

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	Active
Core Processor	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	-
Peripherals	LED, LVD, POR, PWM
Number of I/O	23
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 12x8b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	<a href="https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=mc908jl3emdwe">https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=mc908jl3emdwe</a>

## Chapter 6 Oscillator (OSC)

6.1	Introduction . . . . .	67
6.2	X-tal Oscillator (MC68HC908JL3E/JK3E/JK1E) . . . . .	67
6.3	RC Oscillator (MC68HRC908JL3E/JK3E/JK1E) . . . . .	67
6.4	I/O Signals . . . . .	69
6.4.1	Crystal Amplifier Input Pin (OSC1) . . . . .	69
6.4.2	Crystal Amplifier Output Pin (OSC2/PTA6/RCCLK) . . . . .	69
6.4.3	Oscillator Enable Signal (SIMOSCEN) . . . . .	69
6.4.4	X-tal Oscillator Clock (XTALCLK) . . . . .	69
6.4.5	RC Oscillator Clock (RCCLK) . . . . .	69
6.4.6	Oscillator Out 2 (2OSCOUT) . . . . .	69
6.4.7	Oscillator Out (OSCOUT) . . . . .	69
6.5	Low Power Modes . . . . .	70
6.5.1	Wait Mode . . . . .	70
6.5.2	Stop Mode . . . . .	70
6.6	Oscillator During Break Mode . . . . .	70

## Chapter 7 Monitor ROM (MON)

7.1	Introduction . . . . .	71
7.2	Features . . . . .	71
7.3	Functional Description . . . . .	71
7.3.1	Entering Monitor Mode . . . . .	73
7.3.2	Baud Rate . . . . .	75
7.3.3	Data Format . . . . .	76
7.3.4	Echoing . . . . .	76
7.3.5	Break Signal . . . . .	76
7.3.6	Commands . . . . .	77
7.4	Security . . . . .	79

## Chapter 8 Timer Interface Module (TIM)

8.1	Introduction . . . . .	81
8.2	Features . . . . .	81
8.3	Pin Name Conventions . . . . .	81
8.4	Functional Description . . . . .	82
8.4.1	TIM Counter Prescaler . . . . .	84
8.4.2	Input Capture . . . . .	84
8.4.3	Output Compare . . . . .	84
8.4.3.1	Unbuffered Output Compare . . . . .	84
8.4.3.2	Buffered Output Compare . . . . .	84
8.4.4	Pulse Width Modulation (PWM) . . . . .	85
8.4.4.1	Unbuffered PWM Signal Generation . . . . .	86
8.4.4.2	Buffered PWM Signal Generation . . . . .	86
8.4.4.3	PWM Initialization . . . . .	87

## 1.2 Features

Features of the MC68H(R)C908JL3E include the following:

- EMC enhanced version of MC68H(R)C908JL3/JK3/JK1
- High-performance M68HC08 architecture
- Fully upward-compatible object code with M6805, M146805, and M68HC05 Families
- Low-power design; fully static with stop and wait modes
- Maximum internal bus frequency:
  - 8-MHz at 5V operating voltage
  - 4-MHz at 3V operating voltage
- Oscillator options:
  - Crystal oscillator for MC68HC908JL3E/JK3E/JK1E
  - RC oscillator for MC68HRC908JL3E/JK3E/JK1E
- User program Flash memory with security<sup>(1)</sup> feature
  - 4,096 bytes for MC68H(R)C908JL3E/JK3E
  - 1,536 bytes for MC68H(R)C908JK1E
- 128 bytes of on-chip RAM
- 2-channel, 16-bit timer interface module (TIM)
- 12-channel, 8-bit analog-to-digital converter (ADC)
- 23 general purpose I/O ports for MC68H(R)C908JL3E:
  - 7 keyboard interrupt with internal pull-up  
(6 keyboard interrupt for MC68HC908JL3E)
  - 10 LED drivers (sink)
  - 2 × 25mA open-drain I/O with pull-up
- 15 general purpose I/O ports for MC68H(R)C908JK3E/JK1E:
  - 1 keyboard interrupt with internal pull-up  
(MC68HRC908JK3E/JK1E only)
  - 4 LED drivers (sink)
  - 2 × 25mA open-drain I/O with pull-up
  - 10-channel ADC
- System protection features:
  - Optional computer operating properly (COP) reset
  - Optional low-voltage detection with reset and selectable trip points for 3V and 5V operation
  - Illegal opcode detection with reset
  - Illegal address detection with reset
- Master reset pin with internal pull-up and power-on reset
- $\overline{\text{IRQ}}$  with schmitt-trigger input and programmable pull-up
- 28-pin PDIP, 28-pin SOIC, and 48-pin LQFP packages for MC68H(R)C908JL3E
- 20-pin PDIP and 20-pin SOIC packages for MC68H(R)C908JK3E/JK1E

---

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

### 4.3.5 Condition Code Register

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

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

X = Indeterminate

**Figure 4-6. Condition Code Register (CCR)**

#### V — Overflow Flag

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

- 1 = Overflow
- 0 = No overflow

#### H — Half-Carry Flag

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

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

#### I — Interrupt Mask

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

- 1 = Interrupts disabled
- 0 = Interrupts enabled

**NOTE**

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

After the I bit is cleared, the highest-priority interrupt request is serviced first.

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

#### N — Negative Flag

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

- 1 = Negative result
- 0 = Non-negative result

### Z — Zero Flag

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

1 = Zero result

0 = Non-zero result

### C — Carry/Borrow Flag

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

1 = Carry out of bit 7

0 = No carry out of bit 7

## 4.4 Arithmetic/Logic Unit (ALU)

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

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

## 4.5 Low-Power Modes

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

### 4.5.1 Wait Mode

The WAIT instruction:

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

### 4.5.2 Stop Mode

The STOP instruction:

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

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

## 4.6 CPU During Break Interrupts

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

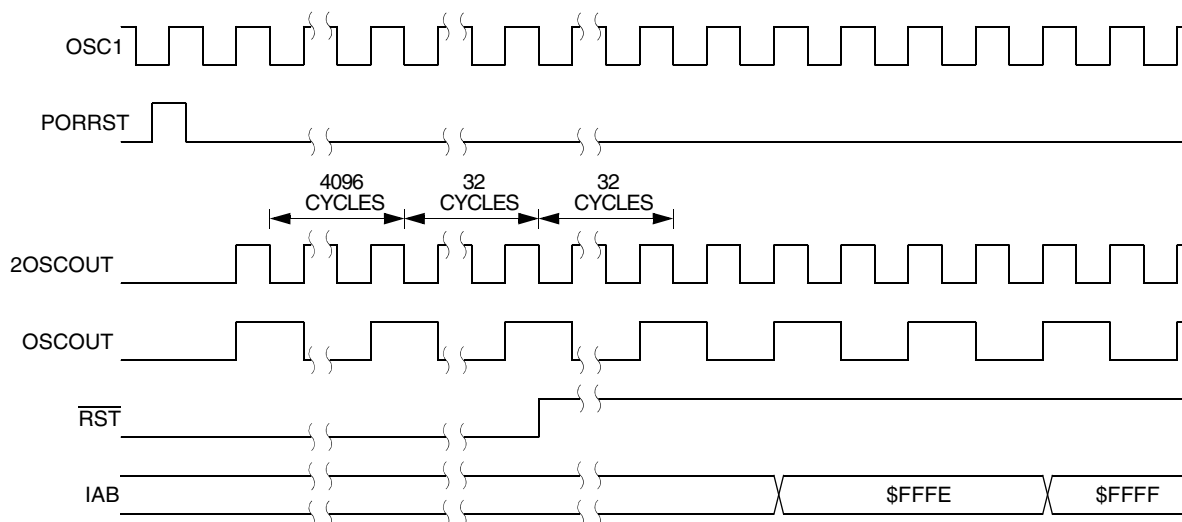
- Loading the instruction register with the SWI instruction
- Loading the program counter with \$FFFC:\$FFFD or with \$FEFC:\$FEFD in monitor mode

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

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

Table 4-1. 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 <i>opr</i> JMP <i>opr</i> JMP <i>opr,X</i> JMP <i>opr,X</i> JMP ,X	Jump	PC ← Jump Address	-	-	-	-	-	-	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff	2 3 4 3 2
JSR <i>opr</i> JSR <i>opr</i> JSR <i>opr,X</i> JSR <i>opr,X</i> JSR ,X	Jump to Subroutine	PC ← (PC) + <i>n</i> ( <i>n</i> = 1, 2, or 3) Push (PCL); SP ← (SP) - 1 Push (PCH); SP ← (SP) - 1 PC ← Unconditional Address	-	-	-	-	-	-	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff	4 5 6 5 4
LDA # <i>opr</i> LDA <i>opr</i> LDA <i>opr</i> LDA <i>opr,X</i> LDA <i>opr,X</i> LDA ,X LDA <i>opr,SP</i> LDA <i>opr,SP</i>	Load A from M	A ← (M)	0	-	-	†	†	-	IMM DIR EXT IX2 IX1 IX SP1 SP2	A6 B6 C6 D6 E6 F6 9EE6 9ED6	ii dd hh ll ee ff ff ee ff	2 3 4 4 3 2 4 5
LDHX # <i>opr</i> LDHX <i>opr</i>	Load H:X from M	H:X ← (M:M + 1)	0	-	-	†	†	-	IMM DIR	45 55	ii jj dd	3 4
LDX # <i>opr</i> LDX <i>opr</i> LDX <i>opr</i> LDX <i>opr,X</i> LDX <i>opr,X</i> LDX ,X LDX <i>opr,SP</i> LDX <i>opr,SP</i>	Load X from M	X ← (M)	0	-	-	†	†	-	IMM DIR EXT IX2 IX1 IX SP1 SP2	AE BE CE DE EE FE 9EEE 9EDE	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
LSL <i>opr</i> LSLA LSLX LSL <i>opr,X</i> LSL ,X LSL <i>opr,SP</i>	Logical Shift Left (Same as ASL)		†	-	-	†	†	†	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	4 1 1 4 3 5
LSR <i>opr</i> LSRA LSRX LSR <i>opr,X</i> LSR ,X LSR <i>opr,SP</i>	Logical Shift Right		†	-	-	0	†	†	DIR INH INH IX1 IX SP1	34 44 54 64 74 9E64	dd ff ff	4 1 1 4 3 5
MOV <i>opr,opr</i> MOV <i>opr,X+</i> MOV # <i>opr,opr</i> MOV X+, <i>opr</i>	Move	(M) <sub>Destination</sub> ← (M) <sub>Source</sub> H:X ← (H:X) + 1 (IX+D, DIX+)	0	-	-	†	†	-	DD DIX+ IMD IX+D	4E 5E 6E 7E	dd dd dd ii dd dd	5 4 4 4
MUL	Unsigned multiply	X:A ← (X) × (A)	-	0	-	-	-	0	INH	42		5
NEG <i>opr</i> NEGA NEGX NEG <i>opr,X</i> NEG ,X NEG <i>opr,SP</i>	Negate (Two's Complement)	M ← -(M) = \$00 - (M) A ← -(A) = \$00 - (A) X ← -(X) = \$00 - (X) M ← -(M) = \$00 - (M) M ← -(M) = \$00 - (M)	†	-	-	†	†	†	DIR INH INH IX1 IX SP1	30 40 50 60 70 9E60	dd ff ff	4 1 1 4 3 5
NOP	No Operation	None	-	-	-	-	-	-	INH	9D		1
NSA	Nibble Swap A	A ← (A[3:0]:A[7:4])	-	-	-	-	-	-	INH	62		3
ORA # <i>opr</i> ORA <i>opr</i> ORA <i>opr</i> ORA <i>opr,X</i> ORA <i>opr,X</i> ORA ,X ORA <i>opr,SP</i> ORA <i>opr,SP</i>	Inclusive OR A and M	A ← (A)   (M)	0	-	-	†	†	-	IMM DIR EXT IX2 IX1 IX SP1 SP2	AA BA CA DA EA FA 9EEA 9EDA	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
PSHA	Push A onto Stack	Push (A); SP ← (SP) - 1	-	-	-	-	-	-	INH	87		2
PSHH	Push H onto Stack	Push (H); SP ← (SP) - 1	-	-	-	-	-	-	INH	8B		2
PSHX	Push X onto Stack	Push (X); SP ← (SP) - 1	-	-	-	-	-	-	INH	89		2



**Figure 5-7. POR Recovery**

### 5.3.2.2 Computer Operating Properly (COP) Reset

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

To prevent a COP module time-out, write any value to location \$FFFF. Writing to location \$FFFF clears the COP counter and stages 12 through 5 of the SIM counter. The SIM counter output, which occurs at least every 4080 2OSCOUT cycles, drives the COP counter. The COP should be serviced as soon as possible out of reset to guarantee the maximum amount of time before the first time-out.

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

### 5.3.2.3 Illegal Opcode Reset

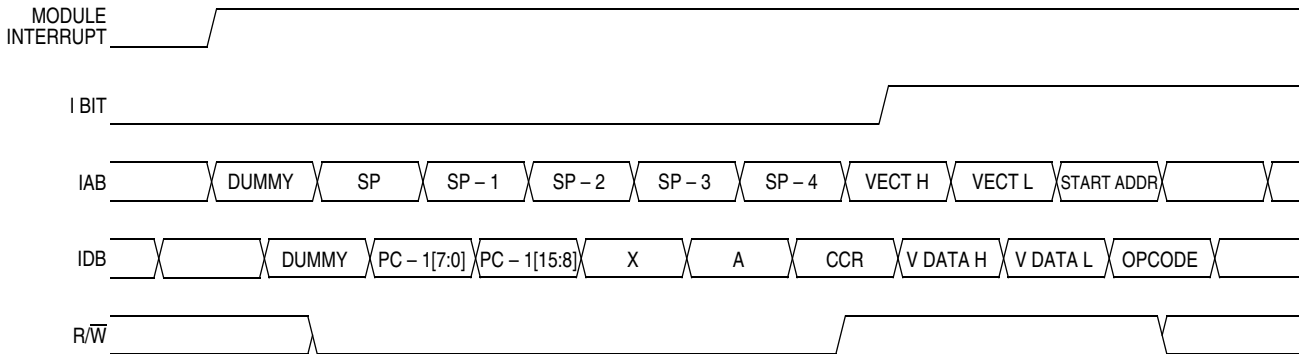
The SIM decodes signals from the CPU to detect illegal instructions. An illegal instruction sets the ILOP bit in the reset status register (RSR) and causes a reset.

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

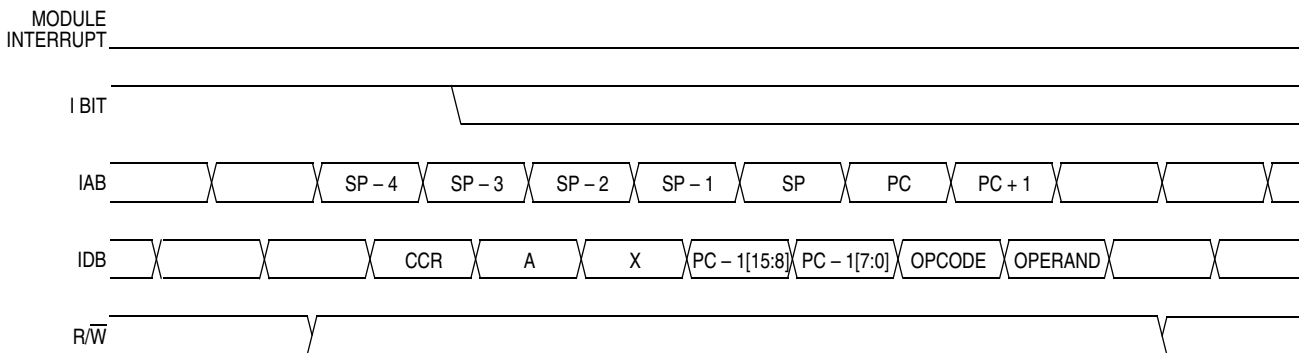
### 5.3.2.4 Illegal Address Reset

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

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



**Figure 5-9. Interrupt Entry**



**Figure 5-10. Interrupt Recovery**

**5.5.1.1 Hardware Interrupts**

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

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



## System Integration Module (SIM)

### IF14 — Interrupt Flags

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

1 = Interrupt request present

0 = No interrupt request present

### Bit 0 to 6 — Always read 0

#### 5.5.2.3 Interrupt Status Register 3

Address: \$FE06

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

R = Reserved

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

### IF15 — Interrupt Flags

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

1 = Interrupt request present

0 = No interrupt request present

### Bit 1 to 7 — Always read 0

#### 5.5.3 Reset

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

#### 5.5.4 Break Interrupts

The break module can stop normal program flow at a software-programmable break point by asserting its break interrupt output. (See Chapter 15 Break Module (BREAK).) The SIM puts the CPU into the break state by forcing it to the SWI vector location. Refer to the break interrupt subsection of each module to see how each module is affected by the break state.

#### 5.5.5 Status Flag Protection in Break Mode

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

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

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

## 6.5 Low Power Modes

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

### 6.5.1 Wait Mode

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

### 6.5.2 Stop Mode

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

## 6.6 Oscillator During Break Mode

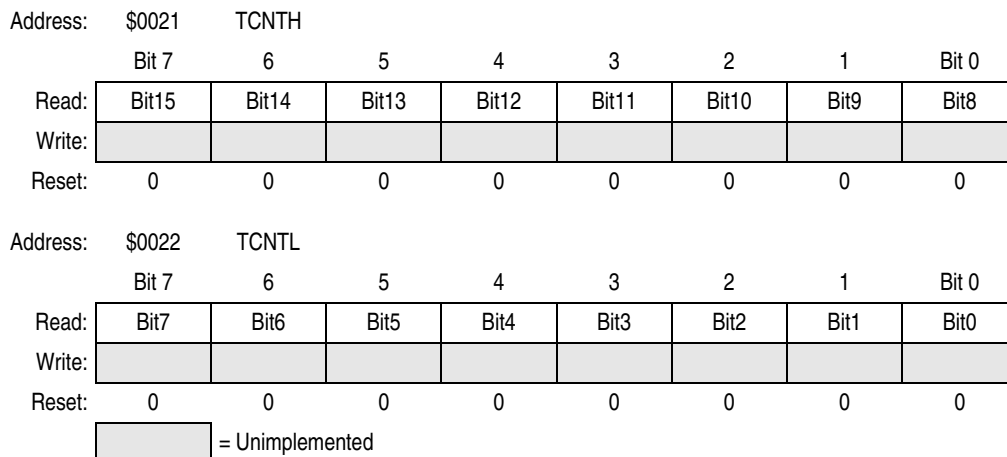
The oscillator continues to drive OSCOUT and 2OSCOUT when the device enters the break state.

### 8.9.2 TIM Counter Registers (TCNTH:TCNTL)

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**

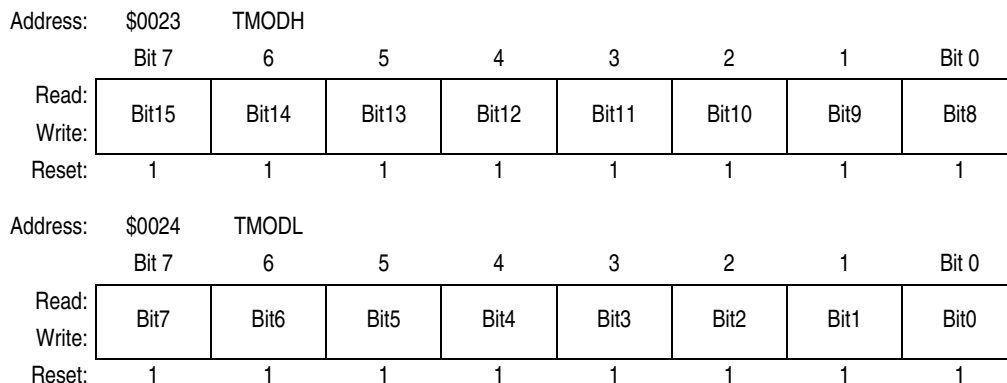
*If you read TCNTH during a break interrupt, be sure to unlatch TCNTL by reading TCNTL before exiting the break interrupt. Otherwise, TCNTL retains the value latched during the break.*



**Figure 8-5. TIM Counter Registers (TCNTH:TCNTL)**

### 8.9.3 TIM Counter Modulo Registers (TMODH:TMODL)

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.



**Figure 8-6. TIM Counter Modulo Registers (TMODH:TMODL)**

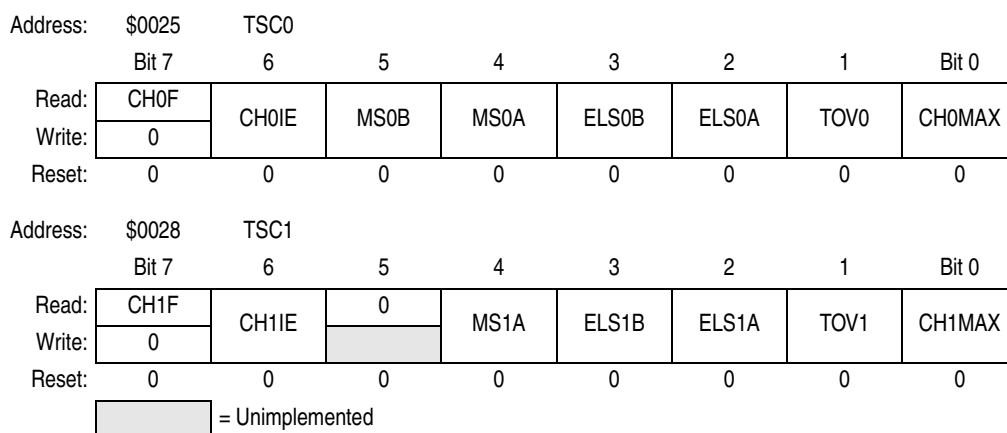
**NOTE**

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

### 8.9.4 TIM Channel Status and Control Registers (TSC0:TSC1)

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



**Figure 8-7. 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.

When TIM CPU interrupt requests are enabled (CHxIE=1), clear CHxF by reading the TIM channel x status and control register with CHxF set and then writing a zero to CHxF. If another interrupt request occurs before the clearing sequence is complete, then writing zero 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 one 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

### 9.3.2 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.*

### 9.3.3 Conversion Time

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

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

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

### 9.3.4 Continuous Conversion

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

### 9.3.5 Accuracy and Precision

The conversion process is monotonic and has no missing codes.

## 9.4 Interrupts

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

## 9.5 Low-Power Modes

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

### 9.5.1 Wait Mode

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

### ADCO — ADC Continuous Conversion Bit

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

1 = Continuous ADC conversion

0 = One ADC conversion

### ADCH[4:0] — ADC Channel Select Bits

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

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

#### NOTE

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

**Table 9-1. MUX Channel Select**

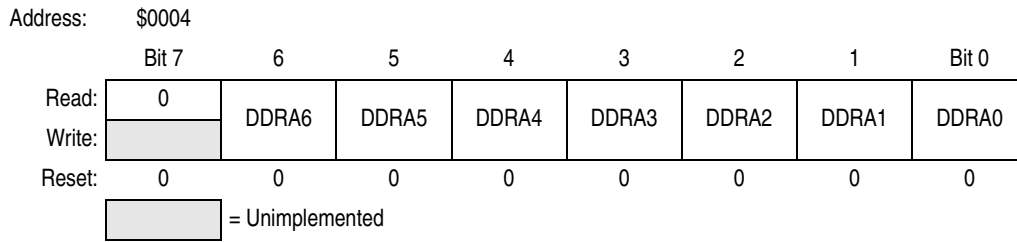
ADCH4	ADCH3	ADCH2	ADCH1	ADCH0	ADC Channel	Input Select
0	0	0	0	0	ADC0	PTB0
0	0	0	0	1	ADC1	PTB1
0	0	0	1	0	ADC2	PTB2
0	0	0	1	1	ADC3	PTB3
0	0	1	0	0	ADC4	PTB4
0	0	1	0	1	ADC5	PTB5
0	0	1	1	0	ADC6	PTB6
0	0	1	1	1	ADC7	PTB7
0	1	0	0	0	ADC8	PTD3
0	1	0	0	1	ADC9	PTD2
0	1	0	1	0	ADC10	PTD1
0	1	0	1	1	ADC11	PTD0
0	1	1	0	0	—	Unused (see Note 1)
:	:	:	:	:	—	
1	1	0	1	0	—	Reserved
1	1	0	1	1	—	Reserved
1	1	1	0	0	—	Unused
1	1	1	0	1		V <sub>DDA</sub> (see Note 2)
1	1	1	1	0		V <sub>SSA</sub> (see Note 2)
1	1	1	1	1		ADC power off

1. If any unused channels are selected, the resulting ADC conversion will be unknown.

2. The voltage levels supplied from internal reference nodes as specified in the table are used to verify the operation of the ADC converter both in production test and for user applications.

### 10.2.2 Data Direction Register A (DDRA)

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



**Figure 10-3. Data Direction Register A (DDRA)**

#### DDRA[6:0] — Data Direction Register A Bits

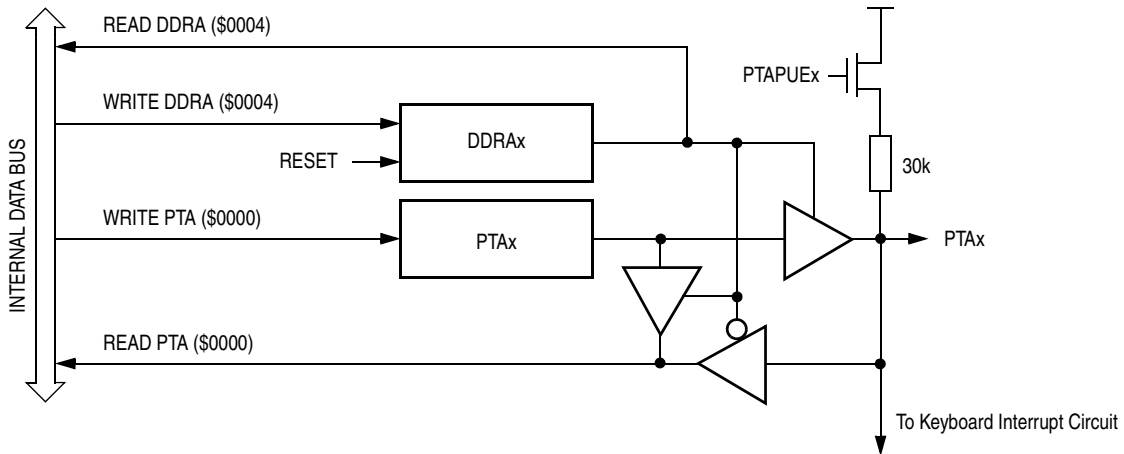
These read/write bits control port A data direction. Reset clears DDRA[6:0], configuring all port A pins as inputs.

- 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 10-4 shows the port A I/O logic.



**Figure 10-4. Port A I/O Circuit**

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.

# Chapter 11

## External Interrupt (IRQ)

### 11.1 Introduction

The IRQ (external interrupt) module provides a maskable interrupt input.

### 11.2 Features

Features of the IRQ module include the following:

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

### 11.3 Functional Description

A logic zero applied to the external interrupt pin can latch a CPU interrupt request. Figure 11-1 shows the structure of the IRQ module.

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

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

The external interrupt pin is falling-edge-triggered and is software-configurable to be either falling-edge or falling-edge and low-level-triggered. The MODE bit in the INTSCR controls the triggering sensitivity of the IRQ pin.

When the interrupt pin is edge-triggered only, the CPU interrupt request remains set until a vector fetch, software clear, or reset occurs.

When the interrupt pin is both falling-edge and low-level-triggered, the CPU interrupt request remains set until both of the following occur:

- Vector fetch or software clear
- Return of the interrupt pin to logic one



### 13.3.7 COPRS (COP Rate Select)

The COPRS signal reflects the state of the COP rate select bit (COPRS) in the configuration register 1.

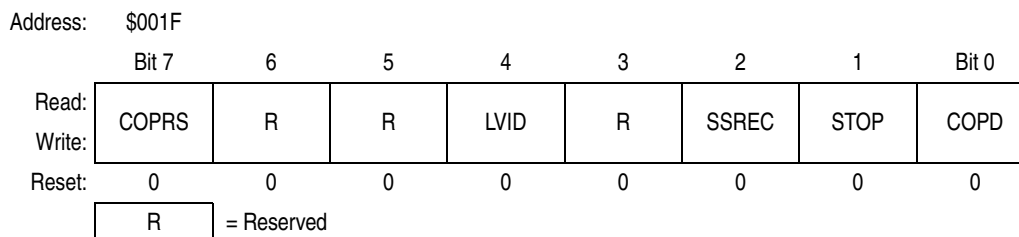


Figure 13-2. Configuration Register 1 (CONFIG1)

#### COPRS — COP Rate Select Bit

COPRS selects the COP timeout period. Reset clears COPRS.

- 1 = COP timeout period is  $8176 \times 2\text{OSCOU}$  cycles
- 0 = COP timeout period is  $262,128 \times 2\text{OSCOU}$  cycles

#### COPD — COP Disable Bit

COPD disables the COP module.

- 1 = COP module disabled
- 0 = COP module enabled

## 13.4 COP Control Register

The COP control register is located at address \$FFFF and overlaps the reset vector. Writing any value to \$FFFF clears the COP counter and starts a new timeout period. Reading location \$FFFF returns the low byte of the reset vector.

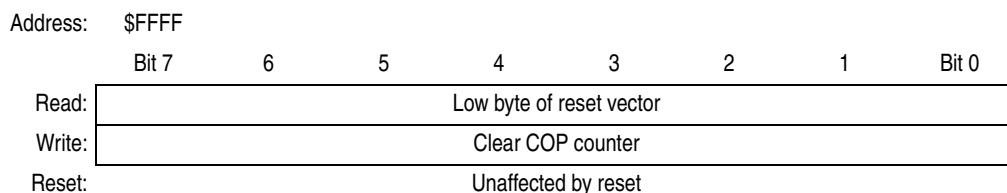


Figure 13-3. COP Control Register (COPCTL)

## 13.5 Interrupts

The COP does not generate CPU interrupt requests.

## 13.6 Monitor Mode

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

## 13.7 Low-Power Modes

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

## Computer Operating Properly (COP)

### 13.7.1 Wait Mode

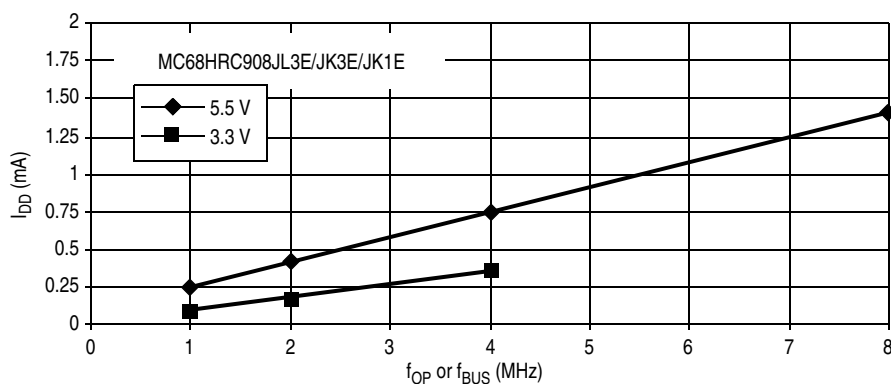
The COP continues to operate during wait mode. To prevent a COP reset during wait mode, periodically clear the COP counter in a CPU interrupt routine.

### 13.7.2 Stop Mode

Stop mode turns off the 2OSCO<sub>UT</sub> 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.

## 13.8 COP Module During Break Mode

The COP is disabled during a break interrupt when  $V_{TST}$  is present on the  $\overline{RST}$  pin.



**Figure 16-6. Typical Wait Mode I<sub>DD</sub> (MC68HRC908JL3E/JK3E/JK1E), with All Modules Turned Off (25 °C)**

## 16.12 ADC Characteristics

**Table 16-10. ADC Characteristics**

Characteristic	Symbol	Min	Max	Unit	Comments
Supply voltage	V <sub>DDAD</sub>	2.7 (V <sub>DD</sub> min)	5.5 (V <sub>DD</sub> max)	V	
Input voltages	V <sub>ADIN</sub>	V <sub>SS</sub>	V <sub>DD</sub>	V	
Resolution	B <sub>AD</sub>	8	8	Bits	
Absolute accuracy	A <sub>AD</sub>	± 0.5	± 1.5	LSB	Includes quantization
ADC internal clock	f <sub>ADIC</sub>	0.5	1.048	MHz	t <sub>AIC</sub> = 1/f <sub>ADIC</sub> , tested only at 1 MHz
Conversion range	R <sub>AD</sub>	V <sub>SS</sub>	V <sub>DD</sub>	V	
Power-up time	t <sub>ADPU</sub>	16		t <sub>AIC</sub> cycles	
Conversion time	t <sub>ADC</sub>	14	15	t <sub>AIC</sub> cycles	
Sample time <sup>(1)</sup>	t <sub>ADS</sub>	5	—	t <sub>AIC</sub> cycles	
Zero input reading <sup>(2)</sup>	Z <sub>ADI</sub>	00	01	Hex	V <sub>IN</sub> = V <sub>SS</sub>
Full-scale reading <sup>(3)</sup>	F <sub>ADI</sub>	FE	FF	Hex	V <sub>IN</sub> = V <sub>DD</sub>
Input capacitance	C <sub>ADI</sub>	—	(20) 8	pF	Not tested
Input leakage <sup>(3)</sup> Port B/port D	—	—	± 1	μA	

1. Source impedances greater than 10 kΩ adversely affect internal RC charging time during input sampling.
2. Zero-input/full-scale reading requires sufficient decoupling measures for accurate conversions.
3. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

### SSREC — Short Stop Recovery Bit

SSREC enables the CPU to exit stop mode with a delay of  $32 \times 2\text{OSCOUT}$  cycles instead of a  $4096 \times 2\text{OSCOUT}$  cycle delay.

- 1 = Stop mode recovery after  $32 \times 2\text{OSCOUT}$  cycles
- 0 = Stop mode recovery after  $4096 \times 2\text{OSCOUT}$  cycles

#### NOTE

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

*If using an external crystal, do not set the SSREC bit.*

### STOP — STOP Instruction Enable

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. (See **Chapter 13 Computer Operating Properly (COP)**.)

- 1 = COP module disabled
- 0 = COP module enabled

## B.5.3 Mask Option Register 2 (MOR2)

Address: \$001E

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

 = Unimplemented

**Figure 18-2. Mask Option Register 2 (MOR2)**

### IRQPUD — $\overline{\text{IRQ}}$ Pin Pull-up control bit

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

### LVIT1, LVIT0 — Low Voltage Inhibit trip voltage selection bits

Detail description of the LVI control signals is given in Chapter 14 Low Voltage Inhibit (LVI)

## B.6 Monitor ROM

The monitor program (monitor ROM: \$FE10–\$FFCF) on the MC68H(R)C08JL3E/JK3E is for device testing only. \$FC00–\$FDFF are unused.

## C.4 Reserved Registers

The following registers are reserved location on the MC68HC908KL3E/KK3E.

Addr.	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
\$003C	Reserved	Read:	R	R	R	R	R	R	R
		Write:							
		Reset:							
\$003D	Reserved	Read:	R	R	R	R	R	R	R
		Write:							
		Reset:							
\$003E	Reserved	Read:	R	R	R	R	R	R	R
		Write:							
		Reset:							

**Figure C-4. Reserved Registers**

## C.5 Reserved Vectors

The following vectors are reserved interrupt vectors on the MC68HC908KL3E/KK3E.

**Table C-2. Reserved Vectors**

Vector Priority	INT Flag	Address	Vector
—	IF15	\$FFDE	Reserved
		\$FFDF	Reserved

## C.6 Order Numbers

**Table C-3. MC68HC908KL3E/KK3E Order Numbers**

MC order number	Package	Operating Temperature	Operating V <sub>DD</sub>	OSC	Flash Memory
MC68HC908KL3ECP	28-pin PDIP	-40 to +85 °C	3V, 5V	XTAL	4096 Bytes
MC68HC908KL3ECDW	28-pin SOIC				
MC68HC908KK3ECP	20-pin PDIP				
MC68HC908KK3ECDW	20-pin SOIC				