



Welcome to [E-XFL.COM](https://www.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, SPI
Peripherals	LVD, POR, PWM
Number of I/O	24
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	384 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 10x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	28-TSSOP
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc908qc8cdre

MC68HC908QC16

MC68HC908QC8

MC68HC908QC4

Data Sheet

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

<http://www.freescale.com>

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

© Freescale Semiconductor, Inc., 2007, 2008. All rights reserved.

Table 1-2. Pin Functions

Pin Name	Description	Input/Output
V _{DD}	Power supply	Power
V _{SS}	Power supply ground	Power
PTA0	PTA0 — General purpose I/O port	Input/Output
	T1CH0 — Timer Channel 0 I/O	Input/Output
	AD0 — A/D channel 0 input	Input
	KBI0 — Keyboard interrupt input 0	Input
PTA1	PTA1 — General purpose I/O port	Input/Output
	T1CH1 — Timer Channel 1 I/O	Input/Output
	AD1 — A/D channel 1 input	Input
	KBI1 — Keyboard interrupt input 1	Input
PTA2 ⁽¹⁾	PTA2 — General purpose input-only port	Input
	IRQ — External interrupt with programmable pullup and Schmitt trigger input	Input
	KBI2 — Keyboard interrupt input 2	Input
	T1CLK — TIM1 timer clock input	Input
PTA3	PTA3 — General purpose I/O port	Input/Output
	RST — Reset input, active low with internal pullup and Schmitt trigger	Input
	KBI3 — Keyboard interrupt input 3	Input
PTA4	PTA4 — General purpose I/O port	Input/Output
	OSC2 — XTAL oscillator output (XTAL option only) RC or internal oscillator output (OSC2EN = 1 in PTAPUE register)	Output Output
	AD2 — A/D channel 2 input	Input
	KBI4 — Keyboard interrupt input 4	Input
PTA5	PTA5 — General purpose I/O port	Input/Output
	OSC1 — XTAL, RC, or external oscillator input	Input
	AD3 — A/D channel 3 input	Input
	KBI5 — Keyboard interrupt input 5	Input
PTB0	PTB0 — General-purpose I/O port	Input/Output
	SPSCK — SPI serial clock	Input/Output
	AD4 — A/D channel 4 input	Input
PTB1	PTB1 — General-purpose I/O port	Input/Output
	MOSI — SPI data transmitted	Input/Output
	T2CH1 ⁽²⁾ — TIM2 channel 1	Input/Output
	AD5 — A/D channel 5 input	Input

— Continued on next page

2.6 FLASH Memory (FLASH)

The FLASH memory is intended primarily for program storage. In-circuit programming allows the operating program to be loaded into the FLASH memory after final assembly of the application product. It is possible to program the entire array through the single-wire monitor mode interface. Because no special voltages are needed for FLASH erase and programming operations, in-application programming is also possible through other software-controlled communication paths.

This subsection describes the operation of the embedded FLASH memory. The FLASH memory can be read, programmed, and erased from the internal V_{DD} supply. The program and erase operations are enabled through the use of an internal charge pump.

The minimum size of FLASH memory that can be erased is 64 bytes; and the maximum size of FLASH memory that can be programmed in a program cycle is 32 bytes (a row). Program and erase operations are facilitated through control bits in the FLASH control register (FLCR). Details for these operations appear later in this section.

NOTE

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

2.6.1 FLASH Control Register

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

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


 = Unimplemented

Figure 2-3. FLASH Control Register (FLCR)

HVEN — High Voltage Enable Bit

This read/write bit enables high voltage from the charge pump to the memory for either program or erase operation. It can only be set if either PGM = 1 or ERASE = 1 and the proper sequence for program or erase is followed.

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

MASS — Mass Erase Control Bit

This read/write bit configures the memory for mass erase operation.

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

ERASE — Erase Control Bit

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

- 1 = Erase operation selected
- 0 = Erase operation unselected

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

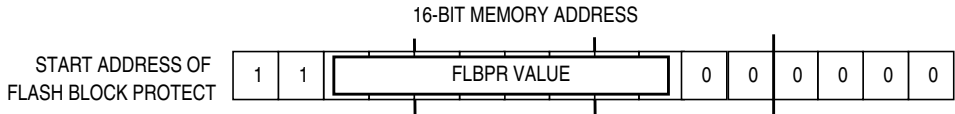


Figure 2-6. FLASH Block Protect Start Address

Table 2-2. Examples of Protect Start Address

BPR[7:0]	Start of Address of Protect Range ⁽¹⁾
\$00 ⁽²⁾	The entire FLASH memory is protected.
\$01 (0000 0001)	\$C040 (1100 0000 0100 0000)
\$02 (0000 0010)	\$C080 (1100 0000 1000 0000)
\$03 (0000 0011)	\$C0C0 (1100 0000 1100 0000)
and so on...	
\$FD (1111 1101)	\$FF40 (1111 1111 0100 0000)
\$FE (1111 1110)	\$FF80 (1111 1111 1000 0000)
\$FF	The entire FLASH memory is not protected.

1. The end address of the protected range is always \$FFFF.
2. \$BE00–\$BFFF is always protected unless the entire FLASH memory is unprotected, BPR[7:0] = \$FF.

2.6.7 EEPROM Memory Emulation Using FLASH Memory

In some applications, the user may want to repeatedly store and read a set of data from an area of nonvolatile memory. This is easily implemented in EEPROM memory because single byte erase is allowed in EEPROM.

When using FLASH memory, the minimum erase size is a page. However, the FLASH can be used as EEPROM memory. This technique is called “EEPROM emulation”.

The basic concept of EEPROM emulation using FLASH is that a page is continuously programmed with a new data set without erasing the previously programmed locations. Once the whole page is completely programmed or the page does not have enough bytes to program a new data set, the user software automatically erases the page and then programs a new data set in the erased page.

In EEPROM emulation when data is read from the page, the user software must find the latest data set in the page since the previous data still remains in the same page. There are many ways to monitor the page erase timing and the latest data set. One example is unprogrammed FLASH bytes are detected by checking programmed bytes (non-\$FF value) in a page. In this way, the end of the data set will contain unprogrammed data (\$FF value).

A couple of application notes, describing how to emulate EEPROM using FLASH, are available on our web site. Titles and order numbers for these application notes are given at the end of this subsection.

3.3.4.4 Code Width and Quantization Error

The ADC10 quantizes the ideal straight-line transfer function into 1024 steps (in 10-bit mode). Each step ideally has the same height (1 code) and width. The width is defined as the delta between the transition points from one code to the next. The ideal code width for an N bit converter (in this case N can be 8 or 10), defined as 1LSB, is:

$$1\text{LSB} = (V_{\text{REFH}} - V_{\text{REFL}}) / 2^N$$

Because of this quantization, there is an inherent quantization error. Because the converter performs a conversion and then rounds to 8 or 10 bits, the code will transition when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be 1/2LSB in 8- or 10-bit mode. As a consequence, however, the code width of the first (\$000) conversion is only 1/2LSB and the code width of the last (\$FF or \$3FF) is 1.5LSB.

3.3.4.5 Linearity Errors

The ADC10 may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors but the user should be aware of them because they affect overall accuracy. These errors are:

- Zero-Scale Error (E_{ZS}) (sometimes called offset) — This error is defined as the difference between the actual code width of the first conversion and the ideal code width (1/2LSB). Note, if the first conversion is \$001, then the difference between the actual \$001 code width and its ideal (1LSB) is used.
- Full-Scale Error (E_{FS}) — This error is defined as the difference between the actual code width of the last conversion and the ideal code width (1.5LSB). Note, if the last conversion is \$3FE, then the difference between the actual \$3FE code width and its ideal (1LSB) is used.
- Differential Non-Linearity (DNL) — This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral Non-Linearity (INL) — This error is defined as the highest-value the (absolute value of the) running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total Unadjusted Error (TUE) — This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function, and therefore includes all forms of error.

3.3.4.6 Code Jitter, Non-Monotonicity and Missing Codes

Analog-to-digital converters are susceptible to three special forms of error. These are code jitter, non-monotonicity, and missing codes.

- Code jitter is when, at certain points, a given input voltage converts to one of two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the converter yields the lower code (and vice-versa). However, even very small amounts of system noise can cause the converter to be indeterminate (between two codes) for a range of input voltages around the transition voltage. This range is normally around 1/2LSB but will increase with noise.
- Non-monotonicity is defined as when, except for code jitter, the converter converts to a lower code for a higher input voltage.
- Missing codes are those which are never converted for any input value. In 8-bit or 10-bit mode, the ADC10 is guaranteed to be monotonic and to have no missing codes.

Configuration Registers (CONFIG1 and CONFIG2)

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

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

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

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

TIM2POS — TIM2 Position Bit

TIM2POS is used to reposition the timer channels for TIM2 to a different pair of pins. This allows the user to free up one of the communication ports based on application needs.

- 1 = TIM2 timer channel pins share PTB4 and PTB5
- 0 = TIM2 timer channel pins share PTB1 and PTB2

ESCIBDSRC — ESCI Baud Rate Clock Source Bit

ESCIBDSRC controls the clock source used for the ESCI. The setting of the bit affects the frequency at which the ESCI operates.

- 1 = Internal data bus clock used as clock source for ESCI
- 0 = BUSCLKX4 used as clock source for ESCI

OSCENINSTOP— Oscillator Enable in Stop Mode Bit

OSCENINSTOP, when set, will allow the clock source to continue to generate clocks in stop mode. This function can be used to keep the periodic wakeup running while the rest of the microcontroller stops. When clear, the clock source is disabled when the microcontroller enters stop mode.

- 1 = Oscillator enabled to operate during stop mode
- 0 = Oscillator disabled during stop mode

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

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

NOTE

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

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	LVITRIP	SSREC	STOP	COPD
Write:								
Reset:	0	0	0	0	U	0	0	0
POR:	0	0	0	0	0	0	0	0

U = Unaffected

Figure 4-2. Configuration Register 1 (CONFIG1)

COPRS — COP Reset Period Selection Bit

- 1 = COP reset short cycle = $8176 \times \text{BUSCLKX4}$
- 0 = COP reset long cycle = $262,128 \times \text{BUSCLKX4}$

6.3.3 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.

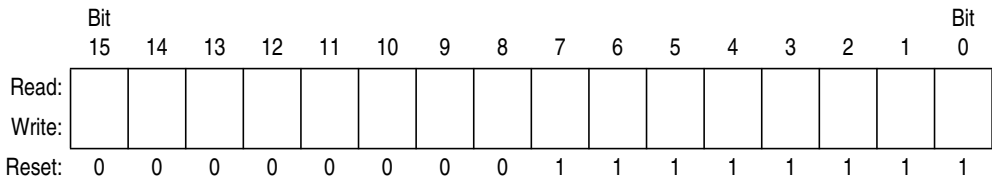


Figure 6-4. Stack Pointer (SP)

NOTE

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

6.3.4 Program Counter

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

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

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

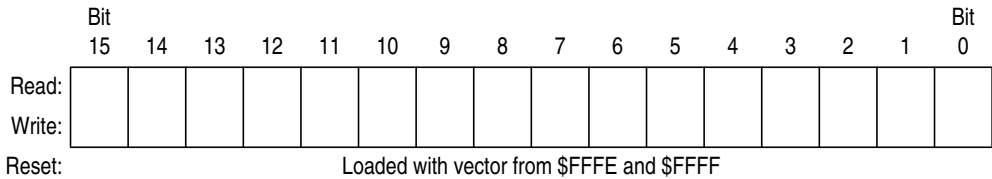
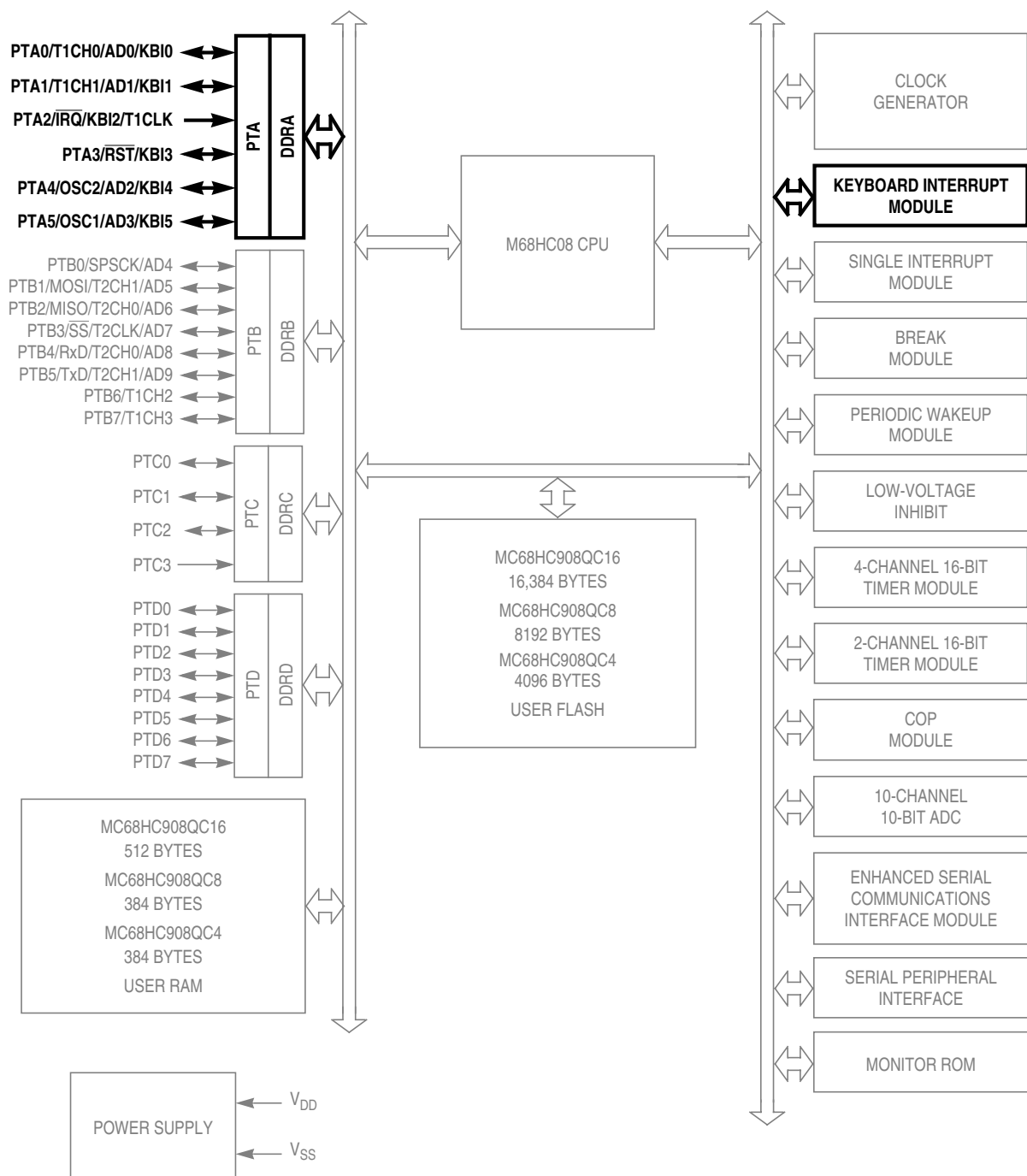


Figure 6-5. Program Counter (PC)



Keyboard Interrupt Module (KBI)



All port pins can be configured with internal pullup
 PTC not available on 16-pin devices (see note in 11.1 Introduction)
 PTD not available on 16-pin or 20-pin devices (see note in 11.1 Introduction)

Figure 8-2. Block Diagram Highlighting KBI Block and Pins

9.4 LVI Interrupts

The LVI module does not generate interrupt requests.

9.5 Low-Power Modes

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

9.5.1 Wait Mode

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

9.5.2 Stop Mode

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

9.6 Registers

The LVI status register (LVISR) contains a status bit that is useful when the LVI is enabled and LVI reset is disabled.

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

 = Unimplemented
 R = Reserved

Figure 9-2. LVI Status Register (LVISR)

LVIOUT — LVI Output Bit

This read-only flag becomes set when the V_{DD} voltage falls below the V_{TRIPF} trip voltage and is cleared when V_{DD} voltage rises above V_{TRIPR} . (See Table 9-1.)

Table 9-1. LVIOUT Bit Indication

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



Oscillator Mode (OSC)

Periodic Wakeup Module (PWU)

PWUON — Periodic Wakeup Enabled Bit

This read/write bit enables or disables the periodic wakeup module.
 1 = Periodic wakeup module is enabled.
 0 = Periodic wakeup module is disabled.

PWUCLKSEL — Periodic Wakeup Clock Select Bit

This read/write bit selects the clock source for the prescaler.
 1 = BUSCLKX4 is selected as the clock source for the prescaler.
 0 = The internal 32 kHz RC oscillator is selected as the clock source for the prescaler.

NOTE

The PWUCLKSEL bit can only be written to when PWUON is clear.

PWUF — Periodic Wakeup Flag Bit

This read-only bit is set when the counter reaches the modulo value programmed in the PWUMOD register. This bit is cleared by writing a 1 to the PWUACK bit or by a PWU interrupt vector fetch.
 1 = Periodic wakeup interrupt pending
 0 = No periodic wakeup interrupt pending.

PWUACK — Periodic Wakeup Acknowledge Bit

Writing a 1 to this write-only bit clears the PWUF. PWUACK always reads as 0.

PWUIE — Periodic Wakeup Interrupt Enable Bit

This read/write bit enables periodic wakeup interrupt requests.
 1 = Periodic wakeup interrupt requests enabled.
 0 = Periodic wakeup interrupt requests disabled.

SMODE — Periodic Wakeup Module Enabled in Stop Mode Bit

This read/write bit is used to allow the PWU module to continue running in stop mode.
 1 = Periodic wakeup module continues to run in stop mode.
 0 = Periodic wakeup module disabled in stop mode.

12.8.2 Periodic Wakeup Prescaler Register

The PWUP register is used to select the clock rate that will be input to the PWU counter. The prescaler generates sixteen clock rates from either the dedicated low-power internal oscillator or from the bus clock source (BUSCLKX4).

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	PS3	PS2	PS1	PS0
Write:								
Reset:	0	0	0	0	0	1	0	0


 = Unimplemented

Figure 12-3. Periodic Wakeup Prescaler Register (PWUP)

PS3–PS0 — Prescaler Select Bits

These read/write bits select one of the sixteen prescaler outputs to be the input to the PWU counter.

NOTE

The PWUP register can only be written to when PWUON is clear.

Fast Data Tolerance

Figure 13-8 shows how much a fast received character can be misaligned without causing a noise error or a framing error. The fast stop bit ends at RT10 instead of RT16 but is still there for the stop bit data samples at RT8, RT9, and RT10.

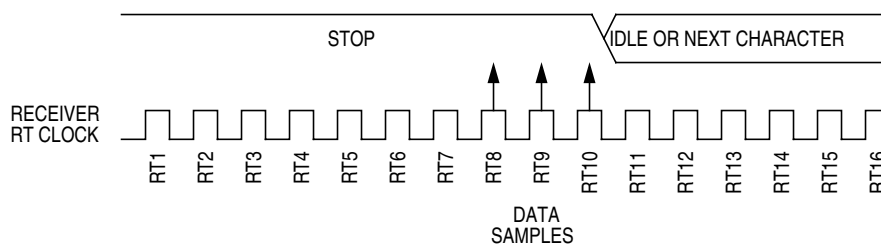


Figure 13-8. Fast Data

For an 8-bit character, data sampling of the stop bit takes the receiver
 $9 \text{ bit times} \times 16 \text{ RT cycles} + 10 \text{ RT cycles} = 154 \text{ RT cycles}$.

With the misaligned character shown in Figure 13-8, the receiver counts 154 RT cycles at the point when the count of the transmitting device is $10 \text{ bit times} \times 16 \text{ RT cycles} = 160 \text{ RT cycles}$.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is

$$\left| \frac{154 - 160}{154} \right| \times 100 = 3.90\%.$$

For a 9-bit character, data sampling of the stop bit takes the receiver
 $10 \text{ bit times} \times 16 \text{ RT cycles} + 10 \text{ RT cycles} = 170 \text{ RT cycles}$.

With the misaligned character shown in Figure 13-8, the receiver counts 170 RT cycles at the point when the count of the transmitting device is $11 \text{ bit times} \times 16 \text{ RT cycles} = 176 \text{ RT cycles}$.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

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

13.3.3.6 Receiver Wakeup

So that the MCU can ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in SCC2 puts the receiver into a standby state during which receiver interrupts are disabled.

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

1. **Address mark** — An address mark is a 1 in the MSB position of a received character. When the WAKE bit is set, an address mark wakes the receiver from the standby state by clearing the RWU bit. The address mark also sets the ESCI receiver full bit, SCRF. Software can then compare the character containing the address mark to the user-defined address of the receiver. If they are the same, the receiver remains awake and processes the characters that follow. If they are not the same, software can set the RWU bit and put the receiver back into the standby state.

ILIE — Idle Line Interrupt Enable Bit

This read/write bit enables the IDLE bit to generate ESCI receiver interrupt requests.

1 = IDLE enabled to generate interrupt requests

0 = IDLE not enabled to generate interrupt requests

TE — Transmitter Enable Bit

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

1 = Transmitter enabled

0 = Transmitter disabled

NOTE

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

RE — Receiver Enable Bit

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

1 = Receiver enabled

0 = Receiver disabled

NOTE

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

RWU — Receiver Wakeup Bit

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

1 = Standby state

0 = Normal operation

SBK — Send Break Bit

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

1 = Transmit break characters

0 = No break characters being transmitted

NOTE

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

13.8.3 ESCI Control Register 3

ESCI control register 3 (SCC3):

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

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R8	T8	R	R	ORIE	NEIE	FEIE	PEIE
Write:								
Reset:	U	0	0	0	0	0	0	0

 = Unimplemented R = Reserved U = Unaffected

Figure 13-11. ESCI Control Register 3 (SCC3)

R8 — Received Bit 8

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

When the ESCI is receiving 8-bit characters, R8 is a copy of the eighth bit (bit 7).

T8 — Transmitted Bit 8

When the ESCI is transmitting 9-bit characters, T8 is the read/write ninth bit (bit 8) of the transmitted character. T8 is loaded into the transmit shift register at the same time that the SCDR is loaded into the transmit shift register.

ORIE — Receiver Overrun Interrupt Enable Bit

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

- 1 = ESCI error interrupt requests from OR bit enabled
- 0 = ESCI error interrupt requests from OR bit disabled

NEIE — Receiver Noise Error Interrupt Enable Bit

This read/write bit enables ESCI error interrupt requests generated by the noise error bit, NE.

- 1 = ESCI error interrupt requests from NE bit enabled
- 0 = ESCI error interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables ESCI error interrupt requests generated by the framing error bit, FE.

- 1 = ESCI error interrupt requests from FE bit enabled
- 0 = ESCI error interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables ESCI receiver interrupt requests generated by the parity error bit, PE.

- 1 = ESCI error interrupt requests from PE bit enabled
- 0 = ESCI error interrupt requests from PE bit disabled

14.8.1 SIM Reset Status Register

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

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
Write:								
POR:	1	0	0	0	0	0	0	0
	<div style="display: inline-block; width: 20px; height: 10px; background-color: #cccccc; border: 1px solid black;"></div> = Unimplemented							

Figure 14-19. SIM Reset Status Register (SRSR)

POR — Power-On Reset Bit

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

PIN — External Reset Bit

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

COP — Computer Operating Properly Reset Bit

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

ILOP — Illegal Opcode Reset Bit

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

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

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

MODRST — Monitor Mode Entry Module Reset bit

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

LVI — Low Voltage Inhibit Reset bit

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

14.8.2 Break Flag Control Register

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

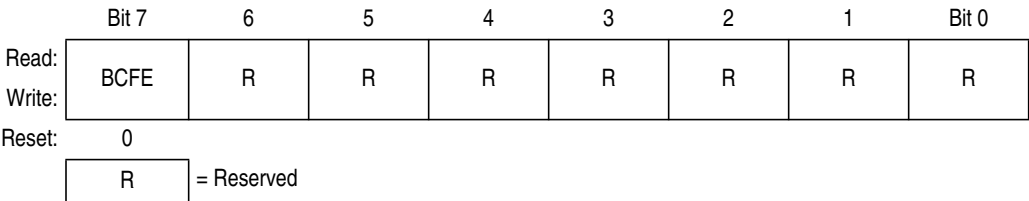


Figure 14-20. Break Flag Control Register (BFCR)

BCFE — Break Clear Flag Enable Bit

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

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

Serial Peripheral Interface (SPI) Module

The error interrupt enable bit (ERRIE) enables both the MODF and OVRF bits to generate a receiver/error interrupt request.

The mode fault enable bit (MODFEN) can prevent the MODF flag from being set so that only the OVRF bit is enabled by the ERRIE bit to generate receiver/error interrupt requests.

The following sources in the SPI status and control register can generate interrupt requests:

- SPI receiver full bit (SPRF) — SPRF becomes set every time a byte transfers from the shift register to the receive data register. If the SPI receiver interrupt enable bit, SPRIE, is also set, SPRF generates an SPI receiver/error interrupt request.
- SPI transmitter empty bit (SPTE) — SPTE becomes set every time a byte transfers from the transmit data register to the shift register. If the SPI transmit interrupt enable bit, SPTIE, is also set, SPTE generates an SPTE interrupt request.

15.5 Low-Power Modes

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

15.5.1 Wait Mode

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

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

To exit wait mode when an overflow condition occurs, enable the OVRF bit to generate interrupt requests by setting the error interrupt enable bit (ERRIE). See 15.4 Interrupts.

15.5.2 Stop Mode

The SPI module is inactive after the execution of a STOP instruction. The STOP instruction does not affect register conditions. SPI operation resumes after an external interrupt. If stop mode is exited by reset, any transfer in progress is aborted, and the SPI is reset.

15.6 SPI During Break Interrupts

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

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

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

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
Write:								
Reset:	0	0	1	0	1	0	0	0

R = Reserved

Figure 15-13. SPI Control Register (SPCR)

SPRIE — SPI Receiver Interrupt Enable Bit

This read/write bit enables interrupt requests generated by the SPRF bit. The SPRF bit is set when a byte transfers from the shift register to the receive data register.

- 1 = SPRF interrupt requests enabled
- 0 = SPRF interrupt requests disabled

SPMSTR — SPI Master Bit

This read/write bit selects master mode operation or slave mode operation.

- 1 = Master mode
- 0 = Slave mode

CPOL — Clock Polarity Bit

This read/write bit determines the logic state of the SPSCCK pin between transmissions. (See Figure 15-4 and Figure 15-6.) To transmit data between SPI modules, the SPI modules must have identical CPOL values.

CPHA — Clock Phase Bit

This read/write bit controls the timing relationship between the serial clock and SPI data. (See Figure 15-4 and Figure 15-6.) To transmit data between SPI modules, the SPI modules must have identical CPHA values. When CPHA = 0, the \overline{SS} pin of the slave SPI module must be high between bytes. (See Figure 15-12.)

SPWOM — SPI Wired-OR Mode Bit

This read/write bit configures pins SPSCCK, MOSI, and MISO so that these pins become open-drain outputs.

- 1 = Wired-OR SPSCCK, MOSI, and MISO pins
- 0 = Normal push-pull SPSCCK, MOSI, and MISO pins

SPE — SPI Enable

This read/write bit enables the SPI module. Clearing SPE causes a partial reset of the SPI. (See 15.3.5 Resetting the SPI.)

- 1 = SPI module enabled
- 0 = SPI module disabled

SPTIE— SPI Transmit Interrupt Enable

This read/write bit enables interrupt requests generated by the SPTE bit. SPTE is set when a byte transfers from the transmit data register to the shift register.

- 1 = SPTE interrupt requests enabled
- 0 = SPTE interrupt requests disabled

18.2.2 Break Module Registers

These registers control and monitor operation of the break module:

- Break status and control register (BRKSCR)
- Break address register high (BRKH)
- Break address register low (BRKL)
- Break status register (BSR)
- Break flag control register (BFCR)

18.2.2.1 Break Status and Control Register

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

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


 = Unimplemented

Figure 18-3. Break Status and Control Register (BRKSCR)

BRKE — Break Enable Bit

This read/write bit enables breaks on break address register matches. Clear BRKE by writing a 0 to bit 7. Reset clears the BRKE bit.

- 1 = Breaks enabled on 16-bit address match
- 0 = Breaks disabled

BRKA — Break Active Bit

This read/write status and control bit is set when a break address match occurs. Writing a 1 to BRKA generates a break interrupt. Clear BRKA by writing a 0 to it before exiting the break routine. Reset clears the BRKA bit.

- 1 = Break address match
- 0 = No break address match

18.2.2.2 Break Address Registers

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

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

Figure 18-4. Break Address Register High (BRKH)

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

Figure 18-5. Break Address Register Low (BRKL)