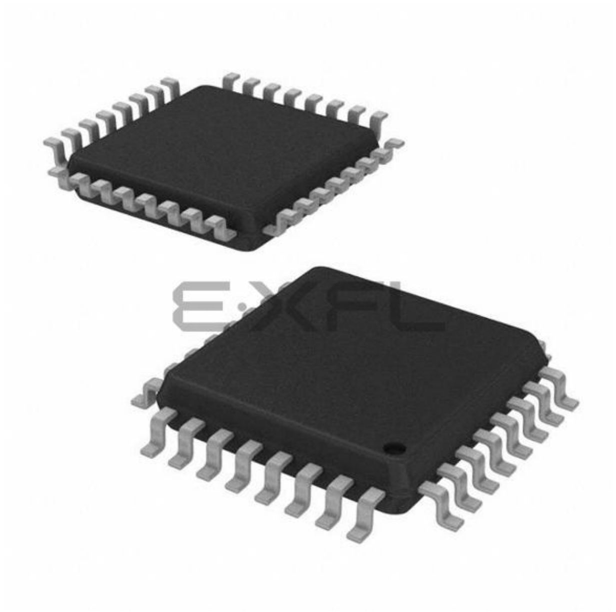
# NXP USA Inc. - MC9S08JM16CLC Datasheet



**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 | 48MHz |
| Connectivity | I²C, LINbus, SCI, SPI, USB |
| Peripherals | LVD, POR, PWM, WDT |
| Number of I/O | 21 |
| Program Memory Size | 16KB (16K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 1K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 5.5V |
| Data Converters | A/D 4x12b |
| Oscillator Type | External |
| 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/product-detail/nxp-semiconductors/mc9s08jm16clc |

# Contents

**Section Number** **Title** **Page**

## Chapter 1
## Device Overview

## Chapter 2
## Pins and Connections

## Chapter 3
## Modes of Operation

## Chapter 4
## Memory

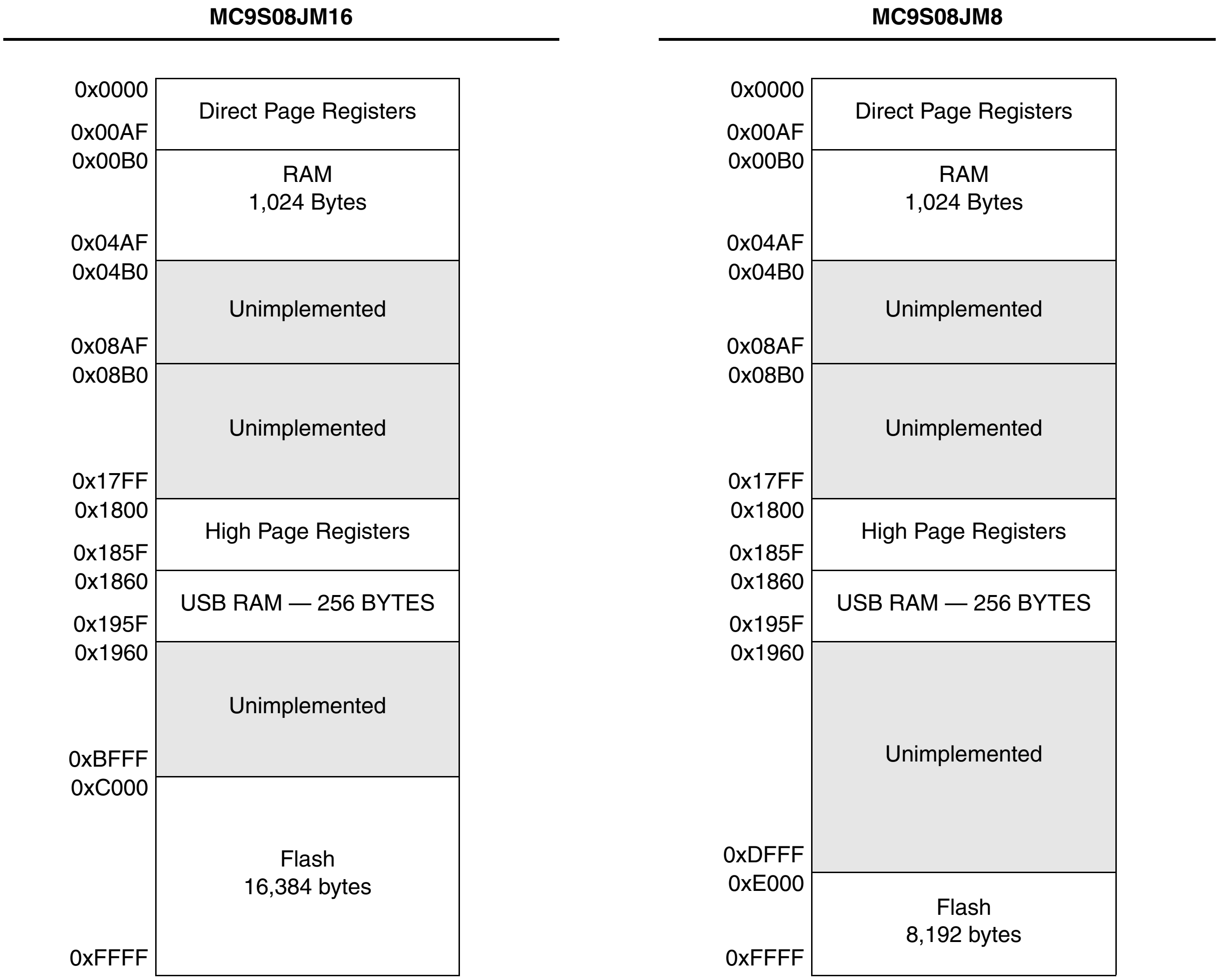


**Figure 4-1. MC9S08JM16 Series Memory Map**

### Table 4-2. Direct-Page Register Summary (Sheet 2 of 4)

| Address | Register Name | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x002A | TPM1C1VL | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x002B | TPM1C2SC | CH2F | CH2IE | MS2B | MS2A | ELS2B | ELS2A | 0 | 0 |
| 0x002C | TPM1C2VH | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| 0x002D | TPM1C2VL | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x002E | TPM1C3SC | CH3F | CH3IE | MS3B | MS3A | ELS3B | ELS3A | 0 | 0 |
| 0x002F | TPM1C3VH | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| 0x0030 | TPM1C3VL | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0031 – 0x0037 | Reserved | — | — | — | — | — | — | — | — |
| 0x0038 | SCI1BDH | LBKDIE | RXEDGIE | 0 | SBR12 | SBR11 | SBR10 | SBR9 | SBR8 |
| 0x0039 | SCI1BDL | SBR7 | SBR6 | SBR5 | SBR4 | SBR3 | SBR2 | SBR1 | SBR0 |
| 0x003A | SCI1C1 | LOOPS | SCISWAI | RSRC | M | WAKE | ILT | PE | PT |
| 0x003B | SCI1C2 | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| 0x003C | SCI1S1 | TDRE | TC | RDRF | IDLE | OR | NF | FE | PF |
| 0x003D | SCI1S2 | LBKDIF | RXEDGIF | 0 | RXINV | RWUID | BRK13 | LBKDE | RAF |
| 0x003E | SCI1C3 | R8 | T8 | TXDIR | TXINV | ORIE | NEIE | FEIE | PEIE |
| 0x003F | SCI1D | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0040 | SCI2BDH | LBKDIE | RXEDGIE | 0 | SBR12 | SBR11 | SBR10 | SBR9 | SBR8 |
| 0x0041 | SCI2BDL | SBR7 | SBR6 | SBR5 | SBR4 | SBR3 | SBR2 | SBR1 | SBR0 |
| 0x0042 | SCI2C1 | LOOPS | SCISWAI | RSRC | M | WAKE | ILT | PE | PT |
| 0x0043 | SCI2C2 | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| 0x0044 | SCI2S1 | TDRE | TC | RDRF | IDLE | OR | NF | FE | PF |
| 0x0045 | SCI2S2 | LBKDIF | RXEDGIF | 0 | RXINV | RWUID | BRK13 | LBKDE | RAF |
| 0x0046 | SCI2C3 | R8 | T8 | TXDIR | TXINV | ORIE | NEIE | FEIE | PEIE |
| 0x0047 | SCI2D | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0048 | MCGC1 | CLKS | | RDIV | | | IREFS | IRCLKEN | IREFSTEN |
| 0x0049 | MCGC2 | BDIV | | RANGE | HGO | LP | EREFS | ERCLKEN | EREFSTEN |
| 0x004A | MCGTRM | TRIM | | | | | | | |
| 0x004B | MCGSC | LOLS | LOCK | PLLST | IREFST | CLKST | | OSCINIT | FTRIM |
| 0x004C | MCGC3 | LOLIE | PLLS | CME | 0 | VDIV | | | |
| 0x004D | MCGT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x004E – 0x004F | Reserved | — | — | — | — | — | — | — | — |
| 0x0050 | SPI1C1 | SPIE | SPE | SPTIE | MSTR | CPOL | CPHA | SSOE | LSBFE |
| 0x0051 | SPI1C2 | SPMIE | SPIMODE | 0 | MODFEN | BIDIROE | 0 | SPISWAI | SPC0 |
| 0x0052 | SPI1BR | 0 | SPPR2 | SPPR1 | SPPR0 | 0 | SPR2 | SPR1 | SPR0 |
| 0x0053 | SPI1S | SPRF | SPMF | SPTEF | MODF | 0 | 0 | 0 | 0 |
| 0x0054 | SPI1DH | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| 0x0055 | SPI1DL | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0056 | SPI1MH | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| 0x0057 | SPI1ML | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0058 | IICA | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | 0 |

is issued, an internal charge pump associated with the flash memory must be enabled to supply high voltage to the array. Upon completion of the command, the charge pump is turned off. When a burst program command is issued, the charge pump is enabled and then remains enabled after completion of the burst program operation if these two conditions are met:

- The next burst program command has been queued before the current program operation has completed.
- The next sequential address selects a byte on the same physical row as the current byte being programmed. A row of flash memory consists of 64 bytes. A byte within a row is selected by addresses A5 through A0. A new row begins when addresses A5 through A0 are all zero.

The first byte of a series of sequential bytes being programmed in burst mode will take the same amount of time to program as a byte programmed in standard mode. Subsequent bytes will program in the burst program time provided that the conditions above are met. In the case the next sequential address is the beginning of a new row, the program time for that byte will be the standard time instead of the burst time. This is because the high voltage to the array must be disabled and then enabled again. If a new burst command has not been queued before the current command completes, then the charge pump will be disabled and high voltage will be removed from the array.

**Table 4-9. Security States**

| SEC01:SEC00 | Description |
|---|---|
| 0:0 | Secure |
| 0:1 | Secure |
| 1:0 | Unsecured |
| 1:1 | Secure |

SEC01:SEC00 changes to 1:0 after successful backdoor key entry or a successful blank check of flash.

### 4.7.3 Flash Configuration Register (FCNFG)

Bits 7 through 5 may be read or written at any time. Bits 4 through 0 always read 0 and cannot be written.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | KEYACC | 0 | 0 | 0 | 0 | 0 |
| W | ▨ | ▨ | | ▨ | ▨ | ▨ | ▨ | ▨ |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

▨ = Unimplemented or Reserved

**Figure 4-7. Flash Configuration Register (FCNFG)**

**Table 4-10. FCNFG Register Field Descriptions**

| Field | Description |
|---|---|
| 5<br>KEYACC | **Enable Writing of Access Key** — This bit enables writing of the backdoor comparison key. For more detailed information about the backdoor key mechanism, refer to Section 4.6, “Security.”<br>0 Writes to 0xFFB0–0xFFB7 are interpreted as the start of a flash programming or erase command.<br>1 Writes to NVBACKKEY (0xFFB0–0xFFB7) are interpreted as comparison key writes. |

### 4.7.4 Flash Protection Register (FPROT and NVPROT)

During reset, the contents of the nonvolatile location NVPROT are copied from flash into FPROT. Bits 0, 1, and 2 are not used and each always reads as 0. This register may be read at any time, but user program writes have no meaning or effect. Background debug commands can write to FPROT.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | FPS7 | FPS6 | FPS5 | FPS4 | FPS3 | FPS2 | FPS1 | FPDIS |
| W | [1] | [1] | [1] | [1] | [1] | [1] | [1] | [1] |
| Reset | This register is loaded from nonvolatile location NVPROT during reset. | | | | | | | |

[1] Background commands can be used to change the contents of these bits in FPROT.

**Figure 4-8. Flash Protection Register (FPROT)**

**Table 5-1. Vector Summary (from Lowest to Highest Priority) (continued)**

| Vector Number | Address (High/Low) | Vector Name | Module | Source | Enable | Description |
|---|---|---|---|---|---|---|
| 24 | 0xFFCE:FFCF | Vsci2tx | SCI2 | TDRE<br>TC | TIE<br>TCIE | SCI2 transmit |
| 23 | 0xFFD0:FFD1 | Vsci2rx | SCI2 | IDLE<br>RDRF | ILIE<br>RIE | SCI2 receive |
| 22 | 0xFFD2:FFD3 | Vsci2err | SCI2 | OR<br>NF<br>FE<br>PF | ORIE<br>NFIE<br>FEIE<br>PFIE | SCI2 error |
| 21 | 0xFFD4:FFD5 | Vsci1tx | SCI1 | TDRE<br>TC | TIE<br>TCIE | SCI1 transmit |
| 20 | 0xFFD6:FFD7 | Vsci1rx | SCI1 | IDLE<br>RDRF | ILIE<br>RIE | SCI1 receive |
| 19 | 0xFFD8:FFD9 | Vsci1err | SCI1 | OR<br>NF<br>FE<br>PF | ORIE<br>NFIE<br>FEIE<br>PFIE | SCI1 error |
| 18 | 0xFFDA:FFDB | Vtpm2ovf | TPM2 | TOF | TOIE | TPM2 overflow |
| 17 | 0xFFDC:FFDD | Vtpm2ch1 | TPM2 | CH1F | CH1IE | TPM2 channel 1 |
| 16 | 0xFFDE:FFDF | Vtpm2ch0 | TPM2 | CH0F | CH0IE | TPM2 channel 0 |
| 15 | 0xFFE0:FFE1 | Vtpm1ovf | TPM1 | TOF | TOIE | TPM1 overflow |
| 14 | 0xFFE2:FFE3 | reserved | reserved | reserved | reserved | reserved |
| 13 | 0xFFE4:FFE5 | reserved | reserved | reserved | reserved | reserved |
| 12 | 0xFFE6:FFE7 | Vtpm1ch3 | TPM1 | CH3F | CH3IE | TPM1 channel 3 |
| 11 | 0xFFE8:FFE9 | Vtpm1ch2 | TPM1 | CH2F | CH2IE | TPM1 channel 2 |
| 10 | 0xFFEA:FFEB | Vtpm1ch1 | TPM1 | CH1F | CH1IE | TPM1 channel 1 |
| 9 | 0xFFEC:FFED | Vtpm1ch0 | TPM1 | CH0F | CH0IE | TPM1 channel 0 |
| 8 | 0xFFEE:FFEF | reserved | — | — | — | — |
| 7 | 0xFFF0:FFF1 | Vusb | USB | STALLF<br>RESUMEF<br>SLEEPF<br>TOKDNEF<br>SOFTOKF<br>ERRORF<br>USBRSTF | STALL<br>RESUME<br>SLEEP<br>TOKDNE<br>SOFTOK<br>ERROR<br>USBRST | USB Status |
| 6 | 0xFFF2:FFF3 | Vspi2 | SPI2 | SPRF<br>MODF<br>SPTEF<br>SPMF | SPIE<br>SPIE<br>SPTIE<br>SPMIE | SPI2 |

### 6.5.2 Port A Pin Control Registers (PTAPE, PTASE, PTADS)

In addition to the I/O control, port A pins are controlled by the registers listed below.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | | | PTAPE5 | | | | | PTAPE0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 6-4. Internal Pullup Enable for Port A (PTAPE)**

**Table 6-3. PTADD Register Field Descriptions**

| Field | Description |
|---|---|
| 5,0<br>PTAPE[5,0] | **Internal Pullup Enable for Port A Bits** — Each of these control bits determines if the internal pullup device is enabled for the associated PTA pin. For port A pins that are configured as outputs, these bits have no effect and the internal pullup devices are disabled.<br>0 Internal pullup device disabled for port A bit n.<br>1 Internal pullup device enabled for port A bit n. |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | | | PTASE5 | | | | | PTASE0 |
| Reset | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figure 6-5. Output Slew Rate Control Enable for Port A (PTASE)**

**Table 6-4. PTASE Register Field Descriptions**

| Field | Description |
|---|---|
| 5,0<br>PTASE[5,0] | **Output Slew Rate Control Enable for Port A Bits** — Each of these control bits determine whether output slew rate control is enabled for the associated PTA pin. For port A pins that are configured as inputs, these bits have no effect.<br>0 Output slew rate control disabled for port A bit n.<br>1 Output slew rate control enabled for port A bit n. |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | | | PTADS5 | | | | | PTADS0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 6-6. Output Drive Strength Selection for Port A (PTASE)**

interrupt service routine, this would allow nesting of interrupts (which is not recommended because it leads to programs that are difficult to debug and maintain).

For compatibility with the earlier M68HC05 MCUs, the high-order half of the H:X index register pair (H) is not saved on the stack as part of the interrupt sequence. The user must use a PSHH instruction at the beginning of the service routine to save H and then use a PULH instruction just before the RTI that ends the interrupt service routine. It is not necessary to save H if you are certain that the interrupt service routine does not use any instructions or auto-increment addressing modes that might change the value of H.

The software interrupt (SWI) instruction is like a hardware interrupt except that it is not masked by the global I bit in the CCR and it is associated with an instruction opcode within the program so it is not asynchronous to program execution.

### 7.4.3 Wait Mode Operation

The WAIT instruction enables interrupts by clearing the I bit in the CCR. It then halts the clocks to the CPU to reduce overall power consumption while the CPU is waiting for the interrupt or reset event that will wake the CPU from wait mode. When an interrupt or reset event occurs, the CPU clocks will resume and the interrupt or reset event will be processed normally.

If a serial BACKGROUND command is issued to the MCU through the background debug interface while the CPU is in wait mode, CPU clocks will resume and the CPU will enter active background mode where other serial background commands can be processed. This ensures that a host development system can still gain access to a target MCU even if it is in wait mode.

### 7.4.4 Stop Mode Operation

Usually, all system clocks, including the crystal oscillator (when used), are halted during stop mode to minimize power consumption. In such systems, external circuitry is needed to control the time spent in stop mode and to issue a signal to wake up the target MCU when it is time to resume processing. Unlike the earlier M68HC05 and M68HC08 MCUs, the HCS08 can be configured to keep a minimum set of clocks running in stop mode. This optionally allows an internal periodic signal to wake the target MCU from stop mode.

When a host debug system is connected to the background debug pin (BKGD) and the ENBDM control bit has been set by a serial command through the background interface (or because the MCU was reset into active background mode), the oscillator is forced to remain active when the MCU enters stop mode. In this case, if a serial BACKGROUND command is issued to the MCU through the background debug interface while the CPU is in stop mode, CPU clocks will resume and the CPU will enter active background mode where other serial background commands can be processed. This ensures that a host development system can still gain access to a target MCU even if it is in stop mode.

Recovery from stop mode depends on the particular HCS08 and whether the oscillator was stopped in stop mode. Refer to the Modes of Operation chapter for more details.

## Table 7-2. . Instruction Set Summary (Sheet 2 of 9)

| Source Form | Operation | Address Mode | Object Code | Cycles | Cyc-by-Cyc Details | Affect on CCR | |
|---|---|---|---|---|---|---|---|
| | | | | | | VH | I N Z C |
| BCLR *n,opr8a* | Clear Bit n in Memory<br>(Mn ← 0) | DIR (b0)<br>DIR (b1)<br>DIR (b2)<br>DIR (b3)<br>DIR (b4)<br>DIR (b5)<br>DIR (b6)<br>DIR (b7) | 11 dd<br>13 dd<br>15 dd<br>17 dd<br>19 dd<br>1B dd<br>1D dd<br>1F dd | 5<br>5<br>5<br>5<br>5<br>5<br>5<br>5 | rfwpp<br>rfwpp<br>rfwpp<br>rfwpp<br>rfwpp<br>rfwpp<br>rfwpp<br>rfwpp | – – | – – – – |
| BCS *rel* | Branch if Carry Bit Set (if C = 1)<br>(Same as BLO) | REL | 25 rr | 3 | ppp | – – | – – – – |
| BEQ *rel* | Branch if Equal (if Z = 1) | REL | 27 rr | 3 | ppp | – – | – – – – |
| BGE *rel* | Branch if Greater Than or Equal To<br>(if N ⊕ V = 0) (Signed) | REL | 90 rr | 3 | ppp | – – | – – – – |
| BGND | Enter active background if ENBDM=1<br>Waits for and processes BDM commands<br>until GO, TRACE1, or TAGGO | INH | 82 | 5+ | fp...ppp | – – | – – – – |
| BGT *rel* | Branch if Greater Than (if Z \| (N ⊕ V) = 0)<br>(Signed) | REL | 92 rr | 3 | ppp | – – | – – – – |
| BHCC *rel* | Branch if Half Carry Bit Clear (if H = 0) | REL | 28 rr | 3 | ppp | – – | – – – – |
| BHCS *rel* | Branch if Half Carry Bit Set (if H = 1) | REL | 29 rr | 3 | ppp | – – | – – – – |
| BHI *rel* | Branch if Higher (if C \| Z = 0) | REL | 22 rr | 3 | ppp | – – | – – – – |
| BHS *rel* | Branch if Higher or Same (if C = 0)<br>(Same as BCC) | REL | 24 rr | 3 | ppp | – – | – – – – |
| BIH *rel* | Branch if IRQ Pin High (if IRQ pin = 1) | REL | 2F rr | 3 | ppp | – – | – – – – |
| BIL *rel* | Branch if IRQ Pin Low (if IRQ pin = 0) | REL | 2E rr | 3 | ppp | – – | – – – – |
| BIT #*opr8i*<br>BIT *opr8a*<br>BIT *opr16a*<br>BIT *oprx16*,X<br>BIT *oprx8*,X<br>BIT ,X<br>BIT *oprx16*,SP<br>BIT *oprx8*,SP | Bit Test<br>(A) & (M)<br>(CCR Updated but Operands Not Changed) | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP2<br>SP1 | A5 ii<br>B5 dd<br>C5 hh ll<br>D5 ee ff<br>E5 ff<br>F5<br>9E D5 ee ff<br>9E E5 ff | 2<br>3<br>4<br>4<br>3<br>3<br>5<br>4 | pp<br>rpp<br>prpp<br>prpp<br>rpp<br>rfp<br>pprpp<br>prpp | 0 – | – ↕ ↕ – |
| BLE *rel* | Branch if Less Than or Equal To<br>(if Z \| (N ⊕ V) = 1) (Signed) | REL | 93 rr | 3 | ppp | – – | – – – – |
| BLO *rel* | Branch if Lower (if C = 1) (Same as BCS) | REL | 25 rr | 3 | ppp | – – | – – – – |
| BLS *rel* | Branch if Lower or Same (if C \| Z = 1) | REL | 23 rr | 3 | ppp | – – | – – – – |
| BLT *rel* | Branch if Less Than (if N ⊕ V = 1) (Signed) | REL | 91 rr | 3 | ppp | – – | – – – – |
| BMC *rel* | Branch if Interrupt Mask Clear (if I = 0) | REL | 2C rr | 3 | ppp | – – | – – – – |
| BMI *rel* | Branch if Minus (if N = 1) | REL | 2B rr | 3 | ppp | – – | – – – – |
| BMS *rel* | Branch if Interrupt Mask Set (if I = 1) | REL | 2D rr | 3 | ppp | – – | – – – – |
| BNE *rel* | Branch if Not Equal (if Z = 0) | REL | 26 rr | 3 | ppp | – – | – – – – |
| BPL *rel* | Branch if Plus (if N = 0) | REL | 2A rr | 3 | ppp | – – | – – – – |

## Table 7-2. . Instruction Set Summary (Sheet 8 of 9)

| Source Form | Operation | Address Mode | Object Code | Cycles | Cyc-by-Cyc Details | Affect on CCR | |
|---|---|---|---|---|---|---|---|
| | | | | | | VH | I N Z C |
| SUB #*opr8i*<br>SUB *opr8a*<br>SUB *opr16a*<br>SUB *oprx16*,X<br>SUB *oprx8*,X<br>SUB ,X<br>SUB *oprx16*,SP<br>SUB *oprx8*,SP | Subtract<br>A ← (A) – (M) | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX<br>SP2<br>SP1 | A0 ii<br>B0 dd<br>C0 hh ll<br>D0 ee ff<br>E0 ff<br>F0<br>9E D0 ee ff<br>9E E0 ff | 2<br>3<br>4<br>4<br>3<br>3<br>5<br>4 | pp<br>rpp<br>prpp<br>prpp<br>rpp<br>rfp<br>pprpp<br>prpp | ↕ – | – ↕ ↕ ↕ |
| SWI | Software Interrupt<br>PC ← (PC) + $0001<br>Push (PCL); SP ← (SP) – $0001<br>Push (PCH); SP ← (SP) – $0001<br>Push (X); SP ← (SP) – $0001<br>Push (A); SP ← (SP) – $0001<br>Push (CCR); SP ← (SP) – $0001<br>I ← 1;<br>PCH ← Interrupt Vector High Byte<br>PCL ← Interrupt Vector Low Byte | INH | 83 | 11 | sssssvvfppp | – – | 1 – – – |
| TAP | Transfer Accumulator to CCR<br>CCR ← (A) | INH | 84 | 1 | p | ↕ ↕ | ↕ ↕ ↕ ↕ |
| TAX | Transfer Accumulator to X (Index Register Low)<br>X ← (A) | INH | 97 | 1 | p | – – | – – – – |
| TPA | Transfer CCR to Accumulator<br>A ← (CCR) | INH | 85 | 1 | p | – – | – – – – |
| TST *opr8a*<br>TSTA<br>TSTX<br>TST *oprx8*,X<br>TST ,X<br>TST *oprx8*,SP | Test for Negative or Zero (M) – $00<br>(A) – $00<br>(X) – $00<br>(M) – $00<br>(M) – $00<br>(M) – $00 | DIR<br>INH<br>INH<br>IX1<br>IX<br>SP1 | 3D dd<br>4D<br>5D<br>6D ff<br>7D<br>9E 6D ff | 4<br>1<br>1<br>4<br>3<br>5 | rfpp<br>p<br>p<br>rfpp<br>rfp<br>prfpp | 0 – | – ↕ ↕ – |
| TSX | Transfer SP to Index Reg.<br>H:X ← (SP) + $0001 | INH | 95 | 2 | fp | – – | – – – – |
| TXA | Transfer X (Index Reg. Low) to Accumulator<br>A ← (X) | INH | 9F | 1 | p | – – | – – – – |

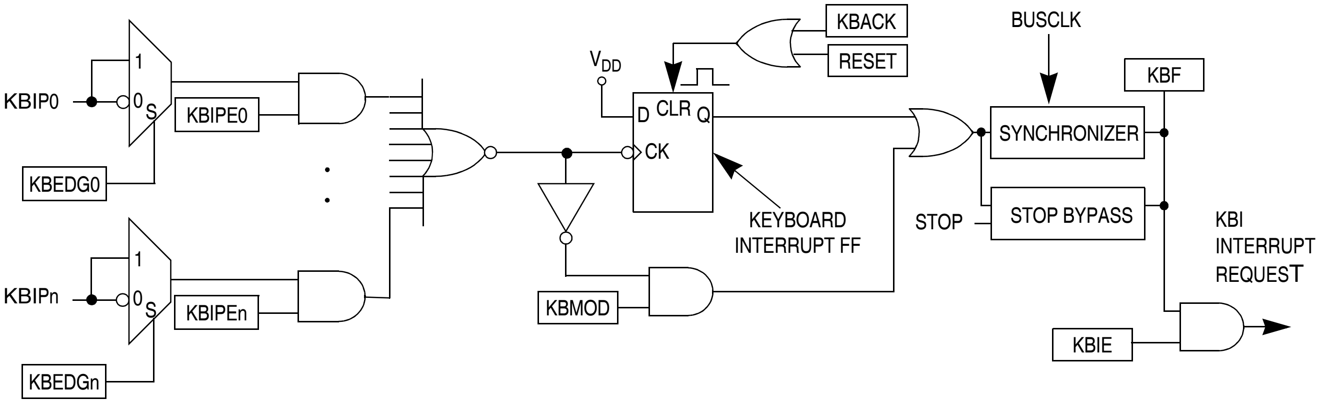


Figure 8-2. KBI Block Diagram

## 8.2 External Signal Description

The KBI input pins can be used to detect either falling edges, or both falling edge and low level interrupt requests. The KBI input pins can also be used to detect either rising edges, or both rising edge and high level interrupt requests.

The signal properties of KBI are shown in Table 8-1.

Table 8-1. Signal Properties

| Signal | Function | I/O |
|---|---|---|
| KBIPn | Keyboard interrupt pins | I |

## 8.3 Register Definition

The KBI includes three registers:

- An 8-bit pin status and control register.
- An 8-bit pin enable register.
- An 8-bit edge select register.

Refer to the direct-page register summary in the Memory chapter for the absolute address assignments for all KBI registers. This section refers to registers and control bits only by their names.

Some MCUs may have more than one KBI, so register names include placeholder characters to identify which KBI is being referenced.

### 8.3.1 KBI Status and Control Register (KBISC)

KBISC contains the status flag and control bits, which are used to configure the KBI.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | KBF | 0 | KBIE | KBMOD |
| W | ▨ | ▨ | ▨ | ▨ | ▨ | KBACK | | |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

▨ = Unimplemented

**Figure 8-3. KBI Status and Control Register**

**Table 8-2. KBISC Register Field Descriptions**

| Field | Description |
|---|---|
| 7:4 | Unused register bits, always read 0. |
| 3<br>KBF | **Keyboard Interrupt Flag** — KBF indicates when a keyboard interrupt is detected. Writes have no effect on KBF.<br>0 No keyboard interrupt detected.<br>1 Keyboard interrupt detected. |
| 2<br>KBACK | **Keyboard Acknowledge** — Writing a 1 to KBACK is part of the flag clearing mechanism. KBACK always reads as 0. |
| 1<br>KBIE | **Keyboard Interrupt Enable** — KBIE determines whether a keyboard interrupt is requested.<br>0 Keyboard interrupt request not enabled.<br>1 Keyboard interrupt request enabled. |
| 0<br>KBMOD | **Keyboard Detection Mode** — KBMOD (along with the KBEDG bits) controls the detection mode of the keyboard interrupt pins.0Keyboard detects edges only.<br>1 Keyboard detects both edges and levels. |

## 8.3.2 KBI Pin Enable Register (KBIPE)

KBIPE contains the pin enable control bits.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | KBIPE7 | KBIPE6 | KBIPE5 | KBIPE4 | KBIPE3 | KBIPE2 | KBIPE1 | KBIPE0 |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 8-4. KBI Pin Enable Register**

**Table 8-3. KBIPE Register Field Descriptions**

| Field | Description |
|---|---|
| 7:0<br>KBIPEn | **Keyboard Pin Enables** — Each of the KBIPEn bits enable the corresponding keyboard interrupt pin.<br>0 Pin not enabled as keyboard interrupt.<br>1 Pin enabled as keyboard interrupt. |

## 8.3.3 KBI Edge Select Register (KBIES)

KBIES contains the edge select control bits.

### 12.1.2 Modes of Operation

There are nine modes of operation for the MCG:

- FLL Engaged Internal (FEI)
- FLL Engaged External (FEE)
- FLL Bypassed Internal (FBI)
- FLL Bypassed External (FBE)
- PLL Engaged External (PEE)
- PLL Bypassed External (PBE)
- Bypassed Low Power Internal (BLPI)
- Bypassed Low Power External (BLPE)
- Stop

For details see Section 12.4.1, “Operational Modes.”

## 12.2 External Signal Description

There are no MCG signals that connect off chip.

Table 12-5. MCG PLL Register Field Descriptions (continued)

| Field | Description |
|---|---|
| 5<br>CME | **Clock Monitor Enable** — Determines if a reset request is made following a loss of external clock indication. The CME bit must only be set to a logic 1 when either the MCG is in an operational mode that uses the external clock (FEE, FBE, PEE, PBE, or BLPE) or the external reference is enabled (ERCLKEN=1 in the MCGC2 register). Whenever the CME bit is set to a logic 1, the value of the RANGE bit in the MCGC2 register must not be changed.<br>0 Clock monitor is disabled.<br>1 Generate a reset request on loss of external clock. |
| 3:0<br>VDIV | **VCO Divider** — Selects the amount to divide down the VCO output of PLL. The VDIV bits establish the multiplication factor (M) applied to the reference clock frequency.<br>0000 Encoding 0 — Reserved.<br>0001 Encoding 1 — Multiply by 4.<br>0010 Encoding 2 — Multiply by 8.<br>0011 Encoding 3 — Multiply by 12.<br>0100 Encoding 4 — Multiply by 16.<br>0101 Encoding 5 — Multiply by 20.<br>0110 Encoding 6 — Multiply by 24.<br>0111 Encoding 7 — Multiply by 28.<br>1000 Encoding 8 — Multiply by 32.<br>1001 Encoding 9 — Multiply by 36.<br>1010 Encoding 10 — Multiply by 40.<br>1011 Encoding 11 — Reserved (default to M=40).<br>11xx Encoding 12-15 — Reserved (default to M=40). |

minimum power consumption, leave the internal reference disabled while in an external clock mode.

3. After the proper configuration bits have been set, wait for the affected bits in the MCGSC register to be changed appropriately, reflecting that the MCG has moved into the proper mode.
   - If ERCLKEN was set in step 1 or the MCG is in FEE, FBE, PEE, PBE, or BLPE mode, and EREFS was also set in step 1, wait here for the OSCINIT bit to become set indicating that the external clock source has finished its initialization cycles and stabilized. Typical crystal startup times are given in Appendix A, “Electrical Characteristics”.
   - If in FEE mode, check to make sure the IREFST bit is cleared and the LOCK bit is set before moving on.
   - If in FBE mode, check to make sure the IREFST bit is cleared, the LOCK bit is set, and the CLKST bits have changed to %10 indicating the external reference clock has been appropriately selected. Although the FLL is bypassed in FBE mode, it is still on and will lock in FBE mode.

To change from FEI clock mode to FBI clock mode, follow this procedure:

1. Change the CLKS bits to %01 so that the internal reference clock is selected as the system clock source.
2. Wait for the CLKST bits in the MCGSC register to change to %01, indicating that the internal reference clock has been appropriately selected.

### 12.5.2 MCG Mode Switching

When switching between operational modes of the MCG, certain configuration bits must be changed in order to properly move from one mode to another. Each time any of these bits are changed (PLLS, IREFS, CLKS, or EREFS), the corresponding bits in the MCGSC register (PLLST, IREFST, CLKST, or OSCINIT) must be checked before moving on in the application software.

Additionally, care must be taken to ensure that the reference clock divider (RDIV) is set properly for the mode being switched to. For instance, in PEE mode, if using a 4 MHz crystal, RDIV must be set to %001 (divide-by-2) or %010 (divide-by-4) in order to divide the external reference down to the required frequency between 1 and 2 MHz.

The RDIV and IREFS bits must always be set properly before changing the PLLS bit so that the FLL or PLL clock has an appropriate reference clock frequency to switch to.

# Chapter 14
# Serial Communications Interface (S08SCIV4)

## 14.1 Introduction

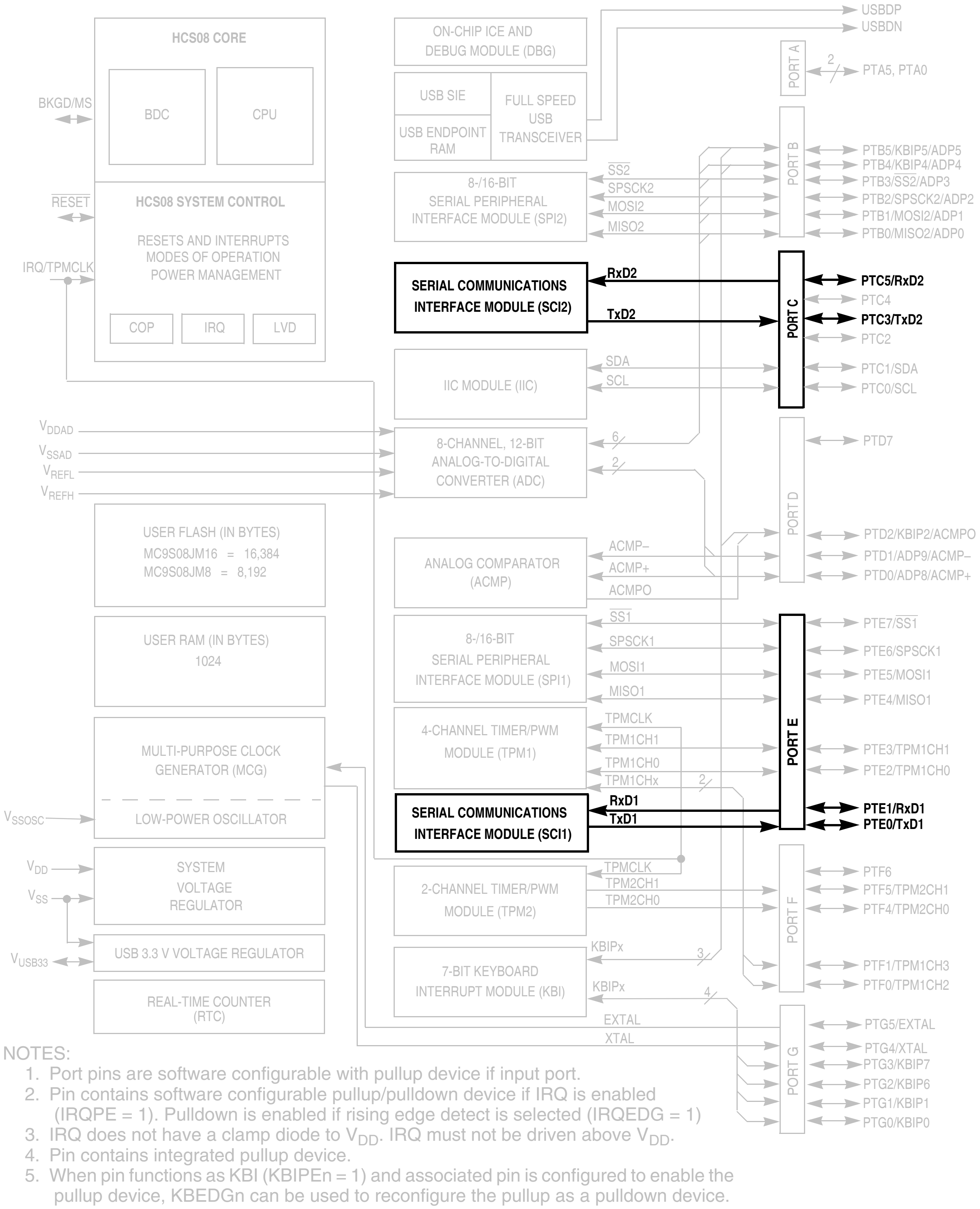
The MC9S08JM16 series include two independent serial communications interface (SCI) modules, which are sometimes called universal asynchronous receiver/transmitters (UARTs). Typically, these systems are used to connect to the RS232 serial input/output (I/O) port of a personal computer or workstation, but they can also be used to communicate with other embedded controllers.

A flexible, 13-bit, modulo-based baud rate generator supports a broad range of standard baud rates beyond 115.2 kbaud. Transmit and receive within the same SCI use a common baud rate, and each SCI module has a separate baud rate generator.

This SCI system offers many advanced features not commonly found on other asynchronous serial I/O peripherals on other embedded controllers. The receiver employs an advanced data sampling technique that ensures reliable communication and noise detection. Hardware parity, receiver wakeup, and double buffering on transmit and receive are also included.

**NOTE**

MC9S08JM16 series devices operate at a higher voltage range (2.7 V to 5.5 V) and do not include stop1 mode. Therefore, please disregard references to stop1.



NOTES:

1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ is enabled (IRQPE = 1). Pulldown is enabled if rising edge detect is selected (IRQEDG = 1)
3. IRQ does not have a clamp diode to $V_{DD}$. IRQ must not be driven above $V_{DD}$.
4. Pin contains integrated pullup device.
5. When pin functions as KBI (KBIPEn = 1) and associated pin is configured to enable the pullup device, KBEDGn can be used to reconfigure the pullup as a pulldown device.

**Figure 14-1. MC9S08JM16 Series Block Diagram Highlighting the SCI Blocks and Pins**

Table 17-12. ERRSTAT Field Descriptions (continued)

| Field | Description |
|---|---|
| 1<br>CRC5 | **CRC5 Interrupt Enable** — Setting this bit will enable CRC5 interrupts.<br>0 Interrupt disabled<br>1 Interrupt enabled |
| 0<br>PIDERR | **PIDERR Interrupt Enable** — Setting this bit will enable PIDERR interrupts.<br>0 Interrupt disabled<br>1 Interrupt enabled |

## 17.3.9 Status Register (STAT)

The STAT reports the transaction status within the USB module. When the MCU receives a TOKDNE interrupt, the STAT is read to determine the status of the previous endpoint communication. The data in the status register is valid only when the TOKDNEF interrupt flag is asserted. The STAT register is actually a read window into a status FIFO maintained by the USB module. When the USB module uses a BD, it updates the status register. If another USB transaction is performed before the TOKDNE interrupt is serviced, the USB module will store the status of the next transaction in the STAT FIFO. Thus, the STAT register is actually a four byte FIFO which allows the microcontroller to process one transaction while the serial interface engine (SIE) is processing the next. Clearing the TOKDNEF bit in the INTSTAT register causes the SIE to update the STAT register with the contents of the next STAT value. If the next data in the STAT FIFO holding register is valid, the SIE will immediately reassert the TOKDNE interrupt.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | ENDP[3:0] | | | | IN | ODD | 0 | 0 |
| W | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

▨ = Unimplemented or Reserved

Figure 17-12. Status Register (STAT)

Table 17-13. STAT Field Descriptions

| Field | Description |
|---|---|
| 7–4<br>ENDP[3:0] | **Endpoint Number** — These four bits encode the endpoint address that received or transmitted the previous token. This allows the microcontroller to determine which BDT entry was updated by the last USB transaction.<br>0000 Endpoint 0<br>0001 Endpoint 1<br>0010 Endpoint 2<br>0011 Endpoint 3<br>0100 Endpoint 4<br>0101 Endpoint 5<br>0110 Endpoint 6 |

# Chapter 18
# Development Support

## 18.1 Introduction

This chapter describes the single-wire background debug mode (BDM), which uses the on-chip background debug controller (BDC) module, and the independent on-chip real-time in-circuit emulation (ICE) system, which uses the on-chip debug (DBG) module.

### 18.1.1 Forcing Active Background

The method for forcing active background mode depends on the specific HCS08 derivative. For the MC9S08JM16 Series, you can force active background mode by holding the BKGD pin low as the MCU exits the reset condition independent of what caused the reset. If no debug pod is connected to the BKGD pin, the MCU will always reset into normal operating mode.

Figure 18-3 shows the host receiving a logic 1 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 perceived start of the bit time in the target MCU. The host holds the BKGD pin low long enough for the target to recognize it (at least two target BDC cycles). The host must release the low drive before the target MCU drives a brief active-high speedup pulse seven cycles after the perceived start of the bit time. The host must sample the bit level about 10 cycles after it started the bit time.

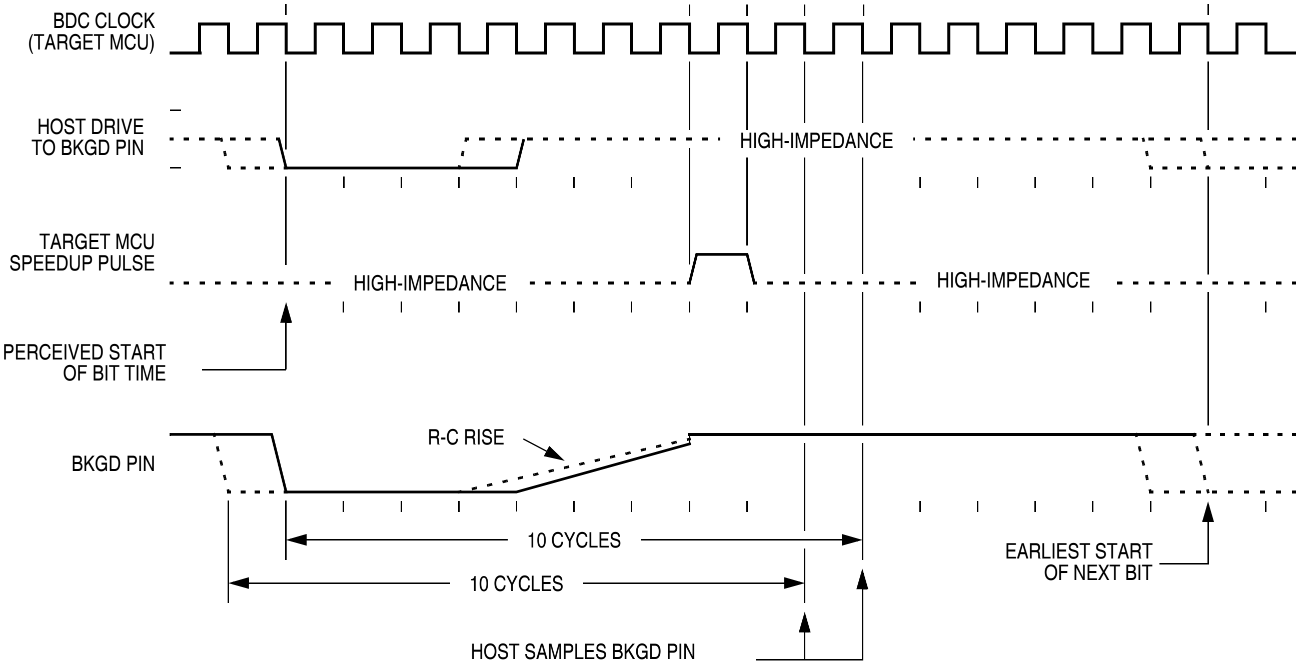


**Figure 18-3. BDC Target-to-Host Serial Bit Timing (Logic 1)**