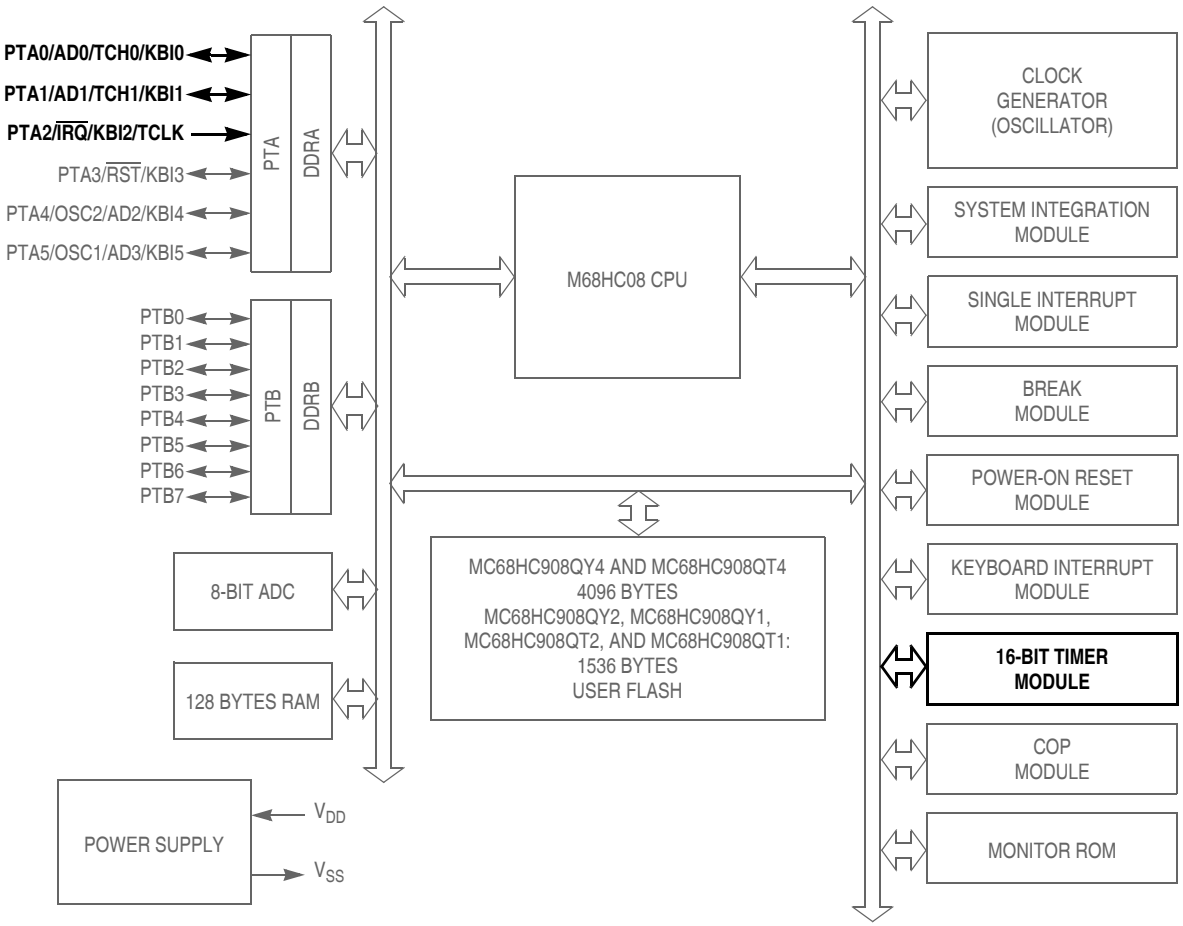
- Two input capture/output compare channels
  - Rising-edge, falling-edge, or any-edge input capture trigger
  - Set, clear, or toggle output compare action
- Buffered and unbuffered pulse width modulation (PWM) signal generation
- Programmable TIM clock input
  - 7-frequency internal bus clock prescaler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

## 14.3 Pin Name Conventions

The TIM shares two input/output (I/O) pins with two port A I/O pins. The full names of the TIM I/O pins are listed in Table 14-1. The generic pin name appear in the text that follows.

**Table 14-1. Pin Name Conventions**

| TIM Generic Pin Names: | TCH0 | TCH1 | TCLK |
|---|---|---|---|
| Full TIM Pin Names: | PTA0/TCH0 | PTA1/TCH1 | PTA2/TCLK |




RST, IRQ: Pins have internal (about 30K Ohms) pull up
**PTA[0:5]: High current sink and source capability**
**PTA[0:5]: Pins have programmable keyboard interrupt and pull up**
PTB[0:7]: Not available on 8-pin devices – MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 (see note in 12.1 Introduction)
ADC: Not available on the MC68HC908QY1 and MC68HC908QT1

**Figure 14-1. Block Diagram Highlighting TIM Block and Pins**

## 14.4 Functional Description

Figure 14-2 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 are programmable independently as input capture or output compare channels.

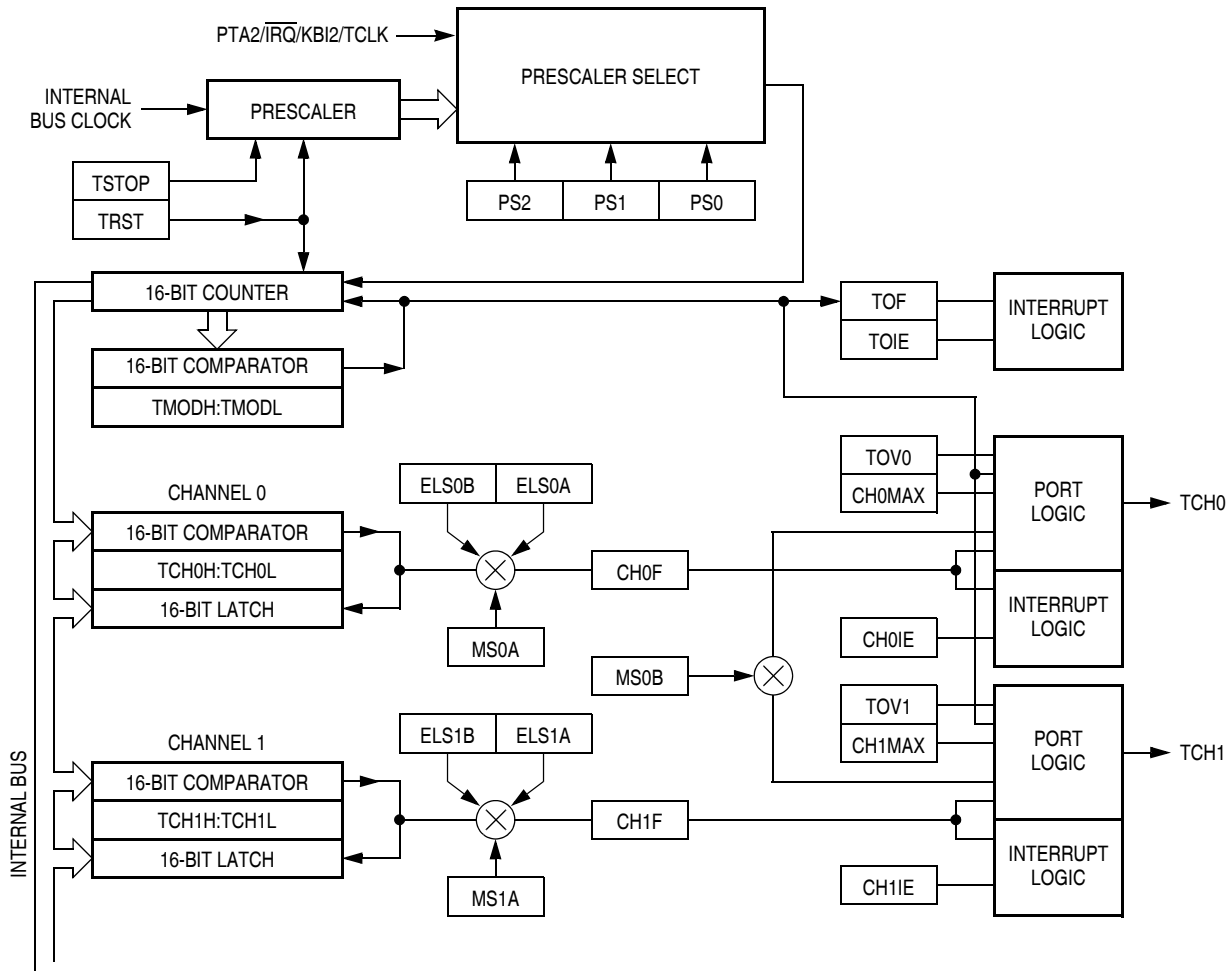



**Figure 14-2. TIM Block Diagram**

### 14.4.1 TIM Counter Prescaler

The TIM clock source is one of the seven prescaler outputs or the TIM clock pin, TCLK. 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 (TSC) select the TIM clock source.

### 14.4.2 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 central processor unit (CPU) interrupt requests.

### 14.4.3 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.

#### *14.4.3.1 Unbuffered Output Compare*

Any output compare channel can generate unbuffered output compare pulses as described in 14.4.3 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.

#### *14.4.3.2 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.*

### 14.4.4 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 14-3 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 (ELSxA = 0). Program the TIM to set the pin if the state of the PWM pulse is logic 0 (ELSxA = 1).

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 14.9.1 TIM Status and Control Register.

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%.

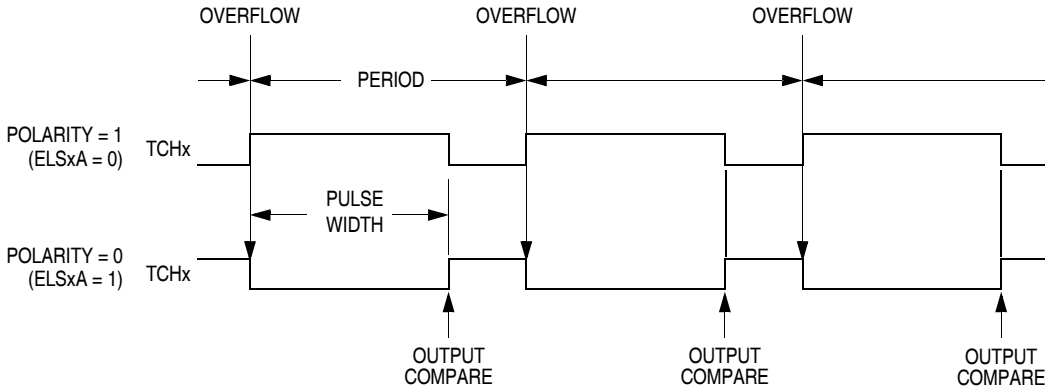



**Figure 14-3. PWM Period and Pulse Width**

#### 14.4.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in 14.4.4 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.*

#### 14.4.4.2 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.*

#### 14.4.4.3 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 14-3.
   b. Write 1 to the toggle-on-overflow bit, TOVx.
   c. Write 1:0 (polarity 1 — to clear output on compare) or 1:1 (polarity 0 — 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 14-3.

   ***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. MS0B takes priority over MS0A.

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 14.9.4 TIM Channel Status and Control Registers.

## 14.5 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.

## 14.6 Wait Mode

The WAIT instruction puts the MCU in low power-consumption standby 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.

## 14.7 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 13.8.2 Break Flag Control Register.

To allow software to clear status bits during a break interrupt, write a 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 0 to the BCFE bit. With BCFE at 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 two-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 0. After the break, doing the second step clears the status bit.

## 14.8 Input/Output Signals

Port A shares three of its pins with the TIM. Two TIM channel I/O pins are PTA0/TCH0 and PTA1/TCH1 and an alternate clock source is PTA2/TCLK.

### 14.8.1 TIM Clock Pin (PTA2/TCLK)

PTA2/TCLK is an external clock input that can be the clock source for the TIM counter instead of the prescaled internal bus clock. Select the PTA2/TCLK input by writing 1s to the three prescaler select bits, PS[2–0]. (See 14.9.1 TIM Status and Control Register.) When the PTA2/TCLK pin is the TIM clock input, it is an input regardless of port pin initialization.

### 14.8.2 TIM Channel I/O Pins (PTA0/TCH0 and PTA1/TCH1)

Each channel I/O pin is programmable independently as an input capture pin or an output compare pin. PTA0/TCH0 can be configured as a buffered output compare or buffered PWM pin.

## 14.9 Input/Output Registers

The following 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 and TSC1)
- TIM channel registers (TCH0H:TCH0L and TCH1H:TCH1L)

### 14.9.1 TIM Status and Control Register

The TIM status and control register (TSC) does the following:

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

Address: $0020

| | 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 14-4. 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 0 to TOF. If another TIM overflow occurs before the clearing sequence is complete, then writing 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 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. When the TSTOP bit is set and the timer is configured for input capture operation, input captures are inhibited until the TSTOP bit is cleared.*

*When using TSTOP to stop the timer counter, see if any timer flags are set. If a timer flag is set, it must be cleared by clearing TSTOP, then clearing the flag, then setting TSTOP again.*

### 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 a 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 either the PTA2/TCLK pin or one of the seven prescaler outputs as the input to the TIM counter as Table 14-2 shows. Reset clears the PS[2:0] bits.

**Table 14-2. 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 | PTA2/TCLK |

## 14.9.2 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.*

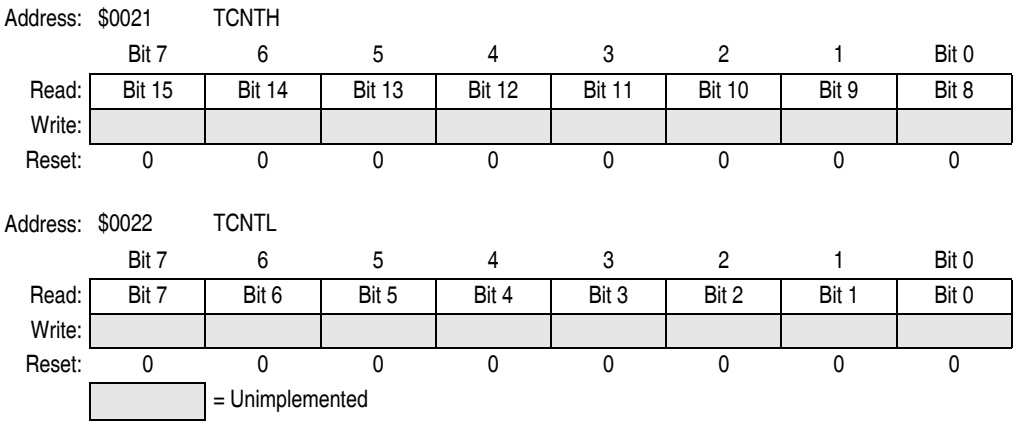



**Figure 14-5. TIM Counter Registers (TCNTH:TCNTL)**

### 14.9.3 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: $0023 TMODH

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

Address: $0024 TMODL

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read:<br>Write: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| Reset: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figure 14-6. TIM Counter Modulo Registers (TMODH:TMODL)**

***NOTE***

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

### 14.9.4 TIM Channel Status and Control Registers

Each of the TIM channel status and control registers does the following:

- 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: $0025 TSC0

| | 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 |

Address: $0028 TSC1

| | 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 |

= Unimplemented

**Figure 14-7. TIM Channel Status and Control Registers (TSC0: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.

Clear CHxF by reading the TIM channel x status and control register with CHxF set and then writing a 0 to CHxF. If another interrupt request occurs before the clearing sequence is complete, then writing a 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 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 TIM channel 0 status and control register.

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:A ≠ 00, this read/write bit selects either input capture operation or unbuffered output compare/PWM operation. See Table 14-3.

1 = Unbuffered output compare/PWM operation
0 = Input capture operation

When ELSxB:A = 00, this read/write bit selects the initial output level of the TCHx pin (see Table 14-3). 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).*

**Table 14-3. Mode, Edge, and Level Selection**

| MSxB | MSxA | ELSxB | ELSxA | Mode | Configuration |
|---|---|---|---|---|---|
| X | 0 | 0 | 0 | Output preset | Pin under port control; initial output level high |
| X | 1 | 0 | 0 | | Pin under port control; initial output level low |
| 0 | 0 | 0 | 1 | Input capture | Capture on rising edge only |
| 0 | 0 | 1 | 0 | | Capture on falling edge only |
| 0 | 0 | 1 | 1 | | Capture on rising or falling edge |
| 0 | 1 | 0 | 0 | Output compare or PWM | Software compare only |
| 0 | 1 | 0 | 1 | | Toggle output on compare |
| 0 | 1 | 1 | 0 | | Clear output on compare |
| 0 | 1 | 1 | 1 | | Set output on compare |
| 1 | X | 0 | 1 | Buffered output compare or buffered PWM | Toggle output on compare |
| 1 | X | 1 | 0 | | Clear output on compare |
| 1 | X | 1 | 1 | | Set output on compare |

### 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 14-3 shows how ELSxB and ELSxA work. Reset clears the ELSxB and ELSxA bits.

***NOTE***

*After initially enabling a TIM channel register for input capture operation and selecting the edge sensitivity, clear CHxF to ignore any erroneous edge detection flags.*

### 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 a 1, setting the CHxMAX bit forces the duty cycle of buffered and unbuffered PWM signals to 100%. As Figure 14-8 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.

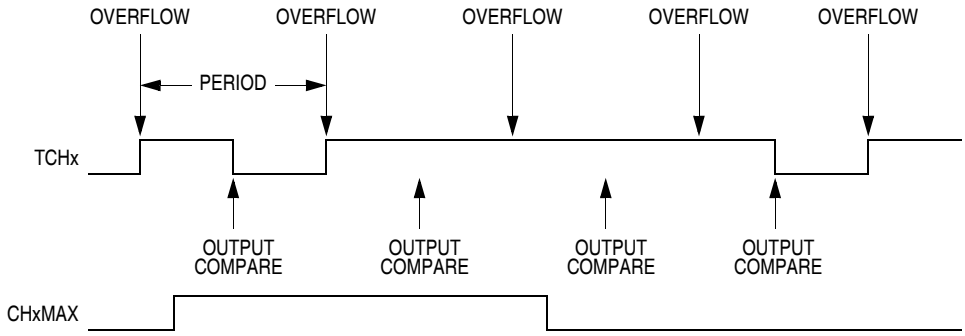



**Figure 14-8. CHxMAX Latency**

## 14.9.5 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: $0026 TCH0H

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

Address: $0027 TCH0L

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read:<br>Write: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| Reset: | Indeterminate after reset | | | | | | | |

Address: $0029 TCH1H

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

Address: $02A TCH1L

| | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Read:<br>Write: | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| Reset: | Indeterminate after reset | | | | | | | |

**Figure 14-9. TIM Channel Registers (TCH0H/L:TCH1H/L)**

# Chapter 15
# Development Support

## 15.1 Introduction

This section describes the break module, the monitor read-only memory (MON), and the monitor mode entry methods.

## 15.2 Break Module (BRK)

The break module can generate a break interrupt that stops normal program flow at a defined address to enter a background program.

Features include:

- Accessible input/output (I/O) registers during the break Interrupt
- Central processor unit (CPU) generated break interrupts
- Software-generated break interrupts
- Computer operating properly (COP) disabling during break interrupts

### 15.2.1 Functional Description

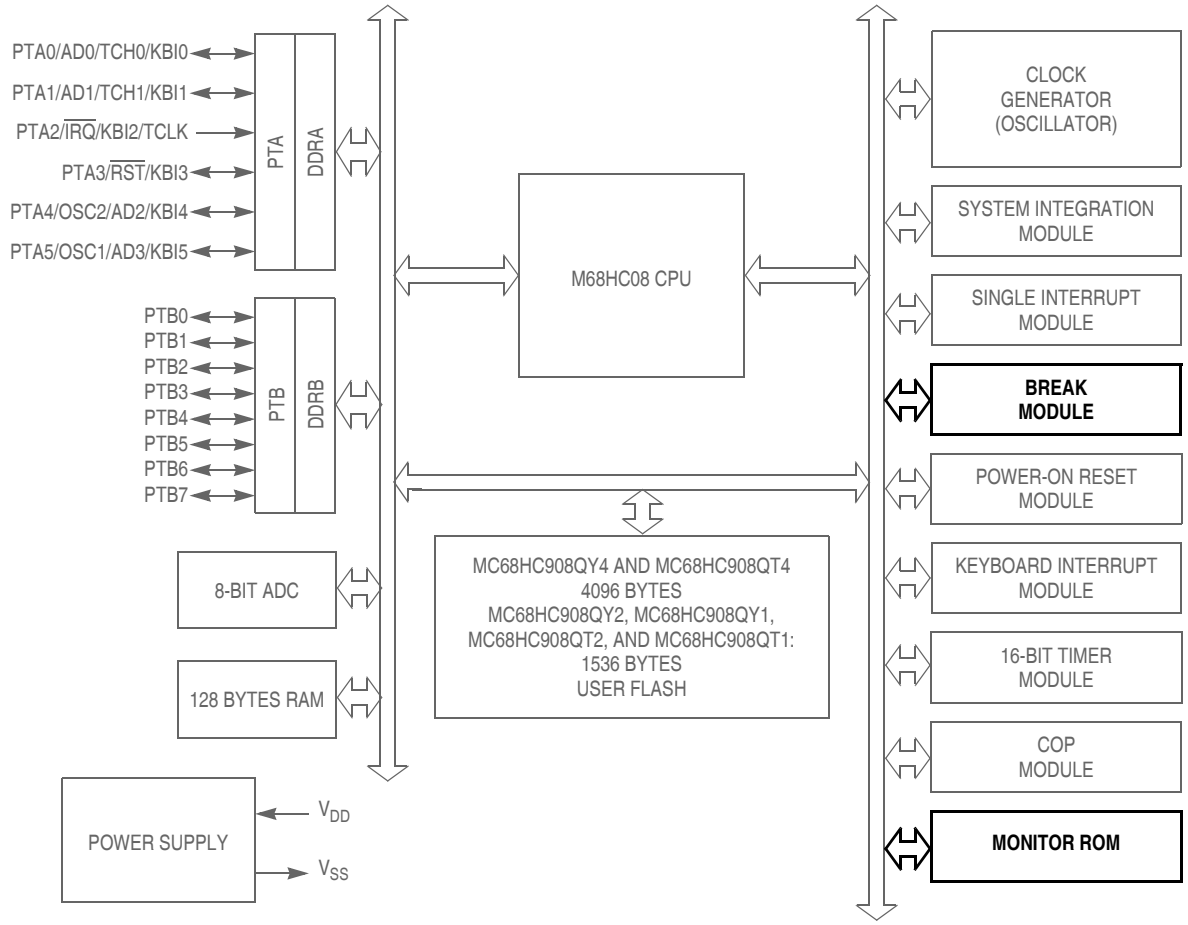
When the internal address bus matches the value written in the break address registers, the break module issues a breakpoint signal ($\overline{\text{BKPT}}$) to the system integration module (SIM). The SIM then causes the CPU to load the instruction register with a software interrupt instruction (SWI). 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 1 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 is generated. A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the microcontroller unit (MCU) to normal operation.

Figure 15-2 shows the structure of the break module.




RST, IRQ: Pins have internal (about 30K Ohms) pull up
PTA[0:5]: High current sink and source capability
PTA[0:5]: Pins have programmable keyboard interrupt and pull up
PTB[0:7]: Not available on 8-pin devices – MC68HC908QT1, MC68HC908QT2, and MC68HC908QT4 (see note in 12.1 Introduction)
ADC: Not available on the MC68HC908QY1 and MC68HC908QT1

**Figure 15-1. Block Diagram Highlighting BRK and MON Blocks**

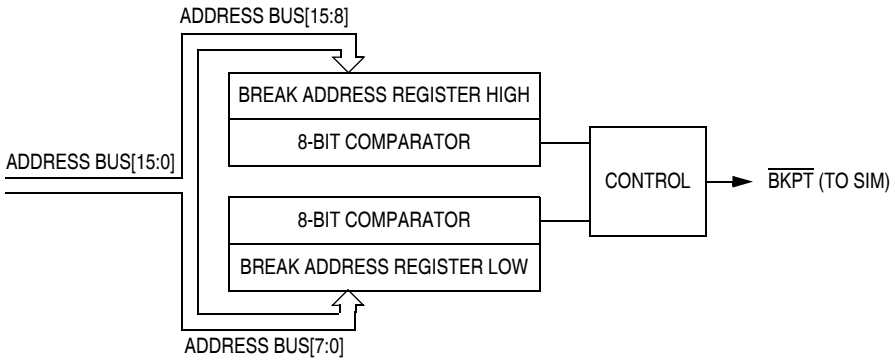



**Figure 15-2. Break Module Block Diagram**

When the internal address bus matches the value written in the break address registers or when software writes a 1 to the BRKA bit in the break status and control register, the CPU starts a break interrupt by:

- Loading the instruction register with the SWI instruction
- Loading the program counter with $FFFC and $FFFD ($FEFC and $FEFD in monitor mode)

The break interrupt timing is:

- When a break address is placed at the address of the instruction opcode, the instruction is not executed until after completion of the break interrupt routine.
- When a break address is placed at an address of an instruction operand, the instruction is executed before the break interrupt.
- When software writes a 1 to the BRKA bit, the break interrupt occurs just before the next instruction is executed.

By updating a break address and clearing the BRKA bit in a break interrupt routine, a break interrupt can be generated continuously.

***CAUTION***

*A break address should be placed at the address of the instruction opcode. When software does not change the break address and clears the BRKA bit in the first break interrupt routine, the next break interrupt will not be generated after exiting the interrupt routine even when the internal address bus matches the value written in the break address registers.*

#### *15.2.1.1 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 13.8.2 Break Flag Control Register and the **Break Interrupts** subsection for each module.

#### *15.2.1.2 TIM During Break Interrupts*

A break interrupt stops the timer counter.

#### *15.2.1.3 COP During Break Interrupts*

The COP is disabled during a break interrupt with monitor mode when BDCOP bit is set in break auxiliary register (BRKAR).

### 15.2.2 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)

### 15.2.2.1 Break Status and Control Register

The break status and control register (BRKSCR) contains break module enable and status bits.

Address: $FE0B

| | 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 15-3. 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 0 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 1 to BRKA generates a break interrupt. Clear BRKA by writing a 0 to it before exiting the break routine. Reset clears the BRKA bit.

1 = Break address match
0 = No break address match

### 15.2.2.2 Break Address Registers

The break address registers (BRKH and BRKL) contain the high and low bytes of the desired breakpoint address. Reset clears the break address registers.

Address: $FE09

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

**Figure 15-4. Break Address Register High (BRKH)**

Address: $FE0A

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

**Figure 15-5. Break Address Register Low (BRKL)**

### 15.2.2.3 Break Auxiliary Register

The break auxiliary register (BRKAR) contains a bit that enables software to disable the COP while the MCU is in a state of break interrupt with monitor mode.

Address: $FE02

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

= Unimplemented

**Figure 15-6. Break Auxiliary Register (BRKAR)**

**BDCOP — Break Disable COP Bit**

This read/write bit disables the COP during a break interrupt. Reset clears the BDCOP bit.

1 = COP disabled during break interrupt
0 = COP enabled during break interrupt

### 15.2.2.4 Break Status Register

The break status register (BSR) contains a flag to indicate that a break caused an exit from wait mode. This register is only used in emulation 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 0 clears SBSW.

**Figure 15-7. Break Status Register (BSR)**

**SBSW — SIM Break Stop/Wait**

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.

1 = Wait mode was exited by break interrupt
0 = Wait mode was not exited by break interrupt

#### *15.2.2.5 Break Flag Control Register*

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

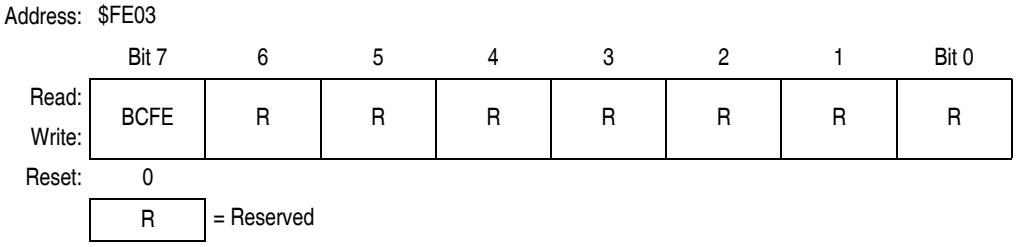



**Figure 15-8. Break Flag Control Register (BFCR)**

**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

### 15.2.3 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power- consumption standby modes. If enabled, the break module will remain enabled in wait and stop modes. However, since the internal address bus does not increment in these modes, a break interrupt will never be triggered.

## 15.3 Monitor Module (MON)

This subsection describes the monitor module (MON) and the monitor mode entry methods. The monitor allows debugging and programming of the microcontroller unit (MCU) through a single-wire interface with a host computer. 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.

Features include:

- Normal user-mode pin functionality on most pins
- One pin dedicated to serial communication between MCU and host computer
- Standard non-return-to-zero (NRZ) communication with host computer
- Execution of code in random-access memory (RAM) or FLASH
- FLASH memory security feature(1)
- FLASH memory programming interface
- Use of external 9.8304 MHz oscillator to generate internal frequency of 2.4576 MHz
- Simple internal oscillator mode of operation (no external clock or high voltage)
- 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 is applied to $\overline{IRQ}$

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

### 15.3.1 Functional Description

Figure 15-9 shows a simplified diagram of monitor mode entry.

The monitor module receives and executes commands from a host computer. Figure 15-10, Figure 15-11, and Figure 15-12 show example circuits used to enter monitor mode and communicate with a host computer via a standard RS-232 interface.

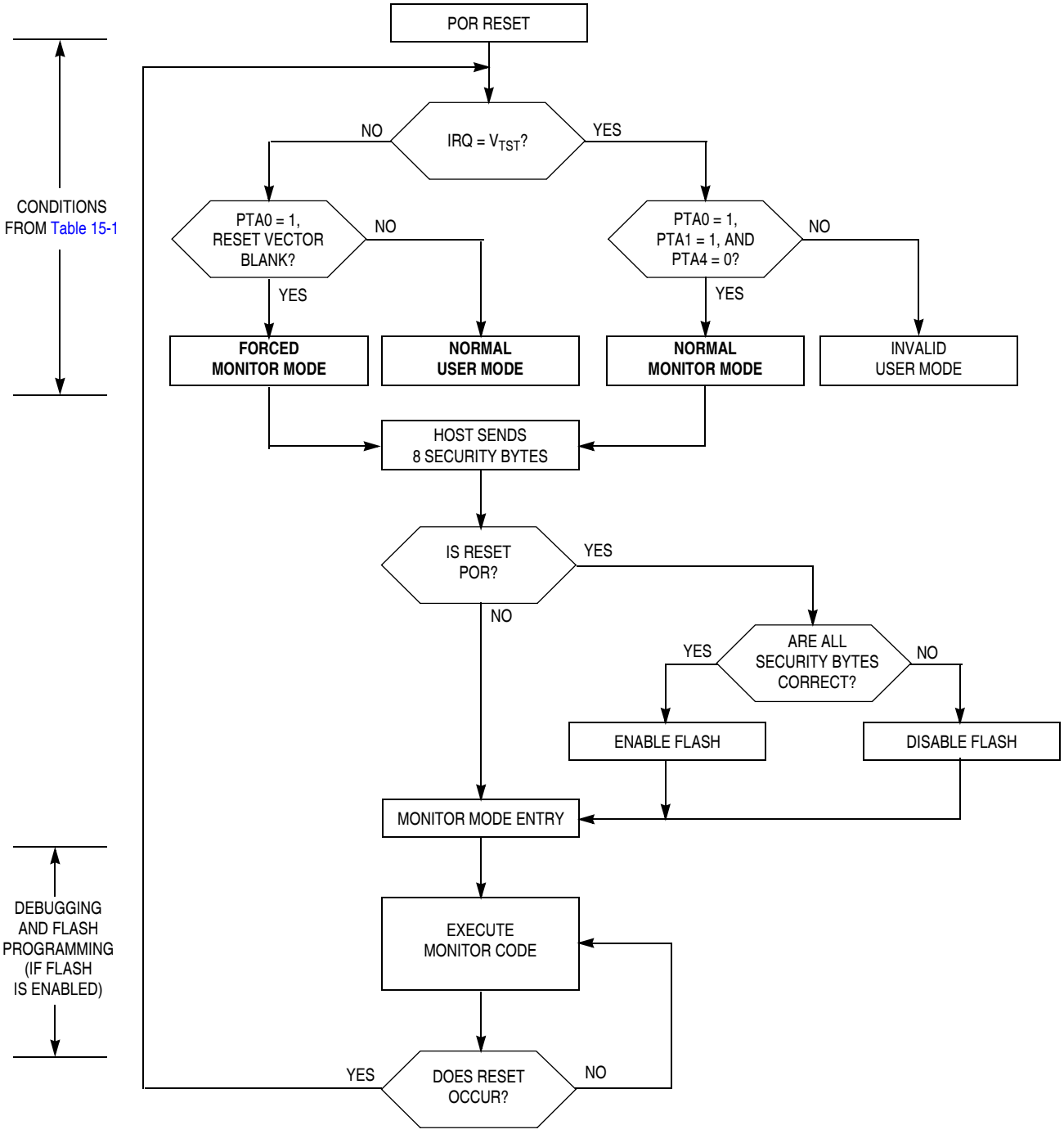



**Figure 15-9. Simplified Monitor Mode Entry Flowchart**

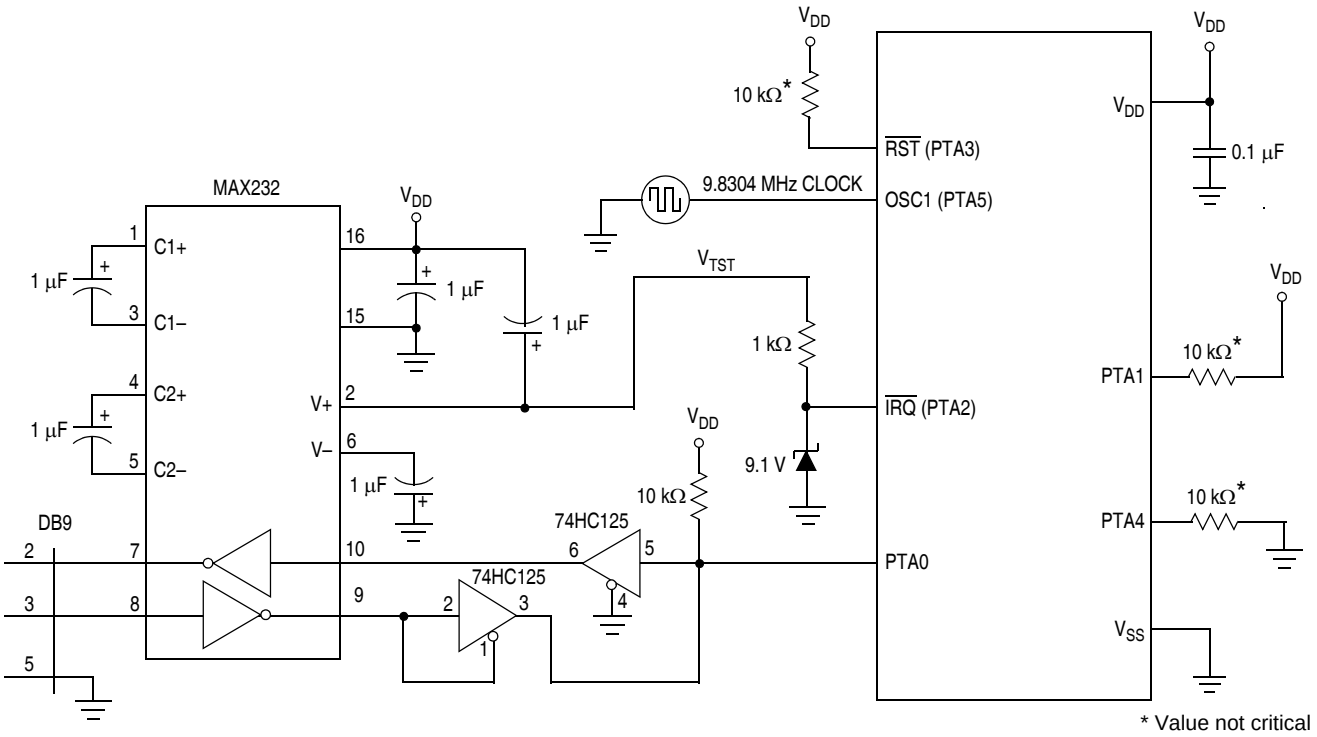



**Figure 15-10. Monitor Mode Circuit (External Clock, with High Voltage)**

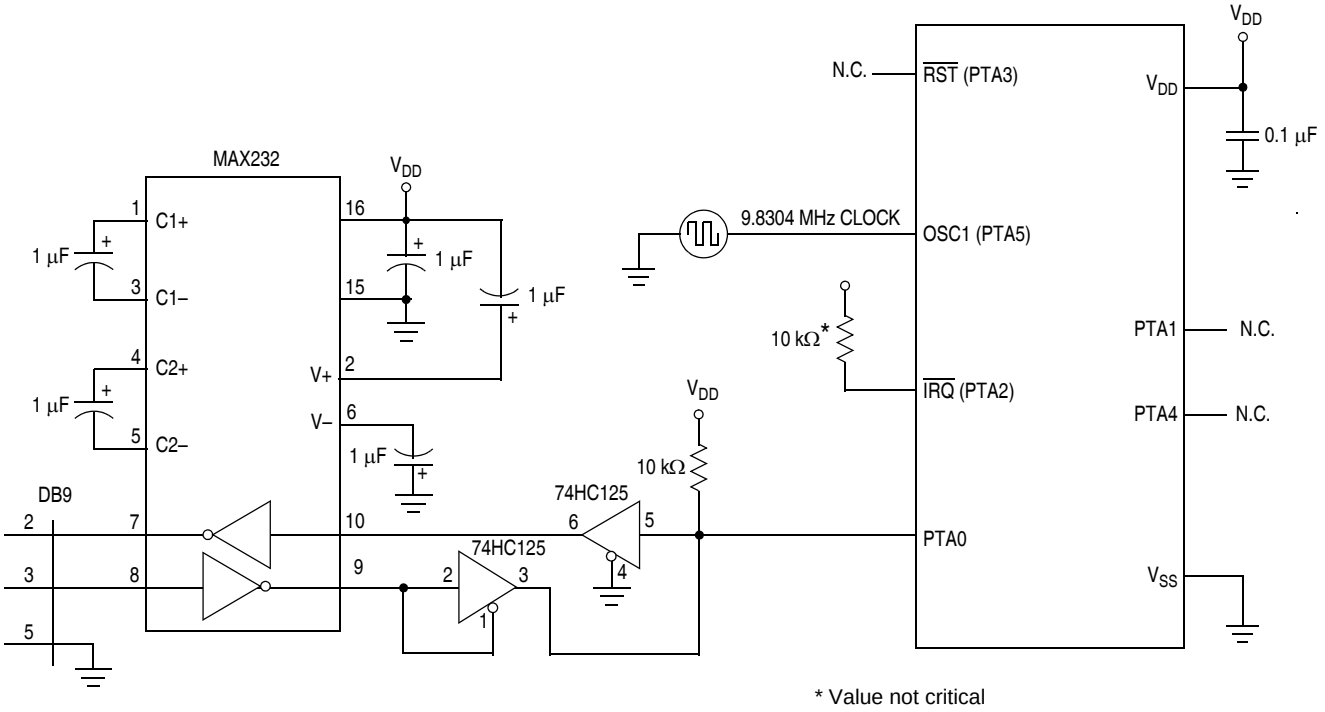



**Figure 15-11. Monitor Mode Circuit (External Clock, No High Voltage)**

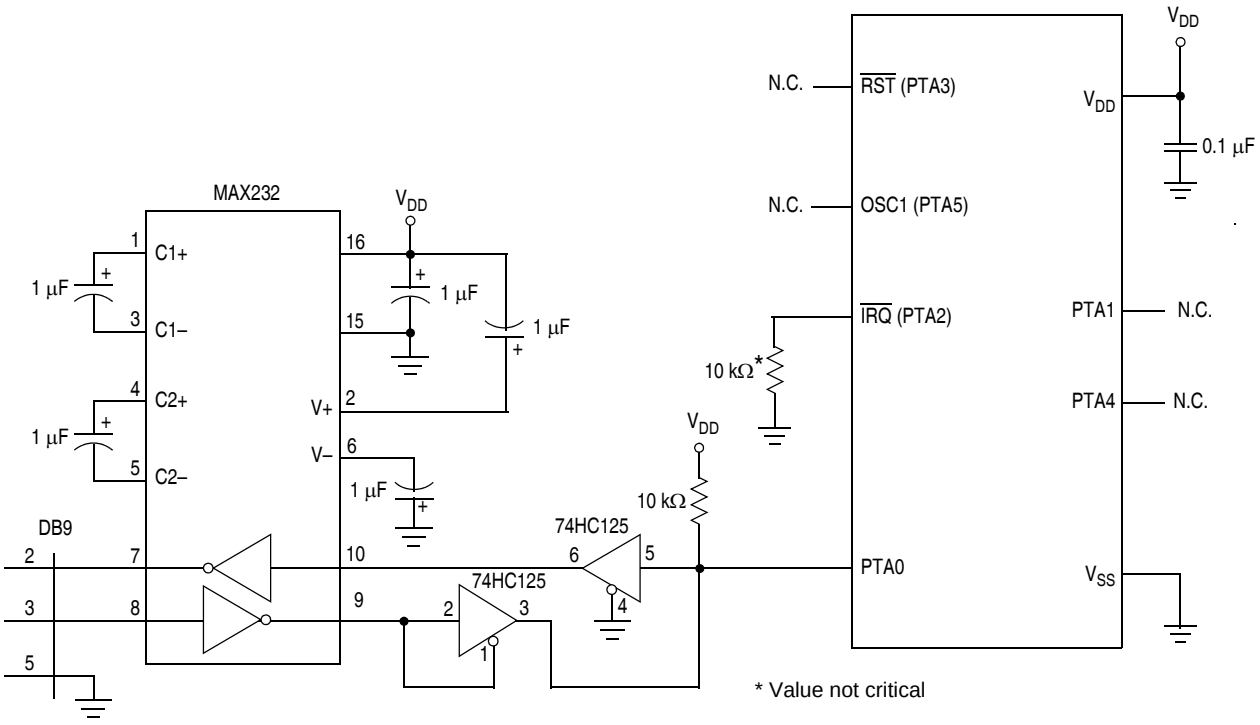



**Figure 15-12. Monitor Mode Circuit (Internal Clock, No High Voltage)**

Simple monitor commands can access any memory address. In monitor mode, the MCU can execute code downloaded into RAM by a host computer while most MCU pins retain normal operating mode functions. All communication between the host computer and the MCU is through the PTA0 pin. A level-shifting and multiplexing interface is required between PTA0 and the host computer. PTA0 is used in a wired-OR configuration and requires a pullup resistor.

The monitor code has been updated from previous versions of the monitor code to allow enabling the internal oscillator to generate the internal clock. This addition, which is enabled when $\overline{IRQ}$ is held low out of reset, is intended to support serial communication/programming at 9600 baud in monitor mode by using the internal oscillator, and the internal oscillator user trim value OSCTRIM (FLASH location $FFC0, if programmed) to generate the desired internal frequency (3.2 MHz). Since this feature is enabled only when $\overline{IRQ}$ is held low out of reset, it cannot be used when the reset vector is programmed (i.e., the value is not $FFFF) because entry into monitor mode in this case requires $V_{TST}$ on $\overline{IRQ}$. The $\overline{IRQ}$ pin must remain low during this monitor session in order to maintain communication.

Table 15-1 shows the pin conditions for entering monitor mode. As specified in the table, monitor mode may be entered after a power-on reset (POR) and will allow communication at 9600 baud provided one of the following sets of conditions is met:

- If $FFFE and $FFFF do not contain $FF (programmed state):
  - The external clock is 9.8304 MHz
  - $\overline{IRQ}$ = $V_{TST}$
- If $FFFE and $FFFF contain $FF (erased state):
  - The external clock is 9.8304 MHz
  - $\overline{IRQ}$ = $V_{DD}$ (this can be implemented through the internal $\overline{IRQ}$ pullup)
- If $FFFE and $FFFF contain $FF (erased state):
  - $\overline{IRQ}$ = $V_{SS}$ (internal oscillator is selected, no external clock required)

**Table 15-1. Monitor Mode Signal Requirements and Options**

| Mode | $\overline{IRQ}$ (PTA2) | $\overline{RST}$ (PTA3) | Reset Vector | Serial Communi-cation | Mode Selection | | COP | Communication Speed | | | Comments |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | PTA0 | PTA1 | PTA4 | | External Clock | Bus Frequency | Baud Rate | |
| Normal Monitor | $V_{TST}$ | $V_{DD}$ | X | 1 | 1 | 0 | Disabled | 9.8304 MHz | 2.4576 MHz | 9600 | Provide external clock at OSC1. |
| Forced Monitor | $V_{DD}$ | X | $FFFF (blank) | 1 | X | X | Disabled | 9.8304 MHz | 2.4576 MHz | 9600 | Provide external clock at OSC1. |
| | $V_{SS}$ | X | $FFFF (blank) | 1 | X | X | Disabled | X | 3.2 MHz (Trimmed) | 9600 | Internal clock is active. |
| User | X | X | Not $FFFF | X | X | X | Enabled | X | X | X | |
| MON08 Function [Pin No.] | $V_{TST}$ [6] | $\overline{RST}$ [4] | — | COM [8] | MOD0 [12] | MOD1 [10] | — | OSC1 [13] | — | — | |

1. PTA0 must have a pullup resistor to $V_{DD}$ in monitor mode.
2. Communication speed in the table is an example to obtain a baud rate of 9600. Baud rate using external oscillator is bus frequency / 256 and baud rate using internal oscillator is bus frequency / 335.
3. External clock is a 9.8304 MHz oscillator on OSC1.
4. X = don't care
5. MON08 pin refers to P&E Microcomputer Systems' MON08-Cyclone 2 by 8-pin connector.

| | | | |
|---|---|---|---|
| NC | 1 | 2 | GND |
| NC | 3 | 4 | $\overline{RST}$ |
| NC | 5 | 6 | $\overline{IRQ}$ |
| NC | 7 | 8 | PTA0 |
| NC | 9 | 10 | PTA4 |
| NC | 11 | 12 | PTA1 |
| OSC1 | 13 | 14 | NC |
| $V_{DD}$ | 15 | 16 | NC |

The rising edge of the internal $\overline{RST}$ signal latches the monitor mode. Once monitor mode is latched, the values on PTA1 and PTA4 pins can be changed.

Once out of reset, the MCU waits for the host to send eight security bytes (see 15.3.2 Security). After the security bytes, the MCU sends a break signal (10 consecutive logic 0s) to the host, indicating that it is ready to receive a command.

#### *15.3.1.1 Normal Monitor Mode*

$\overline{RST}$ and OSC1 functions will be active on the PTA3 and PTA5 pins respectively as long as $V_{TST}$ is applied to the $\overline{IRQ}$ pin. If the $\overline{IRQ}$ pin is lowered (no longer $V_{TST}$) then the chip will still be operating in monitor mode, but the pin functions will be determined by the settings in the configuration registers (see Chapter 5 Configuration Register (CONFIG)) when $V_{TST}$ was lowered. With $V_{TST}$ lowered, the BIH and BIL instructions will read the $\overline{IRQ}$ pin state only if IRQEN is set in the CONFIG2 register.

If monitor mode was entered with $V_{TST}$ on $\overline{IRQ}$, then the COP is disabled as long as $V_{TST}$ is applied to $\overline{IRQ}$.

#### 15.3.1.2 Forced Monitor Mode

If entering monitor mode without high voltage on $\overline{IRQ}$, then startup port pin requirements and conditions, (PTA1/PTA4) are not in effect. This is to reduce circuit requirements when performing in-circuit programming.

**NOTE**

*If the reset vector is blank and monitor mode is entered, the chip will see an additional reset cycle after the initial power-on reset (POR). Once the reset vector has been programmed, the traditional method of applying a voltage, $V_{TST}$, to $\overline{IRQ}$ must be used to enter monitor mode.*

If monitor mode was entered as a result of the reset vector being blank, the COP is always disabled regardless of the state of $\overline{IRQ}$.

If the voltage applied to the $\overline{IRQ}$ is less than $V_{TST}$, the MCU will come out of reset in user mode. Internal circuitry monitors the reset vector fetches and will assert an internal reset if it detects that the reset vectors are erased ($FF). When the MCU comes out of reset, it is forced into monitor mode without requiring high voltage on the $\overline{IRQ}$ pin. Once out of reset, the monitor code is initially executing with the internal clock at its default frequency.

If $\overline{IRQ}$ is held high, all pins will default to regular input port functions except for PTA0 and PTA5 which will operate as a serial communication port and OSC1 input respectively (refer to Figure 15-10). That will allow the clock to be driven from an external source through OSC1 pin.

If $\overline{IRQ}$ is held low, all pins will default to regular input port function except for PTA0 which will operate as serial communication port. Refer to Figure 15-11.

Regardless of the state of the $\overline{IRQ}$ pin, it will not function as a port input pin in monitor mode. Bit 2 of the Port A data register will always read 0. The BIH and BIL instructions will behave as if the $\overline{IRQ}$ pin is enabled, regardless of the settings in the configuration register. See Chapter 5 Configuration Register (CONFIG).

The COP module is disabled in forced monitor mode. Any reset other than a power-on reset (POR) will automatically force the MCU to come back to the forced monitor mode.

#### 15.3.1.3 Monitor Vectors

In monitor mode, the MCU uses different vectors for reset, SWI (software interrupt), and break interrupt than those for user mode. 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.

**NOTE**

*Exiting monitor mode after it has been initiated by having a blank reset vector requires a power-on reset (POR). Pulling $\overline{RST}$ (when $\overline{RST}$ pin available) low will not exit monitor mode in this situation.*

Table 15-2 summarizes the differences between user mode and monitor mode regarding vectors.

**Table 15-2. Mode Difference**

| Modes | Functions | | | | | |
|---|---|---|---|---|---|---|
| | Reset Vector High | Reset Vector Low | Break Vector High | Break Vector Low | SWI Vector High | SWI Vector Low |
| User | $FFFE | $FFFF | $FFFC | $FFFD | $FFFC | $FFFD |
| Monitor | $FEFE | $FEFF | $FEFC | $FEFD | $FEFC | $FEFD |

#### *15.3.1.4 Data Format*

Communication with the monitor ROM is in standard non-return-to-zero (NRZ) mark/space data format. Transmit and receive baud rates must be identical.

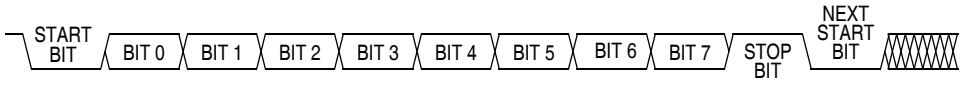



**Figure 15-13. Monitor Data Format**

#### *15.3.1.5 Break Signal*

A start bit (logic 0) followed by nine logic 0 bits is a break signal. When the monitor receives a break signal, it drives the PTA0 pin high for the duration of two bits and then echoes back the break signal.

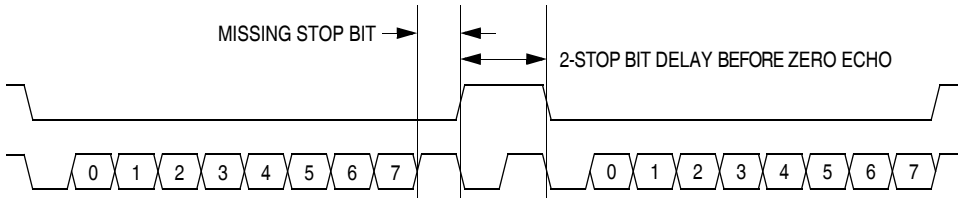



**Figure 15-14. Break Transaction**

#### *15.3.1.6 Baud Rate*

The monitor communication baud rate is controlled by the frequency of the external or internal oscillator and the state of the appropriate pins as shown in Table 15-1.

Table 15-1 also lists the bus frequencies to achieve standard baud rates. The effective baud rate is the bus frequency divided by 256 when using an external oscillator. When using the internal oscillator in forced monitor mode, the effective baud rate is the bus frequency divided by 335.

#### *15.3.1.7 Commands*

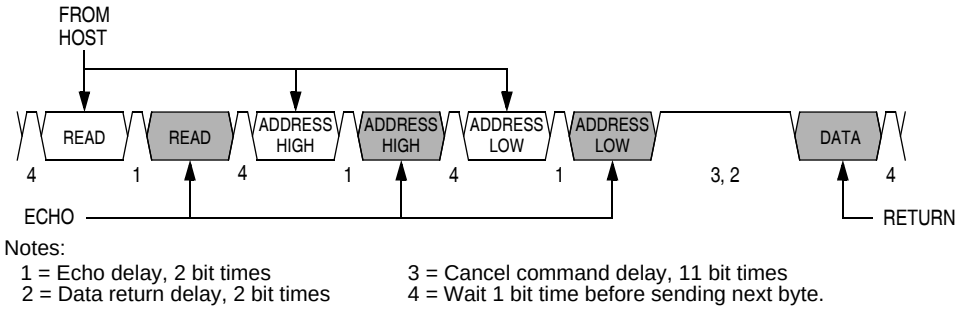
The monitor ROM firmware uses these commands:

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

The monitor ROM firmware echoes each received byte back to the PTA0 pin for error checking. An 11-bit delay at the end of each command allows the host to send a break character to cancel the command. A delay of two bit times occurs before each echo and before READ, IREAD, or READSP data is returned. The data returned by a read command appears after the echo of the last byte of the command.

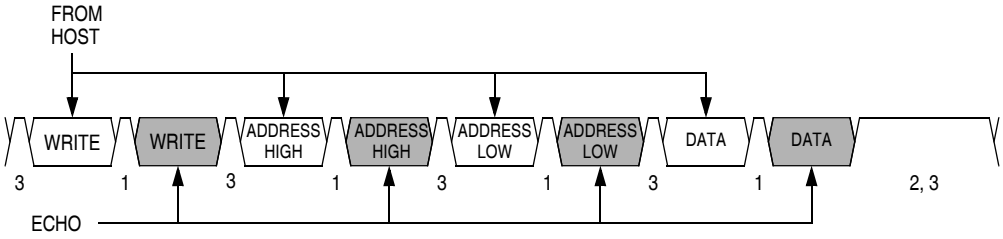
***NOTE***

*Wait one bit time after each echo before sending the next byte.*




Notes:
1 = Echo delay, 2 bit times
2 = Data return delay, 2 bit times
3 = Cancel command delay, 11 bit times
4 = Wait 1 bit time before sending next byte.

**Figure 15-15. Read Transaction**




Notes:
1 = Echo delay, 2 bit times
2 = Cancel command delay, 11 bit times
3 = Wait 1 bit time before sending next byte.

**Figure 15-16. Write Transaction**

A brief description of each monitor mode command is given in Table 15-3 through Table 15-8.

**Table 15-3. READ (Read Memory) Command**

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

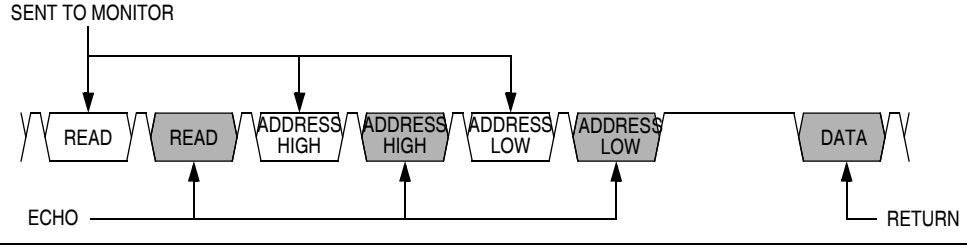


**Table 15-4. WRITE (Write Memory) Command**

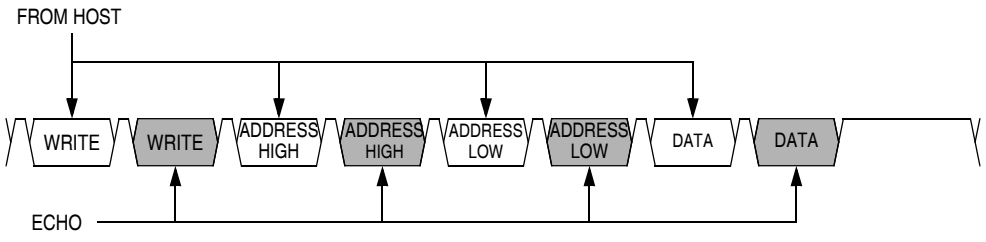
| | |
|---|---|
| **Description** | Write byte to memory |
| **Operand** | 2-byte address in high-byte:low-byte order; low byte followed by data byte |
| **Data Returned** | None |
| **Opcode** | $49 |
| **Command Sequence** |  |



**Table 15-5. IREAD (Indexed Read) Command**

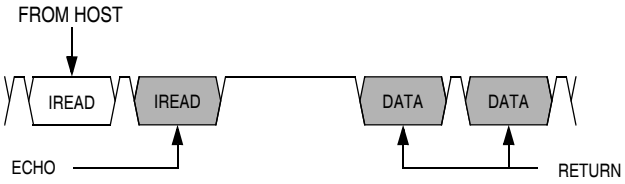
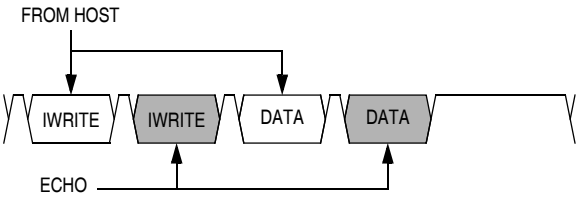
| | |
|---|---|
| **Description** | Read next 2 bytes in memory from last address accessed |
| **Operand** | None |
| **Data Returned** | Returns contents of next two addresses |
| **Opcode** | $1A |
| **Command Sequence** |  |



**Table 15-6. IWRITE (Indexed Write) Command**

| | |
|---|---|
| **Description** | Write to last address accessed + 1 |
| **Operand** | Single data byte |
| **Data Returned** | None |
| **Opcode** | $19 |
| **Command Sequence** |  |



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

**Table 15-7. READSP (Read Stack Pointer) Command**

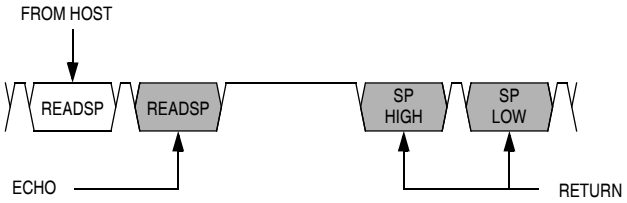
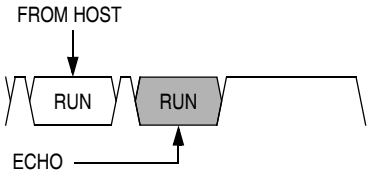
| | |
|---|---|
| **Description** | Reads stack pointer |
| **Operand** | None |
| **Data Returned** | Returns incremented stack pointer value (SP + 1) in high-byte:low-byte order |
| **Opcode** | $0C |
| **Command Sequence** |  |



**Table 15-8. RUN (Run User Program) Command**

| | |
|---|---|
| **Description** | Executes PULH and RTI instructions |
| **Operand** | None |
| **Data Returned** | None |
| **Opcode** | $28 |
| **Command Sequence** |  |



The MCU executes the SWI and PSHH instructions when it enters monitor mode. The RUN command tells the MCU to execute the PULH and RTI instructions. Before sending the RUN command, the host can modify the stacked CPU registers to prepare to run the host program. The READSP command returns the incremented stack pointer value, SP + 1. The high and low bytes of the program counter are at addresses SP + 5 and SP + 6.

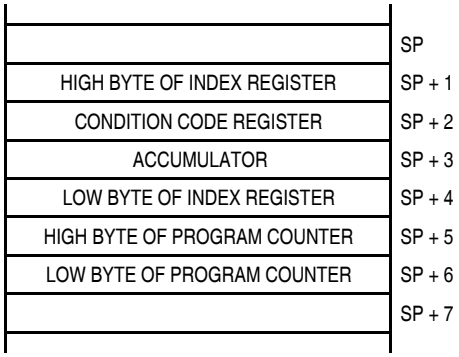



**Figure 15-17. Stack Pointer at Monitor Mode Entry**

### 15.3.2 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 PTA0. 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 15-18.

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 $80 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).

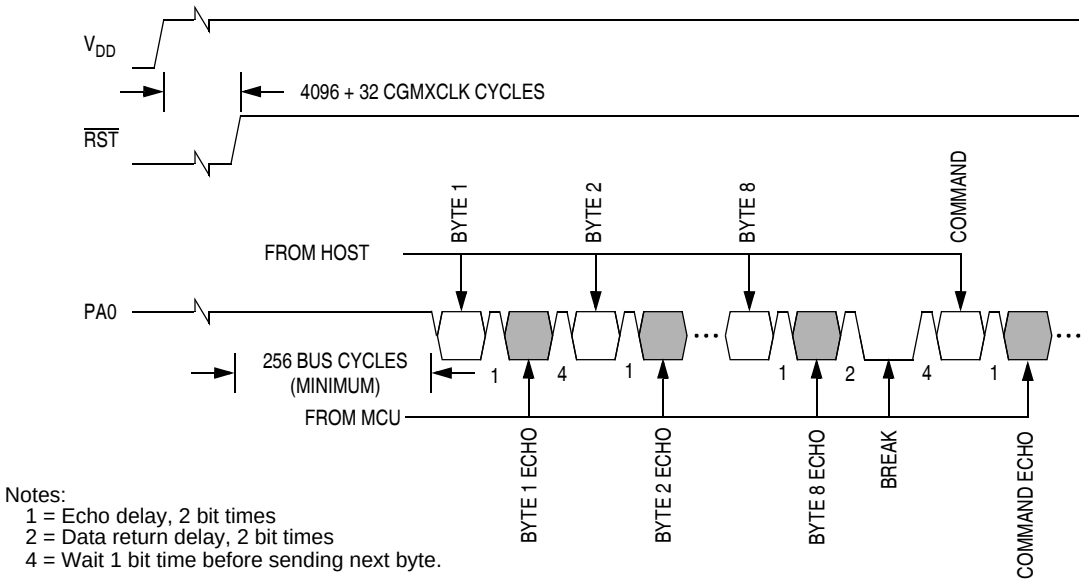



**Figure 15-18. Monitor Mode Entry Timing**

# Chapter 16
# Electrical Specifications

## 16.1 Introduction

This section contains electrical and timing specifications.

## 16.2 Absolute Maximum Ratings

Maximum ratings are the extreme limits to which the microcontroller unit (MCU) can be exposed without permanently damaging it.

***NOTE***

*This device is not guaranteed to operate properly at the maximum ratings. Refer to 16.5 5-V DC Electrical Characteristics and 16.9 3-V DC Electrical Characteristics for guaranteed operating conditions.*

| 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 +9.1 | V |
| Maximum current per pin excluding PTA0–PTA5, $V_{DD}$, and $V_{SS}$ | I | ±15 | mA |
| Maximum current for pins PTA0–PTA5 | $I_{PTA0}$—$I_{PTA5}$ | ±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 references 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 $V_{DD}$.)*

## 16.3 Functional Operating Range

| Characteristic | Symbol | Value | Unit | Temp. Code |
|---|---|---|---|---|
| Operating temperature range | $T_A$ | –40 to +125<br>–40 to +105<br>–40 to +85 | •C | M<br>V<br>C |
| Operating voltage range | $V_{DD}$ | 2.7 to 5.5 | V | — |

## 16.4 Thermal Characteristics

| Characteristic | Symbol | Value | Unit |
|---|---|---|---|
| Thermal resistance<br>8-pin PDIP<br>8-pin SOIC<br>8-pin DFN<br>16-pin PDIP<br>16-pin SOIC<br>16-pin TSSOP | $\theta_{JA}$ | <br>105<br>142<br>173<br>76<br>90<br>133 | •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 |
| Maximum junction temperature | $T_{JM}$ | 150 | •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 $T_A$.

## 16.5 5-V DC Electrical Characteristics

| Characteristic[1] | Symbol | Min | Typ[2] | Max | Unit |
|---|---|---|---|---|---|
| Output high voltage<br>$I_{Load}$ = –2.0 mA, all I/O pins<br>$I_{Load}$ = –10.0 mA, all I/O pins<br>$I_{Load}$ = –15.0 mA, PTA0, PTA1, PTA3–PTA5 only | $V_{OH}$ | <br>$V_{DD}$–0.4<br>$V_{DD}$–1.5<br>$V_{DD}$–0.8 | <br>—<br>—<br>— | <br>—<br>—<br>— | V |
| Maximum combined $I_{OH}$ (all I/O pins) | $I_{OHT}$ | — | — | 50 | mA |
| Output low voltage<br>$I_{Load}$ = 1.6 mA, all I/O pins<br>$I_{Load}$ = 10.0 mA, all I/O pins<br>$I_{Load}$ = 15.0 mA, PTA0, PTA1, PTA3–PTA5 only | $V_{OL}$ | <br>—<br>—<br>— | <br>—<br>—<br>— | <br>0.4<br>1.5<br>0.8 | V |
| Maximum combined $I_{OL}$ (all I/O pins) | $I_{OLT}$ | — | — | 50 | mA |
| Input high voltage<br>PTA0–PTA5, PTB0–PTB7 | $V_{IH}$ | 0.7 x $V_{DD}$ | — | $V_{DD}$ | V |
| Input low voltage<br>PTA0–PTA5, PTB0–PTB7 | $V_{IL}$ | $V_{SS}$ | — | 0.3 x $V_{DD}$ | V |
| Input hysteresis | $V_{HYS}$ | 0.06 x $V_{DD}$ | — | — | V |
| DC injection current, all ports | $I_{INJ}$ | –2 | — | +2 | mA |
| Total dc current injection (sum of all I/O) | $I_{INJTOT}$ | –25 | — | +25 | mA |
| Ports Hi-Z leakage current | $I_{IL}$ | –1 | ±0.1 | +1 | µA |
| Capacitance<br>Ports (as input)<br>Ports (as input) | <br>$C_{IN}$<br>$C_{OUT}$ | <br>—<br>— | <br>—<br>— | <br>12<br>8 | pF |
| POR rearm voltage[3] | $V_{POR}$ | 0 | — | 100 | mV |
| POR rise time ramp rate[4] | $R_{POR}$ | 0.035 | — | — | V/ms |
| Monitor mode entry voltage | $V_{TST}$ | $V_{DD}$ + 2.5 | — | 9.1 | V |
| Pullup resistors[5]<br>PTA0–PTA5, PTB0–PTB7 | $R_{PU}$ | 16 | 26 | 36 | kΩ |
| Low-voltage inhibit reset, trip falling voltage | $V_{TRIPF}$ | 3.90 | 4.20 | 4.50 | V |
| Low-voltage inhibit reset, trip rising voltage | $V_{TRIPR}$ | 4.00 | 4.30 | 4.60 | V |
| Low-voltage inhibit reset/recover hysteresis | $V_{HYS}$ | — | 100 | — | mV |

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. Maximum is highest voltage that POR is guaranteed.
4. If minimum $V_{DD}$ is not reached before the internal POR reset is released, the LVI will hold the part in reset until minimum $V_{DD}$ is reached.
5. $R_{PU}$ is measured at $V_{DD}$ = 5.0 V.

## 16.6 Typical 5-V Output Drive Characteristics

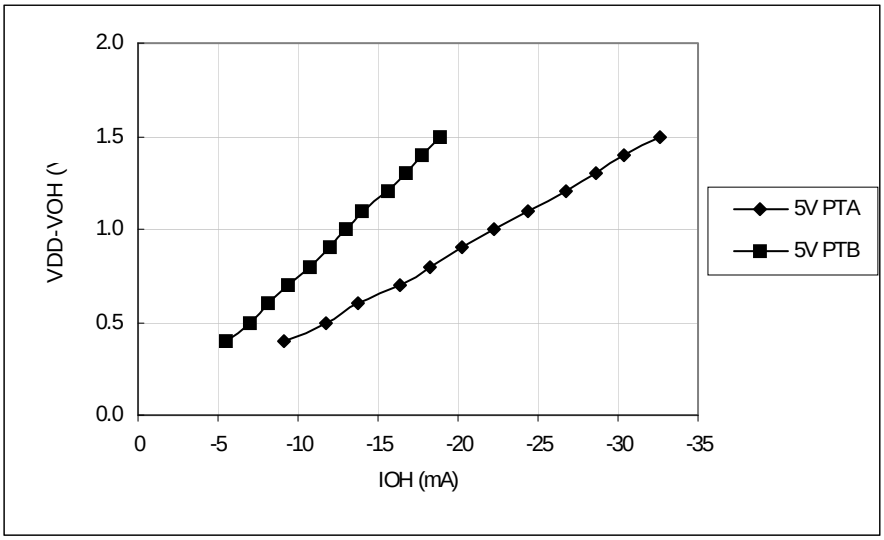



**Figure 16-1. Typical 5-Volt Output High Voltage versus Output High Current (25•C)**

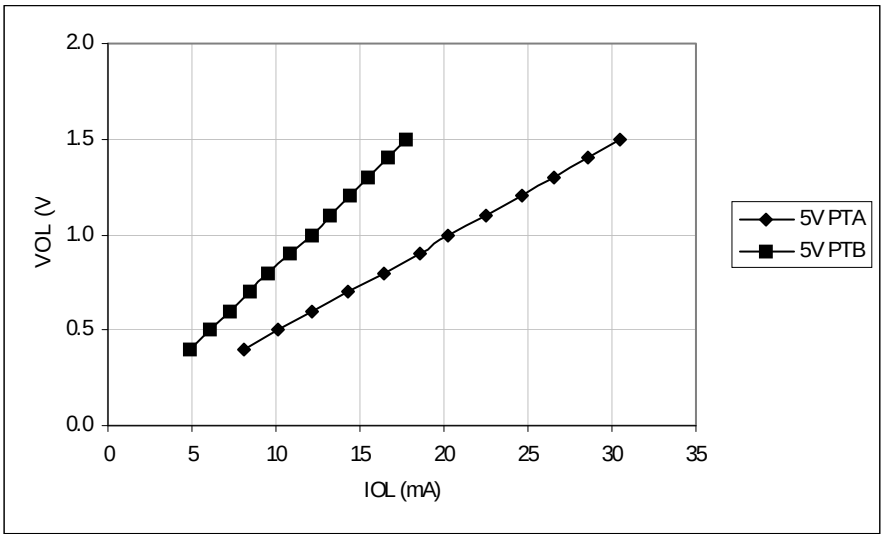



**Figure 16-2. Typical 5-Volt Output Low Voltage versus Output Low Current (25•C)**

## 16.7 5-V Control Timing

| Characteristic[1] | Symbol | Min | Max | Unit |
|---|---|---|---|---|
| Internal operating frequency | $f_{OP}$ ($f_{Bus}$) | — | 8 | MHz |
| Internal clock period ($1/f_{OP}$) | $t_{cyc}$ | 125 | — | ns |
| $\overline{RST}$ input pulse width low | $t_{RL}$ | 100 | — | ns |
| $\overline{IRQ}$ interrupt pulse width low (edge-triggered) | $t_{ILIH}$ | 100 | — | ns |
| $\overline{IRQ}$ interrupt pulse period | $t_{ILIL}$ | Note[2] | — | $t_{cyc}$ |

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. The minimum period is the number of cycles it takes to execute the interrupt service routine plus 1 $t_{cyc}$.

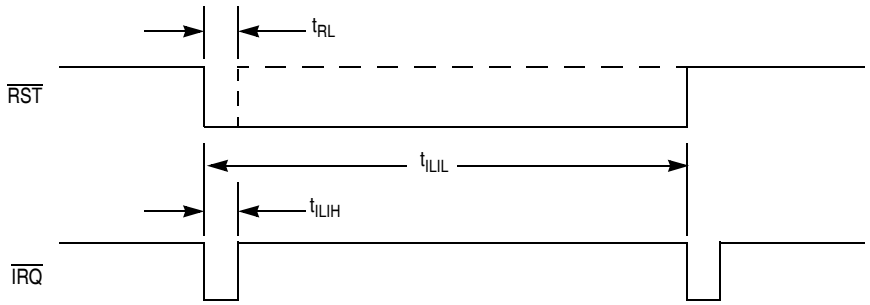



**Figure 16-3. $\overline{RST}$ and $\overline{IRQ}$ Timing**

## 16.8 5-V Oscillator Characteristics

| Characteristic | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Internal oscillator frequency(1) | $f_{INTCLK}$ | — | 12.8 | — | MHz |
| Deviation from trimmed Internal oscillator (2)(3)<br>12.8 MHz, fixed voltage, fixed temp<br>12.8 MHz, $V_{DD}$ ± 10%, 0 to 70°C<br>12.8 MHz, $V_{DD}$ ± 10%, –40 to 125°C | $ACC_{INT}$ | —<br>—<br>— | ± 0.4<br>± 2<br>— | —<br>—<br>± 5 | % |
| Crystal frequency, XTALCLK(1) | $f_{OSCXCLK}$ | 1 | — | 24 | MHz |
| External RC oscillator frequency, RCCLK(1) | $f_{RCCLK}$ | 2 | — | 12 | MHz |
| External clock reference frequency(1) (4) | $f_{OSCXCLK}$ | dc | — | 32 | MHz |
| Crystal load capacitance(5) | $C_L$ | — | 20 | — | pF |
| Crystal fixed capacitance(3) | $C_1$ | — | 2 x $C_L$ | — | — |
| Crystal tuning capacitance(3) | $C_2$ | — | 2 x $C_L$ | — | — |
| Feedback bias resistor | $R_B$ | 0.5 | 1 | 10 | MΩ |
| RC oscillator external resistor | $R_{EXT}$ | See Figure 16-4 | | | — |
| Crystal series damping resistor<br>$f_{OSCXCLK}$ = 1 MHz<br>$f_{OSCXCLK}$ = 4 MHz<br>$f_{OSCXCLK}$ = > 8 MHz | $R_S$ | —<br>—<br>— | 20<br>10<br>0 | —<br>—<br>— | kΩ |

1. Bus frequency, $f_{OP}$, is oscillator frequency divided by 4.
2. Deviation values assumes trimming @25•C and midpoint of voltage range.
3. Values are based on characterization results, not tested in production.
4. No more than 10% duty cycle deviation from 50%.
5. Consult crystal vendor data sheet.

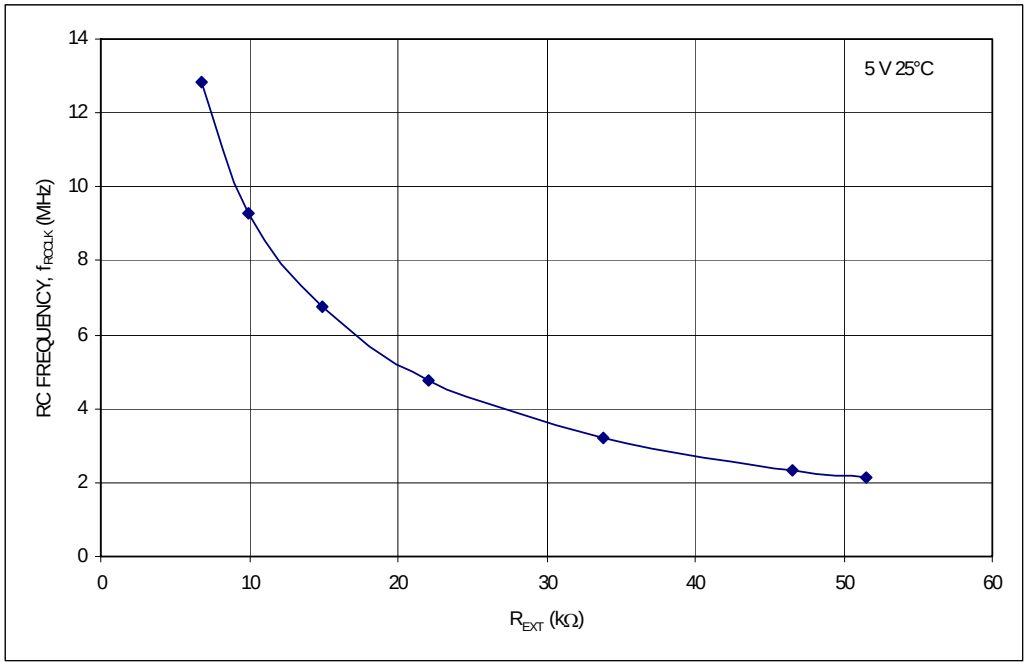



**Figure 16-4. RC versus Frequency (5 Volts @ 25•C)**

## 16.9 3-V DC Electrical Characteristics

| Characteristic[1] | Symbol | Min | Typ[2] | Max | Unit |
|---|---|---|---|---|---|
| Output high voltage<br>$I_{Load}$ = –0.6 mA, all I/O pins<br>$I_{Load}$ = –4.0 mA, all I/O pins<br>$I_{Load}$ = –10.0 mA, PTA0, PTA1, PTA3–PTA5 only | $V_{OH}$ | <br>$V_{DD}$–0.3<br>$V_{DD}$–1.0<br>$V_{DD}$–0.8 | <br>—<br>—<br>— | <br>—<br>—<br>— | V |
| Maximum combined $I_{OH}$ (all I/O pins) | $I_{OHT}$ | — | — | 50 | mA |
| Output low voltage<br>$I_{Load}$ = 0.5 mA, all I/O pins<br>$I_{Load}$ = 6.0 mA, all I/O pins<br>$I_{Load}$ = 10.0 mA, PTA0, PTA1, PTA3–PTA5 only | $V_{OL}$ | <br>—<br>—<br>— | <br>—<br>—<br>— | <br>0.3<br>1.0<br>0.8 | V |
| Maximum combined $I_{OL}$ (all I/O pins) | $I_{OLT}$ | — | — | 50 | mA |
| Input high voltage<br>PTA0–PTA5, PTB0–PTB7 | $V_{IH}$ | 0.7 x $V_{DD}$ | — | $V_{DD}$ | V |
| Input low voltage<br>PTA0–PTA5, PTB0–PTB7 | $V_{IL}$ | $V_{SS}$ | — | 0.3 x $V_{DD}$ | V |
| Input hysteresis | $V_{HYS}$ | 0.06 x $V_{DD}$ | — | — | V |
| DC injection current, all ports | $I_{INJ}$ | –2 | — | +2 | mA |
| Total dc current injection (sum of all I/O) | $I_{INJTOT}$ | –25 | — | +25 | mA |
| Ports Hi-Z leakage current | $I_{IL}$ | –1 | ±0.1 | +1 | µA |
| Capacitance<br>Ports (as input)<br>Ports (as input) | <br>$C_{IN}$<br>$C_{OUT}$ | <br>—<br>— | <br>—<br>— | <br>12<br>8 | pF |
| POR rearm voltage[3] | $V_{POR}$ | 0 | — | 100 | mV |
| POR rise time ramp rate[4] | $R_{POR}$ | 0.035 | — | — | V/ms |
| Monitor mode entry voltage | $V_{TST}$ | $V_{DD}$ + 2.5 | — | $V_{DD}$ + 4.0 | V |
| Pullup resistors[5]<br>PTA0–PTA5, PTB0–PTB7 | $R_{PU}$ | 16 | 26 | 36 | kΩ |
| Low-voltage inhibit reset, trip falling voltage | $V_{TRIPF}$ | 2.40 | 2.55 | 2.70 | V |
| Low-voltage inhibit reset, trip rising voltage | $V_{TRIPR}$ | 2.50 | 2.65 | 2.80 | V |
| Low-voltage inhibit reset/recover hysteresis | $V_{HYS}$ | — | 60 | — | mV |

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. Maximum is highest voltage that POR is guaranteed.
4. If minimum $V_{DD}$ is not reached before the internal POR reset is released, the LVI will hold the part in reset until minimum $V_{DD}$ is reached.
5. $R_{PU}$ are measured at $V_{DD}$ = 3.0 V

## 16.10 Typical 3.0-V Output Drive Characteristics

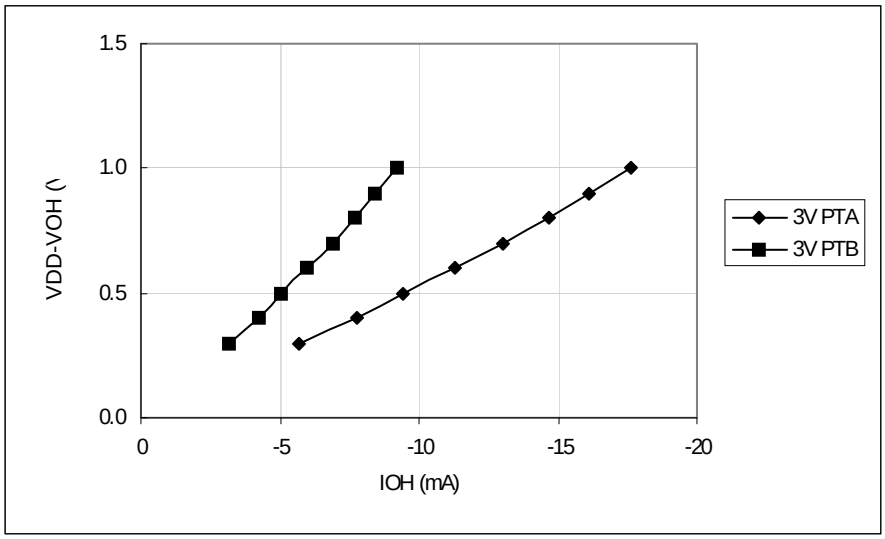



**Figure 16-5. Typical 3-Volt Output High Voltage versus Output High Current (25•C)**

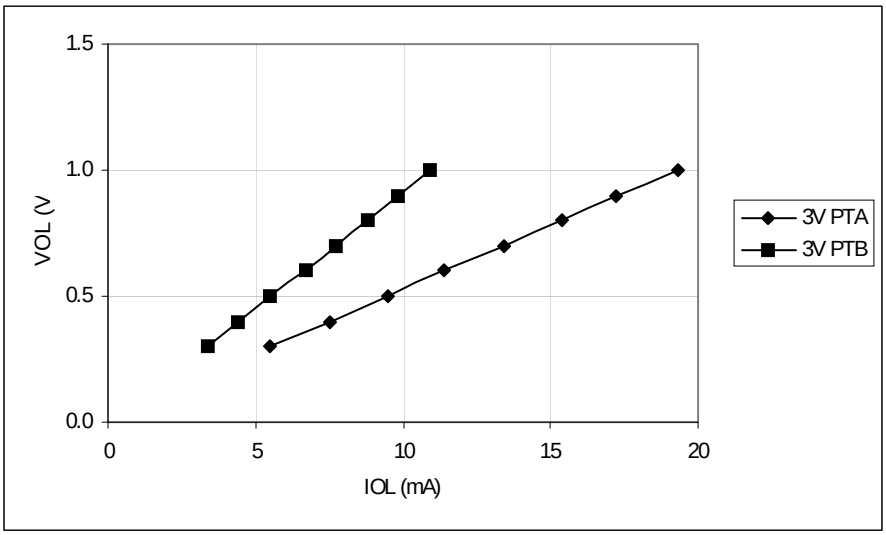



**Figure 16-6. Typical 3-Volt Output Low Voltage versus Output Low Current (25•C)**

## 16.11 3-V Control Timing

| Characteristic[1] | Symbol | Min | Max | Unit |
|---|---|---|---|---|
| Internal operating frequency | $f_{OP}$ ($f_{Bus}$) | — | 4 | MHz |
| Internal clock period ($1/f_{OP}$) | $t_{cyc}$ | 250 | — | ns |
| $\overline{RST}$ input pulse width low | $t_{RL}$ | 200 | — | ns |
| $\overline{IRQ}$ interrupt pulse width low (edge-triggered) | $t_{ILIH}$ | 200 | — | ns |
| $\overline{IRQ}$ interrupt pulse period | $t_{ILIL}$ | Note[2] | — | $t_{cyc}$ |

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. The minimum period is the number of cycles it takes to execute the interrupt service routine plus 1 $t_{cyc}$.

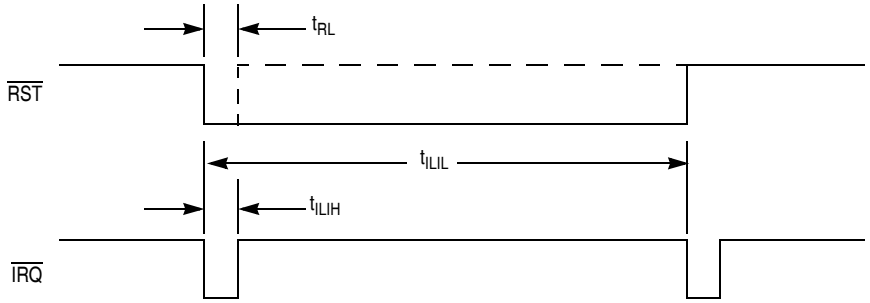



**Figure 16-7. $\overline{RST}$ and $\overline{IRQ}$ Timing**

## 16.12 3-V Oscillator Characteristics

| Characteristic | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Internal oscillator frequency(1) | $f_{INTCLK}$ | — | 12.8 | — | MHz |
| Deviation from trimmed Internal oscillator (2)(3)<br>12.8 MHz, fixed voltage, fixed temp<br>12.8 MHz, $V_{DD}$ ± 10%, 0 to 70°C<br>12.8 MHz, $V_{DD}$ ± 10%, –40 to 125°C | $ACC_{INT}$ | —<br>—<br>— | ± 0.4<br>± 2<br>— | —<br>—<br>± 5 | % |
| Crystal frequency, XTALCLK(1) | $f_{OSCXCLK}$ | 1 | — | 16 | MHz |
| External RC oscillator frequency, RCCLK (1) | $f_{RCCLK}$ | 2 | — | 10 | MHz |
| External clock reference frequency(1) (4) | $f_{OSCXCLK}$ | dc | — | 16 | MHz |
| Crystal load capacitance(5) | $C_L$ | — | 20 | — | pF |
| Crystal fixed capacitance(3) | $C_1$ | — | 2 x $C_L$ | — | — |
| Crystal tuning capacitance(3) | $C_2$ | — | 2 x $C_L$ | — | — |
| Feedback bias resistor | $R_B$ | 0.5 | 1 | 10 | MΩ |
| RC oscillator external resistor | $R_{EXT}$ | See Figure 16-8 | | | — |
| Crystal series damping resistor<br>$f_{OSCXCLK}$ = 1 MHz<br>$f_{OSCXCLK}$ = 4 MHz<br>$f_{OSCXCLK}$ = > 8 MHz | $R_S$ | —<br>—<br>— | 10<br>5<br>0 | —<br>—<br>— | kΩ |

1. Bus frequency, $f_{OP}$, is oscillator frequency divided by 4.
2. Deviation values assumes trimming @25•C and midpoint of voltage range.
3. Values are based on characterization results, not tested in production.
4. No more than 10% duty cycle deviation from 50%
5. Consult crystal vendor data sheet

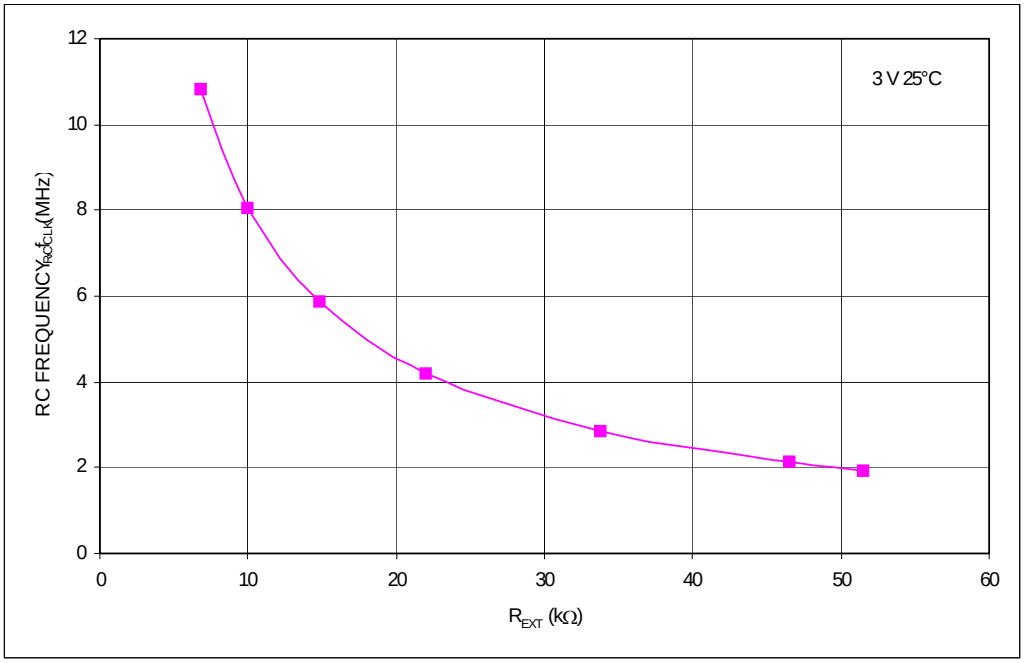



**Figure 16-8. RC versus Frequency (3 Volts @ 25•C)**

## 16.13 Supply Current Characteristics

| Characteristic[1] | Voltage | Bus Frequency (MHz) | Symbol | Typ[2] | Max | Unit |
|---|---|---|---|---|---|---|
| Run Mode $V_{DD}$ supply current[3] | 5.0<br>3.0 | 3.2<br>3.2 | $RI_{DD}$ | 6.0<br>2.5 | 7.0<br>3.2 | mA |
| Wait Mode $V_{DD}$ supply current[4] | 5.0<br>3.0 | 3.2<br>3.2 | $WI_{DD}$ | 1.0<br>0.67 | 1.5<br>1.0 | mA<br>mA |
| Stop Mode $V_{DD}$ supply current[5]<br>–40 to 85•C<br>–40 to 105•C<br>–40 to 125•C<br>25•C with auto wakeup enabled<br>Incremental current with LVI enabled at 25•C | 5.0 | | $SI_{DD}$ | 0.04<br>—<br>—<br>7<br>125 | 1.0<br>2.0<br>5.0<br>—<br>— | µA |
| Stop Mode $V_{DD}$ supply current[5]<br>–40 to 85•C<br>–40 to 105•C<br>–40 to 125•C<br>25•C with auto wakeup enabled<br>Incremental current with LVI enabled at 25•C | 3.0 | | $SI_{DD}$ | 0.02<br>—<br>—<br>5<br>100 | 0.5<br>1.0<br>4.0<br>—<br>— | µA |

1. $V_{SS}$ = 0 Vdc, $T_A = T_L$ to $T_H$, unless otherwise noted.
2. Typical values reflect average measurements at 25•C only.
3. Run (operating) $I_{DD}$ measured using trimmed internal oscillator, ADC off, all other modules enabled. All pins configured as inputs and tied to 0.2 V from rail.
4. Wait $I_{DD}$ measured using trimmed internal oscillator, ADC off, all other modules enabled. All pins configured as inputs and tied to 0.2 V from rail.
5. Stop $I_{DD}$ measured with all pins tied to 0.2 V or less from rail. No dc loads. On the 8-pin versions, port B is configured as inputs with pullups enabled.

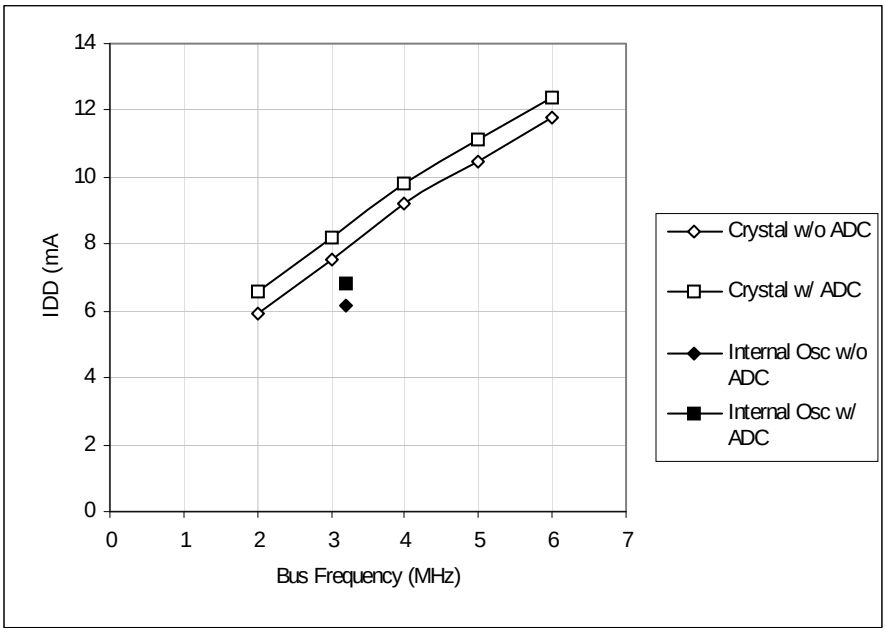



**Figure 16-9. Typical 5-Volt Run Current versus Bus Frequency (25•C)**

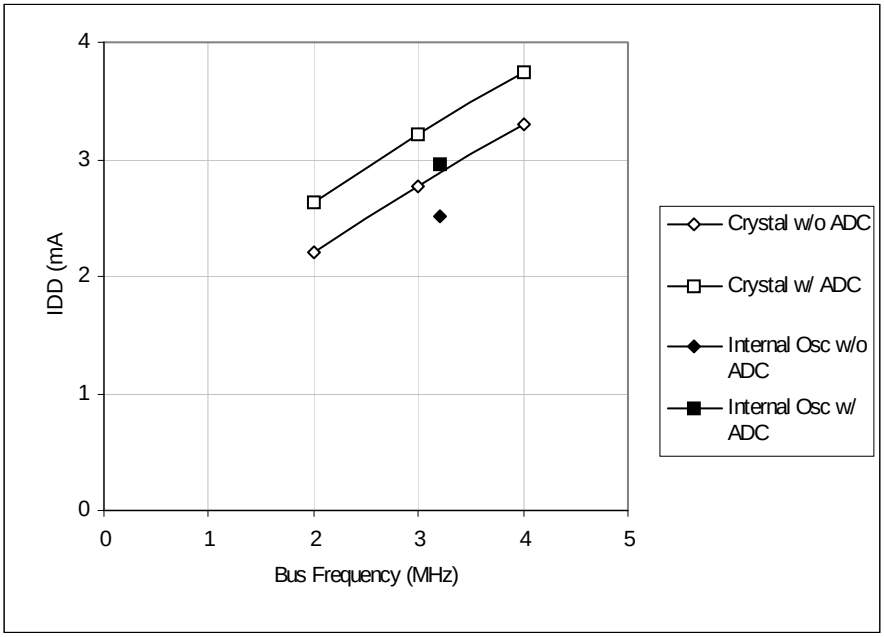



**Figure 16-10. Typical 3-Volt Run Current versus Bus Frequency (25•C)**

## 16.14 Analog-to-Digital Converter Characteristics

| 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 (1 LSB) | RES | 10.5 | 21.5 | mV | — |
| Absolute accuracy (Total unadjusted error) | $E_{TUE}$ | — | ± 1.5 | LSB | Includes quantization |
| ADC internal clock | $f_{ADIC}$ | 0.5 | 1.048 | MHz | $t_{ADIC} = 1/f_{ADIC}$, tested only at 1 MHz |
| Conversion range | $V_{AIN}$ | $V_{SS}$ | $V_{DD}$ | V | — |
| Power-up time | $t_{ADPU}$ | 16 | — | $t_{ADIC}$ cycles | $t_{ADIC} = 1/f_{ADIC}$ |
| Conversion time | $t_{ADC}$ | 16 | 17 | $t_{ADIC}$ cycles | $t_{ADIC} = 1/f_{ADIC}$ |
| Sample time[1] | $t_{ADS}$ | 5 | — | $t_{ADIC}$ cycles | $t_{ADIC} = 1/f_{ADIC}$ |
| 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 | $C_{ADI}$ | — | 8 | pF | Not tested |
| Input leakage[3] | $I_{IL}$ | — | ± 1 | μA | — |
| ADC supply current<br>$V_{DD}$ = 3 V<br>$V_{DD}$ = 5 V | $I_{ADAD}$ | Typical = 0.45<br>Typical = 0.65 | | mA<br>mA | Enabled<br>Enabled |

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.

## 16.15 Timer Interface Module Characteristics

| Characteristic | Symbol | Min | Max | Unit |
|---|---|---|---|---|
| Timer input capture pulse width | $t_{TH}$, $t_{TL}$ | 2 | — | $t_{cyc}$ |
| Timer input capture period | $t_{TLTL}$ | Note(1) | — | $t_{cyc}$ |
| Timer input clock pulse width | $t_{TCL}$, $t_{TCH}$ | $t_{cyc}$ + 5 | — | ns |

1. The minimum period is the number of cycles it takes to execute the interrupt service routine plus 1 $t_{cyc}$.

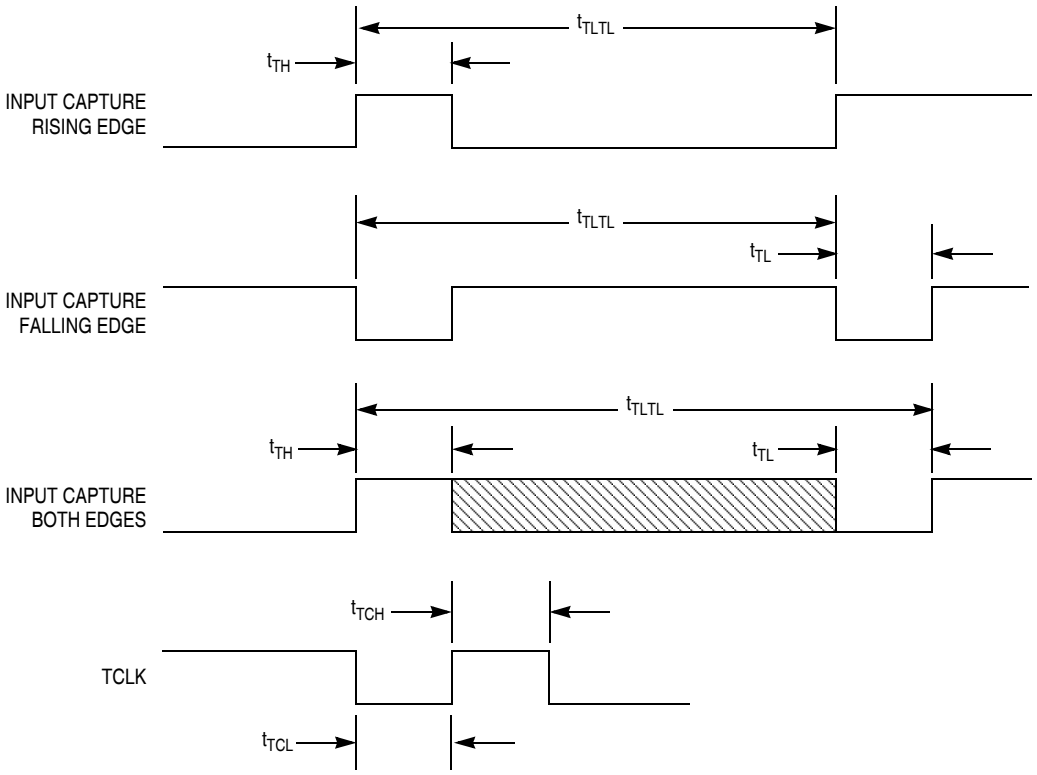



**Figure 16-11. Timer Input Timing**

## 16.16 Memory Characteristics

| Characteristic | Symbol | Min | Typ | 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) | 0 | — | 8 M | Hz |
| FLASH page erase time<br><1 k cycles<br>>1 k cycles | $t_{Erase}$ | <br>0.9<br>3.6 | <br>1<br>4 | <br>1.1<br>5.5 | ms |
| FLASH mass erase time | $t_{MErase}$ | 4 | — | — | ms |
| FLASH PGM/ERASE to HVEN setup 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 | $t_{RCV}$(2) | 1 | — | — | µs |
| FLASH cumulative program HV period | $t_{HV}$(3) | — | — | 4 | ms |
| FLASH endurance(4) | — | 10 k | 100 k | — | Cycles |
| FLASH data retention time(5) | — | 15 | 100 | — | Years |

1. $f_{Read}$ is defined as the frequency range for which the FLASH memory can be read.
2. $t_{RCV}$ 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 0.
3. $t_{HV}$ is defined as the cumulative high voltage programming time to the same row before next erase. $t_{HV}$ must satisfy this condition: $t_{NVS} + t_{NVH} + t_{PGS} + (t_{PROG} \times 32) \leq t_{HV}$ maximum.
4. Typical endurance was evaluated for this product family. For additional information on how Freescale defines *Typical Endurance*, please refer to Engineering Bulletin EB619.
5. Typical data retention values are based on intrinsic capability of the technology measured at high temperature and de-rated to 25•C using the Arrhenius equation. For additional information on how Freescale defines *Typical Data Retention*, please refer to Engineering Bulletin EB618.

# Chapter 17
# Ordering Information and Mechanical Specifications

## 17.1 Introduction

This section contains order numbers for the MC68HC908QY1, MC68HC908QY2, MC68HC908QY4, MC68HC908QT1, MC68HC908QT2, and MC69HC908QT4. Dimensions are given for:

- 8-pin plastic dual in-line package (PDIP)
- 8-pin small outline integrated circuit (SOIC) package
- 8-pin dual flat no lead (DFN) package
- 16-pin PDIP
- 16-pin SOIC
- 16-pin thin shrink small outline package (TSSOP)

## 17.2 MC Order Numbers

**Table 17-1. MC Order Numbers**

| MC Order Number | ADC | FLASH Memory | Package |
|---|---|---|---|
| MC908QY1 | — | 1536 bytes | 16-pins PDIP, SOIC, and TSSOP |
| MC908QY2 | Yes | 1536 bytes | |
| MC908QY4 | Yes | 4096 bytes | |
| MC908QT1 | — | 1536 bytes | 8-pins PDIP, SOIC, and DFN |
| MC908QT2 | Yes | 1536 bytes | |
| MC908QT4 | Yes | 4096 bytes | |

Temperature and package designators:
C = –40•C to +85•C
V = –40•C to +105•C
M = –40•C to +125•C
P = Plastic dual in-line package (PDIP)
DW = Small outline integrated circuit package (SOIC)
DT = Thin shrink small outline package (TSSOP)
FQ = Dual flat no lead (DFN)

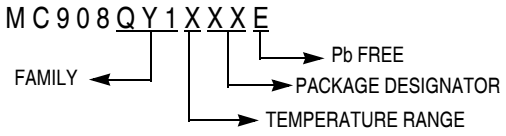
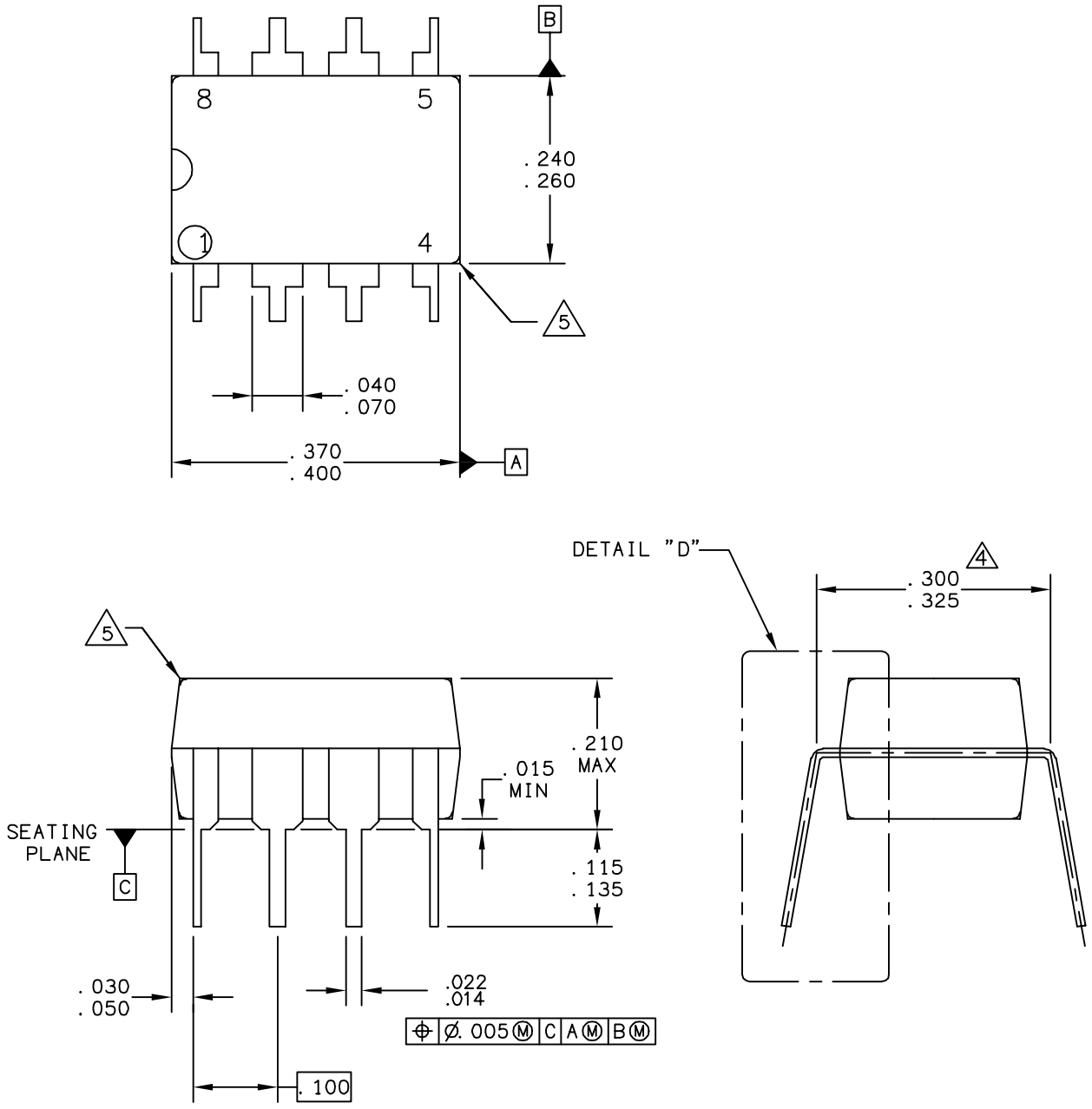


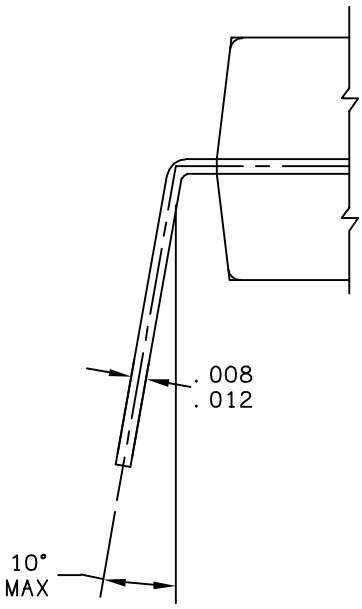

**Figure 17-1. Device Numbering System**

## 17.3 Package Dimensions

Refer to the following pages for detailed package dimensions.




| | | |
|---|---|---|
| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | **MECHANICAL OUTLINE** | PRINT VERSION NOT TO SCALE |
| TITLE: 8 LD PDIP | DOCUMENT NO: 98ASB42420B | REV: N |
| | CASE NUMBER: 626-06 | 19 MAY 2005 |
| | STANDARD: NON-JEDEC | |




DETAIL "D"

© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

| MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|

| TITLE: | DOCUMENT NO: 98ASB42420B | REV: N |
|---|---|---|
| 8 LD PDIP | CASE NUMBER: 626-06 | 19 MAY 2005 |
| | STANDARD: NON-JEDEC | |

NOTES:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M – 1994.
2. ALL DIMENSIONS ARE IN INCHES.
3. 626-03 TO 626-06 OBSOLETE. NEW STANDARD 626-07.
4. DIMENSION TO CENTER OF LEAD WHEN FORMED PARALLEL.
5. PACKAGE CONTOUR OPTIONAL (ROUND OR SQUARE CONERS).

STYLE 1:

| | | | | |
|---|---|---|---|---|
| PIN | 1. | AC IN | 5. | GROUND |
| | 2. | DC + IN | 6. | OUTPUT |
| | 3. | DC – IN | 7. | AUXILIARY |
| | 4. | AC IN | 8. | VCC |

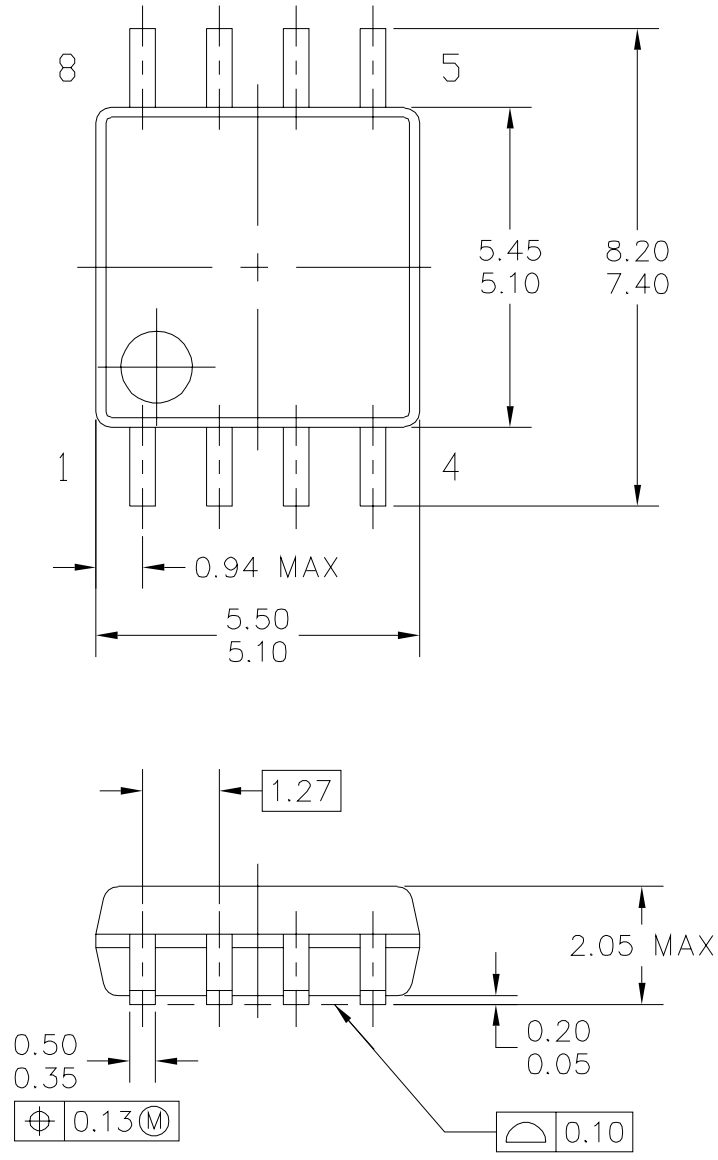
© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

| MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|

| TITLE: | DOCUMENT NO: 98ASB42420B | REV: N |
|---|---|---|
| 8 LD PDIP | CASE NUMBER: 626-06 | 19 MAY 2005 |
| | STANDARD: NON-JEDEC | |

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






TITLE:

8 LEAD MFP

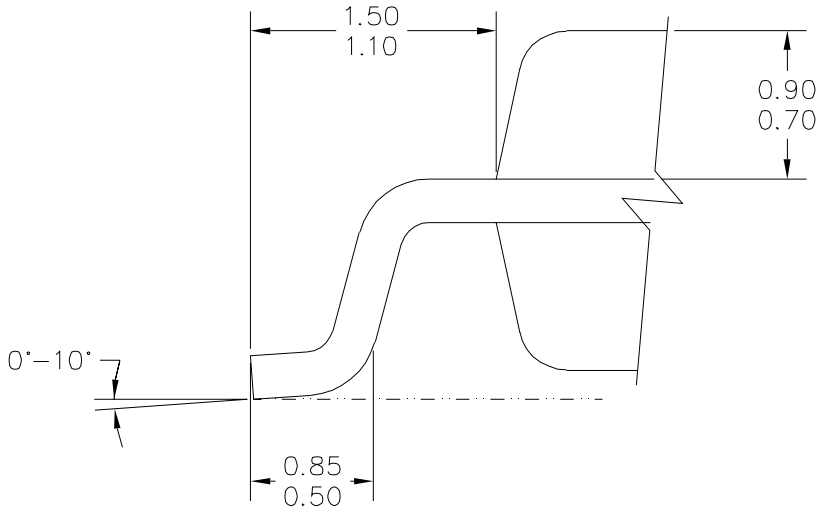
CASE NUMBER: 968-02

STANDARD: EIAJ

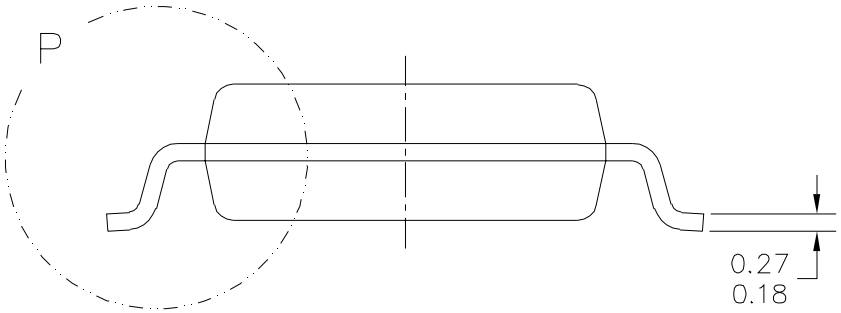
PACKAGE CODE: 6003

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






DETAIL P




TITLE:

8 LEAD MFP

CASE NUMBER: 968-02

STANDARD: EIAJ

PACKAGE CODE: 6003

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.



NOTES:

1. DIMENSIONS AND TOLERANCES PER ASME Y14.5–1994.

2. CONTROLLING DIMENSION: MILLIMETER.

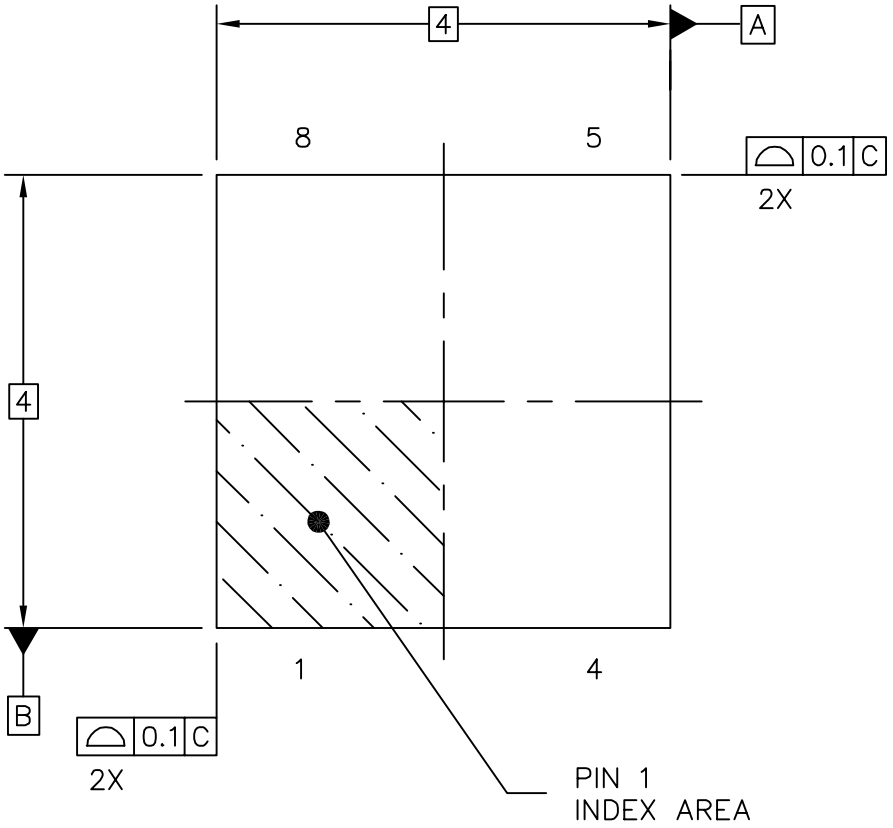
3. DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS AND ARE MEASURED AT THE PARTING LINE. MOLD FLASH OR PROTRUSIONS SHALL NOT EXCEDD 0.15mm PER SIDE.

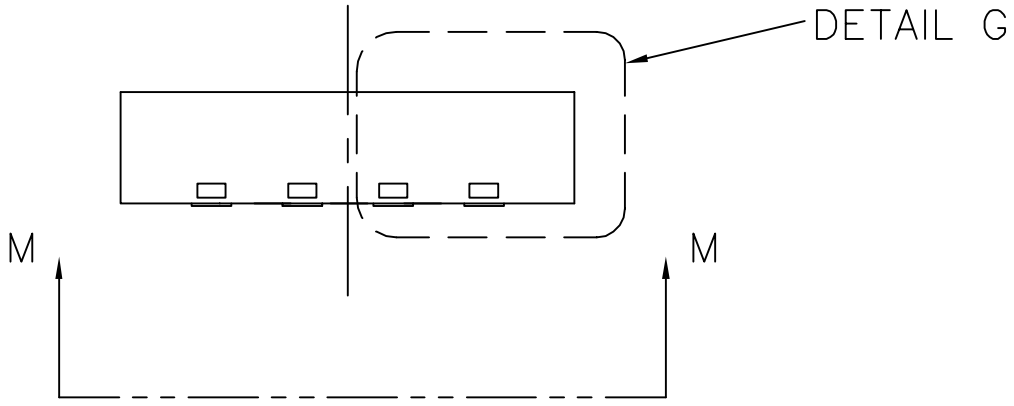
4. TERMINAL NUMBERS ARE SHOWN FOR REFERENCE ONLY.

5. THE LEAD WIDTH DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION SHALL BE 0.08mm TOTAL IN EXCESS OF THE LEAD WIDTH DIMENSION AT MAXIMUM MATERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSIONS AND ADJACENT LEAD TO BE 0.46mm.

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






TITLE: THERMALLY ENHANCED DUAL FLAT NO LEAD PACKAGE (DFN) 8 TERMINAL, 0.8 PITCH (4 X 4 X 1)

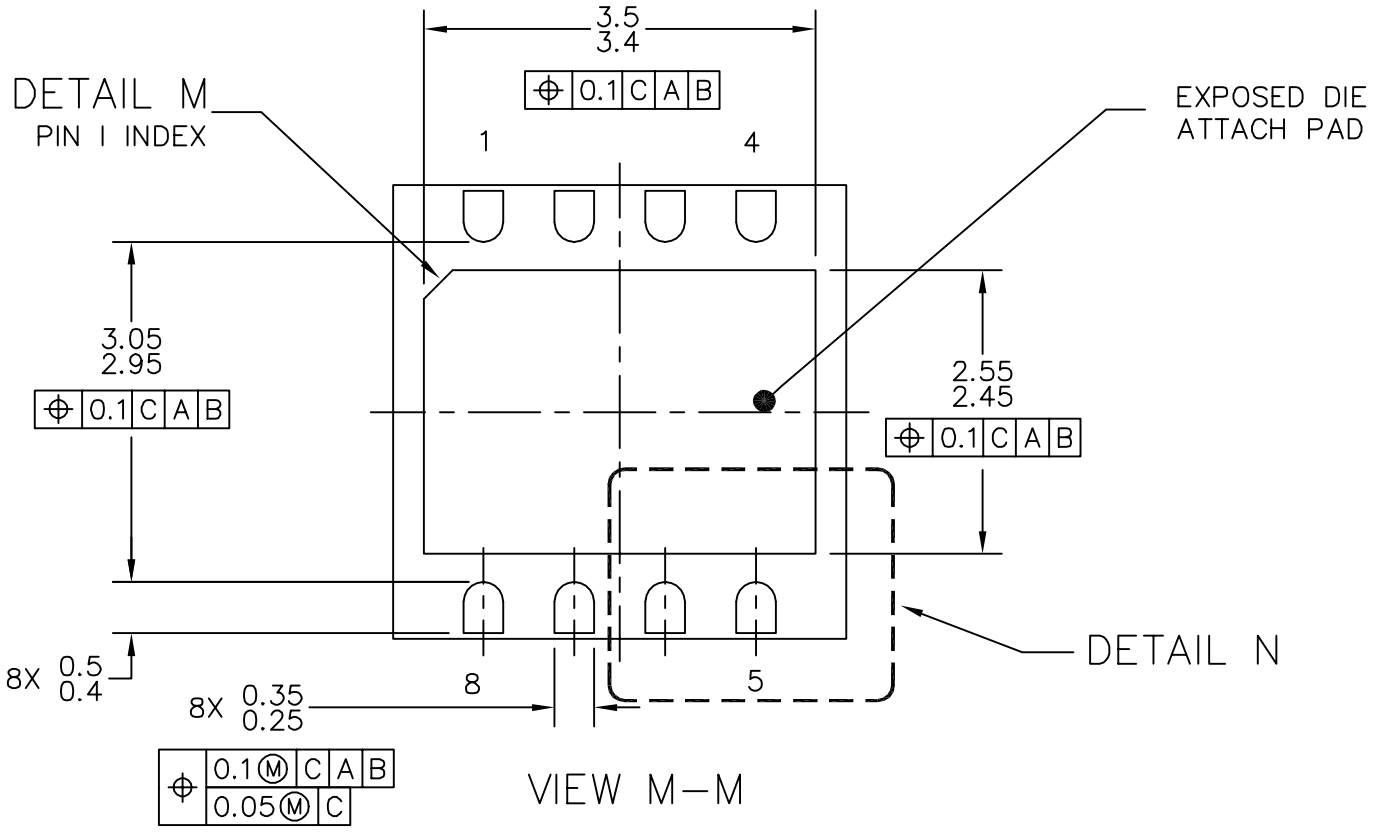
CASE NUMBER: 1452-01

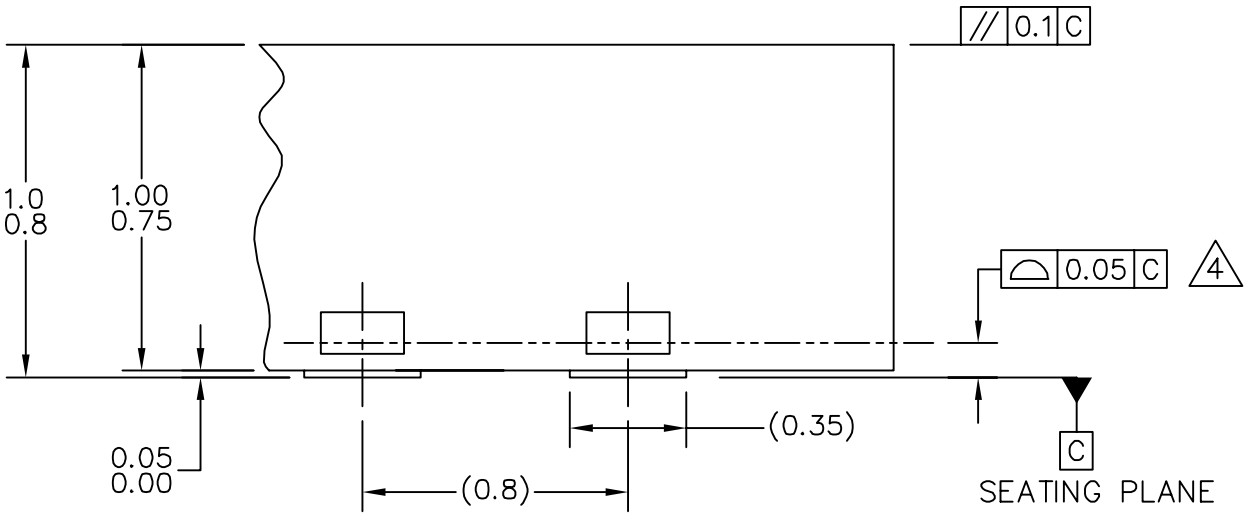
STANDARD: NON-JEDEC

PACKAGE CODE: 6165

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






DETAIL G

TITLE: THERMALLY ENHANCED DUAL FLAT NO LEAD PACKAGE (DFN) 8 TERMINAL, 0.8 PITCH (4 X 4 X 1)

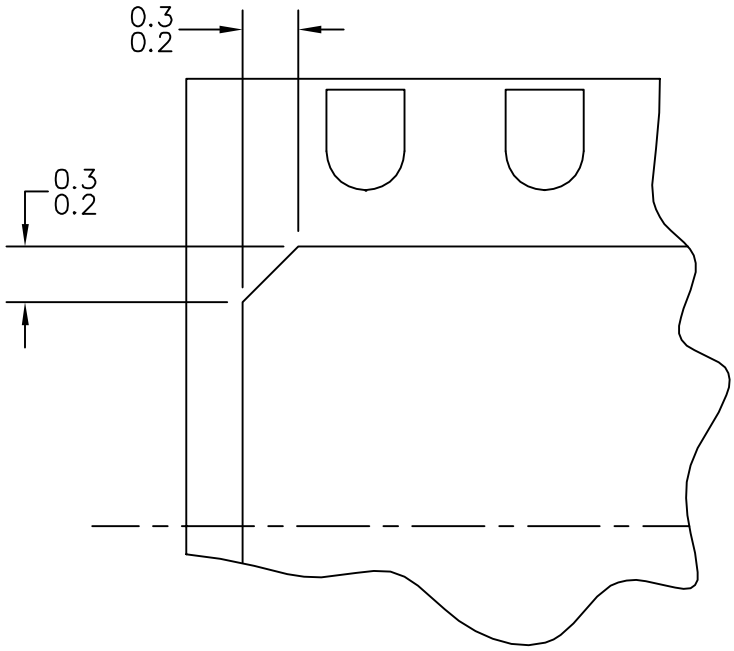
CASE NUMBER: 1452–01

STANDARD: NON–JEDEC

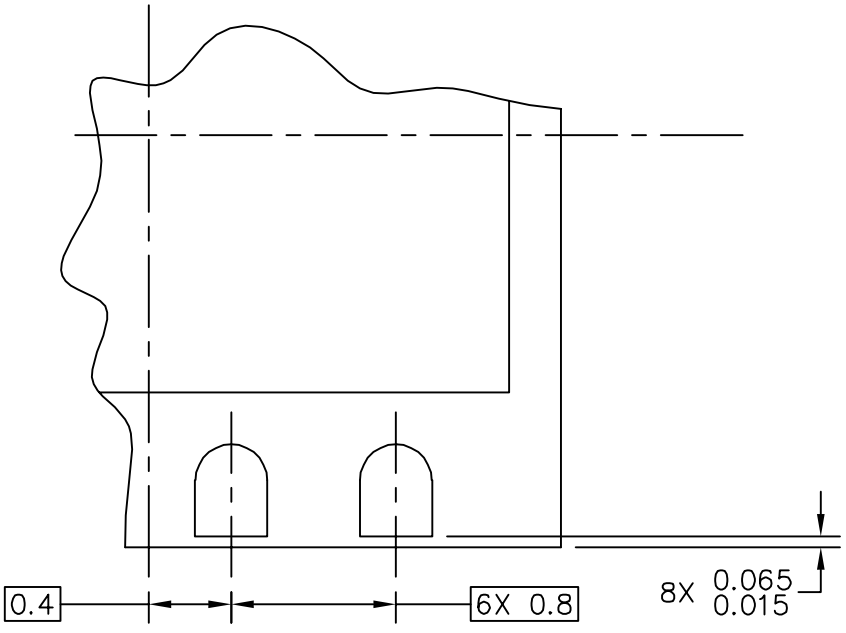
PACKAGE CODE: 6165

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






DETAIL M
BACKSIDE PIN 1 INDEX




DETAIL N

TITLE: THERMALLY ENHANCED DUAL FLAT NO LEAD PACKAGE (DFN) 8 TERMINAL, 0.8 PITCH (4 X 4 X 1)

CASE NUMBER: 1452-01
STANDARD: NON-JEDEC
PACKAGE CODE: 6165
SHEET: 3 OF 5

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.

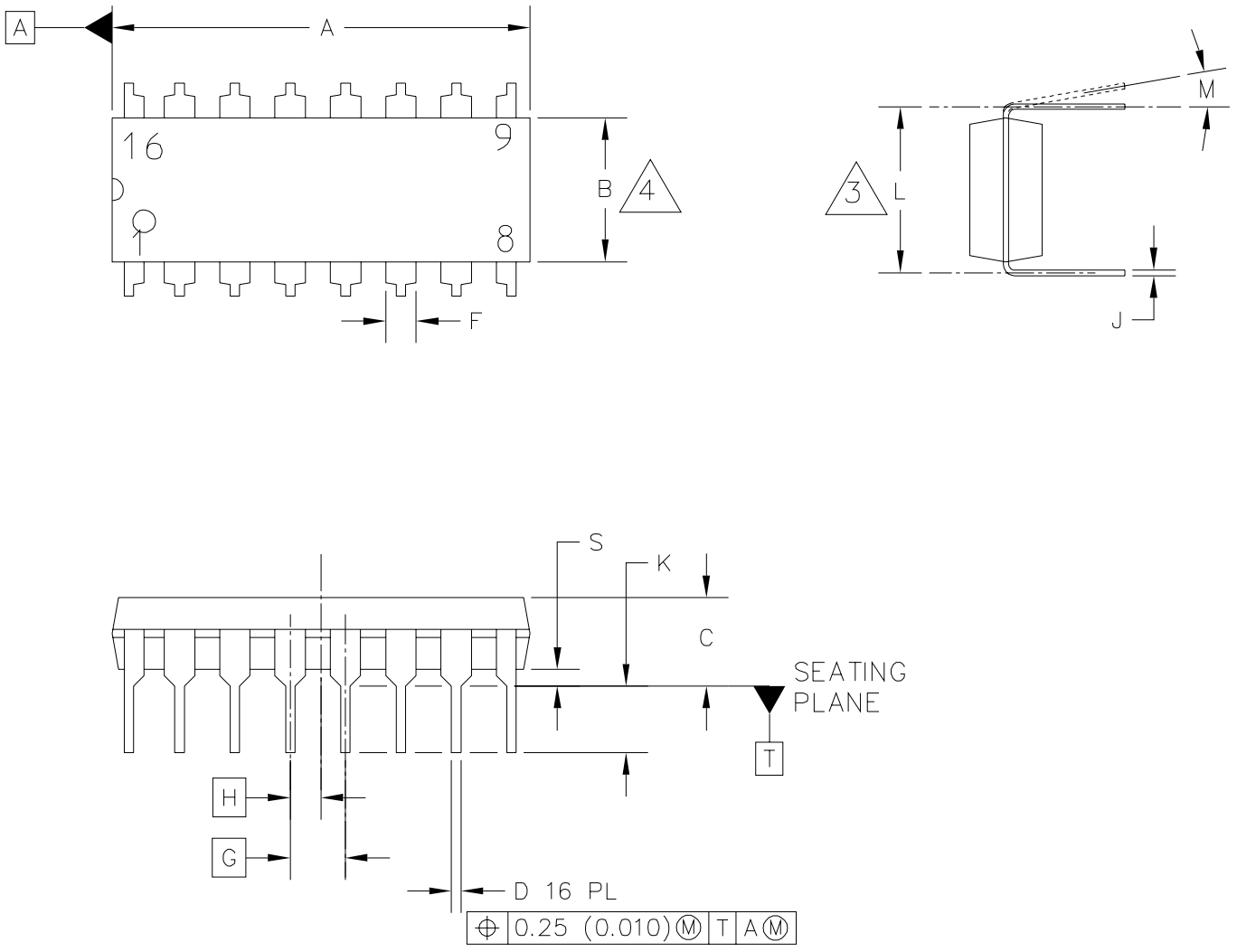


NOTES:

1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
3. THE COMPLETE JEDEC DESIGNATOR FOR THIS PACKAGE IS: HP–VFDFP–N.
4. COPLANARITY APPLIES TO LEADS AND DIE ATTACH PAD.

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






TITLE:

16 LD PDIP

CASE NUMBER: 648-08

STANDARD: NON-JEDEC

PACKAGE CODE: 0006

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.



NOTES:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M, 1994.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION TO CENTER OF LEADS WHEN FORMED PARALLEL.
4. DIMENSIONS DOES NOT INCLUDE MOLD FLASH.
5. ROUNDED CORNERS OPTIONAL.
6. 648-01 THRU -08 OBSOLETE, NEW STANDARD 648-09.

| DIM | MILLIMETERS MIN | MILLIMETERS MAX | INCHES MIN | INCHES MAX | DIM | MILLIMETERS MIN | MILLIMETERS MAX | INCHES MIN | INCHES MAX |
|---|---|---|---|---|---|---|---|---|---|
| A | 18.80 | 19.55 | 0.740 | 0.770 | | | | | |
| B | 6.35 | 6.85 | 0.250 | 0.270 | | | | | |
| C | 3.69 | 4.44 | 0.145 | 0.175 | | | | | |
| D | 0.39 | 0.53 | 0.015 | 0.021 | | | | | |
| F | 1.02 | 1.77 | 0.040 | 0.070 | | | | | |
| G | 2.54 BSC | | 0.100 BSC | | | | | | |
| H | 1.27 BSC | | 0.050 BSC | | | | | | |
| J | 0.21 | 0.38 | 0.008 | 0.015 | | | | | |
| K | 2.80 | 3.30 | 0.110 | 0.130 | | | | | |
| L | 7.50 | 7.74 | 0.295 | 0.305 | | | | | |
| M | 0° | 10° | 0° | 10° | | | | | |
| S | 0.51 | 1.01 | 0.020 | 0.040 | | | | | |

TITLE:

16 LD PDIP

CASE NUMBER: 648-08

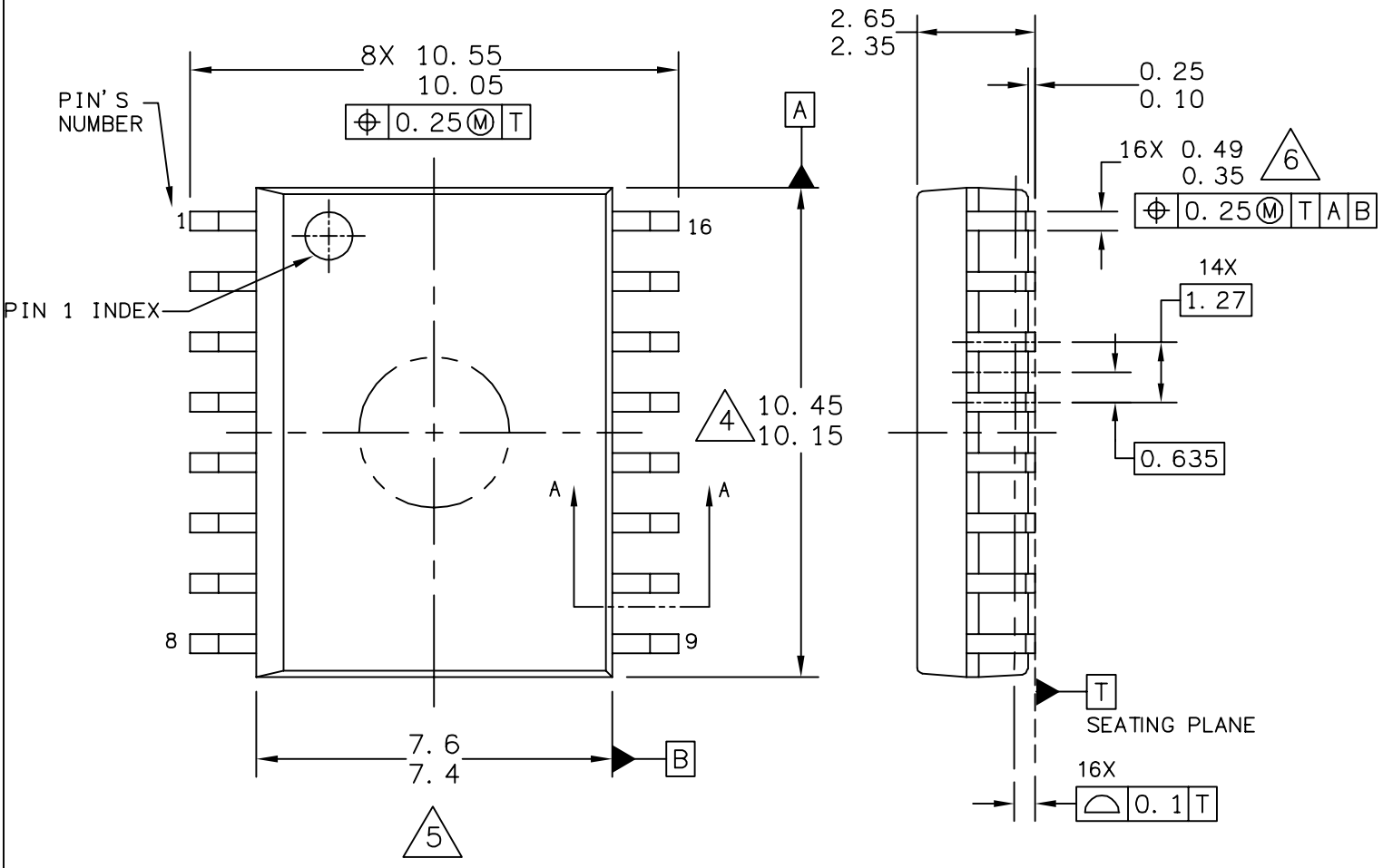
STANDARD: NON-JEDEC

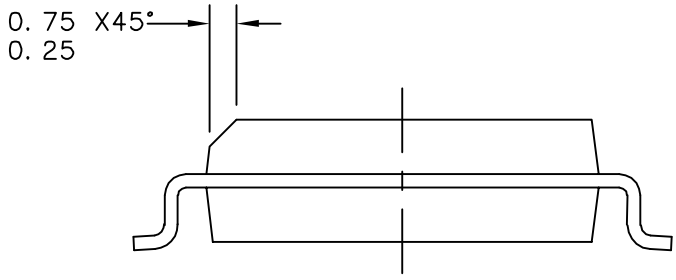
PACKAGE CODE: 0006

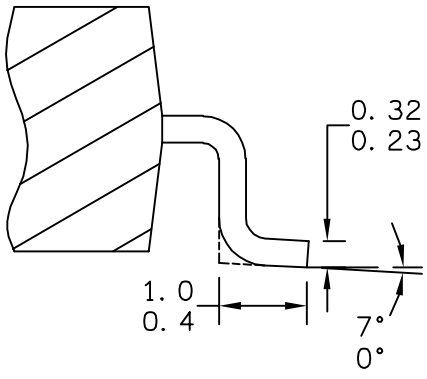
freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.

# MECHANICAL OUTLINES DICTIONARY

DOCUMENT NO: 98ASB42567B
PAGE: 751G
DO NOT SCALE THIS DRAWING
REV: E




SECTION A–A

TITLE:
16LD SOIC W/B, 1.27 PITCH
CASE-OUTLINE

CASE NUMBER: 751G-05
STANDARD: JEDEC MS-013AA
PACKAGE CODE: 2003
SHEET: 1 OF 3

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.

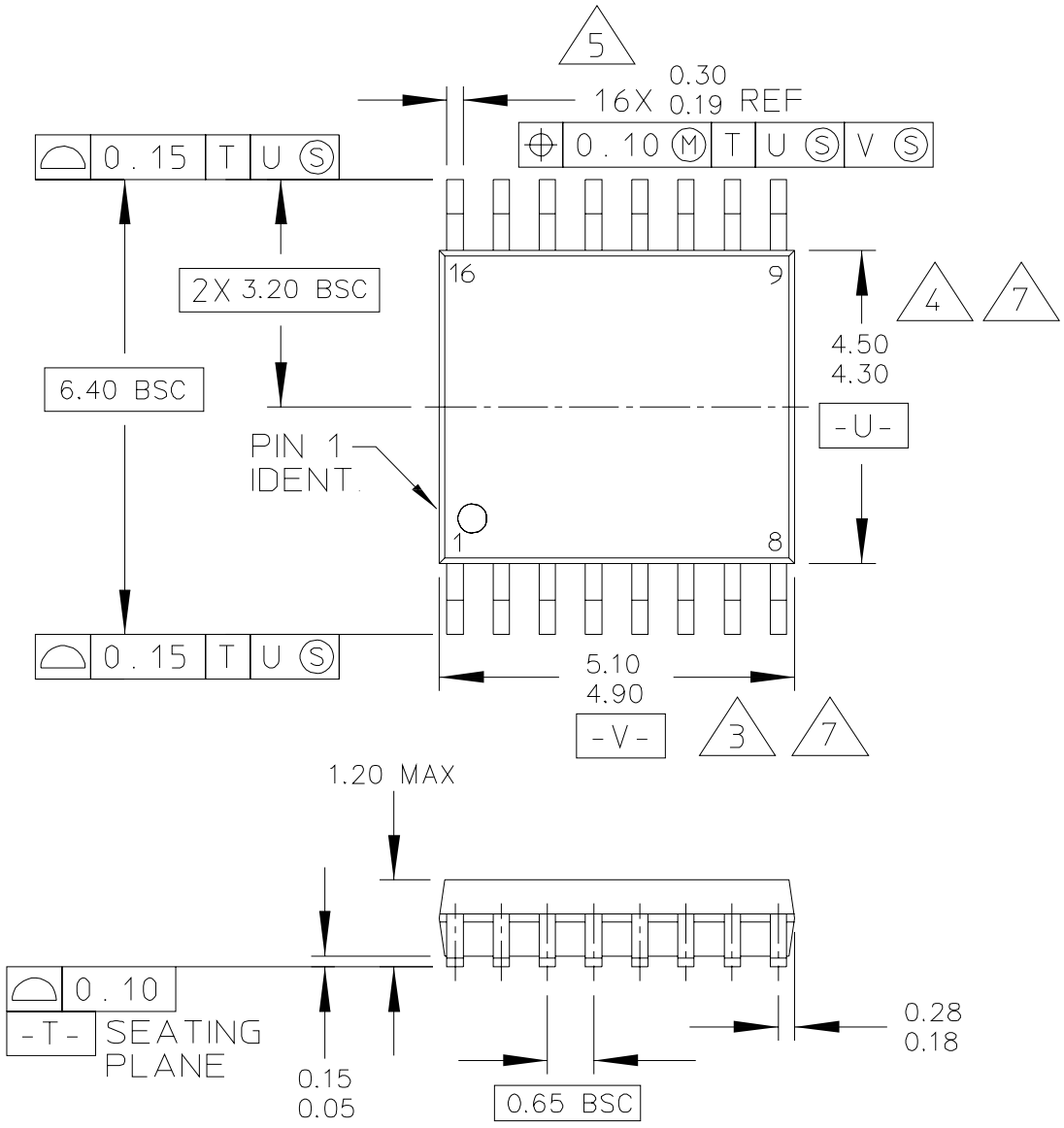


NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.
2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M–1994.
3. DATUMS A AND B TO BE DETERMINED AT THE PLANE WHERE THE BOTTOM OF THE LEADS EXIT THE PLASTIC BODY.
4. THIS DIMENSION DOES NOT INCLUDE MOLD FLASH, PROTRUSION OR GATE BURRS. MOLD FLASH, PROTRUSION OR GATE BURRS SHALL NOT EXCEED 0.15 MM PER SIDE. THIS DIMENSION IS DETERMINED AT THE PLANE WHERE THE BOTTOM OF THE LEADS EXIT THE PLASTIC BODY.
5. THIS DIMENSION DOES NOT INCLUDE INTER–LEAD FLASH OR PROTRUSIONS. INTER–LEAD FLASH AND PROTRUSIONS SHALL NOT EXCEED 0.25 MM PER SIDE. THIS DIMENSION IS DETERMINED AT THE PLANE WHERE THE BOTTOM OF THE LEADS EXIT THE PLASTIC BODY.
6. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.62 mm.

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






TITLE:

16 LD TSSOP, PITCH 0.65MM

CASE NUMBER: 948F-01

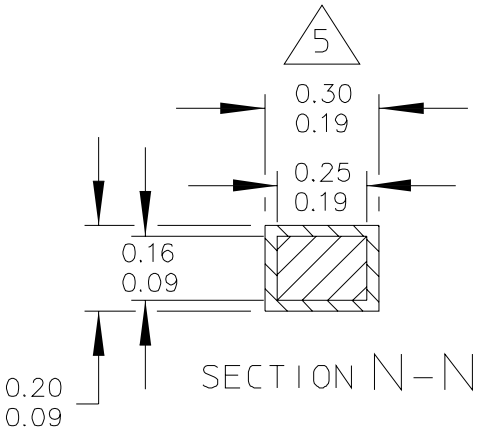
STANDARD: JEDEC

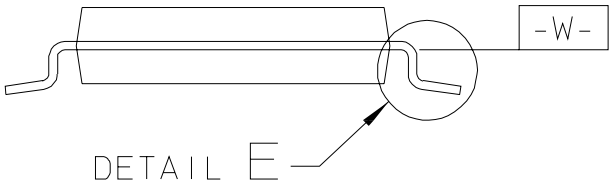
PACKAGE CODE: 6117

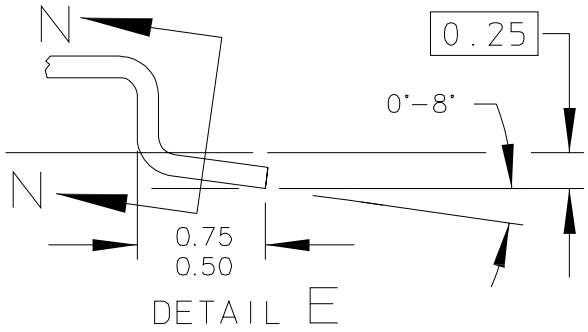
SHEET: 1 OF 4

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.






TITLE:

16 LD TSSOP, PITCH 0.65MM

CASE NUMBER: 948F-01

STANDARD: JEDEC

PACKAGE CODE: 6117

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.



NOTES:

1. CONTROLLING DIMENSION: MILLIMETER
2. DIMENSIONS AND TOLERANCES PER ANSI Y14.5M-1982.
3. DIMENSION DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH OR GATE BURRS SHALL NOT EXCEED 0.15 PER SIDE.
4. DIMENSION DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION. INTERLEAD FLASH OR PROTRUSION SHALL NOT EXCEED 0.25 PER SIDE.
5. DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08 TOTAL IN EXCESS OF THE DIMENSION AT MAXIMUM MATERIAL CONDITION.
6. TERMINAL NUMBERS ARE SHOWN FOR REFERENCE ONLY.
7. DIMENSIONS ARE TO BE DETERMINED AT DATUM PLANE -W-.

## *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. 2005–2010. All rights reserved.

MC68HC908QY4
Rev. 6, 03/2010

freescale™
semiconductor

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View MC68HC908QY2CDTE 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