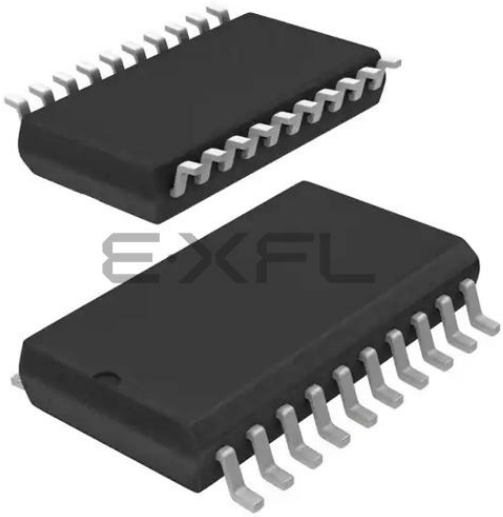

Welcome to E-XFL.COM

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	Obsolete
Core Processor	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	SCI
Peripherals	LED, LVD, POR, PWM
Number of I/O	15
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SOIC (0.295", 7.50mm Width)
Supplier Device Package	20-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68hc908jk8cdw

Table of Contents

Table 1-2. Pin Functions (Continued)

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

NOTE

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

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

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

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

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

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

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

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

Non-volatile FLASH registers; write by programming.

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

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

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

Table 2-1. Vector Addresses

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

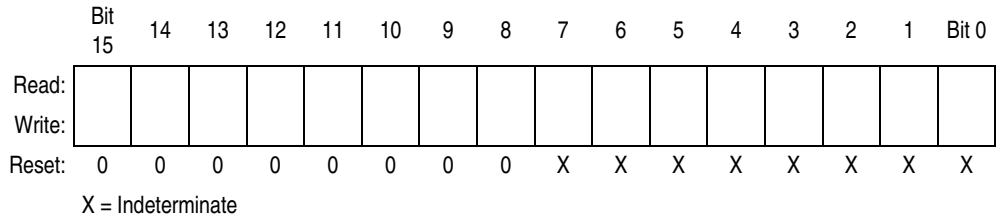

Figure 4-3. Index Register (H:X)

4.3.3 Stack Pointer

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

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

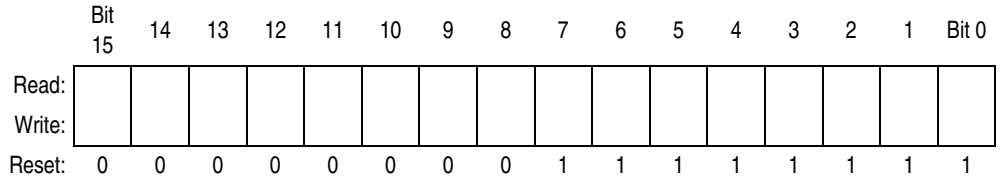

Figure 4-4. Stack Pointer (SP)

NOTE

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

4.3.4 Program Counter

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

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

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

Table 4-1. Instruction Set Summary

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

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

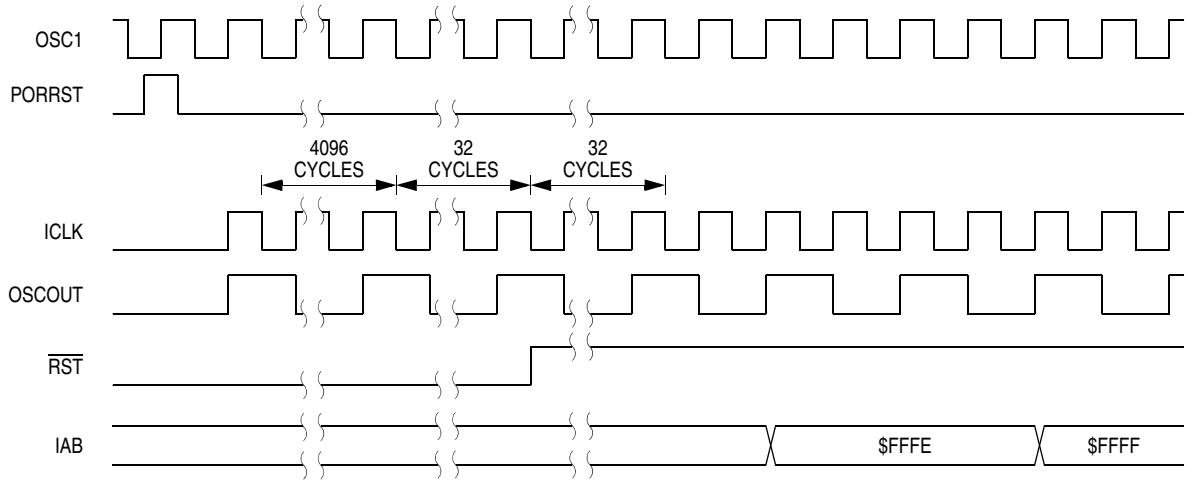

Figure 5-7. POR Recovery

5.3.2.2 Computer Operating Properly (COP) Reset

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

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

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

5.3.2.3 Illegal Opcode Reset

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

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

5.3.2.4 Illegal Address Reset

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

5.3.2.5 Low-Voltage Inhibit (LVI) Reset

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

Chapter 6
Oscillator (OSC)

6.1 Introduction

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

Selectable oscillator — for bus clock

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

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

Non-selectable oscillator — for COP

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

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

6.2 Oscillator Selection

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

NOTE

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

Address: $FFD0

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

Non-volatile FLASH register; write by programming.

R = Reserved

Figure 6-1. Mask Option Register (MOR)

7.5.6 MON_LDRNGE

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

Table 7-16. ICP_LDRNGE Routine

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

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

7.5.7 EE_WRITE

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

Table 7-17. EE_WRITE Routine

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

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

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

7.5.8 EE_READ

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

Table 7-18. EE_READ Routine

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

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

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

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

```
EE_READ         EQU     $FDD0

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

NOTE

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

8.4.3.1 Unbuffered Output Compare

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

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

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

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

8.4.3.2 Buffered Output Compare

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

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

NOTE

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

8.4.4 Pulse Width Modulation (PWM)

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

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

8.9.4 TIM Channel Status and Control Registers

Each of the TIM channel status and control registers:

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

Address: T1SC0, $0025 and T2SC0, $0035

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

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

Address: T1SC1, $0028 and T2SC1, $0038

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

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

CHxF — Channel x Flag Bit

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

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

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

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

CHxIE — Channel x Interrupt Enable Bit

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

Reset clears the CHxIE bit.

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

MSxB — Mode Select Bit B

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

9.4.3.3 Data Sampling

The receiver samples the RxD pin at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock is resynchronized at the following times (see Figure 9-6):

- After every start bit
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0)

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

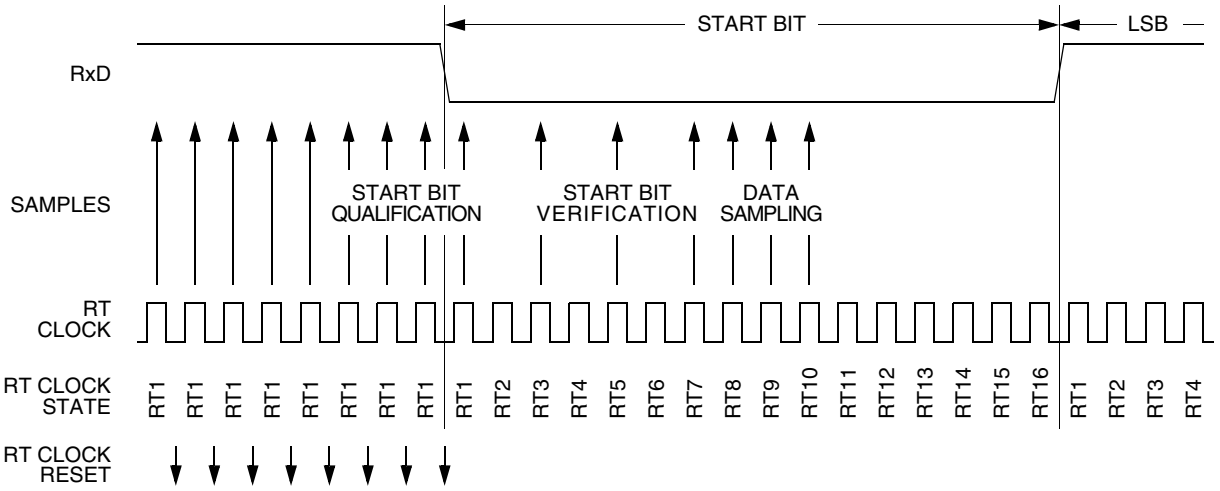

Figure 9-6. Receiver Data Sampling

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

Table 9-2. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

Start bit verification is not successful if any two of the three verification samples are logic 1s. If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

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

Figure 11-1. I/O Port Register Summary

Table 11-1. Port Control Register Bits Summary

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

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

13.4 Functional Description

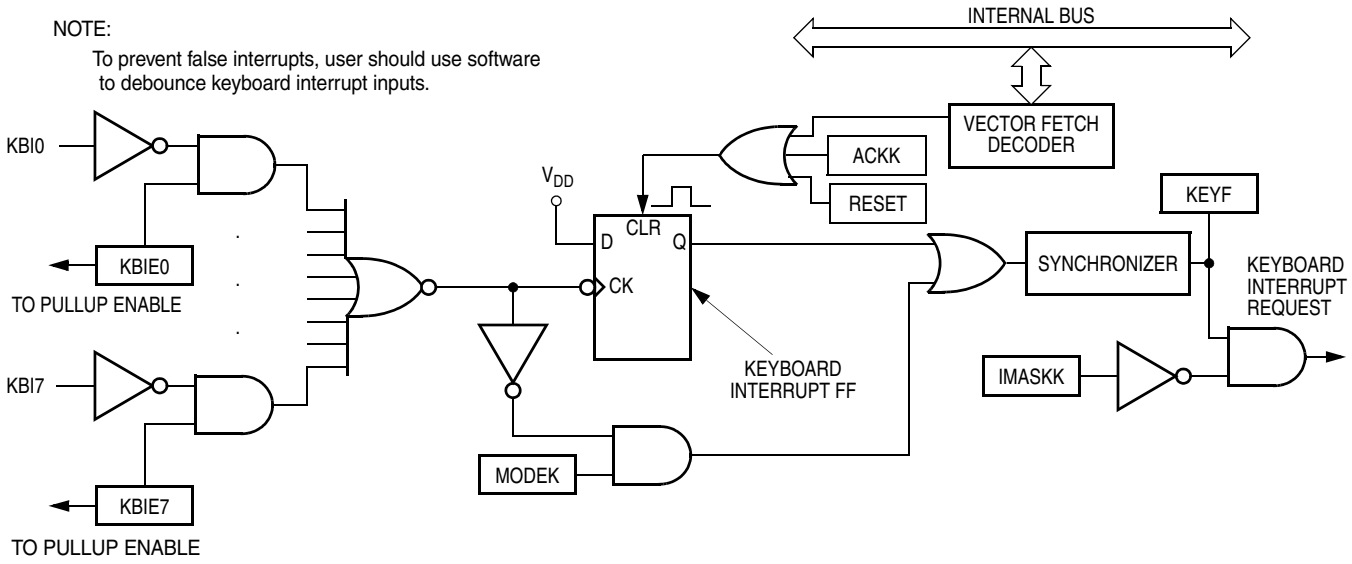

Figure 13-2. Keyboard Interrupt Block Diagram

Writing to the KBIE7–KBIE0 bits in the keyboard interrupt enable register independently enables or disables each port A pin as a keyboard interrupt pin. Enabling a keyboard interrupt pin in port A also enables its internal pull-up device regardless of PTAPUEx bits in the port A input pull-up enable register (see 11.2.3 Port A Input Pull-Up Enable Registers). A logic 0 applied to an enabled keyboard interrupt pin latches a keyboard interrupt request.

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

- If the keyboard interrupt is edge-sensitive only, a falling edge on a keyboard pin does not latch an interrupt request if another keyboard pin is already low. To prevent losing an interrupt request on one pin because another pin is still low, software can disable the latter pin while it is low.
- If the keyboard interrupt is falling edge- and low level-sensitive, an interrupt request is present as long as any keyboard pin is low.

If the MODEK bit is set, the keyboard interrupt pins are both falling edge- and low level-sensitive, and both of the following actions must occur to clear a keyboard interrupt request:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the interrupt request. Software may generate the interrupt acknowledge signal by writing a logic 1 to the ACKK bit in the keyboard status and control register KBSCR. The ACKK bit is useful in applications that poll the keyboard interrupt pins and require software to clear the keyboard interrupt request. Writing to the ACKK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACKK does not affect subsequent transitions on the keyboard interrupt pins. A falling edge that occurs after writing to the ACKK bit latches another interrupt request. If the keyboard interrupt mask bit, IMASKK, is clear, the CPU loads the program counter with the vector address at locations $FFE0 and $FFE1.
- Return of all enabled keyboard interrupt pins to logic 1 — As long as any enabled keyboard interrupt pin is at logic 0, the keyboard interrupt remains set.

The vector fetch or software clear and the return of all enabled keyboard interrupt pins to logic 1 may occur in any order.

The COP counter is a free-running 6-bit counter preceded by the 12-bit system integration module (SIM) counter. If not cleared by software, the COP counter overflows and generates an asynchronous reset after $2^{18} - 2^4$ or $2^{13} - 2^4$ ICLK cycles; depending on the state of the COP rate select bit, COPRS, in configuration register 1. Writing any value to location $FFFF before an overflow occurs prevents a COP reset by clearing the COP counter and stages 12 through 5 of the SIM counter.

NOTE

Service the COP immediately after reset and before entering or after exiting stop mode to guarantee the maximum time before the first COP counter overflow.

A COP reset pulls the $\overline{\text{RST}}$ pin low for 32 × ICLK cycles and sets the COP bit in the reset status register (RSR). (See 5.7.2 Reset Status Register (RSR).).

NOTE

Place COP clearing instructions in the main program and not in an interrupt subroutine. Such an interrupt subroutine could keep the COP from generating a reset even while the main program is not working properly.

14.3 I/O Signals

The following paragraphs describe the signals shown in Figure 14-1.

14.3.1 ICLK

ICLK is the internal oscillator output signal, typically 50-kHz. The ICLK frequency varies depending on the supply voltage. See **Chapter 17 Electrical Specifications** for ICLK parameters.

14.3.2 COPCTL Write

Writing any value to the COP control register (COPCTL) (see **14.4 COP Control Register**) clears the COP counter and clears bits 12 through 5 of the SIM counter. Reading the COP control register returns the low byte of the reset vector.

14.3.3 Power-On Reset

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

14.3.4 Internal Reset

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

14.3.5 Reset Vector Fetch

A reset vector fetch occurs when the vector address appears on the data bus. A reset vector fetch clears the SIM counter.

14.3.6 COPD (COP Disable)

The COPD signal reflects the state of the COP disable bit (COPD) in the configuration register 1 (CONFIG1). (See Chapter 3 Configuration and Mask Option Registers (CONFIG & MOR).)

17.12 Timer Interface Module Characteristics

Table 17-10. Timer Interface Module Characteristics (5V and 3V)

Characteristic	Symbol	Min	Max	Unit
Input capture pulse width	t_{TIH}, t_{TIL}	$1/f_{OP}$	—	
Input clock pulse width (T2CLK pulse width)	t_{LMIN}, t_{HMIN}	$(1/f_{OP}) + 5$ns	—	

17.13 ADC Characteristics

Table 17-11. ADC Characteristics (5V and 3V)

Characteristic	Symbol	Min	Max	Unit	Comments
Supply voltage	V_{DDAD}	2.7 (V_{DD} min)	5.5 (V_{DD} max)	V	
Input voltages	V_{ADIN}	V_{SS}	V_{DD}	V	
Resolution	B_{AD}	8	8	Bits	
Absolute accuracy	A_{AD}	± 0.5	± 1.5	LSB	Includes quantization
ADC internal clock	f_{ADIC}	0.5	1.048	MHz	$t_{AIC} = 1/f_{ADIC}$, tested only at 1 MHz
Conversion range	R_{AD}	V_{SS}	V_{DD}	V	
Power-up time	t_{ADPU}	16		t_{AIC} cycles	
Conversion time	t_{ADC}	14	15	t_{AIC} cycles	
Sample time[1]	t_{ADS}	5	—	t_{AIC} cycles	
Zero input reading[2]	Z_{ADI}	00	01	Hex	$V_{IN} = V_{SS}$
Full-scale reading[3]	F_{ADI}	FE	FF	Hex	$V_{IN} = V_{DD}$
Input capacitance	C_{ADI}	—	(20) 8	pF	Not tested
Input leakage[3] Port B/port D	—	—	± 1	μA	

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