

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	16MHz
Connectivity	-
Peripherals	LVD, POR, PWM, WDT
Number of I/O	4
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 4x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	8-SOIC (0.154", 3.90mm Width)
Supplier Device Package	8-SOIC
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=s9s08qd4j1vsc

9.5	Module Initialization	132
9.5.1	ICS Module Initialization Sequence	132

Chapter 10

Keyboard Interrupt (S08KBIV2)

10.1	Introduction	135
10.1.1	Features	137
10.1.2	Modes of Operation	137
10.1.3	Block Diagram	137
10.2	External Signal Description	138
10.3	Register Definition	138
10.3.1	KBI Status and Control Register (KBISC)	138
10.3.2	KBI Pin Enable Register (KBIPE)	139
10.3.3	KBI Edge Select Register (KBIES)	139
10.4	Functional Description	140
10.4.1	Edge Only Sensitivity	140
10.4.2	Edge and Level Sensitivity	140
10.4.3	KBI Pullup/Pulldown Resistors	141
10.4.4	KBI Initialization	141

Chapter 11

Timer/Pulse-Width Modulator (S08TPMV2)

11.1	Introduction	143
11.1.1	TPM2 Configuration Information	143
11.1.2	TCLK1 and TCLK2 Configuration Information	143
11.1.3	Features	145
11.1.4	Block Diagram	145
11.2	External Signal Description	147
11.2.1	External TPM Clock Sources	147
11.2.2	TPMxCn — TPMx Channel n I/O Pins	147
11.3	Register Definition	147
11.3.1	Timer Status and Control Register (TPMxSC)	148
11.3.2	Timer Counter Registers (TPMxCNTH:TPMxCNTL)	149
11.3.3	Timer Counter Modulo Registers (TPMxMODH:TPMxMODL)	150
11.3.4	Timer Channel n Status and Control Register (TPMxCnSC)	151
11.3.5	Timer Channel Value Registers (TPMxCnVH:TPMxCnVL)	152
11.4	Functional Description	153
11.4.1	Counter	153
11.4.2	Channel Mode Selection	154
11.4.3	Center-Aligned PWM Mode	156
11.5	TPM Interrupts	157
11.5.1	Clearing Timer Interrupt Flags	157
11.5.2	Timer Overflow Interrupt Description	157
11.5.3	Channel Event Interrupt Description	158
11.5.4	PWM End-of-Duty-Cycle Events	158

Chapter 1

Device Overview

1.1 Introduction

MC9S08QD4 series MCUs are 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.

1.2 Devices in the MC9S08QD4 Series

This data sheet covers:

- MC9S08QD4
- MC9S08QD2
- S9S08QD4
- S9S08QD2

NOTE

- The MC9S08QD4 and MC9S08QD2 devices are qualified for, and are intended to be used in, *consumer and industrial* applications.
- The S9S08QD4 and S9S08QD2 devices are qualified for, and are intended to be used in, *automotive* applications.

[Table 1-1](#) summarizes the features available in the MCUs.

5.8 Reset, Interrupt, and System Control Registers and Control Bits

One 8-bit register in the direct page register space and eight 8-bit registers in the high-page register space are related to reset and interrupt systems.

Refer to the direct-page register summary in [Chapter 3, “Modes of Operation,”](#) for the absolute address assignments for all registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

Some control bits in the SOPT1, SOPT2 and SPMSC2 registers are related to modes of operation. Although brief descriptions of these bits are provided here, the related functions are discussed in greater detail in [Chapter 3, “Modes of Operation.”](#)

5.8.1 Interrupt Pin Request Status and Control Register (IRQSC)

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

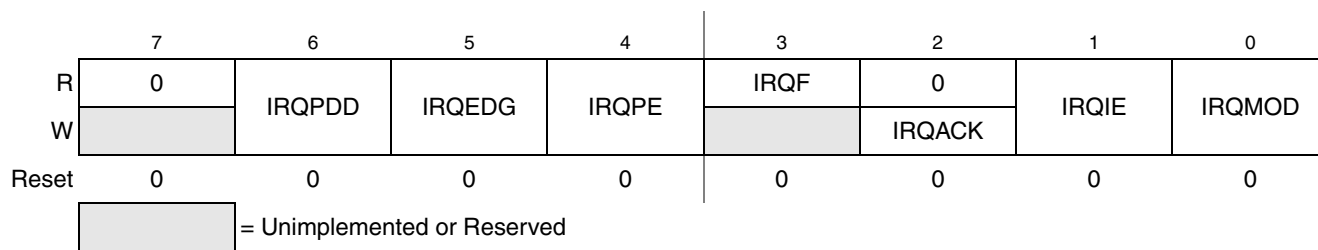


Figure 5-2. Interrupt Request Status and Control Register (IRQSC)

Table 5-3. IRQSC Register Field Descriptions

Field	Description
6 IRQPDD	Interrupt Request (IRQ) Pull Device Disable — This read/write control bit is used to disable the internal pullup/pulldown device when the IRQ pin is enabled (IRQPE = 1) allowing for an external device to be used. 0 IRQ pull device enabled if IRQPE = 1. 1 IRQ pull device disabled if IRQPE = 1.
5 IRQEDG	Interrupt Request (IRQ) Edge Select — This read/write control bit is used to select the polarity of edges or levels on the IRQ pin that cause IRQF to be set. The IRQMOD control bit determines whether the IRQ pin is sensitive to both edges and levels or only edges. When the IRQ pin is enabled as the IRQ input and is configured to detect rising edges. When IRQEDG = 1 and the internal pull device is enabled, the pullup device is reconfigured as an optional pulldown device. 0 IRQ is falling edge or falling edge/low-level sensitive. 1 IRQ is rising edge or rising edge/high-level sensitive.
4 IRQPE	IRQ Pin Enable — This read/write control bit enables the IRQ pin function. When this bit is set the IRQ pin can be used as an interrupt request. 0 IRQ pin function is disabled. 1 IRQ pin function is enabled.
3 IRQF	IRQ Flag — This read-only status bit indicates when an interrupt request event has occurred. 0 No IRQ request. 1 IRQ event detected.

Chapter 7

Central Processor Unit (S08CPUV2)

7.1 Introduction

This section provides summary information about the registers, addressing modes, and instruction set of the CPU of the HCS08 Family. For a more detailed discussion, refer to the *HCS08 Family Reference Manual, volume 1*, Freescale Semiconductor document order number HCS08RMV1/D.

The HCS08 CPU is fully source- and object-code-compatible with the M68HC08 CPU. Several instructions and enhanced addressing modes were added to improve C compiler efficiency and to support a new background debug system which replaces the monitor mode of earlier M68HC08 microcontrollers (MCU).

7.1.1 Features

Features of the HCS08 CPU include:

- Object code fully upward-compatible with M68HC05 and M68HC08 Families
- All registers and memory are mapped to a single 64-Kbyte address space
- 16-bit stack pointer (any size stack anywhere in 64-Kbyte address space)
- 16-bit index register (H:X) with powerful indexed addressing modes
- 8-bit accumulator (A)
- Many instructions treat X as a second general-purpose 8-bit register
- Seven addressing modes:
 - Inherent — Operands in internal registers
 - Relative — 8-bit signed offset to branch destination
 - Immediate — Operand in next object code byte(s)
 - Direct — Operand in memory at 0x0000–0x00FF
 - Extended — Operand anywhere in 64-Kbyte address space
 - Indexed relative to H:X — Five submodes including auto increment
 - Indexed relative to SP — Improves C efficiency dramatically
- Memory-to-memory data move instructions with four address mode combinations
- Overflow, half-carry, negative, zero, and carry condition codes support conditional branching on the results of signed, unsigned, and binary-coded decimal (BCD) operations
- Efficient bit manipulation instructions
- Fast 8-bit by 8-bit multiply and 16-bit by 8-bit divide instructions
- STOP and WAIT instructions to invoke low-power operating modes

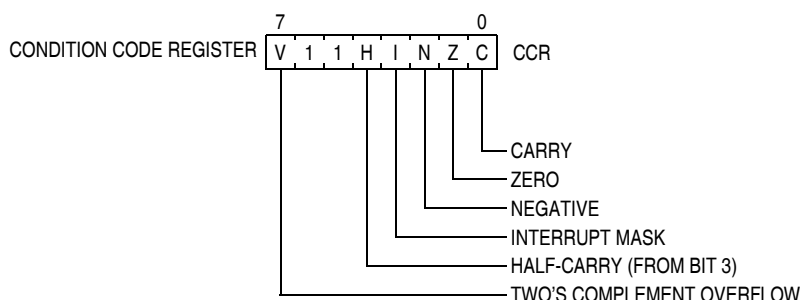


Figure 7-2. Condition Code Register

Table 7-1. CCR Register Field Descriptions

Field	Description
7 V	Two's Complement Overflow Flag — The CPU sets the overflow flag when a two's complement overflow occurs. The signed branch instructions BGT, BGE, BLE, and BLT use the overflow flag. 0 No overflow 1 Overflow
4 H	Half-Carry Flag — The CPU sets the half-carry flag when a carry occurs between accumulator bits 3 and 4 during an add-without-carry (ADD) or add-with-carry (ADC) operation. The half-carry flag is required for binary-coded decimal (BCD) arithmetic operations. The DAA instruction uses the states of the H and C condition code bits to automatically add a correction value to the result from a previous ADD or ADC on BCD operands to correct the result to a valid BCD value. 0 No carry between bits 3 and 4 1 Carry between bits 3 and 4
3 I	Interrupt Mask Bit — When the interrupt mask is set, all maskable CPU interrupts are disabled. CPU interrupts are enabled when the interrupt mask is cleared. When a CPU interrupt occurs, the interrupt mask is set automatically after the CPU registers are saved on the stack, but before the first instruction of the interrupt service routine is executed. Interrupts are not recognized at the instruction boundary after any instruction that clears I (CLI or TAP). This ensures that the next instruction after a CLI or TAP will always be executed without the possibility of an intervening interrupt, provided I was set. 0 Interrupts enabled 1 Interrupts disabled
2 N	Negative Flag — The CPU sets the negative flag when an arithmetic operation, logic operation, or data manipulation produces a negative result, setting bit 7 of the result. Simply loading or storing an 8-bit or 16-bit value causes N to be set if the most significant bit of the loaded or stored value was 1. 0 Non-negative result 1 Negative result
1 Z	Zero Flag — The CPU sets the zero flag when an arithmetic operation, logic operation, or data manipulation produces a result of 0x00 or 0x0000. Simply loading or storing an 8-bit or 16-bit value causes Z to be set if the loaded or stored value was all 0s. 0 Non-zero result 1 Zero result
0 C	Carry/Borrow Flag — The CPU sets the carry/borrow flag when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some instructions — such as bit test and branch, shift, and rotate — also clear or set the carry/borrow flag. 0 No carry out of bit 7 1 Carry out of bit 7

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 up 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.

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

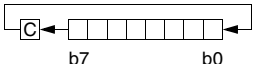
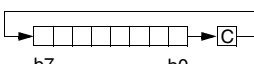
Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR			
						VH	I	N	Z C
MOV <i>opr8a,opr8a</i> MOV <i>opr8a,X+</i> MOV <i>#opr8i,opr8a</i> MOV <i>,X+,opr8a</i>	Move $(M)_{\text{destination}} \leftarrow (M)_{\text{source}}$ In IX+/DIR and DIR/IX+ Modes, $H:X \leftarrow (H:X) + \$0001$	DIR/DIR DIR/IX+ IMM/DIR IX+/DIR	4E dd dd 5E dd 6E ii dd 7E dd	5 5 4 5	rpwpp rfwpp pwpp rfwpp	0	-	-	↑ ↓ -
MUL	Unsigned multiply $X:A \leftarrow (X) \times (A)$	INH	42	5	fffffp	-	0	-	- - - 0
NEG <i>opr8a</i> NEGA NEGX NEG <i>opr8,X</i> NEG <i>,X</i> NEG <i>opr8,SP</i>	Negate $M \leftarrow -(M) = \$00 - (M)$ (Two's Complement) $A \leftarrow -(A) = \$00 - (A)$ $X \leftarrow -(X) = \$00 - (X)$ $M \leftarrow -(M) = \$00 - (M)$ $M \leftarrow -(M) = \$00 - (M)$ $M \leftarrow -(M) = \$00 - (M)$	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	↑	-	-	↑ ↓ ↑
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>opr16,X</i> ORA <i>opr8,X</i> ORA <i>,X</i> ORA <i>opr16,SP</i> ORA <i>opr8,SP</i>	Inclusive OR Accumulator and Memory $A \leftarrow (A) \vee (M)$	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	-	-	↑ ↓ -
PSHA	Push Accumulator onto Stack $\text{Push } (A); SP \leftarrow (SP) - \0001	INH	87	2	sp	-	-	-	- - - -
PSHH	Push H (Index Register High) onto Stack $\text{Push } (H); SP \leftarrow (SP) - \0001	INH	8B	2	sp	-	-	-	- - - -
PSHX	Push X (Index Register Low) onto Stack $\text{Push } (X); SP \leftarrow (SP) - \0001	INH	89	2	sp	-	-	-	- - - -
PULA	Pull Accumulator from Stack $SP \leftarrow (SP + \$0001); \text{Pull } (A)$	INH	86	3	ufp	-	-	-	- - - -
PULH	Pull H (Index Register High) from Stack $SP \leftarrow (SP + \$0001); \text{Pull } (H)$	INH	8A	3	ufp	-	-	-	- - - -
PULX	Pull X (Index Register Low) from Stack $SP \leftarrow (SP + \$0001); \text{Pull } (X)$	INH	88	3	ufp	-	-	-	- - - -
ROL <i>opr8a</i> ROLA ROLX ROL <i>opr8,X</i> ROL <i>,X</i> ROL <i>opr8,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	↑	-	-	↑ ↓ ↑
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 dd 46 56 66 ff 76 9E 66 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑	-	-	↑ ↓ ↑

Table 7-3. Opcode Map (Sheet 2 of 2)

Bit-Manipulation	Branch	Read-Modify-Write				Control				Register/Memory							
					9E60 6 3 SP1 NEG						9ED0 5 4 SP2 SUB	9EE0 4 3 SP1 SUB					
					9E61 6 4 SP1 CBEQ						9ED1 5 4 SP2 CMP	9EE1 4 3 SP1 CMP					
											9ED2 5 4 SP2 SBC	9EE2 4 3 SP1 SBC					
					9E63 6 3 SP1 COM						9ED3 5 4 SP2 CPX	9EE3 4 3 SP1 CPX	9EF3 6 3 SP1 CPHX				
					9E64 6 3 SP1 LSR						9ED4 5 4 SP2 AND	9EE4 4 3 SP1 AND					
											9ED5 5 4 SP2 BIT	9EE5 4 3 SP1 BIT					
					9E66 6 3 SP1 ROR						9ED6 5 4 SP2 LDA	9EE6 4 3 SP1 LDA					
					9E67 6 3 SP1 ASR						9ED7 5 4 SP2 STA	9EE7 4 3 SP1 STA					
					9E68 6 3 SP1 LSL						9ED8 5 4 SP2 EOR	9EE8 4 3 SP1 EOR					
					9E69 6 3 SP1 ROL						9ED9 5 4 SP2 ADC	9EE9 4 3 SP1 ADC					
					9E6A 6 3 SP1 DEC						9EDA 5 4 SP2 ORA	9EEA 4 3 SP1 ORA					
					9E6B 8 4 SP1 DBNZ						9EDB 5 4 SP2 ADD	9EEB 4 3 SP1 ADD					
					9E6C 6 3 SP1 INC												
					9E6D 5 3 SP1 TST												
										9EAE 5 2 IX LDHX	9EBE 6 4 IX2 LDHX	9ECE 5 3 IX1 LDHX	9EDE 5 4 SP2 LDX	9EEE 4 3 SP1 LDX	9EFE 5 3 SP1 LDHX		
					9E6F 6 3 SP1 CLR						9EDF 5 4 SP2 STX	9EEF 4 3 SP1 STX	9EFF 5 3 SP1 STHX				

INH Inherent REL Relative SP1 Stack Pointer, 8-Bit Offset
 IMM Immediate IX Indexed, No Offset SP2 Stack Pointer, 16-Bit Offset
 DIR Direct IX1 Indexed, 8-Bit Offset IX+ Indexed, No Offset with
 EXT Extended IX2 Indexed, 16-Bit Offset Post Increment
 DD DIR to DIR IMD IMM to DIR IX1+ Indexed, 1-Byte Offset with
 IX+D IX+ to DIR DIX+ DIR to IX+ Post Increment

Note: All Sheet 2 Opcodes are Preceded by the Page 2 Prebyte (9E)

Prebyte (9E) and Opcode in
 Hexadecimal 9E60 6
 3 SP1
 Number of Bytes 3 NEG HCS08 Cycles
 SP1 Instruction Mnemonic
 Addressing Mode

8.1.1.1 Channel Assignments

The ADC channel assignments for the MC9S08QD4 series devices are shown in [Table 8-1](#). Reserved channels convert to an unknown value.

Table 8-1. ADC Channel Assignment

ADCH	Channel	Input	Pin Control	ADCH	Channel	Input	Pin Control
00000	AD0	PTA0/ADC1P0	ADPC0	10000	AD16	V _{SS}	N/A
00001	AD1	PTA1/ADC1P1	ADPC1	10001	AD17	V _{SS}	N/A
00010	AD2	PTA2/ADC1P2	ADPC2	10010	AD18	V _{SS}	N/A
00011	AD3	PTA3/ADC1P3	ADPC3	10011	AD19	V _{SS}	N/A
00100	AD4	V _{SS}	N/A	10100	AD20	V _{SS}	N/A
00101	AD5	V _{SS}	N/A	10101	AD21	V _{SS}	N/A
00110	AD6	V _{SS}	N/A	10110	AD22	Reserved	N/A
00111	AD7	V _{SS}	N/A	10111	AD23	Reserved	N/A
01000	AD8	V _{SS}	N/A	11000	AD24	Reserved	N/A
01001	AD9	V _{SS}	N/A	11001	AD25	Reserved	N/A
01010	AD10	V _{SS}	N/A	11010	AD26	Temperature Sensor ¹	N/A
01011	AD11	V _{SS}	N/A	11011	AD27	Internal Bandgap ²	N/A
01100	AD12	V _{SS}	N/A	11100	V _{REFH}	V _{DD}	N/A
01101	AD13	V _{SS}	N/A	11101	V _{REFH}	V _{DD}	N/A
01110	AD14	V _{SS}	N/A	11110	V _{REFL}	V _{SS}	N/A
01111	AD15	V _{SS}	N/A	11111	Module Disabled	None	N/A

¹ For information, see [Section 8.1.1.5, “Temperature Sensor.”](#)

² Requires BGBE =1 in SPMSC1 see [Section 5.8.8, “System Power Management Status and Control 1 Register \(SPMSC1\).”](#)
For value of bandgap voltage reference see [Appendix A.5, “DC Characteristics.”](#)

8.1.1.2 Alternate Clock

The ADC is capable of performing conversions using the MCU bus clock, the bus clock divided by two, or the local asynchronous clock (ADACK) within the module. The alternate clock, ALTCLK, input for the MC9S08QD4 series MCU devices is not implemented.

8.1.1.3 Hardware Trigger

The ADC hardware trigger, ADHWT, is output from the real-time interrupt (RTI) counter. The RTI counter can be clocked by either IC SERCLK or a nominal 32 kHz clock source within the RTI block.

The period of the RTI is determined by the input clock frequency and the RTIS bits. The RTI counter is a free running counter that generates an overflow at the RTI rate determined by the RTIS bits. When the ADC hardware trigger is enabled, a conversion is initiated upon a RTI counter overflow.

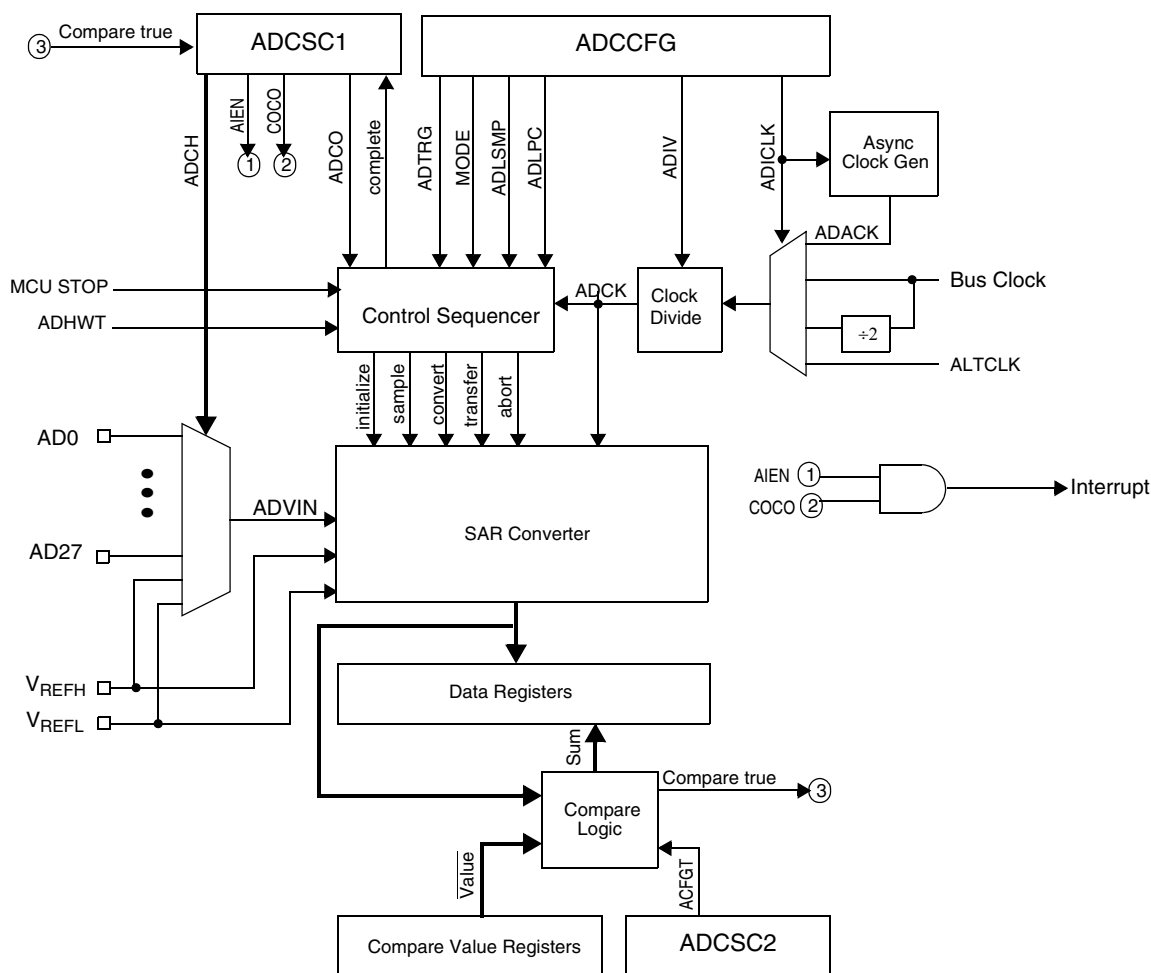


Figure 8-2. ADC Block Diagram

8.2 External Signal Description

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

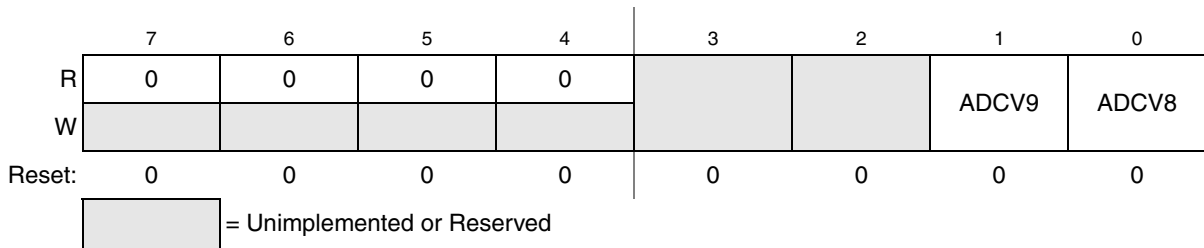
Table 8-2. Signal Properties

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


Figure 8-7. Data Result Low Register (ADCRL)

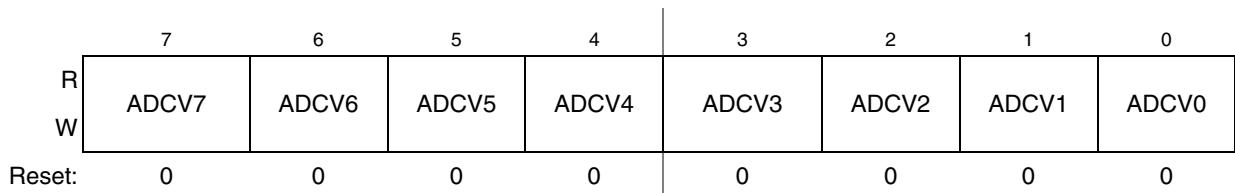
8.3.5 Compare Value High Register (ADCCVH)

This register holds the upper two bits of the 10-bit compare value. These bits are compared to the upper two bits of the result following a conversion in 10-bit mode when the compare function is enabled. In 8-bit operation, ADCCVH is not used during compare.


Figure 8-8. Compare Value High Register (ADCCVH)

8.3.6 Compare Value Low Register (ADCCVL)

This register holds the lower 8 bits of the 10-bit compare value, or all 8 bits of the 8-bit compare value. Bits ADCV7:ADCV0 are compared to the lower 8 bits of the result following a conversion in either 10-bit or 8-bit mode.


Figure 8-9. Compare Value Low Register (ADCCVL)

8.3.7 Configuration Register (ADCCFG)

ADCCFG is used to select the mode of operation, clock source, clock divide, and configure for low power or long sample time.



9.4.1.2 FLL Engaged External (FEE)

The FLL engaged external (FEE) mode is entered when all the following conditions occur:

- CLKS bits are written to 00
- IREFS bit is written to 0
- RDIV bits are written to divide reference clock to be within the range of 31.25 kHz to 39.0625 kHz

In FLL engaged external mode, the ICSOUT clock is derived from the FLL clock which is controlled by the external reference clock. The FLL loop will lock the frequency to 512 times the filter frequency, as selected by the RDIV bits. The ICSLCLK is available for BDC communications, and the external reference clock is enabled.

9.4.1.3 FLL Bypassed Internal (FBI)

The FLL bypassed internal (FBI) mode is entered when all the following conditions occur:

- CLKS bits are written to 01
- IREFS bit is written to 1.
- BDM mode is active or LP bit is written to 0

In FLL bypassed internal mode, the ICSOUT clock is derived from the internal reference clock. The FLL clock is controlled by the internal reference clock, and the FLL loop will lock the FLL frequency to 512 times the Filter frequency, as selected by the RDIV bits. The ICSLCLK will be available for BDC communications, and the internal reference clock is enabled.

9.4.1.4 FLL Bypassed Internal Low Power (FBILP)

The FLL bypassed internal low power (FBILP) mode is entered when all the following conditions occur:

- CLKS bits are written to 01
- IREFS bit is written to 1.
- BDM mode is not active and LP bit is written to 1

In FLL bypassed internal low power mode, the ICSOUT clock is derived from the internal reference clock and the FLL is disabled. The ICSLCLK will be not be available for BDC communications, and the internal reference clock is enabled.

9.4.1.5 FLL Bypassed External (FBE)

The FLL bypassed external (FBE) mode is entered when all the following conditions occur:

- CLKS bits are written to 10.
- IREFS bit is written to 0.
- BDM mode is active or LP bit is written to 0.

In FLL bypassed external mode, the ICSOUT clock is derived from the external reference clock. The FLL clock is controlled by the external reference clock, and the FLL loop will lock the FLL frequency to 512

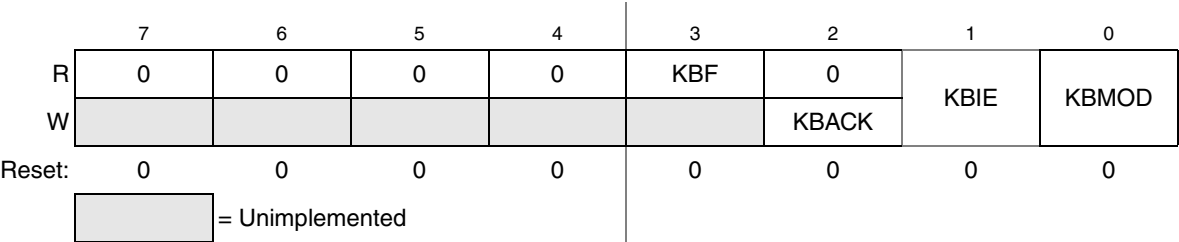


Figure 10-3. KBI Status and Control Register

Table 10-2. KBISC Register Field Descriptions

Field	Description
7:4	Unused register bits, always read 0.
3 KBF	Keyboard Interrupt Flag — KBF indicates when a keyboard interrupt is detected. Writes have no effect on KBF. 0 No keyboard interrupt detected. 1 Keyboard interrupt detected.
2 KBACK	Keyboard Acknowledge — Writing a 1 to KBACK is part of the flag clearing mechanism. KBACK always reads as 0.
1 KBIE	Keyboard Interrupt Enable — KBIE determines whether a keyboard interrupt is requested. 0 Keyboard interrupt request not enabled. 1 Keyboard interrupt request enabled.
0 KBMOD	Keyboard Detection Mode — KBMOD (along with the KBEDG bits) controls the detection mode of the keyboard interrupt pins. 0 Keyboard detects edges only. 1 Keyboard detects both edges and levels.

10.3.2 KBI Pin Enable Register (KBIPE)

KBIPE contains the pin enable control bits.

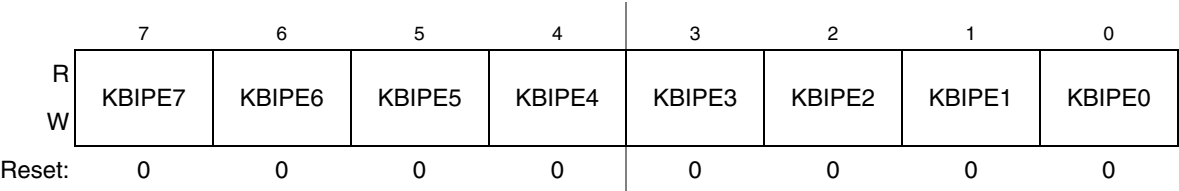


Figure 10-4. KBI Pin Enable Register

Table 10-3. KBIPE Register Field Descriptions

Field	Description
7:0 KBIPEn	Keyboard Pin Enables — Each of the KBIPEn bits enable the corresponding keyboard interrupt pin. 0 Pin not enabled as keyboard interrupt. 1 Pin enabled as keyboard interrupt.

10.3.3 KBI Edge Select Register (KBIES)

KBIES contains the edge select control bits.

KBISC provided all enabled keyboard inputs are at their deasserted levels. KBF will remain set if any enabled KBI pin is asserted while attempting to clear by writing a 1 to KBACK.

10.4.3 KBI Pullup/Pulldown Resistors

The KBI pins can be configured to use an internal pullup/pulldown resistor using the associated I/O port pullup enable register. If an internal resistor is enabled, the KBIES register is used to select whether the resistor is a pullup ($KBEDGn = 0$) or a pulldown ($KBEDGn = 1$).

10.4.4 KBI Initialization

When a keyboard interrupt pin is first enabled it is possible to get a false keyboard interrupt flag. To prevent a false interrupt request during keyboard initialization, the user must do the following:

1. Mask keyboard interrupts by clearing KBIE in KBISC.
2. Enable the KBI polarity by setting the appropriate KBEDGn bits in KBIES.
3. If using internal pullup/pulldown device, configure the associated pullup enable bits in PTxPE.
4. Enable the KBI pins by setting the appropriate KBIPEn bits in KBIPE.
5. Write to KBACK in KBISC to clear any false interrupts.
6. Set KBIE in KBISC to enable interrupts.

12.1.3 Features

Features of the BDC module include:

- Single pin for mode selection and background communications
- BDC registers are not located in the memory map
- SYNC command to determine target communications rate
- Non-intrusive commands for memory access
- Active background mode commands for CPU register access
- GO and TRACE1 commands
- BACKGROUND command can wake CPU from stop or wait modes
- One hardware address breakpoint built into BDC
- Oscillator runs in stop mode, if BDC enabled
- COP watchdog disabled while in active background mode

12.2 Background Debug Controller (BDC)

All MCUs in the HCS08 Family contain a single-wire background debug interface that supports in-circuit programming of on-chip nonvolatile memory and sophisticated non-intrusive debug capabilities. Unlike debug interfaces on earlier 8-bit MCUs, this system does not interfere with normal application resources. It does not use any user memory or locations in the memory map and does not share any on-chip peripherals.

BDC commands are divided into two groups:

- Active background mode commands require that the target MCU is in active background mode (the user program is not running). Active background mode commands allow the CPU registers to be read or written, and allow the user to trace one user instruction at a time, or GO to the user program from active background mode.
- 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.

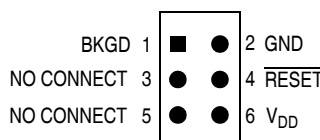


Figure 12-1. BDM Tool Connector

12.2.1 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 12.2.2, "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 12.2.2, "Communication Details,"](#) for more detail.

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

12.2.2 Communication Details

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

BKGD is a pseudo-open-drain pin that can be driven either by an external controller or by the MCU. Data is transferred MSB first at 16 BDC clock cycles per bit (nominal speed). The interface times out if 512 BDC clock cycles occur between falling edges from the host. Any BDC command that was in progress

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.

12.3.1 BDC Registers and Control Bits

The BDC has two registers:

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

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

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

Appendix A Electrical Characteristics

7 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).

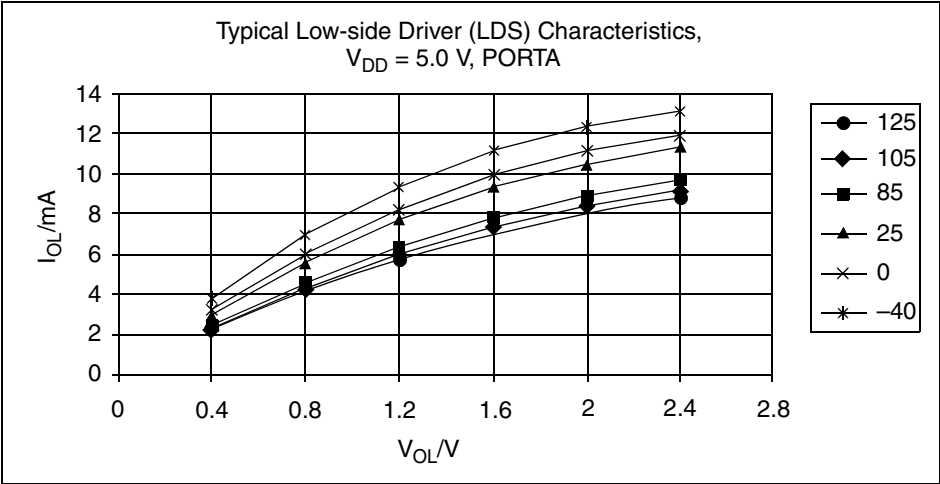


Figure A-1. Typical Low-Side Driver (Sink) Characteristics
Low Drive ($PTxDSn = 0$), $V_{DD} = 5.0\text{V}$, V_{OL} vs. I_{OL}

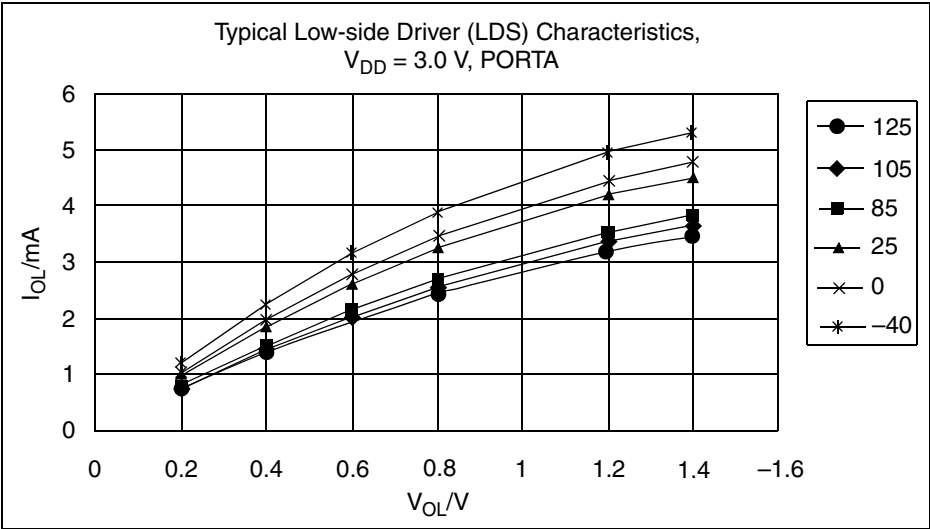


Figure A-2. Typical Low-Side Driver (Sink) Characteristics
Low Drive ($PTxDSn = 0$), $V_{DD} = 3.0 \text{ V}$, V_{OL} vs. I_{OL}