

Welcome to [E-XFL.COM](http://E-XFL.COM)

### What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

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

#### Details

Product Status	Active
Core Processor	S08
Core Size	8-Bit
Speed	40MHz
Connectivity	I <sup>2</sup> C, LINbus, SCI, SPI
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)	2.7V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	<a href="https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=mc9s08sh16cwlr">https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=mc9s08sh16cwlr</a>



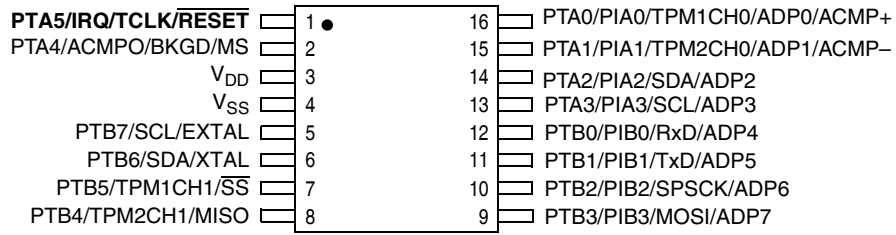


Figure 2-3. 16-Pin TSSOP

Most background commands are not available in stop mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from stop and enter active background mode if the ENBDM bit is set. After entering background debug mode, all background commands are available.

### 3.6.2 Stop2 Mode

Stop2 mode is entered by executing a STOP instruction under the conditions as shown in [Table 3-1](#). Most of the internal circuitry of the MCU is powered off in stop2 with the exception of the RAM. Upon entering stop2, all I/O pin control signals are latched so that the pins retain their states during stop2.

Exit from stop2 is performed by asserting the wake-up pin (PTA5/IRQ/TCLK/ $\overline{\text{RESET}}$ ) on the MCU.

In addition, the real-time counter (RTC) can wake the MCU from stop2, if enabled.

Upon wake-up from stop2 mode, the MCU starts up as from a power-on reset (POR):

- All module control and status registers are reset
- The LVD reset function is enabled and the MCU remains in the reset state if  $V_{DD}$  is below the LVD trip point (low trip point selected due to POR)
- The CPU takes the reset vector

In addition to the above, upon waking up from stop2, the PPDF bit in SPMSC2 is set. This flag is used to direct user code to go to a stop2 recovery routine. PPDF remains set and the I/O pin states remain latched until a 1 is written to PPDACK in SPMSC2.

To maintain I/O states for pins that were configured as general-purpose I/O before entering stop2, the user must restore the contents of the I/O port registers, which have been saved in RAM, to the port registers before writing to the PPDACK bit. If the port registers are not restored from RAM before writing to PPDACK, then the pins will switch to their reset states when PPDACK is written.

For pins that were configured as peripheral I/O, the user must reconfigure the peripheral module that interfaces to the pin before writing to the PPDACK bit. If the peripheral module is not enabled before writing to PPDACK, the pins will be controlled by their associated port control registers when the I/O latches are opened.

### 3.6.3 On-Chip Peripheral Modules in Stop Modes

When the MCU enters any stop mode, system clocks to the internal peripheral modules are stopped. Even in the exception case ( $\text{ENBDM} = 1$ ), where clocks to the background debug logic continue to operate, clocks to the peripheral systems are halted to reduce power consumption. Refer to [Section 3.6.2, “Stop2 Mode,”](#) and [Section 3.6.1, “Stop3 Mode,”](#) for specific information on system behavior in stop modes.

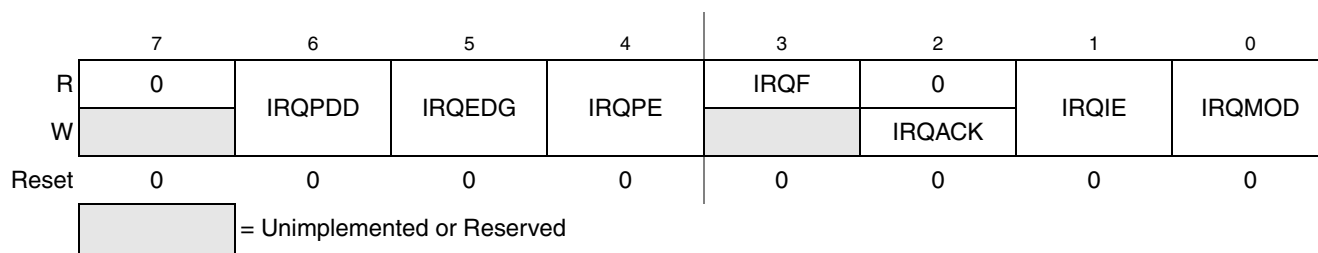
When an interrupt condition occurs, an associated flag bit becomes set. If the associated local interrupt enable is 1, an interrupt request is sent to the CPU. Within the CPU, if the global interrupt mask (I bit in the CCR) is 0, the CPU will finish the current instruction; stack the PCL, PCH, X, A, and CCR CPU registers; set the I bit; and then fetch the interrupt vector for the highest priority pending interrupt. Processing then continues in the interrupt service routine.

**Table 5-2. Vector Summary**

Vector Priority	Vector Number	Address (High/Low)	Vector Name	Module	Source	Enable	Description
Lowest  Highest	31	0xFFC0/0xFFC1	—	—	—	—	—
	30	0xFFC2/0xFFC3	Vacmp	ACMP	ACF	ACIE	Analog comparator
	29	0xFFC4/0xFFC5	—	—	—	—	—
	28	0xFFC6/0xFFC7	—	—	—	—	—
	27	0xFFC8/0xFFC9	—	—	—	—	—
	26	0xFFCA/0xFFCB	Vmtim	MTIM	TOF	TOIE	MTIM overflow
	25	0xFFCC/0xFFCD	Vrtc	RTC	RTIF	RTIE	Real-time interrupt
	24	0xFFCE/0xFFCF	Viic	IIC	IICIS	IICIE	IIC control
	23	0xFFD0/0xFFD1	Vadc	ADC	COCO	AIEN	ADC
	22	0xFFD2/0xFFD3	—	—	—	—	—
	21	0xFFD4/0xFFD5	Vportb	Port B	PTBIF	PTBIE	Port B Pins
	20	0xFFD6/0xFFD7	Vporta	Port A	PTAIF	PTAIE	Port A Pins
	19	0xFFD8/0xFFD9	—	—	—	—	—
	18	0xFFDA/0xFFDB	Vscitx	SCI	TDRE, TC	TIE, TCIE	SCI transmit
	17	0xFFDC/0xFFDD	Vscirx	SCI	IDLE, RDRF, LDBKDIF, RXEDGIF	ILIE, RIE, LBKDIE, RXEDGIE	SCI receive
	16	0xFFDE/0xFFDF	Vscierr	SCI	OR, NF, FE, PF	ORIE, NFIE, FEIE, PFIE	SCI error
	15	0xFFE0/0xFFE1	Vspi	SPI	SPIF, MODF, SPTEF	SPIE, SPIE, SPTIE	SPI
	14	0xFFE2/0xFFE3	Vtpm2ovf	TPM2	TOF	TOIE	TPM2 overflow
	13	0xFFE4/0xFFE5	Vtpm2ch1	TPM2	CH1F	CH1IE	TPM2 channel 1
	12	0xFFE6/0xFFE7	Vtpm2ch0	TPM2	CH0F	CH0IE	TPM2 channel 0
	11	0xFFE8/0xFFE9	Vtpm1ovf	TPM1	TOF	TOIE	TPM1 overflow
	10	0xFFEA/0xFFEB	—	—	—	—	—
	9	0xFFEC/0xFFED	—	—	—	—	—
	8	0xFFEE/0xFFEF	—	—	—	—	—
	7	0xFFFF0/0xFFFF1	—	—	—	—	—
	6	0xFFFF2/0xFFFF3	Vtpm1ch1	TPM1	CH1F	CH1IE	TPM1 channel 1
	5	0xFFFF4/0xFFFF5	Vtpm1ch0	TPM1	CH0F	CH0IE	TPM1 channel 0
	4	0xFFFF6/0xFFFF7	—	—	—	—	—
	3	0xFFFF8/0xFFFF9	Vlvd	System control	LVWF	LVWIE	Low-voltage warning
	2	0xFFFFA/0xFFFFB	Virq	IRQ	IRQF	IRQIE	IRQ pin
	1	0xFFFFC/0xFFFFD	Vswi	Core	SWI Instruction	—	Software interrupt
	0	0xFFFFE/0xFFFFF	Vreset	System control	COP, LVD, RESET pin, Illegal opcode, Illegal address	COPE, LVDRE, —, —	Watchdog timer, Low-voltage detect, External pin, Illegal opcode, Illegal address

### 5.7.1 Interrupt Pin Request Status and Control Register (IRQSC)

This direct page register includes status and control bits, which are used to configure the IRQ function, report status, and acknowledge IRQ events.



**Figure 5-2. Interrupt Request Status and Control Register (IRQSC)**

**Table 5-3. IRQSC Register Field Descriptions**

Field	Description
6 IRQPDD	<b>Interrupt Request (IRQ) Pull Device Disable</b> — This read/write control bit is used to disable the internal pullup device when the IRQ pin is enabled (IRQPE = 1) allowing for an external device to be used. 0 IRQ pull device enabled if IRQPE = 1. 1 IRQ pull device disabled if IRQPE = 1.
5 IRQEDG	<b>Interrupt Request (IRQ) Edge Select</b> — This read/write control bit is used to select the polarity of edges or levels on the IRQ pin that cause IRQF to be set. The IRQMOD control bit determines whether the IRQ pin is sensitive to both edges and levels or only edges. When the IRQ pin is enabled as the IRQ input and is configured to detect rising edges. When IRQEDG = 1 and the internal pull device is enabled, the pull-up device is reconfigured as an optional pull-down device. 0 IRQ is falling edge or falling edge/low-level sensitive. 1 IRQ is rising edge or rising edge/high-level sensitive.
4 IRQPE	<b>IRQ Pin Enable</b> — This read/write control bit enables the IRQ pin function. When this bit is set the IRQ pin can be used as an interrupt request. 0 IRQ pin function is disabled. 1 IRQ pin function is enabled.
3 IRQF	<b>IRQ Flag</b> — This read-only status bit indicates when an interrupt request event has occurred. 0 No IRQ request. 1 IRQ event detected.
2 IRQACK	<b>IRQ Acknowledge</b> — This write-only bit is used to acknowledge interrupt request events (write 1 to clear IRQF). Writing 0 has no meaning or effect. Reads always return 0. If edge-and-level detection is selected (IRQMOD = 1), IRQF cannot be cleared while the IRQ pin remains at its asserted level.
1 IRQIE	<b>IRQ Interrupt Enable</b> — This read/write control bit determines whether IRQ events generate an interrupt request. 0 Interrupt request when IRQF set is disabled (use polling). 1 Interrupt requested whenever IRQF = 1.
0 IRQMOD	<b>IRQ Detection Mode</b> — This read/write control bit selects either edge-only detection or edge-and-level detection. The IRQEDG control bit determines the polarity of edges and levels that are detected as interrupt request events. See <a href="#">Section 5.5.2.2, “Edge and Level Sensitivity,”</a> for more details. 0 IRQ event on falling edges or rising edges only. 1 IRQ event on falling edges and low levels or on rising edges and high levels.

## 5.7.2 System Reset Status Register (SRS)

This high page register includes read-only status flags to indicate the source of the most recent reset. When a debug host forces reset by writing 1 to BDFR in the SBDFR register, none of the status bits in SRS will be set. Writing any value to this register address causes a COP reset when the COP is enabled except the values 0x55 and 0xAA. Writing a 0x55-0xAA sequence to this address clears the COP watchdog timer without affecting the contents of this register. The reset state of these bits depends on what caused the MCU to reset.

	7	6	5	4	3	2	1	0
R	POR	PIN	COP	ILOP	ILAD	0	LVD	0
W	Writing 0x55, 0xAA to SRS address clears COP watchdog timer.							
POR:	1	0	0	0	0	0	1	0
LVR:	u <sup>1</sup>	0	0	0	0	0	1	0
Any other reset:	0	Note <sup>2</sup>	Note <sup>2</sup>	Note <sup>2</sup>	Note <sup>2</sup>	0	0	0

<sup>1</sup> u = unaffected

<sup>2</sup> Any of these reset sources that are active at the time of reset entry will cause the corresponding bit(s) to be set; bits corresponding to sources that are not active at the time of reset entry will be cleared.

**Figure 5-3. System Reset Status (SRS)**

**Table 5-4. SRS Register Field Descriptions**

Field	Description
7 POR	<b>Power-On Reset</b> — Reset was caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR. 1 POR caused reset.
6 PIN	<b>External Reset Pin</b> — Reset was caused by an active-low level on the external reset pin. 0 Reset not caused by external reset pin. 1 Reset came from external reset pin.
5 COP	<b>Computer Operating Properly (COP) Watchdog</b> — Reset was caused by the COP watchdog timer timing out. This reset source can be blocked by COPE = 0. 0 Reset not caused by COP timeout. 1 Reset caused by COP timeout.
4 ILOP	<b>Illegal Opcode</b> — Reset was caused by an attempt to execute an unimplemented or illegal opcode. The STOP instruction is considered illegal if stop is disabled by STOPE = 0 in the SOPT register. The BGND instruction is considered illegal if active background mode is disabled by ENBDM = 0 in the BDCSC register. 0 Reset not caused by an illegal opcode. 1 Reset caused by an illegal opcode.

### 6.6.1.4 Port A Slew Rate Enable Register (PTASE)

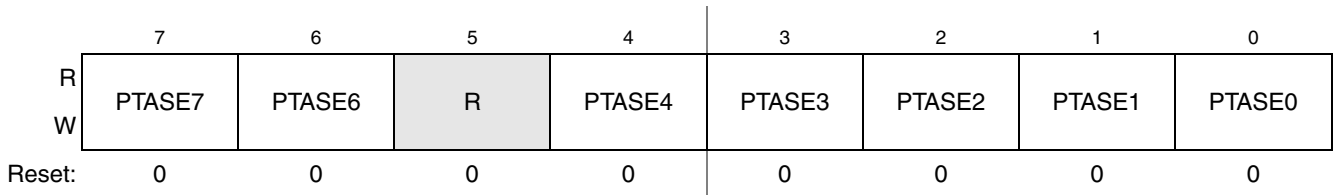


Figure 6-6. Slew Rate Enable for Port A Register (PTASE)

Table 6-5. PTASE Register Field Descriptions

Field	Description
7:6,4:0 PTASE [7:6, 4:0]	<b>Output Slew Rate Enable for Port A Bits</b> — Each of these control bits determines if the 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. 0 Output slew rate control disabled for port A bit n. 1 Output slew rate control enabled for port A bit n.
5 Reserved	<b>Reserved Bits</b> — These bits are unused on this MCU, writes have no affect and could read as 1s or 0s.

### 6.6.1.5 Port A Drive Strength Selection Register (PTADS)

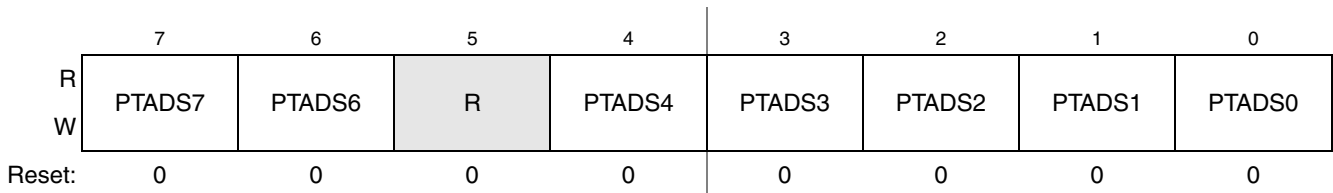


Figure 6-7. Drive Strength Selection for Port A Register (PTADS)

Table 6-6. PTADS Register Field Descriptions

Field	Description
7:6, 4:0 PTADS [7:6, 4:0]	<b>Output Drive Strength Selection for Port A Bits</b> — Each of these control bits selects between low and high output drive for the associated PTA pin. For port A pins that are configured as inputs, these bits have no effect. 0 Low output drive strength selected for port A bit n. 1 High output drive strength selected for port A bit n.
5 Reserved	<b>Reserved Bits</b> — These bits are unused on this MCU, writes have no affect and could read as 1s or 0s.



## 7.2 Programmer's Model and CPU Registers

Figure 7-1 shows the five CPU registers. CPU registers are not part of the memory map.

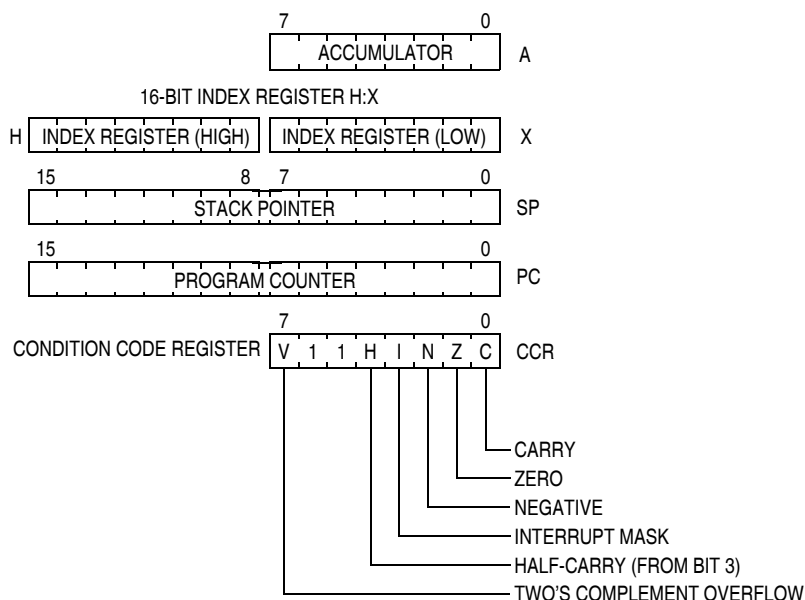


Figure 7-1. CPU Registers

### 7.2.1 Accumulator (A)

The A accumulator is a general-purpose 8-bit register. One operand input to the arithmetic logic unit (ALU) is connected to the accumulator and the ALU results are often stored into the A accumulator after arithmetic and logical operations. The accumulator can be loaded from memory using various addressing modes to specify the address where the loaded data comes from, or the contents of A can be stored to memory using various addressing modes to specify the address where data from A will be stored.

Reset has no effect on the contents of the A accumulator.

### 7.2.2 Index Register (H:X)

This 16-bit register is actually two separate 8-bit registers (H and X), which often work together as a 16-bit address pointer where H holds the upper byte of an address and X holds the lower byte of the address. All indexed addressing mode instructions use the full 16-bit value in H:X as an index reference pointer; however, for compatibility with the earlier M68HC05 Family, some instructions operate only on the low-order 8-bit half (X).

Many instructions treat X as a second general-purpose 8-bit register that can be used to hold 8-bit data values. X can be cleared, incremented, decremented, complemented, negated, shifted, or rotated. Transfer instructions allow data to be transferred from A or transferred to A where arithmetic and logical operations can then be performed.

For compatibility with the earlier M68HC05 Family, H is forced to 0x00 during reset. Reset has no effect on the contents of X.

# Chapter 9

## Analog-to-Digital Converter (S08ADCV1)

### 9.1 Introduction

The 10-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

#### NOTE

- MC9S08SH32 Series devices operate at a higher voltage range (2.7 V to 5.5 V) and do not include stop1 mode. Please ignore references to stop1.
- MC9S08SH32 Series devices have up to 16 analog inputs. Consequently, the APCTL3 register is not available on these devices.

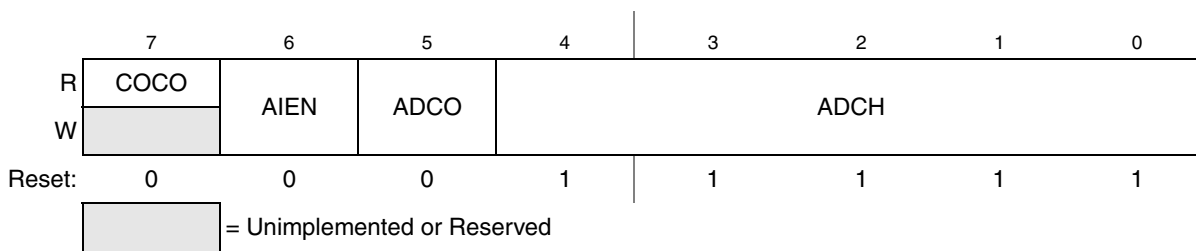
The ADC channel assignments, alternate clock function, and hardware trigger function are configured as described below for the MC9S08SH32 Series family of devices.

#### 9.1.1 Channel Assignments

The ADC channel assignments for the MC9S08SH32 Series devices are shown in [Table 9-1](#). Reserved channels convert to an unknown value. This chapter shows bits for all S08ADCV1 channels. MC9S08SH32 Series MCUs do not use all of these channels. All bits corresponding to channels that are not available on a device are reserved.

**Table 9-1. ADC Channel Assignment**

ADCH	Channel	Input	ADCH	Channel	Input
00000	AD0	PTA0/AD0	10000	AD16	V <sub>SS</sub>
00001	AD1	PTA1/ADP1	10001	AD17	V <sub>SS</sub>
00010	AD2	PTA2/ADP2	10010	AD18	V <sub>SS</sub>
00011	AD3	PTA3/ADP3	10011	AD19	V <sub>SS</sub>
00100	AD4	PTB0/ADP4	10100	AD20	V <sub>SS</sub>
00101	AD5	PTB1/ADP5	10101	AD21	V <sub>SS</sub>
00110	AD6	PTB2/ADP6	10110	AD22	Reserved
00111	AD7	PTB3/ADP7	10111	AD23	Reserved
01000	AD8	PTC0/ADP8	11000	AD24	Reserved
01001	AD9	PTC1/ADP9	11001	AD25	Reserved
01010	AD10	PTC2/ADP10	11010	AD26	Temperature Sensor <sup>1</sup>
01011	AD11	PTC3/ADP11	11011	AD27	Internal Bandgap <sup>2</sup>
01100	AD12	PTC4/ADP12	11100	-	Reserved
01101	AD13	PTC5/ADP13	11101	V <sub>REFH</sub>	V <sub>DD</sub>
01110	AD14	PTC6/ADP14	11110	V <sub>REFL</sub>	V <sub>SS</sub>



**Figure 9-3. Status and Control Register (ADCSC1)**

**Table 9-3. ADCSC1 Register Field Descriptions**

Field	Description
7 COCO	<p><b>Conversion Complete Flag</b> — The COCO flag is a read-only bit which is set each time a conversion is completed when the compare function is disabled (ACFE = 0). When the compare function is enabled (ACFE = 1) the COCO flag is set upon completion of a conversion only if the compare result is true. This bit is cleared whenever ADCSC1 is written or whenever ADCRL is read.</p> <p>0 Conversion not completed 1 Conversion completed</p>
6 AIEN	<p><b>Interrupt Enable</b> — AIEN is used to enable conversion complete interrupts. When COCO becomes set while AIEN is high, an interrupt is asserted.</p> <p>0 Conversion complete interrupt disabled 1 Conversion complete interrupt enabled</p>
5 ADCO	<p><b>Continuous Conversion Enable</b> — ADCO is used to enable continuous conversions.</p> <p>0 One conversion following a write to the ADCSC1 when software triggered operation is selected, or one conversion following assertion of ADHWT when hardware triggered operation is selected. 1 Continuous conversions initiated following a write to ADCSC1 when software triggered operation is selected. Continuous conversions are initiated by an ADHWT event when hardware triggered operation is selected.</p>
4:0 ADCH	<p><b>Input Channel Select</b> — The ADCH bits form a 5-bit field which is used to select one of the input channels. The input channels are detailed in <a href="#">Figure 9-4</a>.</p> <p>The successive approximation converter subsystem is turned off when the channel select bits are all set to 1. This feature allows for explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way will prevent an additional, single conversion from being performed. It is not necessary to set the channel select bits to all 1s to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes.</p>

**Figure 9-4. Input Channel Select**

ADCH	Input Select	ADCH	Input Select
00000	AD0	10000	AD16
00001	AD1	10001	AD17
00010	AD2	10010	AD18
00011	AD3	10011	AD19
00100	AD4	10100	AD20
00101	AD5	10101	AD21
00110	AD6	10110	AD22
00111	AD7	10111	AD23

### 9.4.4.2 Completing Conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, ADCRH and ADCRL. This is indicated by the setting of COCO. An interrupt is generated if AIEN is high at the time that COCO is set.

A blocking mechanism prevents a new result from overwriting previous data in ADCRH and ADCRL if the previous data is in the process of being read while in 10-bit MODE (the ADCRH register has been read but the ADCRL register has not). When blocking is active, the data transfer is blocked, COCO is not set, and the new result is lost. In the case of single conversions with the compare function enabled and the compare condition false, blocking has no effect and ADC operation is terminated. In all other cases of operation, when a data transfer is blocked, another conversion is initiated regardless of the state of ADCO (single or continuous conversions enabled).

If single conversions are enabled, the blocking mechanism could result in several discarded conversions and excess power consumption. To avoid this issue, the data registers must not be read after initiating a single conversion until the conversion completes.

### 9.4.4.3 Aborting Conversions

Any conversion in progress will be aborted when:

- A write to ADCSC1 occurs (the current conversion will be aborted and a new conversion will be initiated, if ADCH are not all 1s).
- A write to ADCSC2, ADCCFG, ADCCVH, or ADCCVL occurs. This indicates a mode of operation change has occurred and the current conversion is therefore invalid.
- The MCU is reset.
- The MCU enters stop mode with ADACK not enabled.

When a conversion is aborted, the contents of the data registers, ADCRH and ADCRL, are not altered but continue to be the values transferred after the completion of the last successful conversion. In the case that the conversion was aborted by a reset, ADCRH and ADCRL return to their reset states.

### 9.4.4.4 Power Control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source, the ADACK clock generator is also enabled.

Power consumption when active can be reduced by setting ADLPC. This results in a lower maximum value for  $f_{ADCK}$  (see the electrical specifications).

### 9.4.4.5 Total Conversion Time

The total conversion time depends on the sample time (as determined by ADLSMP), the MCU bus frequency, the conversion mode (8-bit or 10-bit), and the frequency of the conversion clock ( $f_{ADCK}$ ). After the module becomes active, sampling of the input begins. ADLSMP is used to select between short and long sample times. When sampling is complete, the converter is isolated from the input channel and a successive approximation algorithm is performed to determine the digital value of the analog signal. The

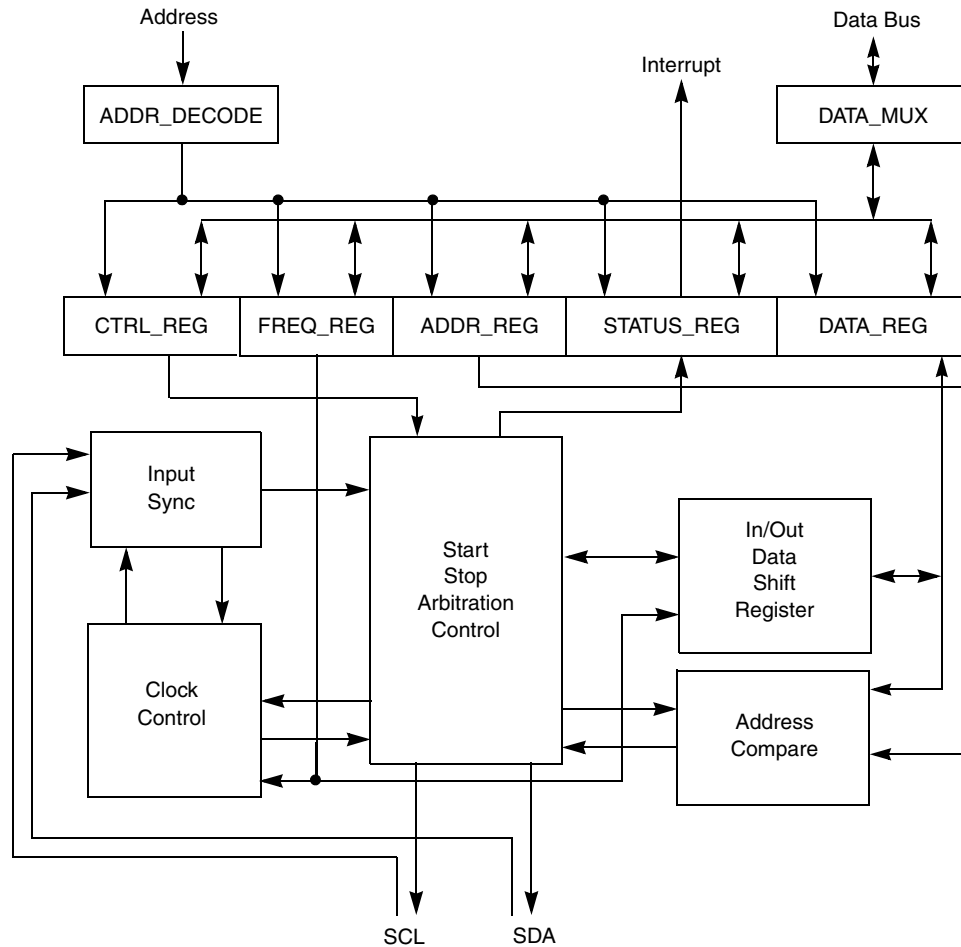


Figure 10-2. IIC Functional Block Diagram

## 10.2 External Signal Description

This section describes each user-accessible pin signal.

### 10.2.1 SCL — Serial Clock Line

The bidirectional SCL is the serial clock line of the IIC system.

### 10.2.2 SDA — Serial Data Line

The bidirectional SDA is the serial data line of the IIC system.

## 10.3 Register Definition

This section consists of the IIC register descriptions in address order.

Refer to the direct-page register summary in the [memory](#) chapter of this document for the absolute address assignments for all IIC registers. This section refers to registers and control bits only by their names. A

### 10.4.1.2 Slave Address Transmission

The first byte of data transferred immediately after the start signal is the slave address transmitted by the master. This is a seven-bit calling address followed by a  $R/\overline{W}$  bit. The  $R/\overline{W}$  bit tells the slave the desired direction of data transfer.

1 = Read transfer, the slave transmits data to the master.

0 = Write transfer, the master transmits data to the slave.

Only the slave with a calling address that matches the one transmitted by the master responds by sending back an acknowledge bit. This is done by pulling the SDA low at the ninth clock (see [Figure 10-9](#)).

No two slaves in the system may have the same address. If the IIC module is the master, it must not transmit an address equal to its own slave address. The IIC cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the IIC reverts to slave mode and operates correctly even if it is being addressed by another master.

### 10.4.1.3 Data Transfer

Before successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the  $R/\overline{W}$  bit sent by the calling master.

All transfers that come after an address cycle are referred to as data transfers, even if they carry sub-address information for the slave device

Each data byte is 8 bits long. Data may be changed only while SCL is low and must be held stable while SCL is high as shown in [Figure 10-9](#). There is one clock pulse on SCL for each data bit, the msb being transferred first. Each data byte is followed by a 9th (acknowledge) bit, which is signalled from the receiving device. An acknowledge is signalled by pulling the SDA low at the ninth clock. In summary, one complete data transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master in the ninth bit time, the SDA line must be left high by the slave. The master interprets the failed acknowledge as an unsuccessful data transfer.

If the master receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets this as an end of data transfer and releases the SDA line.

In either case, the data transfer is aborted and the master does one of two things:

- Relinquishes the bus by generating a stop signal.
- Commences a new calling by generating a repeated start signal.

### 10.4.1.4 Stop Signal

The master can terminate the communication by generating a stop signal to free the bus. However, the master may generate a start signal followed by a calling command without generating a stop signal first. This is called repeated start. A stop signal is defined as a low-to-high transition of SDA while SCL at logical 1 (see [Figure 10-9](#)).

The master can generate a stop even if the slave has generated an acknowledge at which point the slave must release the bus.

### 10.4.1.5 Repeated Start Signal

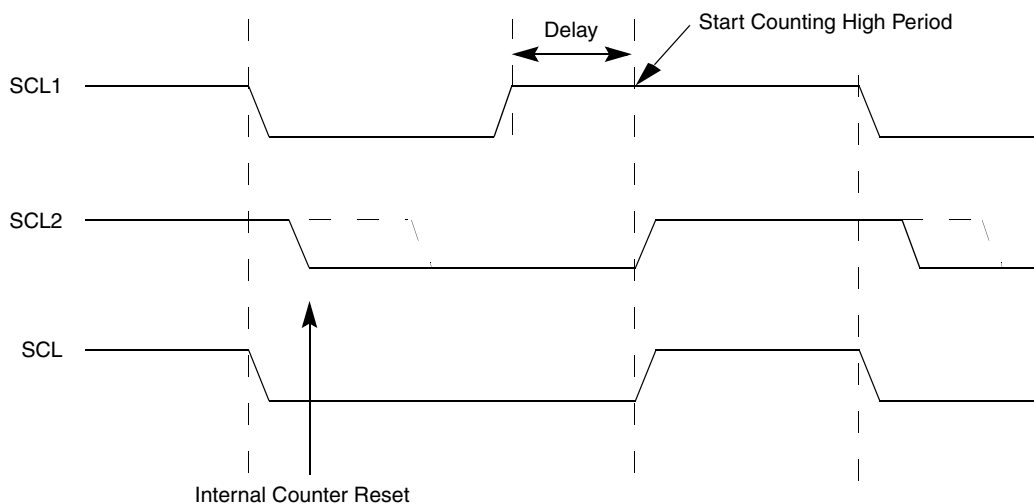
As shown in [Figure 10-9](#), a repeated start signal is a start signal generated without first generating a stop signal to terminate the communication. This is used by the master to communicate with another slave or with the same slave in different mode (transmit/receive mode) without releasing the bus.

### 10.4.1.6 Arbitration Procedure

The IIC bus is a true multi-master bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock, for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. The relative priority of the contending masters is determined by a data arbitration procedure, a bus master loses arbitration if it transmits logic 1 while another master transmits logic 0. The losing masters immediately switch over to slave receive mode and stop driving SDA output. In this case, the transition from master to slave mode does not generate a stop condition. Meanwhile, a status bit is set by hardware to indicate loss of arbitration.

### 10.4.1.7 Clock Synchronization

Because wire-AND logic is performed on the SCL line, a high-to-low transition on the SCL line affects all the devices connected on the bus. The devices start counting their low period and after a device's clock has gone low, it holds the SCL line low until the clock high state is reached. However, the change of low to high in this device clock may not change the state of the SCL line if another device clock is still within its low period. Therefore, synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see [Figure 10-10](#)). When all devices concerned have counted off their low period, the synchronized clock SCL line is released and pulled high. There is then no difference between the device clocks and the state of the SCL line and all the devices start counting their high periods. The first device to complete its high period pulls the SCL line low again.



**Figure 10-10. IIC Clock Synchronization**

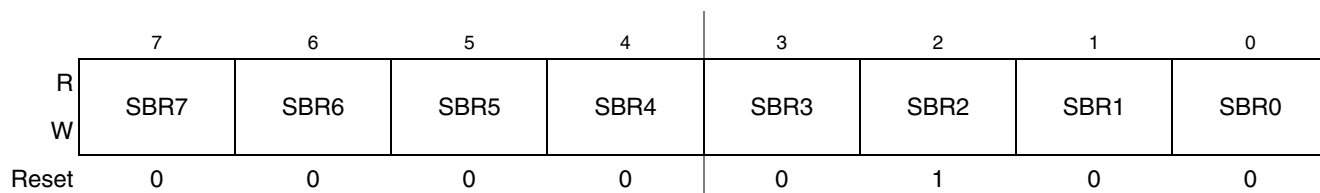
## Chapter 13

# Real-Time Counter (S08RTCV1)

### 13.1 Introduction

The RTC module consists of one 8-bit counter, one 8-bit comparator, several binary-based and decimal-based prescaler dividers, two clock sources, and one programmable periodic interrupt. This module can be used for time-of-day, calendar or any task scheduling functions. It can also serve as a cyclic wake up from low power modes without the need of external components.

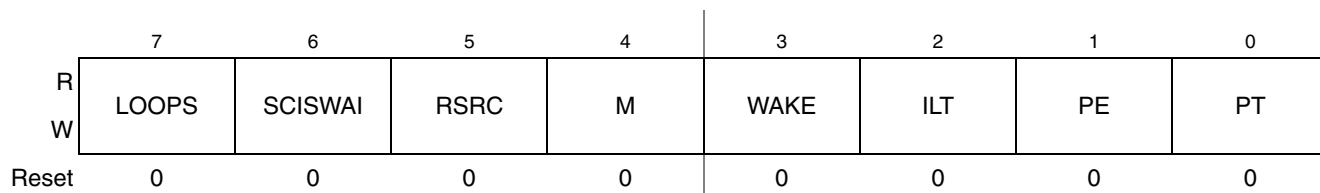



**Figure 14-5. SCI Baud Rate Register (SCIxBDL)**
**Table 14-2. SCIxBDL Field Descriptions**

Field	Description
7:0 SBR[7:0]	<b>Baud Rate Modulo Divisor</b> — These 13 bits in SBR[12:0] 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×BR). See also BR bits in <a href="#">Table 14-1</a> .

## 14.2.2 SCI Control Register 1 (SCIxC1)

This read/write register is used to control various optional features of the SCI system.


**Figure 14-6. SCI Control Register 1 (SCIxC1)**
**Table 14-3. SCIxC1 Field Descriptions**

Field	Description
7 LOOPS	<b>Loop Mode Select</b> — Selects between loop back modes and normal 2-pin full-duplex modes. When LOOPS = 1, the transmitter output is internally connected to the receiver input. 0 Normal operation — RxD and TxD use separate pins. 1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input. (See <a href="#">RSRC</a> bit.) RxD pin is not used by SCI.
6 SCISWAI	<b>SCI Stops in Wait Mode</b> 0 SCI clocks continue to run in wait mode so the SCI can be the source of an interrupt that wakes up the CPU. 1 SCI clocks freeze while CPU is in wait mode.
5 RSRC	<b>Receiver Source Select</b> — This bit has no meaning or effect unless the LOOPS bit is set to 1. When LOOPS = 1, the receiver input is internally connected to the TxD pin and RSRC determines whether this connection is also connected to the transmitter output. 0 Provided LOOPS = 1, RSRC = 0 selects internal loop back mode and the SCI does not use the RxD pins. 1 Single-wire SCI mode where the TxD pin is connected to the transmitter output and receiver input.
4 M	<b>9-Bit or 8-Bit Mode Select</b> 0 Normal — start + 8 data bits (LSB first) + stop. 1 Receiver and transmitter use 9-bit data characters start + 8 data bits (LSB first) + 9th data bit + stop.

**Table 14-7. SCiXC3 Field Descriptions (continued)**

Field	Description
4 TXINV <sup>1</sup>	<b>Transmit Data Inversion</b> — Setting this bit reverses the polarity of the transmitted data output. 0 Transmit data not inverted 1 Transmit data inverted
3 ORIE	<b>Overrun Interrupt Enable</b> — This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0 OR interrupts disabled (use polling). 1 Hardware interrupt requested when OR = 1.
2 NEIE	<b>Noise Error Interrupt Enable</b> — This bit enables the noise flag (NF) to generate hardware interrupt requests. 0 NF interrupts disabled (use polling). 1 Hardware interrupt requested when NF = 1.
1 FEIE	<b>Framing Error Interrupt Enable</b> — This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0 FE interrupts disabled (use polling). 1 Hardware interrupt requested when FE = 1.
0 PEIE	<b>Parity Error Interrupt Enable</b> — This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0 PF interrupts disabled (use polling). 1 Hardware interrupt requested when PF = 1.

<sup>1</sup> Setting TXINV inverts the TxD output for all cases: data bits, start and stop bits, break, and idle.

## 14.2.7 SCI Data Register (SCiXD)

This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for the SCI status flags.

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

**Figure 14-11. SCI Data Register (SCiXD)**

## 14.3 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.

### 14.3.1 Baud Rate Generation

As shown in [Figure 14-12](#), the clock source for the SCI baud rate generator is the bus-rate clock.

## 15.5 Functional Description

An SPI transfer is initiated by checking for the SPI transmit buffer empty flag (SPTEF = 1) and then writing a byte of data to the SPI data register (SPIxD) in the master SPI device. When the SPI shift register is available, this byte of data is moved from the transmit data buffer to the shifter, SPTEF is set to indicate there is room in the buffer to queue another transmit character if desired, and the SPI serial transfer starts.

During the SPI transfer, data is sampled (read) on the MISO pin at one SPSCCK edge and shifted, changing the bit value on the MOSI pin, one-half SPSCCK cycle later. After eight SPSCCK cycles, the data that was in the shift register of the master has been shifted out the MOSI pin to the slave while eight bits of data were shifted in the MISO pin into the master's shift register. At the end of this transfer, the received data byte is moved from the shifter into the receive data buffer and SPRF is set to indicate the data can be read by reading SPIxD. If another byte of data is waiting in the transmit buffer at the end of a transfer, it is moved into the shifter, SPTEF is set, and a new transfer is started.

Normally, SPI data is transferred most significant bit (MSB) first. If the least significant bit first enable (LSBFE) bit is set, SPI data is shifted LSB first.

When the SPI is configured as a slave, its  $\overline{SS}$  pin must be driven low before a transfer starts and  $\overline{SS}$  must stay low throughout the transfer. If a clock format where CPHA = 0 is selected,  $\overline{SS}$  must be driven to a logic 1 between successive transfers. If CPHA = 1,  $\overline{SS}$  may remain low between successive transfers. See [Section 15.5.1, "SPI Clock Formats"](#) for more details.

Because the transmitter and receiver are double buffered, a second byte, in addition to the byte currently being shifted out, can be queued into the transmit data buffer, and a previously received character can be in the receive data buffer while a new character is being shifted in. The SPTEF flag indicates when the transmit buffer has room for a new character. The SPRF flag indicates when a received character is available in the receive data buffer. The received character must be read out of the receive buffer (read SPIxD) before the next transfer is finished or a receive overrun error results.

In the case of a receive overrun, the new data is lost because the receive buffer still held the previous character and was not ready to accept the new data. There is no indication for such an overrun condition so the application system designer must ensure that previous data has been read from the receive buffer before a new transfer is initiated.

### 15.5.1 SPI Clock Formats

To accommodate a wide variety of synchronous serial peripherals from different manufacturers, the SPI system has a clock polarity (CPOL) bit and a clock phase (CPHA) control bit to select one of four clock formats for data transfers. CPOL selectively inserts an inverter in series with the clock. CPHA chooses between two different clock phase relationships between the clock and data.

[Figure 15-10](#) shows the clock formats when CPHA = 1. At the top of the figure, the eight bit times are shown for reference with bit 1 starting at the first SPSCCK edge and bit 8 ending one-half SPSCCK cycle after the sixteenth SPSCCK edge. The MSB first and LSB first lines show the order of SPI data bits depending on the setting in LSBFE. Both variations of SPSCCK polarity are shown, but only one of these waveforms applies for a specific transfer, depending on the value in CPOL. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the

When no debugger pod is connected to the 6-pin BDM interface connector, the internal pullup on BKGD chooses normal operating mode. When a debug pod is connected to BKGD it is possible to force the MCU into active background mode after reset. The specific conditions for forcing active background depend upon the HCS08 derivative (refer to the introduction to this Development Support section). It is not necessary to reset the target MCU to communicate with it through the background debug interface.

## 17.2.2 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.

## 17.3.6 Hardware Breakpoints

The BRKEN control bit in the DBGCR register may be set to 1 to allow any of the trigger conditions described in [Section 17.3.5, “Trigger Modes,”](#) to be used to generate a hardware breakpoint request to the CPU. TAG in DBGCR controls whether the breakpoint request will be treated as a tag-type breakpoint or a force-type breakpoint. A tag breakpoint causes the current opcode to be marked as it enters the instruction queue. If a tagged opcode reaches the end of the pipe, the CPU executes a BGND instruction to go to active background mode rather than executing the tagged opcode. A force-type breakpoint causes the CPU to finish the current instruction and then go to active background mode.

If the background mode has not been enabled (ENBDM = 1) by a serial WRITE\_CONTROL command through the BKGD pin, the CPU will execute an SWI instruction instead of going to active background mode.

## 17.4 Register Definition

This section contains the descriptions of the BDC and DBG registers and control bits.

Refer to the high-page register summary in the device overview chapter of this data sheet for the absolute address assignments for all DBG 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.

### 17.4.1 BDC Registers and Control Bits

The BDC has two registers:

- The BDC status and control register (BDCSCR) is an 8-bit register containing control and status bits for the background debug controller.
- The BDC breakpoint match register (BDCBKPT) holds a 16-bit breakpoint match address.

These registers are accessed with dedicated serial BDC commands and are not located in the memory space of the target MCU (so they do not have addresses and cannot be accessed by user programs).

Some of the bits in the BDCSCR have write limitations; otherwise, these registers may be read or written at any time. For example, the ENBDM control bit may not be written while the MCU is in active background mode. (This prevents the ambiguous condition of the control bit forbidding active background mode while the MCU is already in active background mode.) Also, the four status bits (BDMACT, WS, WSF, and DVF) are read-only status indicators and can never be written by the WRITE\_CONTROL serial BDC command. The clock switch (CLKSW) control bit may be read or written at any time.