

Welcome to [E-XFL.COM](https://www.e-xfl.com)

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

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

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

Details

Product Status	Active
Core Processor	S08
Core Size	8-Bit
Speed	40MHz
Connectivity	I ² C, LINbus, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	16
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	20-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	20-TSSOP
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/s9s08sg8e2vtj

Revision Number	Revision Date	Description of Changes
6	7/2009	<ul style="list-style-type: none"> • Revised NV Register 0xFFAE address to have dashes instead of 0s. • Revised NVOPT register in Table 4-4 and Figure 4-6 so that Reserved is indicated with em dashes (—). • Changed ICS FLL deviation to 1.5% from 2%. • Table A-9, Row 1 and Table A-6 footnote 10: Removed temperature reference. • Table A-9, Row 9: Changed Column C to “D” and Max to “1.5%” • Removed section A.14.2. • Updated Mechanical drawings to point to the Freescale web. • Rebuilt book to ensure proper footers and pagination. • Revised all "Reserved" vector space memory locations in Table 4-1 to read, "Unused Vector Space (available for user program)."
7	7/2011	<ul style="list-style-type: none"> • Revised to include high-temperature (up to 150 °C) devices for 16-pin TSSOP package. • In Table 2-1, added TCLK to row 20 and Alt 3 column. • Updated “How to Reach Us” information.
8	1/2014	<p>Changes done in Chapter 3, “Modes of Operation”. Updated Table 3-1 Updated PPDC column to “x” from “0” for the specification STOPE=1, ENBDM=0, LVDE and LVDSE =both bits must be 1.</p> <p>Changes done in Appendix A, “Electrical Characteristics”</p> <ul style="list-style-type: none"> • In Table A-6. DC Characteristics for column “Characteristic” moved “Reset” from parameter “P” to “C”. • In the Table A-6 added note 11 and 12 for parameter #18. <p>Note 11: Device functionality is guaranteed between the LVD threshold VLVD0 and VDD Min. When VDD is below the minimum operating voltage (VDD Min), the analog parameters for the IO pins, ACMP and ADC, are not guaranteed to meet data sheet performance parameters.</p> <p>Note 12: In addition to LVD, it is recommended to also use the LVW feature. LVW can trigger an interrupt and be used as an indicator to warn that the VDD is dropping, so that the software can take actions accordingly before the VDD drops below VDD Min.</p> <ul style="list-style-type: none"> • Table A-13. Control Timing. For the parameter “Internal low power oscillator period” changed the “Min” value from “800” to “700”.

Chapter 1

Device Overview

The MC9S08SG8 members of the low-cost, high-performance HCS08 family of 8-bit microcontroller units (MCUs). All MCUs in the family use the enhanced HCS08 core and are available with a variety of modules, memory sizes, memory types, and package types. The high-temperature devices have been qualified to meet or exceed AEC Grade 0 requirements to allow them to operate up to 150 °C TA.

1.1 Devices in the MC9S08SG8 Series

Table 1-1 summarizes the feature set available in the MC9S08SG8 series of MCUs.

Table 1-1. MC9S08SG8 Features by MCU and Package

Feature	9S08SG8			9S08SG4		
FLASH size (bytes)	8192			4096		
RAM size (bytes)	512			256		
Pin quantity	20	16	8	20	16	8
ACMP	yes					
ADC channels	12	8	4	12	8	4
DBG	yes					
ICS	yes	yes	yes ¹	yes	yes	yes ¹
IIC	yes					
MTIM	yes					
Pin Interrupts	8	8	4	8	8	4
Pin I/O	16	12	4	16	12	4
RTC	yes					
SCI	yes	yes	no	yes	yes	no
SPI	yes	yes	no	yes	yes	no
TPM1 channels	2	2	1	2	2	1
TPM2 channels	2	2	1	2	2	1
XOSC	yes	yes	no	yes	yes	no

¹ FBE and FEE modes are not available in 8-pin packages.

Chapter 2 Pins and Connections

This section describes signals that connect to package pins. It includes pinout diagrams, recommended system connections, and detailed discussions of signals.

2.1 Device Pin Assignment

Figure 2-1 - Figure 2-3 shows the pin assignments for the MC9S08SG8 devices.

NOTE

20-pin TSSOP package and 8-pin SOIC package are not available for the AEC Grade 0 high-temperature rated devices.

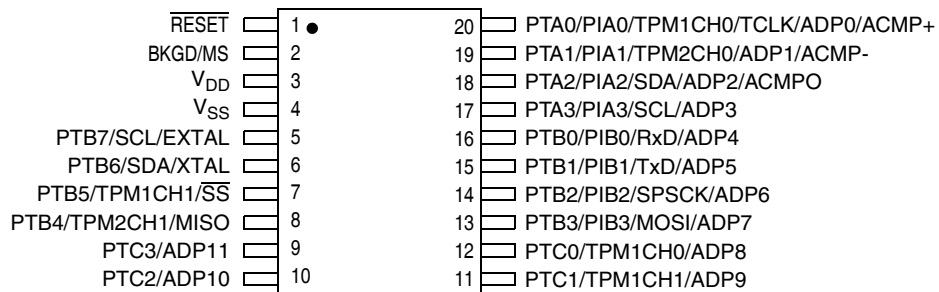


Figure 2-1. 20-Pin TSSOP

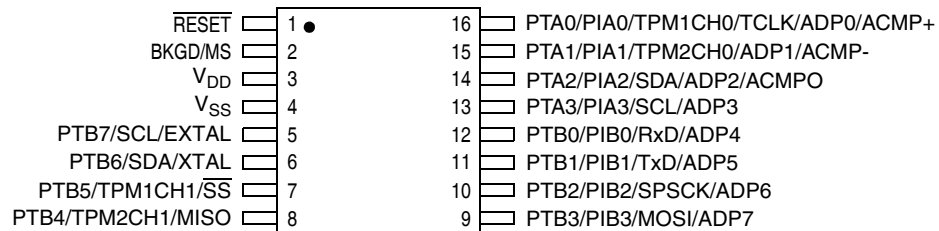


Figure 2-2. 16-Pin TSSOP

recommended for anyone other than the most experienced programmers because it can lead to subtle program errors that are difficult to debug.

The interrupt service routine ends with a return-from-interrupt (RTI) instruction which restores the CCR, A, X, and PC registers to their pre-interrupt values by reading the previously saved information from the stack.

NOTE

For compatibility with M68HC08 devices, the H register is not automatically saved and restored. It is good programming practice to push H onto the stack at the start of the interrupt service routine (ISR) and restore it immediately before the RTI that is used to return from the ISR.

If more than one interrupt is pending when the I bit is cleared, the highest priority source is serviced first (see Table 5-2).

5.5.1 Interrupt Stack Frame

Figure 5-1 shows the contents and organization of a stack frame. Before the interrupt, the stack pointer (SP) points at the next available byte location on the stack. The current values of CPU registers are stored on the stack starting with the low-order byte of the program counter (PCL) and ending with the CCR. After stacking, the SP points at the next available location on the stack which is the address that is one less than the address where the CCR was saved. The PC value that is stacked is the address of the instruction in the main program that would have executed next if the interrupt had not occurred.

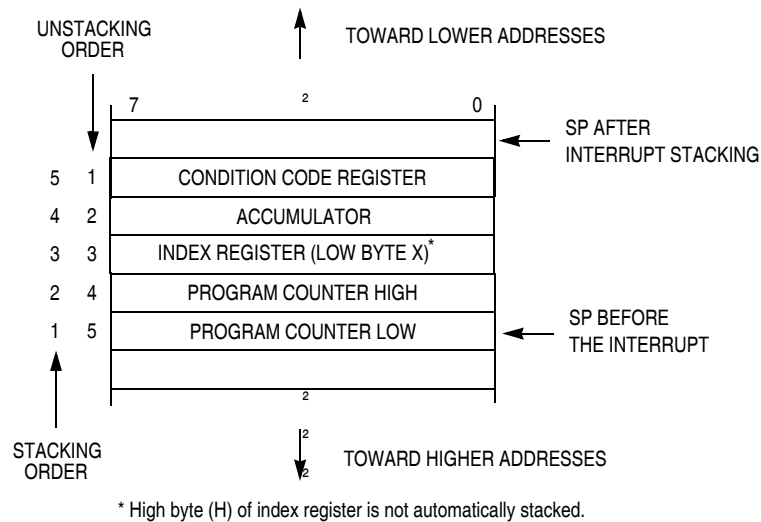


Figure 5-1. Interrupt Stack Frame

When an RTI instruction is executed, these values are recovered from the stack in reverse order. As part of the RTI sequence, the CPU fills the instruction pipeline by reading three bytes of program information, starting from the PC address recovered from the stack.

5.7.7 System Power Management Status and Control 2 Register (SPMSC2)

This register is used to report the status of the low voltage warning function, and to configure the stop mode behavior of the MCU.

	7	6	5	4	3	2	1	0
R	0	0	LVDV ¹	LVWV	PPDF	0	0	PPDC ²
W						PPDACK		
Power-on Reset:	0	0	0	0	0	0	0	0
LVD Reset:	0	0	u	u	0	0	0	0
Any other Reset:	0	0	u	u	0	0	0	0

= Unimplemented or Reserved
 u = Unaffected by reset

¹ This bit can be written only one time after power-on reset. Additional writes are ignored.

² This bit can be written only one time after reset. Additional writes are ignored.

Figure 5-9. System Power Management Status and Control 2 Register (SPMSC2)

Table 5-10. SPMSC2 Register Field Descriptions

Field	Description
5 LVDV	Low-Voltage Detect Voltage Select — This write-once bit selects the low voltage detect (LVD) trip point setting. It also selects the warning voltage range. See Table 5-11 .
4 LVWV	Low-Voltage Warning Voltage Select — This bit selects the low voltage warning (LVW) trip point voltage. See Table 5-11 .
3 PPDF	Partial Power Down Flag — This read-only status bit indicates that the MCU has recovered from stop2 mode. 0 MCU has not recovered from stop2 mode. 1 MCU recovered from stop2 mode.
2 PPDACK	Partial Power Down Acknowledge — Writing a 1 to PPDACK clears the PPDF bit
0 PPDC	Partial Power Down Control — This write-once bit controls whether stop2 or stop3 mode is selected. 0 Stop3 mode enabled. 1 Stop2, partial power down, mode enabled.

Table 5-11. LVD and LVW trip point typical values¹

LVDV:LVWV	LVW Trip Point	LVD Trip Point
0:0	$V_{LVW0} = 2.74 \text{ V}$	$V_{LVD0} = 2.56 \text{ V}$
0:1	$V_{LVW1} = 2.92 \text{ V}$	
1:0	$V_{LVW2} = 4.3 \text{ V}$	$V_{LVD1} = 4.0 \text{ V}$
1:1	$V_{LVW3} = 4.6 \text{ V}$	

¹ See Electrical Characteristics appendix for minimum and maximum values.

7.4.1 Reset Sequence

Reset can be caused by a power-on-reset (POR) event, internal conditions such as the COP (computer operating properly) watchdog, or by assertion of an external active-low reset pin. When a reset event occurs, the CPU immediately stops whatever it is doing (the MCU does not wait for an instruction boundary before responding to a reset event). For a more detailed discussion about how the MCU recognizes resets and determines the source, refer to the [Resets, Interrupts, and System Configuration](#) chapter.

The reset event is considered concluded when the sequence to determine whether the reset came from an internal source is done and when the reset pin is no longer asserted. At the conclusion of a reset event, the CPU performs a 6-cycle sequence to fetch the reset vector from 0xFFFFE and 0xFFFF and to fill the instruction queue in preparation for execution of the first program instruction.

7.4.2 Interrupt Sequence

When an interrupt is requested, the CPU completes the current instruction before responding to the interrupt. At this point, the program counter is pointing at the start of the next instruction, which is where the CPU should return after servicing the interrupt. The CPU responds to an interrupt by performing the same sequence of operations as for a software interrupt (SWI) instruction, except the address used for the vector fetch is determined by the highest priority interrupt that is pending when the interrupt sequence started.

The CPU sequence for an interrupt is:

1. Store the contents of PCL, PCH, X, A, and CCR on the stack, in that order.
2. Set the I bit in the CCR.
3. Fetch the high-order half of the interrupt vector.
4. Fetch the low-order half of the interrupt vector.
5. Delay for one free bus cycle.
6. Fetch three bytes of program information starting at the address indicated by the interrupt vector to fill the instruction queue in preparation for execution of the first instruction in the interrupt service routine.

After the CCR contents are pushed onto the stack, the I bit in the CCR is set to prevent other interrupts while in the interrupt service routine. Although it is possible to clear the I bit with an instruction in the interrupt service routine, this would allow nesting of interrupts (which is not recommended because it leads to programs that are difficult to debug and maintain).

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

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

7.4.3 Wait Mode Operation

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

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

7.4.4 Stop Mode Operation

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

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

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

7.4.5 BGND Instruction

The BGND instruction is new to the HCS08 compared to the M68HC08. BGND would not be used in normal user programs because it forces the CPU to stop processing user instructions and enter the active background mode. The only way to resume execution of the user program is through reset or by a host debug system issuing a GO, TRACE1, or TAGGO serial command through the background debug interface.

Software-based breakpoints can be set by replacing an opcode at the desired breakpoint address with the BGND opcode. When the program reaches this breakpoint address, the CPU is forced to active background mode rather than continuing the user program.

Table 7-2. Instruction Set Summary (Sheet 3 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR				
						VH	I	N	Z	C
BRA <i>rel</i>	Branch Always (if I = 1)	REL	20 rr	3	ppp	--	---	---	---	---
BRCLR <i>n,opr8a,rel</i>	Branch if Bit <i>n</i> in Memory Clear (if (Mn) = 0)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 dd rr 03 dd rr 05 dd rr 07 dd rr 09 dd rr 0B dd rr 0D dd rr 0F dd rr	5 5 5 5 5 5 5 5	rpppp rpppp rpppp rpppp rpppp rpppp rpppp rpppp	--	---	---	---	b
BRN <i>rel</i>	Branch Never (if I = 0)	REL	21 rr	3	ppp	--	---	---	---	---
BRSET <i>n,opr8a,rel</i>	Branch if Bit <i>n</i> in Memory Set (if (Mn) = 1)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 dd rr 02 dd rr 04 dd rr 06 dd rr 08 dd rr 0A dd rr 0C dd rr 0E dd rr	5 5 5 5 5 5 5 5	rpppp rpppp rpppp rpppp rpppp rpppp rpppp rpppp	--	---	---	---	b
BSET <i>n,opr8a</i>	Set Bit <i>n</i> in Memory (Mn ← 1)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 dd 12 dd 14 dd 16 dd 18 dd 1A dd 1C dd 1E dd	5 5 5 5 5 5 5 5	rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp	--	---	---	---	---
BSR <i>rel</i>	Branch to Subroutine PC ← (PC) + 0x0002 push (PCL); SP ← (SP) – 0x0001 push (PCH); SP ← (SP) – 0x0001 PC ← (PC) + <i>rel</i>	REL	AD rr	5	ssppp	--	---	---	---	---
CBEQ <i>opr8a,rel</i> CBEQA # <i>opr8i,rel</i> CBEQX # <i>opr8i,rel</i> CBEQ <i>opr8,X+,rel</i> CBEQ <i>,X+,rel</i> CBEQ <i>opr8,SP,rel</i>	Compare and... 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 dd rr 41 ii rr 51 ii rr 61 ff rr 71 rr 9E 61 ff rr	5 4 4 5 5 6	rpppp pppp pppp rpppp rfppp prpppp	--	---	---	---	---
CLC	Clear Carry Bit (C ← 0)	INH	98	1	p	--	---	---	---	0
CLI	Clear Interrupt Mask Bit (I ← 0)	INH	9A	1	p	--	0	---	---	---
CLR <i>opr8a</i> CLRA CLR X CLR X CLR <i>opr8,X</i> CLR <i>,X</i> CLR <i>opr8,SP</i>	Clear M ← 0x00 A ← 0x00 X ← 0x00 H ← 0x00 M ← 0x00 M ← 0x00 M ← 0x00	DIR INH INH INH IX1 IX SP1	3F dd 4F 5F 8C 6F ff 7F 9E 6F ff	5 1 1 1 5 4 6	rfwpp p p p rfwpp rfwp prfwpp	0	---	---	0	1

Table 7-2. Instruction Set Summary (Sheet 6 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR				
						VH	I	N	Z	C
MOV <i>opr8a,opr8a</i> MOV <i>opr8a,X+</i> MOV <i>#opr8i,opr8a</i> MOV <i>,X+,opr8a</i>	Move $(M)_{\text{destination}} \leftarrow (M)_{\text{source}}$ In IX+/DIR and DIR/IX+ Modes, H:X $\leftarrow (H:X) + 0x0001$	DIR/DIR DIR/IX+ IMM/DIR IX+/DIR	4E dd dd 5E dd 6E ii dd 7E dd	5 5 4 5	rfwpp rfwpp pwpp rfwpp	0-	-	b	b	-
MUL	Unsigned multiply $X:A \leftarrow (X) \times (A)$	INH	42	5	fffffp	-0	-	-	-	0
NEG <i>opr8a</i> NEGA NEGX NEG <i>oprx8,X</i> NEG <i>,X</i> NEG <i>oprx8,SP</i>	Negate $M \leftarrow (M) = 0x00 - (M)$ (Two's Complement) $A \leftarrow (A) = 0x00 - (A)$ $X \leftarrow (X) = 0x00 - (X)$ $M \leftarrow (M) = 0x00 - (M)$ $M \leftarrow (M) = 0x00 - (M)$ $M \leftarrow (M) = 0x00 - (M)$	DIR INH INH IX1 IX SP1	30 dd 40 50 60 ff 70 9E 60 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	b-	-	b	b	b
NOP	No Operation — Uses 1 Bus Cycle	INH	9D	1	p	--	-	-	-	-
NSA	Nibble Swap Accumulator $A \leftarrow (A[3:0]:A[7:4])$	INH	62	1	p	--	-	-	-	-
ORA <i>#opr8i</i> ORA <i>opr8a</i> ORA <i>opr16a</i> ORA <i>oprx16,X</i> ORA <i>oprx8,X</i> ORA <i>,X</i> ORA <i>oprx16,SP</i> ORA <i>oprx8,SP</i>	Inclusive OR Accumulator and Memory $A \leftarrow (A) (M)$	IMM DIR EXT IX2 IX1 IX SP2 SP1	AA ii BA dd CA hh ll DA ee ff EA ff FA 9E DA ee ff 9E EA ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0-	-	b	b	-
PSHA	Push Accumulator onto Stack Push (A); $SP \leftarrow (SP) - 0x0001$	INH	87	2	sp	--	-	-	-	-
PSHH	Push H (Index Register High) onto Stack Push (H); $SP \leftarrow (SP) - 0x0001$	INH	8B	2	sp	--	-	-	-	-
PSHX	Push X (Index Register Low) onto Stack Push (X); $SP \leftarrow (SP) - 0x0001$	INH	89	2	sp	--	-	-	-	-
PULA	Pull Accumulator from Stack $SP \leftarrow (SP + 0x0001)$; Pull (A)	INH	86	3	ufp	--	-	-	-	-
PULH	Pull H (Index Register High) from Stack $SP \leftarrow (SP + 0x0001)$; Pull (H)	INH	8A	3	ufp	--	-	-	-	-
PULX	Pull X (Index Register Low) from Stack $SP \leftarrow (SP + 0x0001)$; Pull (X)	INH	88	3	ufp	--	-	-	-	-
ROL <i>opr8a</i> ROLA ROLX ROL <i>oprx8,X</i> ROL <i>,X</i> ROL <i>oprx8,SP</i>	Rotate Left through Carry 	DIR INH INH IX1 IX SP1	39 dd 49 59 69 ff 79 9E 69 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	b-	-	b	b	b
ROR <i>opr8a</i> RORA RORX ROR <i>oprx8,X</i> ROR <i>,X</i> ROR <i>oprx8,SP</i>	Rotate Right through Carry 	DIR INH INH IX1 IX SP1	36 dd 46 56 66 ff 76 9E 66 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	b-	-	b	b	b

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

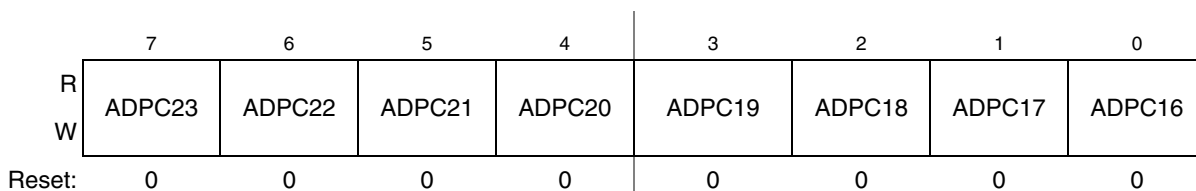
Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR				
						VH	I	N	Z	C
SUB #opr8i SUB opr8a SUB opr16a SUB oprx16,X SUB oprx8,X SUB ,X SUB oprx16,SP SUB oprx8,SP	Subtract $A \leftarrow (A) - (M)$	IMM DIR EXT IX2 IX1 IX SP2 SP1	A0 ii B0 dd C0 hh ll D0 ee ff E0 ff F0 9E D0 ee ff 9E E0 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp		b -	-	b b b	
SWI	Software Interrupt $PC \leftarrow (PC) + 0x0001$ Push (PCL); $SP \leftarrow (SP) - 0x0001$ Push (PCH); $SP \leftarrow (SP) - 0x0001$ Push (X); $SP \leftarrow (SP) - 0x0001$ Push (A); $SP \leftarrow (SP) - 0x0001$ Push (CCR); $SP \leftarrow (SP) - 0x0001$ $I \leftarrow 1$; PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	INH	83	11	sssssvvfppp	--	1	--	--	--
TAP	Transfer Accumulator to CCR $CCR \leftarrow (A)$	INH	84	1	p	b b		b b b b		
TAX	Transfer Accumulator to X (Index Register Low) $X \leftarrow (A)$	INH	97	1	p	--		--	--	--
TPA	Transfer CCR to Accumulator $A \leftarrow (CCR)$	INH	85	1	p	--		--	--	--
TST opr8a TSTA TSTX TST oprx8,X TST ,X TST oprx8,SP	Test for Negative or Zero (M) – 0x00 (A) – 0x00 (X) – 0x00 (M) – 0x00 (M) – 0x00 (M) – 0x00	DIR INH INH IX1 IX SP1	3D dd 4D 5D 6D ff 7D 9E 6D ff	4 1 1 4 3 5	rfpp p p rfpp rfp prfpp	0 -		-	b b -	
TSX	Transfer SP to Index Reg. $H:X \leftarrow (SP) + 0x0001$	INH	95	2	fp	--		--	--	--
TXA	Transfer X (Index Reg. Low) to Accumulator $A \leftarrow (X)$	INH	9F	1	p	--		--	--	--

Table 9-10. APCTL2 Register Field Descriptions (continued)

Field	Description
1 ADPC9	ADC Pin Control 9 — ADPC9 is used to control the pin associated with channel AD9. 0 AD9 pin I/O control enabled 1 AD9 pin I/O control disabled
0 ADPC8	ADC Pin Control 8 — ADPC8 is used to control the pin associated with channel AD8. 0 AD8 pin I/O control enabled 1 AD8 pin I/O control disabled

9.3.10 Pin Control 3 Register (APCTL3)

APCTL3 is used to control channels 16–23 of the ADC module.


Figure 9-13. Pin Control 3 Register (APCTL3)
Table 9-11. APCTL3 Register Field Descriptions

Field	Description
7 ADPC23	ADC Pin Control 23 — ADPC23 is used to control the pin associated with channel AD23. 0 AD23 pin I/O control enabled 1 AD23 pin I/O control disabled
6 ADPC22	ADC Pin Control 22 — ADPC22 is used to control the pin associated with channel AD22. 0 AD22 pin I/O control enabled 1 AD22 pin I/O control disabled
5 ADPC21	ADC Pin Control 21 — ADPC21 is used to control the pin associated with channel AD21. 0 AD21 pin I/O control enabled 1 AD21 pin I/O control disabled
4 ADPC20	ADC Pin Control 20 — ADPC20 is used to control the pin associated with channel AD20. 0 AD20 pin I/O control enabled 1 AD20 pin I/O control disabled
3 ADPC19	ADC Pin Control 19 — ADPC19 is used to control the pin associated with channel AD19. 0 AD19 pin I/O control enabled 1 AD19 pin I/O control disabled
2 ADPC18	ADC Pin Control 18 — ADPC18 is used to control the pin associated with channel AD18. 0 AD18 pin I/O control enabled 1 AD18 pin I/O control disabled

Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

11.3.1 IIC Address Register (IICA)

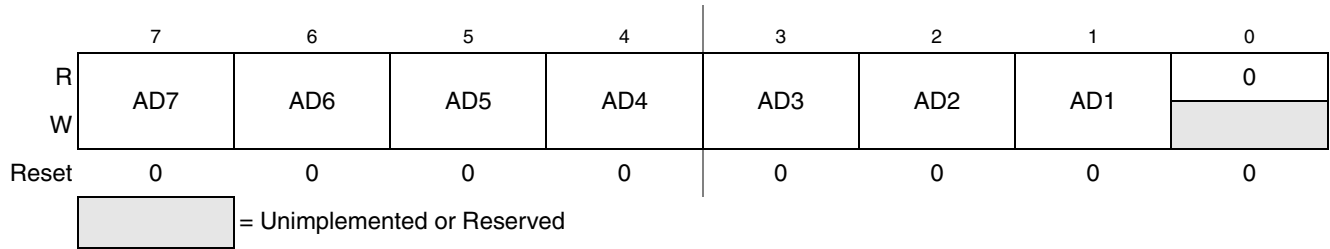


Figure 11-3. IIC Address Register (IICA)

Table 11-2. IICA Field Descriptions

Field	Description
7-1 AD[7:1]	Slave Address. The AD[7:1] field contains the slave address to be used by the IIC module. This field is used on the 7-bit address scheme and the lower seven bits of the 10-bit address scheme.

11.3.2 IIC Frequency Divider Register (IICF)

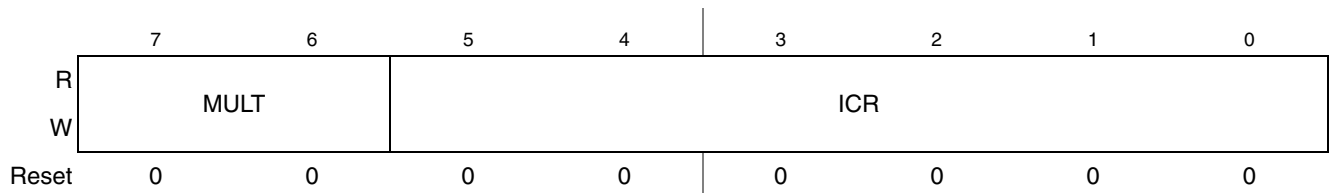


Figure 11-4. IIC Frequency Divider Register (IICF)

11.3.3 IIC Control Register (IICC1)

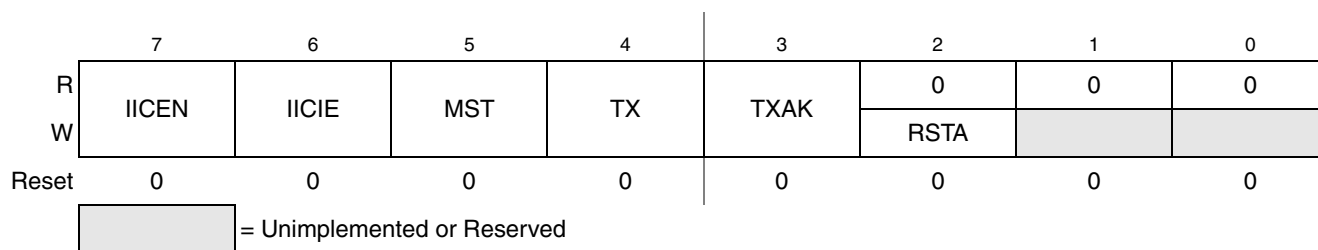


Figure 11-5. IIC Control Register (IICC1)

Table 11-6. IICC1 Field Descriptions

Field	Description
7 IICEN	IIC Enable. The IICEN bit determines whether the IIC module is enabled. 0 IIC is not enabled 1 IIC is enabled
6 IICIE	IIC Interrupt Enable. The IICIE bit determines whether an IIC interrupt is requested. 0 IIC interrupt request not enabled 1 IIC interrupt request enabled
5 MST	Master Mode Select. The MST bit changes from a 0 to a 1 when a start signal is generated on the bus and master mode is selected. When this bit changes from a 1 to a 0 a stop signal is generated and the mode of operation changes from master to slave. 0 Slave mode 1 Master mode
4 TX	Transmit Mode Select. The TX bit selects the direction of master and slave transfers. In master mode, this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit is always high. When addressed as a slave, this bit should be set by software according to the SRW bit in the status register. 0 Receive 1 Transmit
3 TXAK	Transmit Acknowledge Enable. This bit specifies the value driven onto the SDA during data acknowledge cycles for master and slave receivers. 0 An acknowledge signal is sent out to the bus after receiving one data byte 1 No acknowledge signal response is sent
2 RSTA	Repeat start. Writing a 1 to this bit generates a repeated start condition provided it is the current master. This bit is always read as cleared. Attempting a repeat at the wrong time results in loss of arbitration.

11.3.4 IIC Status Register (IICS)

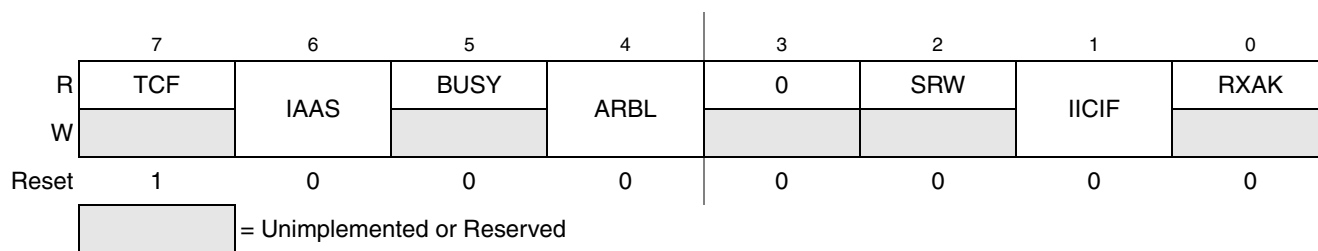


Figure 11-6. IIC Status Register (IICS)

11.4.1.5 Repeated Start Signal

As shown in [Figure 11-9](#), 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.

11.4.1.6 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.

11.4.1.7 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 11-10](#)). 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.

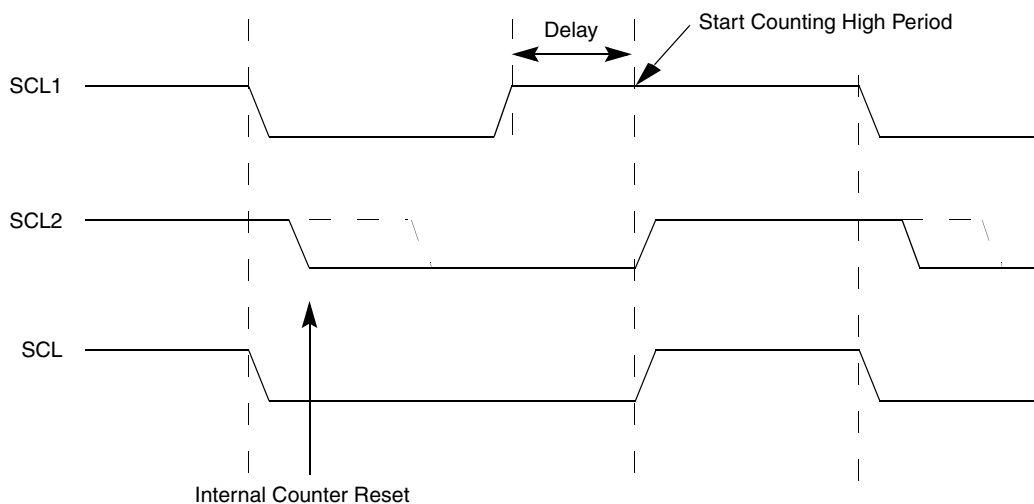
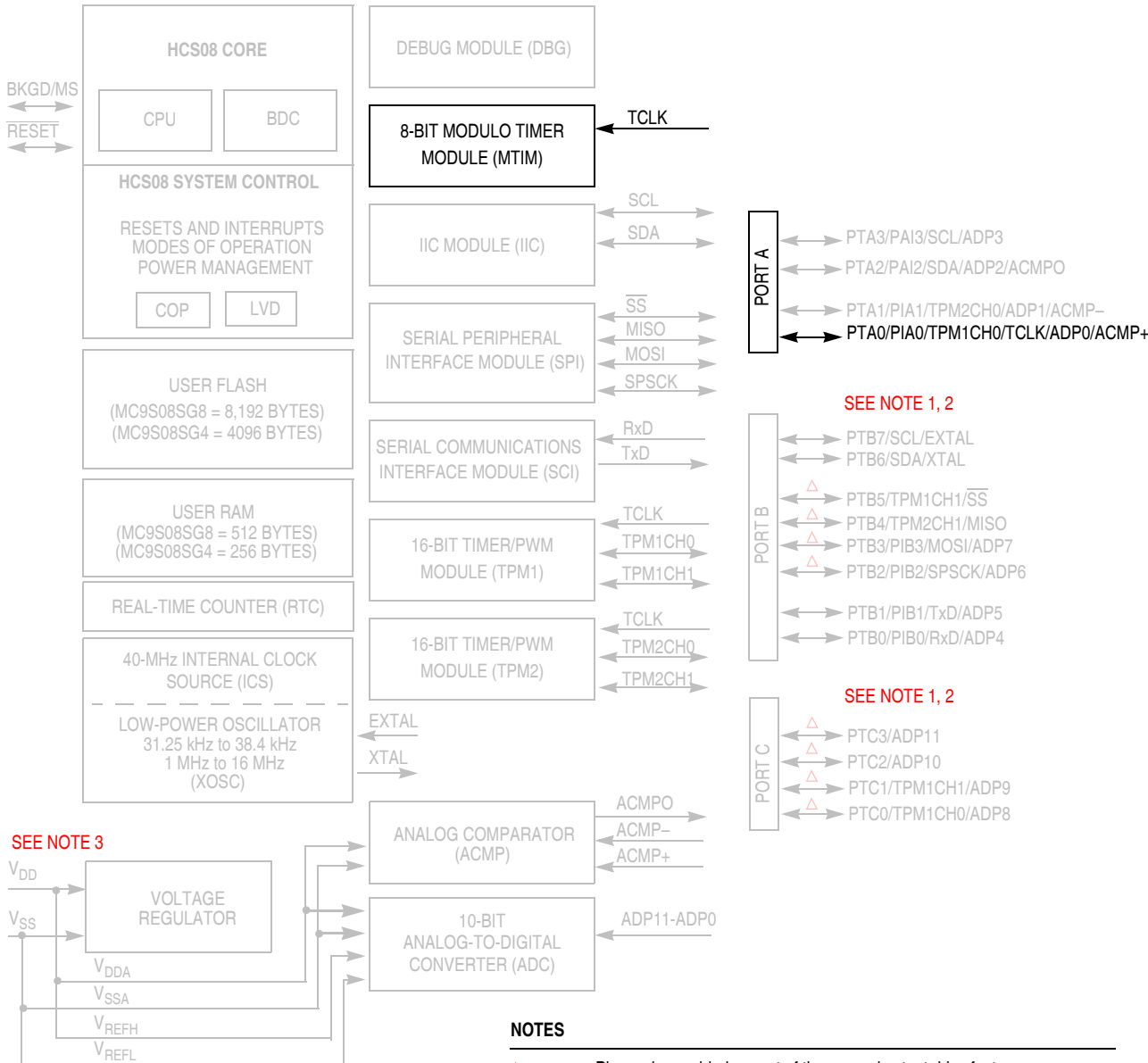


Figure 11-10. IIC Clock Synchronization



- NOTES**
- △ = Pin can be enabled as part of the ganged output drive feature.
 - NOTE 1: Port B not available on 8-pin packages
 - NOTE 2: Port C not available on 8-pin or 16-pin packages.
 - NOTE 3: V_{DDA}/V_{REFH} and V_{SSA}/V_{REFL} are double bonded to V_{DD} and V_{SS} respectively.

Figure 12-1. MC9S08SG8 Block Diagram with MTIM Module Highlighted

14.2 Register Definition

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

Refer to the direct-page register summary in the [Memory](#) chapter of this data sheet for the absolute address assignments for all SCI registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

14.2.1 SCI Baud Rate Registers (SCIBDH, SCIBDL)

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

SCIBDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled (RE or TE bits in SCIC2 are written to 1).

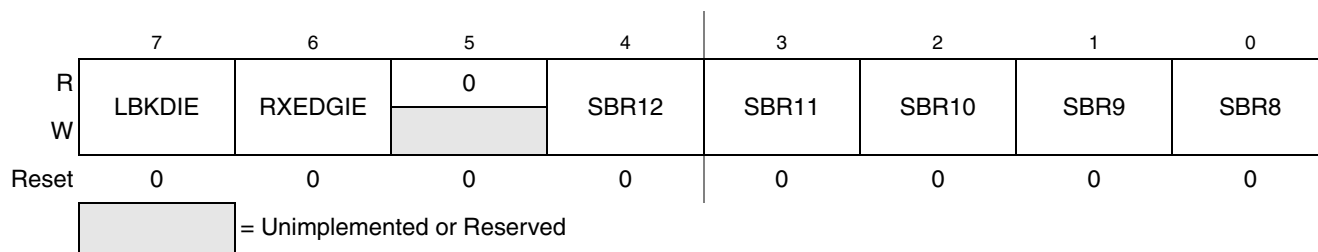
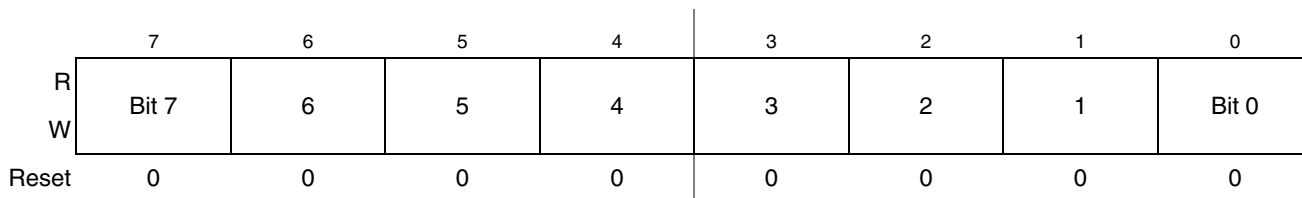


Figure 14-4. SCI Baud Rate Register (SCIBDH)

Table 14-1. SCIBDH Field Descriptions

Field	Description
7 LBKDIE	LIN Break Detect Interrupt Enable (for LBKDIF) 0 Hardware interrupts from LBKDIF disabled (use polling). 1 Hardware interrupt requested when LBKDIF flag is 1.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable (for RXEDGIF) 0 Hardware interrupts from RXEDGIF disabled (use polling). 1 Hardware interrupt requested when RXEDGIF flag is 1.
4:0 SBR[12:8]	Baud Rate Modulo Divisor — The 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 = $BUSCLK/(16 \times BR)$. See also BR bits in Table 14-2 .





Reset the TPM counter before writing to the TPM modulo registers to avoid confusion about when the first counter overflow will occur.

16.3.4 TPM Channel n Status and Control Register (TPMxCnSC)

TPMxCnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

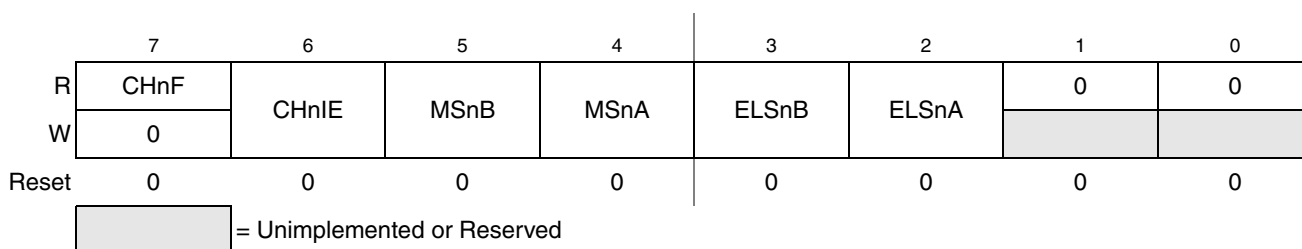


Figure 16-12. TPM Channel n Status and Control Register (TPMxCnSC)

Table 16-8. TPMxCnSC Field Descriptions

Field	Description
7 CHnF	Channel n flag. When channel n is an input-capture channel, this read/write bit is set when an active edge occurs on the channel n pin. When channel n is an output compare or edge-aligned/center-aligned PWM channel, CHnF is set when the value in the TPM counter registers matches the value in the TPM channel n value registers. When channel n is an edge-aligned/center-aligned PWM channel and the duty cycle is set to 0% or 100%, CHnF will not be set even when the value in the TPM counter registers matches the value in the TPM channel n value registers. A corresponding interrupt is requested when CHnF is set and interrupts are enabled (CHnIE = 1). Clear CHnF by reading TPMxCnSC while CHnF is set and then writing a logic 0 to CHnF. If another interrupt request occurs before the clearing sequence is complete, the sequence is reset so CHnF remains set after the clear sequence completed for the earlier CHnF. This is done so a CHnF interrupt request cannot be lost due to clearing a previous CHnF. Reset clears the CHnF bit. Writing a logic 1 to CHnF has no effect. 0 No input capture or output compare event occurred on channel n 1 Input capture or output compare event on channel n
6 CHnIE	Channel n interrupt enable. This read/write bit enables interrupts from channel n. Reset clears CHnIE. 0 Channel n interrupt requests disabled (use for software polling) 1 Channel n interrupt requests enabled
5 MSnB	Mode select B for TPM channel n. When CPWMS=0, MSnB=1 configures TPM channel n for edge-aligned PWM mode. Refer to the summary of channel mode and setup controls in Table 16-9 .

17.4.3.7 Debug Control Register (DBGC)

This register can be read or written at any time.

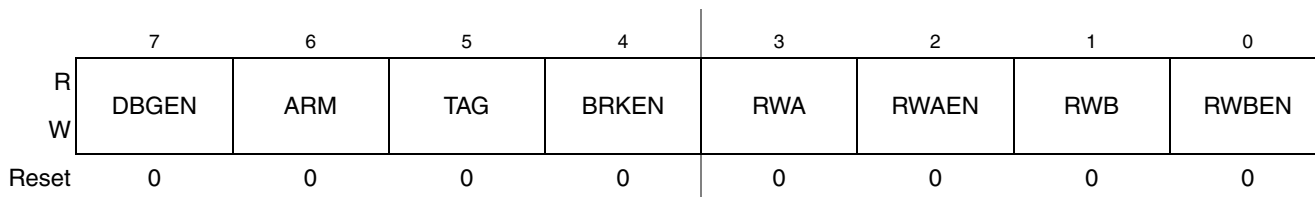


Figure 17-7. Debug Control Register (DBGC)

Table 17-4. DBGC Register Field Descriptions

Field	Description
7 DBGEN	Debug Module Enable — 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	Arm Control — 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	Tag/Force Select — 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	Break Enable — 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	R/W Comparison Value for Comparator A — 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	Enable R/W for Comparator A — 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	R/W Comparison Value for Comparator 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	Enable R/W for Comparator 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

A.13 FLASH Specifications

This section provides details about program/erase times and program-erase endurance for the FLASH memory.

Program and erase operations do not require any special power sources other than the normal V_{DD} supply. For more detailed information about program/erase operations, see the Memory section.

Table A-16. FLASH Characteristics

Num	C	Characteristic	Symbol	Min	Typical	Max	Unit	Temp Rated ¹	
								Stand ard	AEC Grade 0
1	—	Supply voltage for program/erase	$V_{\text{prog/erase}}$	2.7		5.5	V	x	x
2	—	Supply voltage for read operation	V_{Read}	2.7		5.5	V	x	x
3	—	Internal FCLK frequency ²	f_{FCLK}	150		200	kHz	x	x
4	—	Internal FCLK period ($1/f_{\text{FCLK}}$)	t_{FcyC}	5		6.67	μs	x	x
5	—	Byte program time (random location) ³	t_{prog}		9		t_{FcyC}	x	x
6	—	Byte program time (burst mode) ²	t_{Burst}		4		t_{FcyC}	x	x
7	—	Page erase time ²	t_{Page}		4000		t_{FcyC}	x	x
8	—	Mass erase time ²	t_{Mass}		20,000		t_{FcyC}	x	x
9	C	Program/erase endurance ⁴ T_L to $T_H = -40^\circ\text{C}$ to $+125^\circ\text{C}$ T_L to $T_H = -40^\circ\text{C}$ to $+150^\circ\text{C}$ $T = 25^\circ\text{C}$	n_{FLPE}	10,000 10,000 10,000	— — 100,000	— — —	cycles	x — x	— x x
10	C	Data retention ⁵	$t_{\text{D_ret}}$	15	100	—	years	x	x

¹ Electrical characteristics only apply to the temperature rated devices marked with x.

² The frequency of this clock is controlled by a software setting.

³ These values are hardware state machine controlled. User code does not need to count cycles. This information supplied for calculating approximate time to program and erase.

⁴ Typical endurance for FLASH is based on the intrinsic bit cell performance. For additional information on how Freescale defines typical endurance, please refer to Engineering Bulletin EB619/D, Typical Endurance for Nonvolatile Memory.

⁵ Typical data retention values are based on intrinsic capability of the technology measured at high temperature and de-rated to 25°C using the Arrhenius equation. For additional information on how Freescale defines typical data retention, please refer to Engineering Bulletin EB618/D, Typical Data Retention for Nonvolatile Memory.