

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	8MHz
Connectivity	SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	60KB (60K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	32-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc9s08rg60cfje

Section Number	Title	Page
13.3.2.3	MISO1 — Master Data In, Slave Data Out	170
13.3.2.4	$\overline{SS1}$ — Slave Select	170
13.3.3	SPI Interrupts	171
13.3.4	Mode Fault Detection	171
13.4	SPI Registers and Control Bits	171
13.4.1	SPI Control Register 1 (SPI1C1)	172
13.4.2	SPI Control Register 2 (SPI1C2)	173
13.4.3	SPI Baud Rate Register (SPI1BR)	174
13.4.4	SPI Status Register (SPI1S)	176
13.4.5	SPI Data Register (SPI1D)	177

Chapter 14 Analog Comparator (S08ACMPV1)

14.1	Features	180
14.2	Block Diagram	180
14.3	Pin Description	180
14.4	Functional Description	181
14.4.1	Interrupts	181
14.4.2	Wait Mode Operation	181
14.4.3	Stop Mode Operation	181
14.4.4	Background Mode Operation	181
14.5	ACMP Status and Control Register (ACMP1SC)	182

Chapter 15 Development Support

15.1	Introduction	183
15.1.1	Features	183
15.2	Background Debug Controller (BDC)	184
15.2.1	BKGD Pin Description	184
15.2.2	Communication Details	185
15.2.3	BDC Commands	189
15.2.4	BDC Hardware Breakpoint	191
15.3	On-Chip Debug System (DBG)	192
15.3.1	Comparators A and B	192
15.3.2	Bus Capture Information and FIFO Operation	192
15.3.3	Change-of-Flow Information	193
15.3.4	Tag vs. Force Breakpoints and Triggers	193
15.3.5	Trigger Modes	194
15.3.6	Hardware Breakpoints	196
15.4	Register Definition	196
15.4.1	BDC Registers and Control Bits	196
15.4.1.1	BDC Status and Control Register (BDCSCR)	197
15.4.1.2	BDC Breakpoint Match Register (BDCBKPT)	198
15.4.2	System Background Debug Force Reset Register (SBDFR)	198

Chapter 2 Pins and Connections

2.1 Introduction

This section describes signals that connect to package pins. It includes a pinout diagram, a table of signal properties, and detailed discussion of signals.

2.2 Device Pin Assignment

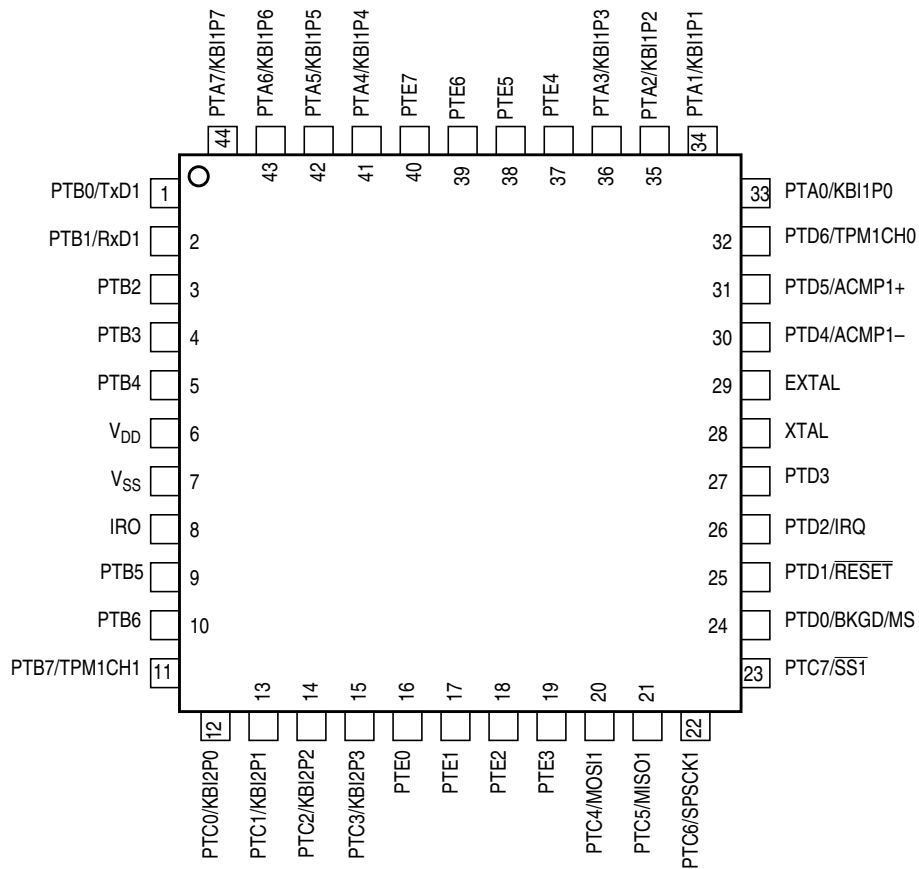


Figure 2-1. MC9S08RC/RD/RE/RG in 44-Pin LQFP Package

3.6.4 Active BDM Enabled in Stop Mode

Entry into the active background mode from run mode is enabled if the ENBDM bit in BDCSCR is set. This register is described in the Development Support chapter of this data sheet. If ENBDM is set when the CPU executes a STOP instruction, the system clocks to the background debug logic remain active when the MCU enters stop mode so background debug communication is still possible. In addition, the voltage regulator does not enter its low-power standby state but maintains full internal regulation. The MCU cannot enter either stop1 mode or stop2 mode if ENBDM is set.

Most background commands are not available in stop mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from stop and enter active background mode if the ENBDM bit is set. After active background mode is entered, all background commands are available. Table 3-2 summarizes the behavior of the MCU in stop when entry into the active background mode is enabled.

Table 3-2. BDM Enabled Stop Mode Behavior

Mode	PDC	PPDC	CPU, Digital Peripherals, FLASH	RAM	OSC	ACMP	Regulator	I/O Pins	RTI
Stop3	Don't care	Don't care	Standby	Standby	On	Standby	On	States held	Optionally on

3.6.5 LVD Reset Enabled

The LVD system is capable of generating either an interrupt or a reset when the supply voltage drops below the LVD voltage. If the LVD reset is enabled in stop by setting the LVDRE bit in SPMSC1 when the CPU executes a STOP instruction, then the voltage regulator remains active during stop mode. If the user attempts to enter either stop1 or stop2 with the LVD reset enabled (LVDRE = 1) the MCU will instead enter stop3. Table 3-3 summarizes the behavior of the MCU in stop when LVD reset is enabled.

Table 3-3. LVD Enabled Stop Mode Behavior

Mode	PDC	PPDC	CPU, Digital Peripherals, FLASH	RAM	OSC	ACMP	Regulator	I/O Pins	RTI
Stop3	Don't care	Don't care	Standby	Standby	On	Standby	On	States held	Optionally on

3.6.6 On-Chip Peripheral Modules in Stop Mode

When the MCU enters any stop mode, system clocks to the internal peripheral modules are stopped. Even in the exception case (ENBDM = 1), where clocks are kept alive to the background debug logic, clocks to the peripheral systems are halted to reduce power consumption.

- Writing to a FLASH address before the internal FLASH clock frequency has been set by writing to the FCDIV register
- Writing to a FLASH address while FCBEF is not set (A new command cannot be started until the command buffer is empty.)
- Writing a second time to a FLASH address before launching the previous command (There is only one write to FLASH for every command.)
- Writing a second time to FCMD before launching the previous command (There is only one write to FCMD for every command.)
- Writing to any FLASH control register other than FCMD after writing to a FLASH address
- Writing any command code other than the five allowed codes (\$05, \$20, \$25, \$40, or \$41) to FCMD
- Accessing (read or write) any FLASH control register other than the write to FSTAT (to clear FCBEF and launch the command) after writing the command to FCMD
- The MCU enters stop mode while a program or erase command is in progress (The command is aborted.)
- Writing the byte program, burst program, or page erase command code (\$20, \$25, or \$40) with a background debug command while the MCU is secured (The background debug controller can only do blank check and mass erase commands when the MCU is secure.)
- Writing 0 to FCBEF to cancel a partial command

4.4.6 FLASH Block Protection

Block protection prevents program or erase changes for FLASH memory locations in a designated address range. Mass erase is disabled when any block of FLASH is protected. The MC9S08RC/RD/RE/RG allows a block of memory at the end of FLASH, and/or the entire FLASH memory to be block protected. A disable control bit and a 3-bit control field, for each of the blocks, allows the user to independently set the size of these blocks. A separate control bit allows block protection of the entire FLASH memory array. All seven of these control bits are located in the FPROT register (see Section 4.6.4, “FLASH Protection Register (FPROT and NVPROT)”).

At reset, the high-page register (FPROT) is loaded with the contents of the NVPROT location that is in the nonvolatile register block of the FLASH memory. The value in FPROT cannot be changed directly from application software so a runaway program cannot alter the block protection settings. If the last 512 bytes of FLASH (which includes the NVPROT register) is protected, the application program cannot alter the block protection settings (intentionally or unintentionally). The FPROT control bits can be written by background debug commands to allow a way to erase a protected FLASH memory.

One use for block protection is to block protect an area of FLASH memory for a bootloader program. This bootloader program then can be used to erase the rest of the FLASH memory and reprogram it. Because the bootloader is protected, it remains intact even if MCU power is lost during an erase and reprogram operation.

The MC9S08RC/RD/RE/RG has these sources for reset:

- Power-on reset (POR)
- Low-voltage detect (LVD)
- Computer operating properly (COP) timer
- Illegal opcode detect
- Illegal address (16K and 8K devices only)
- Background debug forced reset
- The reset pin ($\overline{\text{RESET}}$)

Each of these sources, with the exception of the background debug forced reset, has an associated bit in the system reset status register. Whenever the MCU enters reset, the reset pin is driven low for 34 internal bus cycles where the internal bus frequency is one-half the OSC frequency. After the 34 cycles are completed, the pin is released and will be pulled up by the internal pullup resistor, unless it is held low externally. After the pin is released, it is sampled after another 38 cycles to determine whether the reset pin is the cause of the MCU reset.

5.4 Computer Operating Properly (COP) Watchdog

The COP watchdog is intended to force a system reset when the application software fails to execute as expected. To prevent a system reset from the COP timer (when it is enabled), application software must reset the COP timer periodically. If the application program gets lost and fails to reset the COP before it times out, a system reset is generated to force the system back to a known starting point. The COP watchdog is enabled by the COPE bit in SOPT (see Section 5.8.4, “System Options Register (SOPT),” for additional information). The COP timer is reset by writing any value to the address of SRS. This write does not affect the data in the read-only SRS. Instead, the act of writing to this address is decoded and sends a reset signal to the COP timer.

After any reset, the COP timer is enabled. This provides a reliable way to detect code that is not executing as intended. If the COP watchdog is not used in an application, it can be disabled by clearing the COPE bit in the write-once SOPT register. Also, the COPT bit can be used to choose one of two timeout periods (2^{18} or 2^{20} cycles of the bus rate clock). Even if the application will use the reset default settings in COPE and COPT, the user must write to write-once SOPT during reset initialization to lock in the settings. That way, they cannot be changed accidentally if the application program gets lost.

The write to SRS that services (clears) the COP timer must not be placed in an interrupt service routine (ISR) because the ISR could continue to be executed periodically even if the main application program fails.

When the MCU is in active background mode, the COP timer is temporarily disabled.

5.5 Interrupts

Interrupts provide a way to save the current CPU status and registers, execute an interrupt service routine (ISR), and then restore the CPU status so processing resumes where it was before the interrupt. Other than the software interrupt (SWI), which is a program instruction, interrupts are caused by hardware events such

6.4.2 Internal Pullup Control

An internal pullup device can be enabled for each port pin that is configured as an input ($PTxDDn = 0$). The pullup device is available for a peripheral module to use, provided the peripheral is enabled and is an input function as long as the $PTxDDn = 0$.

NOTE

The voltage measured on the pulled up PTA0 pin will be less than V_{DD} . The internal gates connected to this pin are pulled all the way to V_{DD} . All other pins with enabled pullup resistors will have an unloaded measurement of V_{DD} .

6.5 Stop Modes

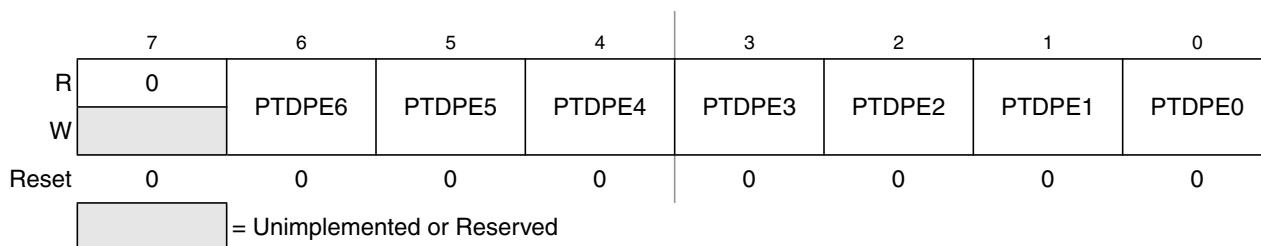
Depending on the stop mode, I/O functions differently as the result of executing a STOP instruction. An explanation of I/O behavior for the various stop modes follows:

- When the MCU enters stop1 mode, all internal registers, including general-purpose I/O control and data registers, are powered down. All of the general-purpose I/O pins assume their reset state: output buffers and pullups turned off. Upon exit from stop1, all I/O must be initialized as if the MCU had been reset.
- When the MCU enters stop2 mode, the internal registers are powered down as in stop1 but the I/O pin states are latched and held. For example, a port pin that is an output driving low continues to function as an output driving low even though its associated data direction and output data registers are powered down internally. Upon exit from stop2, the pins continue to hold their states until a 1 is written to the PPDACK bit. To avoid discontinuity in the pin state following exit from stop2, the user must restore the port control and data registers to the values they held before entering stop2. These values can be stored in RAM before entering stop2 because the RAM is maintained during stop2.
- In stop3 mode, all I/O is maintained because internal logic circuitry stays powered up. Upon recovery, normal I/O function is available to the user.

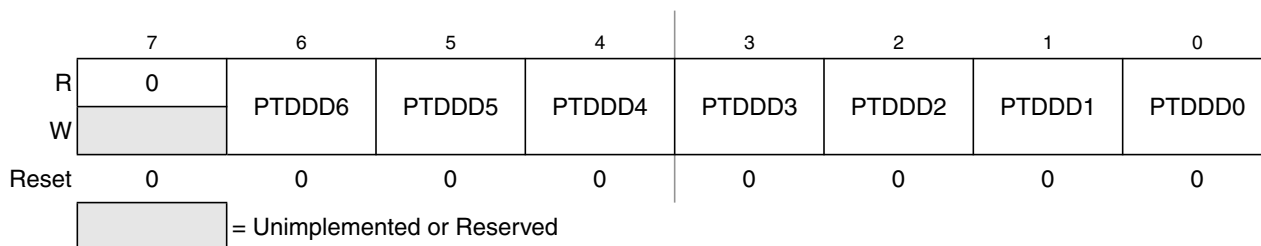
6.6 Parallel I/O Registers and Control Bits

This section provides information about all registers and control bits associated with the parallel I/O ports.

Refer to tables in the Memory chapter for the absolute address assignments for all parallel I/O registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.


Figure 6-16. Pullup Enable for Port D (PTDPE)
Table 6-11. PTDPE Field Descriptions

Field	Description
6:0 PTDPE[6:0]	Pullup Enable for Port D Bits — For port D pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled. For port D pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled. 0 Internal pullup device disabled. 1 Internal pullup device enabled.


Figure 6-17. Data Direction for Port D (PTDDD)
Table 6-12. PTDDD Field Descriptions

Field	Description
6:0 PTDDD[6:0]	Data Direction for Port D Bits — These read/write bits control the direction of port D pins and what is read for PTDD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port D bit n and PTDD reads return the contents of PTDDn.

7.2.3 Stack Pointer (SP)

This 16-bit address pointer register points at the next available location on the automatic last-in-first-out (LIFO) stack. The stack may be located anywhere in the 64-Kbyte address space that has RAM and can be any size up to the amount of available RAM. The stack is used to automatically save the return address for subroutine calls, the return address and CPU registers during interrupts, and for local variables. The AIS (add immediate to stack pointer) instruction adds an 8-bit signed immediate value to SP. This is most often used to allocate or deallocate space for local variables on the stack.

SP is forced to 0x00FF at reset for compatibility with the earlier M68HC05 Family. HCS08 programs normally change the value in SP to the address of the last location (highest address) in on-chip RAM during reset initialization to free up direct page RAM (from the end of the on-chip registers to 0x00FF).

The RSP (reset stack pointer) instruction was included for compatibility with the M68HC05 Family and is seldom used in new HCS08 programs because it only affects the low-order half of the stack pointer.

7.2.4 Program Counter (PC)

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

During normal program execution, the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, interrupt, and return operations load the program counter with an address other than that of the next sequential location. This is called a change-of-flow.

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

7.2.5 Condition Code Register (CCR)

The 8-bit condition code register contains the interrupt mask (I) and five flags that indicate the results of the instruction just executed. Bits 6 and 5 are set permanently to 1. The following paragraphs describe the functions of the condition code bits in general terms. For a more detailed explanation of how each instruction sets the CCR bits, refer to the *HCS08 Family Reference Manual, volume 1*, Freescale Semiconductor document order number HCS08RMv1/D.

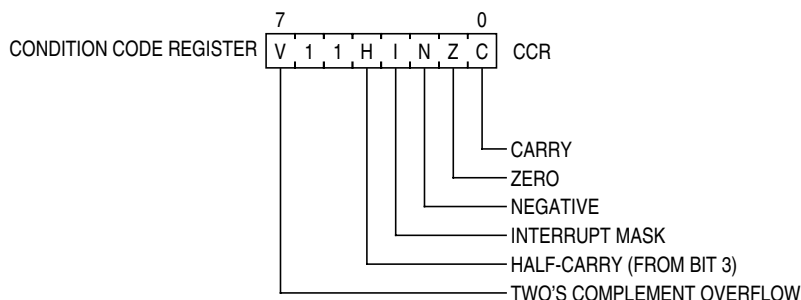


Figure 7-2. Condition Code Register

7.5 HCS08 Instruction Set Summary

Instruction Set Summary Nomenclature

The nomenclature listed here is used in the instruction descriptions in Table 7-2.

Operators

- () = Contents of register or memory location shown inside parentheses
- ← = Is loaded with (read: “gets”)
- & = Boolean AND
- | = Boolean OR
- ⊕ = Boolean exclusive-OR
- × = Multiply
- ÷ = Divide
- :
- + = Add
- = Negate (two’s complement)

CPU registers

- A = Accumulator
- CCR = Condition code register
- H = Index register, higher order (most significant) 8 bits
- X = Index register, lower order (least significant) 8 bits
- PC = Program counter
- PCH = Program counter, higher order (most significant) 8 bits
- PCL = Program counter, lower order (least significant) 8 bits
- SP = Stack pointer

Memory and addressing

- M = A memory location or absolute data, depending on addressing mode
- M:M + 0x0001 = A 16-bit value in two consecutive memory locations. The higher-order (most significant) 8 bits are located at the address of M, and the lower-order (least significant) 8 bits are located at the next higher sequential address.

Condition code register (CCR) bits

- V = Two’s complement overflow indicator, bit 7
- H = Half carry, bit 4
- I = Interrupt mask, bit 3
- N = Negative indicator, bit 2
- Z = Zero indicator, bit 1
- C = Carry/borrow, bit 0 (carry out of bit 7)

CCR activity notation

- = Bit not affected



Chapter 9

Keyboard Interrupt (S08KBIV1)

9.1 Introduction

The MC9S08RC/RD/RE/RG has two KBI modules. One has eight keyboard interrupt inputs that share port A pins. The other KBI module has four inputs that are shared on the upper four pins of port C. See the Pins and Connections chapter for more information about the logic and hardware aspects of these pins.

Port A is an 8-bit port that is shared between the KBI1 keyboard interrupt inputs and general-purpose I/O. The eight KBI1PEn control bits in the KBI1PE register allow selection of any combination of port A pins to be assigned as KBI1 inputs. Any pins that are enabled as KBI1 inputs will be forced to act as inputs and the remaining port A pins are available as general-purpose I/O pins controlled by the port A data (PTAD), data direction (PTADD) and pullup enable (PTAPE) registers. The eight PTAPEn control bits in the PTAPE register allow the user to select whether an internal pullup device is enabled on each port A pin that is configured as a port input or a KBI1 input.

KBI1 inputs can be configured for edge-only sensitivity or edge-and-level sensitivity. Bits 3 through 0 of port A are falling-edge/low-level sensitive and bits 7 through 4 can be configured for rising-edge/high-level or for falling-edge/low-level sensitivity.

Port C is an 8-bit port with its lower four pins shared between the KBI2 keyboard interrupt inputs and general-purpose I/O. The four KBI2PEn control bits in the KBI2PE register allow selection of any combination of the lower four port C pins to be assigned as KBI2 inputs. Any pins that are enabled as KBI2 inputs will be forced to act as inputs and the remaining port C pins are available as general-purpose I/O pins controlled by the port C data (PTCD), data direction (PTCDD) and pullup enable (PTCPE) registers. The eight PTCPEn control bits in the PTCPE register allow the user to select whether an internal pullup device is enabled on each port C pin that is configured as a port input or a KBI2 input.

Any enabled keyboard interrupt can be used to wake the MCU from wait, standby (stop3), partial power-down (stop2) or power-down modes (stop1). In either stop1 or stop2 mode, an input functions as a falling edge/low-level wakeup, therefore it should be configured to use falling-edge sensing if the MCU will be used in stop1 or stop2 modes.

Either KBI1 or KBI2 can be used to wake the MCU from wait or standby (stop3). Only KBI1 can be used to wake the MCU from partial power down (stop2) or power down (stop1). When using KBI1 to wake up from stop2 or stop1, the pins must be configured to use falling-edge/low-level sensing (KBEDG = 0). The KBF bits for both KBI modules must be cleared before entering stop mode, regardless of whether the interrupt is enabled.

NOTE

The voltage measured on the pulled up PTA0 pin will be less than V_{DD} . The internal gates connected to this pin are pulled all the way to V_{DD} . All other pins with enabled pullup resistors will have an unloaded measurement of V_{DD} .

A rising edge is detected when the input signal is seen as a logic 0 during one bus cycle and then a logic 1 during the next cycle.

The KBIMOD control bit can be set to reconfigure the detection logic so that it detects edges and levels. In KBIMOD = 1 mode, the KBF status flag becomes set when an edge is detected (when one or more enabled pins change from the deasserted to the asserted level while all other enabled pins remain at their deasserted levels), but the flag is continuously set (and cannot be cleared) as long as any enabled keyboard input pin remains at the asserted level. When the MCU enters stop mode, the synchronous edge-detection logic is bypassed (because clocks are stopped). In stop mode, KBI inputs act as asynchronous level-sensitive inputs so they can wake the MCU from stop mode.

9.3.3 KBI Interrupt Controls

The KBF status flag becomes set (1) when an edge event has been detected on any KBI input pin. If KBIE = 1 in the KBIxSC register, a hardware interrupt will be requested whenever KBF = 1. The KBF flag is cleared by writing a 1 to the keyboard acknowledge (KBACK) bit.

When KBIMOD = 0 (selecting edge-only operation), KBF is always cleared by writing 1 to KBACK. When KBIMOD = 1 (selecting edge-and-level operation), KBF cannot be cleared as long as any keyboard input is at its asserted level.

9.4 KBI Registers and Control Bits

This section provides information about all registers and control bits associated with the KBI modules.

Refer to the direct-page register summary in the Memory chapter of this data sheet for the absolute address assignments for all KBI registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

Some MCU systems have more than one KBI, so register names include placeholder characters to identify which KBI is being referenced. For example, KBIxSC refers to the KBIx status and control register and KBI2SC is the status and control register for KBI2.

Timer/PWM (TPM)

In input capture mode, reading either byte (TPM1CnVH or TPM1CnVL) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. This latching mechanism also resets (becomes unlatched) when the TPM1CnSC register is written.

In output compare or PWM modes, writing to either byte (TPM1CnVH or TPM1CnVL) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the timer channel value registers. This latching mechanism may be manually reset by writing to the TPM1CnSC register.

This latching mechanism allows coherent 16-bit writes in either order, which is friendly to various compiler implementations.

12.1.3 Block Diagram

Figure 12-1 shows the transmitter portion of the SCI. (Figure 12-2 shows the receiver portion of the SCI.)

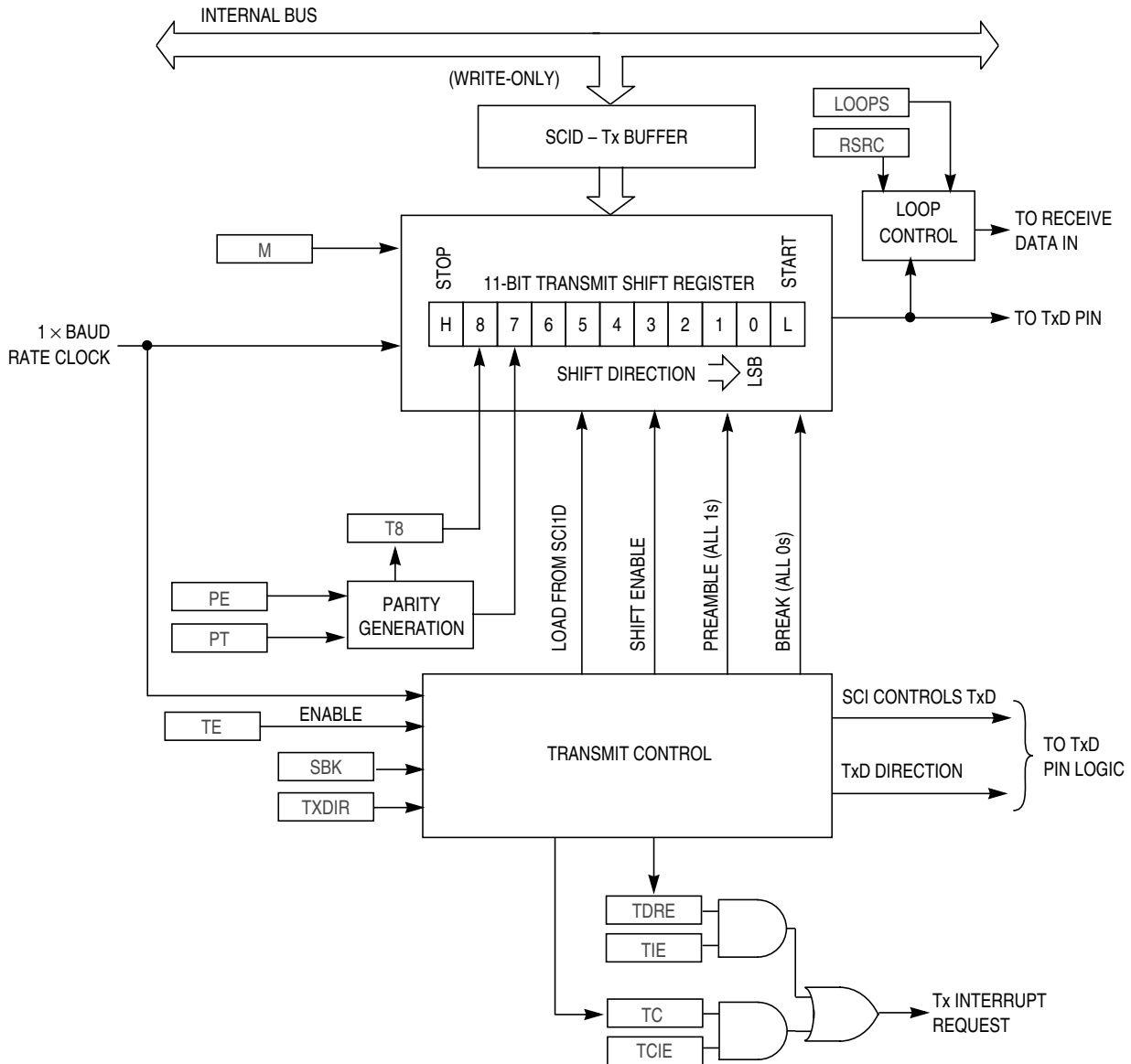


Figure 12-1. SCI Transmitter Block Diagram

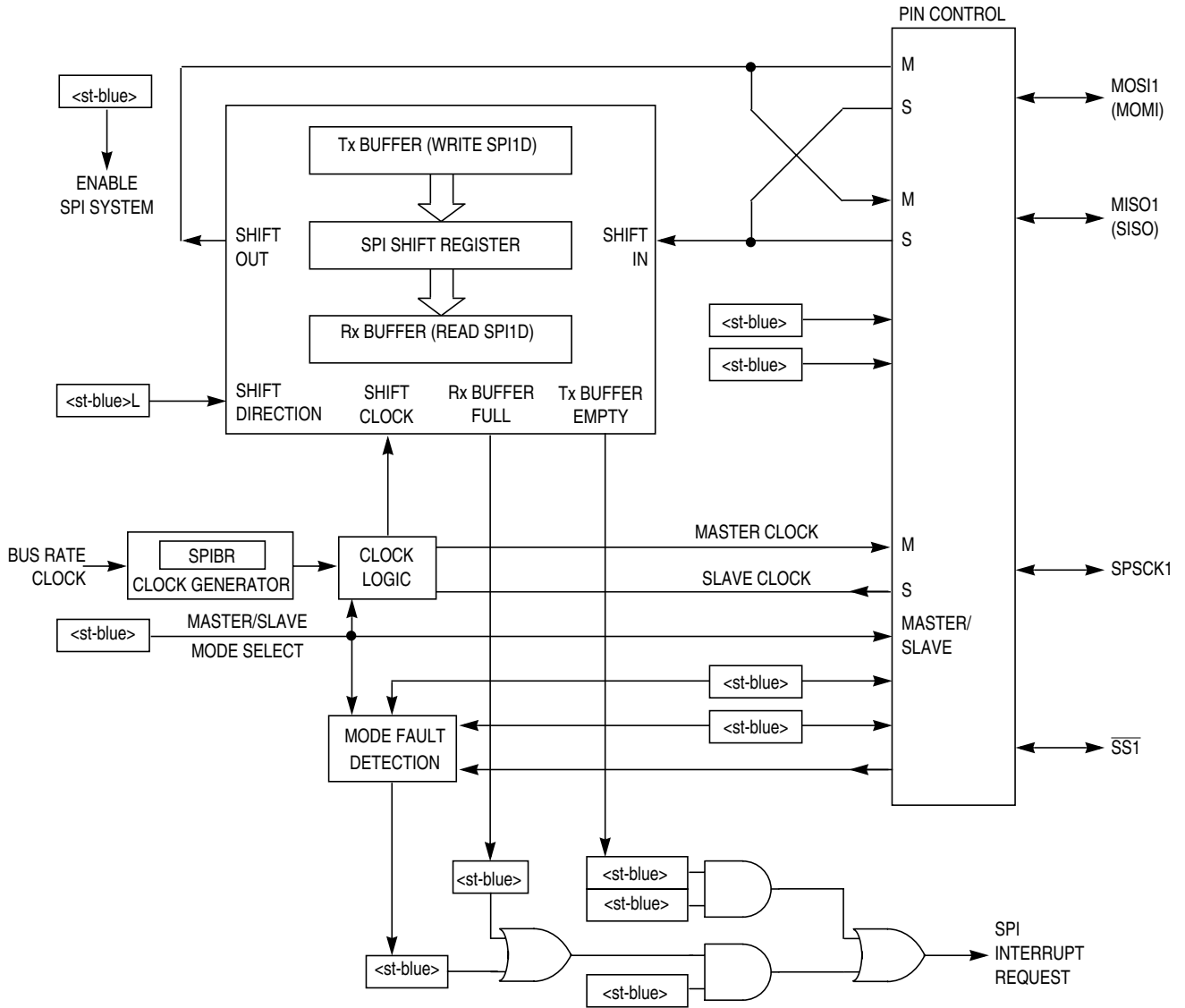


Figure 13-3. SPI Module Block Diagram

14.5 ACMP Status and Control Register (ACMP1SC)

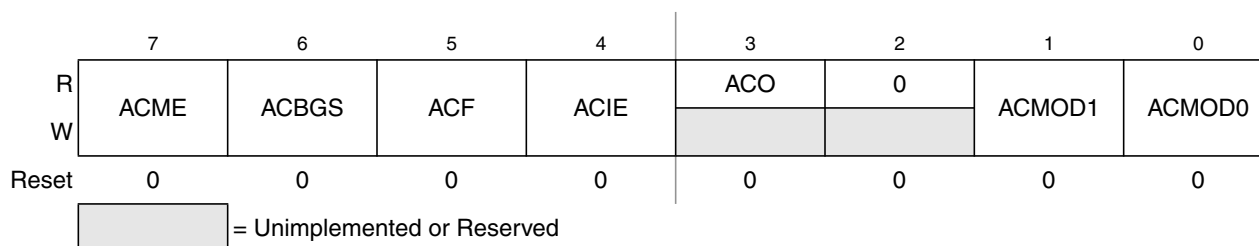


Figure 14-3. ACMP Status and Control Register (ACMP1SC)

Table 14-1. ACMP1SC Field Descriptions

Field	Description
7 ACME	Analog Comparator Module Enable — The ACME bit enables the analog comparator module. When the module is not enabled, it remains in a low-power state. 0 Analog comparator disabled 1 Analog comparator enabled
6 ACBGS	Analog Comparator Bandgap Select — The ACBGS bit is used to select the internal bandgap as the comparator reference. 0 External pin ACMP1+ selected as comparator non-inverting input 1 Internal bandgap reference selected as comparator non-inverting input
5 ACF	Analog Comparator Flag — The ACF bit is set when a compare event occurs. Compare events are defined by the ACMOD0 and ACMOD1 bits. The ACF bit is cleared by writing a 1 to the bit. 0 Compare event has not occurred. 1 Compare event has occurred.
4 ACIE	Analog Comparator Interrupt Enable — The ACIE bit enables the interrupt from the ACMP. When this bit is set, an interrupt will be asserted when the ACF bit is set. 0 Interrupt disabled 1 Interrupt enabled
3 ACO	Analog Comparator Output — Reading the ACO bit will return the current value of the analog comparator output. The register bit is reset to 0 and will read as 0 when the analog comparator module is disabled (ACME = 0).
1:0 ACMOD[1:0]	Analog Comparator Modes — The ACMOD1 and ACMOD0 bits select the flag setting mode that controls the type of compare event that sets the ACF bit. 00 Comparator output falling edge 01 Comparator output rising edge 10 Comparator output falling edge 11 Comparator output either rising or falling edge

Figure 15-4 shows the host receiving a logic 0 from the target HCS08 MCU. Because the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target MCU. The host initiates the bit time but the target HCS08 finishes it. Because the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 BDC clock cycles, then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 cycles after starting the bit time.

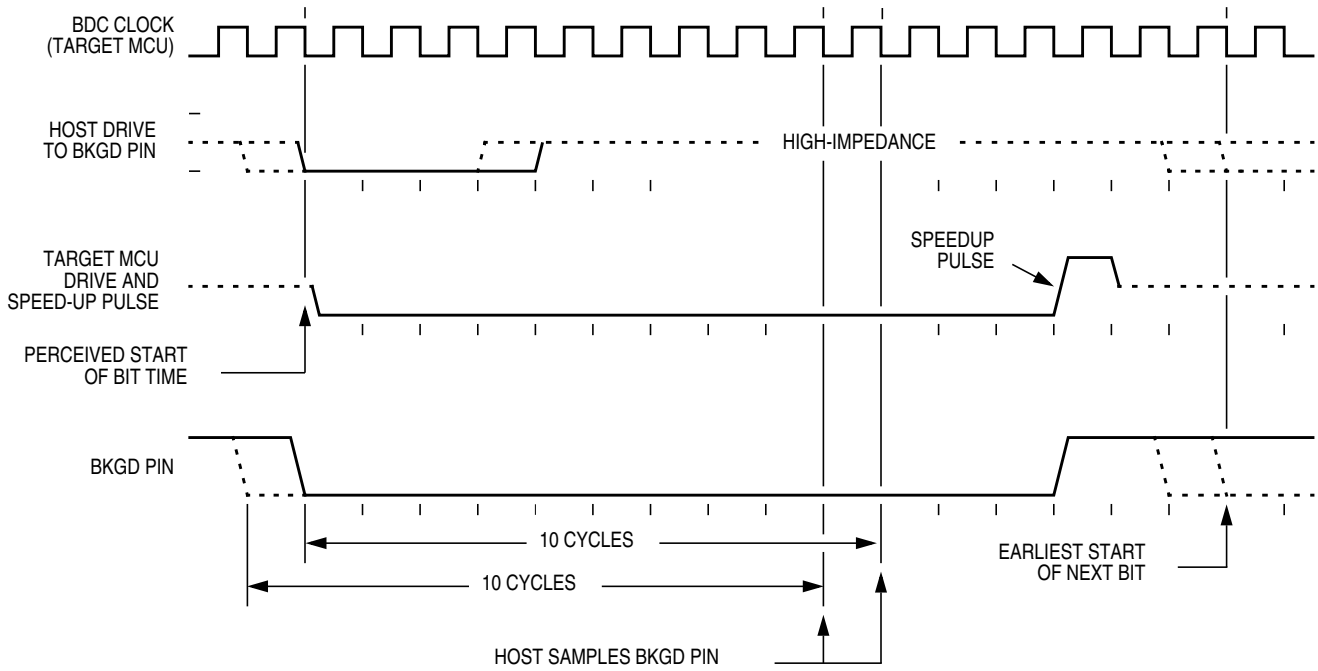


Figure 15-4. BDM Target-to-Host Serial Bit Timing (Logic 0)

15.4.1.1 BDC Status and Control Register (BDCSCR)

This register can be read or written by serial BDC commands (READ_STATUS and WRITE_CONTROL) but is not accessible to user programs because it is not located in the normal memory map of the MCU.

	7	6	5	4	3	2	1	0
R	ENBDM	BDMACT	BKPTEN	FTS	CLKSW	WS	WSF	DVF
W								
Normal Reset	0	0	0	0	0	0	0	0
Reset in Active BDM:	1	1	0	0	1	0	0	0

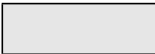
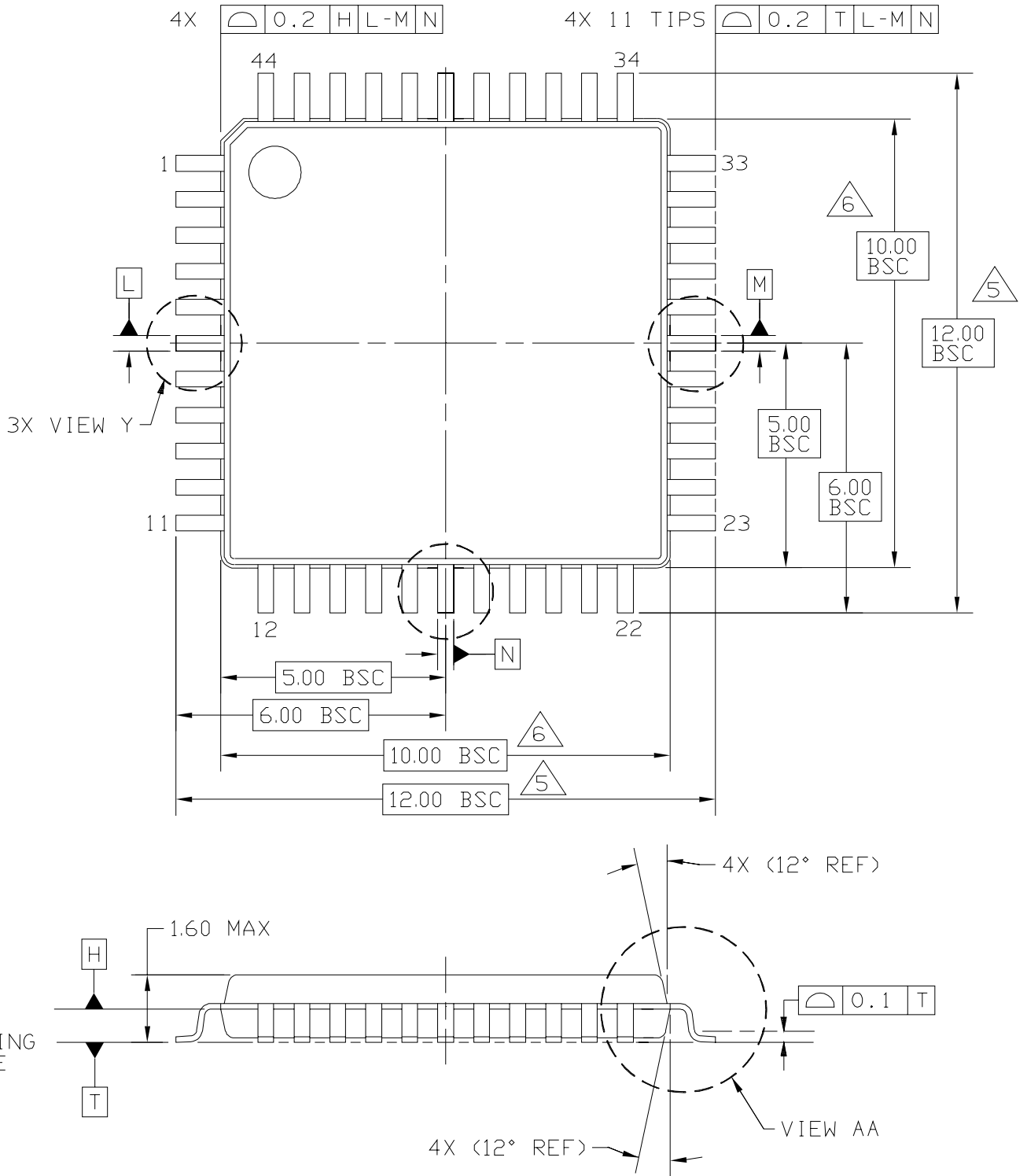
 = Unimplemented or Reserved

Figure 15-5. BDC Status and Control Register (BDCSCR)

Table 15-2. BDCSCR Register Field Descriptions

Field	Description
7 ENBDM	Enable BDM (Permit Active Background Mode) — Typically, this bit is written to 1 by the debug host shortly after the beginning of a debug session or whenever the debug host resets the target and remains 1 until a normal reset clears it. 0 BDM cannot be made active (non-intrusive commands still allowed) 1 BDM can be made active to allow active background mode commands
6 BDMACT	Background Mode Active Status — This is a read-only status bit. 0 BDM not active (user application program running) 1 BDM active and waiting for serial commands
5 BKPTEN	BDC Breakpoint Enable — If this bit is clear, the BDC breakpoint is disabled and the FTS (force tag select) control bit and BDCBKPT match register are ignored. 0 BDC breakpoint disabled 1 BDC breakpoint enabled
4 FTS	Force/Tag Select — When FTS = 1, a breakpoint is requested whenever the CPU address bus matches the BDCBKPT match register. When FTS = 0, a match between the CPU address bus and the BDCBKPT register causes the fetched opcode to be tagged. If this tagged opcode ever reaches the end of the instruction queue, the CPU enters active background mode rather than executing the tagged opcode. 0 Tag opcode at breakpoint address and enter active background mode if CPU attempts to execute that instruction 1 Breakpoint match forces active background mode at next instruction boundary (address need not be an opcode)
3 CLKSW	Select Source for BDC Communications Clock — CLKSW defaults to 0, which selects the alternate BDC clock source. 0 Alternate BDC clock source 1 MCU bus clock



TITLE:
 44 LD TQFP,
 10 X 10 PKG, 0.8 PITCH, 1.4 THICK

CASE NUMBER: 824D-03

STANDARD: JEDEC MS-026-BCB

PACKAGE CODE: 8256

SHEET: 1 OF 3



© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
 ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED
 DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS
 ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.

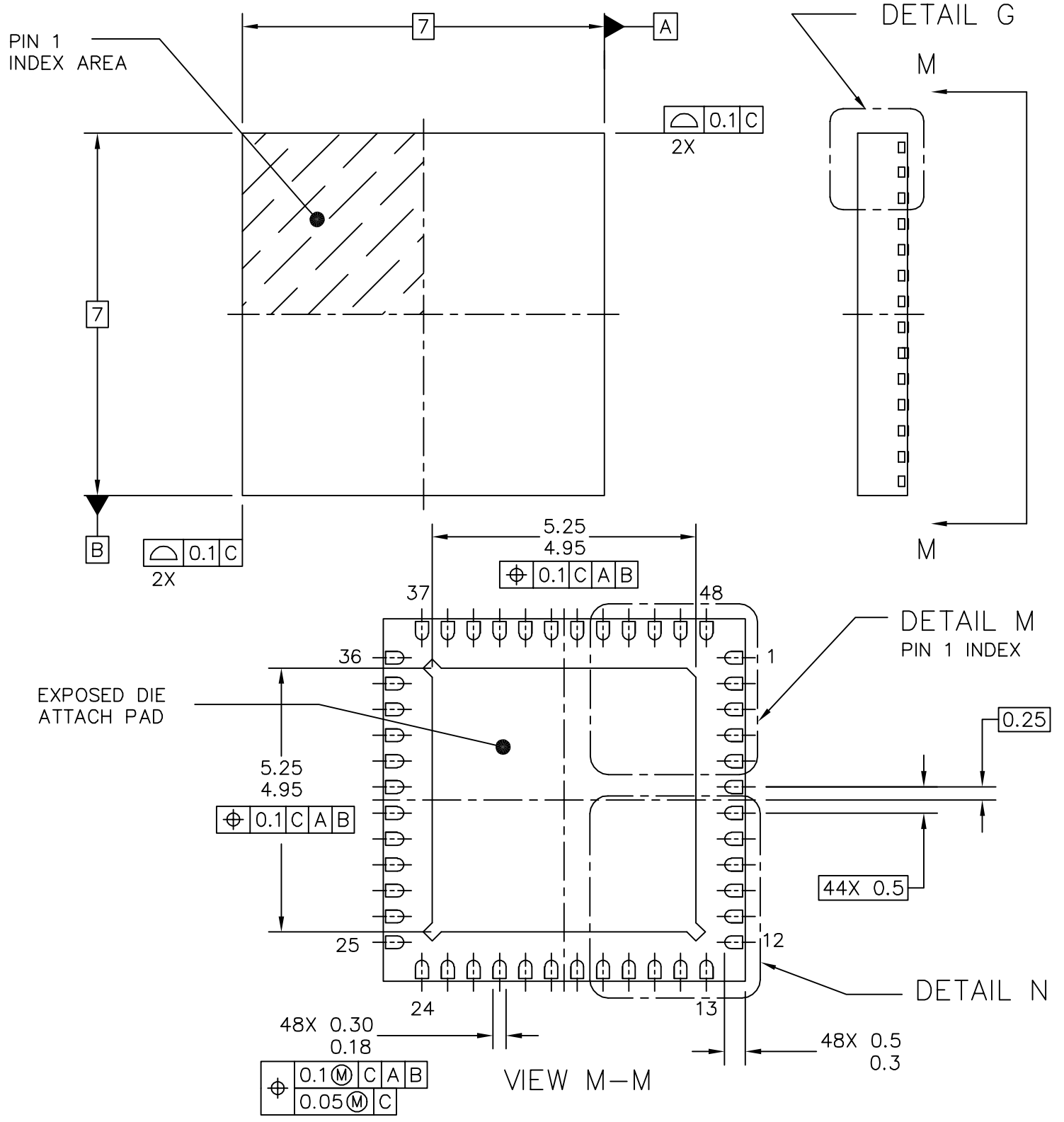
**MECHANICAL OUTLINES
 DICTIONARY**

DOCUMENT NO: 98ARH99048A

PAGE: 1314

DO NOT SCALE THIS DRAWING

REV: D



**TITLE: THERMALLY ENHANCED QUAD
 FLAT NON-LEADED PACKAGE (QFN)
 48 TERMINAL, 0.5 PITCH (7 X 7 X 1)**

CASE NUMBER: 1314-03

STANDARD: JEDEC-MO-220 VKKD-2

PACKAGE CODE: 6152 **SHEET: 1 OF 5**