



Welcome to [E-XFL.COM](#)

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

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

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

Details

Product Status	Active
Core Processor	S08
Core Size	8-Bit
Speed	40MHz
Connectivity	I ² C, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	34
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 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s08aw16mfge

Section Number	Title	Page
8.4.4	FLL Engaged Internal Unlocked	143
8.4.5	FLL Engaged Internal Locked	143
8.4.6	FLL Bypassed, External Clock (FBE) Mode	143
8.4.7	FLL Engaged, External Clock (FEE) Mode	143
8.4.8	FLL Lock and Loss-of-Lock Detection	144
8.4.9	FLL Loss-of-Clock Detection	145
8.4.10	Clock Mode Requirements	146
8.4.11	Fixed Frequency Clock	147
8.4.12	High Gain Oscillator	147
8.5	Initialization/Application Information	147
8.5.1	Introduction	147
8.5.2	Example #1: External Crystal = 32 kHz, Bus Frequency = 4.19 MHz	149
8.5.3	Example #2: External Crystal = 4 MHz, Bus Frequency = 20 MHz	151
8.5.4	Example #3: No External Crystal Connection, 5.4 MHz Bus Frequency	153
8.5.5	Example #4: Internal Clock Generator Trim	155

Chapter 9 Keyboard Interrupt (S08KBIV1)

9.1	Introduction	157
9.2	Keyboard Pin Sharing	157
9.3	Features	158
9.3.1	KBI Block Diagram	160
9.4	Register Definition	160
9.4.1	KBI Status and Control Register (KBI1SC)	161
9.4.2	KBI Pin Enable Register (KBI1PE)	162
9.5	Functional Description	162
9.5.1	Pin Enables	162
9.5.2	Edge and Level Sensitivity	162
9.5.3	KBI Interrupt Controls	163

Chapter 10 Timer/PWM (S08TPMV2)

10.1	Introduction	165
10.2	Features	165
10.2.1	Features	167
10.2.2	Block Diagram	167
10.3	External Signal Description	169
10.3.1	External TPM Clock Sources	169
10.3.2	TPMxCHn — TPMx Channel n I/O Pins	169
10.4	Register Definition	169
10.4.1	Timer x Status and Control Register (TPMxSC)	170
10.4.2	Timer x Counter Registers (TPMxCNTH:TPMxCNTL)	171

When IRQ is configured as the IRQ input and is set to detect rising edges, a pulldown device rather than a pullup device is enabled.

In EMC-sensitive applications, an external RC filter is recommended on the IRQ pin. See Figure 2-4 for an example.

2.3.7 General-Purpose I/O and Peripheral Ports

The remaining pins are shared among general-purpose I/O and on-chip peripheral functions such as timers and serial I/O systems. Immediately after reset, all of these pins are configured as high-impedance general-purpose inputs with internal pullup devices disabled.

NOTE

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

For information about controlling these pins as general-purpose I/O pins, see Chapter 6, “Parallel Input/Output.” For information about how and when on-chip peripheral systems use these pins, refer to the appropriate chapter from Table 2-1.

Table 2-1. Pin Sharing Priority

Lowest <- Pin Function Priority -> Highest			Reference ¹
Port Pins	Alternate Function	Alternate Function	
PTB7–PTB0	AD1P7–AD1P0		Chapter 14, “Analog-to-Digital Converter (S08ADC10V1)”
PTC5, PTC3	RxD2–TxD2		Chapter 11, “Serial Communications Interface (S08SCIV2)”
PTC2	MCLK		Chapter 5, “Resets, Interrupts, and System Configuration”
PTC1–PTC0	SCL1–SDA1		Chapter 13, “Inter-Integrated Circuit (S08IICV1)”
PTD7	KBI1P7	AD1P15	Chapter 14, “Analog-to-Digital Converter (S08ADC10V1)” Chapter 9, “Keyboard Interrupt (S08KBIV1)”
PTD6	TPM1CLK	AD1P14	Chapter 14, “Analog-to-Digital Converter (S08ADC10V1)” Chapter 10, “Timer/PWM (S08TPMV2)”
PTD5	AD1P13	AD1P13	Chapter 14, “Analog-to-Digital Converter (S08ADC10V1)”
PTD4	TPM2CLK	AD1P12	Chapter 14, “Analog-to-Digital Converter (S08ADC10V1)” Chapter 10, “Timer/PWM (S08TPMV2)”
PTD3–PTD2	KBI1P6–KBI1P5	AD1P11–AD1P10	Chapter 14, “Analog-to-Digital Converter (S08ADC10V1)” Chapter 9, “Keyboard Interrupt (S08KBIV1)”
PTD1–PTD0	AD1P9–AD1P8		Chapter 14, “Analog-to-Digital Converter (S08ADC10V1)”
PTE7 PTE6 PTE5 PTE4	SPSCK1 MOSI1 MISO1 SS1		Chapter 12, “Serial Peripheral Interface (S08SPIV3)”
PTE3–PTE2	TPM1CH1– TPM1CH0		Chapter 10, “Timer/PWM (S08TPMV2)”
PTE1–PTE0	RxD1–TxD1		Chapter 11, “Serial Communications Interface (S08SCIV2)”
PTF5–PTF4	TPM2CH1– TPM2CH0		Chapter 10, “Timer/PWM (S08TPMV2)”

Table 2-1. Pin Sharing Priority

Lowest <- Pin Function Priority -> Highest			Reference ¹
Port Pins	Alternate Function	Alternate Function	
PTF3–PTF0	TPM1CH5–TPM1CH2		Chapter 10, “Timer/PWM (S08TPMV2)”
PTG4–PTG0	KBI1P4–KBI1P0		Chapter 9, “Keyboard Interrupt (S08KBIV1)”
PTG6–PTG5	EXTAL–XTAL		Chapter 8, “Internal Clock Generator (S08ICGV4)”

¹ See the listed chapter for information about modules that share these pins.

When an on-chip peripheral system is controlling a pin, data direction control bits still determine what is read from port data registers even though the peripheral module controls the pin direction by controlling the enable for the pin’s output buffer. See the Chapter 6, “Parallel Input/Output” chapter for more details.

Pullup enable bits for each input pin control whether on-chip pullup devices are enabled whenever the pin is acting as an input even if it is being controlled by an on-chip peripheral module. When the PTD7, PTD3, PTD2, and PTG4 pins are controlled by the KBI module and are configured for rising-edge/high-level sensitivity, the pullup enable control bits enable pulldown devices rather than pullup devices.

NOTE

When an alternative function is first enabled it is possible to get a spurious edge to the module, user software should clear out any associated flags before interrupts are enabled. Table 2-1 illustrates the priority if multiple modules are enabled. The highest priority module will have control over the pin. Selecting a higher priority pin function with a lower priority function already enabled can cause spurious edges to the lower priority module. It is recommended that all modules that share a pin be disabled before enabling another module.

the CPU executes a STOP instruction, the MCU will not enter either of the stop modes and an illegal opcode reset is forced. The stop modes are selected by setting the appropriate bits in SPMSC2.

HCS08 devices that are designed for low voltage operation (1.8V to 3.6V) also include stop1 mode. The MC9S08AW60 Series family of devices does not include stop1 mode.

Table 3-1 summarizes the behavior of the MCU in each of the stop modes.

Table 3-1. Stop Mode Behavior

Mode	PPDC	CPU, Digital Peripherals, FLASH	RAM	ICG	ADC1	Regulator	I/O Pins	RTI
Stop2	1	Off	Standby	Off	Disabled	Standby	States held	Optionally on
Stop3	0	Standby	Standby	Off ¹	Optionally on	Standby	States held	Optionally on

¹ Crystal oscillator can be configured to run in stop3. Please see the ICG registers.

3.6.1 Stop2 Mode

The stop2 mode provides very low standby power consumption and maintains the contents of RAM and the current state of all of the I/O pins. To enter stop2, the user must execute a STOP instruction with stop2 selected (PPDC = 1) and stop mode enabled (STOPE = 1). In addition, the LVD must not be enabled to operate in stop (LVDSE = 0 or LVDE = 0). If the LVD is enabled in stop, then the MCU enters stop3 upon the execution of the STOP instruction regardless of the state of PPDC.

Before entering stop2 mode, the user must save the contents of the I/O port registers, as well as any other memory-mapped registers which they want to restore after exit of stop2, to locations in RAM. Upon exit of stop2, these values can be restored by user software before pin latches are opened.

When the MCU is in stop2 mode, all internal circuits that are powered from the voltage regulator are turned off, except for the RAM. The voltage regulator is in a low-power standby state, as is the ADC. Upon entry into stop2, the states of the I/O pins are latched. The states are held while in stop2 mode and after exiting stop2 mode until a logic 1 is written to PPDAK in SPMSC2.

Exit from stop2 is done by asserting either of the wake-up pins: $\overline{\text{RESET}}$ or IRQ, or by an RTI interrupt. IRQ is always an active low input when the MCU is in stop2, regardless of how it was configured before entering stop2.

NOTE

Although this IRQ pin is automatically configured as active low input, the pullup associated with the IRQ pin is not automatically enabled. Therefore, if an external pullup is not used, the internal pullup must be enabled by setting IRQPE in IRQSC.

Upon wake-up from stop2 mode, the MCU will start up as from a power-on reset (POR) except pin states remain latched. The CPU will take the reset vector. The system and all peripherals will be in their default reset states and must be initialized.

Table 3-4. Stop Mode Behavior (continued)

Peripheral	Mode	
	Stop2	Stop3
KBI	Off	Optionally On ³
RTI	Optionally On ⁴	Optionally On ⁴
SCI	Off	Standby
SPI	Off	Standby
TPM	Off	Standby
Voltage Regulator	Standby	Standby
I/O Pins	States Held	States Held

¹ Requires the asynchronous ADC clock and LVD to be enabled, else in standby.

² OSCSTEN set in ICSC1, else in standby. For high frequency range (RANGE in ICSC2 set) requires the LVD to also be enabled in stop3.

³ During stop3, KBI pins that are enabled continue to function as interrupt sources that are capable of waking the MCU from stop3.

⁴ This RTI can be enabled to run in stop2 or stop3 with the internal RTI clock source (RTICKS = 0, in SRTISC). The RTI also can be enabled to run in stop3 with the external clock source (RTICKS = 1 and OSCSTEN = 1).

5.9.7 System Real-Time Interrupt Status and Control Register (SRTISC)

This register contains one read-only status flag, one write-only acknowledge bit, three read/write delay selects, and three unimplemented bits, which always read 0.

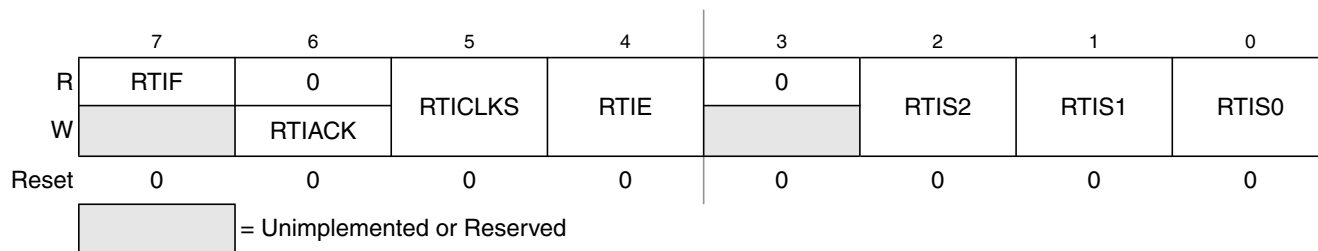


Figure 5-9. System RTI Status and Control Register (SRTISC)

Table 5-9. SRTISC Register Field Descriptions

Field	Description
7 RTIF	Real-Time Interrupt Flag — This read-only status bit indicates the periodic wakeup timer has timed out. 0 Periodic wakeup timer not timed out. 1 Periodic wakeup timer timed out.
6 RTIACK	Real-Time Interrupt Acknowledge — This write-only bit is used to acknowledge real-time interrupt request (write 1 to clear RTIF). Writing 0 has no meaning or effect. Reads always return logic 0.
5 RTICLKs	Real-Time Interrupt Clock Select — This read/write bit selects the clock source for the real-time interrupt. 0 Real-time interrupt request clock source is internal 1-kHz oscillator. 1 Real-time interrupt request clock source is external clock.
4 RTIE	Real-Time Interrupt Enable — This read-write bit enables real-time interrupts. 0 Real-time interrupts disabled. 1 Real-time interrupts enabled.
2:0 RTIS[2:0]	Real-Time Interrupt Delay Selects — These read/write bits select the wakeup delay for the RTI. The clock source for the real-time interrupt is a self-clocked source which oscillates at about 1 kHz, is independent of other MCU clock sources. Using external clock source the delays will be crystal frequency divided by value in RTIS2:RTIS1:RTIS0. See Table 5-10.

Table 5-10. Real-Time Interrupt Frequency

RTIS2:RTIS1:RTIS0	1-kHz Clock Source Delay ¹	Using External Clock Source Delay (Crystal Frequency)
0:0:0	Disable periodic wakeup timer	Disable periodic wakeup timer
0:0:1	8 ms	divide by 256
0:1:0	32 ms	divide by 1024
0:1:1	64 ms	divide by 2048
1:0:0	128 ms	divide by 4096
1:0:1	256 ms	divide by 8192
1:1:0	512 ms	divide by 16384
1:1:1	1.024 s	divide by 32768

¹ Normal values are shown in this column based on $f_{RTI} = 1$ kHz. See Appendix A, “Electrical Characteristics and Timing Specifications,” f_{RTI} for the tolerance on these values.

- 0 = Bit forced to 0
- 1 = Bit forced to 1
- = Bit set or cleared according to results of operation
- U = Undefined after the operation

Machine coding notation

- dd = Low-order 8 bits of a direct address 0x0000–0x00FF (high byte assumed to be 0x00)
- ee = Upper 8 bits of 16-bit offset
- ff = Lower 8 bits of 16-bit offset or 8-bit offset
- ii = One byte of immediate data
- jj = High-order byte of a 16-bit immediate data value
- kk = Low-order byte of a 16-bit immediate data value
- hh = High-order byte of 16-bit extended address
- ll = Low-order byte of 16-bit extended address
- rr = Relative offset

Source form

Everything in the source forms columns, *except expressions in italic characters*, is literal information that must appear in the assembly source file exactly as shown. The initial 3- to 5-letter mnemonic is always a literal expression. All commas, pound signs (#), parentheses, and plus signs (+) are literal characters.

- n* — Any label or expression that evaluates to a single integer in the range 0–7
- opr8i* — Any label or expression that evaluates to an 8-bit immediate value
- opr16i* — Any label or expression that evaluates to a 16-bit immediate value
- opr8a* — Any label or expression that evaluates to an 8-bit value. The instruction treats this 8-bit value as the low order 8 bits of an address in the direct page of the 64-Kbyte address space (0x00xx).
- opr16a* — Any label or expression that evaluates to a 16-bit value. The instruction treats this value as an address in the 64-Kbyte address space.
- opr8* — Any label or expression that evaluates to an unsigned 8-bit value, used for indexed addressing
- opr16* — Any label or expression that evaluates to a 16-bit value. Because the HCS08 has a 16-bit address bus, this can be either a signed or an unsigned value.
- rel* — Any label or expression that refers to an address that is within –128 to +127 locations from the next address after the last byte of object code for the current instruction. The assembler will calculate the 8-bit signed offset and include it in the object code for this instruction.

Address modes

- INH = Inherent (no operands)
- IMM = 8-bit or 16-bit immediate
- DIR = 8-bit direct
- EXT = 16-bit extended

- IX = 16-bit indexed no offset
- IX+ = 16-bit indexed no offset, post increment (CBEQ and MOV only)
- IX1 = 16-bit indexed with 8-bit offset from H:X
- IX1+ = 16-bit indexed with 8-bit offset, post increment (CBEQ only)
- IX2 = 16-bit indexed with 16-bit offset from H:X
- REL = 8-bit relative offset
- SP1 = Stack pointer with 8-bit offset
- SP2 = Stack pointer with 16-bit offset

Table 7-2. HCS08 Instruction Set Summary (Sheet 1 of 7)

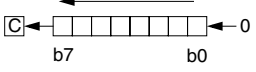
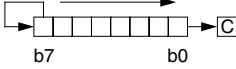
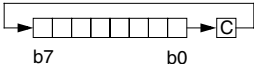
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles ¹
			V	H	I	N	Z	C				
ADC #opr8i ADC opr8a ADC opr16a ADC oprx16,X ADC oprx8,X ADC ,X ADC oprx16,SP ADC oprx8,SP	Add with Carry	$A \leftarrow (A) + (M) + (C)$	↑	↑	—	↑	↑	↑	IMM DIR EXT IX2 IX1 IX SP2 SP1	A9 B9 C9 D9 E9 F9 9ED9 9EE9	ii dd hh ll ee ff ff ff ee ff ff	2 3 4 4 3 3 5 4
ADD #opr8i ADD opr8a ADD opr16a ADD oprx16,X ADD oprx8,X ADD ,X ADD oprx16,SP ADD oprx8,SP	Add without Carry	$A \leftarrow (A) + (M)$	↑	↑	—	↑	↑	↑	IMM DIR EXT IX2 IX1 IX SP2 SP1	AB BB CB DB EB FB 9EDB 9EEB	ii dd hh ll ee ff ff ff ee ff ff	2 3 4 4 3 3 5 4
AIS #opr8i	Add Immediate Value (Signed) to Stack Pointer	$SP \leftarrow (SP) + (M)$ M is sign extended to a 16-bit value	—	—	—	—	—	—	IMM	A7	ii	2
AIX #opr8i	Add Immediate Value (Signed) to Index Register (H:X)	$H:X \leftarrow (H:X) + (M)$ M is sign extended to a 16-bit value	—	—	—	—	—	—	IMM	AF	ii	2
AND #opr8i AND opr8a AND opr16a AND oprx16,X AND oprx8,X AND ,X AND oprx16,SP AND oprx8,SP	Logical AND	$A \leftarrow (A) \& (M)$	0	—	—	↑	↑	—	IMM DIR EXT IX2 IX1 IX SP2 SP1	A4 B4 C4 D4 E4 F4 9ED4 9EE4	ii dd hh ll ee ff ff ff ee ff ff	2 3 4 4 3 3 5 4
ASL opr8a ASLA ASLX ASL oprx8,X ASL ,X ASL oprx8,SP	Arithmetic Shift Left (Same as LSL)		↑	—	—	↑	↑	↑	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	5 1 1 5 4 6
ASR opr8a ASRA ASRX ASR oprx8,X ASR ,X ASR oprx8,SP	Arithmetic Shift Right		↑	—	—	↑	↑	↑	DIR INH INH IX1 IX SP1	37 47 57 67 77 9E67	dd ff ff	5 1 1 5 4 6
BCC rel	Branch if Carry Bit Clear	Branch if (C) = 0	—	—	—	—	—	—	REL	24	rr	3

Table 7-2. HCS08 Instruction Set Summary (Sheet 6 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles ¹
			V	H	I	N	Z	C				
ROR <i>opr8a</i> RORA RORX ROR <i>opr8,X</i> ROR <i>,X</i> ROR <i>opr8,SP</i>	Rotate Right through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	5 1 1 5 4 6
RSP	Reset Stack Pointer	SP ← 0xFF (High Byte Not Affected)	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 0x0001; Pull (CCR) SP ← (SP) + 0x0001; Pull (A) SP ← (SP) + 0x0001; Pull (X) SP ← (SP) + 0x0001; Pull (PCH) SP ← (SP) + 0x0001; Pull (PCL)	↕	↕	↕	↕	↕	↕	INH	80		9
RTS	Return from Subroutine	SP ← SP + 0x0001; Pull (PCH) SP ← SP + 0x0001; Pull (PCL)	–	–	–	–	–	–	INH	81		6
SBC <i>#opr8i</i> SBC <i>opr8a</i> SBC <i>opr16a</i> SBC <i>opr16,X</i> SBC <i>opr8,X</i> SBC <i>,X</i> SBC <i>opr16,SP</i> SBC <i>opr8,SP</i>	Subtract with Carry	A ← (A) – (M) – (C)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP2 SP1	A2 B2 C2 D2 E2 F2 9ED2 9EE2	ii dd hh ll ee ff ff ff ee ff ff	2 3 4 4 3 3 5 4
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask Bit	I ← 1	–	–	1	–	–	–	INH	9B		1
STA <i>opr8a</i> STA <i>opr16a</i> STA <i>opr16,X</i> STA <i>opr8,X</i> STA <i>,X</i> STA <i>opr16,SP</i> STA <i>opr8,SP</i>	Store Accumulator in Memory	M ← (A)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP2 SP1	B7 C7 D7 E7 F7 9ED7 9EE7	dd hh ll ee ff ff ff ee ff ff	3 4 4 3 2 5 4
STHX <i>opr8a</i> STHX <i>opr16a</i> STHX <i>opr8,SP</i>	Store H:X (Index Reg.)	(M:M + 0x0001) ← (H:X)	0	–	–	↕	↕	–	DIR EXT SP1	35 96 9EFF	dd hh ll ff	4 5 5
STOP	Enable Interrupts: Stop Processing Refer to MCU Documentation	I bit ← 0; Stop Processing	–	–	0	–	–	–	INH	8E		2+
STX <i>opr8a</i> STX <i>opr16a</i> STX <i>opr16,X</i> STX <i>opr8,X</i> STX <i>,X</i> STX <i>opr16,SP</i> STX <i>opr8,SP</i>	Store X (Low 8 Bits of Index Register) in Memory	M ← (X)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP2 SP1	BF CF DF EF FF 9EDF 9EEF	dd hh ll ee ff ff ff ee ff ff	3 4 4 3 2 5 4
SUB <i>#opr8i</i> SUB <i>opr8a</i> SUB <i>opr16a</i> SUB <i>opr16,X</i> SUB <i>opr8,X</i> SUB <i>,X</i> SUB <i>opr16,SP</i> SUB <i>opr8,SP</i>	Subtract	A ← (A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP2 SP1	A0 B0 C0 D0 E0 F0 9ED0 9EE0	ii dd hh ll ee ff ff ff ee ff ff	2 3 4 4 3 3 5 4
SWI	Software Interrupt	PC ← (PC) + 0x0001 Push (PCL); SP ← (SP) – 0x0001 Push (PCH); SP ← (SP) – 0x0001 Push (X); SP ← (SP) – 0x0001 Push (A); SP ← (SP) – 0x0001 Push (CCR); SP ← (SP) – 0x0001 I ← 1; PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		11

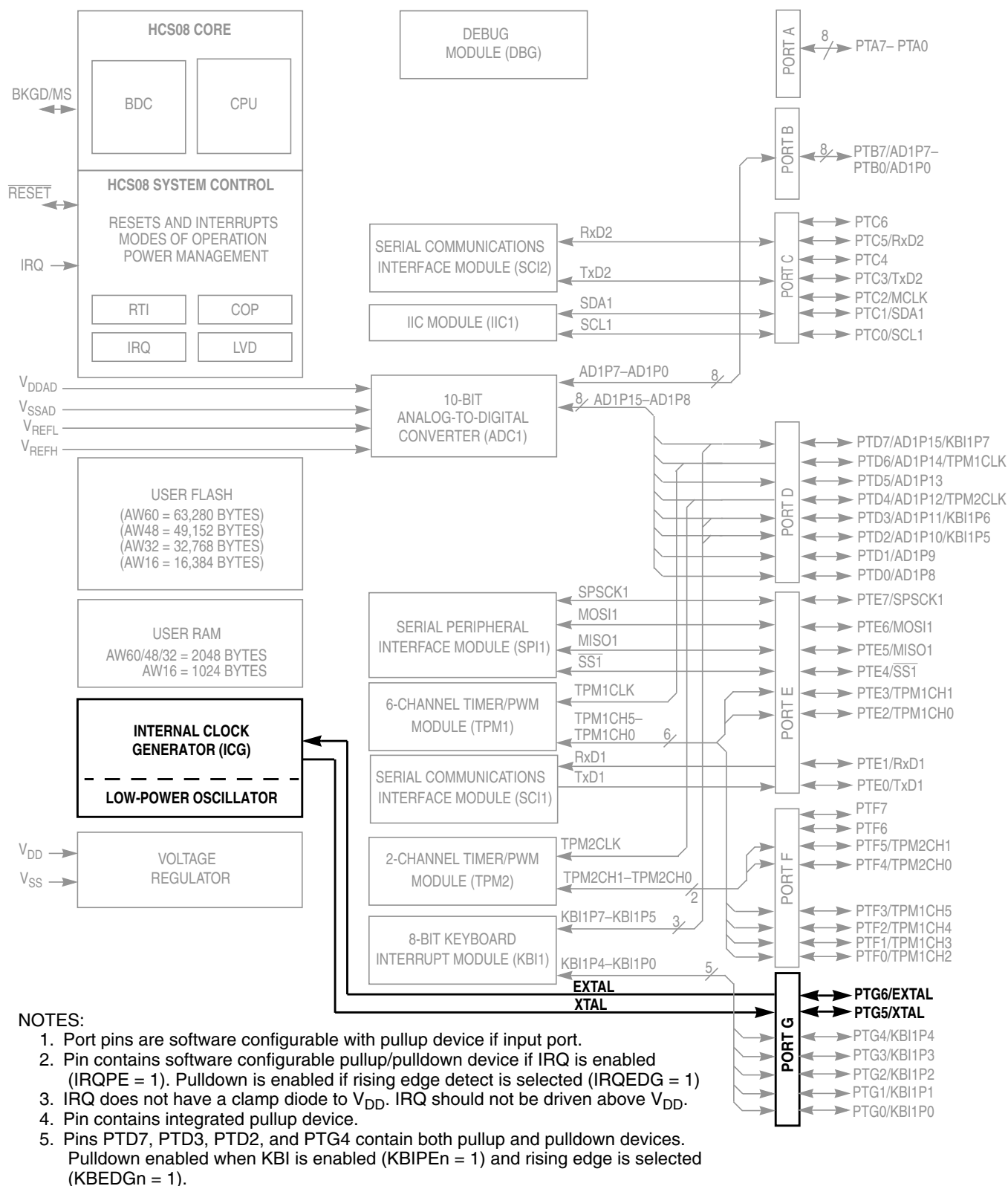


Figure 8-2. Block Diagram Highlighting ICG Module

Table 8-1. ICGC1 Register Field Descriptions (continued)

Field	Description
2 OSCSTEN	Enable Oscillator in Off Mode — The OSCSTEN bit controls whether or not the oscillator circuit remains enabled when the ICG enters off mode. This bit has no effect if HGO = 1 and RANGE = 1. 0 Oscillator disabled when ICG is in off mode unless ENABLE is high, CLKS = 10, and REFST = 1. 1 Oscillator enabled when ICG is in off mode, CLKS = 1X and REFST = 1.
1 LOCD	Loss of Clock Disable 0 Loss of clock detection enabled. 1 Loss of clock detection disabled.

Bits 11:0 FLT No need for user initialization

ICGTRM = \$xx

Bits 7:0 TRIM Only need to write when trimming internal oscillator; not used when external crystal is clock source

Figure 8-14 shows flow charts for three conditions requiring ICG initialization.

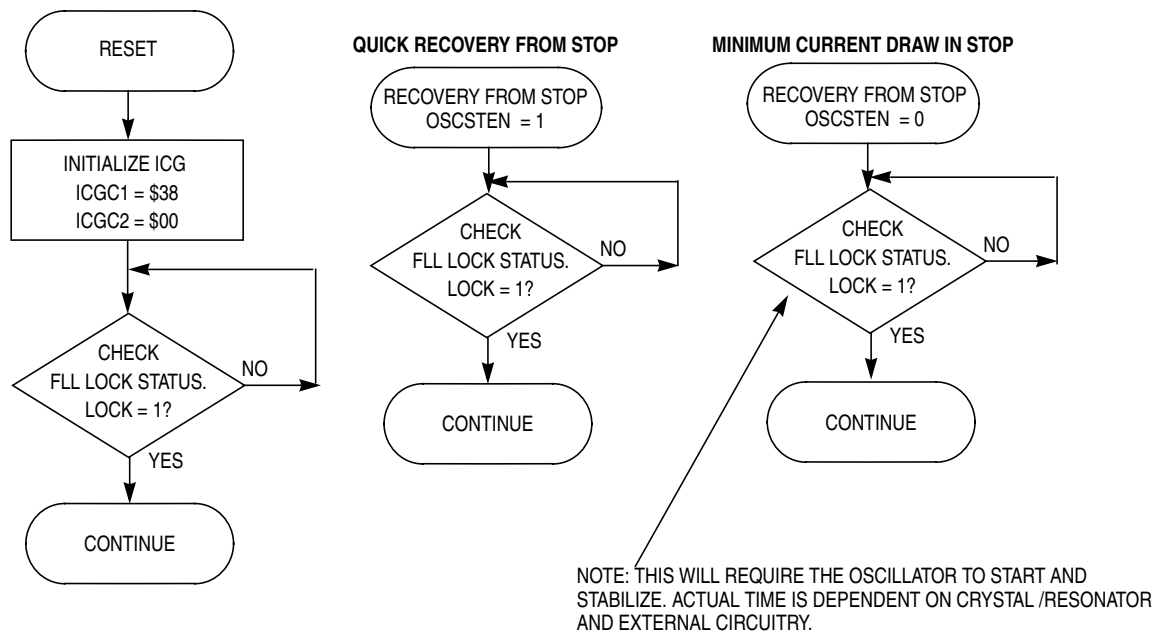


Figure 8-14. ICG Initialization for FEE in Example #1

to Section 11.3.5.1, “8- and 9-Bit Data Modes.” For the remainder of this discussion, we assume the SCI is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (RDRF) status flag is set. If RDRF was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost. Because the SCI receiver is double-buffered, the program has one full character time after RDRF is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full ($RDRF = 1$), it gets the data from the receive data register by reading SCiXD. The RDRF flag is cleared automatically by a 2-step sequence which is normally satisfied in the course of the user's program that handles receive data. Refer to Section 11.3.4, “Interrupts and Status Flags” for more details about flag clearing.

11.3.3.1 Data Sampling Technique

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

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

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

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

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

11.3.3.2 Receiver Wakeup Operation

Receiver wakeup is a hardware mechanism that allows an SCI receiver to ignore the characters in a message that is intended for a different SCI receiver. In such a system, all receivers evaluate the first

Chapter 13

Inter-Integrated Circuit (S08IICV1)

13.1 Introduction

The MC9S08AW60 Series of microcontrollers has an inter-integrated circuit (IIC) module for communication with other integrated circuits. The two pins associated with this module, SCL and SDA, are open-drain outputs and are shared with port C pins 0 and 1, respectively.

Chapter 14

Analog-to-Digital Converter (S08ADC10V1)

14.1 Overview

The 10-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip. The ADC module design supports up to 28 separate analog inputs (AD0-AD27). Only 18 (AD0-AD15, AD26, and AD27) of the possible inputs are implemented on the MC9S08AW60 Series of MCUs. These inputs are selected by the ADCH bits. Some inputs are shared with I/O pins as shown in Figure 14-1. All of the channel assignments of the ADC for the MC9S08AW60 Series devices are summarized in Table 14-1.

14.2 Channel Assignments

The ADC channel assignments for the MC9S08AW60 Series devices are shown in the table below. Channels that are unimplemented are internally connected to V_{REFL} . Reserved channels convert to an unknown value. Channels which are connected to an I/O pin have an associated pin control bit as shown.

Table 14-1. ADC Channel Assignment

ADCH	Channel	Input	Pin Control	ADCH	Channel	Input	Pin Control
00000	AD0	PTB0/ADC1P0	ADPC0	10000	AD16	V_{REFL}	N/A
00001	AD1	PTB1/ADC1P1	ADPC1	10001	AD17	V_{REFL}	N/A
00010	AD2	PTB2/ADC1P2	ADPC2	10010	AD18	V_{REFL}	N/A
00011	AD3	PTB3/ADC1P3	ADPC3	10011	AD19	V_{REFL}	N/A
00100	AD4	PTB4/ADC1P4	ADPC4	10100	AD20	V_{REFL}	N/A
00101	AD5	PTB5/ADC1P5	ADPC5	10101	AD21	V_{REFL}	N/A
00110	AD6	PTB6/ADC1P6	ADPC6	10110	AD22	Reserved	N/A
00111	AD7	PTB7/ADC1P7	ADPC7	10111	AD23	Reserved	N/A
01000	AD8	PTD0/ADC1P8	ADPC8	11000	AD24	Reserved	N/A
01001	AD9	PTD1/ADC1P9	ADPC9	11001	AD25	Reserved	N/A
01010	AD10	PTD2/ADC1P10/ KBI1P5	ADPC10	11010	AD26	Temperature Sensor ¹	N/A
01011	AD11	PTD3/ADC1P11/ KBI1P6	ADPC11	11011	AD27	Internal Bandgap	N/A
01100	AD12	PTD4/ADC1P12/ TPM2CLK	ADPC12	11100	—	Reserved	N/A
01101	AD13	PTD5/ADC1P13	ADPC13	11101	V_{REFH}	V_{REFH}	N/A

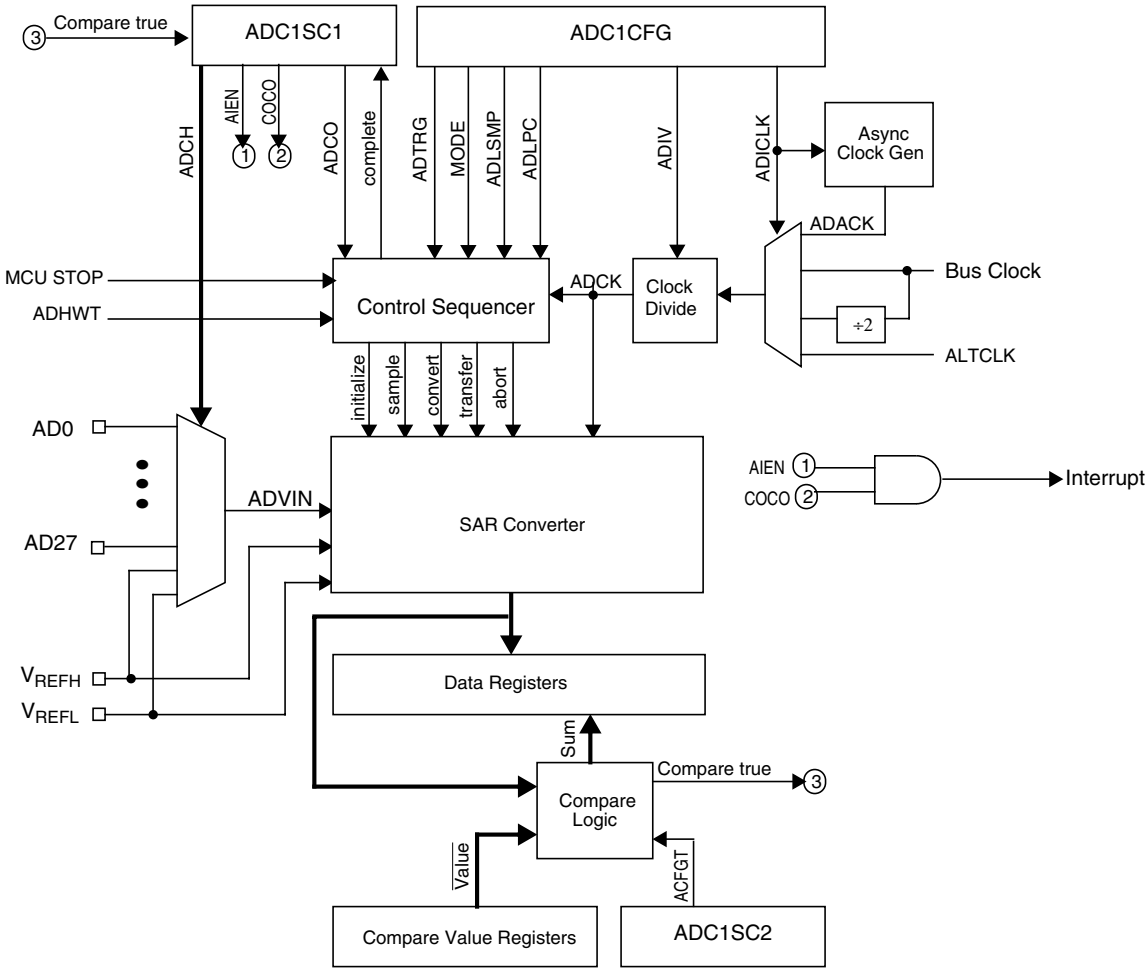


Figure 14-2. ADC Block Diagram

14.3 External Signal Description

The ADC module supports up to 28 separate analog inputs. It also requires four supply/reference/ground connections.

Table 14-2. Signal Properties

Name	Function
AD27–AD0	Analog Channel inputs
V _{REFH}	High reference voltage
V _{REFL}	Low reference voltage
V _{DDAD}	Analog power supply
V _{SSAD}	Analog ground

- Non-intrusive commands can be executed at any time even while the user's program is running. Non-intrusive commands allow a user to read or write MCU memory locations or access status and control registers within the background debug controller.

Typically, a relatively simple interface pod is used to translate commands from a host computer into commands for the custom serial interface to the single-wire background debug system. Depending on the development tool vendor, this interface pod may use a standard RS-232 serial port, a parallel printer port, or some other type of communications such as a universal serial bus (USB) to communicate between the host PC and the pod. The pod typically connects to the target system with ground, the BKGD pin, $\overline{\text{RESET}}$, and sometimes V_{DD} . An open-drain connection to reset allows the host to force a target system reset, which is useful to regain control of a lost target system or to control startup of a target system before the on-chip nonvolatile memory has been programmed. Sometimes V_{DD} can be used to allow the pod to use power from the target system to avoid the need for a separate power supply. However, if the pod is powered separately, it can be connected to a running target system without forcing a target system reset or otherwise disturbing the running application program.

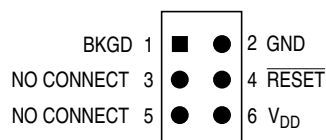


Figure 15-1. BDM Tool Connector

15.2.1 BKGD Pin Description

BKGD is the single-wire background debug interface pin. The primary function of this pin is for bidirectional serial communication of active background mode commands and data. During reset, this pin is used to select between starting in active background mode or starting the user's application program. This pin is also used to request a timed sync response pulse to allow a host development tool to determine the correct clock frequency for background debug serial communications.

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

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

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

the host must perform $((8 - \text{CNT}) - 1)$ dummy reads of the FIFO to advance it to the first significant entry in the FIFO.

In most trigger modes, the information stored in the FIFO consists of 16-bit change-of-flow addresses. In these cases, read DBGFH then DBGFL to get one coherent word of information out of the FIFO. Reading DBGFL (the low-order byte of the FIFO data port) causes the FIFO to shift so the next word of information is available at the FIFO data port. In the event-only trigger modes (see Section 15.3.5, “Trigger Modes”), 8-bit data information is stored into the FIFO. In these cases, the high-order half of the FIFO (DBGFH) is not used and data is read out of the FIFO by simply reading DBGFL. Each time DBGFL is read, the FIFO is shifted so the next data value is available through the FIFO data port at DBGFL.

In trigger modes where the FIFO is storing change-of-flow addresses, there is a delay between CPU addresses and the input side of the FIFO. Because of this delay, if the trigger event itself is a change-of-flow address or a change-of-flow address appears during the next two bus cycles after a trigger event starts the FIFO, it will not be saved into the FIFO. In the case of an end-trace, if the trigger event is a change-of-flow, it will be saved as the last change-of-flow entry for that debug run.

The FIFO can also be used to generate a profile of executed instruction addresses when the debugger is not armed. When $\text{ARM} = 0$, reading DBGFL causes the address of the most-recently fetched opcode to be saved in the FIFO. To use the profiling feature, a host debugger would read addresses out of the FIFO by reading DBGFH then DBGFL at regular periodic intervals. The first eight values would be discarded because they correspond to the eight DBGFL reads needed to initially fill the FIFO. Additional periodic reads of DBGFH and DBGFL return delayed information about executed instructions so the host debugger can develop a profile of executed instruction addresses.

15.3.3 Change-of-Flow Information

To minimize the amount of information stored in the FIFO, only information related to instructions that cause a change to the normal sequential execution of instructions is stored. With knowledge of the source and object code program stored in the target system, an external debugger system can reconstruct the path of execution through many instructions from the change-of-flow information stored in the FIFO.

For conditional branch instructions where the branch is taken (branch condition was true), the source address is stored (the address of the conditional branch opcode). Because BRA and BRN instructions are not conditional, these events do not cause change-of-flow information to be stored in the FIFO.

Indirect JMP and JSR instructions use the current contents of the H:X index register pair to determine the destination address, so the debug system stores the run-time destination address for any indirect JMP or JSR. For interrupts, RTI, or RTS, the destination address is stored in the FIFO as change-of-flow information.

15.3.4 Tag vs. Force Breakpoints and Triggers

Tagging is a term that refers to identifying an instruction opcode as it is fetched into the instruction queue, but not taking any other action until and unless that instruction is actually executed by the CPU. This distinction is important because any change-of-flow from a jump, branch, subroutine call, or interrupt causes some instructions that have been fetched into the instruction queue to be thrown away without being executed.

Table A-7. DC Characteristics (continued)

Num	C	Parameter	Symbol	Min	Typ ¹	Max	Unit
9	P	High Impedance (off-state) leakage current ²	$ I_{OZ} $	—	0.01	1	μA
10	P	Internal pullup resistors ³	R_{PU}	20	45	65	$k\Omega$
11	P	Internal pulldown resistors ⁴	R_{PD}	20	45	65	$k\Omega$
12	C	Input Capacitance; all non-supply pins	C_{In}	—	—	8	pF
13	P	POR rearm voltage	V_{POR}	0.9	1.4	2.0	V
14	D	POR rearm time	t_{POR}	10	—	—	μs
15	P	Low-voltage detection threshold — high range V_{DD} falling V_{DD} rising	V_{LVDH}	4.2 4.3	4.3 4.4	4.4 4.5	V
16	P	Low-voltage detection threshold — low range V_{DD} falling V_{DD} rising	V_{LVDL}	2.48 2.54	2.56 2.62	2.64 2.7	V
17	P	Low-voltage warning threshold — high range V_{DD} falling V_{DD} rising	V_{LVWH}	4.2 4.3	4.3 4.4	4.4 4.5	V
18	P	Low-voltage warning threshold — low range V_{DD} falling V_{DD} rising	V_{LVWL}	2.48 2.54	2.56 2.62	2.64 2.7	V
19	P	Low-voltage inhibit reset/recover hysteresis 5V 3V	V_{hys}	— —	100 60	— —	mV
20	P	Bandgap Voltage Reference Factory trimmed at $V_{DD} = 5.0 V$ Temp = 25 °C	V_{BG}	1.185	1.20	1.215	V
21	D	dc injection current ^{5, 6, 7, 8} DC Injection Current Single pin limit $V_{IN} > V_{DD}$ $V_{IN} < V_{SS}$ Total MCU limit, includes sum of all stressed pins $V_{IN} > V_{DD}$ $V_{IN} < V_{SS}$	$ I_{IC} $	0 0 0 0	- - - -	2 -0.2 25 -5	mA mA mA mA

¹ Typical values are based on characterization data at 25°C unless otherwise stated.

² Measured with $V_{In} = V_{DD}$ or V_{SS} .

³ Measured with $V_{In} = V_{SS}$.

⁴ Measured with $V_{In} = V_{DD}$.

⁵ Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{In} > V_{DD}$) is greater than I_{DD} , the injection current may flow out of V_{DD} and could result in external power supply going out of regulation. Ensure external V_{DD} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power. Examples are: if no system clock is present, or if clock rate is very low which (would reduce overall power consumption).

⁶ All functional non-supply pins are internally clamped to V_{SS} and V_{DD} .

⁷ Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive and negative clamp voltages, then use the larger of the two values.

A.9 Internal Clock Generation Module Characteristics

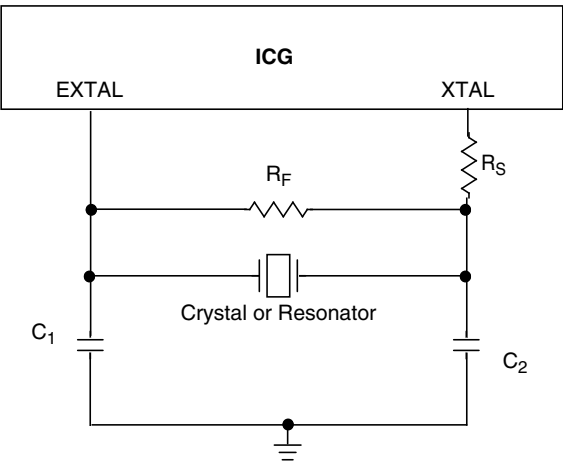


Table A-11. ICG DC Electrical Specifications (Temperature Range = –40 to 125°C Ambient)

Characteristic	Symbol	Min	Typ ¹	Max	Unit
Load capacitors	C ₁ C ₂	See Note ²			
Feedback resistor	R _F		10 1		MΩ MΩ
Low range (32k to 100 kHz)					
High range (1M – 16 MHz)					
Series resistor	R _S		0 100 0 0 10 20		kΩ
Low range					
Low Gain (HGO = 0)					
High Gain (HGO = 1)					
High range					
Low Gain (HGO = 0)					
High Gain (HGO = 1)					
≥ 8 MHz					
4 MHz					
1 MHz					

¹ Typical values are based on characterization data at V_{DD} = 5.0V, 25°C or is typical recommended value.

² See crystal or resonator manufacturer's recommendation.