



Welcome to [E-XFL.COM](http://E-XFL.COM)

### What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

### Applications of "[Embedded - Microcontrollers](#)"

Details	
Product Status	Not For New Designs
Core Processor	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	-
Peripherals	LVD, POR, PWM
Number of I/O	13
Program Memory Size	1.5KB (1.5K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 4x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	16-SOIC (0.295", 7.50mm Width)
Supplier Device Package	16-SOIC
Purchase URL	<a href="https://www.e-xfl.com/product-detail/nxp-semiconductors/mchc908qy2cdwe">https://www.e-xfl.com/product-detail/nxp-semiconductors/mchc908qy2cdwe</a>

## Revision History (Sheet 3 of 3)

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

## 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

# 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<sup>(1)</sup>
  - $\pm 25\%$  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<sup>(2)</sup>

1. The oscillator frequency is guaranteed to  $\pm 5\%$  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.

## 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.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0	
\$0022	TIM Counter Register Low (TCNTL) <a href="#">See page 128.</a>	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	0
\$0023	TIM Counter Modulo Register High (TMODH) <a href="#">See page 129.</a>	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	1
\$0024	TIM Counter Modulo Register Low (TMODL) <a href="#">See page 129.</a>	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	1
\$0025	TIM Channel 0 Status and Control Register (TSC0) <a href="#">See page 130.</a>	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX	
		Write:	0								
		Reset:	0	0	0	0	0	0	0	0	0
\$0026	TIM Channel 0 Register High (TCH0H) <a href="#">See page 132.</a>	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) <a href="#">See page 132.</a>	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) <a href="#">See page 130.</a>	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX	
		Write:	0								
		Reset:	0	0	0	0	0	0	0	0	0
\$0029	TIM Channel 1 Register High (TCH1H) <a href="#">See page 132.</a>	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) <a href="#">See page 132.</a>	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	Read:									
\$0036	Oscillator Status Register (OSCSTAT) <a href="#">See page 96.</a>	Read:	R	R	R	R	R	R	ECGON	ECGST	
		Write:									
		Reset:	0	0	0	0	0	0	0	0	0
\$0037	Unimplemented	Read:									
\$0038	Oscillator Trim Register (OSCTRIM) <a href="#">See page 96.</a>	Read:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0	
		Write:									
		Reset:	1	0	0	0	0	0	0	0	0

= Unimplemented     
  = Reserved     
 U = Unaffected

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

## 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

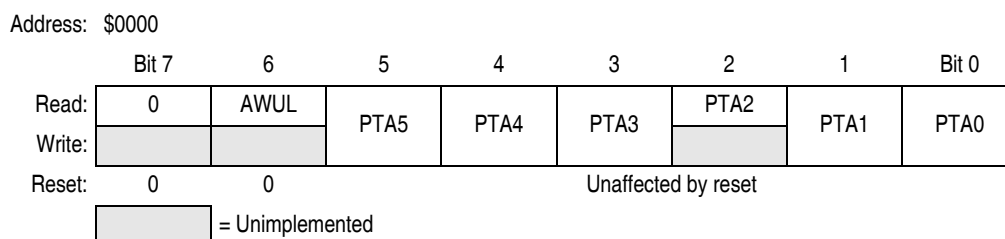
## 4.6 Input/Output Registers

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.



**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

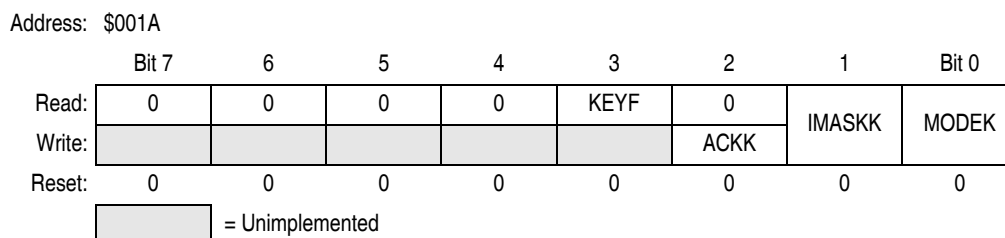
#### NOTE

*PTA5–PTA0 bits are not used in conjunction 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



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



## Auto Wakeup Module (AWU)

### 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.

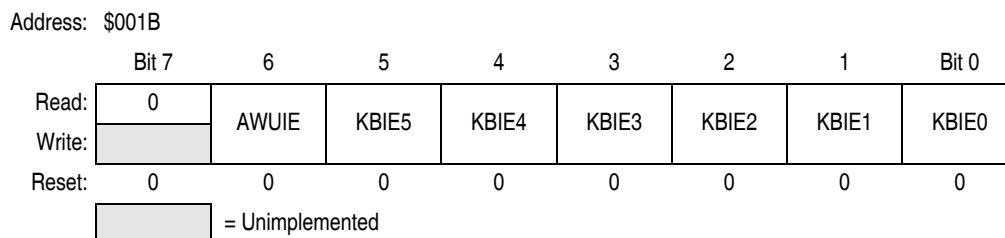
- 1 = Keyboard/auto wakeup interrupt requests masked
- 0 = Keyboard/auto wakeup interrupt requests not masked

#### NOTE

*MODEK is not used in conjunction 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.



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

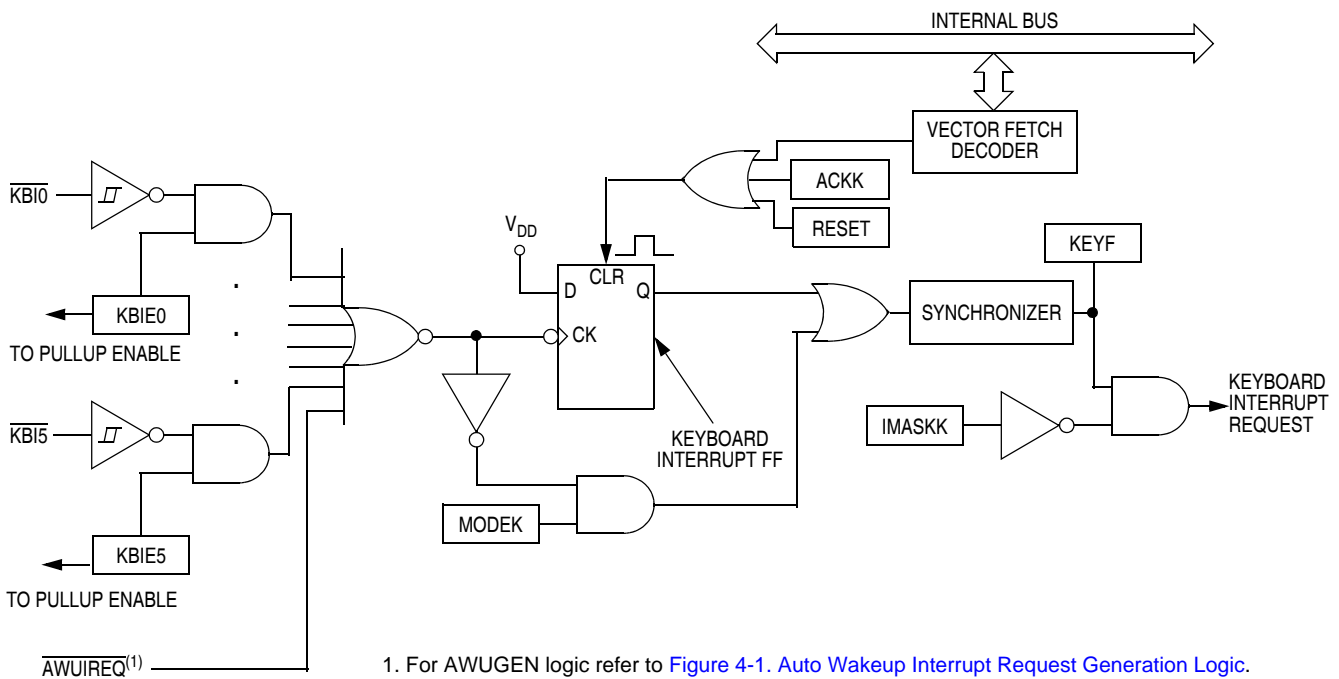
### AWUIE — Auto Wakeup Interrupt Enable Bit

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 conjunction with the auto wakeup feature. To see a description of these bits, see [9.7.2 Keyboard Interrupt Enable Register](#).*



**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.

## Keyboard Interrupt Module (KBI)

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 (KBIE<sub>x</sub>) 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 KBIE<sub>x</sub> 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 KBIE<sub>x</sub> 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.

# 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,  $V_{TRIPF}$ .

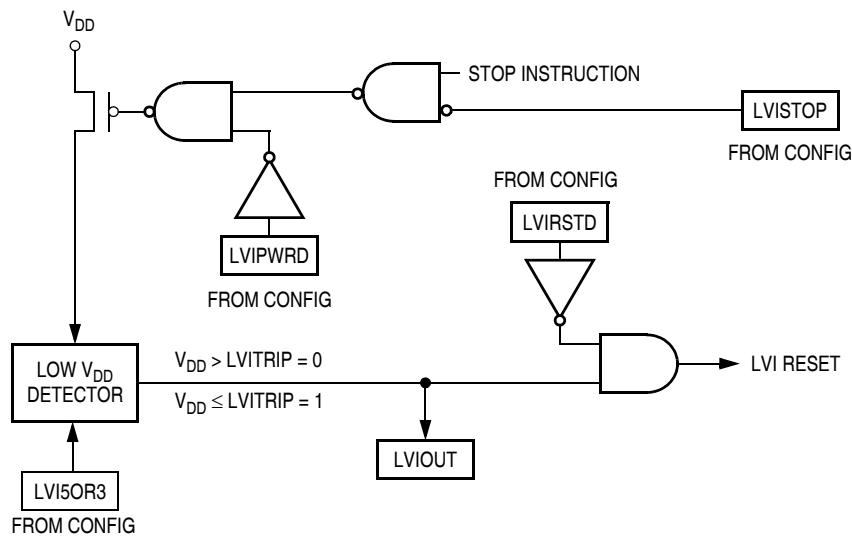
### 10.2 Features

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 \(CONFIG\)](#).

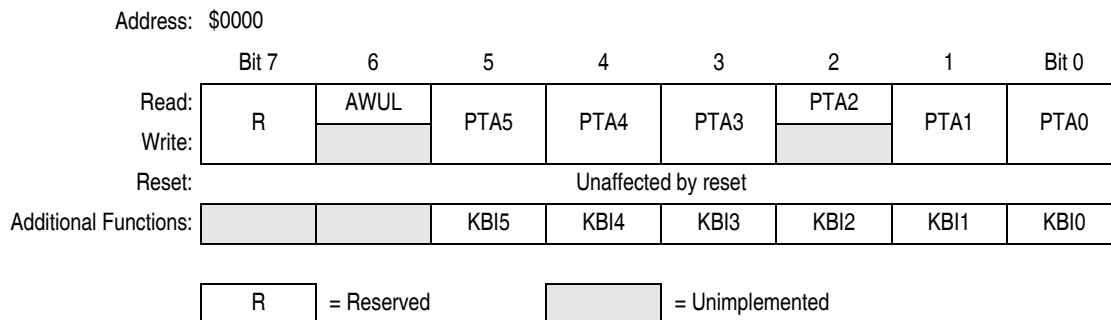


**Figure 10-1. LVI Module Block Diagram**

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,

### 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

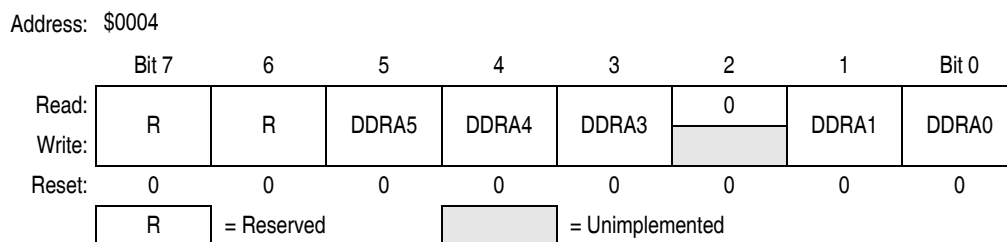
This is a read-only bit which has the value of the auto wakeup interrupt request latch. The wakeup request signal is generated internally (see [Chapter 4 Auto Wakeup Module \(AWU\)](#)). There is no PTA6 port nor any of the associated bits such as PTA6 data register, pullup enable or direction.

#### 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.*

### 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.

### 13.6.3 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.

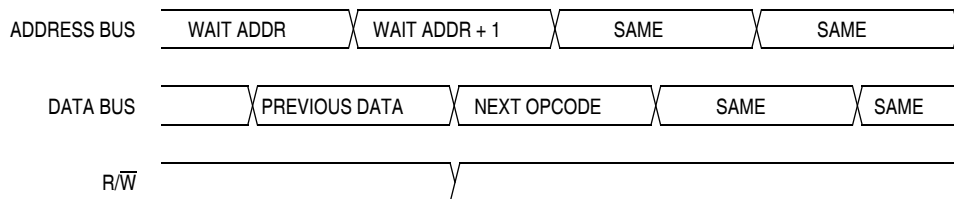
Setting the BCFE bit enables the clearing mechanisms. Once cleared in break mode, a flag remains cleared even when break mode is exited. Status flags with a two-step clearing mechanism — for example, a read of one register followed by the read or write of another — are protected, even when the first 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.



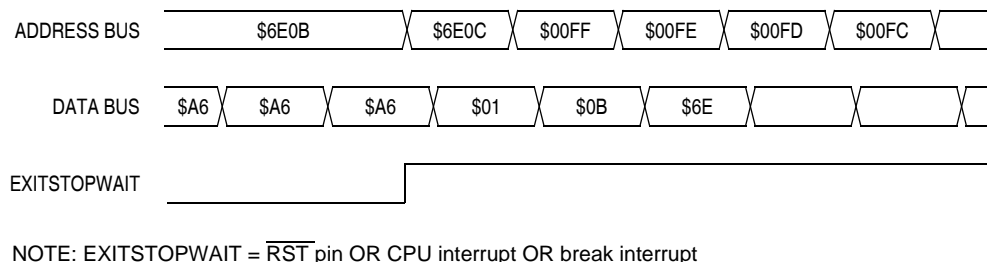
NOTE: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

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

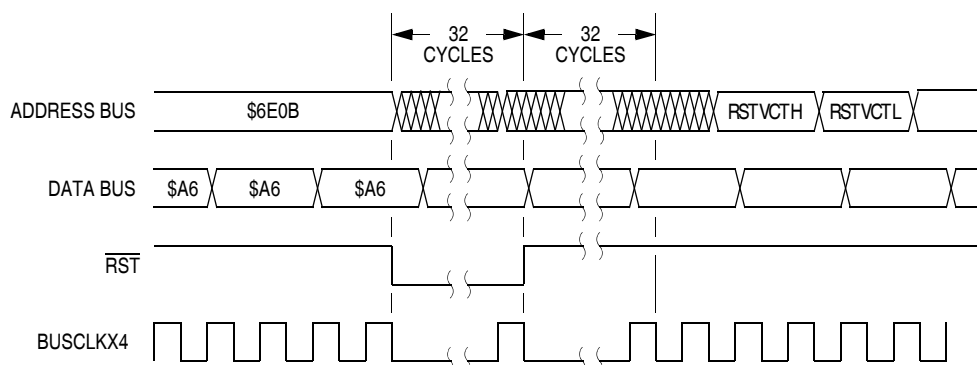
A module that is active during wait mode can wake up the CPU with an interrupt if the interrupt is enabled. Stacking for the interrupt begins one cycle after the WAIT instruction during which the interrupt occurred. In wait mode, the CPU clocks are inactive. Refer to the wait mode subsection of each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



**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.*



# 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**

<b>TIM Generic Pin Names:</b>	<b>TCH0</b>	<b>TCH1</b>	<b>TCLK</b>
<b>Full TIM Pin Names:</b>	PTA0/TCH0	PTA1/TCH1	PTA2/TCLK


## 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

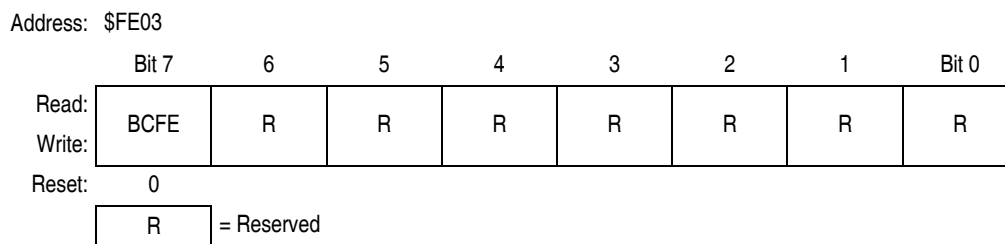
#### **NOTE**

*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.*

### 15.2.2.5 Break Flag Control Register

The break control register (BF CR) 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 (BF CR)**

#### 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.

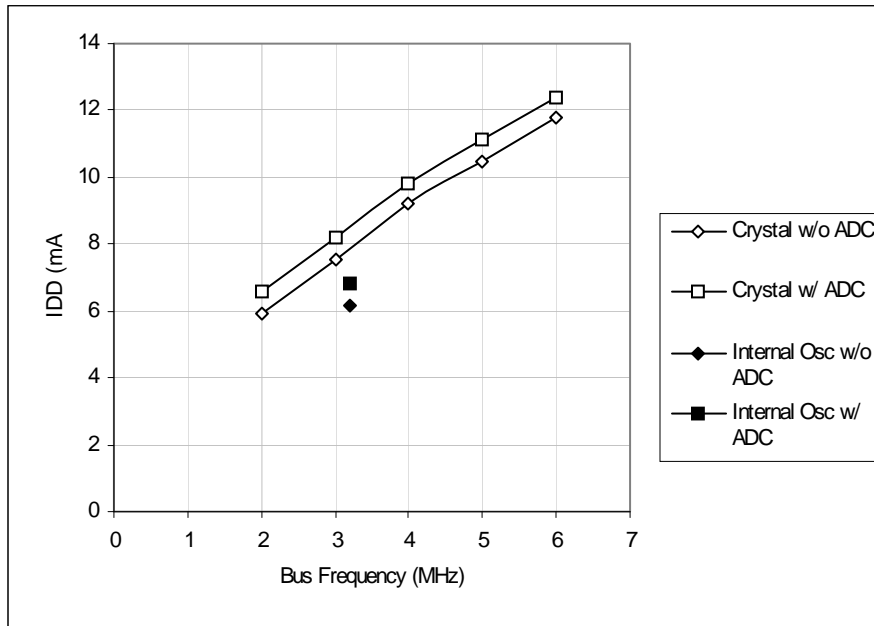
## 15.3 Monitor Module (MON)

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.

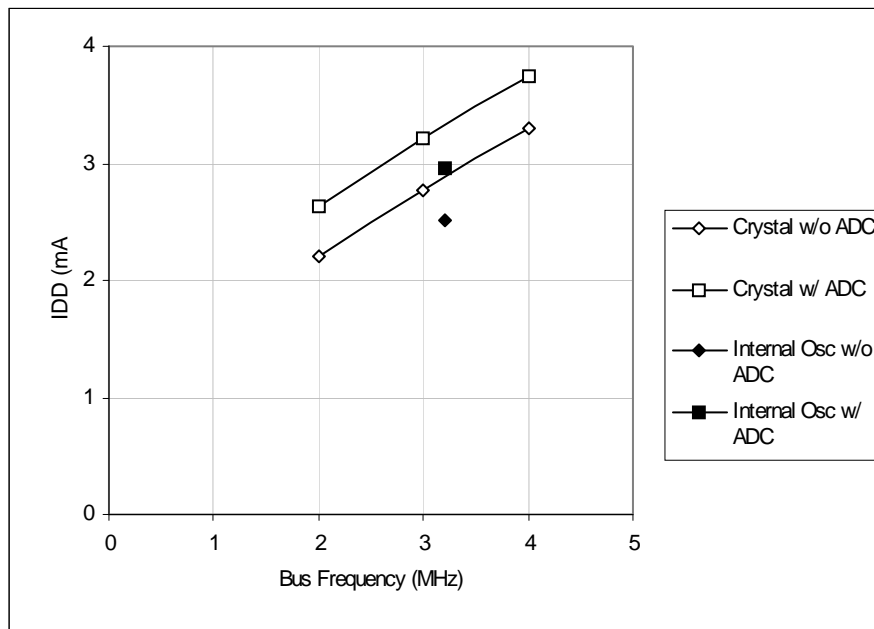
Features include:

- 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<sup>(1)</sup>
- 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.



**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)**

