

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.

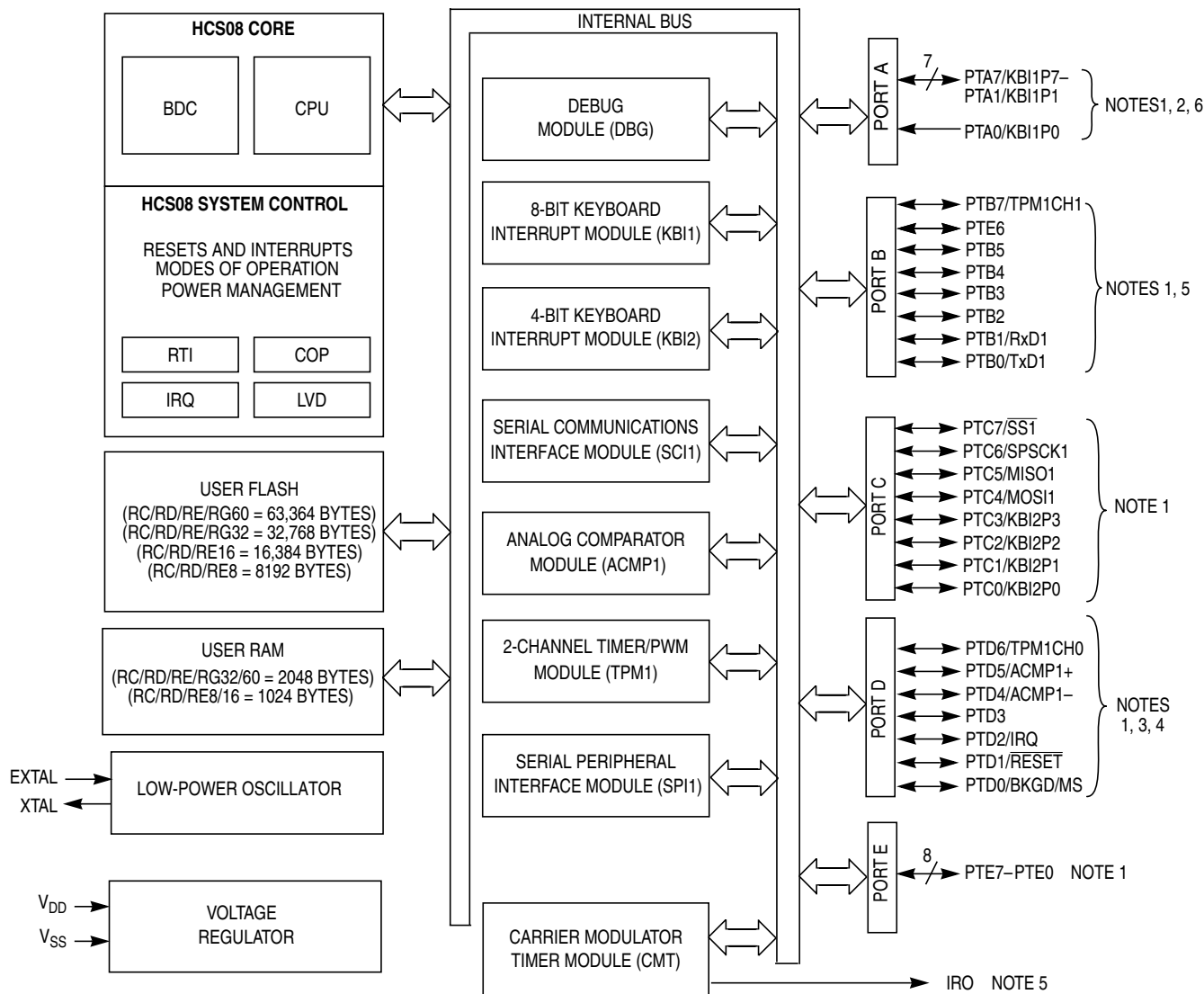
Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Active
Core Processor	S08
Core Size	8-Bit
Speed	8MHz
Connectivity	SCI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	23
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-PDIP
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc9s08rd16pe

1.3 MCU Block Diagram

This block diagram shows the structure of the MC9S08RC/RD/RE/RG MCUs



NOTES:

1. Port pins are software configurable with pullup device if input port
2. PTA0 does not have a clamp diode to V_{DD}. PTA0 should not be driven above V_{DD}. Also, PTA0 does not pullup to V_{DD} when internal pullup is enabled.
3. IRQ pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1)
4. The RESET pin contains integrated pullup device enabled if reset enabled (RSTPE = 1)
5. High current drive
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown enabled when KBI is enabled (KBIPE_n = 1) and rising edge is selected (KBEDG_n = 1).

Figure 1-1. MC9S08RC/RD/RE/RG Block Diagram

Table 1-2 lists the functional versions of the on-chip modules.

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

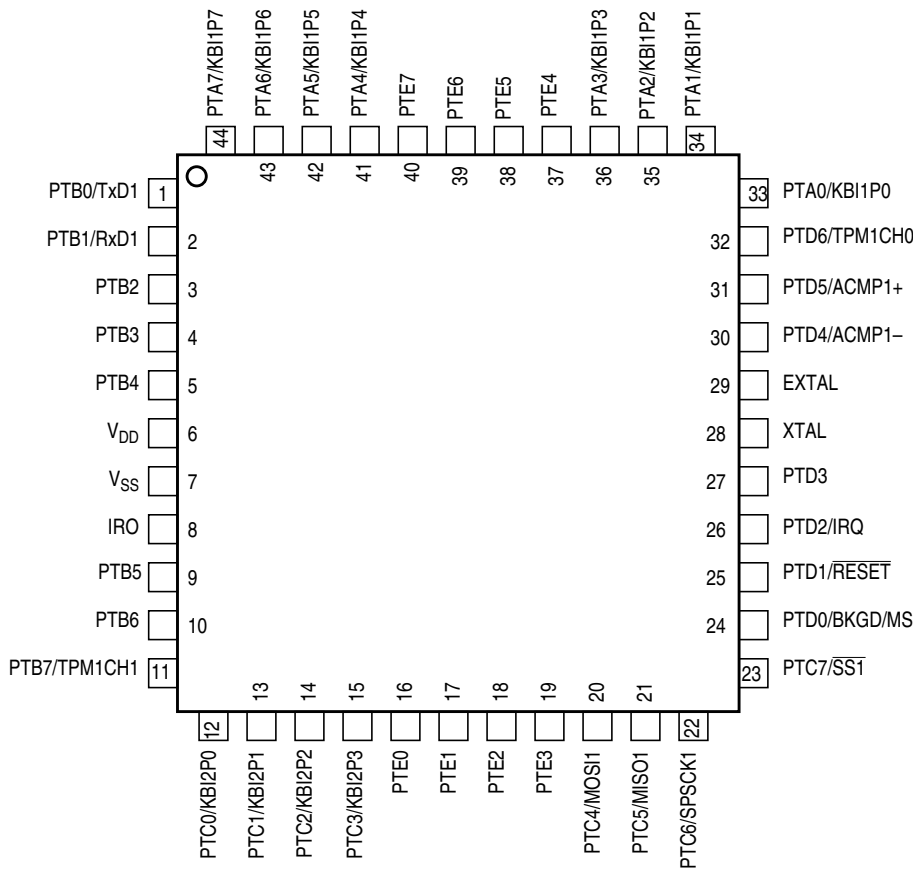


Figure 2-1. MC9S08RC/RD/RE/RG in 44-Pin LQFP Package

Table 2-1. Pin Sharing References

Port Pins	Alternate Function	Reference ⁽¹⁾
PTA7–PTA0	KBI1P7–KBI1P0	Chapter 9, “Keyboard Interrupt (S08KBIV1)”
PTB7	TPM1CH1	Chapter 10, “Timer/PWM Module (S08TPMV1)”
PTB6–PTB2	—	Chapter 6, “Parallel Input/Output”
PTB1 PTB0	RxD1 TxD1	Chapter 11, “Serial Communications Interface (S08SCIV1)”
PTC7 PTC6 PTC5 PTC4	SS1 SPSCK1 MISO1 MOSI1	Chapter 13, “Serial Peripheral Interface (S08SPIV3)”
PTC3–PTC0	KBI2P3–KBI2P0	Chapter 9, “Keyboard Interrupt (S08KBIV1)”
PTD6	TPM1CH0	Chapter 10, “Timer/PWM Module (S08TPMV1)”
PTD5 PTD4	ACMP1+ ACMP1–	Chapter 14, “Analog Comparator (S08ACMPV1)”
PTD2	IRQ	Chapter 5, “Resets, Interrupts, and System Configuration”
PTD1	RESET	
PTD0	BKGD/MS	
PTE7–PTE0	—	Chapter 6, “Parallel Input/Output”

1. See this chapter for information about modules that share these pins.

When an on-chip peripheral system is controlling a pin, data direction control bits still determine what is read from port data registers even though the peripheral module controls the pin direction by controlling the enable for the pin’s output buffer. See the Chapter 6, “Parallel Input/Output,” for more details.

Pullup enable bits for each input pin control whether on-chip pullup devices are enabled whenever the pin is acting as an input even if it is being controlled by an on-chip peripheral module. When the PTA7–PTA4 pins are controlled by the KBI module and are configured for rising-edge/high-level sensitivity, the pullup enable control bits enable pulldown devices rather than pullup devices. Similarly, when PTD2 is configured as the IRQ input and is set to detect rising edges, the pullup enable control bit enables a pulldown device rather than a pullup device.

2.3.7 Signal Properties Summary

Table 2-2 summarizes I/O pin characteristics. These characteristics are determined by the way the common pin interfaces are hardwired to internal circuits.

Chapter 3

Modes of Operation

3.1 Introduction

The operating modes of the MC9S08RC/RD/RE/RG are described in this section. Entry into each mode, exit from each mode, and functionality while in each of the modes are described.

3.2 Features

- Active background mode for code development
- Wait mode:
 - CPU shuts down to conserve power
 - System clocks running
 - Full voltage regulation maintained
- Stop modes:
 - System clocks stopped; voltage regulator in standby
 - Stop1 — Full power down of internal circuits for maximum power savings
 - Stop2 — Partial power down of internal circuits, RAM remains operational
 - Stop3 — All internal circuits powered for fast recovery

3.3 Run Mode

This is the normal operating mode for the MC9S08RC/RD/RE/RG. This mode is selected when the BKGD/MS pin is high at the rising edge of reset. In this mode, the CPU executes code from internal memory with execution beginning at the address fetched from memory at \$FFFE:\$FFFF after reset.

3.4 Active Background Mode

The active background mode functions are managed through the background debug controller (BDC) in the HCS08 core. The BDC, together with the on-chip debug module (DBG), provide the means for analyzing MCU operation during software development.

Active background mode is entered in any of five ways:

- When the BKGD/MS pin is low at the rising edge of reset
- When a BACKGROUND command is received through the BKGD pin
- When a BGND instruction is executed
- When encountering a BDC breakpoint
- When encountering a DBG breakpoint

After active background mode is entered, the CPU is held in a suspended state waiting for serial background commands rather than executing instructions from the user's application program.

Background commands are of two types:

- Non-intrusive commands, defined as commands that can be issued while the user program is running. Non-intrusive commands can be issued through the BKGD pin while the MCU is in run mode; non-intrusive commands can also be executed when the MCU is in the active background mode. Non-intrusive commands include:
 - Memory access commands
 - Memory-access-with-status commands
 - BDC register access commands
 - BACKGROUND command
- Active background commands, which can only be executed while the MCU is in active background mode, include commands to:
 - Read or write CPU registers
 - Trace one user program instruction at a time
 - Leave active background mode to return to the user's application program (GO)

The active background mode is used to program a bootloader or user application program into the FLASH program memory before the MCU is operated in run mode for the first time. When the MC9S08RC/RD/RE/RG is shipped from the Freescale Semiconductor factory, the FLASH program memory is usually erased so there is no program that could be executed in run mode until the FLASH memory is initially programmed. The active background mode can also be used to erase and reprogram the FLASH memory after it has been previously programmed.

For additional information about the active background mode, refer to the Development Support chapter.

3.5 Wait Mode

Wait mode is entered by executing a WAIT instruction. Upon execution of the WAIT instruction, the CPU enters a low-power state in which it is not clocked. The I bit in CCR is cleared when the CPU enters the wait mode, enabling interrupts. When an interrupt request occurs, the CPU exits the wait mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

Only the BACKGROUND command and memory-access-with-status commands are available when the MCU is in wait 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 wait mode and enter active background mode.

4.4.1 Features

Features of the FLASH memory include:

- FLASH Size
 - MC9S08RC/RD/RE/RG60 — 63374 bytes (124 pages of 512 bytes each)
 - MC9S08RC/RD/RE/RG32 — 32768 bytes (64 pages of 512 bytes each)
 - MC9S08RC/RD/RE16 — 16384 bytes (32 pages of 512 bytes each)
 - MC9S08RC/RD/RE8 — 8192 bytes (16 pages of 512 bytes each)
- Single power supply program and erase
- Command interface for fast program and erase operation
- Up to 100,000 program/erase cycles at typical voltage and temperature
- Flexible block protection
- Security feature for FLASH and RAM
- Auto power-down for low-frequency read accesses

4.4.2 Program and Erase Times

Before any program or erase command can be accepted, the FLASH clock divider register (FCDIV) must be written to set the internal clock for the FLASH module to a frequency (f_{FCLK}) between 150 kHz and 200 kHz (see Section 4.6.1, “FLASH Clock Divider Register (FCDIV)”). This register can be written only once, so normally this write is done during reset initialization. FCDIV cannot be written if the access error flag, FACCERR in FSTAT, is set. The user must ensure that FACCERR is not set before writing to the FCDIV register. One period of the resulting clock ($1/f_{FCLK}$) is used by the command processor to time program and erase pulses. An integer number of these timing pulses are used by the command processor to complete a program or erase command.

Table 4-4 shows program and erase times. The bus clock frequency and FCDIV determine the frequency of FCLK (f_{FCLK}). The time for one cycle of FCLK is $t_{FCLK} = 1/f_{FCLK}$. The times are shown as a number of cycles of FCLK and as an absolute time for the case where $t_{FCLK} = 5 \mu s$. Program and erase times shown include overhead for the command state machine and enabling and disabling of program and erase voltages.

Table 4-4. Program and Erase Times

Parameter	Cycles of FCLK	Time if FCLK = 200 kHz
Byte program	9	45 μs
Byte program (burst)	4	20 μs ⁽¹⁾
Page erase	4000	20 ms
Mass erase	20,000	100 ms

1. Excluding start/end overhead

2. Writing the user-entered key values to the NVBACKKEY through NVBACKKEY+7 locations. These writes must be done in order starting with the value for NVBACKKEY and ending with NVBACKKEY+7. STHX must not be used for these writes because these writes cannot be done on adjacent bus cycles. User software normally would get the key codes from outside the MCU system through a communication interface such as a serial I/O.
3. Writing 0 to KEYACC in the FCNFG register. If the 8-byte key that was just written matches the key stored in the FLASH locations, SEC01:SEC00 are automatically changed to 1:0 and security will be disengaged until the next reset.

The security key can be written only from RAM, so it cannot be entered through background commands without the cooperation of a secure user program. The FLASH memory cannot be accessed by read operations while KEYACC is set.

The backdoor comparison key (NVBACKKEY through NVBACKKEY+7) is located in FLASH memory locations in the nonvolatile register space so users can program these locations exactly as they would program any other FLASH memory location. The nonvolatile registers are in the same 512-byte block of FLASH as the reset and interrupt vectors, so block protecting that space also block protects the backdoor comparison key. Block protects cannot be changed from user application programs, so if the vector space is block protected, the backdoor security key mechanism cannot permanently change the block protect, security settings, or the backdoor key.

Security can always be disengaged through the background debug interface by performing these steps:

1. Disable any block protections by writing FPROT. FPROT can be written only with background debug commands, not from application software.
2. Mass erase FLASH if necessary.
3. Blank check FLASH. Provided FLASH is completely erased, security is disengaged until the next reset.

To avoid returning to secure mode after the next reset, program NVOPT so SEC01:SEC00 = 1:0.

4.6 FLASH Registers and Control Bits

The FLASH module has nine 8-bit registers in the high-page register space, three locations in the nonvolatile register space in FLASH memory that are copied into three corresponding high-page control registers at reset. There is also an 8-byte comparison key in FLASH memory. Refer to Table 4-2 and Table 4-3 for the absolute address assignments for all FLASH 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.

4.6.1 FLASH Clock Divider Register (FCDIV)

Bit 7 of this register is a read-only status flag. Bits 6 through 0 may be read at any time but can be written only one time. Before any erase or programming operations are possible, write to this register to set the frequency of the clock for the nonvolatile memory system within acceptable limits.

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

5.8.4 System Options Register (SOPT)

This register may be read at any time. Bits 3 and 2 are unimplemented and always read 0. This is a write-once register so only the first write after reset is honored. Any subsequent attempt to write to SOPT (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT must be written during the user's reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

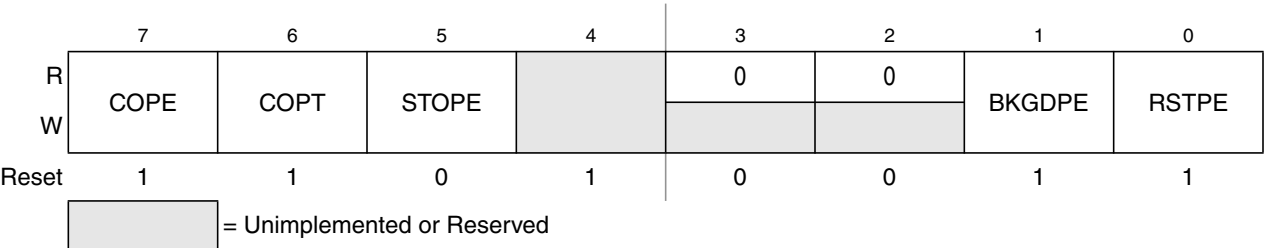


Figure 5-5. System Options Register (SOPT)

Table 5-5. SOPT Field Descriptions

Field	Description
7 COPE	COP Watchdog Enable — This write-once bit defaults to 1 after reset. 0 COP watchdog timer disabled. 1 COP watchdog timer enabled (force reset on timeout).
6 COPT	COP Watchdog Timeout — This write-once bit defaults to 1 after reset. 0 Short timeout period selected (2^{18} cycles of BUSCLK). 1 Long timeout period selected (2^{20} cycles of BUSCLK).
5 STOPE	Stop Mode Enable — This write-once bit defaults to 0 after reset, which disables stop mode. If stop mode is disabled and a user program attempts to execute a STOP instruction, an illegal opcode reset is forced. 0 Stop mode disabled. 1 Stop mode enabled.
1 BKGDPE	Background Debug Mode Pin Enable — The BKGDPE bit enables the PTD0/BKGD/MS pin to function as BKGD/MS. When the bit is clear, the pin will function as PTD0, which is an output only general purpose I/O. This pin always defaults to BKGD/MS function after any reset. 0 BKGD pin disabled. 1 BKGD pin enabled.
0 RSTPE	RESET Pin Enable — The RSTPE bit enables the PTD1/RESET pin to function as RESET. When the bit is clear, the pin will function as PTD1, which is an output only general purpose I/O. This pin always defaults to RESET function after any reset. 0 RESET pin disabled. 1 RESET pin enabled.

5.8.5 System Device Identification Register (SDIDH, SDIDL)

This read-only register is included so host development systems can identify the HCS08 derivative and revision number. This allows the development software to recognize where specific memory blocks, registers, and control bits are located in a target MCU.

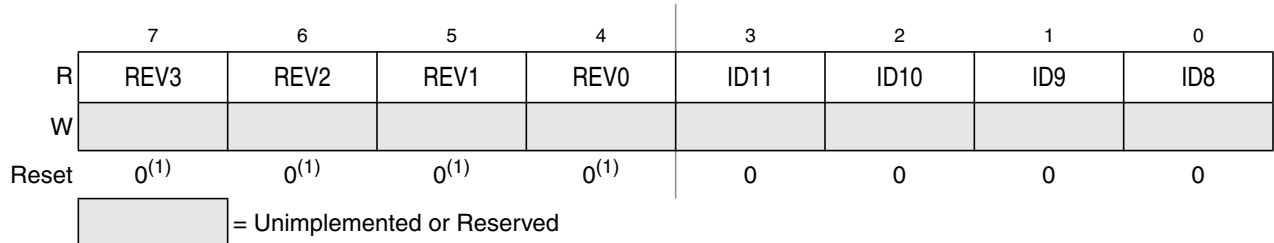


Figure 5-6. System Device Identification Register — High (SDIDH)

1. The revision number that is hard coded into these bits reflects the current silicon revision level.

Table 5-6. SDIDH Field Descriptions

Field	Description
7:4 REV[3:0]	Revision Number — The high-order 4 bits of address \$1806 are hard coded to reflect the current mask set revision number (0–F).
3:0 ID[11:8]	Part Identification Number — Each derivative in the HCS08 Family has a unique identification number. The MC9S08RC/RD/RE/RG32/60 is hard coded to the value \$004 and the MC9S08RC/RD/RE8/16 is hard coded to the value \$003.

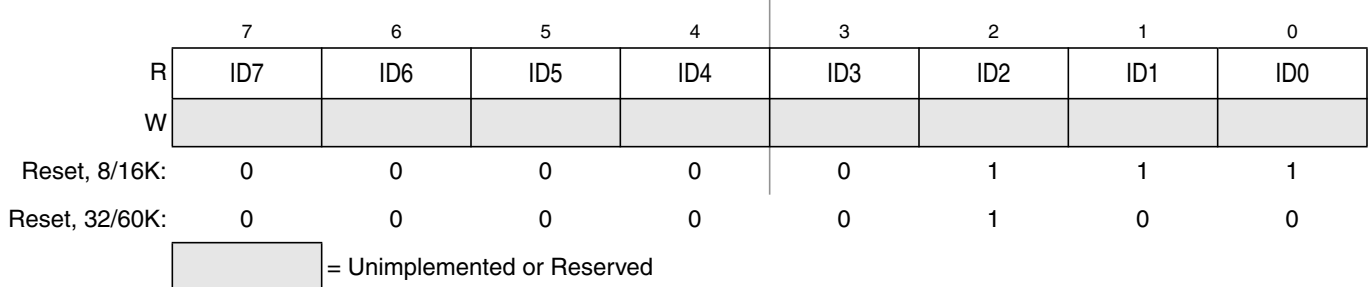
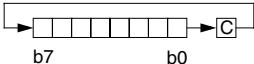


Figure 5-7. System Device Identification Register — Low (SDIDL)

Table 5-7. SDIDL Field Descriptions

Field	Description
7:0 ID[7:0]	Part Identification Number — Each derivative in the HCS08 Family has a unique identification number. The MC9S08RC/RD/RE/RG32/60 is hard coded to the value \$004 and the MC9S08RC/RD/RE8/16 is hard coded to the value \$003.

Table 7-2. HCS08 Instruction Set Summary (Sheet 6 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles ¹
			V	H	I	N	Z	C				
ROR <i>opr8a</i> RORA RORX ROR <i>opr8,X</i> ROR <i>,X</i> ROR <i>opr8,SP</i>	Rotate Right through Carry			–	–				DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	5 1 1 5 4 6
RSP	Reset Stack Pointer	SP ← 0xFF (High Byte Not Affected)	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 0x0001; Pull (CCR) SP ← (SP) + 0x0001; Pull (A) SP ← (SP) + 0x0001; Pull (X) SP ← (SP) + 0x0001; Pull (PCH) SP ← (SP) + 0x0001; Pull (PCL)							INH	80		9
RTS	Return from Subroutine	SP ← SP + 0x0001; Pull (PCH) SP ← SP + 0x0001; Pull (PCL)	–	–	–	–	–	–	INH	81		6
SBC <i>#opr8i</i> SBC <i>opr8a</i> SBC <i>opr16a</i> SBC <i>opr16,X</i> SBC <i>opr8,X</i> SBC <i>,X</i> SBC <i>opr16,SP</i> SBC <i>opr8,SP</i>	Subtract with Carry	A ← (A) – (M) – (C)		–	–				IMM DIR EXT IX2 IX1 IX SP2 SP1	A2 B2 C2 D2 E2 F2 9ED2 9EE2	ii dd hh ll ee ff ff ff ee ff ff	2 3 4 4 3 3 5 4
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask Bit	I ← 1	–	–	1	–	–	–	INH	9B		1
STA <i>opr8a</i> STA <i>opr16a</i> STA <i>opr16,X</i> STA <i>opr8,X</i> STA <i>,X</i> STA <i>opr16,SP</i> STA <i>opr8,SP</i>	Store Accumulator in Memory	M ← (A)	0	–	–			–	DIR EXT IX2 IX1 IX SP2 SP1	B7 C7 D7 E7 F7 9ED7 9EE7	dd hh ll ee ff ff ff ee ff ff	3 4 4 3 2 5 4
STHX <i>opr8a</i> STHX <i>opr16a</i> STHX <i>opr8,SP</i>	Store H:X (Index Reg.)	(M:M + 0x0001) ← (H:X)	0	–	–			–	DIR EXT SP1	35 96 9EFF	dd hh ll ff	4 5 5
STOP	Enable Interrupts: Stop Processing Refer to MCU Documentation	I bit ← 0; Stop Processing	–	–	0	–	–	–	INH	8E		2+
STX <i>opr8a</i> STX <i>opr16a</i> STX <i>opr16,X</i> STX <i>opr8,X</i> STX <i>,X</i> STX <i>opr16,SP</i> STX <i>opr8,SP</i>	Store X (Low 8 Bits of Index Register) in Memory	M ← (X)	0	–	–			–	DIR EXT IX2 IX1 IX SP2 SP1	BF CF DF EF FF 9EDF 9EEF	dd hh ll ee ff ff ff ee ff ff	3 4 4 3 2 5 4
SUB <i>#opr8i</i> SUB <i>opr8a</i> SUB <i>opr16a</i> SUB <i>opr16,X</i> SUB <i>opr8,X</i> SUB <i>,X</i> SUB <i>opr16,SP</i> SUB <i>opr8,SP</i>	Subtract	A ← (A) – (M)		–	–				IMM DIR EXT IX2 IX1 IX SP2 SP1	A0 B0 C0 D0 E0 F0 9ED0 9EE0	ii dd hh ll ee ff ff ff ee ff ff	2 3 4 4 3 3 5 4
SWI	Software Interrupt	PC ← (PC) + 0x0001 Push (PCL); SP ← (SP) – 0x0001 Push (PCH); SP ← (SP) – 0x0001 Push (X); SP ← (SP) – 0x0001 Push (A); SP ← (SP) – 0x0001 Push (CCR); SP ← (SP) – 0x0001 I ← 1; PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		11

independently be configured to operate in input capture, output compare, or buffered edge-aligned PWM mode.

The following sections describe the main 16-bit counter and each of the timer operating modes (input capture, output compare, edge-aligned PWM, and center-aligned PWM). Because details of pin operation and interrupt activity depend on the operating mode, these topics are covered in the associated mode sections.

10.5.1 Counter

All timer functions are based on the main 16-bit counter (TPM1CNTH:TPM1CNTL). This section discusses selection of the clock source, up-counting vs. up-/down-counting, end-of-count overflow, and manual counter reset.

After any MCU reset, CLKS_B:CLKS_A = 0:0 so no clock source is selected and the TPM is inactive. Normally, CLKS_B:CLKS_A would be set to 0:1 so the bus clock drives the timer counter. The clock source for the TPM can be selected to be off, the bus clock (BUSCLK), the fixed system clock (XCLK), or an external input through the TPM1CH0 pin. The maximum frequency allowed for the external clock option is one-fourth the bus rate. Refer to Section 10.7.1, “Timer Status and Control Register (TPM1SC),” and Table 10-2 for more information about clock source selection.

When the microcontroller is in active background mode, the TPM temporarily suspends all counting until the microcontroller returns to normal user operating mode. During stop mode, all TPM clocks are stopped; therefore, the TPM is effectively disabled until clocks resume. During wait mode, the TPM continues to operate normally.

The main 16-bit counter has two counting modes. When center-aligned PWM is selected (CPWMS = 1), the counter operates in up-/down-counting mode. Otherwise, the counter operates as a simple up-counter. As an up-counter, the main 16-bit counter counts from \$0000 through its terminal count and then continues with \$0000. The terminal count is \$FFFF or a modulus value in TPM1MODH:TPM1MODL.

When center-aligned PWM operation is specified, the counter counts upward from \$0000 through its terminal count and then counts downward to \$0000 where it returns to up-counting. Both \$0000 and the terminal count value (value in TPM1MODH:TPM1MODL) are normal length counts (one timer clock period long).

An interrupt flag and enable are associated with the main 16-bit counter. The timer overflow flag (TOF) is a software-accessible indication that the timer counter has overflowed. The enable signal selects between software polling (TOIE = 0) where no hardware interrupt is generated, or interrupt-driven operation (TOIE = 1) where a static hardware interrupt is automatically generated whenever the TOF flag is 1.

The conditions that cause TOF to become set depend on the counting mode (up or up/down). In up-counting mode, the main 16-bit counter counts from \$0000 through \$FFFF and overflows to \$0000 on the next counting clock. TOF becomes set at the transition from \$FFFF to \$0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to \$0000. When the main 16-bit counter is operating in up-/down-counting mode, the TOF flag gets set as the counter changes direction at the transition from the value set in the modulus register and the next lower count value. This corresponds to the end of a PWM period. (The \$0000 count value corresponds to the center of a period.)

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.)

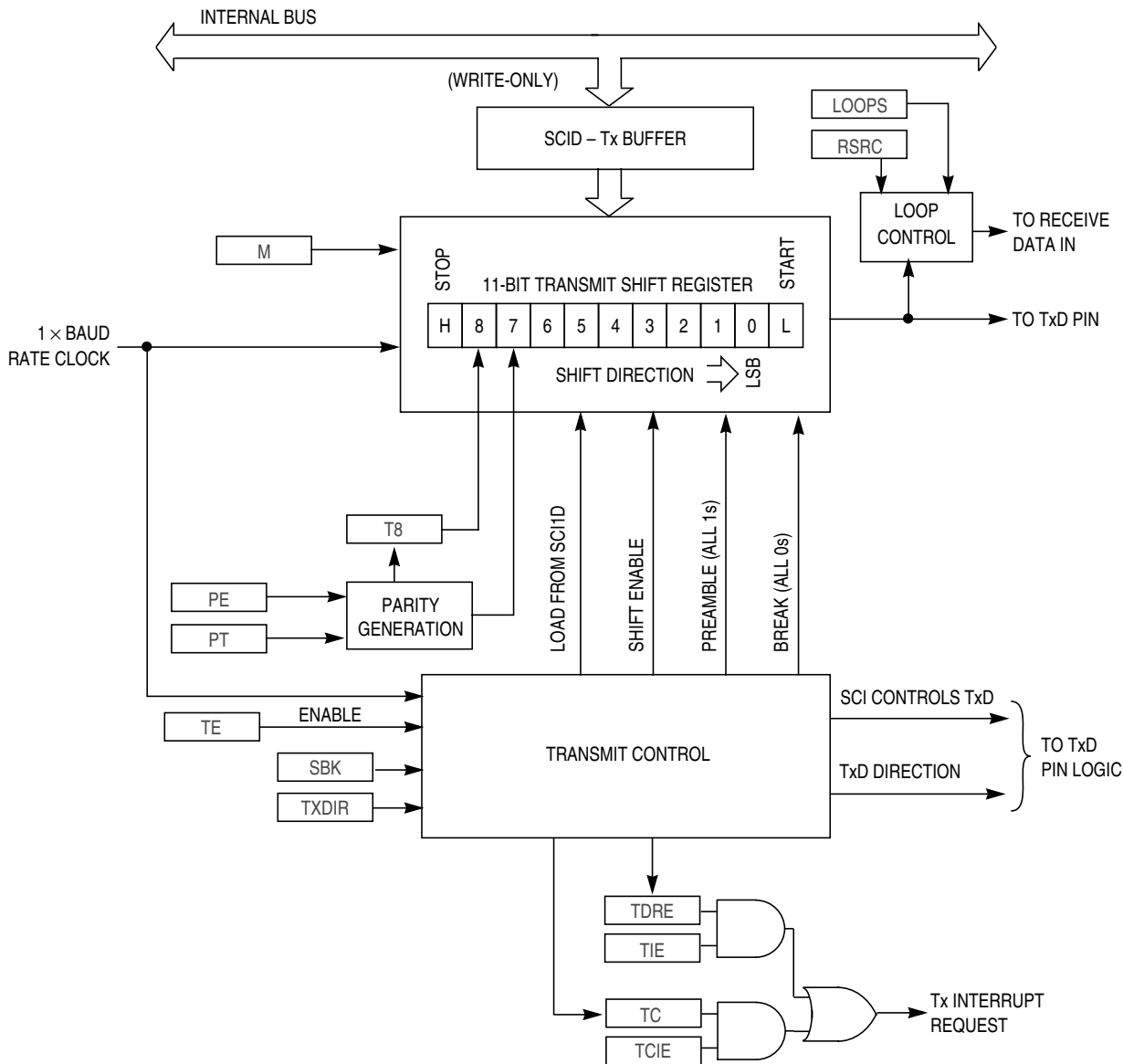


Figure 12-1. SCI Transmitter Block Diagram

12.2 Register Definition

The SCI has eight 8-bit registers to control baud rate, select SCI options, report SCI status, and for transmit/receive data.

Refer to the direct-page register summary in the Memory chapter of this data sheet for the absolute address assignments for all SCI 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.

12.2.1 SCI Baud Rate Registers (SCI1BDH, SCI1BDL)

This pair of registers controls the prescale divisor for SCI baud rate generation. To update the 13-bit baud rate setting [SBR12:SBR0], first write to SCI1BDH to buffer the high half of the new value and then write to SCI1BDL. The working value in SCI1BDH does not change until SCI1BDL is written.

SCI1BDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled (RE or TE bits in SCI1C2 are written to 1).

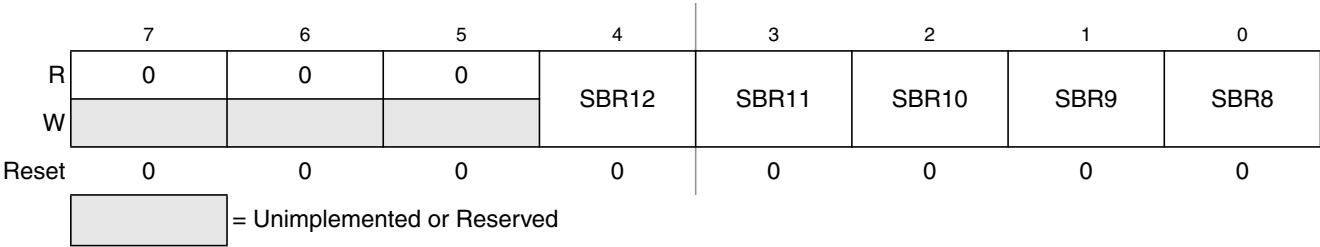


Figure 12-3. SCI Baud Rate Register (SCI1BDH)

Table 12-1. SCI1BDH Register Field Descriptions

Field	Description
4:0 SBR[12:8]	Baud Rate Modulo Divisor — These 13 bits are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR = 0, the SCI baud rate generator is disabled to reduce supply current. When BR = 1 to 8191, the SCI baud rate = $BUSCLK/(16 \times BR)$. See also BR bits in Table 12-2.

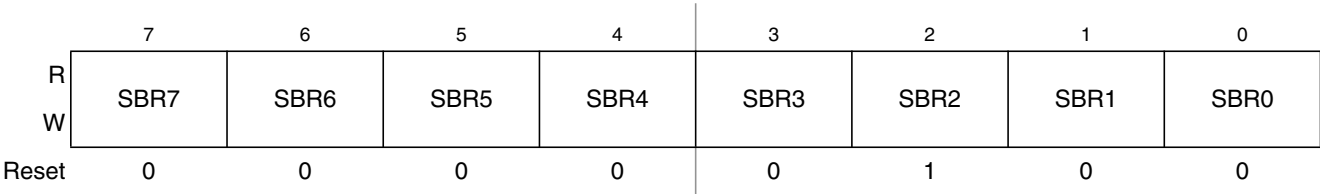


Figure 12-4. SCI Baud Rate Register (SCI1BDL)

Table 12-2. SCI1BDL Register Field Descriptions

Field	Description
7:0 SBR[7:0]	Baud Rate Modulo Divisor — These 13 bits are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR = 0, the SCI baud rate generator is disabled to reduce supply current. When BR = 1 to 8191, the SCI baud rate = $BUSCLK/(16 \times BR)$. See also BR bits in Table 12-1.

SSOE — Slave Select Output Enable

This bit is used in combination with the mode fault enable (MODFEN) bit in SPCR2 and the master/slave (MSTR) control bit to determine the function of the \overline{SS} pin as shown in Table 13-1.

Table 13-1. \overline{SS} Pin Function

MODFEN	SSOE	Master Mode	Slave Mode
0	0	General-purpose I/O (not SPI)	Slave select input
0	1	General-purpose I/O (not SPI)	Slave select input
1	0	\overline{SS} input for mode fault	Slave select input
1	1	Automatic \overline{SS} output	Slave select input

LSBFE — LSB First (Shifter Direction)

- 1 = SPI serial data transfers start with least significant bit.
- 0 = SPI serial data transfers start with most significant bit.

13.4.2 SPI Control Register 2 (SPI1C2)

This read/write register is used to control optional features of the SPI system. Bits 7, 6, 5, and 2 are not implemented and always read 0.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	<st-blue>	<st-blue>	0	<st-blue>	<st-blue>
Write:				MODFEN	BIDIROE		SPISWAI	SPC0
Reset:	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 13-8. SPI Control Register 2 (SPI1C2)

MODFEN — Master Mode-Fault Function Enable

When the SPI is configured for slave mode, this bit has no meaning or effect. (The \overline{SS} pin is the slave select input.) In master mode, this bit determines how the \overline{SS} pin is used (refer to Table 13-1 for more details).

- 1 = Mode fault function enabled, master \overline{SS} pin acts as the mode fault input or the slave select output.
- 0 = Mode fault function disabled, master \overline{SS} pin reverts to general-purpose I/O not controlled by SPI.

Table 13-2. SPI Baud Rate Prescaler Divisor

SPPR2:SPPR1:SPPR0	Prescaler Divisor
0:0:0	1
0:0:1	2
0:1:0	3
0:1:1	4
1:0:0	5
1:0:1	6
1:1:0	7
1:1:1	8

SPR2:SPR1:SPR0 — SPI Baud Rate Divisor

This 3-bit field selects one of eight divisors for the SPI baud rate divider as shown in Figure 13-3. The input to this divider comes from the SPI baud rate prescaler (see Figure 13-4). The output of this divider is the SPI bit rate clock for master mode.

Table 13-3. SPI Baud Rate Divisor

SPR2:SPR1:SPR0	Rate Divisor
0:0:0	2
0:0:1	4
0:1:0	8
0:1:1	16
1:0:0	32
1:0:1	64
1:1:0	128
1:1:1	256

15.3 On-Chip Debug System (DBG)

Because HCS08 devices do not have external address and data buses, the most important functions of an in-circuit emulator have been built onto the chip with the MCU. The debug system consists of an 8-stage FIFO that can store address or data bus information, and a flexible trigger system to decide when to capture bus information and what information to capture. The system relies on the single-wire background debug system to access debug control registers and to read results out of the eight stage FIFO.

The debug module includes control and status registers that are accessible in the user's memory map. These registers are located in the high register space to avoid using valuable direct page memory space.

Most of the debug module's functions are used during development, and user programs rarely access any of the control and status registers for the debug module. The one exception is that the debug system can provide the means to implement a form of ROM patching. This topic is discussed in greater detail in Section 15.3.6, "Hardware Breakpoints."

15.3.1 Comparators A and B

Two 16-bit comparators (A and B) can optionally be qualified with the R/W signal and an opcode tracking circuit. Separate control bits allow you to ignore R/W for each comparator. The opcode tracking circuitry optionally allows you to specify that a trigger will occur only if the opcode at the specified address is actually executed as opposed to only being read from memory into the instruction queue. The comparators are also capable of magnitude comparisons to support the inside range and outside range trigger modes. Comparators are disabled temporarily during all BDC accesses.

The A comparator is always associated with the 16-bit CPU address. The B comparator compares to the CPU address or the 8-bit CPU data bus, depending on the trigger mode selected. Because the CPU data bus is separated into a read data bus and a write data bus, the RWAEN and RWA control bits have an additional purpose, in full address plus data comparisons they are used to decide which of these buses to use in the comparator B data bus comparisons. If RWAEN = 1 (enabled) and RWA = 0 (write), the CPU's write data bus is used. Otherwise, the CPU's read data bus is used.

The currently selected trigger mode determines what the debugger logic does when a comparator detects a qualified match condition. A match can cause:

- Generation of a breakpoint to the CPU
- Storage of data bus values into the FIFO
- Starting to store change-of-flow addresses into the FIFO (begin type trace)
- Stopping the storage of change-of-flow addresses into the FIFO (end type trace)

15.3.2 Bus Capture Information and FIFO Operation

The usual way to use the FIFO is to setup the trigger mode and other control options, then arm the debugger. When the FIFO has filled or the debugger has stopped storing data into the FIFO, you would read the information out of it in the order it was stored into the FIFO. Status bits indicate the number of words of valid information that are in the FIFO as data is stored into it. If a trace run is manually halted by writing 0 to ARM before the FIFO is full (CNT = 1:0:0:0), the information is shifted by one position and

the host must perform $((8 - \text{CNT}) - 1)$ dummy reads of the FIFO to advance it to the first significant entry in the FIFO.

In most trigger modes, the information stored in the FIFO consists of 16-bit change-of-flow addresses. In these cases, read DBGFH then DBGFL to get one coherent word of information out of the FIFO. Reading DBGFL (the low-order byte of the FIFO data port) causes the FIFO to shift so the next word of information is available at the FIFO data port. In the event-only trigger modes (see Section 15.3.5, “Trigger Modes”), 8-bit data information is stored into the FIFO. In these cases, the high-order half of the FIFO (DBGFH) is not used and data is read out of the FIFO by simply reading DBGFL. Each time DBGFL is read, the FIFO is shifted so the next data value is available through the FIFO data port at DBGFL.

In trigger modes where the FIFO is storing change-of-flow addresses, there is a delay between CPU addresses and the input side of the FIFO. Because of this delay, if the trigger event itself is a change-of-flow address or a change-of-flow address appears during the next two bus cycles after a trigger event starts the FIFO, it will not be saved into the FIFO. In the case of an end-trace, if the trigger event is a change-of-flow, it will be saved as the last change-of-flow entry for that debug run.

The FIFO can also be used to generate a profile of executed instruction addresses when the debugger is not armed. When $\text{ARM} = 0$, reading DBGFL causes the address of the most-recently fetched opcode to be saved in the FIFO. To use the profiling feature, a host debugger would read addresses out of the FIFO by reading DBGFH then DBGFL at regular periodic intervals. The first eight values would be discarded because they correspond to the eight DBGFL reads needed to initially fill the FIFO. Additional periodic reads of DBGFH and DBGFL return delayed information about executed instructions so the host debugger can develop a profile of executed instruction addresses.

15.3.3 Change-of-Flow Information

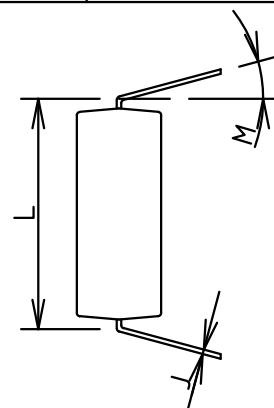
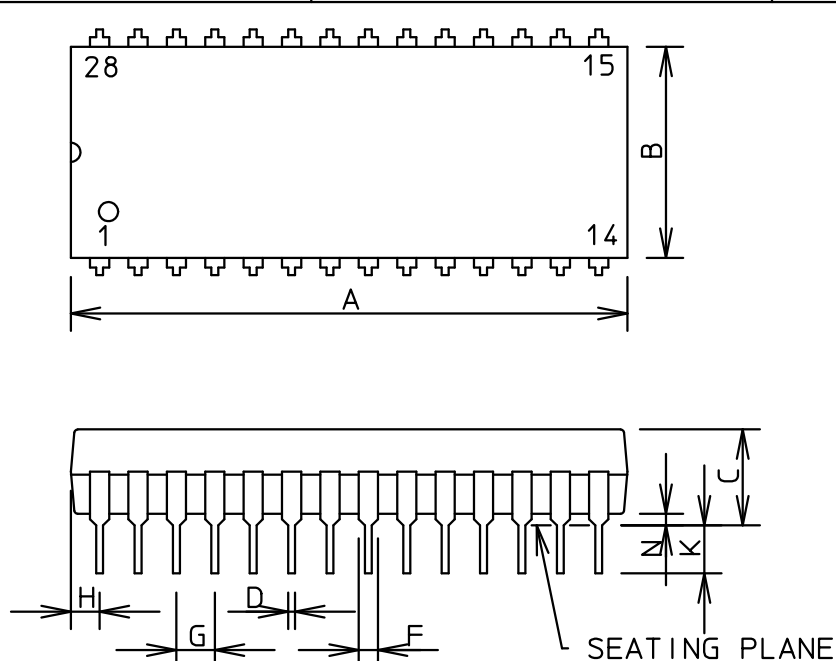
To minimize the amount of information stored in the FIFO, only information related to instructions that cause a change to the normal sequential execution of instructions is stored. With knowledge of the source and object code program stored in the target system, an external debugger system can reconstruct the path of execution through many instructions from the change-of-flow information stored in the FIFO.

For conditional branch instructions where the branch is taken (branch condition was true), the source address is stored (the address of the conditional branch opcode). Because BRA and BRN instructions are not conditional, these events do not cause change-of-flow information to be stored in the FIFO.

Indirect JMP and JSR instructions use the current contents of the H:X index register pair to determine the destination address, so the debug system stores the run-time destination address for any indirect JMP or JSR. For interrupts, RTI, or RTS, the destination address is stored in the FIFO as change-of-flow information.

15.3.4 Tag vs. Force Breakpoints and Triggers

Tagging is a term that refers to identifying an instruction opcode as it is fetched into the instruction queue, but not taking any other action until and unless that instruction is actually executed by the CPU. This distinction is important because any change-of-flow from a jump, branch, subroutine call, or interrupt causes some instructions that have been fetched into the instruction queue to be thrown away without being executed.

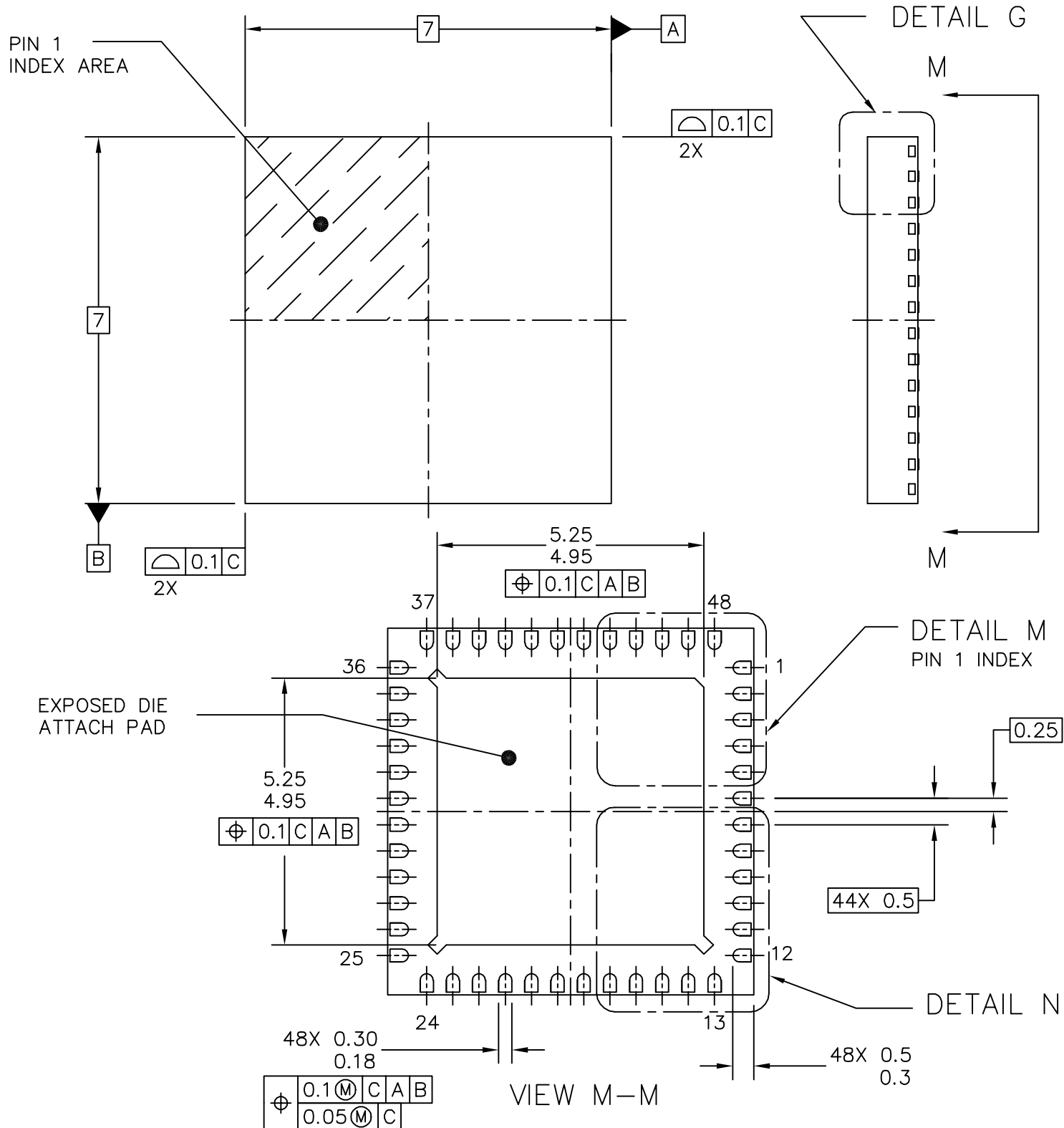


DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	36.45	37.21	1.435	1.465
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

NOTES:

1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
4. 710-01 OBSOLETE, NEW STD 710-02.
5. CONTROLLING DIMENSION: INCH

CASE NO.	710-02
STANDARD	MOT STD
REFERENCE	-
TITLE	28 LD PDIP



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