Freescale Semiconductor

MC9S08LH64RM
Rev. 5.1, 05/2012

MC9S08LH64 Reference Manual

This is the MC9S08LH64 Reference Manual set consisting of the following files:

- MC9S08LH64 Reference Manual Addendum, Rev 1
- MC9S08LH64 Reference Manual, Rev 5

© Freescale Semiconductor, Inc., 2012. All rights reserved.

Freescale Semiconductor
Reference Manual Addendum

MC9S08LH64RMAD
Rev. 1, 05/2012

MC9S08LH64 Reference Manual Addendum

This addendum describes corrections or updates to the *MC9S08LH64Reference Manual*, file named as MC9S08LH64 RM. Please check our website at http://www.freescale.com/coldfire, for the latest updates.

The current version available of the *MC9S08LH64 Reference Manual* is Revision 5.0.

Table of Contents

© Freescale Semiconductor, Inc., 2012. All rights reserved.

1 Addendum for Revision 5.0

Table 1. MC9S08LH64RM Rev 5.0 Addendum

Location	Description

Sub-section “Calibration Procedure for Improved Linearity ” for Chapter 10 ”Analog-to-Digital Converter (S08ADC16V1)”	Added a new sub-section to “Calibration Function” section:
	For applications using the ADC16 in differential mode, improved linearity may be achieved by using an adjusted calibration procedure as detailed below. The ADC16 does perform to the published datasheet specification using the original calibration procedure. The adjusted calibration procedure corrects potential calibration offset errors and diminishes linearity error spikes that may occur near the ¼, ½ and ¾ point of the full scale range.
	
	Adjusted calibration procedure:
	
	• Perform auto calibration as defined in the reference manual.
	• Then, rewrite ADCCLP1-4, ADCCLM0-4, ADCPG and ADCMG using ADCLP0 and the following calculations:
	
	ADCCLP1 = ADCCLP0 << 1; /* CLP1 is 2x CLP0 */
	ADCCLP2 = ADCCLP1 << 1; /* CLP2 is 2x CLP1 */
	ADCCLP3 = ADCCLP2 << 1; /* CLP3 is 2x CLP2 */
	ADCCLP4 = ADCCLP3 << 1; /* CLP4 is 2x CLP3 */
	ADCCLM0 = ADCCLP0; /* minus side calibration values are set equal to the plus side */
	ADCCLM1 = ADCCLP1; /* minus side calibration values are set equal to the plus side */
	ADCCLM2 = ADCCLP2; /* minus side calibration values are set equal to the plus side */
	ADCCLM3 = ADCCLP3; /* minus side calibration values are set equal to the plus side */
	ADCCLM4 = ADCCLP4; /* minus side calibration values are set equal to the plus side */
	
	ADCCLMD = ADCCLPD;
	ADCCLMS = ADCCLPS;
	
	calSum = ADCCLP0 + ADCCLP1 + ADCCLP2 + ADCCLP3 + ADCCLP4 + ADCCLPS; /* recalculate the plus gain factor */
	calSum /= 2;
	calSum += 0x8000;
	ADCPG = calSum;
	
	calSum = 0;
	calSum = ADCCLM0 + ADCCLM1 + ADCCLM2 + ADCCLM3 + ADCCLM4 + ADCCLMS; /* recalculate the minus gain factor */
	calSum /= 2;
	calSum += 0x8000;
	ADCMG = calSum;

2 Revision History

Table 2 provides a revision history for this document.

Table 2. Revision History Table

Rev. Number	Substantive Changes	Date of Release
1.0	Initial release. Added a new section “Calibration Procedure for Improved Linearity ” for Chapter 10 ”Analog-to-Digital Converter (S08ADC16V1)”.	05/2012

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

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)
www.freescale.com/support

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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.© Freescale Semiconductor, Inc. 2012. All rights reserved.

MC9S08LH64RMAD
Rev. 1
05/2012

MC9S08LH64
MC9S08LH36

Reference Manual

HCS08
Microcontrollers

MC9S08LH64RM
Rev. 5
04/2010

freescale.com

MC9S08LH64 Series Features

8-Bit HCS08 Central Processor Unit (CPU)

- Up to 40 MHz CPU at 3.6 V to 2.1 V across temperature range of –40 °C to 85 °C
- Up to 20 MHz at 2.1 V to 1.8 V across temperature range of –40 °C to 85 °C
- HC08 instruction set with added BGND instruction
- Support for up to 32 interrupt/reset sources

On-Chip Memory

- Dual array flash read/program/erase over full operating voltage and temperature
- Random-access memory (RAM)
- Security circuitry to prevent unauthorized access to RAM and flash contents

Power-Saving Modes

- Two low-power stop modes
- Reduced-power wait mode
- Low-power run and wait modes allow peripherals to run while voltage regulator is in standby
- Peripheral clock gating register can disable clocks to unused modules, thereby reducing currents
- Very low-power external oscillator that can be used in stop2 or stop3 modes to provide accurate clock source to time-of-day (TOD) module
- 6 µs typical wakeup time from stop3 mode

Clock Source Options

- Oscillator (XOSC) — Loop-control Pierce oscillator; crystal or ceramic resonator range of 31.25 kHz to 38.4 kHz or 1 MHz to 16 MHz
- Internal Clock Source (ICS) — Internal clock source module containing a frequency-locked-loop (FLL) controlled by internal or external reference; precision trimming of internal reference allows 0.2% resolution and 2% deviation over temperature and voltage; supporting bus frequencies from 1 MHz to 20 MHz

System Protection

- Watchdog computer operating properly (COP) reset with option to run from dedicated 1 kHz internal clock source or bus clock
- Low-voltage warning with interrupt
- Low-voltage detection with reset or interrupt
- Illegal opcode detection with reset; illegal address detection with reset
- Flash block protection

Development Support

- Single-wire background debug interface
- Breakpoint capability to allow single breakpoint setting during in-circuit debugging (plus two more breakpoints in on-chip debug module)
- On-chip in-circuit emulator (ICE) debug module containing three comparators and nine trigger modes

Peripherals

- LCD — Up to 8×36 or 4×40 LCD driver with internal charge pump and option to provide an internally-regulated LCD reference that can be trimmed for contrast control
- ADC —16-bit resolution; with a dedicated differential ADC input, and 8 single-ended ADC inputs; up to 2.5 µs conversion time; hardware averaging; calibration registers, automatic compare function; temperature sensor; operation in stop3; fully functional from 3.6 V to 1.8 V
- IIC — Inter-integrated circuit bus module to operate at up to 100 kbps with maximum bus loading; multi-master operation; programmable slave address; interrupt-driven byte-by-byte data transfer; broadcast mode; 10-bit addressing
- ACMP — Analog comparator with selectable interrupt on rising, falling, or either edge of comparator output; compare option to fixed internal reference voltage; outputs can be optionally routed to TPM module; operation in stop3
- SCIx — Two full-duplex non-return to zero (NRZ) modules (SCI1 and SCI2); LIN master extended break generation; LIN slave extended break detection; wakeup on active edge
- SPI — Full-duplex or single-wire bidirectional; double-buffered transmit and receive; master or slave mode; MSB-first or LSB-first shifting
- TPMx — Two 2-channel (TPM1 and TPM2); selectable input capture, output compare, or buffered edge- or center-aligned PWM on each channel
- TOD — (Time-of-day) 8-bit, quarter second counter with match register; external clock source for precise time base, time-of-day, calendar, or task scheduling functions
- VREFx — Trimmable via an 8-bit register in 0.5 mV steps; automatically loaded with room temperature value upon reset; can be enabled to operate in stop3 mode; trim register is not available in stop modes.

Input/Output

- Dedicated accurate voltage reference output pin, 1.2 V output (VREFOx); trimmable with 0.5 mV resolution
- Up to 39 GPIOs, two output-only pins
- Hysteresis and configurable pullup device on all input pins; configurable slew rate and drive strength on all output pins

Package Options

- 14mm × 14mm 80-pin LQFP, 10 mm × 10 mm 64-pin LQFP

MC9S08LH64 Reference Manual

Covers MC9S08LH64
MC9S08LH36

Related Documentation:

- **MC9S08LH64 (Data Sheet)**
  Contains descriptive feature set, block diagram, and electrical characteristics for the device.

Find the most current versions of all documents at:
http://freescale.com/

MC9S08LH64RM
Rev. 5
04/2010

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc.

© Freescale Semiconductor, Inc., 2008-2010. All rights reserved.

freescale™
semiconductor

Revision History

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

http://freescale.com/

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

Revision Number	Revision Date	Description of Changes
1	10/24/2007	***ORIGINALLY RELEASED AS MC9S08LL64*** Initial alpha customer release.
2	7/2008	***REDEFINED AS MC9S08LH64*** Originally released as MC9S08LL64 Reference Manual, Revision 1. Product redefined and is now MC9S08LH64 Reference Manual, Revision 2. Updated to SPI V4 and TPM V3 modules. Removed references to PTB3. Changed references from VREFO to VREFO1. Updated memory maps and register summary tables. Updated SPMSC1, SOPT2, SCGC1, SCGC2, and VREFSC register descriptions. Updated Stop2 wakeup and register retention information. Included some updates to the ADC Single Ended Channel Assignments and to the DC Characteristics.
3	01/2009	Refreshed the draft to include block module updates, the new VREF module, and the latest revisions.
4	03/2009	Incorporated latest revisions for customer release.
5	4/5/2010	Updated Figure 2-3; updated the registers at 0x0030, 0x0031, 0x1818–0x181B in the Table 4-2 or Table 4-4; updated PPAGE register in the Section 4.5.2.1; updated Section 4.5.2.2; added the descriptions of FLS in the Table 5-15; added Section 10.1.4, “Status and Control Registers”; updated Section 10.1.3 and Section 10.1.5; updated the input and channel at 11101 in the Table 10-3; updated Section 10.1.7.5 and Table 10-4; updated the Reset to bit of registers ADCOFS, ADCPGC, ADCCLPD, ADCCLPS, ADCCLP0–ADCCLP4; updated Table 10-2 for pin control. Added 64-pin LQFP package information for LH36. Updated Chapter 10, “Analog-to-Digital Converter (S08ADC16V1).”

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc.
© Freescale Semiconductor, Inc., 2008-2010. All rights reserved.

List of Chapters

Contents

Section Number **Title** **Page**

Chapter 1
Device Overview

Chapter 2
Pins and Connections

Chapter 3
Modes of Operation

Chapter 4
Memory

Chapter 5
Resets, Interrupts, and General System Control

Chapter 6
Parallel Input/Output Control

Chapter 7
Keyboard Interrupt (S08KBIV2)

Chapter 8
Central Processor Unit (S08CPUV4)

Chapter 9
Analog Comparator (S08ACMPVLPV1)

Chapter 10
Analog-to-Digital Converter (S08ADC16V1)

Chapter 11
Internal Clock Source (S08ICSV3)

Chapter 12
Inter-Integrated Circuit (S08IICV2)

Chapter 13
Liquid Crystal Display Driver (S08LCDV1)

Chapter 14
Serial Communications Interface (S08SCIV4)

Chapter 15
Serial Peripheral Interface (S08SPIV4)

Chapter 16
Time of Day Module (S08TODV1)

Chapter 17
Timer Pulse-Width Modulator (S08TPMV3)

Chapter 18
Voltage Reference Module (S08VREFV1)

Chapter 19
Development Support

Chapter 20
Debug Module (S08DBGV3) (64K)

Chapter 1
Device Overview

The MC9S08LH64 and MC9S08LH36 are members of the low-cost, low-power, 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.

1.1 Devices in the MC9S08LH64 Series

Table 1-1 summarizes the feature set available in the MC9S08LH64 series of MCUs.

Table 1-1. MC9S08LH64 Series Features by MCU and Package

Feature	MC9S08LH64		MC9S08LH36	
Package	**80-pin LQFP**	**64-pin LQFP**	**80-pin LQFP**	**64-pin LQFP**
FLASH	64 KB (32,768 and 32,768 Arrays)		36 KB (24,576 and 12,288 Arrays)	
RAM	4000		4000	
ACMP	yes		yes	
ADC Single-ended Channels	8-ch	8-ch	8-ch	8-ch
ADC Differential Channels[1]	1	0	1	0
IIC	yes		yes	
IRQ	yes		yes	
KBI	8		8	
SCI1	yes		yes	
SCI2	yes		yes	
SPI	yes		yes	
TPM1	2-ch		2-ch	
TPM2	2-ch		2-ch	
TOD	yes		yes	
LCD	8×36 4×40	8×24 4×28	8×36 4×40	8×24 4×28

Table 1-1. MC9S08LH64 Series Features by MCU and Package

Feature	MC9S08LH64		MC9S08LH36	
Package	80-pin LQFP	64-pin LQFP	80-pin LQFP	64-pin LQFP
VREFO1	yes	no	yes	no
VREFO2	no	yes	no	yes
I/O pins[2]	39	37	39	37

1 Each differential channel consists of two pins (DADPx and DADMx).

2 The 39 I/O pins include two output-only pins and 18 LCD GPIO.

1.2 MCU Block Diagram

The block diagram in Figure 1-1 shows the structure of the MC9S08LH64 series MCU.

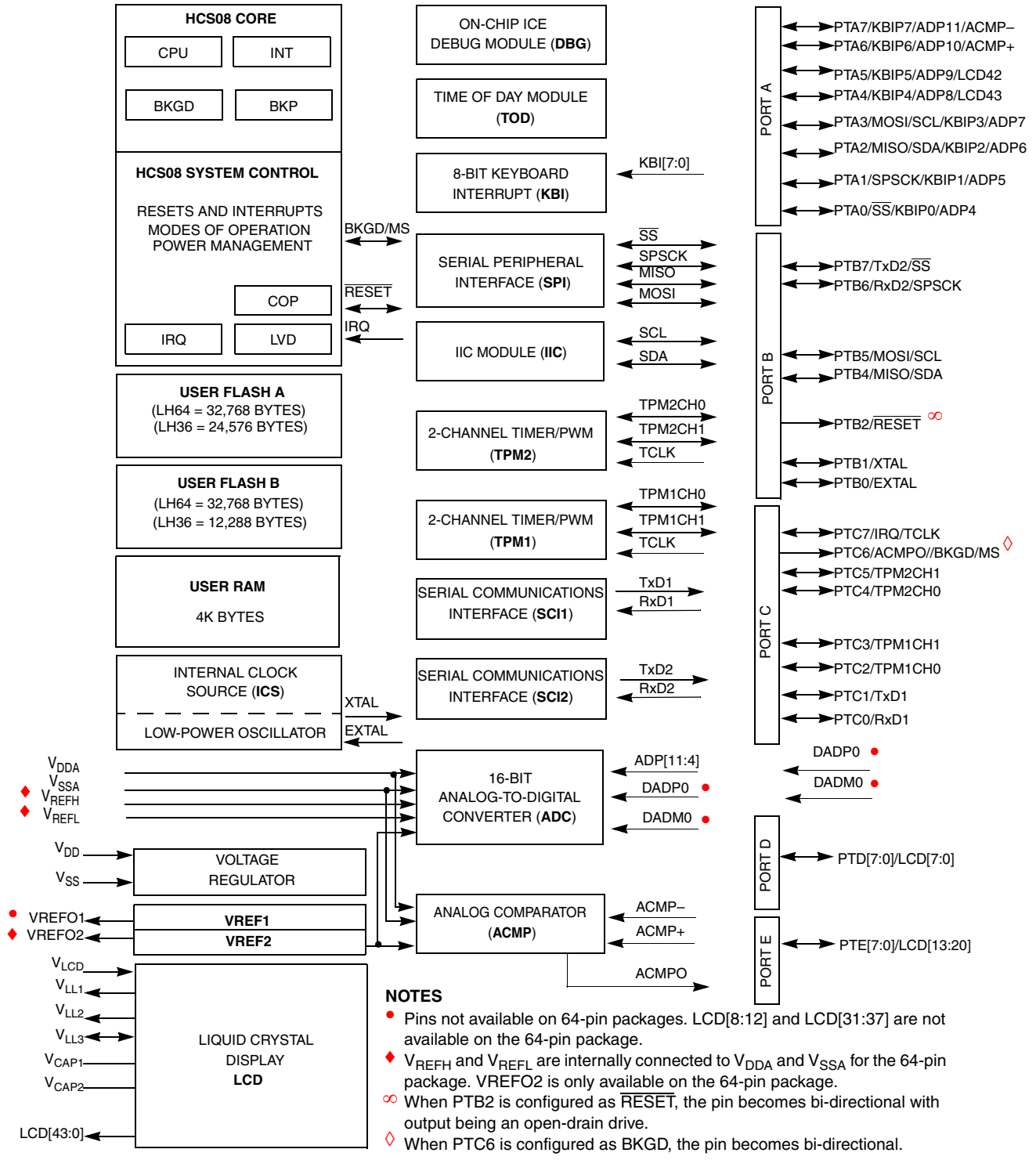


Figure 1-1. MC9S08LH64 Series Block Diagram

Table 1-2 provides the functional version of the on-chip modules.

Table 1-2. Module Versions

Module		Version
Analog Comparator	(ACMPVLP)	1
Analog-to-Digital Converter	(ADC16)	1
Central Processing Unit	(CPU)	4
Keyboard Interrupt	(KBI)	2
Internal Clock Source	(ICS)	3
Inter-Integrated Circuit	(IIC)	2
Low-Power Oscillator	(XOSCVLP)	1
Time-of-Day Counter	(TOD)	1
Serial Communications Interface	(SCI)	4
Serial Peripheral Interface	(SPI)	4
Timer Pulse-Width Modulator	(TPM)	3
Liquid Crystal Display	(LCD)	1
Debug Module	(DBG)	3
Voltage Reference	(VREF)	1

1.3 System Clock Distribution

Figure 1-2 shows a simplified clock connection diagram. Some modules in the MCU have selectable clock inputs as shown. The clock inputs to the modules indicate the clock(s) that are used to drive the module function. All memory-mapped registers associated with the modules are clocked with BUSCLK. The ICS supplies the clock sources:

- ICSOUT — This clock source is used as the CPU clock and is divided by 2 to generate the peripheral bus clock, BUSCLK. Control bits in the ICS control registers determine which of three clock sources is connected:
  - Internal reference clock
  - External reference clock
  - Frequency-locked loop (FLL) output

  See Chapter 11, “Internal Clock Source (S08ICSV3),” for details on configuring the ICSOUT clock.
- ICSLCLK — This clock source is derived from the digitally-controlled oscillator, DCO, of the ICS when the ICS is configured to use the internal or external reference clock as the reference clock. Development tools can select this internal self-clocked source (~ 8 MHz) to speed up BDC communications in systems where the bus clock is slow.
- ICSERCLK — This is the external reference clock and can be selected as the alternate clock for the ADC module. The “Optional External Reference Clock” section in Chapter 11, “Internal Clock Source (S08ICSV3),” explains the ICSERCLK in more detail. See Chapter 10, “Analog-to-Digital Converter (S08ADC16V1),” for more information regarding the use of ICSERCLK with these modules.

- ICSIRCLK — This is the internal reference clock and can be selected as TOD clock source. Chapter 11, “Internal Clock Source (S08ICSV3)” explains the ICSIRCLK in more detail. See Chapter 16, “Time of Day Module (S08TODV1),” for more information regarding the use of ICSIRCLK.
- ICSFFCLK — This generates the fixed-frequency clock (FFCLK) after being synchronized to the bus clock. The FFCLK can be selected as the clock source for the TPM modules. The frequency of the ICSFFCLK is determined by the settings of the ICS. See the Section 11.4.8, “Fixed Frequency Clock,” for details.
- LPOCLK — This clock is generated from an internal low-power oscillator that is completely independent of the ICS module. The LPOCLK can be selected as the clock source to the TOD or COP modules. See Chapter 16, “Time of Day Module (S08TODV1),” and Section 5.4, “Computer Operating Properly (COP) Watchdog” for details on using the LPOCLK with these modules.
- OSCOUT — This is the output of the XOSCVLP module and can be selected as the TOD or LCD clock source. See Chapter 16, “Time of Day Module (S08TODV1)” or Chapter 13, “Liquid Crystal Display Driver (S08LCDV1).”
- TCLK — TCLK is the optional external clock source for the TPM modules. The TCLK must be limited to 1/4th the frequency of the bus clock for synchronization. See Section 17.2.1.1, “EXTCLK — External Clock Source,” for more details.
- TODCLK— TODCLK is generated by the TOD module. The TODCLK bit enables this clock. This clock is connected to the ALTCLK input of the LCD module. See Chapter 16, “Time of Day Module (S08TODV1).”

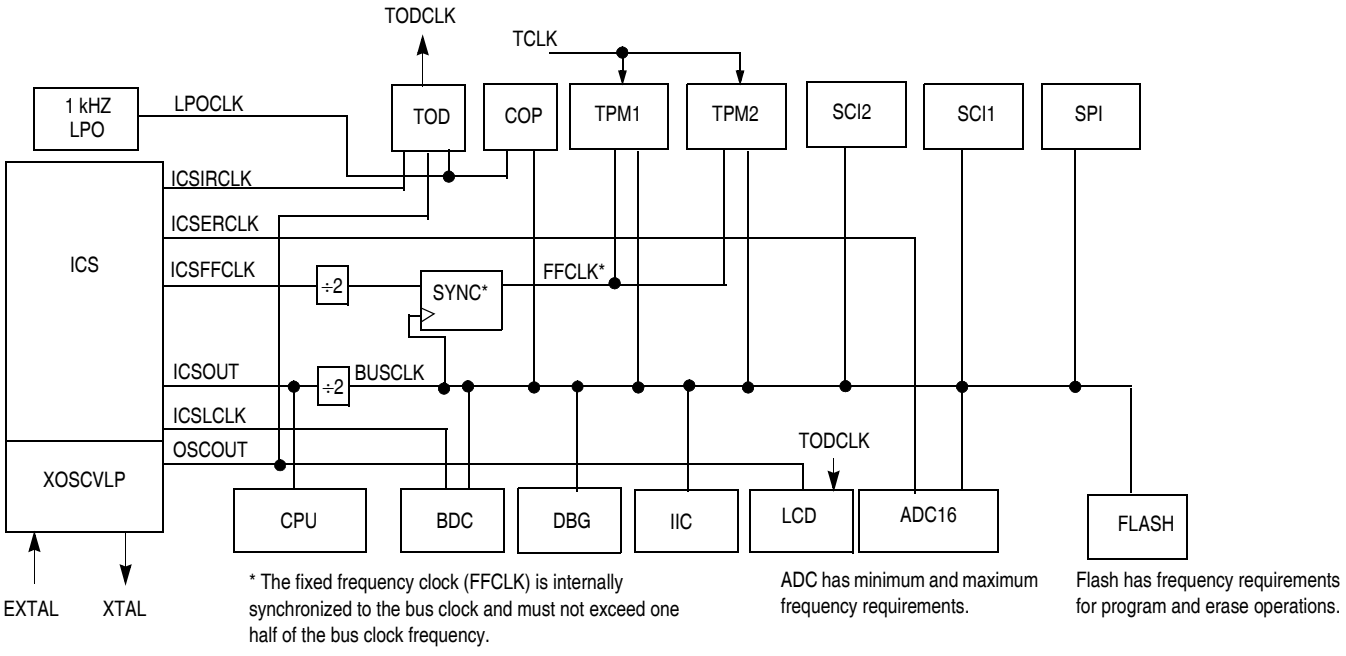


Figure 1-2. System Clock Distribution Diagram

Chapter 2
Pins and Connections

2.1 Introduction

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

2.2 Device Pin Assignment

This section shows the pin assignments for MC9S08LH64 series devices.

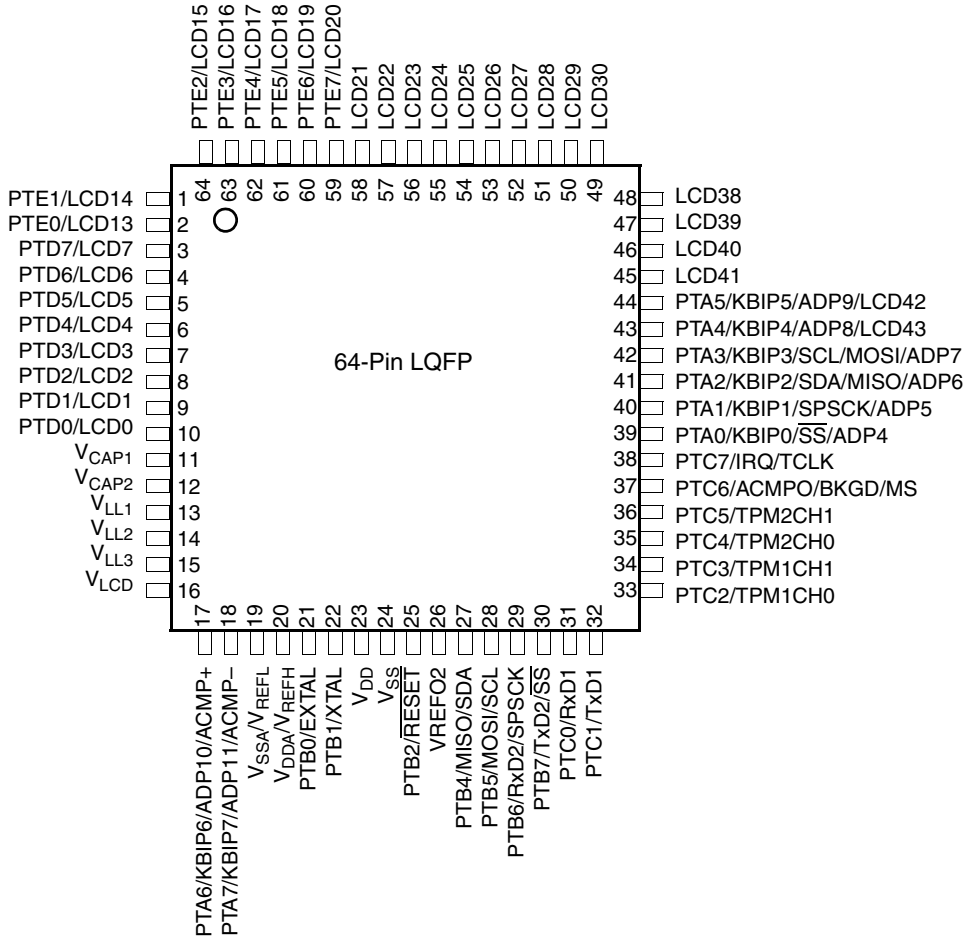


Figure 2-1. 64-Pin LQFP

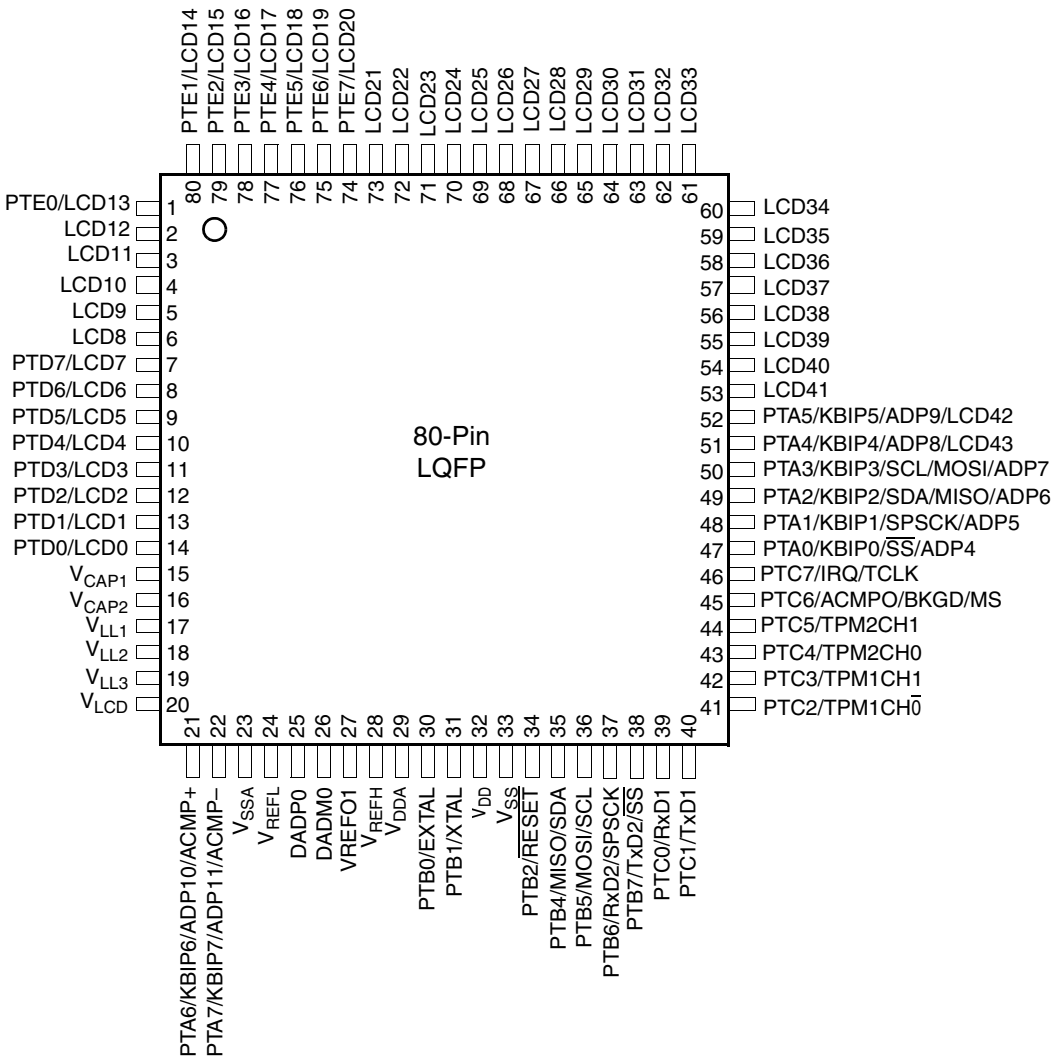
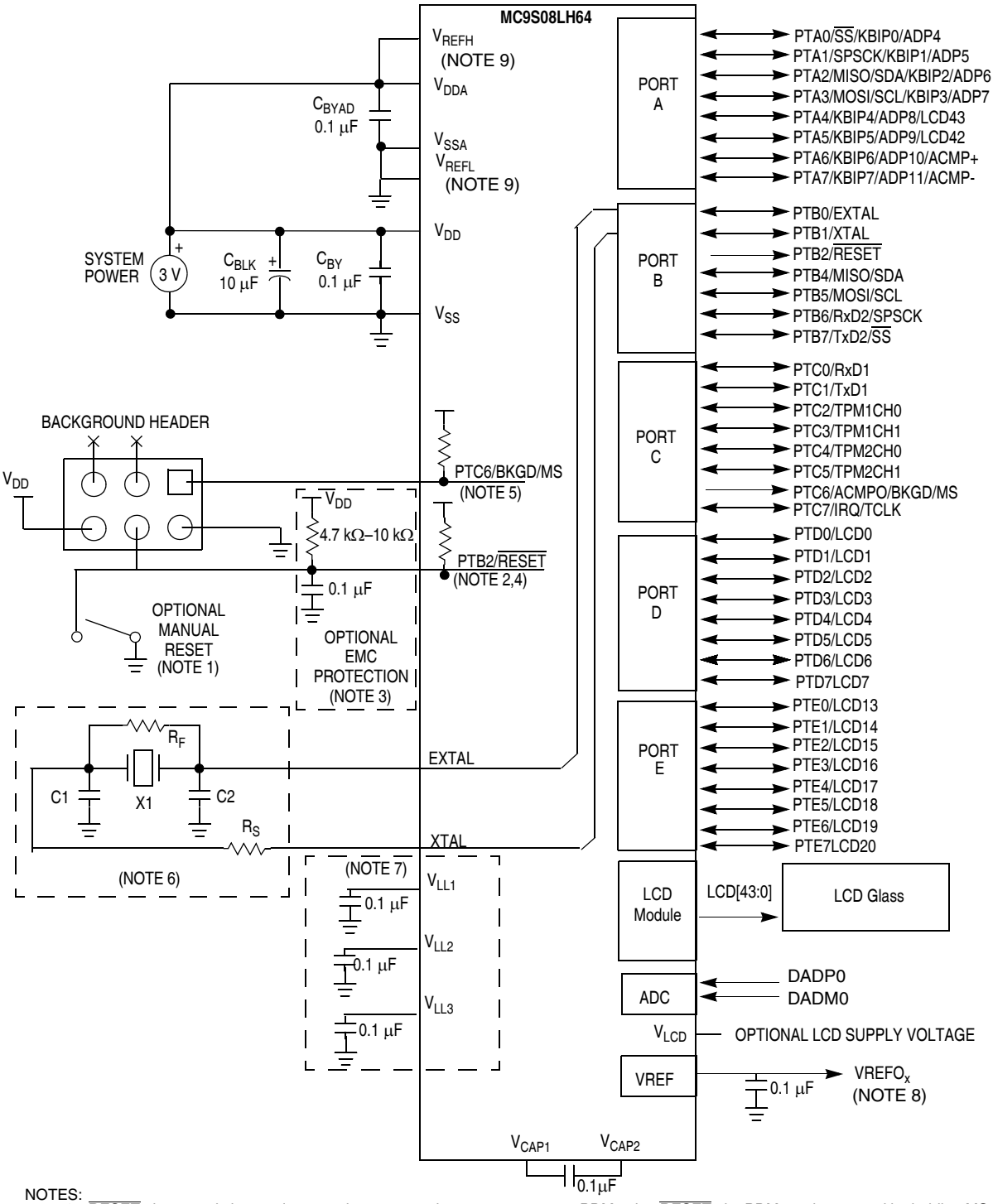


Figure 2-2. 80-Pin LQFP

2.3 Recommended System Connections

Figure 2-3 shows pin connections that are common to MC9S08LH64 series application systems.



NOTES:
1. RESET pin can only be used to reset into user mode, you can not enter BDM using RESET pin. BDM can be entered by holding MS low during POR or writing a 1 to BDFR in SBDFR with MS low after issuing BDM command.
2. RESET and IRQ features have optional internal pullup device.
3. RC filter on RESET pin recommended for noisy environments.
4. When PTB2 is configured as RESET, pin becomes bi-directional with output being open-drain drive containing an internal pullup device.
5. When PTC6 is configured as BKGD, pin becomes bi-directional.
6. When using the XOSCVLP module in low range and low power mode, the external components RF, RS, C1, and C2 are not required.
7. LCD mode shown is for Charge pump enabled, other configurations are necessary for different LCD modes.
8. VREFO1 is only available in the 80-pin package. VREFO2 is only available in the 64-pin package.
9. V_{REFH} and V_{REFL} are only available in the 80-pin package.

Figure 2-3. Basic System Connections

2.3.1 Power

V_{DD} and V_{SS} are the primary power supply pins for the MCU. This voltage source supplies power to all I/O buffer circuitry and to an internal voltage regulator. The internal voltage regulator provides a regulated lower-voltage source for the CPU and other internal circuitry of the MCU.

LCD/GPIO can be powered differently, reference Chapter 6 , “Parallel Input/Output Control” for additional information.

Typically, application systems have two separate capacitors across the power pins. In this case, there should be a bulk electrolytic capacitor, such as a 10 µF tantalum capacitor, to provide bulk charge storage for the overall system and a 0.1 µF ceramic bypass capacitor located as near to the MCU power pins as practical to suppress high-frequency noise.

V_{DDA} and V_{SSA} are the analog power supply pins for the MCU. This voltage source supplies power to the ADC and ACMP modules. A 0.1 µF ceramic bypass capacitor should be located as near to the MCU power pins as practical to suppress high-frequency noise.

The V_{REFH} and V_{REFL} pins are the voltage reference high and voltage reference low inputs, respectively for the ADC module. A 0.1 µF ceramic bypass capacitor should be located as near to the VREFx pins as practical to suppress high-frequency noise.

2.3.2 Oscillator

Immediately after reset, the MCU uses an internally-generated clock provided by the internal clock source (ICS) module. The oscillator can be configured to run in stop2 or stop3 modes. For more information on the ICS, see Chapter 11, “Internal Clock Source (S08ICSV3)”.

The oscillator (XOSCVLP) in this MCU is a Pierce oscillator that can accommodate a crystal or ceramic resonator. An external clock source can optionally be connected to the EXTAL input pin.

Refer to Figure 2-3 for the following discussion. R_S (when used) and R_F should be low-inductance resistors such as carbon composition resistors. Wire-wound resistors and some metal film resistors have too much inductance. C1 and C2 normally should be high-quality ceramic capacitors that are specifically designed for high-frequency applications.

R_F provides a bias path to keep the EXTAL input in its linear range during crystal startup; its value is not generally critical. Typical systems use 1 MΩ to 10 MΩ. Higher values are sensitive to humidity and lower values reduce gain and (in extreme cases) could prevent startup.

C1 and C2 are typically in the 5 pF to 25 pF range and are chosen to match the requirements of a specific crystal or resonator. Be sure to consider printed circuit board (PCB) capacitance and MCU pin capacitance when selecting C1 and C2. The crystal manufacturer typically specifies a load capacitance which is the series combination of C1 and C2 (which are usually the same size). As a first-order approximation, use 10 pF as an estimate of combined pin and PCB capacitance for each oscillator pin (EXTAL and XTAL).

When using the oscillator in low-range and low-gain mode, the external components R_S, R_F, C_1, and C_2 are not required.

2.3.3 RESET

After a power-on reset (POR), the PTB2/RESET pin defaults to RESET. Clearing RSTPE in SOPT1, configures the pin to be an output-only pin with an open-drain drive containing an internal pullup device. When configured as an output, the pin remains as an output until the next POR or LVD reset. When enabled, the RESET pin can be used to reset the MCU from an external source when the pin is driven low.

Internal power-on reset and low-voltage reset circuitry typically make external reset circuitry unnecessary. This pin is normally connected to the standard 6-pin background debug connector so a development system can directly reset the MCU system. A manual external reset can be added by supplying a simple switch to ground (pull reset pin low to force a reset).

Whenever any non-POR reset is initiated (whether from an external signal or from an internal system), the enabled RESET pin is driven low for about 34 bus cycles. The reset circuitry decodes the cause of reset and records it by setting a corresponding bit in the system reset status register (SRS).

NOTE

- This pin does not contain a clamp diode to V_{DD} and must not be driven above V_{DD}.
- The voltage on the internally pulled up RESET pin when measured will be below V_{DD}. The internal gates connected to this pin are pulled to V_{DD}. If the RESET pin is required to drive to a V_{DD} level, an external pullup must be used.
- In EMC-sensitive applications, an external RC filter is recommended on the RESET pin if enabled. See Figure 2-3 for an example.

2.3.4 Background / Mode Select (BKGD/MS)

During a power-on reset (POR) or background debug force reset (see Section 5.8.3, “System Background Debug Force Reset Register (SBDFR),” for more information), the PTC6/ACMPO/BKGD/MS pin functions as a mode select pin. Immediately after any reset, the pin functions as the background pin and can be used for background debug communication. When enabled as the BKGD/MS pin (BKGDPE = 1), an internal pullup device is automatically enabled.

The background debug communication function is enabled when BKGDPE in SOPT1 is set. BKGDPE is set following any reset of the MCU and must be cleared to use the alternative PTC6/ACMPO/BKGD/MS pin functions.

After any reset, if nothing is connected to this pin, the MCU enters normal operating mode. If a debug system is connected to the 6-pin standard background debug header, it can hold BKGD/MS low. It can do this during a POR or after issuing a background debug force reset. This forces the MCU to active background mode.

The BKGD/MS pin is used primarily with BDC communications, and features a custom protocol that uses 16 clock cycles of the target MCU’s BDC clock per bit time. The target MCU’s BDC clock can run as fast as the bus clock, so no significant capacitance should be connected to the BKGD/MS pin that could interfere with background serial communications.

Although the BKGD/MS pin is a pseudo open-drain pin, the background debug communication protocol provides brief, actively driven, high speedup pulses to ensure fast rise times. Small capacitances from cables and the absolute value of the internal pullup device play a minimal role in determining rise and fall times on the BKGD/MS pin.

2.3.5 VREFOx Pins

The VREFO1 and VREFO2 are the voltage reference output pins. The VREFO1 pin is available in the 80-pin package only. The VREFO2 pin is available in the 64-pin package only. A 0.1uF bypass capacitor should be used on these pins.

2.3.6 Dedicated ADC Pins

The DADP0 and DADM0 pins are dedicated differential ADC pins. These pins are not available on the 64-pin package.

2.3.7 LCD Pins

2.3.7.1 LCD Power Pins

The V_{LCD}, V_{LL1}, V_{LL2}, V_{LL3}, V_{CAP1}, and V_{CAP2} pins are dedicated to providing power to the LCD module. For detailed information about LCD pins, see Chapter 13, “Liquid Crystal Display Driver (S08LCDV1).”

2.3.7.2 LCD Driver Pins

The MC9S08LH64 series of MCUs contains 44 LCD driver pins on the 80-pin package, 32 LCD driver pins on the 64-pin package. For both the 80-pin and 64-pin packages, 18 of the LCD driver pins are multiplexed with GPIO. If the LCD module is disabled, the LCD driver pins are high-impedance and LCD/GPIO pins are configured as GPIO. The LCD module is automatically disabled after resets except for stop2 wakeup. For detailed information about LCD driver pins, see Chapter 13, “Liquid Crystal Display Driver (S08LCDV1).”

2.3.8 General-Purpose I/O (GPIO) and Peripheral Ports

The MC9S08LH64 series of MCUs support up to 39 general-purpose I/O pins, 2 output-only pins, which are shared with on-chip peripheral functions (timers, serial I/O, LCD, ADC, ACMP, and so forth). The GPIO output-only pins (PTC6/ACMPO/BKGD/MS and PTB2/$\overline{\text{RESET}}$) are bi-directional when configured as BKGD and $\overline{\text{RESET}}$, respectively.

GPIO that is multiplexed with LCD pins can be configured to reference V_{DD} or V_{LL3}. See Section 6.1, “Introduction” for more info.

When a port pin is configured as a general-purpose output or when a peripheral uses the port pin as an output, software can select one of two drive strengths and enable or disable slew rate control. When a port

pin is configured as a general-purpose input or when a peripheral uses the port pin as an input, software can enable a pullup device.

PTB2 is a special case I/O pin. When the PTB2/$\overline{\text{RESET}}$ pin is configured as PTB2 input with the pullup enabled, the voltage observed on the pin is not pulled to V_{DD}. However, the internal voltage on the PTB2 node will be at V_{DD}.

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. For information about controlling these pins as general-purpose I/O pins, see Chapter 6 , "Parallel Input/Output Control."

NOTE

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

When using the 64-pin device, either enable on-chip pullup devices or change the direction of non-bonded pins to outputs so the pins do not float.

Table 2-1. Pin Availability by Package Pin-Count

		<-- Lowest **Priority** --> Highest				
80	**64**	**Port Pin**	**Alt 1**	**Alt 2**	**Alt3**	**Alt4**
1	2	PTE0	LCD13			
2		LCD12				
3		LCD11				
4		LCD10				
5		LCD9				
6		LCD8				
7	3	PTD7	LCD7			
8	4	PTD6	LCD6			
9	5	PTD5	LCD5			
10	6	PTD4	LCD4			
11	7	PTD3	LCD3			
12	8	PTD2	LCD2			
13	9	PTD1	LCD1			
14	10	PTD0	LCD0			
15	11	V_{CAP1}				
16	12	V_{CAP2}				
17	13	V_{LL1}				
18	14	V_{LL2}				
19	15	V_{LL3}				
20	16	V_{LCD}				

Table 2-1. Pin Availability by Package Pin-Count (continued)

		<-- Lowest **Priority** --> Highest				
80	**64**	**Port Pin**	**Alt 1**	**Alt 2**	**Alt3**	**Alt4**
21	17	PTA6	KBIP6	ADP10	ACMP+	
22	18	PTA7	KBIP7	ADP11	ACMP–	
23	19	V_{SSA}				
24		V_{REFL}				
25		DADP0				
26		DADM0				
27		VREFO1				
28	20	V_{REFH}				
29		V_{DDA}				
30	21	PTB0		EXTAL		
31	22	PTB1		XTAL		
32	23	V_{DD}				
33	24	V_{SS}				
34	25	PTB2	$\overline{\text{RESET}}$			
	26	VREFO2				
35	27	PTB4	MISO	SDA		
36	28	PTB5	MOSI	SCL		
37	29	PTB6	RxD2	SPSCK		
38	30	PTB7	TxD2	$\overline{\text{SS}}$		
39	31	PTC0	RxD1			
40	32	PTC1	TxD1			
41	33	PTC2	TPM1CH0			
42	34	PTC3	TPM1CH1			
43	35	PTC4	TPM2CH0			
44	36	PTC5	TPM2CH1			
45	37	PTC6	ACMPO	BKGD	MS	
46	38	PTC7	IRQ	TCLK		
47	39	PTA0	KBIP0		$\overline{\text{SS}}$	ADP4
48	40	PTA1	KBIP1		SPSCK	ADP5
49	41	PTA2	KBIP2	SDA	MISO	ADP6
50	42	PTA3	KBIP3	SCL	MOSI	ADP7
51	43	PTA4	KBIP4	ADP8	LCD43	
52	44	PTA5	KBIP5	ADP9	LCD42	
53	45	LCD41				
54	46	LCD40				
55	47	LCD39				
56	48	LCD38				
57		LCD37				

Table 2-1. Pin Availability by Package Pin-Count (continued)

		<-- Lowest **Priority** --> Highest				
80	**64**	**Port Pin**	**Alt 1**	**Alt 2**	**Alt3**	**Alt4**
58		LCD36				
59		LCD35				
60		LCD34				
61		LCD33				
62		LCD32				
63		LCD31				
64	49	LCD30				
65	50	LCD29				
66	51	LCD28				
67	52	LCD27				
68	53	LCD26				
69	54	LCD25				
70	55	LCD24				
71	56	LCD23				
72	57	LCD22				
73	58	LCD21				
74	59	PTE7	LCD20			
75	60	PTE6	LCD19			
76	61	PTE5	LCD18			
77	62	PTE4	LCD17			
78	63	PTE3	LCD16			
79	64	PTE2	LCD15			
80	1	PTE1	LCD14			

Chapter 3
Modes of Operation

3.1 Introduction

The operating modes of the MC9S08LH64 series are described in this chapter. Entry into each mode, exit from each mode, and functionality while in each of the modes are described.

3.2 Features

- Active background mode for code development.
- Run mode — CPU clocks can be run at full speed and the internal supply is fully regulated.
- LPRUN mode — CPU and peripheral clocks are restricted to 125 kHz maximum and the internal voltage regulator is in standby.
- Wait mode — CPU shuts down to conserve power; system clocks are running and full regulation is maintained.
- LPWAIT mode — CPU shuts down to conserve power; peripheral clocks are restricted to 125 kHz maximum and the internal voltage regulator is in standby.
- Stop modes — System clocks are stopped and voltage regulator is in standby.
  - Stop3 — All internal circuits are powered for fast recovery.
  - Stop2 — Partial power down of internal circuits, RAM content is retained, LCDPENx, LCDBPENx, and LCDWFx registers' content is retained, and I/O states held.

3.3 Run Mode

This is the normal operating mode for the MC9S08LH64 series. In this mode, the CPU executes code from internal memory with execution beginning at the address fetched from memory at 0xFFFE–0xFFFF after reset.

3.3.1 Low-Power Run Mode (LPRun)

In the low-power run mode, the on-chip voltage regulator is put into its standby state. This state uses the minimum power consumption necessary for CPU functionality. To reduce power consumption further, clear the appropriate bits in the SCGC1 and SCGC2 registers to disable any unused peripheral clocks.

Before entering this mode, the following conditions must be met:

- FBELP is the selected clock mode for the ICS.
- The HGO bit in the ICSC2 register is clear.
- The bus frequency is less than 125 kHz.

- If enabled, configure the ADC to use the asynchronous clock source, ADACK, to meet the ADC minimum frequency requirements. The bandgap channel can be converted if the VREF module is enabled in low-power run mode.
- If enabled, configure the TOD to use an external clock source (OSCOUT) or the 1 kHz low-power oscillator.
- The LVDE or LVDSE bit in SPMSC1 register must be clear. LVD and LVW will automatically be disabled.
- Flash programming/erasing is not allowed.
- The VREF module must be enabled to supply the bandgap for the ACMP option to compare to internal bandgap reference in LPRUN and LPWAIT.

Once these conditions are met, low power run mode can be entered by setting the LPR bit in the SPMSC2 register.

To re-enter standard run mode, clear the LPR bit. The LPRS bit in the SPMSC2 register is a read-only status bit that can be used to determine if the regulator is in full regulation mode or not. When LPRS is ‘0’, the regulator is in full-regulation mode and the MCU can run at full speed in any clock mode.

3.3.1.1 Interrupts in Low-Power Run Mode

Low power run mode provides the option to return to full regulation if any interrupt occurs. This is done by setting the LPWUI bit in the SPMSC2 register. The ICS can then be set for full speed immediately in the interrupt service routine.

If the LPWUI bit is clear, interrupts will be serviced in low power run mode.

If the LPWUI bit is set, LPR and LPRS bits will be cleared and interrupts will be serviced with the regulator in full regulation.

3.3.1.2 Resets in Low-Power Run Mode

Any reset will exit low power run mode, clear the LPR and LPRS bits, and return the device to normal run mode.

3.4 Active Background Mode

The active background mode functions are managed through the BDC in the HCS08 core. The BDC and the on-chip debug module (DBG), provide the means for analyzing MCU operation during software development.

Active background mode is entered in any of six ways:

- When the BKGD/MS pin is low during POR
- When the BKGD/MS pin is low immediately after issuing a background debug force reset (see Section 5.8.3, “System Background Debug Force Reset Register (SBDFR))
- When a BACKGROUND command is received through the BKGD/MS pin
- When a BGND instruction is executed

- When encountering a BDC breakpoint
- When encountering a DBG breakpoint

After entering active background mode, the CPU is held in a suspended state while it waits for serial background commands instead of executing instructions from the user application program.

Background commands are of two types:

- Non-intrusive commands, defined as commands that can be issued while the user program is running. Non-intrusive commands can be issued through the BKGD pin while the MCU is in run mode; non-intrusive commands can also be executed when the MCU is in the active background mode. Non-intrusive commands include:
  - Memory access commands
  - Memory-access-with-status commands
  - BDC register access commands
  - The BACKGROUND command
- Active background commands, which can only be executed while the MCU is in active background mode. Active background commands include commands to:
  - Read or write CPU registers
  - Trace one user program instruction at a time
  - Leave active background mode to return to the user application program (GO)

The active background mode is used to program a bootloader or user application program into the flash program memory before the MCU is operated in run mode for the first time. When the MC9S08LH64 Series is shipped from the Freescale Semiconductor factory, the flash program memory is erased by default unless specifically noted. As a result, no program can be executed in run mode until the flash memory is initially programmed. The active background mode can also be used to erase and reprogram the flash memory after it has been previously programmed.

For additional information about the active background mode, refer to the Development Support chapter.

3.5 Wait Mode

Wait mode is entered by executing a WAIT instruction. Upon execution of the WAIT instruction, the CPU enters a low-power state in which it is not clocked. The I bit in CCR is cleared when the CPU enters the wait mode, enabling interrupts. When an interrupt request occurs, the CPU exits the wait mode and resumes processing, beginning with the stacking operations that lead to the interrupt service routine.

While the MCU is in wait mode, there are some restrictions on which background debug commands can be used. Only the BACKGROUND command and memory-access-with-status commands are available when the MCU is in wait mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from wait mode and enter active background mode.

3.5.1 Low-Power Wait Mode (LPWait)

Low power wait mode is entered by executing a WAIT instruction while the MCU is in low power run mode. In the low power wait mode, the on-chip voltage regulator remains in its standby state (as in the low power run mode). This state uses the minimum power consumption necessary for most modules to maintain functionality. Power consumption is most reduced by disabling the clocks to all unused peripherals by clearing the corresponding bits in the SCGC register.

The same restrictions on the low power run mode apply to low power wait mode.

3.5.1.1 Interrupts in Low-Power Wait Mode

If the LPWUI bit is set when the WAIT instruction is executed, then the voltage regulator will return to full regulation when wait mode is exited. The ICS can be set for full speed immediately in the interrupt service routine.

If the LPWUI bit is clear when the WAIT instruction is executed, an interrupt will return the device to low power run mode.

If the LPWUI bit is set when the WAIT instruction is executed, an interrupt will return the device to normal run mode with full regulation and the LPR and LPRS bits will be cleared.

3.5.1.2 Resets in Low-Power Wait Mode

Any reset will exit low power wait mode, clear the LPR and LPRS bits, and return the device to normal run mode.

3.6 Stop Modes

One of two stop modes (stop2 or stop3) is entered upon execution of a STOP instruction when the STOPE bit in the system option 1 register (SOPT1) is set. In both stop modes, the bus and CPU clocks are halted. In stop3 the voltage regulator is in standby. In stop2 the voltage regulator is in partial powerdown. The ICS module can be configured to leave the reference clocks running. See Chapter 11, “Internal Clock Source (S08ICSV3)” for more information.

If the STOPE bit is not set when the CPU executes a STOP instruction, the MCU will not enter either stop mode and an illegal opcode reset is forced. The stop modes are selected by setting the appropriate bits in the System Power Management Status and Control 2 Register (SPMSC2).

Table 3-1 shows all of the control bits that affect stop mode selection and the mode selected under various conditions. The selected mode is entered following the execution of a STOP instruction.

Table 3-1. Stop Mode Selection

Register	SOPT1	BDCSCR	SPMSC1		SPMSC2	Stop Mode
Bit name	STOPE	ENBDM [1]	LVDE	LVDSE	PPDC	
	0	x	x		x	Stop modes disabled; illegal opcode reset if STOP instruction executed

Table 3-1. Stop Mode Selection (continued)

Register	SOPT1	BDCSCR	SPMSC1		SPMSC2	Stop Mode
Bit name	STOPE	ENBDM [1]	LVDE	LVDSE	PPDC	
	1	1	x		x	Stop3 with BDM enabled [2]
	1	0	Both bits must be 1		x	Stop3 with voltage regulator active
	1	0	Either bit a 0		0	Stop3
	1	0	Either bit a 0		1	Stop2

1 ENBDM is located in the BDCSCR which is only accessible through BDC commands, see the Chapter 19, "Development Support."

2 When in stop3 mode with BDM enabled, the SI_{DD} is near RI_{DD} levels because internal clocks are enabled.

3.6.1 Stop2 Mode

Stop2 mode is entered by executing a STOP instruction under the conditions as shown in Table 3-1. Most of the internal circuitry of the MCU is powered off in stop2 with the exception of the RAM, optionally the TOD, the low power oscillator, voltage reference and the LCD module. Upon entering stop2, all I/O pin control signals are latched so that the pins retain their states during stop2. LCD driver pins continue to drive the signals necessary to display the LCD data.

Exit from stop2 is performed by asserting the wakeup pin (PTB2/$\overline{\text{RESET}}$, PTC7/IRQ, or TOD interrupt) on the MCU.

NOTE

When PTB2/$\overline{\text{RESET}}$ or PTC7/IRQ is used as an active low wakeup source, it must be configured as an input prior to executing a STOP instruction to avoid an immediate exit from stop2. PTB2/$\overline{\text{RESET}}$ and PTC7/IRQ can be disabled as a wakeup if it is configured as a high driven output. For lowest power consumption in stop2, this pin must not be left open if configured as input (enable the internal pullup or tie an external pullup device or set pin as output driving a high value).

In addition, the time of day module (TOD) can wake the MCU from stop2, if enabled.

Upon wakeup from stop2 mode, the MCU starts up as from a power-on reset (POR):

- All module control and status registers are reset, except for SPMSC1-SPMSC3, TODC, TODSC, TODCNT, TODM, LCDPENx, LCDBPENx and LCDWFRx.
- The LVD reset function is enabled and the MCU remains in the reset state if V_{DD} is below the LVD trip point
- The CPU takes the reset vector

In addition to the above, upon waking from stop2, the PPDF bit in SPMSC2 is set. This flag is used to direct user code to go to a stop2 recovery routine. PPDF remains set and the I/O pin states remain latched until a 1 is written to PPDACK in SPMSC2.

If using the low power oscillator during stop2, the user must reconfigure the ICSC2 register which contains oscillator control bits before PPDACK is written.

To maintain I/O states for pins that were configured as GPIO before entering stop2, the user must restore the contents of the I/O port registers to the port registers before writing to the PPDACK bit. If the port registers are not restored from RAM before writing to PPDACK, then the pins will switch to their reset states when PPDACK is written.

For pins that were configured as peripheral I/O, the user must reconfigure the peripheral module that interfaces to the pin before writing to the PPDACK bit. If the peripheral module is not enabled before writing to PPDACK, the pins will be controlled by their associated port control registers when the I/O latches are opened.

If enabled, LCD functionality continues in stop2 mode and upon stop2 recovery the LCD control registers (LCDC0, LCDC1, LCDSUPPLY, LCDRVC, LCDBCTL, and LCDS) must be re-initialized before writing the PPDACK.

3.6.2 Stop3 Mode

Stop3 mode is entered by executing a STOP instruction under the conditions shown in Table 3-1. The states of all of the internal registers and logic, RAM contents, and I/O pin states are maintained.

Stop3 can be exited by asserting $\overline{\text{RESET}}$, or by an interrupt from one of the following sources: the TOD, LVD, LVW, ADC, ACMP, IRQ, SCI, LCD or the KBI.

If stop3 is exited by means of the $\overline{\text{RESET}}$ pin, then the MCU is reset and operation will resume after taking the reset vector. Exit by means of one of the internal interrupt sources results in the MCU taking the appropriate interrupt vector.

3.6.3 Active BDM Enabled in Stop Mode

Entry into the active background mode from run mode is enabled if the ENBDM bit in BDCSCR is set. This register is described in Chapter 19, “Development Support.” If ENBDM is set when the CPU executes a STOP instruction, the system clocks to the background debug logic remain active when the MCU enters stop mode. Because of this, background debug communication remains possible. In addition, the voltage regulator does not enter its low-power standby state but maintains full internal regulation. If the user attempts to enter stop2 with ENBDM set, the MCU will instead enter stop3.

Most background commands are not available in stop mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from stop and enter active background mode if the ENBDM bit is set. After entering background debug mode, all background commands are available.

3.6.4 LVD Enabled in Stop Mode

The LVD system is capable of generating either an interrupt or a reset when the supply voltage drops below the LVD voltage. If the LVD is enabled in stop (LVDE and LVDSE bits in SPMSC1 both set) the voltage

regulator remains active during stop mode. If the user attempts to enter stop2 with the LVD enabled for stop, the MCU will instead enter stop3.

3.6.5 Stop Modes in Low Power Run Mode

Stop2 mode cannot be entered from low power run mode. If the PPDC bit is set, then the LPR bit cannot be set. Likewise, if the LPR bit is set, the PPDC bit cannot be set.

Stop3 mode can be entered from low power run mode by executing the STOP instruction while in low power run. Exiting stop3 with a reset will put the device back into normal run mode. If LPWUI is clear, interrupts will exit stop3 mode, return the device to low power run mode, and then service the interrupt. If LPWUI is set, interrupts will exit stop3 mode, put the device into normal run mode, clear LPR and LPRS bits, and then service the interrupt.

3.7 Mode Selection

Several control signals are used to determine the current operating mode of the device. Table 3-2 shows the conditions for each of the device's operating modes.

Table 3-2. Power Mode Selections

Mode of Operation	BDCSCR BDM	SPMSC1 PMC		SPMSC2 PMC		CPU & Periph CLKs	Affects on Sub-System	
	ENBDM [1]	LVDE	LVDSE	LPR	PPDC		BDM Clock	Voltage Regulator
RUN mode	0	x	x	0	x	on. ICS in any mode.	off	on
		1	1	1				
	1	x	x	x			on	
LPRUN mode	0	0	x	1	0	low freq required. ICS in FBELP mode only.	off	standby
		1	0					
WAIT mode — (Assumes WAIT instruction executed.)	0	x	x	0	x	CPU clock is off; peripheral clocks on. ICS state same as RUN mode.	off	on
		1	1	1				
	1	x	x	x			on	
LPWAIT mode — (Assumes WAIT instruction executed.)	0	0	x	1	0	CPU clock is off; peripheral clocks at low speed. ICS in FBELP mode.	off	standby
		1	0					
STOP3 — (Assumes STOPE bit is set and STOP instruction executed.) Note that STOP3 is used in place of STOP2 if the BDM or LVD is enabled.	0	0	x	x	0	ICS in STOP. ICSERCLK ICSIRCLK, and OSCOUT optionally on[2]	off	standby
	0	1	0	x	0		off	
	0	1	1	x	x		off	on - stop currents will be increased
	1	x	x	x	x	ICSLCLK still active.	on	

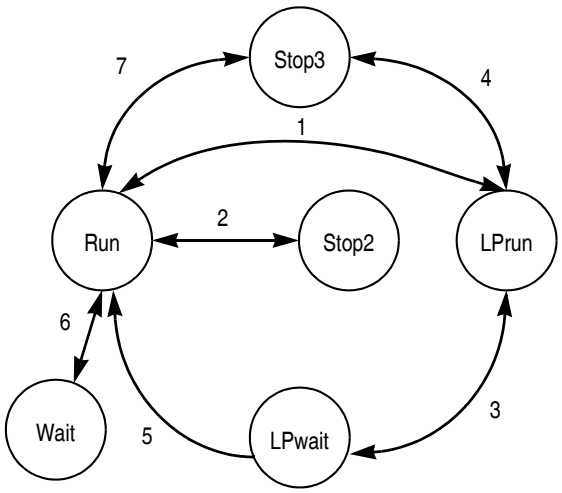
Table 3-2. Power Mode Selections (continued)

Mode of Operation	BDCSCR BDM	SPMSC1 PMC		SPMSC2 PMC		CPU & Periph CLKs	Affects on Sub-System	
	ENBDM [1]	LVDE	LVDSE	LPR	PPDC		BDM Clock	Voltage Regulator
STOP2 — (Assumes STOPE bit is set and STOP instruction executed.) If BDM or LVD is enabled, STOP3 will be invoked rather than STOP2.	0	0	x	0	1	OSCOUT optionally on[2,3]	off	partial powerdown
		1	0					

1 ENBDM is located in the BDC status and control register (BDCSCR) which is write accessible only through BDC commands.

2 Configured within the ICS module based on the settings of IREFSTEN, EFRESTEN, IRCLKEN and ERCLKEN.

3 In stop2, CPU, flash, ICS and all peripheral modules are powered down except for the TOD and LCD.



Mode	Regulator State
RUN	Full on
WAIT	Full on
LPRUN	Standby
LPWAIT	Standby
STOP3	Standby
STOP2	Partial powerdown

Figure 3-1. Allowable Power Mode Transitions for the MC9S08LH64 Series

Figure 3-1 illustrates mode state transitions allowed between the legal states shown in Table 3-1. PTB2/$\overline{\text{RESET}}$ or PTC7/IRQ must be asserted low or a TOD interrupt must occur to exit stop2. Interrupts suffice for the other stop and wait modes.

Table 3-3 defines triggers for the various state transitions shown in Figure 3-1.

Table 3-3. Triggers for Transitions Shown in Figure 3-1.

Transition #	From	To	Trigger
1	RUN	LPRUN	Configure settings shown in Table 3-1, **switch LPR=1 last**
	LPRUN	RUN	Clear LPR
			Interrupt when LPWUI=1

Table 3-3. Triggers for Transitions Shown in Figure 3-1. (continued)

Transition #	From	To	Trigger
2	RUN	STOP2	Pre-configure settings shown in Table 3-1, issue **STOP instruction**
	STOP2	RUN	**assert zero on PTB2/$\overline{\text{RESET}}$[1]or PTC7/IRQ, assert a TOD interrupt**, or reload environment from RAM
3	LPRUN	LPWAIT	**WAIT** instruction
	LPWAIT	LPRUN	**Interrupt** when LPWUI=0
4	LPRUN	STOP3	**STOP** instruction
	STOP3	LPRUN	**Interrupt** when LPWUI=0
5	LPWAIT	RUN	**Interrupt** when LPWUI=1
	RUN	LPWAIT	**NOT SUPPORTED**
6	RUN	WAIT	**WAIT instruction**
	WAIT	RUN	**Interrupt or reset**
7	STOP3	RUN	**Interrupt (**if LPR = 0, or LPR = 1 and LPWUI =1) **or reset**
	RUN	STOP3	**STOP instruction**

1 An analog connection from this pin to the on-chip regulator will wake up the regulator, which will then initiate a power-on-reset sequence.

3.7.1 On-Chip Peripheral Modules in Stop and Low Power Modes

When the MCU enters any stop mode, system clocks to the internal peripheral modules are stopped. Even in the exception case (ENBDM = 1), where clocks to the background debug logic continue to operate, clocks to the peripheral systems are halted to reduce power consumption. Refer to Section 3.6.1, "Stop2 Mode," and Section 3.6.2, "Stop3 Mode," for specific information on system behavior in stop modes.

When the MCU enters LPWait or LPRun modes, system clocks to the internal peripheral modules continue based on the settings of the clock gating control registers (SCGC1 and SCGC2).

Table 3-4. Stop and Low Power Mode Behavior

Peripheral	Mode			
	Stop2	Stop3	LPWait	LPRun
CPU	Off	Standby	Standby	On
RAM	Standby	Standby	Standby	On
FLASH	Off	Standby	Standby	On
Port I/O Registers	Off	Standby	Standby	On
ADC16	Off	Optionally On[1]	Optionally On[1]	Optionally On[1]
ACMP	Off	Optionally On[2]	Optionally On	Optionally On

Table 3-4. Stop and Low Power Mode Behavior (continued)

Peripheral	Mode			
	Stop2	Stop3	LPWait	LPRun
BDM	Off[3]	Optionally On	Off[4]	Off[4]
COP	Off	Off	Optionally On	Optionally On
ICS	Off	Optionally On[5]	On[6]	On[6]
IIC	Off	Standby	Optionally On	Optionally On
IRQ	Wake Up	Optionally On	Optionally On	Optionally On
KBI	Off	Optionally On	Optionally On	Optionally On
LVD/LVW	Off[7]	Optionally On	Off[8]	Off[8]
TOD	Optionally On	Optionally On	Optionally On[9]	Optionally On[9]
LCD	Optionally On	Optionally On	Optionally On[10]	Optionally On[10]
SCIx	Off	Standby	Optionally On	Optionally On
SPI	Off	Standby	Optionally On	Optionally On
TPMx	Off	Standby	Optionally On	Optionally On
VREFx	Optionally On	Optionally On	Optionally On	Optionally On
Voltage Regulator	Partial Powerdown	Optionally On[11]	Standby	Standby
XOSCVLP	Optionally On[12]	Optionally On[12]	Optionally On	Optionally On
I/O Pins	States Held	Peripheral Control	Peripheral Control	On

1 Requires the asynchronous ADC clock. For stop3, The VREF must be enabled to run in stop if converting the bandgap channel.

2 VREF must be enabled to run in stop if using the bandgap as a reference.

3 If ENBDM is set when entering stop2, the MCU will actually enter stop3.

4 If ENBDM is set when entering LPRun or LPWait, the MCU will actually stay in run mode or enter wait mode, respectively.

5 IRCLKEN and IREFSTEN set in ICSC1, else in standby.

6 ICS must be configured for FBELP, bus frequency limited to 125 kHz in LPRUN or LPWAIT.

7 If LVDSE is set when entering stop2, the MCU will actually enter stop3.

8 If LVDSE is set when entering LPRun or LPWait, the MCU will actually enter run or wait mode, respectively.

9 Requires the 1kHz LPO or OSCOUT clocks.

10 Requires OSCOUT or TODCLK.

11 Requires the LVD to be enabled, else in standby. See Section 3.6.4, “LVD Enabled in Stop Mode”.

12 ERCLKEN and EREFSTEN set in ICSC2, else in standby.

Chapter 4
Memory

4.1 Introduction

This chapter describes the MC9S08LH64 series on-chip memory. It details the memory map, various vector and bit assignments, registers and control bits, and other RAM and flash features.

4.2 MC9S08LH64 Series Memory Map

As shown in Figure 4-2, on-chip memory in the MC9S08LH64 series of MCUs consists of RAM, flash program memory for nonvolatile data storage, and I/O and control/status registers. The registers are divided into four groups:

- Direct-page registers (0x0000 through 0x005F)
- LCD data registers (0x1000 through 0x103B)
- High-page registers (0x1800 through 0x18A0)
- Nonvolatile registers (0xFFB0 through 0xFFBF)

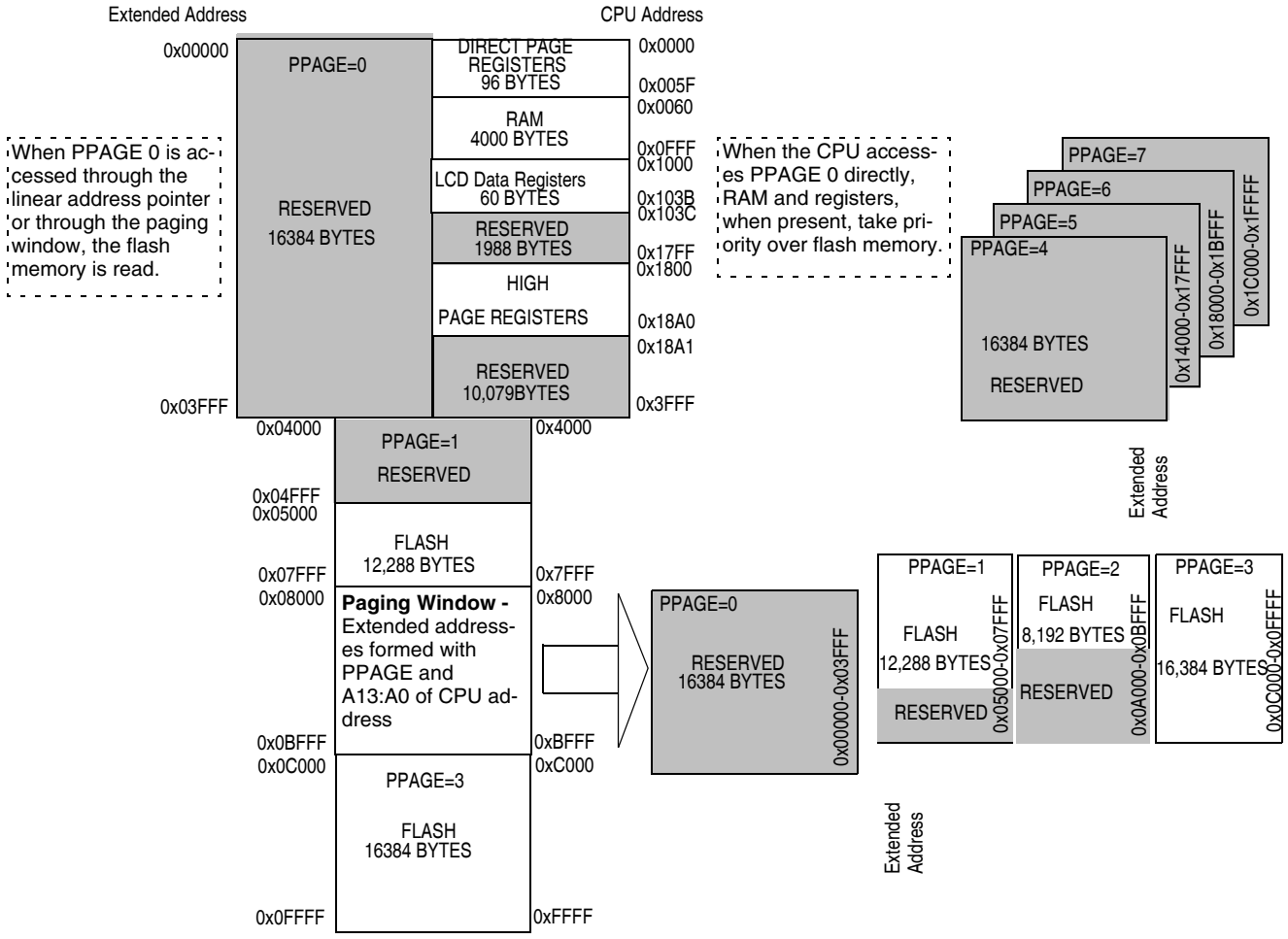


Figure 4-1. MC9S08LH36 Memory Map

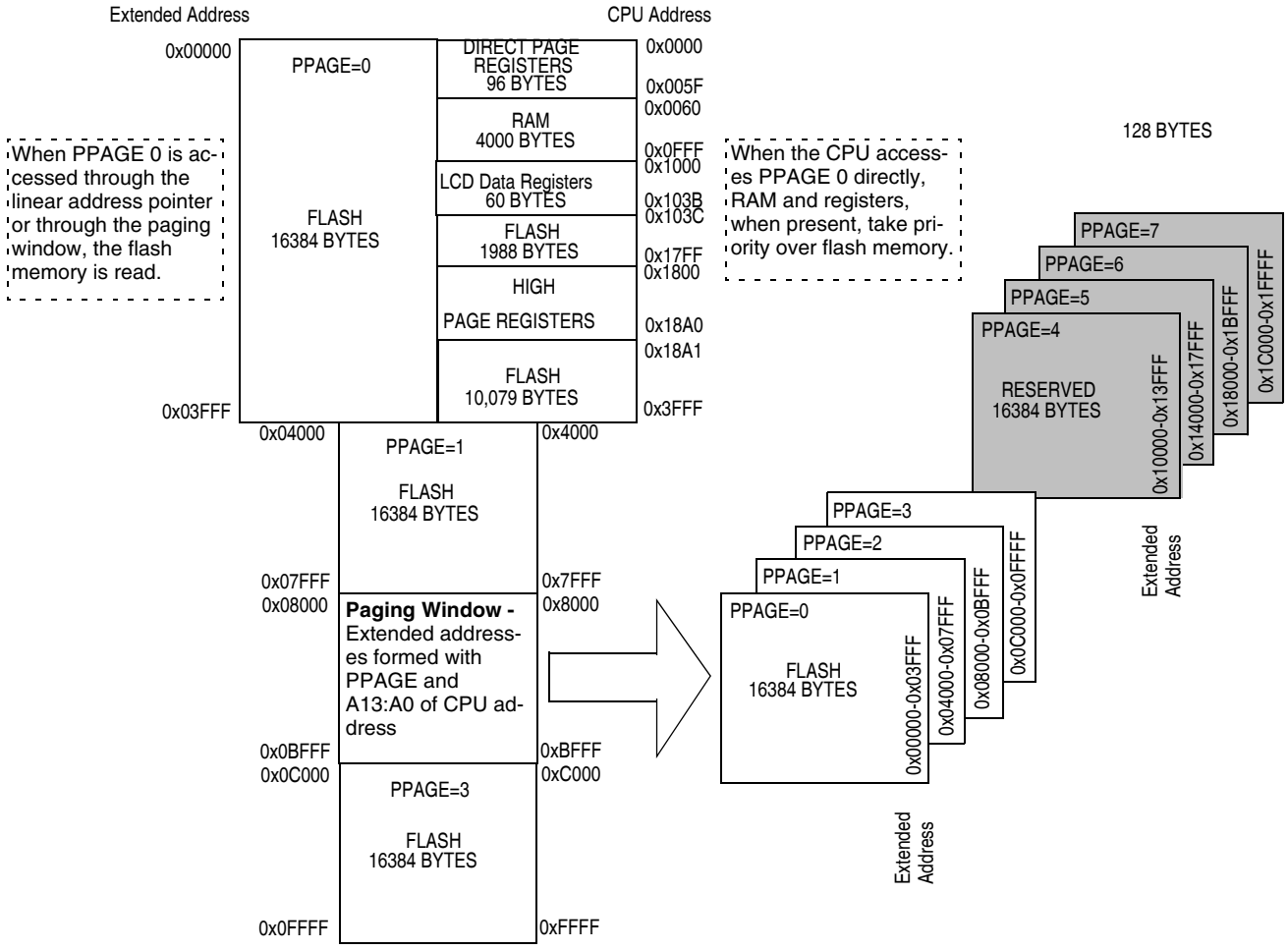


Figure 4-2. MC9S08LH64 Memory Map

4.3 Reset and Interrupt Vector Assignments

Table 4-1 shows address assignments for reset and interrupt vectors. The vector names shown in this table are the labels used in the Freescale Semiconductor provided equate file for the MC9S08LH64 series.

Table 4-1. Reset and Interrupt Vectors

Address (High/Low)	Vector	Vector Name
0xFFC0:FFC1 ⇕ 0xFFD0:FFD1	Unused Vector Space (available for user program)	
0xFFD2:0xFFD3	SCI2 Transmit	Vsci2tx
0xFFD4:0xFFD5	SCI2 Receive	Vsci2rx
0xFFD6:0xFFD7	SCI2 Error	Vsci2err

Table 4-1. Reset and Interrupt Vectors (continued)

Address (High/Low)	Vector	Vector Name
0xFFD8:0xFFD9	TOD	Vtod
0xFFDA:0xFFDB	ACMP	Vacmp
0xFFDC:0xFFDD	ADC16 Conversion	Vadc
0xFFDE:0xFFDF	KBI Interrupt	Vkeyboard
0xFFE0:0xFFE1	IIC	Viic
0xFFE2:0xFFE3	SCI1 Transmit	Vsci1tx
0xFFE4:0xFFE5	SCI1 Receive	Vsci1rx
0xFFE6:0xFFE7	SCI1 Error	Vsci1err
0xFFE8:0xFFE9	SPI	Vspi
0xFFEA:0xFFEB	LCD Frame	Vlcd
0xFFEC:0xFFED	TPM2 Overflow	Vtpm2ovf
0xFFEE:0xFFEF	TPM2 Channel 1	Vtpm2ch1
0xFFF0:0xFFF1	TPM2 Channel 0	Vtpm2ch0
0xFFF2:0xFFF3	TPM1 Overflow	Vtpm1ovf
0xFFF4:0xFFF5	TPM1 Channel 1	Vtpm1ch1
0xFFF6:0xFFF7	TPM1 Channel 0	Vtpm1ch0
0xFFF8:0xFFF9	Low Voltage Detect or Low Voltage Warning	Vlvd
0xFFFA:0xFFFB	IRQ	Virq
0xFFFC:0xFFFD	SWI	Vswi
0xFFFE:0xFFFF	Reset	Vreset

4.4 Register Addresses and Bit Assignments

The registers in the MC9S08LH64 series are divided into these groups:

- Direct-page registers are the ones located in the first 96 locations in the memory map; these are accessible with efficient direct addressing mode instructions.
- LCD data registers, LCDPENx, LCDBPENx, LCDWFx are located at the end of the RAM module.
- High-page registers are used much less often, so they are located above 0x1800 in the memory map. This leaves more room in the direct page for more frequently used registers and RAM.
- The nonvolatile register area consists of a block of 16 locations in flash memory at 0xFFB0–0xFFBF. Nonvolatile register locations include:
  - NVPROT and NVOPT are loaded into working registers at reset
  - An 8-byte backdoor comparison key that optionally allows a user to gain controlled access to secure memory

  Because the nonvolatile register locations are flash memory, they must be erased and programmed like other flash memory locations.

Direct-page registers can be accessed with efficient direct addressing mode instructions. Bit manipulation instructions can be used to access any bit in any direct-page register. Table 4-2 is a summary of all user-accessible direct-page registers and control bits.

The direct-page registers in Table 4-2 can use the more efficient direct addressing mode, which requires only the lower byte of the address. Because of this, the lower byte of the address in column one is shown in bold text. In Table 4-4 and Table 4-5, the whole address in column one is shown in bold. In Table 4-2, Table 4-4, and Table 4-5, the register names in column two are shown in bold to set them apart from the bit names to the right. Cells that are not associated with named bits are shaded. A shaded cell with a 0 indicates this unused bit always reads as a 0. Shaded cells with dashes indicate unused or reserved bit locations that could read as 1s or 0s. When writing to these bits, write a 0 unless otherwise specified.

Table 4-2. Direct-Page Register Summary (Sheet 1 of 3)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x0000	**PTAD**	PTAD7	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
0x0001	**PTADD**	PTADD7	PTADD6	PTADD5	PTADD4	PTADD3	PTADD2	PTADD1	PTADD0
0x0002	**PTBD**	PTBD7	PTBD6	PTBD5	PTBD4	—	PTBD2	PTBD1	PTBD0
0x0003	**PTBDD**	PTBDD7	PTBDD6	PTBDD5	PTBDD4	—	PTBDD2	PTBDD1	PTBDD0
0x0004	**PTCD**	PTCD7	PTCD6	PTCD5	PTCD4	PTCD3	PTCD2	PTCD1	PTCD0
0x0005	**PTCDD**	PTCDD7	PTCDD6	PTCDD5	PTCDD4	PTCDD3	PTCDD2	PTCDD1	PTCDD0
0x0006	**PTDD**	PTDD7	PTDD6	PTDD5	PTDD4	PTDD3	PTDD2	PTDD1	PTDD0
0x0007	**PTDDD**	PTDDD7	PTDDD6	PTDDD5	PTDDD4	PTDDD3	PTDDD2	PTDDD1	PTDDD0
0x0008	**PTED**	PTED7	PTED6	PTED5	PTED4	PTED3	PTED2	PTED1	PTED0
0x0009	**PTEDD**	PTEDD7	PTEDD6	PTEDD5	PTEDD4	PTEDD3	PTEDD2	PTEDD1	PTEDD0
0x000A- 0x000B	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x000C	**KBISC**	0	0	0	0	KBF	KBACK	KBIE	KBIMOD
0x000D	**KBIPE**	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
0x000E	**KBIES**	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBEDG3	KBEDG2	KBEDG1	KBEDG0
0x000F	**IRQSC**	0	IRQPDD	IRQEDG	IRQPE	IRQF	IRQACK	IRQIE	IRQMOD
0x0010	**ADCSC1A**	COCOA	AIENA	DIFFA	ADCHA				
0x0011	**ADCSC1B**	COCOB	AIENB	DIFFB	ADCHB				
0x0012	**ADCCFG1**	ADLPC	ADIV		ADLSMP	MODE		ADICLK	
0x0013	**ADCCFG2**	0	0	0	0	ADACKEN	ADHSC	ADLSTS	
0x0014	**ADCRAH**	—	—	—	—	D[11:8]			
0x0015	**ADCRAL**	D[7:0]							
0x0016	**ADCRBH**	DB[15:8]							
0x0017	**ADCRBL**	DB[7:0]							
0x0018	**LCDC0**	LCDEN	SOURCE	LCLK2	LCLK1	LCLK0	DUTY2	DUTY1	DUTY0
0x0019	**LCDC1**	LCDIEN	0	0	0	0	FCDEN	LCDWAI	LCDSTP
0x001A	**LCDSUPPLY**	CPSEL	HREFSEL	LADJ1	LADJ0	0	BBYPASS	VSUPPLY1	VSUPPLY0
0x001B	**LCDRVC**	RVEN	0	0	0	RVTRIM3	RVTRIM2	RVTRIM1	RVTRIM0
0x001C	**LCDBCTL**	BLINK	ALT	BLANK	0	BMODE	BRATE2	BRATE1	BRATE0
0x001D	**LCDS**	LCDIF	0	0	0	0	0	0	0
0x001E	Reserved	—	—	—	—	—	—	—	—
0x001F	**ACMPSC**	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD	
0x0020	**SCI1BDH**	LBKDIE	RXEDGIE	0	SBR12	SBR11	SBR10	SBR9	SBR8
0x0021	**SCI1BDL**	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
0x0022	**SCI1C1**	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
0x0023	**SCI1C2**	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
0x0024	**SCI1S1**	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
0x0025	**SCI1S2**	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
0x0026	**SCI1C3**	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE

Table 4-2. Direct-Page Register Summary (Sheet 2 of 3)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x0027	SCI1D	Bit 7	6	5	4	3	2	1	Bit 0
0x0028	SPIC1	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
0x0029	SPIC2	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
0x002A	SPIBR	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
0x002B	SPIS	SPRF	0	SPTEF	MODF	0	0	0	0
0x002C	Reserved	0	0	0	0	0	0	0	0
0x002D	SPID	Bit 7	6	5	4	3	2	1	Bit 0
0x002E-0x002F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x0030	PPAGE	0	0	0	0	0	0	XA15	XA14
0x0031	Reserved	—	—	—	—	—	—	—	—
0x0032	LAP1	LA15	LA14	LA13	LA12	LA11	LA10	LA9	LA8
0x0033	LAP0	LA7	LA6	LA5	LA4	LA3	LA2	LA1	LA0
0x0034	LWP	D7	D6	D5	D4	D3	D2	D1	D0
0x0035	LBP	D7	D6	D5	D4	D3	D2	D1	D0
0x0036	LB	D7	D6	D5	D4	D3	D2	D1	D0
0x0037	LAPAB	D7	D6	D5	D4	D3	D2	D1	D0
0x0038	ICSC1	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
0x0039	ICSC2	BDIV		RANGE	HGO	LP	EREFS	ERCLKEN	EREFSTEN
0x003A	ICSTRM	TRIM							
0x003B	ICSSC	DRST		DMX32	IREFST	CLKST		OSCINIT	FTRIM
0x003C-0x003F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x0040	TPM1SC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
0x0041	TPM1CNTH	Bit 15	14	13	12	11	10	9	Bit 8
0x0042	TPM1CNTL	Bit 7	6	5	4	3	2	1	Bit 0
0x0043	TPM1MODH	Bit 15	14	13	12	11	10	9	Bit 8
0x0044	TPM1MODL	Bit 7	6	5	4	3	2	1	Bit 0
0x0045	TPM1C0SC	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
0x0046	TPM1C0VH	Bit 15	14	13	12	11	10	9	Bit 8
0x0047	TPM1C0VL	Bit 7	6	5	4	3	2	1	Bit 0
0x0048	TPM1C1SC	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0
0x0049	TPM1C1VH	Bit 15	14	13	12	11	10	9	Bit 8
0x004A	TPM1C1VL	Bit 7	6	5	4	3	2	1	Bit 0
0x004B-0x004F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x0050	TPM2SC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
0x0051	TPM2CNTH	Bit 15	14	13	12	11	10	9	Bit 8
0x0052	TPM2CNTL	Bit 7	6	5	4	3	2	1	Bit 0

Table 4-2. Direct-Page Register Summary (Sheet 3 of 3)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x0053	**TPM2MODH**	Bit 15	14	13	12	11	10	9	Bit 8
0x0054	**TPM2MODL**	Bit 7	6	5	4	3	2	1	Bit 0
0x0055	**TPM2C0SC**	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
0x0056	**TPM2C0VH**	Bit 15	14	13	12	11	10	9	Bit 8
0x0057	**TPM2C0VL**	Bit 7	6	5	4	3	2	1	Bit 0
0x0058	**TPM2C1SC**	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0
0x0059	**TPM2C1VH**	Bit 15	14	13	12	11	10	9	Bit 8
0x005A	**TPM2C1VL**	Bit 7	6	5	4	3	2	1	Bit 0
0x005B	Reserved	—	—	—	—	—	—	—	—
0x005C	**VREF1TRM**	TRM							
0x005D	**VREF1SC**	VREFEN	0	0	0	0	VREFST	MODE	
0x005E	**VREF2TRM**	TRM							
0x005F	**VREF2SC**	VREFEN	0	0	0	0	VREFST	MODE	

Use the LCD registers shown in table below to enable LCD functionality and display the LCD data.

Table 4-3. LCD Registers (Sheet 1 of 2)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x1000	**LCDPEN0**	PEN7	PEN6	PEN5	PEN4	PEN3	PEN2	PEN1	PEN0
0x1001	**LCDPEN1**	PEN15	PEN14	PEN13	PEN12	PEN11	PEN10	PEN9	PEN8
0x1002	**LCDPEN2**	PEN23	PEN22	PEN21	PEN20	PEN19	PEN18	PEN17	PEN16
0x1003	**LCDPEN3**	PEN31	PEN30	PEN29	PEN28	PEN27	PEN26	PEN25	PEN24
0x1004	**LCDPEN4**	PEN39	PEN38	PEN37	PEN36	PEN35	PEN34	PEN33	PEN32
0x1005	**LCDPEN5**	—	—	—	—	PEN43	PEN42	PEN41	PEN40
0x1006-0x1007	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x1008	**LCDBPEN0**	BPEN7	BPEN6	BPEN5	BPEN4	BPEN3	BPEN2	BPEN1	BPEN0
0x1009	**LCDBPEN1**	BPEN15	BPEN14	BPEN13	BPEN12	BPEN11	BPEN10	BPEN9	BPEN8
0x100A	**LCDBPEN2**	BPEN23	BPEN22	BPEN21	BPEN20	BPEN19	BPEN18	BPEN17	BPEN16
0x100B	**LCDBPEN3**	BPEN31	BPEN30	BPEN29	BPEN28	BPEN27	BPEN26	BPEN25	BPEN24
0x100C	**LCDBPEN4**	BPEN39	BPEN38	BPEN37	BPEN36	BPEN35	BPEN34	BPEN33	BPEN32
0x100D	**LCDBPEN5**	—	—	—	—	BPEN43	BPEN42	BPEN41	BPEN40
0x100E-0x100F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x1010	**LCDWF0**	BPHLCD0	BPGLCD0	BPFLCD0	BPELCD0	BPDLCD0	BPCLCD0	BPBLCD0	BPALCD0
0x1011	**LCDWF1**	BPHLCD1	BPGLCD1	BPFLCD1	BPELCD1	BPDLCD1	BPCLCD1	BPBLCD1	BPALCD1
0x1012	**LCDWF2**	BPHLCD2	BPGLCD2	BPFLCD2	BPELCD2	BPDLCD2	BPCLCD2	BPBLCD2	BPALCD2
0x1013	**LCDWF3**	BPHLCD3	BPGLCD3	BPFLCD3	BPELCD3	BPDLCD3	BPCLCD3	BPBLCD3	BPALCD3
0x1014	**LCDWF4**	BPHLCD4	BPGLCD4	BPFLCD4	BPELCD4	BPDLCD4	BPCLCD4	BPBLCD4	BPALCD4

Table 4-3. LCD Registers (Sheet 2 of 2)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x1015	LCDWF5	BPHLCD5	BPGLCD5	BPFLCD5	BPELCD5	BPDLCD5	BPCLCD5	BPBLCD5	BPALCD5
0x1016	LCDWF6	BPHLCD6	BPGLCD6	BPFLCD6	BPELCD6	BPDLCD6	BPCLCD6	BPBLCD6	BPALCD6
0x1017	LCDWF7	BPHLCD7	BPGLCD7	BPFLCD7	BPELCD7	BPDLCD7	BPCLCD7	BPBLCD7	BPALCD7
0x1018	LCDWF8	BPHLCD8	BPGLCD8	BPFLCD8	BPELCD8	BPDLCD8	BPCLCD8	BPBLCD8	BPALCD8
0x1019	LCDWF9	BPHLCD9	BPGLCD9	BPFLCD9	BPELCD9	BPDLCD9	BPCLCD9	BPBLCD9	BPALCD9
0x101A	LCDWF10	BPHLCD10	BPGLCD10	BPFLCD10	BPELCD10	BPDLCD10	BPCLCD10	BPBLCD10	BPALCD10
0x101B	LCDWF11	BPHLCD11	BPGLCD11	BPFLCD11	BPELCD11	BPDLCD11	BPCLCD11	BPBLCD11	BPALCD11
0x101C	LCDWF12	BPHLCD12	BPGLCD12	BPFLCD12	BPELCD12	BPDLCD12	BPCLCD12	BPBLCD12	BPALCD12
0x101D	LCDWF13	BPHLCD13	BPGLCD13	BPFLCD13	BPELCD13	BPDLCD13	BPCLCD13	BPBLCD13	BPALCD13
0x101E	LCDWF14	BPHLCD14	BPGLCD14	BPFLCD14	BPELCD14	BPDLCD14	BPCLCD14	BPBLCD14	BPALCD14
0x101F	LCDWF15	BPHLCD15	BPGLCD15	BPFLCD15	BPELCD15	BPDLCD15	BPCLCD15	BPBLCD15	BPALCD15
0x1020	LCDWF16	BPHLCD16	BPGLCD16	BPFLCD16	BPELCD16	BPDLCD16	BPCLCD16	BPBLCD16	BPALCD16
0x1021	LCDWF17	BPHLCD17	BPGLCD17	BPFLCD17	BPELCD17	BPDLCD17	BPCLCD17	BPBLCD17	BPALCD17
0x1022	LCDWF18	BPHLCD18	BPGLCD18	BPFLCD18	BPELCD18	BPDLCD18	BPCLCD18	BPBLCD18	BPALCD18
0x1023	LCDWF19	BPHLCD19	BPGLCD19	BPFLCD19	BPELCD19	BPDLCD19	BPCLCD19	BPBLCD19	BPALCD19
0x1024	LCDWF20	BPHLCD20	BPGLCD20	BPFLCD20	BPELCD20	BPDLCD20	BPCLCD20	BPBLCD20	BPALCD20
0x1025	LCDWF21	BPHLCD21	BPGLCD21	BPFLCD21	BPELCD21	BPDLCD21	BPCLCD21	BPBLCD21	BPALCD21
0x1026	LCDWF22	BPHLCD22	BPGLCD22	BPFLCD22	BPELCD22	BPDLCD22	BPCLCD22	BPBLCD22	BPALCD22
0x1027	LCDWF23	BPHLCD23	BPGLCD23	BPFLCD23	BPELCD23	BPDLCD23	BPCLCD23	BPBLCD23	BPALCD23
0x1028	LCDWF24	BPHLCD24	BPGLCD24	BPFLCD24	BPELCD24	BPDLCD24	BPCLCD24	BPBLCD24	BPALCD24
0x1029	LCDWF25	BPHLCD25	BPGLCD25	BPFLCD25	BPELCD25	BPDLCD25	BPCLCD25	BPBLCD25	BPALCD25
0x102A	LCDWF26	BPHLCD26	BPGLCD26	BPFLCD26	BPELCD26	BPDLCD26	BPCLCD26	BPBLCD26	BPALCD26
0x102B	LCDWF27	BPHLCD27	BPGLCD27	BPFLCD27	BPELCD27	BPDLCD27	BPCLCD27	BPBLCD27	BPALCD27
0x102C	LCDWF28	BPHLCD28	BPGLCD28	BPFLCD28	BPELCD28	BPDLCD28	BPCLCD28	BPBLCD28	BPALCD28
0x102D	LCDWF29	BPHLCD29	BPGLCD29	BPFLCD29	BPELCD29	BPDLCD29	BPCLCD29	BPBLCD29	BPALCD29
0x102E	LCDWF30	BPHLCD30	BPGLCD30	BPFLCD30	BPELCD30	BPDLCD30	BPCLCD30	BPBLCD30	BPALCD30
0x102F	LCDWF31	BPHLCD31	BPGLCD31	BPFLCD31	BPELCD31	BPDLCD31	BPCLCD31	BPBLCD31	BPALCD31
0x1030	LCDWF32	BPHLCD32	BPGLCD32	BPFLCD32	BPELCD32	BPDLCD32	BPCLCD32	BPBLCD32	BPALCD32
0x1031	LCDWF33	BPHLCD33	BPGLCD33	BPFLCD33	BPELCD33	BPDLCD33	BPCLCD33	BPBLCD33	BPALCD33
0x1032	LCDWF34	BPHLCD34	BPGLCD34	BPFLCD34	BPELCD34	BPDLCD34	BPCLCD34	BPBLCD34	BPALCD34
0x1033	LCDWF35	BPHLCD35	BPGLCD35	BPFLCD35	BPELCD35	BPDLCD35	BPCLCD35	BPBLCD35	BPALCD35
0x1034	LCDWF36	BPHLCD36	BPGLCD36	BPFLCD36	BPELCD36	BPDLCD36	BPCLCD36	BPBLCD36	BPALCD36
0x1035	LCDWF37	BPHLCD37	BPGLCD37	BPFLCD37	BPELCD37	BPDLCD37	BPCLCD37	BPBLCD37	BPALCD37
0x1036	LCDWF38	BPHLCD38	BPGLCD38	BPFLCD38	BPELCD38	BPDLCD38	BPCLCD38	BPBLCD38	BPALCD38
0x1037	LCDWF39	BPHLCD39	BPGLCD39	BPFLCD39	BPELCD39	BPDLCD39	BPCLCD39	BPBLCD39	BPALCD39
0x1038	LCDWF40	BPHLCD40	BPGLCD40	BPFLCD40	BPELCD40	BPDLCD40	BPCLCD40	BPBLCD40	BPALCD40
0x1039	LCDWF41	BPHLCD41	BPGLCD41	BPFLCD41	BPELCD41	BPDLCD41	BPCLCD41	BPBLCD41	BPALCD41
0x103A	LCDWF42	BPHLCD42	BPGLCD42	BPFLCD42	BPELCD42	BPDLCD42	BPCLCD42	BPBLCD42	BPALCD42
0x103B	LCDWF43	BPHLCD43	BPGLCD43	BPFLCD43	BPELCD43	BPDLCD43	BPCLCD43	BPBLCD43	BPALCD43

High-page registers, shown in Table 4-4, are accessed much less often than other I/O and control registers so they have been located outside the direct addressable memory space, starting at 0x1800.

Table 4-4. High-Page Register Summary (Sheet 1 of 3)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x1800	SRS	POR	PIN	COP	ILOP	ILAD	0	LVD	0
0x1801	SBDFR	0	0	0	0	0	0	0	BDFR
0x1802	SOPT1	COPE	COPT	STOPE	0	0	0	BKGDPE	RSTPE
0x1803	SOPT2	COPCLKS	0	0	0	ADCTRS	SPIPS	IICPS	ACIC
0x1804 – 0x1805	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x1806	SDIDH	—	—	—	—	ID11	ID10	ID9	ID8
0x1807	SDIDL	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
0x1808	SPMSC1	LVDF	LVDACK	LVDIE	LVDRE	LVDSE	LVDE	BGBDS	BGBE
0x1809	SPMSC2	LPR	LPRS	LPWUI	0	PPDF	PPDACK	PPDE	PPDC
0x180A	Reserved	—	—	—	—	—	—	—	—
0x180B	SPMSC3	LVWF	LVWACK	LVDV	LVWV	LVWIE	0	0	0
0x180C	Reserved	—	—	—	—	—	—	—	—
0x180D	Reserved	—	—	—	—	—	—	—	—
0x180E	SCGC1	VREF2	TPM2	TPM1	ADC	VREF1	IIC	SCI2	SCI1
0x180F	SCGC2	DBG	FLS	IRQ	KBI	ACMP	TOD	LCD	SPI
0x1810	DBGCAH	Bit 15	14	13	12	11	10	9	Bit 8
0x1811	DBGCAL	Bit 7	6	5	4	3	2	1	Bit 0
0x1812	DBGCBH	Bit 15	14	13	12	11	10	9	Bit 8
0x1813	DBGCBL	Bit 7	6	5	4	3	2	1	Bit 0
0x1814	DBGCCH	Bit 15	14	13	12	11	10	9	Bit 8
0x1815	DBGCCL	Bit 7	6	5	4	3	2	1	Bit 0
0x1816	DBGFH	Bit 15	14	13	12	11	10	9	Bit 8
0x1817	DBGFL	Bit 7	6	5	4	3	2	1	Bit 0
0x1818	DBGCAX	RWAEN	RWA	PAGSEL	0	0	0	0	bit-16
0x1819	DBGCBX	RWBEN	RWB	PAGSEL	0	0	0	0	bit-16
0x181A	DBGCCX	RWCEN	RWC	PAGSEL	0	0	0	0	bit-16
0x181B	DBGFX	PPACC	0	0	0	0	0	0	0
0x181C	DBGC	DBGEN	ARM	TAG	BRKEN	0	0	0	LOOP1
0x181D	DBGT	TRGSEL	BEGIN	0	0	TRG			
0x181E	DBGS	AF	BF	CF	0	0	0	0	ARMF
0x181F	DBGCNT	0	0	0	0	CNT			
0x1820	FCDIV	DIVLD	PRDIV8	DIV					
0x1821	FOPT	KEYEN	FNORED	0	0	0	0	SEC	
0x1822	Reserved	—	—	—	—	—	—	—	—
0x1823	FCNFG	0	0	KEYACC	0	0	0	0	0
0x1824	FPROT	FPS							FPDIS

Table 4-4. High-Page Register Summary (Sheet 2 of 3)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x1825	FSTAT	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
0x1826	FCMD	FCMD							
0x1827-0x182F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x1830	TODC	TODEN	TODCLKS		TODR	TODCLKEN	TODPS		
0x1831	TODSC	QSECF	SECF	MTCHF	QSECIE	SECIE	MTCHIE	MTCHEN	MTCHWC
0x1832	TODM	TODM						MQSEC	
0x1833	TODCNT	TODCNT							
0x1834-0x183F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x1840	PTAPE	PTAPE7	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
0x1841	PTASE	PTASE7	PTASE6	PTASE5	PTASE4	PTASE3	PTASE2	PTASE1	PTASE0
0x1842	PTADS	PTADS7	PTADS6	PTADS5	PTADS4	PTADS3	PTADS2	PTADS1	PTADS0
0x1843	Reserved	—	—	—	—	—	—	—	—
0x1844	PTBPE	PTBPE7	PTBPE6	PTBPE5	PTBPE4	—	PTBPE2	PTBPE1	PTBPE0
0x1845	PTBSE	PTBSE7	PTBSE6	PTBSE5	PTBSE4	—	PTBSE2	PTBSE1	PTBSE0
0x1846	PTBDS	PTBDS7	PTBDS6	PTBDS5	PTBDS4	—	PTBDS2	PTBDS1	PTBDS0
0x1847	Reserved	—	—	—	—	—	—	—	—
0x1848	PTCPE	PTCPE7	PTCPE6	PTCPE5	PTCPE4	PTCPE3	PTCPE2	PTCPE1	PTCPE0
0x1849	PTCSE	PTCSE7	PTCSE6	PTCSE5	PTCSE4	PTCSE3	PTCSE2	PTCSE1	PTCSE0
0x184A	PTCDS	PTCDS7	PTCDS6	PTCDS5	PTCDS4	PTCDS3	PTCDS2	PTCDS1	PTCDS0
0x184B	Reserved	—	—	—	—	—	—	—	—
0x184C	PTDPE	PTDPE7	PTDPE6	PTDPE5	PTDPE4	PTDPE3	PTDPE2	PTDPE1	PTDPE0
0x184D	PTDSE	PTDSE7	PTDSE6	PTDSE5	PTDSE4	PTDSE3	PTDSE2	PTDSE1	PTDSE0
0x184E	PTDDS	PTDDS7	PTDDS6	PTDDS5	PTDDS4	PTDDS3	PTDDS2	PTDDS1	PTDDS0
0x184F	Reserved	—	—	—	—	—	—	—	—
0x1850	PTEPE	PTEPE7	PTDPE6	PTDPE5	PTDPE4	PTDPE3	PTDPE2	PTDPE1	PTDPE0
0x1851	PTESE	PTESE7	PTDSE6	PTDSE5	PTDSE4	PTDSE3	PTDSE2	PTDSE1	PTDSE0
0x1852	PTEDS	PTEDS7	PTDDS6	PTDDS5	PTDDS4	PTDDS3	PTDDS2	PTDDS1	PTDDS0
0x1853-0x1857	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x1858	SCI2BDH	LBKDIE	RXEDGIE	0	SBR12	SBR11	SBR10	SBR9	SBR8
0x1859	SCI2BDL	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
0x185A	SCI2C1	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
0x185B	SCI2C2	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
0x185C	SCI2S1	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
0x185D	SCI2S2	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
0x185E	SCI2C3	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
0x185F	SCI2D	Bit 7	6	5	4	3	2	1	Bit 0
0x1860	IICA	AD7	AD6	AD5	AD4	AD3	AD2	AD1	0

Table 4-4. High-Page Register Summary (Sheet 3 of 3)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x1861	IICF	MULT		ICR					
0x1862	IICC1	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
0x1863	IICS	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
0x1864	IICD	DATA							
0x1865	IICC2	GCAEN	ADEXT	0	0	0	AD10	AD9	AD8
0x1880	ADCCVH	0	0	0	0	CV[11:8]			
0x1881	ADCCVL	CV[7:0]							
0x1882	ADCCV2H	CV2[15:8]							
0x1883	ADCCV2L	CV2[7:0]							
0x1884	ADCSC2	ADACT	ADTRG	ACFE	ACFGT	ACREN	0	REFSEL	
0x1885	ADCSC3	CAL	CALF	0	0	ADCO	AVGE	AVGS	
0x1886	ADCOFSH	OFS[15:8]							
0x1887	ADCOFSL	OFS[7:0]							
0x1888	ADCPGH	PG[15:8]							
0x1889	ADCPGL	PG[7:0]							
0x188A	ADCMGH	MG[15:8]							
0x188B	ADCMGL	MG[7:0]							
0x188C	ADCCLPD	0	0	CLPD					
0x188D	ADCCLPS	0	0	CLPS					
0x188E	ADCCLP4H	0	0	0	0	0	0	CLP4[9:8]	
0x188F	ADCCLP4L	CLP4[7:0]							
0x1890	ADCCLP3H	0	0	0	0	0	0	0	CLP3[8]
0x1891	ADCCLP3L	CLP3[7:0]							
0x1892	ADCCLP2	CLP2							
0x1893	ADCCLP1	0	CLP1						
0x1894	ADCCLP0	0	0	CLP0					
0x1895	Reserved	—	—	—	—	—	—	—	—
0x1896	ADCCLMD	0	0	CLMD					
0x1897	ADCCLMS	0	0	CLMS					
0x1898	ADCCLM4H	0	0	0	0	0	0	CLM4[9:8]	
0x1899	ADCCLM4L	CLM4[7:0]							
0x189A	ADCCLM3H	0	0	0	0	0	0	0	CLM3[8]
0x189B	ADCCLM3L	CLM3[7:0]							
0x189C	ADCCLM2	CLM2							
0x189D	ADCCLM1	0	CLM1						
0x189E	ADCCLM0	0	0	CLM0					
0x189F	APCTL1	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
0x18A0	APCTL2	ADPC15	ADPC14	ADPC13	ADPC12	ADPC11	ADPC10	ADPC9	ADPC8

Several reserved flash memory locations, shown in Table 4-5, are used for storing values used by several registers. These registers include an 8-byte backdoor key, NVBACKKEY, which can be used to gain access to secure memory resources. During reset events, the contents of NVPROT and NVOPT in the reserved flash memory are transferred into corresponding FPROT and FOPT registers in the high-page registers area to control security and block protection options.

The factory ICS trim value is stored in the IFR and will be loaded into the ICSTRM and ICSSC registers after any reset. The internal reference trim values stored in flash, TRIM and FTRIM, can be programmed by third party programmers and must be copied into the corresponding ICS registers by user code to override the factory trim.

Table 4-5. Reserved Flash Memory Addresses

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0xFFAC	Reserved for Storage of VREF1 TRM	VREF1 TRIM							
0xFFAD	Reserved for Storage of VREF2 TRM	VREF2 TRIM							
0xFFAE	Reserved for Storage of FTRIM	0	0	0	0	0	0	0	FTRIM
0xFFAF	Reserved for Storage of ICSTRM	TRIM							
0xFFB0 – 0xFFB7	NVBACKKEY	8-Byte Comparison Key							
0xFFB8 – 0xFFBC	Reserved	—	—	—	—	—	—	—	—
0xFFBD	NVPROT	FPS							FPDIS
0xFFBE	Reserved	—	—	—	—	—	—	—	—
0xFFBF	NVOPT	KEYEN	FNORED	0	0	0	0	SEC	

Provided the key enable (KEYEN) bit is 1, the 8-byte comparison key can be used to temporarily disengage memory security. This key mechanism can be accessed only through user code running in secure memory. (A security key cannot be entered directly through background debug commands.) This security key can be disabled completely by programming the KEYEN bit to 0. If the security key is disabled, the only way to disengage security is by mass erasing the flash if needed (normally through the background debug interface) and verifying that flash is blank. To avoid returning to secure mode after the next reset, program the security bits (SEC) to the unsecured state (1:0).

4.5 Memory Management Unit

The memory management unit (MMU) allows the program and data space for the HCS08 family of microcontrollers to be extended beyond the 64 KB CPU addressable memory map. The MMU uses a paging scheme similar to that seen on other MCU architectures, such as HCS12. The extended memory when used for data can also be accessed linearly using a linear address pointer and data access registers.

4.5.1 Features

Key features of the MMU module are:

- Memory Management Unit extends the HCS08 memory space
  - — up to 4 MB for program and data space (Maximum)
- Extended program space using paging scheme
  - — PPAGE register used for page selection
  - — fixed 16 KB memory window
  - — architecture supports up to 256 pages at 16 K each
- Extended data space using linear address pointer
  - — up to 22-bit linear address pointer (Maximum)
  - — linear address pointer and data register provided in direct page allows access of complete flash memory map using direct page instructions
  - — optional auto increment of pointer when data accessed
  - — supports an 2s compliment addition/subtraction to address pointer without using any math instructions or memory resources
  - — supports word accesses to any address specified by the linear address pointer when using LDHX, STHX instructions

4.5.2 Register Definition

4.5.2.1 Program Page Register (PPAGE)

The HCS08 core architecture limits the CPU addressable space available to 64 KB. The address space can be extended to 128 KB using a paging window scheme. The program page (PPAGE) allows for selecting one of the 16 KB blocks to be accessed through the program page window located at 0x8000-0xBFFF. The CALL and RTC instructions can load or store the value of PPAGE onto or from the stack during program execution. After any reset, PPAGE is set to PAGE 2.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	XA15	XA14
W								
Reset:	0	0	0	0	0	0	1	0

Figure 4-3. Program Page Register (PPAGE)

Table 4-6. Program Page Register Field Descriptions

Field	Description
1:0 XA15:XA14	When the CPU addresses the paging window, 0x8000-0xBFFF, the value in the PPAGE register along with the CPU addresses A13:A0 are used to create a 17-bit extended address.

4.5.2.2 Linear Address Pointer Registers 1:0 (LAP1:LAP0)

The two registers, LAP1:LAP0 contain the 16-bit linear address that allows the user to access any flash location in the extended address map. This register is used in conjunction with the data registers, linear byte (LB), linear byte post increment (LBP) and linear word post increment (LWP). The contents of LAP1:LAP0 will auto-increment when accessing data using the LBP and LWP registers. The contents of LAP1:LAP0 can be increased by writing an 8-bit value to LAPAB.

R W	LA15	LA14	LA13	LA12	LA11	LA10	LA9	LA8
R W	LA7	LA6	LA5	LA4	LA3	LA2	LA1	LA0
Reset:	0	0	0	0	0	0	0	0

Figure 4-4. Linear Address Pointer Registers 1:0 (LAP1:LAP0)

Table 4-7. Linear Address Pointer Registers 1:0 Field Descriptions

Field	Description
15:0 LA15:LA0	The values in LAP1:LAP0 are used to create a 16-bit linear address pointer. The value in these registers are used as the extended address when accessing any of the data registers LB, LBP and LWP.

4.5.2.3 Linear Word Post Increment Register (LWP)

This register is one of three data registers that the user can use to access any flash memory location in the extended address map. When LWP is accessed the contents of LAP1:LAP0 make up the extended address of the flash memory location to be addressed. When accessing data using LWP, the contents of LAP1:LAP0 will increment after the read or write is complete.

Accessing LWP does the same thing as accessing LBP. The MMU register ordering of LWP followed by LBP, allow the user to access data by words using the LDHX or STHX instructions of the LWP register.

	7	6	5	4	3	2	1	0
R W	D7	D6	D5	D4	D3	D2	D1	D0
Reset:	0	0	0	0	0	0	0	0

Figure 4-5. Linear Word Post Increment Register (LWP)

Table 4-8. Linear Word Post Increment Register Field Descriptions

Field	Description
7:0 D7:D0	Reads of this register will first return the data value pointed to by the linear address pointer, LAP1:LAP0 and then will increment LAP1:LAP0. Writes to this register will first write the data value to the memory location specified by the linear address pointer and then will increment LAP1:LAP0. Writes to this register are most commonly used when writing to the flash block(s) during programming.

4.5.2.4 Linear Byte Post Increment Register (LBP)

This register is one of three data registers that the user can use to access any flash memory location in the extended address map. When LBP is accessed the contents of LAP1:LAP0 make up the extended address of the flash memory location to be addressed. When accessing data using LBP, the contents of LAP1:LAP0 will increment after the read or write is complete.

Accessing LBP does the same thing as accessing LWP. The MMU register ordering of LWP followed by LBP, allow the user to access data by words using the LDHX or STHX instructions with the address of the LWP register.

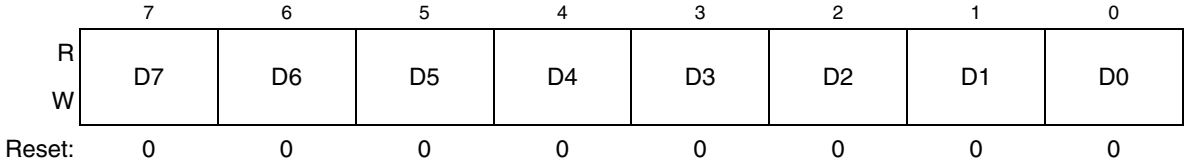


Figure 4-6. Linear Byte Post Increment Register (LBP)

Table 4-9. Linear Byte Post Increment Register Field Descriptions

Field	Description
7:0 D7:D0	Reads of this register will first return the data value pointed to by the linear address pointer, LAP1:LAP0 and then will increment LAP1:LAP0. Writes to this register will first write the data value to the memory location specified by the linear address pointer and then will increment LAP1:LAP0. Writes to this register are most commonly used when writing to the flash block(s) during programming.

4.5.2.5 Linear Byte Register (LB)

This register is one of three data registers that the user can use to access any flash memory location in the extended address map. When LB is accessed the contents of LAP1:LAP0 make up the extended address of the flash memory location to be addressed.

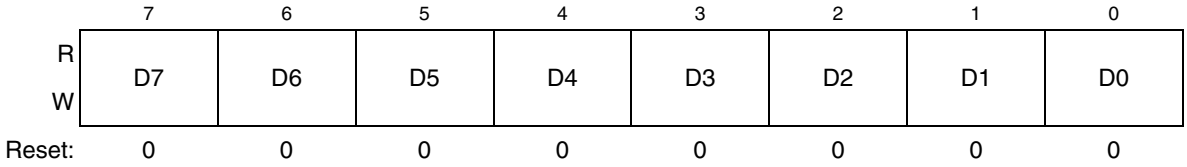


Figure 4-7. Linear Byte Register (LB)

Table 4-10. Linear Data Register Field Descriptions

Field	Description
7:0 D7:D0	Reads of this register returns the data value pointed to by the linear address pointer, LAP1:LAP0. Writes to this register will write the data value to the memory location specified by the linear address pointer. Writes to this register are most commonly used when writing to the flash block(s) during programming.

4.5.2.6 Linear Address Pointer Add Byte Register (LAPAB)

The user can increase or decrease the contents of LAP1:LAP0 by writing a 2s compliment value to LAPAB. The value written will be added to the current contents of LAP1:LAP0.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	D7	D6	D5	D4	D3	D2	D1	D0
Reset:	0	0	0	0	0	0	0	0

Figure 4-8. Linear Address Pointer Add Byte Register (LAPAB)

Table 4-11. Linear Address Pointer Add Byte Register Field Descriptions

Field	Description
7:0 D7:D0	The 2s compliment value written to LAPAB will be added to contents of the linear address pointer register, LAP1:LAP0. Writing a value of 0x7f to LAPAB will increase LAP by 127, a value of 0xff will decrease LAP by 1, and a value of 0x80 will decrease LAP by 128.

4.5.3 Functional Description

4.5.3.1 Memory Expansion

The HCS08 core architecture limits the CPU addressable space available to 64 KB. The program page (PPAGE) allows for integrating up to 128 KB of flash into the system by selecting one of the 16 KB blocks to be accessed through the paging window located at 0x8000-0xBFFF. The MMU module also provides a linear address pointer that allows extension of data access up to 128 KB.

4.5.3.1.1 Program Space

The PPAGE register holds the page select value for the paging window. The value in PPAGE can be manipulated by using normal read and write instructions as well as the CALL and RTC instructions. The user must not change PPAGE directly when running from paged memory, only CALL and RTC are used.

When the MMU detects that the CPU is addressing the paging window, the value currently in PPAGE will be used to create an extended address that the MCU's decode logic will use to select the desired flash location.

As seen in Figure 4-3, the flash blocks in the CPU addressable memory can be accessed directly or using the paging window and PPAGE register. For example, the flash from location 0x4000-0x7FFF can be accessed directly or using the paging window, PPAGE = 1, address 0x8000-0xBFFF.

4.5.3.1.2 CALL and RTC (Return from Call) Instructions

CALL and RTC are instructions that perform automated page switching when executed in the user program. CALL is similar to a JSR instruction, but the subroutine that is called can be located anywhere in the normal 64 KB address space or on any page of program memory.

During the execution of a CALL instruction, the CPU:

- Stacks the return address.
- Pushes the current PPAGE value onto the stack.
- Writes the new instruction-supplied PPAGE value into the PPAGE register.
- Transfers control to the subroutine of the new instruction-supplied address.

This sequence is not interruptible; there is no need to inhibit interrupts during CALL execution. A CALL can be executed from any address in memory to any other address.

The new PPAGE value is provided by an immediate operand in the instruction along with the address within the paging window, 0x8000-0xBFFF.

RTC is similar to an RTS instruction.

The RTC instruction terminates subroutines invoked by a CALL instruction.

During the execution of an RTC instruction, the CPU:

- Pulls the old PPAGE value from the stack and loads it into the PPAGE register
- Pulls the 16-bit return address from the stack and loads it into the PC
- Resumes execution at the return address

This sequence is not interruptible; there is no need to inhibit interrupts during RTC execution. An RTC can be executed from any address in memory.

4.5.3.1.3 Data Space

The linear address pointer registers, LAP1:LAP0 along with the linear data register allow the CPU to read or write any address in the extended flash memory space. This linear address pointer may be used to access data from any memory location while executing code from any location in extended memory, including accessing data from a different PPAGE than the currently executing program.

To access data using the linear address pointer, the user would first setup the extended address in the 17-bit address pointer, LAP1:LAP0. Accessing one of the three linear data registers LB, LBP and LWP will access the extended memory location specified by LAP1:LAP0. The three linear data registers access the memory locations in the same way, however the LBP and LWP will also increment LAP1:LAP0. Accessing either the LBP or LWP registers allows a user program to read successive memory locations without re-writing the linear address pointer. Accessing LBP or LWP does the exact same function. However, because of the address mapping of the registers with LBP following LWP, a user can do word accesses in the extended address space using the LDHX or STHX instructions to access location LWP.

The MMU supports the addition of a 2s compliment value to the linear address pointer without using any math instructions or memory resources. Writes to LAPAB with a 2s compliment value will cause the MMU to add that value to the existing value in LAP1:LAP0.

4.5.3.1.4 PPAGE and Linear Address Pointer to Extended Address

See Figure 4-3, on how the program PPAGE memory pages and the linear address pointer are mapped to extended address space.

4.6 RAM

The MC9S08LH64 series include static RAM. The locations in RAM below 0x0100 can be accessed using the more efficient direct addressing mode, and any single bit in this area can be accessed with the bit manipulation instructions (BCLR, BSET, BRCLR, and BRSET). Locating the most frequently accessed program variables in this area of RAM is preferred.

At power-on, the contents of RAM are uninitialized. RAM data is unaffected by any reset provided that the supply voltage does not drop below the minimum value for RAM retention (V_{RAM}).

For compatibility with M68HC05 MCUs, the HCS08 resets the stack pointer to 0x00FF. In the MC9S08LH64 series, it is usually best to reinitialize the stack pointer to the top of the RAM so the direct-page RAM can be used for frequently accessed RAM variables and bit-addressable program variables. Include the following 2-instruction sequence in your reset initialization routine (where RamLast is equated to the highest address of the RAM in the Freescale Semiconductor-provided equate file).

```
LDHX    #RamLast+1    ;point one past RAM
TXS                   ;SP<-(H:X-1)
```

When security is enabled, the RAM is considered a secure memory resource and is not accessible through BDM or through code executing from non-secure memory. See "Section 4.7.5", for a detailed description of the security feature.

4.7 Flash

The flash memory is intended primarily for program storage. In-circuit programming allows the operating program to be loaded into the flash memory after final assembly of the application product. It is possible to program the entire array through the single-wire background debug interface. Because no special voltages are needed for flash erase and programming operations, in-application programming is also possible through other software-controlled communication paths. For a more detailed discussion of in-circuit and in-application programming, refer to the *HCS08 Family Reference Manual, Volume I,* Freescale Semiconductor document order number HCS08RMv1/D.

Because the MC9S08LH64 Series contains two flash arrays, program and erase operations can be conducted on one array while executing code from the other. The security and protection features treat the two arrays as a single memory entity. Programming and erasing of each flash array is conducted through the same command interface detailed in the following sections.

It is not possible to page erase or program both arrays at the same time. The mass erase command will erase both arrays, and the blank check command will check both arrays.

4.7.1 Features

Features of the flash memory include:

- Flash size
  - MC9S08LH64: 65,326 bytes (Dual 32,768 Arrays)
  - MC9S08LH36: 36,864 bytes (12,288 in flash B and 24,576 in flash A)
- Single power supply program and erase

- Automated program and erase algorithm
- Fast program and erase operation
- Burst program command for faster flash array program times
- Up to 100,000 program/erase cycles at typical voltage and temperature
- Flexible protection scheme to prevent accidental program or erase
- Security feature to prevent unauthorized access to the flash and RAM
- Auto power-down for low-frequency read accesses

4.7.2 Program and Erase Times

Before any program or erase command can be accepted, the flash clock divider register (FCDIV) must be written to set the internal clock for the flash module to a frequency (f_{FCLK}) between 150 kHz and 200 kHz (see Section 4.9.1, "Flash Clock Divider Register (FCDIV)"). This register can be written only once, so normally this write is done during reset initialization. FCDIV cannot be written if the access error flag, FACCERR in FSTAT, is set. The user must ensure that FACCERR is not set before writing to the FCDIV register. One period of the resulting clock ($1/f_{FCLK}$) is used by the command processor to time program and erase pulses. An integer number of these timing pulses is used by the command processor to complete a program or erase command.

Table 4-12 shows program and erase times. The bus clock frequency and FCDIV determine the frequency of FCLK (f_{FCLK}). The time for one cycle of FCLK is $t_{FCLK} = 1/f_{FCLK}$. The times are shown as a number of cycles of FCLK and as an absolute time for the case where $t_{FCLK} = 5$ µs. Program and erase times shown include overhead for the command state machine and enabling and disabling of program and erase voltages.

Table 4-12. Program and Erase Times

Parameter	Cycles of FCLK	Time if FCLK = 200 kHz
Byte program	9	45 µs
Byte program (burst)	4	20 µs[1]
Page erase	4000	20 ms
Mass erase	20,000	100 ms

[1] Excluding start/end overhead

4.7.3 Program and Erase Command Execution

The steps for executing any of the commands are listed below. The FCDIV register must be initialized and any error flags cleared before beginning command execution. The command execution steps are:

1. Write a data value to an address in the flash array. The address and data information from this write is latched into the flash interface. This write is a required first step in any command sequence. For erase and blank check commands, the value of the data is not important. For page erase commands, the address may be any address in the 512-byte page of flash to be erased. For mass erase and blank check commands, the address can be any address in the flash memory. Whole pages of 512 bytes are the smallest block of flash that may be erased.

NOTE

Do not program any byte in the flash more than once after a successful erase operation. Reprogramming bits to a byte that is already programmed is not allowed without first erasing the page in which the byte resides or mass erasing the entire flash memory. Programming without first erasing may disturb data stored in the flash.

2. Write the command code for the desired command to FCMD. The five valid commands are blank check (0x05), byte program (0x20), burst program (0x25), page erase (0x40), and mass erase (0x41). The command code is latched into the command buffer.
3. Write a 1 to the FCBEF bit in FSTAT to clear FCBEF and launch the command (including its address and data information).

A partial command sequence can be aborted manually by writing a 0 to FCBEF any time after the write to the memory array and before writing the 1 that clears FCBEF and launches the complete command. Aborting a command in this way sets the FACCERR access error flag, which must be cleared before starting a new command.

A strictly monitored procedure must be obeyed or the command will not be accepted. This minimizes the possibility of any unintended changes to the flash memory contents. The command complete flag (FCCF) indicates when a command is complete. The command sequence must be completed by clearing FCBEF to launch the command. Figure 4-9 is a flowchart for executing all of the commands except for burst programming. The FCDIV register must be initialized before using any flash commands. This must be done only once following a reset.

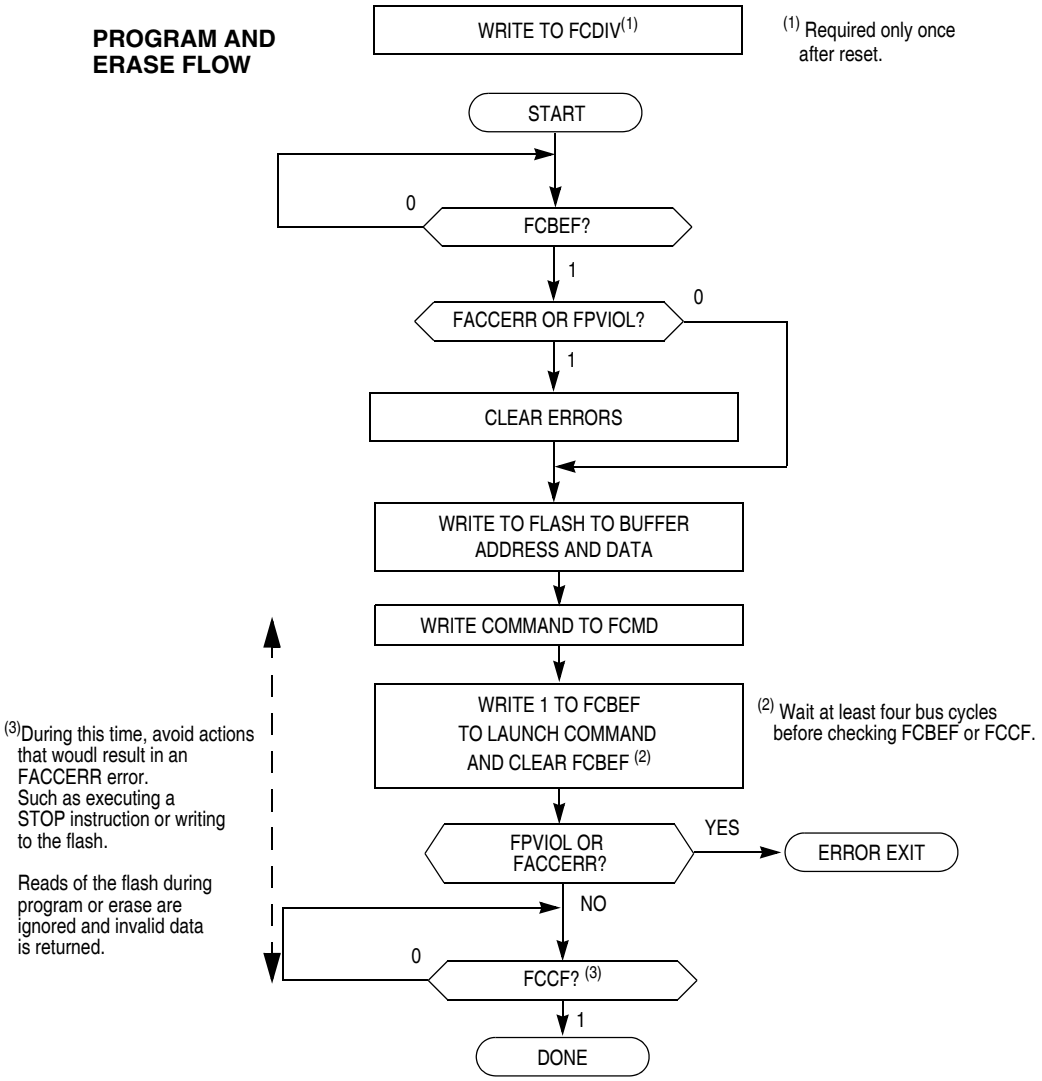


Figure 4-9. Flash Program and Erase Flowchart

4.7.4 Burst Program Execution

The burst program command is used to program sequential bytes of data in less time than would be required using the standard program command. This is possible because the high voltage to the flash array does not need to be disabled between program operations. Ordinarily, when a program or erase command is issued, an internal charge pump associated with the flash memory must be enabled to supply high voltage to the array. Upon completion of the command, the charge pump is turned off. When a burst program command is issued, the charge pump is enabled and then remains enabled after completion of the burst program operation if these two conditions are met:

- The next burst program command has been queued before the current program operation has completed.

- The next sequential address selects a byte on the same physical row as the current byte being programmed. A row of flash memory consists of 64 bytes. A byte within a row is selected by addresses A5 through A0. A new row begins when addresses A5 through A0 are all zero.

The first byte of a series of sequential bytes being programmed in burst mode will take the same amount of time to program as a byte programmed in standard mode. Subsequent bytes will program in the burst program time provided that the conditions above are met. In the case the next sequential address is the beginning of a new row, the program time for that byte will be the standard time instead of the burst time. This is because the high voltage to the array must be disabled and then enabled again. If a new burst command has not been queued before the current command completes, then the charge pump will be disabled and high voltage removed from the array.

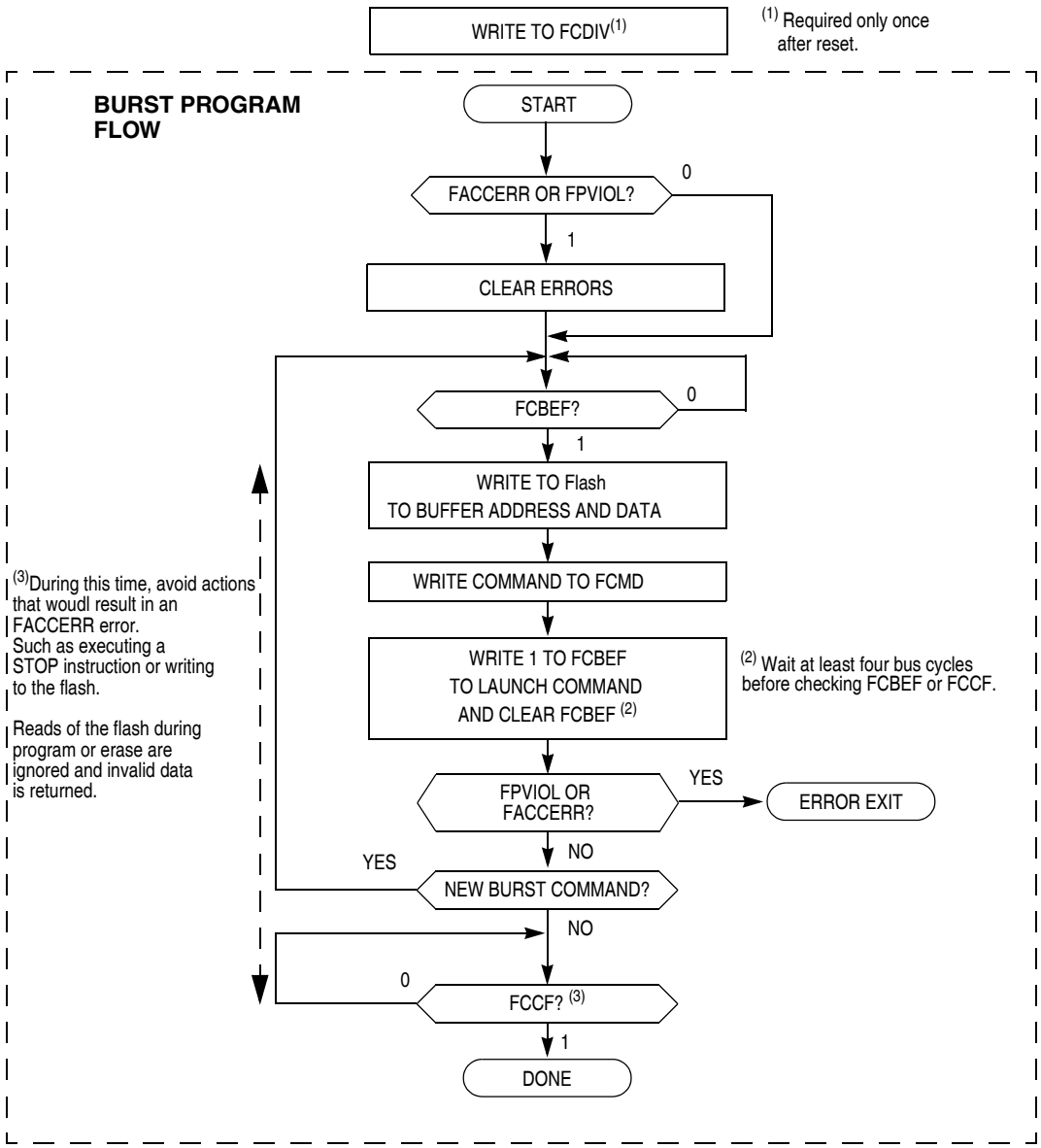


Figure 4-10. Flash Burst Program Flowchart

4.7.5 Access Errors

An access error occurs whenever the command execution protocol is violated.

Any of the following actions will cause the access error flag (FACCERR) in FSTAT to be set. FACCERR must be cleared by writing a 1 to FACCERR in FSTAT before any command can be processed.

- Writing to a flash address before the internal flash clock frequency has been set by writing to the FCDIV register
- Writing to a flash address while FCBEF is not set (A new command cannot start until the command buffer is empty.)
- Writing a second time to a flash address before launching the previous command (There is only one write to flash for every command.)
- Writing a second time to FCMD before launching the previous command (There is only one write to FCMD for every command.)
- Writing to any flash control register other than FCMD after writing to a flash address
- Writing any command code to FCMD other than the five allowed codes (0x05, 0x20, 0x25, 0x40, or 0x41)
- Writing any flash control register other than the write to FSTAT (to clear FCBEF and launch the command) after writing the command to FCMD
- The MCU enters stop mode while a program or erase command is in progress (The command is aborted.)
- Writing the byte program, burst program, or page erase command code (0×20, 0×25, or 0×40) with a background debug command while the MCU is secured (The background debug controller can only do blank check and mass erase commands when the MCU is secure.)
- Writing 0 to FCBEF to cancel a partial command

4.7.6 Flash Block Protection

The block protection feature prevents the protected region of flash from program or erase changes. Block protection is controlled through the flash protection register (FPROT). When enabled, block protection begins at any 512 byte boundary below the last address of flash, 0xFFFF. (See Section 4.9.4, “Flash Protection Register (FPROT and NVPROT)”).

After exit from reset, FPROT is loaded with the contents of the NVPROT location, which is in the nonvolatile register block of the flash memory. FPROT cannot be changed directly from application software to prevent runaway programs from altering the block protection settings. Because NVPROT is within the last 512 bytes of flash, if any amount of memory is protected, NVPROT is itself protected and cannot be altered (intentionally or unintentionally) by the application software. FPROT can be written through background debug commands, which allows a protected flash memory to be erased and reprogrammed.

The block protection mechanism is illustrated in Figure 4-11. The FPS bits are used as the upper bits of the last address of unprotected memory. This address is formed by concatenating FPS7:FPS1 with logic 1 bits as shown. For example, to protect the last 1536 bytes of memory (addresses 0xFA00 through 0xFFFF), the FPS bits must be set to 1111 100, which results in the value 0xF9FF as the last address of unprotected

memory. In addition to programming the FPS bits to the appropriate value, FPDIS (bit 0 of NVPROT) must be programmed to logic 0 to enable block protection. Therefore the value 0xF8 must be programmed into NVPROT to protect addresses 0xFA00 through 0xFFFF.

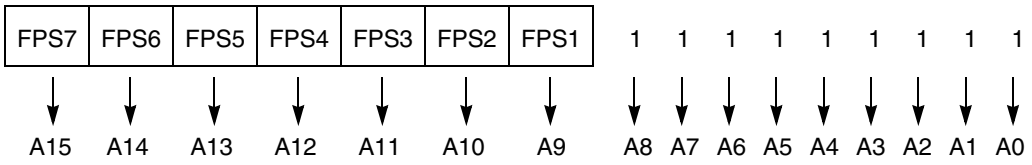


Figure 4-11. Block Protection Mechanism

One use of block protection is to block protect an area of flash memory for a bootloader program. This bootloader program then can be used to erase the rest of the flash memory and reprogram it. Because the bootloader is protected, it remains intact even if MCU power is lost in the middle of an erase and reprogram operation.

4.7.7 Vector Redirection

Whenever any block protection is enabled, the reset and interrupt vectors will be protected. Vector redirection allows users to modify interrupt vector information without unprotecting bootloader and reset vector space. Vector redirection is enabled by programming the FNORED bit in the NVOPT register located at address 0xFFBF to zero. For redirection to occur, at least some portion but not all of the flash memory must be block protected by programming the NVPROT register located at address 0xFFBD. All of the interrupt vectors (memory locations 0xFFC0–0xFFFD) are redirected, though the reset vector (0xFFFE:FFFF) is not.

For example, if 512 bytes of flash are protected, the protected address region is from 0xFE00 through 0xFFFF. The interrupt vectors (0xFFC0–0xFFFD) are redirected to the locations 0xFDC0–0xFDFD. For instance, if an SPI interrupt is taken, the values in the locations 0xFDE0:FDE1 are used for the vector instead of the values in the locations 0xFFE0:FFE1. This allows the user to reprogram the unprotected portion of the flash with new program code including new interrupt vector values while leaving the protected area, which includes the default vector locations, unchanged.

4.8 Security

The MC9S08LH64 series include circuitry to prevent unauthorized access to the contents of flash and RAM memory. When security is engaged, flash and RAM are considered secure resources. Direct-page registers, high-page registers, and the background debug controller are considered unsecured resources. Programs executing within secure memory have normal access to any MCU memory locations and resources. Attempts to access a secure memory location with a program executing from an unsecured memory space or through the background debug interface are blocked (writes are ignored and reads return all 0s).

Security is engaged or disengaged based on the state of two nonvolatile register bits (SEC01:SEC00) in the FOPT register. During reset, the contents of the nonvolatile location NVOPT are copied from flash into the working FOPT register in high-page register space. A user engages security by programming the NVOPT location which can be done at the same time the flash memory is programmed. The 1:0 state

disengages security and the other three combinations engage security. Notice the erased state (1:1) makes the MCU secure. During development, whenever the flash is erased, user code should immediately program the SEC00 bit to 0 in NVOPT so SEC01:SEC00 = 1:0. This would allow the MCU to remain unsecured after a subsequent reset.

The on-chip debug module cannot be enabled while the MCU is secure. The separate background debug controller can still be used for background memory access commands of unsecured resources.

A user can choose to allow or disallow a security unlocking mechanism through an 8-byte backdoor security key. If the nonvolatile KEYEN bit in NVOPT/FOPT is 0, the backdoor key is disabled and there is no way to disengage security without completely erasing all flash locations. If KEYEN is 1, a secure user program can temporarily disengage security by:

1. Writing 1 to KEYACC in the FCNFG register. This makes the flash module interpret writes to the backdoor comparison key locations (NVBACKKEY through NVBACKKEY+7) as values to be compared against the key rather than as the first step in a flash program or erase command.
2. Writing the user-entered key values to the NVBACKKEY through NVBACKKEY+7 locations. These writes must be done in order starting with the value for NVBACKKEY and ending with NVBACKKEY+7. STHX should not be used for these writes because these writes cannot be done on adjacent bus cycles. User software normally would get the key codes from outside the MCU system through a communication interface such as a serial I/O.
3. Writing 0 to KEYACC in the FCNFG register. If the 8-byte key that was just written matches the key stored in the flash locations, SEC01:SEC00 are automatically changed to 1:0 and security will be disengaged until the next reset.

The security key can be written only from secure memory (either RAM or flash), so it cannot be entered through background commands without the cooperation of a secure user program.

The backdoor comparison key (NVBACKKEY through NVBACKKEY+7) is located in flash memory locations in the nonvolatile register space so users can program these locations exactly as they would program any other flash memory location. The nonvolatile registers are in the same 512-byte block of flash as the reset and interrupt vectors, so block protecting that space also block protects the backdoor comparison key. Block protects cannot be changed from user application programs, so if the vector space is block protected, the backdoor security key mechanism cannot permanently change the block protect, security settings, or the backdoor key.

Security can always be disengaged through the background debug interface by taking these steps:

1. Disable any block protections by writing FPROT. FPROT can be written only with background debug commands, not from application software.
2. Mass erase flash if necessary.
3. Blank check flash. Provided flash is completely erased, security is disengaged until the next reset.

   To avoid returning to secure mode after the next reset, program NVOPT so SEC01:SEC00 = 1:0.

4.9 Flash Registers and Control Bits

The flash module has nine 8-bit registers in the high-page register space. Two locations (NVOPT, NVPROT) in the nonvolatile register space in flash memory are copied into corresponding high-page

control registers (FOPT, FPROT) at reset. There is also an 8-byte comparison key in flash memory. Refer to Table 4-4 and Table 4-5 for the absolute address assignments for all flash registers. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file is normally used to translate these names into the appropriate absolute addresses.

4.9.1 Flash Clock Divider Register (FCDIV)

Bit 7 of this register is a read-only flag. DIV Bits 6:0 may be read at any time but can be written only one time. Before any erase or programming operations are possible, write to this register to set the frequency of the clock for the nonvolatile memory system within acceptable limits. Table 4-14 shows the appropriate values for PRDIV8 and DIV for selected bus frequencies.

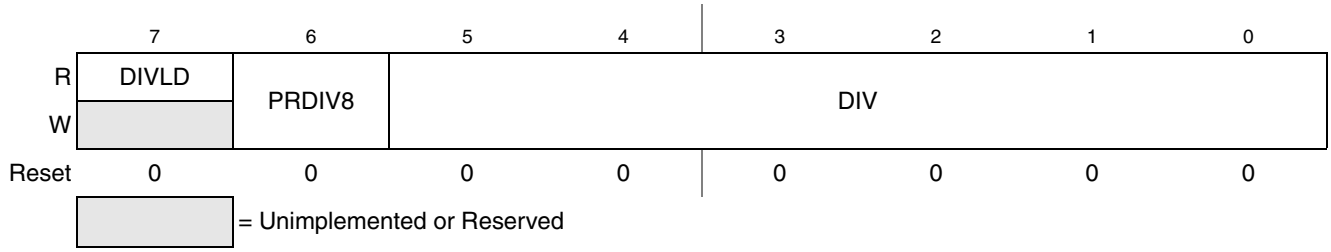


Figure 4-12. Flash Clock Divider Register (FCDIV)

Table 4-13. FCDIV Register Field Descriptions

Field	Description
7 DIVLD	**Divisor Loaded Status Flag** — When set, this read-only status flag indicates that the FCDIV register has been written since reset. Reset clears this bit and the first write to this register causes this bit to become set regardless of the data written. 0 FCDIV has not been written since reset; erase and program operations disabled for flash. 1 FCDIV has been written since reset; erase and program operations enabled for flash.
6 PRDIV8	**Prescale (Divide) Flash Clock by 8** 0 Clock input to the flash clock divider is the bus rate clock. 1 Clock input to the flash clock divider is the bus rate clock divided by 8.
5:0 DIV	**Divisor for Flash Clock Divider** — The flash clock divider divides the bus rate clock (or the bus rate clock divided by 8 if PRDIV8 = 1) by the value in the 6-bit DIV field plus one. The resulting frequency of the internal flash clock must fall within the range of 200 kHz to 150 kHz for proper flash operations. Program/Erase timing pulses are one cycle of this internal flash clock which corresponds to a range of 5 μs to 6.7 μs. The automated programming logic uses an integer number of these pulses to complete an erase or program operation. See Equation 4-1 and Equation 4-2.

$$\text{if PRDIV8} = 0 \text{ — } f_{FCLK} = f_{Bus} \div (\text{DIV} + 1) \qquad \textit{Eqn. 4-1}$$

$$\text{if PRDIV8} = 1 \text{ — } f_{FCLK} = f_{Bus} \div (8 \times (\text{DIV} + 1)) \qquad \textit{Eqn. 4-2}$$

Table 4-14 shows the appropriate values for PRDIV8 and DIV for selected bus frequencies.

Table 4-14. Flash Clock Divider Settings

f_{Bus}	PRDIV8 (Binary)	DIV (Decimal)	f_{FCLK}	Program/Erase Timing Pulse (5 μs Min, 6.7 μs Max)
20 MHz	1	12	192.3 kHz	5.2 μs
10 MHz	0	49	200 kHz	5 μs
8 MHz	0	39	200 kHz	5 μs
4 MHz	0	19	200 kHz	5 μs
2 MHz	0	9	200 kHz	5 μs
1 MHz	0	4	200 kHz	5 μs
200 kHz	0	0	200 kHz	5 μs
150 kHz	0	0	150 kHz	6.7 μs

4.9.2 Flash Options Register (FOPT and NVOPT)

During reset, the contents of the nonvolatile location NVOPT are copied from flash into FOPT. To change the value in this register, erase and reprogram the NVOPT location in flash memory as usual and then issue a new MCU reset.

	7	6	5	4	3	2	1	0
R	KEYEN	FNORED	0	0	0	0	SEC01	SEC00
W								
Reset	This register is loaded from nonvolatile location NVOPT during reset.							

= Unimplemented or Reserved

Figure 4-13. Flash Options Register (FOPT)

Table 4-15. FOPT Register Field Descriptions

Field	Description
7 KEYEN	**Backdoor Key Mechanism Enable** — When this bit is 0, the backdoor key mechanism cannot be used to disengage security. The backdoor key mechanism is accessible only from user (secured) firmware. BDM commands cannot be used to write key comparison values that would unlock the backdoor key. For more detailed information about the backdoor key mechanism, refer to Section 4.8, “Security.” 0 No backdoor key access allowed. 1 If user firmware writes an 8-byte value that matches the nonvolatile backdoor key (NVBACKKEY through NVBACKKEY+7 in that order), security is temporarily disengaged until the next MCU reset.
6 FNORED	**Vector Redirection Disable** — When this bit is 1, then vector redirection is disabled. 0 Vector redirection enabled. 1 Vector redirection disabled.
1:0 SEC0[1:0]	**Security State Code** — This 2-bit field determines the security state of the MCU as shown in Table 4-16. When the MCU is secure, the contents of RAM and flash memory cannot be accessed by instructions from any unsecured source including the background debug interface. SEC01:SEC00 changes to 1:0 after successful backdoor key entry or a successful blank check of flash. For more detailed information about security, refer to Section 4.8, “Security.”

Table 4-16. Security States[1]

SEC01:SEC00	Description
0:0	secure
0:1	secure
1:0	unsecured
1:1	secure

1 SEC01:SEC00 changes to 1:0 after successful backdoor key entry or a successful blank check of flash.

4.9.3 Flash Configuration Register (FCNFG)

	7	6	5	4	3	2	1	0
R	0	0	KEYACC	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-14. Flash Configuration Register (FCNFG)

Table 4-17. FCNFG Register Field Descriptions

Field	Description
5 KEYACC	**Enable Writing of Access Key** — This bit enables writing of the backdoor comparison key. For more detailed information about the backdoor key mechanism, refer to Section 4.8, “Security.” 0 Writes to 0xFFB0–0xFFB7 are interpreted as the start of a flash programming or erase command. 1 Writes to NVBACKKEY (0xFFB0–0xFFB7) are interpreted as comparison key writes.

4.9.4 Flash Protection Register (FPROT and NVPROT)

During reset, the contents of the nonvolatile location NVPROT is copied from flash into FPROT. FPROT can be read at any time. With FPDIS set, all bits are writable, but with FPDIS clear the FPS bits are writable as long as the size of the protected region is being increased. Any FPROT write that attempts to decrease the size of the protected region will be ignored.

	7	6	5	4	3	2	1	0
R W	FPS[(1)]							FPDIS[(1)]
Reset	This register is loaded from nonvolatile location NVPROT during reset.							

1 Background commands can be used to change the contents of these bits in FPROT.

Figure 4-15. Flash Protection Register (FPROT)

Table 4-18. FPROT Register Field Descriptions

Field	Description
7:1 FPS[7:1]	**Flash Protect Select Bits** — When FPDIS = 0, this 7-bit field determines the ending address of unprotected flash locations at the high address end of the flash. Protected flash locations cannot be erased or programmed.
0 FPDIS	**Flash Protection Disable** 0 Flash block specified by FPS7:FPS1 is block protected (program and erase not allowed). 1 No flash block is protected.

4.9.5 Flash Status Register (FSTAT)

	7	6	5	4	3	2	1	0
R	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
W								
Reset	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-16. Flash Status Register (FSTAT)

Table 4-19. FSTAT Register Field Descriptions

Field	Description
7 FCBEF	**Flash Command Buffer Empty Flag** — The FCBEF bit is used to launch commands. It also indicates that the command buffer is empty so that a new command sequence can be executed when performing burst programming. The FCBEF bit is cleared by writing a 1 to it or when a burst program command is transferred to the array for programming. Only burst program commands can be buffered. 0 Command buffer is full (not ready for additional commands). 1 A new burst program command can be written to the command buffer.
6 FCCF	**Flash Command Complete Flag** — FCCF is set automatically when the command buffer is empty and no command is being processed. FCCF is cleared automatically when a new command is started (by writing 1 to FCBEF to register a command). Writing to FCCF has no meaning or effect. 0 Command in progress 1 All commands complete
5 FPVIOL	**Protection Violation Flag** — FPVIOL is set automatically when a command is written that attempts to erase or program a location in a protected block (the erroneous command is ignored). FPVIOL is cleared by writing a 1 to FPVIOL. 0 No protection violation. 1 An attempt was made to erase or program a protected location.

Table 4-19. FSTAT Register Field Descriptions (continued)

Field	Description
4 FACCERR	**Access Error Flag** — FACCERR is set automatically when the proper command sequence is not obeyed exactly (the erroneous command is ignored), if a program or erase operation is attempted before the FCDIV register has been initialized, or if the MCU enters stop while a command was in progress. For a more detailed discussion of the exact actions that are considered access errors, see Section 4.7.5, "Access Errors." FACCERR is cleared by writing a 1 to FACCERR. Writing a 0 to FACCERR has no meaning or effect. 0 No access error. 1 An access error has occurred.
2 FBLANK	**Flash Verified as All Blank (erased) Flag** — FBLANK is set automatically at the conclusion of a blank check command if the entire flash array was verified to be erased. FBLANK is cleared by clearing FCBEF to write a new valid command. Writing to FBLANK has no meaning or effect. 0 After a blank check command is completed and FCCF = 1, FBLANK = 0 indicates the flash array is not completely erased. 1 After a blank check command is completed and FCCF = 1, FBLANK = 1 indicates the flash array is completely erased (all 0xFF).

4.9.6 Flash Command Register (FCMD)

Only five command codes are recognized in normal user modes as shown in Table 4-20. Refer to Section 4.7.3, "Program and Erase Command Execution," for a detailed discussion of flash programming and erase operations.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	FCMD							
Reset	0	0	0	0	0	0	0	0

Figure 4-17. Flash Command Register (FCMD)

Table 4-20. Flash Commands

Command	FCMD	Equate File Label
Blank check	0x05	mBlank
Byte program	0x20	mByteProg
Byte program — burst mode	0x25	mBurstProg
Page erase (512 bytes/page)	0x40	mPageErase
Mass erase (all flash)	0x41	mMassErase

All other command codes are illegal and generate an access error.

It is not necessary to perform a blank check command after a mass erase operation. Only blank check is required as part of the security unlocking mechanism.

Chapter 5
Resets, Interrupts, and General System Control

5.1 Introduction

This section discusses basic reset and interrupt mechanisms and the various sources of reset and interrupt in the MC9S08LH64 series. Some interrupt sources from peripheral modules are discussed in greater detail in other sections of this document. This section gathers basic information about all reset and interrupt sources in one place for easy reference.

5.2 Features

Reset and interrupt features include:

- Multiple sources of reset for flexible system configuration and reliable operation
- Reset status register (SRS) to indicate source of most recent reset
- Separate interrupt vector for most modules (reduces polling overhead) (see Table 5-2)

5.3 MCU Reset

Resetting the MCU provides a way to start processing from an known set of initial conditions. During reset, most control and status registers are forced to initial values and the program counter is loaded from the reset vector (0xFFFE:0xFFFF). On-chip peripheral modules are disabled and I/O pins are initially configured as general-purpose high-impedance inputs with pullup devices disabled. The I bit in the condition code register (CCR) is set to block maskable interrupts so the user program has a chance to initialize the stack pointer (SP) and system control settings. SP is forced to 0x00FF at reset.

The MC9S08LH64 series have the following sources for reset:

- Power-on reset (POR)
- External pin reset (PIN)
- Computer operating properly (COP) timer
- Illegal opcode detect (ILOP)
- Illegal address detect (ILAD)(MC9S08LH36 devices only)
- Low-voltage detect (LVD)
- Background debug forced reset

Each of these sources, with the exception of the background debug forced reset, has an associated bit in the system reset status register (SRS).

5.4 Computer Operating Properly (COP) Watchdog

The COP watchdog can force a system reset when the application software fails to execute as expected. To prevent a system reset from the COP timer (when it is enabled), application software must reset the COP counter periodically. If the application program gets lost and fails to reset the COP counter before it times out, a system reset is generated to force the system back to a known starting point.

After any reset, the COPE becomes set in SOPT1 enabling the COP watchdog (see Section 5.8.4, "System Options Register 1 (SOPT1)," for additional information). If the COP watchdog is not used in an application, it can be disabled by clearing COPE. The COP counter is reset by writing any value to the address of SRS. This write does not affect the data in the read-only SRS. Instead, the act of writing to this address is decoded and sends a reset signal to the COP counter.

The COPCLKS bit in SOPT2 (see Section 5.8.5, "System Options Register 2 (SOPT2)," for additional information) selects the clock source used for the COP timer. The clock source options are either the bus clock or an internal 1 kHz clock source. With each clock source, there is an associated short and long time-out controlled by COPT in SOPT1. Table 5-1 summaries the control functions of the COPCLKS and COPT bits. The COP watchdog defaults to operation from the 1 kHz clock source and the associated long time-out (2^8 cycles).

Table 5-1. COP Configuration Options

Control Bits		Clock Source	COP Overflow Count
COPCLKS	COPT		
0	0	~1 kHz	2^5 cycles (32 ms)[1]
0	1	~1 kHz	2^8 cycles (256 ms)[1]
1	0	Bus	2^{13} cycles
1	1	Bus	2^{18} cycles

[1] Values are shown in this column based on t_{LPO} = 1 ms. See t_{LPO} in the data sheet for the tolerance of this value.

Even if the application will use the reset default settings of COPE, COPCLKS, and COPT, the user must write to the write-once SOPT1 and SOPT2 registers during reset initialization to lock in the settings. That way, they cannot be changed accidentally if the application program gets lost. The initial writes to SOPT1 and SOPT2 will reset the COP counter.

The write to SRS that services (clears) the COP counter must not be placed in an interrupt service routine (ISR) because the ISR could continue to be executed periodically even if the main application program fails.

In background debug mode, the COP counter does not increment.

When the bus clock source is selected, the COP counter does not increment while the system is in stop mode. The COP counter resumes as soon as the MCU exits stop mode.

When the 1-kHz clock source is selected, the COP counter is re-initialized to zero upon entry to stop mode. The COP counter begins from zero after the MCU exits stop mode.

5.5 Interrupts

Interrupts provide a way to save the current CPU status and registers, execute an interrupt service routine (ISR), and then restore the CPU status so processing can resume where it left off before the interrupt. Other than the software interrupt (SWI), which is a program instruction, interrupts are caused by hardware events such as an edge on the IRQ pin or a timer-overflow event. The debug module can also generate an SWI under certain circumstances.

If an event occurs in an enabled interrupt source, an associated read-only status flag will become set. The CPU will not respond unless the local interrupt enable is a 1 (enabled) and the I bit in the CCR is 0 to allow interrupts. The global interrupt mask (I bit) in the CCR is initially set after reset which prevents all maskable interrupt sources. The user program initializes the stack pointer and performs other system setup before clearing the I bit to allow the CPU to respond to interrupts.

When the CPU receives a qualified interrupt request, it completes the current instruction before responding to the interrupt. The interrupt sequence obeys the same cycle-by-cycle sequence as the SWI instruction and consists of:

- Saving the CPU registers on the stack
- Setting the I bit in the CCR to mask further interrupts
- Fetching the interrupt vector for the highest-priority interrupt that is currently pending
- Filling the instruction queue with the first three bytes of program information starting from the address fetched from the interrupt vector locations

While the CPU is responding to the interrupt, the I bit is automatically set to avoid the possibility of another interrupt interrupting the ISR itself (this is called nesting of interrupts). Normally, the I bit is restored to 0 when the CCR is restored from the value stacked on entry to the ISR. In rare cases, the I bit can be cleared inside an ISR (after clearing the status flag that generated the interrupt) so that other interrupts can be serviced without waiting for the first service routine to finish. This practice is not recommended for anyone other than the most experienced programmers because it can lead to subtle program errors that are difficult to debug.

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

NOTE

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

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

5.5.1 Interrupt Stack Frame

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

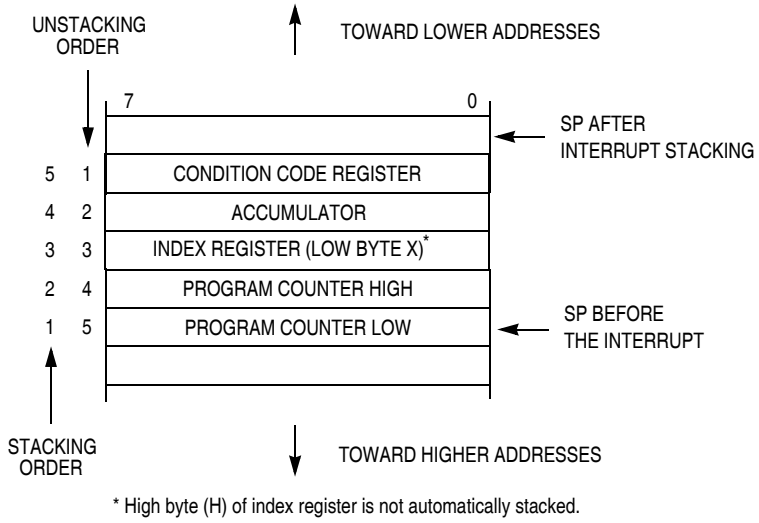


Figure 5-1. Interrupt Stack Frame

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

The status flag corresponding to the interrupt source must be acknowledged (cleared) before returning from the ISR. Typically, the flag is cleared at the beginning of the ISR so that if another interrupt is generated by this same source it will be registered so it can be serviced after completion of the current ISR.

5.5.2 External Interrupt Request (IRQ) Pin

External interrupts are managed by the IRQ status and control register (IRQSC). When the IRQ function is enabled, synchronous logic monitors the pin for edge-only or edge-and-level events. When the MCU is in stop mode and system clocks are shut down, a separate asynchronous path is used so the IRQ pin (if enabled) can wake the MCU.

5.5.2.1 Pin Configuration Options

The IRQ pin enable (IRQPE) control bit in IRQSC must be 1 in order for the IRQ pin to act as the interrupt request (IRQ) input. As an IRQ input, the user can choose the polarity of edges or levels detected (IRQEDG), whether the pin detects edges-only or edges and levels (IRQMOD) and whether an event causes an interrupt or only sets the IRQF flag which can be polled by software (IRQIE).

The IRQ pin, when enabled, defaults to use an internal pull device (IRQPDD = 0), configured as a pullup or pulldown depending on the polarity chosen. If the user desires to use an external pullup or pulldown, the IRQPDD can be written to a 1 to turn off the internal device.

BIH and BIL instructions may be used to detect the level on the IRQ pin when the pin is configured to act as the IRQ input.

5.5.2.2 Edge and Level Sensitivity

The IRQMOD control bit reconfigures the detection logic so it detects edge events and pin levels. In the edge and level detection mode, the IRQF status flag becomes set when an edge is detected (when the IRQ pin changes from the deasserted to the asserted level), but the flag is continuously set (and cannot be cleared) as long as the IRQ pin remains at the asserted level.

5.5.3 Interrupt Vectors, Sources, and Local Masks

Table 5-2 provides a summary of all interrupt sources. Higher-priority sources are located toward the bottom of the table. The high-order byte of the address for the interrupt service routine is located at the first address in the vector address column, and the low-order byte of the address for the interrupt service routine is located at the next higher address.

When an interrupt condition occurs, an associated flag bit becomes set. If the associated local interrupt enable is 1, an interrupt request is sent to the CPU. Within the CPU, if the global interrupt mask (I bit in the CCR) is 0, the CPU will finish the current instruction; stack the PCL, PCH, X, A, and CCR CPU registers; set the I bit; and then fetch the interrupt vector for the highest priority pending interrupt. Processing then continues in the interrupt service routine.

Table 5-2. Interrupt Vectors

Vector Priority	Vector Number	Address (High/Low)	Vector Name	Module	Source	Enable	Description
Lowest	31 through 23	0xFFC0:FFC1 through 0xFFD0:FFD1	Unused Vector Space (available for user program)				
	22	0xFFD2/0xFFD3	Vsci2tx	SCI2	TDRE, TC	TIE, TCIE	SCI2 transmit
	21	0xFFD4/0xFFD5	Vsci2rx	SCI2	IDLE, RDRF, LBKDIF, RXEDGIF	ILIE, RIE, LBKDIE, RXEDGIE	SCI2 receive
	20	0xFFD6/0xFFD7	Vsci2err	SCI2	OR, NF, FE, PF	ORIE, NFIE, FEIE, PFIE	SCI2 error
	19	0xFFD8/0xFFD9	VTOD	TOD	QSECF, SECF, MTCHF	QSECIE, SECIE, MTCHIE	Time of Day interrupt
	18	0xFFDA/0xFFDB	Vacmp	ACMP	ACF	ACIE	Analog comparator
	17	0xFFDC/0xFFDD	Vadc	ADC16	COCO	AIEN	ADC
	16	0xFFDE/0xFFDF	Vkeyboard	KBI	KBF	KBIE	Keyboard pins
	15	0xFFE0/0xFFE1	Viic	IIC	IICIS	IICIE	IIC control
	14	0xFFE2/0xFFE3	Vsci1tx	SCI1	TDRE, TC	TIE, TCIE	SCI1 transmit
	13	0xFFE4/0xFFE5	Vsci1rx	SCI1	IDLE, RDRF, LBKDIF, RXEDGIF	ILIE, RIE, LBKDIE, RXEDGIE	SCI1 receive
	12	0xFFE6/0xFFE7	Vsci1err	SCI1	OR, NF, FE, PF	ORIE, NFIE, FEIE, PFIE	SCI1 error
	11	0xFFE8/0xFFE9	Vspi	SPI	SPIF, MODF, SPTEF	SPIE, SPIE, SPTIE	SPI
	10	0xFFEA/0xFFEB	Vlcd	LCD	LCDF	LCDIE	LCD Frame Interrupt
	9	0xFFEC/0xFFED	Vtpm2ovf	TPM2	TOF	TOIE	TPM2 overflow
	8	0xFFEE/0xFFEF	Vtpm2ch1	TPM2	CH1F	CH1IE	TPM2 channel 1
	7	0xFFF0/0xFFF1	Vtpm2ch0	TPM2	CH0F	CH0IE	TPM2 channel 0
	6	0xFFF2/0xFFF3	Vtpm1ovf	TPM1	TOF	TOIE	TPM1 overflow
	5	0xFFF4/0xFFF5	Vtpm1ch1	TPM1	CH1F	CH1IE	TPM1 channel 1
	4	0xFFF6/0xFFF7	Vtpm1ch0	TPM1	CH0F	CH0IE	TPM1 channel 0
	3	0xFFF8/0xFFF9	Vlvd	System control	LVDF, LVWF	LVDIE, LVWIE	Low-voltage detect, Low-voltage warning
	2	0xFFFA/0xFFFB	Virq	IRQ	IRQF	IRQIE	IRQ pin
	1	0xFFFC/0xFFFD	Vswi	Core	SWI Instruction	—	Software interrupt
Highest	0	0xFFFE/0xFFFF	Vreset	System control	COP LVD $\overline{\text{RESET}}$ pin Illegal opcode Illegal address POR	COPE LVDRE RSTPE — —	Watchdog timer Low-voltage detect External pin Illegal opcode Illegal address

5.6 Low-Voltage Detect (LVD) System

The MC9S08LH64 series include a system to protect against low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system is composed of a power-on reset (POR) circuit and a LVD circuit with a user selectable trip voltage, either high (V_{LVDH}) or low (V_{LVDL}). The LVD circuit is enabled when LVDE in SPMSC1 is high and the trip voltage is selected by LVDV in SPMSC3. The LVD is disabled upon entering either of the stop modes unless LVDSE is set in SPMSC1. If LVDSE and LVDE are both set, then the MCU will enter stop3 instead of stop2, and the current consumption in stop3 with the LVD enabled will be greater.

5.6.1 Power-On Reset Operation

When power is initially applied to the MCU, or when the supply voltage drops below the power-on reset rearm voltage level, V_{POR}, the POR circuit will cause a reset condition. As the supply voltage rises, the LVD circuit will hold the MCU in reset until the supply has risen above the low voltage detection low threshold, V_{LVDL}. Both the POR bit and the LVD bit in SRS are set following a POR.

5.6.2 Low-Voltage Detection (LVD) Reset Operation

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting LVDRE to 1. The low voltage detection threshold is determined by the LVDV bit. After an LVD reset has occurred, the LVD system will hold the MCU in reset until the supply voltage has risen above the low voltage detection threshold. The LVD bit in the SRS register is set following either an LVD reset or POR.

5.6.3 Low-Voltage Detection (LVD) Interrupt Operation

When a low voltage condition is detected and the LVD circuit is configured using SPMSC1 for interrupt operation (LVDE set, LVDIE set, and LVDRE clear), then LVDF in SPMSC1 will be set and an LVD interrupt request will occur. The LVDF bit is cleared by writing a 1 to the LVDACK bit in SPMSC1.

5.6.4 Low-Voltage Warning (LVW) Interrupt Operation

The LVD system has a low voltage warning flag (LVWF) to indicate to the user that the supply voltage is approaching, but is above, the LVD voltage. The LVW also has an interrupt associated with it, enabled by setting the LVWIE bit in the SPMSC3 register. If enabled, an LVW interrupt request will occur when the LVWF is set. LVWF is cleared by writing a 1 to the LVWACK bit in SPMSC3. There are two user selectable trip voltages for the LVW, one high (V_{LVWH}) and one low (V_{LVWL}). The trip voltage is selected by LVWV in SPMSC3.

5.7 Peripheral Clock Gating

The MC9S08LH64 series include a clock gating system to manage the bus clock sources to the individual peripherals. Using this system, the user can enable or disable the bus clock to each of the peripherals at the clock source, eliminating unnecessary clocks to peripherals which are not in use and thereby reducing the overall run and wait mode currents.

Out of reset, all peripheral clocks will be enabled. For lowest possible run or wait currents, user software should disable the clock source to any peripheral not in use. The actual clock will be enabled or disabled immediately following the write to the Clock Gating Control registers (SCGC1 and SCGC2). Any peripheral with a gated clock cannot be used unless its clock is enabled. Writing to the registers of a peripheral with a disabled clock has no effect.

NOTE

User software must disable the peripheral before disabling the clocks to the peripheral. When clocks are re-enabled to a peripheral, the peripheral registers need to be re-initialized by user software.

In stop modes, the bus clock is disabled for all gated peripherals, regardless of the settings in SCGC1 and SCGC2.

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 Table 4-2 and Table 4-4 in Chapter 4 , "Memory," of this reference manual 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 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.

	7	6	5	4	3	2	1	0
R	0	IRQPDD	IRQEDG	IRQPE	IRQF	0	IRQIE	IRQMOD
W						IRQACK		
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

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 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.

Table 5-3. IRQSC Register Field Descriptions (continued)

Field	Description
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.
2 IRQACK	**IRQ Acknowledge** — This write-only bit is used to acknowledge interrupt request events (write 1 to clear IRQF). Writing 0 has no meaning or effect. Reads always return 0. If edge-and-level detection is selected (IRQMOD = 1), IRQF cannot be cleared while the IRQ pin remains at its asserted level.
1 IRQIE	**IRQ Interrupt Enable** — This read/write control bit determines whether IRQ events generate an interrupt request. 0 Interrupt request when IRQF set is disabled (use polling). 1 Interrupt requested whenever IRQF = 1.
0 IRQMOD	**IRQ Detection Mode** — This read/write control bit selects either edge-only detection or edge-and-level detection. The IRQEDG control bit determines the polarity of edges and levels that are detected as interrupt request events. See Section 5.5.2.2, “Edge and Level Sensitivity” for more details. 0 IRQ event on falling edges or rising edges only. 1 IRQ event on falling edges and low levels or on rising edges and high levels.

5.8.2 System Reset Status Register (SRS)

This high-page register includes read-only status flags to indicate the source of the most recent reset. When a debug host forces reset by writing 1 to BDFR in the SBDFR register, none of the status bits in SRS will be set. Writing any value to this register address clears the COP watchdog timer without affecting the contents of this register. The reset state of these bits depends on what caused the MCU to reset.

	7	6	5	4	3	2	1	0
R	POR	PIN	COP	ILOP	ILAD	0	LVD	0
W	Writing any value to SRS address clears COP watchdog timer.							
POR:	1	0	0	0	0	0	1	0
LVD:	u[1]	0	0	0	0	0	1	0
Any other reset:	0	Note[2]	Note[2]	Note[2]	Note[2]	0	0	0

1 u = unaffected

2 Any of these reset sources that are active at the time of reset entry will cause the corresponding bit(s) to be set; bits corresponding to sources that are not active at the time of reset entry will be cleared.

Figure 5-3. System Reset Status (SRS)

Table 5-4. SRS Register Field Descriptions

Field	Description
7 POR	**Power-On Reset** — Reset was caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR. 1 POR caused reset.
6 PIN	**External Reset Pin** — Reset was caused by an active-low level on the external reset pin. 0 Reset not caused by external reset pin. 1 Reset came from external reset pin.
5 COP	**Computer Operating Properly (COP) Watchdog** — Reset was caused by the COP watchdog timer timing out. This reset source can be blocked by COPE = 0. 0 Reset not caused by COP timeout. 1 Reset caused by COP timeout.
4 ILOP	**Illegal Opcode** — Reset was caused by an attempt to execute an unimplemented or illegal opcode. The STOP instruction is considered illegal if stop is disabled by STOPE = 0 in the SOPT register. The BGND instruction is considered illegal if active background mode is disabled by ENBDM = 0 in the BDCSC register. 0 Reset not caused by an illegal opcode. 1 Reset caused by an illegal opcode.
3 ILAD	**Illegal Address** — Reset was caused by an attempt to access either data or an instruction at an unimplemented memory address. 0 Reset not caused by an illegal address 1 Reset caused by an illegal address
1 LVD	**Low Voltage Detect** — If the LVDRE bit is set and the supply drops below the LVD trip voltage, an LVD reset will occur. This bit is also set by POR. 0 Reset not caused by LVD trip or POR. 1 Reset caused by LVD trip or POR.

5.8.3 System Background Debug Force Reset Register (SBDFR)

This high-page register contains a single write-only control bit. A serial background command such as WRITE_BYTE must be used to write to SBDFR. Attempts to write this register from a user program are ignored. Reads always return 0x00.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								BDFR[1]
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

[1] BDFR is writable only through serial background debug commands, not from user programs.

Figure 5-4. System Background Debug Force Reset Register (SBDFR)

Table 5-5. SBDFR Register Field Descriptions

Field	Description
0 BDFR	**Background Debug Force Reset** — A serial background command such as WRITE_BYTE can be used to allow an external debug host to force a target system reset. Writing 1 to this bit forces an MCU reset. This bit cannot be written from a user program.To enter user mode, PTC6/ACMPO/BKGD/MS must be high immediately after issuing WRITE_BYTE command. To enter BDM, PTC6/ACMPO/BKGD/MS must be low immediately after issuing WRITE_BYTE command. See the data sheet for more information.

5.8.4 System Options Register 1 (SOPT1)

This high-page register is a write-once register, so only the first write after reset is honored. It can be read at any time. Any subsequent attempt to write to SOPT1 (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT1 must be written during the user's reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

	7	6	5	4	3	2	1	0
R	COPE	COPT	STOPE	0	0	0	BKGDPE	RSTPE
W								
Reset:	1	1	0	0	0	0	1	u(1)
POR:	1	1	0	0	0	0	1	1
LVR:	1	1	0	0	0	0	1	1

= Unimplemented or Reserved

Figure 5-5. System Options Register 1 (SOPT1)

1 u = unaffected

Table 5-6. SOPT1 Register Field Descriptions

Field	Description
7 COPE	**COP Watchdog Enable** — This write-once bit selects whether the COP watchdog is enabled. 0 COP watchdog timer disabled. 1 COP watchdog timer enabled (force reset on timeout).
6 COPT	**COP Watchdog Timeout** — This write-once bit selects the timeout period of the COP. COPT along with COPCLKS in SOPT2 defines the COP timeout period. 0 Short timeout period selected. 1 Long timeout period selected.
5 STOPE	**Stop Mode Enable** — This write-once bit is used to enable stop mode. If stop mode is disabled and a user program attempts to execute a STOP instruction, an illegal opcode reset is forced. 0 Stop mode disabled. 1 Stop mode enabled.

Table 5-6. SOPT1 Register Field Descriptions (continued)

Field	Description
1 BKGDPE	**Background Debug Mode Pin Enable** — This write-once bit when set enables the PTC6/ACMPO/BKGD/MS pin to function as BKGD/MS. When clear, the pin functions as one of its output only alternative functions. This pin defaults to the BKGD/MS function following any MCU reset. 0 PTC6/ACMPO/BKGD/MS pin functions as PTC6 or ACMPO 1 PTC6/ACMPO/BKGD/MS pin functions as BKGD/MS.
0 RSTPE	**$\overline{\text{RESET}}$ Pin Enable** — This write-once bit when set enables the PTB2/$\overline{\text{RESET}}$ pin to function as $\overline{\text{RESET}}$. When clear, the pin functions as open drain output only. This pin defaults to its $\overline{\text{RESET}}$ function following an MCU POR or LVD. When RSTPE is set, an internal pullup device is enabled on $\overline{\text{RESET}}$. 0 PTB2/$\overline{\text{RESET}}$ pin functions as PTB2. 1 PTB2/$\overline{\text{RESET}}$ pin functions as $\overline{\text{RESET}}$.

5.8.5 System Options Register 2 (SOPT2)

This high page register contains bits to configure MCU specific features on the MC9S08LH64 Series devices.

	7	6	5	4	3	2	1	0
R	COPCLKS[1]	0	0	0	ADCTRS	SPIPS	IICPS	ACIC
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-6. System Options Register 2 (SOPT2)

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

Table 5-7. SOPT2 Register Field Descriptions

Field	Description
7 COPCLKS	**COP Watchdog Clock Select** — This write-once bit selects the clock source of the COP watchdog. 0 Internal 1-kHz clock is source to COP. 1 Bus clock is source to COP.
3 ADCTRS	**ADC Trigger Select** — This bit selects which hardware trigger initiates conversion for the analog to digital converter when the ADC hardware trigger is enabled (ADCTRG is set in ADCSC2 register). 0 Time-of-Day Module Match Interrupt is the source of ADHWT. 1 TPM2 Overflow interrupt is the source of ADHWT.
2 SPIPS	**SPI Pin Select**— This bit selects the location of the SPI pins. 0 MOSI on PTA3, MISO on PTA2, SPSCK on PTA1 and SS on PTA0. 1 MOSI on PTB5, MISO on PTB4, SPSCK on PTB6 and SS on PTB7.
1 IICPS	**IIC Pin Select**— This bit selects the location of the SDA and SCL pins of the IIC module. 0 SDA on PTB4, SCL on PTB5. 1 SDA on PTA2, SCL on PTA3.
0 ACIC	**Analog Comparator to Input Capture Enable**— This bit connects the output of ACMP to TPM2 input channel 0. See Chapter 9, “Analog Comparator (S08ACMPVLPV1) and Chapter 17, “Timer Pulse-Width Modulator (S08TPMV3)” for more details on this feature. 0 ACMP output not connected to TPM2 input channel 0. 1 ACMP output connected to TPM2 input channel 0.

5.8.6 System Device Identification Register (SDIDH, SDIDL)

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

	7	6	5	4	3	2	1	0
R					ID11	ID10	ID9	ID8
W								
Reset:	—	—	—	—	0	0	0	0

= Unimplemented or Reserved

Figure 5-7. System Device Identification Register — High (SDIDH)

Table 5-8. SDIDH Register Field Descriptions

Field	Description
7:4 Reserved	**Bits 7:4 are reserved. Reading these bits will result in an indeterminate value; writes have no effect.**
3:0 ID[11:8]	**Part Identification Number** — Each derivative in the HCS08 Family has a unique identification number. The MC9S08LH64 is hard coded to the value 0x026. See also ID bits in Table 5-9.

	7	6	5	4	3	2	1	0
R	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
W								
Reset:	0	0	1	0	0	1	1	0

= Unimplemented or Reserved

Figure 5-8. System Device Identification Register — Low (SDIDL)

Table 5-9. SDIDL Register Field Descriptions

Field	Description
7:0 ID[7:0]	**Part Identification Number** — Each derivative in the HCS08 Family has a unique identification number. The MC9S08LH64 is hard coded to the value 0x026. See also ID bits in Table 5-8.

5.8.7 System Power Management Status and Control 1 Register (SPMSC1)

This high-page register contains status and control bits to support the low-voltage detect function. To configure the low-voltage detect trip voltage, see Table 5-12 for the LVDV bit description in SPMSC3.

	7	6	5	4	3	2	1	0
R	LVDF	0	LVDIE	LVDRE[1]	LVDSE	LVDE[1]	BGBDS	BGBE
W		LVDACK						
Reset:	0	0	0	1	1	1	0	0
Stop2 wakeup:	u	0	u	u	u	u	0	u

= Unimplemented or Reserved u= Unaffected by reset

[1] This bit can be written only one time after reset. Additional writes are ignored.

Figure 5-9. System Power Management Status and Control 1 Register (SPMSC1)

Table 5-10. SPMSC1 Register Field Descriptions

Field	Description
7 LVDF	**Low-Voltage Detect Flag** — Provided LVDE = 1, this read-only status bit indicates a low-voltage detect event.
6 LVDACK	**Low-Voltage Detect Acknowledge** — This write-only bit is used to acknowledge low voltage detection errors (write 1 to clear LVDF). Reads always return 0.
5 LVDIE	**Low-Voltage Detect Interrupt Enable** — This bit enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling). 1 Request a hardware interrupt when LVDF = 1.
4 LVDRE	**Low-Voltage Detect Reset Enable** — This write-once bit enables LVDF events to generate a hardware reset (provided LVDE = 1). 0 LVDF does not generate hardware resets. 1 Force an MCU reset when LVDF = 1.
3 LVDSE	**Low-Voltage Detect Stop Enable** — Provided LVDE = 1, this read/write bit determines whether the low-voltage detect function operates when the MCU is in stop mode. 0 Low-voltage detect disabled during stop mode. 1 Low-voltage detect enabled during stop mode.
2 LVDE	**Low-Voltage Detect Enable** — This write-once bit enables low-voltage detect logic and qualifies the operation of other bits in this register. 0 LVD logic disabled. 1 LVD logic enabled.
1 BGBDS	**Bandgap Buffer Drive Select** — This bit is used to select the high-drive mode of the bandgap buffer. 0 Bandgap Buffer enabled in low-drive mode if BGBE = 1. 1 Bandgap Buffer enabled in high-drive mode if BGBE = 1.
0 BGBE	**Bandgap Buffer Enable** — This bit enables an internal buffer for the bandgap voltage reference for use by the ADC module on one of its internal channels. 0 Bandgap Buffer is disabled. 1 Bandgap Buffer is enabled.

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

This high-page register contains status and control bits to configure the low power run and wait modes as well as configure the stop mode behavior of the MCU. See Section 3.3.1, "Low-Power Run Mode (LPRun)," Section 3.5.1, "Low-Power Wait Mode (LPWait)," and Section 3.6, "Stop Modes," for more information.

	7	6	5	4	3	2	1	0
R	LPR	LPRS	LPWUI	0	PPDF	0	PPDE[1]	PPDC
W						PPDACK		
Reset:	0	0	0	0	0	0	1	0
Stop2 wakeup:	0	0	u	0	1	0	1	1

= Unimplemented or Reserved u= Unaffected by reset

[1] This bit can be written only one time after reset. Additional writes are ignored.

Figure 5-10. System Power Management Status and Control 2 Register (SPMSC2)

Table 5-11. SPMSC2 Register Field Descriptions

Field	Description
7 LPR	**Low Power Regulator Control** — The LPR bit controls entry into the low power run and wait modes in which the voltage regulator is put into standby. This bit cannot be set if PPDC=1. If PPDC and LPR are set in a single write instruction, only PPDC will actually be set. Automatically cleared when LPWUI is set and an interrupt occurs. 0 Low power run and wait modes are disabled. 1 Low power run and wait modes are enabled.
6 LPRS	**Low Power Regulator Status** — This read-only status bit indicates that the voltage regulator has entered into standby for the low power run or wait mode. 0 The voltage regulator is not currently in standby. 1 The voltage regulator is currently in standby.
5 LPWUI	**Low Power Wake Up on Interrupt** — This bit controls whether or not the voltage regulator exits standby when any active MCU interrupt occurs. 0 The voltage regulator will remain in standby on an interrupt. 1 The voltage regulator will exit standby on an interrupt.
3 PPDF	**Partial Power Down Flag** — This read-only status bit indicates that the MCU has recovered from stop2 mode. 0 MCU has not recovered from stop2 mode. 1 MCU recovered from stop2 mode.
2 PPDACK	**Partial Power Down Acknowledge** — Writing a 1 to PPDACK clears the PPDF bit.
1 PPDE	**Partial Power Down Enable** — The write-once PPDE bit can be used to "lockout" the partial power down mode. 0 Partial power down is not enabled. 1 Partial power down is enabled and controlled via the PPDC bit.
0 PPDC	**Partial Power Down Control** — The PPDC bit controls which power down mode is selected. This bit cannot be set if LPR=1. If PPDC and LPR are set in a single write instruction, only PPDC will actually be set. 0 Stop3 low power mode enabled. 1 Stop2 partial power down mode enabled.

5.8.9 System Power Management Status and Control 3 Register (SPMSC3)

This high-page register is used to report the status of the low voltage warning function and to select the low voltage detect trip voltage.

	7	6	5	4	3	2	1	0
R	LVWF	0	LVDV	LVWV	LVWIE	0	0	0
W		LVWACK						
POR:	0[1]	0	0	0	0	0	0	0
LVR:	0[1]	0	U	U	0	0	0	0
Any other reset:	0[1]	0	U	U	0	0	0	0

= Unimplemented or Reserved U= Unaffected by reset

[1] LVWF will be set in the case when V_{Supply} transitions below the trip point or after reset and V_{Supply} is already below V_{LVW}.

Figure 5-11. System Power Management Status and Control 3 Register (SPMSC3)

Table 5-12. SPMSC3 Register Field Descriptions

Field	Description
7 LVWF	**Low-Voltage Warning Flag** — The LVWF bit indicates the low voltage warning status. 0 Low voltage warning **not** present. 1 Low voltage warning is present or was present.
6 LVWACK	**Low-Voltage Warning Acknowledge** — The LVWF bit indicates the low voltage warning status. Writing a 1 to LVWACK clears LVWF to a 0 if a low voltage warning is not present.
5 LVDV	**Low-Voltage Detect Voltage Select** — The LVDV bit indicates the LVD trip point voltage (V_{LVD}). 0 Low trip point selected ($V_{LVD} = V_{LVOL}$). 1 High trip point selected ($V_{LVD} = V_{LVOH}$).
4 LVWV	**Low-Voltage Detect Voltage Select** — The LVWV bit indicates the LVW trip point voltage (V_{LVW}). 0 Low trip point selected ($V_{LVW} = V_{LVWL}$). 1 High trip point selected ($V_{LVW} = V_{LVWH}$).
3 LVWIE	**Low-Voltage Warning Interrupt Enable** — This bit enables hardware interrupt requests for LVWF. 0 Hardware interrupt disabled (use polling). 1 Request a hardware interrupt when LVWF = 1.

Table 5-13. LVD and LVW trip point typical values[1]

LVDV:LVWV	LVW Trip Point	LVD Trip Point
0:0	V_{LVWL} = 2.16 V	V_{LVDL} = 1.82 V
0:1	V_{LVWH} = 2.46 V	
1:0[2]	V_{LVWL} = 2.16 V	V_{LVDH} = 2.16 V
1:1	V_{LVWH} = 2.46 V	

[1] See data sheet for minimum and maximum values.

[2] This setting is not recommended.

5.8.10 System Clock Gating Control 1 Register (SCGC1)

This high-page register contains control bits to enable or disable the bus clock to the TPMx, ADC, IIC, VREFx, and SCIx modules. Gating off the clocks to unused peripherals reduces the MCU's run and wait currents. See Section 5.7, "Peripheral Clock Gating," for more information.

NOTE

User software must disable the peripheral before disabling the clocks to the peripheral. When clocks are re-enabled to a peripheral, the peripheral registers need to be re-initialized by user software.

	7	6	5	4	3	2	1	0
R W	VREF2	TPM2	TPM1	ADC	VREF1	IIC	SCI2	SCI1
Reset:	1	1	1	1	1	1	1	1

Figure 5-12. System Clock Gating Control 1Register (SCGC1)

Table 5-14. SCGC1 Register Field Descriptions

Field	Description
7 VREF2	**VREF2 Clock Gate Control** — This bit controls the clock gate to the VREF2 module. 0 Bus clock to the VREF2 module is disabled. 1 Bus clock to the VREF2 module is enabled.
6 TPM2	**TPM2 Clock Gate Control** — This bit controls the clock gate to the TPM2 module. 0 Bus clock to the TPM2 module is disabled. 1 Bus clock to the TPM2 module is enabled.
5 TPM1	**TPM1 Clock Gate Control** — This bit controls the clock gate to the TPM1 module. 0 Bus clock to the TPM1 module is disabled. 1 Bus clock to the TPM1 module is enabled.
4 ADC	**ADC Clock Gate Control** — This bit controls the clock gate to the ADC module. 0 Bus clock to the ADC module is disabled. 1 Bus clock to the ADC module is enabled.
3 VREF1	**VREF1 Clock Gate Control** — This bit controls the clock gate to the VREF1 module. 0 Bus clock to the VREF1 module is disabled. 1 Bus clock to the VREF1 module is enabled.
2 IIC	**IIC Clock Gate Control** — This bit controls the clock gate to the IIC module. 0 Bus clock to the IIC module is disabled. 1 Bus clock to the IIC module is enabled.
1 SCI2	**SCI2 Clock Gate Control** — This bit controls the clock gate to the SCI2 module. 0 Bus clock to the SCI2 module is disabled. 1 Bus clock to the SCI2 module is enabled.
0 SCI1	**SCI1 Clock Gate Control** — This bit controls the clock gate to the SCI1 module. 0 Bus clock to the SCI1 module is disabled. 1 Bus clock to the SCI1 module is enabled.

5.8.11 System Clock Gating Control 2 Register (SCGC2)

This high-page register contains control bits to enable or disable the bus clock to the DBG, FLS, IRQ, KBI, ACMP, TOD, LCD, and SPI modules. Gating off the clocks to unused peripherals reduces the MCU's run and wait currents. See Section 5.7, "Peripheral Clock Gating," for more information.

NOTE

User software should disable the peripheral before disabling the clocks to the peripheral. When clocks are re-enabled to a peripheral, the peripheral registers need to be re-initialized by user software.

	7	6	5	4	3	2	1	0
R W	DBG	FLS	IRQ	KBI	ACMP	TOD	LCD	SPI
Reset:	1	1	1	1	1	1	1	1

= Unimplemented or Reserved

Figure 5-13. System Clock Gating Control 2 Register (SCGC2)

Table 5-15. SCGC2 Register Field Descriptions

Field	Description
7 DBG	**DBG Register Clock Gate Control** — This bit controls the bus clock gate to the DBG module. 0 Bus clock to the DBG module is disabled. 1 Bus clock to the DBG module is enabled.
6 FLS	Flash Register Clock Gate Control — This bit controls the bus clock gate to the flash registers. This bit does not affect normal program execution from with the flash array. Only the clock to the flash control registers is affected. 0 Bus clock to the flash registers is disabled. 1 Bus clock to the flash registers is enabled.
5 IRQ	**IRQ Clock Gate Control** — This bit controls the bus clock gate to the IRQ module. 0 Bus clock to the IRQ module is disabled. 1 Bus clock to the IRQ module is enabled.
4 KBI	**KBI Clock Gate Control** — This bit controls the clock gate to the KBI module. 0 Bus clock to the KBI module is disabled. 1 Bus clock to the KBI module is enabled.
3 ACMP	**ACMP Clock Gate Control** — This bit controls the clock gate to both of the ACMP modules. 0 Bus clock to the ACMP modules is disabled. 1 Bus clock to the ACMP modules is enabled.
2 TOD	**TOD Clock Gate Control** — This bit controls the bus clock gate to the TOD module. Only the bus clock is gated, the OSCOUT, ICSIRCLK and LPOCLK are still available to the TOD. 0 Bus clock to the TOD module is disabled. 1 Bus clock to the TOD module is enabled.

Table 5-15. SCGC2 Register Field Descriptions (continued)

Field	Description
1 LCD	**LCD Clock Gate Control** — This bit controls the bus clock gate to the LCD module. Only the bus clock is gated, the OSCOUT, TODCLK and LPOCLK are still available to the LCD. 0 Bus clock to the LCD module is disabled. 1 Bus clock to the LCD module is enabled.
0 SPI	**SPI Clock Gate Control** — This bit controls the clock gate to the SPI module. 0 Bus clock to the SPI module is disabled. 1 Bus clock to the SPI module is enabled.

Chapter 6
Parallel Input/Output Control

6.1 Introduction

This section explains software controls related to parallel input/output (I/O) and pin control. The MC9S08LH64 has five parallel I/O ports which include a total of 39 I/O pins, including two output-only pins. See Chapter 2 , “Pins and Connections,” for more information about pin assignments and external hardware considerations of these pins.

Many of these pins are shared with on-chip peripherals such as timer systems, communication systems, or keyboard interrupts, as shown in Table 2-1. The peripheral modules have priority over the general-purpose I/O functions so that when a peripheral is enabled, the I/O functions associated with the shared pins may be disabled.

After reset, the shared peripheral functions are disabled and the pins are configured as inputs (PTxDDn = 0). The pin control functions for each pin are configured as follows: slew rate control enabled (PTxSEn = 1), low drive strength selected (PTxDSn = 0), and internal pullups disabled (PTxPEn = 0).

Pins that have shared function with the LCD have special behavior based on the state of the VSUPPLY bits in the LCDSUPPLY register. These pins (PTD, PTE and PTA[5:4]) can operate as full complementary drive or open drain drive depending on the VSUPPLY bits. When V_{LL3} is connected to V_{DD} externally, VSUPPLY = 11, FCDEN = 1, and RVEN = 0, the pins operate as full complementary drive. For all other VSUPPLY modes the LCD/GPIO will operate as open drain.

NOTE

Not all general-purpose I/O pins are available on all packages. To avoid extra current drain from floating input pins, the user’s reset initialization routine in the application program must either enable on-chip pullup devices or change the direction of unconnected pins to outputs so the pins do not float.

6.2 Port Data and Data Direction

Reading and writing of parallel I/Os are performed through the port data registers. The direction, either input or output, is controlled through the port data direction registers. The parallel I/O port function for an individual pin is illustrated in the block diagram shown in Figure 6-1.

The data direction control bit (PTxDDn) determines whether the output buffer for the associated pin is enabled, and also controls the source for port data register reads. The input buffer for the associated pin is always enabled unless the pin is enabled as an analog function or is an output-only pin.

When a shared digital function is enabled for a pin, the output buffer is controlled by the shared function. However, the data direction register bit will continue to control the source for reads of the port data register.

When a shared analog function is enabled for a pin, both the input and output buffers are disabled. A value of 0 is read for any port data bit where the bit is an input (PTxDDn = 0) and the input buffer is disabled. In general, whenever a pin is shared with both an alternate digital function and an analog function, the analog function has priority such that if both the digital and analog functions are enabled, the analog function controls the pin.

Write to the port data register before changing the direction of a port pin so it becomes an output. This ensures that the pin will not be driven momentarily with an old data value that happened to be in the port data register.

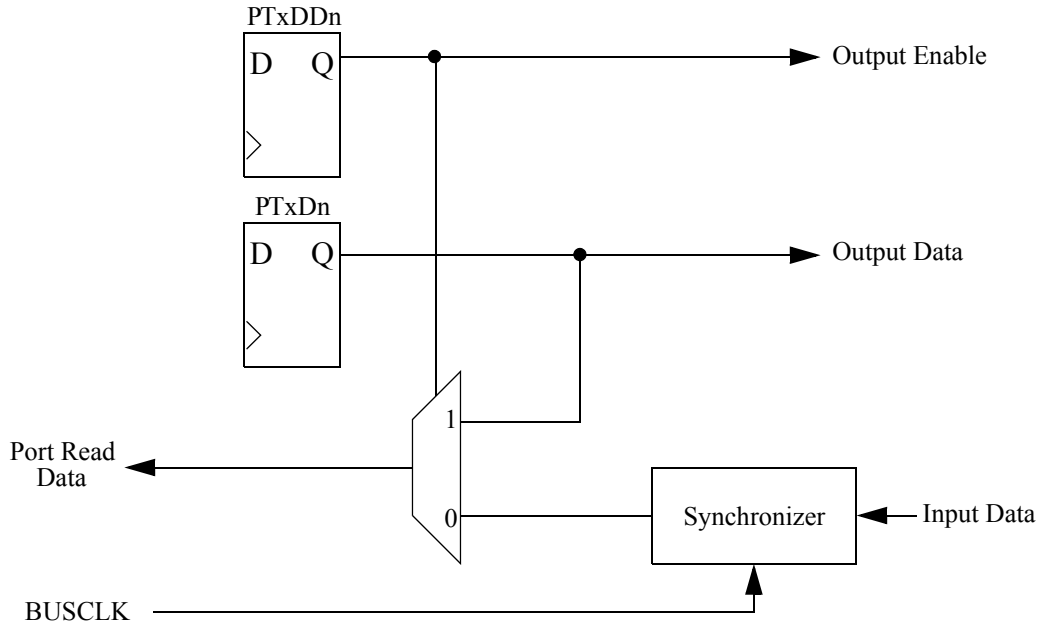


Figure 6-1. Parallel I/O Block Diagram

6.3 Pullup, Slew Rate, and Drive Strength

Associated with the parallel I/O ports is a set of registers located in the high-page register space that operate independently of the parallel I/O registers. These registers are used to control pullups, slew rate, and drive strength for the pins and may be used in conjunction with the peripheral functions on these pins.

6.3.1 Port Internal Pullup Enable

For all GPIO, an internal pullup resistor can be enabled for each port pin by setting the corresponding bit in the pullup enable register (PTxPEn). Typically, GPIO internal pullups are disabled when in output mode. However, for GPIO that are muxed with LCD pins, the internal pullup is not disabled when in open drain, output mode.

The pullup device is disabled if the pin is controlled by an analog function or any shared peripheral function regardless of the state of the corresponding pullup enable register bit.

6.3.2 Port Slew Rate Enable

Slew rate control can be enabled for each port pin by setting the corresponding bit in the slew rate control register (PTxSEn). When enabled, slew control limits the rate at which an output can transition in order to reduce EMC emissions. Slew rate control has no effect on pins that are configured as inputs.

6.3.3 Port Drive Strength Select

An output pin can be configured for high output drive strength by setting the corresponding bit in the drive strength select register (PTxDSn). When high drive is selected, a pin is capable of sourcing and sinking greater current. Even though every I/O pin can be selected as high drive, the user must ensure that the total current source and sink limits for the MCU are not exceeded. Drive strength selection is intended to affect the DC behavior of I/O pins. However, the AC behavior is also affected. High drive allows a pin to drive a greater load with the same switching speed as a low drive enabled pin into a smaller load. Because of this, the EMC emissions may be affected by enabling pins as high drive.

6.4 Open Drain Operation

For most cases, port pins that share functions with the LCD operate as open drain outputs. As an open drain output, the output high of the pin is dependent upon the pullup resistor. The pullup resistor can be an internal resistor enabled by the PTxPEx bit or an external resistor.

- The value of the internal resistor can be in the range of 17.5 to 52.5 kΩ
- The value of an external resistor should be carefully selected to ensure it supports the output loads that are being driven.

6.5 Pin Behavior in Stop Modes

Pin behavior following execution of a STOP instruction depends on the stop mode that is entered. An explanation of pin behavior for the various stop modes follows:

- Stop2 mode is a partial power-down mode, whereby I/O latches are maintained in their pre-STOP instruction state. CPU register status and the state of I/O registers should be saved in RAM before the STOP instruction is executed to place the MCU in stop2 mode. Upon recovery from stop2 mode, before accessing any I/O, the user should examine the state of the PPDF bit in the SPMSC2 register. If the PPDF bit is 0, I/O must be initialized as if a power-on reset had occurred. If the PPDF bit is 1, I/O register states should be restored from the values saved in RAM before the STOP instruction was executed. Peripherals may require initialization or restoration to their pre-stop condition. The user must then write a 1 to the PPDACK bit in the SPMSC2 register. Access to I/O is again permitted in the user application program.
- If the LCD module is configured to operate in stop modes, the drive mode of the GPIO shared with LCD will be retained upon stop recovery.
- In stop3 mode, all I/O is maintained because internal logic circuitry stays powered up. Upon recovery, normal I/O function is available to the user.

6.6 Parallel I/O and Pin Control Registers

This section provides information about the registers associated with the parallel I/O ports. The data and data direction registers are located in page zero of the memory map. The pull up, slew rate, drive strength, and interrupt control registers are located in the high page section of the memory map.

Refer to tables in Chapter 4 , “Memory,” for the absolute address assignments for all parallel I/O and their pin control registers. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file is normally to translate these names into the appropriate absolute addresses.

6.6.1 Port A Registers

Port A is controlled by the registers listed below.

Pins that have shared function with the LCD have special behavior based on the state of the VSUPPLY bits in the LCDSUPPLY register. These pins (PTA[5:4]) can operate as full complementary drive or open drain drive depending on the VSUPPLY bits. When V_{LL3} is connected to V_{DD} externally, VSUPPLY = 11, FCDEN = 1 and RVEN = 0, the pins operate as full complementary drive. For all other VSUPPLY modes the LCD/GPIO will operate as open drain.

6.6.1.1 Port A Data Register (PTAD)

	7	6	5	4	3	2	1	0
R W	PTAD7	PTAD6	PTAD5[1]	PTAD4[1]	PTAD3	PTAD2	PTAD1	PTAD0
Reset:	0	0	0	0	0	0	0	0

1 PTA4 and PTA5 pins are shared with the LCD pins.

Figure 6-2. Port A Data Register (PTAD)

Table 6-1. PTAD Register Field Descriptions

Field	Description
7:0 PTAD[7:0]	**Port A Data Register Bits** — For port A pins that are inputs, reads return the logic level on the pin. For port A pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port A pins that are configured as outputs, the logic level is driven out the corresponding MCU pin. Reset forces PTAD to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups/pulldowns disabled.

6.6.1.2 Port A Data Direction Register (PTADD)

	7	6	5	4	3	2	1	0
R W	PTADD7	PTADD6	PTADD5[1]	PTADD4[1]	PTADD3	PTADD2	PTADD1	PTADD0
Reset:	0	0	0	0	0	0	0	0

[1] PTA4 and PTA5 pins are shared with the LCD pins. When enables as an output, the pin operates in open drain mode for most cases. See Section 6.4, “Open Drain Operation,” for more information.

Figure 6-3. Port A Data Direction Register (PTADD)

Table 6-2. PTADD Register Field Descriptions

Field	Description
7:0 PTADD[7:0]	**Data Direction for Port A Bits** — These read/write bits control the direction of port A pins and what is read for PTAD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port A bit n and PTAD reads return the contents of PTADn.

6.6.1.3 Port A Pull Enable Register (PTAPE)

	7	6	5	4	3	2	1	0
R W	PTAPE7	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
Reset:	0	0	0	0	0	0	0	0

Figure 6-4. Internal Pull Enable for Port A Register (PTAPE)

Table 6-3. PTAPE Register Field Descriptions

Field	Description
7:0 PTAPE[7:0]	**Internal Pull Enable for Port A Bits** — Each of these control bits determines if the internal pullup or pulldown device is enabled for the associated PTA pin. For port A pins that are configured as outputs (except for PTA4 and PTA5), these bits have no effect and the internal pull devices are disabled. 0 Internal pullup/pulldown device disabled for port A bit n. 1 Internal pullup/pulldown device enabled for port A bit n.

6.6.1.4 Port A Slew Rate Enable Register (PTASE)

	7	6	5	4	3	2	1	0
R W	PTASE7	PTASE6	PTASE5	PTASE4	PTASE3	PTASE2	PTASE1	PTASE0
Reset:	1	1	1	1	1	1	1	1

Figure 6-5. Slew Rate Enable for Port A Register (PTASE)

Table 6-4. PTASE Register Field Descriptions

Field	Description
7:0 PTASE[7:0]	**Output Slew Rate Enable for Port A Bits** — Each of these control bits determines if the output slew rate control is enabled for the associated PTA pin. For port A pins that are configured as inputs, these bits have no effect. 0 Output slew rate control disabled for port A bit n. 1 Output slew rate control enabled for port A bit n.

6.6.1.5 Port A Drive Strength Selection Register (PTADS)

	7	6	5	4	3	2	1	0
R W	PTADS7	PTADS6	PTADS5	PTADS4	PTADS3	PTADS2	PTADS1	PTADS0
Reset:	0	0	0	0	0	0	0	0

Figure 6-6. Drive Strength Selection for Port A Register (PTADS)

Table 6-5. PTADS Register Field Descriptions

Field	Description
7:0 PTADS[7:0]	**Output Drive Strength Selection for Port A Bits** — Each of these control bits selects between low and high output drive for the associated PTA pin. For port A pins that are configured as inputs, these bits have no effect. 0 Low output drive strength selected for port A bit n. 1 High output drive strength selected for port A bit n.

6.6.2 Port B Registers

Port B is controlled by the registers listed below.

The pin PTB2 is unique. PTB2 is output only, so the control bits for the input functions have no effect on this pin.

6.6.2.1 Port B Data Register (PTBD)

	7	6	5	4	3	2	1	0
R	PTBD7	PTBD6	PTBD5	PTBD4	0	PTBD2	PTBD1	PTBD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-7. Port B Data Register (PTBD)

Table 6-6. PTBD Register Field Descriptions

Field	Description
7:0 PTBD[7:0]	**Port B Data Register Bits** — For port B pins that are inputs, reads return the logic level on the pin. For port B pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port B pins that are configured as outputs, the logic level is driven out the corresponding MCU pin. Reset forces PTBD to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups/pulldowns disabled.

6.6.2.2 Port B Data Direction Register (PTBDD)

	7	6	5	4	3	2	1	0
R	PTBDD7	PTBDD6	PTBDD5	PTBDD4	0	PTBDD2[1]	PTBDD1	PTBDD0
W								
Reset:	0	0	0	0	0	0	0	0

1 Writing to PTBDD2 has no affect on the output-only PTB2 pin.

Figure 6-8. Port B Data Direction Register (PTBDD)

1

Table 6-7. PTBDD Register Field Descriptions

Field	Description
7:0 PTBDD[7:0]	**Data Direction for Port B Bits** — These read/write bits control the direction of port B pins and what is read for PTBD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port B bit n and PTBD reads return the contents of PTBDn.

6.6.2.3 Port B Pull Enable Register (PTBPE)

	7	6	5	4	3	2	1	0
R	PTBPE7	PTBPE6	PTBPE5	PTBPE4	0	PTBPE2	PTBPE1	PTBPE0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-9. Internal Pull Enable for Port B Register (PTBPE)

Table 6-8. PTBPE Register Field Descriptions

Field	Description
7:0 PTBPE[7:0]	**Internal Pull Enable for Port B Bits** — Each of these control bits determines if the internal pullup or pulldown device is enabled for the associated PTB pin. For port B pins that are configured as outputs, these bits have no effect and the internal pull devices are disabled. 0 Internal pullup/pulldown device disabled for port B bit n. 1 Internal pullup/pulldown device enabled for port B bit n.

6.6.2.4 Port B Slew Rate Enable Register (PTBSE)

	7	6	5	4	3	2	1	0
R	PTBSE7	PTBSE6	PTBSE5	PTBSE4	0	PTBSE2	PTBSE1	PTBSE0
W								
Reset:	1	1	1	1	1	1	1	1

Figure 6-10. Slew Rate Enable for Port B Register (PTBSE)

Table 6-9. PTBSE Register Field Descriptions

Field	Description
7:0 PTBSE[7:0]	**Output Slew Rate Enable for Port B Bits** — Each of these control bits determines if the output slew rate control is enabled for the associated PTB pin. For port B pins that are configured as inputs, these bits have no effect. 0 Output slew rate control disabled for port B bit n. 1 Output slew rate control enabled for port B bit n.

6.6.2.5 Port B Drive Strength Selection Register (PTBDS)

	7	6	5	4	3	2	1	0
R	PTBDS7	PTBDS6	PTBDS5	PTBDS4	0	PTBDS2	PTBDS1	PTBDS0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-11. Drive Strength Selection for Port B Register (PTBDS)

Table 6-10. PTBDS Register Field Descriptions

Field	Description
7:0 PTBDS[7:0]	**Output Drive Strength Selection for Port B Bits** — Each of these control bits selects between low and high output drive for the associated PTB pin. For port B pins that are configured as inputs, these bits have no effect. 0 Low output drive strength selected for port B bit n. 1 High output drive strength selected for port B bit n.

6.6.3 Port C Registers

Port C is controlled by the registers listed below.

The pin PTC6 is unique. PTC6 is an output only, so the control bits for the input functions have no effect on this pin.

6.6.3.1 Port C Data Register (PTCD)

	7	6	5	4	3	2	1	0
R W	PTCD7	PTCD6	PTCD5	PTCD4	PTCD3	PTCD2	PTCD1	PTCD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-12. Port C Data Register (PTCD)

Table 6-11. PTCD Register Field Descriptions

Field	Description
7:0 PTCD[7:0]	**Port C Data Register Bits** — For port C pins that are inputs, reads return the logic level on the pin. For port C pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port C pins that are configured as outputs, the logic level is driven out of the corresponding MCU pin. Reset forces PTCD to all 0s, but these 0s are not driven out of the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups disabled.

6.6.3.2 Port C Data Direction Register (PTCDD)

	7	6	5	4	3	2	1	0
R W	PTCDD7	PTCDD6[1]	PTCDD5	PTCDD4	PTCDD3	PTCDD2	PTCDD1	PTCDD0
Reset:	0	0	0	0	0	0	0	0

[1] Writing to PTCDD6 has no affect to the output-only PTC6 pin.

Figure 6-13. Port C Data Direction Register (PTCDD)

Table 6-12. PTCDD Register Field Descriptions

Field	Description
7:0 PTCDD[7:0]	**Data Direction for Port C Bits** — These read/write bits control the direction of port C pins and what is read for PTCD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port C bit n and PTCD reads return the contents of PTCDn.

6.6.3.3 Port C Pull Enable Register (PTCPE)

	7	6	5	4	3	2	1	0
R W	PTCPE7	PTCPE6	PTCPE5	PTCPE4	PTCPE3	PTCPE2	PTCPE1	PTCPE0
Reset:	0	0	0	0	0	0	0	0

Figure 6-14. Internal Pull Enable for Port C Register (PTCPE)

Table 6-13. PTCPE Register Field Descriptions

Field	Description
7:0 PTCPE[7:0]	**Internal Pull Enable for Port C Bits** — Each of these control bits determines if the internal pullup device is enabled for the associated PTC pin. For port C pins that are configured as outputs, these bits have no effect and the internal pull devices are disabled. 0 Internal pullup device disabled for port C bit n. 1 Internal pullup device enabled for port C bit n.

6.6.3.4 Port C Slew Rate Enable Register (PTCSE)

	7	6	5	4	3	2	1	0
R W	PTCSE7	PTCSE6	PTCSE5	PTCSE4	PTCSE3	PTCSE2	PTCSE1	PTCSE0
Reset:	1	1	1	1	1	1	1	1

Figure 6-15. Slew Rate Enable for Port C Register (PTCSE)

Table 6-14. PTCSE Register Field Descriptions

Field	Description
7:0 PTCSE[7:0]	**Output Slew Rate Enable for Port C Bits** — Each of these control bits determines if the output slew rate control is enabled for the associated PTC pin. For port C pins that are configured as inputs, these bits have no effect. 0 Output slew rate control disabled for port C bit n. 1 Output slew rate control enabled for port C bit n.

6.6.3.5 Port C Drive Strength Selection Register (PTCDS)

	7	6	5	4	3	2	1	0
R W	PTCDS7	PTCDS6	PTCDS5	PTCDS4	PTCDS3	PTCDS2	PTCDS1	PTCDS0
Reset:	0	0	0	0	0	0	0	0

Figure 6-16. Drive Strength Selection for Port C Register (PTCDS)

Table 6-15. PTCDS Register Field Descriptions

Field	Description
7:0 PTCDS[7:0]	**Output Drive Strength Selection for Port C Bits** — Each of these control bits selects between low and high output drive for the associated PTC pin. For port C pins that are configured as inputs, these bits have no effect. 0 Low output drive strength selected for port C bit n. 1 High output drive strength selected for port C bit n.

6.6.4 Port D Registers

Port D is controlled by the registers listed below.

Pins that have shared function with the LCD have special behavior based on the state of the VSUPPLY bits in the LCDSUPPLY register. These pins can operate as full complementary drive or open drain drive depending on the VSUPPLY bits. When V_{LL3} is connected to V_{DD} externally, VSUPPLY = 11, FCDEN = 1 and RVEN = 0, the pins operate as full complementary drive. For all other VSUPPLY modes, the LCD/GPIO operates as an open drain.

6.6.4.1 Port D Data Register (PTDD)

	7	6	5	4	3	2	1	0
R W	PTDD7	PTDD6	PTDD5	PTDD4	PTDD3	PTDD2	PTDD1	PTDD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-17. Port D Data Register (PTDD)

Table 6-16. PTDD Register Field Descriptions

Field	Description
7:0 PTDD[7:0]	**Port D Data Register Bits** — For port D pins that are inputs, reads return the logic level on the pin. For port D pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port D pins that are configured as outputs, the logic level is driven out the corresponding MCU pin. Reset forces PTDD to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups/pulldowns disabled.

6.6.4.2 Port D Data Direction Register (PTDDD)

	7	6	5	4	3	2	1	0
R W	PTDDD7	PTDDD6	PTDDD5	PTDDD4	PTDDD3	PTDDD2	PTDDD1	PTDDD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-18. Port D Data Direction Register (PTDDD)

Table 6-17. PTDDD Register Field Descriptions

Field	Description
7:0 PTDDD[7:0]	**Data Direction for Port D Bits** — These read/write bits control the direction of port D pins and what is read for PTDD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port D bit n and PTDD reads return the contents of PTDDn.

6.6.4.3 Port D Pull Enable Register (PTDPE)

	7	6	5	4	3	2	1	0
R W	PTDPE7	PTDPE6	PTDPE5	PTDPE4	PTDPE3	PTDPE2	PTDPE1	PTDPE0
Reset:	0	0	0	0	0	0	0	0

Figure 6-19. Internal Pull Enable for Port D Register (PTDPE)

Table 6-18. PTDPE Register Field Descriptions

Field	Description
7:0 PTDPE[7:0]	**Internal Pull Enable for Port D Bits** — Each of these control bits determines if the internal pullup or pulldown device is enabled for the associated PTD pin. 0 Internal pullup/pulldown device disabled for port D bit n. 1 Internal pullup/pulldown device enabled for port D bit n.

6.6.4.4 Port D Slew Rate Enable Register (PTDSE)

	7	6	5	4	3	2	1	0
R W	PTDSE7	PTDSE6	PTDSE5	PTDSE4	PTDSE3	PTDSE2	PTDSE1	PTDSE0
Reset:	1	1	1	1	1	1	1	1

Figure 6-20. Slew Rate Enable for Port D Register (PTDSE)

Table 6-19. PTDSE Register Field Descriptions

Field	Description
7:0 PTDSE[7:0]	**Output Slew Rate Enable for Port D Bits** — Each of these control bits determines if the output slew rate control is enabled for the associated PTD pin. 0 Output slew rate control disabled for port D bit n. 1 Output slew rate control enabled for port D bit n.

6.6.4.5 Port D Drive Strength Selection Register (PTDDS)

	7	6	5	4	3	2	1	0
R W	PTDDS7	PTDDS6	PTDDS5	PTDDS4	PTDDS3	PTDDS2	PTDDS1	PTDDS0
Reset:	0	0	0	0	0	0	0	0

Figure 6-21. Drive Strength Selection for Port D Register (PTDDS)

Table 6-20. PTDDS Register Field Descriptions

Field	Description
7:0 PTDDS[7:0]	**Output Drive Strength Selection for Port D Bits** — Each of these control bits selects between low and high output drive for the associated PTD pin. For port D pins that are configured as inputs, these bits have no effect. 0 Low output drive strength selected for port D bit n. 1 High output drive strength selected for port D bit n.

6.6.5 Port E Registers

Port E is controlled by the registers listed below.

Pins that have shared function with the LCD have special behavior based on the state of the VSUPPLY bits in the LCDSUPPLY register. These pins can operate as full complementary drive or open drain drive depending on the VSUPPLY bits. When V_{LL3} is connected to V_{DD} externally, VSUPPLY = 11, FCDEN = 1 and RVEN = 0, the pins operate as full complementary drive. For all other VSUPPLY modes, the LCD/GPIO operates as an open drain.

6.6.5.1 Port E Data Register (PTED)

	7	6	5	4	3	2	1	0
R W	PTED7	PTED6	PTED5	PTED4	PTED3	PTED2	PTED1	PTED0
Reset:	0	0	0	0	0	0	0	0

Figure 6-22. Port E Data Register (PTED)

Table 6-21. PTED Register Field Descriptions

Field	Description
7:0 PTED[7:0]	**Port E Data Register Bits** — For Port E pins that are inputs, reads return the logic level on the pin. For Port E pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For Port E pins that are configured as outputs, the logic level is driven out the corresponding MCU pin. Reset forces PTED to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups/pulldowns disabled.

6.6.5.2 Port E Data Direction Register (PTEDD)

	7	6	5	4	3	2	1	0
R W	PTEDD7	PTEDD6	PTEDD5	PTEDD4	PTEDD3	PTEDD2	PTEDD1	PTEDD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-23. Port E Data Direction Register (PTEDD)

Table 6-22. PTEDD Register Field Descriptions

Field	Description
7:0 PTEDD[7:0]	**Data Direction for Port E Bits** — These read/write bits control the direction of port E pins and what is read for PTED reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for Port E bit n and PTED reads return the contents of PTEDn.

6.6.5.3 Port E Pull Enable Register (PTEPE)

	7	6	5	4	3	2	1	0
R W	PTEPE7	PTEPE6	PTEPE5	PTEPE4	PTEPE3	PTEPE2	PTEPE1	PTEPE0
Reset:	0	0	0	0	0	0	0	0

Figure 6-24. Internal Pull Enable for Port E Register (PTEPE)

Table 6-23. PTEPE Register Field Descriptions

Field	Description
7:0 PTEPE[7:0]	**Internal Pull Enable for Port E Bits** — Each of these control bits determines if the internal pullup or pulldown device is enabled for the associated PTE pin. 0 Internal pullup/pulldown device disabled for port E bit n. 1 Internal pullup/pulldown device enabled for port E bit n.

6.6.5.4 Port E Slew Rate Enable Register (PTESE)

	7	6	5	4	3	2	1	0
R W	PTESE7	PTESE6	PTESE5	PTESE4	PTESE3	PTESE2	PTESE1	PTESE0
Reset:	1	1	1	1	1	1	1	1

Figure 6-25. Slew Rate Enable for Port E Register (PTESE)

Table 6-24. PTESE Register Field Descriptions

Field	Description
7:0 PTESE[7:0]	**Output Slew Rate Enable for Port E Bits** — Each of these control bits determines if the output slew rate control is enabled for the associated PTE pin. 0 Output slew rate control disabled for port E bit n. 1 Output slew rate control enabled for port E bit n.

Chapter 7
Keyboard Interrupt (S08KBIV2)

7.1 Introduction

The keyboard interrupt (KBI) module provides up to eight independently enabled external interrupt sources.

Figure 7-2 shows the device-level block diagram with the KBI module highlighted.

7.1.1 KBI shared with LCD pins

PTA4/KBI4/ADP8/LCD43 and PTA5/KBI5/ADP9/LCD42 share KBI functionality with LCD pins. LCD functionality must be disabled for these pins to operate as KBI pins. As KBI inputs, these pins operate just like other KBI pins.

7.1.2 KBI Clock Gating

The bus clock to the KBI can be gated on and off using the KBI bit in SCGC2. This bit is set after any reset, which enables the bus clock to this module. To conserve power, this bit can be cleared to disable the clock to this module when not in use. See Section 5.7, “Peripheral Clock Gating,” for details.

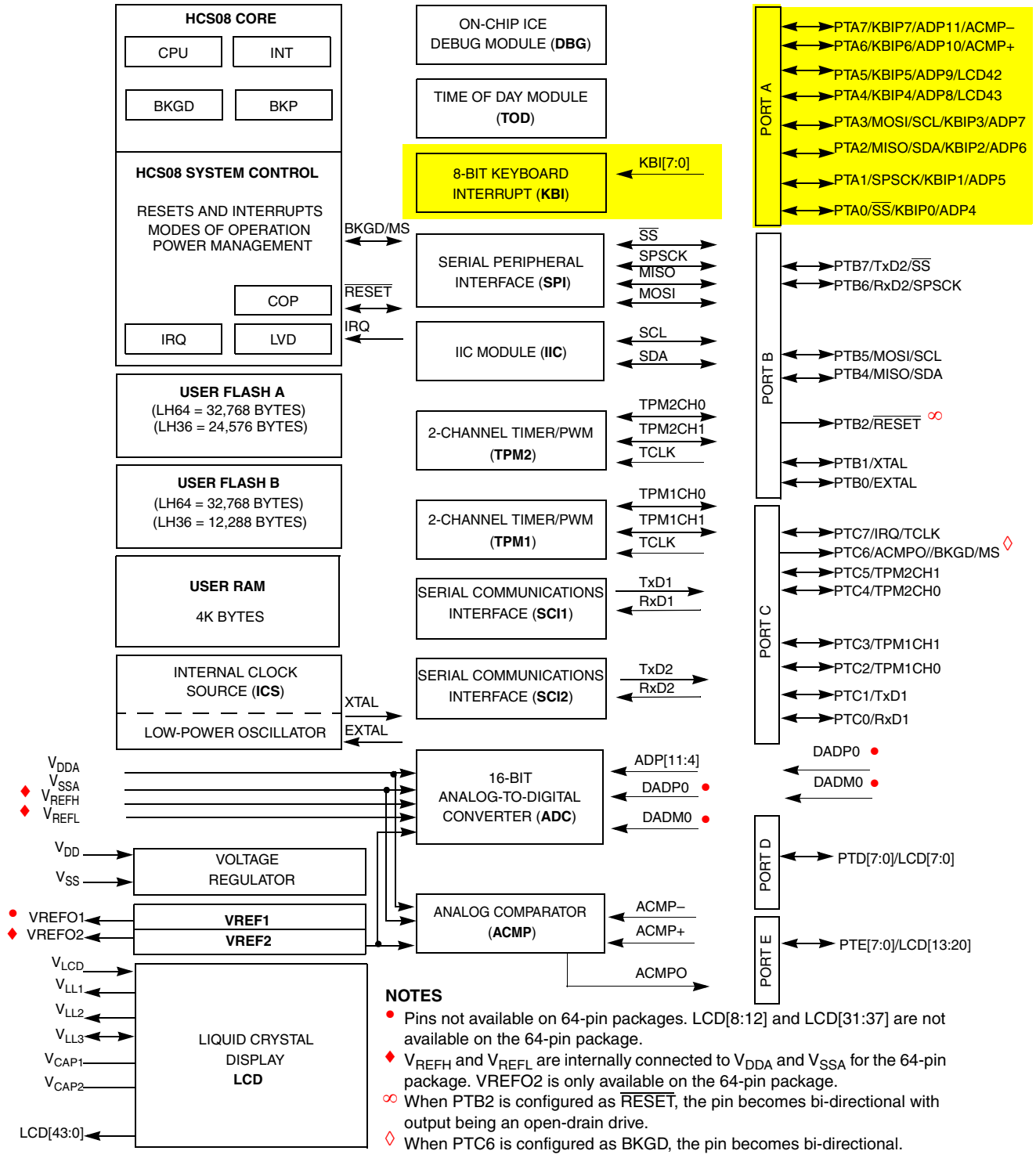


Figure 7-1. MC9S08LH64 Series Block Diagram Highlighting KBI Block and Pins

7.2 Features

The KBI features include:

- Up to eight keyboard interrupt pins with individual pin enable bits.
- Each keyboard interrupt pin is programmable as falling edge (or rising edge) only, or both falling edge and low level (or both rising edge and high level) interrupt sensitivity.
- One software enabled keyboard interrupt.
- Exit from low-power modes.

7.2.1 Modes of Operation

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

7.2.1.1 KBI in Wait Mode

The KBI continues to operate in wait mode if enabled before executing the WAIT instruction. Therefore, an enabled KBI pin (KBIPEx = 1) can be used to bring the MCU out of wait mode if the KBI interrupt is enabled (KBIE = 1).

7.2.1.2 KBI in Stop Modes

The KBI operates asynchronously in stop3 mode if enabled before executing the STOP instruction. Therefore, an enabled KBI pin (KBIPEx = 1) can be used to bring the MCU out of stop3 and stop 2 mode if the KBI interrupt is enabled (KBIE = 1).

During stop2 mode, the KBI is disabled. Upon wake-up from stop2 mode, the KBI module will be in the reset state.

7.2.1.3 KBI in Active Background Mode

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

7.2.2 Block Diagram

The block diagram for the keyboard interrupt module is shown Figure 7-2.

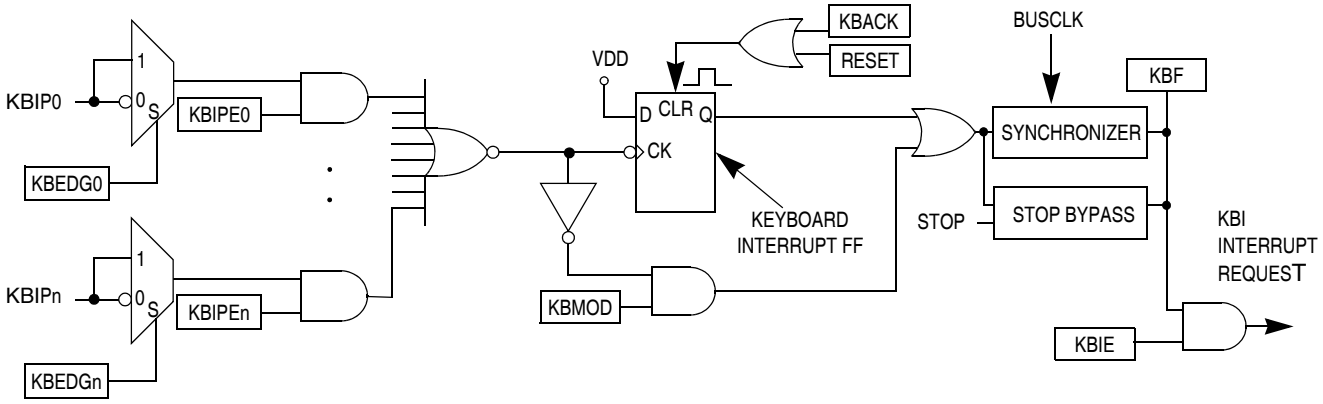


Figure 7-2. Keyboard Interrupt (KBI) Block Diagram

7.3 External Signal Description

The KBI input pins can be used to detect either falling edges, or both falling edge and low level interrupt requests. The KBI input pins can also be used to detect either rising edges, or both rising edge and high level interrupt requests.

Table 7-1. KBI Pin Mapping

Port pin	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
KBI pin	KBIP7	KBIP6	KBIP75	KBIP4	KBIP3	KBIP2	KBIP1	KBIP0

7.4 Register Definition

The KBI includes three registers:

- An 8-bit pin status and control register.
- An 8-bit pin enable register.
- An 8-bit edge select register.

Refer to the direct-page register summary in the Memory chapter for the absolute address assignments for all KBI registers. This section refers to registers and control bits only by their names and relative address offsets.

7.4.1 KBI Interrupt Status and Control Register (KBISC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	KBF	0	KBIE	KBIMOD
W						KBACK		
Reset:	0	0	0	0	0	0	0	0

Figure 7-3. KBI Interrupt Status and Control Register (KBISC)

Table 7-2. KBISC Register Field Descriptions

Field	Description
3 KBF	**KBI Interrupt Flag** — KBF indicates when a KBI interrupt is detected. Writes have no effect on KBF. 0 No KBI interrupt detected. 1 KBI interrupt detected.
2 KBACK	**KBI Interrupt Acknowledge** — Writing a 1 to KBACK is part of the flag clearing mechanism. KBACK always reads as 0.

Table 7-2. KBISC Register Field Descriptions

Field	Description
1 KBIE	**KBI Interrupt Enable** — KBIE determines whether a KBI interrupt is requested. 0 KBI interrupt request not enabled. 1 KBI interrupt request enabled.
0 KBIMOD	**KBI Detection Mode** — KBIMOD (along with the KBIES bits) controls the detection mode of the KBI interrupt pins. 0 KBI pins detect edges only. 1 KBI pins detect both edges and levels.

7.4.2 KBI Interrupt Pin Select Register (KBIPE)

	7	6	5	4	3	2	1	0
R W	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
Reset:	0	0	0	0	0	0	0	0

Figure 7-4. KBI Interrupt Pin Select Register (KBIPE)

Table 7-3. KBIPE Register Field Descriptions

Field	Description
7:0 KBIPE[7:0]	**KBI Interrupt Pin Selects** — Each of the KBIPEn bits enable the corresponding KBI interrupt pin. 0 Pin not enabled as interrupt. 1 Pin enabled as interrupt.

7.4.3 KBI Interrupt Edge Select Register (KBIES)

	7	6	5	4	3	2	1	0
R W	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBEDG3	KBEDG2	KBEDG1	KBEDG0
Reset:	0	0	0	0	0	0	0	0

Figure 7-5. KBI Edge Select Register (KBIES)

Table 7-4. KBIES Register Field Descriptions

Field	Description
7:0 KBEDG[7:0]	**KBI Edge Selects** — Each of the KBEDGn bits serves a dual purpose by selecting the polarity of the active interrupt edge as well as selecting a pullup or pulldown device if enabled. 0 A pullup device is connected to the associated pin and detects falling edge/low level for interrupt generation. 1 A pulldown device is connected to the associated pin and detects rising edge/high level for interrupt generation.

7.5 Functional Description

This on-chip peripheral module is called a keyboard interrupt (KBI) module because originally it was designed to simplify the connection and use of row-column matrices of keyboard switches. However, these inputs are also useful as extra external interrupt inputs and as an external means of waking the MCU from stop or wait low-power modes. The KBI module allows up to eight pins to act as additional interrupt sources.

Writing to the KBIPEn bits in the keyboard interrupt pin enable register (KBIPE) independently enables or disables each port pin. Each port can be configured as edge sensitive or edge and level sensitive based on the KBIMOD bit in the keyboard interrupt status and control register (KBISC). Edge sensitivity can be software programmed to be either falling or rising; the level can be either low or high. The polarity of the edge or edge and level sensitivity is selected using the KBEDGn bits in the keyboard interrupt edge select register (KBIES).

Synchronous logic is used to detect edges. Prior to detecting an edge, enabled port inputs must be at the deasserted logic level. A falling edge is detected when an enabled port input signal is seen as a logic 1 (the deasserted level) during one bus cycle and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input signal is seen as a logic 0 during one bus cycle and then a logic 1 during the next cycle.

7.5.1 Edge-Only Sensitivity

A valid edge on an enabled port pin will set KBF in KBISC. If KBIE in KBISC is set, an interrupt request will be presented to the CPU. Clearing of KBF is accomplished by writing a 1 to KBACK in KBISC.

7.5.2 Edge and Level Sensitivity

A valid edge or level on an enabled port pin will set KBF in KBISC. If KBIE in KBISC is set, an interrupt request will be presented to the CPU. Clearing of KBF is accomplished by writing a 1 to KBACK in KBISC provided all enabled port inputs are at their deasserted levels. KBF will remain set if any enabled port pin is asserted while attempting to clear by writing a 1 to KBACK.

7.5.3 Pullup/Pulldown Resistors

The keyboard interrupt 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).

7.5.4 Keyboard Interrupt Initialization

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

1. Mask interrupts by clearing KBIE in KBISC.
2. Select the pin polarity by setting the appropriate KBEDGn bits in KBIES.
3. If using internal pullup/pulldown device, configure the associated pull enable bits in KBIPE.

4. Enable the interrupt 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.

Chapter 8
Central Processor Unit (S08CPUV4)

8.1 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).

8.1.1 Features

Features of the HCS08 CPU include:

- Object code fully upward-compatible with M68HC05 and M68HC08 Families
- 64-KB CPU address space with banked memory management unit for greater than 64 KB
- 16-bit stack pointer (any size stack anywhere in 64-KB CPU 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

8.2 Programmer's Model and CPU Registers

Figure 8-1 shows the five CPU registers. CPU registers are not part of the memory map.

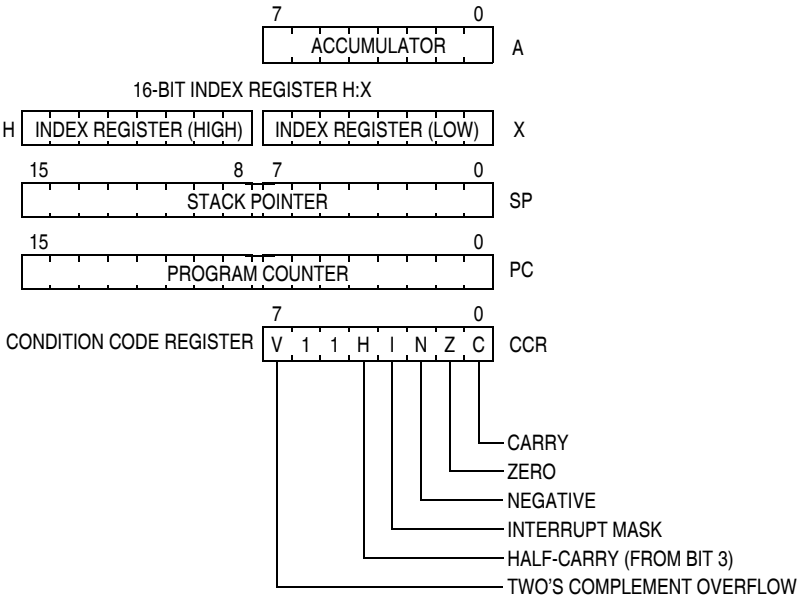


Figure 8-1. CPU Registers

8.2.1 Accumulator (A)

The A accumulator is a general-purpose 8-bit register. One operand input to the arithmetic logic unit (ALU) is connected to the accumulator and the ALU results are often stored into the A accumulator after arithmetic and logical operations. The accumulator can be loaded from memory using various addressing modes to specify the address where the loaded data comes from, or the contents of A can be stored to memory using various addressing modes to specify the address where data from A will be stored.

Reset has no effect on the contents of the A accumulator.

8.2.2 Index Register (H:X)

This 16-bit register is actually two separate 8-bit registers (H and X), which often work together as a 16-bit address pointer where H holds the upper byte of an address and X holds the lower byte of the address. All indexed addressing mode instructions use the full 16-bit value in H:X as an index reference pointer; however, for compatibility with the earlier M68HC05 Family, some instructions operate only on the low-order 8-bit half (X).

Many instructions treat X as a second general-purpose 8-bit register that can be used to hold 8-bit data values. X can be cleared, incremented, decremented, complemented, negated, shifted, or rotated. Transfer instructions allow data to be transferred from A or transferred to A where arithmetic and logical operations can then be performed.

For compatibility with the earlier M68HC05 Family, H is forced to 0x00 during reset. Reset has no effect on the contents of X.

8.2.3 Stack Pointer (SP)

This 16-bit address pointer register points at the next available location on the automatic last-in-first-out (LIFO) stack. The stack may be located anywhere in the 64-Kbyte address space that has RAM and can be any size up to the amount of available RAM. The stack is used to automatically save the return address for subroutine calls, the return address and CPU registers during interrupts, and for local variables. The AIS (add immediate to stack pointer) instruction adds an 8-bit signed immediate value to SP. This is most often used to allocate or deallocate space for local variables on the stack.

SP is forced to 0x00FF at reset for compatibility with the earlier M68HC05 Family. HCS08 programs normally change the value in SP to the address of the last location (highest address) in on-chip RAM during reset initialization to free up direct page RAM (from the end of the on-chip registers to 0x00FF).

The RSP (reset stack pointer) instruction was included for compatibility with the M68HC05 Family and is seldom used in new HCS08 programs because it only affects the low-order half of the stack pointer.

8.2.4 Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next instruction or operand to be fetched.

During normal program execution, the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, interrupt, and return operations load the program counter with an address other than that of the next sequential location. This is called a change-of-flow.

During reset, the program counter is loaded with the reset vector that is located at 0xFFFE and 0xFFFF. The vector stored there is the address of the first instruction that will be executed after exiting the reset state.

8.2.5 Condition Code Register (CCR)

The 8-bit condition code register contains the interrupt mask (I) and five flags that indicate the results of the instruction just executed. Bits 6 and 5 are set permanently to 1. The following paragraphs describe the functions of the condition code bits in general terms. For a more detailed explanation of how each instruction sets the CCR bits, refer to the *HCS08 Family Reference Manual, volume 1,* Freescale Semiconductor document order number HCS08RMv1.

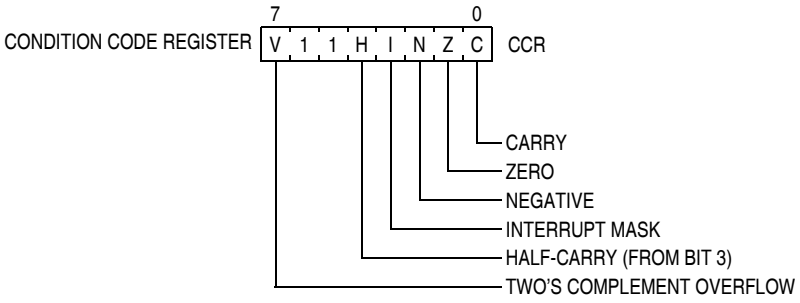


Figure 8-2. Condition Code Register

Table 8-1. 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

8.3 Addressing Modes

Addressing modes define the way the CPU accesses operands and data. In the HCS08, memory, status and control registers, and input/output (I/O) ports share a single 64-Kbyte CPU address space. This arrangement means that the same instructions that access variables in RAM can also be used to access I/O and control registers or nonvolatile program space.

NOTE

For more information about extended addressing modes, see the Memory Management Unit section in the Memory chapter.

MCU derivatives with more than 64-Kbytes of memory also include a memory management unit (MMU) to support extended memory space. A PPAGE register is used to manage 16-Kbyte pages of memory which can be accessed by the CPU through a 16-Kbyte window from 0x8000 through 0xBFFF. The CPU includes two special instructions (CALL and RTC). CALL operates like the JSR instruction except that CALL saves the current PPAGE value on the stack and provides a new PPAGE value for the destination. RTC works like the RTS instruction except RTC restores the old PPAGE value in addition to the PC during the return from the called routine. The MMU also includes a linear address pointer register and data access registers so that the extended memory space operates as if it was a single linear block of memory. For additional information about the MMU, refer to the Memory chapter of this data sheet.

Some instructions use more than one addressing mode. For instance, move instructions use one addressing mode to specify the source operand and a second addressing mode to specify the destination address. Instructions such as BRCLR, BRSET, CBEQ, and DBNZ use one addressing mode to specify the location of an operand for a test and then use relative addressing mode to specify the branch destination address when the tested condition is true. For BRCLR, BRSET, CBEQ, and DBNZ, the addressing mode listed in the instruction set tables is the addressing mode needed to access the operand to be tested, and relative addressing mode is implied for the branch destination.

8.3.1 Inherent Addressing Mode (INH)

In this addressing mode, operands needed to complete the instruction (if any) are located within CPU registers so the CPU does not need to access memory to get any operands.

8.3.2 Relative Addressing Mode (REL)

Relative addressing mode is used to specify the destination location for branch instructions. A signed 8-bit offset value is located in the memory location immediately following the opcode. During execution, if the branch condition is true, the signed offset is sign-extended to a 16-bit value and is added to the current contents of the program counter, which causes program execution to continue at the branch destination address.

8.3.3 Immediate Addressing Mode (IMM)

In immediate addressing mode, the operand needed to complete the instruction is included in the object code immediately following the instruction opcode in memory. In the case of a 16-bit immediate operand,

the high-order byte is located in the next memory location after the opcode, and the low-order byte is located in the next memory location after that.

8.3.4 Direct Addressing Mode (DIR)

In direct addressing mode, the instruction includes the low-order eight bits of an address in the direct page (0x0000–0x00FF). During execution a 16-bit address is formed by concatenating an implied 0x00 for the high-order half of the address and the direct address from the instruction to get the 16-bit address where the desired operand is located. This is faster and more memory efficient than specifying a complete 16-bit address for the operand.

8.3.5 Extended Addressing Mode (EXT)

In extended addressing mode, the full 16-bit address of the operand is located in the next two bytes of program memory after the opcode (high byte first).

8.3.6 Indexed Addressing Mode

Indexed addressing mode has seven variations including five that use the 16-bit H:X index register pair and two that use the stack pointer as the base reference.

8.3.6.1 Indexed, No Offset (IX)

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

8.3.6.2 Indexed, No Offset with Post Increment (IX+)

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

8.3.6.3 Indexed, 8-Bit Offset (IX1)

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

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

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

8.3.6.5 Indexed, 16-Bit Offset (IX2)

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

8.3.6.6 SP-Relative, 8-Bit Offset (SP1)

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

8.3.6.7 SP-Relative, 16-Bit Offset (SP2)

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

8.4 Special Operations

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

8.4.1 Reset Sequence

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

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

8.4.2 Interrupt Sequence

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

The CPU sequence for an interrupt is:

1. Store the contents of PCL, PCH, X, A, and CCR on the stack, in that order.
2. Set the I bit in the CCR.

3. Fetch the high-order half of the interrupt vector.
4. Fetch the low-order half of the interrupt vector.
5. Delay for one free bus cycle.
6. Fetch three bytes of program information starting at the address indicated by the interrupt vector to fill the instruction queue in preparation for execution of the first instruction in the interrupt service routine.

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

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

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

8.4.3 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.

8.4.4 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.

8.4.5 BGND Instruction

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

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

The CALL is similar to a jump-to-subroutine (JSR) instruction, but the subroutine that is called can be located anywhere in the normal 64-Kbyte address space or on any page of program expansion memory. When CALL is executed, a return address is calculated, then it and the current program page register value are stacked, and a new instruction-supplied value is written to PPAGE. The PPAGE value controls which of the possible 16-Kbyte pages is visible through the window in the 64-Kbyte memory map. Execution continues at the address of the called subroutine.

The actual sequence of operations that occur during execution of CALL is:

1. CPU calculates the address of the next instruction after the CALL instruction (the return address) and pushes this 16-bit value onto the stack, low byte first.
2. CPU reads the old PPAGE value and pushes it onto the stack.
3. CPU writes the new instruction-supplied page select value to PPAGE. This switches the destination page into the program overlay window in the CPU address range 0x8000 0xBFFF.
4. Instruction queue is refilled starting from the destination address, and execution begins at the new address.

This sequence of operations is an uninterruptable CPU instruction. There is no need to inhibit interrupts during CALL execution. In addition, a CALL can be performed from any address in memory to any other address. This is a big improvement over other bank-switching schemes, where the page switch operation can be performed only by a program outside the overlay window.

For all practical purposes, the PPAGE value supplied by the instruction can be considered to be part of the effective address. The new page value is provided by an immediate operand in the instruction.

The RTC instruction is used to terminate subroutines invoked by a CALL instruction. RTC unstacks the PPAGE value and the return address, the queue is refilled, and execution resumes with the next instruction after the corresponding CALL.

The actual sequence of operations that occur during execution of RTC is:

1. The return value of the 8-bit PPAGE register is pulled from the stack.
2. The 16-bit return address is pulled from the stack and loaded into the PC.
3. The return PPAGE value is written to the PPAGE register.
4. The queue is refilled and execution begins at the new address.

Since the return operation is implemented as a single uninterruptable CPU instruction, the RTC can be executed from anywhere in memory, including from a different page of extended memory in the overlay window.

The CALL and RTC instructions behave like JSR and RTS, except they have slightly longer execution times. Since extra execution cycles are required, routinely substituting CALL/RTC for JSR/RTS is not recommended. JSR and RTS can be used to access subroutines that are located outside the program overlay window or on the same memory page. However, if a subroutine can be called from other pages, it must be terminated with an RTC. In this case, since RTC unstacks the PPAGE value as well as the return address, all accesses to the subroutine, even those made from the same page, must use CALL instructions.

8.5 HCS08 Instruction Set Summary

Table 8-2 provides a summary of the HCS08 instruction set in all possible addressing modes. The table shows operand construction, execution time in internal bus clock cycles, and cycle-by-cycle details for each addressing mode variation of each instruction.

Table 8-2. Instruction Set Summary (Sheet 1 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 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 ← (A) + (M) + (C)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A9 ii B9 dd C9 hh ll D9 ee ff E9 ff F9 9E D9 ee ff 9E E9 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↕ 1 1 ↕	– ↕ ↕ ↕

ADD #*opr8i*
ADD *opr8a*
ADD *opr16a*
ADD *oprx16*,X
ADD *oprx8*,X
ADD ,X
ADD *oprx16*,SP
ADD *oprx8*,SP
	Add without Carry
	A ← (A) + (M)
		IMM
		DIR
		EXT
		IX2
		IX1
		IX
		SP2
		SP1
			AB ii
			BB dd
			CB hh ll
			DB ee ff
			EB ff
			FB
			9E DB ee ff
			9E EB ff
				2
				3
				4
				4
				3
				3
				5
				4
					pp
					rpp
					prpp
					prpp
					rpp
					rfp
					pprpp
					prpp
						↕ 1 1 ↕	– ↕ ↕ ↕

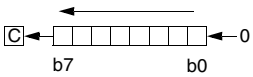
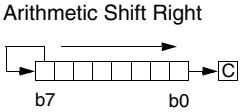
AIS #*opr8i*	Add Immediate Value (Signed) to Stack Pointer SP ← (SP) + (M)	IMM	A7 ii	2	pp	– 1 1 –	– – – –
AIX #*opr8i*	Add Immediate Value (Signed) to Index Register (H:X) H:X ← (H:X) + (M)	IMM	AF ii	2	pp	– 1 1 –	– – – –
AND #*opr8i* AND *opr8a* AND *opr16a* AND *oprx16*,X AND *oprx8*,X AND ,X AND *oprx16*,SP AND *oprx8*,SP	Logical AND A ← (A) & (M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A4 ii B4 dd C4 hh ll D4 ee ff E4 ff F4 9E D4 ee ff 9E E4 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 –	– ↕ ↕ –
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 dd 48 58 68 ff 78 9E 68 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↕ 1 1 –	– ↕ ↕ ↕
ASR *opr8a* ASRA ASRX ASR *oprx8*,X ASR ,X ASR *oprx8*,SP	Arithmetic Shift Right 	DIR INH INH IX1 IX SP1	37 dd 47 57 67 ff 77 9E 67 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↕ 1 1 –	– ↕ ↕ ↕

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

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 H	I N Z C
BCC *rel*	Branch if Carry Bit Clear (if C = 0)	REL	24 rr	3	ppp	– 1 1 –	– – – –
BCLR *n,opr8a*	Clear Bit n in Memory (Mn ← 0)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 dd 13 dd 15 dd 17 dd 19 dd 1B dd 1D dd 1F dd	5 5 5 5 5 5 5 5	rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp	– 1 1 –	– – – –
BCS *rel*	Branch if Carry Bit Set (if C = 1) (Same as BLO)	REL	25 rr	3	ppp	– 1 1 –	– – – –
BEQ *rel*	Branch if Equal (if Z = 1)	REL	27 rr	3	ppp	– 1 1 –	– – – –
BGE *rel*	Branch if Greater Than or Equal To (if N ⊕ V = 0) (Signed)	REL	90 rr	3	ppp	– 1 1 –	– – – –
BGND	Enter active background if ENBDM=1 Waits for and processes BDM commands until GO, TRACE1, or TAGGO	INH	82	5+	fp...ppp	– 1 1 –	– – – –
BGT *rel*	Branch if Greater Than (if Z | (N ⊕ V) = 0) (Signed)	REL	92 rr	3	ppp	– 1 1 –	– – – –
BHCC *rel*	Branch if Half Carry Bit Clear (if H = 0)	REL	28 rr	3	ppp	– 1 1 –	– – – –
BHCS *rel*	Branch if Half Carry Bit Set (if H = 1)	REL	29 rr	3	ppp	– 1 1 –	– – – –
BHI *rel*	Branch if Higher (if C | Z = 0)	REL	22 rr	3	ppp	– 1 1 –	– – – –
BHS *rel*	Branch if Higher or Same (if C = 0) (Same as BCC)	REL	24 rr	3	ppp	– 1 1 –	– – – –
BIH *rel*	Branch if IRQ Pin High (if IRQ pin = 1)	REL	2F rr	3	ppp	– 1 1 –	– – – –
BIL *rel*	Branch if IRQ Pin Low (if IRQ pin = 0)	REL	2E rr	3	ppp	– 1 1 –	– – – –
BIT #*opr8i* BIT *opr8a* BIT *opr16a* BIT *oprx16*,X BIT *oprx8*,X BIT ,X BIT *oprx16*,SP BIT *oprx8*,SP	Bit Test (A) & (M) (CCR Updated but Operands Not Changed)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A5 ii B5 dd C5 hh ll D5 ee ff E5 ff F5 9E D5 ee ff 9E E5 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 –	– ↕ ↕ –
BLE *rel*	Branch if Less Than or Equal To (if Z | (N ⊕ V) = 1) (Signed)	REL	93 rr	3	ppp	– 1 1 –	– – – –
BLO *rel*	Branch if Lower (if C = 1) (Same as BCS)	REL	25 rr	3	ppp	– 1 1 –	– – – –
BLS *rel*	Branch if Lower or Same (if C | Z = 1)	REL	23 rr	3	ppp	– 1 1 –	– – – –
BLT *rel*	Branch if Less Than (if N ⊕ V = 1) (Signed)	REL	91 rr	3	ppp	– 1 1 –	– – – –
BMC *rel*	Branch if Interrupt Mask Clear (if I = 0)	REL	2C rr	3	ppp	– 1 1 –	– – – –
BMI *rel*	Branch if Minus (if N = 1)	REL	2B rr	3	ppp	– 1 1 –	– – – –
BMS *rel*	Branch if Interrupt Mask Set (if I = 1)	REL	2D rr	3	ppp	– 1 1 –	– – – –
BNE *rel*	Branch if Not Equal (if Z = 0)	REL	26 rr	3	ppp	– 1 1 –	– – – –

Table 8-2. Instruction Set Summary (Sheet 3 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 H	I N Z C
BPL *rel*	Branch if Plus (if N = 0)	REL	2A rr	3	ppp	– 1 1 –	– – – –
BRA *rel*	Branch Always (if I = 1)	REL	20 rr	3	ppp	– 1 1 –	– – – –
BRCLR *n*,*oprx8*,*rel*	Branch if Bit *n* in Memory Clear (if (Mn) = 0)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 dd rr 03 dd rr 05 dd rr 07 dd rr 09 dd rr 0B dd rr 0D dd rr 0F dd rr	5 5 5 5 5 5 5 5	rpppp rpppp rpppp rpppp rpppp rpppp rpppp rpppp	– 1 1 –	– – – ↕
BRN *rel*	Branch Never (if I = 0)	REL	21 rr	3	ppp	– 1 1 –	– – – –
BRSET *n*,*oprx8*,*rel*	Branch if Bit *n* in Memory Set (if (Mn) = 1)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 dd rr 02 dd rr 04 dd rr 06 dd rr 08 dd rr 0A dd rr 0C dd rr 0E dd rr	5 5 5 5 5 5 5 5	rpppp rpppp rpppp rpppp rpppp rpppp rpppp rpppp	– 1 1 –	– – – ↕
BSET *n*,*oprx8*	Set Bit *n* in Memory (Mn ← 1)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 dd 12 dd 14 dd 16 dd 18 dd 1A dd 1C dd 1E dd	5 5 5 5 5 5 5 5	rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp	– 1 1 –	– – – –
BSR *rel*	Branch to Subroutine PC ← (PC) + $0002 push (PCL); SP ← (SP) – $0001 push (PCH); SP ← (SP) – $0001 PC ← (PC) + *rel*	REL	AD rr	5	ssppp	– 1 1 –	– – – –
CALL page, opr16a	Call Subroutine	EXT	AC pg hhll	8	ppssspppp	– 1 1 –	– – – –
CBEQ *oprx8*,*rel* CBEQA #*opr8i*,*rel* CBEQX #*opr8i*,*rel* CBEQ *oprx8*,X+,*rel* CBEQ ,X+,*rel* CBEQ *oprx8*,SP,*rel*	Compare and... Branch if (A) = (M) Branch if (A) = (M) Branch if (X) = (M) Branch if (A) = (M) Branch if (A) = (M) Branch if (A) = (M)	DIR IMM IMM IX1+ IX+ SP1	31 dd rr 41 ii rr 51 ii rr 61 ff rr 71 rr 9E 61 ff rr	5 4 4 5 5 6	rpppp pppp pppp rpppp rfppp prpppp	– 1 1 –	– – – –
CLC	Clear Carry Bit (C ← 0)	INH	98	1	p	– 1 1 –	– – – 0
CLI	Clear Interrupt Mask Bit (I ← 0)	INH	9A	1	p	– 1 1 –	0 – – –
CLR *oprx8* CLRA CLRX CLRH CLR *oprx8*,X CLR ,X CLR *oprx8*,SP	Clear M ← $00 A ← $00 X ← $00 H ← $00 M ← $00 M ← $00 M ← $00	DIR INH INH INH IX1 IX SP1	3F dd 4F 5F 8C 6F ff 7F 9E 6F ff	5 1 1 1 5 4 6	rfwpp p p p rfwpp rfwp prfwpp	0 1 1 –	– 0 1 –

Table 8-2. Instruction Set Summary (Sheet 4 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 H	I N Z C

CMP #*oprx8i*
CMP *oprx8a*
CMP *oprx16a*
CMP *oprx16*,X
CMP *oprx8*,X
CMP ,X
CMP *oprx16*,SP
CMP *oprx8*,SP
	Compare Accumulator with Memory
	A – M
	(CCR Updated But Operands Not Changed)
		IMM
		DIR
		EXT
		IX2
		IX1
		IX
		SP2
		SP1
			A1 ii
			B1 dd
			C1 hh ll
			D1 ee ff
			E1 ff
			F1
			9E D1 ee ff
			9E E1 ff
				2
				3
				4
				4
				3
				3
				5
				4
					pp
					rpp
					prpp
					prpp
					rpp
					rfp
					pprpp
					prpp
						↕ 1 1 –	– ↕ ↕ ↕

COM *oprx8a* COMA COMX COM *oprx8*,X COM ,X COM *oprx8*,SP	Complement M ← ($\overline{M}$)= $FF – (M) (One's Complement) A ← ($\overline{A}$) = $FF – (A) X ← ($\overline{X}$) = $FF – (X) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M)	DIR INH INH IX1 IX SP1	33 dd 43 53 63 ff 73 9E 63 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	0 1 1 –	– ↕ ↕ 1
CPHX *oprx16a* CPHX #*oprx16i* CPHX *oprx8a* CPHX *oprx8*,SP	Compare Index Register (H:X) with Memory (H:X) – (M:M + $0001) (CCR Updated But Operands Not Changed)	EXT IMM DIR SP1	3E hh ll 65 jj kk 75 dd 9E F3 ff	6 3 5 6	prrfpp ppp rrfpp prrfpp	↕ 1 1 –	– ↕ ↕ ↕
CPX #*oprx8i* CPX *oprx8a* CPX *oprx16a* CPX *oprx16*,X CPX *oprx8*,X CPX ,X CPX *oprx16*,SP CPX *oprx8*,SP	Compare X (Index Register Low) with Memory X – M (CCR Updated But Operands Not Changed)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A3 ii B3 dd C3 hh ll D3 ee ff E3 ff F3 9E D3 ee ff 9E E3 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↕ 1 1 –	– ↕ ↕ ↕
DAA	Decimal Adjust Accumulator After ADD or ADC of BCD Values	INH	72	1	p	U 1 1 –	– ↕ ↕ ↕
DBNZ *oprx8a*,*rel* DBNZA *rel* DBNZX *rel* DBNZ *oprx8*,X,*rel* DBNZ ,X,*rel* DBNZ *oprx8*,SP,*rel*	Decrement A, X, or M and Branch if Not Zero (if (result) ≠ 0) DBNZX Affects X Not H	DIR INH INH IX1 IX SP1	3B dd rr 4B rr 5B rr 6B ff rr 7B rr 9E 6B ff rr	7 4 4 7 6 8	rfwpppp fppp fppp rfwpppp rfwppp prfwpppp	– 1 1 –	– – – –
DEC *oprx8a* DECA DECX DEC *oprx8*,X DEC ,X DEC *oprx8*,SP	Decrement M ← (M) – $01 A ← (A) – $01 X ← (X) – $01 M ← (M) – $01 M ← (M) – $01 M ← (M) – $01	DIR INH INH IX1 IX SP1	3A dd 4A 5A 6A ff 7A 9E 6A ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↕ 1 1 –	– ↕ ↕ –
DIV	Divide A ← (H:A)÷(X); H ← Remainder	INH	52	6	ffffffp	– 1 1 –	– – ↕ ↕
EOR #*oprx8i* EOR *oprx8a* EOR *oprx16a* EOR *oprx16*,X EOR *oprx8*,X EOR ,X EOR *oprx16*,SP EOR *oprx8*,SP	Exclusive OR Memory with Accumulator A ← (A ⊕ M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A8 ii B8 dd C8 hh ll D8 ee ff E8 ff F8 9E D8 ee ff 9E E8 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 –	– ↕ ↕ –

Table 8-2. Instruction Set Summary (Sheet 5 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR V 1 1 H	Affect on CCR I N Z C
INC *opr8a* INCA INCX INC *oprx8*,X INC ,X INC *oprx8*,SP	Increment M ← (M) + $01 A ← (A) + $01 X ← (X) + $01 M ← (M) + $01 M ← (M) + $01 M ← (M) + $01	DIR INH INH IX1 IX SP1	3C dd 4C 5C 6C ff 7C 9E 6C ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↕ 1 1 –	– ↕ ↕ –
JMP *opr8a* JMP *opr16a* JMP *oprx16*,X JMP *oprx8*,X JMP ,X	Jump PC ← Jump Address	DIR EXT IX2 IX1 IX	BC dd CC hh ll DC ee ff EC ff FC	3 4 4 3 3	ppp pppp pppp ppp ppp	– 1 1 –	– – – –
JSR *opr8a* JSR *opr16a* JSR *oprx16*,X JSR *oprx8*,X JSR ,X	Jump to Subroutine PC ← (PC) + *n* (*n* = 1, 2, or 3) Push (PCL); SP ← (SP) – $0001 Push (PCH); SP ← (SP) – $0001 PC ← Unconditional Address	DIR EXT IX2 IX1 IX	BD dd CD hh ll DD ee ff ED ff FD	5 6 6 5 5	ssppp pssppp pssppp ssppp ssppp	– 1 1 –	– – – –

LDA #*opr8i*
LDA *opr8a*
LDA *opr16a*
LDA *oprx16*,X
LDA *oprx8*,X
LDA ,X
LDA *oprx16*,SP
LDA *oprx8*,SP
	Load Accumulator from Memory
	A ← (M)
		IMM
		DIR
		EXT
		IX2
		IX1
		IX
		SP2
		SP1
			A6 ii
			B6 dd
			C6 hh ll
			D6 ee ff
			E6 ff
			F6
			9E D6 ee ff
			9E E6 ff
				2
				3
				4
				4
				3
				3
				5
				4
					pp
					rpp
					prpp
					prpp
					rpp
					rfp
					pprpp
					prpp
						0 1 1 –	– ↕ ↕ –

LDHX #*opr16i*
LDHX *opr8a*
LDHX *opr16a*
LDHX ,X
LDHX *oprx16*,X
LDHX *oprx8*,X
LDHX *oprx8*,SP
	Load Index Register (H:X)
	H:X ← (M:M + $0001)
		IMM
		DIR
		EXT
		IX
		IX2
		IX1
		SP1
			45 jj kk
			55 dd
			32 hh ll
			9E AE
			9E BE ee ff
			9E CE ff
			9E FE ff
				3
				4
				5
				5
				6
				5
				5
					ppp
					rrpp
					prrpp
					prrfp
					pprrpp
					prrpp
					prrpp
						0 1 1 –	– ↕ ↕ –

LDX #*opr8i*
LDX *opr8a*
LDX *opr16a*
LDX *oprx16*,X
LDX *oprx8*,X
LDX ,X
LDX *oprx16*,SP
LDX *oprx8*,SP
	Load X (Index Register Low) from Memory
	X ← (M)
		IMM
		DIR
		EXT
		IX2
		IX1
		IX
		SP2
		SP1
			AE ii
			BE dd
			CE hh ll
			DE ee ff
			EE ff
			FE
			9E DE ee ff
			9E EE ff
				2
				3
				4
				4
				3
				3
				5
				4
					pp
					rpp
					prpp
					prpp
					rpp
					rfp
					pprpp
					prpp
						0 1 1 –	– ↕ ↕ –

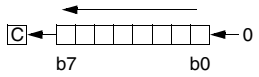
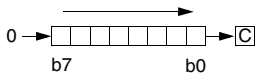
LSL *opr8a* LSLA LSLX LSL *oprx8*,X LSL ,X LSL *oprx8*,SP	Logical Shift Left  (Same as ASL)	DIR INH INH IX1 IX SP1	38 dd 48 58 68 ff 78 9E 68 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↕ 1 1 –	– ↕ ↕ ↕
LSR *opr8a* LSRA LSR*X* LSR *oprx8*,X LSR ,X LSR *oprx8*,SP	Logical Shift Right  	DIR INH INH IX1 IX SP1	34 dd 44 54 64 ff 74 9E 64 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↕ 1 1 –	– 0 ↕ ↕

Table 8-2. Instruction Set Summary (Sheet 6 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR V 1 1 H	Affect on CCR I N Z C

MOV *oprx8*,*oprx8*
MOV *oprx8*,X+
MOV #*oprx8*,*oprx8*
MOV ,X+,*oprx8*
	Move
	$(M)_{destination} \leftarrow (M)_{source}$
	In IX+/DIR and DIR/IX+ Modes,
	H:X ← (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 1 1 –	– ↕ ↕ –

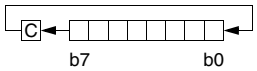
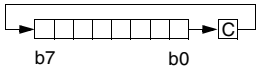
MUL	Unsigned multiply X:A ← (X) × (A)	INH	42	5	ffffp	– 1 1 0	– – – 0
NEG *oprx8* NEGA NEGX NEG *oprx8*,X NEG ,X NEG *oprx8*,SP	Negate M ← – (M) = $00 – (M) (Two's Complement) A ← – (A) = $00 – (A) X ← – (X) = $00 – (X) M ← – (M) = $00 – (M) M ← – (M) = $00 – (M) M ← – (M) = $00 – (M)	DIR INH INH IX1 IX SP1	30 dd 40 50 60 ff 70 9E 60 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↕ 1 1 –	– ↕ ↕ ↕
NOP	No Operation — Uses 1 Bus Cycle	INH	9D	1	p	– 1 1 –	– – – –
NSA	Nibble Swap Accumulator A ← (A[3:0]:A[7:4])	INH	62	1	p	– 1 1 –	– – – –
ORA #*oprx8* ORA *oprx8* ORA *oprx16* ORA *oprx16*,X ORA *oprx8*,X ORA ,X ORA *oprx16*,SP ORA *oprx8*,SP	Inclusive OR Accumulator and Memory A ← (A) | (M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	AA ii BA dd CA hh ll DA ee ff EA ff FA 9E DA ee ff 9E EA ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 –	– ↕ ↕ –
PSHA	Push Accumulator onto Stack Push (A); SP ← (SP) – $0001	INH	87	2	sp	– 1 1 –	– – – –
PSHH	Push H (Index Register High) onto Stack Push (H); SP ← (SP) – $0001	INH	8B	2	sp	– 1 1 –	– – – –
PSHX	Push X (Index Register Low) onto Stack Push (X); SP ← (SP) – $0001	INH	89	2	sp	– 1 1 –	– – – –
PULA	Pull Accumulator from Stack SP ← (SP + $0001); Pull (A)	INH	86	3	ufp	– 1 1 –	– – – –
PULH	Pull H (Index Register High) from Stack SP ← (SP + $0001); Pull (H)	INH	8A	3	ufp	– 1 1 –	– – – –
PULX	Pull X (Index Register Low) from Stack SP ← (SP + $0001); Pull (X)	INH	88	3	ufp	– 1 1 –	– – – –
ROL *oprx8* ROLA ROLX ROL *oprx8*,X ROL ,X ROL *oprx8*,SP	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	↕ 1 1 –	– ↕ ↕ ↕
ROR *oprx8* RORA RORX ROR *oprx8*,X ROR ,X ROR *oprx8*,SP	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	↕ 1 1 –	– ↕ ↕ ↕

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

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 H	I N Z C
RSP	Reset Stack Pointer (Low Byte) SPL ← $FF (High Byte Not Affected)	INH	9C	1	p	– 1 1 –	– – – –
RTC	Return from CALL	INH	8D	7	uuufppp	– 1 1 –	– – – –
RTI	Return from Interrupt SP ← (SP) + $0001; Pull (CCR) SP ← (SP) + $0001; Pull (A) SP ← (SP) + $0001; Pull (X) SP ← (SP) + $0001; Pull (PCH) SP ← (SP) + $0001; Pull (PCL)	INH	80	9	uuuuufppp	↕ 1 1 ↕	↕ ↕ ↕ ↕
RTS	Return from Subroutine SP ← SP + $0001; Pull (PCH) SP ← SP + $0001; Pull (PCL)	INH	81	5	ufppp	– 1 1 –	– – – –
SBC #*opr8i* SBC *opr8a* SBC *opr16a* SBC *oprx16*,X SBC *oprx8*,X SBC ,X SBC *oprx16*,SP SBC *oprx8*,SP	Subtract with Carry A ← (A) – (M) – (C)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A2 ii B2 dd C2 hh ll D2 ee ff E2 ff F2 9E D2 ee ff 9E E2 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↕ 1 1 –	– ↕ ↕ ↕
SEC	Set Carry Bit (C ← 1)	INH	99	1	p	– 1 1 –	– – – 1
SEI	Set Interrupt Mask Bit (I ← 1)	INH	9B	1	p	– 1 1 –	1 – – –
STA *opr8a* STA *opr16a* STA *oprx16*,X STA *oprx8*,X STA ,X STA *oprx16*,SP STA *oprx8*,SP	Store Accumulator in Memory M ← (A)	DIR EXT IX2 IX1 IX SP2 SP1	B7 dd C7 hh ll D7 ee ff E7 ff F7 9E D7 ee ff 9E E7 ff	3 4 4 3 2 5 4	wpp pwpp pwpp wpp wp ppwpp pwpp	0 1 1 –	– ↕ ↕ –
STHX *opr8a* STHX *opr16a* STHX *oprx8*,SP	Store H:X (Index Reg.) (M:M + $0001) ← (H:X)	DIR EXT SP1	35 dd 96 hh ll 9E FF ff	4 5 5	wwpp pwwpp pwwpp	0 1 1 –	– ↕ ↕ –
STOP	Enable Interrupts: Stop Processing Refer to MCU Documentation I bit ← 0; Stop Processing	INH	8E	2	fp...	– 1 1 –	0 – – –
STX *opr8a* STX *opr16a* STX *oprx16*,X STX *oprx8*,X STX ,X STX *oprx16*,SP STX *oprx8*,SP	Store X (Low 8 Bits of Index Register) in Memory M ← (X)	DIR EXT IX2 IX1 IX SP2 SP1	BF dd CF hh ll DF ee ff EF ff FF 9E DF ee ff 9E EF ff	3 4 4 3 2 5 4	wpp pwpp pwpp wpp wp ppwpp pwpp	0 1 1 –	– ↕ ↕ –

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

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 H	I N Z C

SUB #*opr8i*
SUB *opr8a*
SUB *opr16a*
SUB *oprx16*,X
SUB *oprx8*,X
SUB ,X
SUB *oprx16*,SP
SUB *oprx8*,SP
	Subtract
	A ← (A) – (M)
		IMM
		DIR
		EXT
		IX2
		IX1
		IX
		SP2
		SP1
			A0 ii
			B0 dd
			C0 hh ll
			D0 ee ff
			E0 ff
			F0
			9E D0 ee ff
			9E E0 ff
				2
				3
				4
				4
				3
				3
				5
				4
					pp
					rpp
					prpp
					prpp
					rpp
					rfp
					pprpp
					prpp
						↕ 1 1 –	– ↕ ↕ ↕

SWI	Software Interrupt PC ← (PC) + $0001 Push (PCL); SP ← (SP) – $0001 Push (PCH); SP ← (SP) – $0001 Push (X); SP ← (SP) – $0001 Push (A); SP ← (SP) – $0001 Push (CCR); SP ← (SP) – $0001 I ← 1; PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	INH	83	11	ssssvvfppp	– 1 1 –	1 – – –
TAP	Transfer Accumulator to CCR CCR ← (A)	INH	84	1	p	↕ 1 1 ↕	↕ ↕ ↕ ↕
TAX	Transfer Accumulator to X (Index Register Low) X ← (A)	INH	97	1	p	– 1 1 –	– – – –
TPA	Transfer CCR to Accumulator A ← (CCR)	INH	85	1	p	– 1 1 –	– – – –
TST *opr8a* TSTA TSTX TST *oprx8*,X TST ,X TST *oprx8*,SP	Test for Negative or Zero (M) – $00 (A) – $00 (X) – $00 (M) – $00 (M) – $00 (M) – $00	DIR INH INH IX1 IX SP1	3D dd 4D 5D 6D ff 7D 9E 6D ff	4 1 1 4 3 5	rfpp p p rfpp rfp prfpp	0 1 1 –	– ↕ ↕ –
TSX	Transfer SP to Index Reg. H:X ← (SP) + $0001	INH	95	2	fp	– 1 1 –	– – – –
TXA	Transfer X (Index Reg. Low) to Accumulator A ← (X)	INH	9F	1	p	– 1 1 –	– – – –

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

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 H	I N Z C
TXS	Transfer Index Reg. to SP SP ← (H:X) – $0001	INH	94	2	fp	– 1 1 –	– – – –
WAIT	Enable Interrupts; Wait for Interrupt I bit ← 0; Halt CPU	INH	8F	2+	fp...	– 1 1 –	0 – – –

Source Form: Everything in the source forms columns, *except expressions in italic characters*, is literal information which must appear in the assembly source file exactly as shown. The initial 3- to 5-letter mnemonic and the characters (# , ( ) and +) are always a 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 direct-page address ($00xx).
opr16a	Any label or expression that evaluates to a 16-bit address.
oprx8	Any label or expression that evaluates to an unsigned 8-bit value, used for indexed addressing.
oprx16	Any label or expression that evaluates to a 16-bit value, used for indexed addressing.
rel	Any label or expression that refers to an address that is within –128 to +127 locations from the start of the next instruction.

Operation Symbols:

A	Accumulator
CCR	Condition code register
H	Index register high byte
M	Memory location
n	Any bit
opr	Operand (one or two bytes)
PC	Program counter
PCH	Program counter high byte
PCL	Program counter low byte
rel	Relative program counter offset byte
SP	Stack pointer
SPL	Stack pointer low byte
X	Index register low byte
&	Logical AND
|	Logical OR
⊕	Logical EXCLUSIVE OR
( )	Contents of
+	Add
–	Subtract, Negation (two's complement)
×	Multiply
÷	Divide
#	Immediate value
←	Loaded with
:	Concatenated with

Addressing Modes:

DIR	Direct addressing mode
EXT	Extended addressing mode
IMM	Immediate addressing mode
INH	Inherent addressing mode
IX	Indexed, no offset addressing mode
IX1	Indexed, 8-bit offset addressing mode
IX2	Indexed, 16-bit offset addressing mode
IX+	Indexed, no offset, post increment addressing mode
IX1+	Indexed, 8-bit offset, post increment addressing mode
REL	Relative addressing mode
SP1	Stack pointer, 8-bit offset addressing mode
SP2	Stack pointer 16-bit offset addressing mode

Cycle-by-Cycle Codes:

f	Free cycle. This indicates a cycle where the CPU does not require use of the system buses. An f cycle is always one cycle of the system bus clock and is always a read cycle.
p	Program fetch; read from next consecutive location in program memory
r	Read 8-bit operand
s	Push (write) one byte onto stack
u	Pop (read) one byte from stack
v	Read vector from $FFxx (high byte first)
w	Write 8-bit operand

CCR Bits:

V	Overflow bit
H	Half-carry bit
I	Interrupt mask
N	Negative bit
Z	Zero bit
C	Carry/borrow bit

CCR Effects:

↕	Set or cleared
–	Not affected
U	Undefined

Table 8-3. Opcode Map (Sheet 1 of 2)

Bit-Manipulation		Branch	Read-Modify-Write					Control		Register/Memory					
00 5 BRSET0 3 DIR	10 5 BSET0 2 DIR	20 3 BRA 2 REL	30 5 NEG 2 DIR	40 1 NEGA 1 INH	50 1 NEGX 1 INH	60 5 NEG 2 IX1	70 4 NEG 1 IX	80 9 RTI 1 INH	90 3 BGE 2 REL	A0 2 SUB 2 IMM	B0 3 SUB 2 DIR	C0 4 SUB 3 EXT	D0 4 SUB 3 IX2	E0 3 SUB 2 IX1	F0 3 SUB 1 IX
01 5 BRCLR0 3 DIR	11 5 BCLR0 2 DIR	21 3 BRN 2 REL	31 5 CBEQ 3 DIR	41 4 CBEQA 3 IMM	51 4 CBEQX 3 IMM	61 5 CBEQ 3 IX1+	71 5 CBEQ 2 IX+	81 6 RTS 1 INH	91 3 BLT 2 REL	A1 2 CMP 2 IMM	B1 3 CMP 2 DIR	C1 4 CMP 3 EXT	D1 4 CMP 3 IX2	E1 3 CMP 2 IX1	F1 3 CMP 1 IX
02 5 BRSET1 3 DIR	12 5 BSET1 2 DIR	22 3 BHI 2 REL	32 5 LDHX 3 EXT	42 5 MUL 1 INH	52 6 DIV 1 INH	62 1 NSA 1 INH	72 1 DAA 1 INH	82 5+ BGND 1 INH	92 3 BGT 2 REL	A2 2 SBC 2 IMM	B2 3 SBC 2 DIR	C2 4 SBC 3 EXT	D2 4 SBC 3 IX2	E2 3 SBC 2 IX1	F2 3 SBC 1 IX
03 5 BRCLR1 3 DIR	13 5 BCLR1 2 DIR	23 3 BLS 2 REL	33 5 COM 2 DIR	43 1 COMA 1 INH	53 1 COMX 1 INH	63 5 COM 2 IX1	73 4 COM 1 IX	83 11 SWI 1 INH	93 3 BLE 2 REL	A3 2 CPX 2 IMM	B3 3 CPX 2 DIR	C3 4 CPX 3 EXT	D3 4 CPX 3 IX2	E3 3 CPX 2 IX1	F3 3 CPX 1 IX
04 5 BRSET2 3 DIR	14 5 BSET2 2 DIR	24 3 BCC 2 REL	34 5 LSR 2 DIR	44 1 LSRA 1 INH	54 1 LSRX 1 INH	64 5 LSR 2 IX1	74 4 LSR 1 IX	84 1 TAP 1 INH	94 2 TXS 1 INH	A4 2 AND 2 IMM	B4 3 AND 2 DIR	C4 4 AND 3 EXT	D4 4 AND 3 IX2	E4 3 AND 2 IX1	F4 3 AND 1 IX
05 5 BRCLR2 3 DIR	15 5 BCLR2 2 DIR	25 3 BCS 2 REL	35 4 STHX 2 DIR	45 3 LDHX 3 IMM	55 4 LDHX 2 DIR	65 3 CPHX 3 IMM	75 5 CPHX 2 DIR	85 1 TPA 1 INH	95 2 TSX 1 INH	A5 2 BIT 2 IMM	B5 3 BIT 2 DIR	C5 4 BIT 3 EXT	D5 4 BIT 3 IX2	E5 3 BIT 2 IX1	F5 3 BIT 1 IX
06 5 BRSET3 3 DIR	16 5 BSET3 2 DIR	26 3 BNE 2 REL	36 5 ROR 2 DIR	46 1 RORA 1 INH	56 1 RORX 1 INH	66 5 ROR 2 IX1	76 4 ROR 1 IX	86 3 PULA 1 INH	96 5 STHX 3 EXT	A6 2 LDA 2 IMM	B6 3 LDA 2 DIR	C6 4 LDA 3 EXT	D6 4 LDA 3 IX2	E6 3 LDA 2 IX1	F6 3 LDA 1 IX
07 5 BRCLR3 3 DIR	17 5 BCLR3 2 DIR	27 3 BEQ 2 REL	37 5 ASR 2 DIR	47 1 ASRA 1 INH	57 1 ASRX 1 INH	67 5 ASR 2 IX1	77 4 ASR 1 IX	87 2 PSHA 1 INH	97 1 TAX 1 INH	A7 2 AIS 2 IMM	B7 3 STA 2 DIR	C7 4 STA 3 EXT	D7 4 STA 3 IX2	E7 3 STA 2 IX1	F7 2 STA 1 IX
08 5 BRSET4 3 DIR	18 5 BSET4 2 DIR	28 3 BHCC 2 REL	38 5 LSL 2 DIR	48 1 LSLA 1 INH	58 1 LSLX 1 INH	68 5 LSL 2 IX1	78 4 LSL 1 IX	88 3 PULX 1 INH	98 1 CLC 1 INH	A8 2 EOR 2 IMM	B8 3 EOR 2 DIR	C8 4 EOR 3 EXT	D8 4 EOR 3 IX2	E8 3 EOR 2 IX1	F8 3 EOR 1 IX
09 5 BRCLR4 3 DIR	19 5 BCLR4 2 DIR	29 3 BHCS 2 REL	39 5 ROL 2 DIR	49 1 ROLA 1 INH	59 1 ROLX 1 INH	69 5 ROL 2 IX1	79 4 ROL 1 IX	89 2 PSHX 1 INH	99 1 SEC 1 INH	A9 2 ADC 2 IMM	B9 3 ADC 2 DIR	C9 4 ADC 3 EXT	D9 4 ADC 3 IX2	E9 3 ADC 2 IX1	F9 3 ADC 1 IX
0A 5 BRSET5 3 DIR	1A 5 BSET5 2 DIR	2A 3 BPL 2 REL	3A 5 DEC 2 DIR	4A 1 DECA 1 INH	5A 1 DECX 1 INH	6A 5 DEC 2 IX1	7A 4 DEC 1 IX	8A 3 PULH 1 INH	9A 1 CLI 1 INH	AA 2 ORA 2 IMM	BA 3 ORA 2 DIR	CA 4 ORA 3 EXT	DA 4 ORA 3 IX2	EA 3 ORA 2 IX1	FA 3 ORA 1 IX
0B 5 BRCLR5 3 DIR	1B 5 BCLR5 2 DIR	2B 3 BMI 2 REL	3B 7 DBNZ 3 DIR	4B 4 DBNZA 2 INH	5B 4 DBNZX 2 INH	6B 7 DBNZ 3 IX1	7B 6 DBNZ 2 IX	8B 2 PSHH 1 INH	9B 1 SEI 1 INH	AB 2 ADD 2 IMM	BB 3 ADD 2 DIR	CB 4 ADD 3 EXT	DB 4 ADD 3 IX2	EB 3 ADD 2 IX1	FB 3 ADD 1 IX
0C 5 BRSET6 3 DIR	1C 5 BSET6 2 DIR	2C 3 BMC 2 REL	3C 5 INC 2 DIR	4C 1 INCA 1 INH	5C 1 INCX 1 INH	6C 5 INC 2 IX1	7C 4 INC 1 IX	8C 1 CLRH 1 INH	9C 1 RSP 1 INH	AC 8 CALL 4 EXT	BC 3 JMP 2 DIR	CC 4 JMP 3 EXT	DC 4 JMP 3 IX2	EC 3 JMP 2 IX1	FC 3 JMP 1 IX
0D 5 BRCLR6 3 DIR	1D 5 BCLR6 2 DIR	2D 3 BMS 2 REL	3D 4 TST 2 DIR	4D 1 TSTA 1 INH	5D 1 TSTX 1 INH	6D 4 TST 2 IX1	7D 3 TST 1 IX	8D 7 RTC 1 INH	9D 1 NOP 1 INH	AD 5 BSR 2 REL	BD 5 JSR 2 DIR	CD 6 JSR 3 EXT	DD 6 JSR 3 IX2	ED 5 JSR 2 IX1	FD 5 JSR 1 IX
0E 5 BRSET7 3 DIR	1E 5 BSET7 2 DIR	2E 3 BIL 2 REL	3E 6 CPHX 3 EXT	4E 5 MOV 3 DD	5E 5 MOV 2 DIX+	6E 4 MOV 3 IMD	7E 5 MOV 2 IX+D	8E 2+ STOP 1 INH	9E Page 2	AE 2 LDX 2 IMM	BE 3 LDX 2 DIR	CE 4 LDX 3 EXT	DE 4 LDX 3 IX2	EE 3 LDX 2 IX1	FE 3 LDX 1 IX
0F 5 BRCLR7 3 DIR	1F 5 BCLR7 2 DIR	2F 3 BIH 2 REL	3F 5 CLR 2 DIR	4F 1 CLRA 1 INH	5F 1 CLRX 1 INH	6F 5 CLR 2 IX1	7F 4 CLR 1 IX	8F 2+ WAIT 1 INH	9F 1 TXA 1 INH	AF 2 AIX 2 IMM	BF 3 STX 2 DIR	CF 4 STX 3 EXT	DF 4 STX 3 IX2	EF 3 STX 2 IX1	FF 2 STX 1 IX

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 Post Increment**
EXT	**Extended**	**IX2**	**Indexed, 16-Bit Offset**	**IX1+**	**Indexed, 1-Byte Offset with Post Increment**
DD	**DIR to DIR**	**IMD**	**IMM to DIR**		
IX+D	**IX+ to DIR**	**DIX+**	**DIR to IX+**		

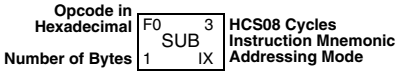


Table 8-3. Opcode Map (Sheet 2 of 2)

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

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 Post Increment**
EXT	**Extended**	**IX2**	**Indexed, 16-Bit Offset**		
DD	**DIR to DIR**	**IMD**	**IMM to DIR**	**IX1+**	**Indexed, 1-Byte Offset with Post Increment**
IX+D	**IX+ to DIR**	**DIX+**	**DIR to IX+**		

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

Prebyte (9E) and Opcode in Hexadecimal
Number of Bytes

9E60 6
NEG
3 SP1

HCS08 Cycles
Instruction Mnemonic
Addressing Mode

Chapter 9
Analog Comparator (S08ACMPVLPV1)

9.1 Introduction

The analog comparator module (ACMP) provides a circuit for comparing two analog input voltages or for comparing one analog input voltage to an internal reference voltage. The comparator circuit is designed to operate across the full range of the supply voltage (rail-to-rail operation).

In LPRUN and LPWAIT the VREF output must be enabled in order to use the ACMP option to compare to internal bandgap.

Figure 9-1 shows the MC9S08LH64 series block diagram with the ACMP highlighted.

9.1.1 ACMP Configuration Information

When using the bandgap reference voltage for input to ACMP+, the user must enable the VREF output to supply a reference signal to the ACMP. See Chapter 18, "Voltage Reference Module (S08VREFV1)," for information on how to enable the VREF output. For value of bandgap voltage reference see the data sheet.

9.1.2 ACMP/TPM Configuration Information

The ACMP module can be configured to connect the output of the analog comparator to TPM2 input capture channel 0 by setting the ACIC bit in SOPT2. With ACIC set, the TPM2CH0 pin is not available externally regardless of the configuration of the TPM2 module.

9.1.3 ACMP Clock Gating

The bus clock to the ACMP can be gated on and off using the ACMP bit in SCGC2. This bit is set after any reset, which enables the bus clock to this module. To conserve power, the ACMP bit can be cleared to disable the clock to this module when not in use. See Section 5.7, "Peripheral Clock Gating," for details.

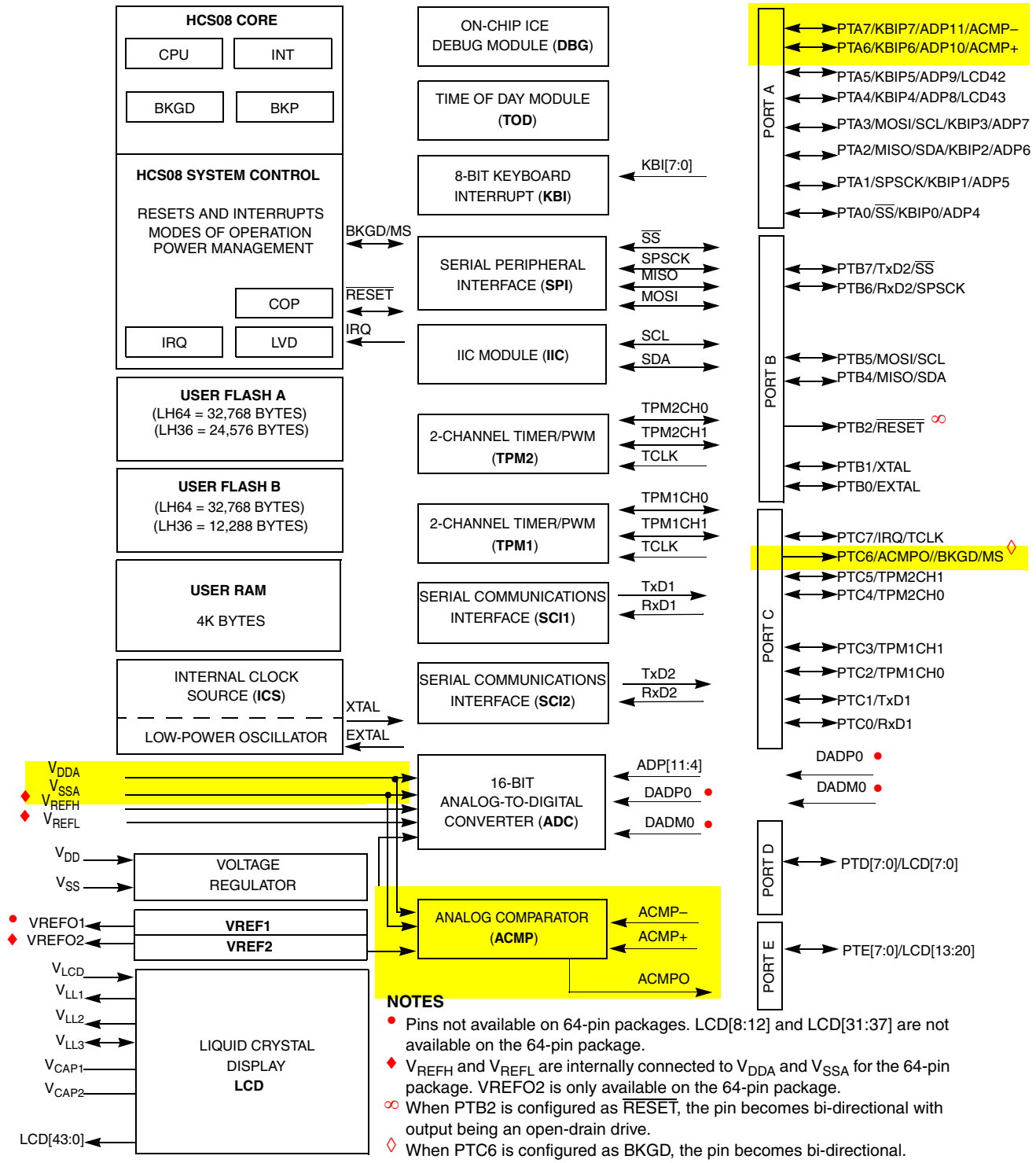


Figure 9-1. MC9S08LH64 Series Block Diagram Highlighting ACMP Block and Pins

9.1.4 Features

The ACMP has the following features:

- Full rail-to-rail supply operation
- Low input offset and hysteresis
- Selectable interrupt on rising edge, falling edge, or rising and falling edges of comparator output
- Option to compare to fixed internal bandgap reference voltage

9.1.5 Modes of Operation

9.1.5.1 Wait Mode Operation

During wait mode the ACMP, if enabled, continues to operate normally. Also, if enabled, the interrupt can wake the MCU.

9.1.5.2 Stop3 Mode Operation

If enabled, the ACMP continues to operate in stop3 mode and compare operation remains active. If ACOPE is enabled, comparator output operates in the normal operating mode and comparator output is placed onto the external pin. The MCU is brought out of stop when a compare event occurs and ACIE is enabled; ACF flag sets accordingly.

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

9.1.5.3 Stop2 Mode Operation

During stop2 mode, the ACMP module is fully powered down. Upon wakeup from stop2 mode, the ACMP module is in the reset state.

9.1.5.4 Active Background Mode Operation

When the microcontroller is in active background mode, the ACMP continues to operate normally. When ACMPO is shared with the BKGD pin and the BKGD pin is enabled, the ACMPO function is not available.

9.1.6 Block Diagram

The block diagram for the ACMP module follows.

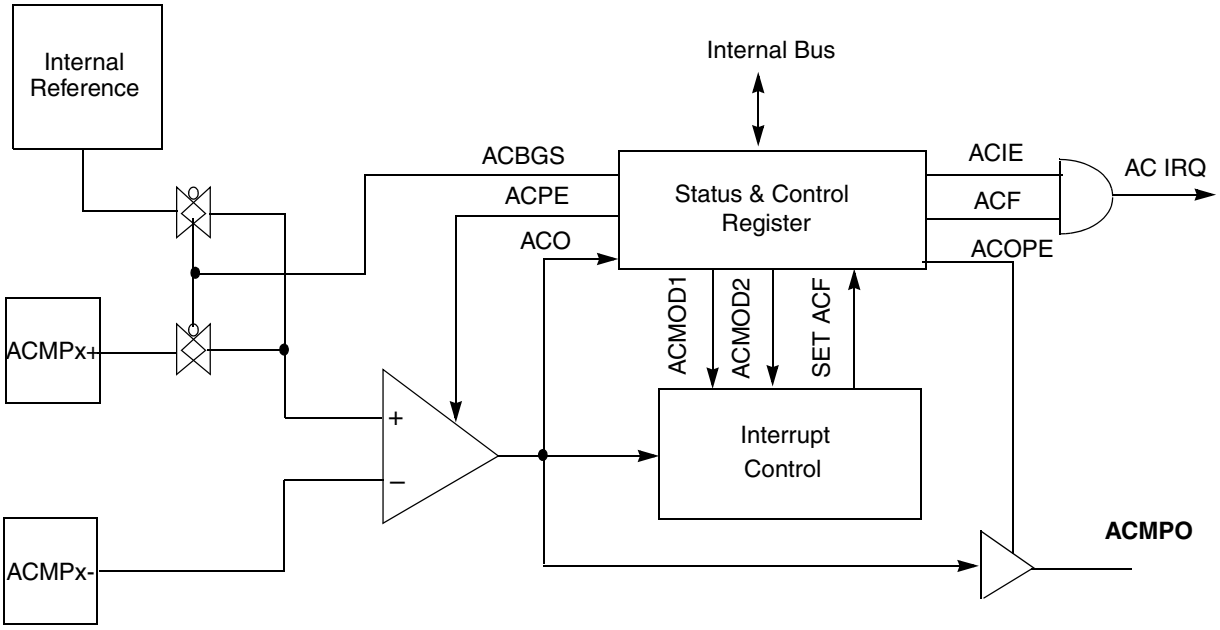


Figure 9-2. Analog Comparator Module Block Diagram

9.2 External Signal Description

The ACMP has two analog input pins: ACMP- and ACMP+. Each of these pins can accept an input voltage that varies across the full operating voltage range of the MCU. If the module is not enabled, each of these pins can be used as digital inputs or outputs. Consult the specific MCU documentation to determine what functions are shared with these analog inputs. As shown in the block diagram, the ACMP+ pin is connected to the comparator non-inverting input if ACBGS is equal to logic zero, and the ACMP- pin is connected to the inverting input of the comparator.

9.3 Register Definition

9.3.1 Status and Control Register (ACMPSC)

	7	6	5	4	3	2	1	0
R	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD1	ACMOD0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 9-3. ACMP Status and Control Register (ACMPSC)

Table 9-1. ACMPSC Field Descriptions

Field	Description
7 ACME	**Analog Comparator Module Enable** — The ACME bit enables the ACMP module. When the module is not enabled, it remains in a low power state. 0 Analog Comparator disabled. 1 Analog Comparator enabled.
6 ACBGS	**Analog Comparator Bandgap Select** — The ACBGS bit selects the internal bandgap as the comparator reference. 0 External pin ACMP+ selected as comparator non-inverting input. 1 Internal bandgap reference selected as comparator non-inverting input.
5 ACF	**Analog Comparator Flag** — The ACF bit is set when a compare event occurs. Compare events are defined by the ACMOD0 and ACMOD1 bits. The ACF bit is cleared by writing a logic one to the bit. 0 Compare event has not occurred. 1 Compare event has occurred.
4 ACIE	**Analog Comparator Interrupt Enable** — The ACIE bit enables the interrupt from the ACM. When this bit is set, an interrupt is asserted when the ACF bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 ACO	**Analog Comparator Output** — Reading the ACO bit returns the current value of the analog comparator output. The register bit is reset to zero and reads as logic zero when the ACMP module is disabled (ACME = 0).
2 ACOPE	**Analog Comparator Output Pin Enable** — ACOPE enables the comparator output to be placed onto the external pin, ACMPO. 0 Analog comparator output not available on ACMPO. 1 Analog comparator output is driven out on ACMPO.
1:0 ACMOD	**Analog Comparator Modes** — The ACMOD1 and ACMOD0 bits select the flag setting mode that controls the type of compare event that sets the ACF bit. 00 Comparator output falling edge. 01 Comparator output rising edge. 10 Comparator output falling edge. 11 Comparator output rising and falling edge.

9.4 Functional Description

The ACMP module can be used to compare:

- Two analog input voltages applied to ACMP- and ACMP+ or
- An analog input voltage applied to ACMP- with an internal bandgap reference voltage

The ACBGS bit selects the mode of operation. The comparator output is high when the non-inverting input is greater than the inverting input, and low when the non-inverting input is less than the inverting input. The ACMOD0 and ACMOD1 bits select the condition that causes the ACF bit to be set. The ACF bit can be set on a rising edge of the comparator output, a falling edge of the comparator output, or a rising and a falling edge (toggle). The comparator output can be read directly through the ACO bit.

9.5 Interrupts

The ACMP module is capable of generating an interrupt on a compare event. The interrupt request is asserted when both the ACIE bit and the ACF bit are set. The interrupt is deasserted by clearing either the

ACIE bit or the ACF bit. The ACIE bit is cleared by writing a logic zero and the ACF bit is cleared by writing a logic one.

Chapter 10
Analog-to-Digital Converter (S08ADC16V1)

10.1 Introduction

The 16-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

Figure 10-1 shows the MC9S08LH64 Series with the ADC module highlighted.

10.1.1 ADC shared with LCD

PTA4/KBI4/ADP8/LCD43 and PTA5/KBI4/ADP9/LCD42 share ADC functionality with LCD pins. LCD functionality must be disabled for these pins to operate as ADC pins. As ADC inputs, these pins operate just like other ADC pins.

10.1.2 Dedicated ADC Pins

The DADP0 and DADM0 pins are dedicated differential ADC pins. These pins are not available on the 64-pin package.

10.1.3 ADC Supply and Reference Pins

The 80-pin package contains separate V_{REFH} and V_{REFL} pins. On the 64-pin package, the V_{REFH} and V_{REFL} pins are connected to V_{DDA} and V_{SSA} respectively. External filtering may be necessary to ensure clean V_{DDA} and V_{REFH} for good results.

10.1.4 Status and Control Registers

This device has two ADCSC1 registers to control and monitor the conversion activity on two ADC channels. They are identified as ADCSC1A and SCDSC1B. The bit names in the registers are also identified with the A or B suffix. See Section 10.3.1, “Status and Control Registers 1 (ADCSC1A:ADCSC1n),” for a for a complete description of these registers.

10.1.5 ADC Reference Selection

The MC9S08LH64 has the ability to select from several different reference voltages for the ADC. The following table describes the options available for this device. Even if the external V_{REFH} is not selected, external filtering on V_{REFH} may be necessary to ensure good results.

Table 10-1. Reference Assignment

REFSEL	Reference Source
00	PAD VREFH and PAD VREFL
01	Internal VREF module
10	bandgap
11	Reserved

REFSEL is at the lowest 2 bits of register ADCSC2 and selects the voltage reference for the ADC.

- If REFSEL = 00, then PAD V_{REFL} and PAD V_{REFH} act as the ADC conversion reference.
- If REFSEL = 01, then the internal VREF module output acts as the ADC conversion reference.
- If REFSEL = 10, then the 1.2 V PMC bandgap output acts as the ADC conversion reference.
- If REFSEL = 11 (reserved), then selects the default voltage reference same as REFSEL = 2'b00.

10.1.6 ADC Clock Gating

The bus clock to the ADC can be gated on and off using the ADC bit in SCGC1 register. This bit is set after any reset, which enables the bus clock to this module. To conserve power, the ADC bit can be cleared to disable the clock to this module when not in use. See Section 5.7, "Peripheral Clock Gating," for details.

10.1.7 Module Configurations

This section provides device-specific information for configuring the ADC on MC9S08LH64 Series.

10.1.7.1 Configurations for Stop Modes

The ADC, if enabled, must be configured to use the asynchronous clock source, ADACK, to meet the ADC minimum frequency requirements. The VREF output must be enabled in order to convert the bandgap channel in stop mode.

10.1.7.2 Differential Channel Assignments

The ADC differential channel assignments for the MC9S08LH64 Series devices are shown in Table 10-2. Differential channels are selected when the DIFFn bit in the corresponding ADSC1n register is set (DIFFn= 1). Reserved channels convert to an unknown value.

Table 10-2. ADC Differential Channel Assignment

ADCH	Channel	Input	Pin Control
00000	DAD0	DADP0 and DADM0	ADPC0
00001	DAD1	Reserved	N/A
00010	DAD2	Reserved	N/A
00011	DAD3	Reserved	N/A

ADCH	Channel	Input	Pin Control
10000	R	Reserved	N/A
10001	R	Reserved	N/A
10010	R	Reserved	N/A
10011	R	Reserved	N/A

Table 10-2. ADC Differential Channel Assignment (continued)

ADCH	Channel	Input	Pin Control
00100	R	Reserved	N/A
00101	R	Reserved	N/A
00110	R	Reserved	N/A
00111	R	Reserved	N/A
01000	R	Reserved	N/A
01001	R	Reserved	N/A
01010	R	Reserved	N/A
01011	R	Reserved	N/A
01100	R	Reserved	N/A
01101	R	Reserved	N/A
01110	R	Reserved	N/A
01111	R	Reserved	N/A

ADCH	Channel	Input	Pin Control
10100	R	Reserved	N/A
10101	R	Reserved	N/A
10110	R	Reserved	N/A
10111	R	Reserved	N/A
11000	R	Reserved	N/A
11001	R	Reserved	N/A
11010	Temp-Differential	Temperature Sensor[1]	N/A
11011	Bandgap	Bandgap	N/A
11100	R	Reserved	N/A
11101	V_{REFH}	V_{REFH}	N/A
11110	R	Reserved	N/A
11111	Module Disabled	None	N/A

[1] For information, see Section 10.1.7.6, "Temperature Sensor."

10.1.7.3 Single-Ended Channel Assignments

The ADC single-ended channel assignments for the MC9S08LH64 Series devices are shown in Table 10-3. Single-ended channels are selected when the DIFFn bit in the corresponding ADSC1n register is cleared (DIFFn= 0). Reserved channels convert to an unknown value.

Table 10-3. ADC Single-Ended Channel Assignment

ADCH	Channel	Input	Pin Control
00000	DADP0	DADP0	ADPC0
00001	DADP1	Reserved	ADPC1
00010	DADP2	Reserved	ADPC2
00011	DADP3	Reserved	ADPC3
00100	AD4	PTA0/ADP4	ADPC4
00101	AD5	PTA1/ADP5	ADPC5
00110	AD6	PTA2/ADP6	ADPC6
00111	AD7	PTA3/ADP7	ADPC7
01000	AD8	PTA4/ADP8	ADPC8
01001	AD9	PTA5/ADP9	ADPC9
01010	AD10	PTA6/ADP10	ADPC10
01011	AD11	PTA7/ADP11	ADPC11

ADCH	Channel	Input	Pin Control
10000	AD16	Reserved	N/A
10001	AD17	Reserved	N/A
10010	AD18	Reserved	N/A
10011	AD19	VREFO	N/A
10100	AD20	Reserved	N/A
10101	AD21	Reserved	N/A
10110	AD22	Reserved	N/A
10111	AD23	VLCD	N/A
11000	AD24	VLL1	N/A
11001	R	Reserved	N/A
11010	Temp Single-ended	Temperature Sensor[1]	N/A
11011	Bandgap	Bandgap	N/A

Table 10-3. ADC Single-Ended Channel Assignment (continued)

ADCH	Channel	Input	Pin Control
01100	AD12	DADM0	ADPC12
01101	AD13	Reserved	N/A
01110	AD14	Reserved	N/A
01111	AD15	Reserved	N/A

ADCH	Channel	Input	Pin Control
11100	R	Reserved	N/A
11101	V_{REFH}	V_{REFH}	N/A
11110	V_{REFL}	V_{REFL}	N/A
11111	Module Disabled	None	N/A

[1] For information, see Section 10.1.7.6, "Temperature Sensor."

NOTE

Enable the VREF output to supply the bandgap voltage. See Chapter 18, "Voltage Reference Module (S08VREFV1)," for information on how to enable the VREF output. For the value of the bandgap voltage reference, see the data sheet.

10.1.7.4 Alternate Clock

The ADC is capable of performing conversions using the MCU bus clock, the bus clock divided by two, the local asynchronous clock (ADACK) within the module, or the alternate clock (ALTCLK). The ALTCLK on the MC9S08LH64 Series is connected to the ICSERCLK. See Chapter 11, "Internal Clock Source (S08ICSV3)" for more information.

10.1.7.5 Hardware Trigger

The ADC hardware trigger can be provided from either the TPM or the TOD module depending on the state of the ADCTRS bit in the SOPT2 register.

- When ADCTRS = 1, the ADCHWT (Hardware Trigger) is provided by the TPM2. TPM2 can be enabled as a hardware trigger manager for the ADC module by setting the ADTRG bit in the ADCSC2 register.
  - When enabled, the ADC will be triggered by TPM output. The timer overflow interrupt does not have to be enabled to trigger the ADC.
  - The TPM2 channel 0 output will trigger a conversion on channel A
  - The TPM2 channel 1 output will trigger a conversion on channel B.
  - With this configuration, conversion selection between ADSC1A and ADSC1B can be controlled using the TPM.

NOTE

If Trigger A or B occurs while a conversion is in progress the trigger is ignored.

The TPM must not be configured to trigger A and B at the same time.

- When ADCTRS = 0, ADCSC1A is the source for the conversion and the TOD generates the ADCHWT (Hardware Trigger).

— To enable the TOD as a hardware trigger for the ADC module, set the ADTRG bit in the ADCSC2 register when ADCTRS = 0. Once enabled, the ADC is triggered every time a TOD match condition occurs. The TOD match interrupt does not have to be enabled to trigger the ADC.

— The TOD can be configured to cause a hardware trigger in MCU run, wait, low-power run, low-power wait, and stop3.

Table 10-4. Configuring the ADC Trigger for Initiating Conversions

ADC Trigger Mode	ADCSC2 Register ADTRG Bit Value	SOPT2 Register ADCTRS Bit Value	ADCHWT	ADC triggers when ...	Conversion Source		Stop Mode Functionality
Hardware	ADTRG = 1	ADCTRS = 1	TPM2 channel output	TPM2 channel output raising edge.	ADCSC1A *or* ADCSC1B is selected by TPM2 for the source for the conversion.	TPM2 channel 0 output — Sets the ADHWTSA signal until flag is cleared.	No
						TPM2 channel 1 output — Sets the ADHWTSB signal until flag is cleared.	No
		ADCTRS = 0	TOD Match Flag	TOD match condition occurs.	ADCSC1A is the source for the conversion.		Yes
Software	ADTRG = 0	—		ADCSC1A is written.	ADCSC1A is the source for the conversion.		No

10.1.7.6 Temperature Sensor

The ADC module includes a temperature sensor whose output is connected to one of the ADC analog channel inputs. Equation 10-1 provides an approximate transfer function of the temperature sensor.

$$Temp = 25 - ((V_{TEMP} - V_{TEMP25}) \div m) \qquad \textit{Eqn. 10-1}$$

where:

— V_{TEMP} is the voltage of the temperature sensor channel at the ambient temperature.

— V_{TEMP25} is the voltage of the temperature sensor channel at 25°C.

— m is the hot or cold voltage versus temperature slope in V/°C.

For temperature calculations, use the V_{TEMP25} and m values in the data sheet.

In application code, the user reads the temperature sensor channel, calculates V_{TEMP}, and compares it to V_{TEMP25}. If V_{TEMP} is greater than V_{TEMP25} the cold slope value is applied in Equation 10-1. If V_{TEMP} is less than V_{TEMP25} the hot slope value is applied in Equation 10-1.

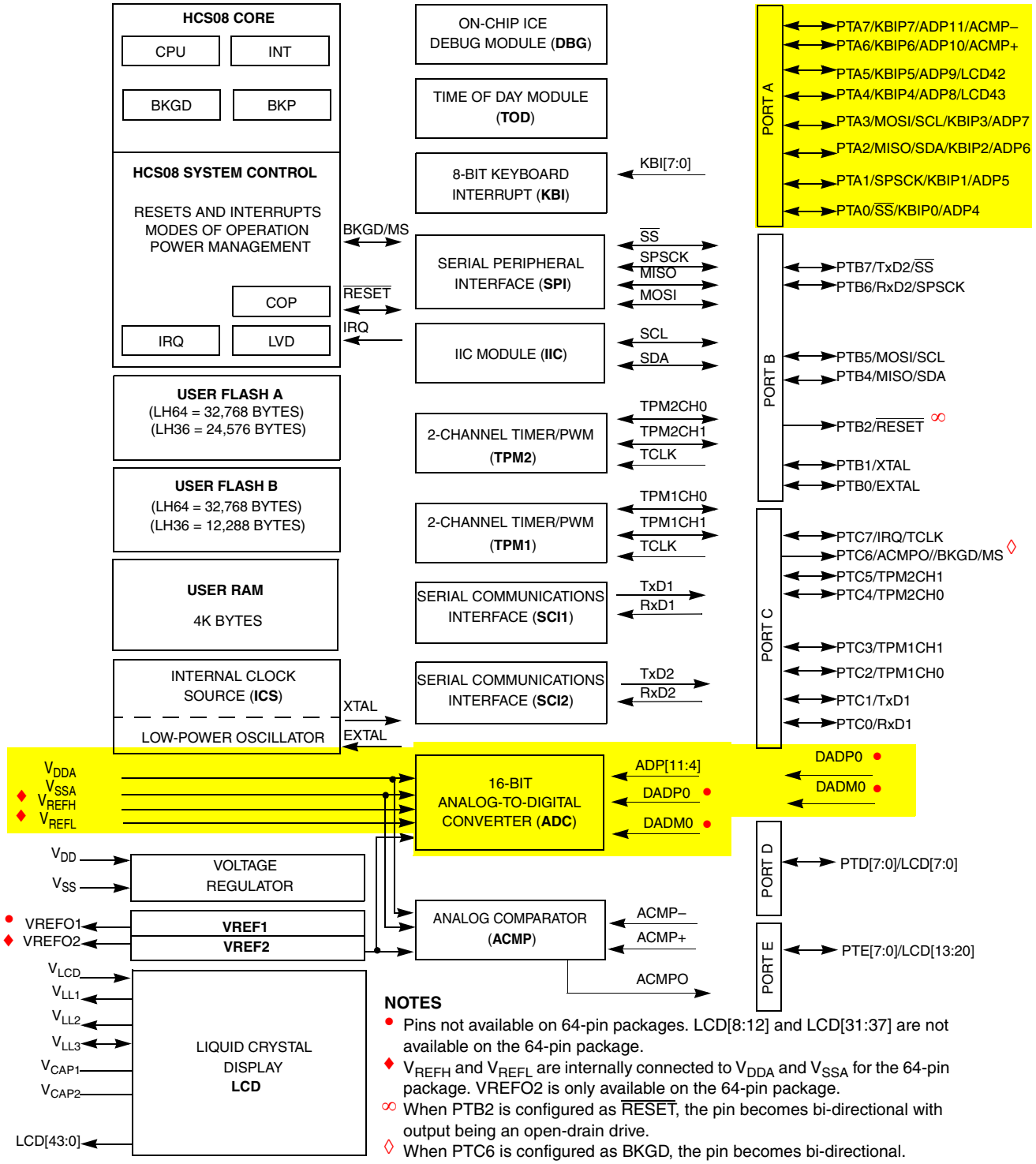


Figure 10-1. MC9S08LH64 Series Block Diagram Highlighting ADC Block and Pins

10.1.8 Features

Features of the ADC module include:

- Linear successive approximation algorithm with up to 16-bit resolution
- Up to four pairs of differential and 24 single-ended external analog inputs
- Output Modes:
  - Differential 16-bit, 13-bit, 11-bit, and 9-bit modes
  - Single-ended 16-bit, 12-bit, 10-bit, and 8-bit modes
- Output formatted in 2's complement 16b sign extended for differential modes
- Output in right-justified unsigned format for single-ended
- Single or continuous conversion (automatic return to idle after single conversion)
- Configurable sample time and conversion speed/power
- Conversion complete / Hardware average complete flag and interrupt
- Input clock selectable from up to four sources
- Operation in wait or stop3 modes for lower noise operation
- Asynchronous clock source for lower noise operation with option to output the clock
- Selectable asynchronous hardware conversion trigger with hardware channel select
- Automatic compare with interrupt for less-than, greater-than or equal-to, within range, or out-of-range, programmable value
- Temperature sensor
- Hardware average function
- Selectable voltage reference
  - Internal
  - External
  - Alternate
- Self-Calibration mode

10.1.9 Block Diagram

Figure 10-2 provides a block diagram of the ADC module.

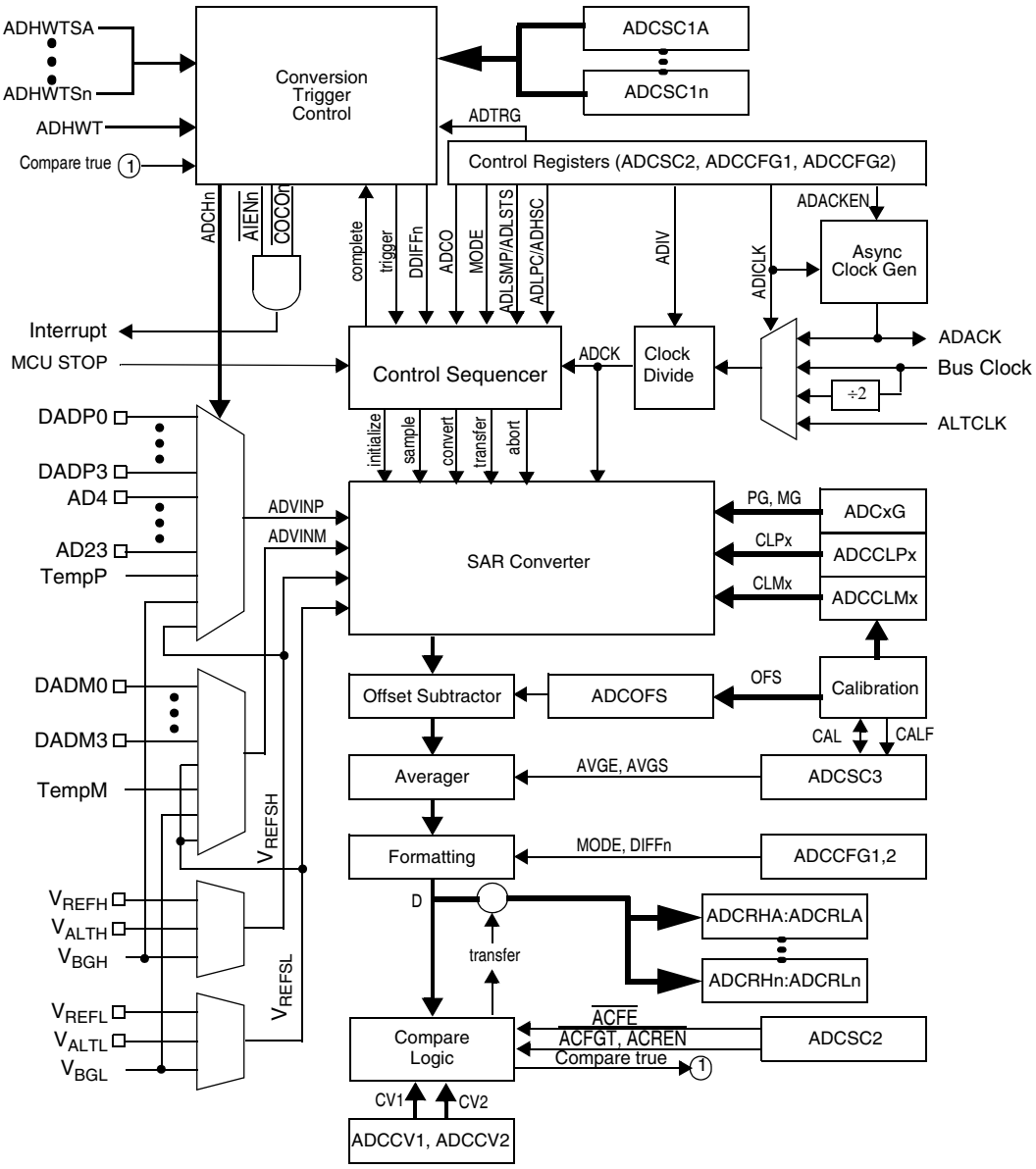


Figure 10-2. ADC Block Diagram

10.2 External Signal Description

The ADC module supports up to four pairs of differential inputs and 24 single-ended inputs. Each differential pair requires two inputs, DADPx and DADMx.The ADC also requires four supply/reference/ground connections.

Table 10-5. Signal Properties

Name	Function
DADP0-DADP3	Differential Analog Channel Inputs
DADM0-DADM3	Differential Analog Channel Inputs

Table 10-5. Signal Properties (continued)

Name	Function
AD4–AD23	Analog Channel inputs
V_{REFSH}	Voltage Reference Select High
V_{REFSL}	Voltage Reference Select Low
V_{DDAD}	Analog power supply
V_{SSAD}	Analog ground

10.2.1 Analog Power (V_{DDAD})

The ADC analog portion uses V_{DDAD} as its power connection. In some packages, V_{DDAD} is connected internally to V_{DD}. If externally available, connect the V_{DDAD} pin to the same voltage potential as V_{DD}. External filtering may be necessary to ensure clean V_{DDAD} for good results.

10.2.2 Analog Ground (V_{SSAD})

The ADC analog portion uses V_{SSAD} as its ground connection. In some packages, V_{SSAD} is connected internally to V_{SS}. If externally available, connect the V_{SSAD} pin to the same voltage potential as V_{SS}.

10.2.3 Voltage Reference Select High (V_{REFSH})

V_{REFSH} is the high-reference voltage for the converter.

The ADC can be configured to accept one of three voltage reference pairs for V_{REFSH}. Each pair contains a positive reference which must be between the minimum Ref Voltage High (defined in data sheet) and V_{DDAD}, and a ground reference which must be at the same potential as V_{SSAD}.

The three pairs are:

- external (V_{REFH} and V_{REFL})
- alternate (V_{ALTH} and V_{ALTL})
- internal bandgap (V_{BGH} and V_{BGL})

These voltage references are selected using the REFSEL bits in the ADCSC2 register.The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration.Consult the module introduction for information on the Voltage References specific to this MCU.

In some packages, V_{REFH} is connected in the package to V_{DDAD}. If externally available, the positive reference(s) may be connected to the same potential as V_{DDAD} or may be driven by an external source to a level between the minimum Ref Voltage High (defined in the Data Sheet) and the V_{DDAD} potential (V_{REFH} must never exceed V_{DDAD}).

10.2.4 Voltage Reference Select Low (V_{REFL})

V_{REFSL} is the low reference voltage for the converter.The ADC can be configured to accept one of three voltage reference pairs for V_{REFSL}. Each pair contains a positive reference which must be between the minimum Ref Voltage High (defined in Appendix A) and V_{DDAD}, and a ground reference which must be at the same potential as V_{SSAD}. The three pairs are external (V_{REFH} and V_{REFL}), alternate (V_{ALTH} and V_{ALTL}) and the internal bandgap (V_{BGH} and V_{BGL}). These voltage references are selected using the REFSEL bits.The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration.Consult the module introduction for information on the Voltage References specific to this MCU.

In some packages, V_{REFL} is connected in the package to V_{SSAD}. If externally available, connect the ground reference(s) to the same voltage potential as V_{SSAD}.

10.2.5 Analog Channel Inputs (ADx)

The ADC module supports up to 24 single-ended analog inputs. A single-ended input is selected for conversion through the ADCHn channel select bits when the DIFFn bit in the ADCSC1n register is low.

10.2.6 Differential Analog Channel Inputs (DADx)

The ADC module supports up to four differential analog channel inputs. Each differential analog input is a pair of external pins (DADPx and DADMx) referenced to each other to provide the most accurate analog to digital readings. A differential input is selected for conversion through the ADCHn channel select bits when the DIFFn bit in the ADCSC1n register bit is high.

10.3 Register Definition

These memory-mapped registers control and monitor operation of the ADC:

- Status and channel control registers, ADCSC1A:ADCSC1n
- Configuration registers, ADCCFG1 and ADCCFG2
- Data result registers, ADCRHA:ADCRLA to ADCRHn:ADCRLn
- Compare value registers, ADCCV1H, ADCCV1L, ADCCV2H, and ADCCV2L
- General status and control registers, ADCSC2 and ADCSC3
- Configuration registers, ADCCFG1 and ADCCFG2
- Offset Correction Registers, ADCOFSH and ADCOFSL
- Plus-input gain registers, ADCPGH and ADCPGL
- Minus-input gain registers, ADCMGH and ADCMGL
- Plus-side general calibration registers, ADCCLP0, ADCCLP1, ADCCLP2, ADCCLP3H, ADCCLP3L, ADCCLP4H, ADCCLP4L, ADCCLSP, ADCCLDP
- Minus-side general calibration registers, ADCCLM0, ADCCLM1, ADCCLM2, ADCCLM3H, ADCCLM3L, ADCCLM4H, ADCCLM4L, ADCCLSM, ADCCLDM
- Pin enable registers, APCTL1, APCTL2, APCTL3, and APCTL4

10.3.1 Status and Control Registers 1 (ADCSC1A:ADCSC1n)

This section describes the function of the ADC status and channel control registers, ADCSC1A through ADCSC1n. ADCSC1A is used for both software and hardware trigger modes of operation. ADCSC1B-ADCSC1n indicate potentially multiple ADCSC1 registers for use only in hardware trigger mode. Consult the module introduction for information on the number of ADCSC1n registers specific to this MCU. The ADCSC1A to ADCSC1n registers have identical fields, and are used in a “ping-pong” approach to control ADC operation. At any one point in time, only one of the ADCSC1A to ADCSC1n registers is actively controlling ADC conversions. Updating ADCSC1A while ADCSC1n is actively controlling a conversion is allowed (and vice-versa for any of the ADCSC1n registers specific to this MCU). Writing ADCSC1A while ADCSC1A is actively controlling a conversion aborts the current conversion. In software trigger mode (ADTRG=0), writes to ADCSC1A subsequently initiates a new conversion (if the ADCHn bits are equal to a value other than all 1s). Similarly, writing any of the ADCSC1n registers while that specific ADCSC1n register is actively controlling a conversion aborts the current conversion. Any of the ADCSC1B -ADCSC1n registers are not used for software trigger operation and therefore writes to the ADCSC1B -ADCSC1n registers do not initiate a new conversion.

	7	6	5	4	3	2	1	0
R	COCOn	AIENn	DIFFn	ADCHn				
W								
Reset:	0	0	0	1	1	1	1	1

Figure 10-3. Status and Channel Control Register 1n (ADCSC1n)

Table 10-6. ADCSC1:ADCSC1n Field Descriptions

Field	Description
7 COCOn	**Conversion Complete Flag** - The COCOn flag is a read-only bit that is set each time a conversion is completed when the compare function is disabled (ACFE=0) and the hardware average function is disabled (AVGE=0). When the compare function is enabled (ACFE=1), the COCOn flag is set upon completion of a conversion only if the compare result is true. When the hardware average function is enabled (AVGE=1), the COCOn flag is set upon completion of the selected number of conversions (determined by the AVGS bits). The COCO1 flag will also set at the completion of a Calibration sequence. The COCOn bit is cleared when the respective ADCSCn is written or when the respective ADCRLn is read. 0 Conversion not completed 1 Conversion completed
6 AIENn	**Interrupt Enable** - AIENn enables conversion complete interrupts. When COCOn becomes set while the respective AIENn is high, an interrupt is asserted. 0 Conversion complete interrupt disabled 1 Conversion complete interrupt enabled

Table 10-6. ADCSC1:ADCSC1n Field Descriptions (continued)

Field	Description
5 DIFFn	**Differential Mode Enable** - DIFFn configures the ADC to operate in differential mode. When enabled this mode automatically selects from the differential channels, changes the conversion algorithm and the number of cycles to complete a conversion. 0 Single-ended conversions and input channels are selected 1 Differential conversions and input channels are selected
4:0 ADCHn[4:0]	**Input Channel Select** - The ADCHn bits form a 5-bit field that selects one of the input channels. The input channel decode is dependent upon the value of the DIFFn bit as detailed in Table 10-7. The successive approximation converter subsystem is turned off when the channel select bits are all set(ADCHn = 11111). This feature allows for explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way prevents an additional, single conversion from being performed. It is not necessary to set the channel select bits to all ones to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes.

Table 10-7. Input Channel Select

ADCHn	Input Selected when DIFFn=0	Input Selected when DIFFn=1
00000–00011	DADP0-DADP3	DAD0-DAD3[1]
00100-10111	AD4-AD23	Reserved
11000-11001	Reserved	Reserved
11010	Temp Sensor (single-ended)	Temp Sensor (differential)
11011	Bandgap (single-ended)	Bandgap (differential)
11100	Reserved	Reserved
11101	V_{REFSH}[2]	$-V_{REFSH}$[2] (differential)
11110	V_{REFSL}[2]	Reserved
11111	Module disabled	

1 DAD0-DAD3 are associated with the input pin pairs DADPx and DADMx.

2 Voltage Reference selected is determined by the REFSEL bits in the ADCSC2 register. Refer to Section 10.4.3 for more information on voltage reference selection.

10.3.2 Configuration Register 1 (ADCCFG1)

ADCCFG1 selects the mode of operation, clock source, clock divide, and configure for low power or long sample time.

	7	6	5	4	3	2	1	0
R W	ADLPC	ADIV		ADLSMP	MODE		ADICLK	
Reset:	0	0	0	0	0	0	0	0

Figure 10-4. Configuration Register (ADCCFG1)

Table 10-8. ADCCFG1 Register Field Descriptions

Field	Description
7 ADLPC	**Low-Power Configuration -** ADLPC controls the power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required. 0 Normal power configuration 1 Low-power configuration: The power is reduced at the expense of maximum clock speed.
6:5 ADIV[6:5]	**Clock Divide Select -** ADIV selects the divide ratio used by the ADC to generate the internal clock ADCK. Table 10-9 shows the available clock configurations.
4 ADLSMP	**Sample Time Configuration -** ADLSMP selects between different sample times based on the conversion mode selected. This bit adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required. **When ADLSMP=1, the Long Sample Time Select bits (**ADLSTS[1:0]) can select the extent of the long sample time. 0 Short sample time 1 Long sample time (The ADLTS bits can select the extent of the long sample time)
3:2 MODE[3:2]	**Conversion Mode Selection -** MODE bits are used to select between the ADC resolution mode. See Table 10-10.
1:0 ADICLK[1:0]	**Input Clock Select -** ADICLK bits select the input clock source to generate the internal clock ADCK. See Table 10-11.

Table 10-9. Clock Divide Select

ADIV	Divide Ratio	Clock Rate
00	1	Input clock
01	2	Input clock ÷ 2
10	4	Input clock ÷ 4
11	8	Input clock ÷ 8

Table 10-10. Conversion Modes

MODE	DIFFn	Conversion Mode Description
00	0	single-ended 8-bit conversion
00	1	Differential 9-bit conversion with 2s complement output
01	0	single-ended 12-bit conversion
01	1	Differential 13-bit conversion with 2s complement output
10	0	single-ended 10-bit conversion
10	1	Differential 11-bit conversion with 2s complement output
11	0	single-ended 16-bit conversion
11	1	Differential 16-bit conversion with 2s complement output

Table 10-11. Input Clock Select

ADICLK	Selected Clock Source
00	Bus clock
01	Bus clock divided by 2
10	Alternate clock (ALTCLK)
11	Asynchronous clock (ADACK)

10.3.3 Configuration Register 2 (ADCCFG2)

ADCCFG2 selects differential mode, the special high speed configuration for very high speed conversions, and selects the long sample time duration during long sample mode.

	7	6	5	4	3	2	1	0
R	0	0	0	0	ADACKEN	ADHSC	ADLSTS	
W								
Reset:	0	0	0	0	0	0	0	0

Figure 10-5. Configuration Register 2(ADCCFG2)

Table 10-12. ADCCFG2 Register Field Descriptions

Field	Description
3 ADACKEN	**Asynchronous clock output enable** - ADACKEN enables the ADC's asynchronous clock source and the clock source output regardless of the conversion and input clock select (ADICLK bits) status of the ADC. Based on MCU configuration the asynchronous clock may be used by other modules (see module introduction section). Setting this bit allows the clock to be used even while the ADC is idle or operating from a different clock source. Also, latency of initiating a single or first-continuous conversion with the asynchronous clock selected is reduced since the ADACK clock is already operational. 0 Asynchronous clock output disabled; Asynchronous clock only enabled if selected by ADICLK and a conversion is active 1 Asynchronous clock and clock output enabled regardless of the state of the ADC
2 ADHSC	**High Speed Configuration- ADHSC configures the ADC for very high speed operation. The conversion sequence is altered (4 ADCK cycles added to the conversion time) to allow higher speed conversion clocks.** 0 Normal conversion sequence selected 1 High speed conversion sequence selected (4 additional ADCK cycles to total conversion time)
1:0 ADLSTS	**Long Sample Time Select -** ADLSTS selects between the extended sample times when long sample time is selected (ADLSMP=1). This allows higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required. 00 Default longest sample time (20 extra ADCK cycles; 24 ADCK cycles total) 01 12 extra ADCK cycles; 16 ADCK cycles total sample time 10 6 extra ADCK cycles; 10 ADCK cycles total sample time 11 2 extra ADCK cycles; 6 ADCK cycles total sample time

10.3.4 Data Result Registers (ADCRHA:ADCRLA to ADCRHn:ADCRLn)

The Data Result Registers (ADCRHA:ADCRLA to ADCRHn:ADCRLn) contain the result of an ADC conversion of the channel selected by the respective status and channel control register (ADCSC1A:ADCSC1n). For every ADCSC1A:ADCSC1n status and channel control register, there is a respective ADCRHA:ADCRLA to ADCRHn:ADCRLn data result register. Consult the module introduction for information on the number of ADCRHn:ADCRLn registers specific to this MCU.Reading ADCRHn prevents the ADC from transferring subsequent conversion results into the result registers until ADCRLn is read. If ADCRLn is not read until after the next conversion is completed, the intermediate conversion result is lost. In 8-bit single-ended mode, there is no interlocking with ADCRLn.

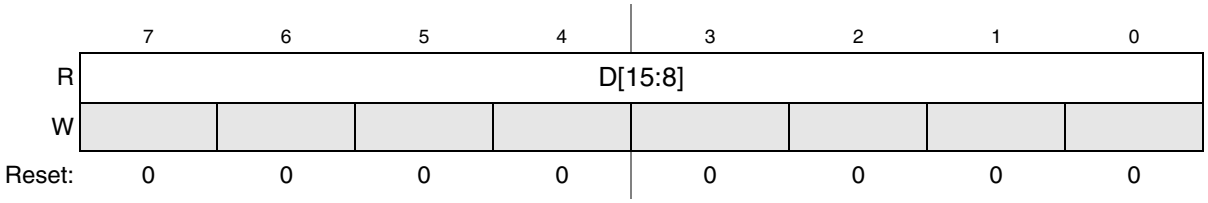


Figure 10-6. Data Result High Register (ADCRHn)

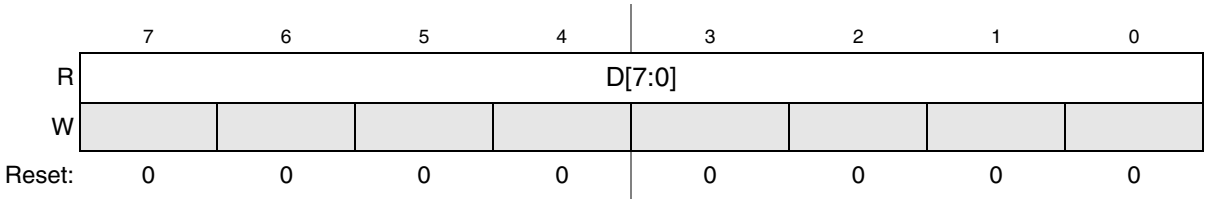


Figure 10-7. Data Result Low Register (ADCRLn)

ADCRHn contains the upper bits of the result of a conversion based on the conversion mode. ADCRLn contains the lower eight bits of the result of a conversion, or all eight bits of an 8-bit single-ended conversion. Unused bits in the ADCRHn register are cleared in unsigned right justified modes and carry the sign bit (MSB) in sign extended 2's complement modes. For example when configured for 10-bit single-ended mode, D[15:10] are cleared. When configured for 11-bit differential mode, D[15:10] carry the sign bit (bit 10 extended through bit 15).

Table 10-13 describes the behavior of the data result registers in the different modes of operation.

Table 10-13. Data Result Register Description

Conversion Mode	Data Result Register bits																Format
	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
16b differential	S	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	signed 2's complement
16b single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	unsigned right justified
13b differential	S	S	S	S	D	D	D	D	D	D	D	D	D	D	D	D	sign extended 2's complement
12b single-ended	0	0	0	0	D	D	D	D	D	D	D	D	D	D	D	D	unsigned right justified
11b differential	S	S	S	S	S	S	D	D	D	D	D	D	D	D	D	D	sign extended 2's complement

Table 10-13. Data Result Register Description

Conversion Mode	Data Result Register bits																Format
	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
10b single-ended	0	0	0	0	0	0	D	D	D	D	D	D	D	D	D	D	unsigned right justified
9b differential	S	S	S	S	S	S	S	S	D	D	D	D	D	D	D	D	sign extended 2's complement
8b single-ended	0	0	0	0	0	0	0	0	D	D	D	D	D	D	D	D	unsigned right justified

S: Sign bit or sign bit extension.
D: Data (2's complement data if indicated).

10.3.5 Compare Value Registers (ADCCV1H:ADCCV1L and ADCCV2H:ADCCV2L)

The Compare Value Registers (ADCCV1H:ADCCV1L & ADCCV2H:ADCCV2L) contain a compare value used to compare with the conversion result when the compare function is enabled (ACFE=1). This register is formatted the same for both bit position definition and value format (unsigned or sign-extended 2's complement) as the Data Result Registers (ADCRHn:ADCRLn) in the different modes of operation (See Table 10-13). Therefore, the compare function only uses the compare value register bits that are related to the ADC mode of operation.

The compare value 2 registers (ADCCV2H:ADCCV2L) are utilized only when the compare range function is enabled (ACREN=1).

In all modes except 8-bit single-ended conversions, the ADCCV1H register holds the upper bits of the first compare value. In 8-bit single-ended mode, ADCCV1H is not used during compare. In all conversion modes, the ADCCV1L register holds the lower 8 bits of the first compare value. The compare function is further detailed in Section 10.4.6.

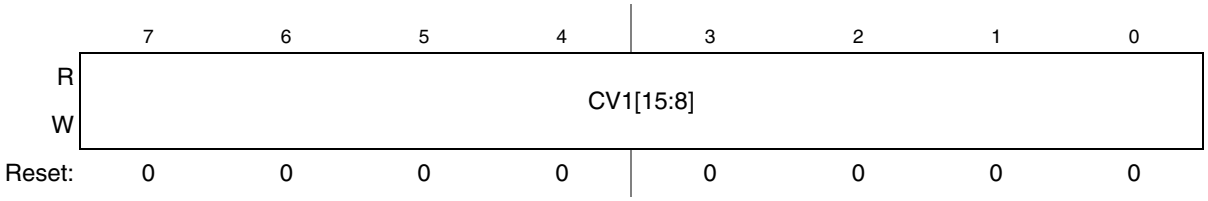


Figure 10-8. Compare Value 1 High Register (ADCCV1H)

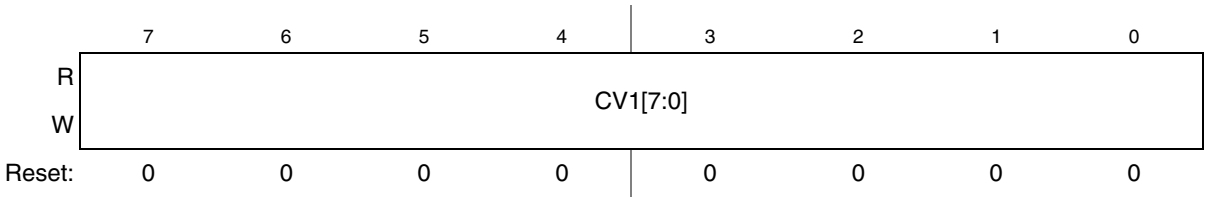


Figure 10-9. Compare Value 1 Low Register(ADCCV1L)

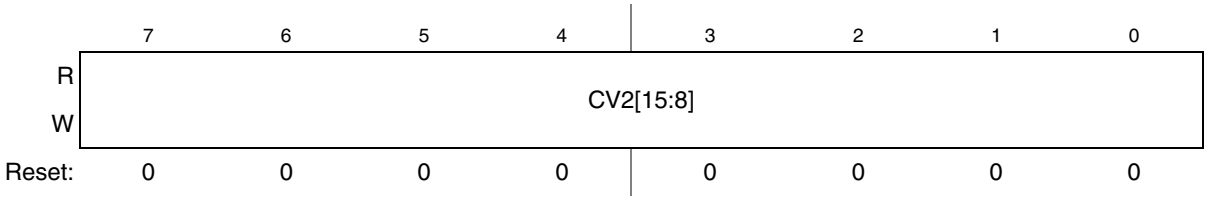


Figure 10-10. Compare Value 2 High Register (ADCCV2H)

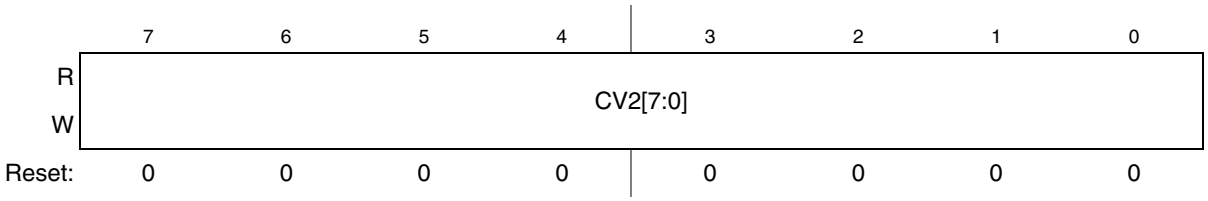


Figure 10-11. Compare Value 2Low Register(ADCCV2L)

10.3.6 Status and Control Register 2 (ADCSC2)

The ADCSC2 register contains the conversion active, hardware/software trigger select, compare function and voltage reference select of the ADC module.

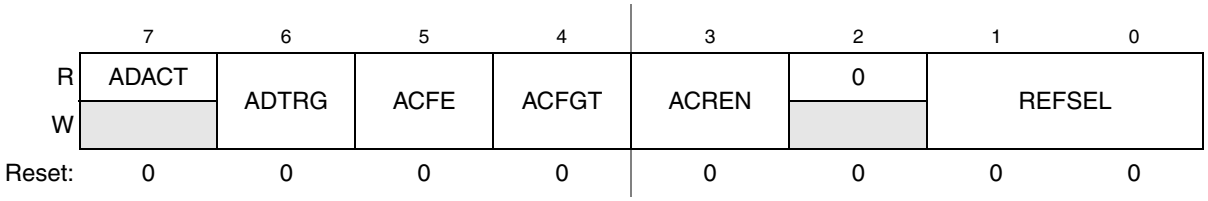


Figure 10-12. Status and Control Register 2 (ADCSC2)

Table 10-14. ADCSC2 Register Field Descriptions

Field	Description
7 ADACT	**Conversion Active** - ADACT indicates that a conversion or hardware averaging is in progress. ADACT is set when a conversion is initiated and cleared when a conversion is completed or aborted. 0 Conversion not in progress 1 Conversion in progress
6 ADTRG	**Conversion Trigger Select** - ADTRG selects the type of trigger used for initiating a conversion. Two types of trigger are selectable: software trigger and hardware trigger. When software trigger is selected, a conversion is initiated following a write to ADCSC1A. When hardware trigger is selected, a conversion is initiated following the assertion of the ADHWT input after a pulse of the ADHWTSn input.Refer to Section 10.4.5.1 for more information on initiating conversions. 0 Software trigger selected 1 Hardware trigger selected
5 ACFE	**Compare Function Enable** - ACFE enables the compare function. 0 Compare function disabled 1 Compare function enabled

Table 10-14. ADCSC2 Register Field Descriptions (continued)

Field	Description
4 ACFGT	**Compare Function Greater Than Enable** - ACFGT configures the compare function to check the conversion result relative to the compare value register(s) (ADCCV1H:ADCCV1L and ADCCV2H:ADCCV2L) based upon the value of ACREN. The ACFE bit must be set for ACFGT to have any effect. The compare function modes are further detailed in Table 10-27 in Section 10.4.6. 0 Configures Less Than Threshold, Outside Range Not Inclusive and Inside Range Not Inclusive functionality based on the values placed in the ADCCV1 and ADCCV2 registers. 1 Configures GreaterThan Or EqualTo Threshold, Outside Range Inclusive and Inside Range Inclusive functionality based on the values placed in the ADCCV1 and ADCCV2 registers.
3 ACREN	**Compare Function Range Enable** - ACREN configures the compare function to check the conversion result of the input being monitored is either between or outside the range formed by the compare value registers (ADCCV1H:ADCCV1L and ADCCV2H:ADCCV2L) determined by the value of ACFGT.The ACFE bit must be set for ACFGT to have any effect. The compare function modes are further detailed in Table 10-27 in Section 10.4.6. 0 Range function disabled. Only the compare value 1 register (ADCCV1H:ADCCV1L) is compared. 1 Range function enabled. Both compare value registers (ADCCV1H:ADCCV1L and ADCCV2H:ADCCV2L) are compared.
1:0 REFSEL [1:0]	**Voltage Reference Selection** - REFSEL bits select the voltage reference source used for conversions.Refer to Section 10.4.3 for more information on voltage reference selection. 00 Default voltage reference pin pair (External pins V_{REFH} and V_{REFL}). 01 Alternate reference pair (V_{ALTH} and V_{ALTL}). This pair may be additional external pins or internal sources depending on MCU configuration.Consult the module introduction for information on the Voltage Reference specific to this MCU. 10 Internal bandgap reference and associated ground reference (V_{BGH} and V_{BGL}). 11 Reserved - Selects default voltage reference (V_{REFH} and V_{REFL}) pin pair.

10.3.7 Status and Control Register 3 (ADCSC3)

The ADCSC3 register controls the calibration, continuous convert and hardware averaging functions of the ADC module.

	7	6	5	4	3	2	1	0
R	CAL	CALF	0	0	ADCO	AVGE	AVGS	
W								
Reset:	0	0	0	0	0	0	0	0

Figure 10-13. Status and Control Register 3 (ADCSC3)

Table 10-15. ADCSC3 Register Field Descriptions

Field	Description
7 CAL	**Calibration - CAL begins the calibration sequence when set. This bit stays set while the calibration is in progress and is cleared when the calibration sequence is complete. The CALF bit must be checked to determine the result of the calibration sequence. Once started, the calibration routine cannot be interrupted by writes to the ADC registers or the results will be invalid and the CALF bit will set. Setting the CAL bit will abort any current conversion.**
6 CALF	**Calibration Failed Flag -** CALF displays the result of the calibration sequence. The calibration sequence will fail if ADTRG = 1, any ADC register is written, or any stop mode is entered before the calibration sequence completes.The CALF bit is cleared by writing a 1 to this bit. 0 Calibration completed normally. 1 Calibration failed. ADC accuracy specifications are not guaranteed.
3 ADCO	**Continuous Conversion Enable** - ADCO enables continuous conversions.Refer to Section 10.4.5.1 for more information on initiating conversions. 0 One conversion or one set of conversions if the hardware average function is enabled (AVGE=1) after initiating a conversion. 1 Continuous conversions or sets of conversions if the hardware average function is enabled (AVGE=1) after initiating a conversion.
2 AVGE	**Hardware average enable - AVGE enables the hardware average function of the ADC.** 0 Hardware average function disabled 1 Hardware average function enabled
1:0 AVGS	**Hardware Average select** - AVGS determine how many ADC conversions will be averaged to create the ADC average result. 00 - 4 Samples averaged 01 - 8 Samples averaged 10 - 16 Samples averaged 11 - 32 Samples averaged

10.3.8 ADC Offset Correction Register (ADCOFSH:ADCOFSL)

The ADC Offset Correction Register (ADCOFSH:ADCOFSL) contains the user-selected or calibration-generated offset error correction value.

This register is a 2’s complement, left justified, 16b value formed by the concatenation of:

- ADCOFSH
- ADCOFSL.

The value in the offset correction registers (ADCOFSH:ADCOFSL) is subtracted from the conversion and the result is transferred into the result registers (ADCRHn:ADCRLn).

NOTE

If the result is above the maximum or below the minimum result value, it is forced to the appropriate limit for the current mode of operation.For additional information, please see Section 10.4.8.

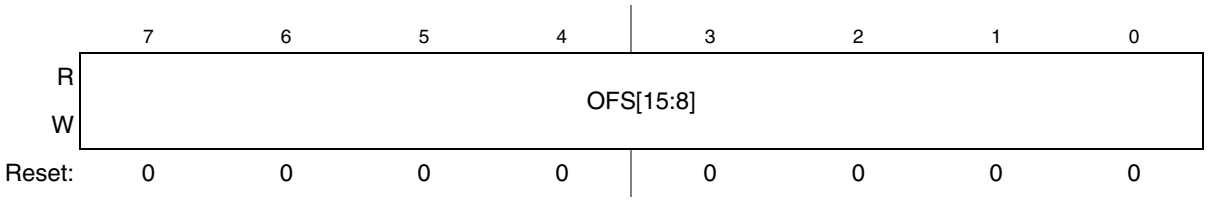


Figure 10-14. Offset Calibration High Register (ADCOFSH)

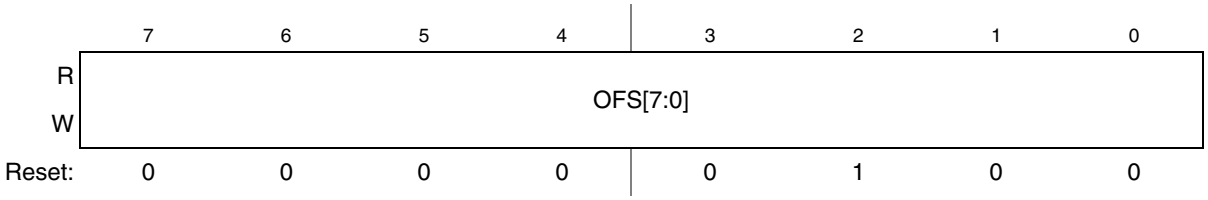


Figure 10-15. Offset Calibration Low Register (ADCOFSL)

10.3.9 ADC Plus-Side Gain Register (ADCPGH:ADCPGL)

The Plus-Side Gain Register (ADCPGH:ADCPGL) contains the gain error correction for the plus-side input in differential mode or the overall conversion in single-ended mode. ADCPGH:ADCPGL represent a 16 bit floating point number representation of the gain adjustment factor, with the decimal point fixed between PG15 and PG14. This register must be written by the user with the value described in the calibration procedure or the gain error specifications may not be met.

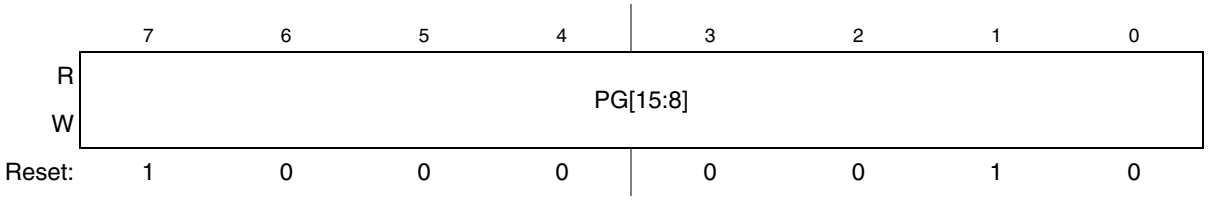


Figure 10-16. ADC Plus Gain High Register (ADCPGH)

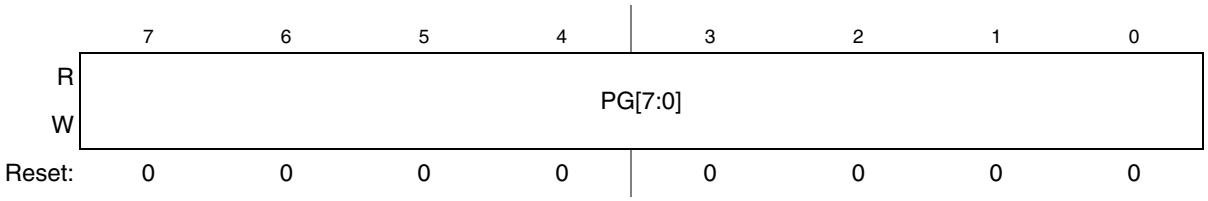


Figure 10-17. ADC Plus Gain Low Register (ADCPGL)

10.3.10 ADC Minus-Side Gain Register (ADCMGH:ADCMGL)

The Minus-Side Gain Register (ADCMGH:ADCMGL) contains the gain error correction for the minus-side input in differential mode. This register is ignored in single-ended mode. ADCMGH:ADCMGL represent a 16 bit floating point number representation of the gain adjustment

factor, with the decimal point fixed between MG15 and MG14. This register must be written by the user with the value described in the calibration procedure or the gain error specifications may not be met.

	7	6	5	4	3	2	1	0
R W	MG[15:8]							
Reset:	1	0	0	0	0	0	1	1

Figure 10-18. ADC Minus-Side Gain Register (ADCMGH)

	7	6	5	4	3	2	1	0
R W	MG[7:0]							
Reset:	0	0	0	1	0	0	0	0

Figure 10-19. ADC Minus-Side Gain Register (ADCMGL)

10.3.11 ADC Plus-Side General Calibration Value Registers (ADCCLPx)

The Plus-Side General Calibration Value Registers (ADCCLPx) contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLP0[5:0], CLP1[6:0], CLP2[7:0], CLP3[8:0], CLP4[9:0], CLPS[5:0], and CLPD[5:0]. ADCCLPx are automatically set once the self calibration sequence is done (CAL is cleared). If these registers are written by the user after calibration, the linearity error specifications may not be met.

	7	6	5	4	3	2	1	0
R	0	0	CLPD[5:0]					
W								
Reset:	0	0	0	0	1	0	0	1

Figure 10-20. Plus-Side General Calibration Register (ADCCLPD)

	7	6	5	4	3	2	1	0
R	0	0	CLPS[5:0]					
W								
Reset:	0	0	1	0	0	0	0	0

Figure 10-21. Plus-Side General Calibration Register (ADCCLPS)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	CLP4[9:8]	
W								
Reset:	0	0	0	0	0	0	1	1

Figure 10-22. Plus-Side General Calibration Register (ADCCLP4H)

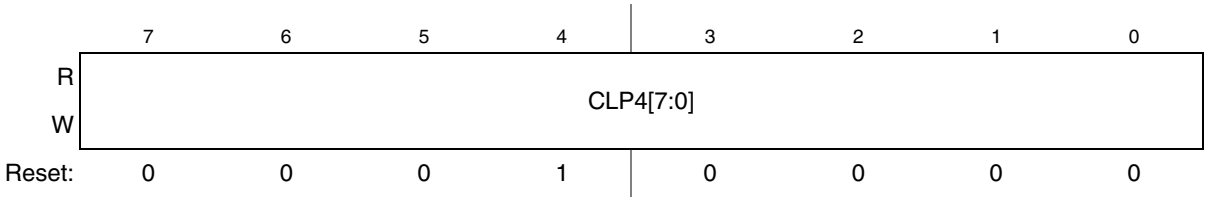


Figure 10-23. Plus-Side General Calibration Register (ADCCLP4L)

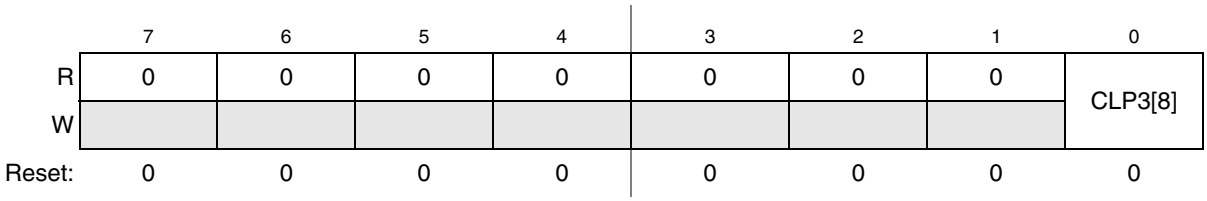


Figure 10-24. Plus-Side General Calibration Register (ADCCLP3H)

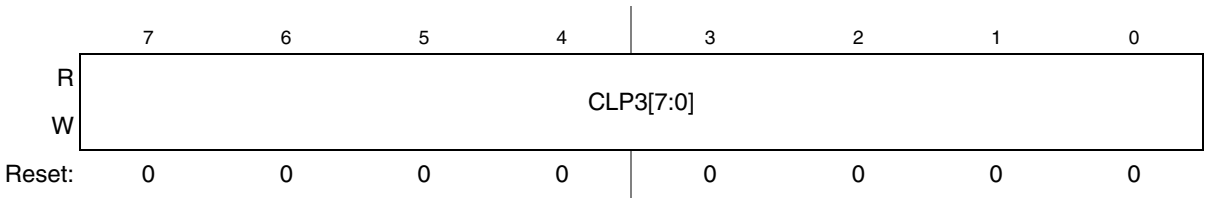


Figure 10-25. Plus-Side General Calibration Register (ADCCLP3L)

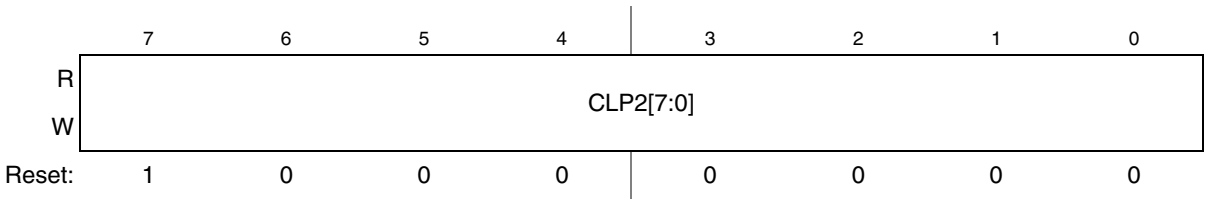


Figure 10-26. Plus-Side General Calibration Register (ADCCLP2)

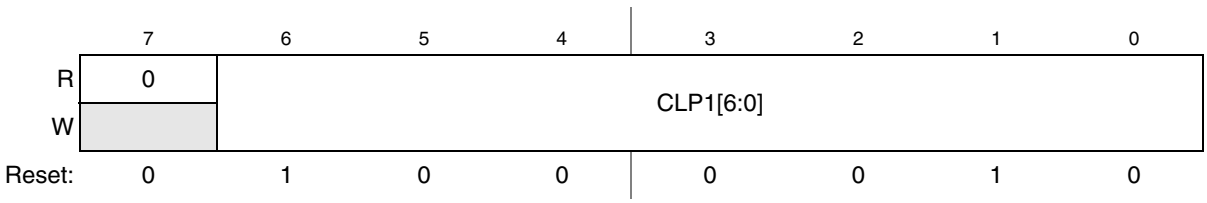


Figure 10-27. Plus-Side General Calibration Register (ADCCLP1)

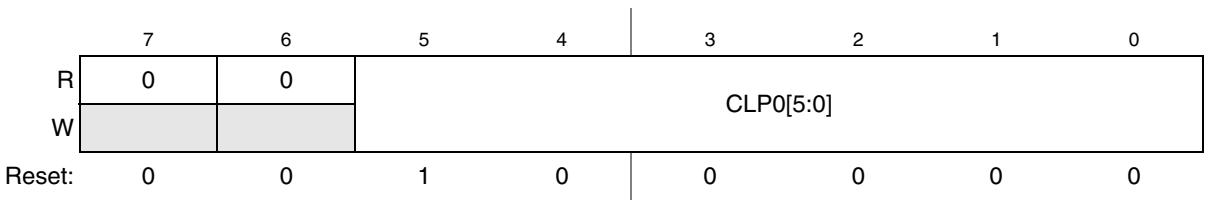


Figure 10-28. Plus-Side General Calibration Register (ADCCLP0)

10.3.12 ADC Minus-Side General Calibration Value Registers (ADCCLMx)

ADCCLMx contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLM0[5:0], CLM1[6:0], CLM2[7:0], CLM3[8:0], CLM4[9:0], CLMS[5:0], and CLMD[5:0]. ADCCLMx are automatically set once the self calibration sequence is done (CAL is cleared). If these registers are written by the user after calibration, the linearity error specifications may not be met.

	7	6	5	4	3	2	1	0
R	0	0	CLMD[5:0]					
W								
Reset:	0	0	0	1	1	1	0	1

Figure 10-29. Minus-Side General Calibration Register (ADCCLMD)

	7	6	5	4	3	2	1	0
R	0	0	CLMS[5:0]					
W								
Reset:	0	0	1	1	0	0	0	1

Figure 10-30. Minus-Side General Calibration Register (ADCCLMS)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	CLM4[9:8]	
W								
Reset:	0	0	0	0	0	0	1	1

Figure 10-31. Minus-Side General Calibration Register (ADCCLM4H)

	7	6	5	4	3	2	1	0
R	CLM4[7:0]							
W								
Reset:	0	0	0	1	0	0	0	0

Figure 10-32. Minus-Side General Calibration Register (ADCCLM4L)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CLM3[8]
W								
Reset:	0	0	0	0	0	0	0	1

Figure 10-33. Minus-Side General Calibration Register (ADCCLM3H)

	7	6	5	4	3	2	1	0
R W	CLM3[7:0]							
Reset:	1	0	0	0	1	0	0	0

Figure 10-34. Minus-Side General Calibration Register (ADCCLM3L)

	7	6	5	4	3	2	1	0
R W	CLM2[7:0]							
Reset:	1	1	0	0	0	1	0	0

Figure 10-35. Minus-Side General Calibration Register (ADCCLM2)

	7	6	5	4	3	2	1	0
R	0	CLM1[6:0]						
W								
Reset:	0	1	1	0	0	0	1	0

Figure 10-36. Minus-Side General Calibration Register (ADCCLM1)

	7	6	5	4	3	2	1	0
R	0	0	CLM0[5:0]					
W								
Reset:	0	0	1	1	0	0	0	1

Figure 10-37. Minus-Side General Calibration Register (ADCCLM0)

10.3.13 Pin Control 1 Register (APCTL1)

The pin control registers disable the I/O port control of MCU pins used as analog inputs. APCTL1 is used to control the pins associated with channels 0–7 of the ADC module.

	7	6	5	4	3	2	1	0
R W	ADPC[7:0]							
Reset:	0	0	0	0	0	0	0	0

Figure 10-38. Pin Control 1 Register (APCTL1)

Table 10-16. APCTL1 Register Field Descriptions

Field	Description
7 ADPC7	**ADC Pin Control 7 -** ADPC7 controls the pin associated with channel AD7. 0 AD7 pin I/O control enabled 1 AD7 pin I/O control disabled
6 ADPC6	**ADC Pin Control 6** - ADPC6 controls the pin associated with channel AD6. 0 AD6 pin I/O control enabled 1 AD6 pin I/O control disabled
5 ADPC5	**ADC Pin Control 5**- ADPC5 controls the pin associated with channel AD5. 0 AD5 pin I/O control enabled 1 AD5 pin I/O control disabled
4 ADPC4	**ADC Pin Control 4** - ADPC4 controls the pin associated with channel AD4. 0 AD4 pin I/O control enabled 1 AD4 pin I/O control disabled
3 ADPC3	**ADC Pin Control 3** - ADPC3 controls the pin associated with channel AD3. 0 AD3 pin I/O control enabled 1 AD3 pin I/O control disabled
2 ADPC2	**ADC Pin Control 2** - ADPC2 controls the pin associated with channel AD2. 0 AD2 pin I/O control enabled 1 AD2 pin I/O control disabled
1 ADPC1	**ADC Pin Control 1** - ADPC1 controls the pin associated with channel AD1. 0 AD1 pin I/O control enabled 1 AD1 pin I/O control disabled
0 ADPC0	**ADC Pin Control 0** - ADPC0 controls the pin associated with channel AD0. 0 AD0 pin I/O control enabled 1 AD0 pin I/O control disabled

10.3.14 Pin Control 2 Register (APCTL2)

APCTL2 controls channels 8–15 of the ADC module.

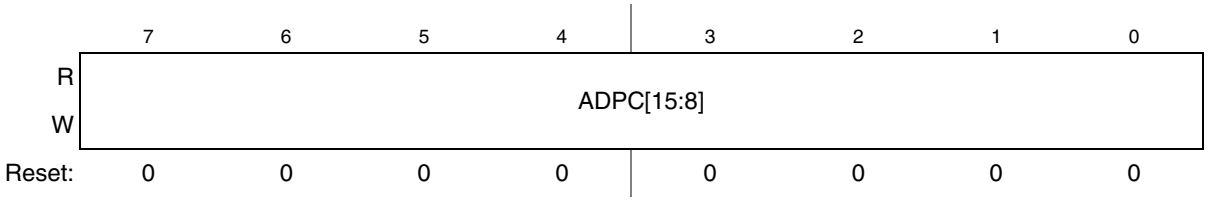


Figure 10-39. Pin Control 2 Register (APCTL2)

Table 10-17. APCTL2 Register Field Descriptions

Field	Description
7 ADPC15	**ADC Pin Control 15 -** ADPC15 controls the pin associated with channel AD15. 0 AD15 pin I/O control enabled 1 AD15 pin I/O control disabled
6 ADPC14	**ADC Pin Control 14** - ADPC14 controls the pin associated with channel AD14. 0 AD14 pin I/O control enabled 1 AD14 pin I/O control disabled
5 ADPC13	**ADC Pin Control 13** - ADPC13 controls the pin associated with channel AD13. 0 AD13 pin I/O control enabled 1 AD13 pin I/O control disabled
4 ADPC12	**ADC Pin Control 12** - ADPC12 controls the pin associated with channel AD12. 0 AD12 pin I/O control enabled 1 AD12 pin I/O control disabled
3 ADPC11	**ADC Pin Control 11** - ADPC11 controls the pin associated with channel AD11. 0 AD11 pin I/O control enabled 1 AD11 pin I/O control disabled
2 ADPC10	**ADC Pin Control 10** - ADPC10 controls the pin associated with channel AD10. 0 AD10 pin I/O control enabled 1 AD10 pin I/O control disabled
1 ADPC9	**ADC Pin Control 9** - ADPC9 controls the pin associated with channel AD9. 0 AD9 pin I/O control enabled 1 AD9 pin I/O control disabled
0 ADPC8	**ADC Pin Control 8** - ADPC8 controls the pin associated with channel AD8. 0 AD8 pin I/O control enabled 1 AD8 pin I/O control disabled

10.3.15 Pin Control 3 Register (APCTL3)

APCTL3 controls channels 23–16 of the ADC module.

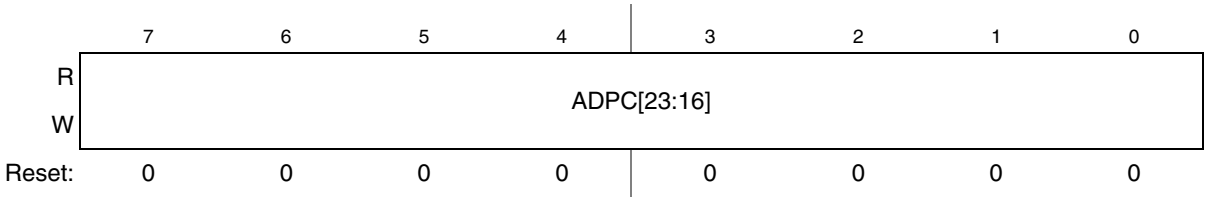


Figure 10-40. Pin Control 3 Register (APCTL3)

Table 10-18. APCTL3 Register Field Descriptions

Field	Description
7 ADPC23	**ADC Pin Control 23** - ADPC23 controls the pin associated with channel AD23. 0 AD23 pin I/O control enabled 1 AD23 pin I/O control disabled
6 ADPC22	**ADC Pin Control 22** - ADPC22 controls the pin associated with channel AD22. 0 AD22 pin I/O control enabled 1 AD22 pin I/O control disabled
5 ADPC21	**ADC Pin Control 21** - ADPC21 controls the pin associated with channel AD21. 0 AD21 pin I/O control enabled 1 AD21 pin I/O control disabled
4 ADPC20	**ADC Pin Control 20** - ADPC20 controls the pin associated with channel AD20. 0 AD20 pin I/O control enabled 1 AD20 pin I/O control disabled
3 ADPC19	**ADC Pin Control 19** - ADPC19 controls the pin associated with channel AD19. 0 AD19 pin I/O control enabled 1 AD19 pin I/O control disabled
2 ADPC18	**ADC Pin Control 18** - ADPC18 controls the pin associated with channel AD18. 0 AD18 pin I/O control enabled 1 AD18 pin I/O control disabled
1 ADPC17	**ADC Pin Control 17** - ADPC17 controls the pin associated with channel AD17. 0 AD17 pin I/O control enabled 1 AD17 pin I/O control disabled
0 ADPC16	**ADC Pin Control 16** - ADPC16 controls the pin associated with channel AD16. 0 AD16 pin I/O control enabled 1 AD16 pin I/O control disabled

10.3.16 Pin Control 4 Register (APCTL4)

APCTL4 controls channels DAD0-DAD3 of the ADC module. When DIFFn=1, channels DAD0-DAD3 use the input pin pairs DADPx and DADMx to form a differential conversion. When DIFFn=0, channels DAD0-DAD3 use only the input pins DADPx and the DADMx input pins are ingored.

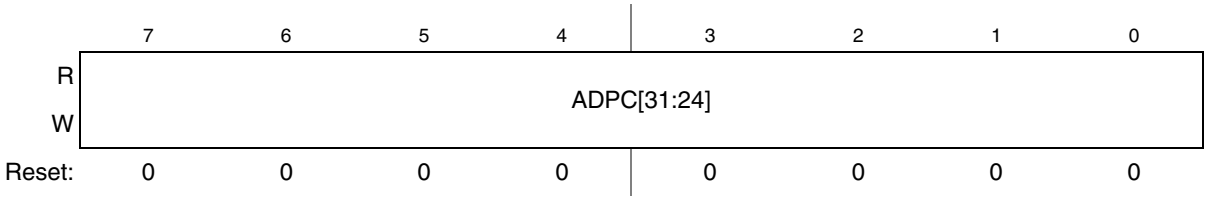


Figure 10-41. Pin Control 4 Register (APCTL4)

Table 10-19. APCTL4 Register Field Descriptions

Field	Description
7 ADPC31	**ADC Pin Control 31** - ADPC31 controls the pin associated with channel AD31. 0 AD31 pin I/O control enabled 1 AD31 pin I/O control disabled
6 ADPC30	**ADC Pin Control 30** - ADPC30 controls the pin associated with channel AD30. 0 AD30 pin I/O control enabled 1 AD30 pin I/O control disabled
5 ADPC29	**ADC Pin Control 29** - ADPC29 controls the pin associated with channel AD29. 0 AD29 pin I/O control enabled 1 AD29 pin I/O control disabled
4 ADPC28	**ADC Pin Control 28** - ADPC28 controls the pin associated with channel AD28. 0 AD28 pin I/O control enabled 1 AD28 pin I/O control disabled
3 ADPC27	**ADC Pin Control 27** - ADPC27 controls the pin associated with channel AD27. 0 AD27 pin I/O control enabled 1 AD27 pin I/O control disabled
2 ADPC26	**ADC Pin Control 26** - ADPC26 controls the pin associated with channel AD26. 0 AD26 pin I/O control enabled 1 AD26 pin I/O control disabled
1 ADPC25	**ADC Pin Control 25** - ADPC25 controls the pin associated with channel AD25. 0 AD25 pin I/O control enabled 1 AD25 pin I/O control disabled
0 ADPC24	**ADC Pin Control 24** - ADPC24 controls the pin associated with channel AD24. 0 AD24 pin I/O control enabled 1 AD24 pin I/O control disabled

10.4 Functional Description

The ADC module is disabled during reset, stop2 or when the ADCHn bits are all high. The module is idle when a conversion has completed and another conversion has not been initiated. When idle and the asynchronous clock output enable is disabled (ADACKEN=0), the module is in its lowest power state. The ADC can perform an analog-to-digital conversion on any of the software selectable channels. All modes perform conversion by a successive approximation algorithm.

To meet accuracy specifications, the ADC module must be calibrated using the on chip calibration function. Calibration is recommended to be done after any reset. See Section 10.4.7 for details on how to perform calibration.

When the conversion is completed, the result is placed in the data registers (ADCRHn and ADCRLn). The conversion complete flag (COCOn) is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled (AIENn=1).

The ADC module has the capability of automatically comparing the result of a conversion with the contents of the compare value registers. The compare function is enabled by setting the ACFE bit and operates with any of the conversion modes and configurations.

The ADC module has the capability of automatically averaging the result of multiple conversions. The hardware average function is enabled by setting the AVGE bit and operates with any of the conversion modes and configurations.

10.4.1 Clock Select and Divide Control

One of four clock sources can be selected as the clock source for the ADC module. This clock source is then divided by a configurable value to generate the input clock to the converter (ADCK). The clock is selected from one of the following sources by means of the ADICLK bits.

- The bus clock, which is equal to the frequency at which software is executed. This is the default selection following reset.
- The bus clock divided by two. For higher bus clock rates, this allows a maximum divide by 16 of the bus clock with using the ADIV bits.
- ALTCLK, as defined for this MCU (See module section introduction).
- The asynchronous clock (ADACK). This clock is generated from a clock source within the ADC module. Conversions are possible using ADACK as the input clock source while the MCU is in stop3 mode. Refer to Section 10.4.5.4 for more information.

Whichever clock is selected, its frequency must fall within the specified frequency range for ADCK. If the available clocks are too slow, the ADC may not perform according to specifications. If the available clocks are too fast, the clock must be divided to the appropriate frequency. This divider is specified by the ADIV bits and can be divide-by 1, 2, 4, or 8.

10.4.2 Input Select and Pin Control

The pin control registers (APCTL1, APCTL2, APCTL3, and APCTL4) disable the I/O port control of the pins used as analog inputs.When a pin control register bit is set, the following conditions are forced for the associated MCU pin:

- The output buffer is forced to its high impedance state.
- The input buffer is disabled. A read of the I/O port returns a zero for any pin with its input buffer disabled.
- The pullup is disabled.

10.4.3 Voltage Reference Selection

The ADC can be configured to accept one of three voltage reference pairs as the reference voltage (V_{REFSH} and V_{REFSL}) used for conversions. Each pair contains a positive reference which must be between the minimum Ref Voltage High (defined in Appendix A) and V_{DDAD}, and a ground reference which must be at the same potential as V_{SSAD}. The three pairs are external (V_{REFH} and V_{REFL}), alternate (V_{ALTH} and V_{ALTL}) and the internal bandgap (V_{BGH} and V_{BGL}). These voltage references are selected using the REFSEL bits in the ADCSC2 register.The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration.Consult the module introduction for information on the Voltage References specific to this MCU.

10.4.4 Hardware Trigger and Channel Selects

The ADC module has a selectable asynchronous hardware conversion trigger, ADHWT, that is enabled when the ADTRG bit is set and a hardware trigger select event (ADHWTSn) has occurred. This source is not available on all MCUs. Consult the module introduction for information on the ADHWT source and the ADHWTSn configurations specific to this MCU.

When the ADHWT source is available and hardware trigger is enabled (ADTRG=1), a conversion is initiated on the rising edge of the ADHWT after a hardware trigger select event (ADHWTSn) has occurred. If a conversion is in progress when a rising edge of a trigger occurs, the rising edge is ignored. In continuous convert configuration, only the initial rising edge to launch continuous conversions is observed and until conversion gets aborted the ADC will continue to do conversions on the same ADC Status and Control register that initiated the conversion. The hardware trigger function operates in conjunction with any of the conversion modes and configurations.

The hardware trigger select event (ADHWTSn) must be set prior to and during the receipt of the ADHWT signal. If these conditions are not met the converter may ignore the trigger or use the incorrect configuration. If a hardware trigger select event gets asserted during a conversion, it must stay asserted until end of current conversion and remain set until the receipt of the ADHWT signal to trigger a new conversion. The channel and status fields selected for the conversion will depend on the active trigger select signal (ADHWTSA active selects ADCSC1A; ADHWTSn active selects ADCSC1n).

NOTE

Asserting more than one hardware trigger select signal (ADHWTSn) at the same time will result in unknown results. To avoid this, only select one hardware trigger select signal (ADHWTSn) prior to the next intended conversion.

When the conversion is completed, the result is placed in the data registers associated with the ADHWTSn received (ADHWTSA active selects ADCRHA:ADCRLA; ADHWTSn active selects ADCRHn:ADCRLn). The conversion complete flag associated with the ADHWTSn received (COCOn) is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled (AIENn=1).

10.4.5 Conversion Control

Conversions can be performed as determined by the MODE bits and the DIFFn bit as shown in Table 10-10.

Conversions can be initiated by a software or hardware trigger. In addition, the ADC module can be configured for low power operation, long sample time, continuous conversion, hardware average and automatic compare of the conversion result to a software determined compare value.

10.4.5.1 Initiating Conversions

A conversion is initiated:

- Following a write to ADCSC1A (with ADCHA bits not all 1’s) if software triggered operation is selected (ADTRG=0).

- Following a hardware trigger (ADHWT) event if hardware triggered operation is selected (ADTRG=1) and a hardware trigger select event (ADHWTSn) has occurred. The channel and status fields selected will depend on the active trigger select signal (ADHWTSA active selects ADCSC1A; ADHWTSn active selects ADCSC1n; if neither is active the off condition is selected).

NOTE

Selecting more than one hardware trigger select signal (ADHWTSn) prior to a conversion completion will result in unknown results. To avoid this, only select one hardware trigger select signal (ADHWTSn) prior to a conversion completion.

- Following the transfer of the result to the data registers when continuous conversion is enabled (ADCO=1).

If continuous conversions are enabled, a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation (ADTRG=0), continuous conversions begin after ADCSC1A is written and continue until aborted. In hardware triggered operation(ADTRG=1 and one ADHWTSn event has occurred), continuous conversions begin after a hardware trigger event and continue until aborted.

If hardware averaging is enabled, a new conversion is automatically initiated after the completion of the current conversion until the correct number of conversions is completed. In software triggered operation, conversions begin after ADCSC1A is written. In hardware triggered operation, conversions begin after a hardware trigger. If continuous conversions are also enabled, a new set of conversions to be averaged are initiated following the last of the selected number of conversions.

10.4.5.2 Completing Conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, ADCRHn and ADCRLn. If the compare functions are disabled, this is indicated by the setting of COCOn. If hardware averaging is enabled, COCOn sets only if the last of the selected number of conversions is complete. If the compare function is enabled, COCOn sets and conversion result data is transferred only if the compare condition is true. If both hardware averaging and compare functions are enabled then COCOn sets only if the last of the selected number of conversions is complete and the compare condition is true. An interrupt is generated if AIENn is high at the time that COCOn is set.In all modes except 8-bit single-ended conversions, a blocking mechanism prevents a new result from overwriting previous data in ADCRHn and ADCRLn if the previous data is in the process of being read (the ADCRHn register has been read but the ADCRLn register has not). When blocking is active, the conversion result data transfer is blocked, COCOn is not set, and the new result is lost. In the case of single conversions with the compare function enabled and the compare condition false, blocking has no effect and ADC operation is terminated. In all other cases of operation, when a conversion result data transfer is blocked, another conversion is initiated regardless of the state of ADCO (single or continuous conversions enabled).

NOTE

If continuous conversions are enabled, the blocking mechanism could result in the loss of data occurring at specific timepoints. To avoid this issue, the data must be read in fewer cycles than an ADC conversion time, accounting for interrupt or software polling loop latency.

If single conversions are enabled, the blocking mechanism could result in several discarded conversions and excess power consumption. To avoid this issue, the data registers must not be read after initiating a single conversion until the conversion completes.

10.4.5.3 Aborting Conversions

Any conversion in progress is aborted when:

- Writing ADCSC1A while ADCSC1A is actively controlling a conversion aborts the current conversion. In software trigger mode (ADTRG=0), a write to ADCSC1A initiates a new conversion (if the ADCHA bits are equal to a value other than all 1s). Writing any of the ADCSC1(B-n) registers while that specific ADCSC1(B-n) register is actively controlling a conversion aborts the current conversion.The ADCSC1(B-n) registers are not used for software trigger operation and therefore writes to the ADCSC1(B-n) registers do not initiate a new conversion.
- A write to any ADC register besides the ADCSC1A:ADCSC1n registers occurs. This indicates a mode of operation change has occurred and the current conversion is therefore invalid.
- The MCU is reset or enters stop2 mode.
- The MCU enters stop3 mode with ADACK not enabled.

When a conversion is aborted, the contents of the data registers, ADCRHn and ADCRLn, are not altered. The data registers continue to be the values transferred after the completion of the last successful conversion. If the conversion was aborted by a reset or stop2, ADCRHA:ADCRLA and ADCRHn:ADCRLn return to their reset states.

10.4.5.4 Power Control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source but the asynchronous clock output is disabled (ADACKEN=0), the ADACK clock generator will also remain in its idle state (disabled) until a conversion is initiated. If the asynchronous clock output is enabled (ADACKEN=1), it will remain active regardless of the state of the ADC or the MCU power mode.

Power consumption when the ADC is active can be reduced by setting ADLPC. This results in a lower maximum value for f_{ADCK} (see the electrical specifications).

10.4.5.5 Sample Time and Total Conversion Time

The total conversion time depends upon: the sample time (as determined by ADLSMP and ADLSTS bits), the MCU bus frequency, the conversion mode (as determined by MODE and DIFFn bits), the high speed configuration (ADHSC bit), and the frequency of the conversion clock (f_{ADCK}).

The ADHSC bit is used to configure a higher clock input frequency. This will allow faster overall conversion times. In order to meet internal A/D converter timing requirements the ADHSC bit adds additional ADCK cycles. Conversions with ADHSC = 1 take four more ADCK cycles. ADHSC should be used when the ADCLK exceeds the limit for ADHSC = 0.

After the module becomes active, sampling of the input begins. ADLSMP and ADLSTS select between sample times based on the conversion mode that is selected. When sampling is complete, the converter is isolated from the input channel and a successive approximation algorithm is performed to determine the digital value of the analog signal. The result of the conversion is transferred to ADCRHn and ADCRLn upon completion of the conversion algorithm.

If the bus frequency is less than the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled (ADLSMP=0). The maximum total conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is selectable by the ADICLK bits, and the divide ratio is specified by the ADIV bits. The maximum total conversion time for all configurations is summarized in the equation below. Refer to Table 10-20 through Table 10-24 for the variables referenced in the equation.

Conversion Time Equation ***Eqn. 10-2***

$$ConversionTime = SFCAdder + AverageNum \times (BCT + LSTAdder + HSCAdder)$$

Table 10-20. Single or First Continuous Time Adder (SFCAdder)

ADLSMP	ADACKEN	ADICLK	Single or First Continuous Time Adder (SFCAdder)
1	x	0x, 10	3 ADCK cycles + 5 bus clock cycles
1	1	11	3 ADCK cycles + 5 bus clock cycles[1]
1	0	11	5μs + 3 ADCK cycles + 5 bus clock cycles
0	x	0x, 10	5 ADCK cycles + 5 bus clock cycles
0	1	11	5 ADCK cycles + 5 bus clock cycles[1]
0	0	11	5μs + 5 ADCK cycles + 5 bus clock cycles

[1] ADACKEN must be 1 for at least 5us prior to the conversion is initiated to achieve this time

Table 10-21. Average Number Factor (AverageNum)

AVGE	AVGS[1:0]	Average Number Factor (AverageNum)
0	xx	1
1	00	4
1	01	8
1	10	16
1	11	32

Table 10-22. Base Conversion Time (BCT)

Mode	Base Conversion Time (BCT)
8b se	17 ADCK cycles
9b diff	27 ADCK cycles
10b s.e.	20 ADCK cycles
11b diff	30 ADCK cycles
12b s.e.	20 ADCK cycles
13b diff	30 ADCK cycles
16b s.e.	25 ADCK cycles
16b diff	34 ADCK cycles

Table 10-23. Long Sample Time Adder (LSTAdder)

ADLSMP	ADLSTS	Long Sample Time Adder (LSTAdder)
0	xx	0 ADCK cycles
1	00	20 ADCK cycles
1	01	12 ADCK cycles
1	10	6 ADCK cycles
1	11	2 ADCK cycles

Table 10-24. High-Speed Conversion Time Adder (HSCAdder)

ADHSC	High Speed Conversion Time Adder (HSCAdder)
0	0 ADCK cycles
1	4 ADCK cycles

NOTE

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

10.4.5.6 Conversion Time Examples

The following examples use Equation 10-3 and the information provided in Table 10-20 through Table 10-24.

10.4.5.6.1 Typical conversion time configuration

A typical configuration for ADC conversion is: 10-bit mode, with the bus clock selected as the input clock source, the input clock divide-by-1 ratio selected, and a bus frequency of 8 MHz, long sample time disabled and high speed conversion disabled. The conversion time for a single conversion is calculated by using Equation 10-3 and the information provided in Table 10-20 through Table 10-24. The table below lists the variables of Equation 10-3.

Table 10-25. Typical Conversion Time

Variable	Time
SFCAdder	5 ADCK cycles + 5 bus clock cycles
AverageNum	1
BCT	20 ADCK cycles
LSTAdder	0
HSCAdder	0

The resulting conversion time is generated using the parameters listed in Table 10-24. So, for Bus clock equal to 8 MHz and ADCK equal to 8 MHz, the resulting conversion time is 3.75 us.

10.4.5.6.2 Long conversion time configuration

A configuration for long ADC conversion is: 16-bit differential mode, with the bus clock selected as the input clock source, the input clock divide-by-8 ratio selected, and a bus frequency of 8 MHz, long sample time enabled and configured for longest adder and high speed conversion disabled. Average enabled for 32 conversions. The conversion time for this conversion is calculated by using Equation 10-3 and the information provided in Table 10-20 through Table 10-24. The table below list the variables of Equation 10-3.

Table 10-26. Typical Conversion Time

Variable	Time
SFCAdder	3 ADCK cycles + 5 bus clock cycles
AverageNum	32
BCT	34 ADCK cycles
LSTAdder	20 ADCK cycles
HSCAdder	0

The resulting conversion time is generated using the parameters listed in Table 10-24. So, for Bus clock equal to 8 MHz and ADCK equal to 1 MHz the resulting conversion time is 57.625 us (AverageNum). This results in a total conversion time of 1.844 ms.

10.4.5.7 Hardware Average Function

The hardware average function can be enabled (AVGE=1) to perform a hardware average of multiple conversions. The number of conversions is determined by the AVGS[1:0] bits, which select 4, 8, 16 or 32 conversions to be averaged. While the hardware average function is in progress, the ADACT bit is set.

After the selected input is sampled and converted, the result is placed in an accumulator from which an average is calculated once the selected number of conversions has been completed. When hardware averaging is selected the completion of a single conversion will not set the COCOn bit.

If the compare function is either disabled or evaluates true, after the selected number of conversions are completed, the average conversion result is transferred into the data result registers, ADCRHn and ADCRLn, and the COCOn bit is set. An ADC interrupt is generated upon the setting of COCOn if the respective ADC interrupt is enabled (AIENn=1).

NOTE

The hardware average function can perform conversions on a channel while the MCU is in wait or stop3 mode. The ADC interrupt wakes the MCU when the hardware average is complete if AIENn was set.

10.4.6 Automatic Compare Function

The compare function can be configured to check if the result is less than or greater-than-or-equal-to a single compare value, or if the result falls within or outside a range determined by two compare values. The compare mode is determined by ACFGT, ACREN and the values in the compare value registers(ADCCV1H:ADCCV1L and ADCCV2H:ADCCV2L). After the input is sampled and converted,

the compare values (ADCCV1H:ADCCV1L and ADCCV2H:ADCCV2L) are used as described in Table 10-27.There are six compare modes as shown in Table 10-27.

Table 10-27. Compare Modes

ACFGT	ACREN	ADCCV1 relative to ADCCV2	Function	Compare Mode Description
0	0	-	Less than threshold	Compare true if the result is less than the ADCCV1 registers.
1	0	-	Greater than or equal to threshold	Compare true if the result is greater than or equal to ADCCV1 registers.
0	1	Less than or equal	Outside range, not inclusive	Compare true if the result is less than ADCCV1 **Or** the result is Greater than ADCCV2
0	1	Greater than	Inside range, not inclusive	Compare true if the result is less than ADCCV1 **And** the result is greater than ADCCV2
1	1	Less Than or equal	Inside range, inclusive	Compare true if the result is greater than or equal to ADCCV1 **And** the result is less than or equal to ADCCV2
1	1	Greater than	Outside range, inclusive	Compare true if the result is greater than or equal to ADCCV1 **Or** the result is less than or equal to ADCCV2

With the ADC range enable bit set, ADCREN =1, if compare value register 1(ADCCV1 value) is less than or equal to the compare value register 2 (ADCCV2 value), setting ACFGT will select a trigger-if-inside-compare-range, inclusive-of-endpoints function. Clearing ACFGT will select a trigger-if-outside-compare-range, not-inclusive-of-endpoints function.

If ADCCV1 is greater than the ADCCV2, setting ACFGT will select a trigger-if-outside-compare-range, inclusive-of-endpoints function. Clearing ACFGT will select a trigger-if-inside-compare-range, not-inclusive-of-endpoints function.

If the condition selected evaluates true, COCOn is set.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, COCOn is not set and the conversion result data will not be transferred to the result register. If the hardware averaging function is enabled, the compare function compares the averaged result to the compare values. The same compare function definitions apply. An ADC interrupt is generated upon the setting of COCOn if the respective ADC interrupt is enabled (AIENn=1).

NOTE

The compare function can monitor the voltage on a channel while the MCU is in wait or stop3 mode. The ADC interrupt wakes the MCU when the compare condition is met.

10.4.7 Calibration Function

The ADC contains a self-calibration function that is required to achieve the specified accuracy. Calibration must be run or valid calibration values written after any reset and before a conversion is initiated. The calibration function sets the offset calibration value and the plus-side and minus-side calibration values.

The offset calibration value is automatically stored in the ADC Offset Correction Registers (ADCOFSH and ADCOFSL) and the plus-side and minus-side calibration values are automatically stored in the ADC Plus-Side and Minus-Side Calibration registers (CLPD, CLPS, CLP4, CLP3, CLP2, CLP1, CLP0 and CLMD, CLMS, CLP4, CLM3, CLM2, CLM1, CLM0). The user must configure the ADC correctly prior to calibration, and must generate the plus-side and minus-side gain calibration results and store them in the ADC GAIN registers (ADCPGH and ADCPGL) after the calibration function completes.

Prior to calibration, the user must configure the ADC's clock frequency to be between 2 MHz and 4 MHz, set to high speed mode (ADLPC = 0, ADHSC = 1), and set to maximum averaging (AVGE = 1, AVGS = 11). It is recommended that calibration be run at $V_{DDA} = V_{REFH} \geq 3V$. The input channel, conversion mode continuous function, compare function, resolution mode, and differential/single-ended mode are all ignored during the calibration function.

To initiate calibration, the user sets the CAL bit and the calibration will automatically begin if the ADTRG bit = 0. If ADTRG = 1, the CAL bit will not get set and the calibration fail flag (CALF) will be set. While calibration is active, no ADC register can be written and no stop mode may be entered or the calibration routine will be aborted causing the CAL bit to clear and the CALF bit to set. At the end of a calibration sequence the COCO bit of the ADSC1A register will be set. The AIEN1 bit can be used to allow an interrupt to occur at the end of a calibration sequence. If at the end of calibration routine the CALF bit is not set, the automatic calibration routine completed successfully.

To complete calibration, the user must generate the gain calibration values using the following procedure:

- Initialize (clear) a 16b variable in RAM.
- Add the following plus-side calibration results CLP0, CLP1, CLP2, CLP3, CLP4, and CLPS to the variable.
- Divide the variable by two.
- Set the MSB of the variable.
- The previous two steps can be achieved by setting the carry bit, rotating-right through the carry bit on the high byte and again on the low byte.
- Store the value in the plus-side gain calibration registers ADCPGH and ADCPGL.
- Repeat procedure for the minus-side gain calibration value.

When complete the user may reconfigure and use the ADC as desired. A second calibration may also be performed if desired by clearing and again setting the CAL bit.

Overall the calibration routine may take as many as 14000 ADCK cycles and 100 bus cycles, depending on the results and the clock source chosen. For an 8 MHz clock source this is about 1.7 msec. To reduce this latency, the calibration values (offset, plus- and minus-side gain, and plus- and minus-side calibration values) may be stored in flash after an initial calibration and recovered prior to the first ADC conversion. This should reduce the calibration latency to 20 register store operations on all subsequent power, reset, or stop2 mode recoveries.

10.4.8 User Defined Offset Function

The ADC Offset Correction Register (ADCOFSH:ADCOFSL) contains the user selected or calibration generated offset error correction value. This register is a 2’s complement, left justified, 16b value formed

by the concatenation of ADCOFSH and ADCOFSL. The value in the offset correction registers (ADCOFSH:ADCOFSL) is subtracted from the conversion and the result is transferred into the result registers (ADCRHn:ADCRLn). If the result is above the maximum or below the minimum result value, it is forced to the appropriate limit for the current mode of operation.For additional information please see Section 10.4.8

The formatting of the ADC Offset Correction Register is different from the Data Result Registers (ADCRHn:ADCRLn) to preserve the resolution of the calibration value regardless of the conversion mode selected. Lower order bits are ignored in lower resolution modes. For example, in 8b single-ended mode, the bits OFS[14:7] are subtracted from D[7:0]; bit OFS[15] indicates the sign (negative numbers are effectively added to the result) and bits OFS[6:0] are ignored. The same bits are used in 9b differential mode since bit OFS[15] indicates the sign bit, which maps to bit D[8]. For 16b differential mode, all bits OFS[15:0] are directly subtracted from the conversion result data D[15:0]. Finally, in 16b single-ended mode, there is no bit in the Offset Correction Register corresponding to the least significant result bit D[0], so odd values (-1 or +1, etc.) cannot be subtracted from the result.ADCOFSH is automatically set according to calibration requirements once the self calibration sequence is done (CAL is cleared). Write ADCOFSH:ADCOFSL to override the calibration result if desired. If the Offset Correction Register is written to a value that is different from the calibration value, the ADC error specifications may not be met. It is recommended that the value generated by the calibration function be stored in memory before overwriting with a specified value.

NOTE

There is an effective limit to the values of Offset that can be set. If the magnitude of the offset is too great, the results of the conversions cap off at the limits.

Use the offset calibration function to remove application offsets or DC bias values. The Offset Correction Registers ADCOFSH and ADCOFSL may be written with a number in 2's complement format and this offset will be subtracted from the result (or hardware averaged value). To add an offset, store the negative offset in 2's complement format and the effect will be an addition. An offset correction that results in an out-of-range value will be forced to the minimum or maximum value (the minimum value for single-ended conversions is 0x0000; for a differential conversion 0x8000).

To preserve accuracy, the calibrated offset value initially stored in the ADCOFS registers must be added to the user defined offset. For applications which may change the offset repeatedly during operation, it is recommended to store the initial offset calibration value in flash so that it can be recovered and added to any user offset adjustment value -nd the sum stored in the ADCOFS registers.

10.4.9 Temperature Sensor

The ADC module includes a temperature sensor whose output is connected to one of the ADC analog channel inputs. Equation 10-3 provides an approximate transfer function of the temperature sensor.

$$\text{Temp} = 25 - ((V_{TEMP} - V_{TEMP25}) \div m) \qquad \textit{Eqn. 10-3}$$

where:

— V_{TEMP} is the voltage of the temperature sensor channel at the ambient temperature.

— V_{TEMP25} is the voltage of the temperature sensor channel at 25°C.
— m is the hot or cold voltage versus temperature slope in V/°C.

For temperature calculations, use the V_{TEMP25} and m values from the ADC Electricals table.

In application code, the user reads the temperature sensor channel, calculates V_{TEMP}, and compares to V_{TEMP25}. If V_{TEMP} is greater than V_{TEMP25} the cold slope value is applied in Equation 10-3. If V_{TEMP} is less than V_{TEMP25} the hot slope value is applied in Equation 10-3.

For more information on using the temperature sensor, consult AN3031.

10.4.10 MCU Wait Mode Operation

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

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

If the compare and hardware averaging functions are disabled, a conversion complete event sets the COCOn and generates an ADC interrupt to wake the MCU from wait mode if the respective ADC interrupt is enabled (AIENn=1). If the hardware averaging function is enabled the COCOn will set (and generate an interrupt if enabled) when the selected number of conversions are complete. If the compare function is enabled the COCOn will set (and generate an interrupt if enabled) only if the compare conditions are met. If a single conversion is selected and the compare trigger is not met, the ADC will return to its idle state and cannot wake the MCU from wait mode unless a new conversion is initiated by the hardware trigger.

10.4.11 MCU Stop3 Mode Operation

Stop mode is a low power-consumption standby mode during which most or all clock sources on the MCU are disabled.

10.4.11.1 Stop3 Mode With ADACK Disabled

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

10.4.11.2 Stop3 Mode With ADACK Enabled

If ADACK is selected as the conversion clock, the ADC continues operation during stop3 mode. For guaranteed ADC operation, the MCU's voltage regulator must remain active during stop3 mode. Consult the module introduction for configuration information for this MCU.

If a conversion is in progress when the MCU enters stop3 mode, it continues until completion. Conversions can be initiated while the MCU is in stop3 mode by means of the hardware trigger or if continuous conversions are enabled.

If the compare and hardware averaging functions are disabled, a conversion complete event sets the COCOn and generates an ADC interrupt to wake the MCU from stop3 mode if the respective ADC interrupt is enabled (AIENn = 1). The result register will contain the data from the first completed conversion that occurred during stop3 mode. If the hardware averaging function is enabled the COCOn will set (and generate an interrupt if enabled) when the selected number of conversions are complete. If the compare function is enabled the COCOn will set (and generate an interrupt if enabled) only if the compare conditions are met. If a single conversion is selected and the compare is not true, the ADC will return to its idle state and cannot wake the MCU from stop3 mode unless a new conversion is initiated by another hardware trigger.

NOTE

The ADC module can wake the system from low-power stop and cause the MCU to begin consuming run-level currents without generating a system level interrupt. To prevent this scenario, software should ensure the conversion result data transfer blocking mechanism (discussed in Section 10.4.5.2, “Completing Conversions) is cleared when entering stop3 and continuing ADC conversions.

10.4.12 MCU Stop2 Mode Operation

The ADC module is automatically disabled when the MCU enters stop2 mode. All module registers contain their reset values following exit from stop2. Therefore, the module must be re-enabled and re-configured following exit from stop2.

10.5 Initialization Information

This section gives an example that provides some basic direction on how to initialize and configure the ADC module. You can configure the module for 8-, 10-, 12-, or 16-bit single-ended resolution or 9-, 11-, 13-, or 16-bit differential resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. Refer to Table 10-9, Table 10-10, and Table 10-11 for information used in this example.

NOTE

Hexadecimal values designated by a preceding 0x, binary values designated by a preceding %, and decimal values have no preceding character.

10.5.1 ADC Module Initialization Example

10.5.1.1 Initialization Sequence

Before the ADC module can be used to complete conversions, an initialization procedure must be performed. A typical sequence is as follows:

1. Calibrate the ADC by following the calibration instructions in Section 10.4.7.
2. Update the configuration register (ADCCFG) to select the input clock source and the divide ratio used to generate the internal clock, ADCK. This register is also used for selecting sample time and low-power configuration.
3. Update status and control register 2 (ADCSC2) to select the conversion trigger (hardware or software) and compare function options, if enabled.
4. Update status and control register 3 (ADSC3) to select whether conversions will be continuous or completed only once (ADCO) and to select whether to perform hardware averaging.
5. Update status and control register (ADCSC1:ADCSC1n) to select whether conversions will be single-ended or differential and to enable or disable conversion complete interrupts. The input channel on which conversions will be performed is also selected here.

10.5.1.2 Pseudo-Code Example

In this example, the ADC module is set up with interrupts enabled to perform a single 10-bit conversion at low power with a long sample time on input channel 1, where the internal ADCK clock is derived from the bus clock divided by 1.

ADCCFG = 0x98 (%10011000)

```
Bit 7    ADLPC    1        Configures for low power (lowers maximum clock speed.
Bit 6:5  ADIV     00       Sets the ADCK to the input clock ÷ 1.
Bit 4    ADLSMP   1        Configures for long sample time.
Bit 3:2  MODE     10       Sets mode at 10-bit conversions.
Bit 1:0  ADICLK   00       Selects bus clock as input clock source.
```

ADCSC2 = 0x00 (%00000000)

```
Bit 7    ADACT    0        Flag indicates if a conversion is in progress.
Bit 6    ADTRG    0        Software trigger selected.
Bit 5    ACFE     0        Compare function disabled.
Bit 4    ACFGT    0        Not used in this example.
Bit 3:2           00       Reserved, always reads zero.
Bit 1:0           00       Reserved for Freescale's internal use; always write zero.
```

ADCSC1A = 0x41 (%01000001)

```
Bit 7    COCOA    0        Read-only flag which is set when a conversion completes.
Bit 6    AIENA    1        Conversion complete interrupt enabled.
Bit 5    ADCOA    0        One conversion only (continuous conversions disabled).
Bit 4:0  ADCHA    00001    Input channel 1 selected as ADC input channel.
```

ADCRHA/LA = 0xxx

```
Holds results of conversion. Read high byte (ADCRHA) before low byte (ADCRLA) so that
conversion data cannot be overwritten with data from the next conversion.
```

ADCCVH/L = 0xxx

Holds compare value when compare function enabled.

APCTL1=0x02

AD1 pin I/O control disabled. All other AD pins remain general purpose I/O pins.

APCTL2=0x00

All other AD pins remain general purpose I/O pins.

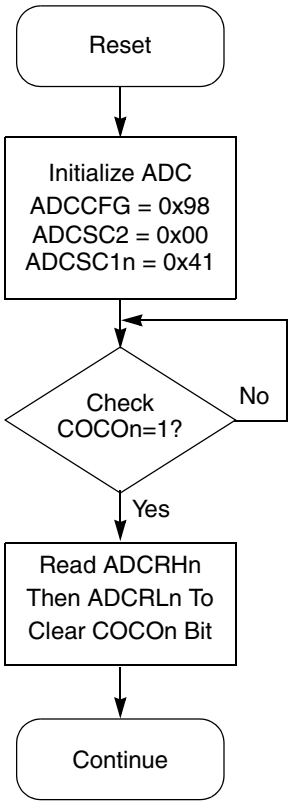


Figure 10-42. Initialization Flowchart for Example

10.6 Application Information

This section contains information for using the ADC module in applications. The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an A/D converter.

10.6.1 External Pins and Routing

The following sections discuss the external pins associated with the ADC module and how they should be used for best results.

10.6.1.1 Analog Supply Pins

The ADC module has analog power and ground supplies (V_{DDAD} and V_{SSAD}) available as separate pins on some devices. V_{SSAD} is shared on the same pin as the MCU digital V_{SS} on some devices. On other devices, V_{SSAD} and V_{DDAD} are shared with the MCU digital supply pins. In these cases, there are separate pads for the analog supplies bonded to the same pin as the corresponding digital supply so that some degree of isolation between the supplies is maintained.

When available on a separate pin, both V_{DDAD} and V_{SSAD} must be connected to the same voltage potential as their corresponding MCU digital supply (V_{DD} and V_{SS}) and must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

If separate power supplies are used for analog and digital power, the ground connection between these supplies must be at the V_{SSAD} pin. This should be the only ground connection between these supplies if possible. The V_{SSAD} pin makes a good single point ground location.

10.6.1.2 Analog Voltage Reference Pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs used by the converter, V_{REFSH} and V_{REFSL}. V_{REFSH} is the high reference voltage for the converter. V_{REFSL} is the low reference voltage for the converter.

The ADC can be configured to accept one of three voltage reference pairs for V_{REFSH} and V_{REFSL}. Each pair contains a positive reference and a ground reference. The three pairs are external (V_{REFH} and V_{REFL}), alternate (V_{ALTH} and V_{ALTL}) and the internal bandgap (V_{BGH} and V_{BGL}). These voltage references are selected using the REFSEL bits in the ADCSC2 register. The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration. Consult the module introduction for information on the Voltage References specific to this MCU.

In some packages, the external or alternate pairs are connected in the package to V_{DDAD} and V_{SSAD}, respectively. One of these positive references may be shared on the same pin as V_{DDAD} on some devices. One of these ground references may be shared on the same pin as V_{SSAD} on some devices.

If externally available, the positive reference may be connected to the same potential as V_{DDAD} or may be driven by an external source to a level between the minimum Ref Voltage High (defined in Appendix A) and the V_{DDAD} potential (the positive reference must never exceed V_{DDAD}). If externally available, the ground reference must be connected to the same voltage potential as V_{SSAD}. The voltage reference pairs must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

AC current in the form of current spikes required to supply charge to the capacitor array at each successive approximation step is drawn through the V_{REFH} and V_{REFL} loop. The best external component to meet this current demand is a 0.1 μF capacitor with good high frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as near as possible to the package pins. Resistance in the path is not recommended because the current causes a voltage drop that could result in conversion errors. Inductance in this path must be minimum (parasitic only).

10.6.1.3 Analog Input Pins

The external analog inputs are typically shared with digital I/O pins on MCU devices. The pin I/O control is disabled by setting the appropriate control bit in one of the pin control registers. Conversions can be performed on inputs without the associated pin control register bit set. It is recommended that the pin control register bit always be set when using a pin as an analog input. This avoids problems with contention because the output buffer is in its high impedance state and the pullup is disabled. Also, the input buffer draws DC current when its input is not at V_{DD} or V_{SS}. Setting the pin control register bits for all pins used as analog inputs should be done to achieve lowest operating current.

Empirical data shows that capacitors on the analog inputs improve performance in the presence of noise or when the source impedance is high. Use of 0.01 μF capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used they must be placed as near as possible to the package pins and be referenced to V_{SSA}.

For proper conversion, the input voltage must fall between V_{REFH} and V_{REFL}. If the input is equal to or exceeds V_{REFH}, the converter circuit converts the signal to 0xFFF (full scale 12-bit representation), 0x3FF (full scale 10-bit representation) or 0xFF (full scale 8-bit representation). If the input is equal to or less than V_{REFL}, the converter circuit converts it to 0x000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. There is a brief current associated with V_{REFL} when the sampling capacitor is charging.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins should not be transitioning during conversions.

10.6.2 Sources of Error

Several sources of error exist for A/D conversions. These are discussed in the following sections.

10.6.2.1 Sampling Error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy. Given the maximum input resistance of approximately 7kΩ and input capacitance of approximately 5.5 pF, sampling to within 1/4LSB (at 12-bit resolution) can be achieved within the minimum sample window (3.5 cycles @ 8 MHz maximum ADCK frequency) provided the resistance of the external analog source (R_{AS}) is kept below 2 kΩ.

Higher source resistances or higher-accuracy sampling is possible by setting ADLSMP and changing the ADLSTS bits (to increase the sample window) or decreasing ADCK frequency to increase sample time.

10.6.2.2 Pin Leakage Error

Leakage on the I/O pins can cause conversion error if the external analog source resistance (R_{AS}) is high. If this error cannot be tolerated by the application, keep R_{AS} lower than $V_{DDAD} / (2^N * I_{LEAK})$ for less than 1/4LSB leakage error (N = 8 in 8-bit, 10 in 10-bit or 12 in 12-bit mode).

10.6.2.3 Noise-Induced Errors

System noise that occurs during the sample or conversion process can affect the accuracy of the conversion. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1 μF low-ESR capacitor from V_{REFH} to V_{REFL}.
- There is a 0.1 μF low-ESR capacitor from V_{DDAD} to V_{SSAD}.
- If inductive isolation is used from the primary supply, an additional 1 μF capacitor is placed from V_{DDAD} to V_{SSAD}.
- V_{SSAD} (and V_{REFL}, if connected) is connected to V_{SS} at a quiet point in the ground plane.
- Operate the MCU in wait or stop3 mode before initiating (hardware triggered conversions) or immediately after initiating (hardware or software triggered conversions) the ADC conversion.
  - For software triggered conversions, immediately follow the write to ADCSC1 with a wait instruction or stop instruction.
  - For stop3 mode operation, select ADACK as the clock source. Operation in stop3 reduces V_{DD} noise but increases effective conversion time due to stop recovery.
- There is no I/O switching, input or output, on the MCU during the conversion.

There are some situations where external system activity causes radiated or conducted noise emissions or excessive V_{DD} noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in wait or stop3 or I/O activity cannot be halted, these recommended actions may reduce the effect of noise on the accuracy:

- Place a 0.01 μF capacitor (C_{AS}) on the selected input channel to V_{REFL} or V_{SSAD} (this improves noise issues, but affects the sample rate based on the external analog source resistance).
- Average the result by converting the analog input many times in succession and dividing the sum of the results. Four samples are required to eliminate the effect of a 1LSB, one-time error.
- Reduce the effect of synchronous noise by operating off the asynchronous clock (ADACK) and averaging. Noise that is synchronous to ADCK cannot be averaged out.

10.6.2.4 Code Width and Quantization Error

The ADC quantizes the ideal straight-line transfer function into 4096 steps (in 12-bit mode). Each step ideally has the same height (1 code) and width. The width is defined as the delta between the transition points to one code and the next. The ideal code width for an N bit converter (in this case N can be 8, 10 or 12), defined as 1LSB, is:

$$1\ lsb = (V_{REFH} - V_{REFL}) / 2^N \qquad \textit{Eqn. 10-4}$$

There is an inherent quantization error due to the digitalization of the result. For 8-bit or 10-bit conversions the code transitions when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be ± 1/2 lsb in 8- or 10-bit mode. As a consequence, however, the code width of the first (0x000) conversion is only 1/2 lsb and the code width of the last (0xFF or 0x3FF) is 1.5 lsb.

For 12-bit conversions the code transitions only after the full code width is present, so the quantization error is –1 lsb to 0 lsb and the code width of each step is 1 lsb.

10.6.2.5 Linearity Errors

The ADC may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors but the system should be aware of them because they affect overall accuracy. These errors are:

- Zero-scale error (E_{ZS}) (sometimes called offset) — This error is defined as the difference between the actual code width of the first conversion and the ideal code width (1/2 lsb in 8-bit or 10-bit modes and 1 lsb in 12-bit mode). If the first conversion is 0x001, the difference between the actual 0x001 code width and its ideal (1 lsb) is used.
- Full-scale error (E_{FS}) — This error is defined as the difference between the actual code width of the last conversion and the ideal code width (1.5 lsb in 8-bit or 10-bit modes and 1LSB in 12-bit mode). If the last conversion is 0x3FE, the difference between the actual 0x3FE code width and its ideal (1LSB) is used.
- Differential non-linearity (DNL) — This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral non-linearity (INL) — This error is defined as the highest-value the (absolute value of the) running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total unadjusted error (TUE) — This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function and includes all forms of error.

10.6.2.6 Code Jitter, Non-Monotonicity, and Missing Codes

Analog-to-digital converters are susceptible to three special forms of error. These are code jitter, non-monotonicity, and missing codes.

Code jitter is when, at certain points, a given input voltage converts to one of two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the converter yields the lower code (and vice-versa). However, even small amounts of system noise can cause the converter to be indeterminate (between two codes) for a range of input voltages around the transition voltage. This range is normally around ±1/2 lsb in 8-bit or 10-bit mode, or around 2 lsb in 12-bit mode, and increases with noise.

This error may be reduced by repeatedly sampling the input and averaging the result. Additionally the techniques discussed in Section 10.6.2.3 reduces this error.

Non-monotonicity is defined as when, except for code jitter, the converter converts to a lower code for a higher input voltage. Missing codes are those values never converted for any input value.

In 8-bit or 10-bit mode, the ADC is guaranteed to be monotonic and have no missing codes.

Chapter 11
Internal Clock Source (S08ICSV3)

11.1 Introduction

The internal clock source (ICS) module provides clock source choices for the MCU. For this device, the ICS can generate up to maximum MCU speed at 40 MHz or at 20 MHz bus speed. The module contains a frequency-locked loop (FLL) as a clock source that is controllable by either an internal or an external reference clock. The module can provide this FLL clock or either of the internal or external reference clocks as a source for the MCU system clock. There are also signals provided to control a low power oscillator (XOSCVLP) module to allow the use of an external crystal/resonator as the external reference clock.

The ICSTRM and FTRIM bits are normally reset to the factory trim values on any reset. However, any reset that puts the device into BDM (a POR with the BKGD pin held low or a development tool setting SBDFR[BDFR]) results in the ICSTRM and FTRIM bits being set. When debugging the MCU, the factory trim value can be used by copying the trim values from the Flash locations shown in Table 4-5.

Whichever clock source is chosen, it is passed through a reduced bus divider (BDIV) which allows a lower final output clock frequency to be derived.

11.1.1 External Oscillator

The external oscillator module (XOSCVLP) provides the external clock options to the ICS module. The output of this submodule (OSCOUT) can be used as the TOD module (TOD) clock source.

11.1.2 Stop2 Mode Considerations

If you are using a low range oscillator during stop2, reconfigure the ICSC2 register (the oscillator control bits) before PPDACK is written. The low range (RANGE=0) oscillator can operate in stop2 to be the clock source for the TOD module. If the low range oscillator is active when entering stop2, it remains active in stop2 regardless of the value of EREFSTEN. To disable the oscillator in stop2, switch the ICS into FBI or FEI mode before executing the STOP instruction.

Figure 11-1 shows the MC9S08LH64 Series block diagram with the ICS highlighted.

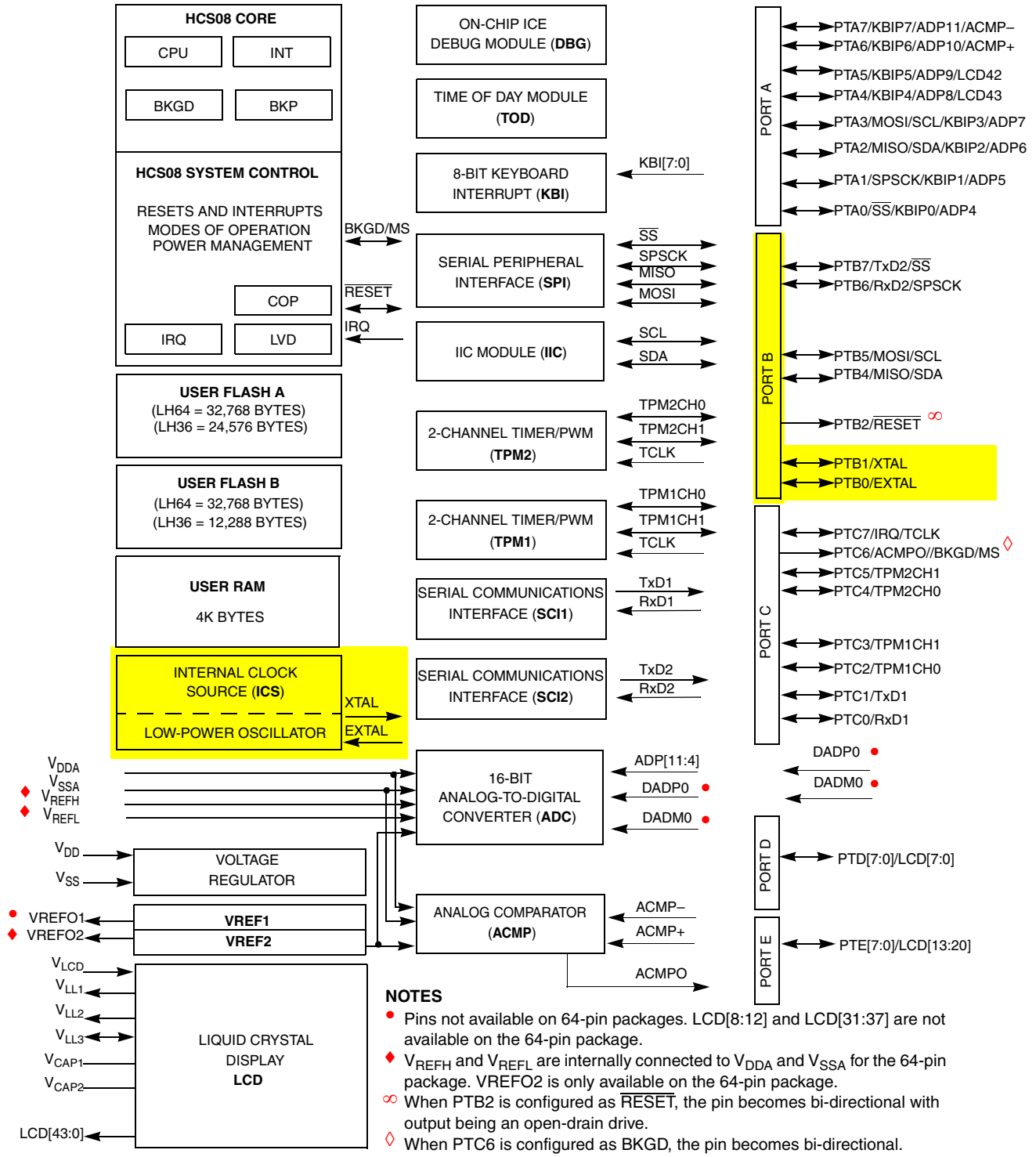


Figure 11-1. MC9S08LH64 Series Block Diagram Highlighting ICS Block and Pins

11.1.3 Features

Key features of the ICS module are:

- Frequency-locked loop (FLL) is trimmable for accuracy
- Internal or external reference clocks can be used to control the FLL
- Reference divider is provided for external clock
- Internal reference clock has 9 trim bits available
- Internal or external reference clocks can be selected as the clock source for the MCU
- Whichever clock is selected as the source can be divided down
  - 2-bit select for clock divider is provided
    - Allowable dividers are: 1, 2, 4, 8
- Control signals for a low power oscillator clock generator (OSCOUT) as the ICS external reference clock are provided
  - HGO, RANGE, EREFS, ERCLKEN, EREFSTEN
- FLL Engaged Internal mode is automatically selected out of reset
- BDC clock is provided as a constant divide by 2 of the low range DCO output
- Three selectable digitally-controlled oscillators (DCO) optimized for different frequency ranges.
- Option to maximize output frequency for a 32768 Hz external reference clock source.

11.1.4 Block Diagram

Figure 11-2 is the ICS block diagram.

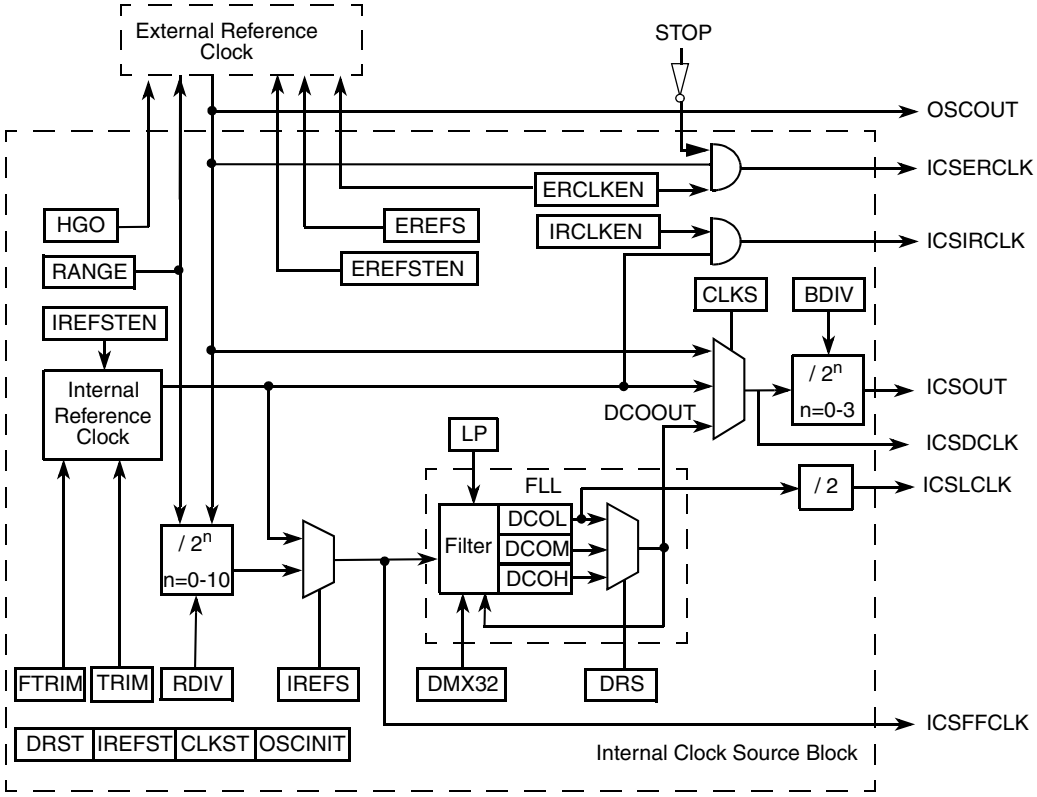


Figure 11-2. Internal Clock Source (ICS) Block Diagram

11.1.5 Modes of Operation

There are seven modes of operation for the ICS: FEI, FEE, FBI, FBILP, FBE, FBELP, and stop.

11.1.5.1 FLL Engaged Internal (FEI)

In FLL engaged internal mode, which is the default mode, the ICS supplies a clock derived from the FLL which is controlled by the internal reference clock. The BDC clock is supplied from the FLL.

11.1.5.2 FLL Engaged External (FEE)

In FLL engaged external mode, the ICS supplies a clock derived from the FLL which is controlled by an external reference clock source. The BDC clock is supplied from the FLL.

11.1.5.3 FLL Bypassed Internal (FBI)

In FLL bypassed internal mode, the FLL is enabled and controlled by the internal reference clock, but is bypassed. The ICS supplies a clock derived from the internal reference clock. The BDC clock is supplied from the FLL.

11.1.5.4 FLL Bypassed Internal Low Power (FBILP)

In FLL bypassed internal low power mode, the FLL is disabled and bypassed, and the ICS supplies a clock derived from the internal reference clock. The BDC clock is not available.

11.1.5.5 FLL Bypassed External (FBE)

In FLL bypassed external mode, the FLL is enabled and controlled by an external reference clock, but is bypassed. The ICS supplies a clock derived from the external reference clock source. The BDC clock is supplied from the FLL.

11.1.5.6 FLL Bypassed External Low Power (FBELP)

In FLL bypassed external low power mode, the FLL is disabled and bypassed, and the ICS supplies a clock derived from the external reference clock. The BDC clock is not available.

11.1.5.7 Stop (STOP)

In stop mode, the FLL is disabled and the internal or the ICS external reference clocks source (OSCOUT) can be selected to be enabled or disabled. The BDC clock is not available and the ICS does not provide an MCU clock source.

NOTE

The DCO frequency changes from the pre-stop value to its reset value and the FLL will need to re-acquire the lock before the frequency is stable. Timing sensitive operations should wait for the FLL acquistition time, tAquire, before executing.

11.2 External Signal Description

There are no ICS signals that connect off chip.

11.3 Register Definition

Figure 11-1 is a summary of ICS registers.

Table 11-1. ICS Register Summary

Name		7	6	5	4	3	2	1	0
ICSC1	R	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
	W								
ICSC2	R	BDIV		RANGE	HGO	LP	EREFS	ERCLKEN	EREFSTEN
	W								
ICSTRM	R	TRIM							
	W								

Table 11-1. ICS Register Summary (continued)

Name		7	6	5	4	3	2	1	0
ICSSC	R	DRST		DMX32	IREFST	CLKST		OSCINIT	FTRIM
	W	DRS							

11.3.1 ICS Control Register 1 (ICSC1)

	7	6	5	4	3	2	1	0
R W	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
Reset:	0	0	0	0	0	1	0	0

Figure 11-3. ICS Control Register 1 (ICSC1)

Table 11-2. ICS Control Register 1 Field Descriptions

Field	Description
7:6 CLKS	**Clock Source Select** — Selects the clock source that controls the bus frequency. The actual bus frequency depends on the value of the BDIV bits. 00 Output of FLL is selected. 01 Internal reference clock is selected. 10 External reference clock is selected. 11 Reserved, defaults to 00.
5:3 RDIV	**Reference Divider** — Selects the amount to divide down the external reference clock. Resulting frequency must be in the range 31.25 kHz to 39.0625 kHz. See Table 11-3 for the divide-by factors.
2 IREFS	**Internal Reference Select** — The IREFS bit selects the reference clock source for the FLL. 1 Internal reference clock selected. 0 External reference clock selected.
1 IRCLKEN	**Internal Reference Clock Enable** — The IRCLKEN bit enables the internal reference clock for use as ICSIRCLK. 1 ICSIRCLK active. 0 ICSIRCLK inactive.
0 IREFSTEN	**Internal Reference Stop Enable** — The IREFSTEN bit controls whether or not the internal reference clock remains enabled when the ICS enters stop mode. 1 Internal reference clock stays enabled in stop if IRCLKEN is set before entering stop. 0 Internal reference clock is disabled in stop.

Table 11-3. Reference Divide Factor

RDIV	RANGE=0	RANGE=1
0	1[1]	32
1	2	64
2	4	128
3	8	256

Table 11-3. Reference Divide Factor

RDIV	RANGE=0	RANGE=1
4	16	512
5	32	1024
6	64	Reserved
7	128	Reserved

[1] Reset default

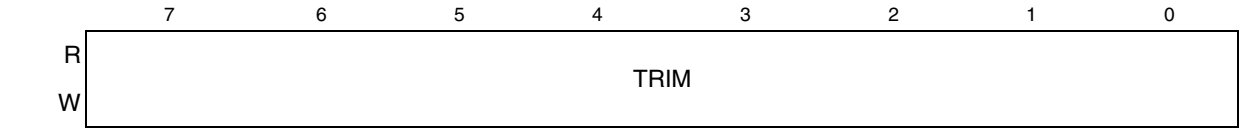
11.3.2 ICS Control Register 2 (ICSC2)

Figure 11-4. ICS Control Register 2 (ICSC2)

Table 11-4. ICS Control Register 2 Field Descriptions

Field	Description
7:6 BDIV	**Bus Frequency Divider** — Selects the amount to divide down the clock source selected by the CLKS bits. This controls the bus frequency. 00 Encoding 0 — Divides selected clock by 1. 01 Encoding 1 — Divides selected clock by 2 (reset default). 10 Encoding 2 — Divides selected clock by 4. 11 Encoding 3 — Divides selected clock by 8.
5 RANGE	**Frequency Range Select** — Selects the frequency range for the external oscillator. 1 High frequency range selected for the external oscillator. 0 Low frequency range selected for the external oscillator.
4 HGO	**High Gain Oscillator Select** — The HGO bit controls the external oscillator mode of operation. 1 Configure external oscillator for high gain operation. 0 Configure external oscillator for low power operation.
3 LP	**Low Power Select** — The LP bit controls whether the FLL is disabled in FLL bypassed modes. 1 FLL is disabled in bypass modes unless BDM is active. 0 FLL is not disabled in bypass mode.
2 EREFS	**External Reference Select** — The EREFS bit selects the source for the external reference clock. 1 Oscillator requested. 0 External Clock Source requested.
1 ERCLKEN	**External Reference Enable** — The ERCLKEN bit enables the external reference clock for use as ICSERCLK. 1 ICSERCLK active. 0 ICSERCLK inactive.
0 EREFSTEN	**External Reference Stop Enable** — The EREFSTEN bit controls whether or not the external reference clock source (OSCOUT) remains enabled when the ICS enters stop mode. 1 External reference clock source stays enabled in stop if ERCLKEN is set before entering stop. 0 External reference clock source is disabled in stop.

11.3.3 ICS Trim Register (ICSTRM)



Reset: Note: TRIM is loaded during reset from a factory programmed location when not in BDM mode. If in a BDM mode, a default value of 0x80 is loaded.

Figure 11-5. ICS Trim Register (ICSTRM)

Table 11-5. ICS Trim Register Field Descriptions

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

11.3.4 ICS Status and Control (ICSSC)

	7	6	5	4	3	2	1	0
R	DRST		DMX32	IREFST	CLKST		OSCINIT	FTRIM[1]
W	DRS							
Reset:	0	0	0	1	0	0	0	

Figure 11-6. ICS Status and Control Register (ICSSC)

1 FTRIM is loaded during reset from a factory programmed location when not in any BDM mode. If in a BDM mode, FTRIM gets loaded with a value of 1'b0.

Table 11-6. ICS Status and Control Register Field Descriptions

Field	Description
7-6 DRST DRS	**DCO Range Status** — The DRST read field indicates the current frequency range for the FLL output, DCOOUT. See Table 11-7. The DRST field does not update immediately after a write to the DRS field due to internal synchronization between clock domains. Writing the DRS bits to 2'b11 is ignored and the DRST bits remain with the current setting. **DCO Range Select** — The DRS field selects the frequency range for the FLL output, DCOOUT. Writes to the DRS field while the LP bit is set are ignored. 00 Low range. 01 Mid range. 10 High range. 11 Reserved.
5 DMX32	**DCO Maximum frequency with 32.768 kHz reference** — The DMX32 bit controls whether or not the DCO frequency range is narrowed to its maximum frequency with a 32.768 kHz reference. See Table 11-7. 0 DCO has default range of 25%. 1 DCO is fined tuned for maximum frequency with 32.768 kHz reference.
4 IREFST	**Internal Reference Status** — The IREFST bit indicates the current source for the reference clock. The IREFST bit does not update immediately after a write to the IREFS bit due to internal synchronization between clock domains. 0 Source of reference clock is external clock. 1 Source of reference clock is internal clock.

Table 11-6. ICS Status and Control Register Field Descriptions (continued)

Field	Description
3-2 CLKST	**Clock Mode Status** — The CLKST bits indicate the current clock mode. The CLKST bits don't update immediately after a write to the CLKS bits due to internal synchronization between clock domains. 00 Output of FLL is selected. 01 FLL Bypassed, Internal reference clock is selected. 10 FLL Bypassed, External reference clock is selected. 11 Reserved.
1 OSCINIT	**OSC Initialization** — If the external reference clock is selected by ERCLKEN or by the ICS being in FEE, FBE, or FBELP mode, and if EREFS is set, then this bit is set after the initialization cycles of the external oscillator clock have completed. This bit is only cleared when either ERCLKEN or EREFS are cleared.
0 FTRIM	**ICS Fine Trim** — The FTRIM bit controls the smallest adjustment of the internal reference clock frequency. Setting FTRIM will increase the period and clearing FTRIM will decrease the period by the smallest amount possible.

Table 11-7. DCO frequency range[1]

DRS	DMX32	Reference range	FLL factor	DCO range
00	**0**	31.25 - 39.0625 kHz	512	16 - 20 MHz
	1	32.768 kHz	608	19.92 MHz
01	**0**	31.25 - 39.0625 kHz	1024	32 - 40 MHz
	1	32.768 kHz	1216	39.85 MHz
10	**0**	31.25 - 39.0625 kHz	1536	48 - 60 MHz
	1	32.768 kHz	1824	59.77 MHz
11	Reserved			

[1] The resulting bus clock frequency should not exceed the maximum specified bus clock frequency of the device.

11.4 Functional Description

11.4.1 Operational Modes

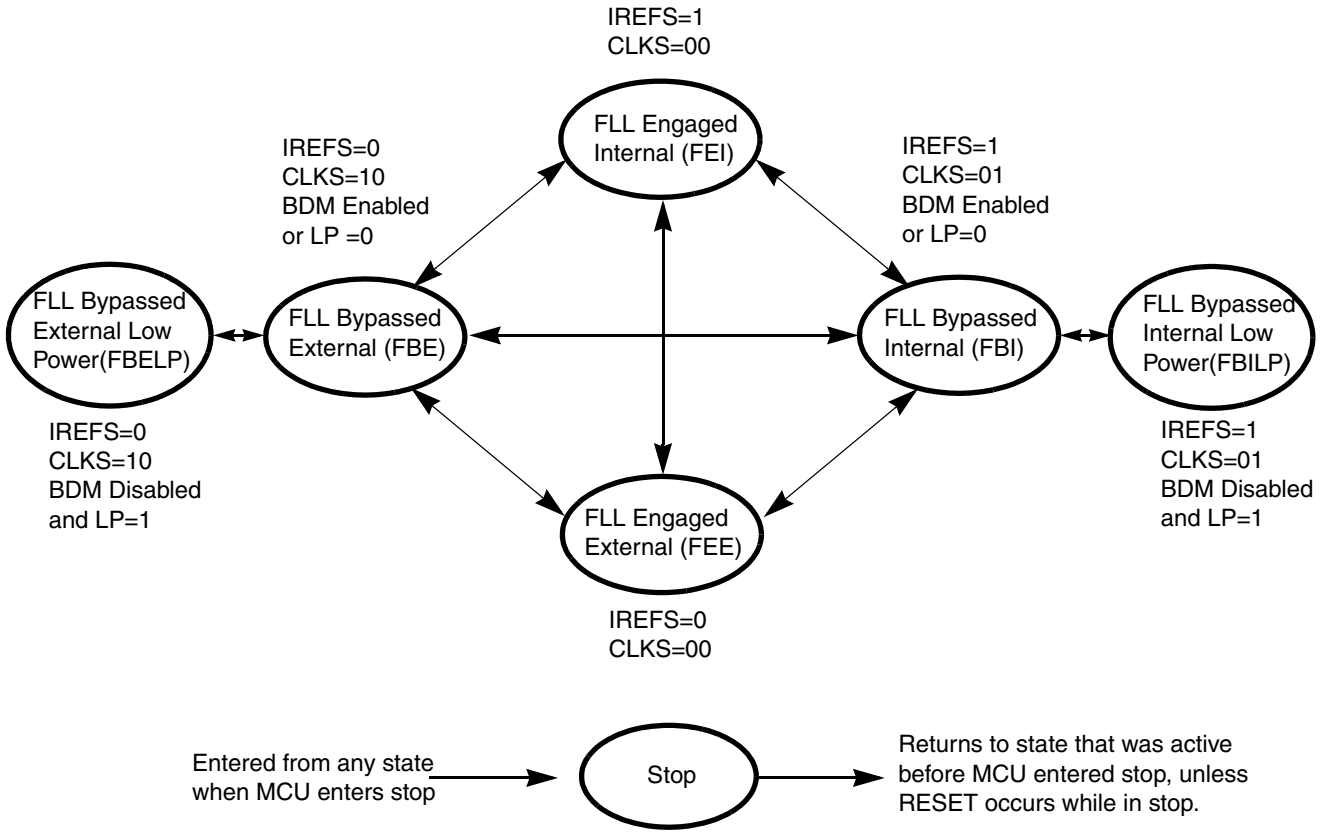


Figure 11-7. Clock Switching Modes

The seven states of the ICS are shown as a state diagram and are described below. The arrows indicate the allowed movements between the states.

11.4.1.1 FLL Engaged Internal (FEI)

FLL engaged internal (FEI) is the default mode of operation and is entered when all the following conditions occur:

- CLKS bits are written to 00.
- IREFS bit is written to 1.

In FLL engaged internal mode, the ICSOUT clock is derived from the FLL clock, which is controlled by the internal reference clock. The FLL loop locks the frequency to the FLL factor times the internal reference frequency. The ICSLCLK is available for BDC communications, and the internal reference clock is enabled.

11.4.1.2 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 external 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 source.The FLL loop locks the frequency to the FLL factor times the external reference frequency, as selected by the RDIV bits. The ICSLCLK is available for BDC communications, and the external reference clock is enabled.

11.4.1.3 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 locks the FLL frequency to the FLL factor times the internal reference frequency. The ICSLCLK will be available for BDC communications, and the internal reference clock is enabled.

11.4.1.4 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.

11.4.1.5 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.
- RDIV bits are written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz.
- 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 source. The FLL clock is controlled by the external reference clock, and the FLL loop locks the FLL frequency to the FLL factor times the external reference frequency, as selected by the RDIV bits, so that the ICSLCLK will be available for BDC communications, and the external reference clock is enabled.

11.4.1.6 FLL Bypassed External Low Power (FBELP)

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

- CLKS bits are written to 10.
- IREFS bit is written to 0.
- BDM mode is not active and LP bit is written to 1.

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

11.4.1.7 Stop

Stop mode is entered whenever the MCU enters a STOP state. In this mode, all ICS clock signals are static except in the following cases:

ICSIRCLK will be active in stop mode when all the following conditions occur:

- IRCLKEN bit is written to 1.
- IREFSTEN bit is written to 1.

OSCOUT will be active in stop mode when all the following conditions occur:

- ERCLKEN bit is written to 1.
- EREFSTEN bit is written to 1.

11.4.2 Mode Switching

The IREF bit can be changed at anytime, but the actual switch to the newly selected clock is shown by the IREFST bit. When switching between FLL engaged internal (FEI) and FLL engaged external (FEE) modes, the FLL begins locking again after the switch is completed.

The CLKS bits can also be changed at anytime, but the actual switch to the newly selected clock is shown by the CLKST bits. If the newly selected clock is not available, the previous clock remains selected.

The DRS bits can be changed at anytime except when LP bit is 1. If the DRS bits are changed while in FLL engaged internal (FEI) or FLL engaged external (FEE), the bus clock remains at the previous DCO range until the new DCO starts. When the new DCO starts the bus clock switches to it. After switching to the new DCO the FLL remains unlocked for several reference cycles. Once the selected DCO startup time is over, the FLL is locked. The completion of the switch is shown by the DRST bits.

11.4.3 Bus Frequency Divider

The BDIV bits can be changed at anytime and the actual switch to the new frequency occurs immediately.

11.4.4 Low Power Bit Usage

The low power bit (LP) is provided to allow the FLL to be disabled and thus conserve power when it is not being used. The DRS bits can not be written while LP bit is 1.

However, in some applications it may be desirable to allow the FLL to be enabled and to lock for maximum accuracy before switching to an FLL engaged mode. To do this, write the LP bit to 0.

11.4.5 DCO Maximum Frequency with 32.768 kHz Oscillator

The FLL has an option to change the clock multiplier for the selected DCO range such that it results in the maximum bus frequency with a common 32.768 kHz crystal reference clock.

11.4.6 Internal Reference Clock

When IRCLKEN is set the internal reference clock signal is presented as ICSIRCLK, which can be used as an additional clock source. To re-target the ICSIRCLK frequency, write a new value to the TRIM bits in the ICSTRM register to trim the period of the internal reference clock:

- Writing a larger value slows down the ICSIRCLK frequency.
- Writing a smaller value to the ICSTRM register speeds up the ICSIRCLK frequency.

The TRIM bits effect the ICSOUT frequency if the ICS is in FLL engaged internal (FEI), FLL bypassed internal (FBI), or FLL bypassed internal low power (FBILP) mode.

Until ICSIRCLK is trimmed, programming low reference divider (RDIV) factors may result in ICSOUT frequencies that exceed the maximum chip-level frequency and violate the chip-level clock timing specifications (see the Device Overview chapter).

If IREFSTEN is set and the IRCLKEN bit is written to 1, the internal reference clock keeps running during stop mode in order to provide a fast recovery upon exiting stop.

All MCU devices are factory programmed with a trim value in a reserved memory location. This value is uploaded to the ICSTRM register and ICS FTRIM register during any reset initialization. For finer precision, trim the internal oscillator in the application and set the FTRIM bit accordingly.

11.4.7 External Reference Clock

The ICS module supports an external reference clock with frequencies between 31.25 kHz to 40 MHz in all modes. When the ERCLKEN is set, the external reference clock signal is presented as ICSERCLK, which can be used as an additional clock source in run mode. When IREFS = 1, the external reference clock is not used by the FLL and will only be used as ICSERCLK. In these modes, the frequency can be equal to the maximum frequency the chip-level timing specifications support (see the Device Overview chapter).

If EREFSTEN is set and the ERCLKEN bit is written to 1, the external reference clock source (OSCOUT) keeps running during stop mode in order to provide a fast recovery upon exiting stop.

11.4.8 Fixed Frequency Clock

The ICS presents the divided FLL reference clock as ICSFFCLK for use as an additional clock source. ICSFFCLK frequency must be no more than 1/4 of the ICSOUT frequency to be valid. Because of this requirement, in bypass modes the ICSFFCLK is valid only in bypass external modes (FBE and FBELP) for the following combinations of BDIV, RDIV and RANGE values:

- RANGE=1
- BDIV=00 (divide by 1), RDIV ≥ 010
- BDIV=01 (divide by 2), RDIV ≥ 011
- BDIV=10 (divide by 4), RDIV ≥ 100
- BDIV=11 (divide by 8), RDIV ≥ 101

11.4.9 Local Clock

The ICS presents the low range DCO output clock divided by two as ICSLCLK for use as a clock source for BDC communications. ICSLCLK is not available in FLL bypassed internal low power (FBILP) and FLL bypassed external low power (FBELP) modes.

Chapter 12
Inter-Integrated Circuit (S08IICV2)

12.1 Introduction

The inter-integrated circuit (IIC) provides a method of communication between a number of devices. The interface is designed to operate up to 100 kbps with maximum bus loading and timing. The device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF.

NOTE

MC9S08LH64 Series devices do not include stop1 mode. Please ignore references to stop1.

NOTE

The SDA and SCL should not be driven above V_{DD}. These pins are pseudo open-drain and contain a protection diode to V_{DD}.

12.1.1 Module Configuration

The IIC module pins, SDA and SCL can be repositioned under software control using IICPS in SOPT2 as as shown in Table 12-1. IICPS in SOPT2 selects which general-purpose I/O ports are associated with IIC operation.

Table 12-1. IIC Position Options

IICPS in SOPT2	Port Pin for SDA	Port Pin for SCL
0 (default)	PTB4	PTB5
1	PTA2	PTA3

12.1.2 IIC Clock Gating

The bus clock to the IIC can be gated on and off using the IIC bit in SCGC1. This bit is set after any reset, which enables the bus clock to this module. To conserve power, the IIC bit can be cleared to disable the clock to this module when not in use. See Section 5.7, “Peripheral Clock Gating,” for details.

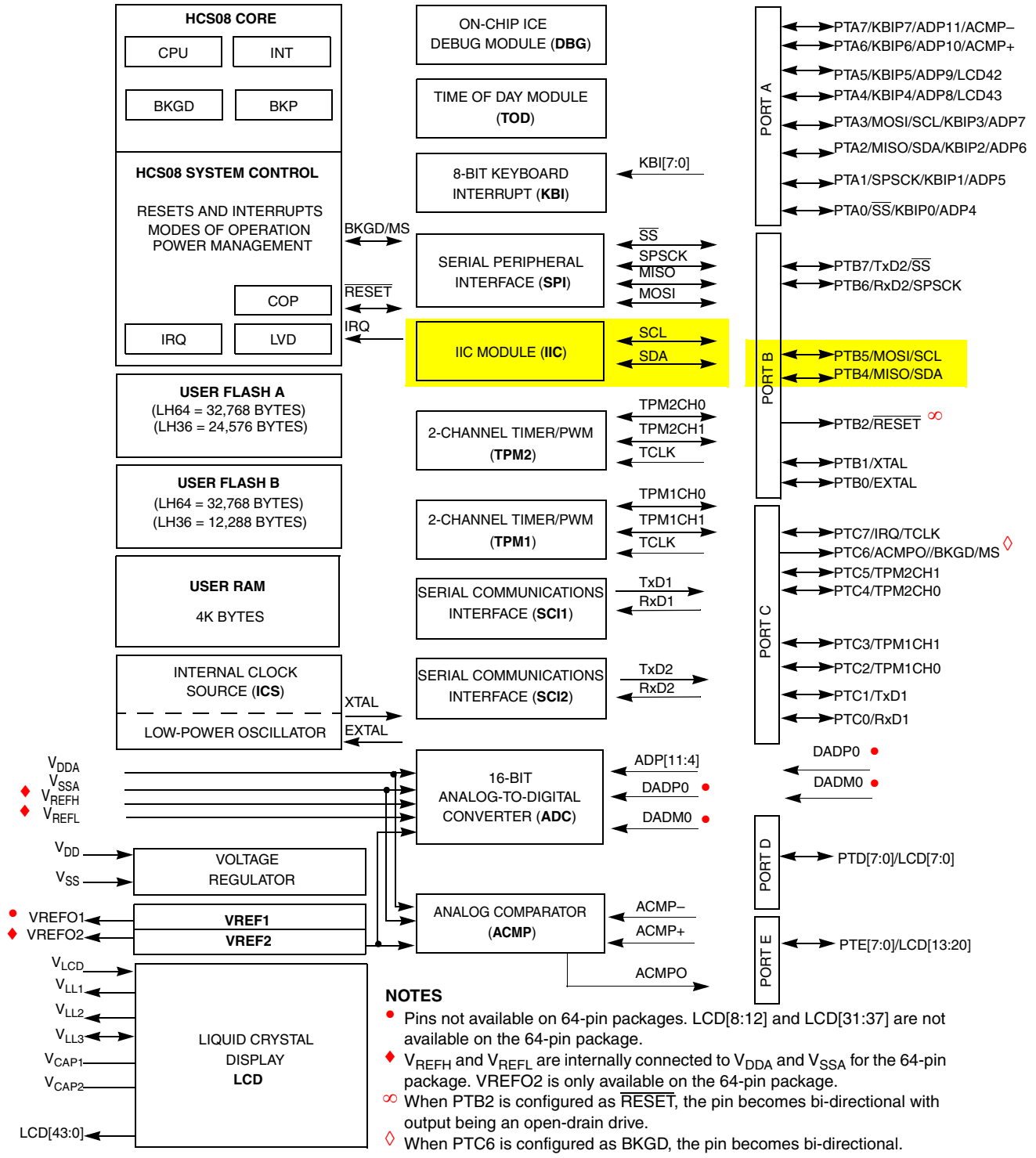


Figure 12-1. MC9S08LH64 Block Diagram Highlighting the IIC Block and Pins

12.1.3 Features

The IIC includes these distinctive features:

- Compatible with IIC bus standard
- Multi-master operation
- Software programmable for one of 64 different serial clock frequencies
- Software selectable acknowledge bit
- Interrupt driven byte-by-byte data transfer
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation
- Acknowledge bit generation/detection
- Bus busy detection
- General call recognition
- 10-bit address extension

12.1.4 Modes of Operation

A brief description of the IIC in the various MCU modes is given here.

- **Run mode** — This is the basic mode of operation. To conserve power in this mode, disable the module.
- **Wait mode** — The module continues to operate while the MCU is in wait mode and can provide a wake-up interrupt.
- **Stop mode** — The IIC is inactive in stop3 mode for reduced power consumption. The stop instruction does not affect IIC register states. Stop2 resets the register contents.

12.1.5 Block Diagram

Figure 12-2 is a block diagram of the IIC.

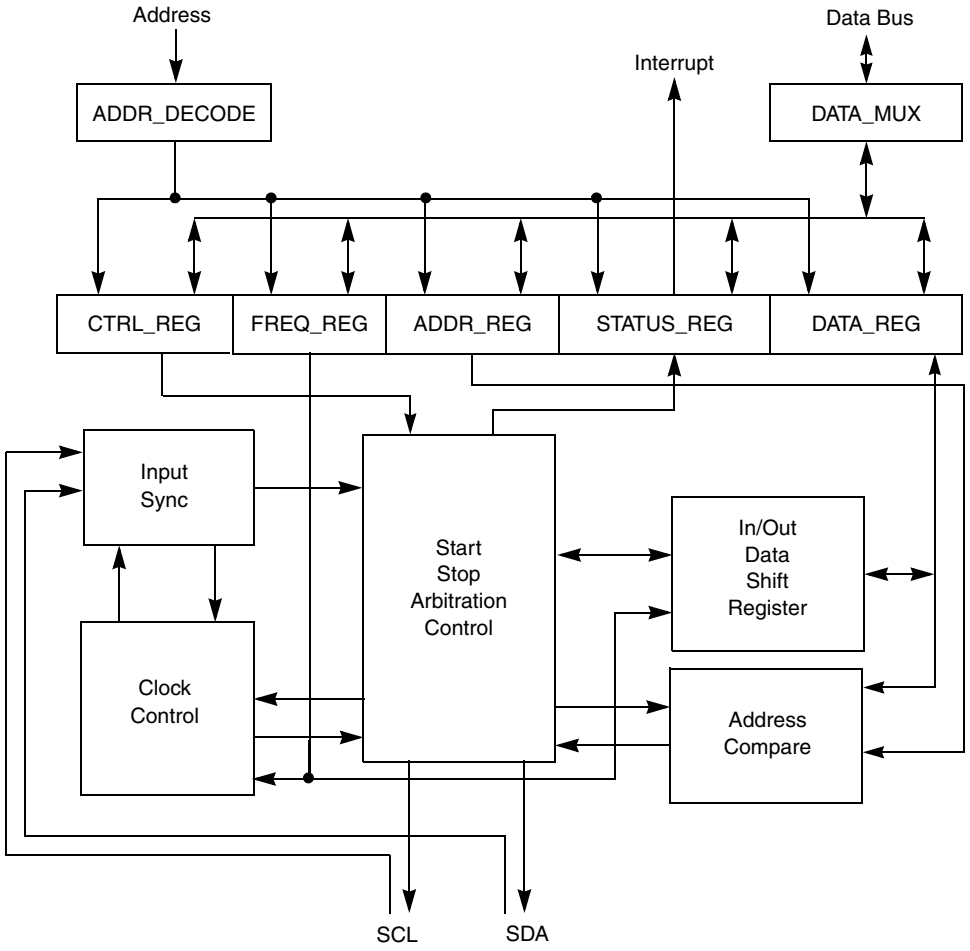


Figure 12-2. IIC Functional Block Diagram

12.2 External Signal Description

This section describes each user-accessible pin signal.

12.2.1 SCL — Serial Clock Line

The bidirectional SCL is the serial clock line of the IIC system.

12.2.2 SDA — Serial Data Line

The bidirectional SDA is the serial data line of the IIC system.

12.3 Register Definition

This section consists of the IIC register descriptions in address order.

Refer to the direct-page register summary in the memory chapter of this document for the absolute address assignments for all IIC 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.

12.3.1 IIC Address Register (IICA)

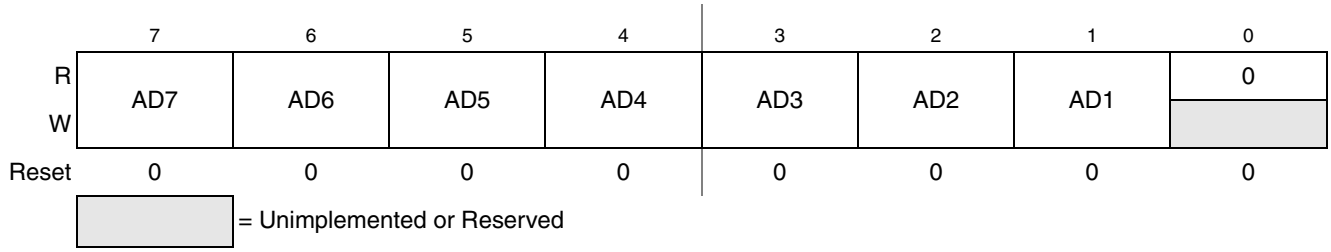


Figure 12-3. IIC Address Register (IICA)

Table 12-2. IICA Field Descriptions

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

12.3.2 IIC Frequency Divider Register (IICF)

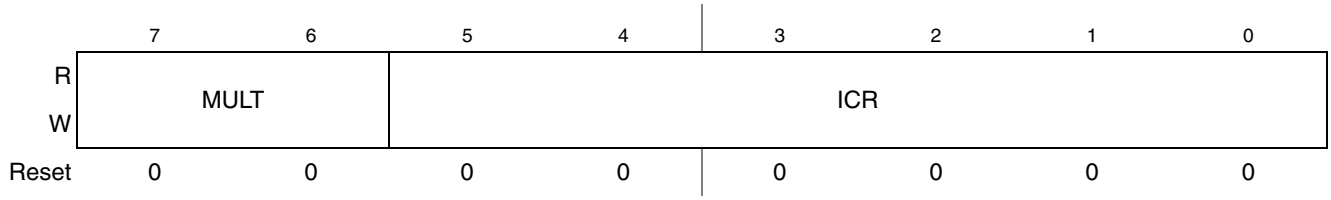


Figure 12-4. IIC Frequency Divider Register (IICF)

Table 12-3. IICF Field Descriptions

Field	Description
7–6 MULT	**IIC Multiplier Factor**. The MULT bits define the multiplier factor, mul. This factor, along with the SCL divider, generates the IIC baud rate. The multiplier factor mul as defined by the MULT bits is provided below. 00 mul = 01 01 mul = 02 10 mul = 04 11 Reserved
5–0 ICR	**IIC Clock Rate**. The ICR bits are used to prescale the bus clock for bit rate selection. These bits and the MULT bits determine the IIC baud rate, the SDA hold time, the SCL Start hold time, and the SCL Stop hold time. Table 12-5 provides the SCL divider and hold values for corresponding values of the ICR. The SCL divider multiplied by multiplier factor mul generates IIC baud rate. $\text{IIC baud rate} = \frac{\text{bus speed (Hz)}}{\text{mul} \times \text{SCLdivider}}$ ***Eqn. 12-1*** SDA hold time is the delay from the falling edge of SCL (IIC clock) to the changing of SDA (IIC data). SDA hold time = bus period (s) × mul × SDA hold value ***Eqn. 12-2*** SCL start hold time is the delay from the falling edge of SDA (IIC data) while SCL is high (Start condition) to the falling edge of SCL (IIC clock). SCL Start hold time = bus period (s) × mul × SCL Start hold value ***Eqn. 12-3*** SCL stop hold time is the delay from the rising edge of SCL (IIC clock) to the rising edge of SDA SDA (IIC data) while SCL is high (Stop condition). SCL Stop hold time = bus period (s) × mul × SCL Stop hold value ***Eqn. 12-4***

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

Table 12-4. Hold Time Values for 8 MHz Bus Speed

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

Table 12-5. IIC Divider and Hold Values

ICR (hex)	SCL Divider	SDA Hold Value	SCL Hold (Start) Value	SCL Hold (Stop) Value
00	20	7	6	11
01	22	7	7	12
02	24	8	8	13
03	26	8	9	14
04	28	9	10	15
05	30	9	11	16
06	34	10	13	18
07	40	10	16	21
08	28	7	10	15
09	32	7	12	17
0A	36	9	14	19
0B	40	9	16	21
0C	44	11	18	23
0D	48	11	20	25
0E	56	13	24	29
0F	68	13	30	35
10	48	9	18	25
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97
1F	240	33	118	121

ICR (hex)	SCL Divider	SDA Hold Value	SCL Hold (Start) Value	SCL Hold (Stop) Value
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921

12.3.3 IIC Control Register (IICC1)

	7	6	5	4	3	2	1	0
R	IICEN	IICIE	MST	TX	TXAK	0	0	0
W						RSTA		
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-5. IIC Control Register (IICC1)

Table 12-6. IICC1 Field Descriptions

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

12.3.4 IIC Status Register (IICS)

	7	6	5	4	3	2	1	0
R	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
W								
Reset	1	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-6. IIC Status Register (IICS)

Table 12-7. IICS Field Descriptions

Field	Description
7 TCF	**Transfer Complete Flag.** This bit is set on the completion of a byte transfer. This bit is only valid during or immediately following a transfer to the IIC module or from the IIC module.The TCF bit is cleared by reading the IICD register in receive mode or writing to the IICD in transmit mode. 0 Transfer in progress 1 Transfer complete
6 IAAS	**Addressed as a Slave.** The IAAS bit is set when the calling address matches the programmed slave address or when the GCAEN bit is set and a general call is received. Writing the IICC register clears this bit. 0 Not addressed 1 Addressed as a slave
5 BUSY	**Bus Busy.** The BUSY bit indicates the status of the bus regardless of slave or master mode. The BUSY bit is set when a start signal is detected and cleared when a stop signal is detected. 0 Bus is idle 1 Bus is busy
4 ARBL	**Arbitration Lost.** This bit is set by hardware when the arbitration procedure is lost. The ARBL bit must be cleared by software by writing a 1 to it. 0 Standard bus operation 1 Loss of arbitration
2 SRW	**Slave Read/Write.** When addressed as a slave, the SRW bit indicates the value of the R/W command bit of the calling address sent to the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IICIF	**IIC Interrupt Flag.** The IICIF bit is set when an interrupt is pending. This bit must be cleared by software, by writing a 1 to it in the interrupt routine. One of the following events can set the IICIF bit: • One byte transfer completes • Match of slave address to calling address • Arbitration lost 0 No interrupt pending 1 Interrupt pending
0 RXAK	**Receive Acknowledge**. When the RXAK bit is low, it indicates an acknowledge signal has been received after the completion of one byte of data transmission on the bus. If the RXAK bit is high it means that no acknowledge signal is detected. 0 Acknowledge received 1 No acknowledge received

12.3.5 IIC Data I/O Register (IICD)

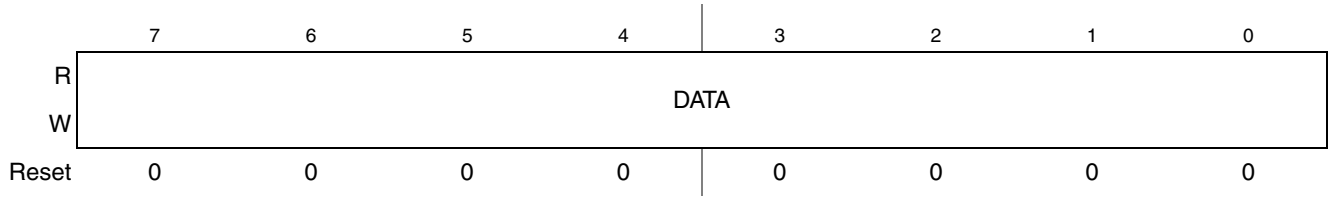


Figure 12-7. IIC Data I/O Register (IICD)

Table 12-8. IICD Field Descriptions

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

NOTE

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

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

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

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

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

12.3.6 IIC Control Register 2 (IICC2)

	7	6	5	4	3	2	1	0
R	GCAEN	ADEXT	0	0	0	AD10	AD9	AD8
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-8. IIC Control Register (IICC2)

Table 12-9. IICC2 Field Descriptions

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

12.4 Functional Description

This section provides a complete functional description of the IIC module.

12.4.1 IIC Protocol

The IIC bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfer. All devices connected to it must have open drain or open collector outputs. A logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors is system dependent.

Normally, a standard communication is composed of four parts:

- Start signal
- Slave address transmission
- Data transfer
- Stop signal

The stop signal should not be confused with the CPU stop instruction. The IIC bus system communication is described briefly in the following sections and illustrated in Figure 12-9.

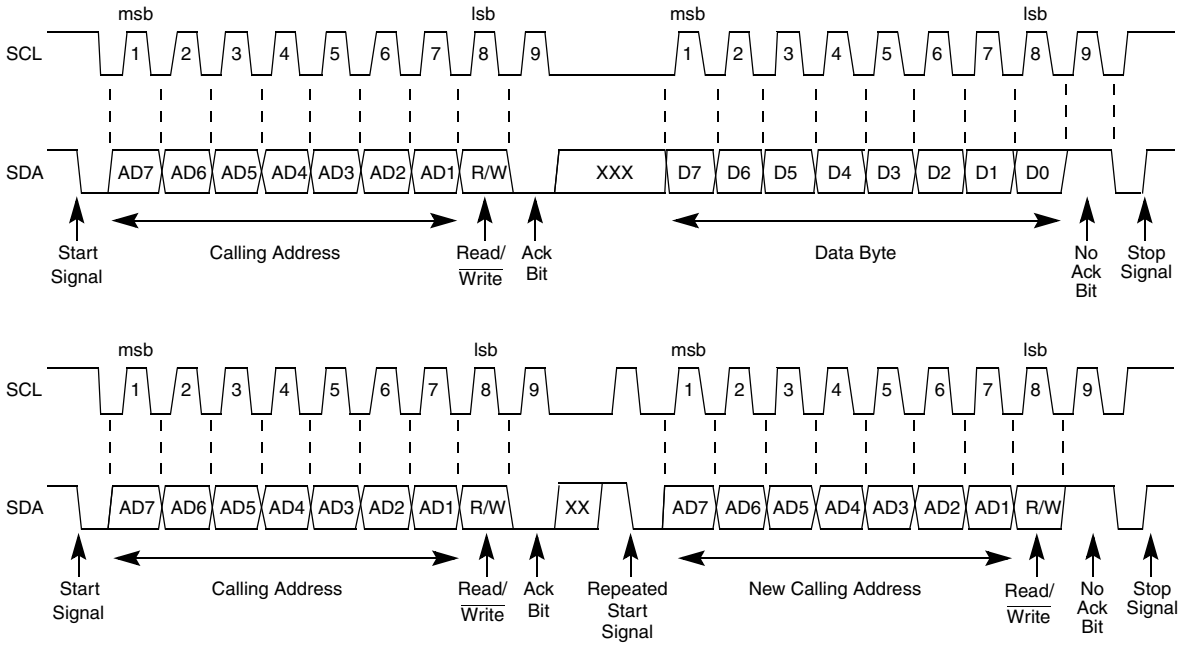


Figure 12-9. IIC Bus Transmission Signals

12.4.1.1 Start Signal

When the bus is free, no master device is engaging the bus (SCL and SDA lines are at logical high), a master may initiate communication by sending a start signal. As shown in Figure 12-9, a start signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer may contain several bytes of data) and brings all slaves out of their idle states.

12.4.1.2 Slave Address Transmission

The first byte of data transferred immediately after the start signal is the slave address transmitted by the master. This is a seven-bit calling address followed by a R/$\overline{W}$ bit. The R/$\overline{W}$ bit tells the slave the desired direction of data transfer.

1 = Read transfer, the slave transmits data to the master.
0 = Write transfer, the master transmits data to the slave.

Only the slave with a calling address that matches the one transmitted by the master responds by sending back an acknowledge bit. This is done by pulling the SDA low at the ninth clock (see Figure 12-9).

No two slaves in the system may have the same address. If the IIC module is the master, it must not transmit an address equal to its own slave address. The IIC cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the IIC reverts to slave mode and operates correctly even if it is being addressed by another master.

12.4.1.3 Data Transfer

Before successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the R/$\overline{W}$ bit sent by the calling master.

All transfers that come after an address cycle are referred to as data transfers, even if they carry sub-address information for the slave device

Each data byte is 8 bits long. Data may be changed only while SCL is low and must be held stable while SCL is high as shown in Figure 12-9. There is one clock pulse on SCL for each data bit, the msb being transferred first. Each data byte is followed by a 9th (acknowledge) bit, which is signalled from the receiving device. An acknowledge is signalled by pulling the SDA low at the ninth clock. In summary, one complete data transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master in the ninth bit time, the SDA line must be left high by the slave. The master interprets the failed acknowledge as an unsuccessful data transfer.

If the master receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets this as an end of data transfer and releases the SDA line.

In either case, the data transfer is aborted and the master does one of two things:

- Relinquishes the bus by generating a stop signal.
- Commences a new calling by generating a repeated start signal.

12.4.1.4 Stop Signal

The master can terminate the communication by generating a stop signal to free the bus. However, the master may generate a start signal followed by a calling command without generating a stop signal first. This is called repeated start. A stop signal is defined as a low-to-high transition of SDA while SCL at logical 1 (see Figure 12-9).

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

12.4.1.5 Repeated Start Signal

As shown in Figure 12-9, a repeated start signal is a start signal generated without first generating a stop signal to terminate the communication. This is used by the master to communicate with another slave or with the same slave in different mode (transmit/receive mode) without releasing the bus.

12.4.1.6 Arbitration Procedure

The IIC bus is a true multi-master bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock, for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. The relative priority of the contending masters is determined by a data arbitration procedure, a bus master loses arbitration if it transmits logic 1 while another master transmits logic 0. The losing masters immediately switch over to slave receive mode and stop driving SDA output. In this case, the transition from master to slave mode does not generate a stop condition. Meanwhile, a status bit is set by hardware to indicate loss of arbitration.

12.4.1.7 Clock Synchronization

Because wire-AND logic is performed on the SCL line, a high-to-low transition on the SCL line affects all the devices connected on the bus. The devices start counting their low period and after a device's clock has gone low, it holds the SCL line low until the clock high state is reached. However, the change of low to high in this device clock may not change the state of the SCL line if another device clock is still within its low period. Therefore, synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see Figure 12-10). When all devices concerned have counted off their low period, the synchronized clock SCL line is released and pulled high. There is then no difference between the device clocks and the state of the SCL line and all the devices start counting their high periods. The first device to complete its high period pulls the SCL line low again.

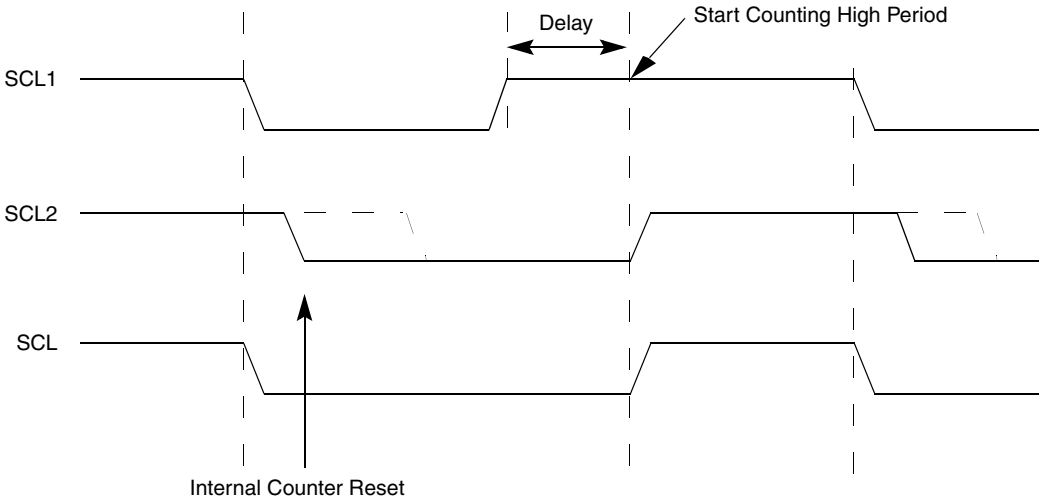


Figure 12-10. IIC Clock Synchronization

12.4.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte transfer (9 bits). In such a case, it halts the bus clock and forces the master clock into wait states until the slave releases the SCL line.

12.4.1.9 Clock Stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master has driven SCL low the slave can drive SCL low for the required period and then release it. If the slave SCL low period is greater than the master SCL low period then the resulting SCL bus signal low period is stretched.

12.4.2 10-bit Address

For 10-bit addressing, 0x11110 is used for the first 5 bits of the first address byte. Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing.

12.4.2.1 Master-Transmitter Addresses a Slave-Receiver

The transfer direction is not changed (see Table 12-10). When a 10-bit address follows a start condition, each slave compares the first seven bits of the first byte of the slave address (11110XX) with its own address and tests whether the eighth bit (R/$\overline{W}$ direction bit) is 0. More than one device can find a match and generate an acknowledge (A1). Then, each slave that finds a match compares the eight bits of the second byte of the slave address with its own address. Only one slave finds a match and generates an acknowledge (A2). The matching slave remains addressed by the master until it receives a stop condition (P) or a repeated start condition (Sr) followed by a different slave address.

S	Slave Address 1st 7 bits 11110 + AD10 + AD9	R/W 0	A1	Slave Address 2nd byte AD[8:1]	A2	Data	A	...	Data	A/A	P

Table 12-10. Master-Transmitter Addresses Slave-Receiver with a 10-bit Address

After the master-transmitter has sent the first byte of the 10-bit address, the slave-receiver sees an IIC interrupt. Software must ensure the contents of IICD are ignored and not treated as valid data for this interrupt.

12.4.2.2 Master-Receiver Addresses a Slave-Transmitter

The transfer direction is changed after the second R/$\overline{W}$ bit (see Table 12-11). Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated start condition (Sr), a matching slave remembers that it was addressed before. This slave then checks whether the first seven bits of the first byte of the slave address following Sr are the same as they were after the start condition (S) and tests whether the eighth (R/$\overline{W}$) bit is 1. If there is a match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3. The slave-transmitter remains addressed until it receives a stop condition (P) or a repeated start condition (Sr) followed by a different slave address.

After a repeated start condition (Sr), all other slave devices also compare the first seven bits of the first byte of the slave address with their own addresses and test the eighth (R/$\overline{W}$) bit. However, none of them are addressed because R/$\overline{W}$ = 1 (for 10-bit devices) or the 11110XX slave address (for 7-bit devices) does not match.

S	Slave Address 1st 7 bits 11110 + AD10 + AD9	R/W 0	A1	Slave Address 2nd byte AD[8:1]	A2	Sr	Slave Address 1st 7 bits 11110 + AD10 + AD9	R/W 1	A3	Data	A	...	Data	A	P

Table 12-11. Master-Receiver Addresses a Slave-Transmitter with a 10-bit Address

After the master-receiver has sent the first byte of the 10-bit address, the slave-transmitter sees an IIC interrupt. Software must ensure the contents of IICD are ignored and not treated as valid data for this interrupt.

12.4.3 General Call Address

General calls can be requested in 7-bit address or 10-bit address. If the GCAEN bit is set, the IIC matches the general call address as well as its own slave address. When the IIC responds to a general call, it acts as a slave-receiver and the IAAS bit is set after the address cycle. Software must read the IICD register after the first byte transfer to determine whether the address matches is its own slave address or a general call. If the value is 00, the match is a general call. If the GCAEN bit is clear, the IIC ignores any data supplied from a general call address by not issuing an acknowledgement.

12.5 Resets

The IIC is disabled after reset. The IIC cannot cause an MCU reset.

12.6 Interrupts

The IIC generates a single interrupt.

An interrupt from the IIC is generated when any of the events in Table 12-12 occur, provided the IICIE bit is set. The interrupt is driven by bit IICIF (of the IIC status register) and masked with bit IICIE (of the IIC control register). The IICIF bit must be cleared by software by writing a 1 to it in the interrupt routine. You can determine the interrupt type by reading the status register.

Table 12-12. Interrupt Summary

Interrupt Source	Status	Flag	Local Enable
Complete 1-byte transfer	TCF	IICIF	IICIE
Match of received calling address	IAAS	IICIF	IICIE
Arbitration Lost	ARBL	IICIF	IICIE

12.6.1 Byte Transfer Interrupt

The TCF (transfer complete flag) bit is set at the falling edge of the ninth clock to indicate the completion of byte transfer.

12.6.2 Address Detect Interrupt

When the calling address matches the programmed slave address (IIC address register) or when the GCAEN bit is set and a general call is received, the IAAS bit in the status register is set. The CPU is interrupted, provided the IICIE is set. The CPU must check the SRW bit and set its Tx mode accordingly.

12.6.3 Arbitration Lost Interrupt

The IIC is a true multi-master bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, the relative priority of the contending masters is determined by a data arbitration procedure. The IIC module asserts this interrupt when it loses the data arbitration process and the ARBL bit in the status register is set.

Arbitration is lost in the following circumstances:

- SDA sampled as a low when the master drives a high during an address or data transmit cycle.
- SDA sampled as a low when the master drives a high during the acknowledge bit of a data receive cycle.
- A start cycle is attempted when the bus is busy.
- A repeated start cycle is requested in slave mode.
- A stop condition is detected when the master did not request it.

This bit must be cleared by software writing a 1 to it.

Subject to Change — Non-Disclosure Agreement Required

12.7 Initialization/Application Information

Module Initialization (Slave)

1. Write: IICC2
   — to enable or disable general call
   — to select 10-bit or 7-bit addressing mode
2. Write: IICA
   — to set the slave address
3. Write: IICC1
   — to enable IIC and interrupts
4. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
5. Initialize RAM variables used to achieve the routine shown in Figure 12-12

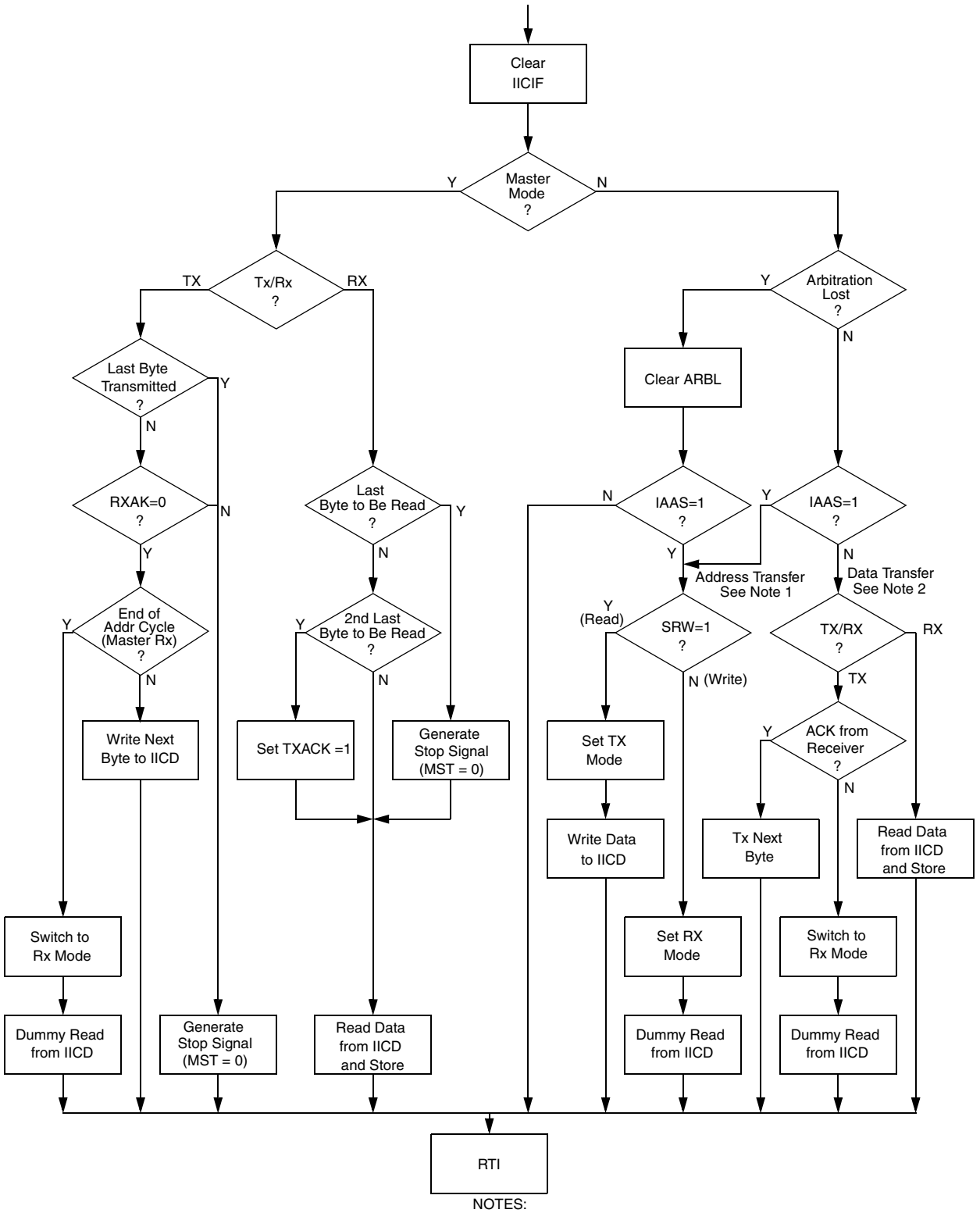
Module Initialization (Master)

1. Write: IICF
   — to set the IIC baud rate (example provided in this chapter)
2. Write: IICC1
   — to enable IIC and interrupts
3. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
4. Initialize RAM variables used to achieve the routine shown in Figure 12-12
5. Write: IICC1
   — to enable TX

Register Model

IICA	AD[7:1]							0
	When addressed as a slave (in slave mode), the module responds to this address							
IICF	MULT		ICR					
	Baud rate = BUSCLK / (2 x MULT x (SCL DIVIDER))							
IICC1	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
	Module configuration							
IICS	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
	Module status flags							
IICD	DATA							
	Data register; Write to transmit IIC data read to read IIC data							
IICC2	GCAEN	ADEXT	0	0	0	AD10	AD9	AD8
	Address configuration							

Figure 12-11. IIC Module Quick Start



NOTES:

1. If general call is enabled, a check must be done to determine whether the received address was a general call address (0x00). If the received address was a general call address, then the general call must be handled by user software.
2. When 10-bit addressing is used to address a slave, the slave sees an interrupt following the first byte of the extended address. User software must ensure that for this interrupt, the contents of IICD are ignored and not treated as a valid data transfer.

Figure 12-12. Typical IIC Interrupt Routine

Chapter 13
Liquid Crystal Display Driver (S08LCDV1)

13.1 Introduction

On the MC9S08LH64 series, the LCD module (LCD) controls the 44 LCD pins to generate the waveforms necessary to drive a liquid crystal display. On the 80 pin package the 44 LCD pins can be used to generate 4×40 or 8×36 configuration. On the 64-pin package there are 32 LCD pins available for driving 4×28 or 8×24 configurations.

The maximum power supply voltage (V_{DD}) for the MC9S08LH64 is specified in the Data Sheet. The voltage on the VLL3 pin must not exceed V_{DD}.

13.1.1 LCD Clock Sources

The LCD module on MC9S08LH64 series can be clocked from the OSCOUT or the ALTCLK. ALTCLK is connected to TODCLK a signal provided by the TOD module. The TOD must be configured to provide the TODCLK if this clock source is to be used.

13.1.2 LCD Modes of Operation

The LCD module can be configured to operate in stop modes by clearing the LCDSTP bit. All clock sources are available in all modes except stop2. In stop2 mode, only OSCOUT is available.

13.1.3 LCD Status after Stop2 Wakeup

The LCD control registers will be set to their default state. These registers should be re-initialized before setting the PPDACK. The contents of the LCD data registers (LCD waveform, LCD pin enable, and LCD backplane enable) will be retained.

13.1.4 LCD Clock Gating

The bus clock to the LCD can be gated on and off using the LCD bit in SCGC2. This bit is set after any reset, which enables the bus clock to this module. To conserve power, the LCD bit can be cleared to disable the clock to this module when not in use. See Section 5.7, “Peripheral Clock Gating,” for details.

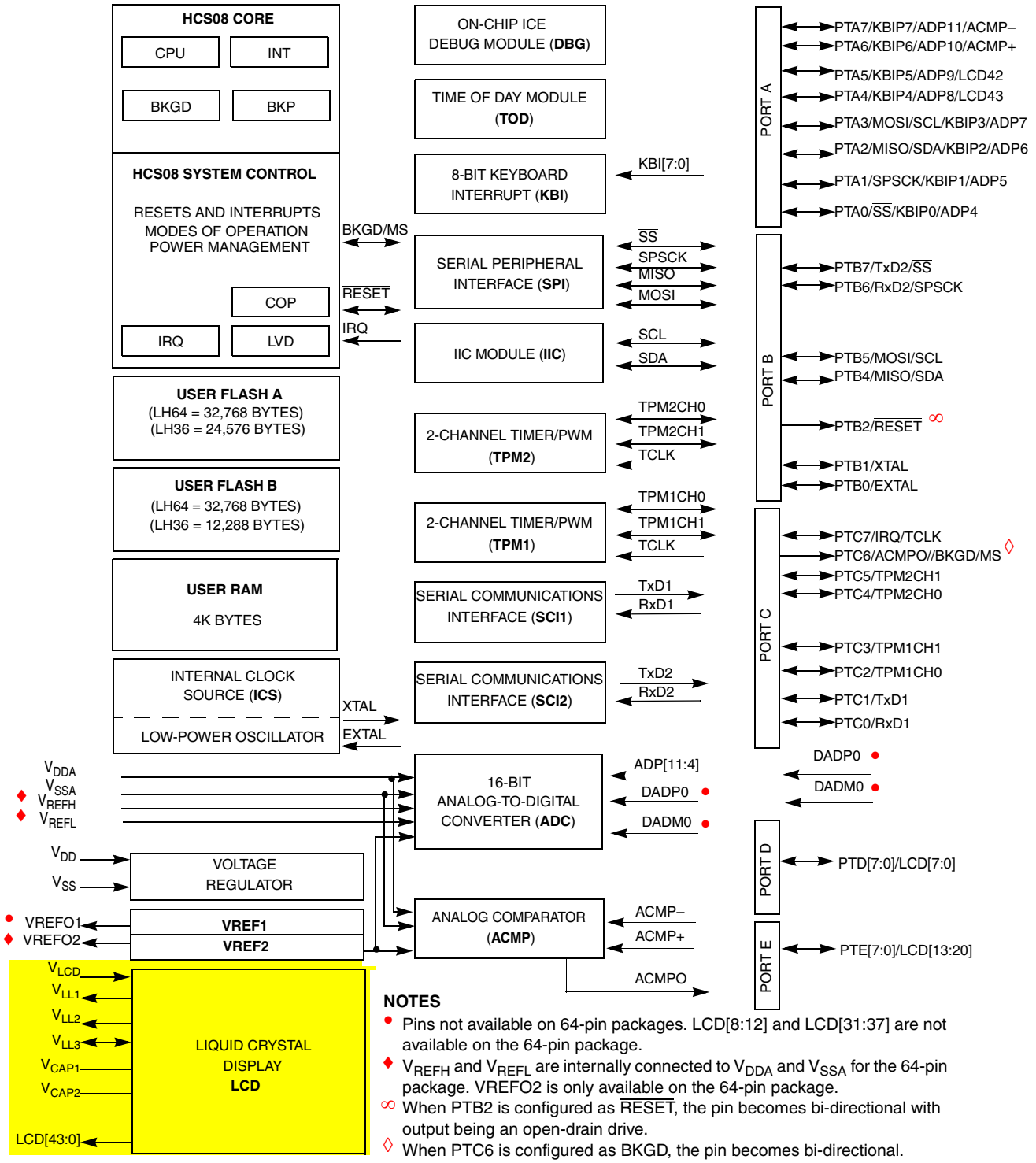


Figure 13-1. MC9S08LH64 Block Diagram Highlighting LCD Block and Pins

13.1.5 Features

The LCD module driver features include:

- LCD waveforms functional in LP_{Run}, LP_{Wait}, wait, stop2 and stop3 low-power modes
- 64 LCD (LCD[63:0]) pins with selectable frontplane/backplane configuration
  - Generate up to 63 frontplane signals
  - Generate up to 8 backplanes signals
- Programmable LCD frame frequency
- Programmable blink modes and frequency
  - All segments blank during blink period
  - Alternate display for each LCD segment in x4 or less mode
  - Blink operation in low-power modes
- Programmable LCD power supply switch, making it an ideal solution for battery-powered and board-level applications
  - Charge pump requires only four external capacitors
  - Internal LCD power using V_{DD} (1.8 to 3.6 V)
  - External V_{LCD} power supply option (.9 to 1.8 V)
  - Internal V_{IREG} regulated power supply option for 3 or 5V LCD glass
  - External V_{LL3} power supply option (3V)
- Internal regulated voltage source with a 4-bit trim register to apply contrast control
- Integrated charge pump for generating LCD bias voltages
  - Hardware configurable to drive 3-V or 5-V LCD panels
  - On-chip generation of bias voltages
- Waveform storage registers LCDWF
- Backplane reassignment to assist in vertical scrolling on dot-matrix displays
- Software configurable LCD frame frequency interrupt
- Internal ADC channels are connected to V_{LL1} and V_{LCD} to monitor their magnitudes. This feature allows software to adjust the contrast.

13.1.6 Modes of Operation

The LCD module supports the following operation modes:

Table 13-1. LCD-Module Operation Modes

Mode	Operation
Stop2	Depending on the state of the LCDSTP bit, the LCD module can operate an LCD panel in stop2 mode. If LCDSTP = 1, LCD module clock generation is turned off and the LCD module enters a power conservation state and is disabled.If LCDSTP = 0, the LCD module can operate an LCD panel in stop2, and the LCD module continues to display the current LCD panel contents based on the LCD operation prior to the stop2 event. If the LCD is enabled in stop2, the selected LCD clock source, OSCOUT, must be enabled to operate in stop2. The LCD frame interrupt does not cause the MCU to exit stop2.
Stop3	Depending on the state of the LCDSTP bit, the LCD module can operate an LCD panel in stop3 mode. If LCDSTP = 1, LCD module clock generation is turned off and the LCD module enters a power conservation state and is disabled. If LCDSTP = 0, the LCD module can operate an LCD panel in stop3, and the LCD module continues displaying the current LCD panel contents based on the LCD operation prior to the stop3 event. If the LCD is enabled in stop3, the selected LCD clock source, OSCOUT or the Alternate Clock, must be enabled to operate in stop3. In stop3 mode the LCD frame interrupt can cause the MCU to exit stop3.
LP_{Wait}, Wait	Depending on the configuration, the LCD module can operate an LCD panel in wait mode. If LCDWAI = 1, the LCD module clock generation is turned off and the LCD module enters a power-conservation state and is disabled. If LCDWAI = 0, the LCD module can operate an LCD panel in wait, and the LCD module continues displaying the current LCD panel contents base on the LCDWF registers. In wait mode, the LCD frame interrupt can cause the MCU to exit wait.

Stop2 provides the lowest power consumption state where the LCD module is functional. To operate the LCD in stop2 mode, use an external crystal.

13.1.7 Block Diagram

Figure 13-2 is a block diagram of the LCD module.

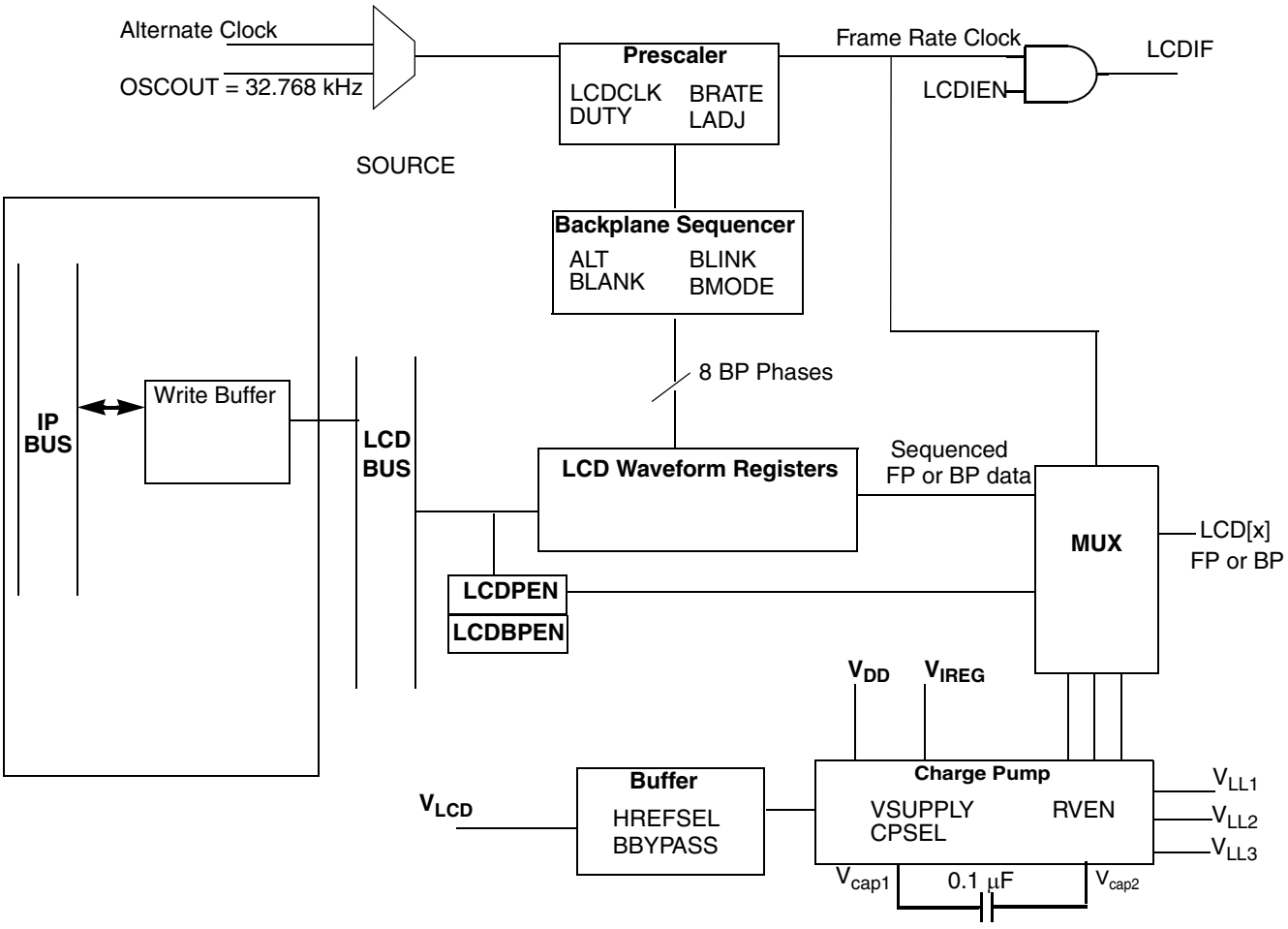


Figure 13-2. LCD Driver Block Diagram

13.2 External Signal Description

The LCD module has several external pins dedicated to power supply and LCD frontplane/backplane signaling. The LCD module can be configured to support eight backplane signals. The table below itemizes all the LCD external pins. See the Pins and Connections chapter for device-specific pin configurations.

Table 13-2. Signal Properties

Name	Port	Function	Reset State
64 LCD frontplane/backplane	LCD[63:0]	Switchable frontplane/backplane driver that connects directly to the display LCD[63:0] can operate as GPIO pins	High impedance
LCD voltage	V_{LCD}	LCD supply voltage	—
LCD bias voltages	V_{LL1}, V_{LL2}, V_{LL3}	LCD bias voltages	—
LCD charge pump capacitance	V_{cap1}, V_{cap2}	Charge pump capacitor pins	—

13.2.1 LCD[63:0]

When LCD functionality is enabled by the PEN[63:0] bits in the LCDPEN registers, the corresponding LCD[63:0] pin will generate a frontplane or backplane waveform depending on the configuration of the backplane-enable bit field (BPEN[63:0]).

13.2.2 V_{LCD}

V_{LCD} can be a source for LCD module-waveform generation. V_{LCD} is connected to a switch capacitor charge pump DC/DC converter that can generate double or triple V_{LCD} to support 3-V or 5-V LCD glass. A voltage source in the range from .9V to 1.8 V can be placed on $V_{LCD.}$ V_{LCD} is also connected internally to an ADC channel in order to monitor the V_{LCD} magnitude.

13.2.3 V_{LL1}, V_{LL2}, V_{LL3}

V_{LL1}, V_{LL2}, and V_{LL3} are bias voltages for the LCD module driver waveforms which can be internally generated using the internal charge pump (when enabled). The charge pump can also be configured to accept V_{LL3} as an input and generate V_{LL1} and V_{LL2}. V_{LL3} should never be set to a voltage other than V_{DD}. Refer to VSUPPLY[1:0] bits explanation.

13.2.4 V_{cap1}, V_{cap2}

The charge pump capacitor is used to transfer charge from the input supply to the regulated output. Use a ceramic capacitor.

13.3 Register Definition

This section consists of register descriptions. Each description includes a standard register diagram. Details of register bit and field function follow the register diagrams, in bit order.

13.3.1 LCD Control Register 0 (LCDC0)

	7	6	5	4	3	2	1	0
R W	LCDEN	SOURCE	LCLK2	LCLK1	LCLK0	DUTY2	DUTY1	DUTY0
Reset	0	0	0	0	0	0	1	1

= Unimplemented or Reserved

Figure 13-3. LCD Control Register 0 (LCDC0)

Read: anytime

Write: LCDEN anytime. Do not change SOURCE LCLCK OR DUTY while LCDEN = 1.

Table 13-3. LCDC0 Field Descriptions

Field	Description
7 LCDEN	**LCD Driver Enable** — LCDEN starts LCD-module-waveform generator. 0 All frontplane and backplane pins are disabled. The LCD module system is also disabled, and all LCD waveform generation clocks are stopped. V_{LL3} is connected to V_{DD} internally 1 LCD module driver system is enabled and frontplane and backplane waveforms are generated. All LCD pins enabled using the LCD pin enable register (LCDPEN[x]) will output an LCD module driver waveform.The backplane pins will output an LCD module driver backplane waveform based on the settings of DUTY[2:0]. Chargepump or resistor bias is enabled.
6 SOURCE	**LCD Clock Source Select** — The LCD module has two possible clock sources. This bit is used to select which clock source is the basis for LCDCLK. 0 Selects the OSCOUT (external clock reference) as the LCD clock source. 1 Selects the alternate clock as the LCD clock source.
5:3 LCLK[2:0]	**LCD Clock Prescaler** — Used as a clock divider to generate the LCD module frame frequency as shown in Equation 13-1. LCD-module-duty-cycle configuration is used to determine the LCD module frame frequency. LCD module frame frequency calculations are provided in Table 13-13./p.263. ***Eqn. 13-1*** $\text{LCD Module Frame Frequency} = \frac{\text{LCDCLK}}{((\text{DUTY}+1) \times 8 \times (4 + \text{LCLK[2:0]}) \times Y)}$ where 30< LCDCLK < 39.063 kHz where Y = 2,2,3,3,4,5,8,16 chosen by module duty cycle configuration
2:0 DUTY[2:0]	**LCD Duty Select** — DUTY[2:0] bits select the duty cycle of the LCD module driver. 000 Use 1 BP (1/1 duty cycle). 001 Use 2 BP (1/2 duty cycle). 010 Use 3 BP (1/3 duty cycle). 011 Use 4 BP (1/4 duty cycle). (Default) 100 Use 5 BP (1/5 duty cycle). 101 Use 6 BP (1/6 duty cycle). 110 Use 7 BP (1/7 duty cycle). 111 Use 8 BP (1/8 duty cycle).

13.3.2 LCD Control Register 1 (LCDC1)

	7	6	5	4	3	2	1	0
R	LCDIEN	0	0	0	0	FCDEN	LCDWAI	LCDSTP
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 13-4. LCD Control Register 1 (LCDC1)

Read: anytimeWrite: anytime

Table 13-4. LCDC1 Field Descriptions

Field	Description
7 LCDIEN	**LCD Module Frame Frequency Interrupt Enable** — Enables an LCD interrupt event that coincides with the LCD module frame frequency. 0 No interrupt request is generated by this event. 1 The start of the LCD module frame causes an LCD module frame frequency interrupt request.
2 FCDEN	**Full Complementary Drive Enable** — This bit allows GPIO that are shared with LCD pins to operate as full complementary if the other conditions necessary have been met. The other conditions are: VSUPPLY = 11 and RVEN = 0. 0 GPIO shared with LCD operate as open drain outputs, input levels and internal pullup resistors are referenced to V_{DD}. 1 If VSUPPLY = 11 and RVEN = 0, GPIO shared with LCD operate as full complementary outputs. Input levels and internal pullup resistors are referenced to V_{LL3}.
1 LCDWAI	**LCD Module Driver and Charge Pump Stop While in Wait Mode** 0 Allows the LCD driver and charge pump to continue running during wait mode. 1 Disables the LCD driver and charge pump when MCU goes into wait mode.
0 LCDSTP	**LCD Module Driver and Charge Pump Stop While in Stop2 or Stop3 Mode** 0 Allows LCD module driver and charge pump to continue running during stop2 or stop3. 1 Disables LCD module driver and charge pump when MCU goes into stop2 or stop3.

13.3.3 LCD Voltage Supply Register (LCDSUPPLY)

Module Base + 0x0022

	7	6	5	4	3	2	1	0
R	CPSEL	HREFSEL	LADJ1	LADJ0	0	BBYPASS	VSUPPLY1	VSUPPLY0
W								
Reset	0	0	1	1	0	1	0	1

Unimplemented or Reserved

Figure 13-5. LCD Voltage Supply Register (LCDSUPPLY)

Read: anytime

Write: anytime.

For proper operation, do not modify VSUPPLY[1:0] while the LCDEN bit is asserted. VSUPPLY[1:0] must also be configured according to the external hardware power supply configuration.

Table 13-5. LCDSUPPLY Field Descriptions

Field	Description
7 CPSEL	**Charge Pump or Resistor Bias Select** — Selects LCD module charge pump or a resistor network to supply the LCD voltages V_{LL1}, V_{LL2}, and V_{LL3}. See Figure 13-16 for more detail. 0 LCD charge pump is disabled. Resistor network selected (The internal 1/3-bias is forced.) 1 LCD charge pump is selected. Resistor network disabled (The internal 1/3-bias is forced.)
6 HREFSEL	**High Reference Select**— When using the V_{LCD} or V_{IREG} inputs, this bit configures internal circuits to supply V_{LL1}. 0 Divide input, V_{LCD} IN = V_{LCDEXT} * 2/3, V_{IREG} = 1.0V 1 Do not divide the input, V_{LCD} IN = V_{LCDEXT} * 3/3, V_{IREG} = 1.67 V
5:4 LADJ[1:0]	**LCD Module Load Adjust** — The LCD load adjust bits are used to configure the LCD module to handle different LCD glass capacitance. For CPSEL = 1 Adjust the clock source for the charge pump. Higher loads require higher charge pump clock rates. 00 - Fastest clock source for charge pump (LCD glass capacitance 8000pf or lower) 01 - Intermediate clock source for charge pump (LCD glass capacitance 6000pf or lower)) 10 - Intermediate clock source for charge pump (LCD glass capacitance 4000pf or lower) 11 - Slowest clock source for charge pump (LCD glass capacitance 2000pf or lower) For CPSEL = 0 Adjust the resistor bias network for different LCD glass capacitance 00 - Low Load (LCD glass capacitance 2000pf or lower) 01 - Low Load (LCD glass capacitance 2000pf or lower) 10 - High Load (LCD glass capacitance 8000pf or lower) 11 - High Load (LCD glass capacitance 8000pf or lower)
2 BBYPASS	**Op Amp Control** — Determines whether the internal LCD op amp buffer is bypassed. 0 Buffered mode 1 Unbuffered mode
1:0 VSUPPLY[1:0]	**Voltage Supply Control** — Configures whether the LCD module power supply is external or internal. Avoid modifying this bit field while the LCD module is enabled (e.g., LCDEN = 1). See Figure 13-16 for more detail. 00 Drive V_{LL2} internally from V_{DD} 01 Drive V_{LL3} internally from V_{DD} 10 Drive V_{LL1} internally from V_{LCD} 11 Drive V_{LL3} externally Or V_{IREG}

13.3.4 LCD Regulated Voltage Control Register (LCDRVC)

	7	6	5	4	3	2	1	0
R	RVEN	0	0	0	RVTRIM3	RVTRIM2	RVTRIM1	RVTRIM0
W								
Reset	0	0	0	0	1	0	0	0

Unimplemented or Reserved

Figure 13-6. LCD Regulated Voltage Control Register (LCDRVC)

Read: anytime.

Write: anytime.

The regulated voltage can be used to generate a reference signal to the LCD charge pump for 3V or 5V LCD operation dependant on the HREFSEL bit.

Table 13-6. LCDRVC Field Descriptions

Field	Description
7 RVEN	**Regulated Voltage Enable** — Enables internal voltage regulator, Must have charge pump enabled. 0 Regulated voltage disabled. 1 Regulated voltage enabled.
3:0 RVTRIM[3:0]	**Regulated Voltage Trim**—This 4 bit trim register is used to adjust the regulated input. Each bit in the register has equal weight. The Regulated input is changed by 1.5% for each count.

13.3.5 LCD Blink Control Register (LCDBCTL)

	7	6	5	4	3	2	1	0
R	BLINK	ALT	BLANK	0	BMODE	BRATE2	BRATE1	BRATE0
W								
Reset	0	0	0	0	0	0	0	0

Unimplemented or Reserved

Figure 13-7. LCD Blink Control Register (LCDBCTL)

Read: anytime

Write: anytime

Table 13-7. LCDBCTL Field Descriptions

Field	Description
7 BLINK	**Blink Command** — Starts or stops LCD module blinking 0 Disables blinking 1 Starts blinking at blinking frequency specified by LCD blink rate calculation (see Equation 13-2)
6 ALT	**Alternate Display Mode** — For four backplanes or less the LCD backplane sequencer changes to output an alternate display. ALT bit is ignored if Duty is 5 or greater. 0 Normal Display 1 Alternate display mode
5 BLANK	**Blank Display Mode** — Asserting this bit clears all segments in the LCD display. 0 Normal or Alternate Display 1 Blank Display Mode

Table 13-7. LCDBCTL Field Descriptions (continued)

Field	Description
3 BMODE	**Blink Mode** — Selects the blink mode displayed during the blink period. See Table 13-7 for more information on how BMODE affects the LCD display. 0 Display blank during the blink period 1 Display alternate display during blink period (Ignored if duty is 5 or greater)
2:0 BRATE[2:0]	**Blink-Rate Configuration**— Selects frequency at which the LCD display blinks when the BLINK is asserted. Equation 13-2 shows how BRATE[2:0] bit field is used in the LCD blink-rate calculation. Equation 13-2 provides an expression for the LCD module blink rate ***Eqn. 13-2*** $\text{LCD module blink rate} = \frac{\text{LCDCLK}}{2^{(12+\text{ BRATE[2:0]})}}$ LCD module blink rate calculations are provided in 13.4.3.2/p.270.

13.3.6 LCD Status Register (LCDS)

	7	6	5	4	3	2	1	0
R	LCDIF	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Unimplemented or Reserved

Figure 13-8. LCD Status Register (LCDS)

Read: anytime

Write: anytime

Table 13-8. LCDS Field Descriptions

Field	Description
7 LCDIF	**LCD Interrupt Flag** — LCDIF indicates an interrupt condition occurred. To clear the interrupt write a 1 to LCDIF. 0 interrupt condition has not occurred. 1 interrupt condition has occurred.

13.3.7 LCD Pin Enable Registers 0–7 (LCDPEN0–LCDPEN7)

When LCDEN = 1, these bits enable the corresponding LCD pin for LCD operation.

These registers should only be written with instructions that perform byte writes, using instructions that perform word writes will lead to invalid data being placed in the register. Initialize these registers before enabling the LCD module. Exiting Stop2 mode does not require reinitializing the LCDPEN registers.

		7	6	5	4	3	2	1	0
LCDPEN0	R W	PEN7	PEN6	PEN5	PEN4	PEN3	PEN2	PEN1	PEN0
	Reset	Indeterminate after reset							
LCDPEN1	R W	PEN15	PEN14	PEN13	PEN12	PEN11	PEN10	PEN9	PEN8
	Reset	Indeterminate after reset							
LCDPEN2	R W	PEN23	PEN22	PEN21	PEN20	PEN19	PEN18	PEN17	PEN16
	Reset	Indeterminate after reset							
LCDPEN3	R W	PEN31	PEN30	PEN29	PEN28	PEN27	PEN26	PEN25	PEN24
	Reset	Indeterminate after reset							
LCDPEN4	R W	PEN39	PEN38	PEN37	PEN36	PEN35	PEN34	PEN33	PEN32
	Reset	Indeterminate after reset							
LCDPEN5	R W	PEN47	PEN46	PEN45	PEN44	PEN43	PEN42	PEN41	PEN40
	Reset	Indeterminate after reset							
LCDPEN6	R W	PEN55	PEN54	PEN53	PEN52	PEN51	PEN50	PEN49	PEN48
	Reset	Indeterminate after reset							
LCDPEN7	R W	PEN63	PEN62	PEN61	PEN60	PEN59	PEN58	PEN57	PEN56
	Reset	Indeterminate after reset							

Unimplemented or Reserved

Figure 13-9. LCD Pin Enable Registers 0–7 (LCDPEN0–LCDPEN7)

Read: anytime

Write: anytime

Table 13-9. LCDPEN0–LCDPEN7 Field Descriptions

Field	Description
PEN[63:0]	**LCD Pin Enable** — The PEN[63:0] bit enables the LCD[63:0] pin for LCD operation. Each LCD[63:0] pin can be configured as a backplane or a frontplane based on the corresponding BPEN[n] bit in the Backplane Enable Register (LCDBPEN[7:0]). If LCDEN = 0, these bits have no effect on the state of the I/O pins. Set PEN[63:0] bits before LCDEN is set. 0 LCD operation disabled on LCDnn. 1 LCD operation enabled on LCDnn.

13.3.8 Backplane Enable Registers 0–7 (BPEN0–BPEN7)

When PEN[n] = 1, BPEN[x] configures the corresponding LCD pin to operate as an LCD backplane or an LCD frontplane. Most applications set a maximum of eight of these bits. Initialize these registers before enabling the LCD module. Exiting Stop2 mode does not require reinitializing the LCDBPEN registers.

These registers should only be written with instructions that perform byte writes, using instructions that perform word writes will lead to invalid data being placed in the register.

		7	6	5	4	3	2	1	0
LCDBPEN0	R W	BPEN7	BPEN6	BPEN5	BPEN4	BPEN3	BPEN2	BPEN1	BPEN0
	Reset	Indeterminate after reset							
LCDBPEN1	R W	BPEN15	BPEN14	BPEN13	BPEN12	BPEN11	BPEN10	BPEN9	BPEN8
	Reset	Indeterminate after reset							
LCDBPEN2	R W	BPEN23	BPEN22	BPEN21	BPEN20	BPEN19	BPEN18	BPEN17	BPEN16
	Reset	Indeterminate after reset							
LCDBPEN3	R W	BPEN31	BPEN30	BPEN29	BPEN28	BPEN27	BPEN26	BPEN25	BPEN24
	Reset	Indeterminate after reset							
LCDBPEN4	R W	BPEN39	BPEN38	BPEN37	BPEN36	BPEN35	BPEN34	BPEN33	BPEN32
	Reset	Indeterminate after reset							
LCDBPEN5	R W	BPEN47	BPEN46	BPEN45	BPEN44	BPEN43	BPEN42	BPEN41	BPEN40
	Reset	Indeterminate after reset							
LCDBPEN6	R W	BPEN55	BPEN54	BPEN53	BPEN52	BPEN51	BPEN50	BPEN49	BPEN48
	Reset	Indeterminate after reset							
LCDBPEN7	R W	BPEN63	BPEN62	BPEN61	BPEN60	BPEN59	BPEN58	BPEN57	BPEN56
	Reset	Indeterminate after reset							

Unimplemented or Reserved

Figure 13-10. Backplane Enable Registers 0–7 (BPEN0–BPEN7)

Read: anytime

Write: anytime

Table 13-10. LCDBPEN0–LCDBPEN7 Field Descriptions

Field	Description
BPEN[63:0]	**Backplane Enable** — The BPEN[63:0] bit configures the LCD[63:0] pin to operate as an LCD backplane or LCD frontplane. If LCDEN = 0, these bits have no effect on the state of the I/O pins. It is recommended to set BPEN[63:0] bits before LCDEN is set. 0 Frontplane operation enabled on LCD[n]. 1 Backplane operation enabled on LCD[n].

13.3.9 LCD Waveform Registers (LCDWF[63:0])

Each frontplane segment is associated with a backplane phase (A-H). For an LCD pin configured as a frontplante the LCDWF registers control the on/off state for frontplane segments.

For an LCD pin configured as a backplane the LCDWF registers controls the phase (A-H) in which the associated backplane pin is active.

These registers should only be written with instructions that perform byte writes, using instructions that perform word writes will lead to invalid data being placed in the register.

After reset, the LCDWF contents are indeterminate as indicated by Figure 13-11. Exiting Stop2 mode does not require reinitializing the LCDWF registers.

		7	6	5	4	3	2	1	0
LCDWF0	R W	BPHLCD0	BPGLCD0	BPFLCD0	BPELCD0	BPDLCD0	BPCLCD0	BPBLCD0	BPALCD0
	Reset	Indeterminate after reset							
LCDWF1	R W	BPHLCD1	BPGLCD1	BPFLCD1	BPELCD1	BPDLCD1	BPCLCD1	BPBLCD1	BPALCD1
	Reset	Indeterminate after reset							
LCDWF2	R W	BPHLCD2	BPGLCD2	BPFLCD2	BPELCD2	BPDLCD2	BPCLCD2	BPBLCD2	BPALCD2
	Reset	Indeterminate after reset							
LCDWF3	R W	BPHLCD3	BPGLCD3	BPFLCD3	BPELCD3	BPDLCD3	BPCLCD3	BPBLCD3	BPALCD3
	Reset	Indeterminate after reset							
LCDWF4	R W	BPHLCD4	BPGLCD4	BPFLCD4	BPELCD4	BPDLCD4	BPCLCD4	BPBLCD4	BPALCD4
	Reset	Indeterminate after reset							
LCDWF5	R W	BPHLCD5	BPGLCD5	BPFLCD5	BPELCD5	BPDLCD5	BPCLCD5	BPBLCD5	BPALCD5
	Reset	Indeterminate after reset							
LCDWF6	R W	BPHLCD6	BPGLCD6	BPFLCD6	BPELCD6	BPDLCD6	BPCLCD6	BPBLCD6	BPALCD6
	Reset	Indeterminate after reset							
LCDWF7	R W	BPHLCD7	BPGLCD7	BPFLCD7	BPELCD7	BPDLCD7	BPCLCD7	BPBLCD7	BPALCD7
	Reset	Indeterminate after reset							
LCDWF8	R W	BPHLCD8	BPGLCD8	BPFLCD8	BPELCD8	BPDLCD8	BPCLCD8	BPBLCD8	BPALCD8
	Reset	Indeterminate after reset							
LCDWF9	R W	BPHLCD9	BPGLCD9	BPFLCD9	BPELCD9	BPDLCD9	BPCLCD9	BPBLCD9	BPALCD9
	Reset	Indeterminate after reset							
LCDWF10	R W	BPHLCD10	BPGLCD10	BPFLCD10	BPELCD10	BPDLCD10	BPCLCD10	BPBLCD10	BPALCD10

Figure 13-11. LCD Waveform Registers (LCDWF[63:0])

	Reset	Indeterminate after reset							
LCDWF11	R W	BPHLCD11	BPGLCD11	BPFLCD11	BPELCD11	BPDLCD11	BPCLCD11	BPBLCD11	BPALCD11
	Reset	Indeterminate after reset							
LCDWF12	R W	BPHLCD12	BPGLCD12	BPFLCD12	BPELCD12	BPDLCD12	BPCLCD12	BPBLCD12	BPALCD12
	Reset	Indeterminate after reset							
LCDWF13	R W	BPHLCD13	BPGLCD13	BPFLCD13	BPELCD13	BPDLCD13	BPCLCD13	BPBLCD13	BPALCD13
	Reset	Indeterminate after reset							
LCDWF14	R W	BPHLCD14	BPGLCD14	BPFLCD14	BPELCD14	BPDLCD14	BPCLCD14	BPBLCD14	BPALCD14
	Reset	Indeterminate after reset							
LCDWF15	R W	BPHLCD15	BPGLCD15	BPFLCD15	BPELCD15	BPDLCD15	BPCLCD15	BPBLCD15	BPALCD15
	Reset	Indeterminate after reset							
LCDWF16	R W	BPHLCD16	BPGLCD16	BPFLCD16	BPELCD16	BPDLCD16	BPCLCD16	BPBLCD16	BPALCD16
	Reset	Indeterminate after reset							
LCDWF17	R W	BPHLCD17	BPGLCD17	BPFLCD17	BPELCD17	BPDLCD17	BPCLCD17	BPBLCD17	BPALCD17
	Reset	Indeterminate after reset							
LCDWF18	R W	BPHLCD18	BPGLCD18	BPFLCD18	BPELCD18	BPDLCD18	BPCLCD18	BPBLCD18	BPALCD18
	Reset	Indeterminate after reset							
LCDWF19	R W	BPHLCD19	BPGLCD19	BPFLCD19	BPELCD19	BPDLCD19	BPCLCD19	BPBLCD19	BPALCD19
	Reset	Indeterminate after reset							
LCDWF20	R W	BPHLCD20	BPGLCD20	BPFLCD20	BPELCD20	BPDLCD20	BPCLCD20	BPBLCD20	BPALCD20
	Reset	Indeterminate after reset							
LCDWF21	R W	BPHLCD21	BPGLCD21	BPFLCD21	BPELCD21	BPDLCD21	BPCLCD21	BPBLCD21	BPALCD21
	Reset	Indeterminate after reset							
LCDWF22	R W	BPHLCD22	BPGLCD22	BPFLCD22	BPELCD22	BPDLCD22	BPCLCD22	BPBLCD22	BPALCD22
	Reset	Indeterminate after reset							
LCDWF23	R W	BPHLCD23	BPGLCD23	BPFLCD23	BPELCD23	BPDLCD23	BPCLCD23	BPBLCD23	BPALCD23
	Reset	Indeterminate after reset							

Figure 13-11. LCD Waveform Registers (LCDWF[63:0]) (continued)

LCDWF24	R W	BPHLCD24	BPGLCD24	BPFLCD24	BPELCD24	BPDLCD24	BPCLCD24	BPBLCD24	BPALCD24
	Reset	Indeterminate after reset							
LCDWF25	R W	BPHLCD25	BPGLCD25	BPFLCD25	BPELCD25	BPDLCD25	BPCLCD25	BPBLCD25	BPALCD25
	Reset	Indeterminate after reset							
LCDWF26	R W	BPHLCD26	BPGLCD26	BPFLCD26	BPELCD26	BPDLCD26	BPCLCD26	BPBLCD26	BPALCD26
	Reset	Indeterminate after reset							
LCDWF27	R W	BPHLCD27	BPGLCD27	BPFLCD27	BPELCD27	BPDLCD27	BPCLCD27	BPBLCD27	BPALCD27
	Reset	Indeterminate after reset							
LCDWF28	R W	BPHLCD28	BPGLCD28	BPFLCD28	BPELCD28	BPDLCD28	BPCLCD28	BPBLCD28	BPALCD28
	Reset	Indeterminate after reset							
LCDWF29	R W	BPHLCD29	BPGLCD29	BPFLCD29	BPELCD29	BPDLCD29	BPCLCD29	BPBLCD29	BPALCD29
	Reset	Indeterminate after reset							
LCDWF30	R W	BPHLCD30	BPGLCD30	BPFLCD30	BPELCD30	BPDLCD30	BPCLCD30	BPBLCD30	BPALCD30
	Reset	Indeterminate after reset							
LCDWF31	R W	BPHLCD31	BPGLCD31	BPFLCD31	BPELCD31	BPDLCD31	BPCLCD31	BPBLCD31	BPALCD31
	Reset	Indeterminate after reset							
LCDWF32	R W	BPHLCD32	BPGLCD32	BPFLCD32	BPELCD32	BPDLCD32	BPCLCD32	BPBLCD32	BPALCD32
	Reset	Indeterminate after reset							
LCDWF33	R W	BPHLCD33	BPGLCD33	BPFLCD33	BPELCD33	BPDLCD33	BPCLCD33	BPBLCD33	BPALCD33
	Reset	Indeterminate after reset							
LCDWF34	R W	BPHLCD34	BPGLCD34	BPFLCD34	BPELCD34	BPDLCD34	BPCLCD34	BPBLCD34	BPALCD34
	Reset	Indeterminate after reset							
LCDWF35	R W	BPHLCD35	BPGLCD35	BPFLCD35	BPELCD35	BPDLCD35	BPCLCD35	BPBLCD35	BPALCD35
	Reset	Indeterminate after reset							
LCDWF36	R W	BPHLCD36	BPGLCD36	BPFLCD36	BPELCD36	BPDLCD36	BPCLCD36	BPBLCD36	BPALCD36
	Reset	Indeterminate after reset							
LCDWF37	R W	BPHLCD37	BPGLCD37	BPFLCD37	BPELCD37	BPDLCD37	BPCLCD37	BPBLCD37	BPALCD37

Figure 13-11. LCD Waveform Registers (LCDWF[63:0]) (continued)

	Reset	Indeterminate after reset							
LCDWF38	R W	BPHLCD38	BPGLCD38	BPFLCD38	BPELCD38	BPDLCD38	BPCLCD38	BPBLCD38	BPALCD38
	Reset	Indeterminate after reset							
LCDWF39	R W	BPHLCD39	BPGLCD39	BPFLCD39	BPELCD39	BPDLCD39	BPCLCD39	BPBLCD39	BPALCD39
	Reset	Indeterminate after reset							
LCDWF40	R W	BPHLCD40	BPGLCD40	BPFLCD40	BPELCD40	BPDLCD40	BPCLCD40	BPBLCD40	BPALCD40
	Reset	Indeterminate after reset							
LCDWF41	R W	BPHLCD41	BPGLCD41	BPFLCD41	BPELCD41	BPDLCD41	BPCLCD41	BPBLCD41	BPALCD41
	Reset	Indeterminate after reset							
LCDWF42	R W	BPHLCD42	BPGLCD42	BPFLCD42	BPELCD42	BPDLCD42	BPCLCD42	BPBLCD42	BPALCD42
	Reset	Indeterminate after reset							
LCDWF43	R W	BPHLCD43	BPGLCD43	BPFLCD43	BPELCD43	BPDLCD43	BPCLCD43	BPBLCD43	BPALCD43
	Reset	Indeterminate after reset							
LCDWF44	R W	BPHLCD44	BPGLCD44	BPFLCD44	BPELCD44	BPDLCD44	BPCLCD44	BPBLCD44	BPALCD44
	Reset	Indeterminate after reset							
LCDWF45	R W	BPHLCD45	BPGLCD45	BPFLCD45	BPELCD45	BPDLCD45	BPCLCD45	BPBLCD45	BPALCD45
	Reset	Indeterminate after reset							
LCDWF46	R W	BPHLCD46	BPGLCD46	BPFLCD46	BPELCD46	BPDLCD46	BPCLCD46	BPBLCD46	BPALCD46
	Reset	Indeterminate after reset							
LCDWF47	R W	BPHLCD47	BPGLCD47	BPFLCD47	BPELCD47	BPDLCD47	BPCLCD47	BPBLCD47	BPALCD47
	Reset	Indeterminate after reset							
LCDWF48	R W	BPHLCD48	BPGLCD48	BPFLCD48	BPELCD48	BPDLCD48	BPCLCD48	BPBLCD48	BPALCD48
	Reset	Indeterminate after reset							
LCDWF49	R W	BPHLCD49	BPGLCD49	BPFLCD49	BPELCD49	BPDLCD49	BPCLCD49	BPBLCD49	BPALCD49
	Reset	Indeterminate after reset							
LCDWF50	R W	BPHLCD50	BPGLCD50	BPFLCD50	BPELCD50	BPDLCD50	BPCLCD50	BPBLCD50	BPALCD50
	Reset	Indeterminate after reset							

Figure 13-11. LCD Waveform Registers (LCDWF[63:0]) (continued)

LCDWF51	R W	BPHLCD51	BPGLCD51	BPFLCD51	BPELCD51	BPDLCD51	BPCLCD51	BPBLCD51	BPALCD51
	Reset	Indeterminate after reset							
LCDWF52	R W	BPHLCD52	BPGLCD52	BPFLCD52	BPELCD52	BPDLCD52	BPCLCD52	BPBLCD52	BPALCD52
	Reset	Indeterminate after reset							
LCDWF53	R W	BPHLCD53	BPGLCD53	BPFLCD53	BPELCD53	BPDLCD53	BPCLCD53	BPBLCD53	BPALCD53
	Reset	Indeterminate after reset							
LCDWF54	R W	BPHLCD54	BPGLCD54	BPFLCD54	BPELCD54	BPDLCD54	BPCLCD54	BPBLCD54	BPALCD54
	Reset	Indeterminate after reset							
LCDWF55	R W	BPHLCD55	BPGLCD55	BPFLCD55	BPELCD55	BPDLCD55	BPCLCD55	BPBLCD55	BPALCD55
	Reset	Indeterminate after reset							
LCDWF56	R W	BPHLCD56	BPGLCD56	BPFLCD56	BPELCD56	BPDLCD56	BPCLCD56	BPBLCD56	BPALCD56
	Reset	Indeterminate after reset							
LCDWF57	R W	BPHLCD57	BPGLCD57	BPFLCD57	BPELCD57	BPDLCD57	BPCLCD57	BPBLCD57	BPALCD57
	Reset	Indeterminate after reset							
LCDWF58	R W	BPHLCD58	BPGLCD58	BPFLCD58	BPELCD58	BPDLCD58	BPCLCD58	BPBLCD58	BPALCD58
	Reset	Indeterminate after reset							
LCDWF59	R W	BPHLCD59	BPGLCD59	BPFLCD59	BPELCD59	BPDLCD59	BPCLCD59	BPBLCD59	BPALCD59
	Reset	Indeterminate after reset							
LCDWF60	R W	BPHLCD60	BPGLCD60	BPFLCD60	BPELCD60	BPDLCD60	BPCLCD60	BPBLCD60	BPALCD60
	Reset	Indeterminate after reset							
LCDWF61	R W	BPHLCD61	BPGLCD61	BPFLCD61	BPELCD61	BPDLCD61	BPCLCD61	BPBLCD61	BPALCD61
	Reset	Indeterminate after reset							
LCDWF62	R W	BPHLCD62	BPGLCD62	BPFLCD62	BPELCD62	BPDLCD62	BPCLCD62	BPBLCD62	BPALCD62
	Reset	Indeterminate after reset							
LCDWF63	R W	BPHLCD63	BPGLCD63	BPFLCD63	BPELCD63	BPDLCD63	BPCLCD63	BPBLCD63	BPALCD63
	Reset	Indeterminate after reset							

Figure 13-11. LCD Waveform Registers (LCDWF[63:0]) (continued)

Table 13-11. LCDWF Field Descriptions

Field	Description
BP[y]LCD[x]	**Segment-on-Frontplane Operation** — If the LCD[x] pin is enabled and configured to operate as a frontplane, the BP[y]LCD[x] bit in the LCDWF registers controls the on/off state for the LCD segment connected between LCD[x] and BP[y].BP[y] corresponds to an LCD[63:0] pin enabled and configured to operate as a backplane that is active in phase [y]. Asserting BP[y]LCD[x] displays (turns on) the LCD segment connected between LCD[x] and BP[y]. 0 LCD segment off 1 LCD segment on **Segment-on-Backplane Operation** — If the LCD[x] pin is enabled and configured to operate as a backplane, the BP[y] LCD[x] bit in the LCDWF registers controls the phase (A-H) in which the LCD[x] pin is active.Backplane phase assignment is done using this method. 0 LCD backplane inactive for phase[y] 1 LCD backplane active for phase[y].

13.4 Functional Description

This section provides a complete functional description of the LCD block, detailing the operation of the design from the end-user perspective.

Before enabling the LCD module by asserting the LCDEN bit in the LCDC0 register, configure the LCD module based on the end application requirements. Out of reset, the LCD module is configured with default settings, but these settings are not optimal for every application. The LCD module provides several versatile configuration settings and options to support varied implementation requirements, including:

- Frame frequency
- Duty cycle (number of backplanes)
- Backplane assignment (which LCD[63:0] pins operate as backplanes)
- Frame frequency interrupt enable
- Blinking frequency and options
- Power-supply configurations

The LCD module also provides an LCD pin enable control. Setting the LCD pin enable bit (PEN[x] in the LCDPEN[y] register) for a particular LCD[y] pin enables the LCD module functionality of that pin once the LCDEN bit is set. When the BPEN[x] bit in the LCDBPEN[y] is set, the associated pin operates as a backplane. The LCDWF registers can then activate (display) the corresponding LCD segments on an LCD panel.

The LCDWF registers control the on/off state for the segments controlled by the LCD pins defined as front planes and the active phase for the backplanes. Blank display modes do not use the data from the LCDWF registers. When using the LCDWF register for frontplane operation, writing a 0 turns the segment off.

For pins enabled as backplane, the phase of the backplane (A-H) is assigned by the LCDWF register for the corresponding backplane pin. For a detailed description of LCD module operation for a basic seven-segment LCD display, see Section 13.6.1, “LCD Seven Segment Example Description.”.

13.4.1 LCD Driver Description

The LCD module driver has 8 modes of operation:

- 1/1 duty (1 backplane) (Phase A), 1/1 bias (2 voltage levels)
- 1/2 duty (2 backplanes) (Phase A, B), 1/3 bias (4 voltage levels)
- 1/3 duty (3 backplanes) (Phase A, B, C), 1/3 bias (4 voltage levels)
- 1/4 duty (4 backplanes) (Phase A, B, C, D), 1/3 bias (4 voltage levels)
- 1/5 duty (5 backplanes) (Phase A, B, C, D, E), 1/3 bias (4 voltage levels)
- 1/6 duty (6 backplanes) (Phase A, B, C, D, E, F), 1/3 bias (4 voltage levels)
- 1/7 duty (7 backplanes) (Phase A, B, C, D, E, F, G), 1/3 bias (4 voltage levels)
- 1/8 duty (8 backplanes) (Phase A, B, C, D, E, F, G, H), 1/3 bias (4 voltage levels)

All modes are 1/3 bias. These modes of operation are described in more detail in the following sections.

13.4.1.1 LCD Duty Cycle

The denominator of the duty cycle indicates the number of LCD panel segments capable of being driven by each individual frontplane output driver. Depending on the duty cycle, the LCD waveform drive can be categorized as static or multiplexed.

In static-driving method, the LCD is driven with two square waveforms. The static-driving method is the most basic method to drive an LCD panel, but because each frontplane driver can drive only one LCD segment, static driving limits the LCD segments that can be driven with a given number of frontplane pins. In static mode, only one backplane is required.

In multiplexed mode, the LCD waveforms are multi-level and depend on the bias mode. Multiplex mode, depending on the number of backplanes, can drive multiple LCD segments with a single frontplane driver. This reduces the number of driver circuits and connections to LCD segments. For multiplex mode operation, at least two backplane drivers are needed. The LCD module is optimized for multiplex mode.

The duty cycle indicates the amount of time the LCD panel segment is energized during each LCD module frame cycle. The denominator of the duty cycle indicates the number of backplanes that are being used to drive an LCD panel.

The duty cycle is used by the backplane phase generator to set the phase outputs. The phase outputs A-H are driven according to the sequence shown below. The sequence is repeated at the LCD frame frequency. The duty cycle is configured using the DUTY[2:0] bit field in the LCDC0 register, as shown in Table 13-12.

Table 13-12. LCD Module Duty Cycle Modes

Duty	LCDC0 Register			Number of Backplanes	Phase Sequence
	DUTY2	DUTY1	DUTY0		
1/1	0	0	0	1	A
1/2	0	0	1	2	A B

Table 13-12. LCD Module Duty Cycle Modes

Duty	LCDC0 Register			Number of Backplanes	Phase Sequence
	DUTY2	DUTY1	DUTY0		
1/3	0	1	0	3	A B C
1/4	0	1	1	4	A B C D
1/5	1	0	0	5	A B C D E
1/6	1	0	1	6	A B C D E F
1/7	1	1	0	7	A B C D E F G
1/8	1	1	1	8	A B C D E F G H

13.4.1.2 LCD Bias

Because a single frontplane driver is configured to drive more and more individual LCD segments, 3 voltage levels are required to generate the appropriate waveforms to drive the segment. The LCD module is designed to operate using the 1/3 bias mode.

13.4.1.3 LCD Module Base Clock and Frame Frequency

The LCD module is optimized to operate using a 32.768-kHz clock input. Two clock sources are available to the LCD module, which are selectable by configuring the SOURCE bit in the LCDC0 register. The two clock sources include:

- External crystal — OSCOUT (SOURCE = 0)
- Alternate clock (SOURCE = 1)

Figure 13-12 shows the LCD clock tree. The clock tree shows the two possible clock sources and the LCD frame frequency and blink frequency clock source. The LCD blink frequency is discussed in Section 13.4.3.2, “Blink Frequency.”

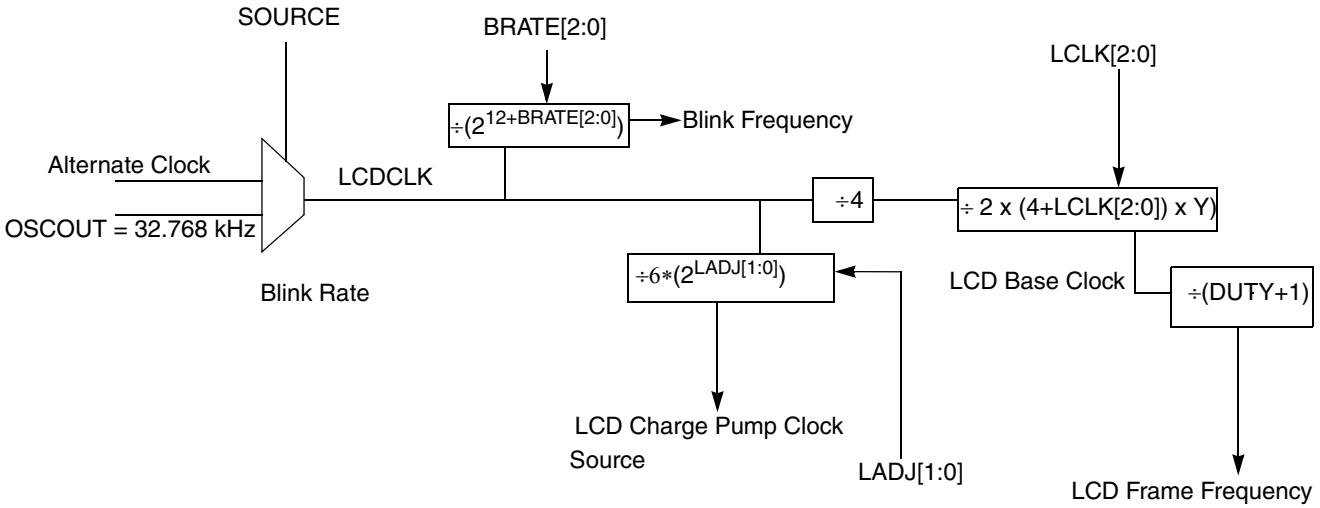


Figure 13-12. LCD Clock Tree

An external 32.768-kHz clock input is required to achieve lowest power consumption.

The value of LCDCLK is important because it is used to generate the LCD module frame frequency. Equation 13-1 provides an expression for the LCD module frame frequency calculation.

The LCD module frame frequency is a function of the LCD module duty cycle as shown in Equation 13-1. Table 13-14 and Table 13-13 show LCD module frame frequency calculations that consider several possible LCD module configurations of LCLK[2:0] and DUTY[2:0].

The LCD module frame frequency is defined as the number of times the LCD segments are energized per second. The LCD module frame frequency must be selected to prevent the LCD display from flickering (LCD module frame frequency is too low) or ghosting (LCD module frame frequency is too high). To avoid these issues, an LCD module frame frequency in the range of 28 to 58 Hz is required. LCD module frame frequencies less than 28 Hz or greater than 58 Hz are out of specification, and so are invalid. Selecting lower values for the LCD base and frame frequency results in lower current consumption for the LCD module.

The LCD module base clock frequency is the LCD module frame frequency multiplied by the number of backplane phases that are being generated. The number of backplane phases is selected using the DUTY[2:0] bits. The LCD module base clock is used by the backplane sequencer to generate the LCD waveform data for the enabled phases (A-H).

Table 13-13. LCD Module Frame Frequency Calculations[1]

Duty Cycle	1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8
Y	16	8	5	4	3	3	2	2
LCLK[2:0]								
0	64	64	68.3	64	68.3	56.9	73.1	64
1	51.2	51.2	54.6	51.2	54.6	45.5	58.5	51.2
2	42.7	42.7	45.5	42.7	45.5	37.9	48.8	42.7
3	36.6	36.6	39	36.6	39	32.5	41.8	36.6
4	32	32	34.1	32	34.1	28.4	36.6	32
5	28.4	28.4	30.3	28.4	30.3	25.3	32.5	28.4
6	25.6	25.6	27.3	25.6	27.3	22.8	29.3	25.6
7	23.3	23.3	24.8	23.3	24.8	20.7	26.6	23.3

[1] LCD clock input ~ 32.768 kHz

Shaded table entries are out of specification and invalid.

Table 13-14. LCD Module Frame Frequency Calculations[1]

Duty Cycle	1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8
Y	16	8	5	4	3	3	2	2
LCLK[2:0]								
0	76.3	76.3	81.4	76.3	81.4	67.8	87.2	76.3
1	61	61	65.1	61	65.1	54.3	69.8	61
2	50.9	50.9	54.3	50.9	54.3	45.2	58.1	50.9
3	43.6	43.6	46.5	43.6	46.5	38.8	49.8	43.6
4	38.1	38.1	40.7	38.1	40.7	33.9	43.6	38.1
5	33.9	33.9	36.2	33.9	36.2	30.1	38.8	33.9
6	30.5	30.5	32.6	30.5	32.6	27.1	34.9	30.5
7	27.7	27.7	29.6	27.7	29.6	24.7	31.7	27.7

[1] LCD clock input ~ 39.063 kHz

Shaded table entries are out of specification and invalid.

13.4.1.4 LCD Waveform Examples

This section shows the timing examples of the LCD output waveforms for the several modes of operation. As shown in Table 13-15, all examples use 1/3 bias mode.

Table 13-15. Configurations for Example LCD Waveforms

	Bias Mode	DUTY[2:0]	Duty Cycle
Example 1	1/3	001	1/2
Example 2		011	1/4
Example 3		111	1/8

13.4.1.4.1 1/2 Duty Multiplexed with 1/3 Bias Mode (Low-power Waveform)

Duty=1/2:DUTY[2:0] = 001

LCD pin 0 (LCD[0])and LCD pin 1, LCD[1] enabled as backplanes:

BPEN0 =1 and BPEN1 =1 in the LCDBPEN0

LCD[0] assigned to Phase A: LCDWF0 = 0x01

LCD[1] assigned to Phase B: LCDWF1 = 0x02

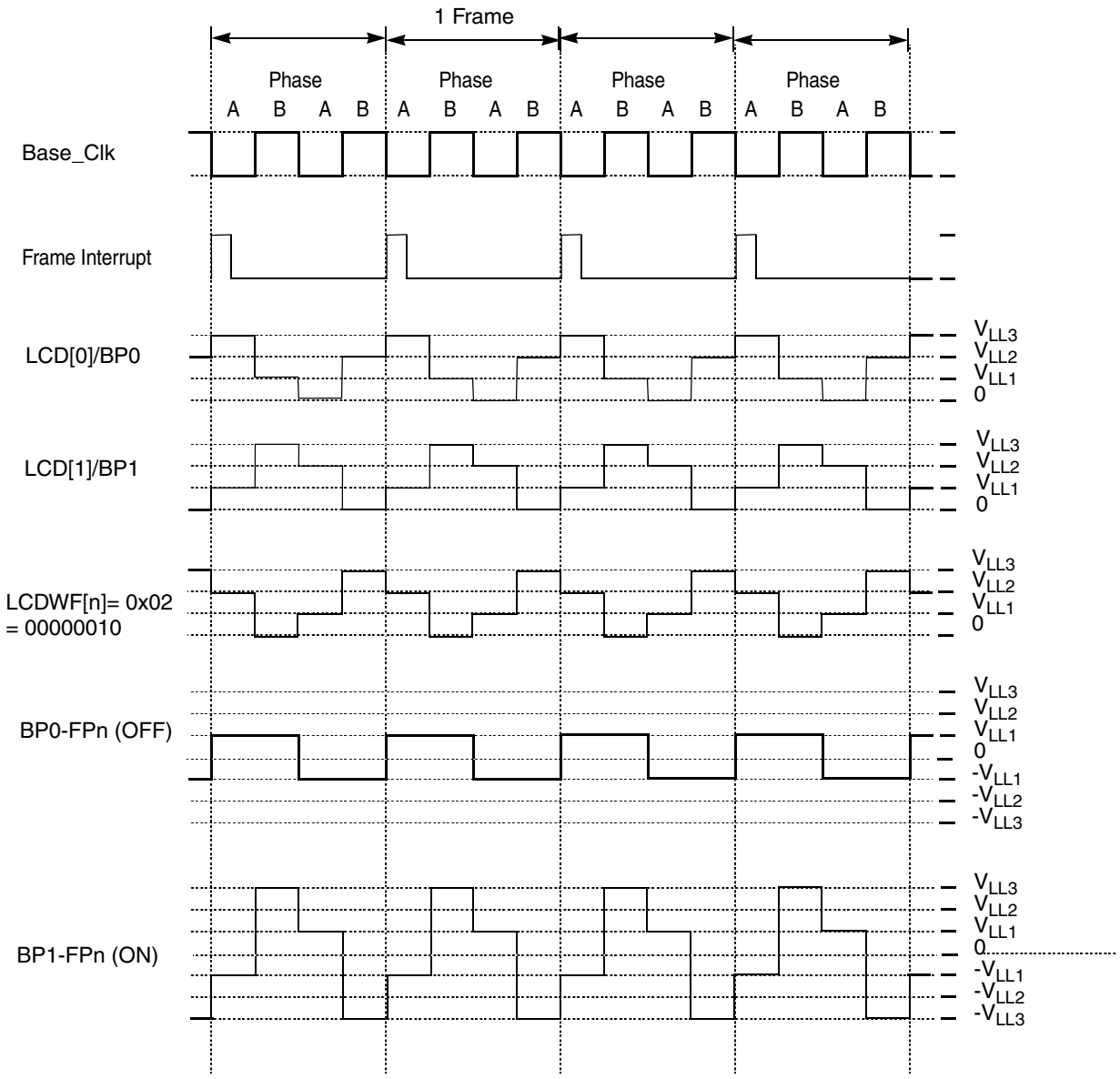


Figure 13-13. 1/2 Duty and 1/3 Bias (Low-Power Waveform)

13.4.1.4.2 1/4 Duty Multiplexed with 1/3 Bias Mode (Low-power Waveform)

Duty = 1/4: DUTY[2:0] = 011

LCD pins 0 – 3 enabled as backplanes: LCDBPEN0 = 0x0F

LCD[0] assigned to Phase A: LCDWF0 = 0x01

LCD[1] assigned to Phase B: LCDWF1 = 0x02

LCD[2] assigned to Phase C: LCDWF2 = 0x04

LCD[3] assigned to Phase D: LCDWF3 = 0x08

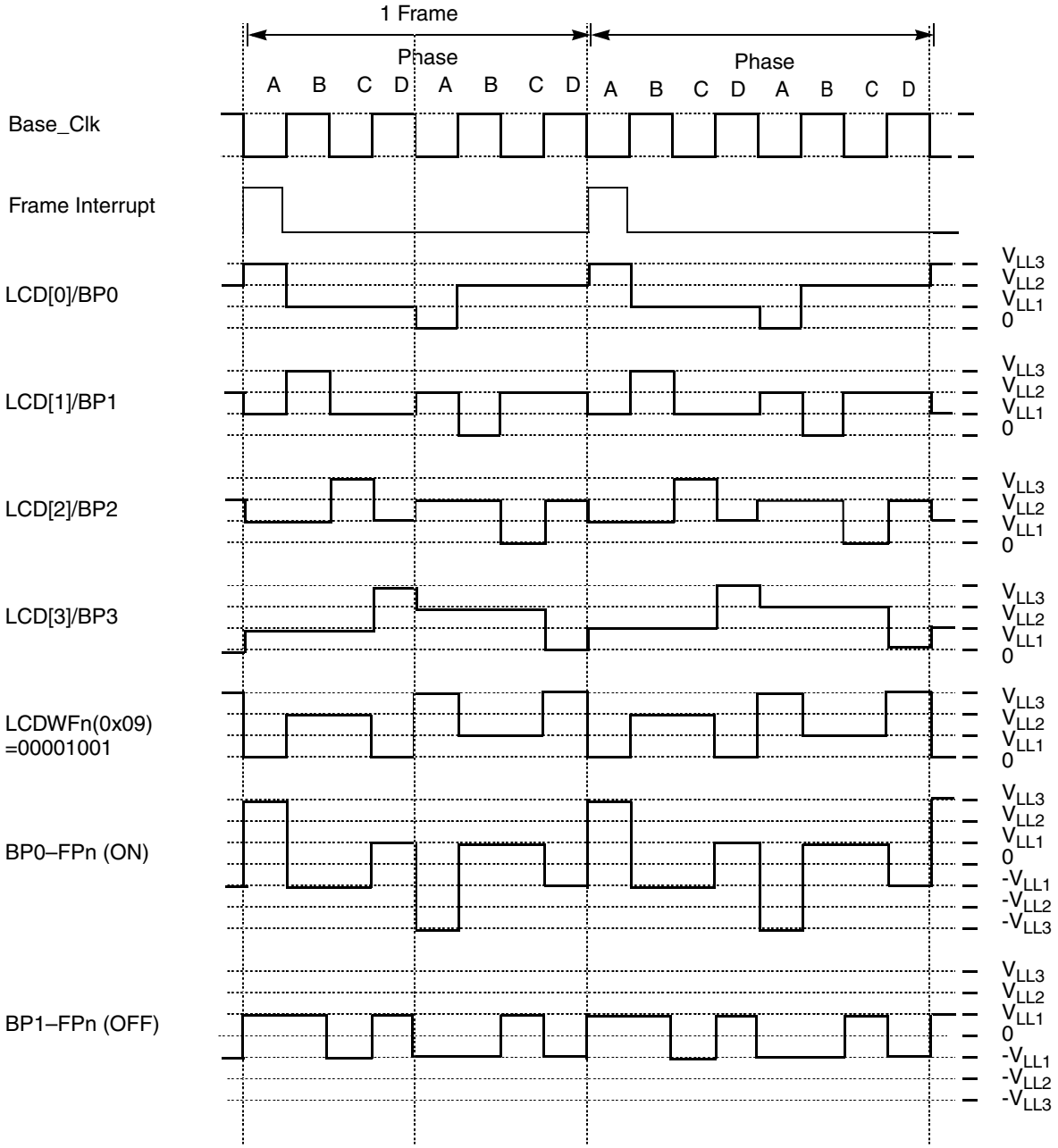


Figure 13-14. 1/4 Duty and 1/3 Bias (Low-Power Waveform)

13.4.1.4.3 1/8 Duty Multiplexed with 1/3 Bias Mode (Low-power Waveform)

Duty = 1/8:DUTY[2:0] = 111

LCD pins 0 – 7 enabled as backplanes: LCDBPEN0 = 0xFF

LCD[0] assigned to Phase A: LCDWF0 = 0x01

LCD[1] assigned to Phase B: LCDWF1 = 0x02

LCD[2] assigned to Phase C: LCDWF2 = 0x04

LCD[3] assigned to Phase D: LCDWF3 = 0x08

LCD[4] assigned to Phase E: LCDWF4 = 0x10

LCD[5] assigned to Phase F: LCDWF5 = 0x20

LCD[6] assigned to Phase G: LCDWF6 = 0x40

LCD[7] assigned to Phase H: LCDWF7 = 0x80

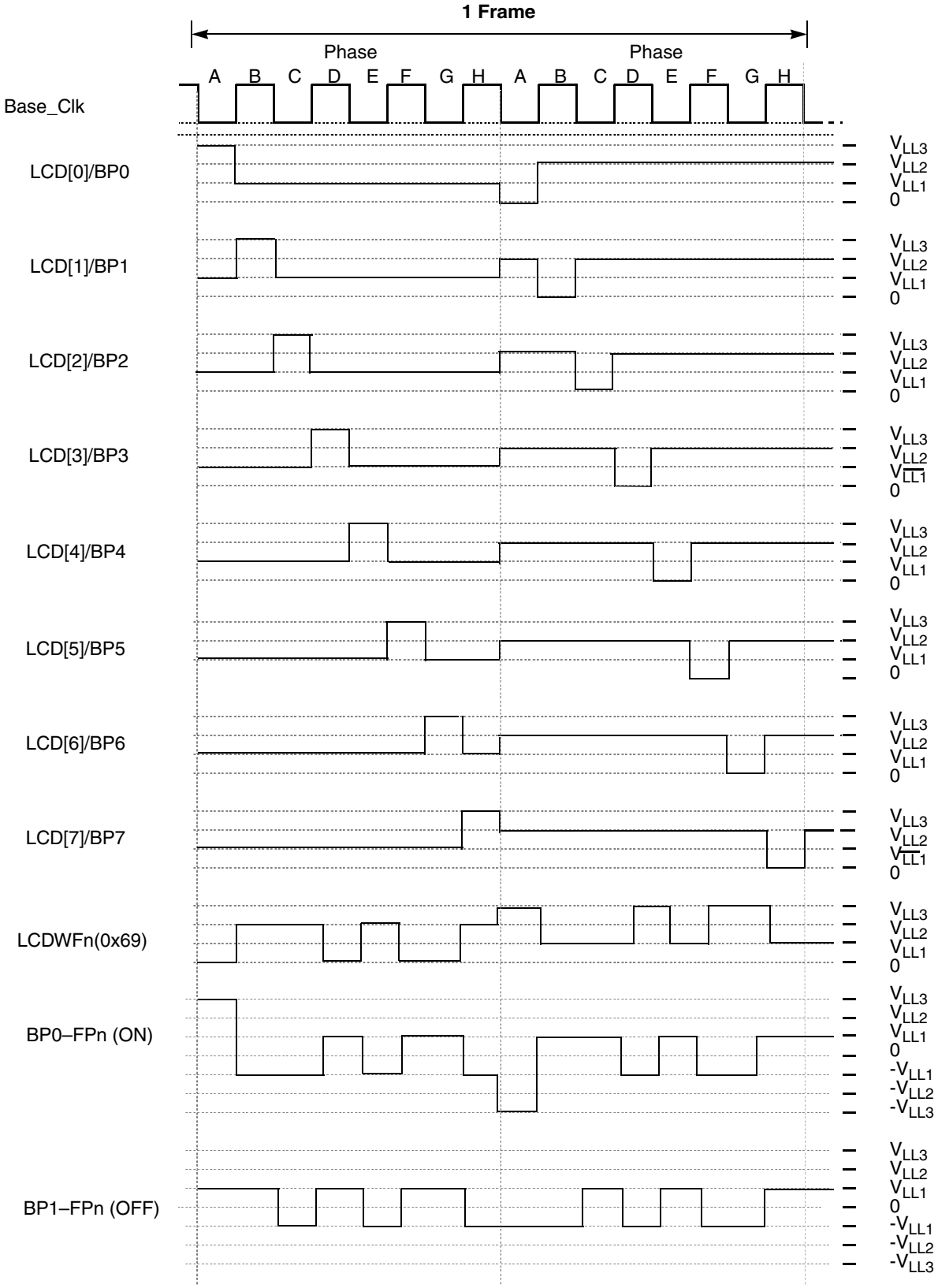


Figure 13-15. 1/8 Duty and 1/3 Bias (Low-power Waveform)

13.4.2 LCDWF Registers

For a segment on the LCD panel to be displayed, data must be written to the LCDWF registers. For LCD pins enabled as frontplanes, each bit in the LCDWF registers corresponds to a segment on an LCD panel. The different phases A-H represent the different backplanes of the LCD panel. The selected LCD duty cycle controls the number of implemented phases. Refer to Table 13-12 for normal LCD operation the phases follow the sequence shown.

For LCD pins enabled as a backplane, the LCDWF assigns the phase in which the backplane pin is active. This is how backplane assignment is done.

An example of normal operation follows: enable LCD pin 0 to operate as backplane 0. Enable the LCD pin 0 by setting PEN0 bit in the LCDPEN0 register. Configure LCD pin 0 as a backplane pin by setting the BPEN0 bit in the LCDBPEN0 register. Finally, the BPALCD0 bit in the LCDWF0 is set to associate LCD pin 0 with backplane phase A.This will configure LCD0 to operate as a backplane that is active in Phase A.

For LCD pins enabled as a frontplane, writing a 1 to a given LCDWF location results in the corresponding display segment being driven with the differential root mean square (RMS) voltage necessary to turn the segment on during the phase selected. Writing a 0 to a given location results in the corresponding display segment being driven with the differential RMS voltage necessary to turn the segment off during the phase selected.

13.4.3 LCD Display Modes

The LCD module can be configured to implement several different display modes. The bits ALT and BLANK in the LCD-blink-control register (LCDBCTL) configure the different display modes. In normal display mode (default), LCD segments are controlled by the data placed in the LCDWF registers, as described in Section 13.4.2, “LCDWF Registers.” For blank-display mode, the LCDWF data is bypassed and the frontplane and backplane pins are configured to clear all segments.

For alternate-display mode, the backplane sequence is modified for duty cycles of 1/4, 1/3, 1/2, and 1/1. For four backplanes or less, the backplane sequence is modified as shown below. The altered sequence allows two complete displays to be placed in the LDCDWF registers. The first display is placed in phases A-D and the second in phases E-H in the case of four backplanes. If the LCD duty cycle is five backplanes or greater, the ALT bit is ignored and creates a blank display. Refer to Table 13-17 for additional information.

Using the alternate display function an inverse display can be accomplished for x4 mode and less by placing inverse data in the alternate phases of the LCDWF registers.

Table 13-16. Alternate Display Backplane Sequence

Duty	Backplane Sequence	Alt. Backplane Sequence
1/1	A	E
1/2	A B	E F

Table 13-16. Alternate Display Backplane Sequence (continued)

Duty	Backplane Sequence	Alt. Backplane Sequence
1/3	A B C	E F G
1/4	A B C D	E F G H

13.4.3.1 LCD Blink Modes

The blink mode is used as a means of alternating among different LCD display modes at a defined frequency. The LCD module can be configured to implement two blink modes. The BMODE bit in the LCD-blink-control register (LCDBCTL) configures the different blink modes. Blink modes are activated by setting the BLINK bit in the LCDBCTL register. If BLINK = 0, the LCD module operates normally as described Section 13.4.3, "LCD Display Modes". If BLINK = 1, BMODE bit configures the blinking operation. During a blink, the display data driven by the LCD module changes to the mode selected by the BMODE bit. The BMODE bit selects two different blink modes, blank and alternate modes operate in the same way, as defined in Section 13.4.3, "LCD Display Modes." The table below shows the interaction between display modes and blink modes. If the LCD duty cycle is five backplanes or greater, BMODE = 1 is ignored and will revert to create a blank display during the blink period.

Table 13-17. Display Mode Interaction

BLANK	ALT	BMODE	LCD Duty	BLINK = 1	
				Normal Period	Blink Period
0	0	0	1-4	Normal Display	Blank Display
0	0	1	1-4	Normal Display	Alternate display
0	1	0	1-4	Alternate display	Blank Display
0	1	1	1-4	Alternate Display	Alternate display
1	X	0	1-4	Blank Display	Blank DIsplay
1	X	1	1-4	Blank Display	Alternate display
0	X	X	5-8	Normal Display	Blank Display
1	X	X	5-8	Blank Display	Blank Display

13.4.3.2 Blink Frequency

The LCD clock is the basis for the calculation of the LCD module blink frequency. The LCD module blink frequency is equal to the LCD clock (LCDCLK) divided by the factor selected by the BRATE[2:0] bits. Table 13-18 shows LCD module blink frequency calculations for all values of BRATE[2:0] at a few common LCDCLK selections.

Table 13-18. Blink Frequency Calculations
(Blink Rate = LCD Clock(Hz) ÷ Blink Divider)

BRATE[2:0]	0	1	2	3	4	5	6	7
LCD Clock	**Blink Frequency (Hz)**							
30 khz	7.32	3.66	1.831	.916	.46	.23	.11	.06
32.768 khz	8	4	2	1	.5	.25	.13	.06
39.063 khz	9.54	4.77	2.38	1.19	.6	.30	.15	.075

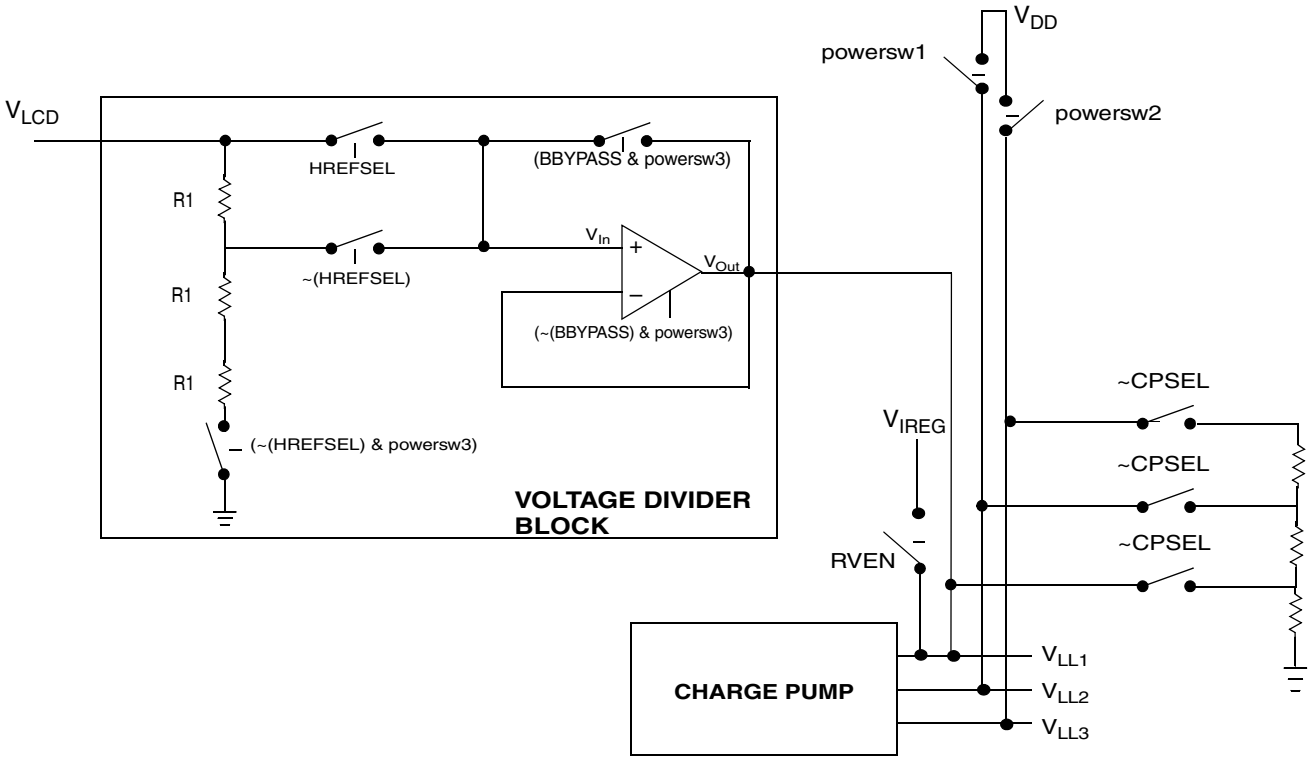
13.4.4 LCD Charge Pump, Voltage Divider, and Power Supply Operation

This section describes the LCD charge pump, voltage divider, and LCD power supply configuration options. Figure 13-16 provides a block diagram for the LCD charge pump and the resistor divider network.

The LCD bias voltages (V_{LL1}, V_{LL2} and V_{LL3}) can be generated by the LCD charge pump or a resistor divider network that is connected using the CPSEL bit. The input source to the LCD charge pump is controlled by the VSUPPLY[1:0] bit field.

VSUPPLY[1:0] indicates the state of internal signals used to configure power switches as shown in the table in Figure 13-16. The block diagram in Figure 13-16 illustrates several potential operational modes for the LCD module including configuration of the LCD module power supply source using V_{DD}, internal regulated voltage V_{IREG} or an external supply on the V_{LL3} or V_{LCD} pins. V_{LL3} should never exceed V_{DD}.

Upon Reset the VSUPPLY[1:0] bits are configured to connect V_{LL3} to V_{DD}. This configuration should be changed to match the application requirements before the LCD module is enabled.



VSUPPLY[1:0]	Configuration	powersw1	powersw2	powersw3
00	Drive V_{LL2} internally from V_{DD}	1	0	0
01	Drive V_{LL3} internally from V_{DD}	0	1	0
10	Drive V_{LL1}internally from the V_{LCD} pin	0	0	1
11	Drive V_{LL3} externally from V_{DD} Or Drive V_{LL1} internally from V_{IREG}	0	0	0

Figure 13-16. LCD Charge Pump and V_{LCD} Voltage Divider Block Diagram

Figure 13-16 also illustrates a buffer, a voltage follower with an ideal op amp. The buffer, if enabled, gives $V_{In} = V_{Out}$, and, because the input impedance of the op amp is very high, V_{In} is isolated from V_{Out} This isolation can protect V_{In} from current draw from V_{Out}; however, if the buffer is disabled ((~BBYPASS & powersw3) = 0), the output and input will be configured in a tri-state condition; that is, they will be floating.

NOTE:

The charge pump is optimized for 1/3 bias mode operation only.

During the first 16 timebase clock cycles after the LCDCPEN bit is set, all the LCD frontplane and backplane outputs are disabled, regardless of the state of the LCDEN bit.

The charge pump requires external capacitance for its operation. To provide this external capacitance, the V_{cap1} and V_{cap2} external pins are provided. It is recommended that a ceramic capacitor be used. Proper orientation is imperative when using a polarized capacitor. The recommended value for the external capacitor is 0.1 µF.

13.4.4.1 LCD Charge Pump and Voltage Divider

Using the voltage divider and charge pump, the LCD module can effectively double or triple the input voltage placed on the V_{LCD} pin.{statement} This LCD module configurability makes the LCD module compatible with both 3-V or 5-V LCD glass.

The LCD module high reference select bit (HREFSEL) in the LCDSUPPLY register configures the LCD module operational mode as a voltage doubler or a voltage tripler. In Figure 13-16, HREFSEL bit signal is used to control switches within the voltage divider block to enable or disable the two-thirds ($2/3 * V_{LCD}$) voltage divider. If HREFSEL = 0, the LCD module is configured as a voltage doubler, by enabling the voltage divider. With this configuration if V_{LCD} = 1.5 V the bias voltages generated will be:

$V_{LCD} * 2/3 = 1.5\ V * 2/3 = 1\ V = V_{LL1}$

$V_{LL1} * 2 = 2\ V = V_{LL2}$

$V_{LL1} * 3 = 3\ V = V_{LL3}$

If HREFSEL = 1, the LCD module is configured as a voltage tripler by disabling the voltage divider. ($V_{LL1} = V_{LCD}$) The HREFSEL configuration depends on the LCD panel operating voltage specification in the application.

13.4.4.1.1 CPSEL: LCD Charge Pump or Resistor Bias Enable

The CPSEL bit in the LCDSUPPLY register selects the charge pump. When the charge pump is selected (CPSEL = 1), an internal charge pump is used to generate the LCD bias voltages; V_{LL1}, V_{LL2}, and V_{LL3}.

When the charge pump is unselected (CPSEL = 0), V_{LL3} must be supplied and V_{LL1}, V_{LL2} are generated by a resistor bias network.

13.4.4.2 LCD Power Supply and Voltage Buffer Configuration

The LCD bias voltages can be internally derived from V_{DD}, internally derived from a voltage source (must not exceed V_{DD}) connected to V_{LL3}, internally derived from a regulated voltage source that can be configured to supply 1.0 or 1.67 V (V_{IREG}), or it can be internally derived from a voltage source in the range between .9 to 1.8 V that is applied to the V_{LCD} pin. Table 13-20 provides a more detailed description of the power state of the LCD module which depends on the configuration of the VSUPPLY[1:0], HREFSEL, BBYPASS, CPSEL and RVEN bits.

Table 13-20 shows all possible configurations of the LCD Power Supply. All other combinations of the configuration bits above are not permissible LCD power supply modes and should be avoided.

Table 13-20. LCD Power Supply Options

LCD Operational State	LCD Power Supply Configuration	VSUPPLY[1:0]	HREFSEL	BBYPASS	CPSEL	RVEN
V_{LL2} connected to V_{DD} internally for 3 or 5 V glass operation.	For 3 V glass operation V_{DD} must equal 2 V. For 5 V glass operation V_{DD} must equal 3.33 V Charge pump is used to generate V_{LL1} and V_{LL3} V_{LCD} pin is not connected	00	X	0	1	0
V_{LL3} connected to V_{DD} internally for 3 V or 5 V glass operation	For 3 V glass operation V_{DD} must equal 3 V. For 5 V glass operation V_{DD} must equal 5 V. Charge pump is used to generate V_{LL1} and V_{LL2} V_{LCD} pin is not connected	01	X	0	1	0
V_{LCD} * 3/3 connected to V_{LL1} for 3 V glass operation. Unbuffered mode.	For 3 V glass operation V_{LCD} must equal 1 V. V_{LCD} (1 V) is connected to V_{LL1} HREFSEL = 1 to set $V_{LL1} = V_{LCD}$ * 3/3 Charge pump is used to generate V_{LL2}, and V_{LL3}	10	1	1	1	0
V_{LCD} * 2/3 connected to V_{LL1} for 3 V glass operation. Buffered mode.	For 3 V glass operation V_{LCD} must equal 1.5 V. 2/3 V_{LCD} (1 V) is connected to V_{LL1} HREFSEL = 0 to set $V_{LL1} = V_{LCD}$ * 2/3 Charge pump is used to generate V_{LL2}, and V_{LL3}	10	0	0	1	0

Table 13-20. LCD Power Supply Options (continued)

LCD Operational State	LCD Power Supply Configuration	VSUPPLY[1:0]	HREFSEL	BBYPASS	CPSEL	RVEN
V_{LCD} * 3/3 connected to V_{LL1} for 3 V glass operation. Buffered mode.	For 3 V glass operation V_{LCD} must equal 1 V. V_{LCD} (1 V) is connected to V_{LL1} HREFSEL = 1 to set $V_{LL1} = V_{LCD}$ * 3/3 Charge pump is used to generate V_{LL2}, and V_{LL3}	10	1	0	1	0
V_{LCD} connected to V_{LL1} for 5 V glass operation. Unbuffered mode.	For 5 V glass operation V_{LCD} must equal 1.67 V. V_{LCD} (1.67 V) is connected to V_{LL1} HREFSEL = 1 to set $V_{LL1} = V_{LCD}$ * 3/3 Charge pump is used to generate V_{LL2}, and V_{LL3}	10	1	1	1	0
V_{LCD} connected to V_{LL1} for 5 V glass operation. Buffered mode.	For 5 V glass operation V_{LCD} must equal 1.67 V. V_{LCD} (1.67 V) is connected to V_{LL1} HREFSEL = 1 to set $V_{LL1} = V_{LCD}$ * 3/3 Charge pump is used to generate V_{LL2}, and V_{LL3}	10	1	0	1	0
V_{LL3} is driven externally for 3 V LCD Glass operation.	For 3 V glass operation V_{LL3} must equal 3 V. V_{LCD} is not connected Charge pump is used to generate V_{LL1} and V_{LL2}	11	X	0	1	0
V_{LL3} is driven externally for 5 V LCD Glass operation. V_{LL3} must equal V_{DD} This operation is not allowed for 1.8 V to 3.6 V parts	For 5 V glass operation V_{LL3} must equal 5 V. V_{LCD} is not connected Charge pump is used to generate V_{LL1} and V_{LL2}	11	X	0	1	0
V_{LL3} is driven externally for 3 V LCD Glass operation. Resistor Bias Network enabled.	For 3 V glass operation V_{LL3} must equal 3 V. V_{LCD} is not connected. Charge pump is disabled. Resistor Bias network is used to create V_{LL1} and V_{LL2}	11	X	0	0	0

Table 13-20. LCD Power Supply Options (continued)

LCD Operational State	LCD Power Supply Configuration	VSUPPLY[1:0]	HREFSEL	BBYPASS	CPSEL	RVEN
V_{LL3} is driven externally for 5 V LCD Glass operation. Resistor Bias network enabled. V_{LL3} must equal V_{DD} This operation is not allowed for 1.8 V to 3.6 V parts	For 5 V glass operation V_{LL3} must equal 5 V. V_{LCD} is not connected. Charge pump is disabled. Resistor Bias network is used to create V_{LL1} and V_{LL2}.	11	X	0	0	0
V_{IREG} is connected to V_{LL1} for 5 V glass operation. The HREFSEL bit is used to select 1.0 or 1.67 V range for V_{IREG}	For 5 V glass operation HREFSEL = 1, V_{IREG} = 1.67 V. V_{IREG} is connected V_{LL1} internally. V_{LCD} is not connected. Charge pump is used to generate V_{LL2} and V_{LL3} .	11	1	0	1	1
V_{IREG} is connected to V_{LL1} for 3 V glass operation. The HREFSEL bit is used to select 1.0 or 1.67 V range for V_{IREG}	For 3 V glass operation HREFSEL= 0, V_{IREG} = 1 V. V_{IREG} is connected V_{LL1} internally. V_{LCD} is not connected. Charge pump is used to generate V_{LL2} and V_{LL3} .	11	0	0	1	1

13.4.4.2.1 VSUPPLY[1:0] = 10

If VSUPPLY[1:0] = 10, the LCD power supply is configured to be internally derived from the V_{LCD} pin.

External V_{LCD} supply

When VSUPPLY[1:0] = 10, only the powersw3 signal is asserted, and the LCD module is configured to be powered via an input. (V_{LCD} in the range from .9 V to 1.8 V).{statement} Figure 13-16 shows that V_{LCD} can be an input to the voltage divider block and is related to $V_{LL1.}$ The voltage-divider block uses the states of HREFSEL, BBYPASS, and powersw3 to derive a state for V_{LL1}.

The output of the voltage divider block is be connected to $V_{LL1.}$ V_{LL1} is connected to the internal charge pump. Using the charge pump, the value of V_{LL1} is tripled and output as $V_{LL3.}$ V_{LL3}, an LCD bias voltage, is equal to the voltage required to energize the LCD panel, $V_{LCDON.}$ For 3-V LCD glass, V_{LL3} should be approximately 3 V; while for 5-V LCD glass, V_{LL3} should be approximately 5 V.

Depending on the HREFSEL bit configuration, V_{LL3} will be equal to $3*V_{LCD}$ or $3*(2/3*V_{LCD})$ (see Section 13.4.4, "LCD Charge Pump, Voltage Divider, and Power Supply Operation). Table 13-21 shows the selected V_{LL1} and V_{LL3} values based on the input value of V_{LCD}.

Table 13-21. V_{LL1} Typical Values

V_{LCD}	HREFSEL = 0 Voltage Doubler		HREFSEL = 1 Voltage Tripler	
	$V_{LL1} = V_{ref}$	$V_{LL3} = 3 \times V_{ref}$	$V_{LL1} = V_{ref}$	$V_{LL3} = 3 \times V_{ref}$
1.4 V	(2/3) × 1.4 V	2.8 V	1.4 V	4.2 V
1.5 V	(2/3) × 1.5 V	3.0 V	1.5 V	4.5 V
1.7 V	(2/3) × 1.7 V	3.4 V	1.7 V	5.1 V
1.8 V	(2/3) × 1.8 V	3.6 V	1.8 V	5.4 V

In addition to V_{LL1} and V_{LL3}, V_{LL2} is also generated internally when the charge pump is enabled (CPSEL = 1). For a typical LCD panel, the bias voltages in 1/3 bias mode are:

- $V_3 = V_{LL3} = V_{LCDON} = 3 \times V_{ref}$
- $V_2 = V_{LL2} = 2 \times V_{ref}$
- $V_1 = V_{LL1} = V_{ref}$
- $V_0 = V_{SS}$

NOTE

V_{LCDON} is the LCD panel driving voltage required to turn on an LCD segment. Since V_{LL3} and V_{LCDON} are equivalent, V_{LL3} should be configured so that it is 3 V or 5 V, depending on the LCD panel specification.

13.4.4.2.2 LCD External Power Supply, VSUPPLY[1:0] = 11

When VSUPPLY[1:0] = 11, powersw1, powersw2, and powersw3 are deasserted. With powersw3 deasserted, the buffer is disabled ((~BBYPASS & pwersw3) = 0), so V_{LCD} will be configured in a tri-state condition. V_{DD} is not available to power the LCD module internally, so the LCD module requires an external power source for V_{LL1}, V_{LL2}, and V_{LL3} when the charge pump is disabled.

If the charge pump is enabled, external power must be applied to V_{LL3}. With this configuration, the charge pump will generate the other LCD bias voltages V_{LL1} and V_{LL2}.

Internal V_{IREG}

If the charge pump is enabled the internal regulated voltage, V_{IREG}, can be used as an input to generate the LCD bias voltages. In this state external voltage source should not be connected to V_{LL1}. V_{IREG} is controlled by the LCD regulated voltage control (LCDRVC) register. The figure above, Figure 13-16, shows that V_{IREG} is connected to V_{LL1} when the RVEN bit is set.{statement}

V_{LL1} is connected to the internal charge pump. Using the charge pump, the value of V_{LL1} is tripled and output as V_{LL3}. V_{LL3}, an LCD bias voltage, is equal to the voltage required to energize the LCD panel,

V_{LCDON}. For 3-V LCD glass, V_{LL3} should be approximately 3 V; while for 5-V LCD glass, V_{LL3} should be approximately 5 V.

The HREFSEL bit in the LCDSUPPLY register is used to set V_{IREG} to approximately 1.0 V or 1.67 V as shown in Table 13-22. The LCDRVC contains trim bits that can be used to make changes to the regulated voltage. The trim can be used to increase or decrease the regulated voltage by 1.5% for each count. A total of ±12% of change can be done to the regulated voltage.

Table 13-22 shows the selected LCD bias voltages V_{LL1}, V_{LL2}, and V_{LL3} values based on the value of V_{IREG}

Table 13-22. Bias Voltage Typical Values

HREFSELL	V_{IREG}	$V_{LL1} = V_{ref}$	$V_{LL2} = 2 \times V_{ref}$	$V_{LL3} = 3 \times V_{ref}$
HREFSEL = 0	1.0 V	1.0 V	2.0 V	3.0 V
HREFSEL =1	1.67 V	1.67 V	3.33 V	5.0 V

13.4.4.2.3 LCD Internal Power Supply, VSUPPLY[1:0] = 00 or 01

V_{DD} is used as the LCD module power supply when VSUPPLY[1:0] = 00 or 11 (Table 13-23**).** When powering the LCD module using V_{DD}, the charge pump must be enabled (CPSEL = 1). Table 13-23 provides recommendations regarding configuration of the VSUPPLY[1:0] bit field when using both 3-V and 5-V LCD panels.

Table 13-23. V_{DD} Switch Option

VSUPPLY[1:0]	V_{DD} Switch Option	Recommend Use for 3-V LCD Panels	Recommend Use for 5-V LCD Panels
00	V_{LL2} is generated from V_{DD}	• V_{LL1} = 1 V • **$V_{DD} = V_{LL2}$ = 2 V** • V_{LL3} = 3 V	• V_{LL1} = 1.67 V • **$V_{DD} = V_{LL2}$ = 3.3 V** • V_{LL3} = 5 V
01	V_{LL3} is generated from V_{DD}	• V_{LL1} = 1 V • V_{LL2} = 2 V • **$V_{DD} = V_{LL3}$ = 3 V**	• V_{LL1} = 1.67 V • V_{LL2} = 3.33 V • **$V_{DD} = V_{LL3}$ = 5 V**

13.4.5 Resets

During a reset, the LCD module system is configured in the default mode. The default mode includes the following settings:

- LCDEN is cleared, thereby forcing all frontplane and backplane driver outputs to the high impedance state.
- 1/4 duty
- 1/3 bias
- LCLK[2:0], VSUPPLY[1:0], BBYPASS, CPSEL, RVEN, and BRATE[2:0] revert to their reset values

13.4.6 Interrupts

When an LCD module frame-frequency interrupt event occurs, the LCDIF bit in the LCDS register is asserted. The LCDIF bit remains asserted until software clears the LCD-module-frame-frequency interrupt. The interrupt can be cleared by software writing a 1 to the LCDIF bit.

If both the LCDIF bit in the LCDS register and the LCDIEN bit in the LCDC1 register are set, an LCD interrupt signal asserts.

13.5 Initialization Section

This section provides a recommended initialization sequence for the LCD module and also includes initialization examples for several LCD application scenarios.

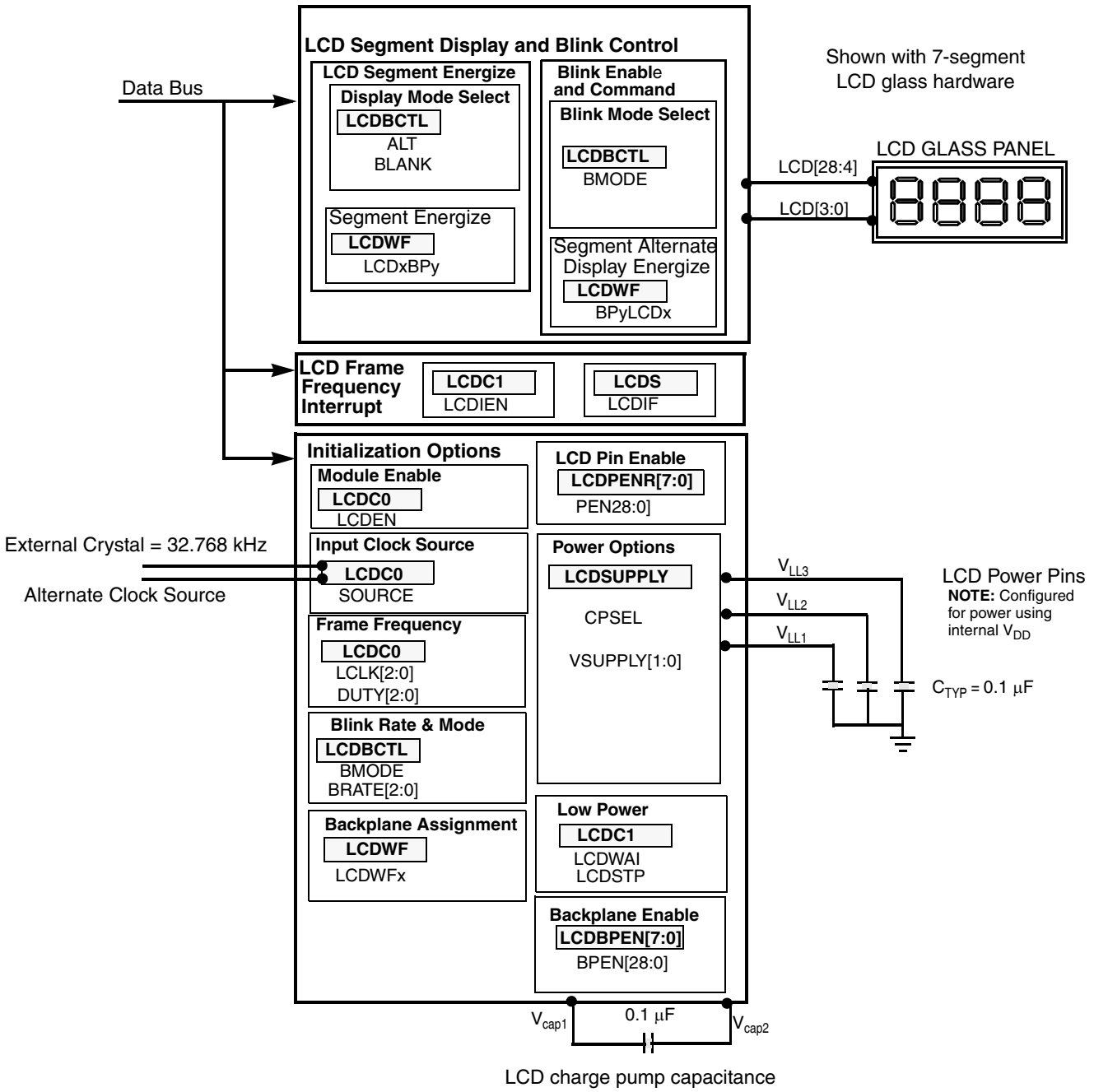
13.5.1 Initialization Sequence

The below list provides a recommended initialization sequence for the LCD module.

You must write to all LCDPEN, LCDBPEN, and LCDWF registers to initialize their values after a reset.

1. LCDC0 register
   a) Configure LCD clock source (SOURCE bit).
2. LCDRVC register (If the application uses the internally regulated voltage)
   a) Select 1.0 V or 1.67 V for 3 or 5 V glass (HREFSEL).
   b) Enable regulated voltage (RVEN).
   c) Trim the regulated voltage (RVTRIM).
3. LCDSUPPLY register
   a) Enable charge pump (CPSEL bit).
   b) Configure LCD module for doubler or tripler mode (HREFSEL bit).
   c) Configure charge pump clock (LADJ[1:0]).
   d) Configure op amp switch (BBYPASS bit).
   e) Configure LCD power supply (VSUPPLY[1:0]).
4. LCDC1 register
   a) Configure LCD frame frequency interrupt (LCDIEN bit).
   b) Configure LCD behavior in low power mode (LCDWAI and LCDSTP bits).
5. LCDC0 register
   a) Configure LCD duty cycle (DUTY[2:0]).
   b) Select and configure LCD frame frequency (LCLK[2:0]).
6. LCDBCTL register
   a) Configure display mode (ALT and BLANK bits).
   b) Configure blink mode (BMODE).
   c) Configure blink frequency (BRATE[2:0]).

7. LCDPEN[7:0] register
   a) Enable LCD module pins (PEN[63:0] bits).
8. LCDBPEN[7:0]
   a) Enable LCD pins to operate as an LCD backplane (BPEN[63:0]).
9. LCDC0 register
   a) Enable LCD module (LCDEN bit).


LCD Segment Display and Blink Control
LCD Segment Energize
Display Mode Select
LCDBCTL
ALT
BLANK
Segment Energize
LCDWF
LCDxBPy
Blink Enable and Command
Blink Mode Select
LCDBCTL
BMOD
Segment Alternate Display Energize
LCDWF
BPyLCDx
Shown with 7-segment LCD glass hardware
LCD GLASS PANEL
LCD[28:4]
LCD[3:0]
Data Bus
LCD Frame Frequency Interrupt
LCDC1
LCDIEN
LCDS
LCDIF
Initialization Options
Module Enable
LCDC0
LCDEN
LCD Pin Enable
LCDPENR[7:0]
PEN28:0]
External Crystal = 32.768 kHz
Alternate Clock Source
Input Clock Source
LCDC0
SOURCE
Power Options
LCDSUPPLY
CPSEL
VSUPPLY[1:0]
VLL3
VLL2
VLL1
LCD Power Pins
NOTE: Configured for power using internal VDD
CTYP = 0.1 µF
Frame Frequency
LCDC0
LCLK[2:0]
DUTY[2:0]
Blink Rate & Mode
LCDBCTL
BMODE
BRATE[2:0]
Backplane Assignment
LCDWF
LCDWFx
Low Power
LCDC1
LCDWAI
LCDSTP
Backplane Enable
LCDBPEN[7:0]
BPEN[28:0]
Vcap1
0.1 µF
Vcap2
LCD charge pump capacitance

13.5.2 Initialization Examples

This section provides initialization information for LCD configuration. Each example details the register and bit-field values required to achieve the appropriate LCD configuration for a given LCD application scenario. The table below lists each example and the setup requirements.

Table 13-24. LCD Application Scenario

Example	Operating Voltage, V_{DD}	LCD Clock Source	LCD Glass Operating Voltage	Required LCD segments	LCD Frame Rate	Blinking Mode/Rate	Behavior in WAIT/STOP modes	LCD Power Input
1	1.8 V	External 32.768 kHz	3 V	128	30 Hz	None	WAIT: on STOP: on	Power via V_{LCD}
2	3.6 V	Internal 39.063 kHz	3 V	100	50 Hz	Alternate 0.5 Hz	WAIT: on STOP: off	Power via V_{DD}
3	3.0 V	External 32.768 kHz	5 V	168	30 Hz	Blank 2.0 Hz	WAIT: off STOP: off	Power via V_{IREG}

These examples illustrate the flexibility of the LCD module to be configured to meet a wide range of application requirements including:

- clock inputs/sources
- LCD power supply
- LCD glass operating voltage
- LCD segment count
- varied blink modes/frequencies
- LCD frame rate

13.5.2.1 Initialization Example 1

Table 13-25. LCD Setup Requirements for Example 1

Example	Operating Voltage, V_{DD}	LCD Clock Source	LCD Glass Operating Voltage	Required LCD segments	LCD Frame Rate	Blinking Mode/Rate	Behavior in STOP and WAIT modes	LCD Power Input
1	1.8-V	External 32.768 kHz	3-V	128	30 Hz	None	WAIT: on STOP: on	Power via V_{LCD}

The table below lists the setup values required to initialize the LCD as specified by Example 1:

Table 13-26. Initialization Register Values for Example 1

Register	bit or bit field	Binary Value	Comment
LCDSUPPLY 1000-010	CPSEL	1	Enable charge pump
	HREFSEL	0	For 3-V LCD glass, select doubler mode; Doubler mode = 0; Tripler mode = 1
	LADJ[1:0]	00	Configure LCD charge pump clock source
	BBYPASS	0	Buffer Bypass; Buffer mode = 0; Unbuffered mode = 1
	VSUPPLY[1:0]	10	When VSUPPLY[1:0] = 10, the LCD must be externally powered via V_{LCD} (see Table 13-20). For 3-V glass, the nominal value of V_{LCD} should be 1.5-V.
LCDC1 0------00	LCDIEN	0	LCD frame interrupts disabled
	LCDWAI	0	LCD is “on” in WAIT mode
	LCDSTP	0	LCD is “on” in STOP mode
LCDC0 00101111	LCDEN	0	Initialization is done before initializing the LCD module
	SOURCE	0	Selects the external clock reference as the LCD clock input (OSCOUT)
	LCLK[2:0]	101	For 1/8 duty cycle, select closest value to the desired 30 Hz LCD frame frequency (see Table 13-13)
	DUTY[2:0]	111	For 128 segments (8x16), select 1/8 duty cycle
LCDBCTL 0XX-XXXX	BLINK	0	No blinking
	ALT	X	Alternate bit is configured during LCD operation
	BLANK	X	Blank bit is configured during LCD operation
	BMODE	X	N/A; Blink Blank = 0; Blink Alternate = 1
	BRATE[2:0]	XXX	N/A
LCDPEN[7:0]	LCDPEN0 LCDPEN1 LCDPEN2 LCDPEN3	11111111 11111111 11111111 00000000	Only 24 LCD pins need to be enabled. **Note:** Any of the 63 LCD pins can be used, this allows flexibility in the hardware design.
LCDBPEN[7:0]	LCDBPEN0 LCDBPEN1 LCDBPEN2 LCDBPEN3	11111111 00000000 00000000 00000000	Eight backplane pins needed. **Note:** Any enabled LCD pin can be enabled to operate as a backplane.
LCDWF[63:0]	LCDWF0 LCDWF1 LCDWF2 LCDWF3 LCDWF4 LCDWF5 LCDWF6 LCDWF7	00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000	Configure which phase the eight backplane pins will be active in. This configuration sets LCD[0] to be active in Phase A, LCD[1] to be active in Phase B...etc **Note:** Any backplane pin can be active in any phase.

13.5.2.2 Initialization Example 2

Example 2 LCD setup requirements are reiterated in the table below:

Example	Operating Voltage, V_{DD}	LCD Clock Source	LCD Glass Operating Voltage	Required LCD segments	LCD Frame Rate	Blinking Mode/Rate	Behavior in STOP3 and WAIT modes	LCD Power Input
2	3.6 V	Internal 39.063 kHz	3 V	100	50 Hz	Alternate 0.5 Hz	WAIT: on STOP: off	Power via V_{DD}

The table below lists the required setup values required to initialize the LCD as specified by Example 2:

Table 13-27. Initialization Register Values for Example 2

Register	bit or bit field	Binary Value	Comment
LCDSUPPLY 1X00-001	CPSEL	1	Enable charge pump
	HREFSEL	X	Don't Care since power is from internal V_{DD}; Doubler mode = 0; Tripler mode = 1
	LADJ[1:0]	00	Configure LCD charge pump clock source
	BBYPASS	0	Buffer Bypass; Buffer mode = 0; Unbuffered mode = 1
	VSUPPLY[1:0]	01	Generate V_{LL3} from V_{DD} (See Table 13-20)
LCDC1 0-----01	LCDIEN	0	LCD Frame Interrupts disabled
	LCDWAI	0	LCD is "on" in WAIT mode
	LCDSTP	1	LCD is "off" in STOP mode
LCDC0 01010011	LCDEN	0	Initialization is done before initializing the LCD module
	SOURCE	1	Selects the alternate-clock reference as the LCD clock input (ALTCLK) This clock source is configured by the ICS TRIM bits to be 39.063Khz.
	LCLK[2:0]	010	For 1/4 duty cycle, select closest value to the desired 50 Hz LCD frame frequency (Table 13-14)
	DUTY[2:0]	011	For 100 segments (4x25), select 1/4 duty cycle
LCDBCTL 1XX-1100	BLINK	1	Blinking is turned on or off during LCD operation
	ALT	X	Alternate bit is configured during LCD operation
	BLANK	X	Blank bit is configured during LCD operation
	BMODE	1	Blink Alternate = 1
	BRATE[2:0]	100	Select.5 Hz blink frequency using Table 13-18
LCDPEN[7:0]	LCDPEN0 LCDPEN1 LCDPEN2 LCDPEN3	11111111 11111111 11111111 00011111	29 LCD pins need to be enabled.

Table 13-27. Initialization Register Values for Example 2 (continued)

Register	bit or bit field	Binary Value	Comment
LCDBPEN[7:0]	LCDBPEN0 LCDBPEN1 LCDBPEN2 LCDBPEN3	00001111 00000000 00000000 00000000	Four backplane pins needed. **Note:** Any enabled LCD pin can be enabled to operate as a backplane.
LCDWF[63:0]	LCDWF0 LCDWF1 LCDWF2 LCDWF3	00000001 00000010 00000100 00001000	Configure which phase the four backplane pins will be active in. This configuration sets LCD[0] to be active in Phase A, LCD[1] to be active in Phase B etc. **Note:** Any backplane pin can be active in any of the phases.

13.5.2.3 Initialization Example 3

Example 3 LCD setup requirements are reiterated in the table below:

Example	Operating Voltage, V_{DD}	LCD Clock Source	LCD Glass Operating Voltage	Required LCD segments	LCD Frame Rate	Blinking Mode/Rate	Behavior in STOP3 and WAIT modes	LCD Power Input
3	3.0 V	External 32.768 kHz	5 V	168	30 Hz	Blank 2.0 Hz	WAIT: off STOP: off	Power via V_{IREG}

The table below lists the required setup values required to initialize the LCD as specified by Example 3:

Table 13-28. Initialization Register Values for Example 3

Register	bit or bit field	Binary Value	Comment
LCDRVC 1---XXXX	RVEN	1	Enable V_{IREG} so that it can be used to supply the LCD
	RVTRIM	XXXX	Trim value is determined by characterization
LCDSUPPLY 1100-000	CPSEL	1	Enable charge pump
	HREFSEL	1	For 5V glass must enable 1.67V
	LADJ[1:0]	00	Configure LCD charge pump clock source
	BBYPASS	0	Buffer Bypass; Buffer mode = 0; Unbuffered mode = 1
	VSUPPLY[1:0]	11	When VSUPPLY[1:0] = 11, the LCD can be powered via V_{IREG} (see Table 13-20). For 5-V glass, the nominal value of V_{IREG} should be 1.67 V
LCDC1 0-----11	LCDIEN	0	LCD Frame Interrupts disabled
	LCDWAI	1	LCD is “off” in WAIT mode
	LCDSTP	1	LCD is “off” in STOP mode

Table 13-28. Initialization Register Values for Example 3 (continued)

Register	bit or bit field	Binary Value	Comment
LCDC0 00101111	LCDEN	0	Initialization is done before initializing the LCD module
	SOURCE	0	Selects OSCOUT as the LCD clock source (32.768 Khz crystal)
	LCLK[2:0]	101	For 1/8 duty cycle, select closest value to the desired 30 Hz LCD frame frequency (see Table 13-13)
	DUTY[2:0]	111	For 168 segments (8x21), select 1/8 duty cycle
LCDBCTL XXX-0010	BLINK	X	Blinking is turned on or off during LCD operation
	ALT	X	Alternate bit is configured during LCD operation
	BLANK	X	Blank bit is configured during LCD operation
	BMODE	0	Blink to a blank mode
	BRATE[2:0]	010	Select 2 Hz blink frequency using Table 13-18
LCDPEN[7:0]	LCDPEN0 LCDPEN1 LCDPEN2 LCDPEN3	11111111 11111111 11111111 00011111	29 LCD pins need to be enabled.
LCDBPEN[7:0]	LCDBPEN0 LCDBPEN1 LCDBPEN2 LCDBPEN3	11111111 00000000 00000000 00000000	Eight backplane pins needed. **Note:** Any enabled LCD pin can be enabled to operate as a backplane
LCDWF[63:0]	LCDWF0 LCDWF1 LCDWF2 LCDWF3 LCDWF4 LCDWF5 LCDWF6 LCDWF7	00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000	Configure which phase the eight backplane pins will be active in. This configuration sets LCD[0] to be active in Phase A, LCD[1] to be active in Phase B. . . etc. This configuration sets LCD pins 0-7 to represent backplane 1-8. **Note:** Any backplane pin can be active in any phase

13.6 Application Information

Figure 13-17 is a programmer's model of the LCD module. The programmer's model groups the LCD module register bit and bit field into functional groups. The model is a high-level illustration of the LCD module showing the module's functional hierarchy including initialization and runtime control.

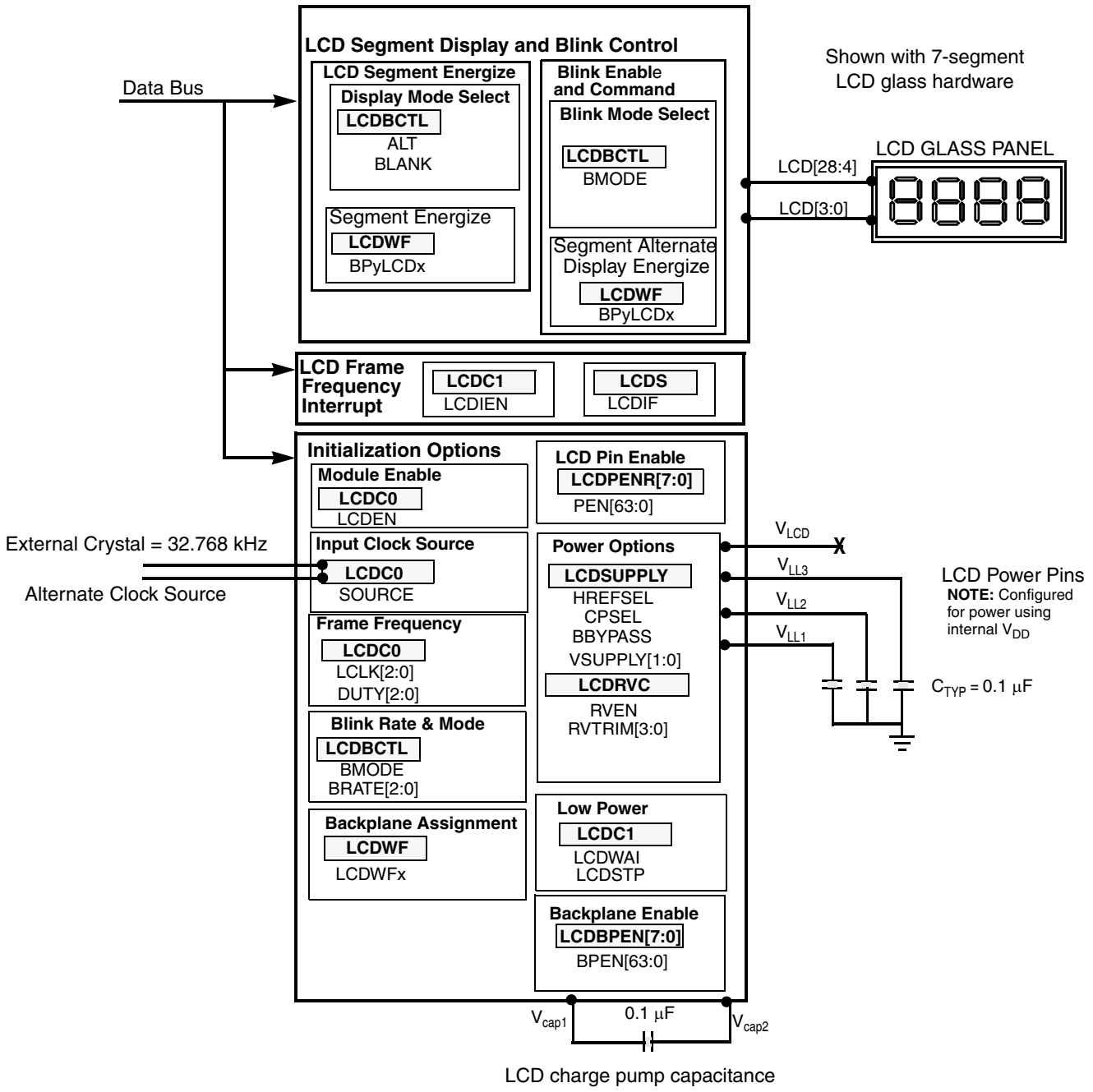
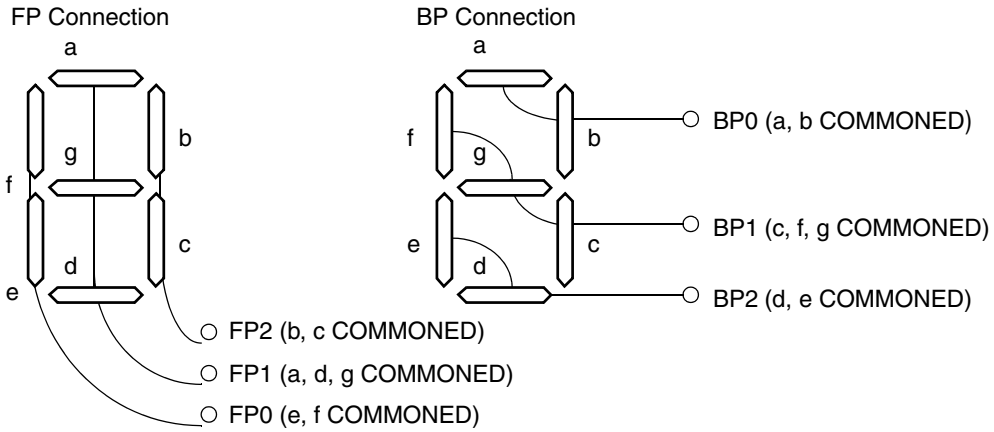


Figure 13-17. LCD Programmer's Model Diagram

13.6.1 LCD Seven Segment Example Description

A description of the connection between the LCD module and a seven segment LCD character is illustrated below to provide a basic example for a 1/3 duty cycle LCD implementation. The example uses three backplane pins (LCD[3], LCD[4] and LCD[5] and 3 frontplane pins (LCD[0], LCD[1], and LCD[2]).

LCDWF contents and output waveforms are also shown. Output waveforms are illustrated in Figure 13-18 and Figure 13-19.



The above segment assignments are provided by the specification for the LCD glass for this example.
For this LCD Module any of the LCD pins can be configured to be Frontplane 0-2 or Backplane 0-2.
For this example we will set LCD[0] as FP0, LCD[1] as FP1, and LCD[2] as FP2.
For this example we will set LCD[3] as BP0, LCD[4] as BP1 and LCD[5] as BP2.

Backplane assignment is done in the LCDWF register as shown below:

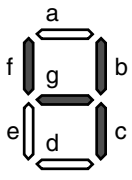
LCDWF3	0	0	0	0	0	0	0	1
LCDWF4	0	0	0	0	0	0	1	0
LCDWF5	0	0	0	0	0	1	0	0

With the above conditions the segment assignment is shown below:

LCDWF0	–	–	–	–	–	e	f	–
LCDWF1	–	–	–	–	–	d	g	a
LCDWF2	–	–	–	–	–	–	c	b

To display the character “4”: LCDWF0 = XXXXX01X, LCDWF1 = XXXXX010, LCDWF2 =XXXXXX11

LCDWF0	X	X	X	X	X	0	1	X
LCDWF1	X	X	X	X	X	0	1	0
LCDWF2	X	X	X	X	X	X	1	1



X = don’t care

Figure 13-18. Waveform Output from LCDWF Registers

13.6.1.1 LCD Module Waveforms

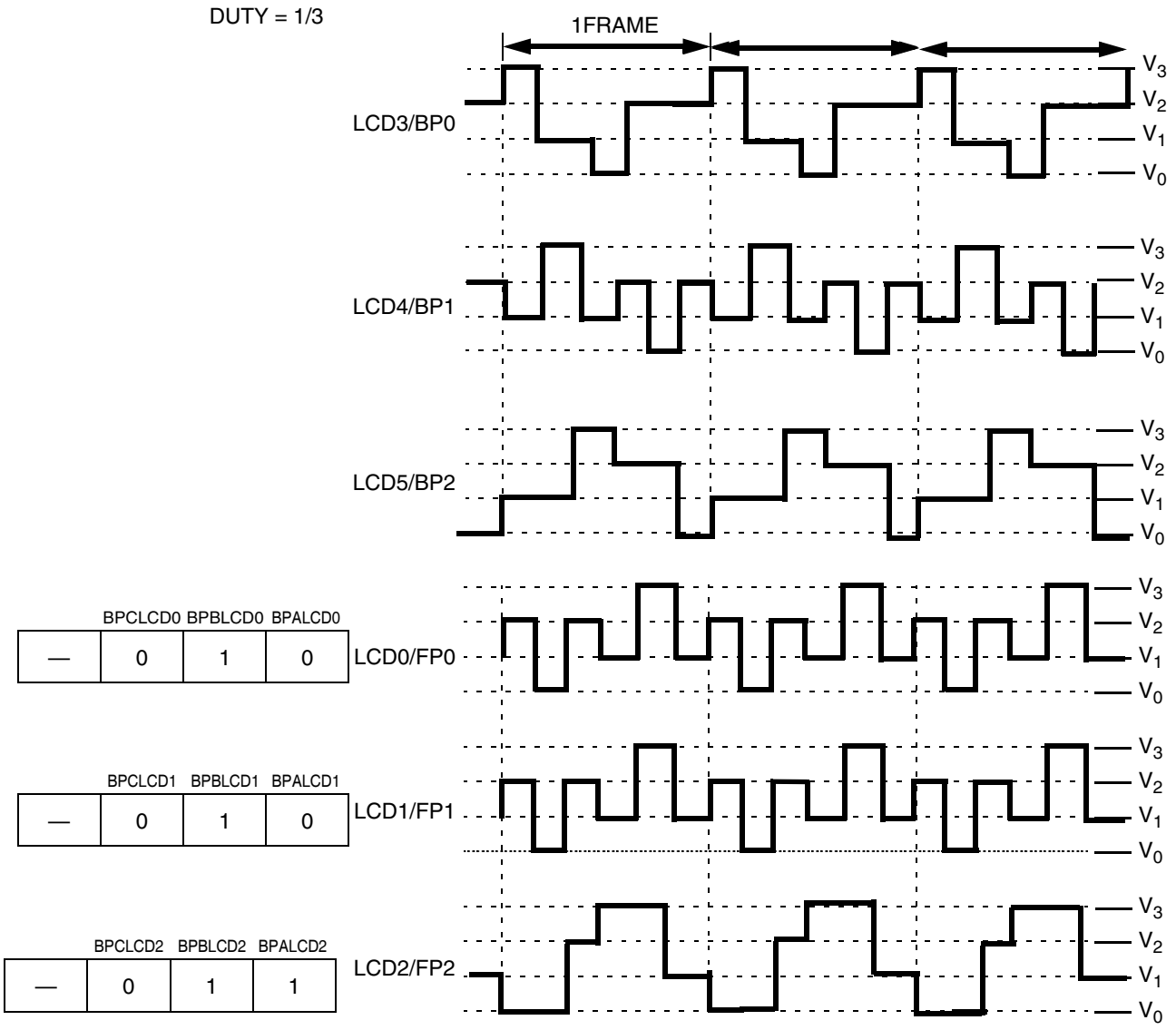


Figure 13-19. LCD Waveforms

13.6.1.2 Segment On Driving Waveform

The voltage waveform across the “f” segment of the LCD (between LCD[4]/BP1 and LCD[0]/FP0) is illustrated in Figure 13-20. As shown in the waveform, the voltage level reaches the value V_3 therefore the segment will be on.

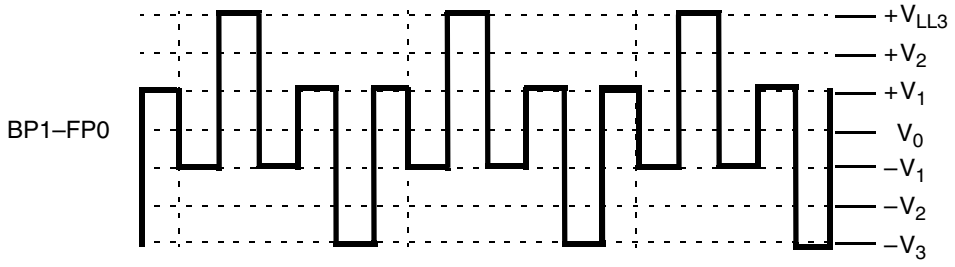


Figure 13-20. “f” Segment Voltage Waveform

13.6.1.3 Segment Off Driving Waveform

The voltage waveform across the “e” segment of the LCD (between LCD[5]/BP2 and LCD[0]/FP0) is illustrated in Figure 13-21. As shown in the waveform, the voltage does not reach the voltage V_3 threshold therefore the segment will be off.

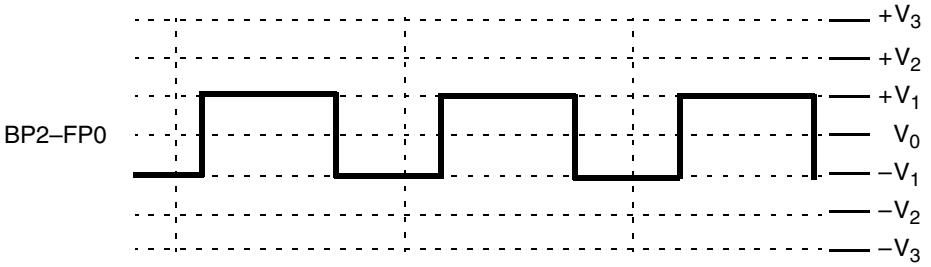


Figure 13-21. “e” Segment Voltage Waveform

13.6.2 LCD Contrast Control

Contrast control for the LCD module is achieved when the LCD power supply is adjusted above and below the LCD threshold voltage. The LCD threshold voltage is the nominal voltage required to energize the LCD segments. For 3-V LCD glass, the LCD threshold voltage is 3 V; while for 5-V LCD glass it is 5 V. By increasing the value of the LCD voltage, the energized segments on the LCD glass will become more opaque. Decreasing the value of the LCD voltage makes the energized segments on the LCD glass become more transparent. The LCD power supply can be adjusted to facilitate contrast control by using external components like a variable resistor. Figure 13-22 shows two circuits that can be used to implement contrast control.

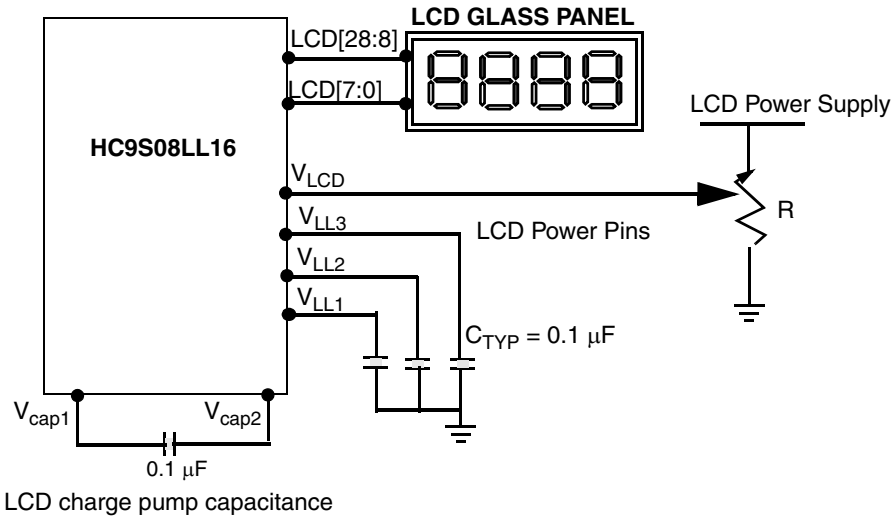
Additionally, if the internally regulated voltage source (V_{IREG}) is used to power the LCD glass, contrast control can be achieved by software alone. Using the RVTRIM[3:0] bits, the V_{IREG} can be increased or decreased by steps of 1.5% for each count in the trim register. Just as with an external circuit, increasing the value of the LCD voltage will cause the energized segments on the LCD glass to become more opaque.

Decreasing the value of the LCD voltage makes the energized segments on the LCD glass more transparent.

NOTE:
Contrast control configuration when LCD is powered using external V_{LCD}

This is the recommended configuration for contrast control.

V_{LCD} specified between 1.4 and 1.8 V.



NOTE:
Contrast control configuration when LCD is powered using internal V_{DD}

V_{DD} is specified between 1.8 and 3.6 V.

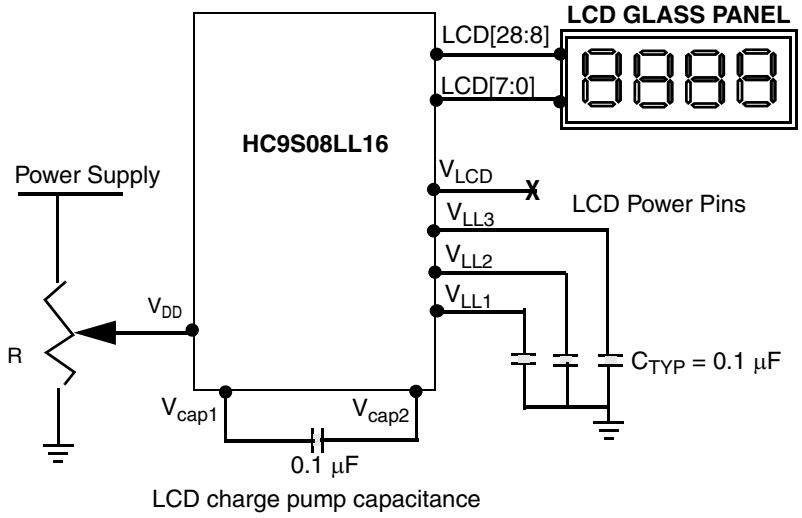


Figure 13-22. Power Connections for Contrast Control

13.6.3 Stop Mode Recovery

When the MCU recovers from stop2 mode a reset sequence is initiated. All Control Registers should be re-written before the stop2 recovery acknowledge bit is set. The Registers LCDBPEN, LCDPEN and the LCDFWF retain their values upon stop2 recovery and do not need to be re-written. For more information on how to perform stop recovery please refer to AN2493.

Chapter 14
Serial Communications Interface (S08SCIV4)

14.1 Introduction

Figure 14-1 shows the MC9S08LH64 series block diagram with the SCI highlighted.

14.1.1 SCI Clock Gating

The bus clock to the SCI can be gated on and off using the SCI bit in SCGC1. This bit is set after any reset, which enables the bus clock to this module. To conserve power, the SCI bit can be cleared to disable the clock to this module when not in use. See Section 5.7, “Peripheral Clock Gating,” for details.

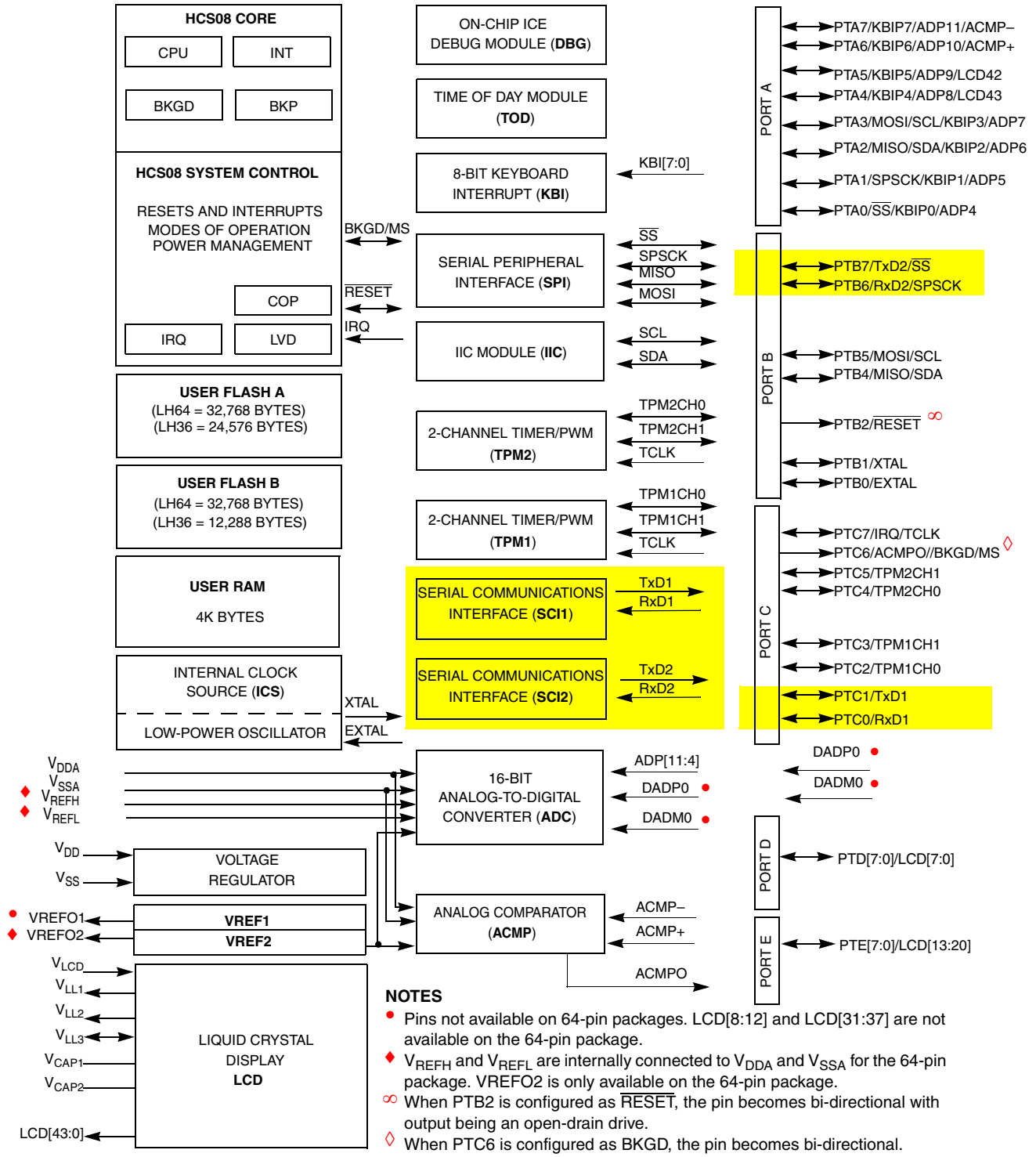


Figure 14-1. MC9S08LH64 Series Block Diagram Highlighting SCI Blocks and Pins

14.1.2 Features

Features of SCI module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Double-buffered transmitter and receiver with separate enables
- Programmable baud rates (13-bit modulo divider)
- Interrupt-driven or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
- Hardware parity generation and checking
- Programmable 8-bit or 9-bit character length
- Receiver wakeup by idle-line or address-mark
- Optional 13-bit break character generation / 11-bit break character detection
- Selectable transmitter output polarity

14.1.3 Modes of Operation

See Section 14.3, “Functional Description,” for details concerning SCI operation in these modes:

- 8- and 9-bit data modes
- Stop mode operation
- Loop mode
- Single-wire mode

14.1.4 Block Diagram

Figure 14-2 shows the transmitter portion of the SCI.

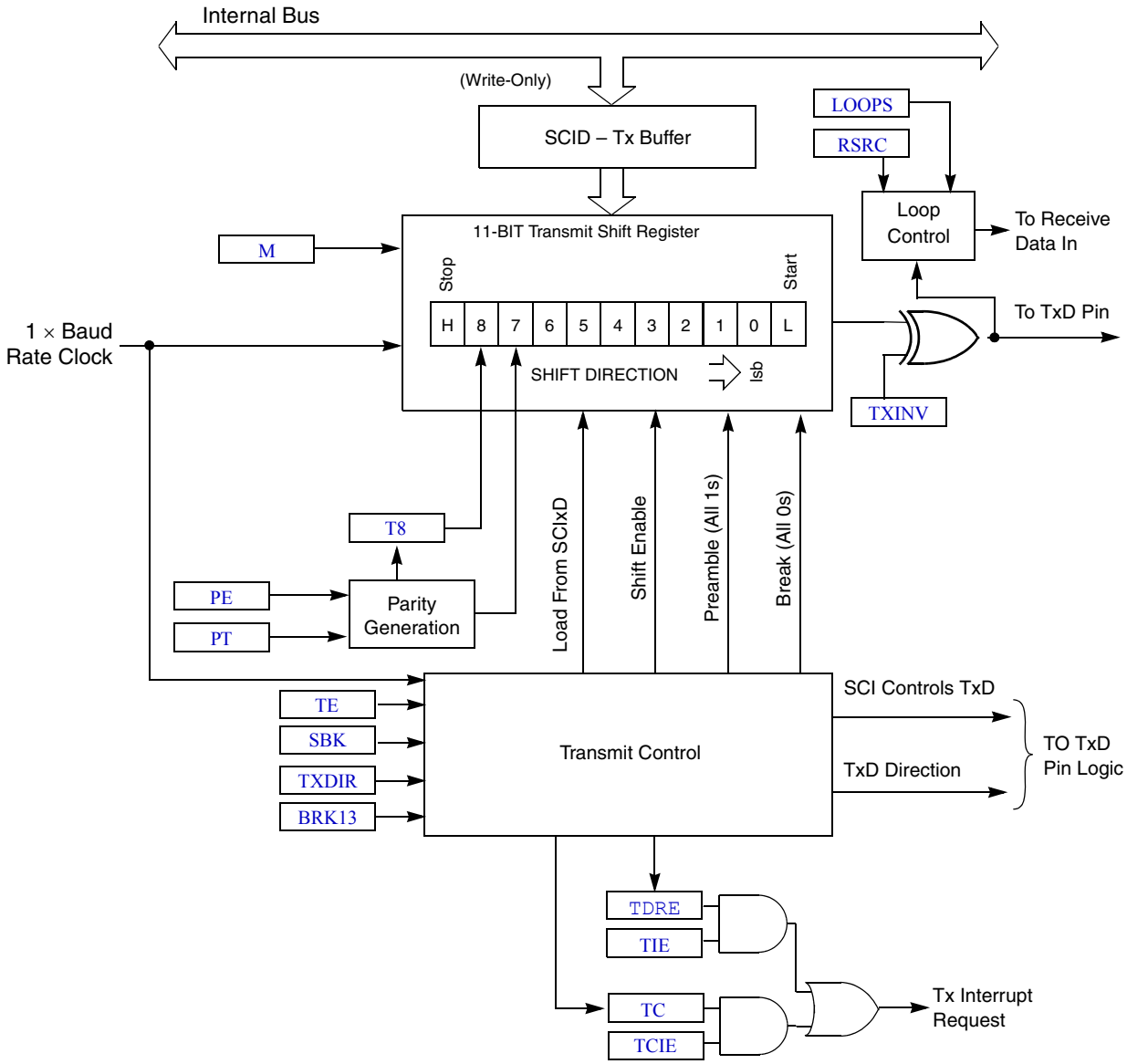


Figure 14-2. SCI Transmitter Block Diagram

Figure 14-3 shows the receiver portion of the SCI.

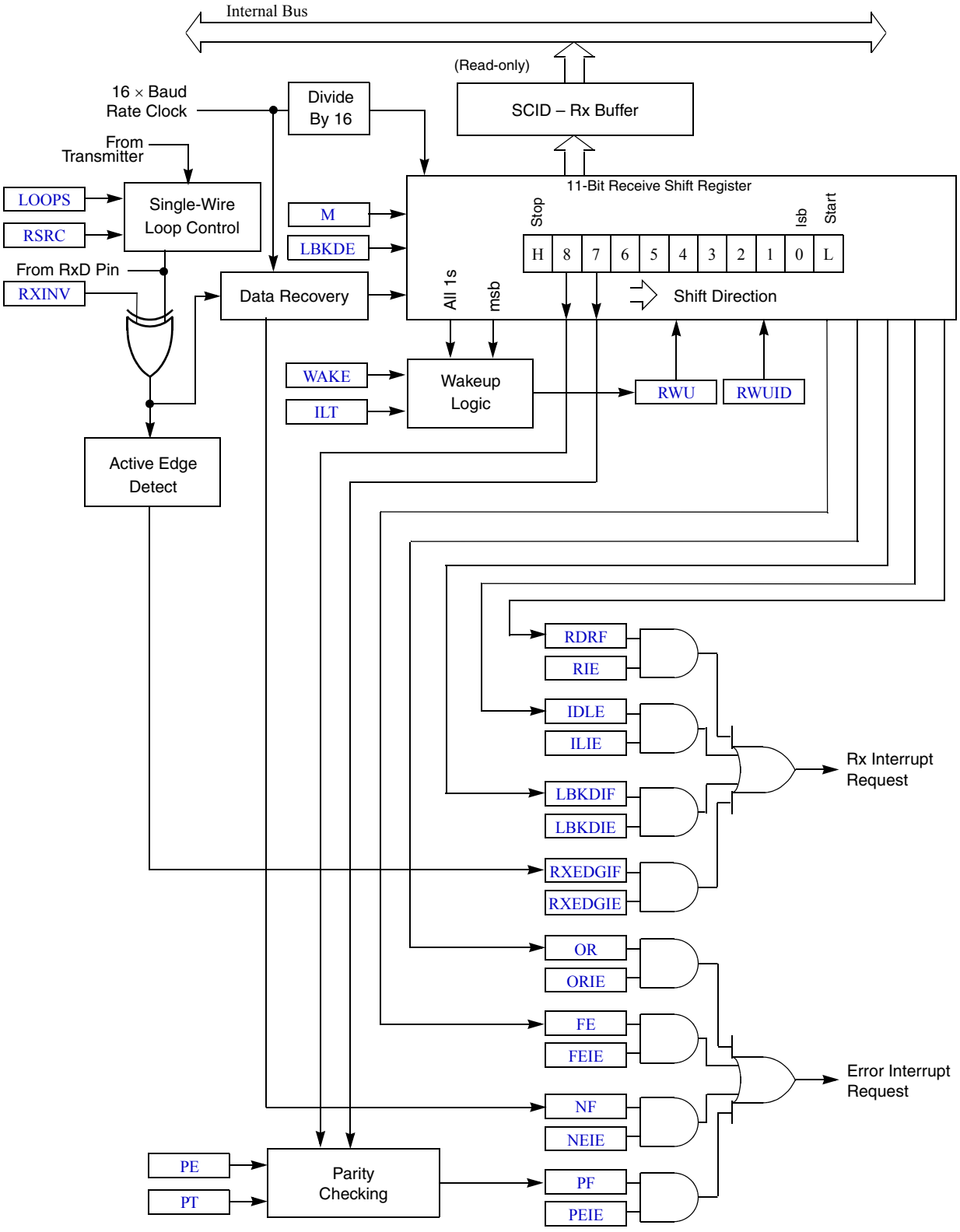


Figure 14-3. SCI Receiver Block Diagram

14.2 Register Definition

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

Refer to the direct-page register summary in Chapter 4 , “Memory,” or the absolute address assignments for all SCI registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

14.2.1 SCI Baud Rate Registers (SCIxBDH, SCIxBDL)

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

SCIxBDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled (RE or TE bits in SCIxC2 are written to 1).

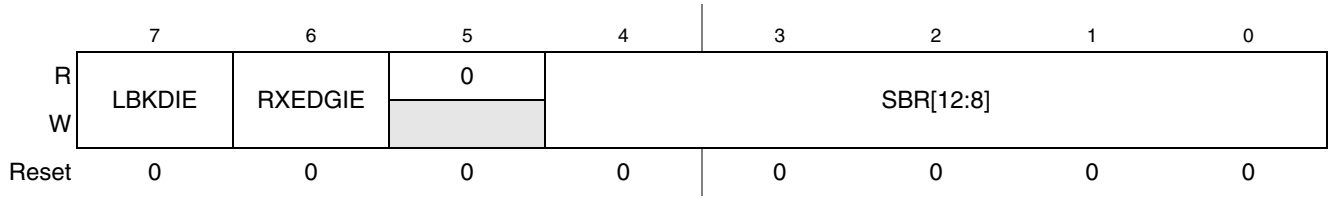


Figure 14-4. SCI Baud Rate Register (SCIxBDH)

Table 14-1. SCIxBDH Field Descriptions

Field	Description
7 LBKDIE	LIN Break Detect Interrupt Enable (for LBKDIF) 0 Hardware interrupts from LBKDIF disabled (use polling). 1 Hardware interrupt requested when LBKDIF flag is 1.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable (for RXEDGIF) 0 Hardware interrupts from RXEDGIF disabled (use polling). 1 Hardware interrupt requested when RXEDGIF flag is 1.
4–0 SBR[12:8]	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR is cleared, the SCI baud rate generator is disabled to reduce supply current. When BR is 1 – 8191, the SCI baud rate equals SCI module clock/(16 × BR). See also BR bits in Table 14-2.

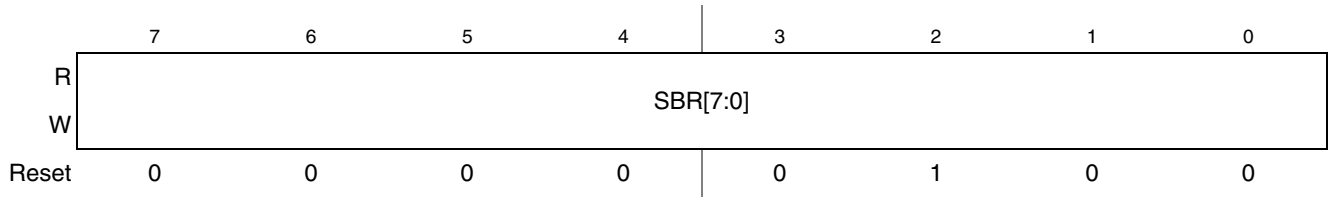


Figure 14-5. SCI Baud Rate Register (SCIxBDL)

Table 14-2. SCIxBDL Field Descriptions

Field	Description
7–0 SBR[7:0]	Baud Rate Modulo Divisor. These 13 bits in SBR[12:0] are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR is cleared, the SCI baud rate generator is disabled to reduce supply current. When BR is 1 – 8191, the SCI baud rate equals SCI module clock/(16 × BR). See also BR bits in Table 14-1.

14.2.2 SCI Control Register 1 (SCIxC1)

This read/write register controls various optional features of the SCI system.

	7	6	5	4	3	2	1	0
R W	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

Figure 14-6. SCI Control Register 1 (SCIxC1)

Table 14-3. SCIxC1 Field Descriptions

Field	Description
7 LOOPS	Loop Mode Select. Selects between loop back modes and normal 2-pin full-duplex modes. When LOOPS is set, the transmitter output is internally connected to the receiver input. 0 Normal operation — RxD and TxD use separate pins. 1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input. (See RSRC bit.) RxD pin is not used by SCI.
6 SCISWAI	SCI Stops in Wait Mode 0 SCI clocks continue to run in wait mode so the SCI can be the source of an interrupt that wakes up the CPU. 1 SCI clocks freeze while CPU is in wait mode.
5 RSRC	Receiver Source Select. This bit has no meaning or effect unless the LOOPS bit is set to 1. When LOOPS is set, the receiver input is internally connected to the TxD pin and RSRC determines whether this connection is also connected to the transmitter output. 0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the SCI does not use the RxD pins. 1 Single-wire SCI mode where the TxD pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 0 Normal — start + 8 data bits (lsb first) + stop. 1 Receiver and transmitter use 9-bit data characters start + 8 data bits (lsb first) + 9th data bit + stop.
3 WAKE	Receiver Wakeup Method Select. Refer to Section 14.3.3.2, “Receiver Wakeup Operation” for more information. 0 Idle-line wakeup. 1 Address-mark wakeup.
2 ILT	Idle Line Type Select. Setting this bit to 1 ensures that the stop bit and logic 1 bits at the end of a character do not count toward the 10 or 11 bit times of logic high level needed by the idle line detection logic. Refer to Section 14.3.3.2.1, “Idle-Line Wakeup” for more information. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.

Table 14-3. SCIxC1 Field Descriptions (continued)

Field	Description
1 PE	Parity Enable. Enables hardware parity generation and checking. When parity is enabled, the most significant bit (msb) of the data character (eighth or ninth data bit) is treated as the parity bit. 0 No hardware parity generation or checking. 1 Parity enabled.
0 PT	Parity Type. Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even. 0 Even parity. 1 Odd parity.

14.2.3 SCI Control Register 2 (SCIxC2)

This register can be read or written at any time.

	7	6	5	4	3	2	1	0
R W	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

Figure 14-7. SCI Control Register 2 (SCIxC2)

Table 14-4. SCIxC2 Field Descriptions

Field	Description
7 TIE	Transmit Interrupt Enable (for TDRE) 0 Hardware interrupts from TDRE disabled (use polling). 1 Hardware interrupt requested when TDRE flag is 1.
6 TCIE	Transmission Complete Interrupt Enable (for TC) 0 Hardware interrupts from TC disabled (use polling). 1 Hardware interrupt requested when TC flag is 1.
5 RIE	Receiver Interrupt Enable (for RDRF) 0 Hardware interrupts from RDRF disabled (use polling). 1 Hardware interrupt requested when RDRF flag is 1.
4 ILIE	Idle Line Interrupt Enable (for IDLE) 0 Hardware interrupts from IDLE disabled (use polling). 1 Hardware interrupt requested when IDLE flag is 1.

Table 14-4. SCIxC2 Field Descriptions (continued)

Field	Description
3 TE	Transmitter Enable 0 Transmitter off. 1 Transmitter on. TE must be 1 to use the SCI transmitter. When TE is set, the SCI forces the TxD pin to act as an output for the SCI system. When the SCI is configured for single-wire operation (LOOPS = RSRC = 1), TXDIR controls the direction of traffic on the single SCI communication line (TxD pin). TE can also queue an idle character by clearing TE then setting TE while a transmission is in progress. Refer to Section 14.3.2.1, “Send Break and Queued Idle” for more details. When TE is written to 0, the transmitter keeps control of the port TxD pin until any data, queued idle, or queued break character finishes transmitting before allowing the pin to revert to a general-purpose I/O pin.
2 RE	Receiver Enable. When the SCI receiver is off, the RxD pin reverts to being a general-purpose port I/O pin. If LOOPS is set the RxD pin reverts to being a general-purpose I/O pin even if RE is set. 0 Receiver off. 1 Receiver on.
1 RWU	Receiver Wakeup Control. This bit can be written to 1 to place the SCI receiver in a standby state where it waits for automatic hardware detection of a selected wakeup condition. The wakeup condition is an idle line between messages (WAKE = 0, idle-line wakeup) or a logic 1 in the most significant data bit in a character (WAKE = 1, address-mark wakeup). Application software sets RWU and (normally) a selected hardware condition automatically clears RWU. Refer to Section 14.3.3.2, “Receiver Wakeup Operation,” for more details. 0 Normal SCI receiver operation. 1 SCI receiver in standby waiting for wakeup condition.
0 SBK	Send Break. Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 or 11 (13 or 14 if BRK13 = 1) bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK. Refer to Section 14.3.2.1, “Send Break and Queued Idle” for more details. 0 Normal transmitter operation. 1 Queue break character(s) to be sent.

14.2.4 SCI Status Register 1 (SCIxS1)

This register has eight read-only status flags. Writes have no effect. Special software sequences (which do not involve writing to this register) clear these status flags.

	7	6	5	4	3	2	1	0
R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W								
Reset	1	1	0	0	0	0	0	0

Figure 14-8. SCI Status Register 1 (SCIxS1)

Table 14-5. SCIxS1 Field Descriptions

Field	Description
7 TDRE	Transmit Data Register Empty Flag. TDRE is set out of reset and when a transmit data value transfers from the transmit data buffer to the transmit shifter, leaving room for a new character in the buffer. To clear TDRE, read SCIxS1 with TDRE set and then write to the SCI data register (SCIxD). 0 Transmit data register (buffer) full. 1 Transmit data register (buffer) empty.
6 TC	Transmission Complete Flag. TC is set out of reset and when TDRE is set and no data, preamble, or break character is being transmitted. 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete). TC is cleared automatically by reading SCIxS1 with TC set and then doing one of the following: • Write to the SCI data register (SCIxD) to transmit new data • Queue a preamble by changing TE from 0 to 1 • Queue a break character by writing 1 to SBK in SCIxC2
5 RDRF	Receive Data Register Full Flag. RDRF becomes set when a character transfers from the receive shifter into the receive data register (SCIxD). To clear RDRF, read SCIxS1 with RDRF set and then read the SCI data register (SCIxD). 0 Receive data register empty. 1 Receive data register full.
4 IDLE	Idle Line Flag. IDLE is set when the SCI receive line becomes idle for a full character time after a period of activity. When ILT is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bit time count toward the full character time of logic high (10 or 11 bit times depending on the M control bit) needed for the receiver to detect an idle line. When ILT is set, the receiver doesn't start counting idle bit times until after the stop bit. The stop bit and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line. To clear IDLE, read SCIxS1 with IDLE set and then read the SCI data register (SCIxD). After IDLE has been cleared, it cannot become set again until after a new character has been received and RDRF has been set. IDLE is set only once even if the receive line remains idle for an extended period. 0 No idle line detected. 1 Idle line was detected.
3 OR	Receiver Overrun Flag. OR is set when a new serial character is ready to be transferred to the receive data register (buffer), but the previously received character has not been read from SCIxD yet. In this case, the new character (and all associated error information) is lost because there is no room to move it into SCIxD. To clear OR, read SCIxS1 with OR set and then read the SCI data register (SCIxD). 0 No overrun. 1 Receive overrun (new SCI data lost).
2 NF	Noise Flag. The advanced sampling technique used in the receiver takes seven samples during the start bit and three samples in each data bit and the stop bit. If any of these samples disagrees with the rest of the samples within any bit time in the frame, the flag NF is set at the same time as RDRF is set for the character. To clear NF, read SCIxS1 and then read the SCI data register (SCIxD). 0 No noise detected. 1 Noise detected in the received character in SCIxD.

Table 14-5. SCIxS1 Field Descriptions (continued)

Field	Description
1 FE	Framing Error Flag. FE is set at the same time as RDRF when the receiver detects a logic 0 where the stop bit was expected. This suggests the receiver was not properly aligned to a character frame. To clear FE, read SCIxS1 with FE set and then read the SCI data register (SCIxD). 0 No framing error detected. This does not guarantee the framing is correct. 1 Framing error.
0 PF	Parity Error Flag. PF is set at the same time as RDRF when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, read SCIxS1 and then read the SCI data register (SCIxD). 0 No parity error. 1 Parity error.

14.2.5 SCI Status Register 2 (SCIxS2)

This register contains one read-only status flag.

	7	6	5	4	3	2	1	0
R	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-9. SCI Status Register 2 (SCIxS2)

Table 14-6. SCIxS2 Field Descriptions

Field	Description
7 LBKDIF	LIN Break Detect Interrupt Flag. LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it. 0 No LIN break character has been detected. 1 LIN break character has been detected.
6 RXEDGIF	RxD Pin Active Edge Interrupt Flag. RXEDGIF is set when an active edge (falling if RXINV = 0, rising if RXINV=1) on the RxD pin occurs. RXEDGIF is cleared by writing a 1 to it. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
4 RXINV[1]	Receive Data Inversion. Setting this bit reverses the polarity of the received data input. 0 Receive data not inverted 1 Receive data inverted
3 RWUID	Receive Wake Up Idle Detect. RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. 0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. 1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character.
2 BRK13	Break Character Generation Length. BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. 0 Break character is transmitted with length of 10 bit times (11 if M = 1) 1 Break character is transmitted with length of 13 bit times (14 if M = 1)

Table 14-6. SCIxS2 Field Descriptions (continued)

Field	Description
1 LBKDE	LIN Break Detection Enable. LBKDE selects a longer break character detection length. While LBKDE is set, framing error (FE) and receive data register full (RDRF) flags are prevented from setting. 0 Break character is detected at length of 10 bit times (11 if M = 1). 1 Break character is detected at length of 11 bit times (12 if M = 1).
0 RAF	Receiver Active Flag. RAF is set when the SCI receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. This status flag can be used to check whether an SCI character is being received before instructing the MCU to go to stop mode. 0 SCI receiver idle waiting for a start bit. 1 SCI receiver active (RxD input not idle).

[1] Setting RXINV inverts the RxD input for all cases: data bits, start and stop bits, break, and idle.

When using an internal oscillator in a LIN system, it is necessary to raise the break detection threshold one bit time. Under the worst case timing conditions allowed in LIN, it is possible that a 0x00 data character can appear to be 10.26 bit times long at a slave running 14% faster than the master. This would trigger normal break detection circuitry designed to detect a 10-bit break symbol. When the LBKDE bit is set, framing errors are inhibited and the break detection threshold changes from 10 bits to 11 bits, preventing false detection of a 0x00 data character as a LIN break symbol.

14.2.6 SCI Control Register 3 (SCIxC3)

	7	6	5	4	3	2	1	0
R	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-10. SCI Control Register 3 (SCIxC3)

Table 14-7. SCIxC3 Field Descriptions

Field	Description
7 R8	Ninth Data Bit for Receiver. When the SCI is configured for 9-bit data (M = 1), R8 can be thought of as a ninth receive data bit to the left of the msb of the buffered data in the SCIxD register. When reading 9-bit data, read R8 before reading SCIxD because reading SCIxD completes automatic flag clearing sequences that could allow R8 and SCIxD to be overwritten with new data.
6 T8	Ninth Data Bit for Transmitter. When the SCI is configured for 9-bit data (M = 1), T8 may be thought of as a ninth transmit data bit to the left of the msb of the data in the SCIxD register. When writing 9-bit data, the entire 9-bit value is transferred to the SCI shift register after SCIxD is written so T8 should be written (if it needs to change from its previous value) before SCIxD is written. If T8 does not need to change in the new value (such as when it is used to generate mark or space parity), it need not be written each time SCIxD is written.
5 TXDIR	TxD Pin Direction in Single-Wire Mode. When the SCI is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the TxD pin. 0 TxD pin is an input in single-wire mode. 1 TxD pin is an output in single-wire mode.

Table 14-7. SCIxC3 Field Descriptions (continued)

Field	Description
4 TXINV[1]	Transmit Data Inversion. Setting this bit reverses the polarity of the transmitted data output. 0 Transmit data not inverted 1 Transmit data inverted
3 ORIE	Overrun Interrupt Enable. This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0 OR interrupts disabled (use polling). 1 Hardware interrupt requested when OR is set.
2 NEIE	Noise Error Interrupt Enable. This bit enables the noise flag (NF) to generate hardware interrupt requests. 0 NF interrupts disabled (use polling). 1 Hardware interrupt requested when NF is set.
1 FEIE	Framing Error Interrupt Enable. This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0 FE interrupts disabled (use polling). 1 Hardware interrupt requested when FE is set.
0 PEIE	Parity Error Interrupt Enable. This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0 PF interrupts disabled (use polling). 1 Hardware interrupt requested when PF is set.

[1] Setting TXINV inverts the TxD output for all cases: data bits, start and stop bits, break, and idle.

14.2.7 SCI Data Register (SCIxD)

This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for the SCI status flags.

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

Figure 14-11. SCI Data Register (SCIxD)

14.3 Functional Description

The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The SCI comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. During normal operation, the MCU monitors the status of the SCI, writes the data to be transmitted, and processes received data. The following describes each of the blocks of the SCI.

14.3.1 Baud Rate Generation

As shown in Figure 14-12, the clock source for the SCI baud rate generator is the SCI module clock.

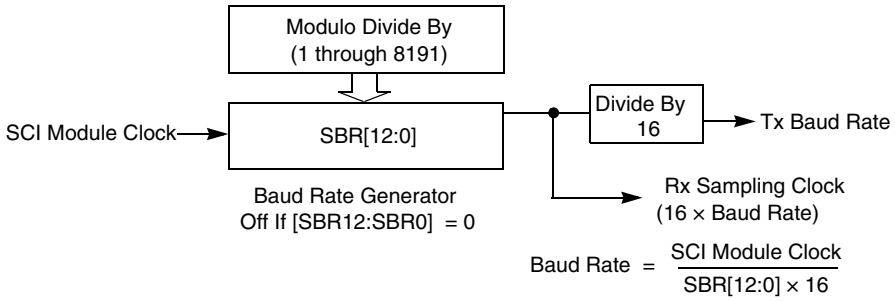


Figure 14-12. SCI Baud Rate Generation

SCI communications require the transmitter and receiver (which typically derive baud rates from independent clock sources) to use the same baud rate. Allowed tolerance on this baud frequency depends on the details of how the receiver synchronizes to the leading edge of the start bit and how bit sampling is performed.

The MCU resynchronizes to bit boundaries on every high-to-low transition. In the worst case, there are no such transitions in the full 10- or 11-bit time character frame so any mismatch in baud rate is accumulated for the whole character time. For a Freescale Semiconductor SCI system whose bus frequency is driven by a crystal, the allowed baud rate mismatch is about ±4.5 percent for 8-bit data format and about ±4 percent for 9-bit data format. Although baud rate modulo divider settings do not always produce baud rates that exactly match standard rates, it is normally possible to get within a few percent, which is acceptable for reliable communications.

14.3.2 Transmitter Functional Description

This section describes the overall block diagram for the SCI transmitter, as well as specialized functions for sending break and idle characters. The transmitter block diagram is shown in Figure 14-2.

The transmitter output (TxD) idle state defaults to logic high (TXINV is cleared following reset). The transmitter output is inverted by setting TXINV. The transmitter is enabled by setting the TE bit in SCIxC2. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the SCI data register (SCIxD).

The central element of the SCI transmitter is the transmit shift register that is 10 or 11 bits long depending on the setting in the M control bit. For the remainder of this section, assume M is cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new SCI character, the value waiting in the transmit data register is transferred to the shift register (synchronized with the baud rate clock) and the transmit data register empty (TDRE) status flag is set to indicate another character may be written to the transmit data buffer at SCIxD.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the TxD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TxD high, waiting for more characters to transmit.

Writing 0 to TE does not immediately release the pin to be a general-purpose I/O pin. Any transmit activity in progress must first be completed. This includes data characters in progress, queued idle characters, and queued break characters.

14.3.2.1 Send Break and Queued Idle

The SBK control bit in SCIxC2 sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0 (10 bit times including the start and stop bits). A longer break of 13 bit times can be enabled by setting BRK13. Normally, a program would wait for TDRE to become set to indicate the last character of a message has moved to the transmit shifter, then write 1 and then write 0 to the SBK bit. This action queues a break character to be sent as soon as the shifter is available. If SBK remains 1 when the queued break moves into the shifter (synchronized to the baud rate clock), an additional break character is queued. If the receiving device is another Freescale Semiconductor SCI, the break characters are received as 0s in all eight data bits and a framing error (FE = 1) occurs.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for TDRE to become set to indicate the last character of a message has moved to the transmit shifter, then write 0 and then write 1 to the TE bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while TE is cleared, the SCI transmitter never actually releases control of the TxD pin. If there is a possibility of the shifter finishing while TE is cleard, set the general-purpose I/O controls so the pin shared with TxD is an output driving a logic 1. This ensures that the TxD line looks like a normal idle line even if the SCI loses control of the port pin between writing 0 and then 1 to TE.

The length of the break character is affected by the BRK13 and M bits as shown below.

Table 14-8. Break Character Length

BRK13	M	Break Character Length
0	0	10 bit times
0	1	11 bit times
1	0	13 bit times
1	1	14 bit times

14.3.3 Receiver Functional Description

In this section, the receiver block diagram (Figure 14-3) is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, two variations of the receiver wakeup function are explained.

The receiver input is inverted by setting RXINV. The receiver is enabled by setting the RE bit in SCIxC2. Character frames consist of a start bit of logic 0, eight (or nine) data bits (lsb first), and a stop bit of logic 1. For information about 9-bit data mode, refer to Section •, “8- and 9-bit data modes”. For the remainder of this discussion, 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 two-step sequence normally satisfied in the course of the user's program that manages receive data. Refer to Section 14.3.4, "Interrupts and Status Flags," for more details about flag clearing.

14.3.3.1 Data Sampling Technique

The SCI receiver uses a 16× 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× baud rate clock divides 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) is set when the received character is transferred to the receive data buffer.

The falling edge detection logic continuously looks for falling edges. 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 remains set.

14.3.3.2 Receiver Wakeup Operation

Receiver wakeup is a hardware mechanism that allows an SCI receiver to ignore the characters in a message intended for a different SCI receiver. In such a system, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up (RWU) control bit in SCIxC2. When RWU bit is set, the status flags associated with the receiver (with the exception of the idle bit, IDLE, when RWUID bit is set) are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At

the end of a message, or at the beginning of the next message, all receivers automatically force RWU to 0 so all receivers wake up in time to look at the first character(s) of the next message.

14.3.3.2.1 Idle-Line Wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, RWU is cleared automatically when the receiver detects a full character time of the idle-line level. The M control bit selects 8-bit or 9-bit data mode that determines how many bit times of idle are needed to constitute a full character time (10 or 11 bit times because of the start and stop bits).

When RWU is one and RWUID is zero, the idle condition that wakes up the receiver does not set the IDLE flag. The receiver wakes up and waits for the first data character of the next message that sets the RDRF flag and generates an interrupt if enabled. When RWUID is one, any idle condition sets the IDLE flag and generates an interrupt if enabled, regardless of whether RWU is zero or one.

The idle-line type (ILT) control bit selects one of two ways to detect an idle line. When ILT is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle. When ILT is set, the idle bit counter does not start until after a stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

14.3.3.2.2 Address-Mark Wakeup

When wake is set, the receiver is configured for address-mark wakeup. In this mode, RWU is cleared automatically when the receiver detects a logic 1 in the most significant bit of a received character (eighth bit when M is cleared and ninth bit when M is set).

Address-mark wakeup allows messages to contain idle characters, but requires the msb be reserved for use in address frames. The logic 1 msb of an address frame clears the RWU bit before the stop bit is received and sets the RDRF flag. In this case, the character with the msb set is received even though the receiver was sleeping during most of this character time.

14.3.4 Interrupts and Status Flags

The SCI system has three separate interrupt vectors to reduce the amount of software needed to isolate the cause of the interrupt. One interrupt vector is associated with the transmitter for TDRE and TC events. Another interrupt vector is associated with the receiver for RDRF, IDLE, RXEDGIF, and LBKDIF events. A third vector is used for OR, NF, FE, and PF error conditions. Each of these ten interrupt sources can be separately masked by local interrupt enable masks. The flags can be polled by software when the local masks are cleared to disable generation of hardware interrupt requests.

The SCI transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty (TDRE) indicates when there is room in the transmit data buffer to write another transmit character to SCIxD. If the transmit interrupt enable (TIE) bit is set, a hardware interrupt is requested when TDRE is set. Transmit complete (TC) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TxD at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (TCIE) bit is set, a hardware interrupt is requested when TC is set. Instead of hardware

interrupts, software polling may be used to monitor the TDRE and TC status flags if the corresponding TIE or TCIE local interrupt masks are cleared.

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 by reading SCIxS1 while RDRF is set and then reading SCIxD.

When polling is used, this sequence is naturally satisfied in the normal course of the user program. If hardware interrupts are used, SCIxS1 must be read in the interrupt service routine (ISR). Normally, this is done in the ISR anyway to check for receive errors, so the sequence is automatically satisfied.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the RxD line remains idle for an extended period of time. IDLE is cleared by reading SCIxS1 while IDLE is set and then reading SCIxD. After IDLE has been cleared, it cannot become set again until the receiver has received at least one new character and has set RDRF.

If the associated error was detected in the received character that caused RDRF to be set, the error flags — noise flag (NF), framing error (FE), and parity error flag (PF) — are set at the same time as RDRF. These flags are not set in overrun cases.

If RDRF was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (OR) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

At any time, an active edge on the RxD serial data input pin causes the RXEDGIF flag to set. The RXEDGIF flag is cleared by writing a 1 to it. This function does depend on the receiver being enabled (RE = 1).

14.3.5 Additional SCI Functions

The following sections describe additional SCI functions.

14.3.5.1 8- and 9-Bit Data Modes

The SCI system (transmitter and receiver) can be configured to operate in 9-bit data mode by setting the M control bit in SCIxC1. In 9-bit mode, there is a ninth data bit to the left of the msb of the SCI data register. For the transmit data buffer, this bit is stored in T8 in SCIxC3. For the receiver, the ninth bit is held in R8 in SCIxC3.

For coherent writes to the transmit data buffer, write to the T8 bit before writing to SCIxD.

If the bit value to be transmitted as the ninth bit of a new character is the same as for the previous character, it is not necessary to write to T8 again. When data is transferred from the transmit data buffer to the transmit shifter, the value in T8 is copied at the same time data is transferred from SCIxD to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. In custom protocols, the ninth bit can also serve as a software-controlled marker.

14.3.5.2 Stop Mode Operation

During all stop modes, clocks to the SCI module are halted.

In stop1 and stop2 modes, all SCI register data is lost and must be re-initialized upon recovery from these two stop modes. No SCI module registers are affected in stop3 mode.

The receive input active edge detect circuit remains active in stop3 mode, but not in stop2. An active edge on the receive input brings the CPU out of stop3 mode if the interrupt is not masked (RXEDGIE = 1).

Because the clocks are halted, the SCI module resumes operation upon exit from stop (only in stop3 mode). Software should ensure stop mode is not entered while there is a character being transmitted out of or received into the SCI module.

14.3.5.3 Loop Mode

When LOOPS is set, the RSRC bit in the same register chooses between loop mode (RSRC = 0) or single-wire mode (RSRC = 1). Loop mode is sometimes used to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the RxD pin is not used by the SCI, so it reverts to a general-purpose port I/O pin.

14.3.5.4 Single-Wire Operation

When LOOPS is set, the RSRC bit in the same register chooses between loop mode (RSRC = 0) or single-wire mode (RSRC = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TxD pin. The RxD pin is not used and reverts to a general-purpose port I/O pin.

In single-wire mode, the TXDIR bit in SCIxC3 controls the direction of serial data on the TxD pin. When TXDIR is cleared, the TxD pin is an input to the SCI receiver and the transmitter is temporarily disconnected from the TxD pin so an external device can send serial data to the receiver. When TXDIR is set, the TxD pin is an output driven by the transmitter. In single-wire mode, the internal loop back connection from the transmitter to the receiver causes the receiver to receive characters that are sent out by the transmitter.

Chapter 15
Serial Peripheral Interface (S08SPIV4)

15.1 Introduction

Figure 15-1 shows the MC9S08LH64 series block diagram with the SPI highlighted.

15.1.1 Module Configuration

The SPI module pins, MISO, MOSI, SPSCK and $\overline{SS}$ can be repositioned under software control using SPIPS in SOPT2 as shown in Table 15-1. SPIPS in SOPT2 selects which general-purpose I/O ports are associated with SPI operation.

Table 15-1. SPI Position Options

SPIPS in SOPT2	Port Pin for MISO	Port Pin for MOSI	Port Pin for SPSCK	Port Pin for $\overline{SS}$
0 (default)	PTA2	PTA3	PTA1	PTA0
1	PTB4	PTB5	PTB6	PTB7

15.1.2 SPI Clock Gating

The bus clock to the SPI can be gated on and off using the SPI bit in SCGC2. These bits are set after any reset, which enables the bus clock to this module. To conserve power, these bits can be cleared to disable the clock to this module when not in use. See Section 5.7, “Peripheral Clock Gating,” for details.

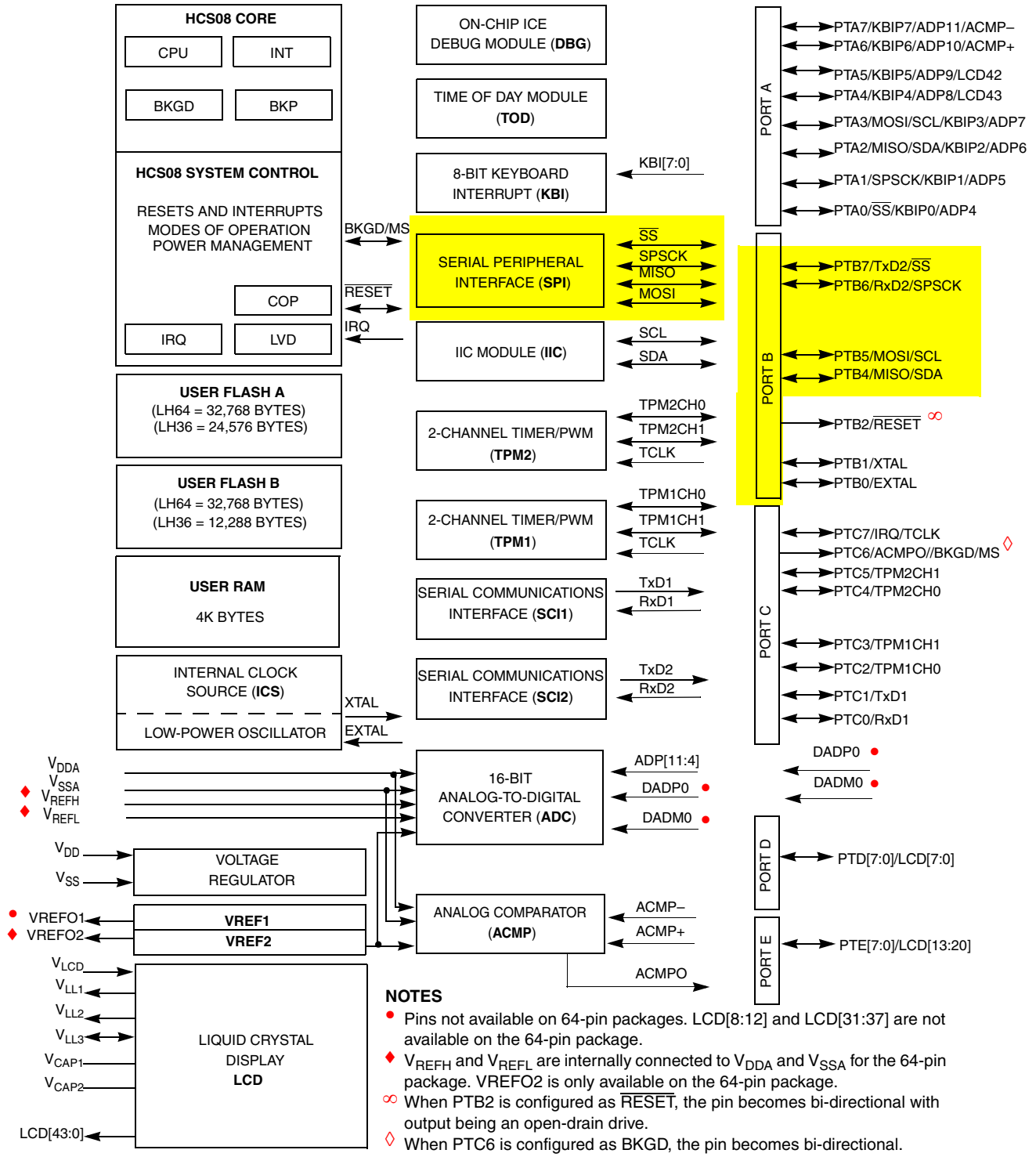


Figure 15-1. MC9S08LH64 Series Block Diagram Highlighting SPI Block and Pins

15.1.3 Features

Features of the SPI module include:

- Master or slave mode operation
- Full-duplex or single-wire bidirectional option
- Programmable transmit bit rate
- Double-buffered transmit and receive
- Serial clock phase and polarity options
- Slave select output
- Selectable MSB-first or LSB-first shifting

15.1.4 Block Diagrams

This section includes block diagrams showing SPI system connections, the internal organization of the SPI module, and the SPI clock dividers that control the master mode bit rate.

15.1.4.1 SPI System Block Diagram

Figure 15-2 shows the SPI modules of two MCUs connected in a master-slave arrangement. The master device initiates all SPI data transfers. During a transfer, the master shifts data out (on the MOSI pin) to the slave while simultaneously shifting data in (on the MISO pin) from the slave. The transfer effectively exchanges the data that was in the SPI shift registers of the two SPI systems. The SPSCK signal is a clock output from the master and an input to the slave. The slave device must be selected by a low level on the slave select input ($\overline{SS}$ pin). In this system, the master device has configured its $\overline{SS}$ pin as an optional slave select output.

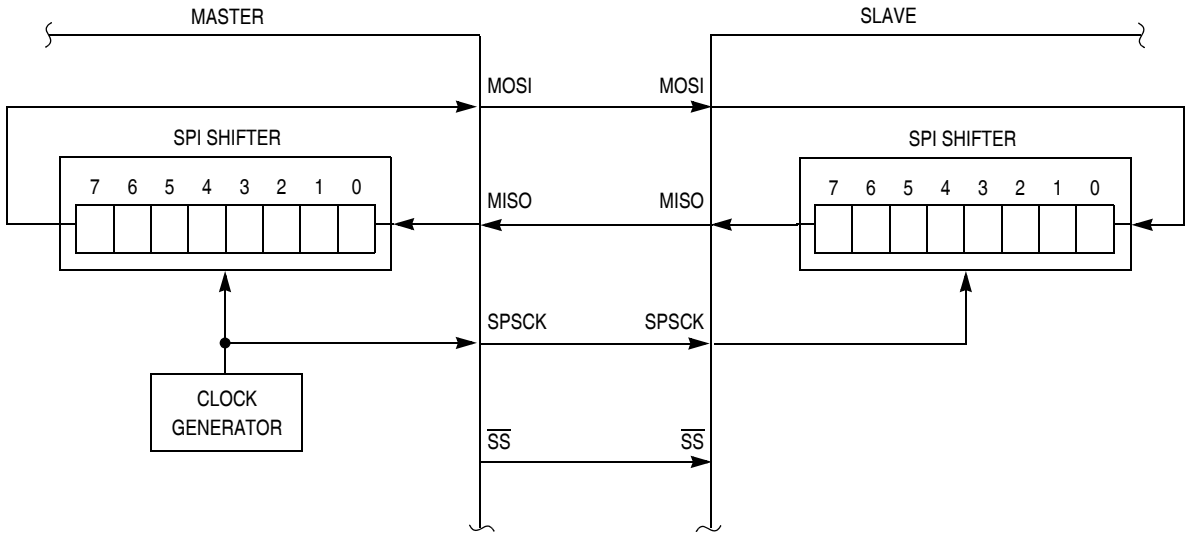


Figure 15-2. SPI System Connections

The most common uses of the SPI system include connecting simple shift registers for adding input or output ports or connecting small peripheral devices such as serial A/D or D/A converters. Although Figure 15-2 shows a system where data is exchanged between two MCUs, many practical systems involve simpler connections where data is unidirectionally transferred from the master MCU to a slave or from a slave to the master MCU.

15.1.4.2 SPI Module Block Diagram

Figure 15-3 is a block diagram of the SPI module. The central element of the SPI is the SPI shift register. Data is written to the double-buffered transmitter (write to SPID) and gets transferred to the SPI shift register at the start of a data transfer. After shifting in a byte of data, the data is transferred into the double-buffered receiver where it can be read (read from SPID). Pin multiplexing logic controls connections between MCU pins and the SPI module.

When the SPI is configured as a master, the clock output is routed to the SPSCK pin, the shifter output is routed to MOSI, and the shifter input is routed from the MISO pin.

When the SPI is configured as a slave, the SPSCK pin is routed to the clock input of the SPI, the shifter output is routed to MISO, and the shifter input is routed from the MOSI pin.

In the external SPI system, simply connect all SPSCK pins to each other, all MISO pins together, and all MOSI pins together. Peripheral devices often use slightly different names for these pins.

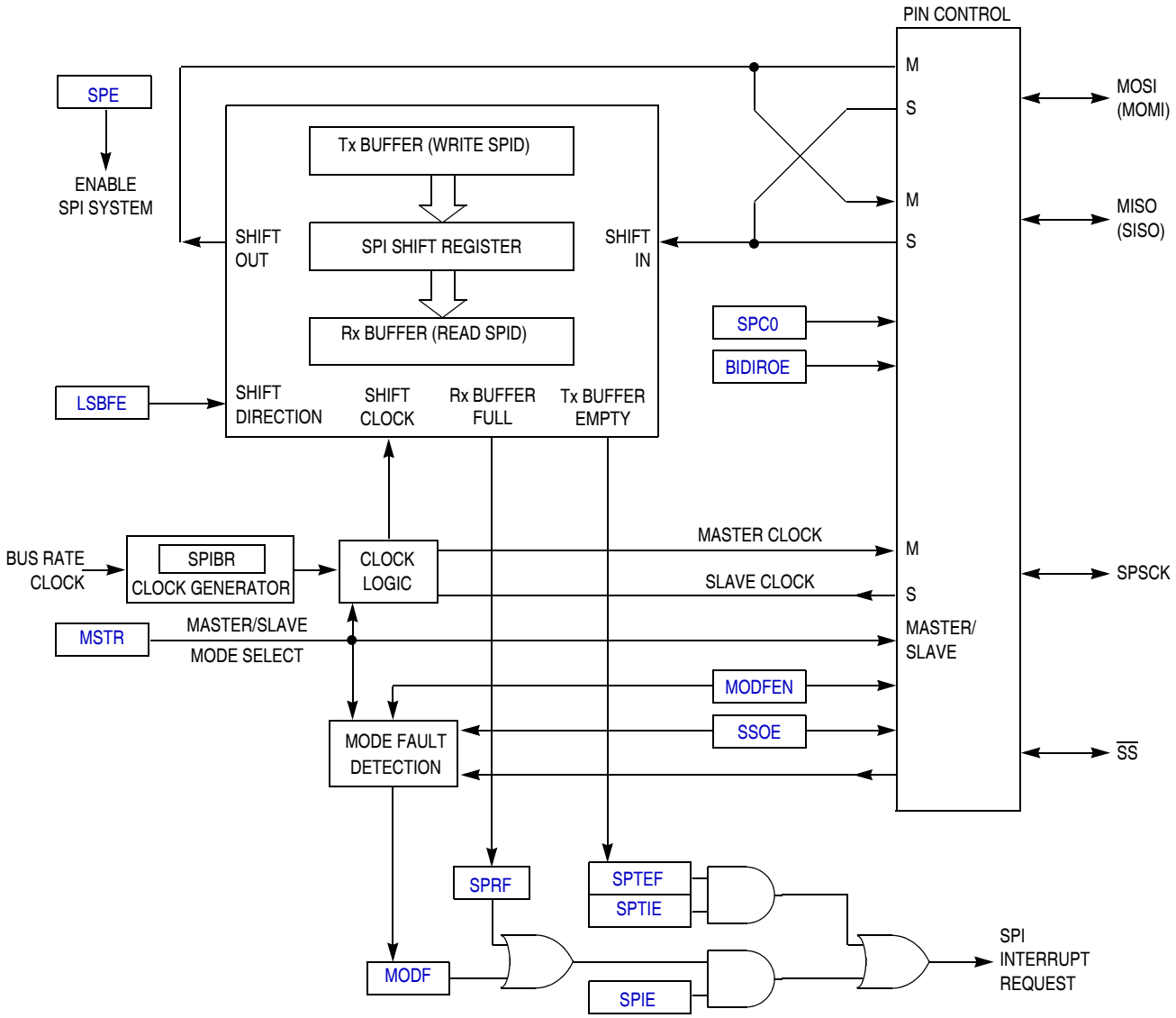


Figure 15-3. SPI Module Block Diagram

15.1.5 SPI Baud Rate Generation

As shown in Figure 15-4, the clock source for the SPI baud rate generator is the bus clock. The three prescale bits (SPPR2:SPPR1:SPPR0) choose a prescale divisor of 1, 2, 3, 4, 5, 6, 7, or 8. The three rate select bits (SPR3:SPR2:SPR1:SPR0) divide the output of the prescaler stage by 2, 4, 8, 16, 32, 64, 128, 256,or 512 to get the internal SPI master mode bit-rate clock.

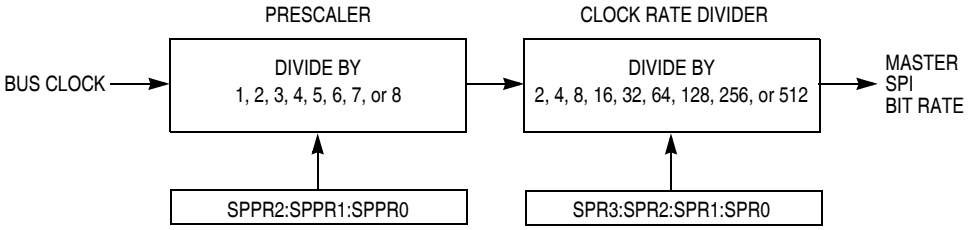


Figure 15-4. SPI Baud Rate Generation

15.2 External Signal Description

The SPI optionally shares four port pins. The function of these pins depends on the settings of SPI control bits. When the SPI is disabled (SPE = 0), these four pins revert to being general-purpose port I/O pins that are not controlled by the SPI.

15.2.1 SPSCK — SPI Serial Clock

When the SPI is enabled as a slave, this pin is the serial clock input. When the SPI is enabled as a master, this pin is the serial clock output.

15.2.2 MOSI — Master Data Out, Slave Data In

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data output. When the SPI is enabled as a slave and SPC0 = 0, this pin is the serial data input. If SPC0 = 1 to select single-wire bidirectional mode, and master mode is selected, this pin becomes the bidirectional data I/O pin (MOMI). Also, the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE = 0) or an output (BIDIROE = 1). If SPC0 = 1 and slave mode is selected, this pin is not used by the SPI and reverts to being a general-purpose port I/O pin.

15.2.3 MISO — Master Data In, Slave Data Out

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data input. When the SPI is enabled as a slave and SPC0 = 0, this pin is the serial data output. If SPC0 = 1 to select single-wire bidirectional mode, and slave mode is selected, this pin becomes the bidirectional data I/O pin (SISO) and the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE = 0) or an output (BIDIROE = 1). If SPC0 = 1 and master mode is selected, this pin is not used by the SPI and reverts to being a general-purpose port I/O pin.

15.2.4 $\overline{SS}$ — Slave Select

When the SPI is enabled as a slave, this pin is the low-true slave select input. When the SPI is enabled as a master and mode fault enable is off (MODFEN = 0), this pin is not used by the SPI and reverts to being a general-purpose port I/O pin. When the SPI is enabled as a master and MODFEN = 1, the slave select output enable bit determines whether this pin acts as the mode fault input (SSOE = 0) or as the slave select output (SSOE = 1).

15.3 Modes of Operation

15.3.1 SPI in Stop Modes

The SPI is disabled in all stop modes, regardless of the settings before executing the STOP instruction. During either stop1 or stop2 mode, the SPI module will be fully powered down. Upon wake-up from stop1 or stop2 mode, the SPI module will be in the reset state. During stop3 mode, clocks to the SPI module are halted. No registers are affected. If stop3 is exited with a reset, the SPI will be put into its reset state. If stop3 is exited with an interrupt, the SPI continues from the state it was in when stop3 was entered.

15.4 Register Definition

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

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

15.4.1 SPI Control Register 1 (SPIC1)

This read/write register includes the SPI enable control, interrupt enables, and configuration options.

	7	6	5	4	3	2	1	0
R W	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
Reset	0	0	0	0	0	1	0	0

Figure 15-5. SPI Control Register 1 (SPIC1)

Table 15-2. SPIC1 Field Descriptions

Field	Description
7 SPIE	**SPI Interrupt Enable (for SPRF and MODF)** — This is the interrupt enable for SPI receive buffer full (SPRF) and mode fault (MODF) events. 0 Interrupts from SPRF and MODF inhibited (use polling) 1 When SPRF or MODF is 1, request a hardware interrupt
6 SPE	**SPI System Enable** — Disabling the SPI halts any transfer that is in progress, clears data buffers, and initializes internal state machines. SPRF is cleared and SPTEF is set to indicate the SPI transmit data buffer is empty. 0 SPI system inactive 1 SPI system enabled
5 SPTIE	**SPI Transmit Interrupt Enable** — This is the interrupt enable bit for SPI transmit buffer empty (SPTEF). 0 Interrupts from SPTEF inhibited (use polling) 1 When SPTEF is 1, hardware interrupt requested

Table 15-2. SPIC1 Field Descriptions (continued)

Field	Description
4 MSTR	**Master/Slave Mode Select** 0 SPI module configured as a slave SPI device 1 SPI module configured as a master SPI device
3 CPOL	**Clock Polarity** — This bit effectively places an inverter in series with the clock signal from a master SPI or to a slave SPI device. Refer to Section 15.5.4, "SPI Clock Formats" for more details. 0 Active-high SPI clock (idles low) 1 Active-low SPI clock (idles high)
2 CPHA	**Clock Phase** — This bit selects one of two clock formats for different kinds of synchronous serial peripheral devices. Refer to Section 15.5.4, "SPI Clock Formats" for more details. 0 First edge on SPSCK occurs at the middle of the first cycle of an 8-cycle data transfer 1 First edge on SPSCK occurs at the start of the first cycle of an 8-cycle data transfer
1 SSOE	**Slave Select Output Enable** — This bit is used in combination with the mode fault enable (MODFEN) bit in SPCR2 and the master/slave (MSTR) control bit to determine the function of the $\overline{SS}$ pin as shown in Table 15-3.
0 LSBFE	**LSB First (Shifter Direction)** 0 SPI serial data transfers start with most significant bit 1 SPI serial data transfers start with least significant bit

Table 15-3. $\overline{SS}$ Pin Function

MODFEN	SSOE	Master Mode	Slave Mode
0	0	General-purpose I/O (not SPI)	Slave select input
0	1	General-purpose I/O (not SPI)	Slave select input
1	0	$\overline{SS}$ input for mode fault	Slave select input
1	1	Automatic $\overline{SS}$ output	Slave select input

NOTE

Ensure that the SPI should not be disabled (SPE=0) at the same time as a bit change to the CPHA bit. These changes should be performed as separate operations or unexpected behavior may occur.

15.4.2 SPI Control Register 2 (SPIC2)

This read/write register is used to control optional features of the SPI system. Bits 7, 6, 5, and 2 are not implemented and always read 0.

	7	6	5	4	3	2	1	0
R	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-6. SPI Control Register 2 (SPIC2)

Table 15-4. SPIC2 Register Field Descriptions

Field	Description
4 MODFEN	**Master Mode-Fault Function Enable** — When the SPI is configured for slave mode, this bit has no meaning or effect. (The $\overline{SS}$ pin is the slave select input.) In master mode, this bit determines how the $\overline{SS}$ pin is used (refer to Table 15-3 for more details). 0 Mode fault function disabled, master $\overline{SS}$ pin reverts to general-purpose I/O not controlled by SPI 1 Mode fault function enabled, master $\overline{SS}$ pin acts as the mode fault input or the slave select output
3 BIDIROE	**Bidirectional Mode Output Enable** — When bidirectional mode is enabled by SPI pin control 0 (SPC0) = 1, BIDIROE determines whether the SPI data output driver is enabled to the single bidirectional SPI I/O pin. Depending on whether the SPI is configured as a master or a slave, it uses either the MOSI (MOMI) or MISO (SISO) pin, respectively, as the single SPI data I/O pin. When SPC0 = 0, BIDIROE has no meaning or effect. 0 Output driver disabled so SPI data I/O pin acts as an input 1 SPI I/O pin enabled as an output
1 SPISWAI	**SPI Stop in Wait Mode** 0 SPI clocks continue to operate in wait mode 1 SPI clocks stop when the MCU enters wait mode
0 SPC0	**SPI Pin Control 0** — The SPC0 bit chooses single-wire bidirectional mode. If MSTR = 0 (slave mode), the SPI uses the MISO (SISO) pin for bidirectional SPI data transfers. If MSTR = 1 (master mode), the SPI uses the MOSI (MOMI) pin for bidirectional SPI data transfers. When SPC0 = 1, BIDIROE is used to enable or disable the output driver for the single bidirectional SPI I/O pin. 0 SPI uses separate pins for data input and data output 1 SPI configured for single-wire bidirectional operation

15.4.3 SPI Baud Rate Register (SPIBR)

This register is used to set the prescaler and bit rate divisor for an SPI master. This register may be read or written at any time.

	7	6	5	4	3	2	1	0
R	0	SPPR2	SPPR1	SPPR0	SPR3	SPR2	SPR1	SPR0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-7. SPI Baud Rate Register (SPIBR)

Table 15-5. SPIBR Register Field Descriptions

Field	Description
6:4 SPPR[2:0]	**SPI Baud Rate Prescale Divisor** — This 3-bit field selects one of eight divisors for the SPI baud rate prescaler as shown in Table 15-6. The input to this prescaler is the bus rate clock (BUSCLK). The output of this prescaler drives the input of the SPI baud rate divider (see Figure 15-4).
2:0 SPR[3:0]	**SPI Baud Rate Divisor** — This 4-bit field selects one of nine divisors for the SPI baud rate divider as shown in Table 15-7. The input to this divider comes from the SPI baud rate prescaler (see Figure 15-4). The output of this divider is the SPI bit rate clock for master mode.

Table 15-6. SPI Baud Rate Prescaler Divisor

SPPR2:SPPR1:SPPR0	Prescaler Divisor
0:0:0	1
0:0:1	2
0:1:0	3
0:1:1	4
1:0:0	5
1:0:1	6
1:1:0	7
1:1:1	8

Table 15-7. SPI Baud Rate Divisor

SPR3:SPR2:SPR1:SPR0	Rate Divisor
0:0:0:0	2
0:0:0:1	4
0:0:1:0	8
0:0:1:1	16
0:1:0:0	32
0:1:0:1	64
0:1:1:0	128
0:1:1:1	256
1:0:0:0	512
All other combinations	reserved

15.4.4 SPI Status Register (SPIS)

This register has three read-only status bits. Bits 6, 3, 2, 1, and 0 are not implemented and always read 0. Writes have no meaning or effect.

	7	6	5	4	3	2	1	0
R	SPRF	0	SPTEF	MODF	0	0	0	0
W								
Reset	0	0	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-8. SPI Status Register (SPIS)

Table 15-8. SPIS Register Field Descriptions

Field	Description
7 SPRF	**SPI Read Buffer Full Flag** — SPRF is set at the completion of an SPI transfer to indicate that received data may be read from the SPI data register (SPID). SPRF is cleared by reading SPRF while it is set, then reading the SPI data register. 0 No data available in the receive data buffer 1 Data available in the receive data buffer
5 SPTEF	**SPI Transmit Buffer Empty Flag** — This bit is set when there is room in the transmit data buffer. It is cleared by reading SPIS with SPTEF set, followed by writing a data value to the transmit buffer at SPID. SPIS must be read with SPTEF = 1 before writing data to SPID or the SPID write will be ignored. SPTEF generates an SPTEF CPU interrupt request if the SPTIE bit in the SPIC1 is also set. SPTEF is automatically set when a data byte transfers from the transmit buffer into the transmit shift register. For an idle SPI (no data in the transmit buffer or the shift register and no transfer in progress), data written to SPID is transferred to the shifter almost immediately so SPTEF is set within two bus cycles allowing a second 8-bit data value to be queued into the transmit buffer. After completion of the transfer of the value in the shift register, the queued value from the transmit buffer will automatically move to the shifter and SPTEF will be set to indicate there is room for new data in the transmit buffer. If no new data is waiting in the transmit buffer, SPTEF simply remains set and no data moves from the buffer to the shifter. 0 SPI transmit buffer not empty 1 SPI transmit buffer empty
4 MODF	**Master Mode Fault Flag** — MODF is set if the SPI is configured as a master and the slave select input goes low, indicating some other SPI device is also configured as a master. The $\overline{SS}$ pin acts as a mode fault error input only when MSTR = 1, MODFEN = 1, and SSOE = 0; otherwise, MODF will never be set. MODF is cleared by reading MODF while it is 1, then writing to SPI control register 1 (SPIC1). 0 No mode fault error 1 Mode fault error detected

15.4.5 SPI Data Register (SPID)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 15-9. SPI Data Register (SPID)

Reads of this register return the data read from the receive data buffer. Writes to this register write data to the transmit data buffer. When the SPI is configured as a master, writing data to the transmit data buffer initiates an SPI transfer.

Data should not be written to the transmit data buffer unless the SPI transmit buffer empty flag (SPTEF) is set, indicating there is room in the transmit buffer to queue a new transmit byte.

Data may be read from SPID any time after SPRF is set and before another transfer is finished. Failure to read the data out of the receive data buffer before a new transfer ends causes a receive overrun condition and the data from the new transfer is lost.

15.5 Functional Description

15.5.1 General

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI Control Register 1. While the SPE bit is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select (SS)
- Serial clock (SPSCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

An SPI transfer is initiated in the master SPI device by reading the SPI status register (SPIxS) when SPTEF = 1 and then writing data to the transmit data buffer (write to SPIxD). When a transfer is complete, received data is moved into the receive data buffer. The SPIxD register acts as the SPI receive data buffer for reads and as the SPI transmit data buffer for writes.

The clock phase control bit (CPHA) and a clock polarity control bit (CPOL) in the SPI Control Register 1 (SPIxC1) select one of four possible clock formats to be used by the SPI system. The CPOL bit simply selects a non-inverted or inverted clock. The CPHA bit is used to accommodate two fundamentally different protocols by sampling data on odd numbered SPSCK edges or on even numbered SPSCK edges.

The SPI can be configured to operate as a master or as a slave. When the MSTR bit in SPI control register 1 is set, master mode is selected, when the MSTR bit is clear, slave mode is selected.

15.5.2 Master Mode

The SPI operates in master mode when the MSTR bit is set. Only a master SPI module can initiate transmissions. A transmission begins by reading the SPIxS register while SPTEF = 1 and writing to the master SPI data registers. If the shift register is empty, the byte immediately transfers to the shift register. The data begins shifting out on the MOSI pin under the control of the serial clock.

- SPSCK

The SPR3, SPR2, SPR1, and SPR0 baud rate selection bits in conjunction with the SPPR2, SPPR1, and SPPR0 baud rate preselection bits in the SPI Baud Rate register control the baud rate generator and determine the speed of the transmission. The SPSCK pin is the SPI clock output. Through the SPSCK pin, the baud rate generator of the master controls the shift register of the slave peripheral.

- MOSI, MISO pin

In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and BIDIROE control bits.

- SS pin

If MODFEN and SSOE bit are set, the SS pin is configured as slave select output. The SS output becomes low during each transmission and is high when the SPI is in idle state.

If MODFEN is set and SSOE is cleared, the SS pin is configured as input for detecting mode fault error. If the SS input becomes low this indicates a mode fault error where another master tries to drive the MOSI

and SPSCK lines. In this case, the SPI immediately switches to slave mode, by clearing the MSTR bit and also disables the slave output buffer MISO (or SISO in bidirectional mode). So the result is that all outputs are disabled and SPSCK, MOSI and MISO are inputs. If a transmission is in progress when the mode fault occurs, the transmission is aborted and the SPI is forced into idle state.

This mode fault error also sets the mode fault (MODF) flag in the SPI Status Register (SPIxS). If the SPI interrupt enable bit (SPIE) is set when the MODF flag gets set, then an SPI interrupt sequence is also requested.

When a write to the SPI Data Register in the master occurs, there is a half SPSCK-cycle delay. After the delay, SPSCK is started within the master. The rest of the transfer operation differs slightly, depending on the clock format specified by the SPI clock phase bit, CPHA, in SPI Control Register 1 (see Section 15.5.4, “SPI Clock Formats).

NOTE

A change of the bits CPOL, CPHA, SSOE, LSBFE, MODFEN, SPC0, BIDIROE with SPC0 set, SPPR2-SPPR0 and SPR3-SPR0 in master mode will abort a transmission in progress and force the SPI into idle state. The remote slave cannot detect this, therefore the master has to ensure that the remote slave is set back to idle state.

15.5.3 Slave Mode

The SPI operates in slave mode when the MSTR bit in SPI Control Register1 is clear.

- SPSCK

In slave mode, SPSCK is the SPI clock input from the master.

- MISO, MOSI pin

In slave mode, the function of the serial data output pin (MISO) and serial data input pin (MOSI) is determined by the SPC0 bit and BIDIROE bit in SPI Control Register 2.

- SS pin

The SS pin is the slave select input. Before a data transmission occurs, the SS pin of the slave SPI must be low. SS must remain low until the transmission is complete. If SS goes high, the SPI is forced into idle state.

The SS input also controls the serial data output pin, if SS is high (not selected), the serial data output pin is high impedance, and, if SS is low the first bit in the SPI Data Register is driven out of the serial data output pin. Also, if the slave is not selected (SS is high), then the SPSCK input is ignored and no internal shifting of the SPI shift register takes place.

Although the SPI is capable of duplex operation, some SPI peripherals are capable of only receiving SPI data in a slave mode. For these simpler devices, there is no serial data out pin.

NOTE

When peripherals with duplex capability are used, take care not to simultaneously enable two receivers whose serial outputs drive the same system slave's serial data output line.

As long as no more than one slave device drives the system slave's serial data output line, it is possible for several slaves to receive the same transmission from a master, although the master would not receive return information from all of the receiving slaves.

If the CPHA bit in SPI Control Register 1 is clear, odd numbered edges on the SPSCK input cause the data at the serial data input pin to be latched. Even numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

If the CPHA bit is set, even numbered edges on the SPSCK input cause the data at the serial data input pin to be latched. Odd numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

When CPHA is set, the first edge is used to get the first data bit onto the serial data output pin. When CPHA is clear and the SS input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the eighth shift, the transfer is considered complete and the received data is transferred into the SPI data registers. To indicate transfer is complete, the SPRF flag in the SPI Status Register is set.

NOTE

A change of the bits CPOL, CPHA, SSOE, LSBFE, MODFEN, SPC0 and BIDIROE with SPC0 set in slave mode will corrupt a transmission in progress and has to be avoided.

15.5.4 SPI Clock Formats

To accommodate a wide variety of synchronous serial peripherals from different manufacturers, the SPI system has a clock polarity (CPOL) bit and a clock phase (CPHA) control bit to select one of four clock formats for data transfers. CPOL selectively inserts an inverter in series with the clock. CPHA chooses between two different clock phase relationships between the clock and data.

Figure 15-10 shows the clock formats when CPHA = 1. At the top of the figure, the eight bit times are shown for reference with bit 1 starting at the first SPSCK edge and bit 8 ending one-half SPSCK cycle after the sixteenth SPSCK edge. The MSB first and LSB first lines show the order of SPI data bits depending on the setting in LSBFE. Both variations of SPSCK polarity are shown, but only one of these waveforms applies for a specific transfer, depending on the value in CPOL. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the MOSI output pin from a master and the MISO waveform applies to the MISO output from a slave. The $\overline{SS}$ OUT waveform applies to the slave select output from a master (provided MODFEN and SSOE = 1). The master $\overline{SS}$ output goes to active low one-half SPSCK cycle before the start of the transfer and goes back high at the end of the eighth bit time of the transfer. The $\overline{SS}$ IN waveform applies to the slave select input of a slave.

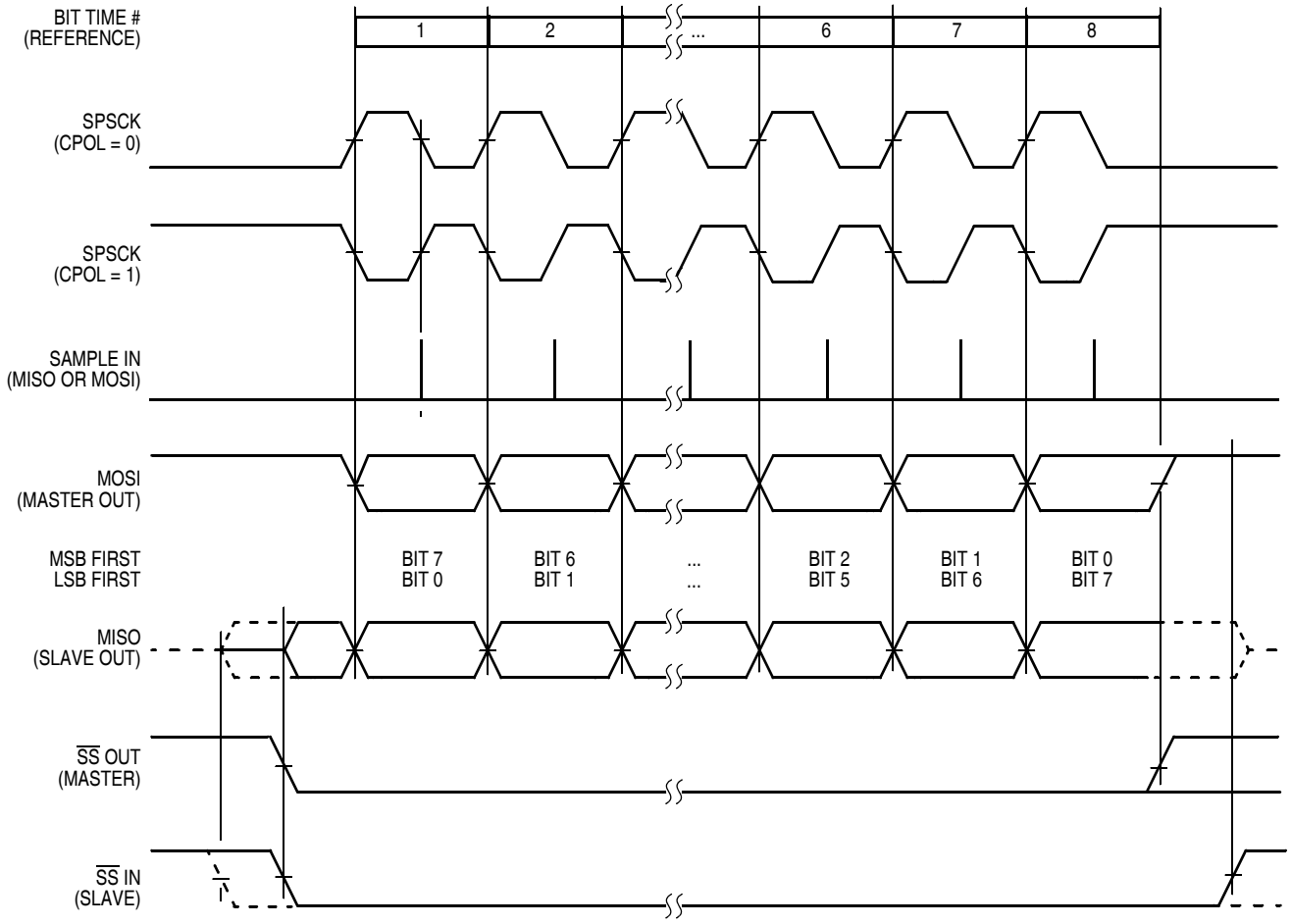


Figure 15-10. SPI Clock Formats (CPHA = 1)

When CPHA = 1, the slave begins to drive its MISO output when $\overline{SS}$ goes to active low, but the data is not defined until the first SPSCK edge. The first SPSCK edge shifts the first bit of data from the shifter onto the MOSI output of the master and the MISO output of the slave. The next SPSCK edge causes both the master and the slave to sample the data bit values on their MISO and MOSI inputs, respectively. At the third SPSCK edge, the SPI shifter shifts one bit position which shifts in the bit value that was just sampled, and shifts the second data bit value out the other end of the shifter to the MOSI and MISO outputs of the master and slave, respectively. When CHPA = 1, the slave's $\overline{SS}$ input is not required to go to its inactive high level between transfers.

Figure 15-11 shows the clock formats when CPHA = 0. At the top of the figure, the eight bit times are shown for reference with bit 1 starting as the slave is selected ($\overline{SS}$ IN goes low), and bit 8 ends at the last SPSCK edge. The MSB first and LSB first lines show the order of SPI data bits depending on the setting in LSBFE. Both variations of SPSCK polarity are shown, but only one of these waveforms applies for a specific transfer, depending on the value in CPOL. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the MOSI output pin from a master and the MISO waveform applies to the MISO output from a slave. The $\overline{SS}$ OUT waveform applies to the slave select output from a master (provided MODFEN and SSOE = 1). The master $\overline{SS}$ output goes to active low at the start of the first bit time of the transfer and goes back high one-half SPSCK cycle after

the end of the eighth bit time of the transfer. The $\overline{SS}$ IN waveform applies to the slave select input of a slave.

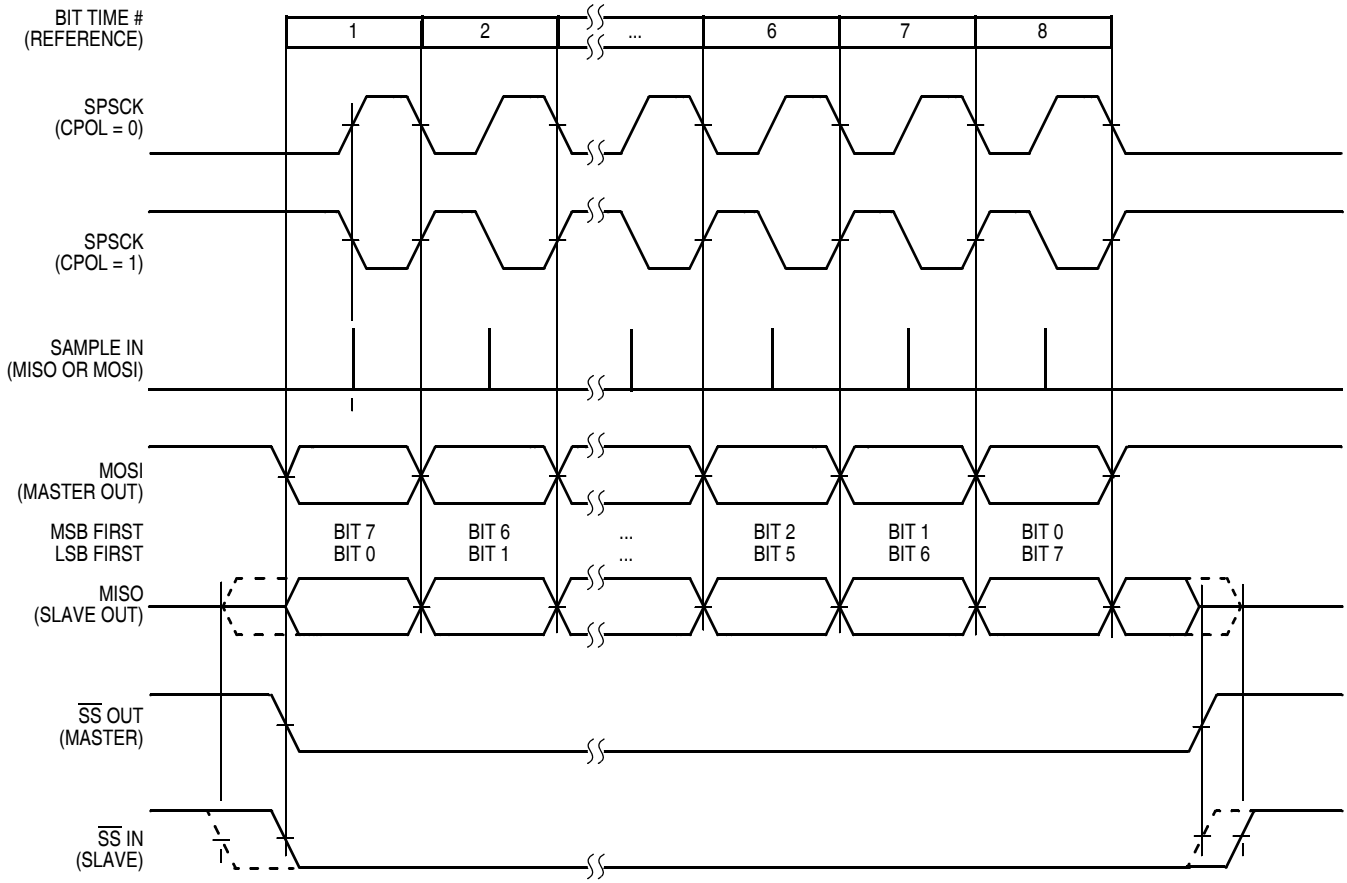


Figure 15-11. SPI Clock Formats (CPHA = 0)

When CPHA = 0, the slave begins to drive its MISO output with the first data bit value (MSB or LSB depending on LSBFE) when $\overline{SS}$ goes to active low. The first SPSCK edge causes both the master and the slave to sample the data bit values on their MISO and MOSI inputs, respectively. At the second SPSCK edge, the SPI shifter shifts one bit position which shifts in the bit value that was just sampled and shifts the second data bit value out the other end of the shifter to the MOSI and MISO outputs of the master and slave, respectively. When CPHA = 0, the slave's $\overline{SS}$ input must go to its inactive high level between transfers.

15.5.5 Special Features

15.5.5.1 SS Output

The SS output feature automatically drives the SS pin low during transmission to select external devices and drives it high during idle to deselect external devices. When SS output is selected, the SS output pin is connected to the SS input pin of the external device.

The SS output is available only in master mode during normal SPI operation by asserting the SSOE and MODFEN bits as shown in Table 15-3., “SS Pin Function.

The mode fault feature is disabled while SS output is enabled.

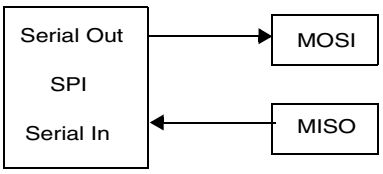
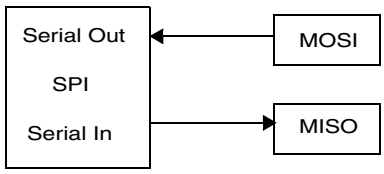
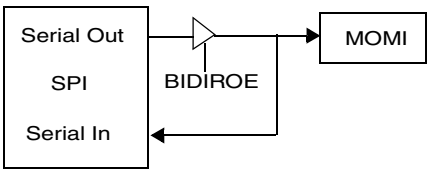
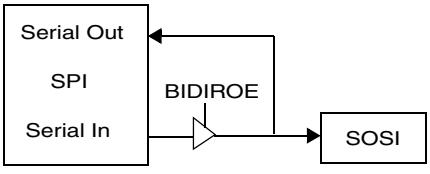
NOTE

Care must be taken when using the SS output feature in a multi-master system since the mode fault feature is not available for detecting system errors between masters.

15.5.5.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI Control Register 2 (see Section Table). In this mode, the SPI uses only one serial data pin for the interface with external device(s). The MSTR bit decides which pin to use. The MOSI pin becomes the serial data I/O (MOMI) pin for the master mode, and the MISO pin becomes serial data I/O (SISO) pin for the slave mode. The MISO pin in master mode and MOSI pin in slave mode are not used by the SPI.

Table 15-9. Normal Mode and Bidirectional Mode

When SPE = 1	Master Mode MSTR = 1	Slave Mode MSTR = 0
Normal Mode SPC0 = 0		
Bidirectional Mode SPC0 = 1		

The direction of each serial I/O pin depends on the BIDIROE bit. If the pin is configured as an output, serial data from the shift register is driven out on the pin. The same pin is also the serial input to the shift register.

The SPSCK is output for the master mode and input for the slave mode.

The SS is the input or output for the master mode, and it is always the input for the slave mode.

The bidirectional mode does not affect SPSCK and SS functions.

NOTE

In bidirectional master mode, with mode fault enabled, both data pins MISO and MOSI can be occupied by the SPI, though MOSI is normally used for transmissions in bidirectional mode and MISO is not used by the SPI. If a mode fault occurs, the SPI is automatically switched to slave mode, in this case MISO becomes occupied by the SPI and MOSI is not used. This has to be considered, if the MISO pin is used for another purpose.

15.5.6 SPI Interrupts

There are three flag bits, two interrupt mask bits, and one interrupt vector associated with the SPI system. The SPI interrupt enable mask (SPIE) enables interrupts from the SPI receiver full flag (SPRF) and mode fault flag (MODF). The SPI transmit interrupt enable mask (SPTIE) enables interrupts from the SPI transmit buffer empty flag (SPTEF). When one of the flag bits is set, and the associated interrupt mask bit is set, a hardware interrupt request is sent to the CPU. If the interrupt mask bits are cleared, software can poll the associated flag bits instead of using interrupts. The SPI interrupt service routine (ISR) should check the flag bits to determine what event caused the interrupt. The service routine should also clear the flag bit(s) before returning from the ISR (usually near the beginning of the ISR).

15.5.7 Mode Fault Detection

A mode fault occurs and the mode fault flag (MODF) becomes set when a master SPI device detects an error on the $\overline{SS}$ pin (provided the $\overline{SS}$ pin is configured as the mode fault input signal). The $\overline{SS}$ pin is configured to be the mode fault input signal when MSTR = 1, mode fault enable is set (MODFEN = 1), and slave select output enable is clear (SSOE = 0).

The mode fault detection feature can be used in a system where more than one SPI device might become a master at the same time. The error is detected when a master's $\overline{SS}$ pin is low, indicating that some other SPI device is trying to address this master as if it were a slave. This could indicate a harmful output driver conflict, so the mode fault logic is designed to disable all SPI output drivers when such an error is detected.

When a mode fault is detected, MODF is set and MSTR is cleared to change the SPI configuration back to slave mode. The output drivers on the SPSCK, MOSI, and MISO (if not bidirectional mode) are disabled.

MODF is cleared by reading it while it is set, then writing to the SPI control register 1 (SPIC1). User software should verify the error condition has been corrected before changing the SPI back to master mode.

Chapter 16
Time of Day Module (S08TODV1)

16.1 Introduction

The time of day module (TOD) consists of one 8-bit counter, one 6-bit match register, several binary-based and decimal-based prescaler dividers, three clock source options, and one interrupt that can be used for quarter second, one second and match conditions. This module can be used for time-of-day, calendar or any task scheduling functions. It can also serve as a cyclic wakeup from low-power modes without the need of external components.

16.1.1 ADC Hardware Trigger

To enable the TOD as a hardware trigger for the ADC module, set the ADTRG bit in the ADCSC2 register. When enabled, the ADC is triggered whenever a TOD match condition occurs. The TOD match interrupt does not have to be enabled to trigger the ADC, but the MTCHEN bit must be set.

16.1.2 TOD Clock Sources

The TOD module on MC9S08LH64 series can be clocked from the ICSIRCLK, OSCOUT or the LPO. When configured to use the ICSIRCLK, the IRCLKEN bit in the ICSC1 register must be set.

16.1.3 TOD Modes of Operation

All clock sources are available in all modes except stop2. The OSCOUT and LPO can be enabled as the clock source of the TOD in stop2.

16.1.4 TOD Status after Stop2 Wakeup

The registers associated with the TOD will be unaffected after a stop2 wakeup.

16.1.5 TOD Clock Gating

The bus clock to the TOD can be gated on and off using the TOD bit in SCGC2. This bit is set after any reset, which enables the bus clock to this module. To conserve power, the TOD bit can be cleared to disable the clock to this module when not in use. See Section 5.7, “Peripheral Clock Gating,” for details.

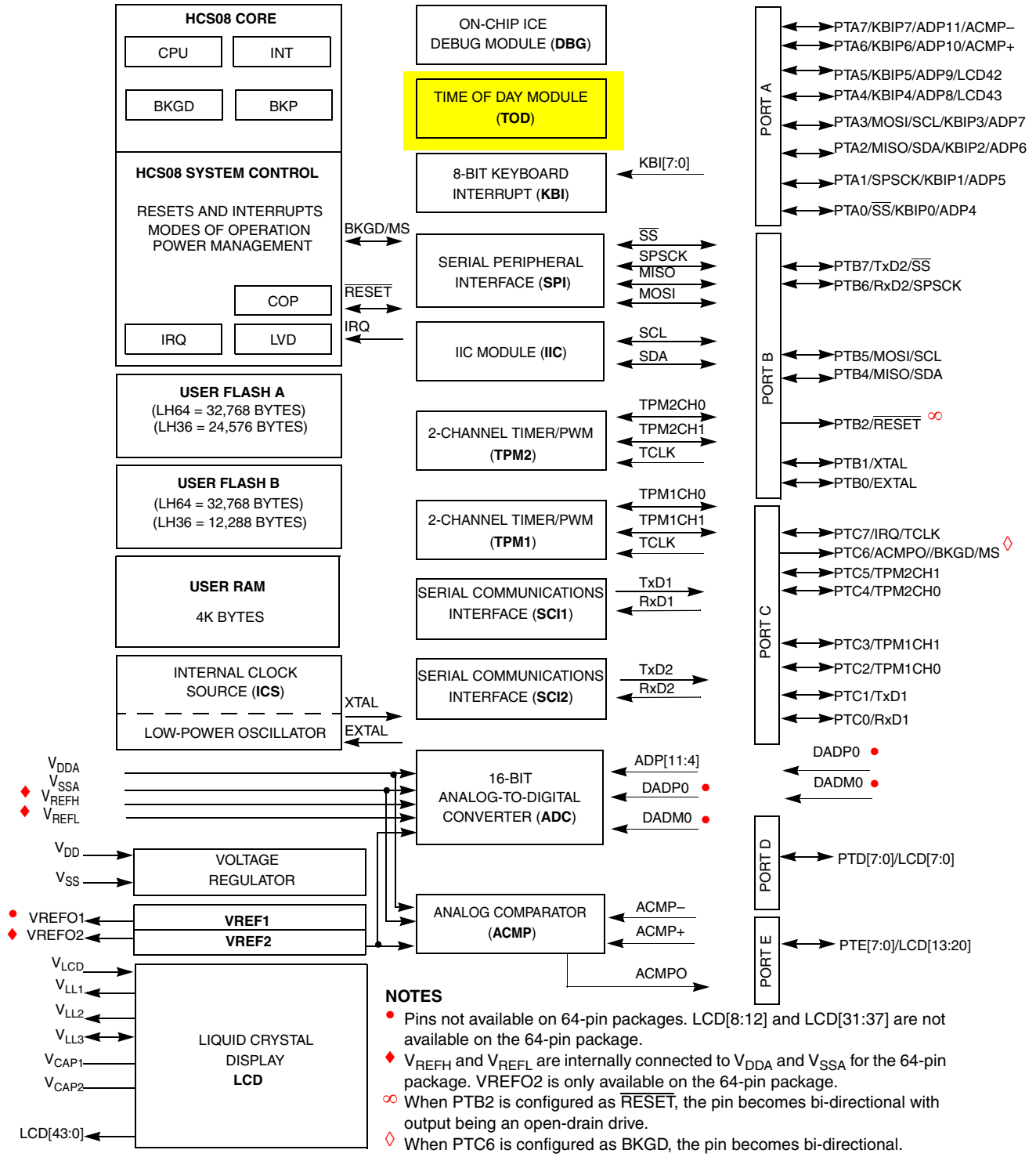


Figure 16-1. MC9S08LH64 Block Diagram Highlighting TOD Block and Pins

16.1.6 Features

Time of day module features include:

- 8-bit counter register that increments every 0.25 seconds
- Prescaler that handles standard input frequencies
- Configurable interrupts
  - — Quarter second
  - — One second
  - — Match
- Match functionality enabled by a control bit
- Counter reset on match to facilitate time-keeping software
- Operation in low-power modes: LPrun, wait, LPwait, stop3, and stop2.
- Option to use internal 1 kHz low-power oscillator (LPO)
- TOD clock output (TODCLK)

16.1.7 Modes of Operation

The TOD module supports five low power operation modes :

Table 16-1. TOD Operation Modes

Mode	Operation
Stop2	In stop2 mode the TOD module can use an external clock source or the internal 1 KHz LPO as an input. The TOD clock input ICSIRCLK is not available in stop2 mode. In stop2 mode all interrupt sources (quarter second, 1 second, and match) are capable of waking the MCU from stop2 mode. If selected, the external clock source must be enabled to operate in stop mode. (EREFSTEN) If enabled the TODCLK signal is generated.
Stop3	In stop3 mode the TOD module can use an external clock source, the internal 1 kHz LPO or ICSIRCLK as an input. In stop3 mode all interrupt sources (quarter second, 1 second, and match) are capable of waking the MCU from stop3 mode. If selected, the external clock source must be enabled to operate in stop mode. (EREFSTEN) If selected, the internal clock source must be enabled to operate in stop mode. (IREFSTEN) If enabled the TODCLK signal is generated.
LPWait	In low-power wait mode the TOD reference clock continues to run. All interrupt sources (quarter second, 1 second, and match) are capable of waking the MCU from low-power wait mode. If enabled the TODCLK signal is generated.

Table 16-1. TOD Operation Modes

Mode	Operation
Wait	In wait mode the TOD reference clock continues to run. All interrupt sources (quarter second, 1 second, and match) are capable of waking the MCU from wait mode. If enabled the TODCLK signal is generated.
LPRun	In low-power run mode the TOD reference clock continues to run. All interrupt sources (quarter second, 1 second, and match) are capable of waking the MCU from low-power run mode. If enabled the TODCLK signal is generated.

16.1.8 Block Diagram

Figure 16-2 is a block diagram of the TOD module.

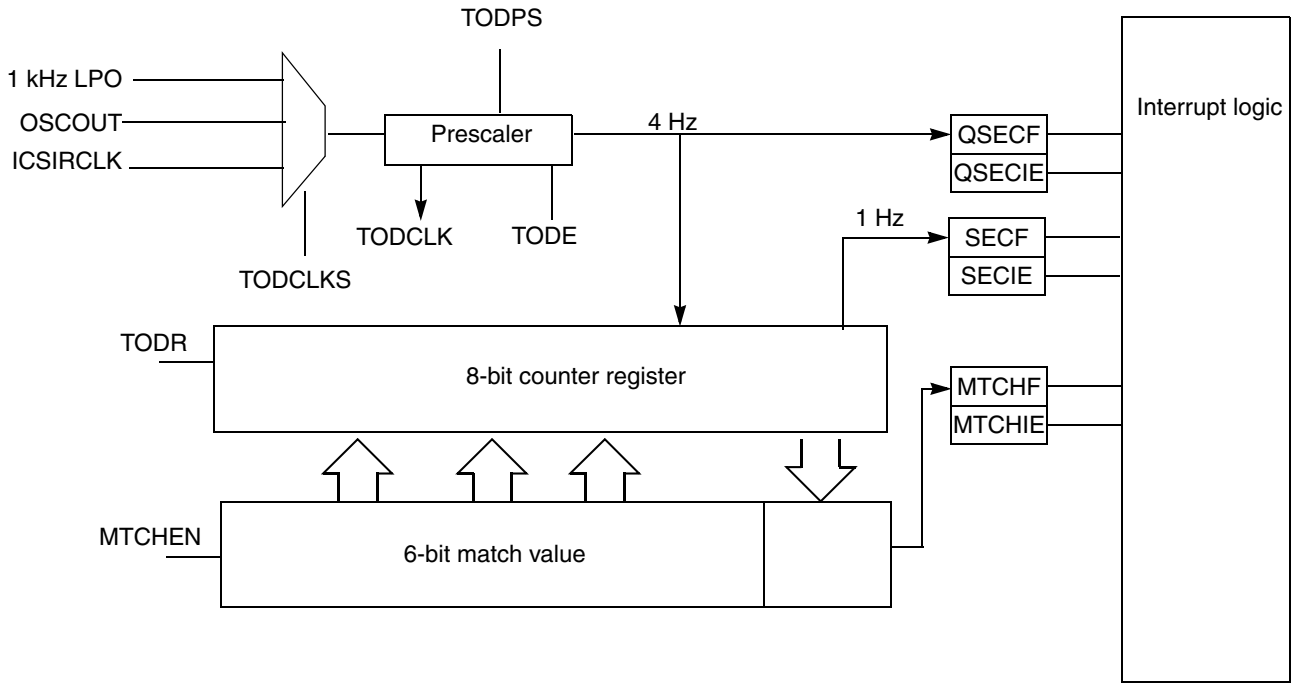


Figure 16-2. TOD Block Diagram

16.2 External Signal Description

The TOD module can output TODCLK. This clock can be used by other modules.

Table 16-2. Signal Properties

Name	Function
TODCLK	Clock output to be used by other modules.
TODMTCHS	TOD match signal

16.2.1 TOD Clock (TODCLK)

A clock can be provided for other modules. The TODPS bits are used to create the TOD clock. The TODCLKEN bit in the TODC register can enable or disable the TOD clock. If enabled the TODCLK signal will be generated in low power modes.

16.2.2 TOD Match Signal (TODMTCHS)

The TOD module has output (TODMTCHS) that is set when the TOD match condition occurs. This output does not depend the value on the MTCHIE bit. For example, this output can be used to start an ADC conversion (ADC hardware trigger). This output is cleared automatically.

16.3

This section consists of register descriptions. Each description includes a standard register diagram. Details of register bit and field function follow the register diagrams, in bit order.

16.3.1 TOD Control Register (TODC)

	7	6	5	4	3	2	1	0
R	TODEN	TODCLKS		0	TODCLKEN	TODPS2	TODPS1	TODPS0
W				TODR				
Reset	0	0	0	0	0	0	0	1

= Unimplemented or Reserved

Figure 16-3. TOD Control Register (TODC)

Read: anytime

Write:TODEN, TODR anytime.

Table 16-3. TODC Field Descriptions

Field	Description
7 TODEN	**Time of Day Enable—** The TODEN bit enables the TOD module . 0 TOD module disabled and the TODCNT is reset to $00 and the 4 Hz generator is cleared. 1 TOD module enabled.
6:5 TODCLKS	**TOD Clock Source—** The TOD module has three possible clock sources. This bit field is used to select which clock source is the basis for the TOD. Do not change TODCLKS if TODEN = 1. 00 Selects the OSCOUT clock as the TOD clock source 01 1 kHz LPO as the TOD clock source 10 Selects the ICS internal reference clock as the TOD clock source 11 Reserved
4 TODR	**TOD Reset—** Reset the TOD counter to $00 and the 4 Hz generator is cleared. When this bit is set, all TOD counts are cleared, allowing counting to begin at exactly $00. This bit always reads as 0. 0 No operation. 1 TOD counter register reset to $00.
3 TODCLKEN	**TOD Clock Enable—** The TODCLKEN bit enables the TOD clock, which can be used by other MCU peripherals. 0 TOD clock output disabled. 1 TOD clock output enabled.
2:0 TODPS[2:0]	**TOD Prescaler Bits—** The TOD prescaler bits are used to divide TOD reference clocks to create a 4 Hz timebase. Do not change TODPS if TODEN = 1. 000 1kHz prescaler (Use for 1 kHz LPO) 001 Use for 32.768 kHz (TODCLK = 32.768kHz) 010 Use for 32 kHz (TODCLK = 32kHz) 011 Use for 38.4 kHz (TODCLK = 38.4kHz) 100 Use for 4.9152 MHz (TODCLK = 38.4 kHz) 101 Use for 4 MHz (TODCLK = 32kHz) 110 Use for 8 MHz (TODCLK = 32kHz) 111 Use for 16 MHz (TODCLK = 32kHz)

16.3.2 TOD Status and Control Register (TODSC)

	7	6	5	4	3	2	1	0
R	QSECF	SECF	MTCHF	QSECIE	SECIE	MTCHIE	MTCHEN	MTCHWC
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-4. TOD Status Register (TODSC)

Read: anytime

Write: anytime

Table 16-4. TODSC Field Descriptions

Field	Description
7 QSECF	**Quarter-Second Interrupt Flag—**QSECF indicates a quarter-second condition occurred. Write a 1 to QSECF to clear the interrupt flag . 0 Quarter -second interrupt condition has not occurred. 1 Quarter-second interrupt condition has occurred.
6 SECF	**Second Interrupt Flag—** SECF indicates a one-second condition occurred. Write a 1 to SECF to clear the interrupt flag . 0 One-second interrupt condition has not occurred. 1 One-second interrupt condition has occurred.
5 MTCHF	**Match Interrupt Flag—** MTCHF indicates the match condition occurred. Write a 1 to the MTCHF to clear the interrupt flag. 0 Match interrupt condition has not occurred. 1 Match interrupt condition has occurred.
4 QSECIE	**Quarter-Second Interrupt Enable—** Enables or disables quarter-second interrupt. The quarter-second interrupt occurs every time the TOD counter register increments. 0 Quarter-second interrupt disabled. 1 Quarter-second interrupt enabled.
3 SECIE	**Second Interrupt Enable—** Enables or disables one-second interrupt. The one-second interrupt occurs every fourth count of the TOD counter register. 0 One-second interrupt disabled. 1 One-second interrupt enabled.
2 MTCHIE	**Match Interrupt Enable—** Enables/disables the match interrupt. The match interrupt condition occurs when the match functionality is enabled (MTCHEN = 1) and the TOD counter register reaches the value in the match register 0 Match interrupt disabled 1 Match interrupt enabled
1 MTCHEN	**Match Function Enable—** Enables/disables the match functionality. This bit must be set for the match interrupt to occur when the TOD counter register reaches the value in the match register. When a match occurs, the upper 6 bits of the counter register are reset to 0. 0 Match function disabled. 1 Match function enabled.
0 MTCHWC	**Match Write Complete—** This bit indicates when writes to the match register have been completed. When the match register is written this bit is set, when the match register is completed this bit is cleared automatically. 0 Last write to the TOD match register is comlete. 1 Write to the TOD match register is not complete.

16.3.3 TOD Match Register (TODM)

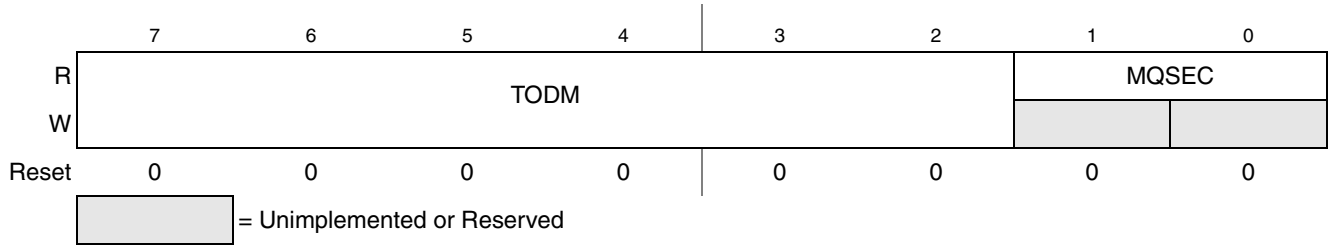


Figure 16-5. TOD Match Register (TODM)

Read: anytime

Table 16-5. TODM Field Descriptions

Field	Description
7:2 TODM	**TOD Match Value—** The 6 bit match value is compared against the upper 6 bits of the TOD counter register and a match occurs when the TOD counter register reaches the value in the TOD match register. The lower 2 bits of the TOD match register (MQSEC) are preloaded by hardware with the lower 2 bits of TODCNT when the match value is written. When a match occurs and MTCHEN = 1, the match Interrupt flag is asserted and the upper 6 bits of the TOD counter register are reset to 0.
1:0 MQSEC	**Match Quarter-Second Bits—**These bits are preloaded with the lower 2 bits of TODCNT when the match value is written. These bits are read only.

16.3.4 TOD Counter Register (TODCNT)

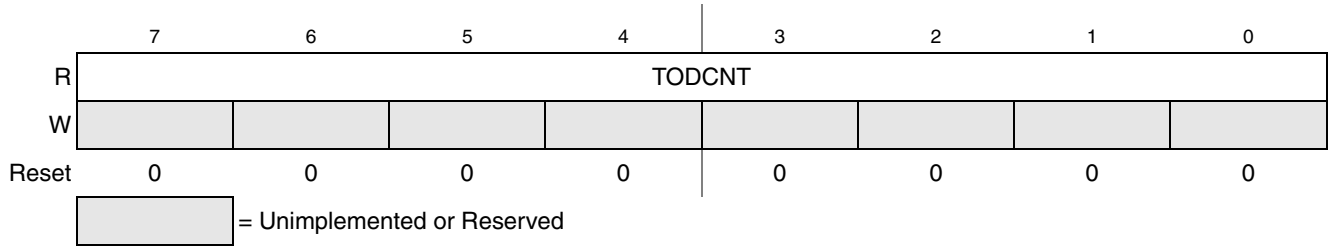


Figure 16-6. TOD Counter Register (TODCNT)

Read: anytime

Table 16-6. TODCNT Field Descriptions

Field	Description
7:0 TODCNT	**TOD Counter Register—** The TOD count register contains the current state of the TOD quarter-second counter.

16.4 Functional Description

This section provides a complete functional description of the TOD block, detailing the operation of the design from the end-user perspective.

Before enabling the TOD module by asserting the TODEN bit in the TODC register, the proper clock source and prescaler must be selected. Out of reset, the TOD module is configured with default settings, but these settings are not optimal for every application. The default settings configure TOD operation for an external 32.768 KHz oscillator. The TOD module provides a prescaler that allows several versatile configuration settings for the input clock source Refer to the TODPS bit field description in Table 16-3 to see the different input clock sources that can be used.

The TOD clock source and the TOD prescaler must be modified only if the TOD is disabled. (TODEN = 0)

Selection of the proper clock source and prescaler settings is critical because these settings are used to create the 4 Hz internal time base that is the basis for the TOD counter register. This ensures that quarter-second and one-second interrupts will occur at the appropriate times. The match interrupt can be used to generate interrupts at intervals that are multiple seconds.

Time of day can be maintained using software and the TOD module.

16.4.1 TOD Counter Register

The TOD counter is an 8-bit counter that starts at $00 and increments until $FF. After $FF is reached, the counter rolls over to $00. This creates up to 256 counts.

A 4 Hz signal is the reference clock to the TOD counter. Each tick in the TOD counter is 0.25 of a second and 4 ticks are a second. The second and quarter-second interrupts depend on proper initialization of the TOD prescaler and clock source bits.

The TODR bit in the TODC register can be used to reset the TOD counter to $00. The TODR bit also resets the 4 Hz generator ensuring that all TOD counting is starting at absolute zero time.

If the match function is enabled (MTCHEN = 1), the match flag is set when the TOD counter register reaches the value in the TOD match register and the upper 6 bits of the TOD counter register are reset to 0.

16.4.2 TOD Match Value

The TOD match value is used to generate interrupts at multiple second intervals. The TOD match value is a 6-bit value that can be set to any value from $00 to $3F. To enable match functionality, the MTCHEN bit must be set. Always configure the TODM value before setting the MTCHEN bit, this ensures that the next Match condition will occur at the desired setting.

When TODM is written,the lower 2 bits of the TODCNT are preloaded into MQSEC (the lower 2 bits of the TODM register). When TODCNT reaches the value in the TOD match register, a match condition is generated, the MTCHF is set, and the upper 6 bits of the TOD counter register are reset to 0.

For example, if the match value is set to $3C and the lower 2 bits of the TODCNT are 00 a match condition will occur when the counter transitions from $EF to $F0.

TOD Counter Register = $F0							
1	1	1	1	0	0	0	0

TOD Match Value = $3C						MQSEC	
1	1	1	1	0	0	0	0

When the match condition occurs, the upper 6 bits of the TOD counter register are reset to 0. The lower 2 bits are not affected, ensureing that one-second and quarter-second interrupt functions are not affected by a match. No time counts are lost during this time, therefore, the time interval between match conditions is exactly the same. For this case, as long as the match value is not changed, the next match condition will occur after 240 TOD counts or 1 minute. This operation facilitates timekeeping by generating a match condition after every minute.

The match value can also be used to facilitate alarm timeouts. For example, to generate an interrupt in 1 minute past the current time.

So if the TOD counter register is $50, the TOD match value must be set to $10 to generate an interrupt 1 minute past the time when the counter was $50. This can be calculated by the algorithm below.

Match value = (TOD counter register/4 + The TOD match value for 1 minute) and mask overvflow

Match value = ($50/4 + $3C) and $3F = $10

When the match condition occurs, the upper 6 bits of the TOD counter register reset to 0. If the match register is not changed, the next match condition will occur when the uper 6 bits of the TOD counter matches the match value ($10). This will occur when the TOD counter reaches $40 or 64 TOD quarter-second counts. The next match condition will occur in 16 seconds, not 1 minute.

If the TOD match value is written to a value that is below the current value of the upper 6 bits of the TOD counter register the match condition will not occur until the TOD counter has rolled over and matched the TOD match value. For example, if the TOD counter is $FF and the TOD match value is written to $01 then the match interrupt will occur after 8 TOD counter ticks (2 seconds). Subsequent match interrupts will occur every second.

If the TOD match value is written from a value that is close to the current value of the TODCNT register to a new value the match condition may still occur at the old TOD match value. The TOD match value will be valid if it is changed when the TODCNT is 2 TODCNT values from the current TOD match value.

If the TOD match value is written to $00 and the TODCNT is reset by the TODR bit or by disabling and enabling the TOD module, the TOD match condition will not be valid. Care should be taken when writing the TOD match value to $00 because the TOD match interrupt may not occur at the expected time interval if the TOD match value is written to $00 and the TODCNT is reset.

The MTCHIE bit is used to enable or disable an interrupt when a match condition occurs.

16.4.3 Match Write Complete

When writing the TOD Match Value it may be necessary to monitor the MTCHWC bit for the case of back to back writes of the TOD match value. If the second write is done while MTCHWC = 1, the write will not occur. The MTCHWC can be used to verify that the last TOD match register write is complete. Normal TOD use does not require back to back writes of the TOD match register.

16.4.4 TOD Clock Select and Prescaler

The TOD module defaults to operate using a 32.768 kHz clock input. Three possible clock sources are available to the TOD module:

OSCOUT (TODCLKS = 00)

Internal 1kHz LPO (TODCLKS =01)

ICSIRCLK (TODCLKS =10)

NOTE

Appropiately select clock source for the desired accuracy. OSCOUT is as accurate as the external source, ICSIRCLK is as accurate as the ICS specifications, and the 1 kHz LPO is the least accurate clock source.

Table 16-7 shows the TOD clock tree. The clock tree shows the three possible clock sources and the TOD clock output.

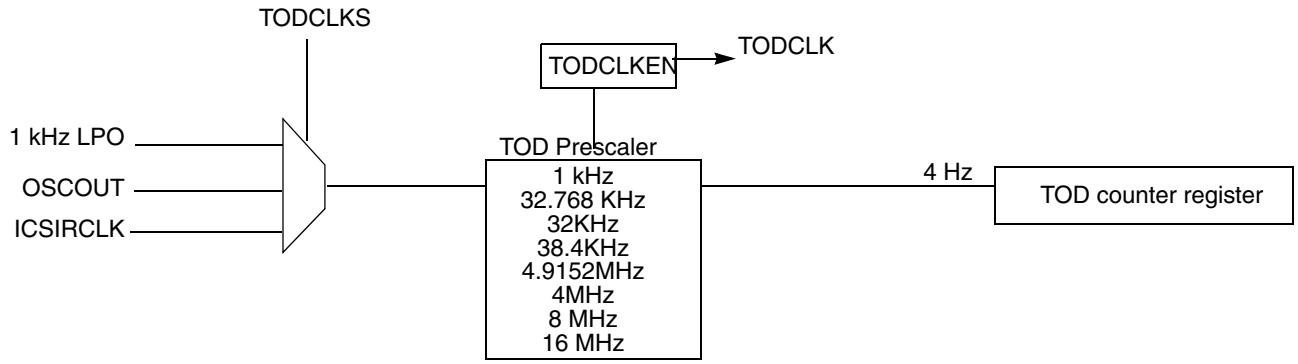


Figure 16-7. TOD Clock Tree

The TOD prescaler has dividers to generate the 4 Hz reference input to the TOD counter register. Setting the TODPS bits correctly configures internal dividers to generate the 4 Hz signal. The TOD prescaler also uses dividers to generate the TOD clock output. The TODPS must not be changed if TODEN = 1.

16.4.4.1 TOD Clock Output

The TOD module can be configured to generate an TOD clock. The TOD clock is available for use by other peripherals of the microcontroller as determined by the SOC guide. The TOD clock output is enabled with the TODCLKEN bit in the TODC register.

The TOD prescaler contains dividers for low and high frequency inputs as described in the TODPS bit description. The table below shows the value of TOD clock for various inputs.

Input	TODPS[2:0]	TOD Clock
1 kHz LPO	000	-
32.768 KHz	001	32.768 KHz
32 KHz	010	32 KHz
38.4 KHz	011	38.4 KHz
4.9152 MHz	100	38.4 kHz
4 MHz	101	32 kHz
8 MHz	110	32kHz
16 MHz	111	32 kHz

16.4.5 Quarter-Second, One-Second, and Match Interrupts

The TOD module contains quarter-second, one-second, and match interrupts to facilitate time of day applications. These interrupts have corresponding flags (QSECF, SECF, and MTCHF) that are set when the conditions are met and for the case of MTCHF the match functionality must be enabled (MTCHEN =1). These flags will be set regardless if the interrupt is enabled using the coresponding interrupt enable bits. These flags must be cleared to allow the next interrupt to occur.

16.4.5.1 Quarter-Second Interrupt

The TOD counter register is a quarter second counter. Each tick in this counter represents a quarter second. If the quarter second interrupt is enabled (QSECIE = 1) each tick in this register will result in a TOD interrupt. For example, when the TOD counter register changes from $01 to $02 QSECF will be set and the interrupt will be generated. An interrupt service routine must clear the QSECF by writing a 1 to it so that the next quarter-econd interrupt can occur.

16.4.5.2 One-Second Interrupt

The one-second interrupt is enabled by the SECIE bit in the TODSC register. When the one-second interrupt is enabled an interrupt will be generated every four counts of the TOD counter register. For example, when the TOD counter register changes from $03 to $04 the SECF will be set and an interrupt will be generated. An interrupt service routine must clear the SECF by writing a 1 to it so that the next second interrupt can occur.

If both the quarter-second and one-second interrupts are enabled, only one interrupt will be generated when the TOD counter register changes from $03 to $04. The QSECF and the SECF will be set and both must be cleared by the TOD interrupt service routine.

16.4.5.3 Match Interrupt

The TOD match functionality can be used to generate interrupts at multiple-second time intervals. This functionality must be enabled using the MTCHEN bit. If MTCHEN = 1, the TOD match interrupt will occur whenthe TOD counter register reaches the value that is placed in the TODM and the MTCHIE bit is set.

NOTE

The TOD match value (TODM) should always be written before enabling the match functionality (MTCHEN = 1).

To generate an interrupt after 4 seconds, the TOD match value must be written to a value that is 4 counts past the upper 6 bits of the TOD counter register. See Section 16.3.3, "TOD Match Register (TODM)," for an example. When the TOD counter register reaches the value placed in TODM, the MTCHF is set and an interrupt is generated. The upper 6 bits of the TOD counter are reset. An interrupt service routine must clear the MTCHF by writing a 1 to it so that the next match interrupt can occur. All interrupt flags that are set must be cleared by the TOD interrupt service routine so that the interrupt may occur again.

If the MTCHIE bit is not set when MTCHEN =1, then match conditions will occur, the MTCHF will be set, and the upper 6 bits of the TOD counter register will reset to 0 after the match condition has occured.

16.4.6 Resets

During a reset, the TOD module is configured in the default mode. The default mode includes the following settings:

- TODEN is cleared, TOD is disabled
- TODCLKS is cleared and TODPS = 001, defaults to 32.768 kHz external oscillator clock source
- All TOD interrupts are disabled
- MTCHEN is cleared, match condition is disabled
- TOD counter register and TOD match register are cleared

16.4.7 Interrupts

See Section 16.4.5, "Quarter-Second, One-Second, and Match Interrupts."

16.5 Initialization

This section provides a recommended initialization sequence for the TOD module and also includes initialization examples for several possible TOD application scenarios.

16.5.1 Initialization Sequence

This list provides a recommended initialization sequence for the TOD module.

1. Configure the TODC register
   b) Configure TOD clock source (TODCLKS bit)

c) Configure the proper TOD prescaler (TODPS bits)

2. Write the TOD match register (optional)
   d) Write the TOD match value to the desired value (TODM register)
3. Enable the TODSC register(optional)
   a) Enable match functionality
   b) Enable desired TOD interrupts (QSECIE, SECIE, MTCHIE bits)
4. Enable TODC register
   a) (Optional) Enable TOD clock output (TODCLKEN bit)
   b) Enable the TOD module (TODEN bit)

16.5.2 Initialization Examples

This section provides initialization information for configurating the TOD. Each example details the register and bit field values required to achieve the appropriate TOD configuration for a given TOD application scenario. Table 16-7 lists each example and the setup requirements.

Table 16-7. TOD Application Scenarios

Example	TOD Clock Source	Required Interrupt Timeout	TOD Clock Output
1	External 32.768 kHz	0.25 sec.	no
2	Internal 38.4 kHz	1 sec.	no
3	Internal 1 kHz LPO	approx. 1 min.	no

These examples illustrate the flexibility of the TOD module to be configured to meet a range of application requirements including:

- Clock inputs/sources
- Required interrupt timeout
- TOD clock output

16.5.2.1 Initialization Example 1

TOD setup requirements for example 1 are reiterated in Table 16-8:

Table 16-8. TOD Setup Requirements (Example 1)

Example	TOD Clock Source	Required Interrupt Timeout	TOD clock Output
1	External 32.768 kHz	0.25 sec.	no

Table 16-9 lists the required setup values required to initialize the TOD as specified by example 1:

Table 16-9. Initialization Register Values for Example 1

Register	Bit or Bit Field	Binary Value	Comment
TODC 100X1001	TODEN	1	Enable the TOD module, last step of initialization
	TODCLKS	00	Use OSCOUT for external 32.768 KHz TOD clock source
	TODR	X	TOD reset optional
	TODCLKEN	0	TOD clock output is disabled
	TODPS	001	For 32.768 KHz, TODPS is set to 001
TODSC 1XX1000x	QSECF	1	Clear the quarter-second interrupt QSECF
	SECF	X	One-second interrupt is not used
	MTCHF	X	Match interrupt is not used
	QSECIE	1	Enable quarter-second interrupts
	SECIE	0	One-second interrupt is not used
	MTCHIE	0	Match interrupt is not used
	MTCHEN	0	Match functionality is disabled
	MTCHWC	X	Match complete flag not used
TODM $XX	TODM	$XX	Match value is not used

16.5.2.2 Initialization Example 2

TOD setup requirements for example 2 are reiterated in Table 16-10:

Table 16-10. TOD Setup Requirements (Example 2)

Example	TOD Clock Source	Required Interrupt Timeout	TOD Clock Output
2	Internal 38.4 kHz	1 sec.	no

Table 16-11 lists the required setup values required to initialize the TOD as specified by example 2:

Table 16-11. Initialization Register Values for Example 2

Register	Bit or Bit Field	Binary Value	Comment
TODC 110X1010	TODEN	1	Enable the TOD module, last step of initialization
	TODCLKS	10	Use ICSIRCLK for internal 38.4 kHz TOD clock source
	TODR	X	TOD Reset optional
	TODCLKEN	0	TODCLK output is disabled
	TODPS	011	For 38.4 KHz, TODPS is set to 011
TODSC X1X0100x	QSECF	X	Quarter-second interrupt is not used
	SECF	1	Clear one-second interrupt flag
	MTCHF	X	Match interrupt is not used
	QSECIE	0	Quarter-second interrupt is not used
	SECIE	1	Enable one-second interrupt
	MTCHIE	0	Match interrupt is not used
	MTCHEN	0	Match functionality is disabled
	MTCHWC	X	Match complete flag not used
TODM $XX	TODM	$XX	Match value is not used

16.5.2.3 Initialization Example 3

TOD setup requirements for example 3 are reiterated in Table 16-12:

Table 16-12. TOD Setup Requirements (Example 3)

Example	TOD Clock Source	Required Interrupt Timeout	TOD Clock Output
3	Internal 1 kHz LPO	approx. 1 min.	no

Table 16-13 lists the required setup values required to initialize the TOD as specified by Example 3:

Table 16-13. Initialization Register Values for Example 3

Register	Bit or Bit field	Binary Value	Comment
TODC 10110000	TODEN	1	Enable the TOD module, last step of initialization
	TODCLKS	01	Use for internal 1 kHz low-power oscillator TOD clock source
	TODR	1	TOD reset to begin TOD counts at $00
	TODCLKEN	0	TODCLK output disabled
	TODPS	000	For 1 KHz low-power oscillator,TODPS is set to 000
TODSC XX10011x	QSECF	X	Quarter-second interrupt is not used
	SECF	X	One-second interrupt is not used
	MTCHF	1	Clear match interrupt flag
	QSECIE	0	Quarter-second interrupt is not used
	SECIE	0	One-second interrupt is not used
	MTCHIE	1	Match interrupt is enabled
	MTCHEN	1	Match functionality is enabled
	MTCHWC	X	Match complete flag not used
TODM 111100XX	TODM	$3C	Match value is set to $3C to generate an interrupt after $F0 or 240 counts of the TOD counter register. Always write the match value before setting the MTCHEN bit.

16.6 Application Information

The most common application for the TOD module is keeping the time of day. The TOD module can be used to keep track of the second, minute, hour, day, month, and year. To accomplish these tasks, software is used to count quarter seconds or seconds. Seconds are added up in RAM until they reach minutes, minutes are added up until they reach hours, and so on.

Chapter 17
Timer Pulse-Width Modulator (S08TPMV3)

17.1 Introduction

Figure 17-1 shows the MC9S08LH64 series block diagram with the TPM highlighted.

17.1.1 ACMP/TPM Configuration Information

The ACMP module can be configured to connect the output of the analog comparator to the TPM2 input capture channel 0 by setting the corresponding ACIC bit in SOPT2. With ACIC set, the TPM2CH0 pin is not available externally regardless of the configuration of the TPM2 module.

17.1.2 ADC/TPM Configuration Information

The ADC module can be configured to use the TPM2 as a hardware trigger manager.

- When ADCTRS = 1, the ADCHWT (Hardware Trigger) is provided by the TPM2. TPM2 can be enabled as a hardware trigger manager for the ADC module by setting the ADTRG bit in the ADCSC2 register.
  - When enabled, the ADC will be triggered by TPM output. The timer overflow interrupt does not have to be enabled to trigger the ADC.
  - The TPM2 channel 0 output will trigger a conversion on channel A
  - The TPM2 channel 1 output will trigger a conversion on channel B.
  - With this configuration, conversion selection between ADSC1A and ADSC1B can be controlled using the TPM.

NOTE

If Trigger A or B occurs while a conversion is in progress the trigger is ignored.

The TPM must not be configured to trigger A and B at the same time.

17.1.3 TPM External Clock

The TPM modules on the MC9S08LH64 series use the TCLK pin.

17.1.4 TPM Clock Gating

The bus clock to the TPMs can be gated on and off using the TPM2 bit and TPM1 bit in SCGC1. These bits are set after any reset, which enables the bus clock to this module. To conserve power, these bits can

be cleared to disable the clock to this module when not in use. See Section 5.7, "Peripheral Clock Gating," for details.

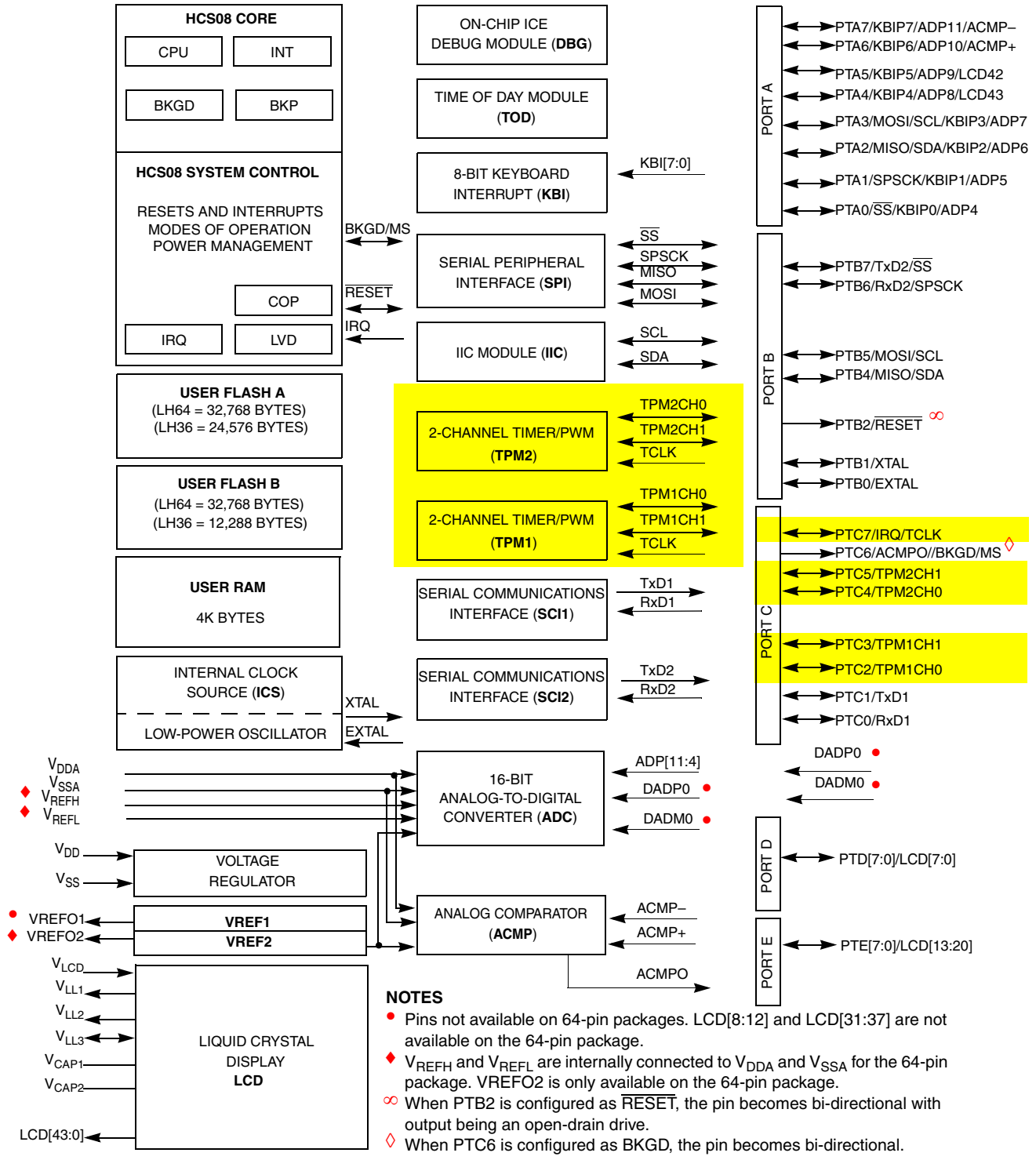


Figure 17-1. MC9S08LH64 Series Block Diagram Highlighting TPM Blocks and Pins

17.1.5 Features

The TPM includes these distinctive features:

- One to eight channels:
  - Each channel is input capture, output compare, or edge-aligned PWM
  - Rising-edge, falling-edge, or any-edge input capture trigger
  - Set, clear, or toggle output compare action
  - Selectable polarity on PWM outputs
- Module is configured for buffered, center-aligned pulse-width-modulation (CPWM) on all channels
- Timer clock source selectable as bus clock, fixed frequency clock, or an external clock
  - Prescale taps for divide-by 1, 2, 4, 8, 16, 32, 64, or 128 used for any clock input selection
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than bus clock
  - Selecting external clock connects TPM clock to a chip level input pin therefore allowing to synchronize the TPM counter with an off chip clock source
- 16-bit free-running or modulus count with up/down selection
- One interrupt per channel and one interrupt for TPM counter overflow

17.1.6 Modes of Operation

In general, TPM channels are independently configured to operate in input capture, output compare, or edge-aligned PWM modes. A control bit allows the whole TPM (all channels) to switch to center-aligned PWM mode. When center-aligned PWM mode is selected, input capture, output compare, and edge-aligned PWM functions are not available on any channels of this TPM module.

When the MCU is in active BDM background or BDM foreground mode, the TPM temporarily suspends all counting until the MCU returns to normal user operating mode. During stop mode, all TPM input clocks are stopped, so the TPM is effectively disabled until clocks resume. During wait mode, the TPM continues to operate normally. If the TPM does not need to produce a real time reference or provide the interrupt sources needed to wake the MCU from wait mode, the power can then be saved by disabling TPM functions before entering wait mode.

- Input capture mode

  When a selected edge event occurs on the associated MCU pin, the current value of the 16-bit timer counter is captured into the channel value register and an interrupt flag bit is set. Rising edges, falling edges, any edge, or no edge (disable channel) are selected as the active edge that triggers the input capture.
- Output compare mode

  When the value in the timer counter register matches the channel value register, an interrupt flag bit is set, and a selected output action is forced on the associated MCU pin. The output compare action is selected to force the pin to zero, force the pin to one, toggle the pin, or ignore the pin (used for software timing functions).
- Edge-aligned PWM mode

The value of a 16-bit modulo register plus 1 sets the period of the PWM output signal. The channel value register sets the duty cycle of the PWM output signal. You can also choose the polarity of the PWM output signal. Interrupts are available at the end of the period and at the duty-cycle transition point. This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period that is same for all channels within a TPM.

- Center-aligned PWM mode

  Twice the value of a 16-bit modulo register sets the period of the PWM output, and the channel-value register sets the half-duty-cycle duration. The timer counter counts up until it reaches the modulo value and then counts down until it reaches zero. As the count matches the channel value register while counting down, the PWM output becomes active. When the count matches the channel value register while counting up, the PWM output becomes inactive. This type of PWM signal is called center-aligned because the centers of the active duty cycle periods for all channels are aligned with a count value of zero. This type of PWM is required for types of motors used in small appliances.

This is a high-level description only. Detailed descriptions of operating modes are in later sections.

17.1.7 Block Diagram

The TPM uses one input/output (I/O) pin per channel, TPMxCHn (timer channel n) where n is the channel number (1–8). The TPM shares its I/O pins with general purpose I/O port pins (refer to I/O pin descriptions in full-chip specification for the specific chip implementation).

Figure 17-2 shows the TPM structure. The central component of the TPM is the 16-bit counter that can operate as a free-running counter or a modulo up/down counter. The TPM counter (when operating in normal up-counting mode) provides the timing reference for the input capture, output compare, and edge-aligned PWM functions. The timer counter modulo registers, TPMxMODH:TPMxMODL, control the modulo value of the counter (the values 0x0000 or 0xFFFF effectively make the counter free running). Software can read the counter value at any time without affecting the counting sequence. Any write to either half of the TPMxCNT counter resets the counter, regardless of the data value written.

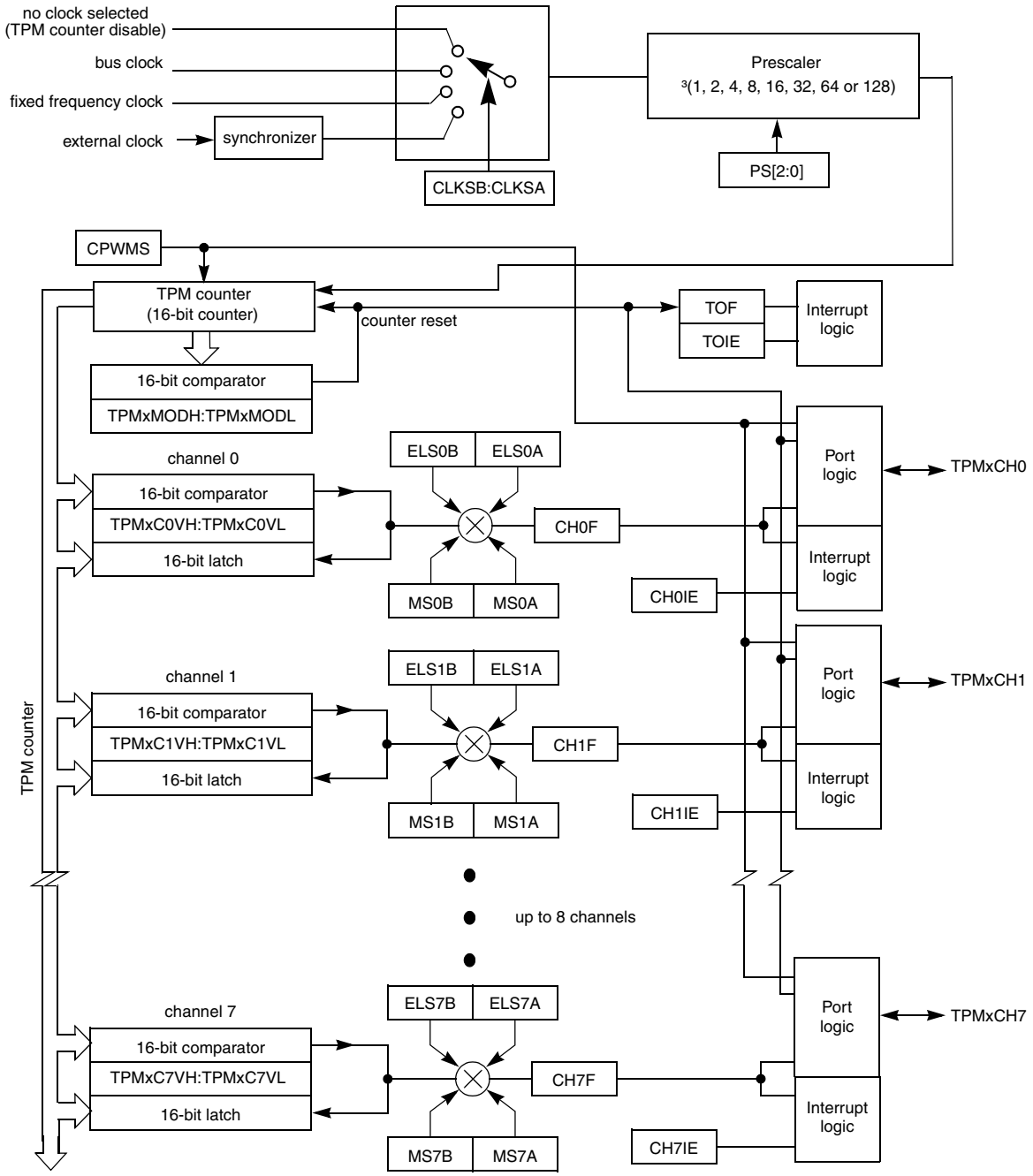


Figure 17-2. TPM Block Diagram

The TPM channels are programmable independently as input capture, output compare, or edge-aligned PWM channels. Alternately, the TPM can be configured to produce CPWM outputs on all channels. When the TPM is configured for CPWMs (the counter operates as an up/down counter) input capture, output compare, and EPWM functions are not practical.

17.2 Signal Description

Table 17-1 shows the user-accessible signals for the TPM. The number of channels are varied from one to eight. When an external clock is included, it can be shared with the same pin as any TPM channel; however, it could be connected to a separate input pin. Refer to the I/O pin descriptions in full-chip specification for the specific chip implementation.

Table 17-1. Signal Properties

Name	Function
EXTCLK[1]	External clock source that is selected to drive the TPM counter.
TPMxCHn[2]	I/O pin associated with TPM channel n.

1 The external clock pin can be shared with any channel pin. However, depending upon full-chip implementation, this signal could be connected to a separate external pin.

2 n = channel number (1–8)

17.2.1 Detailed Signal Descriptions

17.2.1.1 EXTCLK — External Clock Source

The external clock signal can share the same pin as a channel pin, however the channel pin can not be used for channel I/O function when external clock is selected. If this pin is used as an external clock (CLKSB:CLKSA = 1:1), the channel can still be configured to output compare mode therefore allowing its use as a timer (ELSnB:ELSnA = 0:0).

For proper TPM operation, the external clock frequency must not exceed one-fourth of the bus clock frequency.

17.2.1.2 TPMxCHn — TPM Channel n I/O Pins

The TPM channel does not control the I/O pin when ELSnB:ELSnA or CLKSB:CLKSA are cleared so it normally reverts to general purpose I/O control. When CPWMS is set and ELSnB:ELSnA are not cleared, all TPM channels are configured for center-aligned PWM and the TPMxCHn pins are all controlled by TPM. When CPWMS is cleared, the MSnB:MSnA control bits determine whether the channel is configured for input capture, output compare, or edge-aligned PWM.

When a channel is configured for input capture (CPWMS = 0, MSnB:MSnA = 0:0, and ELSnB:ELSnA ≠ 0:0), the TPMxCHn pin is forced to act as an edge-sensitive input to the TPM. ELSnB:ELSnA control bits determine what polarity edge or edges trigger input capture events. The channel input signal is synchronized on the bus clock. This implies the minimum pulse width—that can

be reliably detected—on an input capture pin is four bus clock periods (with ideal clock pulses as near as two bus clocks can be detected).

When a channel is configured for output compare (CPWMS = 0, MSnB:MSnA = 0:1, and ELSnB:ELSnA ≠ 0:0), the TPMxCHn pin is an output controlled by the TPM. The ELSnB:ELSnA bits determine whether the TPMxCHn pin is toggled, cleared, or set each time the 16-bit channel value register matches the TPM counter.

When the output compare toggle mode is initially selected, the previous value on the pin is driven out until the next output compare event, the pin is then toggled.

When a channel is configured for edge-aligned PWM (CPWMS = 0, MSnB = 1, and ELSnB:ELSnA ≠ 0:0), the TPMxCHn pin is an output controlled by the TPM, and ELSnB:ELSnA bits control the polarity of the PWM output signal. When ELSnB is set and ELSnA is cleared, the TPMxCHn pin is forced high at the start of each new period (TPMxCNT=0x0000), and it is forced low when the channel value register matches the TPM counter. When ELSnA is set, the TPMxCHn pin is forced low at the start of each new period (TPMxCNT=0x0000), and it is forced high when the channel value register matches the TPM counter.

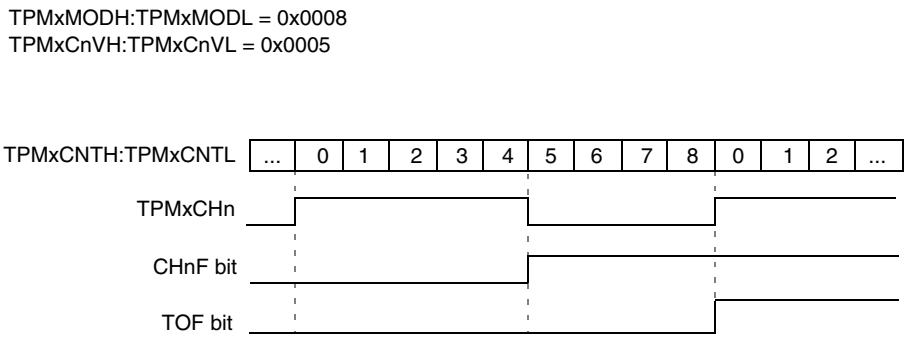


Figure 17-3. High-true pulse of an edge-aligned PWM

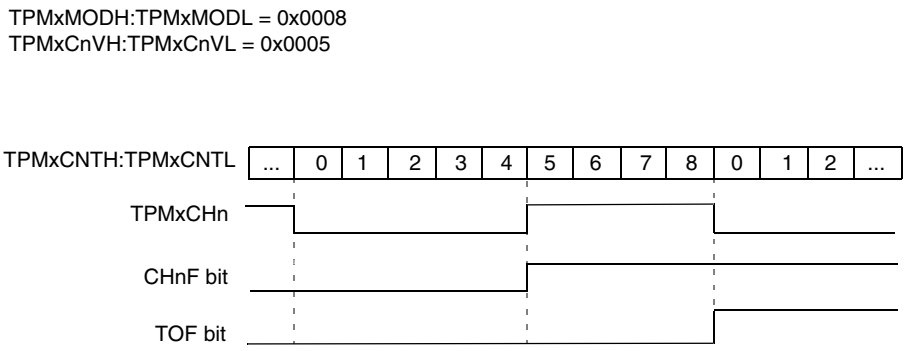


Figure 17-4. Low-true pulse of an edge-aligned PWM

When the TPM is configured for center-aligned PWM (CPWMS = 1 and ELSnB:ELSnA ≠ 0:0), the TPMxCHn pins are outputs controlled by the TPM, and ELSnB:ELSnA bits control the polarity of the PWM output signal. If ELSnB is set and ELSnA is cleared, the corresponding TPMxCHn pin is cleared when the TPM counter is counting up, and the channel value register matches the TPM counter; and it is

set when the TPM counter is counting down, and the channel value register matches the TPM counter. If ELSnA is set, the corresponding TPMxCHn pin is set when the TPM counter is counting up and the channel value register matches the TPM counter; and it is cleared when the TPM counter is counting down and the channel value register matches the TPM counter.

TPMxMODH:TPMxMODL = 0x0008
TPMxCnVH:TPMxCnVL = 0x0005

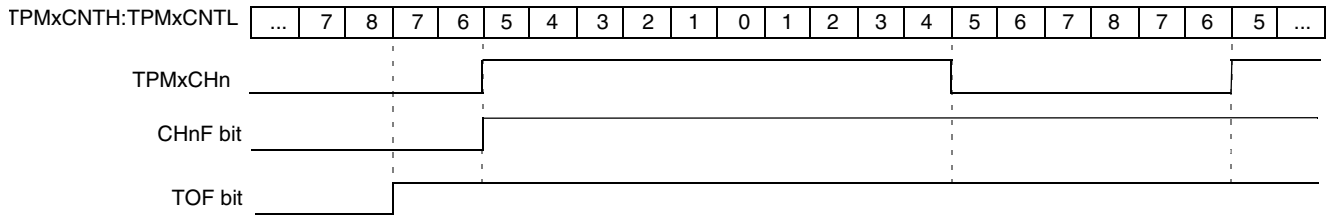


Figure 17-5. High-true pulse of a center-aligned PWM

TPMxMODH:TPMxMODL = 0x0008
TPMxCnVH:TPMxCnVL = 0x0005

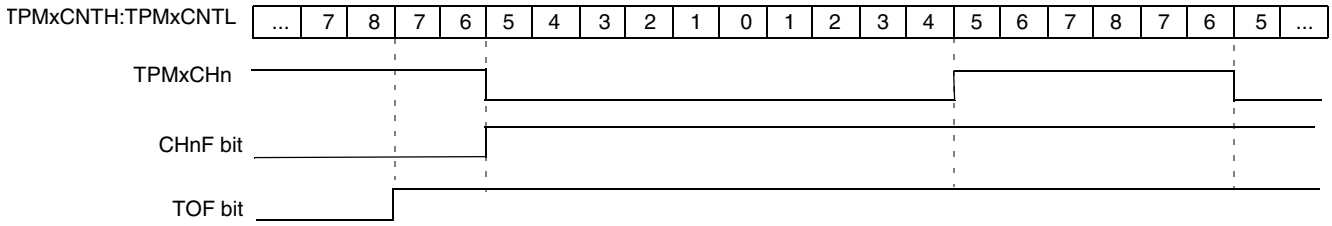


Figure 17-6. Low-true pulse of a center-aligned PWM

17.3 Register Definition

17.3.1 TPM Status and Control Register (TPMxSC)

TPMxSC contains the overflow status flag and control bits used to configure the interrupt enable, TPM configuration, clock source, and prescale factor. These controls relate to all channels within this timer module.

	7	6	5	4	3	2	1	0
R	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
W	0							
Reset	0	0	0	0	0	0	0	0

Figure 17-7. TPM Status and Control Register (TPMxSC)

Table 17-2. TPMxSC Field Descriptions

Field	Description
7 TOF	Timer overflow flag. This read/write flag is set when the TPM counter resets to 0x0000 after reaching the modulo value programmed in the TPM counter modulo registers. Clear TOF by reading the TPM status and control register when TOF is set and then writing a logic 0 to TOF. If another TPM overflow occurs before the clearing sequence is completed, the sequence is reset so TOF remains set after the clear sequence was completed for the earlier TOF. This is done so a TOF interrupt request cannot be lost during the clearing sequence for a previous TOF. Reset clears TOF. Writing a logic 1 to TOF has no effect. 0 TPM counter has not reached modulo value or overflow. 1 TPM counter has overflowed.
6 TOIE	Timer overflow interrupt enable. This read/write bit enables TPM overflow interrupts. If TOIE is set, an interrupt is generated when TOF equals one. Reset clears TOIE. 0 TOF interrupts inhibited (use for software polling). 1 TOF interrupts enabled.
5 CPWMS	Center-aligned PWM select. This read/write bit selects CPWM operating mode. By default, the TPM operates in up-counting mode for input capture, output compare, and edge-aligned PWM functions. Setting CPWMS reconfigures the TPM to operate in up/down counting mode for CPWM functions. Reset clears CPWMS. 0 All channels operate as input capture, output compare, or edge-aligned PWM mode as selected by the MSnB:MSnA control bits in each channel's status and control register. 1 All channels operate in center-aligned PWM mode.
4–3 CLKS[B:A]	Clock source selection bits. As shown in Table 17-3, this 2-bit field is used to disable the TPM counter or select one of three clock sources to TPM counter and counter prescaler.
2–0 PS[2:0]	Prescale factor select. This 3-bit field selects one of eight division factors for the TPM clock as shown in Table 17-4. This prescaler is located after any clock synchronization or clock selection so it affects the clock selected to drive the TPM counter. The new prescale factor affects the selected clock on the next bus clock cycle after the new value is updated into the register bits.

Table 17-3. TPM Clock Selection

CLKSB:CLKSA	TPM Clock to Prescaler Input
00	No clock selected (TPM counter disable)

Table 17-3. TPM Clock Selection

CLKSB:CLKSA	TPM Clock to Prescaler Input
01	Bus clock
10	Fixed frequency clock
11	External clock

Table 17-4. Prescale Factor Selection

PS[2:0]	TPM Clock Divided-by
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

17.3.2 TPM-Counter Registers (TPMxCNTH:TPMxCNTL)

The two read-only TPM counter registers contain the high and low bytes of the value in the TPM counter. Reading either byte (TPMxCNTH or TPMxCNTL) latches the contents of both bytes into a buffer where they remain latched until the other half is read. This allows coherent 16-bit reads in big-endian or little-endian order that makes this more friendly to various compiler implementations. The coherency mechanism is automatically restarted by an MCU reset or any write to the timer status/control register (TPMxSC).

Reset clears the TPM counter registers. Writing any value to TPMxCNTH or TPMxCNTL also clears the TPM counter (TPMxCNTH:TPMxCNTL) and resets the coherency mechanism, regardless of the data involved in the write.

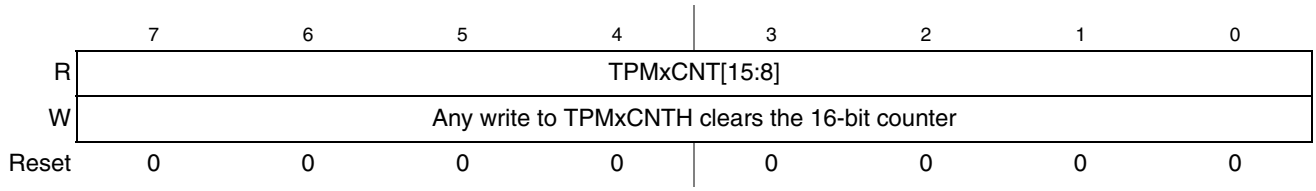


Figure 17-8. TPM Counter Register High (TPMxCNTH)

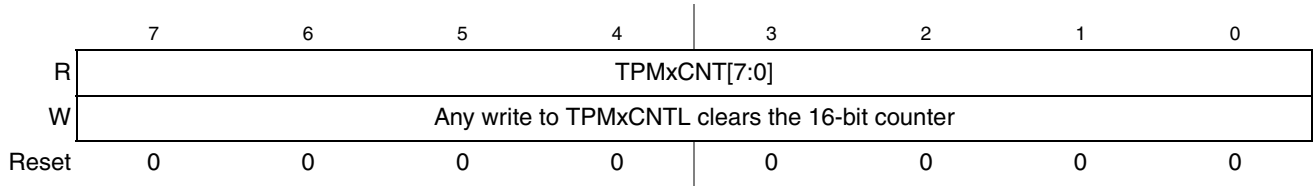


Figure 17-9. TPM Counter Register Low (TPMxCNTL)

When BDM is active, the timer counter is frozen (this is the value you read). The coherency mechanism is frozen so the buffer latches remain in the state they were in when the BDM became active, even if one or both counter halves are read while BDM is active. This assures that if you were in the middle of reading a 16-bit register when BDM became active, it reads the appropriate value from the other half of the 16-bit value after returning to normal execution.

In BDM mode, writing any value to TPMxSC, TPMxCNTH, or TPMxCNTL registers resets the read coherency mechanism of the TPMxCNTH:TPMxCNTL registers, regardless of the data involved in the write.

17.3.3 TPM Counter Modulo Registers (TPMxMODH:TPMxMODL)

The read/write TPM modulo registers contain the modulo value for the TPM counter. After the TPM counter reaches the modulo value, the TPM counter resumes counting from 0x0000 at the next clock, and the overflow flag (TOF) becomes set. Writing to TPMxMODH or TPMxMODL inhibits the TOF bit and overflow interrupts until the other byte is written. Reset sets the TPM counter modulo registers to 0x0000 that results in a free running timer counter (modulo disabled).

Writes to any of the registers TPMxMODH and TPMxMODL actually writes to buffer registers and the registers are updated with the value of their write buffer according to the value of CLKSB:CLKSA bits:

- If CLKSB and CLKSA are cleared, the registers are updated when the second byte is written
- If CLKSB and CLKSA are not cleared, the registers are updated after both bytes were written, and the TPM counter changes from (TPMxMODH:TPMxMODL – 1) to (TPMxMODH:TPMxMODL). If the TPM counter is a free-running counter, the update is made when the TPM counter changes from 0xFFFE to 0xFFFF

The latching mechanism is manually reset by writing to the TPMxSC address (whether BDM is active or not).

When BDM is active, the coherency mechanism is frozen (unless reset by writing to TPMxSC register) so the buffer latches remain in the state they were in when the BDM became active, even if one or both halves of the modulo register are written while BDM is active. Any write to the modulo registers bypasses the buffer latches and directly writes to the modulo register while BDM is active.

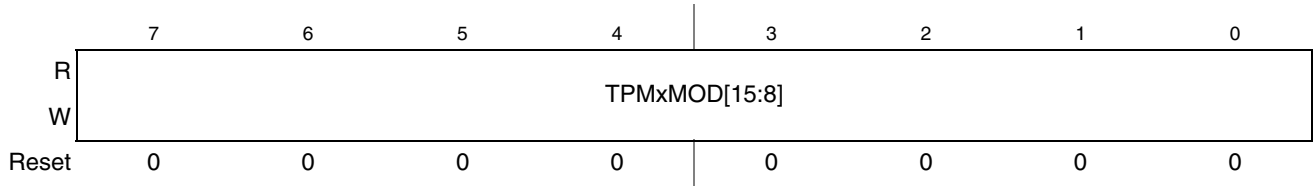


Figure 17-10. TPM Counter Modulo Register High (TPMxMODH)

	7	6	5	4	3	2	1	0
R W	TPMxMOD[7:0]							
Reset	0	0	0	0	0	0	0	0

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

17.3.4 TPM Channel n Status and Control Register (TPMxCnSC)

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

	7	6	5	4	3	2	1	0
R	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	0	0
W	0							
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 17-12. TPM Channel n Status and Control Register (TPMxCnSC)

Table 17-5. TPMxCnSC Field Descriptions

Field	Description
7 CHnF	Channel n flag. When channel n is an input capture channel, this read/write bit is set when an active edge occurs on the channel n input. When channel n is an output compare or edge-aligned/center-aligned PWM channel, CHnF is set when the value in the TPM counter registers matches the value in the TPM channel n value registers. When channel n is an edge-aligned/center-aligned PWM channel and the duty cycle is set to 0% or 100%, CHnF is not set even when the value in the TPM counter registers matches the value in the TPM channel n value registers. A corresponding interrupt is requested when this bit is set and channel n interrupt is enabled (CHnIE = 1). Clear CHnF by reading TPMxCnSC while this bit is set and then writing a logic 0 to it. If another interrupt request occurs before the clearing sequence is completed CHnF remains set. This is done so a CHnF interrupt request is not lost due to clearing a previous CHnF. Reset clears this bit. Writing a logic 1 to CHnF has no effect. 0 No input capture or output compare event occurred on channel n. 1 Input capture or output compare event on channel n.
6 CHnIE	Channel n interrupt enable. This read/write bit enables interrupts from channel n. Reset clears this bit. 0 Channel n interrupt requests disabled (use for software polling). 1 Channel n interrupt requests enabled.
5 MSnB	Mode select B for TPM channel n. When CPWMS is cleared, setting the MSnB bit configures TPM channel n for edge-aligned PWM mode. Refer to the summary of channel mode and setup controls in Table 17-6.

Table 17-5. TPMxCnSC Field Descriptions (continued)

Field	Description
4 MSnA	Mode select A for TPM channel n. When CPWMS and MSnB are cleared, the MSnA bit configures TPM channel n for input capture mode or output compare mode. Refer to Table 17-6 for a summary of channel mode and setup controls. **Note:** If the associated port pin is not stable for at least two bus clock cycles before changing to input capture mode, it is possible to get an unexpected indication of an edge trigger.
3–2 ELSnB ELSnA	Edge/level select bits. Depending upon the operating mode for the timer channel as set by CPWMS:MSnB:MSnA and shown in Table 17-6, these bits select the polarity of the input edge that triggers an input capture event, select the level that is driven in response to an output compare match, or select the polarity of the PWM output. If ELSnB and ELSnA bits are cleared, the channel pin is not controlled by TPM. This configuration can be used by software compare only, because it does not require the use of a pin for the channel.

Table 17-6. Mode, Edge, and Level Selection

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	XX	00	Pin is not controlled by TPM. It is reverted to general purpose I/O or other peripheral control	
0	00	01	Input capture	Capture on rising edge only
		10		Capture on falling edge only
		11		Capture on rising or falling edge
	01	00	Output compare	Software compare only
		01		Toggle output on channel match
		10		Clear output on channel match
		11		Set output on channel match
	1X	10	Edge-aligned PWM	High-true pulses (clear output on channel match)
		X1		Low-true pulses (set output on channel match)
1	XX	10	Center-aligned PWM	High-true pulses (clear output on channel match when TPM counter is counting up)
		X1		Low-true pulses (set output on channel match when TPM counter is counting up)

17.3.5 TPM Channel Value Registers (TPMxCnVH:TPMxCnVL)

These read/write registers contain the captured TPM counter value of the input capture function or the output compare value for the output compare or PWM functions. The channel registers are cleared by reset.

Figure 17-13. TPM Channel Value Register High (TPMxCnVH)

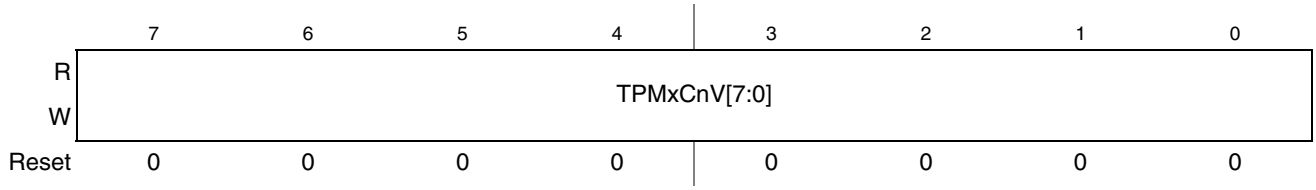


Figure 17-14. TPM Channel Value Register Low (TPMxCnVL)

In input capture mode, reading either byte (TPMxCnVH or TPMxCnVL) latches the contents of both bytes into a buffer where they remain latched until the other half is read. This latching mechanism also resets (becomes unlatched) when the TPMxCnSC register is written (whether BDM mode is active or not). Any write to the channel registers is ignored during the input capture mode.

When BDM is active, the coherency mechanism is frozen (unless reset by writing to TPMxCnSC register) so the buffer latches remain in the state they were in when the BDM became active, even if one or both halves of the channel register are read while BDM is active. This assures that if you were in the middle of reading a 16-bit register when BDM became active, it reads the appropriate value from the other half of the 16-bit value after returning to normal execution. The value read from the TPMxCnVH and TPMxCnVL registers in BDM mode is the value of these registers and not the value of their read buffer.

In output compare or PWM modes, writing to either byte (TPMxCnVH or TPMxCnVL) latches the value into a buffer. After both bytes were written, they are transferred as a coherent 16-bit value into the timer-channel registers according to the value of CLKSB:CLKSA bits and the selected mode:

- If CLKSB and CLKSA are cleared, the registers are updated when the second byte is written.
- If CLKSB and CLKSA are not cleared and in output compare mode, the registers are updated after the second byte is written and on the next change of the TPM counter (end of the prescaler counting).
- If CLKSB and CLKSA are not cleared and in EPWM or CPWM modes, the registers are updated after both bytes were written, and the TPM counter changes from (TPMxMODH:TPMxMODL – 1) to (TPMxMODH:TPMxMODL). If the TPM counter is a free-running counter, the update is made when the TPM counter changes from 0xFFFE to 0xFFFF.

The latching mechanism is manually reset by writing to the TPMxCnSC register (whether BDM mode is active or not). This latching mechanism allows coherent 16-bit writes in either big-endian or little-endian order that is friendly to various compiler implementations.

When BDM is active, the coherency mechanism is frozen so the buffer latches remain in the state they were in when the BDM became active even if one or both halves of the channel register are written while BDM is active. Any write to the channel registers bypasses the buffer latches and directly write to the channel register while BDM is active. The values written to the channel register while BDM is active are used for PWM and output compare operation after normal execution resumes. Writes to the channel registers while BDM is active do not interfere with partial completion of a coherency sequence. After the coherency mechanism is fully exercised, the channel registers are updated using the buffered values (while BDM was not active).

17.4 Functional Description

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

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

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

17.4.1 Counter

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

17.4.1.1 Counter Clock Source

The 2-bit field, CLKSB:CLKSA, in the timer status and control register (TPMxSC) disables the TPM counter or selects one of three clock sources to TPM counter (Table 17-3). After any MCU reset, CLKSB and CLKSA are cleared so no clock is selected and the TPM counter is disabled (TPM is in a very low power state). You can read or write these control bits at any time. Disabling the TPM counter by writing 00 to CLKSB:CLKSA bits, does not affect the values in the TPM counter or other registers.

The fixed frequency clock is an alternative clock source for the TPM counter that allows the selection of a clock other than the bus clock or external clock. This clock input is defined by chip integration. You can refer chip specific documentation for further information. Due to TPM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed the bus clock frequency. The fixed frequency clock has no limitations for low frequency operation.

The external clock passes through a synchronizer clocked by the bus clock to assure that counter transitions are properly aligned to bus clock transitions.Therefore, in order to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the bus clock frequency.

When the external clock source is shared with a TPM channel pin, this pin must not be used in input capture mode. However, this channel can be used in output compare mode with ELSnB:ELSnA = 0:0 for software timing functions. In this case, the channel output is disabled, but the channel match events continue to set the appropriate flag.

17.4.1.2 Counter Overflow and Modulo Reset

An interrupt flag and enable are associated with the 16-bit main counter. The flag (TOF) is a software-accessible indication that the timer counter has overflowed. The enable signal selects between software polling (TOIE = 0) where no interrupt is generated, or interrupt-driven operation (TOIE = 1) where the interrupt is generated whenever the TOF is set.

The conditions causing TOF to become set depend on whether the TPM is configured for center-aligned PWM (CPWMS = 1). If CPWMS is cleared and there is no modulus limit, the 16-bit timer counter counts from 0x0000 through 0xFFFF and overflows to 0x0000 on the next counting clock. TOF is set at the transition from 0xFFFF to 0x0000. When a modulus limit is set, TOF is set at the transition from the value set in the modulus register to 0x0000. When the TPM is in center-aligned PWM mode (CPWMS = 1), the TOF flag is set as the counter changes direction at the end of the count value set in the modulus register (at the transition from the value set in the modulus register to the next lower count value). This corresponds to the end of a PWM period (the 0x0000 count value corresponds to the center of a period).

17.4.1.3 Counting Modes

The main timer counter has two counting modes. When center-aligned PWM is selected (CPWMS = 1), the counter operates in up/down counting mode. Otherwise, the counter operates as a simple up counter. As an up counter, the timer counter counts from 0x0000 through its terminal count and continues with 0x0000. The terminal count is 0xFFFF or a modulus value in TPMxMODH:TPMxMODL.

When center-aligned PWM operation is specified, the counter counts up from 0x0000 through its terminal count and then down to 0x0000 where it changes back to up counting. The terminal count value and 0x0000 are normal length counts (one timer clock period long). In this mode, the timer overflow flag (TOF) is set at the end of the terminal-count period (as the count changes to the next lower count value).

17.4.1.4 Manual Counter Reset

The main timer counter can be manually reset at any time by writing any value to TPMxCNTH or TPMxCNTL. Resetting the counter in this manner also resets the coherency mechanism in case only half of the counter was read before resetting the count.

17.4.2 Channel Mode Selection

If CPWMS is cleared, MSnB and MSnA bits determine the basic mode of operation for the corresponding channel. Choices include input capture, output compare, and edge-aligned PWM.

17.4.2.1 Input Capture Mode

With the input capture function, the TPM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TPM latches the contents of the TPM counter into the channel-value registers (TPMxCnVH:TPMxCnVL). Rising edges, falling edges, or any edge is chosen as the active edge that triggers an input capture.

In input capture mode, the TPMxCnVH and TPMxCnVL registers are read only.

When either half of the 16-bit capture register is read, the other half is latched into a buffer to support coherent 16-bit accesses in big-endian or little-endian order. The coherency sequence can be manually reset by writing to TPMxCnSC.

An input capture event sets a flag bit (CHnF) that optionally generates a CPU interrupt request.

While in BDM, the input capture function works as configured. When an external event occurs, the TPM latches the contents of the TPM counter (frozen because of the BDM mode) into the channel value registers and sets the flag bit.

17.4.2.2 Output Compare Mode

With the output compare function, the TPM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter reaches the value in TPMxCnVH:TPMxCnVL registers of an output compare channel, the TPM can set, clear, or toggle the channel pin.

Writes to any of TPMxCnVH and TPMxCnVL registers actually write to buffer registers. In output compare mode, the TPMxCnVH:TPMxCnVL registers are updated with the value of their write buffer only after both bytes were written and according to the value of CLKSB:CLKSA bits:

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

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

An output compare event sets a flag bit (CHnF) that optionally generates a CPU interrupt request.

17.4.2.3 Edge-Aligned PWM Mode

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

The time between the modulus overflow and the channel match value (TPMxCnVH:TPMxCnVL) is the pulse width or duty cycle (Figure 17-15). If ELSnA is cleared, the counter overflow forces the PWM signal high, and the channel match forces the PWM signal low. If ELSnA is set, the counter overflow forces the PWM signal low, and the channel match forces the PWM signal high.

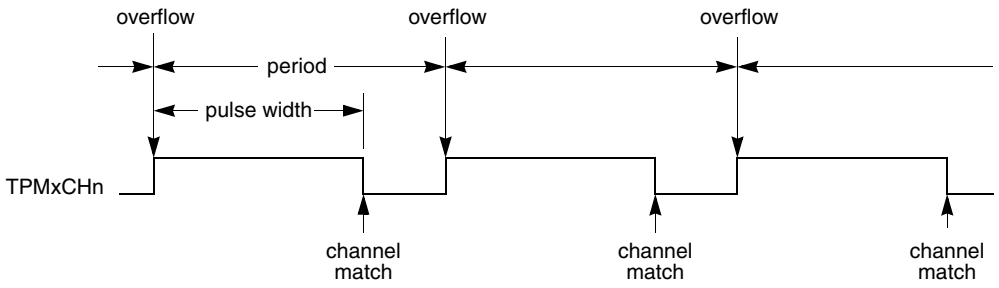


Figure 17-15. EPWM period and pulse width (ELSnA=0)

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

The timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to any of the registers TPMxCnVH and TPMxCnVL actually write to buffer registers. In edge-aligned PWM mode, the TPMxCnVH:TPMxCnVL registers are updated with the value of their write buffer according to the value of CLKSB:CLKSA bits:

- If CLKSB and CLKSA are cleared, the registers are updated when the second byte is written
- If CLKSB and CLKSA are not cleared, the registers are updated after both bytes were written, and the TPM counter changes from (TPMxMODH:TPMxMODL – 1) to (TPMxMODH:TPMxMODL). If the TPM counter is a free-running counter, the update is made when the TPM counter changes from 0xFFFE to 0xFFFF.

17.4.2.4 Center-Aligned PWM Mode

This type of PWM output uses the up/down counting mode of the timer counter (CPWMS=1). The channel match value in TPMxCnVH:TPMxCnVL determines the pulse width (duty cycle) of the PWM signal while the period is determined by the value in TPMxMODH:TPMxMODL. TPMxMODH:TPMxMODL must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results. ELSnA determines the polarity of the CPWM signal.

pulse width = 2 × (TPMxCnVH:TPMxCnVL)

period = 2 × (TPMxMODH:TPMxMODL); TPMxMODH:TPMxMODL = 0x0001–0x7FFF

If TPMxCnVH:TPMxCnVL is zero or negative (bit 15 set), the duty cycle is 0%. If TPMxCnVH:TPMxCnVL is a positive value (bit 15 clear) and is greater than the non-zero modulus setting, the duty cycle is 100% because the channel match never occurs. This implies the usable range of periods set by the modulus register is 0x0001 through 0x7FFE (0x7FFF if you do not need to generate 100% duty cycle). This is not a significant limitation. The resulting period is much longer than required for normal applications.

All zeros in TPMxMODH:TPMxMODL is a special case that must not be used with center-aligned PWM mode. When CPWMS is cleared, this case corresponds to the counter running free from 0x0000 through 0xFFFF. When CPWMS is set, the counter needs a valid match to the modulus register somewhere other than at 0x0000 in order to change directions from up-counting to down-counting.

The channel match value in the TPM channel registers (times two) determines the pulse width (duty cycle) of the CPWM signal (Figure 17-16). If ELSnA is cleared, a channel match occurring while counting up clears the CPWM output signal and a channel match occurring while counting down sets the output. The counter counts up until it reaches the modulo setting in TPMxMODH:TPMxMODL, then counts down until it reaches zero. This sets the period equal to two times TPMxMODH:TPMxMODL.

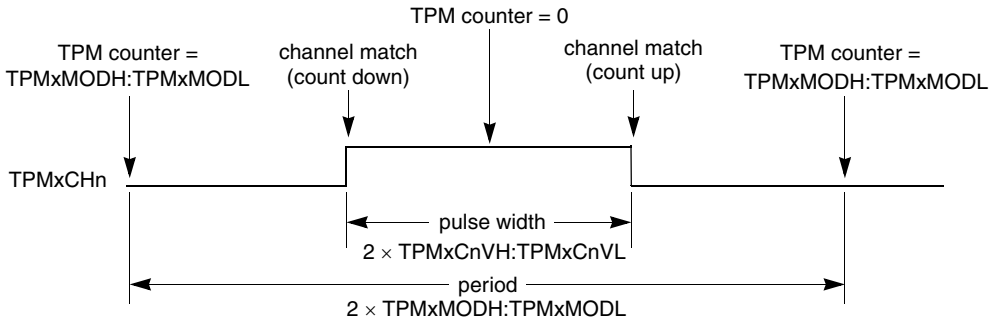


Figure 17-16. CPWM period and pulse width (ELSnA=0)

Center-aligned PWM outputs typically produce less noise than edge-aligned PWMs because fewer I/O pin transitions are lined up at the same system clock edge. This type of PWM is also required for some types of motor drives.

Input capture, output compare, and edge-aligned PWM functions do not make sense when the counter is operating in up/down counting mode so this implies that all active channels within a TPM must be used in CPWM mode when CPWMS is set.

The timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to any of the registers TPMxCnVH and TPMxCnVL actually write to buffer registers. In center-aligned PWM mode, the TPMxCnVH:TPMxCnVL registers are updated with the value of their write buffer according to the value of CLKSB:CLKSA bits:

- If CLKSB and CLKSA are cleared, the registers are updated when the second byte is written
- If CLKSB and CLKSA are not cleared, the registers are updated after both bytes were written, and the TPM counter changes from (TPMxMODH:TPMxMODL – 1) to (TPMxMODH:TPMxMODL). If the TPM counter is a free-running counter, the update is made when the TPM counter changes from 0xFFFE to 0xFFFF.

When TPMxCNTH:TPMxCNTL equals TPMxMODH:TPMxMODL, the TPM can optionally generate a TOF interrupt (at the end of this count).

17.5 Reset Overview

17.5.1 General

The TPM is reset whenever any MCU reset occurs.

17.5.2 Description of Reset Operation

Reset clears TPMxSC that disables TPM counter clock and overflow interrupt (TOIE=0). CPWMS, MSnB, MSnA, ELSnB, and ELSnA are all cleared. This configures all TPM channels for input capture operation and the associated pins are not controlled by TPM.

17.6 Interrupts

17.6.1 General

The TPM generates an optional interrupt for the main counter overflow and an interrupt for each channel. The meaning of channel interrupts depends on each channel's mode of operation. If the channel is configured for input capture, the interrupt flag is set each time the selected input capture edge is recognized. If the channel is configured for output compare or PWM modes, the interrupt flag is set each time the main timer counter matches the value in the 16-bit channel value register.

All TPM interrupts are listed in Table 17-7.

Table 17-7. Interrupt Summary

Interrupt	Local Enable	Source	Description
TOF	TOIE	Counter overflow	Set each time the TPM counter reaches its terminal count (at transition to its next count value)
CHnF	CHnIE	Channel event	An input capture event or channel match took place on channel n

The TPM module provides high-true interrupt signals.

17.6.2 Description of Interrupt Operation

For each interrupt source in the TPM, a flag bit is set upon recognition of the interrupt condition such as timer overflow, channel input capture, or output compare events. This flag is read (polled) by software to determine that the action has occurred, or an associated enable bit (TOIE or CHnIE) can be set to enable the interrupt generation. While the interrupt enable bit is set, the interrupt is generated whenever the associated interrupt flag is set. Software must perform a sequence of steps to clear the interrupt flag before returning from the interrupt-service routine.

TPM interrupt flags are cleared by a two-step process including a read of the flag bit while it is set followed by a write of zero to the bit. If a new event is detected between these two steps, the sequence is reset and the interrupt flag remains set after the second step to avoid the possibility of missing the new event.

17.6.2.1 Timer Overflow Interrupt (TOF) Description

The meaning and details of operation for TOF interrupts varies slightly depending upon the mode of operation of the TPM system (general purpose timing functions versus center-aligned PWM operation). The flag is cleared by the two step sequence described above.

17.6.2.1.1 Normal Case

When CPWMS is cleared, TOF is set when the timer counter changes from the terminal count (the value in the modulo register) to 0x0000. If the TPM counter is a free-running counter, the update is made when the TPM counter changes from 0xFFFF to 0x0000.

17.6.2.1.2 Center-Aligned PWM Case

When CPWMS is set, TOF is set when the timer counter changes direction from up-counting to down-counting at the end of the terminal count (the value in the modulo register).

17.6.2.2 Channel Event Interrupt Description

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

17.6.2.2.1 Input Capture Events

When a channel is configured as an input capture channel, the ELSnB:ELSnA bits select if channel pin is not controlled by TPM, rising edges, falling edges, or any edge as the edge that triggers an input capture event. When the selected edge is detected, the interrupt flag is set. The flag is cleared by the two-step sequence described in Section 17.6.2, "Description of Interrupt Operation."

17.6.2.2.2 Output Compare Events

When a channel is configured as an output compare channel, the interrupt flag is set each time the main timer counter matches the 16-bit value in the channel value register. The flag is cleared by the two-step sequence described in Section 17.6.2, "Description of Interrupt Operation."

17.6.2.2.3 PWM End-of-Duty-Cycle Events

When the channel is configured for edge-aligned PWM, the channel flag is set when the timer counter matches the channel value register that marks the end of the active duty cycle period. When the channel is configured for center-aligned PWM, the timer count matches the channel value register twice during each PWM cycle. In this CPWM case, the channel flag is set at the start and at the end of the active duty cycle period when the timer counter matches the channel value register. The flag is cleared by the two-step sequence described in Section 17.6.2, "Description of Interrupt Operation."

Chapter 18
Voltage Reference Module (S08VREFV1)

18.1 Introduction

The voltage reference is intended to supply an accurate voltage output that is trimmable by an 8-bit register in 0.5 mV steps.

This voltage reference can be used in medical applications such as glucose meters to provide a reference voltage to biosensors or as a reference to analog peripherals such as the ADC, DAC, or ACMP. The voltage reference has three operating modes that provide different levels of load regulation and power consumption.

NOTE

Write the same values to both VREF modules to make code compatible with either package that the firmware initialize both VREF modules.

The MC9S08LH64 contains two Voltage Reference modules; VREF1 and VREF2, each with their own control and trim registers.

Figure 18-1 shows the MC9S08LH64 series block diagram with the VREF highlighted.

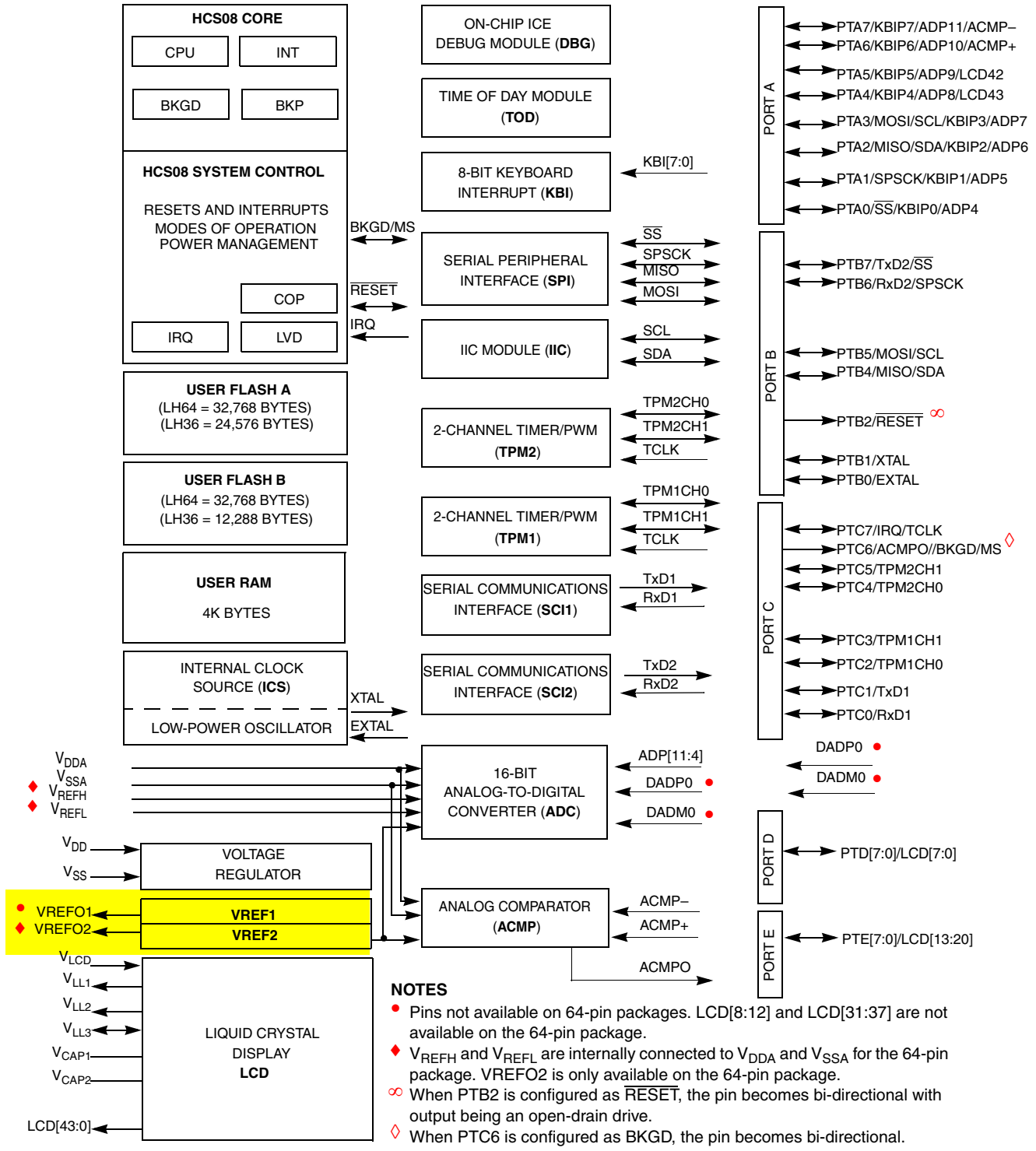
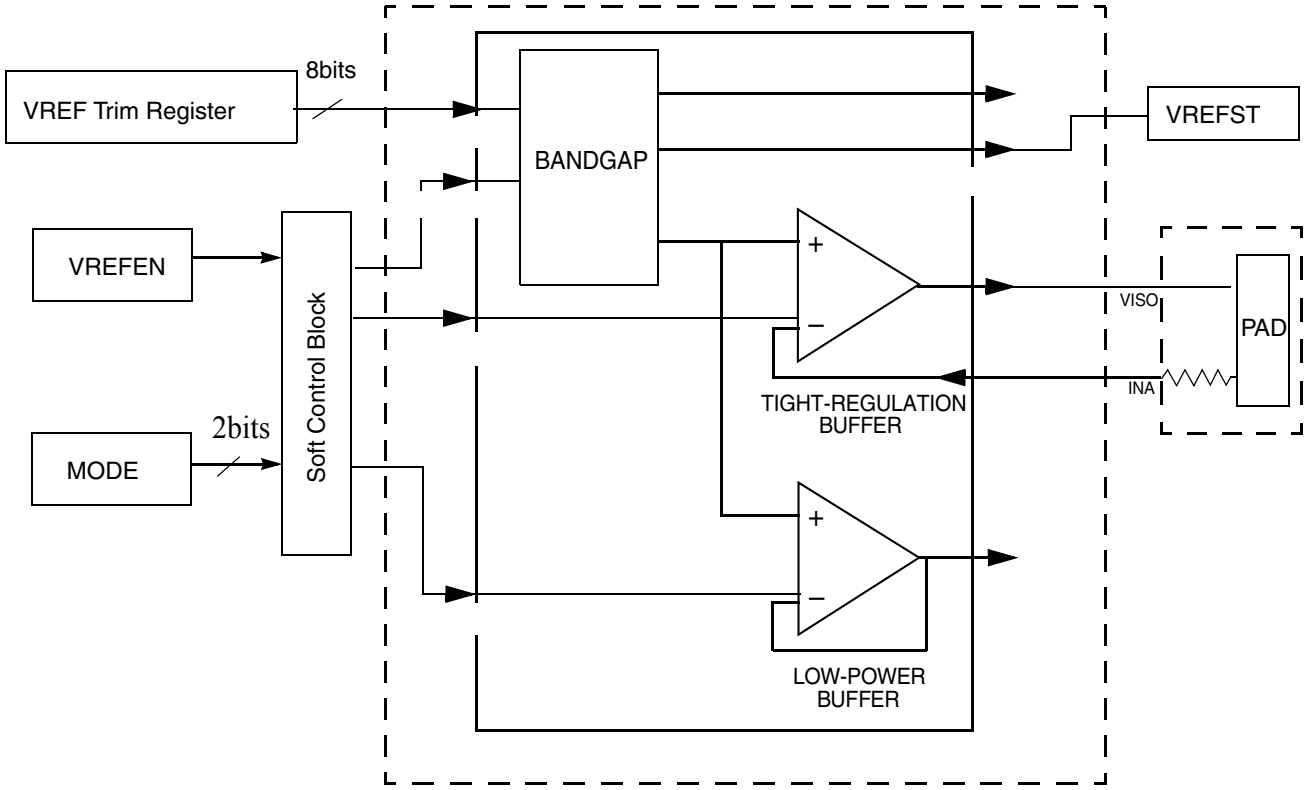


Figure 18-1. MC9S08LH64 Series Block Diagram Highlighting VREF Blocks and Pins

18.1.1 Overview

The voltage reference is an optimization of a ICS bandgap and bandgap buffer system. Unity gain amplifiers are used to ease the design and keep power consumption low. The 8-bit trim register is accessible by the ip bus so that the voltage reference can be trimmed in application.



1 in tight regulation mode, one 100 nf cap must be provided between vrefo and ground.

2 Low power buffer's output also connects to vrefo pad.

Figure 18-2. Voltage Reference Block Diagram

The voltage reference operates from 1.8 V to 3.6 V and its operation maximum current consumption is between 80 μA and 1.15 mA (unloaded) depending on the mode used. See the *MC9S08LH64 Data Sheet* for more information. The Voltage Reference contains trim resolution of 0.5 mV per step. The buffer has modes with improved high-current swing output in order to achieve between 20 μV/mA and 100 μV/mA load regulation. In addition, a low-current buffer mode is available for use with ADCs or DACs. The voltage reference can be output to a dedicated output pin.

18.1.2 Features

The voltage reference module has the following features:

- Programmable trim register with 0.5 mV steps, automatically loaded with room temp value upon reset

- Programmable mode selection:
  - Off
  - Bandgap out (or stabilization delay)
  - Low-power buffer mode
  - Tight-regulation buffer mode
- 1.2 V output at room temperature, 40 ppm/C
- Dedicated output pin VREFO
- Load Regulation in tight-regulation mode of 100 uV/mA max
- PSR of ± .1 mVDC and -60 dB AC

18.1.3 VREF Configuration Information

The MC9S08LH64 series use the VREF module to provide the bandgap signal. The bandgap signal is provided to the ADC and ACMP peripherals.

18.1.4 VREF Clock Gating

The bus clock to the VREF can be gated on and off using the VREF bit in SCGC1. This bit is set after any reset, which enables the bus clock to this module. To conserve power, this bit can be cleared to disable the clock to this module when not in use. See Section 5.7, “Peripheral Clock Gating,” for details.

18.1.5 VREF Enable

Only one of the two VREF modules is enabled after reset:

- 80-pin packages — VREF1 enabled and VREF2 disabled.
- 64-pin packages — VREF1 disabled and VREF2 enabled.

NOTE

If a VREF module is disabled, the registers associated with that module are not in the memory map.

18.1.6 VREF Output

The VREF module is output to a pin. Which VREF module outputs to the pin depends upon the package being used:

- 80-pin packages — VREF1 is output.
- 64-pin packages — VREF2 is output.

18.1.7 Overview

The Voltage Reference optimizes the existing bandgap and bandgap buffer system. Unity gain amplifiers ease the design and keep power consumption low. The 8-bit trim register is user accessible so that the voltage reference can be trimmed in application.

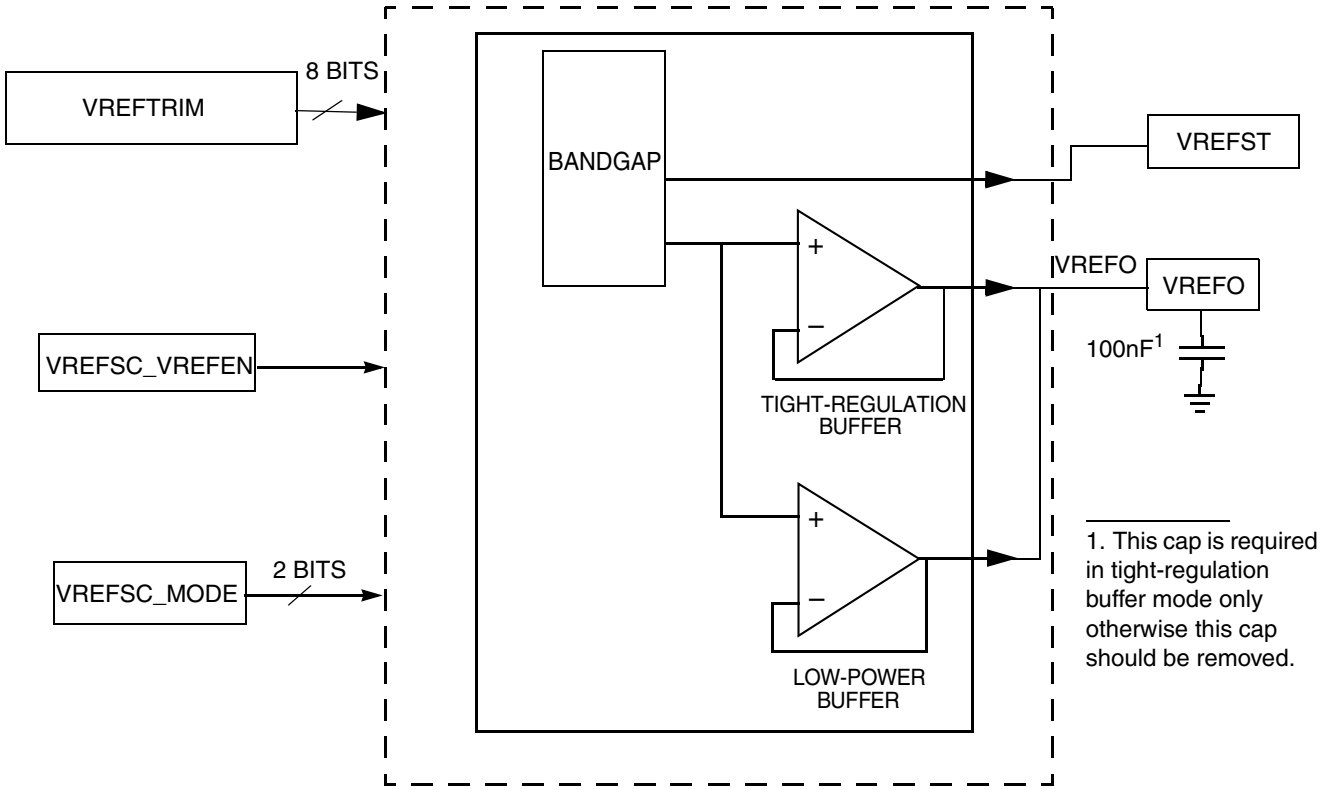


Figure 18-3. Voltage Reference Block Diagram

The voltage reference contains trim resolution of 0.5mV per step. The buffer has modes with improved high-current swing output. In addition, a low-power buffer mode is available for use with ADCs or DACs. The voltage reference can be output on a dedicated output pin[1].

DEDICATED OUTPUT PIN

18.1.8 Features

The Voltage Reference module has the following features:

- Programmable trim register with 0.5mV steps, automatically loaded with factory trimmed value upon reset
- Programmable mode selection:
  - — Off
  - — Bandgap out (or stabilization delay)
  - — Low-power buffer mode

1. In Tight-Regulation buffer mode, a 100nF capacitor is required to be connected between the VREFO pad and the ground.

— Tight-regulation buffer mode
- 1.2 V output at room temperature, 40 ppm/C
- Dedicated output pin VREFO
- Load regulation in tight-regulation mode of 100 uV/mA max
- PSR of 0±0.1 mV DC and -60dB AC

18.1.9 Modes of Operation

The Voltage Reference continues normal operation in Run, Wait, LPRun, and LPWait modes. The Voltage Reference module can be enabled to operate in STOP3 mode.

18.1.10 External Signal Description

Table 18-1 shows the Voltage Reference signals properties.

Table 18-1. Signal Properties

Name	Function	I/O	Reset	Pull Up
VREFO	Internally generated voltage reference output	O	—	—

18.2 Memory Map and Register Definition

18.2.1 VREF Trim Register (VREFTRM)

The VREFTRM register contains eight bits that contain the trim data for the Voltage Reference as described in Table 18-2.

Register address: Base + 0x00

	7	6	5	4	3	2	1	0
R W	TRM[7:0]							
Reset:	X	X	X	X	X	X	X	X

Figure 18-4. VREF Trim Register (VREFTRM)

Table 18-2. VREFTRM Register Field Descriptions

Field	Description
7:0 TRM[7:0]	Trim Bits 7:0 These bits change the resulting VREF output by ±0.5mV for each step.

Table 18-3. VREFTRM Register Settings

TRM7	TRM6	TRM5	TRM4	TRM3	TRM2	TRM1	TRM0	VREF (mV)
1	0	0	0	0	0	0	0	max
1	0	0	0	0	0	0	1	max-0.5
1	0	0	0	0	0	1	0	max-1.0
1	0	0	0	0	0	1	1	max-1.5
1	. . . . .							. . . . .
1	1	1	1	1	1	0	0	mid+1.5
1	1	1	1	1	1	0	1	mid+1.0
1	1	1	1	1	1	1	0	mid+0.5
1	1	1	1	1	1	1	1	mid
0	0	0	0	0	0	0	0	mid-0.5
0	0	0	0	0	0	0	1	mid-1.0
0	0	0	0	0	0	1	0	mid-1.5
0	0	0	0	0	0	1	1	mid-2.0
0	. . . . .							. . . . .
0	1	1	1	1	1	0	0	min+1.5
0	1	1	1	1	1	0	1	min+1.0
0	1	1	1	1	1	1	0	min+0.5
0	1	1	1	1	1	1	1	min

18.2.2 VREF Status and Control Register (VREFSC)

The VREF status and control register contains the control bits used to enable the internal voltage reference and select the buffer mode to be used.

Register address: Base + 0x01

	7	6	5	4	3	2	1	0
R	VREFEN	0	0	0	0	VREFST	MODE	
W								
Reset:	0	0	0	0	0	0	0	0

Figure 18-5. Voltage Reference Control Register (VREFSC)

Table 18-4. VREFSC Register Field Descriptions

Field	Description
7 VREFEN	**Internal Voltage Reference Enable** — This bit is used to enable the Voltage Reference module. 0 the Voltage Reference module is disabled 1 the Voltage Reference module is enabled
2 VREFST	**Internal Voltage Reference Stable** — This bit indicates that the Voltage Reference module has completed its startup and stabilization. 0 Voltage Reference module is disabled or not stable 1 Voltage Reference module is stable
1:0 MODE[1:0]	**Mode selection** — These bits are used to select the modes of operation for the Voltage Reference module. 00 Bandgap on only, for stabilization and startup 01 Low-power buffer enabled 10 Tight-regulation buffer enabled 11 RESERVED

18.3 Functional Description

The Voltage Reference is a bandgap buffer system. Unity gain amplifiers are used.

The Voltage Reference can be used in two main cases. It can be used as a reference to analog peripherals such as an ADC channel or analog comparator input. For this case, the low-power buffer can be used. When the tight-regulation buffer is enabled, VREFO can be used both internally and externally.

Table 18-5 shows all possible functional configurations of the Voltage Reference.

Table 18-5. Voltage Reference Functional Configurations

VREFEN	MODE[1:0]	Configuration	Functionality
0	X	Voltage Reference Disabled	Off
1	00	Voltage Reference Enabled, Bandgap on only	Startup and Standby
1	01	Voltage Reference Enabled, Low-Power buffer on	Can be used for internal peripherals only and VREFO pin should not be loaded
1	10	Voltage Reference Enabled, Tight-Regulation buffer on	Can be used both internally and externally. A 100nF capacitor is required on VREFO and 10mA max drive strength is allowed.
1	11	Voltage Reference Disabled	RESERVED

18.3.1 Voltage Reference Disabled, VREFEN=0

When VREFEN=0, the Voltage Reference is disabled, all bandgap and buffers are disabled. The Voltage Reference is in off mode.

18.3.2 Voltage Reference Enabled, VREFEN=1

When VREFEN=1, the Voltage Reference is enabled, and different modes should be set by the Mode[1:0] bits.

18.3.2.1 Mode[1:0]=00

The internal bandgap is on to generate an accurate voltage output that can be trimmed by the bits TRM[7:0] in 0.5 mV steps. The bandgap requires some time for startup and stabilization. The VREFST bit can be monitored to determine if the stabilization and startup is complete.

Both low-power buffer and tight-regulation buffer are disabled in this mode, and there is no buffered voltage output. The Voltage Reference is in standby mode.

18.3.2.2 Mode[1:0]=01

The internal bandgap is on.

The low-power buffer is enabled to generate a buffered internal voltage. It can be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

18.3.2.3 Mode[1:0]=10

The internal bandgap is on.

The tight-regulation buffer is enabled to generate a buffered voltage to VREFO with load regulation less than 100 uV/mA. A 100nF capacitor is required on VREFO pin and it is allowed to drive 10 mA maximum current. VREFO can be used internally and/or externally.

18.3.2.4 Mode[1:0]=11

RESERVED

18.4 Initialization Information

The Voltage Reference requires some time for startup and stabilization. Once the VREFEN bit is set, the VREFST bit can be monitored to determine if the stabilization and startup is complete.

When the Voltage Reference is already enabled and stabilized, changing the Mode selection bits (MODE[1:0]) will not clear the VREFST bit, but there will be some startup time before the output voltage is stabilized when the low-power buffer or tight-regulation buffer is enabled, and there will be some setting time when a step change of the load current occurs.

Chapter 19
Development Support

19.1 Introduction

Development support systems in the HCS08 include the background debug controller (BDC) and the on-chip debug module (DBG). The BDC provides a single-wire debug interface to the target MCU that provides a convenient interface for programming the on-chip Flash and other nonvolatile memories. The BDC is also the primary debug interface for development and allows non-intrusive access to memory data and traditional debug features such as CPU register modify, breakpoints, and single instruction trace commands.

In the HCS08 Family, address and data bus signals are not available on external pins. Debug is done through commands fed into the target MCU via the single-wire background debug interface. The debug module provides a means to selectively trigger and capture bus information so an external development system can reconstruct what happened inside the MCU on a cycle-by-cycle basis without having external access to the address and data signals.

19.1.1 Forcing Active Background

The method for forcing active background mode depends on the specific HCS08 derivative. For the MC9S08LH64 Series, you can force active background after a power-on reset by holding the BKGD pin low as the device exits the reset condition. You can also force active background by driving BKGD low immediately after a serial background command that writes a one to the BDFR bit in the SBDFR register. Other causes of reset, including an external pin reset or an internally generated error reset ignore the state of the BKGD pin and reset into normal user mode. If no debug pod is connected to the BKGD pin, the MCU will always reset into normal operating mode.

19.1.2 Module Configuration

The alternate BDC clock source is the ICSLCLK. This clock source is selected by clearing the CLKSW bit in the BDCSCR register. For details on ICSLCLK, see the “Functional Description” section of the ICS chapter.

19.1.3 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

19.2 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.

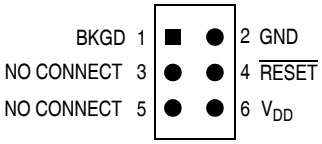


Figure 19-1. BDM Tool Connector

19.2.1 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 19.2.2, "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 19.2.2, "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.

19.2.2 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

when this timeout occurs is aborted without affecting the memory or operating mode of the target MCU system.

The custom serial protocol requires the debug pod to know the target BDC communication clock speed.

The clock switch (CLKSW) control bit in the BDC status and control register allows the user to select the BDC clock source. The BDC clock source can either be the bus or the alternate BDC clock source.

The BKGD pin can receive a high or low level or transmit a high or low level. The following diagrams show timing for each of these cases. Interface timing is synchronous to clocks in the target BDC, but asynchronous to the external host. The internal BDC clock signal is shown for reference in counting cycles.

Figure 19-2 shows an external host transmitting a logic 1 or 0 to the BKGD pin of a target HCS08 MCU. The host is asynchronous to the target so there is a 0-to-1 cycle delay from the host-generated falling edge to where the target perceives the beginning of the bit time. Ten target BDC clock cycles later, the target senses the bit level on the BKGD pin. Typically, the host actively drives the pseudo-open-drain BKGD pin during host-to-target transmissions to speed up rising edges. Because the target does not drive the BKGD pin during the host-to-target transmission period, there is no need to treat the line as an open-drain signal during this period.

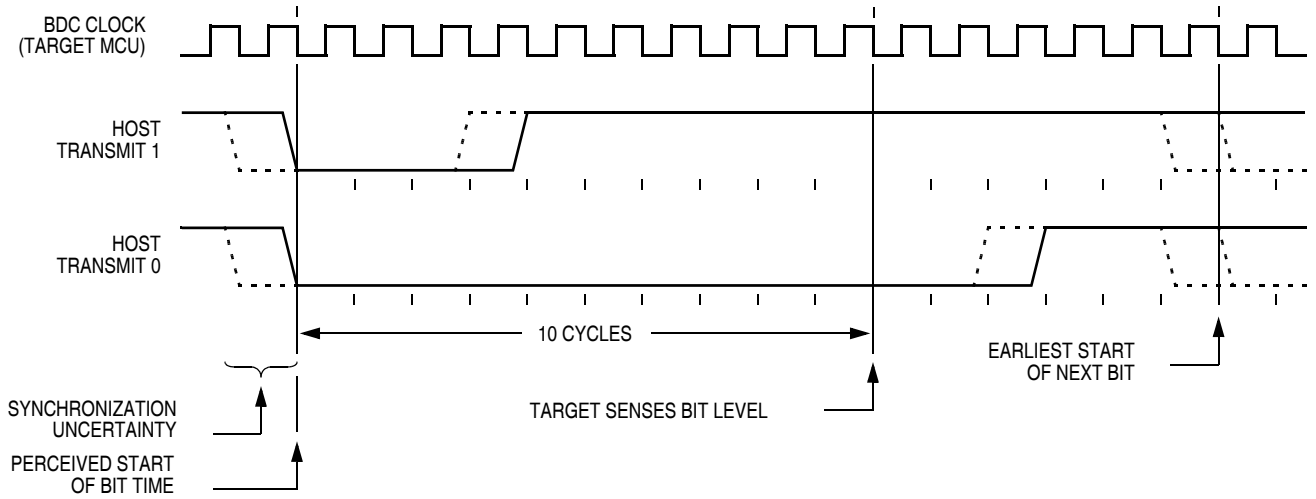


Figure 19-2. BDC Host-to-Target Serial Bit Timing

Figure 19-3 shows the host receiving a logic 1 from the target HCS08 MCU. Because the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target MCU. The host holds the BKGD pin low long enough for the target to recognize it (at least two target BDC cycles). The host must release the low drive before the target MCU drives a brief active-high speedup pulse seven cycles after the perceived start of the bit time. The host should sample the bit level about 10 cycles after it started the bit time.

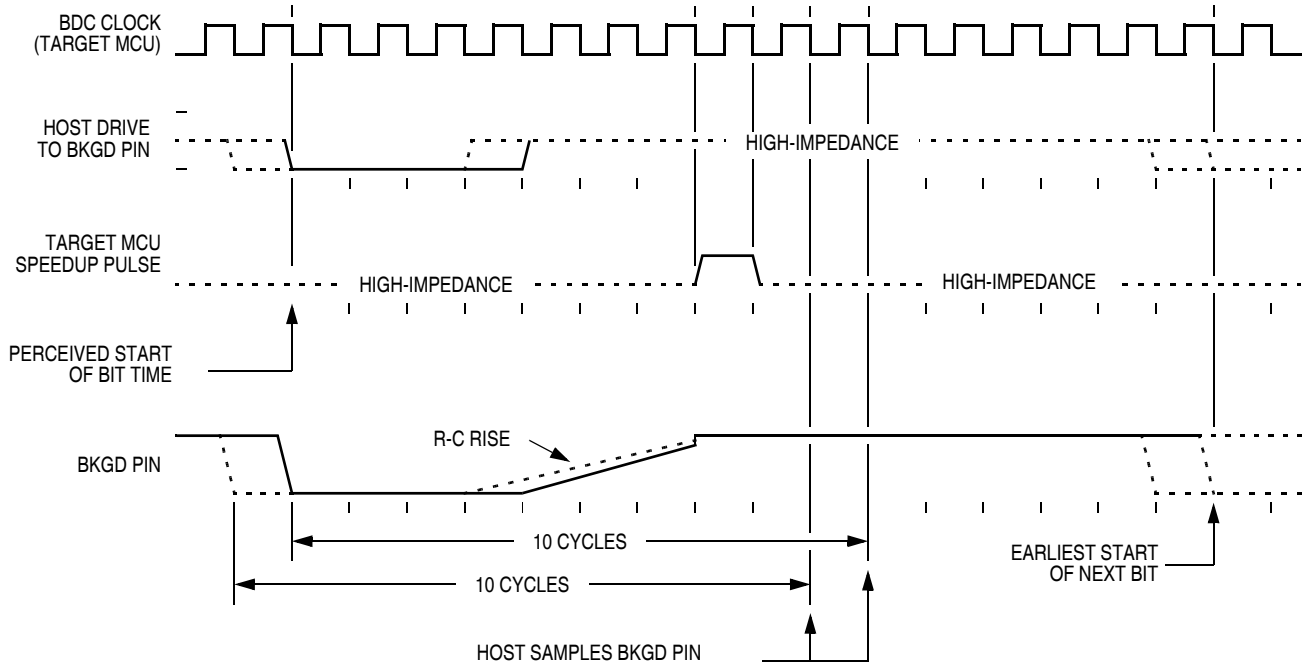


Figure 19-3. BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 19-4 shows the host receiving a logic 0 from the target HCS08 MCU. Because the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target MCU. The host initiates the bit time but the target HCS08 finishes it. Because the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 BDC clock cycles, then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 cycles after starting the bit time.

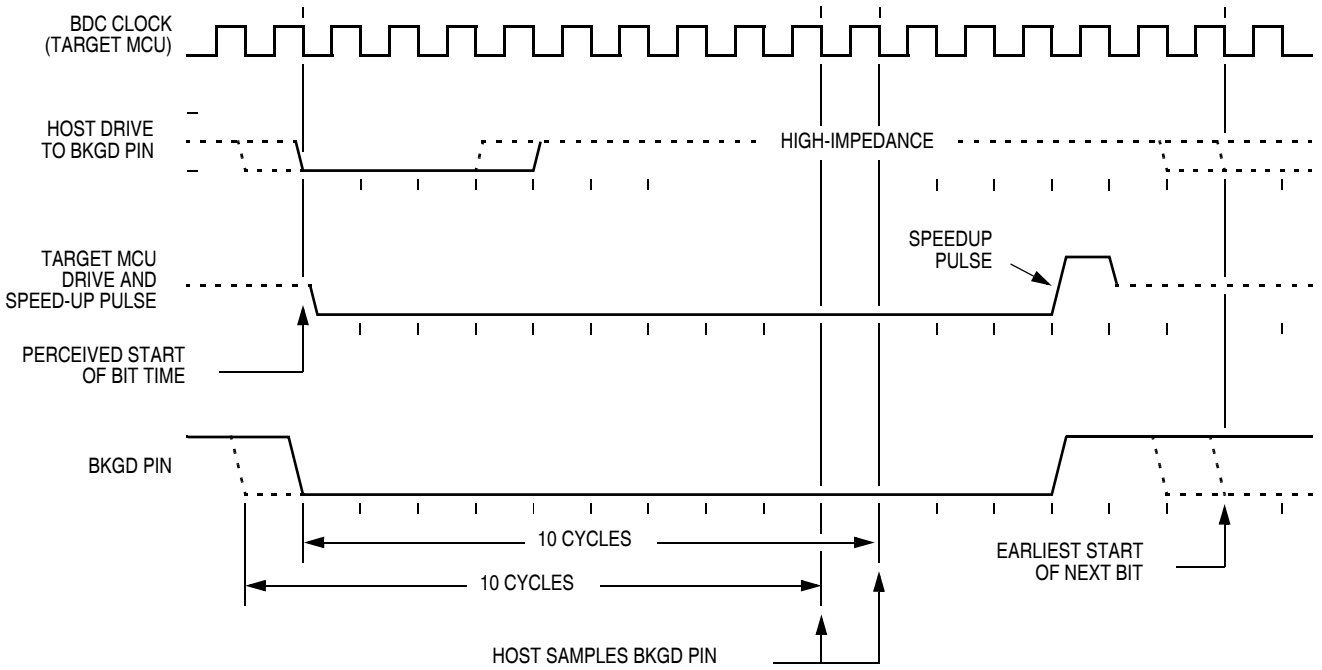


Figure 19-4. BDM Target-to-Host Serial Bit Timing (Logic 0)

19.2.3 BDC Commands

BDC commands are sent serially from a host computer to the BKGD pin of the target HCS08 MCU. All commands and data are sent MSB-first using a custom BDC communications protocol. Active background mode commands require that the target MCU is currently in the active background mode while non-intrusive commands may be issued at any time whether the target MCU is in active background mode or running a user application program.

Table 19-1 shows all HCS08 BDC commands, a shorthand description of their coding structure, and the meaning of each command.

Coding Structure Nomenclature

This nomenclature is used in Table 19-1 to describe the coding structure of the BDC commands.

Commands begin with an 8-bit hexadecimal command code in the host-to-target direction (most significant bit first)

/ = separates parts of the command

d = delay 16 target BDC clock cycles

AAAA = a 16-bit address in the host-to-target direction

RD = 8 bits of read data in the target-to-host direction

WD = 8 bits of write data in the host-to-target direction

RD16 = 16 bits of read data in the target-to-host direction

WD16 = 16 bits of write data in the host-to-target direction

SS = the contents of BDCSCR in the target-to-host direction (STATUS)

CC = 8 bits of write data for BDCSCR in the host-to-target direction (CONTROL)

RBKP = 16 bits of read data in the target-to-host direction (from BDCBKPT breakpoint register)

WBKP = 16 bits of write data in the host-to-target direction (for BDCBKPT breakpoint register)

Table 19-1. BDC Command Summary

Command Mnemonic	Active BDM/ Non-intrusive	Coding Structure	Description
SYNC	Non-intrusive	n/a[1]	Request a timed reference pulse to determine target BDC communication speed
ACK_ENABLE	Non-intrusive	D5/d	Enable acknowledge protocol. Refer to Freescale document order no. HCS08RMv1/D.
ACK_DISABLE	Non-intrusive	D6/d	Disable acknowledge protocol. Refer to Freescale document order no. HCS08RMv1/D.
BACKGROUND	Non-intrusive	90/d	Enter active background mode if enabled (ignore if ENBDM bit equals 0)
READ_STATUS	Non-intrusive	E4/SS	Read BDC status from BDCSCR
WRITE_CONTROL	Non-intrusive	C4/CC	Write BDC controls in BDCSCR
READ_BYTE	Non-intrusive	E0/AAAA/d/RD	Read a byte from target memory
READ_BYTE_WS	Non-intrusive	E1/AAAA/d/SS/RD	Read a byte and report status
READ_LAST	Non-intrusive	E8/SS/RD	Re-read byte from address just read and report status
WRITE_BYTE	Non-intrusive	C0/AAAA/WD/d	Write a byte to target memory
WRITE_BYTE_WS	Non-intrusive	C1/AAAA/WD/d/SS	Write a byte and report status
READ_BKPT	Non-intrusive	E2/RBKP	Read BDCBKPT breakpoint register
WRITE_BKPT	Non-intrusive	C2/WBKP	Write BDCBKPT breakpoint register
GO	Active BDM	08/d	Go to execute the user application program starting at the address currently in the PC

Table 19-1. BDC Command Summary (continued)

Command Mnemonic	Active BDM/ Non-intrusive	Coding Structure	Description
TRACE1	Active BDM	10/d	Trace 1 user instruction at the address in the PC, then return to active background mode
TAGGO	Active BDM	18/d	Same as GO but enable external tagging (HCS08 devices have no external tagging pin)
READ_A	Active BDM	68/d/RD	Read accumulator (A)
READ_CCR	Active BDM	69/d/RD	Read condition code register (CCR)
READ_PC	Active BDM	6B/d/RD16	Read program counter (PC)
READ_HX	Active BDM	6C/d/RD16	Read H and X register pair (H:X)
READ_SP	Active BDM	6F/d/RD16	Read stack pointer (SP)
READ_NEXT	Active BDM	70/d/RD	Increment H:X by one then read memory byte located at H:X
READ_NEXT_WS	Active BDM	71/d/SS/RD	Increment H:X by one then read memory byte located at H:X. Report status and data.
WRITE_A	Active BDM	48/WD/d	Write accumulator (A)
WRITE_CCR	Active BDM	49/WD/d	Write condition code register (CCR)
WRITE_PC	Active BDM	4B/WD16/d	Write program counter (PC)
WRITE_HX	Active BDM	4C/WD16/d	Write H and X register pair (H:X)
WRITE_SP	Active BDM	4F/WD16/d	Write stack pointer (SP)
WRITE_NEXT	Active BDM	50/WD/d	Increment H:X by one, then write memory byte located at H:X
WRITE_NEXT_WS	Active BDM	51/WD/d/SS	Increment H:X by one, then write memory byte located at H:X. Also report status.

[1] The SYNC command is a special operation that does not have a command code.

The SYNC command is unlike other BDC commands because the host does not necessarily know the correct communications speed to use for BDC communications until after it has analyzed the response to the SYNC command.

To issue a SYNC command, the host:

- Drives the BKGD pin low for at least 128 cycles of the slowest possible BDC clock (The slowest clock is normally the reference oscillator/64 or the self-clocked rate/64.)
- Drives BKGD high for a brief speedup pulse to get a fast rise time (This speedup pulse is typically one cycle of the fastest clock in the system.)
- Removes all drive to the BKGD pin so it reverts to high impedance

- Monitors the BKGD pin for the sync response pulse

The target, upon detecting the SYNC request from the host (which is a much longer low time than would ever occur during normal BDC communications):

- Waits for BKGD to return to a logic high
- Delays 16 cycles to allow the host to stop driving the high speedup pulse
- Drives BKGD low for 128 BDC clock cycles
- Drives a 1-cycle high speedup pulse to force a fast rise time on BKGD
- Removes all drive to the BKGD pin so it reverts to high impedance

The host measures the low time of this 128-cycle sync response pulse and determines the correct speed for subsequent BDC communications. Typically, the host can determine the correct communication speed within a few percent of the actual target speed and the communication protocol can easily tolerate speed errors of several percent.

19.2.4 BDC Hardware Breakpoint

The BDC includes one relatively simple hardware breakpoint that compares the CPU address bus to a 16-bit match value in the BDCBKPT register. This breakpoint can generate a forced breakpoint or a tagged breakpoint. A forced breakpoint causes the CPU to enter active background mode at the first instruction boundary following any access to the breakpoint address. The tagged breakpoint causes the instruction opcode at the breakpoint address to be tagged so that the CPU will enter active background mode rather than executing that instruction if and when it reaches the end of the instruction queue. This implies that tagged breakpoints can only be placed at the address of an instruction opcode while forced breakpoints can be set at any address.

The breakpoint enable (BKPTEN) control bit in the BDC status and control register (BDCSCR) is used to enable the breakpoint logic (BKPTEN = 1). When BKPTEN = 0, its default value after reset, the breakpoint logic is disabled and no BDC breakpoints are requested regardless of the values in other BDC breakpoint registers and control bits. The force/tag select (FTS) control bit in BDCSCR is used to select forced (FTS = 1) or tagged (FTS = 0) type breakpoints.

19.3 Register Definition

This section contains the descriptions of the BDC registers and control bits.

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.

19.3.1 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.

19.3.1.1 BDC Status and Control Register (BDCSCR)

This register can be read or written by serial BDC commands (READ_STATUS and WRITE_CONTROL) but is not accessible to user programs because it is not located in the normal memory map of the MCU.

	7	6	5	4	3	2	1	0
R	ENBDM	BDMACT	BKPTEN	FTS	CLKSW	WS	WSF	DVF
W								
Normal Reset	0	0	0	0	0	0	0	0
Reset in Active BDM:	1	1	0	0	1	0	0	0

= Unimplemented or Reserved

Figure 19-5. BDC Status and Control Register (BDCSCR)

Table 19-2. BDCSCR Register Field Descriptions

Field	Description
7 ENBDM	**Enable BDM (Permit Active Background Mode)** — Typically, this bit is written to 1 by the debug host shortly after the beginning of a debug session or whenever the debug host resets the target and remains 1 until a normal reset clears it. 0 BDM cannot be made active (non-intrusive commands still allowed) 1 BDM can be made active to allow active background mode commands
6 BDMACT	**Background Mode Active Status** — This is a read-only status bit. 0 BDM not active (user application program running) 1 BDM active and waiting for serial commands
5 BKPTEN	**BDC Breakpoint Enable** — If this bit is clear, the BDC breakpoint is disabled and the FTS (force tag select) control bit and BDCBKPT match register are ignored. 0 BDC breakpoint disabled 1 BDC breakpoint enabled

Table 19-2. BDCSCR Register Field Descriptions (continued)

Field	Description
4 FTS	**Force/Tag Select** — When FTS = 1, a breakpoint is requested whenever the CPU address bus matches the BDCBKPT match register. When FTS = 0, a match between the CPU address bus and the BDCBKPT register causes the fetched opcode to be tagged. If this tagged opcode ever reaches the end of the instruction queue, the CPU enters active background mode rather than executing the tagged opcode. 0 Tag opcode at breakpoint address and enter active background mode if CPU attempts to execute that instruction 1 Breakpoint match forces active background mode at next instruction boundary (address need not be an opcode)
3 CLKSW	**Select Source for BDC Communications Clock** — CLKSW defaults to 0, which selects the alternate BDC clock source. 0 Alternate BDC clock source 1 MCU bus clock
2 WS	**Wait or Stop Status** — When the target CPU is in wait or stop mode, most BDC commands cannot function. However, the BACKGROUND command can be used to force the target CPU out of wait or stop and into active background mode where all BDC commands work. Whenever the host forces the target MCU into active background mode, the host should issue a READ_STATUS command to check that BDMACT = 1 before attempting other BDC commands. 0 Target CPU is running user application code or in active background mode (was not in wait or stop mode when background became active) 1 Target CPU is in wait or stop mode, or a BACKGROUND command was used to change from wait or stop to active background mode
1 WSF	**Wait or Stop Failure Status** — This status bit is set if a memory access command failed due to the target CPU executing a wait or stop instruction at or about the same time. The usual recovery strategy is to issue a BACKGROUND command to get out of wait or stop mode into active background mode, repeat the command that failed, then return to the user program. (Typically, the host would restore CPU registers and stack values and re-execute the wait or stop instruction.) 0 Memory access did not conflict with a wait or stop instruction 1 Memory access command failed because the CPU entered wait or stop mode
0 DVF	**Data Valid Failure Status** — This status bit is not used in the MC9S08LH64 Series because it does not have any slow access memory. 0 Memory access did not conflict with a slow memory access 1 Memory access command failed because CPU was not finished with a slow memory access

19.3.1.2 BDC Breakpoint Match Register (BDCBKPT)

This 16-bit register holds the address for the hardware breakpoint in the BDC. The BKPTEN and FTS control bits in BDCSCR are used to enable and configure the breakpoint logic. Dedicated serial BDC commands (READ_BKPT and WRITE_BKPT) are used to read and write the BDCBKPT register but is not accessible to user programs because it is not located in the normal memory map of the MCU. Breakpoints are normally set while the target MCU is in active background mode before running the user application program. For additional information about setup and use of the hardware breakpoint logic in the BDC, refer to Section 19.2.4, “BDC Hardware Breakpoint.”

19.3.2 System Background Debug Force Reset Register (SBDFR)

This register contains a single write-only control bit. A serial background mode command such as WRITE_BYTE must be used to write to SBDFR. Attempts to write this register from a user program are ignored. Reads always return 0x00.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								BDFR[1]
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

[1] BDFR is writable only through serial background mode debug commands, not from user programs.

Figure 19-6. System Background Debug Force Reset Register (SBDFR)

Table 19-3. SBDFR Register Field Description

Field	Description
0 BDFR	**Background Debug Force Reset** — A serial active background mode command such as WRITE_BYTE allows an external debug host to force a target system reset. Writing 1 to this bit forces an MCU reset. This bit cannot be written from a user program.

Chapter 20
Debug Module (S08DBGV3) (64K)

20.1 Introduction

The DBG module implements an on-chip ICE (in-circuit emulation) system and allows non-intrusive debug of application software by providing an on-chip trace buffer with flexible triggering capability. The trigger also can provide extended breakpoint capacity. The on-chip ICE system is optimized for the HCS08 8-bit architecture and supports 64K bytes or 128K bytes of memory space.

20.1.1 Features

The on-chip ICE system includes these distinctive features:

- Three comparators (A, B, and C) with ability to match addresses in 128K space
  - Dual mode, Comparators A and B used to compare addresses
  - Full mode, Comparator A compares address and Comparator B compares data
  - Can be used as triggers and/or breakpoints
  - Comparator C can be used as a normal hardware breakpoint
  - Loop1 capture mode, Comparator C is used to track most recent COF event captured into FIFO
- Tag and Force type breakpoints
- Nine trigger modes
  - A
  - A Or B
  - A Then B
  - A And B, where B is data (Full mode)
  - A And Not B, where B is data (Full mode)
  - Event Only B, store data
  - A Then Event Only B, store data
  - Inside Range, A ≤ Address ≤ B
  - Outside Range, Address < A or Address > B
- FIFO for storing change of flow information and event only data
  - Source address of conditional branches taken
  - Destination address of indirect JMP and JSR instruction
  - Destination address of interrupts, RTI, RTC, and RTS instruction
  - Data associated with Event B trigger modes

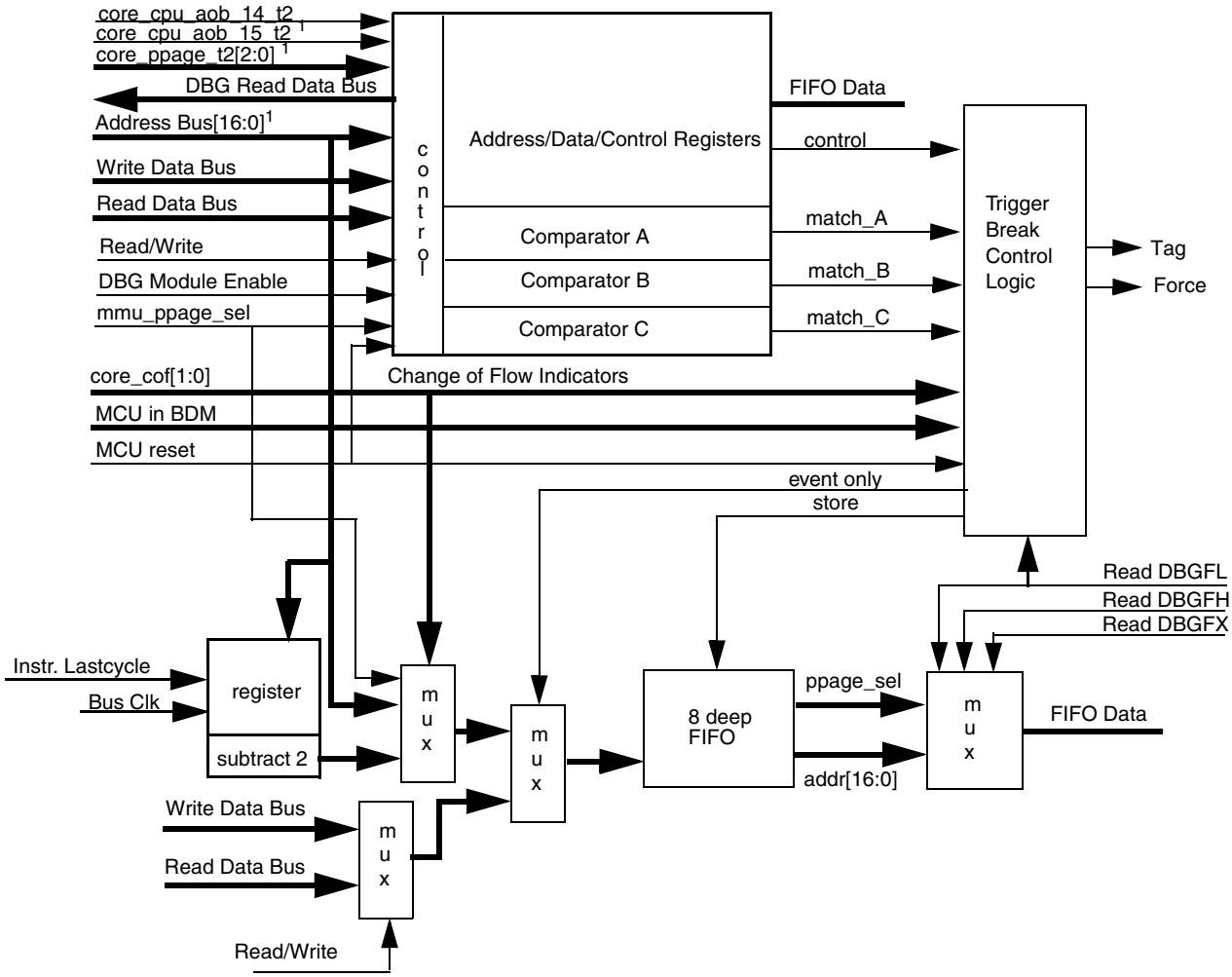
- Ability to End-trace until reset and Begin-trace from reset

20.1.2 Modes of Operation

The on-chip ICE system can be enabled in all MCU functional modes. The DBG module is disabled if the MCU is secure. The DBG module comparators are disabled when executing a Background Debug Mode (BDM) command.

20.1.3 Block Diagram

Figure 20-1 shows the structure of the DBG module.



1.In 64 KB versions of this module, there are only 16 address lines [15:0], there are no core_cpu_aob_15_t2 and core_ppage_t2[2] signals.

Figure 20-1. DBG Block Diagram

20.2 Signal Description

The DBG module contains no external signals.

20.3 Memory Map and Registers

This section provides a detailed description of all DBG registers accessible to the end user.

20.3.1 Module Memory Map

Table 20-1 shows the registers contained in the DBG module.

Table 20-1. Module Memory Map

Address	Use	Access
Base + $0000	Debug Comparator A High Register (DBGCAH)	Read/write
Base + $0001	Debug Comparator A Low Register (DBGCAL)	Read/write
Base + $0002	Debug Comparator B High Register (DBGCBH)	Read/write
Base + $0003	Debug Comparator B Low Register (DBGCBL)	Read/write
Base + $0004	Debug Comparator C High Register (DBGCCH)	Read/write
Base + $0005	Debug Comparator C Low Register (DBGCCL)	Read/write
Base + $0006	Debug FIFO High Register (DBGFH)	Read only
Base + $0007	Debug FIFO Low Register (DBGFL)	Read only
Base + $0008	Debug Comparator A Extension Register (DBGCAX)	Read/write
Base + $0009	Debug Comparator B Extension Register (DBGCBX)	Read/write
Base + $000A	Debug Comparator C Extension Register (DBGCCX)	Read/write
Base + $000B	Debug FIFO Extended Information Register (DBGFX)	Read only
Base + $000C	Debug Control Register (DBGC)	Read/write
Base + $000D	Debug Trigger Register (DBGT)	Read/write
Base + $000E	Debug Status Register (DBGS)	Read only
Base + $000F	Debug FIFO Count Register (DBGCNT)	Read only

20.3.2 Register Bit Summary

Table 20-2. Register Bit Summary

	7	6	5	4	3	2	1	0
DBGCAH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
DBGCAL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBGCBH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
DBGCBL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBGCCH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
DBGCCL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBGFH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
DBGFL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBGCAX	RWAEN	RWA	PAGSEL	0	0	0	0	bit-16
DBGCBX	RWBEN	RWB	PAGSEL	0	0	0	0	bit-16
DBGCCX	RWCEN	RWC	PAGSEL	0	0	0	0	bit-16
DBGFX	PPACC	0	0	0	0	0	0	0
DBGC	DBGEN	ARM	TAG	BRKEN	—	—	—	LOOP1
DBGT	TRGSEL	BEGIN	0	0	TRG[3:0]			
DBGS	AF	BF	CF	0	0	0	0	ARMF
DBGCNT	0	0	0	0	CNT[3:0]			

20.3.3 Register Descriptions

This section consists of the DBG register descriptions in address order.

Note: For all registers below, consider: U = Unchanged, bit maintain its value after reset.

20.3.3.1 Debug Comparator A High Register (DBGCAH)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
POR or non-end-run	1	1	1	1	1	1	1	1
Reset end-run[1]	U	U	U	U	U	U	U	U

Figure 20-2. Debug Comparator A High Register (DBGCAH)

1 In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-3. DBGCAH Field Descriptions

Field	Description
Bits 15–8	**Comparator A High Compare Bits** — The Comparator A High compare bits control whether Comparator A will compare the address bus bits [15:8] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.2 Debug Comparator A Low Register (DBGCAL)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
POR or non-end-run	1	1	1	1	1	1	1	0
Reset end-run[1]	U	U	U	U	U	U	U	U

Figure 20-3. Debug Comparator A Low Register (DBGCAL)

1 In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-4. DBGCAL Field Descriptions

Field	Description
Bits 7–0	**Comparator A Low Compare Bits** — The Comparator A Low compare bits control whether Comparator A will compare the address bus bits [7:0] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.3 Debug Comparator B High Register (DBGCBH)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	U	U	U	U	U

Figure 20-4. Debug Comparator B High Register (DBGCBH)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-5. DBGCBH Field Descriptions

Field	Description
Bits 15–8	**Comparator B High Compare Bits** — The Comparator B High compare bits control whether Comparator B will compare the address bus bits [15:8] to a logic 1 or logic 0. Not used in Full mode. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.4 Debug Comparator B Low Register (DBGCBL)

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	U	U	U	U	U

Figure 20-5. Debug Comparator B Low Register (DBGCBL)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-6. DBGCBL Field Descriptions

Field	Description
Bits 7–0	**Comparator B Low Compare Bits** — The Comparator B Low compare bits control whether Comparator B will compare the address bus or data bus bits [7:0] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0, compares to data if in Full mode 1 Compare corresponding address bit to a logic 1, compares to data if in Full mode

20.3.3.5 Debug Comparator C High Register (DBGCCH)

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	U	U	U	U	U

Figure 20-6. Debug Comparator C High Register (DBGCCH)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-7. DBGCCH Field Descriptions

Field	Description
Bits 15–8	**Comparator C High Compare Bits** — The Comparator C High compare bits control whether Comparator C will compare the address bus bits [15:8] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.6 Debug Comparator C Low Register (DBGCCL)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	U	U	U	U	U

Figure 20-7. Debug Comparator C Low Register (DBGCCL)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-8. DBGCCL Field Descriptions

Field	Description
Bits 7–0	**Comparator C Low Compare Bits** — The Comparator C Low compare bits control whether Comparator C will compare the address bus bits [7:0] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.7 Debug FIFO High Register (DBGFH)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
W								
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	U	U	U	U	U

= Unimplemented or Reserved

Figure 20-8. Debug FIFO High Register (DBGFH)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-9. DBGFH Field Descriptions

Field	Description
Bits 15–8	**FIFO High Data Bits** — The FIFO High data bits provide access to bits [15:8] of data in the FIFO. This register is not used in event only modes and will read a $00 for valid FIFO words.

20.3.3.8 Debug FIFO Low Register (DBGFL)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
W								
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	U	U	U	U	U

= Unimplemented or Reserved

Figure 20-9. Debug FIFO Low Register (DBGFL)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-10. DBGFL Field Descriptions

Field	Description
Bits 7–0	**FIFO Low Data Bits** — The FIFO Low data bits contain the least significant byte of data in the FIFO. When reading FIFO words, read DBGFX and DBGFH before reading DBGFL because reading DBGFL causes the FIFO pointers to advance to the next FIFO location. In event-only modes, there is no useful information in DBGFX and DBGFH so it is not necessary to read them before reading DBGFL.

20.3.3.9 Debug Comparator A Extension Register (DBGCAX)

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	RWAEN	RWA	PAGSEL	0	0	0	0	Bit 16
W								
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	0	0	0	0	U

= Unimplemented or Reserved

Figure 20-10. Debug Comparator A Extension Register (DBGCAX)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-11. DBGCAX Field Descriptions

Field	Description
7 RWAEN	**Read/Write Comparator A Enable Bit** — The RWAEN bit controls whether read or write comparison is enabled for Comparator A. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
6 RWA	**Read/Write Comparator A Value Bit** — The RWA bit controls whether read or write is used in compare for Comparator A. The RWA bit is not used if RWAEN = 0. 0 Write cycle will be matched 1 Read cycle will be matched
5 PAGSEL	**Comparator A Page Select Bit** — This PAGSEL bit controls whether Comparator A will be qualified with the internal signal (mmu_ppage_sel) that indicates an extended access through the PPAGE mechanism. When mmu_ppage_sel = 1, the 17-bit core address is a paged program access, and the 17-bit core address is made up of PPAGE[2:0]:addr[13:0]. When mmu_ppage_sel = 0, the 17-bit core address is either a 16-bit CPU address with a leading 0 in bit 16, or a 17-bit linear address pointer value. 0 Match qualified by mmu_ppage_sel = 0 so address bits [16:0] correspond to a 17-bit CPU address with a leading zero at bit 16, or a 17-bit linear address pointer address 1 Match qualified by mmu_ppage_sel = 1 so address bits [16:0] compare to flash memory address made up of PPAGE[2:0]:addr[13:0]
0 Bit 16	**Comparator A Extended Address Bit 16 Compare Bit** — The Comparator A bit 16 compare bit controls whether Comparator A will compare the core address bus bit 16 to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.10 Debug Comparator B Extension Register (DBGCBX)

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	RWBEN	RWB	PAGSEL	0	0	0	0	Bit 16
W								
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	0	0	0	0	U

= Unimplemented or Reserved

Figure 20-11. Debug Comparator B Extension Register (DBGCBX)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-12. DBGCBX Field Descriptions

Field	Description
7 RWBEN	**Read/Write Comparator B Enable Bit** — The RWBEN bit controls whether read or write comparison is enabled for Comparator B. In full modes, RWAEN and RWA are used to control comparison of R/W and RWBEN is ignored. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
6 RWB	**Read/Write Comparator B Value Bit** — The RWB bit controls whether read or write is used in compare for Comparator B. The RWB bit is not used if RWBEN = 0. In full modes, RWAEN and RWA are used to control comparison of R/W and RWB is ignored. 0 Write cycle will be matched 1 Read cycle will be matched
5 PAGSEL	**Comparator B Page Select Bit** — This PAGSEL bit controls whether Comparator B will be qualified with the internal signal (mmu_papge_sel) that indicates an extended access through the PPAGE mechanism. When mmu_ppage_sel = 1, the 17-bit core address is a paged program access, and the 17-bit core address is made up of PPAGE[2:0]:addr[13:0]. When mmu_papge_sel = 0, the 17-bit core address is either a 16-bit CPU address with a leading 0 in bit 16, or a 17-bit linear address pointer value. This bit is not used in full modes where comparator B is used to match the data value. 0 Match qualified by mmu_ppage_sel = 0 so address bits [16:0] correspond to a 17-bit CPU address with a leading zero at bit 16, or a 17-bit linear address pointer address 1 Match qualified by mmu_ppage_sel = 1 so address bits [16:0] compare to flash memory address made up of PPAGE[2:0]:addr[13:0]
0 Bit 16	**Comparator B Extended Address Bit 16 Compare Bit** — The Comparator B bit 16 compare bit controls whether Comparator B will compare the core address bus bit 16 to a logic 1 or logic 0. This bit is not used in full modes where comparator B is used to match the data value. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.11 Debug Comparator C Extension Register (DBGCCX)

Module Base + 0x000A

	7	6	5	4	3	2	1	0
R	RWCEN	RWC	PAGSEL	0	0	0	0	Bit 16
W								
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	U	U	0	0	0	0	U

= Unimplemented or Reserved

Figure 20-12. Debug Comparator C Extension Register (DBGCCX)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-13. DBGCCX Field Descriptions

Field	Description
7 RWCEN	**Read/Write Comparator C Enable Bit** — The RWCEN bit controls whether read or write comparison is enabled for Comparator C. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
6 RWC	**Read/Write Comparator C Value Bit** — The RWC bit controls whether read or write is used in compare for Comparator C. The RWC bit is not used if RWCEN = 0. 0 Write cycle will be matched 1 Read cycle will be matched
5 PAGSEL	**Comparator C Page Select Bit** — This PAGSEL bit controls whether Comparator C will be qualified with the internal signal (mmu_papge_sel) that indicates an extended access through the PPAGE mechanism. When mmu_ppage_sel = 1, the 17-bit core address is a paged program access, and the 17-bit core address is made up of PPAGE[2:0]:addr[13:0]. When mmu_papge_sel = 0, the 17-bit core address is either a 16-bit CPU address with a leading 0 in bit 16, or a 17-bit linear address pointer value. 0 Match qualified by mmu_ppage_sel = 0 so address bits [16:0] correspond to a 17-bit CPU address with a leading zero at bit 16, or a 17-bit linear address pointer address 1 Match qualified by mmu_ppage_sel = 1 so address bits [16:0] compare to flash memory address made up of PPAGE[2:0]:addr[13:0]
0 Bit 16	**Comparator C Extended Address Bit 16 Compare Bit** — The Comparator C bit 16 compare bit controls whether Comparator C will compare the core address bus bit 16 to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

20.3.3.12 Debug FIFO Extended Information Register (DBGFX)

Module Base + 0x000B

	7	6	5	4	3	2	1	0
R	PPACC	0	0	0	0	0	0	0
W								
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	U	0	0	0	0	0	0	U

= Unimplemented or Reserved

Figure 20-13. Debug FIFO Extended Information Register (DBGFX)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the bits in this register do not change after reset.

Table 20-14. DBGFX Field Descriptions

Field	Description
7 PPACC	**PPAGE Access Indicator Bit** — This bit indicates whether the captured information in the current FIFO word is associated with an extended access through the PPAGE mechanism or not. This is indicated by the internal signal mmu_ppage_sel which is 1 when the access is through the PPAGE mechanism. 0 The information in the corresponding FIFO word is event-only data or an unpaged 17-bit CPU address with bit-16 = 0 1 The information in the corresponding FIFO word is a 17-bit flash address with PPAGE[2:0] in the three most significant bits and CPU address[13:0] in the 14 least significant bits

20.3.3.13 Debug Control Register (DBGC)

Module Base + 0x000C

	7	6	5	4	3	2	1	0
R	DBGEN	ARM	TAG	BRKEN	0	0	0	LOOP1
W								
POR or non-end-run	1	1	0	0	0	0	0	0
Reset end-run[1]	U	0	U	0	0	0	0	U

= Unimplemented or Reserved

Figure 20-14. Debug Control Register (DBGC)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the ARM and BRKEN bits are cleared but the remaining control bits in this register do not change after reset.

Table 20-15. DBGC Field Descriptions

Field	Description
7 DBGEN	**DBG Module Enable Bit** — The DBGEN bit enables the DBG module. The DBGEN bit is forced to zero and cannot be set if the MCU is secure. 0 DBG not enabled 1 DBG enabled
6 ARM	**Arm Bit** — The ARM bit controls whether the debugger is comparing and storing data in FIFO. See Section 20.4.4.2, “Arming the DBG Module” for more information. 0 Debugger not armed 1 Debugger armed
5 TAG	**Tag or Force Bit** — The TAG bit controls whether a debugger or comparator C breakpoint will be requested as a tag or force breakpoint to the CPU. The TAG bit is not used if BRKEN = 0. 0 Force request selected 1 Tag request selected
4 BRKEN	**Break Enable Bit** — The BRKEN bit controls whether the debugger will request a breakpoint to the CPU at the end of a trace run, and whether comparator C will request a breakpoint to the CPU. 0 CPU break request not enabled 1 CPU break request enabled
0 LOOP1	**Select LOOP1 Capture Mode** — This bit selects either normal capture mode or LOOP1 capture mode. LOOP1 is not used in event-only modes. 0 Normal operation - capture COF events into the capture buffer FIFO 1 LOOP1 capture mode enabled. When the conditions are met to store a COF value into the FIFO, compare the current COF address with the address in comparator C. If these addresses match, override the FIFO capture and do not increment the FIFO count. If the address does not match comparator C, capture the COF address, including the PPACC indicator, into the FIFO and into comparator C.

20.3.3.14 Debug Trigger Register (DBGT)

Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	TRGSEL	BEGIN	0	0	TRG			
W[2]								
POR or non-end-run	0	1	0	0	0	0	0	0
Reset end-run[1]	U	U	0	0	U	U	U	U

= Unimplemented or Reserved

Figure 20-15. Debug Trigger Register (DBGT)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the control bits in this register do not change after reset.

[2] The DBG trigger register (DBGT) can not be changed unless ARM=0.

Table 20-16. DBGT Field Descriptions

Field	Description
7 TRGSEL	**Trigger Selection Bit** — The TRGSEL bit controls the triggering condition for the comparators. See Section 20.4.4, "Trigger Break Control (TBC)" for more information. 0 Trigger on any compare address access 1 Trigger if opcode at compare address is executed
6 BEGIN	**Begin/End Trigger Bit** — The BEGIN bit controls whether the trigger begins or ends storing of data in FIFO. 0 Trigger at end of stored data 1 Trigger before storing data
3–0 TRG	**Trigger Mode Bits** — The TRG bits select the trigger mode of the DBG module as shown in Table 20-17.

Table 20-17. Trigger Mode Encoding

TRG Value	Meaning
0000	A Only
0001	A Or B
0010	A Then B
0011	Event Only B
0100	A Then Event Only B
0101	A And B (Full Mode)
0110	A And Not B (Full mode)
0111	Inside Range
1000	Outside Range

Table 20-17. Trigger Mode Encoding

TRG Value	Meaning
1001 ↓ 1111	No Trigger

NOTE

The DBG trigger register (DBGT) can not be changed unless ARM=0.

20.3.3.15 Debug Status Register (DBGS)

Module Base + 0x000E

	7	6	5	4	3	2	1	0
R	AF	BF	CF	0	0	0	0	ARMF
W								
POR or non-end-run	0	0	0	0	0	0	0	1
Reset end-run[1]	U	U	U	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-16. Debug Status Register (DBGS)

1 In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, ARMF gets cleared by reset but AF, BF, and CF do not change after reset.

Table 20-18. DBGS Field Descriptions

Field	Description
7 AF	**Trigger A Match Bit** — The AF bit indicates if Trigger A match condition was met since arming. 0 Comparator A did not match 1 Comparator A match
6 BF	**Trigger B Match Bit** — The BF bit indicates if Trigger B match condition was met since arming. 0 Comparator B did not match 1 Comparator B match
5 CF	**Trigger C Match Bit** — The CF bit indicates if Trigger C match condition was met since arming. 0 Comparator C did not match 1 Comparator C match
0 ARMF	**Arm Flag Bit** — The ARMF bit indicates whether the debugger is waiting for trigger or waiting for the FIFO to fill. While DBGEN = 1, this status bit is a read-only image of the ARM bit in DBGC. See Section 20.4.4.2, “Arming the DBG Module” for more information. 0 Debugger not armed 1 Debugger armed

20.3.3.16 Debug Count Status Register (DBGCNT)

Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	0	0	0	0	CNT			
W								
POR or non-end-run	0	0	0	0	0	0	0	0
Reset end-run[1]	0	0	0	0	U	U	U	U

= Unimplemented or Reserved

Figure 20-17. Debug Count Status Register (DBGCNT)

[1] In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the CNT[3:0] bits do not change after reset.

Table 20-19. DBGS Field Descriptions

Field	Description
3–0 CNT	**FIFO Valid Count Bits** — The CNT bits indicate the amount of valid data stored in the FIFO. Table 20-20 shows the correlation between the CNT bits and the amount of valid data in FIFO. The CNT will stop after a count to eight even if more data is being stored in the FIFO. The CNT bits are cleared when the DBG module is armed, and the count is incremented each time a new word is captured into the FIFO. The host development system is responsible for checking the value in CNT[3:0] and reading the correct number of words from the FIFO because the count does not decrement as data is read out of the FIFO at the end of a trace run.

Table 20-20. CNT Bits

CNT Value	Meaning
0000	No data valid
0001	1 word valid
0010	2 words valid
0011	3 words valid
0100	4 words valid
0101	5 words valid
0110	6 words valid
0111	7 words valid
1000	8 words valid

20.4 Functional Description

This section provides a complete functional description of the on-chip ICE system. The DBG module is enabled by setting the DBGEN bit in the DBGC register. Enabling the module allows the arming, triggering and storing of data in the FIFO. The DBG module is made up of three main blocks, the Comparators, Trigger Break Control logic and the FIFO.

20.4.1 Comparator

The DBG module contains three comparators, A, B, and C. Comparator A compares the core address bus with the address stored in the DBGCAX, DBGCAH, and DBGCAL registers. Comparator B compares the core address bus with the address stored in the DBGCBX, DBGCBH, and DBGCBL registers except in full mode, where it compares the data buses to the data stored in the DBGCBL register. Comparator C compares the core address bus with the address stored in the DBGCCX, DBGCCH, and DBGCCL registers. Matches on Comparators A, B, and C are signaled to the Trigger Break Control (TBC) block.

20.4.1.1 RWA and RWAEN in Full Modes

In full modes ("A And B" and "A And Not B") RWAEN and RWA are used to select read or write comparisons for both comparators A and B. To select write comparisons and the write data bus in Full Modes set RWAEN=1 and RWA=0, otherwise read comparisons and the read data bus will be selected. The RWBEN and RWB bits are not used and will be ignored in Full Modes.

20.4.1.2 Comparator C in LOOP1 Capture Mode

Normally comparator C is used as a third hardware breakpoint and is not involved in the trigger logic for the on-chip ICE system. In this mode, it compares the core address bus with the address stored in the DBGCCX, DBGCCH, and DBGCCL registers. However, in LOOP1 capture mode, comparator C is managed by logic in the DBG module to track the address of the most recent change-of-flow event that was captured into the FIFO buffer. In LOOP1 capture mode, comparator C is not available for use as a normal hardware breakpoint.

When the ARM and DBGEN bits are set to one in LOOP1 capture mode, comparator C value registers are cleared to prevent the previous contents of these registers from interfering with the LOOP1 capture mode operation. When a COF event is detected, the address of the event is compared to the contents of the DBGCCX, DBGCCH, and DBGCCL registers to determine whether it is the same as the previous COF entry in the capture FIFO. If the values match, the capture is inhibited to prevent the FIFO from filling up with duplicate entries. If the values do not match, the COF event is captured into the FIFO and the DBGCCX, DBGCCH, and DBGCCL registers are updated to reflect the address of the captured COF event. When comparator C is updated, the PAGSEL bit (bit-7 of DBGCCX) is updated with the PPACC value that is captured into the FIFO. This bit indicates whether the COF address was a paged 17-bit program address using the PPAGE mechanism (PPACC=1) or a 17-bit CPU address that resulted from an unpaged CPU access.

20.4.2 Breakpoints

A breakpoint request to the CPU at the end of a trace run can be created if the BRKEN bit in the DBGC register is set. The value of the BEGIN bit in DBGT register determines when the breakpoint request to the CPU will occur. If the BEGIN bit is set, begin-trigger is selected and the breakpoint request will not occur until the FIFO is filled with 8 words. If the BEGIN bit is cleared, end-trigger is selected and the breakpoint request will occur immediately at the trigger cycle.

When traditional hardware breakpoints from comparators A or B are desired, set BEGIN=0 to select an end-trace run and set the trigger mode to either 0x0 (A-only) or 0x1 (A OR B) mode.

There are two types of breakpoint requests supported by the DBG module, tag-type and force-type. Tagged breakpoints are associated with opcode addresses and allow breaking just before a specific instruction executes. Force breakpoints are not associated with opcode addresses and allow breaking at the next instruction boundary. The TAG bit in the DBGC register determines whether CPU breakpoint requests will be a tag-type or force-type breakpoints. When TAG=0, a force-type breakpoint is requested and it will take effect at the next instruction boundary after the request. When TAG=1, a tag-type breakpoint is registered into the instruction queue and the CPU will break if/when this tag reaches the head of the instruction queue and the tagged instruction is about to be executed.

20.4.2.1 Hardware Breakpoints

Comparators A, B, and C can be used as three traditional hardware breakpoints whether the on-chip ICE real-time capture function is required or not. To use any breakpoint or trace run capture functions set DBGEN=1. BRKEN and TAG affect all three comparators. When BRKEN=0, no CPU breakpoints are enabled. When BRKEN=1, CPU breakpoints are enabled and the TAG bit determines whether the breakpoints will be tag-type or force-type breakpoints. To use comparators A and B as hardware breakpoints, set DBGT=0x81 for tag-type breakpoints and 0x01 for force-type breakpoints. This sets up an end-type trace with trigger mode “A OR B”.

Comparator C is not involved in the trigger logic for the on-chip ICE system.

20.4.3 Trigger Selection

The TRGSEL bit in the DBGT register is used to determine the triggering condition of the on-chip ICE system. TRGSEL applies to both trigger A and B except in the event only trigger modes. By setting the TRGSEL bit, the comparators will qualify a match with the output of opcode tracking logic. The opcode tracking logic is internal to each comparator and determines whether the CPU executed the opcode at the compare address. With the TRGSEL bit cleared a comparator match is all that is necessary for a trigger condition to be met.

NOTE

If the TRGSEL is set, the address stored in the comparator match address registers must be an opcode address for the trigger to occur.

20.4.4 Trigger Break Control (TBC)

The TBC is the main controller for the DBG module. Its function is to decide whether data should be stored in the FIFO based on the trigger mode and the match signals from the comparator. The TBC also determines whether a request to break the CPU should occur.

The TAG bit in DBGC controls whether CPU breakpoints are treated as tag-type or force-type breakpoints. The TRGSEL bit in DBGT controls whether a comparator A or B match is further qualified by opcode tracking logic. Each comparator has a separate circuit to track opcodes because the comparators could correspond to separate instructions that could be propagating through the instruction queue at the same time.

In end-type trace runs (BEGIN=0), when the comparator registers match, including the optional R/W match, this signal goes to the CPU break logic where BRKEN determines whether a CPU break is requested and the TAG control bit determines whether the CPU break will be a tag-type or force-type breakpoint. When TRGSEL is set, the R/W qualified comparator match signal also passes through the opcode tracking logic. If/when it propagates through this logic, it will cause a trigger to the ICE logic to begin or end capturing information into the FIFO. In the case of an end-type (BEGIN=0) trace run, the qualified comparator signal stops the FIFO from capturing any more information.

If a CPU breakpoint is also enabled, you would want TAG and TRGSEL to agree so that the CPU break occurs at the same place in the application program as the FIFO stopped capturing information. If TRGSEL was 0 and TAG was 1 in an end-type trace run, the FIFO would stop capturing as soon as the comparator address matched, but the CPU would continue running until a TAG signal could propagate through the CPUs instruction queue which could take a long time in the case where changes of flow caused the instruction queue to be flushed. If TRGSEL was one and TAG was zero in an end-type trace run, the CPU would break before the comparator match signal could propagate through the opcode tracking logic to end the trace run.

In begin-type trace runs (BEGIN=1), the start of FIFO capturing is triggered by the qualified comparator signals, and the CPU breakpoint (if enabled by BRKEN=1) is triggered when the FIFO becomes full. Since this FIFO full condition does not correspond to the execution of a tagged instruction, it would not make sense to use TAG=1 for a begin-type trace run.

20.4.4.1 Begin- and End-Trigger

The definition of begin- and end-trigger as used in the DBG module are as follows:

- Begin-trigger: Storage in FIFO occurs after the trigger and continues until 8 locations are filled.
- End-trigger: Storage in FIFO occurs until the trigger with the least recent data falling out of the FIFO if more than 8 words are collected.

20.4.4.2 Arming the DBG Module

Arming occurs by enabling the DBG module by setting the DBGEN bit and by setting the ARM bit in the DBGC register. The ARM bit in the DBGC register and the ARMF bit in the DBGS register are cleared when the trigger condition is met in end-trigger mode or when the FIFO is filled in begin-trigger mode. In the case of an end-trace where DBGEN=1 and BEGIN=0, ARM and ARMF are cleared by any reset to

end the trace run that was in progress. The ARMF bit is also cleared if ARM is written to zero or when the DBGEN bit is low. The TBC logic determines whether a trigger condition has been met based on the trigger mode and the trigger selection.

20.4.4.3 Trigger Modes

The on-chip ICE system supports nine trigger modes. The trigger modes are encoded as shown in Table 20-17. The trigger mode is used as a qualifier for either starting or ending the storing of data in the FIFO. When the match condition is met, the appropriate flag AF or BF is set in DBGS register. Arming the DBG module clears the AF, BF, and CF flags in the DBGS register. In all trigger modes except for the event only modes change of flow addresses are stored in the FIFO. In the event only modes only the value on the data bus at the trigger event B comparator match address will be stored.

20.4.4.3.1 A Only

In the A Only trigger mode, if the match condition for A is met, the AF flag in the DBGS register is set.

20.4.4.3.2 A Or B

In the A Or B trigger mode, if the match condition for A or B is met, the corresponding flag(s) in the DBGS register are set.

20.4.4.3.3 A Then B

In the A Then B trigger mode, the match condition for A must be met before the match condition for B is compared. When the match condition for A or B is met, the corresponding flag in the DBGS register is set.

20.4.4.3.4 Event Only B

In the Event Only B trigger mode, if the match condition for B is met, the BF flag in the DBGS register is set. The Event Only B trigger mode is considered a begin-trigger type and the BEGIN bit in the DBGT register is ignored.

20.4.4.3.5 A Then Event Only B

In the A Then Event Only B trigger mode, the match condition for A must be met before the match condition for B is compared. When the match condition for A or B is met, the corresponding flag in the DBGS register is set. The A Then Event Only B trigger mode is considered a begin-trigger type and the BEGIN bit in the DBGT register is ignored.

20.4.4.3.6 A And B (Full Mode)

In the A And B trigger mode, Comparator A compares to the address bus and Comparator B compares to the data bus. In the A and B trigger mode, if the match condition for A and B happen on the same bus cycle, both the AF and BF flags in the DBGS register are set. If a match condition on only A or only B happens, no flags are set.

For Breakpoint tagging operation with an end-trigger type trace, only matches from Comparator A will be used to determine if the Breakpoint conditions are met and Comparator B matches will be ignored.

20.4.4.3.7 A And Not B (Full Mode)

In the A And Not B trigger mode, comparator A compares to the address bus and comparator B compares to the data bus. In the A And Not B trigger mode, if the match condition for A and Not B happen on the same bus cycle, both the AF and BF flags in the DBGS register are set. If a match condition on only A or only Not B occur no flags are set.

For Breakpoint tagging operation with an end-trigger type trace, only matches from Comparator A will be used to determine if the Breakpoint conditions are met and Comparator B matches will be ignored.

20.4.4.3.8 Inside Range, A ≤ address ≤ B

In the Inside Range trigger mode, if the match condition for A and B happen on the same bus cycle, both the AF and BF flags in the DBGS register are set. If a match condition on only A or only B occur no flags are set.

20.4.4.3.9 Outside Range, address < A or address > B

In the Outside Range trigger mode, if the match condition for A or B is met, the corresponding flag in the DBGS register is set.

The four control bits BEGIN and TRGSEL in DBGT, and BRKEN and TAG in DBGC, determine the basic type of debug run as shown in Table 1.21. Some of the 16 possible combinations are not used (refer to the notes at the end of the table).

Table 20-21. Basic Types of Debug Runs

BEGIN	TRGSEL	BRKEN	TAG	Type of Debug Run
0	0	0	x[1]	Fill FIFO until trigger address (No CPU breakpoint - keep running)
0	0	1	0	Fill FIFO until trigger address, then force CPU breakpoint
0	0	1	1	Do not use[2]
0	1	0	x[1]	Fill FIFO until trigger opcode about to execute (No CPU breakpoint - keep running)
0	1	1	0	Do not use[3]
0	1	1	1	Fill FIFO until trigger opcode about to execute (trigger causes CPU breakpoint)
1	0	0	x[1]	Start FIFO at trigger address (No CPU breakpoint - keep running)
1	0	1	0	Start FIFO at trigger address, force CPU breakpoint when FIFO full
1	0	1	1	Do not use[4]
1	1	0	x[1]	Start FIFO at trigger opcode (No CPU breakpoint - keep running)
1	1	1	0	Start FIFO at trigger opcode, force CPU breakpoint when FIFO full
1	1	1	1	Do not use[4]

[1] When BRKEN = 0, TAG is do not care (x in the table).

[2] In end trace configurations (BEGIN = 0) where a CPU breakpoint is enabled (BRKEN = 1), TRGSEL should agree with TAG. In this case, where TRGSEL = 0 to select no opcode tracking qualification and TAG = 1 to specify a tag-type CPU breakpoint, the CPU breakpoint would not take effect until sometime after the FIFO stopped storing values. Depending on program loops or interrupts, the delay could be very long.

[3] In end trace configurations (BEGIN = 0) where a CPU breakpoint is enabled (BRKEN = 1), TRGSEL should agree with TAG. In this case, where TRGSEL = 1 to select opcode tracking qualification and TAG = 0 to specify a force-type CPU breakpoint, the CPU breakpoint would erroneously take effect before the FIFO stopped storing values and the debug run would not complete normally.

4 In begin trace configurations (BEGIN = 1) where a CPU breakpoint is enabled (BRKEN = 1), TAG should not be set to 1. In begin trace debug runs, the CPU breakpoint corresponds to the FIFO full condition which does not correspond to a taggable instruction fetch.

20.4.5 FIFO

The FIFO is an eight word deep FIFO. In all trigger modes except for event only, the data stored in the FIFO will be change of flow addresses. In the event only trigger modes only the data bus value corresponding to the event is stored. In event only trigger modes, the high byte of the valid data from the FIFO will always read a 0x00 and the extended information byte in DBGFX will always read 0x00.

20.4.5.1 Storing Data in FIFO

In all trigger modes except for the event only modes, the address stored in the FIFO will be determined by the change of flow indicators from the core. The signal core_cof[1] indicates the current core address is the destination address of an indirect JSR or JMP instruction, or a RTS, RTC, or RTI instruction or interrupt vector and the destination address should be stored. The signal core_cof[0] indicates that a conditional branch was taken and that the source address of the conditional branch should be stored.

20.4.5.2 Storing with Begin-Trigger

Storing with Begin-Trigger can be used in all trigger modes. Once the DBG module is enabled and armed in the begin-trigger mode, data is not stored in the FIFO until the trigger condition is met. Once the trigger condition is met the DBG module will remain armed until 8 words are stored in the FIFO. If the core_cof[1] signal becomes asserted, the current address is stored in the FIFO. If the core_cof[0] signal becomes asserted, the address registered during the previous last cycle is decremented by two and stored in the FIFO.

20.4.5.3 Storing with End-Trigger

Storing with End-Trigger cannot be used in event-only trigger modes. Once the DBG module is enabled and armed in the end-trigger mode, data is stored in the FIFO until the trigger condition is met. If the core_cof[1] signal becomes asserted, the current address is stored in the FIFO. If the core_cof[0] signal becomes asserted, the address registered during the previous last cycle is decremented by two and stored in the FIFO. When the trigger condition is met, the ARM and ARMF will be cleared and no more data will be stored. In non-event only end-trigger modes, if the trigger is at a change of flow address the trigger event will be stored in the FIFO.

20.4.5.4 Reading Data from FIFO

The data stored in the FIFO can be read using BDM commands provided the DBG module is enabled and not armed (DBGEN=1 and ARM=0). The FIFO data is read out first-in-first-out. By reading the CNT bits

in the DBGCNT register at the end of a trace run, the number of valid words can be determined. The FIFO data is read by optionally reading the DBGFX and DBGFH registers followed by the DBGFL register. Each time the DBGFL register is read the FIFO is shifted to allow reading of the next word however the count does not decrement. In event-only trigger modes where the FIFO will contain only the data bus values stored, to read the FIFO only DBGFL needs to be accessed.

The FIFO is normally only read while ARM and ARMF=0, however reading the FIFO while the DBG module is armed will return the data value in the oldest location of the FIFO and the TBC will not allow the FIFO to shift. This action could cause a valid entry to be lost because the unexpected read blocked the FIFO advance.

If the DBG module is not armed and the DBGFL register is read, the TBC will store the current opcode address. Through periodic reads of the DBGFX, DBGFH, and DBGFL registers while the DBG module is not armed, host software can provide a histogram of program execution. This is called profile mode. Since the full 17-bit address and the signal that indicates whether an address is in paged extended memory are captured on each FIFO store, profile mode works correctly over the entire extended memory map.

20.4.6 Interrupt Priority

When TRGSEL is set and the DBG module is armed to trigger on begin- or end-trigger types, a trigger is not detected in the condition where a pending interrupt occurs at the same time that a target address reaches the top of the instruction pipe. In these conditions, the pending interrupt has higher priority and code execution switches to the interrupt service routine.

When TRGSEL is clear and the DBG module is armed to trigger on end-trigger types, the trigger event is detected on a program fetch of the target address, even when an interrupt becomes pending on the same cycle. In these conditions, the pending interrupt has higher priority, the exception is processed by the core and the interrupt vector is fetched. Code execution is halted before the first instruction of the interrupt service routine is executed. In this scenario, the DBG module will have cleared ARM without having recorded the change-of-flow that occurred as part of the interrupt exception. Note that the stack will hold the return addresses and can be used to reconstruct execution flow in this scenario.

When TRGSEL is clear and the DBG module is armed to trigger on begin-trigger types, the trigger event is detected on a program fetch of the target address, even when an interrupt becomes pending on the same cycle. In this scenario, the FIFO captures the change of flow event. Because the system is configured for begin-trigger, the DBG remains armed and does not break until the FIFO has been filled by subsequent change of flow events.

20.5 Resets

The DBG module cannot cause an MCU reset.

There are two different ways this module will respond to reset depending upon the conditions before the reset event. If the DBG module was setup for an end trace run with DBGEN=1 and BEGIN=0, ARM, ARMF, and BRKEN are cleared but the reset function on most DBG control and status bits is overridden so a host development system can read out the results of the trace run after the MCU has been reset. In all other cases including POR, the DBG module controls are initialized to start a begin trace run starting from when the reset vector is fetched. The conditions for the default begin trace run are:

- DBGCAX=0x00, DBGCAH=0xFF, DBGCAL=0xFE so comparator A is set to match when the 16-bit CPU address 0xFFFE appears during the reset vector fetch
- DBGC=0xC0 to enable and arm the DBG module
- DBGT=0x40 to select a force-type trigger, a BEGIN trigger, and A-only trigger mode

20.6 Interrupts

The DBG contains no interrupt source.

20.7 Electrical Specifications

The DBG module contain no electrical specifications.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

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)
www.freescale.com/support

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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
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 +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale’s Environmental Products program, go to http://www.freescale.com/epp.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2008-2010. All rights reserved.

MC9S08LH64
Rev. 5, 04/2010

freescale™
semiconductor