Freescale Semiconductor
Application Note

Document Number: AN2717
Rev. 1, 08/2006

M68HC08 to HCS08 Transition

by: Gordon Borland
East Kilbride, Scotland
Scott Pape
Austin, TX

1 Introduction

This application note introduces users of the M68HC08 family of microcontroller units (MCUs) to the changes on the HCS08 family of MCUs. This does not describe in detail how to use new features of the HCS08 family but simply highlights the differences. Programmers and designers should consult the specific HCS08 MCU data sheet for details.

NOTE

With the exception of mask set errata documents, if any other Freescale document contains information that conflicts with the information in the device data sheet, the data sheet should be considered to have the most current and correct data.

Contents

freescale semiconductor

1.1 Why HCS08?

The HCS08 family is intended to be an evolutionary step from the M68HC08 family. The HCS08 family improves low-voltage/low-power performance without sacrificing CPU performance.

HCS08 MCUs are created by an advanced wafer-fabrication process. This gives HCS08s a normal operating voltage of 1.8 V to 2.7 V. As a result, HCS08s are capable of higher bus speeds at lower operating voltages. Also as a result, an on-board voltage regulator is required to accept supply voltages above 2.7 V.

Most M68HC08s were designed to operate directly from a 5-V supply. The first versions of the HCS08 MCU have been designed to operate from a maximum 3.6-V supply. The trade-off for limiting the maximum voltage is that the HCS08s have much higher performance at voltages as low as 1.8 V, and the voltage regulator requires less current than a 5-V regulator would.

2 New Modes of Operation

The HCS08 family introduces two new low-power modes and one new debug mode.

2.1 Low-Power Modes

The new low-power modes are stop1 and stop2. Stop1 is a full power-down mode in which the internal voltage regulator is turned off to conserve the most power possible. Therefore, all internal systems of the MCU are powered down, including RAM and registers. As a result, the MCU requires re-initialization upon waking from stop1 mode.

Stop2 is a partial power-down mode in which most internal systems are turned off, but RAM remains powered. The registers are powered down in this mode, so they must be re-initialized; however, because RAM is powered in this mode, register values can be saved and restored from RAM. The voltage regulator is also turned off in this mode. The I/O pins remain latched in the state they were in upon entering stop2. Stop2 mode uses more power than stop1 and less power than stop3.

Stop3 is basically the same as the traditional M68HC08 stop mode. In this mode, the regulator is put into a “loose regulation” mode in which it consumes little current but can still support the static state of RAM and registers. One difference between stop3 and M68HC08 stop mode is the way stop recovery is managed. On the M68HC08 family, stop recovery is a timed event, lasting either 32 or 4096 oscillator cycles plus the time the oscillator source uses to start. On the HCS08 family, stop recovery is based on the voltage regulator’s time to power up and return to full regulation.

For more information on HCS08 stop modes, see Freescale document AN2493: *MC9S08GB/GT Low Power Modes*.

2.2 Background Debug vs. Monitor

HCS08s use a new module for in-circuit debugging that offers significant advantages over other debugging modes. The M68HC08 monitor mode is an on-chip debug mode with five commands: READ, WRITE, IREAD, IWRITE, and RUN. Monitor mode is implemented by a combination of hardware and firmware that redirects the reset and SWI vectors when enabled. These redirected vectors force the CPU to execute

the monitor firmware instead of the normal user program. Therefore, monitor mode is always intrusive because it disrupts the normal program flow. This code takes control of one of the I/O port pins, usually PTA0, so a pin is lost during debug sessions. Monitor mode is also dependent on the internal bus frequency. Changing the bus frequency during a debug session will cause the host to lose communication with the MCU.

The background debugger on HCS08 MCUs comprises two hardware modules and no on-chip firmware. The background debug controller (BDC) module provides a true single-pin communication interface to an HCS08 MCU. Usually, this pin is reserved for BDC, so no additional general-purpose I/O port pin is needed during debugging. BDC can run from the bus clock or an alternate 8-MHz clock source from the ICG when the bus clock is too slow for communication.

The BDC has 30 commands; 13 are non-intrusive and allow reads and writes of on-chip memory without interrupting CPU operation. Also included is a sync command that allows the host to determine the communication speed so the bus clock can be changed during a debug session without losing communication. Unlike the M68HC08 monitor mode, the BDC never requires high voltage on any pin.

The other hardware module that supports debugging in the HCS08 family is the on-chip in-circuit emulator (ICE). Unlike M68HC08s, HCS08s do not have any form of expanded bus mode. To perform ICE functions, the emulator's most important aspects have been included on-chip. This on-chip ICE system has the following features:

- Two trigger comparators that can trigger on two addresses or one address and one data
- One 8-word capture buffer that can capture change-of-flow addresses or event-only data
- Two types of breakpoints or triggers: address or instruction opcode
- Nine trigger modes

The reason for these advanced debugging features is to eliminate the need for a costly external emulator system. With the BDC, the user can perform these debugging functions using the actual silicon instead of an emulator that may or may not perform exactly like the real silicon.

For an in-depth comparison of the M68HC08 and HCS08 debug modes, see Freescale document AN2497: *HCS08 Background Debug Mode versus M68HC08 Monitor Mode*.

3 CPU

The CPU of the HCS08 has a few enhancements over the M68HC08 CPU but still maintains backwards code compatibility. Several new opcodes have been added to improve C code compiler efficiency. Also, many instructions have slightly different cycle counts compared to the M68HC08 family.

3.1 New Opcodes

The instruction to enter the new background debug mode, BGND, has been added to the original M68HC08 opcode map. To improve C code compiler efficiency, several addressing modes for manipulating the 16-bit H:X index register have been added to the original three instructions. A total of 10 new opcodes have been added to the M68HC08 opcode map, as shown in Table 1.

Table 1. . New HCS08 Opcodes

Instruction	Addressing Mode(s)	New Opcodes
LDHX	EXT,IX,IX1,IX2,SP1	$32,$9EAE,$9ECE,$9EBE,$9EFE
CPHX	EXT,SP1	$3E,$9EF3
STHX	EXT,SP1	$96,$9EFF
BGND	INH	$82

3.2 Instruction Cycle Counts

Many of the instructions from the M68HC08 family have increased or decreased cycle counts by one or two cycles. For the instructions increased by one or two cycles, remember that the increased bus speed performance of the HCS08 will easily offset any cycle count increase. Because of these cycle changes, the HCS08 is capable of much faster bus speeds than the M68HC08. Today, the M68HC08 is capable of an 8 MHz bus speed, and the HCS08 is capable of a 20 MHz bus speed. Future HCS08 MCUs may be capable of bus speeds greater than 20 MHz.

Code written for the M68HC08 using software delay loops may require rewriting for the HCS08, depending on which instructions were used. For this reason, Freescale strongly encourages the use of the timer module to generate delays. Use of the timer instead of software loops greatly simplifies porting code from an M68HC08 MCU to an HCS08 MCU. Refer to Table 2, Table 3, Table 4, and Table 5 for lists of cycle changes on HCS08 MCUs.

Table 2. HCS08 instructions Reduced by One Cycle

Instruction	Address Modes	M68HC08 Cycles	HCS08 Cycles
DIV	INH	7	6
DAA	INH	2	1
TAP	INH	2	1
CLI	INH	2	1
SEI	INH	2	1

Table 3. HCS08 Instructions Reduced by Two Cycles

Instruction	Address Modes	M68HC08 Cycles	HCS08 Cycles
NSA	INH	3	1

Table 4. HCS08 Instructions Increased by One Cycle (Higher Speed of HCS08 Offsets Increase)

Instruction	Address Modes	M68HC08 Cycles	HCS08 Cycles
DBNZA	INH	3	4
DBNZX	INH	3	4

Table 4. HCS08 Instructions Increased by One Cycle (continued)
(Higher Speed of HCS08 Offsets Increase)

Instruction	Address Modes	M68HC08 Cycles	HCS08 Cycles
PULA	INH	2	3
PULX	INH	2	3
PULH	INH	2	3
STOP	INH	1	2
WAIT	INH	1	2
BSETn	DIR	4	5
BCLRn	DIR	4	5
CPHX	DIR	4	5
NEG	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
COM	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
ASL	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
ASR	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
LSL	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
LSR	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
ROL	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
ROR	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
DEC	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
INC	DIR,IX,IX1,SP1	4,3,4,5	5,4,5,6
TST	DIR,IX,IX1,SP1	3,2,3,4	4,3,4,5
JSR	DIR,EXT,IX	4,5,4	5,6,5
JMP	DIR,EXT,IX	2,3,2	3,4,3
BSR	REL	4	5
CBEQ	IX+	4	5
ADC	IX	2	3
ADD	IX	2	3
AND	IX	2	3
BIT	IX	2	3
CMP	IX	2	3
CPX	IX	2	3
EOR	IX	2	3
LDA	IX	2	3
LDX	IX	2	3
ORA	IX	2	3
SBC	IX	2	3
SUB	IX	2	3

Table 5. HCS08 Instructions Increased by Two Cycles (Higher Speed of HCS08 Offsets Increase)

Instruction	Address Modes	M68HC08 Cycles	HCS08 Cycles
DBNZ	DIR,IX,IX1,SP1	3,2,3,4	5,4,5,6
CLR	DIR,IX,IX1,SP1	5,4,5,6	7,6,7,8
RTI	INH	7	9
RTS	INH	4	6
SWI	INH	9	11

4 Clocks

This section describes the differences between the system clocks on M68HC08 and HCS08 MCUs. Section 6.1, “Internal Clock Generator (ICG),” compares the ICG modules of the two MCU families.

Div2 vs. Div4

On M68HC08 MCUs, the output of the primary clock source (whether it is an external crystal, PLL, internal oscillator, etc.) is divided by four to create the system bus clock. To obtain an 8-MHz bus clock, the oscillator must run at 32 MHz.

On the HCS08 MCUs, the clock source is divided by two instead of four. So to obtain the same 8-MHz bus, only a 16-MHz oscillator is required.

4.1 CPU vs. Bus Clock

The M68HC08 CPU clock is equal to the bus clock. The HCS08 CPU clock is twice the speed of the bus clock. A typical HCS08 has a maximum bus speed of 20 MHz and a maximum CPU speed of 40 MHz. The documented cycle times for the CPU instructions are still referenced to bus cycles.

5 Memory

5.1 Elimination of Monitor ROM

All M68HC08s, whether they are ROM-based or Flash-based, have some amount of ROM on-chip. In monitor mode, a small (approximately 200 byte) software routine resides in monitor ROM. In addition, these monitor ROMs sometimes include test firmware and/or Flash programming routines.

As discussed in Section 2, “New Modes of Operation,” monitor mode has been eliminated completely in HCS08 MCUs and replaced with the non-intrusive background debug module. This eliminates the need for any on-chip ROM other than for user program memory on ROM-based devices.

5.2 Flash Command Interface (CI)

HCS08s have a simplified command interface (CI) for programming, erasing, and blank checking the Flash array. The CI makes modifying the Flash array much easier from within the MCU application for tasks such as using a block of Flash as EEPROM or upgrading the application firmware in the field.

The M68HC08 Flash interface is much less automated and requires the user to generate several delays between setting register bits to latch data and enable the programming voltage. As a comparison, programming one byte of Flash on an HCS08 MCU requires six steps, and none require a time delay. Programming one byte of Flash on an M68HC08 MCU requires 13 steps, four require the user to generate a time delay.

Key advantages of HCS08 CI:

- Handles all timing delays automatically. The user simply must set the Flash clock divider register (FCDIV) to generate the specified Flash clock frequency from the bus frequency. On the M68HC08, the user is responsible for calculating all timing delays.
- Has a blank check feature to verify that the entire array is blank. M68HC08s do not have any such feature.
- Has an error check feature that is set if the Flash command procedure is not followed correctly. M68HC08s do not have this feature.

5.3 Vector Redirection

Flash-based HCS08s have an optional vector redirection feature; Flash-based M68HC08s do not. This feature is used in conjunction with the Flash block protection. The MCU’s vectors are protected from programming and erasing whenever the minimal level of Flash block protection is enabled. Vector redirection allows for redirecting all interrupt vectors (excluding the reset vector) to unprotected Flash locations.

This feature is valuable if in-application reprogramming will be used to upgrade the MCU’s firmware. A portion of code that does not change can be protected along with the reset vector so if an interruption occurs during the reprogramming process, the device will still have a basic program in the memory and can attempt the programming process again. However, flexibility is maintained so, if necessary, all interrupt vectors can be redirected to new addresses in the upgraded firmware.

This redirection occurs in hardware, so the application’s interrupt service routines have the same timing as non-redirected vectors. When a redirected interrupt occurs, the vector is fetched directly from the unprotected Flash address instead of the usual Flash address in the protected area.

5.4 EEPROM

On HCS08s the Flash simplified command interface (CI) is shared by EEPROM for programming, erasing, and blank checking the EEPROM array. The CI makes modifying the EEPROM array easier from within the MCU application for tasks such as storing diagnostic data or upgrading the application firmware in the field.

Key advantages of the HCS08 CI:

- The HCS08 CI has a blank check feature to verify that the entire array is blank. M68HC08s do not have any such feature.
- The HCS08 CI has an error check feature that is set if the EEPROM command procedure is not followed correctly. M68HC08s do not have this feature.

The EEPROM registers set from the MC9S08DZ60 are shown in Table 6.

Table 6. MC9S08DZ60 EEPROM Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FCDIV	DIVLD	PRDIV8	DIV					
FOPT	KEYEN	FNORED	EPGMOD	0	0	0	SEC	
FTSTMOD	0	MRDS		0	0	0	0	0
FCNFG	0	EPGSEL	KEYACC	ECCDIS	0	0	0	0
FPROT	EPS		FPS					
FSTAT	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
FCMD	FCMD							

6 Peripheral Changes

6.1 Internal Clock Generator (ICG)

A new internal clock generator module (ICG) was designed for the HCS08 family. The MC68HC908GT16 has a similar ICG module, so we will use it for this comparison. Both ICG versions have a frequency-locked loop (FLL) circuit for generating multiple frequencies from a single clock reference. Both have an internal oscillator to use as the FLL reference clock.

Key enhancements:

- The HCS08 ICG has four modes of operation:
  - Self-clocked mode (SCM)
  - FLL-engaged internal reference (FEI)
  - FLL-engaged external reference (FEE)
  - FLL-bypassed external reference (FBE)

The M68HC08 has only two modes of operation: FEI and FBE.

HCS08 MCUs and M68HC08 MCUs allow switching between available clock modes. However, the M68HC08 requires the user to enable the new clock source and monitor when it becomes stable (through a status bit) before making the switch. The HCS08 ICG automates this process so that the user simply selects the new clock source and the ICG will make the switch when the new clock source is stable. Status bits are available to determine the current status of the ICG.

The HCS08 ICG has a multiplication factor and division factor to generate multiple bus frequencies. The multiplier is used only in the FLL-engaged clock modes and is selectable from the eight even integer

values from 4 through 18. The frequency divider is available in all four clock modes and is selectable from the eight values equal to 2^n, where n={0 through 7}. The M68HC08 ICG has only one multiplication factor equal to any integer from 1 to 127.

HCS08s and M68HC08s have clock monitor circuits to protect against unexpected loss of clock source. M68HC08s allow for only interrupts when a clock source is lost. HCS08s allow for either an interrupt or a reset when loss of clock is detected.

Table 7. MC9S08GB60A ICG Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ICGC1	HG0	RANGE	REFS	CLKS1	CLKS0	OSCSTEN	LOCD	0
ICGC2	LOLRE	MFD2	MFD1	MFD0	LOCRE	RFD2	RFD1	RFD0
ICGS1	CLKST1	CLKST0	REFST	LOLS	LOCK	LOCS	ERCS	ICGIF
ICGS2	0	0	0	0	0	0	0	DCOS
ICGFLTU	0	0	0	0	FLT11	FLT10	FLT9	FLT8
ICGFLTL	FLT7	FLT6	FLT5	FLT4	FLT3	FLT2	FLT1	FLT0
ICGTRM	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0

Table 8. MC68HC908GT16 ICG Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ICGCR	CMIE	CMF	CMON	CS	ICGON	ICGS	ECGON	ECGS
ICGMR		N6	N5	N4	N3	N2	N1	N0
ICGTR	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
ICGDVR					DDIV3	DDIV2	DDIV1	DDIV0
ICGDSR	DSTG7	DSTG6	DSTG5	DSTG4	DSTG3	DSTG2	DSTG1	DSTG0

= Reserved

6.2 Internal Clock Source (ICS)

The internal clock-source module (ICS) is a new module designed for the HCS08 family. It is very similar to the previously described ICG module with a number of significant differences:

Key differences:

- HCS08 ICS has three new operation modes:
  - FLL-bypassed internal reference (FBI)
  - FLL-bypassed internal reference low power (FBILP)
  - FLL-bypassed external reference low power (FBELP)

  The FLL is disabled in FBILP and FBELP modes to save power.
- The HCS08 ICS does not have self-clocked mode (SCM)
- The HCS08 ICS has a fixed FLL multiplier of 1024 unlike the HCS08 ICG that uses a multiplication and division factor to generate multiple bus frequencies. The HCS08 ICS overcomes

this by using a bus frequency divider on the FLL output to generate different bus frequencies. The bus frequency divider can be changed at anytime, and the switch to the new frequency will occur immediately

- The HCS08 ICS requires a reference frequency within the range of 31.25 kHz to 39.0625 kHz for its FLL input. A reference divider is provided to divide down the selected clock source to the required value. The HCS08 ICG does not have a reference divider. It can use any reference frequency within the ranges of 32 kHz to 100 kHz or 2 MHz to 10 MHz.
- The HCS08 ICS allows switching between clock modes similar to the HCS08 ICG; however, when switching between FEI mode and FEE mode, be careful to ensure that the FLL reference frequency stays within the range 31.25 kHz to 39.0625 kHz.
- The HCS08 ICS has 9 trim bits. The HCS08 ICG has only 8 trim bits
- The HCS08 ICS does not have a LOCK status bit.
- The HCS08 does not have a clock monitor.

Table 9. MC9S08QG8 ICS Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ICSC1	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
ICSC2	BDIV		RANGE	HGO	LP	EREFS	ERCLKEN	EREFSTEN
ICSTRM	TRIM							
ICSSC	0	0	0	0	CLKST		OSCINIT	FTRIM

= Reserved

6.3 Multi-Purpose Clock Generator (MCG)

The multi-purpose clock generator (MCG) is a new clock module designed for the high-end HCS08 family which provides several clock source choices for the MCU. It can be considered an amalgamation of the ICG and ICS modules. We shall compare it to the clock-generator module of a similar high end M68HC08 part, the AZ60A clock-generator module (CGM).

Key enhancements of HCS08 MCG:

- HCS08 MCG has eight clock modes
  - FLL-engaged internal (FEI)
  - FLL-engaged external (FEE)
  - FLL-bypassed internal reference (FBI)
  - FLL-bypassed external reference (FBE)
  - PLL-engaged external (PEE)
  - PLL-bypassed external reference (PBE)
  - FLL-bypassed internal reference low power (FBILP)
  - FLL-bypassed external reference low power (FBELP)

- The HCS08 MCG contains a frequency-locked loop (FLL) and a phase-locked loop (PLL) that are controllable by an internal or external reference clock. The M68HC08 CGM contains a PLL and no internal oscillator.
- The HCS08 MCG does not require any external components. The M68HC08 CGM requires a dedicated pin for the PLL filter components.
- The HCS08 MCG is capable of generating a 20 MHz bus clock. The M68HC08 CGM can generate an 8 MHz bus clock.
- The HCS08 MCG has a VCO divider and a bus divider to generate multiple bus frequencies. The VCO divider is used only in PLL engaged modes and is selectable in integer steps of four from 4 through 40. FLL engaged modes have a fixed multiplication factor of 1024. The bus divider is available in all clock modes and is selectable from 1, 2, 4 or 8. The M68HC08 CGM has only a multiplication factor equal to any integer from 1 to 15.
- The HCS08 MCG uses a reference divider to ensure that the reference clock falls within the input frequency ranges for the FLL and PLL. The FLL has an input range of 31.25 kHz to 39.0625 kHz. The PLL has a reference range of 1 MHz to 2 MHz. The M68HC08 CGM can select an appropriate VCO frequency range for its PLL.
- HCS08 MCUs allow switching between available clock modes. The HCS08 MCG automates this process so the user simply selects the new clock source and reference clock, while ensuring the reference frequency remains within the range required by the PLL or FLL. The MCG will make the switch when the new clock source is stable. Status bits are available to determine the current status of the MCG.
- The HCS08 MCG and the M68HC08 CGM have clock-monitor circuits that can detect a loss of lock and force an interrupt. The HCS08 MCG also allows for a reset when a loss of external clock is detected.

Table 10 shows the MCG registers for the MC9S08DZ60. Table 11 shows the CGM registers for the MC68HC908AZ32A.

Table 10. MC9S08DZ60 MCG Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MCGC1	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
MCGC2	BDIV		RANGE	HGO	LP	EREFS	ERCLKEN	EREFSTEN
MCGTRM	TRIM							
MCGSC	LOLS	LOCK	PLLST	IREFST	CLKST		OSCINIT	FTRIM
MCGC3	LOLIE	PLLS	CME	0	VDIV			
MCGT	0	0	0	0	0	0	0	0

= Reserved

Table 11. MC68HC908AZ32A CGM Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCTL	PLLIE	PLLF	PLLON	BCS	1	1	1	1
PBWC	AUTO	LOCK	$\overline{\text{ACQ}}$	XLD	0	0	0	0
PPG	MUL7	MUL6	MUL5	MUL4	VRS7	VRS6	VRS5	VRS4

6.4 Timer (TPM)

A new timer called the timer/PWM module (TPM) was designed for the HCS08 family. It performs all the functions of the M68HC08 family's timer interface module (TIM). The HCS08 TPM also reduces the complexity of the M68HC08 TIM functions and improves the use of MCU resources.

Key enhancements:

- The TPM has an up/down count mode. The TIM only counts up.
- The TPM clock source can be the bus clock, an external clock, or the fixed system clock (XCLK, typically the external oscillator input to the ICG). The TIM clock source is limited to the bus clock or an external clock.
- The TPM has eight selectable prescalers; the TIM has seven.
- Any single channel can be configured for buffered PWM on the TPM.
  The TIM requires two channels to generate a buffered PWM.
- Center-aligned PWM signals can be created with the TPM. This is not possible with the TIM.

The register interface has been modified to make programming the TPM easier. The TPM and the TIM registers are provided in Table 12 and Table 13

Table 12. MC9S08GB60A TPM Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TPMxSC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
TPMxCNTH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
TPMxCNTL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TPMxMODH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
TPMxMODL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TPMxCnSC	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	0	0
TPMxCnVH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
TPMxCnVL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Table 13. MC68HC908GT16 TIM Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TSC	TOF0	TOIE	TSTOP	TRST	0	PS2	PS1	PS0
TCNTH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
TCNTL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMODH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
TMODL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TSCn	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	TOVn	CHnMAX
TCHnH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
TCHnL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

The 16-bit registers (e.g., TPMxCNTH:TPMxCNTL) on TPM can be accessed either high byte or low byte first. Accessing one byte latches the counter until the other byte is accessed. On the M68HC08 TIM, the high byte must always be accessed first to latch the current values.

6.4.1 Configuring Buffered PWM M68HC08

Configuring a buffered PWM is easier for an HCS08. Steps for configuring a buffered PWM for the M68HC08 TIM:

1. Stop the TIM counter by setting TSTOP and reset the counter by setting TRST, both in the TSC register. Also, the clock prescaler can be selected by writing the PS2:PS1:PS0 bits in the same write.
2. Write the appropriate value for the PWM period in the TMODH:TMODL registers. TMODH must be written first, or the timer overflow function will not operate.
3. Write the appropriate value for the duty cycle to an even numbered TIM channel's TCHnH:TCHnL registers. An even numbered channel must be used when configuring a buffered PWM. The next channel, n+1, will not be available for other functions because the TCHn+1H:TCHn+1L registers will be used to change the duty cycle.
4. Write 1:0 to the MSnB:MSnA bits of the even numbered TSCn register to select buffered PWM. Also, set the TOVn bit to toggle output on overflow and write either 1:0 (clear output on compare) or 1:1 (set output on compare) to the ELSnB:ELSnA bits. Do not set up to toggle output on compare (0:1) because this prevents reliable 0% duty cycle generation.
5. When it is time to start the PWM, clear the TSTOP bit in the TSC register.
6. To update the duty cycle, write the new value to TCHn+1H:TCHn+1L registers. The update will take place on the next timer overflow. The following update must be made to the TCHnH:TCHnL registers because the duty cycle is determined by the last channel's registers to be written. Therefore, software must keep track of which channel was written last.

6.4.2 Configuring Buffered PWM HCS08

For the HCS08 family, the steps to configure for buffered PWM are:

1. Stop the TPM by writing 0:0 to the clock-source select bits, CLKSB:CLKSA, in the TPMxSC register. During this write, also set the desired clock prescaler by writing to the PS2:PS1:PS0 bits and set the CPWMS bit if center-aligned PWM is desired (center-aligned PWM is not an option on the M68HC08 TIM).
2. Write the appropriate value for the PWM period in the TPMxMODH:TPMxMODL registers. Either the high or low byte can be written first.
3. Write the appropriate value for the duty cycle to any TPM channel's TPMxCnVH:TPMxCnVL registers. Either the high or low byte can be written first.
4. If edge-aligned PWM has been selected (CPWMS = 0), set the MSnB bit of the TPMxCnSC register. If CPWMS = 1, the MSnB bit has no effect. For either edge- or center-aligned PWM, write 1:0 to the ELSBnB:ELSnA bits to clear output on compare or X:1 to set output on compare.
5. When it is time to start the PWM, write the appropriate value to the clock-select bits (CLKSB:CLKSA) in the TPMxSC register to select the desired clock source for the TPM.
6. To update duty cycle, write the new value to the TPMxCnVH:TPMxCnVL registers. The high or low byte can be written first, and the new value will take effect the next time the counter matches the modulus value.

Comparing these steps shows that the TPM is much easier to program for buffered PWM signals, mainly because the TPM does not require tracking two different registers to perform buffered updates. The other functions—input captures, output compares, and unbuffered PWM signals (actually the TPM always buffers the PWM)— of the two timer modules are configured in a similar manner. However, the TPM will always be a little easier to configure; no limitations are placed on the order of accessing the 16-bit registers.

6.5 Serial Peripheral Interface (SPI)

The HCS08 family SPI module is based on the M68HC08 family SPI module. However, several enhancements were made to improve functionality. The HCS08 SPI can perform most functions of the M68HC08 SPI.

Key enhancements of HCS08 SPI:

- The HCS08 SPI has eight selectable baud-rate prescalers in addition to eight selectable baud rates. The M68HC08 SPI has only four selectable baud rates.
- The HCS08 has a double-buffered receiver; the M68HC08 does not.
- The HCS08 SPI has a bidirectional mode; the M68HC08 SPI does not.
- The HCS08 SPI can be configured for LSB- or MSB-first operation. The M68HC08 SPI always sends the MSB first.
- The HCS08 SPI can be configured to shut down automatically in wait mode to conserve power; the M68HC08 SPI cannot.

- When configured as a master, the HCS08 SPI can use the $\overline{SS}$ pin as an automatic slave select output, a master mode fault detect input, or a general-purpose I/O pin. On the M68HC08 family, the $\overline{SS}$ pin can be used only as a master mode fault detect input or a general-purpose I/O pin.

Functions of the M68HC08 SPI eliminated on HCS08 SPI:

- The M68HC08 family supports wired-OR-mode; the HCS08 family does not.
- The M68HC08 family has an interruptible error flag for receiver overruns (when a new byte is received before the previous byte has been read). The HCS08 family has a double-buffered receiver and, therefore, does not require a flag or interrupt for this condition.
- The M68HC08 family has a separate interrupt enable bit (ERRIE) for two error conditions: receiver overflow (OVRF) or master mode fault detect (MODF). The HCS08 family uses the SPI interrupt enable bit (SPIE) to enable both the receive buffer full (SPRF) and MODF interrupts.
- Table 14 and Table 15 show the register sets for the HCS08 and M68HC08 SPI modules.

Table 14. MC9S08GB60A Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPIC1	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
SPIC2	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
SPIBR	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
SPIS	SPRF	0	SPTEF	MODF	0	0	0	0
SPID	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Table 15. MC68HC908GT16 SPI Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPIC	SPIE		SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
SPIS	SPRF	ERRIE	OVRF	MODF	SPTEF	MODFEN	SPR1	SPR0
SPID	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

= Reserved

The most significant differences between the two SPI modules are the baud-rate generation and the double-buffered receiver. The baud-rate generation of the M68HC08 SPI is limited to four baud rate choices based from the internal bus clock. The four choices are the bus clock divided by 2, 8, 32, or 128.

The HCS08 has many more choices due to the baud-rate prescaler. The prescaler can be one of eight integer values (from 1 to 8). The baud rate is then selected from one of eight divisor values (2, 4, 8, 16, 32, 64, 128, or 256). The resulting baud rate is then determined by the following equation:

baud rate = bus clock ÷ prescaler ÷ divisor

This equation results in 64 possible HCS08 baud rates for a given bus frequency. Several of those possible values will be repeated; for example, the two following combinations would result in the same baud rate value:

- prescaler = 1, divisor = 16
- prescaler = 2, divisor = 8

6.6 Inter-Integrated Circuit (IIC)

The HCS08 IIC module was derived from the HCS12 IIC and, therefore, has several changes from the M68HC08 MMIIC. We will compare the standard M68HC08AP64A MMIIC module to the HCS08 IIC because these are the basic IIC modules for each family.

Key enhancements of HCS08 IIC:

- The HCS08 IIC is software programmable for one of 64 different serial-clock frequencies. The M68HC08 is limited to eight serial-clock frequencies.
- The HCS08 IIC features a 10-bit address extension. The M68HC08 MMIIC does not have this feature.
- The HCS08 IIC has a general call recognition feature in 7-bit and 10-bit addressing. When this feature is enabled, the IIC matches the general call address ($00) as well as its own slave address. The M68HC08 MMIIC does not have this feature.
- The HCS08 IIC can generate an interrupt on a match of a received calling address to its slave address. The M68HC08 MMIIC will generate an interrupt on every received calling address.
- The HCS08 IIC uses the same register for transmit and receive data. The M68HC08 MMIIC uses two separate registers.
- The HCS08 uses a transfer complete flag (TCF) to indicate that a byte of data has been successfully transferred or received by the IIC. The M68HC08 MMIIC uses two separate flags for this function.
- The HCS08 IIC uses a single interrupt flag (IICIF) for Transmit Empty, Receive Full and Arbitration Loss. The M68HC08 MMIIC has separate interrupt flags for all of these.

Key features not available on the HCS08 IIC:

- The M68HC08 MMIIC is SMBus (System Mangement Bus) V1.0 and V1.1 compatible with CRC generation. The HCS08 IIC does not support this feature.
- The M68HC08 MMIIC offers automatic switching of transmit or receive mode; this has to be performed by user software on the HCS08 IIC.

Table 16. MC9S08DZ60 IIC Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IICA	AD7	AD6	AD5	AD4	AD3	AD2	AD1	0
IICF	MULT		ICR					
IICC1	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
IICS	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
IICD	DATA							
IICC2	GCAEN	ADEXT	0	0	0	AD10	AD9	AD8

Table 17. M68HC908AP64 IIC Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MMADR	MMAD7	MMAD6	MMAD5	MMAD4	MMAD3	MMAD2	MMAD1	MMEXTAD
MMCR1	MMEN	MMIEN	MMCLRBB	0	MMTXAK	REPSEN	MMCRCBYTE	0
MMCR2	MMALIF	MMNAKIF	MMBB	MMAST	MMRW	0	0	MMCRCEF
MMSR	MMRXIF	MMTXIF	MMATCH	MMSRW	MMRXAK	MMCRCBF	MMTXBE	MMRXBF
MMDTR	MMTD7	MMTD6	MMTD5	MMTD4	MMTD3	MMTD2	MMTD1	MMTD0
MMDRR	MMRD7	MMRD6	MMRD5	MMRD4	MMRD3	MMRD2	MMRD1	MMRD0
MMCRDR	MMCRCD7	MMCRCD6	MMCRCD5	MMCRCD4	MMCRCD3	MMCRCD2	MMCRCD1	MMCRCD0
MMFDR	0	0	0	0	0	MMBR2	MMBR1	MMBR0

6.7 Freescale's Controller Area Network (MSCAN)

The HCS08 MSCAN is an extended version of the M68HC08 MSCAN. Both MSCANs support the CAN protocol specification version 2 A and B. Furthermore both MSCAN:

- Support remote frame handling
- Use triple-buffered transmit storage scheme with internal prioritization
- Have a programmable wake-up functionality with low-pass filter
- Have a programmable loop-back mode
- Have separate signalling and interrupts capabilities for all CAN receiver and transmitter error states
- Support receive and transmit time stamping
- Have a low power sleep mode
- Have a programmable clock source: CPU bus clock or crystal oscillator output
- Have a flexible maskable identifier filter

The HCS08 MSCAN has additional features and enhancements compared to the M68HC08 MSCAN. Additional features implemented in the HCS08 MSCAN:

- Listen only mode for monitoring the CAN bus

- Programmable bus off recovery functionality
- Internal timer for time stamping

Table 18. Differences Between M68HC08 and HCS08 Implementation

Enhancements	M68HC08	HCS08
Numbers of receive buffers	2 (one foreground and one background buffer)	5 (one foreground and four background buffer)
Number of identifier filters	one 32 bit Filter or two 16 bit Filters or four 8 bit Filters	two 32 bit Filters or four 16 bit Filters or eight 8 bit Filters
Time stamp storage	Timer link to timer module with software storage	Automatic storage in transmit or receive buffer with using MSCAN internal 16 bit timer
Transmit buffers memory map	Direct access to all three transmit buffers	The three transmit buffer are paged (single memory address space) and are accessible via transmit buffer selection flag
Number of control register	9	13
Memory space for entire MSCAN module	128	64

Table 19. MC9S08DZ60 MSCAN Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CANCTL0	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
CANCTL1	CANE	CLKSRC	LOOPB	LISTEN	0	WUPM	SLPAK	INITAK
CANBTR0	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
CANBTR1	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
CANRFLG	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
CANRIER	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
CANTFLG	0	0	0	0	0	TXE2	TXE1	TXE0
CANTIER	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
CANTARQ	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
CANTAAK	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
CANTBSEL	0	0	0	0	0	TX2	TX1	TX0
CANIDAC	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
Reserved	0	0	0	0	0	0	0	0
CANMISC	0	0	0	0	0	0	0	BOHOLD
CANRXERR	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
CANTXERR	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
CANIDAR0 – CANIDAR3	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0

Table 19. MC9S08DZ60 MSCAN Registers (continued)

CANIDMR0 – CANIDMR3	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
CANIDAR4 – CANIDAR7	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
CANIDMR4 – CANIDMR7	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
CANTTSRH	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
CANTTSRL	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
CANRIDR0	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
CANRIDR1	ID2	ID1	ID0	RTR	IDE	—	—	—
CANRIDR2	—	—	—	—	—	—	—	—
CANRIDR3	—	—	—	—	—	—	—	—
CANRDSR0 – CANRDSR7	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
CANRDLR	—	—	—	—	DLC3	DLC2	DLC1	DLC0
Reserved	—	—	—	—	—	—	—	—
CANRTSRH	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
CANRTSRL	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
CANTIDR0	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
CANTIDR1	ID2	ID1	ID0	RTR	IDE	—	—	—
CANTIDR2	—	—	—	—	—	—	—	—
CANTIDR3	—	—	—	—	—	—	—	—
CANTDSR0 – CANTDSR7	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
CANTDLR	—	—	—	—	DLC3	DLC2	DLC1	DLC0
CANTTBPR	PRIO7	PRIO6	PRIO5	PRIO4	PRIO3	PRIO2	PRIO1	PRIO0

Table 20. MC68HC908AZ32A MSCAN08 Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMCR0	0	0	0	SYNCH	TLNKEN	SLPAK	SLPRQ	SFTRES
CMCR1	0	0	0	0	0	LOOPB	WUPM	CLKSRC
CBTR0	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
CBTR1	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
CRFLG	WUPIF	RWRNIF	TWRNIF	RERRIF	TERRIF	BOFFIF	OVRIF	RXF
CRIER	WUPIE	RWRNIE	TWRNIE	RERRIE	TERRIE	BOFFIE	OVRIE	RXFIE
CTFLG	0	ABTAK2	ABTAK1	ABTAK0	0	TXE2	TXE1	TXE0

Table 20. MC68HC908AZ32A MSCAN08 Registers (continued)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CTCR	0	ABTRQ2	ABTRQ1	ABTRQ0	0	TXEIE2	TXEIE1	TXEIE0
CIDAC	0	0	IDAM1	IDAM0	0	0	IDHIT1	IDHIT0
CRXERR	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
CTXERR	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
CIDAR0	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
CIDAR1	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
CIDAR2	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
CIDAR3	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
CIDMR0	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
CIDMR1	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
CIDMR2	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
CIDMR3	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0

6.8 Serial Communications Interface (SCI)

The HCS08 SCI module was derived from the HCS12 SCI and, therefore, has several changes from the M68HC08 SCI. The M68HC08 family actually has two different SCI modules: standard SCI and enhanced ESCI. We will compare the standard M68HC08 SCI module to the HCS08 SCI because these are the basic SCI modules for each family.

Key enhancements of HCS08 SCI:

- The HCS08 SCI has a 13-bit baud rate register that can be set to any integer value from 1 to 8191. The M68HC08 SCI has a 2-bit prescaler value that can be set to divide by 1, 3, 4, or 13 and a 3-bit baud rate selector that can be set to divide by 1, 2, 4, 8, 16, 32, 64, or 128 for a total of 32 combinations.
- HCS08 SCI's maximum baud rate is the bus frequency ÷ 16. For the M68HC08 SCI, it is the bus frequency ÷ 64.
- The HCS08 SCI offers a single wire mode; the M68HC08 SCI does not.
- The HCS08 SCI can be configured to stop automatically when wait mode is entered. M68HC08 SCI must be turned off by software if it is not needed in wait mode.

Key features not available on HCS08 SCI:

- The M68HC08 SCI can invert all transmitter data by setting a single control bit. The HCS08 SCI has no such function.
- The M68HC08 SCI has a flag to indicate when a break character has been received. In the HCS08 SCI, a break signal is received as a $00 character with a framing error flag set.

The SCI registers for the HCS08 family and M68HC08 family are provided in Table 21 and Table 22, respectively.

Table 21. MC9S08DZ60 SCI Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SCIxBDH	LBKDIE	RXEDGIE	0	SBR12	SBR11	SBR10	SBR9	SBR8
SCIxBDL	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
SCIxC1	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
SCIxC2	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
SCIxS1	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
SCIxS2	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
SCIxC3	R8	T8	TXDIR	0	ORIE	NEIE	FEIE	PEIE
SCIxD	R7	R6	R5	R4	R3	R2	R1	R0
	T7	T6	T5	T4	T3	T2	T1	T0

Table 22. MC68HC908AZ32A SCI Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SCC1	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
SCC2	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
SCC3	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
SCS1	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
SCS2	0	0	0	0	0	0	BKF	RPF
SCDR	R7	R6	R5	R4	R3	R2	R1	R0
	T7	T6	T5	T4	T3	T2	T1	T0
SCBR	0	0	SCP1	SCP0	R	SCR2	SCR1	SCR0

6.9 Analog-to-Digital Converters

Two new ADC modules have been designed for the HCS08 family. Both modules are 10-bit successive approximation register (SAR) architectures with sample and hold. The M68HC08 family has several ADC modules. For this comparison, we will compare the MC9S08GB60 family ADC module to the 10-bit SAR module that is found on the MC68HC908LJ12, MC68HC908MR32, and several other MCUs. Afterwards, we will also take a look at the other HCS08 ADC that is used on new HCS08 family members, such as the MC9S08DZ60.

Key enhancements:

- Each ADC pin on the HCS08 ADC has a pin-enable bit in addition to the input channel select. This allows pins that will be used for the ADC to be reserved. This also masks pins not to be used as ADC inputs so they cannot accidentally be selected by the ADC input channel select. The M68HC08 ADC does not have this feature.
- The HCS08 ADC has a power on/off bit separate from input channel select bits. Therefore, the ADC can be powered down without changing the value in ATDCH[4:0]. The M68HC08 ADC is powered down by writing %11111 to the ADCH[4:0] bits.

- The HCS08 ADC has both 8-bit and 10-bit signed modes. The M68HC08 family has only a 10-bit signed mode.
- The HCS08 family has 16 clock prescaler values, even integers from 2 to 32. The M68HC08 family has only 5: divide by 1, 2, 4, 8, or 16.
- The HCS08 ADC has a maximum clock frequency of 2.0 MHz. The M68HC08 ADC has a maximum frequency of 1 MHz.

Key features not available on the HCS08 ADC:

- The M68HC08 ADC can select CGMXCLK (the external MCU clock reference) as the ADC clock reference in addition to the bus clock. The HCS08 can use only the bus clock.
- The M68HC08 ADC requires 17 ADC clock cycles for each conversion. The HCS08 family requires 28 ADC cycles for each conversion. However, the higher possible frequency for the HCS08 ADC can actually result in a faster overall conversion time. The M68HC08 family has an effective sampling rate of 59,000 samples per second; the HCS08 ADC has a sampling rate of 71,000 samples per second.

The ADC registers for the MC9S08GB60 and MC68HC908LJ12 are provided in Table 23 and Table 24, respectively.

Table 23. MC9S08GB60A Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ATDC	ATDPU	DJM	RES8	SGN	PRS3	PRS2	PRS1	PRS0
ATDSC	CCF	ATDIE	ATDCO	ATDCH4	ATDCH3	ATDCH2	ATDCH1	ATDCH0
ATDRH	AD9 or 0	AD8 or 0	AD7 or 0	AD6 or 0	AD5 or 0	AD4 or 0	AD3 or AD9	AD2 or AD8
ATDRL	AD1 or AD7	AD0 or AD6	0 or AD5	0 or AD4	0 or AD3	0 or AD2	0 or AD1	0 or AD0
ATDPE	ATDPE7	ATDPE6	ATDPE5	ATDPE4	ATDPE3	ATDPE2	ATDPE1	ATDPE0

Table 24. M68HC908LJ12 ADC Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADSCR	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
ADRH	AD9 or 0	AD8 or 0	AD7 or 0	AD6 or 0	AD5 or 0	AD4 or 0	AD3 or AD9	AD2 or AD8
ADRL	AD1 or AD7	AD0 or AD6	0 or AD5	0 or AD4	0 or AD3	0 or AD2	0 or AD1	0 or AD0
ADCLK	ADIV2	ADIV1	ADIV0	ADICLK	MODE1	MODE0	0	0

6.9.1 New HCS08 ADC Module

Some key differences between the M9S08GB60 family ADC and the newer version of the HCS08 ADC:

- The newer ADC has a programmable sample time to add ADC clock cycles for high-impedance inputs; the M9S08GB60 ADC does not have this function.
- The newer ADC has a software-selectable low-power mode to allow operation at lower voltages than the M9S08GB60 ADC.
- The newer ADC automatically powers down after a conversion has completed. The M9S08GB60 ADC is powered down by clearing the ATDPU bit in the ATDC register.
- The newer ADC has a built-in asynchronous clock source that reduces noise during a conversion and allows operation when the bus clock is too slow for ADC operation. The M9S08GB60 ADC does not have this feature.
- The newer ADC can operate in stop3 mode when the asynchronous clock is used. The M9S08GB60 ADC cannot operate in stop3.
- The newer ADC has a hardware trigger mode. On the first MCU with this ADC, this trigger will be connected to the real-time interrupt module to start conversions automatically, even when the MCU is in stop3. The M9S08GB60 ADC does not have this feature.
- The newer ADC will have an internal channel connected to a constant voltage source to allow software compensation for voltage drift of the MCU power supply. The M9S08GB60 ADC does not have this function.
- The newer ADC has an automatic compare function. This function allows the ADC result to be compared to a value written in the ADCV registers, and the conversion complete flag (COCO) is set only if the result is greater-than or equal-to the compare value (if ACFGT = 1) or lower than the compare value (if ACFGT = 0). The M9S08GB60 ADC does not have this function.
- The newer ADC requires 19 ADC clock cycles per 10-bit conversion and 17 cycles for an 8-bit conversion. The M9S08GB60 ADC requires 28 ADC clock cycles for a 10-bit or 8-bit conversion.
- The newer ADC can operate at two effective sampling rates: 315k or 210k samples per second. The M9S08GB60 ADC has only one effective sampling rate of 71k samples per second.
- The newer ADC can run from an alternate clock source in addition to the bus clock and the asynchronous clock. This alternate source may vary depending on the configuration of the MCU.

A few features from the M9S08GB60 ADC not available on the newer ADC:

- Left-justified conversion mode has been removed from the newer ADC.
- The newer ADC has fewer ADC clock prescaler choices. The M9S08GB60 ADC has 16 selectable prescaler choices. The newer ADC has five prescaler choices when the bus clock is the selected clock input and four prescaler choices when the alternative clock is the selected input.

Table 25. MC9S08DZ60 ADC Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADSC1	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
ADSC2	ADACT	ADTRG	ACFE	ACFGT	0	0	0	0
ADRH	0	0	0	0	0	0	ADR9	ADR8
ADRL	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
ADCVH	0	0	0	0	0	0	ADCV9	ADCV8
ADCVL	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0
ADCFG	ADLPC	ADIV1	ADIV0	ADLSMP	MODE1	MODE0	ADICLK1	ADICLK0
APCTL1	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
APCTL2	ADPC15	ADPC14	ADPC13	ADPC12	ADPC11	ADPC10	ADPC9	ADPC8
APCTL3	ADPC23	ADPC22	ADPC21	ADPC20	ADPC19	ADPC18	ADPC17	ADPC16

6.10 Analog Comparator (ACMP)

The analogue comparator (ACMP) is a new module for the HCS08 family and provides a circuit for comparing two analogue voltages or for comparing one analogue voltage to an internal reference. As there is no direct M68HC08 equivalent here is a summary of the HCS08 ACMP.

Key features of the HCS08 ACMP:

- Full rail-to-rail operation.
- Selectable interrupt on rising edge, falling edge or either rising or falling edges of comparator output.
- Option to compare to fixed internal bandgap reference voltage.
- Option to allow comparator output to be visible on a pin.
- The comparator output is a digital signal that is high when the non-inverting input of the ACMP is greater than the inverting input, and is low when the non-inverting input of the ACMP is less than the inverting input.

Table 26. MC9S08DZ60 ACMP Register Set

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ACMP1SC	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD1	ACMOD0
ACMP2SC	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD1	ACMOD0

7 Conclusion

Freescale’s newest family of 8-bit microcontrollers is an evolutionary step from the highly successful M68HC08 family. Although the HCS08 family has many enhancements from the M68HC08 family, these are a natural progression and should not be barriers for programmers and designers familiar with the M68HC08 family’s functionality and features.

THIS PAGE IS INTENTIONALLY BLANK

THIS PAGE IS INTENTIONALLY BLANK

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN2717
Rev. 1
08/2006

freescale™
semiconductor