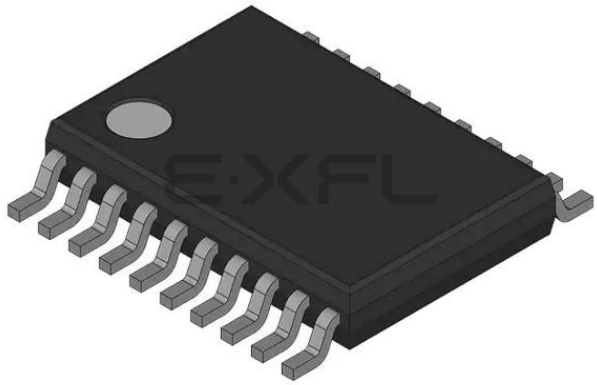

Welcome to E-XFL.COM

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	Active
Core Processor	S08
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, LINbus, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	16
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	20-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	20-TSSOP
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc9s08sl8vtj

Table 1-2 provides the functional version of the on-chip modules

Table 1-2. Module Versions

Module		Version
Central Processor Unit	(CPU)	3
Internal Clock Source	(ICS)	2
5-V Analog Comparator	(ACMP_5V)	2
Analog-to-Digital Converter	(ADC)	1
Inter-Integrated Circuit	(IIC)	2
Slave LIN Interface Controller	(SLIC)	1
Serial Peripheral Interface	(SPI)	3
Serial Communications Interface	(SCI)	4
Real-Time Counter	(RTC)	1
Timer Pulse Width Modulator	(TPM)	2
On-Chip ICE Debug	(DBG)	2

4.5.11.6 FLASH and EEPROM Command Register (FCMD)

Only six command codes are recognized in normal user modes as shown in Table 4-15. All other command codes are illegal and generate an access error. Refer to Section 4.5.3, "Program and Erase Command Execution," for a detailed discussion of FLASH and EEPROM programming and erase operations.

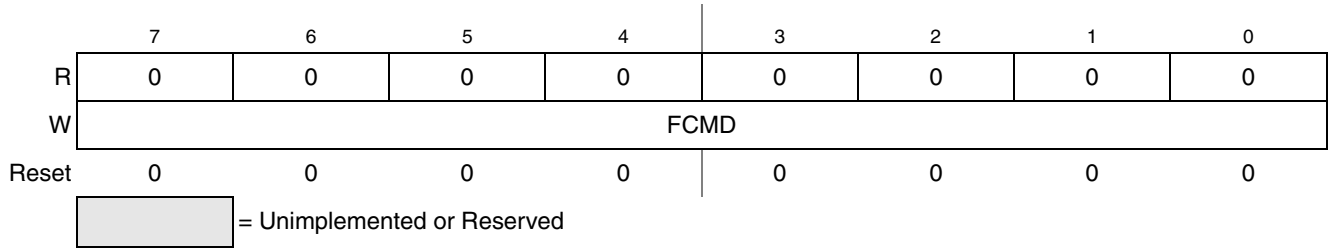

Figure 4-10. FLASH and EEPROM Command Register (FCMD)

Table 4-15. FLASH and EEPROM Commands

Command	FCMD	Equate File Label
Blank check	0x05	mBlank
Byte program	0x20	mByteProg
Burst program	0x25	mBurstProg
Sector erase	0x40	mSectorErase
Mass erase	0x41	mMassErase
Sector erase abort	0x47	mEraseAbort

It is not necessary to perform a blank check command after a mass erase operation. Only blank check is required as part of the security unlocking mechanism.

6.5.2.5 Port B Drive Strength Selection Register (PTBDS)

	7	6	5	4	3	2	1	0
R W	PTBDS7	PTBDS6	PTBDS5	PTBDS4	PTBDS3	PTBDS2	PTBDS1	PTBDS0
Reset:	0	0	0	0	0	0	0	0

Figure 6-15. Drive Strength Selection for Port B Register (PTBDS)

Table 6-13. PTBDS Register Field Descriptions

Field	Description
7:0 PTBDS[7:0]	**Output Drive Strength Selection for Port B Bits** — Each of these control bits selects between low and high output drive for the associated PTB pin. For port B pins that are configured as inputs, these bits have no effect. 0 Low output drive strength selected for port B bit n. 1 High output drive strength selected for port B bit n.

6.5.2.6 Port B Interrupt Status and Control Register (PTBSC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTBIF	0	PTBIE	PTBMOD
W						PTBACK		
Reset:	0	0	0	0	0	0	0	0

Figure 6-16. Port B Interrupt Status and Control Register (PTBSC)

Table 6-14. PTBSC Register Field Descriptions

Field	Description
3 PTBIF	**Port B Interrupt Flag** — PTBIF indicates when a Port B interrupt is detected. Writes have no effect on PTBIF. 0 No Port B interrupt detected. 1 Port B interrupt detected.
2 PTBACK	**Port B Interrupt Acknowledge** — Writing a 1 to PTBACK is part of the flag clearing mechanism. PTBACK always reads as 0.
1 PTBIE	**Port B Interrupt Enable** — PTBIE determines whether a port B interrupt is requested. 0 Port B interrupt request not enabled. 1 Port B interrupt request enabled.
0 PTBMOD	**Port B Detection Mode** — PTBMOD (along with the PTBES bits) controls the detection mode of the port B interrupt pins. 0 Port B pins detect edges only. 1 Port B pins detect both edges and levels.

Chapter 8
Internal Clock Source (S08ICSV2)

8.1 Introduction

The internal clock source (ICS) module provides clock source choices for the MCU. The module contains a frequency-locked loop (FLL) as a clock source that is controllable by either an internal or an external reference clock. The module can provide this FLL clock or either of the internal or external reference clocks as a source for the MCU system clock. There are also signals provided to control a low power oscillator (XOSC) module to allow the use of an external crystal/resonator as the external reference clock.

Whichever clock source is chosen, it is passed through a reduced bus divider (BDIV) which allows a lower final output clock frequency to be derived.

The bus frequency is half of the ICSOUT frequency. After reset, the ICS is configured for FEI mode and BDIV resets to 01 to introduce an extra divide-by-two before ICSOUT. Therefore, the bus frequency is $f_{dco}/4$. At POR, the TRIM and FTRIM are reset to 0x80 and 0, respectively. Therefore, the dco frequency is f_{dco_ut}. For other resets, the trim settings keep the value that was present before the reset.

NOTE

Refer to Section 1.3, “System Clock Distribution”, for a detailed view of the distribution of clock sources throughout the MCU.

8.1.1 Module Configuration

When the internal reference is enabled in stop mode (IREFSTEN = 1), the voltage regulator must also be enabled in stop mode by setting the LVDE and LVDSE bits in the SPMSC1 register.

Figure 8-1 shows the MC9S08EL32 block diagram with the ICS highlighted.

If EREFSTEN is set and the ERCLKEN bit is written to 1, the external reference clock will keep running during stop mode in order to provide a fast recovery upon exiting stop.

8.4.7 Fixed Frequency Clock

The ICS presents the divided FLL reference clock as ICSFFCLK for use as an additional clock source for peripheral modules. The ICS provides an output signal (ICSFFE) which indicates when the ICS is providing ICSOUT frequencies four times or greater than the divided FLL reference clock (ICSFFCLK). In FLL Engaged mode (FEI and FEE) this is always true and ICSFFE is always high. In ICS Bypass modes, ICSFFE will get asserted for the following combinations of BDIV and RDIV values:

- BDIV=00 (divide by 1), RDIV ≥ 010
- BDIV=01 (divide by 2), RDIV ≥ 011
- BDIV=10 (divide by 4), RDIV ≥ 100
- BDIV=11 (divide by 8), RDIV ≥ 101

10.6.2 Sources of Error

Several sources of error exist for A/D conversions. These are discussed in the following sections.

10.6.2.1 Sampling Error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy. Given the maximum input resistance of approximately 7kΩ and input capacitance of approximately 5.5 pF, sampling to within 1/4LSB (at 10-bit resolution) can be achieved within the minimum sample window (3.5 cycles @ 8 MHz maximum ADCK frequency) provided the resistance of the external analog source (R_{AS}) is kept below 5 kΩ.

Higher source resistances or higher-accuracy sampling is possible by setting ADLSMP (to increase the sample window to 23.5 cycles) or decreasing ADCK frequency to increase sample time.

10.6.2.2 Pin Leakage Error

Leakage on the I/O pins can cause conversion error if the external analog source resistance (R_{AS}) is high. If this error cannot be tolerated by the application, keep R_{AS} lower than V_{DDAD} / (2^N*I_{LEAK}) for less than 1/4LSB leakage error (N = 8 in 8-bit mode or 10 in 10-bit mode).

10.6.2.3 Noise-Induced Errors

System noise which occurs during the sample or conversion process can affect the accuracy of the conversion. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1 µF low-ESR capacitor from V_{REFH} to V_{REFL}.
- There is a 0.1 µF low-ESR capacitor from V_{DDAD} to V_{SSAD}.
- If inductive isolation is used from the primary supply, an additional 1 µF capacitor is placed from V_{DDAD} to V_{SSAD}.
- V_{SSAD} (and V_{REFL}, if connected) is connected to V_{SS} at a quiet point in the ground plane.
- Operate the MCU in wait or stop3 mode before initiating (hardware triggered conversions) or immediately after initiating (hardware or software triggered conversions) the ADC conversion.
  - For software triggered conversions, immediately follow the write to the ADCSC1 with a WAIT instruction or STOP instruction.
  - For stop3 mode operation, select ADACK as the clock source. Operation in stop3 reduces V_{DD} noise but increases effective conversion time due to stop recovery.
- There is no I/O switching, input or output, on the MCU during the conversion.

There are some situations where external system activity causes radiated or conducted noise emissions or excessive V_{DD} noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in wait or stop3 or I/O activity cannot be halted, these recommended actions may reduce the effect of noise on the accuracy:

- Place a 0.01 µF capacitor (C_{AS}) on the selected input channel to V_{REFL} or V_{SSAD} (this will improve noise issues but will affect sample rate based on the external analog source resistance).

11.3.3 IIC Control Register (IICC1)

	7	6	5	4	3	2	1	0
R	IICEN	IICIE	MST	TX	TXAK	0	0	0
W						RSTA		
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-5. IIC Control Register (IICC1)

Table 11-6. IICC1 Field Descriptions

Field	Description
7 IICEN	**IIC Enable.** The IICEN bit determines whether the IIC module is enabled. 0 IIC is not enabled 1 IIC is enabled
6 IICIE	**IIC Interrupt Enable.** The IICIE bit determines whether an IIC interrupt is requested. 0 IIC interrupt request not enabled 1 IIC interrupt request enabled
5 MST	**Master Mode Select.** The MST bit changes from a 0 to a 1 when a start signal is generated on the bus and master mode is selected. When this bit changes from a 1 to a 0 a stop signal is generated and the mode of operation changes from master to slave. 0 Slave mode 1 Master mode
4 TX	**Transmit Mode Select.** The TX bit selects the direction of master and slave transfers. In master mode, this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit is always high. When addressed as a slave, this bit should be set by software according to the SRW bit in the status register. 0 Receive 1 Transmit
3 TXAK	**Transmit Acknowledge Enable.** This bit specifies the value driven onto the SDA during data acknowledge cycles for master and slave receivers. 0 An acknowledge signal is sent out to the bus after receiving one data byte 1 No acknowledge signal response is sent
2 RSTA	**Repeat start.** Writing a 1 to this bit generates a repeated start condition provided it is the current master. This bit is always read as cleared. Attempting a repeat at the wrong time results in loss of arbitration.

11.3.4 IIC Status Register (IICS)

	7	6	5	4	3	2	1	0
R	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
W								
Reset	1	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-6. IIC Status Register (IICS)

	7	6	5	4	3	2	1	0
R	SLCACT	0	INITACK	0	0	0	0	SLCF
W								
Reset	0	0	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-8. SLIC Status Register (SLCS)

Table 12-6. SLCS Field Descriptions

Field	Description
7 SLCACT[1]	**SLIC Active (Oscillator Trim Blocking Semaphore)** — SLCACT is used to indicate if it is safe to trim the oscillator based upon current SLIC activity in LIN mode. This bit indicates that the SLIC module might be currently receiving a message header, synchronization byte, ID byte, or sending or receiving data bytes. This bit is read-only. This bit has no meaning in BTM mode (BTM =1). 0 SLIC module not active (safe to trim oscillator) SLCACT is cleared by the SLIC module only upon assertion of the RX Message Buffer Full Checksum OK (SLCSV = 0x10) or the TX Message Buffer Empty Checksum Transmitted (SLCSV = 0x08) interrupt sources. 1 SLIC module activity (not safe to trim oscillator) SLCACT is automatically set to 1 if a falling edge is seen on the SLCRX pin and has successfully been passed through the digital RX filter. This edge is the potential beginning of a LIN message frame.
5 INITACK	**Initialization Mode Acknowledge** — INITACK indicates whether the SLIC module is in the reset mode as a result of writing INITREQ in SLCC1. INITACK = 1 causes all SLIC register bits (except SLCWCM: write once) to be held in their reset state and become not writable until INITACK has been cleared. Clear INITACK by clearing INITREQ in SLCC1. After INITACK is cleared, the SLIC module proceeds to SLIC DISABLED mode (see Figure 12-2) in which the other SLIC register bits are writable and can be configured to the desired SLIC operating mode. INITACK is a read-only bit. 0 Normal operation 1 SLIC module is in reset state
0 SLCF	**SLIC Interrupt Flag** — The SLCF interrupt flag indicates if a SLIC module interrupt is pending. If set, the SLCSV is then used to determine what interrupt is pending. This flag is cleared by writing a 1 to the bit. If additional interrupt sources are pending, the bit will be automatically set to 1 again by the SLIC. 0 No SLIC interrupt pending 1 SLIC interrupt pending

[1] SLCACT may not be clear during all idle times of the bus. For example, if IMSG was used to ignore the data interrupts of an extended message frame, SLCACT will remain set until another LIN message is received and either the RX Message Buffer Full Checksum OK (SLCSV = 0x10) or the TX Message Buffer Empty Checksum Transmitted (SLCSV = 0x08) interrupt sources are asserted and cleared. When clear, SLCACT always indicates times when the SLIC module is not active, but it is possible for the SLIC module to be not active with SLCACT set. SLCACT has no meaning in BTM mode.

12.3.5 SLIC State Vector Register (SLCSV)

SLIC state vector register (SLCSV) is provided to substantially decrease the CPU overhead associated with servicing interrupts while under operation of a LIN protocol. It provides an index offset that is directly related to the LIN module's current state, which can be used with a user supplied jump table to rapidly enter an interrupt service routine. This eliminates the need for the user to maintain a duplicate state machine in software.

method was employed for this message frame. Refer to the LIN specification for more details on the calculations.

- **Byte Framing Error**
  This error comes from the standard UART definition for byte encoding and occurs when the STOP bit is sampled and reads back as a 090. STOP should always read as 1.

NOTE

A byte framing error can also be an indication that the number of data bytes received in a LIN message frame does not match the value written to the SLCDLC register. See Section 12.6.7, "Handling LIN Message Headers," for more details.

- **Identifier Received Successfully**
  This interrupt source indicates that a LIN identifier byte has been received with correct parity and is waiting in the LIN identifier buffer (SLCID). Upon reading this interrupt source from SLCSV, the user can then decode the identifier in software to determine the nature of the LIN message frame. To clear this source, SLCID must be read.
- **Identifier-Parity-Error**
  A parity error in the identifier (i.e., corrupted identifier) will be flagged. Typical LIN slave applications do not distinguish between an unknown but valid identifier, and a corrupted identifier. However, it is mandatory for all slave nodes to evaluate in case of a known identifier all eight bits of the ID-Field and distinguish between a known and a corrupted identifier. The received identifier value is reported in SLCID so that the user software can choose to acknowledge or ignore the parity error message. Once the ID parity error has been detected, the SLIC will begin looking for another LIN header and will not receive message data, even if it appears on the bus.
- **Wakeup**
  The wakeup interrupt source indicates that the SLIC module has entered SLIC run mode from SLIC stop mode.

12.3.5.2 Byte Transfer Mode Operation

When byte transfer mode is enabled (BTM = 1), many of the interrupt sources for the SLCSV no longer apply, as they are specific to LIN operations. Table 12-9 shows those interrupt sources which are applicable to BTM operations. The value of the SLCSV for each interrupt source remains the same, as well as the priority of the interrupt source.

Table 12-9. Interrupt Sources Summary (BTM = 1)

SLCSV	I3	I2	I1	I0	Interrupt Source	Priority
0x00	0	0	0	0	No Interrupts Pending	0 (Lowest)
0x0C	0	0	1	1	TX Message Buffer Empty	3
0x14	0	1	0	1	RX Data Buffer Full No Errors	5
0x18	0	1	1	0	Bit-Error	6
0x1C	0	1	1	1	Receiver Buffer Overrun	7

- The third section of the message frame header is the IDENTIFIER FIELD (ID). The identifier is covered more in Section 12.6.8, “Handling Command Message Frames,” and Section 12.6.9, “Handling Request LIN Message Frames.”

The SLIC automatically reads the incoming pattern of the SYNCHRONIZATION BREAK and FIELD and determines the bit rate of the LIN data frame, as well as checking for errors in form and discerning between a genuine BREAK/FIELD combination and a similar byte pattern somewhere in the data stream. After the header has been verified to be valid and has been processed, the SLIC module updates the SLIC bit time register (SLCBT) with the value obtained from the SYNCH FIELD and begins to receive the ID.

After the ID for the message frame has been received, an interrupt is generated by the SLIC and will trigger an MCU interrupt request if unmasked. At this point, it might be possible that the ID was received with errors such as a parity error (based on the LIN specification) or a byte framing error. If the ID did not have any errors, it will be copied into the SLCD for the software to read. The SLCSV will indicate the type error or that the ID was received correctly.

In a LIN system, the meaning and function of all messages, and therefore all message identifiers, is pre-defined by the system designer. This information can be collected and stored in a standardized format file, called a Configuration Language Description (CLD) file. In using the SLIC module, it is the responsibility of the user software to determine the nature of the incoming message, and therefore how to further handle that message.

The simplest case is when the SLIC receives a message which the user software determines is of no interest to the application. In other words, the slave node does not need to receive or transmit any data for this message frame. This might also apply to messages with zero data bytes (which is allowed by the LIN specification). At this point, the user can set the IMSG control bit, and exit the interrupt service routine by clearing the SLCIF flag. Because there is no data to be sent or received, the SLIC will not generate another interrupt until the next message frame header or bus goes idle long enough to trigger a “No-Bus-Activity” error according to the LIN specification.

NOTE

IMSG will prevent another interrupt from occurring for the current message frame; however, if data bytes are appearing on the bus they may be received and copied into the message buffer. This will delete any previous data which might have been present in the buffer, even though no interrupt is triggered to indicate the arrival of this data.

At the time the ID is read, the user might also choose to read SLCBT and copy this value out to an application variable. This data can then be used at a time appropriate to both the application software and the LIN communications to adjust the trim of the internal oscillator. This operation must be handled very carefully to avoid adjusting the base timing of the MCU at the wrong time, adversely affecting the operation of the SLIC module or of the application itself. More information about this is contained in Section 12.6.17, “Oscillator Trimming with SLIC.”

If the user software determines that the ID read out of the SLCD corresponds to a command or request message for which this node needs to receive or transmit data (respectively), it will then move on to procedures described in Section 12.6.8, “Handling Command Message Frames,” and Section 12.6.9, “Handling Request LIN Message Frames.”

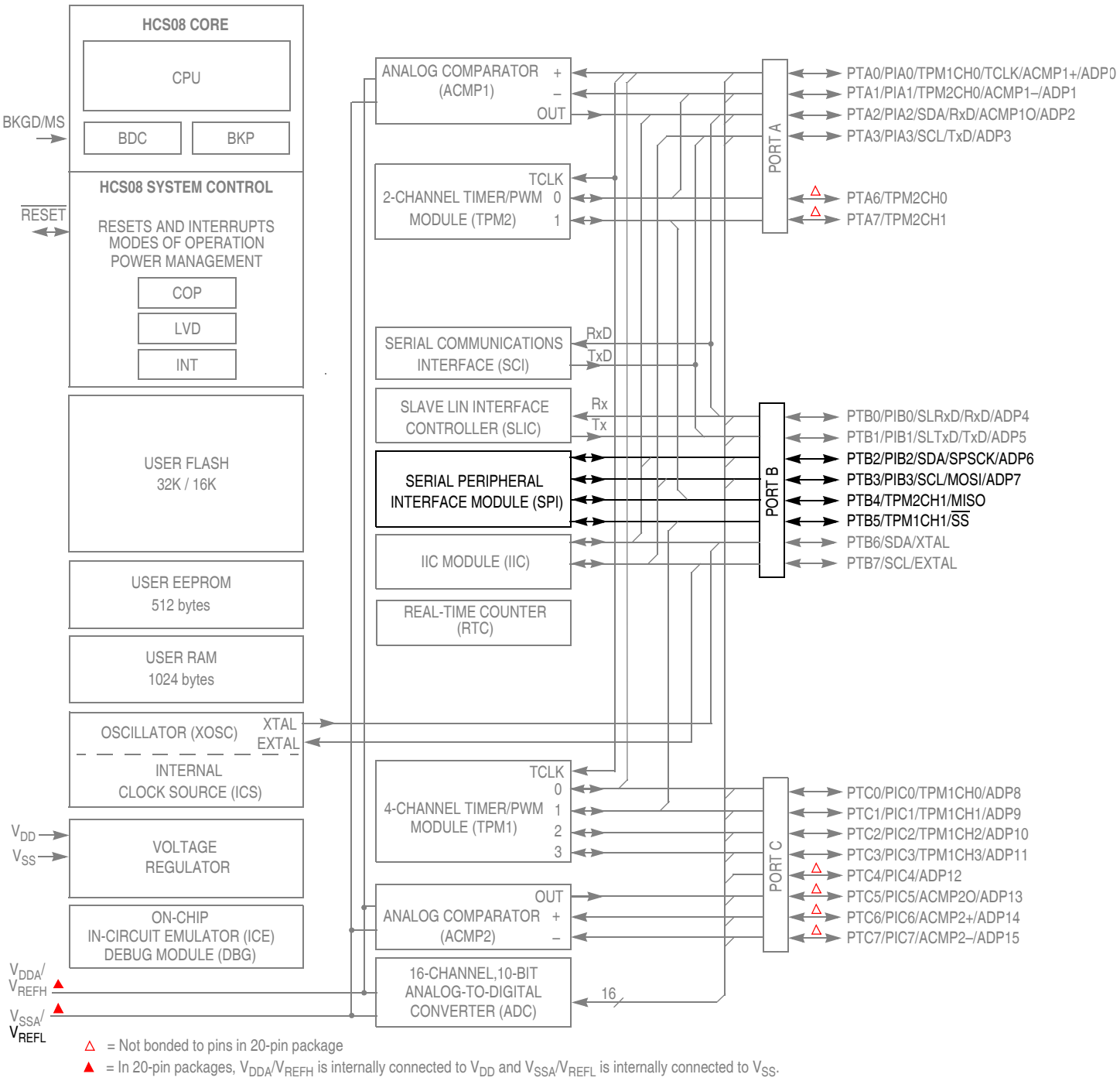

Figure 13-1. MC9S08EL32 Block Diagram Highlighting SPI Block and Pins

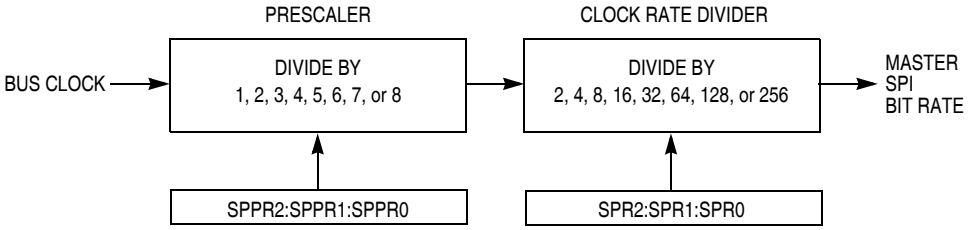

Figure 13-4. SPI Baud Rate Generation

13.2 External Signal Description

The SPI optionally shares four port pins. The function of these pins depends on the settings of SPI control bits. When the SPI is disabled (SPE = 0), these four pins revert to being general-purpose port I/O pins that are not controlled by the SPI.

13.2.1 SPSCK — SPI Serial Clock

When the SPI is enabled as a slave, this pin is the serial clock input. When the SPI is enabled as a master, this pin is the serial clock output.

13.2.2 MOSI — Master Data Out, Slave Data In

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data output. When the SPI is enabled as a slave and SPC0 = 0, this pin is the serial data input. If SPC0 = 1 to select single-wire bidirectional mode, and master mode is selected, this pin becomes the bidirectional data I/O pin (MOMI). Also, the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE = 0) or an output (BIDIROE = 1). If SPC0 = 1 and slave mode is selected, this pin is not used by the SPI and reverts to being a general-purpose port I/O pin.

13.2.3 MISO — Master Data In, Slave Data Out

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data input. When the SPI is enabled as a slave and SPC0 = 0, this pin is the serial data output. If SPC0 = 1 to select single-wire bidirectional mode, and slave mode is selected, this pin becomes the bidirectional data I/O pin (SISO) and the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE = 0) or an output (BIDIROE = 1). If SPC0 = 1 and master mode is selected, this pin is not used by the SPI and reverts to being a general-purpose port I/O pin.

13.2.4 $\overline{SS}$ — Slave Select

When the SPI is enabled as a slave, this pin is the low-true slave select input. When the SPI is enabled as a master and mode fault enable is off (MODFEN = 0), this pin is not used by the SPI and reverts to being a general-purpose port I/O pin. When the SPI is enabled as a master and MODFEN = 1, the slave select output enable bit determines whether this pin acts as the mode fault input (SSOE = 0) or as the slave select output (SSOE = 1).

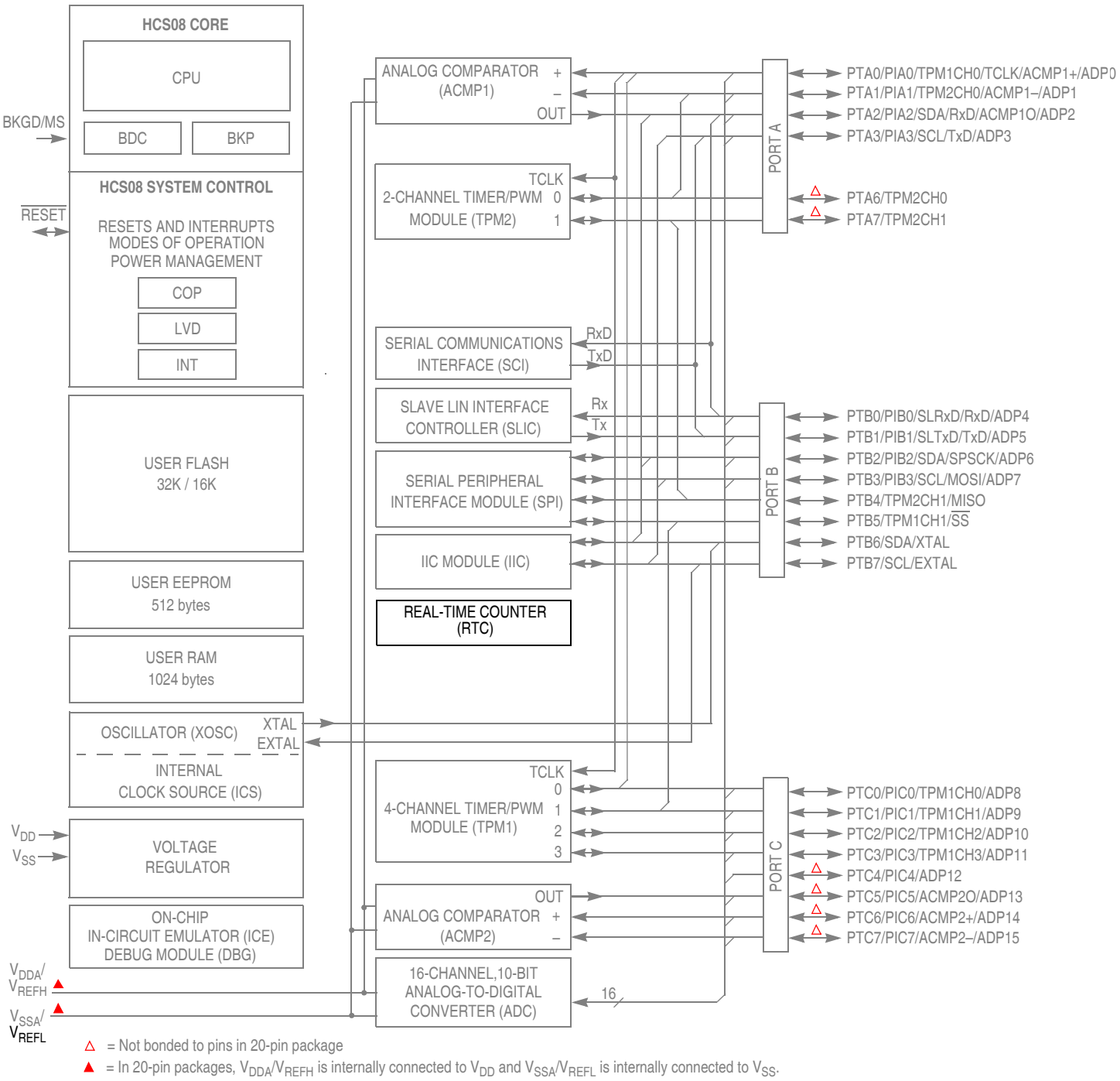

Figure 15-1. MC9S08EL32 Block Diagram Highlighting RTC Block

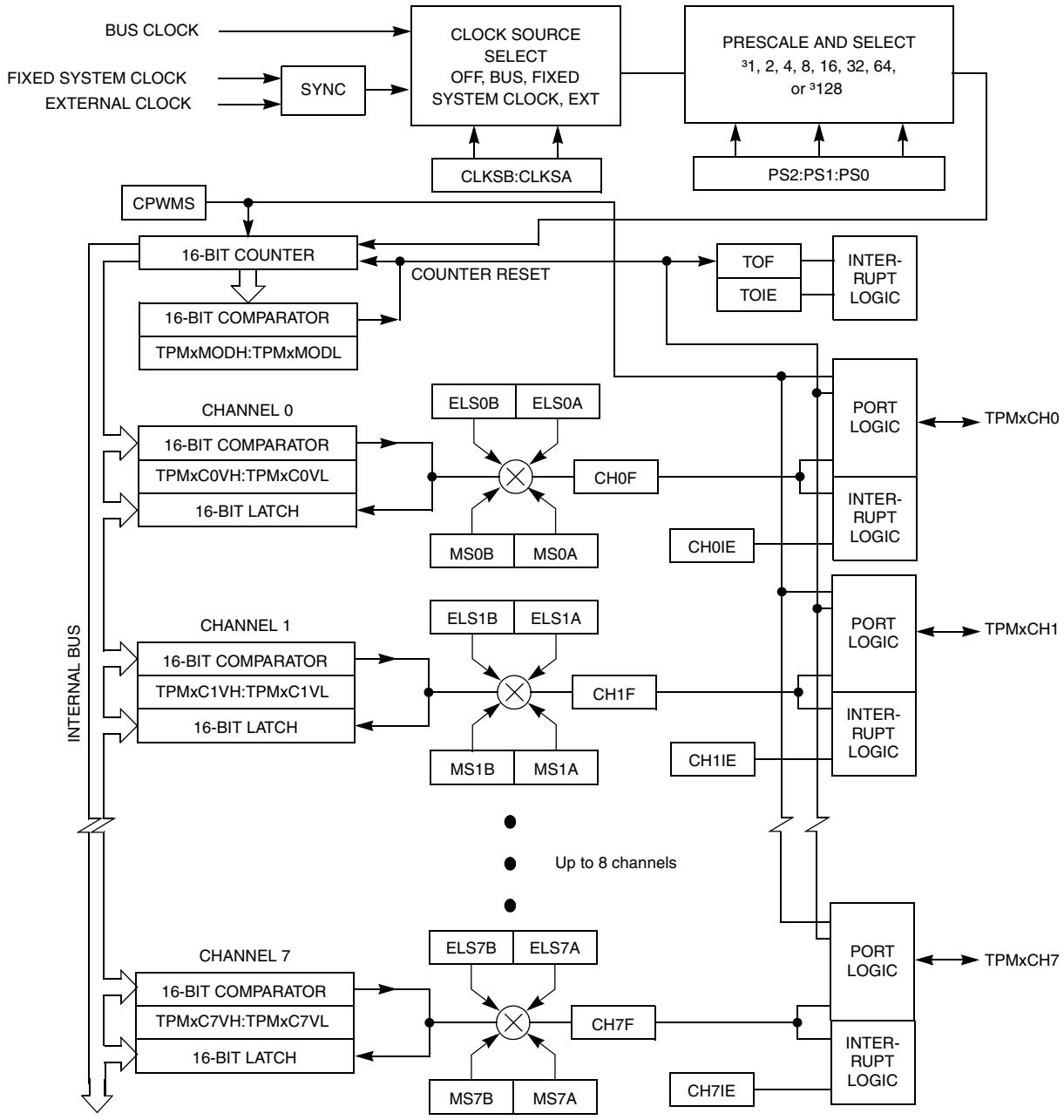

Figure 16-2. TPM Block Diagram

17.1.2 Features

Features of the BDC module include:

- Single pin for mode selection and background communications
- BDC registers are not located in the memory map
- SYNC command to determine target communications rate
- Non-intrusive commands for memory access
- Active background mode commands for CPU register access
- GO and TRACE1 commands
- BACKGROUND command can wake CPU from stop or wait modes
- One hardware address breakpoint built into BDC
- Oscillator runs in stop mode, if BDC enabled
- COP watchdog disabled while in active background mode

Features of the ICE system include:

- Two trigger comparators: Two address + read/write (R/W) or one full address + data + R/W
- Flexible 8-word by 16-bit FIFO (first-in, first-out) buffer for capture information:
  - Change-of-flow addresses or
  - Event-only data
- Two types of breakpoints:
  - Tag breakpoints for instruction opcodes
  - Force breakpoints for any address access
- Nine trigger modes:
  - Basic: A-only, A OR B
  - Sequence: A then B
  - Full: A AND B data, A AND NOT B data
  - Event (store data): Event-only B, A then event-only B
  - Range: Inside range ($A \leq address \leq B$), outside range ($address < A$ or $address > B$)

17.2 Background Debug Controller (BDC)

All MCUs in the HCS08 Family contain a single-wire background debug interface that supports in-circuit programming of on-chip nonvolatile memory and sophisticated non-intrusive debug capabilities. Unlike debug interfaces on earlier 8-bit MCUs, this system does not interfere with normal application resources. It does not use any user memory or locations in the memory map and does not share any on-chip peripherals.

BDC commands are divided into two groups:

- Active background mode commands require that the target MCU is in active background mode (the user program is not running). Active background mode commands allow the CPU registers to be read or written, and allow the user to trace one user instruction at a time, or GO to the user program from active background mode.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								BDFR[1]
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

1 BDFR is writable only through serial background mode debug commands, not from user programs.

Figure 17-7. System Background Debug Force Reset Register (SBDFR)

Table 17-3. SBDFR Register Field Description

Field	Description
0 BDFR	**Background Debug Force Reset** — A serial active background mode command such as WRITE_BYTE allows an external debug host to force a target system reset. Writing 1 to this bit forces an MCU reset. This bit cannot be written from a user program.

17.4.3 DBG Registers and Control Bits

The debug module includes nine bytes of register space for three 16-bit registers and three 8-bit control and status registers. These registers are located in the high register space of the normal memory map so they are accessible to normal application programs. These registers are rarely if ever accessed by normal user application programs with the possible exception of a ROM patching mechanism that uses the breakpoint logic.

17.4.3.1 Debug Comparator A High Register (DBGCAH)

This register contains compare value bits for the high-order eight bits of comparator A. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

17.4.3.2 Debug Comparator A Low Register (DBGCAL)

This register contains compare value bits for the low-order eight bits of comparator A. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

17.4.3.3 Debug Comparator B High Register (DBGCBH)

This register contains compare value bits for the high-order eight bits of comparator B. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

17.4.3.4 Debug Comparator B Low Register (DBGCBL)

This register contains compare value bits for the low-order eight bits of comparator B. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

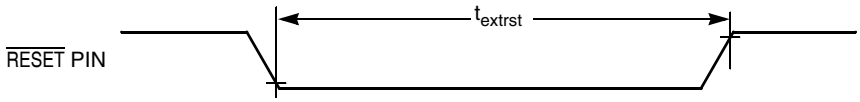

Figure A-10. Reset Timing

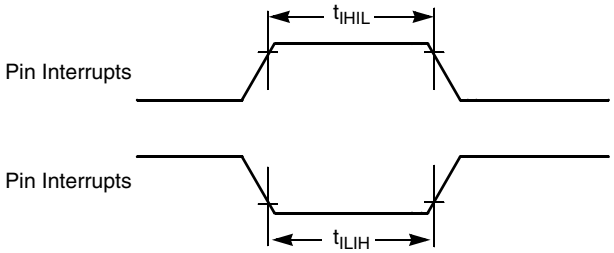

Figure A-11. Pin Interrupt Timing

A.12.2 TPM/MTIM Module Timing

Synchronizer circuits determine the shortest input pulses that can be recognized or the fastest clock that can be used as the optional external source to the timer counter. These synchronizers operate from the current bus rate clock.

Table A-14. TPM Input Timing

Num	C	Rating	Symbol	Min	Max	Unit
1	—	External clock frequency ($1/t_{TCLK}$)	f_{TCLK}	dc	$f_{Bus}/4$	MHz
2	—	External clock period	t_{TCLK}	4	—	t_{cyc}
3	—	External clock high time	t_{clkh}	1.5	—	t_{cyc}
4	—	External clock low time	t_{clkl}	1.5	—	t_{cyc}
5	—	Input capture pulse width	t_{ICPW}	1.5	—	t_{cyc}

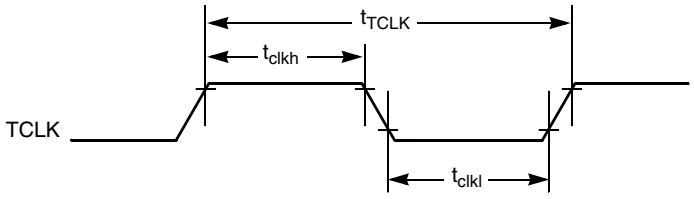

Figure A-12. Timer External Clock