

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	12
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	16-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	16-TSSOP
Purchase URL	<a href="https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=s9s08sg4e2mtg">https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=s9s08sg4e2mtg</a>

## Revision History

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

<http://freescale.com/>

The following revision history table summarizes changes contained in this document.

Revision Number	Revision Date	Description of Changes
0	15 Dec 2006	Initial alpha customer release version; Preliminary
1	June 2007	Samples Draft. Updated book with the latest TPM v3 module. Includes some minor edits to the IIC module to update the Module Quick Start. Fixed the SOPT1 bits 1 and 0 to be RESERVED for both READ and WRITE. Changed all the Reset states of the Slew Rate Enable Registers (PTASE, PTBSE, and PTCSE) bits from 1 to 0 due to silicon functional change.
2	11/2007	Market Launch. Updated the Electricals and Device Numbering scheme information.
3	12/2007	<ul style="list-style-type: none"> <li>Fixed typos: Chapter 7 heading corrected version to v2, and Figure 16-1. title corrected to read ..."TPM Modules Highlighted."</li> <li>Table A-3. Thermal Characteristics row 1, V and M entries were transposed. V now refers to value -40 to 105°C and M now refers to value -40 to 125°C. Added row 2, parameter classification of "D" and row 4 symbol of "<math>\theta_{JA}</math>."</li> <li>Table A-6. DC Characteristics, row 8 Input Hysteresis, corrected units from mV to V.</li> </ul>
4	3/2008	<ul style="list-style-type: none"> <li>SPI block corrected to be version 3 of the module.</li> <li>Temperature Sensor values corrected to reflect the ADC 5V in Section 9.1.4 Temperature Sensor and Table A-12. ADC Characteristics.</li> <li>Provided Maximum juncture temperature for C, V, and M Temperature ranges.</li> <li>Corrected Table A-6, row 10 separated to two pins: PTB6/SDA/XTAL, RESET.</li> <li>Corrected block diagrams User Flash and User RAM listing typos to be SG8 and SG4 instead of SH8 and SH4.</li> <li>Updated the Revision History for Revision Number 1 to include the information on the Slew Rate Enable Register changes that occurred for that revision.</li> </ul>
5	6/2008	<ul style="list-style-type: none"> <li>Added ICS over Temperature graph to Electricals appendix.</li> </ul>

© Freescale Semiconductor, Inc., 2006-2014. All rights reserved.

This product incorporates SuperFlash<sup>®</sup> Technology licensed from SST.

# Contents

Section Number	Title	Page
<b>Chapter 1</b>		
<b>Device Overview</b>		
1.1	Devices in the MC9S08SG8 Series .....	21
1.2	MCU Block Diagram .....	22
1.3	System Clock Distribution .....	24
<b>Chapter 2</b>		
<b>Pins and Connections</b>		
2.1	Device Pin Assignment .....	25
2.2	Recommended System Connections .....	27
2.2.1	Power .....	27
2.2.2	Oscillator (XOSC) .....	28
2.2.3	$\overline{\text{RESET}}$ Pin .....	28
2.2.4	Background / Mode Select (BKGD/MS) .....	29
2.2.5	General-Purpose I/O and Peripheral Ports .....	29
<b>Chapter 3</b>		
<b>Modes of Operation</b>		
3.1	Introduction .....	33
3.2	Features .....	33
3.3	Run Mode .....	33
3.4	Active Background Mode .....	33
3.5	Wait Mode .....	34
3.6	Stop Modes .....	34
3.6.1	Stop3 Mode .....	35
3.6.2	Stop2 Mode .....	36
3.6.3	On-Chip Peripheral Modules in Stop Modes .....	36
<b>Chapter 4</b>		
<b>Memory</b>		
4.1	MC9S08SG8 Memory Map .....	39
4.2	Reset and Interrupt Vector Assignments .....	40
4.3	Register Addresses and Bit Assignments .....	41
4.4	RAM .....	48
4.5	FLASH .....	48
4.5.1	Features .....	49
4.5.2	Program and Erase Times .....	49

Section Number	Title	Page
9.5	Initialization Information .....	139
9.5.1	ADC Module Initialization Example .....	139
9.6	Application Information .....	141
9.6.1	External Pins and Routing .....	141
9.6.2	Sources of Error .....	143

## Chapter 10 Internal Clock Source (S08ICSV2)

10.1	Introduction .....	147
10.1.1	Module Configuration .....	147
10.1.2	Features .....	149
10.1.3	Block Diagram .....	149
10.1.4	Modes of Operation .....	150
10.2	External Signal Description .....	151
10.3	Register Definition .....	151
10.3.1	ICS Control Register 1 (ICSC1) .....	152
10.3.2	ICS Control Register 2 (ICSC2) .....	153
10.3.3	ICS Trim Register (ICSTRM) .....	154
10.3.4	ICS Status and Control (ICSSC) .....	154
10.4	Functional Description .....	155
10.4.1	Operational Modes .....	155
10.4.2	Mode Switching .....	157
10.4.3	Bus Frequency Divider .....	158
10.4.4	Low Power Bit Usage .....	158
10.4.5	Internal Reference Clock .....	158
10.4.6	Optional External Reference Clock .....	158
10.4.7	Fixed Frequency Clock .....	159

## Chapter 11 Inter-Integrated Circuit (S08IICV2)

11.1	Introduction .....	161
11.1.1	Module Configuration .....	161
11.1.2	Features .....	163
11.1.3	Modes of Operation .....	163
11.1.4	Block Diagram .....	163
11.2	External Signal Description .....	164
11.2.1	SCL — Serial Clock Line .....	164
11.2.2	SDA — Serial Data Line .....	164
11.3	Register Definition .....	164
11.3.1	IIC Address Register (IICA) .....	165
11.3.2	IIC Frequency Divider Register (IICF) .....	165
11.3.3	IIC Control Register (IICC1) .....	168

The MC9S08SG8 devices contain a ganged output drive feature that allows a safe and reliable method of allowing pins to be tied together externally to produce a higher output current drive. See [Section 6.3](#), “Ganged Output” for more information for configuring the port pins for ganged output drive.

#### NOTE

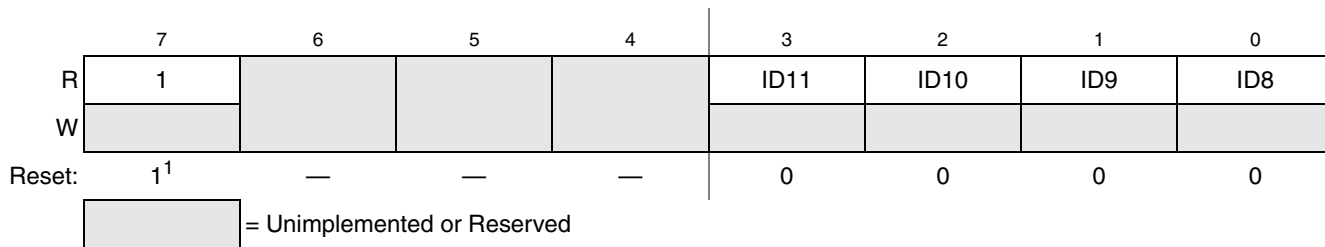
To avoid extra current drain from floating input pins, the reset initialization routine in the application program should either enable on-chip pull-up devices or change the direction of unused pins to outputs so they do not float.

When using the 8-pin devices, the user must either enable on-chip pullup devices or change the direction of non-bonded out port B and port C pins to outputs so the pins do not float.

When using the 16-pin devices, the user must either enable on-chip pullup devices or change the direction of non-bonded out port C pins to outputs so the pins do not float.

### 5.7.5 System Device Identification Register (SDIDH, SDIDL)

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

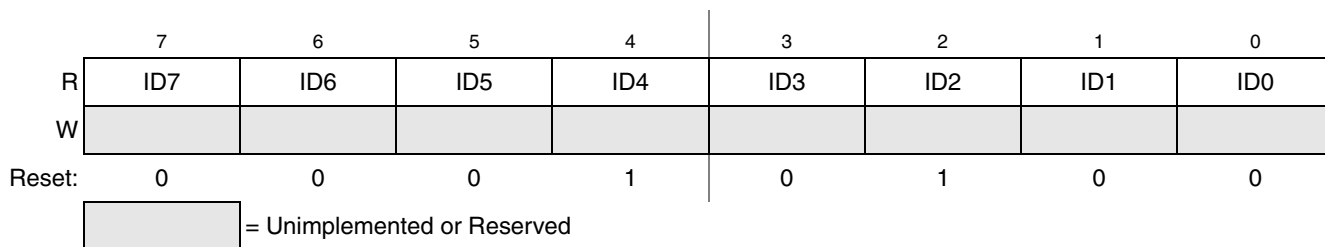


<sup>1</sup> - Bit 7 is a mask option tie off that is used internally to determine that the device is a MC9S08SG8.

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

**Table 5-7. SDIDH Register Field Descriptions**

Field	Description
7	Bit 7 will read as a 1 for the MC9S08SG8 devices; writes have no effect.
6:4 Reserved	Bits 6:4 are reserved. Reading these bits will result in an indeterminate value; writes have no effect.
3:0 ID[11:8]	<b>Part Identification Number</b> — Each derivative in the HCS08 Family has a unique identification number. The MC9S08SG8 is hard coded to the value 0x014. See also ID bits in <a href="#">Table 5-8</a> .



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

**Table 5-8. SDIDL Register Field Descriptions**

Field	Description
7:0 ID[7:0]	<b>Part Identification Number</b> — Each derivative in the HCS08 Family has a unique identification number. The MC9S08SG8 is hard coded to the value 0x014. See also ID bits in <a href="#">Table 5-7</a> .

### 7.3.6.1 Indexed, No Offset (IX)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair as the address of the operand needed to complete the instruction.

### 7.3.6.2 Indexed, No Offset with Post Increment (IX+)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair as the address of the operand needed to complete the instruction. The index register pair is then incremented ( $H:X = H:X + 0x0001$ ) after the operand has been fetched. This addressing mode is only used for MOV and CBEQ instructions.

### 7.3.6.3 Indexed, 8-Bit Offset (IX1)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction.

### 7.3.6.4 Indexed, 8-Bit Offset with Post Increment (IX1+)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction. The index register pair is then incremented ( $H:X = H:X + 0x0001$ ) after the operand has been fetched. This addressing mode is used only for the CBEQ instruction.

### 7.3.6.5 Indexed, 16-Bit Offset (IX2)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus a 16-bit offset included in the instruction as the address of the operand needed to complete the instruction.

### 7.3.6.6 SP-Relative, 8-Bit Offset (SP1)

This variation of indexed addressing uses the 16-bit value in the stack pointer (SP) plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction.

### 7.3.6.7 SP-Relative, 16-Bit Offset (SP2)

This variation of indexed addressing uses the 16-bit value in the stack pointer (SP) plus a 16-bit offset included in the instruction as the address of the operand needed to complete the instruction.

## 7.4 Special Operations

The CPU performs a few special operations that are similar to instructions but do not have opcodes like other CPU instructions. In addition, a few instructions such as STOP and WAIT directly affect other MCU circuitry. This section provides additional information about these operations.

Table 7-2. Instruction Set Summary (Sheet 4 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR				
						VH	I	N	Z	C
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 ii B1 dd C1 hh ll D1 ee ff E1 ff F1 9E D1 ee ff 9E E1 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	b	-	b	b	b
COM opr8a COMA COMX COM oprx8,X COM ,X COM oprx8,SP	Complement (One's Complement) M ← (M̄) = 0xFF – (M) A ← (Ā) = 0xFF – (A) X ← (X̄) = 0xFF – (X) M ← (M̄) = 0xFF – (M) M ← (M̄) = 0xFF – (M) M ← (M̄) = 0xFF – (M)	DIR INH INH IX1 IX SP1	33 dd 43 53 63 ff 73 9E 63 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	0	-	b	b	1
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 hh ll 65 jj kk 75 dd 9E F3 ff	6 3 5 6	prrfpp ppp rrfpp prrfpp	b	-	b	b	b
CPX #opr8i CPX opr8a CPX opr16a CPX oprx16,X CPX oprx8,X CPX ,X CPX oprx16,SP CPX oprx8,SP	Compare X (Index Register Low) with Memory X – M (CCR Updated But Operands Not Changed)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A3 ii B3 dd C3 hh ll D3 ee ff E3 ff F3 9E D3 ee ff 9E E3 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	b	-	b	b	b
DAA	Decimal Adjust Accumulator After ADD or ADC of BCD Values	INH	72	1	p	U	-	b	b	b
DBNZ opr8a,rel DBNZA rel DBNZX rel DBNZ oprx8,X,rel DBNZ ,X,rel DBNZ oprx8,SP,rel	Decrement A, X, or M and Branch if Not Zero (if (result) ≠ 0) DBNZX Affects X Not H	DIR INH INH IX1 IX SP1	3B dd rr 4B rr 5B rr 6B ff rr 7B rr 9E 6B ff rr	7 4 4 7 6 8	rfwpppp fppp fppp rfwpppp rfwppp prfwpppp	-	-	-	-	-
DEC opr8a DECA DECX DEC oprx8,X DEC ,X DEC oprx8,SP	Decrement M ← (M) – 0x01 A ← (A) – 0x01 X ← (X) – 0x01 M ← (M) – 0x01 M ← (M) – 0x01 M ← (M) – 0x01	DIR INH INH IX1 IX SP1	3A dd 4A 5A 6A ff 7A 9E 6A ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	b	-	b	b	-
DIV	Divide A ← (H:A) ÷ (X); H ← Remainder	INH	52	6	fffffp	-	-	-	b	b
EOR #opr8i EOR opr8a EOR opr16a EOR oprx16,X EOR oprx8,X EOR ,X EOR oprx16,SP EOR oprx8,SP	Exclusive OR Memory with Accumulator A ← (A ⊕ M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A8 ii B8 dd C8 hh ll D8 ee ff E8 ff F8 9E D8 ee ff 9E E8 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0	-	b	b	-



## 8.1.4 Features

The ACMP has the following features:

- Full rail to rail supply operation.
- Selectable interrupt on rising edge, falling edge, or either rising or falling edges of comparator output.
- Option to compare to fixed internal bandgap reference voltage.
- Option to allow comparator output to be visible on a pin, ACMPO.
- Can operate in stop3 mode

## 8.1.5 Modes of Operation

This section defines the ACMP operation in wait, stop and background debug modes.

### 8.1.5.1 ACMP in Wait Mode

The ACMP continues to run in wait mode if enabled before executing the WAIT instruction. Therefore, the ACMP can be used to bring the MCU out of wait mode if the ACMP interrupt, ACIE is enabled. For lowest possible current consumption, the ACMP should be disabled by software if not required as an interrupt source during wait mode.

### 8.1.5.2 ACMP in Stop Modes

#### 8.1.5.2.1 Stop3 Mode Operation

The ACMP continues to operate in Stop3 mode if enabled and compare operation remains active. If ACOPE is enabled, comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. The MCU is brought out of stop when a compare event occurs and ACIE is enabled; ACF flag sets accordingly.

If stop is exited with a reset, the ACMP will be put into its reset state.

#### 8.1.5.2.2 Stop2 and Stop1 Mode Operation

During either Stop2 and Stop1 mode, the ACMP module will be fully powered down. Upon wake-up from Stop2 or Stop1 mode, the ACMP module will be in the reset state.

### 8.1.5.3 ACMP in Active Background Mode

When the microcontroller is in active background mode, the ACMP will continue to operate normally.

## 8.1.6 Block Diagram

The block diagram for the Analog Comparator module is shown [Figure 8-2](#).

result of the conversion is transferred to ADCRH and ADCRL upon completion of the conversion algorithm.

If the bus frequency is less than the  $f_{ADCK}$  frequency, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled (ADLSMP=0). If the bus frequency is less than 1/11th of the  $f_{ADCK}$  frequency, precise sample time for continuous conversions cannot be guaranteed when long sample is enabled (ADLSMP=1).

The maximum total conversion time for different conditions is summarized in [Table 9-12](#).

**Table 9-12. Total Conversion Time vs. Control Conditions**

Conversion Type	ADICLK	ADLSMP	Max Total Conversion Time
Single or first continuous 8-bit	0x, 10	0	20 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit	0x, 10	0	23 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	0x, 10	1	40 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit	0x, 10	1	43 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	11	0	5 $\mu$ s + 20 ADCK + 5 bus clock cycles
Single or first continuous 10-bit	11	0	5 $\mu$ s + 23 ADCK + 5 bus clock cycles
Single or first continuous 8-bit	11	1	5 $\mu$ s + 40 ADCK + 5 bus clock cycles
Single or first continuous 10-bit	11	1	5 $\mu$ s + 43 ADCK + 5 bus clock cycles
Subsequent continuous 8-bit; $f_{BUS} \geq f_{ADCK}$	xx	0	17 ADCK cycles
Subsequent continuous 10-bit; $f_{BUS} \geq f_{ADCK}$	xx	0	20 ADCK cycles
Subsequent continuous 8-bit; $f_{BUS} \geq f_{ADCK}/11$	xx	1	37 ADCK cycles
Subsequent continuous 10-bit; $f_{BUS} \geq f_{ADCK}/11$	xx	1	40 ADCK cycles

The maximum total conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is selectable by the ADICLK bits, and the divide ratio is specified by the ADIV bits. For example, in 10-bit mode, with the bus clock selected as the input clock source, the input clock divide-by-1 ratio selected, and a bus frequency of 8 MHz, then the conversion time for a single conversion is:

$$\text{Conversion time} = \frac{23 \text{ ADCK cyc}}{8 \text{ MHz}/1} + \frac{5 \text{ bus cyc}}{8 \text{ MHz}} = 3.5 \mu\text{s}$$

$$\text{Number of bus cycles} = 3.5 \mu\text{s} \times 8 \text{ MHz} = 28 \text{ cycles}$$

#### NOTE

The ADCK frequency must be between  $f_{ADCK}$  minimum and  $f_{ADCK}$  maximum to meet ADC specifications.

## 9.4.5 Automatic Compare Function

The compare function can be configured to check for either an upper limit or lower limit. After the input is sampled and converted, the result is added to the two's complement of the compare value (ADCCVH and ADCCVL). When comparing to an upper limit (ACFGT = 1), if the result is greater-than or equal-to the compare value, COCO is set. When comparing to a lower limit (ACFGT = 0), if the result is less than the compare value, COCO is set. The value generated by the addition of the conversion result and the two's complement of the compare value is transferred to ADCRH and ADCRL.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, COCO is not set and no data is transferred to the result registers. An ADC interrupt is generated upon the setting of COCO if the ADC interrupt is enabled (AIEN = 1).

### NOTE

The compare function can be used to monitor the voltage on a channel while the MCU is in either wait or stop3 mode. The ADC interrupt will wake the MCU when the compare condition is met.

## 9.4.6 MCU Wait Mode Operation

The WAIT instruction puts the MCU in a lower power-consumption standby mode from which recovery is very fast because the clock sources remain active. If a conversion is in progress when the MCU enters wait mode, it continues until completion. Conversions can be initiated while the MCU is in wait mode by means of the hardware trigger or if continuous conversions are enabled.

The bus clock, bus clock divided by two, and ADACK are available as conversion clock sources while in wait mode. The use of ALTCLK as the conversion clock source in wait is dependent on the definition of ALTCLK for this MCU. Consult the module introduction for information on ALTCLK specific to this MCU.

A conversion complete event sets the COCO and generates an ADC interrupt to wake the MCU from wait mode if the ADC interrupt is enabled (AIEN = 1).

## 9.4.7 MCU Stop3 Mode Operation

The STOP instruction is used to put the MCU in a low power-consumption standby mode during which most or all clock sources on the MCU are disabled.

### 9.4.7.1 Stop3 Mode With ADACK Disabled

If the asynchronous clock, ADACK, is not selected as the conversion clock, executing a STOP instruction aborts the current conversion and places the ADC in its idle state. The contents of ADCRH and ADCRL are unaffected by stop3 mode. After exiting from stop3 mode, a software or hardware trigger is required to resume conversions.

### 10.3.3 ICS Trim Register (ICSTRM)

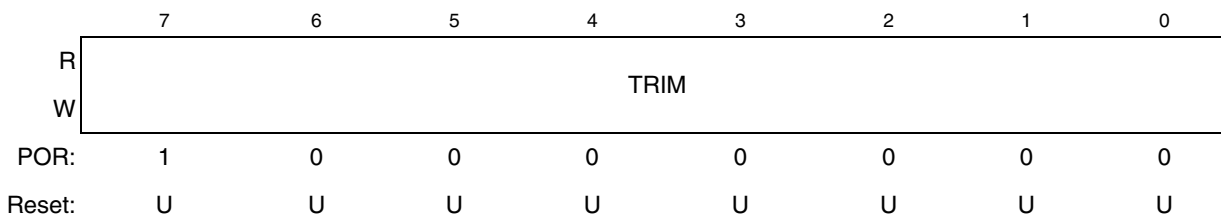


Figure 10-5. ICS Trim Register (ICSTRM)

Table 10-4. ICS Trim Register Field Descriptions

Field	Description
7:0 TRIM	<p><b>ICS Trim Setting</b> — The TRIM bits control the internal reference clock frequency by controlling the internal reference clock period. The bits' effect are binary weighted (i.e., bit 1 will adjust twice as much as bit 0). Increasing the binary value in TRIM will increase the period, and decreasing the value will decrease the period.</p> <p>An additional fine trim bit is available in ICSSC as the FTRIM bit.</p>

### 10.3.4 ICS Status and Control (ICSSC)

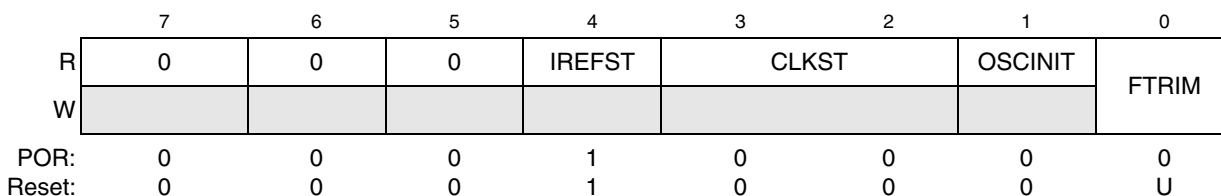


Figure 10-6. ICS Status and Control Register (ICSSC)

Table 10-5. ICS Status and Control Register Field Descriptions

Field	Description
7:5	Reserved, should be cleared.
4 IREFST	<p><b>Internal Reference Status</b> — The IREFST bit indicates the current source for the reference clock. The IREFST bit does not update immediately after a write to the IREFS bit due to internal synchronization between clock domains.</p> <p>0 Source of reference clock is external clock. 1 Source of reference clock is internal clock.</p>
3-2 CLKST	<p><b>Clock Mode Status</b> — The CLKST bits indicate the current clock mode. The CLKST bits don't update immediately after a write to the CLKST bits due to internal synchronization between clock domains.</p> <p>00 Output of FLL is selected. 01 FLL Bypassed, Internal reference clock is selected. 10 FLL Bypassed, External reference clock is selected. 11 Reserved.</p>

**Table 11-3. IICF Field Descriptions**

Field	Description
7–6 MULT	<p><b>IIC Multiplier Factor.</b> The MULT bits define the multiplier factor, mul. This factor, along with the SCL divider, generates the IIC baud rate. The multiplier factor mul as defined by the MULT bits is provided below.</p> <p>00 mul = 01 01 mul = 02 10 mul = 04 11 Reserved</p>
5–0 ICR	<p><b>IIC Clock Rate.</b> The ICR bits are used to prescale the bus clock for bit rate selection. These bits and the MULT bits determine the IIC baud rate, the SDA hold time, the SCL Start hold time, and the SCL Stop hold time. <a href="#">Table 11-5</a> provides the SCL divider and hold values for corresponding values of the ICR.</p> <p>The SCL divider multiplied by multiplier factor mul generates IIC baud rate.</p> $\text{IIC baud rate} = \frac{\text{bus speed (Hz)}}{\text{mul} \times \text{SCLdivider}} \quad \text{Eqn. 11-1}$ <p>SDA hold time is the delay from the falling edge of SCL (IIC clock) to the changing of SDA (IIC data).</p> $\text{SDA hold time} = \text{bus period (s)} \times \text{mul} \times \text{SDA hold value} \quad \text{Eqn. 11-2}$ <p>SCL start hold time is the delay from the falling edge of SDA (IIC data) while SCL is high (Start condition) to the falling edge of SCL (IIC clock).</p> $\text{SCL Start hold time} = \text{bus period (s)} \times \text{mul} \times \text{SCL Start hold value} \quad \text{Eqn. 11-3}$ <p>SCL stop hold time is the delay from the rising edge of SCL (IIC clock) to the rising edge of SDA (IIC data) while SCL is high (Stop condition).</p> $\text{SCL Stop hold time} = \text{bus period (s)} \times \text{mul} \times \text{SCL Stop hold value} \quad \text{Eqn. 11-4}$

For example, if the bus speed is 8 MHz, the table below shows the possible hold time values with different ICR and MULT selections to achieve an IIC baud rate of 100 kbps.

**Table 11-4. Hold Time Values for 8 MHz Bus Speed**

MULT	ICR	Hold Times (μs)		
		SDA	SCL Start	SCL Stop
0x2	0x00	3.500	3.000	5.500
0x1	0x07	2.500	4.000	5.250
0x1	0x0B	2.250	4.000	5.250
0x0	0x14	2.125	4.250	5.125
0x0	0x18	1.125	4.750	5.125

**Table 11-8. IICD Field Descriptions**

Field	Description
7–0 DATA	<b>Data</b> — In master transmit mode, when data is written to the IICD, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates receiving of the next byte of data.

**NOTE**

When transitioning out of master receive mode, the IIC mode should be switched before reading the IICD register to prevent an inadvertent initiation of a master receive data transfer.

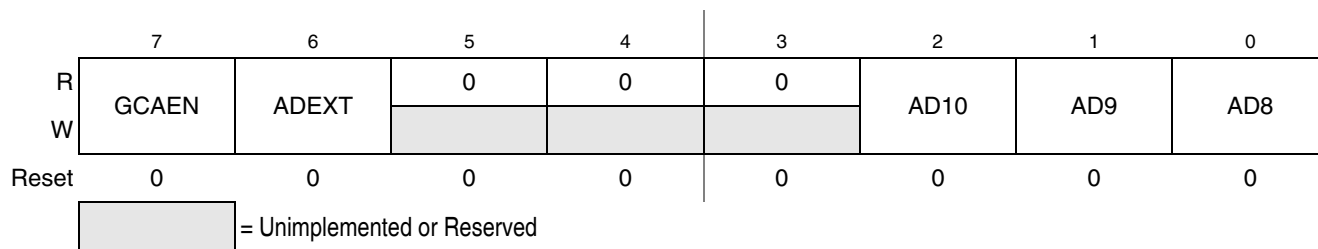
In slave mode, the same functions are available after an address match has occurred.

The TX bit in IICC must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For instance, if the IIC is configured for master transmit but a master receive is desired, reading the IICD does not initiate the receive.

Reading the IICD returns the last byte received while the IIC is configured in master receive or slave receive modes. The IICD does not reflect every byte transmitted on the IIC bus, nor can software verify that a byte has been written to the IICD correctly by reading it back.

In master transmit mode, the first byte of data written to IICD following assertion of MST is used for the address transfer and should comprise of the calling address (in bit 7 to bit 1) concatenated with the required R/W bit (in position bit 0).

**11.3.6 IIC Control Register 2 (IICC2)**



**Figure 11-8. IIC Control Register (IICC2)**

**Table 11-9. IICC2 Field Descriptions**

Field	Description
7 GCAEN	<b>General Call Address Enable.</b> The GCAEN bit enables or disables general call address. 0 General call address is disabled 1 General call address is enabled
6 ADEXT	<b>Address Extension.</b> The ADEXT bit controls the number of bits used for the slave address. 0 7-bit address scheme 1 10-bit address scheme
2–0 AD[10:8]	<b>Slave Address.</b> The AD[10:8] field contains the upper three bits of the slave address in the 10-bit address scheme. This field is only valid when the ADEXT bit is set.

### 11.4.1.2 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 11-9](#)).

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.

### 11.4.1.3 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 11-9](#). 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.

### 11.4.1.4 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 11-9](#)).

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

### 13.1.3 Block Diagram

The block diagram for the RTC module is shown in Figure 13-2.

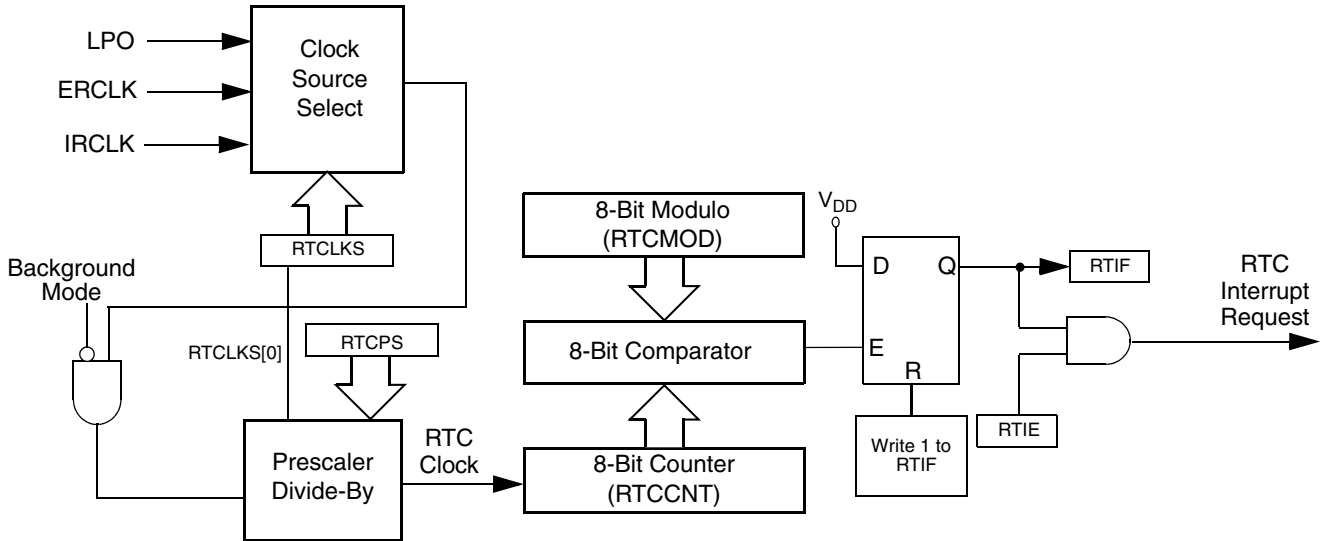


Figure 13-2. Real-Time Counter (RTC) Block Diagram

### 13.2 External Signal Description

The RTC does not include any off-chip signals.

### 13.3 Register Definition

The RTC includes a status and control register, an 8-bit counter register, and an 8-bit modulo register.

Refer to the direct-page register summary in the memory section of this document for the absolute address assignments for all RTC registers. This section refers to registers and control bits only by their names and relative address offsets.

Table 13-1 is a summary of RTC registers.

Table 13-1. RTC Register Summary

Name		7	6	5	4	3	2	1	0
RTCSC	R	RTIF	RTCLKS		RTIE	RTCPS			
	W								
RTCCNT	R	RTCCNT							
	W								
RTCMOD	R	RTCMOD							
	W								





are used for PWM & output compare operation once normal execution resumes. Writes to the channel registers while BDM is active do not interfere with partial completion of a coherency sequence. After the coherency mechanism has been fully exercised, the channel registers are updated using the buffered values written (while BDM was not active) by the user.

## 16.4 Functional Description

All TPM functions are associated with a central 16-bit counter which allows flexible selection of the clock source and prescale factor. There is also a 16-bit modulo register associated with the main counter.

The CPWMS control bit chooses between center-aligned PWM operation for all channels in the TPM (CPWMS=1) or general purpose timing functions (CPWMS=0) where each channel can independently be configured to operate in input capture, output compare, or edge-aligned PWM mode. The CPWMS control bit is located in the main TPM status and control register because it affects all channels within the TPM and influences the way the main counter operates. (In CPWM mode, the counter changes to an up/down mode rather than the up-counting mode used for general purpose timer functions.)

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

### 16.4.1 Counter

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

#### 16.4.1.1 Counter Clock Source

The 2-bit field, CLKS<sub>B</sub>:CLKS<sub>A</sub>, in the timer status and control register (TPMxSC) selects one of three possible clock sources or OFF (which effectively disables the TPM). See [Table 16-6](#). After any MCU reset, CLKS<sub>B</sub>:CLKS<sub>A</sub>=0:0 so no clock source is selected, and the TPM is in a very low power state. These control bits may be read or written at any time and disabling the timer (writing 00 to the CLKS<sub>B</sub>:CLKS<sub>A</sub> field) does not affect the values in the counter or other timer registers.

In output compare mode, values are transferred to the corresponding timer channel registers only after both 8-bit halves of a 16-bit register have been written and according to the value of CLKSB:CLKSA bits, so:

- If (CLKSB:CLKSA = 0:0), the registers are updated when the second byte is written
- If (CLKSB:CLKSA not = 0:0), the registers are updated at the next change of the TPM counter (end of the prescaler counting) after the second byte is written.

The coherency sequence can be manually reset by writing to the channel status/control register (TPMxCnSC).

An output compare event sets a flag bit (CHnF) which may optionally generate a CPU-interrupt request.

### 16.4.2.3 Edge-Aligned PWM Mode

This type of PWM output uses the normal up-counting mode of the timer counter (CPWMS=0) and can be used when other channels in the same TPM are configured for input capture or output compare functions. The period of this PWM signal is determined by the value of the modulus register (TPMxMODH:TPMxMODL) plus 1. The duty cycle is determined by the setting in the timer channel register (TPMxCnVH:TPMxCnVL). The polarity of this PWM signal is determined by the setting in the ELSnA control bit. 0% and 100% duty cycle cases are possible.

The output compare value in the TPM channel registers determines the pulse width (duty cycle) of the PWM signal (Figure 16-15). The time between the modulus overflow and the output compare is the pulse width. If ELSnA=0, the counter overflow forces the PWM signal high, and the output compare forces the PWM signal low. If ELSnA=1, the counter overflow forces the PWM signal low, and the output compare forces the PWM signal high.

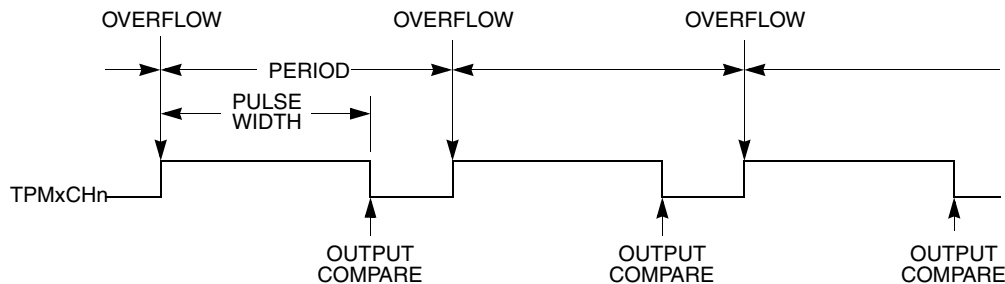


Figure 16-15. PWM Period and Pulse Width (ELSnA=0)

When the channel value register is set to 0x0000, the duty cycle is 0%. 100% duty cycle can be achieved by setting the timer-channel register (TPMxCnVH:TPMxCnVL) to a value greater than the modulus setting. This implies that the modulus setting must be less than 0xFFFF in order to get 100% duty cycle.

Because the TPM may be used in an 8-bit MCU, the settings in the timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to any of the registers TPMxCnVH and TPMxCnVL, actually write to buffer registers. In edge-aligned PWM mode, values are transferred to the corresponding timer-channel registers according to the value of CLKSB:CLKSA bits, so:

- If (CLKSB:CLKSA = 0:0), the registers are updated when the second byte is written
- If (CLKSB:CLKSA not = 0:0), the registers are updated after the both bytes were written, and the TPM counter changes from (TPMxMODH:TPMxMODL - 1) to (TPMxMODH:TPMxMODL). If

### 16.6.2.1.2 Center-Aligned PWM Case

When CPWMS=1, TOF gets set when the timer counter changes direction from up-counting to down-counting at the end of the terminal count (the value in the modulo register). In this case the TOF corresponds to the end of a PWM period.

## 16.6.2.2 Channel Event Interrupt Description

The meaning of channel interrupts depends on the channel's current mode (input-capture, output-compare, edge-aligned PWM, or center-aligned PWM).

### 16.6.2.2.1 Input Capture Events

When a channel is configured as an input capture channel, the ELSnB:ELSnA control bits select no edge (off), rising edges, falling edges or any edge as the edge which triggers an input capture event. When the selected edge is detected, the interrupt flag is set. The flag is cleared by the two-step sequence described in [Section 16.6.2, "Description of Interrupt Operation."](#)

### 16.6.2.2.2 Output Compare Events

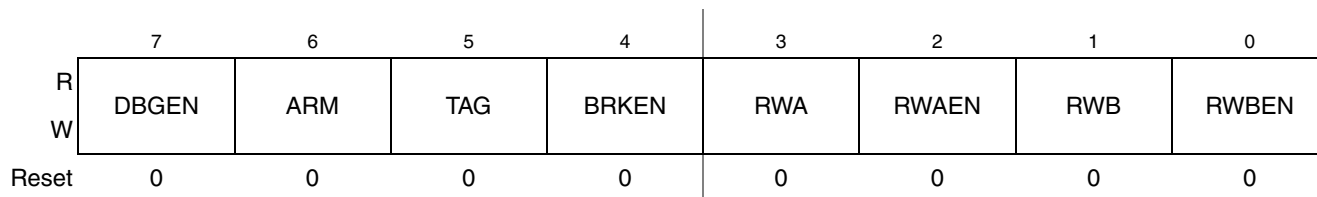
When a channel is configured as an output compare channel, the interrupt flag is set each time the main timer counter matches the 16-bit value in the channel value register. The flag is cleared by the two-step sequence described [Section 16.6.2, "Description of Interrupt Operation."](#)

### 16.6.2.2.3 PWM End-of-Duty-Cycle Events

For channels configured for PWM operation there are two possibilities. When the channel is configured for edge-aligned PWM, the channel flag gets set when the timer counter matches the channel value register which marks the end of the active duty cycle period. When the channel is configured for center-aligned PWM, the timer count matches the channel value register twice during each PWM cycle. In this CPWM case, the channel flag is set at the start and at the end of the active duty cycle period which are the times when the timer counter matches the channel value register. The flag is cleared by the two-step sequence described [Section 16.6.2, "Description of Interrupt Operation."](#)

### 17.4.3.7 Debug Control Register (DBGC)

This register can be read or written at any time.



**Figure 17-7. Debug Control Register (DBGC)**

**Table 17-4. DBGC Register Field Descriptions**

Field	Description
7 DBGEN	<b>Debug Module Enable</b> — Used to enable the debug module. DBGEN cannot be set to 1 if the MCU is secure. 0 DBG disabled 1 DBG enabled
6 ARM	<b>Arm Control</b> — Controls whether the debugger is comparing and storing information in the FIFO. A write is used to set this bit (and ARMF) and completion of a debug run automatically clears it. Any debug run can be manually stopped by writing 0 to ARM or to DBGEN. 0 Debugger not armed 1 Debugger armed
5 TAG	<b>Tag/Force Select</b> — Controls whether break requests to the CPU will be tag or force type requests. If BRKEN = 0, this bit has no meaning or effect. 0 CPU breaks requested as force type requests 1 CPU breaks requested as tag type requests
4 BRKEN	<b>Break Enable</b> — Controls whether a trigger event will generate a break request to the CPU. Trigger events can cause information to be stored in the FIFO without generating a break request to the CPU. For an end trace, CPU break requests are issued to the CPU when the comparator(s) and R/W meet the trigger requirements. For a begin trace, CPU break requests are issued when the FIFO becomes full. TRGSEL does not affect the timing of CPU break requests. 0 CPU break requests not enabled 1 Triggers cause a break request to the CPU
3 RWA	<b>R/W Comparison Value for Comparator A</b> — When RWAEN = 1, this bit determines whether a read or a write access qualifies comparator A. When RWAEN = 0, RWA and the R/W signal do not affect comparator A. 0 Comparator A can only match on a write cycle 1 Comparator A can only match on a read cycle
2 RWAEN	<b>Enable R/W for Comparator A</b> — Controls whether the level of R/W is considered for a comparator A match. 0 R/W is not used in comparison A 1 R/W is used in comparison A
1 RWB	<b>R/W Comparison Value for Comparator B</b> — When RWBEN = 1, this bit determines whether a read or a write access qualifies comparator B. When RWBEN = 0, RWB and the R/W signal do not affect comparator B. 0 Comparator B can match only on a write cycle 1 Comparator B can match only on a read cycle
0 RWBEN	<b>Enable R/W for Comparator B</b> — Controls whether the level of R/W is considered for a comparator B match. 0 R/W is not used in comparison B 1 R/W is used in comparison B