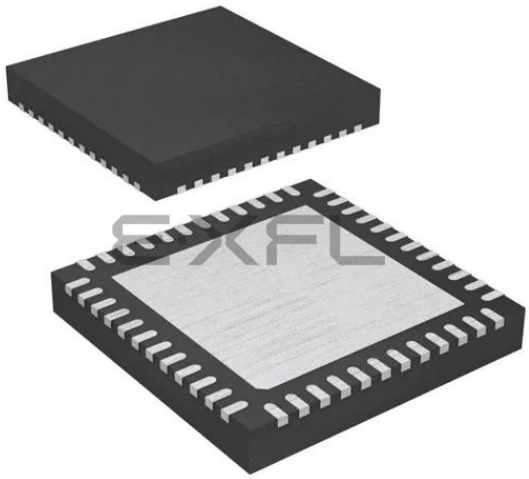

Welcome to E-XFL.COM

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	Obsolete
Core Processor	S08
Core Size	8-Bit
Speed	8MHz
Connectivity	SCI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	39
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	Surface Mount
Package / Case	48-VFQFN Exposed Pad
Supplier Device Package	48-QFN-EP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s08re16fder

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

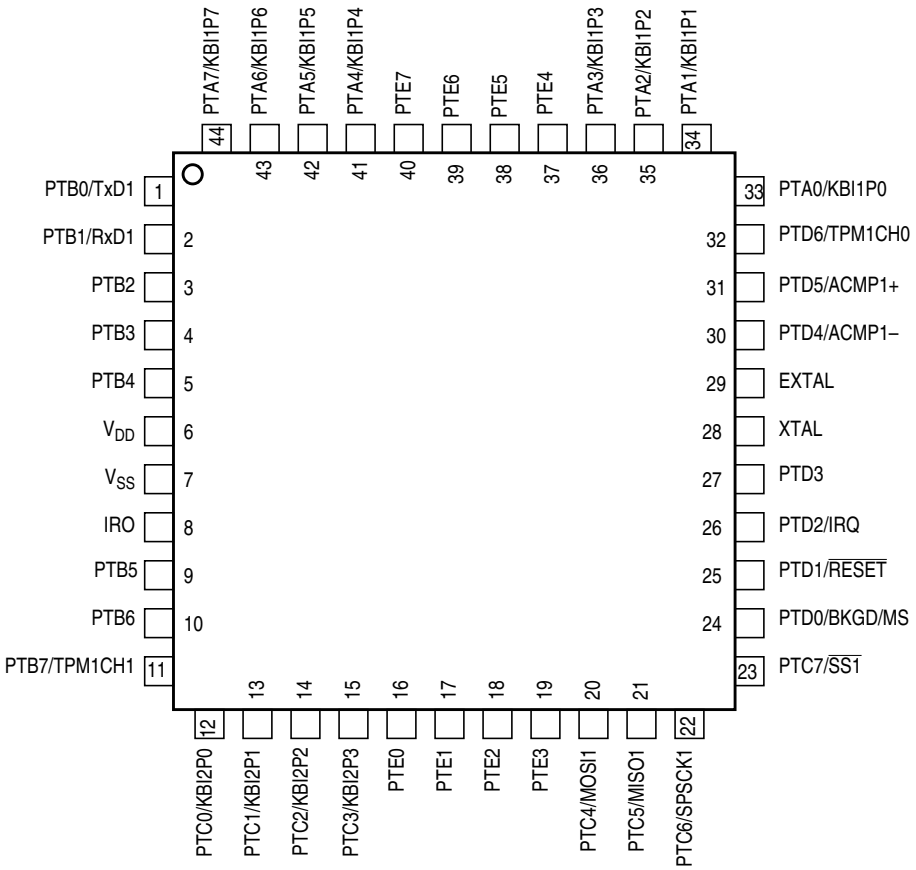

Figure 2-1. MC9S08RC/RD/RE/RG in 44-Pin LQFP Package

4.4.3 Program and Erase Command Execution

The steps for executing any of the commands are listed below. The FCDIV register must be initialized and any error flags cleared before beginning command execution. The command execution steps are:

1. Write a data value to an address in the FLASH array. The address and data information from this write is latched into the FLASH interface. This write is a required first step in any command sequence. For erase and blank check commands, the value of the data is not important. For page erase commands, the address may be any address in the 512-byte page of FLASH to be erased. For mass erase and blank check commands, the address can be any address in the FLASH memory. Whole pages of 512 bytes are the smallest block of FLASH that may be erased. In the 60K version, there are two instances where the size of a block that is accessible to the user is less than 512 bytes: the first page following RAM, and the first page following the high page registers. These pages are overlapped by the RAM and high page registers respectively.

   NOTE

   Do not program any byte in the FLASH more than once after a successful erase operation. Reprogramming bits to a byte which is already programmed is not allowed without first erasing the page in which the byte resides or mass erasing the entire FLASH memory. Programming without first erasing may disturb data stored in the FLASH.

2. Write the command code for the desired command to FCMD. The five valid commands are blank check ($05), byte program ($20), burst program ($25), page erase ($40), and mass erase ($41). The command code is latched into the command buffer.
3. Write a 1 to the FCBEF bit in FSTAT to clear FCBEF and launch the command (including its address and data information).

A partial command sequence can be aborted manually by writing a 0 to FCBEF any time after the write to the memory array and before writing the 1 that clears FCBEF and launches the complete command. Aborting a command in this way sets the FACCERR access error flag, which must be cleared before starting a new command.

A strictly monitored procedure must be adhered to, or the command will not be accepted. This minimizes the possibility of any unintended changes to the FLASH memory contents. The command complete flag (FCCF) indicates when a command is complete. The command sequence must be completed by clearing FCBEF to launch the command. Figure 4-3 is a flowchart for executing all of the commands except for burst programming. The FCDIV register must be initialized before using any FLASH commands. This must be done only once following a reset.

- 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.

Table 7-2. HCS08 Instruction Set Summary (Sheet 3 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
BRCLR *n,opr8a,rel*	Branch if Bit *n* in Memory Clear	Branch if (Mn) = 0	–	–	–	–	–		DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRN *rel*	Branch Never	Uses 3 Bus Cycles	–	–	–	–	–	–	REL	21	rr	3
BRSET *n,opr8a,rel*	Branch if Bit *n* in Memory Set	Branch if (Mn) = 1	–	–	–	–	–		DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BSET *n,opr8a*	Set Bit *n* in Memory	Mn ← 1	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BSR *rel*	Branch to Subroutine	PC ← (PC) + 0x0002 push (PCL); SP ← (SP) – 0x0001 push (PCH); SP ← (SP) – 0x0001 PC ← (PC) + *rel*	–	–	–	–	–	–	REL	AD	rr	5
CBEQ *oprx8,rel* CBEQA *#opr8i,rel* CBEQX *#opr8i,rel* CBEQ *oprx8*,X+,*rel* CBEQ ,X+,*rel* CBEQ *oprx8*,SP,*rel*	Compare and Branch if Equal	Branch if (A) = (M) Branch if (A) = (M) Branch if (X) = (M) Branch if (A) = (M) Branch if (A) = (M) Branch if (A) = (M)	–	–	–	–	–	–	DIR IMM IMM IX1+ IX+ SP1	31 41 51 61 71 9E61	dd rr ii rr ii rr ff rr rr ff rr	5 4 4 5 5 6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask Bit	I ← 0	–	–	0	–	–	–	INH	9A		1
CLR *oprx8*,X CLR ,X CLR *oprx8*,SP	Clear	M ← 0x00 A ← 0x00 X ← 0x00 H ← 0x00 M ← 0x00 M ← 0x00 M ← 0x00	0	–	–	0	1	–	DIR INH INH INH IX1 IX SP1	3F 4F 5F 8C 6F 7F 9E6F	dd ff ff	5 1 1 1 5 4 6
CMP *#opr8i* CMP *opr8a* CMP *opr16a* CMP *oprx16*,X CMP *oprx8*,X CMP ,X CMP *oprx16*,SP CMP *oprx8*,SP	Compare Accumulator with Memory	(A) – (M) (CCR Updated But Operands Not Changed)		–	–				IMM DIR EXT IX2 IX1 IX SP2 SP1	A1 B1 C1 D1 E1 F1 9ED1 9EE1	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
COM *oprx8*,X COM ,X COM *oprx8*,SP	Complement (One's Complement)	M ← (M)= 0xFF – (M) A ← (A) = 0xFF – (A) X ← (X) = 0xFF – (X) M ← (M) = 0xFF – (M) M ← (M) = 0xFF – (M) M ← (M) = 0xFF – (M)	0	–	–			1	DIR INH INH IX1 IX SP1	33 43 53 63 73 9E63	dd ff ff	5 1 1 5 4 6
CPHX *opr16a* CPHX *#opr16i* CPHX *opr8a* CPHX *oprx8*,SP	Compare Index Register (H:X) with Memory	(H:X) – (M:M + 0x0001) (CCR Updated But Operands Not Changed)		–	–				EXT IMM DIR SP1	3E 65 75 9EF3	hh ll jj kk dd ff	6 3 5 6

The duty cycle of the carrier signal is controlled by varying the ratio of high time to low + high time. As the input clock period is fixed, the duty cycle resolution will be proportional to the number of counts required to generate the desired carrier period.

$$\text{Duty Cycle} = \frac{\text{Highcount}}{\text{Highcount} + \text{Lowcount}} \qquad \textit{Eqn. 8-4}$$

8.5.2 Modulator

The modulator has three main modes of operation:

- Gate the carrier onto the modulator output (time mode)
- Control the logic level of the modulator output (baseband mode)
- Count carrier periods and instruct the carrier generator to alternate between two carrier frequencies whenever a modulation period (mark + space counts) expires (FSK mode)

The modulator includes a 17-bit down counter with underflow detection. The counter is loaded from the 16-bit modulation mark period buffer registers, CMTCMD1 and CMTCMD2. The most significant bit is loaded with a logic zero and serves as a sign bit. When the counter holds a positive value, the modulator gate is open and the carrier signal is driven to the transmitter block.

When the counter underflows, the modulator gate is closed and a 16-bit comparator is enabled that compares the logical complement of the value of the down-counter with the contents of the modulation space period register (which has been loaded from the registers CMTCMD3 and CMTCMD4).

When a match is obtained the cycle repeats by opening the modulator gate, reloading the counter with the contents of CMTCMD1 and CMTCMD2, and reloading the modulation space period register with the contents of CMTCMD3 and CMTCMD4.

If the contents of the modulation space period register are all zeroes, the match will be immediate and no space period will be generated (for instance, for FSK protocols that require successive bursts of different frequencies).

The MCGEN bit in the CMTMSC register must be set to enable the modulator timer.

9.2 KBI Block Diagram

Figure 9-2 shows the block diagram for a KBI module.

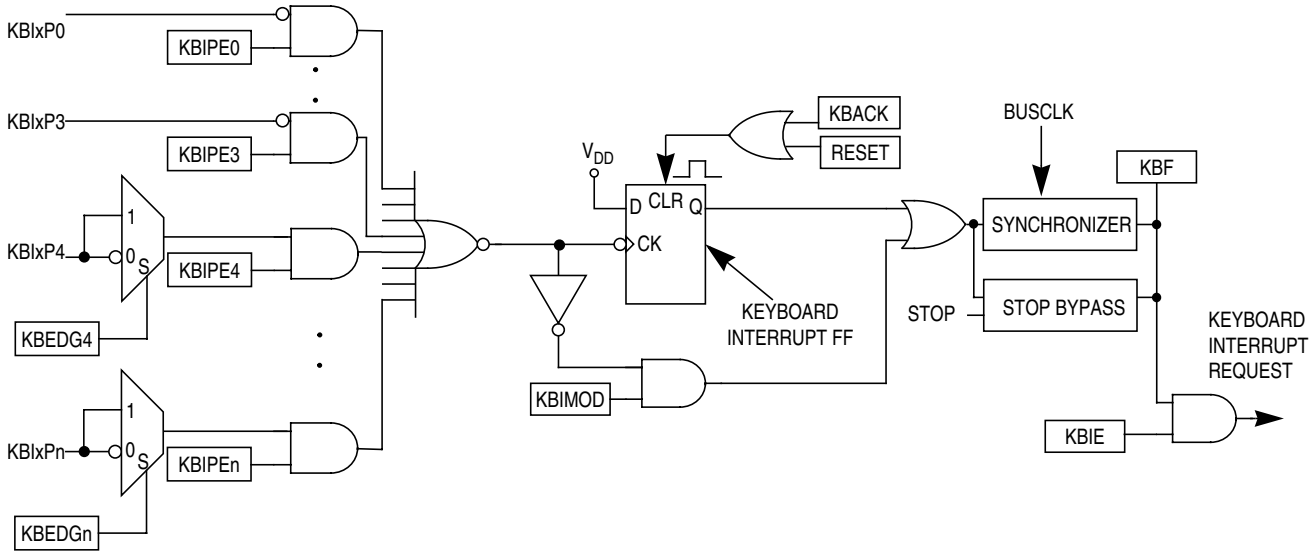

Figure 9-2. KBI Block Diagram

The KBI module allows up to eight pins to act as additional interrupt sources. Four of these pins allow falling-edge sensing while the other four can be configured for either rising-edge sensing or falling-edge sensing. The sensing mode for all eight pins can also be modified to detect edges and levels instead of only edges.

9.3 Keyboard Interrupt (KBI) Module

This on-chip peripheral module is called a keyboard interrupt (KBI) module because originally it was designed to simplify the connection and use of row-column matrices of keyboard switches. However, these inputs are also useful as extra external interrupt inputs and as an external means of waking up the MCU from stop or wait low-power modes.

9.3.1 Pin Enables

The KBIPEn control bits in the KBIxPE register allow a user to enable (KBIPEn = 1) any combination of KBI-related port pins to be connected to the KBI module. Pins corresponding to 0s in KBIxPE are general-purpose I/O pins that are not associated with the KBI module.

9.3.2 Edge and Level Sensitivity

Synchronous logic is used to detect edges. Prior to detecting an edge, enabled keyboard inputs in a KBI module must be at the deasserted logic level.

A falling edge is detected when an enabled keyboard input signal is seen as a logic 1 (the deasserted level) during one bus cycle and then a logic 0 (the asserted level) during the next cycle.

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.

9.4.1 KBI x Status and Control Register (KBIxSC)

	7	6	5	4	3	2	1	0
R	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBF	0	KBIE	KBIMOD
W						KBACK		
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-3. KBI x Status and Control Register (KBIxSC)

Table 9-1. KBIxSC Field Descriptions

Field	Description
7:4 KBEDG[7:4]	**Keyboard Edge Select for KBI Port Bits** — Each of these read/write bits selects the polarity of the edges and/or levels that are recognized as trigger events on the corresponding KBI port pin when it is configured as a keyboard interrupt input (KBIPEn = 1). Also see the KBIMOD control bit, which determines whether the pin is sensitive to edges-only or edges and levels. 0 Falling edges/low levels. 1 Rising edges/high levels.
3 KBF	**Keyboard Interrupt Flag** — This read-only status flag is set whenever the selected edge event has been detected on any of the enabled KBI port pins. This flag is cleared by writing a logic 1 to the KBACK control bit. The flag will remain set if KBIMOD = 1 to select edge-and-level operation and any enabled KBI port pin remains at the asserted level. 0 No KBI interrupt pending. 1 KBI interrupt pending. KBF can be used as a software pollable flag (KBIE = 0) or it can generate a hardware interrupt request to the CPU (KBIE = 1). KBF must be cleared before entering stop mode.
2 KBACK	**Keyboard Interrupt Acknowledge** — This write-only bit (reads always return 0) is used to clear the KBF status flag by writing a logic 1 to KBACK. When KBIMOD = 1 to select edge-and-level operation and any enabled KBI port pin remains at the asserted level, KBF is being continuously set so writing 1 to KBACK does not clear the KBF flag.
1 KBIE	**Keyboard Interrupt Enable** — This read/write control bit determines whether hardware interrupts are generated when the KBF status flag equals 1. When KBIE = 0, no hardware interrupts are generated, but KBF can still be used for software polling. 0 KBF does not generate hardware interrupts (use polling). 1 KBI hardware interrupt requested when KBF = 1.
0 KBIMOD	**Keyboard Detection Mode** — This read/write control bit selects either edge-only detection or edge-and-level detection. KBI port bits 3 through 0 can detect falling edges-only or falling edges and low levels. KBI port bits 7 through 4 can be configured to detect either: • Rising edges-only or rising edges and high levels (KBEDGn = 1) • Falling edges-only or falling edges and low levels (KBEDGn = 0) 0 Edge-only detection. 1 Edge-and-level detection.

9.4.2 KBI x Pin Enable Register (KBIxPE)

	7	6	5	4	3	2	1	0
R W	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
Reset	0	0	0	0	0	0	0	0

Figure 9-4. KBI x Pin Enable Register (KBIxPE)

Table 9-2. KBIxPE Field Descriptions

Field	Description
7:0 KBIPE[7:0]	**Keyboard Pin Enable for KBI Port Bits** — Each of these read/write bits selects whether the associated KBI port pin is enabled as a keyboard interrupt input or functions as a general-purpose I/O pin. 0 Bit n of KBI port is a general-purpose I/O pin not associated with the KBI. 1 Bit n of KBI port enabled as a keyboard interrupt input

Table 12-4. SCI1C2 Register Field Descriptions (continued)

Field	Description
1 RWU	**Receiver Wakeup Control** — This bit can be written to 1 to place the SCI receiver in a standby state where it waits for automatic hardware detection of a selected wakeup condition. The wakeup condition is either an idle line between messages (WAKE = 0, idle-line wakeup), or a logic 1 in the most significant data bit in a character (WAKE = 1, address-mark wakeup). Application software sets RWU and (normally) a selected hardware condition automatically clears RWU. Refer to Section 12.3.3.2, "Receiver Wakeup Operation," for more details. 0 Normal SCI receiver operation. 1 SCI receiver in standby waiting for wakeup condition.
0 SBK	**Send Break** — Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 or 11 bit times of logic 0 are queued as long as SBK = 1. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK. Refer to Section 12.3.2.1, "Send Break and Queued Idle," for more details. 0 Normal transmitter operation. 1 Queue break character(s) to be sent.

12.2.4 SCI Status Register 1 (SCI1S1)

This register has eight read-only status flags. Writes have no effect. Special software sequences (which do not involve writing to this register) are used to clear these status flags.

	7	6	5	4	3	2	1	0
R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W								
Reset	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-7. SCI Status Register 1 (SCI1S1)

Table 12-5. SCI1S1 Register Field Descriptions

Field	Description
7 TDRE	**Transmit Data Register Empty Flag** — TDRE is set immediately after reset and when a transmit data value transfers from the transmit data buffer to the transmit shifter, leaving room for a new character in the buffer. To clear TDRE, read SCI1S1 with TDRE = 1 and then write to the SCI data register (SCI1D). 0 Transmit data register (buffer) full. 1 Transmit data register (buffer) empty.
6 TC	**Transmission Complete Flag** — TC is set immediately after reset and when TDRE = 1 and no data, preamble, or break character is being transmitted. 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete). TC is cleared automatically by reading SCI1S1 with TC = 1 and then doing one of the following three things: • Write to the SCI data register (SCI1D) to transmit new data • Queue a preamble by changing TE from 0 to 1 • Queue a break character by writing 1 to SBK in SCI1C2

12.3 Functional Description

The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The SCI comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. During normal operation, the MCU monitors the status of the SCI, writes the data to be transmitted, and processes received data. The following describes each of the blocks of the SCI.

12.3.1 Baud Rate Generation

As shown in Figure 12-11, the clock source for the SCI baud rate generator is the bus-rate clock.

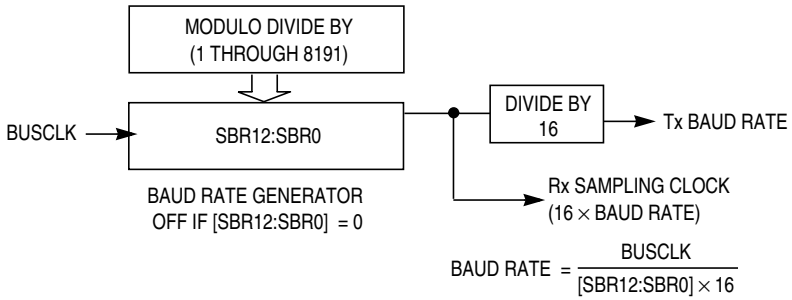

Figure 12-11. SCI Baud Rate Generation

SCI communications require the transmitter and receiver (which typically derive baud rates from independent clock sources) to use the same baud rate. Allowed tolerance on this baud frequency depends on the details of how the receiver synchronizes to the leading edge of the start bit and how bit sampling is performed.

The MCU resynchronizes to bit boundaries on every high-to-low transition, but in the worst case, there are no such transitions in the full 10- or 11-bit time character frame so any mismatch in baud rate is accumulated for the whole character time. For a Freescale Semiconductor SCI system whose bus frequency is driven by a crystal, the allowed baud rate mismatch is about ±4.5 percent for 8-bit data format and about ±4 percent for 9-bit data format. Although baud rate modulo divider settings do not always produce baud rates that exactly match standard rates, it is normally possible to get within a few percent, which is acceptable for reliable communications.

12.3.2 Transmitter Functional Description

This section describes the overall block diagram for the SCI transmitter (Figure 12-1), as well as specialized functions for sending break and idle characters.

The transmitter is enabled by setting the TE bit in SCI1C2. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the SCI data register (SCI1D).

The central element of the SCI transmitter is the transmit shift register that is either 10 or 11 bits long depending on the setting in the M control bit. For the remainder of this section, we will assume M = 0, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new SCI character, the value waiting in

has one full character time after RDRF is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (RDRF = 1), it gets the data from the receive data register by reading SCI1D. The RDRF flag is cleared automatically by a 2-step sequence which is normally satisfied in the course of the user's program that handles receive data. Refer to Section 12.3.4, "Interrupts and Status Flags," for more details about flag clearing.

12.3.3.1 Data Sampling Technique

The SCI receiver uses a 16× baud rate clock for sampling. The receiver starts by taking logic level samples at 16 times the baud rate to search for a falling edge on the RxD1 serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The 16× baud rate clock is used to divide the bit time into 16 segments labeled RT1 through RT16. When a falling edge is located, three more samples are taken at RT3, RT5, and RT7 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a receive character.

The receiver then samples each bit time, including the start and stop bits, at RT8, RT9, and RT10 to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. In the case of the start bit, the bit is assumed to be 0 if at least two of the samples at RT3, RT5, and RT7 are 0 even if one or all of the samples taken at RT8, RT9, and RT10 are 1s. If any sample in any bit time (including the start and stop bits) in a character frame fails to agree with the logic level for that bit, the noise flag (NF) will be set when the received character is transferred to the receive data buffer.

The falling edge detection logic continuously looks for falling edges, and if an edge is detected, the sample clock is resynchronized to bit times. This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

In the case of a framing error, provided the received character was not a break character, the sampling logic that searches for a falling edge is filled with three logic 1 samples so that a new start bit can be detected almost immediately.

In the case of a framing error, the receiver is inhibited from receiving any new characters until the framing error flag is cleared. The receive shift register continues to function, but a complete character cannot transfer to the receive data buffer if FE is still set.

12.3.3.2 Receiver Wakeup Operation

Receiver wakeup is a hardware mechanism that allows an SCI receiver to ignore the characters in a message that is intended for a different SCI receiver. In such a system, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up (RWU) control bit in SCI1C2. When RWU = 1, it inhibits setting of the status flags associated with the receiver, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force RWU to 0 so all receivers wake up in time to look at the first character(s) of the next message.

12.3.3.2.1 Idle-Line Wakeup

When WAKE = 0, the receiver is configured for idle-line wakeup. In this mode, RWU is cleared automatically when the receiver detects a full character time of the idle-line level. The M control bit selects 8-bit or 9-bit data mode that determines how many bit times of idle are needed to constitute a full character time (10 or 11 bit times because of the start and stop bits). The idle-line type (ILT) control bit selects one of two ways to detect an idle line:

- When ILT = 0, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle.
- When ILT = 1, the idle bit counter doesn't start until after a stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

12.3.3.2.2 Address-Mark Wakeup

When WAKE = 1, the receiver is configured for address-mark wakeup. In this mode, RWU is cleared automatically when the receiver detects a logic 1 in the most significant bit of a received character (eighth bit in M = 0 mode and ninth bit in M = 1 mode).

12.3.4 Interrupts and Status Flags

The SCI system has three separate interrupt vectors to reduce the amount of software needed to isolate the cause of the interrupt. One interrupt vector is associated with the transmitter for TDRE and TC events. Another interrupt vector is associated with the receiver for RDRF and IDLE events, and a third vector is used for OR, NF, FE, and PF error conditions. Each of these eight interrupt sources can be separately masked by local interrupt enable masks. The flags can still be polled by software when the local masks are cleared to disable generation of hardware interrupt requests.

The SCI transmitter has two status flags that optionally can generate hardware interrupt requests. Transmit data register empty (TDRE) indicates when there is room in the transmit data buffer to write another transmit character to SCI1D. If the transmit interrupt enable (TIE) bit is set, a hardware interrupt will be requested whenever TDRE = 1. Transmit complete (TC) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TxD1 high. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (TCIE) bit is set, a hardware interrupt will be requested whenever TC = 1. Instead of hardware interrupts, software polling may be used to monitor the TDRE and TC status flags if the corresponding TIE or TCIE local interrupt masks are 0s.

When a program detects that the receive data register is full (RDRF = 1), it gets the data from the receive data register by reading SCI1D. The RDRF flag is cleared by reading SCI1S1 while RDRF = 1 and then reading SCI1D.

When polling is used, this sequence is naturally satisfied in the normal course of the user program. If hardware interrupts are used, SCI1S1 must be read in the interrupt service routine (ISR). Normally, this is done in the ISR anyway to check for receive errors, so the sequence is automatically satisfied.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the RxD1 line remains idle for an extended period of time. IDLE is cleared by reading SCI1S1 while IDLE = 1 and then reading

When CPHA = 0, the slave begins to drive its MISO output with the first data bit value (MSB or LSB depending on LSBFE) when $\overline{SS1}$ goes to active low. The first SPSCK edge causes both the master and the slave to sample the data bit values on their MISO and MOSI inputs, respectively. At the second SPSCK edge, the SPI shifter shifts one bit position which shifts in the bit value that was just sampled and shifts the second data bit value out the other end of the shifter to the MOSI and MISO outputs of the master and slave, respectively. When CPHA = 0, the slave's $\overline{SS}$ input must go to its inactive high level between transfers.

13.3.2 SPI Pin Controls

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

13.3.2.1 SPSCK1 — SPI Serial Clock

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

13.3.2.2 MOSI1 — Master Data Out, Slave Data In

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

13.3.2.3 MISO1 — Master Data In, Slave Data Out

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

13.3.2.4 $\overline{SS1}$ — Slave Select

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

13.4.5 SPI Data Register (SPI1D)

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

Figure 13-11. SPI Data Register (SPI1D)

Reads of this register return the data read from the receive data buffer. Writes to this register write data to the transmit data buffer. When the SPI is configured as a master, writing data to the transmit data buffer initiates an SPI transfer.

Data should not be written to the transmit data buffer unless the SPI transmit buffer empty flag (SPTEF) is set, indicating there is room in the transmit buffer to queue a new transmit byte.

Data may be read from SPI1D any time after SPRF is set and before another transfer is finished. Failure to read the data out of the receive data buffer before a new transfer ends causes a receive overrun condition and the data from the new transfer is lost.

BDC serial communications use a custom serial protocol first introduced on the M68HC12 Family of microcontrollers. This protocol assumes the host knows the communication clock rate that is determined by the target BDC clock rate. All communication is initiated and controlled by the host that drives a high-to-low edge to signal the beginning of each bit time. Commands and data are sent most significant bit first (MSB first). For a detailed description of the communications protocol, refer to Section 15.2.2, "Communication Details."

If a host is attempting to communicate with a target MCU that has an unknown BDC clock rate, a SYNC command may be sent to the target MCU to request a timed sync response signal from which the host can determine the correct communication speed.

BKGD is a pseudo-open-drain pin and there is an on-chip pullup so no external pullup resistor is required. Unlike typical open-drain pins, the external RC time constant on this pin, which is influenced by external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speedup pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Section 15.2.2, "Communication Details," for more detail.

When no debugger pod is connected to the 6-pin BDM interface connector, the internal pullup on BKGD chooses normal operating mode. When a development system is connected, it can pull both BKGD and $\overline{\text{RESET}}$ low, release $\overline{\text{RESET}}$ to select active background mode rather than normal operating mode, then release BKGD. It is not necessary to reset the target MCU to communicate with it through the background debug interface.

15.2.2 Communication Details

The BDC serial interface requires the external controller to generate a falling edge on the BKGD pin to indicate the start of each bit time. The external controller provides this falling edge whether data is transmitted or received.

BKGD is a pseudo-open-drain pin that can be driven either by an external controller or by the MCU. Data is transferred MSB first at 16 BDC clock cycles per bit (nominal speed). The interface times out if 512 BDC clock cycles occur between falling edges from the host. Any BDC command that was in progress when this timeout occurs is aborted without affecting the memory or operating mode of the target MCU system.

The custom serial protocol requires the debug pod to know the target BDC communication clock speed.

The clock switch (CLKSW) control bit in the BDC status and control register allows the user to select the BDC clock source. The BDC clock source can either be the bus or the alternate BDC clock source.

The BKGD pin can receive a high or low level or transmit a high or low level. The following diagrams show timing for each of these cases. Interface timing is synchronous to clocks in the target BDC, but asynchronous to the external host. The internal BDC clock signal is shown for reference in counting cycles.

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

A-Only — Trigger when the address matches the value in comparator A

A OR B — Trigger when the address matches either the value in comparator A or the value in comparator B

A Then B — Trigger when the address matches the value in comparator B but only after the address for another cycle matched the value in comparator A. There can be any number of cycles after the A match and before the B match.

A AND B Data (Full Mode) — This is called a full mode because address, data, and R/W (optionally) must match within the same bus cycle to cause a trigger event. Comparator A checks address, the low byte of comparator B checks data, and R/W is checked against RWA if RWAEN = 1. The high-order half of comparator B is not used.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

A AND NOT B Data (Full Mode) — Address must match comparator A, data must not match the low half of comparator B, and R/W must match RWA if RWAEN = 1. All three conditions must be met within the same bus cycle to cause a trigger.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

Event-Only B (Store Data) — Trigger events occur each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

A Then Event-Only B (Store Data) — After the address has matched the value in comparator A, a trigger event occurs each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

Inside Range (A ≤ Address ≤ B) — A trigger occurs when the address is greater than or equal to the value in comparator A and less than or equal to the value in comparator B at the same time.

Outside Range (Address < A or Address > B) — A trigger occurs when the address is either less than the value in comparator A or greater than the value in comparator B.

A.4 Electrostatic Discharge (ESD) Protection Characteristics

Although damage from static discharge is much less common on these devices than on early CMOS circuits, normal handling precautions should be used to avoid exposure to static discharge. Qualification tests are performed to ensure that these devices can withstand exposure to reasonable levels of static without suffering any permanent damage. All ESD testing is in conformity with CDF-AEC-Q00 Stress Test Qualification for Automotive Grade Integrated Circuits. (http://www.aecouncil.com/) A device is considered to have failed if, after exposure to ESD pulses, the device no longer meets the device specification requirements. Complete dc parametric and functional testing is performed per the applicable device data sheet at room temperature followed by hot temperature, unless specified otherwise in the device data sheet.

Table A-3. ESD Protection Characteristics

Parameter	Symbol	Value	Unit
ESD Target for Machine Model (MM) MM circuit description	V_{THMM}	200	V
ESD Target for Human Body Model (HBM) HBM circuit description	V_{THHBM}	2000	V

A.5 DC Characteristics

This section includes information about power supply requirements, I/O pin characteristics, and power supply current in various operating modes.

Table A-4. DC Characteristics (Temperature Range = 0 to 70°C Ambient)

Parameter	Symbol	Min	Typical	Max	Unit
Low-voltage detection threshold	V_{LVD}	1.82	1.875	1.90	V
Power on reset (POR) voltage	V_{POR}	0.8	0.9	1.1	V
Maximum low-voltage safe state re-arm[1]	V_{REARM}	1.90	2.24	2.60	V

1. If SAFE bit is set, V_{DD} must be above re-arm voltage to allow MCU to accept interrupts, refer to Section 5.6, “Low-Voltage Detect (LVD) System.”

Table A-5. DC Characteristics (Temperature Range = –40 to 85°C Ambient)

Parameter	Symbol	Min	Typical	Max	Unit
Supply voltage (run, wait and stop modes.) $0 < f_{Bus} < 8$ MHz	V_{DD}	1.8		3.6	V
Minimum RAM retention supply voltage applied to V_{DD}	V_{RAM}	V_{POR}[1], [2]		—	V
Low-voltage detection threshold (V_{DD} falling) (V_{DD} rising)	V_{LVD}	 1.82 1.92	 1.88 1.96	 1.93 2.01	V
Low-voltage warning threshold (V_{DD} falling) (V_{DD} rising)	V_{LVW}	 2.07 2.16	 2.13 2.21	 2.18 2.26	V
Power on reset (POR) voltage	V_{POR}	0.85	1.0	1.2	V

Table A-5. DC Characteristics (Temperature Range = –40 to 85°C Ambient) (continued)

Parameter	Symbol	Min	Typical	Max	Unit
Maximum low-voltage safe state re-arm[3]	V_{REARM}	—	—	3.0	V
Input high voltage (V_{DD} > 2.3 V) (all digital inputs)	V_{IH}	0.70 × V_{DD}		—	V
Input high voltage (1.8 V ≤ V_{DD} ≤ 2.3 V) (all digital inputs)	V_{IH}	0.85 × V_{DD}		—	V
Input low voltage (V_{DD} > 2.3 V) (all digital inputs)	V_{IL}	—		0.35 × V_{DD}	V
Input low voltage (1.8 V ≤ V_{DD} ≤ 2.3 V) (all digital inputs)	V_{IL}	—		0.30 × V_{DD}	V
Input hysteresis (all digital inputs)	V_{hys}	0.06 × V_{DD}		—	V
Input leakage current (Per pin) V_{In} = V_{DD} or V_{SS}, all input only pins	$\lvert I_{In}\rvert$	—	0.025	1.0	µA
High impedance (off-state) leakage current (per pin) V_{In} = V_{DD} or V_{SS}, all input/output	$\lvert I_{OZ}\rvert$	—	0.025	1.0	µA
Internal pullup resistors[4] [5]	R_{PU}	17.5		52.5	κW
Internal pulldown resistor (IRQ)	R_{PD}	17.5		52.5	κW
Output high voltage (V_{DD} ≥ 1.8 V) I_{OH} = –2 mA (ports A, C, D and E)	V_{OH}	V_{DD} – 0.5		—	V
Output high voltage (port B and IRO) I_{OH} = –10 mA (V_{DD} ≥ 2.7 V) I_{OH} = –6 mA (V_{DD} ≥ 2.3 V) I_{OH} = –3 mA (V_{DD} ≥ 1.8 V)		V_{DD} – 0.5		— — —	
Maximum total I_{OH} for all port pins	$\lvert I_{OHT}\rvert$	—		60	mA
Output low voltage (V_{DD} ≥ 1.8 V) I_{OL} = 2.0 mA (ports A, C, D and E)	V_{OL}	—		0.5	V
Output low voltage (port B) I_{OL} = 10.0 mA (V_{DD} ≥ 2.7 V) I_{OL} = 6 mA (V_{DD} ≥ 2.3 V) I_{OL} = 3 mA (V_{DD} ≥ 1.8 V)		— — —		0.5 0.5 0.5	
Output low voltage (IRO) I_{OL} = 16 mA (V_{DD} ≥ 2.7 V) I_{OL} = 6 mA (V_{DD} ≥ 2.3 V) I_{OL} = 3 mA (V_{DD} ≥ 1.8 V)		— — —		1.2 1.2 1.2	
Maximum total I_{OL} for all port pins	I_{OLT}	—		60	mA
dc injection current[2], [6], [7], [8],, [9] V_{IN} < V_{SS}, V_{IN} > V_{DD} Single pin limit Total MCU limit, includes sum of all stressed pins	$\lvert I_{IC}\rvert$	 — —		 0.2 5	 mA mA
Input capacitance (all non-supply pins)	C_{In}	—		7	pF

1. RAM will retain data down to POR voltage. RAM data not guaranteed to be valid following a POR.
2. This parameter is characterized and not tested on each device.
3. If SAFE bit is set, V_{DD} must be above re-arm voltage to allow MCU to accept interrupts, refer to Section 5.6, “Low-Voltage Detect (LVD) System.”
4. Measurement condition for pull resistors: V_{In} = V_{SS} for pullup and V_{In} = V_{DD} for pulldown.
5. The PTA0 pullup resistor may not pull up to the specified minimum V_{IH}. However, all ports are functionally tested to guarantee that a logic 1 will be read on any port input when the pullup is enabled and no dc load is present on the pin.
6. All functional non-supply pins are internally clamped to V_{SS} and V_{DD}.