MC9S08LC60
MC9S08LC36

Data Sheet: Technical Data

HCS08
Microcontrollers

MC9S08LC60
Rev. 4
07/2007

freescale.com

MC9S08LC60 Series Features

8-Bit HCS08 Central Processor Unit (CPU)

- 40-MHz HCS08 CPU
- HC08 instruction set with added BGND instruction
- Background debugging system
- Breakpoint capability to allow single breakpoint setting during in-circuit debugging (plus two more breakpoints in on-chip debug module)
- In-Circuit Emulator (ICE) debug module containing two comparators and nine trigger modes. Eight deep FIFO for storing change-of-flow addresses and event-only data. ICE debug module supports both tag and force breakpoints.
- Support for up to 32 interrupt/reset sources

Memory Options

- Dual on-chip in-circuit programmable FLASH memories with block protection and security options; 60K and 36K options available
- Program/erase of one FLASH array while executing from another
- On-chip random-access memory (RAM); 4K and 2.5K options available

Power-Saving Features

- Wait plus three stops
- Software disable of clock monitor and low-voltage interrupt (LVI) for lowest stop current
- Software-generated real-time clock (RTC) functions using real-time interrupt (RTI)

Configurable Clock Source

- Clock source options include crystal, resonator, external clock, or internally generated clock with precision nonvolatile memory (NVM) trimming
- Automatic clock monitor function

System Protection

- Optional computer operating properly (COP) reset
- Low-voltage detection with reset or interrupt
- Illegal opcode detection with reset

Package Options

- 64-pin low-profile quad flat package (LQFP)
- 80-pin LQFP

Peripherals

- **LCD** (liquid crystal display driver) — Compatible with 5-V or 3-V LCD glass displays; functional in wait and stop3 low-power modes; selectable frontplane and backplane configurations:
  - 4 x 40 or 3 x 41 (80-pin package)
  - 4 x 32 or 3 x 33 (64-pin package)
- **ACMP** (analog comparator) — option to compare to internal reference voltage; output is software selectable to be driven to the input capture of TPM1 channel 0.
- **ADC** (analog-to-digital converter) — 8-channel, 12-bit with automatic compare function, asynchronous clock source, temperature sensor and internal bandgap reference channel. ADC is hardware triggerable using the RTI counter.
- **SCI** (serial communications interface) **—** available single-wire mode
- **SPI1** and **SPI2** — Two serial peripheral interface modules
- **KBI** — Two 8-pin keyboard interrupt modules with software selectable rising or falling edge detect
- **IIC** — Inter-integrated circuit bus module capable of operation up to 100 kbps with maximum bus loading; capable of higher baudrates with reduced loading
- **TPM1** and **TPM2** — Two timer/pulse-width modulators with selectable input capture, output compare, and edge-aligned PWM capability on each channel. Each timer module may be configured for buffered, centered PWM (CPWM) on all channels.

Input/Output

- Up to 24 general-purpose input/output (I/O) pins; includes two output-only pins and one input-only pin
- Software selectable pullups on ports when used as input. Selection is on an individual port bit basis.
- Software selectable slew rate control on ports when used as outputs (selection is on an individual port bit basis)
- Software selectable drive strength control on ports when used as outputs (selection is on an individual port bit basis)
- Internal pullup on $\overline{\text{RESET}}$ and IRQ pin to reduce customer system cost

MC9S08LC60 Series Data Sheet

Covers MC9S08LC60
MC9S08LC36

MC9S08LC60
Rev. 4
07/2007

This document contains information on a new product. Specifications and information herein are subject to change without notice.

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	02/2007	Initial advance information release.
2	05/2007	Incorporated changes to the LCDSupply Field Descriptions for the CPCADJ field, added a Run Idd chart, performed some minor formatting edits and fixed a couple of typos.
3	06/2007	Updated the Appendix with ESD tables, package info, and mechanical drawings.
4	07/2007	Updated the Appendix with ESD tables, package info, and mechanical drawings for the 80-pin LQFP package.

List of Chapters

Chapter	Title	Page
Chapter 1	Device Overview	21
Chapter 2	Pins and Connections	25
Chapter 3	Modes of Operation	33
Chapter 4	Memory	39
Chapter 5	Resets, Interrupts, and System Configuration	63
Chapter 6	Parallel Input/Output	81
Chapter 7	Keyboard Interrupt (S08KBIV2)	95
Chapter 8	Central Processor Unit (S08CPUV2)	103
Chapter 9	Liquid Crystal Display Driver (S08LCDV1)	123
Chapter 10	Internal Clock Generator (S08ICGV4)	169
Chapter 11	Timer Pulse-Width Modulator (S08TPMV2)	197
Chapter 12	Serial Communications Interface (S08SCIV3)	213
Chapter 13	Serial Peripheral Interface (S08SPIV3)	233
Chapter 14	Inter-Integrated Circuit (S08IICV1)	249
Chapter 15	Analog-to-Digital Converter (S08ADC12V1)	267
Chapter 16	Analog Comparator (S08ACMPV2)	293
Chapter 17	Development Support	301
Appendix A	Electrical Characteristics	323
Appendix B	Ordering Information and Mechanical Drawings	351

Table of Contents

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

Chapter 1
Device Overview

1.1 Introduction 21
1.2 Devices in the MC9S08LC60 Series 21
1.3 MCU Block Diagram 22
1.4 System Clock Distribution 23

Chapter 2
Pins and Connections

2.1 Introduction 25
2.2 Device Pin Assignment 25
2.3 Recommended System Connections 27
2.3.1 Power (V_{DD}, V_{SS}, V_{DDAD}, V_{SSAD}) 29
2.3.2 ADC Reference Pins (V_{REFH}, V_{REFL}) 29
2.3.3 Oscillator (XTAL, EXTAL) 29
2.3.4 $\overline{\text{RESET}}$ Pin 30
2.3.5 Background / Mode Select (BKGD/MS) 30
2.3.6 LCD Pins 31
2.3.6.1 LCD Power Pins 31
2.3.6.2 LCD Frontplane and Backplane Driver Pins 31
2.3.7 General-Purpose I/O and Peripheral Ports 31

Chapter 3
Modes of Operation

3.1 Introduction 33
3.2 Features 33
3.3 Run Mode 33
3.4 Active Background Mode 33
3.5 Wait Mode 34
3.6 Stop Modes 35
3.6.1 Stop3 Mode 35
3.6.1.1 LVD Enabled in Stop Mode 36
3.6.1.2 Active BDM Enabled in Stop Mode 36
3.6.2 Stop2 Mode 36
3.6.3 Stop1 Mode 37
3.6.4 On-Chip Peripheral Modules in Stop Modes 37

Chapter 4
Memory

4.1 MC9S08LC60 Series Memory Map 39
4.1.1 Reset and Interrupt Vector Assignments 40
4.2 Register Addresses and Bit Assignments 42
4.3 RAM 47
4.4 FLASH 48
4.4.1 Features 49
4.4.2 Program and Erase Times 49
4.4.3 Program and Erase Command Execution 50
4.4.4 Burst Program Execution 51
4.4.5 Access Errors 53
4.4.6 FLASH Block Protection 53
4.4.7 Vector Redirection 54
4.5 Security 54
4.6 FLASH Registers and Control Bits 56
4.6.1 FLASH Clock Divider Register (FCDIV) 56
4.6.2 FLASH Options Register (FOPT and NVOPT) 57
4.6.3 FLASH Configuration Register (FCNFG) 58
4.6.4 FLASH Protection Register (FPROT and NVPROT) 58
4.6.5 FLASH Status Register (FSTAT) 60
4.6.6 FLASH Command Register (FCMD) 61

Chapter 5
Resets, Interrupts, and System Configuration

5.1 Introduction 63
5.2 Features 63
5.3 MCU Reset 63
5.4 Computer Operating Properly (COP) Watchdog 64
5.5 Interrupts 65
5.5.1 Interrupt Stack Frame 66
5.5.2 External Interrupt Request (IRQ) Pin 67
5.5.2.1 Pin Configuration Options 67
5.5.2.2 Edge and Level Sensitivity 67
5.5.3 Interrupt Vectors, Sources, and Local Masks 67
5.6 Low-Voltage Detect (LVD) System 69
5.6.1 Power-On Reset Operation 69
5.6.2 LVD Reset Operation 69
5.6.3 LVD Interrupt Operation 69
5.6.4 Low-Voltage Warning (LVW) 69
5.7 Real-Time Interrupt (RTI) 69
5.8 Reset, Interrupt, and System Control Registers and Control Bits 70
5.8.1 Interrupt Pin Request Status and Control Register (IRQSC) 70
5.8.2 System Reset Status Register (SRS) 72

5.8.3 System Background Debug Force Reset Register (SBDFR) 73
5.8.4 System Options Register (SOPT1) 73
5.8.5 System Options Register (SOPT2) 74
5.8.6 System Device Identification Register (SDIDH, SDIDL) 75
5.8.7 System Real-Time Interrupt Status and Control Register (SRTISC) 76
5.8.8 System Power Management Status and Control 1 Register (SPMSC1) 77
5.8.9 System Power Management Status and Control 2 Register (SPMSC2) 78
5.8.10 System Power Management Status and Control 3 Register (SPMSC3) 79

Chapter 6
Parallel Input/Output

6.1 Pin Behavior in Stop Modes 83
6.2 Parallel I/O Registers 83
6.2.1 Port A Registers 83
6.2.1.1 Port A Data Registers (PTAD) 84
6.2.1.2 Port A Data Direction Registers (PTADD) 84
6.2.2 Port A Control Registers 85
6.2.2.1 Internal Pullup Enable (PTAPE) 85
6.2.2.2 Output Slew Rate Control Enable (PTASE) 86
6.2.2.3 Output Drive Strength Select (PTADS) 86
6.2.3 Port B Registers 87
6.2.3.1 Port B Data Registers (PTBD) 87
6.2.3.2 Port B Data Direction Registers (PTBDD) 88
6.2.4 Port B Control Registers 88
6.2.4.1 Internal Pullup Enable (PTBPE) 88
6.2.4.2 Output Slew Rate Control Enable (PTBSE) 89
6.2.4.3 Output Drive Strength Select (PTBDS) 90
6.2.5 Port C Registers 90
6.2.5.1 Port C Data Registers (PTCD) 91
6.2.5.2 Port C Data Direction Registers (PTCDD) 91
6.2.6 Port C Control Registers 91
6.2.6.1 Internal Pullup Enable (PTCPE) 92
6.2.6.2 Output Slew Rate Control Enable (PTCSE) 93
6.2.6.3 Output Drive Strength Select (PTCDS) 93

Chapter 7
Keyboard Interrupt (S08KBIV2)

7.1 Introduction 95
7.1.1 Features 97
7.1.2 Modes of Operation 97
7.1.2.1 KBI in Wait Mode 97
7.1.2.2 KBI in Stop Modes 97
7.1.2.3 KBI in Active Background Mode 97
7.1.3 Block Diagram 97

7.2 External Signal Description 98
7.3 Register Definition 99
7.3.1 KBIx Status and Control Register (KBIxSC) 99
7.3.2 KBIx Pin Enable Register (KBIxPE) 99
7.3.3 KBIx Edge Select Register (KBIxES) 100
7.4 Functional Description 100
7.4.1 Edge Only Sensitivity 101
7.4.2 Edge and Level Sensitivity 101
7.4.3 KBI Pullup/Pulldown Resistors 101
7.4.4 KBI Initialization 101

Chapter 8
Central Processor Unit (S08CPUV2)

8.1 Introduction 103
8.1.1 Features 103
8.2 Programmer's Model and CPU Registers 104
8.2.1 Accumulator (A) 104
8.2.2 Index Register (H:X) 104
8.2.3 Stack Pointer (SP) 105
8.2.4 Program Counter (PC) 105
8.2.5 Condition Code Register (CCR) 105
8.3 Addressing Modes 107
8.3.1 Inherent Addressing Mode (INH) 107
8.3.2 Relative Addressing Mode (REL) 107
8.3.3 Immediate Addressing Mode (IMM) 107
8.3.4 Direct Addressing Mode (DIR) 107
8.3.5 Extended Addressing Mode (EXT) 108
8.3.6 Indexed Addressing Mode 108
8.3.6.1 Indexed, No Offset (IX) 108
8.3.6.2 Indexed, No Offset with Post Increment (IX+) 108
8.3.6.3 Indexed, 8-Bit Offset (IX1) 108
8.3.6.4 Indexed, 8-Bit Offset with Post Increment (IX1+) 108
8.3.6.5 Indexed, 16-Bit Offset (IX2) 108
8.3.6.6 SP-Relative, 8-Bit Offset (SP1) 108
8.3.6.7 SP-Relative, 16-Bit Offset (SP2) 109
8.4 Special Operations 109
8.4.1 Reset Sequence 109
8.4.2 Interrupt Sequence 109
8.4.3 Wait Mode Operation 110
8.4.4 Stop Mode Operation 110
8.4.5 BGND Instruction 111
8.5 HCS08 Instruction Set Summary 112

Chapter 9
Liquid Crystal Display Driver (S08LCDV1)

9.1 Introduction 123
9.1.1 Features 125
9.1.2 Modes of Operation 125
9.1.3 Block Diagram 126
9.2 External Signal Description 127
9.2.1 BP[2:0] 127
9.2.2 FP[39:0] 127
9.2.3 BP3/FP40 127
9.2.4 V_{LCD} 127
9.2.5 V_{LL1}, VLL2, V_{LL3} 127
9.2.6 Vcap1, Vcap2 128
9.3 Register Definition 128
9.3.1 LCD Control Register 0 (LCDCR0) 128
9.3.2 LCD Control Register 1 (LCDCR1) 129
9.3.3 LCD Frontplane Enable Registers 0–5 (FPENR0–FPENR5) 130
9.3.4 LCDRAM Registers (LCDRAM) 131
9.3.4.1 LCDRAM Registers as On/Off Selector (LCDDRMS = 0) 133
9.3.4.2 LCDRAM Registers as Blink Enable/Disable (LCDDRMS = 1) 133
9.3.5 LCD Clock Source Register (LCDCLKS) 133
9.3.6 LCD Voltage Supply Register (LCDSUPPLY) 134
9.3.7 LCD Blink Control Register (LCDBCTL) 135
9.3.8 LCD Command and Status Register (LCDCMD) 136
9.4 Functional Description 137
9.4.1 LCD Driver Description 138
9.4.1.1 LCD Duty Cycle 138
9.4.1.2 LCD Bias 139
9.4.1.3 LCD Module Waveform Base Clock and Frame Frequency 139
9.4.1.4 LCD Waveform Examples 141
9.4.2 LCDRAM Registers 149
9.4.2.1 LCDRAM Data Clear Command 149
9.4.2.2 LCDRAM Data Blank Command 149
9.4.3 LCD Blinking 149
9.4.3.1 LCD Segment Blinking 150
9.4.3.2 Blink Frequency 150
9.4.4 LCD Charge Pump, Voltage Divider, and Power Supply Operation 150
9.4.4.1 LCD Charge Pump and Voltage Divider 152
9.4.4.2 LCD Power Supply and Voltage Buffer Configuration 153
9.4.5 Resets 155
9.4.6 Interrupts 155
9.5 Initialization Section 155
9.5.1 Initialization Sequence 156
9.5.2 Initialization Examples 157

9.5.2.1 Initialization Example 1 158
9.5.2.2 Initialization Example 2 159
9.5.2.3 Initialization Example 3 161
9.5.2.4 Initialization Example 4 162
9.6 Application Information 163
9.6.1 LCD Seven Segment Example Description 163
9.6.1.1 LCD Module Waveforms 165
9.6.1.2 Segment On Driving Waveform 166
9.6.1.3 Segment Off Driving Waveform 166
9.6.2 LCD Contrast Control 166
9.6.3 LCD Power Consumption 167

Chapter 10
Internal Clock Generator (S08ICGV4)

10.1 Introduction 169
10.2 Introduction 171
10.2.1 Features 171
10.2.2 Modes of Operation 172
10.2.3 Block Diagram 173
10.3 External Signal Description 173
10.3.1 EXTAL — External Reference Clock / Oscillator Input 173
10.3.2 XTAL — Oscillator Output 173
10.3.3 External Clock Connections 174
10.3.4 External Crystal/Resonator Connections 174
10.4 Register Definition 175
10.4.1 ICG Control Register 1 (ICGC1) 175
10.4.2 ICG Control Register 2 (ICGC2) 177
10.4.3 ICG Status Register 1 (ICGS1) 178
10.4.4 ICG Status Register 2 (ICGS2) 179
10.4.5 ICG Filter Registers (ICGFLTU, ICGFLTL) 179
10.4.6 ICG Trim Register (ICGTRM) 180
10.5 Functional Description 180
10.5.1 Off Mode (Off) 181
10.5.1.1 BDM Active 181
10.5.1.2 OSCSTEN Bit Set 181
10.5.1.3 Stop/Off Mode Recovery 181
10.5.2 Self-Clocked Mode (SCM) 181
10.5.3 FLL Engaged, Internal Clock (FEI) Mode 182
10.5.4 FLL Engaged Internal Unlocked 183
10.5.5 FLL Engaged Internal Locked 183
10.5.6 FLL Bypassed, External Clock (FBE) Mode 183
10.5.7 FLL Engaged, External Clock (FEE) Mode 183
10.5.7.1 FLL Engaged External Unlocked 184
10.5.7.2 FLL Engaged External Locked 184

10.5.8 FLL Lock and Loss-of-Lock Detection 184
10.5.9 FLL Loss-of-Clock Detection 185
10.5.10 Clock Mode Requirements 186
10.5.11 Fixed Frequency Clock 187
10.5.12 High Gain Oscillator 187
10.6 Initialization/Application Information 187
10.6.1 Introduction 187
10.6.2 Example #1: External Crystal = 32 kHz, Bus Frequency = 4.19 MHz 189
10.6.3 Example #2: External Crystal = 4 MHz, Bus Frequency = 20 MHz 191
10.6.4 Example #3: No External Crystal Connection, 5.4 MHz Bus Frequency 193
10.6.5 Example #4: Internal Clock Generator Trim 195

Chapter 11
Timer Pulse-Width Modulator (S08TPMV2)

11.1 Introduction 197
11.1.1 Features 199
11.1.2 Block Diagram 199
11.2 External Signal Description 201
11.2.1 External TPM Clock Sources 201
11.2.2 TPMxCHn — TPMx Channel n I/O Pins 201
11.3 Register Definition 201
11.3.1 Timer x Status and Control Register (TPMxSC) 202
11.3.2 Timer x Counter Registers (TPMxCNTH:TPMxCNTL) 203
11.3.3 Timer x Counter Modulo Registers (TPMxMODH:TPMxMODL) 204
11.3.4 Timer x Channel n Status and Control Register (TPMxCnSC) 205
11.3.5 Timer x Channel Value Registers (TPMxCnVH:TPMxCnVL) 206
11.4 Functional Description 207
11.4.1 Counter 207
11.4.2 Channel Mode Selection 208
11.4.2.1 Input Capture Mode 208
11.4.2.2 Output Compare Mode 209
11.4.2.3 Edge-Aligned PWM Mode 209
11.4.3 Center-Aligned PWM Mode 210
11.5 TPM Interrupts 211
11.5.1 Clearing Timer Interrupt Flags 211
11.5.2 Timer Overflow Interrupt Description 211
11.5.3 Channel Event Interrupt Description 212
11.5.4 PWM End-of-Duty-Cycle Events 212

Chapter 12
Serial Communications Interface (S08SCIV3)

12.1 Introduction 213
12.1.1 Features 216
12.1.2 Modes of Operation 216

12.1.3 Block Diagram 217
12.2 Register Definition 219
12.2.1 SCI Baud Rate Registers (SCIBDH, SCIBHL) 219
12.2.2 SCI Control Register 1 (SCIC1) 220
12.2.3 SCI Control Register 2 (SCIC2) 221
12.2.4 SCI Status Register 1 (SCIS1) 222
12.2.5 SCI Status Register 2 (SCIS2) 224
12.2.6 SCI Control Register 3 (SCIC3) 224
12.2.7 SCI Data Register (SCID) 225
12.3 Functional Description 226
12.3.1 Baud Rate Generation 226
12.3.2 Transmitter Functional Description 226
12.3.2.1 Send Break and Queued Idle 227
12.3.3 Receiver Functional Description 228
12.3.3.1 Data Sampling Technique 228
12.3.3.2 Receiver Wakeup Operation 229
12.3.4 Interrupts and Status Flags 229
12.4 Additional SCI Functions 230
12.4.1 8- and 9-Bit Data Modes 230
12.4.2 Stop Mode Operation 231
12.4.3 Loop Mode 231
12.4.4 Single-Wire Operation 231

Chapter 13
Serial Peripheral Interface (S08SPIV3)

13.1 Introduction 233
13.1.1 Features 235
13.1.2 Block Diagrams 235
13.1.2.1 SPI System Block Diagram 235
13.1.2.2 SPI Module Block Diagram 236
13.1.3 SPI Baud Rate Generation 237
13.2 External Signal Description 238
13.2.1 SPSCK — SPI Serial Clock 238
13.2.2 MOSI — Master Data Out, Slave Data In 238
13.2.3 MISO — Master Data In, Slave Data Out 238
13.2.4 $\overline{SS}$ — Slave Select 238
13.3 Modes of Operation 239
13.3.1 SPI in Stop Modes 239
13.4 Register Definition 239
13.4.1 SPI Control Register 1 (SPIxC1) 239
13.4.2 SPI Control Register 2 (SPIxC2) 240
13.4.3 SPI Baud Rate Register (SPIxBR) 241
13.4.4 SPI Status Register (SPIxS) 242
13.4.5 SPI Data Register (SPIxD) 243

13.5 Functional Description 244
13.5.1 SPI Clock Formats 244
13.5.2 SPI Interrupts 247
13.5.3 Mode Fault Detection 247

Chapter 14
Inter-Integrated Circuit (S08IICV1)

14.1 Introduction 249
14.1.1 Features 251
14.1.2 Modes of Operation 251
14.1.3 Block Diagram 252
14.2 External Signal Description 252
14.2.1 SCL — Serial Clock Line 252
14.2.2 SDA — Serial Data Line 252
14.3 Register Definition 252
14.3.1 IIC Address Register (IICA) 253
14.3.2 IIC Frequency Divider Register (IICF) 253
14.3.3 IIC Control Register (IICC) 256
14.3.4 IIC Status Register (IICS) 257
14.3.5 IIC Data I/O Register (IICD) 258
14.4 Functional Description 259
14.4.1 IIC Protocol 259
14.4.1.1 START Signal 260
14.4.1.2 Slave Address Transmission 260
14.4.1.3 Data Transfer 260
14.4.1.4 STOP Signal 261
14.4.1.5 Repeated START Signal 261
14.4.1.6 Arbitration Procedure 261
14.4.1.7 Clock Synchronization 261
14.4.1.8 Handshaking 262
14.4.1.9 Clock Stretching 262
14.5 Resets 262
14.6 Interrupts 262
14.6.1 Byte Transfer Interrupt 263
14.6.2 Address Detect Interrupt 263
14.6.3 Arbitration Lost Interrupt 263
14.7 Initialization/Application Information 264

Chapter 15
Analog-to-Digital Converter (S08ADC12V1)

15.1 Introduction 267
15.1.1 ADC Configuration Information 267
15.1.1.1 Channel Assignments 267
15.1.1.2 Alternate Clock 268

15.1.1.3 Hardware Trigger 268
15.1.1.4 Analog Pin Enables 268
15.1.1.5 Temperature Sensor 268
15.1.1.6 Low-Power Mode Operation 269
15.1.2 Features 270
15.1.3 Block Diagram 270
15.2 External Signal Description 271
15.2.1 Analog Power (V_{DDAD}) 272
15.2.2 Analog Ground (V_{SSAD}) 272
15.2.3 Voltage Reference High (V_{REFH}) 272
15.2.4 Voltage Reference Low (V_{REFL}) 272
15.2.5 Analog Channel Inputs (ADx) 272
15.3 Register Definition 272
15.3.1 Status and Control Register 1 (ADCSC1) 272
15.3.2 Status and Control Register 2 (ADCSC2) 274
15.3.3 Data Result High Register (ADCRH) 275
15.3.4 Data Result Low Register (ADCRL) 275
15.3.5 Compare Value High Register (ADCCVH) 276
15.3.6 Compare Value Low Register (ADCCVL) 276
15.3.7 Configuration Register (ADCCFG) 276
15.3.8 Pin Control 1 Register (APCTL1) 278
15.3.9 Pin Control 2 Register (APCTL2) 279
15.3.10 Pin Control 3 Register (APCTL3) 280
15.4 Functional Description 281
15.4.1 Clock Select and Divide Control 281
15.4.2 Input Select and Pin Control 282
15.4.3 Hardware Trigger 282
15.4.4 Conversion Control 282
15.4.4.1 Initiating Conversions 282
15.4.4.2 Completing Conversions 283
15.4.4.3 Aborting Conversions 283
15.4.4.4 Power Control 283
15.4.4.5 Sample Time and Total Conversion Time 283
15.4.5 Automatic Compare Function 285
15.4.6 MCU Wait Mode Operation 285
15.4.7 MCU Stop3 Mode Operation 285
15.4.7.1 Stop3 Mode With ADACK Disabled 285
15.4.7.2 Stop3 Mode With ADACK Enabled 286
15.4.8 MCU Stop1 and Stop2 Mode Operation 286
15.5 Initialization Information 286
15.5.1 ADC Module Initialization Example 286
15.5.1.1 Initialization Sequence 286
15.5.1.2 Pseudo — Code Example 287
15.6 Application Information 288

15.6.1 External Pins and Routing 288
15.6.1.1 Analog Supply Pins 288
15.6.1.2 Analog Reference Pins 289
15.6.1.3 Analog Input Pins 289
15.6.2 Sources of Error 290
15.6.2.1 Sampling Error 290
15.6.2.2 Pin Leakage Error 290
15.6.2.3 Noise-Induced Errors 290
15.6.2.4 Code Width and Quantization Error 291
15.6.2.5 Linearity Errors 291
15.6.2.6 Code Jitter, Non-Monotonicity and Missing Codes 292

Chapter 16
Analog Comparator (S08ACMPV2)

16.1 Introduction 293
16.1.1 ACMP/TPM1 Configuration Information 293
16.1.2 AMCPO Availability 293
16.1.3 Features 295
16.1.4 Modes of Operation 295
16.1.4.1 ACMP in Wait Mode 295
16.1.4.2 ACMP in Stop Modes 295
16.1.4.3 ACMP in Active Background Mode 295
16.1.5 Block Diagram 295
16.2 External Signal Description 297
16.3 Register Definition 297
16.3.1 ACMP Status and Control Register (ACMPSC) 298
16.4 Functional Description 299

Chapter 17
Development Support

17.1 Introduction 301
17.1.1 Features 301
17.2 Background Debug Controller (BDC) 302
17.2.1 BKGD Pin Description 302
17.2.2 Communication Details 303
17.2.3 BDC Commands 307
17.2.4 BDC Hardware Breakpoint 309
17.3 On-Chip Debug System (DBG) 310
17.3.1 Comparators A and B 310
17.3.2 Bus Capture Information and FIFO Operation 310
17.3.3 Change-of-Flow Information 311
17.3.4 Tag vs. Force Breakpoints and Triggers 311
17.3.5 Trigger Modes 312
17.3.6 Hardware Breakpoints 314

17.4 Register Definition 314
17.4.1 BDC Registers and Control Bits 314
17.4.1.1 BDC Status and Control Register (BDCSCR) 315
17.4.1.2 BDC Breakpoint Match Register (BDCBKPT) 316
17.4.2 System Background Debug Force Reset Register (SBDFR) 316
17.4.3 DBG Registers and Control Bits 317
17.4.3.1 Debug Comparator A High Register (DBGCAH) 317
17.4.3.2 Debug Comparator A Low Register (DBGCAL) 317
17.4.3.3 Debug Comparator B High Register (DBGCBH) 317
17.4.3.4 Debug Comparator B Low Register (DBGCBL) 317
17.4.3.5 Debug FIFO High Register (DBGFH) 318
17.4.3.6 Debug FIFO Low Register (DBGFL) 318
17.4.3.7 Debug Control Register (DBGC) 319
17.4.3.8 Debug Trigger Register (DBGT) 320
17.4.3.9 Debug Status Register (DBGS) 321

Appendix A
Electrical Characteristics

A.1 Introduction 323
A.2 Absolute Maximum Ratings 323
A.3 Thermal Characteristics 324
A.4 Electrostatic Discharge (ESD) Protection Characteristics 325
A.5 DC Characteristics 326
A.6 Supply Current Characteristics 330
A.7 ADC Characteristics 333
A.8 LCD Characteristics 336
A.9 Internal Clock Generation Module Characteristics 339
A.9.1 ICG Frequency Specifications 339
A.10 AC Characteristics 341
A.10.1 Control Timing 342
A.10.2 Timer/PWM (TPM) Module Timing 343
A.10.3 SPI Timing 344
A.11 FLASH Specifications 347
A.12 EMC Performance 348
A.12.1 Radiated Emissions 348
A.12.2 Conducted Transient Susceptibility 349

Appendix B
Ordering Information and Mechanical Drawings

B.1 Ordering Information 351
B.2 Mechanical Drawings 351

Chapter 1
Device Overview

1.1 Introduction

MC9S08LC60 Series MCUs are members of the low-cost, high-performance HCS08 Family of 8-bit microcontroller units (MCUs). All MCUs in the family use the enhanced HCS08 core and are available with a variety of modules, memory sizes, memory types, and package types.

1.2 Devices in the MC9S08LC60 Series

Table 1-1 lists the devices available in the MC9S08LC60 Series and summarizes the differences among them.

Table 1-1. Devices in the MC9S08LC60 Series

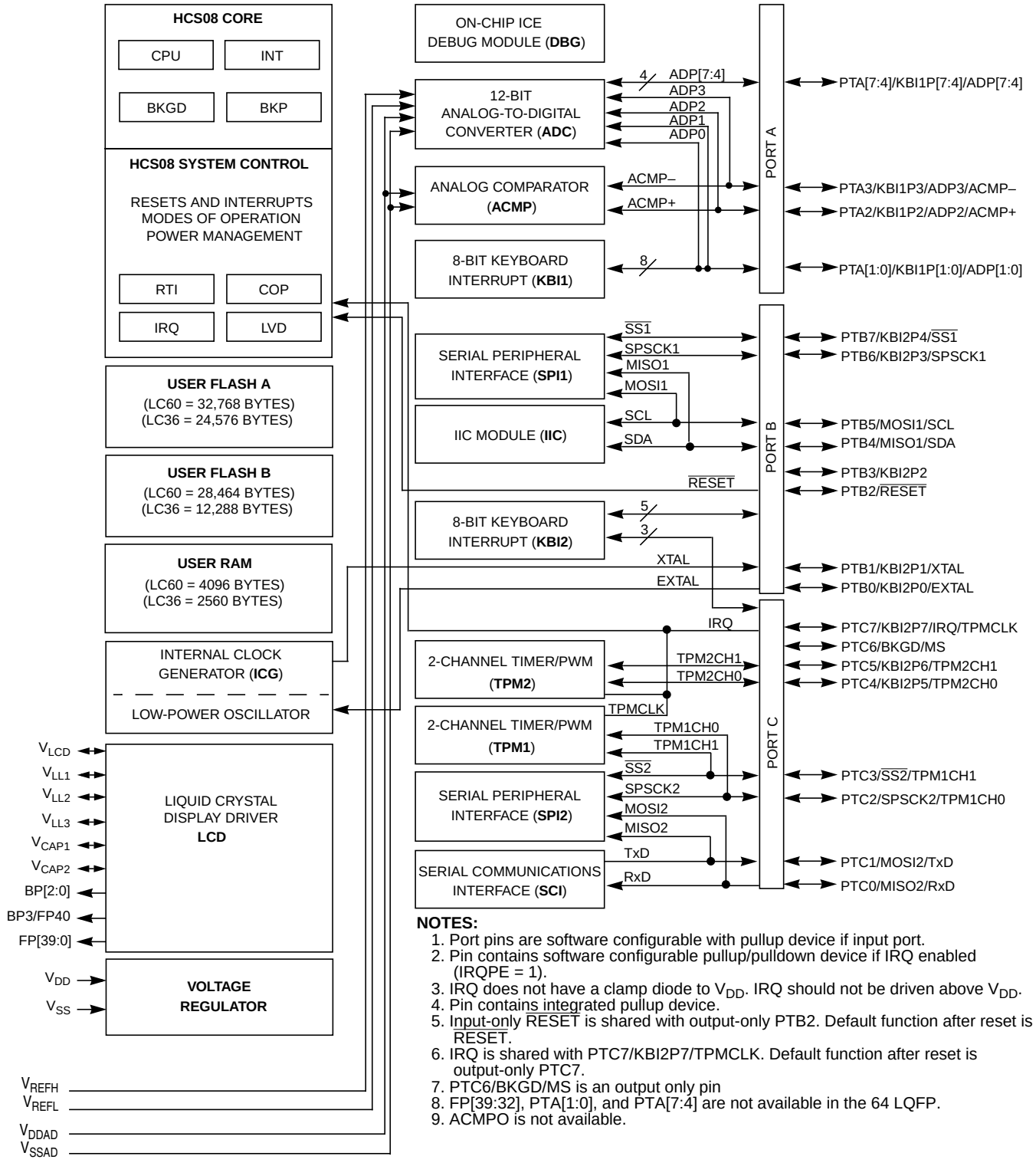
Device	FLASH A	FLASH B	RAM	Package
MC9S08LC60	32K	28K	4K	80 LQFP 64 LQFP
MC9S08LC36	24K	12K	2.5K	

Table 1-2. Package Options by Feature

Feature	Package	
	80-Pin	64-Pin
ACMP	yes	yes
ADC	8-ch	2-ch
IIC	yes	yes
IRQ	yes	yes
KBI1 KBI2	8 8	2 8
SCI	yes	yes
SPI1 SPI2	yes yes	yes yes
TPM1 TPM2	2-ch 2-ch	2-ch 2-ch
Shared I/O pins (max)	24 - I/O 2 - Output only 1- Input only	18 - I/O 2 - Output only 1- Input only
LCD	4x40 3x41	4x32 3x33

1.3 MCU Block Diagram

Figure 1-1 shows the structure of the MC9S08LC60 Series MCUs.



NOTES:
1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 1-1. MC9S08LC60 Series Block Diagram

Table 1-3 lists the functional versions of the on-chip modules.

Table 1-3. Module Versions

Module	Version
Analog Comparator (ACMP)	2
Analog-to-Digital Converter (ADC)	1
Internal Clock Generator (ICG)	4
Inter-Integrated Circuit (IIC)	1
Keyboard Interrupt (KBI)	2
Serial Communications Interface (SCI)	3
Serial Peripheral Interface (SPI)	3
Timer Pulse-Width Modulator (TPM)	2
Liquid Crystal Display Module (LCD)	1
Central Processing Unit (CPU)	2

1.4 System Clock Distribution

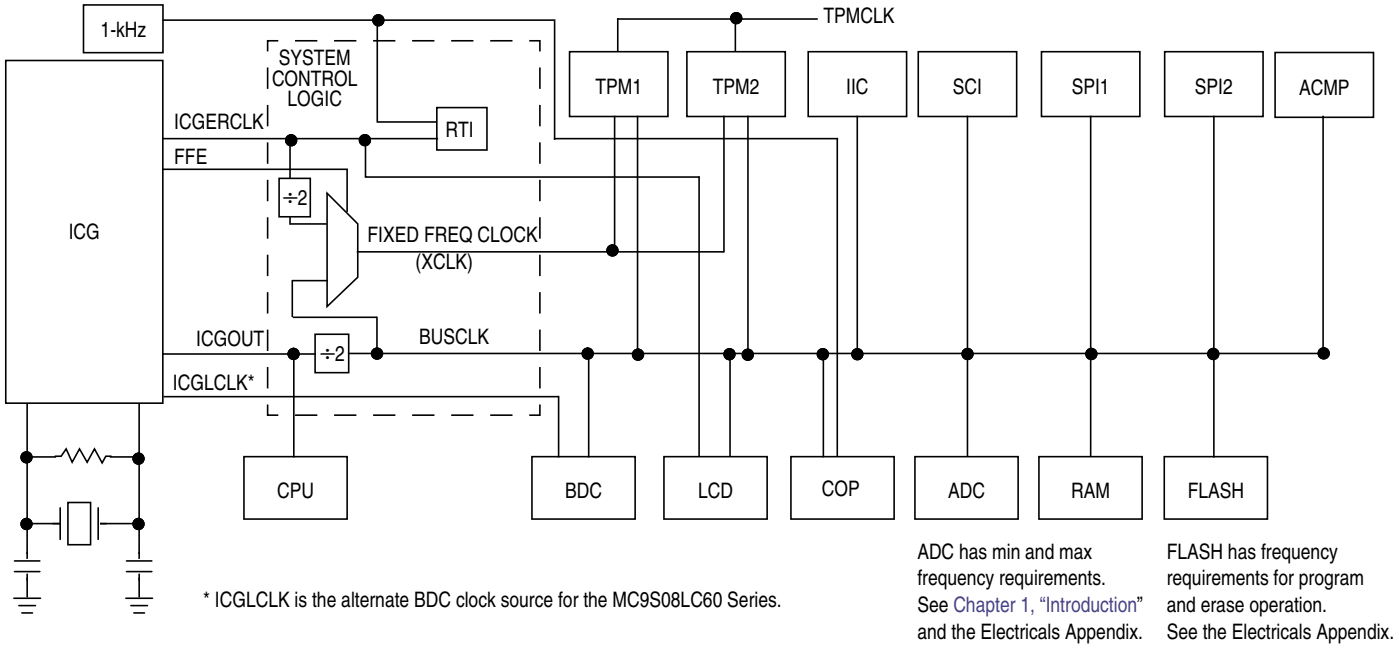


Figure 1-2. System Clock Distribution Diagram

Some of the modules inside the MCU have clock source choices. Figure 1-2 shows a simplified clock connection diagram. The ICG supplies the clock sources:

- ICGOUT is an output of the ICG module. It is one of the following:
  - The external crystal oscillator
  - An external clock source
  - The output of the digitally-controlled oscillator (DCO) in the frequency-locked loop sub-module

Control bits inside the ICG determine which source is connected.

- FFE is a control signal generated inside the ICG. If the frequency of ICGOUT > 4 × the frequency of ICGERCLK, this signal is a logic 1 and the fixed-frequency clock will be the ICGERCLK. Otherwise the fixed-frequency clock will be BUSCLK.
- ICGLCLK — Development tools can select this internal self-clocked source (~ 8 MHz) to speed up BDC communications in systems where the bus clock is slow.
- ICGERCLK — External reference clock can be selected as the real-time interrupt clock source.

Chapter 2
Pins and Connections

2.1 Introduction

This section describes signals that connect to package pins. It includes a pinout diagram, a table of signal properties, and detailed discussion of signals.

2.2 Device Pin Assignment

Figure 2-1 and Figure 2-2 show the pin assignments for the MC9S08LC60 Series devices in its available packages.

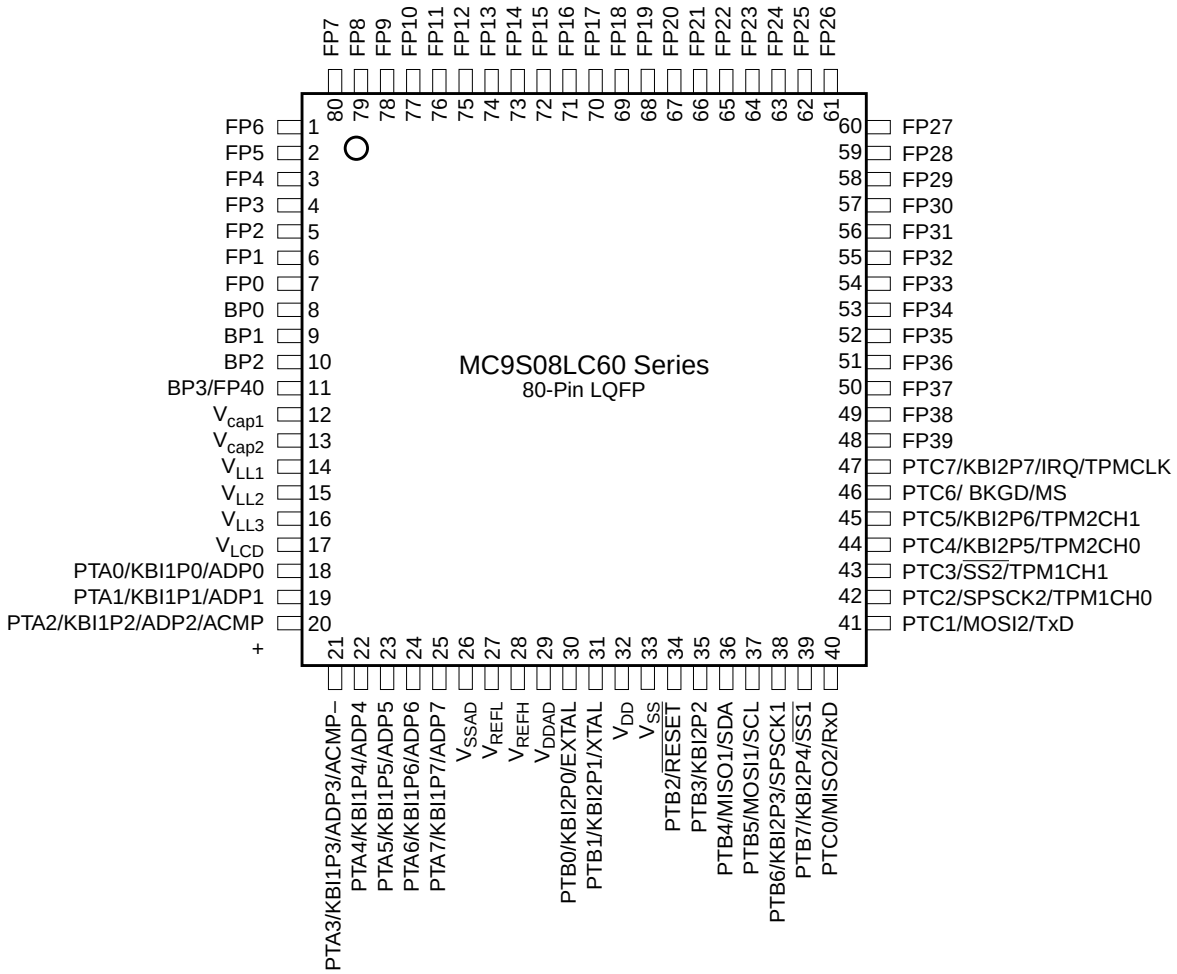
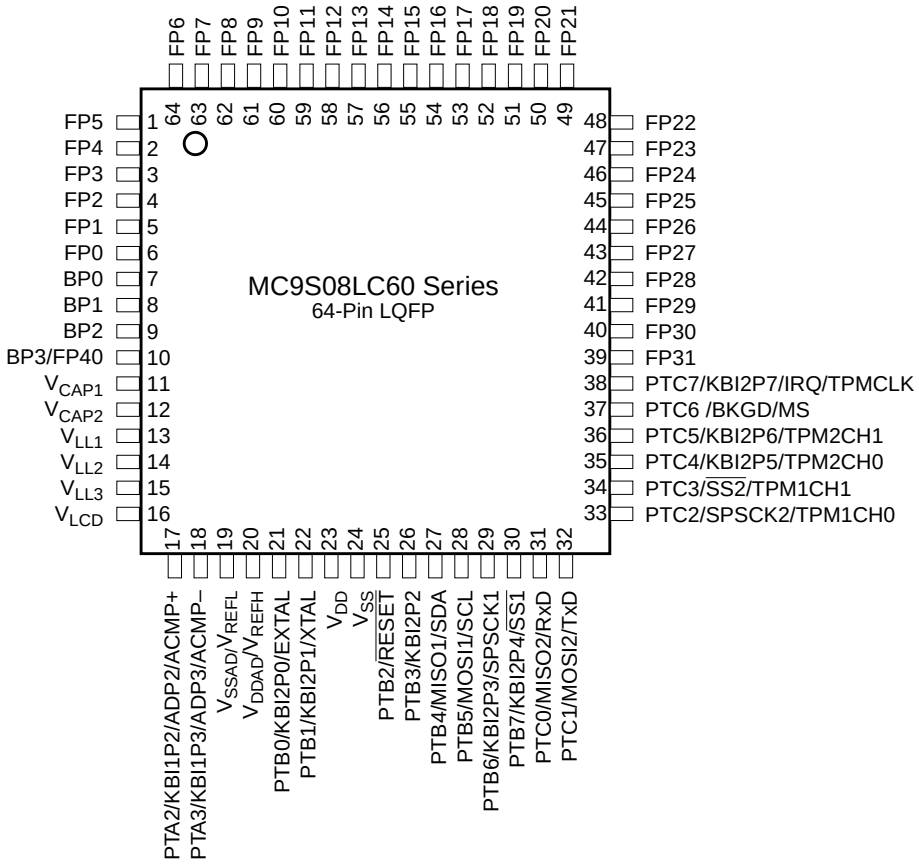


Figure 2-1. MC9S08LC60 Series in 80-Pin LQFP Package

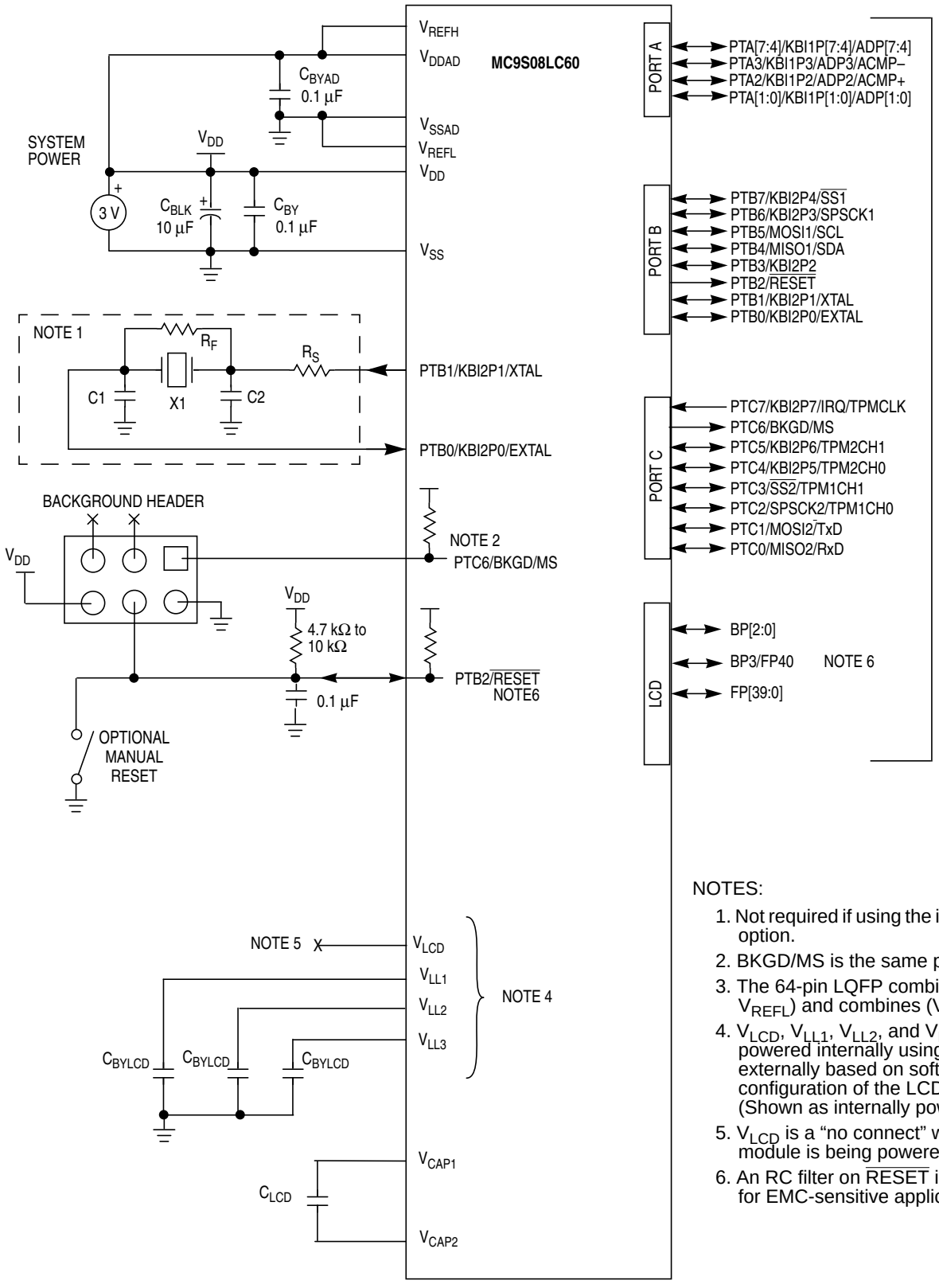


Note: V_{REFH}/V_{REFL} are internally connected to V_{DDAD}/V_{SSAD} in the 64-pin package.

Figure 2-2. MC9S08LC60 Series in 64-Pin LQFP Package

2.3 Recommended System Connections

Figure 2-3 shows pin connections that are common to most MC9S08LC60 Series application systems in the 80-pin package. Connections will be similar to the 64-pin package except for the number of I/O pins available. A more detailed discussion of system connections follows.



NOTES:

1. Not required if using the internal oscillator option.
2. BKGD/MS is the same pin as PTC6.
3. The 64-pin LQFP combines (V_{SSAD} to V_{REFL}) and combines (V_{DDAD} to V_{REFH})
4. V_{LCD}, V_{LL1}, V_{LL2}, and V_{LL3} can be powered internally using V_{DD} or externally based on software configuration of the LCD module. (Shown as internally powered).
5. V_{LCD} is a “no connect” when the LCD module is being powered via V_{DD}.
6. An RC filter on $\overline{\text{RESET}}$ is recommended for EMC-sensitive applications.

Figure 2-3. Basic System Connections

2.3.1 Power (V_{DD}, V_{SS}, V_{DDAD}, V_{SSAD})

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 regulated lower-voltage source to the CPU and other internal circuitry of the MCU.

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 close to the MCU power pins as practical to suppress high-frequency noise.

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

2.3.2 ADC Reference Pins (V_{REFH}, V_{REFL})

$V_{REFH\ and}$ V_{REFL} are the voltage reference high and voltage reference low pins, respectively, for the ADC module. In the 64-pin package, $V_{REFH\ and}$ V_{REFL} are shared with V_{DDAD} and V_{SSAD}, respectively.

2.3.3 Oscillator (XTAL, EXTAL)

Out of reset, the MCU uses an internally generated clock (self-clocked mode — f_{Self_reset}), that is approximately equivalent to an 8-MHz crystal rate. This frequency source is used during reset startup and can be enabled as the clock source for stop recovery to avoid the need for a long crystal startup delay. This MCU also contains a trimmable internal clock generator (ICG) module that can be used to run the MCU. For more information on the ICG, see Chapter 10, “Internal Clock Generator (S08ICGV4).”

The oscillator in this MCU is a Pierce oscillator that can accommodate a crystal or ceramic resonator in either of two frequency ranges selected by the RANGE bit in the ICGC1 register. Rather than a crystal or ceramic resonator, an external oscillator can be connected to the EXTAL input pin, and the XTAL output pin can be used as general I/O.

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 is used to provide a bias path to keep the EXTAL input in its linear range during crystal startup and 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 take into account printed circuit board (PCB) capacitance and MCU pin capacitance when sizing 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).

2.3.4 $\overline{\text{RESET}}$ Pin

After POR, the configuration of the PTB2/$\overline{\text{RESET}}$ pin defaults to $\overline{\text{RESET}}$. Clearing the RSTPE bit in SOPT1 register configures the pin to be the PTB2 general-purpose, output only pin. After configured as PTB2, the pin will remain PTB2 until the next reset. The $\overline{\text{RESET}}$ pin can be used to reset the MCU from an external source when the pin is driven low.

When enabled as the $\overline{\text{RESET}}$ pin (RSTPE = 1), an internal pullup device is automatically enabled. It has input hysteresis, a high current output driver, and no output slew rate control. Internal power-on reset and low-voltage reset circuitry typically make external reset circuitry unnecessary.

The PTB2/$\overline{\text{RESET}}$ pin will default to the $\overline{\text{RESET}}$ pin when a POR enters active background mode. This pin is normally connected to the standard 6-pin background debug connector so a development system can directly reset the MCU system. If desired, when the pin is configured as the RESET pin, a manual external reset can be added by supplying a simple switch to ground (pull reset pin low to force a reset).

Whenever any reset is initiated (whether from an external signal or from an internal system), the reset pin is driven low for approximately 34 cycles of f_{Self_reset}, released, and sampled again approximately 38 cycles of f_{Self_reset} later. If reset was caused by an internal source such as low-voltage reset or watchdog timeout, the circuitry expects the reset pin sample to return a logic 1. The reset circuitry decodes the cause of reset and records it by setting a corresponding bit in the system control reset status register (SRS).

In EMC-sensitive applications, an external RC filter is recommended on the reset pin. See Figure 2-3 for an example.

2.3.5 Background / Mode Select (BKGD/MS)

The background / mode select (BKGD/MS) shares its function with an output-only port pin, PTC6. While in reset, the pin functions as a mode select pin. Immediately after reset rises the pin functions as the background pin and can be used for background debug communication. While functioning as a background/mode select pin (BKGDPE = 1), the pin includes an internal pullup device, input hysteresis, a standard output driver, and no output slew rate control. When used as an I/O port, the pin is limited to output only.

If nothing is connected to this pin, the MCU will enter normal operating mode at the rising edge of reset. If a debug system is connected to the 6-pin standard background debug header, it can hold BKGD/MS low during the rising edge of reset which forces the MCU to active background mode.

The BKGD pin is used primarily for background debug controller (BDC) communications using a custom protocol that uses 16 clock cycles of the target MCU's BDC clock per bit time. The target MCU's BDC clock could be as fast as the bus clock rate, so there should never be any significant capacitance connected to the BKGD/MS pin that could interfere with background serial communications.

Although the BKGD 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 almost no role in determining rise and fall times on the BKGD pin.

2.3.6 LCD Pins

2.3.6.1 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 these pins see the LCD chapter.

2.3.6.2 LCD Frontplane and Backplane Driver Pins

44 pins are dedicated to frontplane and backplane drivers; on the 64-pin package, 36 pins are dedicated. Immediately after reset, the LCD driver pins are high-impedance. For detailed information about LCD frontplane and backplane driver pins, see the LCD chapter.

2.3.7 General-Purpose I/O and Peripheral Ports

MC9S08LC60 Series MCUs support up to 24 general-purpose I/O pins which are shared with on-chip peripheral functions (timers, serial I/O, ADC, keyboard interrupts, etc.). On each MC9S08LC60 Series device, there is one input-only and two output-only port pins. When a port pin is configured as a general-purpose output or 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 a peripheral uses the port pin as an input, software can enable a pullup device. For information about controlling these pins as general-purpose I/O pins, see Chapter 6, “Parallel Input/Output.”

Immediately after reset, all pins that are not output-only are configured as high-impedance general-purpose inputs with internal pullup devices disabled. After reset, the output-only port function is not enabled but is configured for low output drive strength with slew rate control enabled. The PTC6 pin defaults to BKGD/MS on any reset.

NOTE

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

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

Pin Number		<-- Lowest **Priority** --> Highest			
80	**64**	**Port Pin**	**Alt 1**	**Alt 2**	**Alt3**
1	64		FP6		
2	1		FP5		
3	2		FP4		
4	3		FP3		
5	4		FP2		
6	5		FP1		
7	6		FP0		
8	7		BP0		
9	8		BP1		
10	9		BP2		
11	10		BP3	FP40	
12	11		V_{cap1}		
13	12		V_{cap2}		
14	13		V_{LL1}		
15	14		V_{LL2}		
16	15		V_{LL3}		
17	16		V_{LCD}		
18	—	PTA0	KBI1P0	ADP0	
19	—	PTA1	KBI1P1	ADP1	
20	17	PTA2	KBI1P2	ADP2	ACMP+
21	18	PTA3	KBI1P3	ADP3	ACMP–
22	—	PTA4	KBI1P4	ADP4	
23	—	PTA5	KBI1P5	ADP5	
24	—	PTA6	KBI1P6	ADP6	
25	—	PTA7	KBI1P7	ADP7	
26	19				V_{SSAD}
27					V_{REFL}
28	20				V_{REFH}
29					V_{DDAD}
30	—	PTB0	KBI2P0	EXTAL	
31	—	PTB1	KBI2P1	XTAL	
32	23				V_{DD}
33	24				V_{SS}
34	25	PTB2	$\overline{\text{RESET}}$		
35	26	PTB3	KBI2P2		
36	27	PTB4	MISO1	SDA	
37	28	PTB5	MOSI1	SCL	
38	29	PTB6	KBI2P3	SPSCK1	
39	30	PTB7	KBI2P4	$\overline{\text{SS1}}$	
40	31	PTC0	MISO2	RxD	
41	32	PTC1	MOSI2	TxD	
42	33	PTC2	SPSCK2	TPM1CH0	
43	34	PTC3	$\overline{\text{SS2}}$	TPM1CH1	
44	35	PTC4	KBI2P5	TPM2CH0	
45	36	PTC5	KBI2P6	TPM2CH1	
46	37	PTC6	BKGD	MS	
47	38	PTC7	KBI2P7	IRQ	TPMCLK
48	—		FP39		
49	—		FP38		
50	—		FP37		
51	—		FP36		
52	—		FP35		
53	—		FP34		
54	—		FP33		
55	—		FP32		
56	39		FP31		
57	40		FP30		
58	41		FP29		
59	42		FP28		
60	43		FP27		
61	44		FP26		
62	45		FP25		
63	46		FP24		
64	47		FP23		
65	48		FP22		
66	49		FP21		
67	50		FP20		
68	51		FP19		
69	52		FP18		
70	53		FP17		
71	54		FP16		
72	55		FP15		
73	56		FP14		
74	57		FP13		
75	58		FP12		
76	59		FP11		
77	60		FP10		
78	61		FP9		
78	62		FP8		
80	63		FP7		

Chapter 3
Modes of Operation

3.1 Introduction

The operating modes of the MC9S08LC60 Series are described in this section. 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
- Wait mode:
  - CPU halts operation to conserve power
  - System clocks running
  - Full voltage regulation is maintained
- Stop modes:
  - CPU and bus clocks stopped
  - Stop1: Full power-down of internal circuits for maximum power savings
  - Stop2: Partial power-down of internal circuits, RAM contents retained
  - Stop3: All internal circuits powered for fast recovery; RAM and register contents are retained; LCD module can be configured to remain operational

3.3 Run Mode

Run is the normal operating mode for the MC9S08LC60 Series. This mode is selected upon the MCU exiting reset if the BKGD/MS pin is high. In this mode, the CPU executes code from internal memory with execution beginning at the address fetched from memory at 0xFFFE:0xFFFF after reset.

3.4 Active Background Mode

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

Active background mode is entered in any of five ways:

- When the BKGD/MS pin is low at the time the MCU exits reset
- When a BACKGROUND command is received through the BKGD 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 waiting for serial background commands rather than executing instructions from the user's 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's 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 MC9S08LC60 Series MCUs are shipped from the Freescale factory, the FLASH program memory is erased by default unless specifically noted, so there is no program that could 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 the condition code register (CCR) is cleared when the CPU enters wait mode, enabling interrupts. When an interrupt request occurs, the CPU exits wait mode and resumes processing, beginning with the stacking operations leading 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 while 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.

3.6 Stop Modes

One of three stop modes is entered upon execution of a STOP instruction when STOPE in SOPT1 is set. In any stop mode, the bus and CPU clocks are halted. The ICG module can be configured to leave the reference clocks running. see Chapter 10, “Internal Clock Generator (S08ICGV4)” for more information.

Table 3-1 shows all of the control bits that affect stop mode selection and the mode selected under various conditions. The stop mode behavior of the MCU is configured by setting the appropriate bits in the SPMSC1 and SPMSC2 registers The selected mode is entered following the execution of a STOP instruction.

Table 3-1. Stop Mode Selection

STOPE	ENBDM [1]	LVDE and LVDSE	PDC	PPDC	Stop Mode
0	x	x	x	x	Stop modes disabled; illegal opcode reset if STOP instruction executed
1	1	x	x	x	Stop3 with BDM enabled [2]
1	0	1	x	x	Stop3 with voltage regulator active
1	0	0	0	x	Stop3[3]
1	0	0	1	1	Stop2
1	0	0	1	0	Stop1

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

2 When in Stop3 mode with BDM enabled, The S_{IDD} will be near R_{IDD} levels because internal clocks are enabled.

3 The LCD module can operate in stop3 if LCDSTP3 in LCDCR1 is asserted.

3.6.1 Stop3 Mode

Stop3 mode is entered by executing a STOP instruction under the conditions as 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 real-time interrupt (RTI), LVD, ADC, IRQ 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.

A separate self-clocked source (≈1 kHz) for the real-time interrupt allows a wakeup from stop2 or stop3 mode with no external components. When RTIS2:RTIS1:RTIS0 = 0:0:0, the real-time interrupt function and this 1-kHz source are disabled. Power consumption is lower when the 1-kHz source is disabled, but in that case the real-time interrupt cannot wake the MCU from stop.

3.6.1.1 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) at the time the CPU executes a STOP instruction, then the voltage regulator remains active during stop mode.

For the ADC to operate the LVD must be left enabled when entering stop3.

3.6.1.2 Active BDM Enabled in Stop Mode

Entry into the active background mode from run mode is enabled if ENBDM in BDCSCR is set. This register is described in the Development Support chapter of this data sheet. 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.

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.

3.6.2 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 as in stop1 with the exception of the RAM. Upon entering stop2, all I/O pin control signals are latched so that the pins retain their states during stop2.

Exit from stop2 is performed by asserting the wake-up pins or $\overline{\text{RESET}}$ or IRQ.

NOTE

IRQ always functions as an active-low wakeup input when the MCU is in stop2, regardless of how the pin is configured before entering stop2.

In addition, the real-time interrupt (RTI) can wake the MCU from stop2 if enabled.

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

- All module control and status registers are reset
- The LVD reset function is enabled and the MCU remains in the reset state if V_{DD} is below the LVD trip point (low trip point selected due to POR)
- The CPU takes the reset vector

In addition to the above, upon waking up 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.

To maintain I/O states for pins that were configured as general-purpose I/O before entering stop2, the user must restore the contents of the I/O port registers, which have been saved in RAM, 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.

3.6.3 Stop1 Mode

Stop1 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 stop1, providing the lowest possible standby current. Upon entering stop1, all I/O pins automatically transition to their default reset states.

Exit from stop1 is performed by asserting the wake-up pins or $\overline{\text{RESET}}$ or IRQ.

NOTE

IRQ always functions as an active-low wakeup input when the MCU is in stop1, regardless of how the pin is configured before entering stop1.

In addition, the real-time interrupt (RTI) can wake the MCU from stop1 if enabled.

Upon wake-up from stop1 mode, the MCU starts up as from a power-on reset (POR):

- All module control and status registers are reset
- The LVD reset function is enabled and the MCU remains in the reset state if V_{DD} is below the LVD trip point (low trip point selected due to POR)
- The CPU takes the reset vector

In addition to the above, upon waking up from stop1, the PDF bit in SPMSC2 is set. This flag is used to direct user code to go to a stop1 recovery routine. PDF remains set until a 1 is written to PPDACK in SPMSC2.

3.6.4 On-Chip Peripheral Modules in Stop 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 3.6.3, “Stop1 Mode,” Section 3.6.2, “Stop2 Mode,” and Section 3.6.1, “Stop3 Mode,” for specific information on system behavior in stop modes.

Table 3-2. Stop Mode Behavior

Peripheral	Mode		
	Stop1	Stop2	Stop3
CPU	Off	Off	Standby
RAM	Off	Standby	Standby
FLASH	Off	Off	Standby
Parallel Port Registers	Off	Off	Standby
ADC	Off	Off	Optionally On[1]
ACMP	Off	Off	Standby
ICG	Off	Off	Optionally On[2]
IIC	Off	Off	Standby
LCD	Off	Off	Optionally On[3]
SCI	Off	Off	Standby
SPI	Off	Off	Standby
TPM	Off	Off	Standby
Voltage Regulator	Off	Standby	Standby
I/O Pins	Hi-Z	States Held	States Held

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

2 OSCSPEN set in ICGC1, else in standby.

3 LCDSTP3 = 1 in the LCDCR1 register.

Chapter 4
Memory

4.1 MC9S08LC60 Series Memory Map

As shown in Figure 4-1, on-chip memory in the MC9S08LC60 Series consists of RAM, FLASH program memory for nonvolatile data storage, plus I/O and control/status registers. The registers are divided into three groups:

- Direct-page registers (0x0000 through 0x005F)
- High-page registers (0x1800 through 0x186F)
- Nonvolatile registers (0xFFB0 through 0xFFBF)

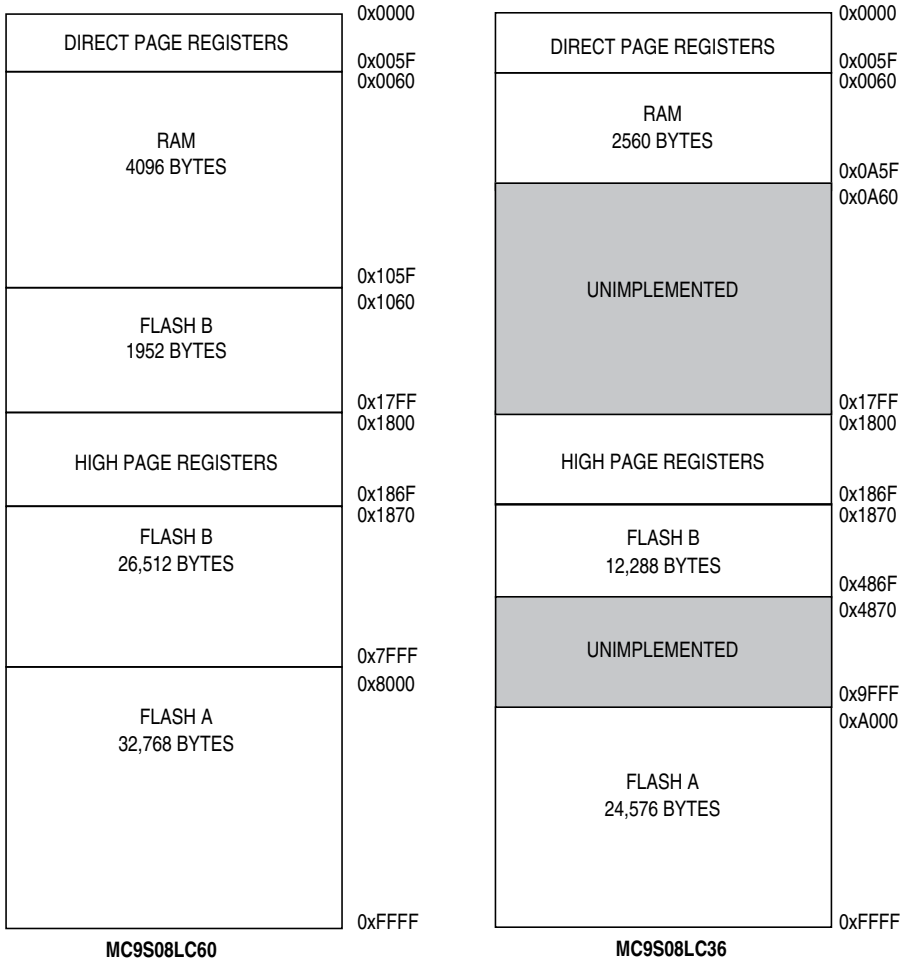


Figure 4-1. MC9S08LC60 Series Memory Map

4.1.1 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-provided equate file for the MC9S08LC60 Series. For more details about resets, interrupts, interrupt priority, and local interrupt mask controls, refer to Chapter 5, "Resets, Interrupts, and System Configuration."

Table 4-1. Reset and Interrupt Vectors

Address (High/Low)	Vector	Vector Name
0xFFC0–0xFFC1 ⇕ 0xFFD0–0xFFD1	Unused Vector Space (available for user program)	
0xFFD2–0xFFD3	LCD	Vlcd

Table 4-1. Reset and Interrupt Vectors

Address (High/Low)	Vector	Vector Name
0xFFD4–0xFFD5	RTI	Vrti
0xFFD6–0xFFD7	IIC	Viic
0xFFD8–0xFFD9	ACMP	Vacmp
0xFFDA–0xFFDB	ADC Conversion	Vadc
0xFFDC–0xFFDD	Keyboard 2	Vkeyboard2
0xFFDE–0xFFDF	Keyboard 1	Vkeyboard1
0xFFE0–0xFFE1	SCI Transmit	Vscitx
0xFFE2–0xFFE3	SCI Receive	Vscirx
0xFFE4–0xFFE5	SCI Error	Vscierr
0xFFE6–0xFFE7	SPI 2	Vspi2
0xFFE8–0xFFE9	SPI 1	Vspi1
0xFFEA–0xFFEB	TPM2 Overflow	Vtpm2ovf
0xFFEC–0xFFED	TPM2 Channel 1	Vtpm2ch1
0xFFEE–0xFFEF	TPM2 Channel 0	Vtpm2ch0
0xFFF0–0xFFF1	TPM1 Overflow	Vtpm1ovf
0xFFF2–0xFFF3	TPM1 Channel 1	Vtpm1ch1
0xFFF4–0xFFF5	TPM1 Channel 0	Vtpm1ch0
0xFFF6–0xFFF7	ICG	Vicg
0xFFF8–0xFFF9	Low Voltage Detect	Vlvd
0xFFFA–0xFFFB	IRQ	Virq
0xFFFC–0xFFFD	SWI	Vswi
0xFFFE–0xFFFF	Reset	Vreset

4.2 Register Addresses and Bit Assignments

The registers in the MC9S08LC60 Series are divided into these three groups:

- Direct-page registers are located in the first 128 locations in the memory map, so they are accessible with efficient direct addressing mode instructions.
- 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 variables.
- 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 which 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 only requires 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 4-3 and Table 4-4 the whole address in column one is shown in bold. In Table 4-2, Table 4-3, and Table 4-4, 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.

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	PTBD3	PTBD2	PTBD1	PTBD0
0x0003	**PTBDD**	PTBDD7	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2	PTBDD1	PTBDD0
0x0004	**PTCD**	PTCD7	PTCD6	PTCD5	PTCD4	PTCD3	PTCD2	PTCD1	PTCD0
0x0005	**PTCDD**	PTCDD7	PTCDD6	PTCDD5	PTCDD4	PTCDD3	PTCDD2	PTCDD1	PTCDD0
0x0006	**IRQSC**	0	IRQPDD	IRQEDG	IRQPE	IRQF	IRQACK	IRQIE	IRQMOD
0x0007	Reserved	—	—	—	—	—	—	—	—
0x0008	**KBI1SC**	0	0	0	0	KBF	KBACK	KBIE	KBIMOD
0x0009	**KBI1PE**	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
0x000A	**KBI1ES**	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBEDG3	KBEDG2	KBEDG1	KBEDG0
0x000B	Reserved	—	—	—	—	—	—	—	—
0x000C	**KBI2SC**	0	0	0	0	KBF	KBACK	KBIE	KBIMOD
0x000D	**KBI2PE**	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
0x000E	**KBI2ES**	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBEDG3	KBEDG2	KBEDG1	KBEDG0
0x000F	**ACMPSC**	ACME	ACBGS	ACF	ACIE	ACO	R[1]	ACMOD	
0x0010	**ADCSC1**	COCO	AIEN	ADCO	ADCH				
0x0011	**ADCSC2**	ADACT	ADTRG	ACFE	ACFGT	—	—	—	—
0x0012	**ADCRH**	0	0	0	0	ADR11	ADR10	ADR9	ADR8
0x0013	**ADCRL**	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
0x0014	**ADCCVH**	0	0	0	0	ADCV11	ADCV10	ADCV9	ADCV8
0x0015	**ADCCVL**	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0
0x0016	**ADCCFG**	ADLPC	ADIV		ADLSMP	MODE		ADICLK	
0x0017	**APCTL1**	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
0x0018	**IICA**	ADDR							0
0x0019	**IICF**	MULT		ICR					
0x001A	**IICC**	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
0x001B	**IICS**	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
0x001C	**IICD**	DATA							
0x001D	Reserved	—	—	—	—	—	—	—	—
0x001E	Reserved	—	—	—	—	—	—	—	—
0x001F	Reserved	—	—	—	—	—	—	—	—
0x0020	**SCIBDH**	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
0x0021	**SCIBDL**	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
0x0022	**SCIC1**	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
0x0023	**SCIC2**	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
0x0024	**SCIS1**	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
0x0025	**SCIS2**	0	0	0	0	0	BRK13	0	RAF
0x0026	**SCIC3**	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
0x0027	**SCID**	Bit 7	6	5	4	3	2	1	Bit 0
0x0028	**SPI1C1**	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
0x0029	**SPI1C2**	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0

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

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x002A	**SPI1BR**	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
0x002B	**SPI1S**	SPRF	0	SPTEF	MODF	0	0	0	0
0x002C	Reserved	0	0	0	0	0	0	0	0
0x002D	**SPI1D**	Bit 7	6	5	4	3	2	1	Bit 0
0x002E	Reserved	0	0	0	0	0	0	0	0
0x002F	Reserved	0	0	0	0	0	0	0	0
0x0030	**SPI2C1**	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
0x0031	**SPI2C2**	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
0x0032	**SPI2BR**	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
0x0033	**SPI2S**	SPRF	0	SPTEF	MODF	0	0	0	0
0x0034	Reserved	0	0	0	0	0	0	0	0
0x0035	**SPI2D**	Bit 7	6	5	4	3	2	1	Bit 0
0x0036	Reserved	0	0	0	0	0	0	0	0
0x0037	Reserved	0	0	0	0	0	0	0	0
0x0038	**ICGC1**	HGO	RANGE	REFS	CLKS		OSCSTEN	LOCD	0
0x0039	**ICGC2**	LOLRE	MFD			LOCRE	RFD		
0x003A	**ICGS1**	CLKST		REFST	LOLS	LOCK	LOCS	ERCS	ICGIF
0x003B	**ICGS2**	0	0	0	0	0	0	0	DCOS
0x003C	**ICGFLTU**	0	0	0	0	FLT			
0x003D	**ICGFLTL**	FLT							
0x003E	**ICGTRM**	TRIM							
0x003F	Reserved	0	0	0	0	0	0	0	0
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	Reserved	—	—	—	—	—	—	—	—
0x004C	Reserved	—	—	—	—	—	—	—	—
0x004D	Reserved	—	—	—	—	—	—	—	—
0x004E	Reserved	—	—	—	—	—	—	—	—
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
0x0053	**TPM2MODH**	Bit 15	14	13	12	11	10	9	Bit 8

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

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
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–0x005F	Reserved	—	—	—	—	—	—	—	—

[1] For the MC9S08LC60 Series, the AMCPO pin is not available, so the ACOPE bit in the ACMPSC register is reserved and does not have any effect.

High-page registers, shown in Table 4-3, 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 4-3. High-Page Register Summary

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x1800	**SRS**	POR	PIN	COP	ILOP	0	ICG	LVD	0
0x1801	**SBDFR**	0	0	0	0	0	0	0	BDFR
0x1802	**SOPT1**	COPE	COPT	STOPE	—	0	0	BKGDPE	RSTPE
0x1803	**SOPT2**	COPCLKS	0	0	0	0	0	0	ACIC
0x1804	Reserved	—	—	—	—	—	—	—	—
0x1805	Reserved	—	—	—	—	—	—	—	—
0x1806	**SDIDH**	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
0x1807	**SDIDL**	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
0x1808	**SRTISC**	RTIF	RTIACK	RTICLKS	RTIE	0	RTIS2	RTIS1	RTIS0
0x1809	**SPMSC1**	LVDF	LVDACK	LVDIE	LVDRE	LVDSE	LVDE	0	BGBE
0x180A	**SPMSC2**	0	0	0	PDF	PPDF	PPDACK	PDC	PPDC
0x180B	Reserved	—	—	—	—	—	—	—	—
0x180C	**SPMSC3**	LVWF	LVWACK	LVDV	LVWV	0	0	0	0
0x180D–0x180F	Reserved	—	—	—	—	—	—	—	—
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	**DBGFH**	Bit 15	14	13	12	11	10	9	Bit 8
0x1815	**DBGFL**	Bit 7	6	5	4	3	2	1	Bit 0
0x1816	**DBGC**	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
0x1817	**DBGT**	TRGSEL	BEGIN	0	0	TRG3	TRG2	TRG1	TRG0
0x1818	**DBGS**	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0
0x1819–0x181F	Reserved	—	—	—	—	—	—	—	—
0x1820	**FCDIV**	DIVLD	PRDIV8	DIV5	DIV4	DIV3	DIV2	DIV1	DIV0
0x1821	**FOPT**	KEYEN	FNORED	0	0	0	0	SEC01	SEC00
0x1822	Reserved	—	—	—	—	—	—	—	—
0x1823	**FCNFG**	0	0	KEYACC	0	0	0	0	0

Table 4-3. High-Page Register Summary (continued)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x1824	**FPROT**	FPS7	FPS6	FPS5	FPS4	FPS3	FPS2	FPS1	FPDIS
0x1825	**FSTAT**	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
0x1826	**FCMD**	FCMD7	FCMD6	FCMD5	FCMD4	FCMD3	FCMD2	FCMD1	FCMD0
0x1827–0x182F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
0x1830	**PTAPE**	PTAPE7	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
0x1831	**PTASE**	PTASE7	PTASE6	PTASE5	PTASE4	PTASE3	PTASE2	PTASE1	PTASE0
0x1832	**PTADS**	PTADS7	PTADS6	PTADS5	PTADS4	PTADS3	PTADS2	PTADS1	PTADS0
0x1833	Reserved	0	0	0	0	0	0	0	0
0x1834	**PTBPE**	PTBPE7	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
0x1835	**PTBSE**	PTBSE7	PTBSE6	PTBSE5	PTBSE4	PTBSE3	PTBSE2	PTBSE1	PTBSE0
0x1836	**PTBDS**	PTBDS7	PTBDS6	PTBDS5	PTBDS4	PTBDS3	PTBDS2	PTBDS1	PTBDS0
0x1837	Reserved	0	0	0	0	0	0	0	0
0x1838	**PTCPE**	PTCPE7	PTCPE6	PTCPE5	PTCPE4	PTCPE3	PTCPE2	PTCPE1	PTCPE0
0x1839	**PTCSE**	PTCSE7	PTCSE6	PTCSE5	PTCSE4	PTCSE3	PTCSE2	PTCSE1	PTCSE0
0x183A	**PTCDS**	PTCDS7	PTCDS6	PTCDS5	PTCDS4	PTCDS3	PTCDS2	PTCDS1	PTCDS0
0x183B-0x187F	Reserved	—	—	—	—	—	—	—	—
0x1840	**LCDCR0**	LCDEN	LPWAVE	LCLK2	LCLK1	LCLK0	0	DUTY1	DUTY0
0x1841	**LCDCR1**	LCDIEN	0	0	0	0	0	LCDWAI	LCDSTP3
0x1842	**FPENR0**	FP7EN	FP6EN	FP5EN	FP4EN	FP3EN	FP2EN	FP1EN	FP0EN
0x1843	**FPENR1**	FP15EN	FP14EN	FP13EN	FP12EN	FP11EN	FP10EN	FP9EN	FP8EN
0x1844	**FPENR2**	FP23EN	FP22EN	FP21EN	FP20EN	FP19EN	FP18EN	FP17EN	FP16EN
0x1845	**FPENR3**	FP31EN	FP30EN	FP29EN	FP28EN	FP27EN	FP26EN	FP25EN	FP24EN
0x1846	**FPENR4**	FP39EN	FP38EN	FP37EN	FP36EN	FP35EN	FP34EN	FP33EN	FP32EN
0x1847	**FPENR5**	0	0	0	0	0	0	0	FP40EN
0x1848	**LCDRAM0**	FP1BP3	FP1BP2	FP1BP1	FP1BP0	FP0BP3	FP0BP2	FP0BP1	FP0BP0
0x1849	**LCDRAM1**	FP3BP3	FP3BP2	FP3BP1	FP3BP0	FP2BP3	FP2BP2	FP2BP1	FP2BP0
0x184A	**LCDRAM2**	FP5BP3	FP5BP2	FP5BP1	FP5BP0	FP4BP3	FP4BP2	FP4BP1	FP4BP0
0x184B	**LCDRAM3**	FP7BP3	FP7BP2	FP7BP1	FP7BP0	FP6BP3	FP6BP2	FP6BP1	FP6BP0
0x184C	**LCDRAM4**	FP9BP3	FP9BP2	FP9BP1	FP9BP0	FP8BP3	FP8BP2	FP8BP1	FP8BP0
0x184D	**LCDRAM5**	FP11BP3	FP11BP2	FP11BP1	FP11BP0	FP10BP3	FP10BP2	FP10BP1	FP10BP0
0x184E	**LCDRAM6**	FP13BP3	FP13BP2	FP13BP1	FP13BP0	FP12BP3	FP12BP2	FP12BP1	FP12BP0
0x184F	**LCDRAM7**	FP15BP3	FP15BP2	FP15BP1	FP15BP0	FP14BP3	FP14BP2	FP14BP1	FP14BP0
0x1850	**LCDRAM8**	FP17BP3	FP17BP2	FP17BP1	FP17BP0	FP16BP3	FP16BP2	FP16BP1	FP16BP0
0x1851	**LCDRAM9**	FP19BP3	FP19BP2	FP19BP1	FP19BP0	FP18BP3	FP18BP2	FP18BP1	FP18BP0
0x1852	**LCDRAM10**	FP21BP3	FP21BP2	FP21BP1	FP21BP0	FP20BP3	FP20BP2	FP20BP1	FP20BP0
0x1853	**LCDRAM11**	FP23BP3	FP23BP2	FP23BP1	FP23BP0	FP22BP3	FP22BP2	FP22BP1	FP22BP0
0x1854	**LCDRAM12**	FP25BP3	FP25BP2	FP25BP1	FP25BP0	FP24BP3	FP24BP2	FP24BP1	FP24BP0
0x1855	**LCDRAM13**	FP27BP3	FP27BP2	FP27BP1	FP27BP0	FP26BP3	FP26BP2	FP26BP1	FP26BP0
0x1856	**LCDRAM14**	FP29BP3	FP29BP2	FP29BP1	FP29BP0	FP28BP3	FP28BP2	FP28BP1	FP28BP0
0x1857	**LCDRAM15**	FP31BP3	FP31BP2	FP31BP1	FP31BP0	FP30BP3	FP30BP2	FP30BP1	FP30BP0
0x1858	**LCDRAM16**	FP33BP3	FP33BP2	FP33BP1	FP33BP0	FP32BP3	FP32BP2	FP32BP1	FP32BP0
0x1859	**LCDRAM17**	FP35BP3	FP35BP2	FP35BP1	FP35BP0	FP34BP3	FP34BP2	FP34BP1	FP34BP0
0x185A	**LCDRAM18**	FP37BP3	FP37BP2	FP37BP1	FP37BP0	FP36BP3	FP36BP2	FP36BP1	FP36BP0
0x185B	**LCDRAM19**	FP39BP3	FP39BP2	FP39BP1	FP39BP0	FP38BP3	FP38BP2	FP38BP1	FP38BP0
0x185C	**LCDRAM20**	0	0	0	0	FP40BP3	FP40BP2	FP40BP1	FP40BP0

Table 4-3. High-Page Register Summary (continued)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x185D–0x1861	Reserved	—	—	—	—	—	—	—	—
0x1862	LCDCLKS	SOURCE	DIV16	CLKADJ5	CLKADJ4	CLKADJ3	CLKADJ2	CLKADJ1	CLKADJ0
0x1863	LCDSUPPLY	LCDCPEN	LCDCPMS	CPCADJ1	CPCADJ0	HDRVBUF	BBYPASS	VSUPPLY1	VSUPPLY0
0x1864	LCDBCTL	BLINK	0	0	0	BLKMODE	BRATE2	BRATE1	BRATE0
0x1865	LCDCMD	LCDIF	0	0	0	LCDDRMS	0	LCDCLR	BLANK
0x1866–0x186F	Reserved	—	—	—	—	—	—	—	—

Nonvolatile FLASH registers, shown in Table 4-4, are located in the FLASH memory. These registers include an 8-byte backdoor key which optionally can be used to gain access to secure memory resources. During reset events, the contents of NVPROT and NVOPT in the nonvolatile register area of the FLASH memory are transferred into corresponding FPROT and FOPT working registers in the high-page registers to control security and block protection options.

Table 4-4. Nonvolatile Register Summary

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0xFFB0–0xFFB7	NVBACKKEY	8-Byte Comparison Key							
0xFFB8–0xFFBC	Reserved	—	—	—	—	—	—	—	—
0xFFBD	NVPROT	FPS7	FPS6	FPS5	FPS4	FPS3	FPS2	FPS1	FPDIS
0xFFBE	NVICGTRM[1]	NVTRIM							
0xFFBF	NVOPT	KEYEN	FNORED	0	0	0	0	SEC01	SEC00

1 Freescale Semiconductor provides a factory trim to set the FIRG to 243 kHz. If user code changes the value of the NVICGTRM register, the factory trim value will be lost. User code should save the content of NVICGTRM before any mass erase operation.

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 (SEC01:SEC00) to the unsecured state (1:0).

4.3 RAM

The MC9S08LC60 Series includes 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.

The RAM retains data when the MCU is in low-power wait, stop2, or stop3 mode. At power-on or after wakeup from stop1, 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.

For compatibility with older M68HC05 MCUs, the HCS08 resets the stack pointer to 0x00FF. In the MC9S08LC60 Series, it is usually best to re-initialize 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-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 4.5, “Security” for a detailed description of the security feature.

4.4 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 MC9S08LC60 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.

4.4.1 Features

Features of the FLASH memory include:

- FLASH Size
  - MC9S08LC60 — 63,232 bytes (28,464 bytes in Flash B, 32,768 bytes in Flash A)
  - MC9S08LC36 — 36,864 bytes (12,288 bytes in Flash B, 24,576 bytes in Flash A)
- Single power supply program and erase
- Command interface for fast program and erase operation
- Up to 100,000 program/erase cycles at typical voltage and temperature
- Flexible block protection
- Security feature for FLASH and RAM
- Auto power-down for low-frequency read accesses to minimize run I_{DD}
- FLASH read/program/erase over full operating voltage or temperature

4.4.2 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 4.6.1, “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 4-5 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 4-5. 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

4.4.3 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. In some boundary conditions with RAM or high page registers, the size of a block that is accessible to the user is less than 512 bytes.

   NOTE

   Do not program any byte in the FLASH more than once after a successful erase operation. Reprogramming bits in a byte which 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 4-2 is a flowchart for executing all of the commands except for burst programming. The FCDIV register must be initialized before using any FLASH commands.

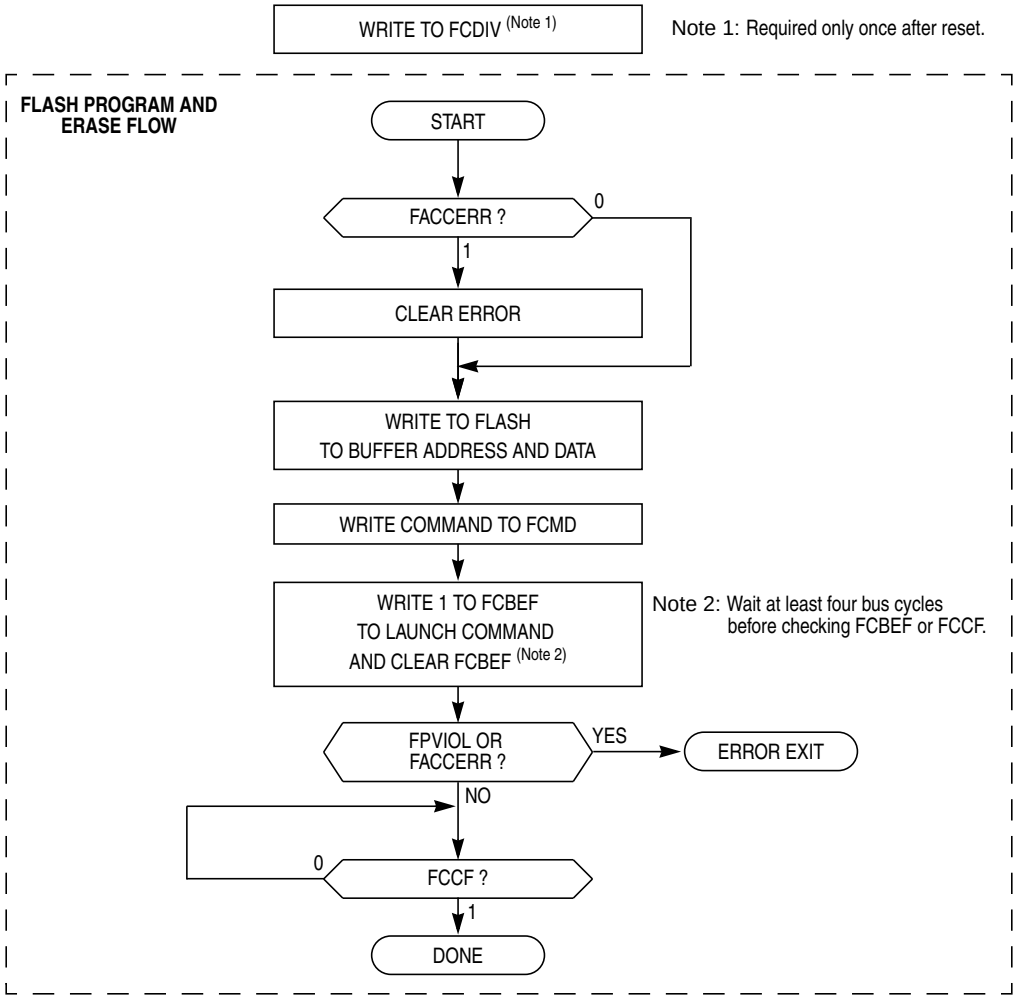


Figure 4-2. FLASH Program and Erase Flowchart

4.4.4 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. For the MC9S08LC60 Series, it is possible to burst across FLASH array boundaries as long as the addresses are consecutive. 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.

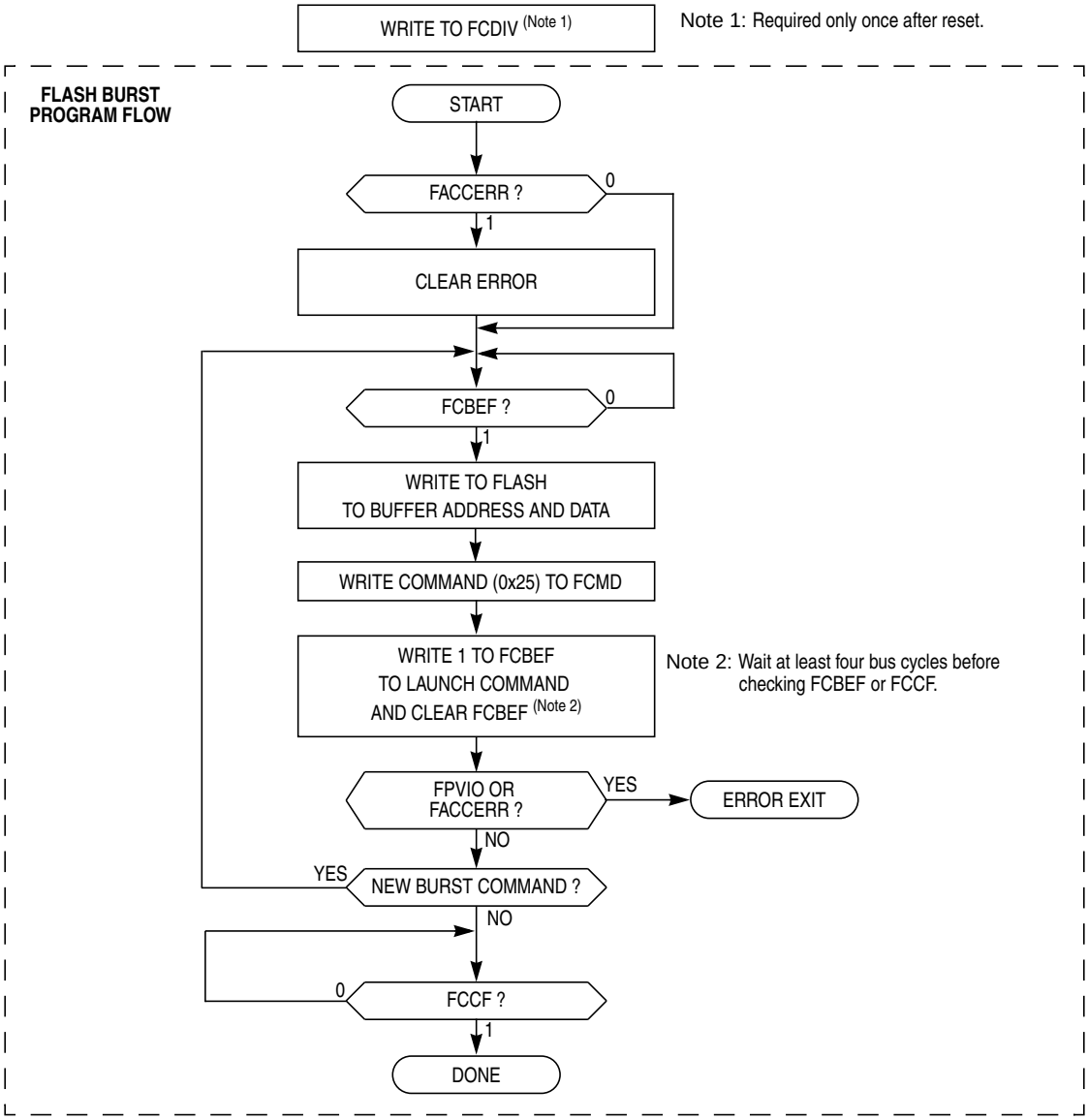


Figure 4-3. FLASH Burst Program Flowchart

4.4.5 Access Errors

An access error occurs whenever the command execution protocol is violated. Any of the following specific 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 be started 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 other than the five allowed codes (0x05, 0x20, 0x25, 0x40, or 0x41) to FCMD
- Accessing (read or write) 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 (0x20, 0x25, or 0x40) with a background debug command while the MCU is secured (The background debug controller can do blank check and mass erase commands only while the MCU is secure.)
- Writing 0 to FCBEF to cancel a partial command

4.4.6 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 4.6.4, “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. In user mode, if FPDIS is set, all FPROT bits are writeable. In user mode, if FPDIS is clear, the FPS bits are writeable as long as the size of the protected region is being increased. Because NVPROT is within the last sector 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 provide a way to erase and reprogram protected FLASH memory.

The block protection mechanism is illustrated in Figure 4-4. 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, in order to protect the last 8192 bytes of memory (addresses 0xE000 through 0xFFFF), the FPS bits must be set to 1101 111 which results in the value 0xDFFF 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 0xDE must be programmed into NVPROT to protect addresses 0xE000 through 0xFFFF.

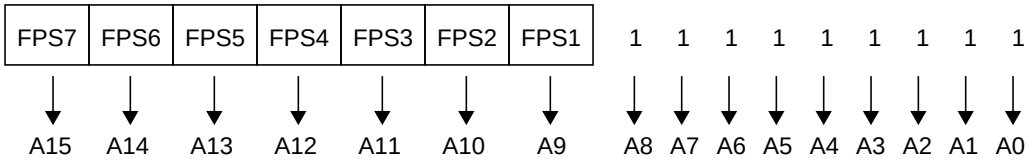


Figure 4-4. Block Protection Mechanism

One use for block protection is to block protect an area of FLASH memory for a bootloader program. This bootloader program can call a routine outside of FLASH that can be used to sector erase and re-program the rest of the FLASH memory. The bootloader is protected even if MCU power is lost during an erase and reprogram operation.

4.4.7 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, while the reset vector (0xFFFE–0xFFFF) is not. When more than 32K is protected, vector redirection must not be enabled.

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. Now, if an SPI interrupt is taken for instance, the values in the locations 0xFDE0–0xFDE1 are used for the vector instead of the values in the locations 0xFFE0–0xFFE1. 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.

4.5 Security

The MC9S08LC60 Series includes 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 while the other three combinations engage security. Notice the erased state (1:1)

makes the MCU secure. During development, whenever the FLASH is erased, it is good practice to 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, but the MCU cannot enter active background mode except by holding BKGD/MS low at the rising edge of reset.

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 just 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 performing 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.

4.6 FLASH Registers and Control Bits

The FLASH module consists of high-page including nonvolatile registers that are copied into corresponding high-page control registers at reset. There is also an 8-byte comparison key in FLASH memory. Refer to Table 4-3 and Table 4-4 for the absolute address assignments for all FLASH registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.

4.6.1 FLASH Clock Divider Register (FCDIV)

Bit 7 of this register is a read-only status flag. Bits 6 through 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.

	7	6	5	4	3	2	1	0
R	DIVLD	PRDIV8	DIV5	DIV4	DIV3	DIV2	DIV1	DIV0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-5. FLASH Clock Divider Register (FCDIV)

Table 4-6. FCDIV 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 DIV[5:0]	**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 DIV5:DIV0 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. Table 4-7 shows the appropriate values for PRDIV8 and DIV5:DIV0 for selected bus frequencies.

$$\text{if PRDIV8 = 0 — } f_{FCLK} = f_{Bus} \div ([DIV5:DIV0] + 1) \qquad \textit{Eqn. 4-1}$$

$$\text{if PRDIV8 = 1 — } f_{FCLK} = f_{Bus} \div (8 \times ([DIV5:DIV0] + 1)) \qquad \textit{Eqn. 4-2}$$

Table 4-7 shows the appropriate values for PRDIV8 and DIV5:DIV0 for selected bus frequencies.

Table 4-7. FLASH Clock Divider Settings

f_{Bus}	PRDIV8 (Binary)	DIV5:DIV0 (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

4.6.2 FLASH Options Register (FOPT and NVOPT)

During reset, the contents of the nonvolatile location NVOPT are copied from FLASH into FOPT. Bits 5 through 2 are not used and always read 0. This register may be read at any time, but writes have no meaning or effect. 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 4-6. FLASH Options Register (FOPT)

Table 4-8. FOPT 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 4.5, “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	**Flash No Redirect (**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 4-9. 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. For more detailed information about security, refer to Section 4.5, “Security.”

Table 4-9. Security States

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

[1] The 0:1 bit pattern is the recommended value to be used since it requires two bit changes before going to the unsecured state.

4.6.3 FLASH Configuration Register (FCNFG)

Bits 7 through 5 may be read or written at any time. Bits 4 through 0 always read 0 and cannot be written.

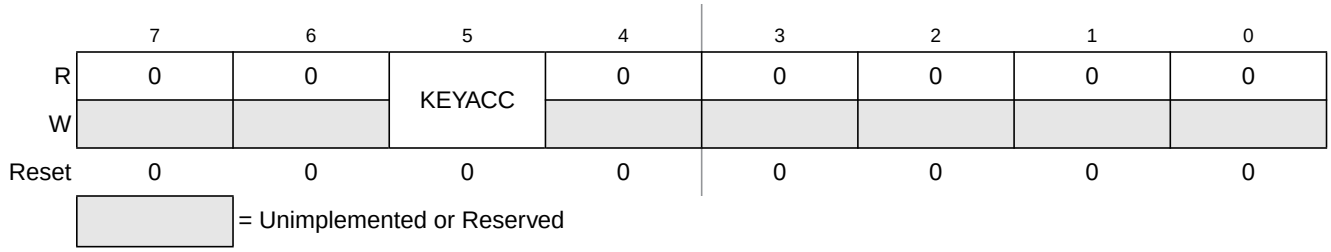


Figure 4-7. FLASH Configuration Register (FCNFG)

Table 4-10. FCNFG 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 4.5, “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.

4.6.4 FLASH Protection Register (FPROT and NVPROT)

The FPROT register defines which FLASH sectors are protected against program and erase operations. The FPROT register is also used to determine whether FLASH protection is disabled.

During the reset sequence, the FPROT register is loaded from the nonvolatile location NVPROT. To change the protection that will be loaded during the reset sequence, the sector containing NVPROT must be unprotected and erased, then NVPROT can be reprogrammed. With FPDIS set all FPROT 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 write to FPROT that attempts to decrease the size of the protected memory will be ignored.

Trying to alter data in any protected area will result in a protection violation error and the FPVIOL flag will be set in the FSTAT register. Mass erase is not possible if any one of the sectors is protected.

In order to change the data flash block protection on a temporary basis, the FPROT register EPS bits can be written to. To change the data flash block protection that will be loaded during the reset sequence, the

FLASH block must first be unprotected, then 0xFFBD in the flash configuration field must be reprogrammed.

	7	6	5	4	3	2	1	0
R	FPS7	FPS6	FPS5	FPS4	FPS3	FPS2	FPS1	FPDIS
W	(Note 1)	(Note 1)	(Note 1)	(Note 1)	(Note 1)	(Note 1)	(Note 1)	(Note 1)
Reset	This register is loaded from nonvolatile location NVPROT during reset.							

= Unimplemented or Reserved

Figure 4-8. FLASH Protection Register (FPROT)

1 If FPDIS is set, these bits are writeable in user mode. Background commands can be used to change the contents of these bits in FPROT in any mode.

Table 4-11. FPROT 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 FPS[7:1] is block protected (program and erase not allowed). 1 No FLASH block is protected.

4.6.5 FLASH Status Register (FSTAT)

Bits 3, 1, and 0 always read 0 and writes have no meaning or effect. The remaining five bits are status bits that can be read at any time. Writes to these bits have special meanings that are discussed in the bit descriptions.

	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 4-9. FLASH Status Register (FSTAT)

Table 4-12. FSTAT 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 may 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	**FLASH Protection Violation Flag** — FPVIOL is set automatically when a command 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.
4 FACCERR	**FLASH Access Error Flag** — FACCERR is set automatically when the proper command sequence is not followed 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 4.4.5, “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 has occurred. 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 0x00FF).

4.6.6 FLASH Command Register (FCMD)

Only five command codes are recognized in normal user modes as shown in Table 4-14. Refer to Section 4.4.3, “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	FCMD7	FCMD6	FCMD5	FCMD4	FCMD3	FCMD2	FCMD1	FCMD0
Reset	0	0	0	0	0	0	0	0

Figure 4-10. FLASH Command Register (FCMD)

Table 4-13. FCMD Field Descriptions

Field	Description
7:0 FCMD[7:0]	See Table 4-14 for a description of FCMD[7:0].

Table 4-14. FLASH Commands[1]

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

1 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. Blank check is required only as part of the security unlocking mechanism.

Chapter 5
Resets, Interrupts, and System Configuration

5.1 Introduction

This section discusses basic reset and interrupt mechanisms and the various sources of reset and interrupts in the MC9S08LC60 Series. Some interrupt sources from peripheral modules are discussed in greater detail within other sections of this data manual. This section gathers basic information about all reset and interrupt sources in one place for easy reference. A few reset and interrupt sources, including the computer operating properly (COP) watchdog and real-time interrupt (RTI), are not part of on-chip peripheral systems with their own sections but are part of the system control logic.

5.2 Features

Reset and interrupt features include:

- Multiple sources of reset for flexible system configuration and reliable operation:
  - Power-on detection (POR)
  - Low voltage detection (LVD) with enable
  - External $\overline{\text{RESET}}$ pin with enable
  - COP watchdog with enable and two timeout choices
  - Illegal opcode
  - Serial command from a background debug host
- Reset status register (SRS) to indicate source of most recent reset
- Separate interrupt vectors for each module (reduces polling overhead) (see Table 5-2)

5.3 MCU Reset

Resetting the MCU provides a way to start processing from a 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 MC9S08LC60 Series has seven sources for reset:

- Power-on reset (POR)
- Low-voltage detect (LVD)
- Computer operating properly (COP) timer

- Illegal opcode detect
- Background debug forced reset
- External pin reset (PIN) — can be disabled using RSTPE in SOPT2
- Clock generator loss of lock and loss of clock reset

Each of these sources, with the exception of the background debug forced reset, has an associated bit in the system reset status register. Whenever the MCU enters reset, the internal clock generator (ICG) module switches to self-clocked mode with the frequency of f_{Self_reset} selected. The reset pin is driven low for 34 internal bus cycles where the internal bus frequency is half the ICG frequency. After the 34 cycles are completed, the pin is released and will be pulled up by the internal pullup resistor, unless it is held low externally. After the pin is released, it is sampled after another 38 cycles to determine whether the reset pin is the cause of the MCU reset.

5.4 Computer Operating Properly (COP) Watchdog

The COP watchdog is intended to 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 timer periodically. If the application program gets lost and fails to reset the COP 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 (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 (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 1kHz 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 1kHz 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_{RTI} = 1 ms. See t_{RTI} in the appendix Section A.10.1, “Control Timing,” 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.

When the bus clock source is selected, the COP counter does not increment while the MCU is in background debug mode or while the system is in stop mode. The COP counter resumes once the MCU exits background debug mode.

When the 1-kHz clock source is selected, the COP counter is re-initialized to zero upon entry to either background debug mode or stop mode. The COP counter begins from zero once the MCU exits background debug mode or 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 resumes 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 until and unless the local interrupt enable is set to 1 to enable the interrupt. 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 masks (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 follows 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 may 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 off the stack.

NOTE

For compatibility with the M68HC08, 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 just before the RTI that is used to return from the ISR.

When two or more interrupts are pending when the I bit is cleared, the highest priority source is serviced first (see Table 5-2).

5.5.1 Interrupt Stack Frame

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

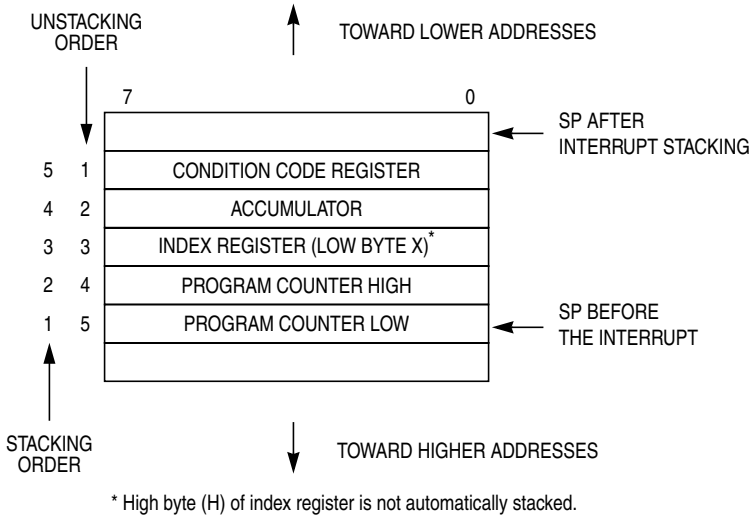


Figure 5-1. Interrupt Stack Frame

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

The status flag causing the interrupt must be acknowledged (cleared) before returning from the ISR. Typically, the flag should be 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 IRQSC status and control register. 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 (if enabled) can wake the MCU.

5.5.2.1 Pin Configuration Options

The IRQ pin enable (IRQPE) control bit in the IRQSC register must be 1 for the IRQ pin to act as the interrupt request (IRQ) input. When the pin is configured 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).

When the IRQ pin is configured to detect rising edges, an optional pulldown resistor is available rather than a pullup resistor. 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.

NOTE

The voltage measured on the pulled up IRQ pin may be as low as $V_{DD} - 0.7$ V. The internal gates connected to this pin are pulled all the way to V_{DD}. All other pins with enabled pullup resistors will have an unloaded measurement of V_{DD}.

5.5.2.2 Edge and Level Sensitivity

The IRQMOD control bit re-configures the detection logic so it detects edge events and pin levels. In this edge 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. Vector Summary

Vector Priority	Vector Number	Address (High/Low)	Vector Name	Module	Source	Enable	Description
Lower	23 through 31	0xFFC0/0xFFC1 through 0xFFD0/0xFFD1	Unused Vector Space (available for user program)				
	22	0xFFD2/0xFFD3	Vlcd	LCD	LCDIF	LCDIEN	LCD interrupt
	21	0xFFD4/0xFFD5	Vrti	System control	RTIF	RTIE	Real-time interrupt
	20	0xFFD6/0xFFD7	Viic	IIC	IICIS	IICIE	IIC control
	19	0xFFD8/0xFFD9	Vacmp	ACMP	ACF	ACIE	ACMP
	18	0xFFDA/0xFFDB	Vadc	ADC	COCO	AIEN	AD conversion complete
	17	0xFFDC/0xFFDD	Vkeyboard2	KBI2	KBF	KBIE	Keyboard 2 pins
	16	0xFFDE/0xFFDF	Vkeyboard1	KBI1	KBF	KBIE	Keyboard 1 pins
	15	0xFFE0/0xFFE1	Vscitx	SCI	TDRE TC	TIE TCIE	SCI transmit
	14	0xFFE2/0xFFE3	Vscirx	SCI	IDLE RDRF	ILIE RIE	SCI receive
	13	0xFFE4/0xFFE5	Vscierr	SCI	OR NF FE PF	ORIE NFIE FEIE PFIE	SCI error
	12	0xFFE6/0xFFE7	Vspi2	SPI2	SPIF MODF SPTEF	SPIE SPIE SPTIE	SPI 2
	11	0xFFE8/0xFFE9	Vspi1	SPI1	SPIF MODF SPTEF	SPIE SPIE SPTIE	SPI 1
	10	0xFFEA/0xFFEB	Vtpm2ovf	TPM2	TOF	TOIE	TPM2 overflow
	9	0xFFEC/0xFFED	Vtpm2ch1	TPM2	CH1F	CH1IE	TPM2 channel 1
	8	0xFFEE/0xFFEF	Vtpm2ch0	TPM2	CH0F	CH0IE	TPM2 channel 0
	7	0xFFF0/0xFFF1	Vtpm1ovf	TPM1	TOF	TOIE	TPM1 overflow
	6	0xFFF2/0xFFF3	Vtpm1ch1	TPM1	CH1F	CH1IE	TPM1 channel 1
	5	0xFFF4/0xFFF5	Vtpm1ch0	TPM1	CH0F	CH0IE	TPM1 channel 0
	4	0xFFF6/0xFFF7	Vicg	ICG	ICGIF (LOLS/LOCS)	LOLRE/LOCRE	ICG
	3	0xFFF8/0xFFF9	Vlvd	System control	LVDF	LVDIE	Low-voltage detect
	2	0xFFFA/0xFFFB	Virq	IRQ	IRQF	IRQIE	IRQ pin
	1	0xFFFC/0xFFFD	Vswi	Core	SWI Instruction	—	Software interrupt
Higher	0	0xFFFE/0xFFFF	Vreset	System control	COP ICG LVD POR $\overline{\text{RESET}}$ pin Illegal opcode	COPE LVDRE — — —	Watchdog timer Low-voltage detect External pin Illegal opcode

5.6 Low-Voltage Detect (LVD) System

The MC9S08LC60 Series includes a system to protect against low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system comprises a power-on reset (POR) circuit and an 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 SPMSC2. The LVD is disabled upon entering any of the stop modes unless the LVDSE bit is set. If LVDSE and LVDE are both set, then the MCU cannot enter stop1 or 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 V_{POR} level, the POR circuit will cause a reset condition. As the supply voltage rises, the LVD circuit will hold the chip in reset until the supply has risen above the V_{LVDL} level. Both the POR bit and the LVD bit in SRS are set following a POR.

5.6.2 LVD Reset Operation

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

5.6.3 LVD Interrupt Operation

When a low voltage condition is detected and the LVD circuit is configured for interrupt operation (LVDE set, LVDIE set, and LVDRE clear), then LVDF will be set and an LVD interrupt will occur.

5.6.4 Low-Voltage Warning (LVW)

The LVD system has a low voltage warning flag to indicate to the user that the supply voltage is approaching, but is still above, the LVD voltage. The LVW does not have an interrupt associated with it. 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 Real-Time Interrupt (RTI)

The real-time interrupt function can be used to generate periodic interrupts. The RTI can accept two sources of clocks, the 1-kHz internal clock or an external clock if available. The RTICLKS bit in SRTISC is used to select the RTI clock source.

Either clock source can be used when the MCU is in run, wait or stop3 mode. When using the external oscillator in stop3, it must be enabled in stop (EREFSTEN = 1) and configured for low frequency operation (RANGE = 0). Only the internal 1-kHz clock source can be selected to wake the MCU from stop1 or stop2 modes.

The SRTISC register includes a read-only status flag, a write-only acknowledge bit, and a 3-bit control value (RTIS) used to select one of seven wakeup periods. The RTI has a local interrupt enable, RTIE, to allow masking of the real-time interrupt. The RTI can be disabled by writing each bit of RTIS to zeroes, and no interrupts will be generated. See Section 5.8.7, “System Real-Time Interrupt Status and Control Register (SRTISC)” for detailed information about this register.

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

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

Refer to the direct-page register summary in Chapter 4, “Memory” of this data sheet for the absolute address assignments for all registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

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

5.8.1 Interrupt Pin Request Status and Control Register (IRQSC)

These bits 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 Field Descriptions

Field	Description
6 IRQPDD	**Interrupt Request (IRQ) Pull Device Disable**— This read/write control bit is used to disable the internal pullup/pulldown device when the IRQ pin is enabled (IRQPE = 1) allowing for an external device to be used. 0 IRQ pull device enabled if IRQPE = 1. 1 IRQ pull device disabled if IRQPE = 1.
5 IRQEDG	**Interrupt Request (IRQ) Edge Select** — This read/write control bit is used to select the polarity of edges or levels on the IRQ pin that cause IRQF to be set. The IRQMOD control bit determines whether the IRQ pin is sensitive to both edges and levels or only edges. When the IRQ pin is enabled as the IRQ input and is configured to detect rising edges, the optional pullup resistor is re-configured as an optional pulldown resistor. 0 IRQ is falling edge or falling edge/low-level sensitive. 1 IRQ is rising edge or rising edge/high-level sensitive.

Table 5-3. IRQSC Field Descriptions (continued)

Field	Description
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. Also, when this bit is set, either an internal pull-up or an internal pull-down resistor is enabled depending on the state of the IRQMOD bit. 0 IRQ pin function is disabled. 1 IRQ pin function is enabled.
3 IRQF	**IRQ Flag —** This read-only status bit indicates when an interrupt request event has occurred. 0 No IRQ request. 1 IRQ event detected.
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 a hardware interrupt request. 0 Hardware interrupt requests from IRQF disabled (use polling). 1 Hardware 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 register includes six 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	0	ICG	LVD	0
W	Writing any value to SRS address clears COP watchdog timer.							
POR:	1	0	0	0	0	0	1	0
LVR:	u	0	0	0	0	0	1	0
Any other reset:	0	Note(1)	Note(1)	Note(1)	0	Note(1)	0	0

u = Unaffected by reset

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

Figure 5-3. System Reset Status (SRS)

Table 5-4. SRS 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 may 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.
2 ICG	**Internal Clock Generation Module Reset** — Reset was caused by an ICG module reset. 0 Reset not caused by ICG module. 1 Reset caused by ICG module.
1 LVD	**Low Voltage Detect** — If the LVD reset is enabled (LVDE = LVDRE = 1) and the supply drops below the LVD trip voltage, an LVD reset occurs. The LVD function is disabled when the MCU enters stop. To maintain LVD operation in stop, the LVDSE bit must be set. 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 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 0x0000.

	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 Field Descriptions

Field	Description
0 BDFR	**Background Debug Force Reset** — A serial 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.

5.8.4 System Options Register (SOPT1)

This register may be read at any time. Bits 3 and 2 are unimplemented and always read 0. This is a write-once register so only the first write after reset is honored. Any subsequent attempt to write to SOPT1 (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT1 should 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	BKGDPE	RSTPE
W								
Reset	1	1	0	1	0	0	1	1

= Unimplemented or Reserved

Figure 5-5. System Options Register (SOPT1)

Table 5-6. SOPT1 Field Descriptions

Field	Description
7 COPE	**COP Watchdog Enable** — This write-once bit defaults to 1 after reset. 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 defaults to 0 after reset, which disables 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.
1 BKGDPE	**Background Debug Mode Pin Enable** — The BKGDPE bit enables the PTG0/BKGD/MS pin to function as BKGD/MS. When the bit is clear, the pin will function as PTG0, which is an output-only general-purpose I/O. This pin always defaults to BKGD/MS function after any reset. 0 BKGD pin disabled. 1 BKGD pin enabled.
0 RSTPE	**Reset Pin Enable** — This write-once bit when set enables the PTB2/$\overline{\text{RESET}}$/ pin to function as RESET. When clear, the pin functions as one of its input only alternative functions. This pin defaults to its input-only port function following an MCU POR. Once configured for RESET pin, only POR can disable the RESET pin function. When RSTPE is set, an internal pullup device is enabled on RESET. 0 PTB2/$\overline{\text{RESET}}$/ pin functions as PTB2. 1 PTB2/$\overline{\text{RESET}}$/ pin functions as RESET.

5.8.5 System Options Register (SOPT2)

This register may be read at any time.

	7	6	5	4	3	2	1	0
R	COPCLKS[1]	0	0	0	0	0	0	ACIC
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-6. System Options Register (SOPT2)

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

Table 5-7. SOPT2 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.
0 ACIC	**Analog Comparator to Input Capture Enable**— This bit connects the output of ACMP to TPM1 input channel 0. 0 ACMP output not connected to TPM1 input channel 0 1 ACMP output connected to TPM1 input channel 0.

5.8.6 System Device Identification Register (SDIDH, SDIDL)

This read-only register is included so host development systems can identify the HCS08 derivative and revision number. 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	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
W								
Reset	0[1]	0(1)	0(1)	0(1)	0	0	0	0

= Unimplemented or Reserved

1 The revision number that is hard coded into these bits reflects the current silicon revision level.

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

Table 5-8. SDIDH Field Descriptions

Field	Description
7:4 REV[3:0]	**Revision Number** — The high-order 4 bits of SDIDH are hard-coded to reflect the current mask set revision number (0–F).
3:0 ID[11:8]	**Part Identification Number** — Each derivative in the HCS08 Family has a unique identification number. The MC9S08LC60 Series is hard-coded to the value 0x0. 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	0	0	1	1	0	0

= Unimplemented or Reserved

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

Table 5-9. SDIDL Field Descriptions

Field	Description
7:0 ID[7:0]	**Part Identification Number** — Each derivative in the HCS08 Family has a unique identification number. The MC9S08LC60 Series is hard coded to the value 0x0C. See also ID bits in Table 5-8.

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

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

	7	6	5	4	3	2	1	0
R	RTIF	0	RTICLKS	RTIE	0	RTIS2	RTIS1	RTIS0
W		RTIACK						
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

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

Table 5-10. SRTISC Field Descriptions

Field	Description
7 RTIF	**Real-Time Interrupt Flag** — This read-only status bit indicates the periodic wakeup timer has timed out. 0 Periodic wakeup timer not timed out. 1 Periodic wakeup timer timed out.
6 RTIACK	**Real-Time Interrupt Acknowledge** — This write-only bit is used to acknowledge real-time interrupt request (write 1 to clear RTIF). Writing 0 has no meaning or effect. Reads always return 0.
5 RTICLKS	**Real-Time Interrupt Clock Select** — This read/write bit selects the clock source for the real-time interrupt. 0 Real-time interrupt request clock source is internal 1-kHz oscillator. 1 Real-time interrupt request clock source is external clock.
4 RTIE	**Real-Time Interrupt Enable** — This read/write bit enables real-time interrupts. 0 Real-time interrupts disabled. 1 Real-time interrupts enabled.
2:0 RTIS[2:0]	**Real-Time Interrupt Period Selects** — These read/write bits select the wakeup period for the RTI. See Table 5-11.

Table 5-11. Real-Time Interrupt Period

RTIS2:RTIS1:RTIS0	Internal 1 kHz Clock Source [1] (t_{RTI} = 1 ms, Nominal)	External Clock Source [2] Period = t_{ext}
0:0:0	Disable RTI	Disable RTI
0:0:1	8 ms	t_{ext} x 256
0:1:0	32 ms	t_{ext} x 1024
0:1:1	64 ms	t_{ext} x 2048
1:0:0	128 ms	t_{ext} x 4096
1:0:1	256 ms	t_{ext} x 8192
1:1:0	512 ms	t_{ext} x 16384
1:1:1	1.024 s	t_{ext} x 32768

1 See Table A-13 t_{RTI} in Appendix A, "Electrical Characteristics," for the tolerance on these values.

2 t_{ext} is based on the external clock source, resonator, or crystal selected by the ICG configuration. See Table A-12 for details.

5.8.8 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, and to enable the bandgap voltage reference for use by the ADC module. To configure the low voltage detect trip voltage, see Table 5-14 for the LVDV bit description in SPMSC3.

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

= Unimplemented or Reserved

1 Bit 1 is a reserved bit that must always be written to 0.

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

Figure 5-10. System Power Management Status and Control 1 Register (SPMSC1)

Table 5-12. SPMSC1 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 read/write 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 read/write 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 read/write bit enables low-voltage detect logic and qualifies the operation of other bits in this register. 0 LVD logic disabled. 1 LVD logic enabled.
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 or the ACMP. 0 Bandgap buffer disabled. 1 Bandgap buffer enabled.

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

This register is used to configure the stop mode behavior of the MCU. For more information concerning partial power down mode, see Section 3.6, “Stop Modes.”

	7	6	5	4	3	2	1	0
R	0	0	0	PDF	PPDF	0	PDC[1]	PPDC[1]
W						PPDACK		
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

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

Figure 5-11. System Power Management Status and Control 2 Register (SPMSC2)

Table 5-13. SPMSC2 Field Descriptions

Field	Description
4 PDF	**Power Down Flag —** This read-only status bit indicates the MCU has recovered from stop1 mode. 0 MCU has not recovered from stop1 mode. 1 MCU recovered from stop1 mode.
3 PPDF	**Partial Power Down Flag** — The PPDF bit indicates that the MCU has exited the stop2 mode. 0 Not stop2 mode recovery. 1 Stop2 mode recovery.
2 PPDACK	**Partial Power Down Acknowledge** — Writing a 1 to PPDACK clears the PPDF bit.
1 PDC	**Power Down Control** — The write-once PDC bit controls entry into the power down (stop2 and stop1) modes. 0 Power down modes are disabled. 1 Power down modes are enabled.
0 PPDC	**Partial Power Down Control** — The write-once PPDC bit controls which power down mode, stop1 or stop2, is selected. 0 Stop1, full power down, mode enabled if PDC set. 1 Stop2, partial power down, mode enabled if PDC set.

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

This register is used to report the status of the low voltage warning function behavior of the MCU.

	7	6	5	4	3	2	1	0
R	LVWF	0	LVDV	LVWV	0	0	0	0
W		LVWACK						
Power-on reset (POR):	0[1]	0	0	0	0	0	0	0
LVD reset (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 5-12. System Power Management Status and Control 3 Register (SPMSC3)

Table 5-14. SPMSC3 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 LVWACK bit is the low-voltage warning acknowledge. Writing a 1 to LVWACK clears LVWF to 0 if a low voltage warning is not present.
5 LVDV	**Low-Voltage Detect Voltage Select** — The LVDV bit selects the LVD trip point voltage (V_{LVD}). 0 Low trip point selected ($V_{LVD} = V_{LVDL}$). 1 High trip point selected ($V_{LVD} = V_{LVDH}$).
4 LVWV	**Low-Voltage Warning Voltage Select** — The LVWV bit selects the LVW trip point voltage (V_{LVW}). 0 Low trip point selected ($V_{LVW} = V_{LVWL}$). 1 High trip point selected ($V_{LVW} = V_{LVWH}$).

Chapter 6
Parallel Input/Output

This section explains software controls related to parallel input/output (I/O). The MC9S08LC60 Series has three I/O ports which include a total of up to 24 general-purpose I/O pins (pin availability depends on device and package option, see Table 1-2 for details). See Chapter 2, “Pins and Connections,” for more information about the logic and hardware aspects of these pins.

Many of these pins are shared with on-chip peripherals such as timer systems, SPI, SCI, IIC, external interrupts, or keyboard interrupts as shown in Table 2-1. The peripheral modules have priority over the I/Os so that when a peripheral is enabled, the I/O functions associated with the shared pins are disabled. After reset, the shared peripheral functions are disabled so that the pins are controlled by the I/O. All of the I/Os are configured as inputs (PTxDDn = 0) with pullup devices disabled (PTxPEn = 0), except for output-only pin PTC6 which defaults to BKGD/MS pin and PTB2 which defaults to the $\overline{\text{RESET}}$ function.

When these other modules are not controlling the port pins, they revert to general-purpose I/O control. As a general-purpose I/O control, a port data bit provides access to input (read) and output (write) data, a data direction bit controls the direction of the pin, and a pullup enable bit enables an internal pullup device (provided the pin is configured as an input), and a slew rate control bit controls the rise and fall times of the pins.

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 should either enable on-chip pullup devices or change the direction of unconnected pins to outputs so the pins do not float.

Reading and writing of parallel I/Os is 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.

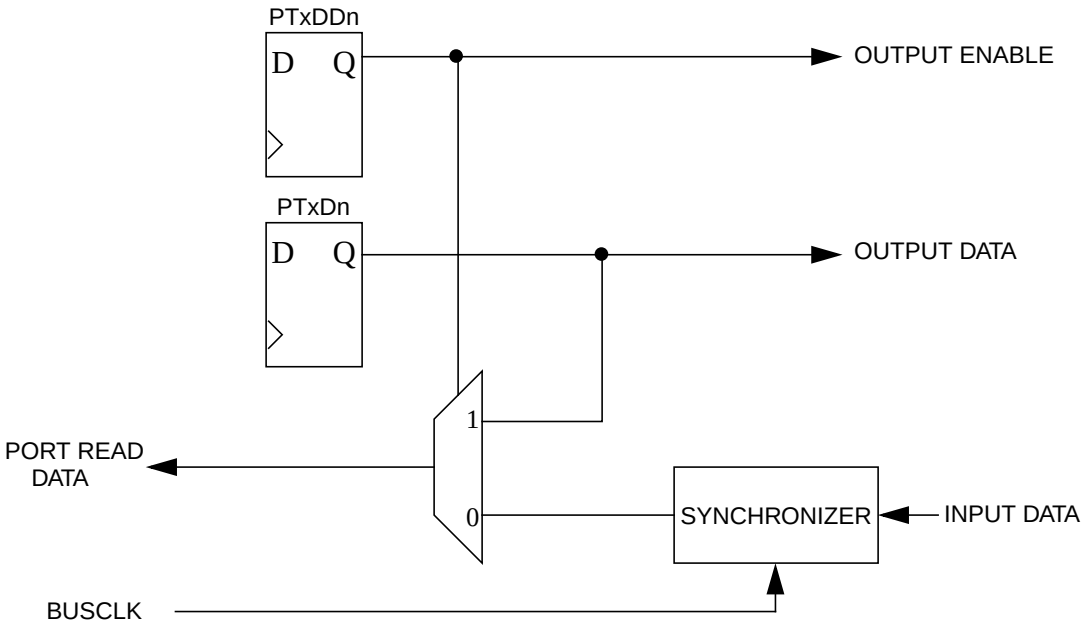


Figure 6-1. Parallel I/O Block Diagram

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.

It is a good programming practice to write to the port data register before changing the direction of a port pin to become 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.

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. See Section 6.2.1, “Port A Registers for more information.

6.1 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:

- In stop1 mode, all internal registers including parallel I/O control and data registers are powered off. Each of the pins assumes its default reset state (output buffer and internal pullup disabled). Upon exit from stop1, all pins must be reconfigured the same as if the MCU had been reset.
- Stop2 mode is a partial power-down mode, whereby latches maintain the pin state as before the STOP instruction was executed. CPU register status and the state of I/O registers must 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 must 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 data previously stored in RAM, before the STOP instruction was executed, peripherals previously enabled will require being initialized and restored to their pre-stop condition. The user must then write a 1 to the PPDACK bit in the SPMSC2 register. Access of pins is now permitted again in the user's application program.
- In stop3 mode, all pin states are maintained because internal logic stays powered up. Upon recovery, all pin functions are the same as before entering stop3.

6.2 Parallel I/O Registers

6.2.1 Port A Registers

This section provides information about all registers and control bits associated with the parallel I/O ports. The parallel I/O registers are located in page zero of the memory map.

Refer to tables in Chapter 3, "Modes of Operation" for the absolute address assignments for all parallel I/O registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.

6.2.1.1 Port A Data Registers (PTAD)

Port A parallel I/O function is controlled by the data and data direction registers in this section.

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

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

Table 6-1. PTAD 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 disabled.

6.2.1.2 Port A Data Direction Registers (PTADD)

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

Figure 6-5. Data Direction for Port A (PTADD)

Table 6-2. PTADD 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.

6.2.2 Port A Control Registers

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 associated pins and may be used in conjunction with the peripheral functions on these pins for most modules.

The pins associated with port A are controlled by the registers in this section. These registers control the pin pullup, slew rate and drive strength of the port A pins independent of the parallel I/O registers.

6.2.2.1 Internal Pullup Enable (PTAPE)

An internal pullup device can be enabled for each port pin by setting the corresponding bit in the pullup enable register (PTAPEn). The pullup device is disabled if the pin is configured as an output by the parallel I/O control logic or any shared peripheral function regardless of the state of the corresponding pullup enable register bit. The pullup device is also disabled if the pin is controlled by an analog function.

	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 6-7. Pullup Enable for Port A (PTAPE)

Table 6-3. PTAPE Field Descriptions

Field	Description
7:0 PTAPE[7:0]	**Pullup Enable for Port A Bits** — For port A pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled provided the corresponding PTADDn is 0. For port A pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled. When any of bits 7 through 0 of port A are enabled as KBI inputs and are configured to detect rising edges/high levels, the pullup enable bits enable pulldown rather than pullup devices. 0 Internal pullup device disabled. 1 Internal pullup device enabled.

6.2.2.2 Output Slew Rate Control Enable (PTASE)

Slew rate control can be enabled for each port pin by setting the corresponding bit in the slew rate control register (PTASEn). 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 which are configured as inputs.

	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 6-8. Slew Rate Control Enable for Port A (PTASE)

Table 6-4. PTASE Field Descriptions

Field	Description
7:0 PTASE[7:0]	**Slew Rate Control Enable for Port A Bits** — For port A pins that are outputs, these read/write control bits determine whether the slew rate controlled outputs are enabled. For port A pins that are configured as inputs, these bits are ignored. 0 Slew rate control disabled. 1 Slew rate control enabled.

6.2.2.3 Output Drive Strength Select (PTADS)

An output pin can be selected to have high output drive strength by setting the corresponding bit in the drive strength select register (PTADSn). 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 chip 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.

	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 6-9. Drive Strength Selection for Port A (PTADS)

Table 6-5. PTADS 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.

6.2.3 Port B Registers

This section provides information about all registers and control bits associated with the parallel I/O ports. The parallel I/O registers are located in page zero of the memory map.

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

6.2.3.1 Port B Data Registers (PTBD)

Port B parallel I/O function is controlled by the data and data direction registers in this section.

	7	6	5	4	3	2	1	0
R W	PTBD7	PTBD6	PTBD5	PTBD4	PTBD3	PTBD2[1]	PTBD1	PTBD0
Reset	0	0	0	0	0	0	0	0

Figure 6-10. Port B Data Register (PTBD)

1 Reads of PTBD2 always return the contents of PTBD2, regardless of the value stored in the bit PTBDD2

Table 6-6. PTBD 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 disabled.

6.2.3.2 Port B Data Direction Registers (PTBDD)

	7	6	5	4	3	2	1	0
R W	PTBDD7	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2[1]	PTBDD1	PTBDD0
Reset	0	0	0	0	0	0	0	0

Figure 6-13. Data Direction for Port B (PTBDD)

1 PTBDD2 has no effect on the output-only PTB2 pin.

Table 6-7. PTBDD 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.

6.2.4 Port B Control Registers

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 associated pins and may be used in conjunction with the peripheral functions on these pins for most modules.

The pins associated with Port B are controlled by the registers in this section. These registers control the pin pullup, slew rate and drive strength of the Port B pins independent of the parallel I/O registers.

6.2.4.1 Internal Pullup Enable (PTBPE)

An internal pullup device can be enabled for each port pin by setting the corresponding bit in the pullup enable register (PTBPEn). The pullup device is disabled if the pin is configured as an output by the parallel I/O control logic or any shared peripheral function regardless of the state of the corresponding pullup enable register bit. The pullup device is also disabled if the pin is controlled by an analog function.

	7	6	5	4	3	2	1	0
R W	PTBPE7	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
Reset	0	0	0	0	0	0	0	0

Figure 6-15. Pullup Enable for Port B (PTBPE)

Table 6-8. PTBPE Field Descriptions

Field	Description
7:0 PTBPE[7:0]	**Pullup Enable for Port B Bits** — For port B pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled provided the corresponding PTBDDn is 0. For port B pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled. When bit 0, 1, 3, 6, or 7 of port B is enabled as KBI inputs and are configured to detect rising edges/high levels, the pullup enable bits enable pulldown rather than pullup devices. 0 Internal pullup device disabled. 1 Internal pullup device enabled.

6.2.4.2 Output Slew Rate Control Enable (PTBSE)

Slew rate control can be enabled for each port pin by setting the corresponding bit in the slew rate control register (PTBSEn). 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 which are configured as inputs.

	7	6	5	4	3	2	1	0
R W	PTBSE7	PTBSE6	PTBSE5	PTBSE4	PTBSE3	PTBSE2	PTBSE1	PTBSE0
Reset	1	1	1	1	1	1	1	1

Figure 6-16. Slew Rate Control Enable for Port B (PTBSE)

Table 6-9. PTBSE Field Descriptions

Field	Description
7:0 PTBSE[7:0]	**Slew Rate Control Enable for Port B Bits** — For port B pins that are outputs, these read/write control bits determine whether the slew rate controlled outputs are enabled. For port B pins that are configured as inputs, these bits are ignored. 0 Slew rate control disabled. 1 Slew rate control enabled.

6.2.4.3 Output Drive Strength Select (PTBDS)

An output pin can be selected to have high output drive strength by setting the corresponding bit in the drive strength select register (PTBDSn). 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 chip 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.

	7	6	5	4	3	2	1	0
R W	PTBDS7	PTBDS6	PTBDS5	PTBDS4	PTBDS3	PTBDS2	PTBDS1	PTBDS0
Reset	0	0	0	0	0	0	0	0

Figure 6-17. Drive Strength Selection for Port B (PTBDS)

Table 6-10. PTBDS 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.

6.2.5 Port C Registers

This section provides information about all registers and control bits associated with the parallel I/O ports. The parallel I/O registers are located in page zero of the memory map.

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

6.2.5.1 Port C Data Registers (PTCD)

Port C parallel I/O function is controlled by the data and data direction registers in this section.

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

Figure 6-18. Port C Data Register (PTCD)

1 Reads of PTCD7 always return the pin value of PTC7, regardless of the value stored in the bit PTCDD7

2 Reads of PTCD6 always return the contents of PTCD6, regardless of the value stored in the bit PTCDD6.

Table 6-11. PTCD 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 the corresponding MCU pin. Reset forces PTCD 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 disabled.

6.2.5.2 Port C Data Direction Registers (PTCDD)

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

Figure 6-21. Data Direction for Port C (PTCDD)

1 PTCDD7 has no effect on the input-only PTC7 pin.

2 PTCDD6 has no effect on the output-only PTC6 pin

Table 6-12. PTCDD 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.

6.2.6 Port C Control Registers

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 associated pins and may be used in conjunction with the peripheral functions on these pins for most modules.

The pins associated with Port C are controlled by the registers in this section. These registers control the pin pullup, slew rate and drive strength of the Port C pins independent of the parallel I/O registers.

6.2.6.1 Internal Pullup Enable (PTCPE)

An internal pullup device can be enabled for each port pin by setting the corresponding bit in the pullup enable register (PTCPEn). The pullup device is disabled if the pin is configured as an output by the parallel I/O control logic or any shared peripheral function regardless of the state of the corresponding pullup enable register bit. The pullup device is also disabled if the pin is controlled by an analog function.

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

Figure 6-23. Pullup Enable for Port C (PTCPE)

1 PTCPE7 has no effect on the output-only PTC7 pin.

2 PTCPE6 has no effect on the output-only PTC6 pin.

Table 6-13. PTCPE Field Descriptions

Field	Description
7:0 PTCPE[7:0]	**Pullup Enable for Port C Bits** — For port C pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled provided the corresponding PTCDDn is 0. For port C pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled. When bits 4, 5, or 7 of port C are enabled as KBI inputs and are configured to detect rising edges/high levels, the pullup enable bits enable pulldown rather than pullup devices. 0 Internal pullup device disabled. 1 Internal pullup device enabled.

6.2.6.2 Output Slew Rate Control Enable (PTCSE)

Slew rate control can be enabled for each port pin by setting the corresponding bit in the slew rate control register (PTCSEn). 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 which are configured as inputs.

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

Figure 6-24. Slew Rate Control Enable for Port C (PTCSE)

1 PTCSE7 has no effect on the input-only PTC7 pin.

2 Reads of PTCD6 always return the contents of PTCD6, regardless of the value stored in the bit PTCDD6.

Table 6-14. PTCSE Field Descriptions

Field	Description
7:0 PTCSE[7:0]	**Slew Rate Control Enable for Port C Bits** — For port C pins that are outputs, these read/write control bits determine whether the slew rate controlled outputs are enabled. For port C pins that are configured as inputs, these bits are ignored. 0 Slew rate control disabled. 1 Slew rate control enabled.

6.2.6.3 Output Drive Strength Select (PTCDS)

An output pin can be selected to have high output drive strength by setting the corresponding bit in the drive strength select register (PTCDSn). 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 chip 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.

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

Figure 6-25. Drive Strength Selection for Port C (PTCDS)

1 PTCDS7 has no effect on the input-only PTC7 pin.

2 PTCDD6 has no effect on the output-only PTC6 pin.

Table 6-15. PTCDS 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.

Chapter 7
Keyboard Interrupt (S08KBIV2)

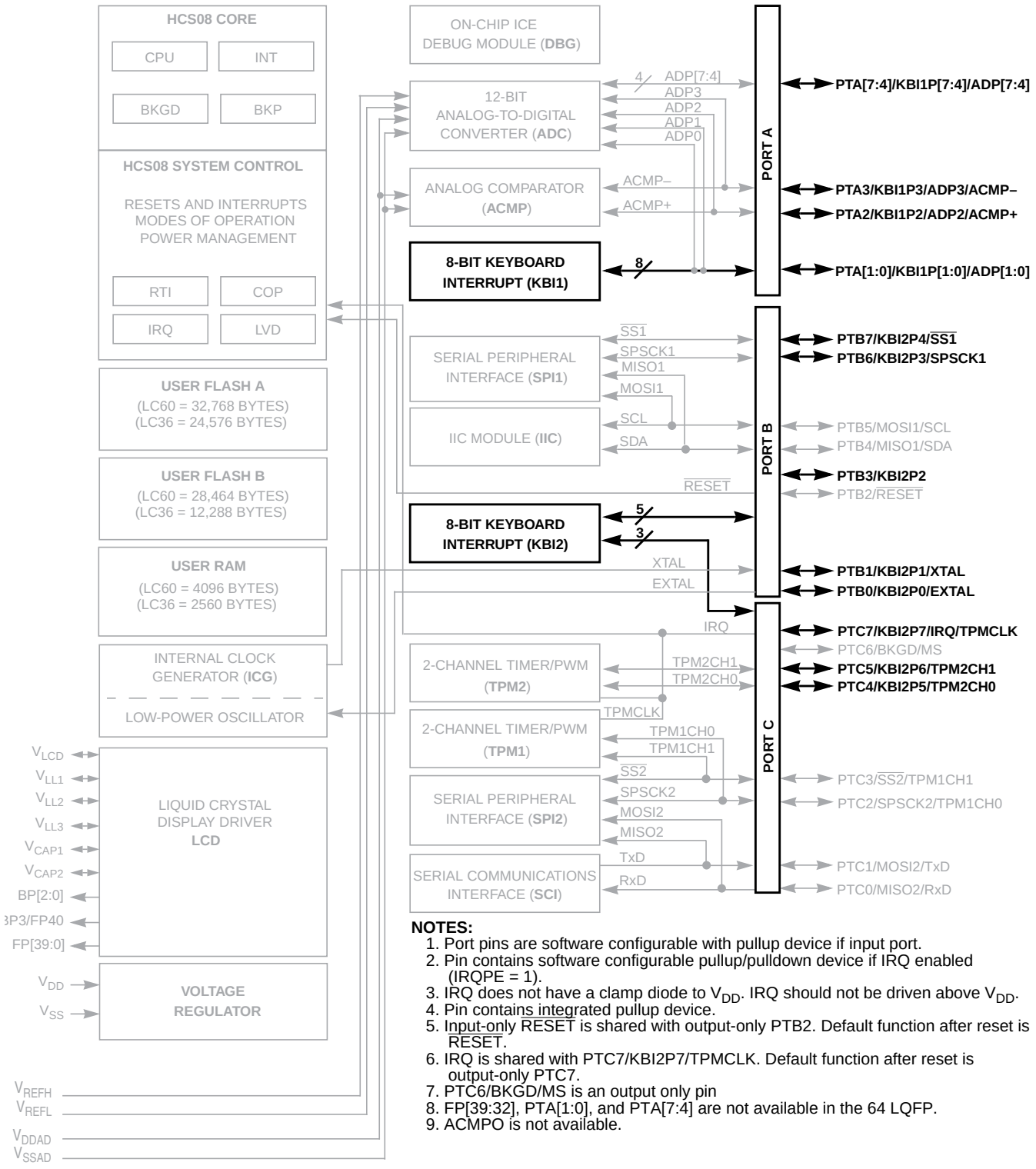
7.1 Introduction

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 up the MCU from stop or wait low-power modes.

The KBI module allows up to eight pins to act as additional interrupt sources. These pins can be configured for either rising-edge sensing or falling-edge sensing. The sensing mode for all eight pins can also be modified to detect edges and levels instead of only edges.

MC9S08LC60 Series MCUs have two KBIs. When they are described individually, they are called KBI1 and/or KBI2. When referring to the module in general or both KBIs collectively, they are called KBIx.

Figure 7-1 Shows the MC9S08LC60 Series block guide with the KBIs highlighted.



NOTES:

1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 7-1. MC9S08LC60 Series Block Diagram Highlighting KBI Block and Pins

7.1.1 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.

7.1.2 Modes of Operation

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

7.1.2.1 KBI in Wait Mode

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

7.1.2.2 KBI in Stop Modes

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

During either stop1 or stop2 mode, the KBI is disabled. In some systems, the pins associated with the KBI may be sources of wakeup from stop1 or stop2, see the stop modes section in the Modes of Operation chapter. Upon wake-up from stop1 or stop2 mode, the KBI module will be in the reset state.

7.1.2.3 KBI in Active Background Mode

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

7.1.3 Block Diagram

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

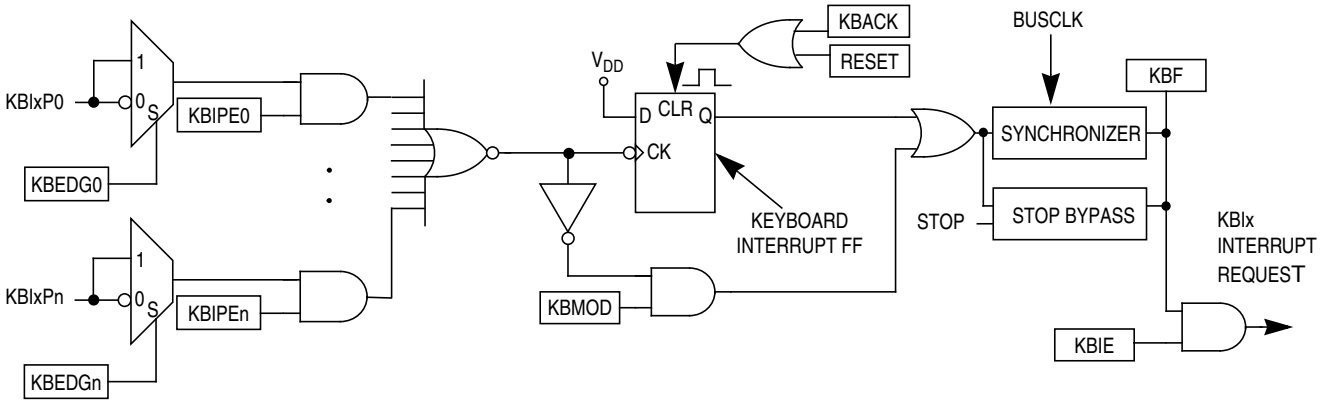


Figure 7-2. Keyboard Interrupt (KBI) Block Diagram

7.2 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.

The signal properties of KBI are shown in Table 7-1.

Table 7-1. Signal Properties

Signal	Function	I/O
KBIxPn	Keyboard interrupt pins	I

7.3 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.

Some MCUs may have more than one KBI, so register names include placeholder characters to identify which KBI is being referenced.

7.3.1 KBIx Status and Control Register (KBIxSC)

KBIxSC contains the status flag and control bits, which are used to configure the KBI.

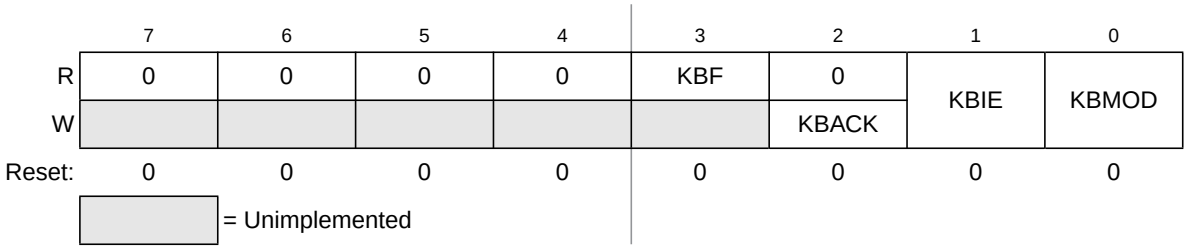


Figure 7-3. KBIx Status and Control Register

Table 7-2. KBIxSC Register Field Descriptions

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

7.3.2 KBIx Pin Enable Register (KBIxPE)

KBIxPE contains the pin enable control bits.

	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. KBIx Pin Enable Register

Table 7-3. KBIxPE Register Field Descriptions

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

7.3.3 KBIx Edge Select Register (KBIxES)

KBIxES contains the edge select control bits.

	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. KBIx Edge Select Register

Table 7-4. KBIxES Register Field Descriptions

Field	Description
7:0 KBEDGn	**Keyboard Edge Selects** — Each of the KBEDGn bits selects the falling edge/low level or rising edge/high level function of the corresponding pin). 0 Falling edge/low level. 1 Rising edge/high level.

7.4 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 (KBIxPE) independently enables or disables each KBI pin. Each KBI pin can be configured as edge sensitive or edge and level sensitive based on the KBMOD bit in the keyboard interrupt status and control register (KBIxSC). Edge sensitive 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 (KBIxES).

Synchronous logic is used to detect edges. Prior to detecting an edge, enabled keyboard inputs must be at the deasserted logic level. A falling edge is detected when an enabled keyboard 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.

7.4.1 Edge Only Sensitivity

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

7.4.2 Edge and Level Sensitivity

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

7.4.3 KBI Pullup/Pulldown Resistors

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

7.4.4 KBI Initialization

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

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

Chapter 8
Central Processor Unit (S08CPUV2)

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

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.

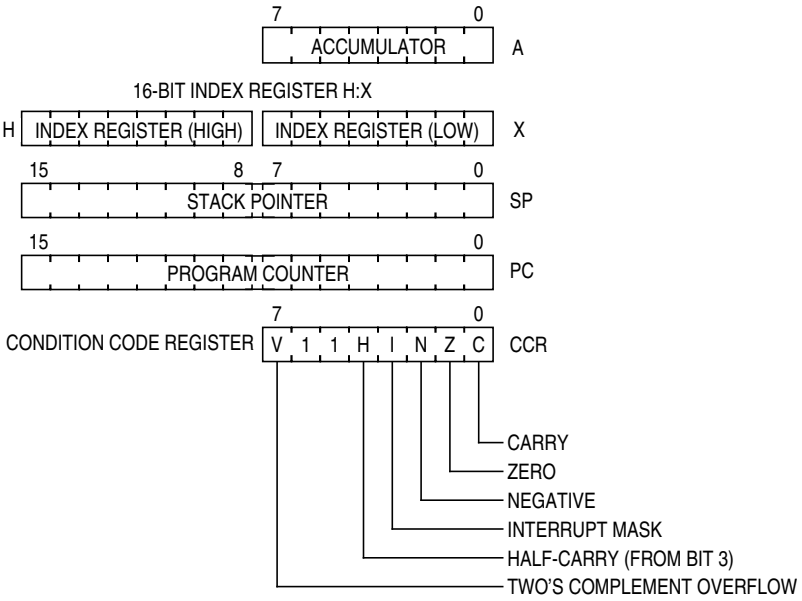


Figure 8-1. CPU Registers

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.

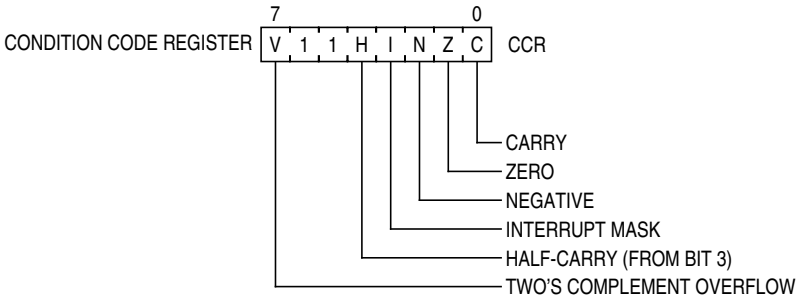


Figure 8-2. Condition Code Register

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, all memory, status and control registers, and input/output (I/O) ports share a single 64-Kbyte linear address space so a 16-bit binary address can uniquely identify any memory location. 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.

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.

8.5 HCS08 Instruction Set Summary

Instruction Set Summary Nomenclature

The nomenclature listed here is used in the instruction descriptions in Table 8-2.

Operators

( )	=	Contents of register or memory location shown inside parentheses
←	=	Is loaded with (read: “gets”)
&	=	Boolean AND
|	=	Boolean OR
⊕	=	Boolean exclusive-OR
×	=	Multiply
÷	=	Divide
:	=	Concatenate
+	=	Add
–	=	Negate (two’s complement)

CPU registers

A	=	Accumulator
CCR	=	Condition code register
H	=	Index register, higher order (most significant) 8 bits
X	=	Index register, lower order (least significant) 8 bits
PC	=	Program counter
PCH	=	Program counter, higher order (most significant) 8 bits
PCL	=	Program counter, lower order (least significant) 8 bits
SP	=	Stack pointer

Memory and addressing

M	=	A memory location or absolute data, depending on addressing mode
M:M + 0x0001	=	A 16-bit value in two consecutive memory locations. The higher-order (most significant) 8 bits are located at the address of M, and the lower-order (least significant) 8 bits are located at the next higher sequential address.

Condition code register (CCR) bits

V	=	Two’s complement overflow indicator, bit 7
H	=	Half carry, bit 4
I	=	Interrupt mask, bit 3
N	=	Negative indicator, bit 2
Z	=	Zero indicator, bit 1
C	=	Carry/borrow, bit 0 (carry out of bit 7)

CCR activity notation

–	=	Bit not affected

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

Machine coding notation

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

Source form

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

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

Address modes

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

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

Table 8-2. HCS08 Instruction Set Summary (Sheet 1 of 7)

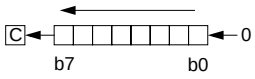
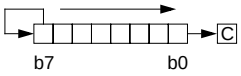
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
ADC *#opr8i* ADC *opr8a* ADC *opr16a* ADC *oprx16*,X ADC *oprx8*,X ADC ,X ADC *oprx16*,SP ADC *oprx8*,SP	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP2 SP1	A9 B9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

Table 8-2. HCS08 Instruction Set Summary (Sheet 2 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
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 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	Branch if (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ *rel*	Branch if Equal	Branch if (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE *rel*	Branch if Greater Than or Equal To (Signed Operands)	Branch if $(N \oplus V) = 0$	–	–	–	–	–	–	REL	90	rr	3
BGND	Enter Active Background if ENBDM = 1	Waits For and Processes BDM Commands Until GO, TRACE1, or TAGGO	–	–	–	–	–	–	INH	82		5+
BGT *rel*	Branch if Greater Than (Signed Operands)	Branch if $(Z) \mid (N \oplus V) = 0$	–	–	–	–	–	–	REL	92	rr	3
BHCC *rel*	Branch if Half Carry Bit Clear	Branch if (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS *rel*	Branch if Half Carry Bit Set	Branch if (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI *rel*	Branch if Higher	Branch if (C) | (Z) = 0	–	–	–	–	–	–	REL	22	rr	3
BHS *rel*	Branch if Higher or Same (Same as BCC)	Branch if (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH *rel*	Branch if IRQ Pin High	Branch if IRQ pin = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL *rel*	Branch if IRQ Pin Low	Branch if IRQ pin = 0	–	–	–	–	–	–	REL	2E	rr	3
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)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A5 B5 C5 D5 E5 F5 9ED5 9EE5	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
BLE *rel*	Branch if Less Than or Equal To (Signed Operands)	Branch if $(Z) \mid (N \oplus V) = 1$	–	–	–	–	–	–	REL	93	rr	3
BLO *rel*	Branch if Lower (Same as BCS)	Branch if (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	Branch if (C) | (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT *rel*	Branch if Less Than (Signed Operands)	Branch if $(N \oplus V) = 1$	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	Branch if (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI *rel*	Branch if Minus	Branch if (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	Branch if (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	Branch if (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL *rel*	Branch if Plus	Branch if (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA *rel*	Branch Always	No Test	–	–	–	–	–	–	REL	20	rr	3

Table 8-2. HCS08 Instruction Set Summary (Sheet 3 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
BRCLR *n*,*opr8a*,*rel*	Branch if Bit *n* in Memory Clear	Branch if (Mn) = 0	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRN *rel*	Branch Never	Uses 3 Bus Cycles	–	–	–	–	–	–	REL	21	rr	3
BRSET *n*,*opr8a*,*rel*	Branch if Bit *n* in Memory Set	Branch if (Mn) = 1	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BSET *n*,*opr8a*	Set Bit *n* in Memory	Mn ← 1	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BSR *rel*	Branch to Subroutine	PC ← (PC) + 0x0002 push (PCL); SP ← (SP) – 0x0001 push (PCH); SP ← (SP) – 0x0001 PC ← (PC) + *rel*	–	–	–	–	–	–	REL	AD	rr	5
CBEQ *opr8a*,*rel* CBEQA *#opr8i*,*rel* CBEQX *#opr8i*,*rel* CBEQ *oprx8*,X+,*rel* CBEQ ,X+,*rel* CBEQ *oprx8*,SP,*rel*	Compare and Branch if Equal	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 41 51 61 71 9E61	dd rr ii rr ii rr ff rr rr ff rr	5 4 4 5 5 6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask Bit	I ← 0	–	–	0	–	–	–	INH	9A		1
CLR *opr8a* CLRA CLRX CLRH CLR *oprx8*,X CLR ,X CLR *oprx8*,SP	Clear	M ← 0x00 A ← 0x00 X ← 0x00 H ← 0x00 M ← 0x00 M ← 0x00 M ← 0x00	0	–	–	0	1	–	DIR INH INH INH IX1 IX SP1	3F 4F 5F 8C 6F 7F 9E6F	dd ff ff	5 1 1 1 5 4 6
CMP *#opr8i* CMP *opr8a* CMP *opr16a* 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 B1 C1 D1 E1 F1 9ED1 9EE1	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
COM *opr8a* COMA COMX COM *oprx8*,X COM ,X COM *oprx8*,SP	Complement (One's Complement)	M ← (M̄)= 0xFF – (M) A ← (Ā) = 0xFF – (A) X ← (X̄) = 0xFF – (X) M ← (M̄) = 0xFF – (M) M ← (M̄) = 0xFF – (M) M ← (M) = 0xFF – (M)	0	–	–	↕	↕	1	DIR INH INH IX1 IX SP1	33 43 53 63 73 9E63	dd ff ff	5 1 1 5 4 6
CPHX *opr16a* CPHX *#opr16i* CPHX *opr8a* CPHX *oprx8*,SP	Compare Index Register (H:X) with Memory	(H:X) – (M:M + 0x0001) (CCR Updated But Operands Not Changed)	↕	–	–	↕	↕	↕	EXT IMM DIR SP1	3E 65 75 9EF3	hh ll jj kk dd ff	6 3 5 6

Table 8-2. HCS08 Instruction Set Summary (Sheet 4 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
CPX #*opr8i* CPX *opr8a* CPX *opr16a* 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 B3 C3 D3 E3 F3 9ED3 9EE3	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
DAA	Decimal Adjust Accumulator After ADD or ADC of BCD Values	$(A)_{10}$	U	–	–	↕	↕	↕	INH	72		1
DBNZ *opr8a*,*rel* DBNZA *rel* DBNZX *rel* DBNZ *oprx8*,X,*rel* DBNZ ,X,*rel* DBNZ *oprx8*,SP,*rel*	Decrement and Branch if Not Zero	Decrement A, X, or M Branch if (result) ≠ 0 DBNZX Affects X Not H	–	–	–	–	–	–	DIR INH INH IX1 IX SP1	3B 4B 5B 6B 7B 9E6B	dd rr rr rr ff rr rr ff rr	7 4 4 7 6 8
DEC *opr8a* DECA DECX DEC *oprx8*,X DEC ,X DEC *oprx8*,SP	Decrement	M ← (M) – 0x01 A ← (A) – 0x01 X ← (X) – 0x01 M ← (M) – 0x01 M ← (M) – 0x01 M ← (M) – 0x01	↕	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3A 4A 5A 6A 7A 9E6A	dd ff ff	5 1 1 5 4 6
DIV	Divide	A ← (H:A)÷(X) H ← Remainder	–	–	–	–	↕	↕	INH	52		6
EOR #*opr8i* EOR *opr8a* EOR *opr16a* EOR *oprx16*,X EOR *oprx8*,X EOR ,X EOR *oprx16*,SP EOR *oprx8*,SP	Exclusive OR Memory with Accumulator	A ← (A ⊕ M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A8 B8 C8 D8 E8 F8 9ED8 9EE8	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
INC *opr8a* INCA INCX INC *oprx8*,X INC ,X INC *oprx8*,SP	Increment	M ← (M) + 0x01 A ← (A) + 0x01 X ← (X) + 0x01 M ← (M) + 0x01 M ← (M) + 0x01 M ← (M) + 0x01	↕	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3C 4C 5C 6C 7C 9E6C	dd ff ff	5 1 1 5 4 6
JMP *opr8a* JMP *opr16a* JMP *oprx16*,X JMP *oprx8*,X JMP ,X	Jump	PC ← Jump Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff 	3 4 4 3 3
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) – 0x0001 Push (PCH); SP ← (SP) – 0x0001 PC ← Unconditional Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff 	5 6 6 5 5
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)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A6 B6 C6 D6 E6 F6 9ED6 9EE6	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
LDHX #*opr16i* LDHX *opr8a* LDHX *opr16a* LDHX ,X LDHX *oprx16*,X LDHX *oprx8*,X LDHX *oprx8*,SP	Load Index Register (H:X) from Memory	H:X ← (M:M + 0x0001)	0	–	–	↕	↕	–	IMM DIR EXT IX IX2 IX1 SP1	45 55 32 9EAE 9EBE 9ECE 9EFE	jj kk dd hh ll ee ff ff ff	3 4 5 5 6 5 5

Table 8-2. HCS08 Instruction Set Summary (Sheet 5 of 7)

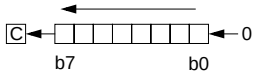
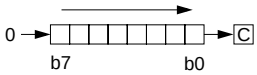
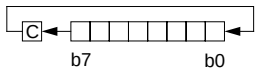
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
LDX *#opr8i* LDX *opr8a* LDX *opr16a* LDX *oprx16*,X LDX *oprx8*,X LDX ,X LDX *oprx16*,SP LDX *oprx8*,SP	Load X (Index Register Low) from Memory	X ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	AE BE CE DE EE FE 9EDE 9EEE	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
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 48 58 68 78 9E68	dd ff ff	5 1 1 5 4 6
LSR *opr8a* LSRA LSR*X* LSR *oprx8*,X LSR ,X LSR *oprx8*,SP	Logical Shift Right		↕	–	–	0	↕	↕	DIR INH INH IX1 IX SP1	34 44 54 64 74 9E64	dd ff ff	5 1 1 5 4 6
MOV *oprx8*,*oprx8* MOV *oprx8*,X+ MOV *#opr8i*,*oprx8* MOV ,X+,*oprx8*	Move	$(M)_{destination} \leftarrow (M)_{source}$ H:X ← (H:X) + 0x0001 in IX+/DIR and DIR/IX+ Modes	0	–	–	↕	↕	–	DIR/DIR DIR/IX+ IMM/DIR IX+/DIR	4E 5E 6E 7E	dd dd dd ii dd dd	5 5 4 5
MUL	Unsigned multiply	X:A ← (X) × (A)	–	0	–	–	–	0	INH	42		5
NEG *opr8a* NEGA NEGX NEG *oprx8*,X NEG ,X NEG *oprx8*,SP	Negate (Two's Complement)	M ← – (M) = 0x00 – (M) A ← – (A) = 0x00 – (A) X ← – (X) = 0x00 – (X) M ← – (M) = 0x00 – (M) M ← – (M) = 0x00 – (M) M ← – (M) = 0x00 – (M)		–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	30 40 50 60 70 9E60	dd ff ff	5 1 1 5 4 6
NOP	No Operation	Uses 1 Bus Cycle	–	–	–	–	–	–	INH	9D		1
NSA	Nibble Swap Accumulator	A ← (A[3:0]:A[7:4])	–	–	–	–	–	–	INH	62		1
ORA *#opr8i* ORA *opr8a* ORA *opr16a* ORA *oprx16*,X ORA *oprx8*,X ORA ,X ORA *oprx16*,SP ORA *oprx8*,SP	Inclusive OR Accumulator and Memory	A ← (A) | (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	AA BA CA DA EA FA 9EDA 9EEA	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
PSHA	Push Accumulator onto Stack	Push (A); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	87		2
PSHH	Push H (Index Register High) onto Stack	Push (H); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	8B		2
PSHX	Push X (Index Register Low) onto Stack	Push (X); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	89		2
PULA	Pull Accumulator from Stack	SP ← (SP + 0x0001); Pull (A)	–	–	–	–	–	–	INH	86		3
PULH	Pull H (Index Register High) from Stack	SP ← (SP + 0x0001); Pull (H)	–	–	–	–	–	–	INH	8A		3
PULX	Pull X (Index Register Low) from Stack	SP ← (SP + 0x0001); Pull (X)	–	–	–	–	–	–	INH	88		3
ROL *opr8a* ROLA ROLX ROL *oprx8*,X ROL ,X ROL *oprx8*,SP	Rotate Left through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	39 49 59 69 79 9E69	dd ff ff	5 1 1 5 4 6

Table 8-2. HCS08 Instruction Set Summary (Sheet 6 of 7)

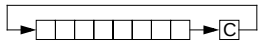
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
ROR *oprx8*,X ROR ,X ROR *oprx8*,SP	Rotate Right through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	5 1 1 5 4 6
RSP	Reset Stack Pointer	SP ← 0xFF (High Byte Not Affected)	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 0x0001; Pull (CCR) SP ← (SP) + 0x0001; Pull (A) SP ← (SP) + 0x0001; Pull (X) SP ← (SP) + 0x0001; Pull (PCH) SP ← (SP) + 0x0001; Pull (PCL)	↕	↕	↕	↕	↕	↕	INH	80		9
RTS	Return from Subroutine	SP ← SP + 0x0001; Pull (PCH) SP ← SP + 0x0001; Pull (PCL)	–	–	–	–	–	–	INH	81		6
SBC *#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 B2 C2 D2 E2 F2 9ED2 9EE2	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask Bit	I ← 1	–	–	1	–	–	–	INH	9B		1
STA *opr8a* STA *opr16a* STA *oprx16*,X STA *oprx8*,X STA ,X STA *oprx16*,SP STA *oprx8*,SP	Store Accumulator in Memory	M ← (A)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP2 SP1	B7 C7 D7 E7 F7 9ED7 9EE7	dd hh ll ee ff ff ee ff ff	3 4 4 3 2 5 4
STHX *opr8a* STHX *opr16a* STHX *oprx8*,SP	Store H:X (Index Reg.)	(M:M + 0x0001) ← (H:X)	0	–	–	↕	↕	–	DIR EXT SP1	35 96 9EFF	dd hh ll ff	4 5 5
STOP	Enable Interrupts: Stop Processing Refer to MCU Documentation	I bit ← 0; Stop Processing	–	–	0	–	–	–	INH	8E		2+
STX *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)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP2 SP1	BF CF DF EF FF 9EDF 9EEF	dd hh ll ee ff ff ee ff ff	3 4 4 3 2 5 4
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 B0 C0 D0 E0 F0 9ED0 9EE0	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
SWI	Software Interrupt	PC ← (PC) + 0x0001 Push (PCL); SP ← (SP) – 0x0001 Push (PCH); SP ← (SP) – 0x0001 Push (X); SP ← (SP) – 0x0001 Push (A); SP ← (SP) – 0x0001 Push (CCR); SP ← (SP) – 0x0001 I ← 1; PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		11

Table 8-2. HCS08 Instruction Set Summary (Sheet 7 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
TAP	Transfer Accumulator to CCR	CCR ← (A)	↕	↕	↕	↕	↕	↕	INH	84		1
TAX	Transfer Accumulator to X (Index Register Low)	X ← (A)	–	–	–	–	–	–	INH	97		1
TPA	Transfer CCR to Accumulator	A ← (CCR)	–	–	–	–	–	–	INH	85		1
TST *opr8a* TSTA TSTX TST *oprx8*,X TST ,X TST *oprx8*,SP	Test for Negative or Zero	(M) – 0x00 (A) – 0x00 (X) – 0x00 (M) – 0x00 (M) – 0x00 (M) – 0x00	0	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3D 4D 5D 6D 7D 9E6D	dd ff ff	4 1 1 4 3 5
TSX	Transfer SP to Index Reg.	H:X ← (SP) + 0x0001	–	–	–	–	–	–	INH	95		2
TXA	Transfer X (Index Reg. Low) to Accumulator	A ← (X)	–	–	–	–	–	–	INH	9F		1
TXS	Transfer Index Reg. to SP	SP ← (H:X) – 0x0001	–	–	–	–	–	–	INH	94		2
WAIT	Enable Interrupts; Wait for Interrupt	I bit ← 0; Halt CPU	–	–	0	–	–	–	INH	8F		2+

[1] Bus clock frequency is one-half of the CPU clock frequency.

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		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		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+**		

Opcode in Hexadecimal F0 3 SUB
Number of Bytes 1 IX

HCS08 Cycles
Instruction Mnemonic
Addressing Mode

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

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

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

9E60 6
NEG
3 SP1

HCS08 Cycles
Instruction Mnemonic
Addressing Mode

Chapter 9
Liquid Crystal Display Driver (S08LCDV1)

9.1 Introduction

The LCD driver module is a CMOS charge pump voltage inverter that is designed for low-voltage, low-power operation. The LCD driver module is designed to generate the appropriate waveforms to drive multiplexed numeric, alpha-numeric, or custom LCD panels. Depending on LCD module hardware and software configuration, the LCD panels can be either 3 V or 5 V and be driven by different waveform modes. The LCD module also has several timing and control settings that can be software configured depending on the applications requirements. Timing and control consists of registers and control logic for:

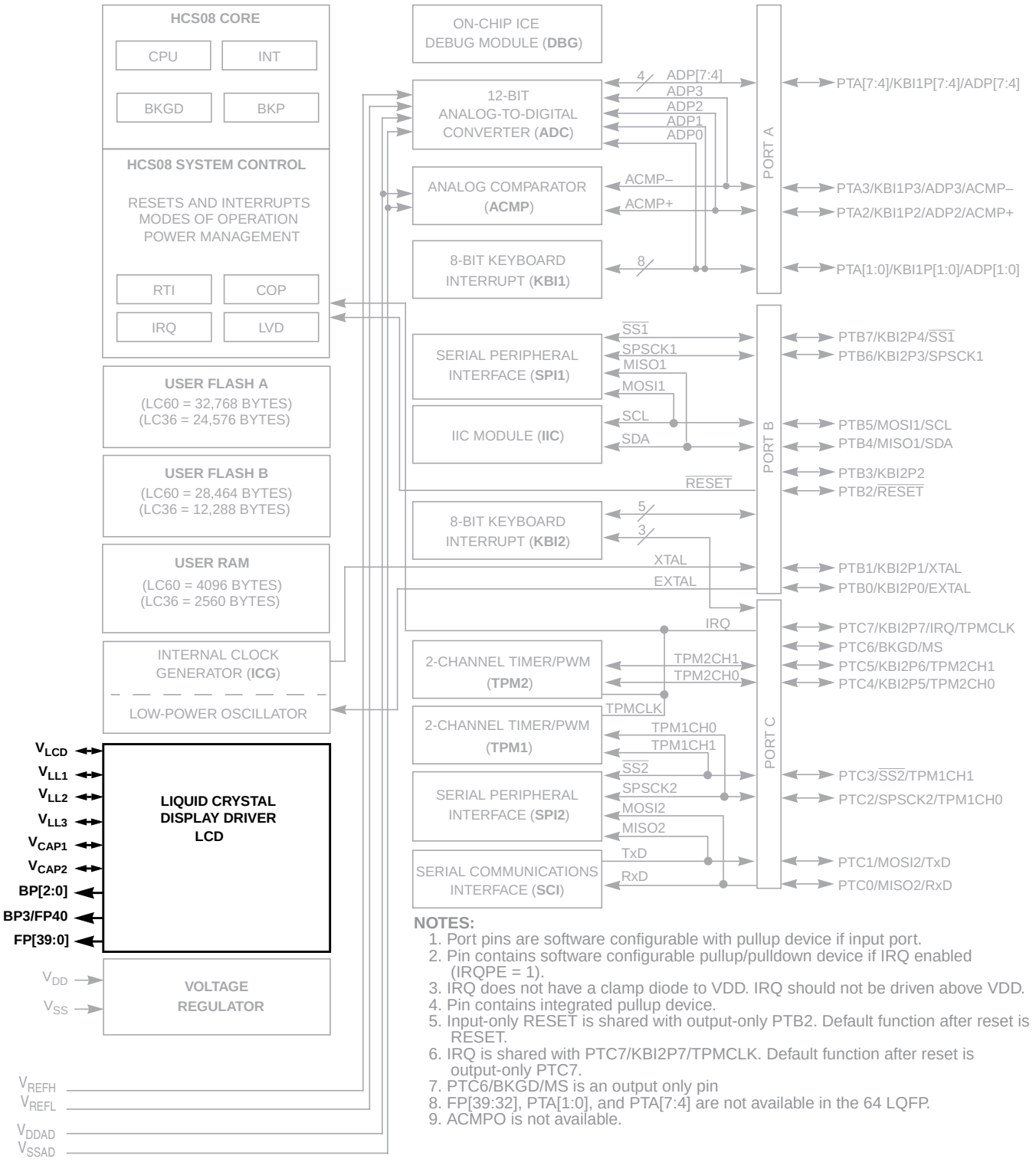
- LCD frame frequency
- Duty cycle select
- Frontplane/backplane select and enable
- Blink modes and frequency
- Operation in low-power modes

In a 64-pin package, the LCD module can be configured to drive 4 backplanes/32 frontplanes (128 segments) or 3 backplanes/33 frontplanes (99 segments). In a 80-pin package, the LCD module can be configured to drive a maximum of 4 backplanes/40 frontplanes (160 segments) or 3 backplanes/41 frontplanes (123 segments). These configurations are summarized in Table 9-1.

Table 9-1. Configuration Options by Package Type

Package Type	No. of Backplanes	No. of Frontplanes	No. of Segments
64-pin	4	32	128
	3	33	99
80-pin	4	40	160
	3	41	123

Figure 9-1 shows the MC9S08LC60 Series block diagram with the LCD highlighted.



NOTES:
1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to VDD. IRQ should not be driven above VDD.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 9-1. MC9S08LC60 Series Block Diagram Highlighting LCD Block and Pins

9.1.1 Features

The LCD module driver module features include:

- Low-power LCD waveform mode
- LCD waveforms functional in wait and stop3 low-power modes
- Selectable frontplane/backplane configuration
  - — Up to 41 frontplanes
  - — Up to 4 backplanes
- Programmable LCD frame frequency
- Programmable blink modes and frequency
  - — Independent blink control for each LCD segment
  - — Blink function remains active in stop3 mode
- Programmable LCD power supply switch making it an ideal solution for battery powered and board level applications
  - — Requires only four external capacitors
  - — Internal LCD power using V_{DD} (1.8 to 3.6 V)
  - — External V_{LCD} power supply option (0.9 to 1.8 V)
- Integrated charge pump for LCD
  - — Hardware configurable to drive 3-V or 5-V LCD panels
  - — On-chip generation of bias voltages, adjustable charge pump frequency
- Dedicated LCD RAM
- LCD frame frequency interrupt event
- Internal ADC channels are connected to V_{LL1} and V_{LCD} to monitor their magnitudes.This feature makes it possible to adjust the contrast by software.
- Low-power, low-voltage CMOS technology

9.1.2 Modes of Operation

The LCD module supports the following operation modes:

Table 9-2. Modes of Operation

Mode	Operation
Stop1	all LCD operation is suspended. It is recommended to blank the display before entering stop1
Stop2	all LCD operation is suspended. It is recommended to blank the display before entering stop2.

Table 9-2. Modes of Operation (continued)

Mode	Operation
Stop3	Depending on the state of the LCDSTP3 bit, the LCD module can operate an LCD panel in stop3 mode. If LCDSTP3 = 1, LCD module clock generation is turned off and the LCD module enters a power conservation state.If LCDSTP3 = 0, the LCD module can operate an LCD panel in stop3, and the LCD module continues to display the current LCD panel contents base on the LCDRAM registers. If the LCD is enabled in stop3, the ICGERCLK must be selected as the clock source since the bus clock is not available in stop3.
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.If LCDWAI = 0, the LCD module can operate an LCD panel in wait, and the LCD module continues to display the current LCD panel contents base on the LCDRAM registers.

Blinking functionality remains active in wait or stop3 mode. If the MCU is in wait or stop3, the LCDRAM registers cannot be changed. Stop3 provides the lowest power consumption state where the LCD module is functional.

9.1.3 Block Diagram

Figure 9-2 is a block diagram of the LCD module.

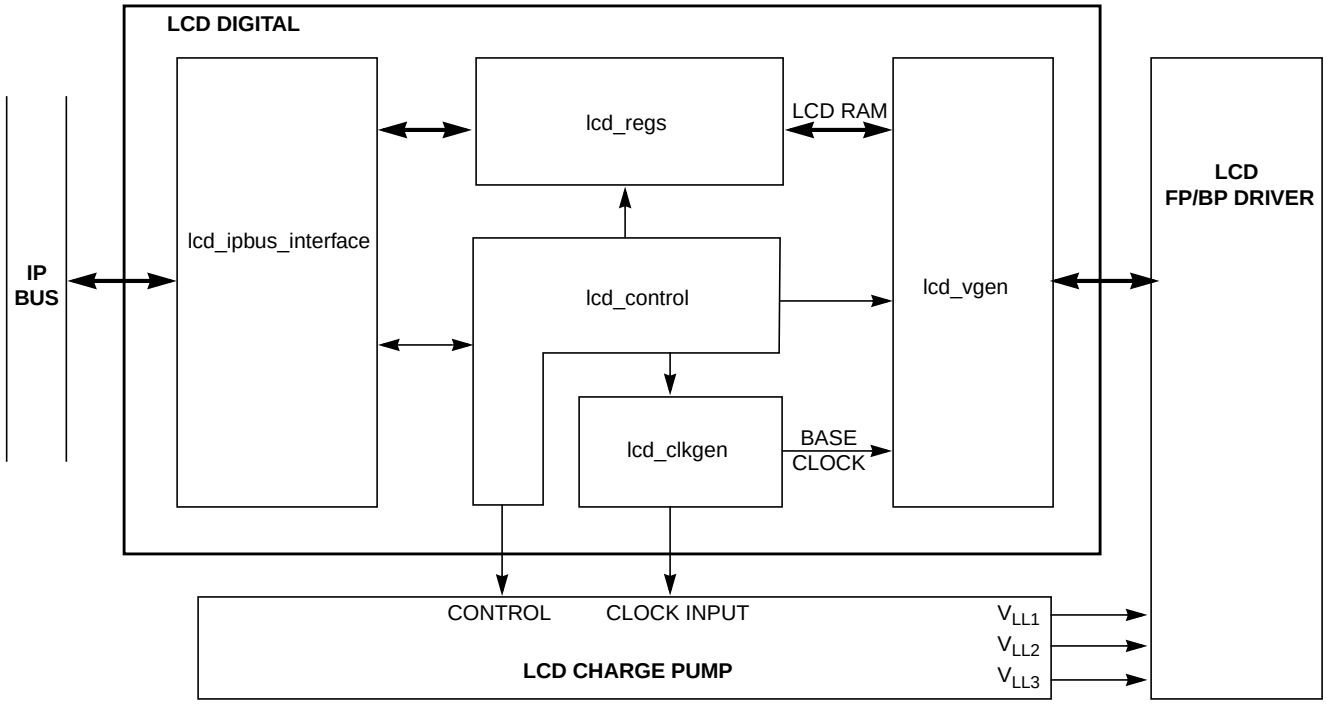


Figure 9-2. LCD Driver Block Diagram

9.2 External Signal Description

The LCD module has several external pins dedicated to power supply and also LCD frontplane/backplane signaling. Table 9-3 itemizes all the LCD external pins. See the Pins and Connections chapter for device-specific pin configurations.

Table 9-3. Signal Properties

Name	Port	Function	Reset State
3 backplanes	BP[2:0],	Backplane waveform signals that connect directly to the pads	High impedance
40 frontplanes	FP[39:0]	Frontplane waveform signals that connect directly to the pads	High impedance
Frontplane/backplane	BP3/FP40	Switchable frontplane/backplane signal that connects directly to the pads	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	—

9.2.1 BP[2:0]

This output signal vector represents the analog backplane waveforms of the LCD module. These signals are connected to the back plane of the LCD panel.

9.2.2 FP[39:0]

This output signal vector represents the analog frontplane waveforms of the LCD module. These signals are connected to the front plane of the LCD panel.

9.2.3 BP3/FP40

This signal vector represents either an analog frontplane or backplane waveform of the LCD module depending on the configuration of the DUTY[1:0] bit field.

9.2.4 V_{LCD}

V_{LCD} is the positive supply voltage for the LCD module waveform generation. V_{LCD} can internally derive from V_{DD} or externally derive from voltage source range from 0.9 to 1.8 Volts. V_{LCD} is connected to a switch capacitor charge pump DC/DC converter (voltage doubler and voltage tripler) which is able to generate double or triple V_{LCD} in order to support either 3-V or 5-V LCD glass. V_{LCD} is also connected internally to an internal ADC channel in order to monitor the V_{LCD} magnitude.

9.2.5 V_{LL1}, V_{LL2}, V_{LL3}

V_{LL1}, V_{LL2}, and V_{LL3} are bias voltages for the LCD module driver waveforms. Internally generated using the internal charge pump when it is enabled.

9.2.6 V_{cap1}, V_{cap2}

The charge pump capacitor is used to transfer charge from the input supply to the regulated output. It is recommended that a low equivalent series resistance (ESR) capacitor be used. Proper orientation is imperative when using a polarized capacitor.

9.3 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.

9.3.1 LCD Control Register 0 (LCDCR0)

	7	6	5	4	3	2	1	0
R	LCDEN	LPWAVE	LCLK2	LCLK1	LCLK0	0	DUTY1	DUTY0
W								
Reset	0	1	0	0	0	0	1	1

= Unimplemented or Reserved

Figure 9-3. LCD Control Register 0 (LCDCR0)

Read: anytime

Write: LCDEN anytime. To avoid segment flicker the clock prescaler bits (LCLK[2:0]) and the duty select bits (DUTY[1:0]) must not be changed when the LCD module is enabled.

Table 9-4. LCDCR0 Field Descriptions

Field	Description
7 LCDEN	**LCD Driver Enable** — The LCDEN bit starts the LCD module waveform generator. 0 All frontplane and backplane pins are disabled. In addition, the LCD module system is disabled and all LCD waveform generation clocks are stopped. 1 LCD module driver system is enabled. All frontplanes pins enabled using the frontplane enable register will output an LCD module driver waveform.The backplane pins will output an LCD module driver waveform based on the settings of DUTY0 and DUTY1.
6 LPWAVE	**LCD Waveform** — The LPWAVE bit allows selection of two types of LCD waveforms. 0 Normal waveforms selected. 1 Low-power waveforms selected.

Table 9-4. LCDCR0 Field Descriptions (continued)

Field	Description
5:3 LCLK[2:0]	**LCD Clock Prescaler** — The LCD module clock prescaler bits are used as a clock divider to generate the LCD waveform base clock as shown in Equation 9-1. The waveform base clock is used, with the LCD module duty cycle configuration to determine the LCD module frame frequency.LCD module frame frequency calculations are provided in 9.4.1.3/p.139. ***Eqn. 9-1*** $\text{LCD Waveform Base Clock} = \frac{\text{LCDCLK}}{16 \times 2^{\text{LCLK[2:0]}}}$ where LCDCLK ≅ 32.768 kHz
1:0 DUTY[1:0]	**LCD Duty Select** — DUTY[1:0] bits select the duty cycle of the LCD module driver. 00 Reserved. 01 Use BP[1:0] (1/2 duty cycle). This mode forces the multiplexed BP3/FP40 pin to be configured as FP40 10 Use BP[2:0] (1/3 duty cycle). This mode forces the multiplexed BP3/FP40 pin to be configured as FP40 11 Use BP[3:0] (1/4 duty cycle). This mode forces the multiplexed BP3/FP40 pin to be configured as BP3 (default)

9.3.2 LCD Control Register 1 (LCDCR1)

	7	6	5	4	3	2	1	0
R	LCDIEN	0	0	0	0	0	LCDWAI	LCDSTP3
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-4. LCD Control Register 1 (LCDCR1)

Read: anytime

Table 9-5. LCDCR1 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 (LPWAVE=0) or sub-frame frequency (LPWAVE=1). 0 The start of the LCD module frame causes a LCD module interrupt request. 1 No Interrupt request is generated by this event.
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 whenever the MCU goes into wait mode.
0 LCDSTP3	**LCD Module Driver and Charge Pump Stop While in Stop3 Mode** 0 Allows the LCD module driver and charge pump to continue running during stop3. 1 Disables the LCD module driver and charge pump whenever the MCU goes into stop3.

9.3.3 LCD Frontplane Enable Registers 0–5 (FPENR0–FPENR5)

When LCDEN = 1, these bits enable the frontplane output waveform on the corresponding frontplane pin.

		7	6	5	4	3	2	1	0
FPENR0	R W	FP7EN	FP6EN	FP5EN	FP4EN	FP3EN	FP2EN	FP1EN	FP0EN
	Reset	0	0	0	0	0	0	0	0
FPENR1	R W	FP15EN	FP14EN	FP13EN	FP12EN	FP11EN	FP10EN	FP9EN	FP8EN
	Reset	0	0	0	0	0	0	0	0
FPENR2	R W	FP23EN	FP22EN	FP21EN	FP20EN	FP19EN	FP18EN	FP17EN	FP16EN
	Reset	0	0	0	0	0	0	0	0
FPENR3	R W	FP31EN	FP30EN	FP29EN	FP28EN	FP27EN	FP26EN	FP25EN	FP24EN
	Reset	0	0	0	0	0	0	0	0
FPENR4	R W	FP39EN	FP38EN	FP37EN	FP36EN	FP35EN	FP34EN	FP33EN	FP32EN
	Reset	0	0	0	0	0	0	0	0
FPENR5	R	0	0	0	0	0	0	0	FP40EN
	W								
	Reset	0	0	0	0	0	0	0	0

Unimplemented or Reserved

Read: anytime

Write: anytime

Table 9-6. FPENR0–FPENR5 Field Descriptions

Field	Description
40:0 FP[40:0]EN	**Frontplane Output Enable** — The FP[40:0]EN bit enables the frontplane driver outputs. If LCDEN = 0, these bits have no effect on the state of the I/O pins.It is recommended to set FP[40:0]EN bits before LCDEN is set. 0 Frontplane driver output disabled on FPnn. 1 Frontplane driver output enabled on FPnn.

9.3.4 LCDRAM Registers (LCDRAM)

The LCDRAM registers control the on/off state for frontplane drivers or the blink enables/disables for each individual LCD segment depending on the state of the LCDDRMS bit in the LCDCMD register. After reset the LCDRAM contents will be indeterminate (I), as indicated by Figure 9-5.

	7	6	5	4	3	2	1	0
R W	FP1BP3	FP1BP2	FP1BP1	FP1BP0	FP0BP3	FP0BP2	FP0BP1	FP0BP0
Reset	I	I	I	I	I	I	I	I
R W	FP3BP3	FP3BP2	FP3BP1	FP3BP0	FP2BP3	FP2BP2	FP2BP1	FP2BP0
Reset	I	I	I	I	I	I	I	I
R W	FP5BP3	FP5BP2	FP5BP1	FP5BP0	FP4BP3	FP4BP2	FP4BP1	FP4BP0
Reset	I	I	I	I	I	I	I	I
R W	FP7BP3	FP7BP2	FP7BP1	FP7BP0	FP6BP3	FP6BP2	FP6BP1	FP6BP0
Reset	I	I	I	I	I	I	I	I
R W	FP9BP3	FP9BP2	FP9BP1	FP9BP0	FP8BP3	FP8BP2	FP8BP1	FP8BP0
Reset	I	I	I	I	I	I	I	I
R W	FP11BP3	FP11BP2	FP11BP1	FP11BP0	FP10BP3	FP10BP2	FP10BP1	FP10BP0
Reset	I	I	I	I	I	I	I	I
R W	FP13BP3	FP13BP2	FP13BP1	FP13BP0	FP12BP3	FP12BP2	FP12BP1	FP12BP0
Reset	I	I	I	I	I	I	I	I
R W	FP15BP3	FP15BP2	FP15BP1	FP15BP0	FP14BP3	FP14BP2	FP14BP1	FP14BP0
Reset	I	I	I	I	I	I	I	I
R W	FP17BP3	FP17BP2	FP17BP1	FP17BP0	FP16BP3	FP16BP2	FP16BP1	FP16BP0
Reset	I	I	I	I	I	I	I	I
R W	FP19BP3	FP19BP2	FP19BP1	FP19BP0	FP18BP3	FP18BP2	FP18BP1	FP18BP0
Reset	I	I	I	I	I	I	I	I
R W	FP21BP3	FP21BP2	FP21BP1	FP21BP0	FP20BP3	FP20BP2	FP20BP1	FP20BP0
Reset	I	I	I	I	I	I	I	I

I = Value is indeterminate

Figure 9-5. LCD Data Registers (LCDRAM)

R W	FP23BP3	FP23BP2	FP23BP1	FP23BP0	FP22BP3	FP22BP2	FP22BP1	FP22BP0
Reset	I	I	I	I	I	I	I	I
R W	FP25BP3	FP25BP2	FP25BP1	FP25BP0	FP24BP3	FP24BP2	FP24BP1	FP24BP0
Reset	I	I	I	I	I	I	I	I
R W	FP27BP3	FP27BP2	FP27BP1	FP27BP0	FP26BP3	FP26BP2	FP26BP1	FP26BP0
Reset	I	I	I	I	I	I	I	I
R W	FP29BP3	FP29BP2	FP29BP1	FP29BP0	FP28BP3	FP28BP2	FP28BP1	FP28BP0
Reset	I	I	I	I	I	I	I	I
R W	FP31BP3	FP31BP2	FP31BP1	FP31BP0	FP30BP3	FP30BP2	FP30BP1	FP30BP0
Reset	I	I	I	I	I	I	I	I
R W	FP33BP3	FP33BP2	FP33BP1	FP33BP0	FP32BP3	FP32BP2	FP32BP1	FP32BP0
Reset	I	I	I	I	I	I	I	I
R W	FP35BP3	FP35BP2	FP35BP1	FP35BP0	FP34BP3	FP34BP2	FP34BP1	FP34BP0
Reset	I	I	I	I	I	I	I	I
R W	FP37BP3	FP37BP2	FP37BP1	FP37BP0	FP36BP3	FP36BP2	FP36BP1	FP36BP0
Reset	I	I	I	I	I	I	I	I
R W	FP39BP3	FP39BP2	FP39BP1	FP39BP0	FP38BP3	FP38BP2	FP38BP1	FP38BP0
Reset	I	I	I	I	I	I	I	I
R	0	0	0	0	FP40BP3	FP40BP2	FP40BP1	FP40BP0
W								
Reset	I	I	I	I	I	I	I	I

I = Value is indeterminate

Figure 9-5. LCD Data Registers (LCDRAM) (continued)

Read: anytime

Write: anytime

The LCDRAM registers provide access to two different register groups. Access to each register group is controlled by the state of the LCDDRMS bit in the LCDCMD register. Each LCDRAM register location provides the waveforms for up to two frontplane drivers.

9.3.4.1 LCDRAM Registers as On/Off Selector (LCDDRMS = 0)

If LCDDRMS bit in the LCDCMD register is deasserted, the LCDRAM register accesses a register bank that controls the on/off state for frontplane drivers.

Table 9-7. LCDRAM Field Descriptions (when LCDDRMS = 0)

Field	Description
FP[n]BP[x]	**Segment On** — If LCDDRMS in the LCDCMD is deasserted (LCDDRMS=0), the FP[n]BP[x] bit in the LCDRAM registers controls on/off state for the LCD segment connected between FP[n] and BP[x].Asserting the FP[n]BP[x] bit displays (turns on) the LCD segment connected between FP[n] and BP[x]. 0 LCD segment off. 1 LCD segment on.

9.3.4.2 LCDRAM Registers as Blink Enable/Disable (LCDDRMS = 1)

If LCDDRMS in the LCDCMD register is asserted, the LCDRAM register accesses a register bank that controls the blink enables/disables for each individual LCD segment.

Table 9-8. LCDRAM Field Descriptions (when LCCDRMS = 1)

Field	Description
FP[n]BP[x]	**LCD Segment Blink Enable** — If LCDDRMS bit in the LCDCMD is asserted (LCDDRMS=1), the FP[n]BP[x] bit in the LCDRAM registers controls blink mode enable/disable state for the LCD segment connected between FP[n] and BP[x].Asserting the FP[n]BP[x] bit enable the blink mode for the LCD segment connected between FP[n] and BP[x] if the associated bit when LCDDRMS = 0 is also set. 0 Disables blink enable for LCD segment. 1 Enables blink enable for LCD segment.

9.3.5 LCD Clock Source Register (LCDCLKS)

	7	6	5	4	3	2	1	0
R W	SOURCE	DIV16	CLKADJ5	CLKADJ4	CLKADJ3	CLKADJ2	CLKADJ1	CLKADJ0
Reset	0	0	0	0	0	1	0	1

Figure 9-6. LCD Clock Source Register (LCDCLKS)

Read: anytime

Write: anytime.It is recommended that CLKADJ[5:0], DIV16, and SOURCE not be modified while the LCDEN bit is asserted.

Table 9-9. LCDCLKS Field Descriptions

Field	Description
7 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 ICGERCLK (external clock reference) as the LCD clock source. 1 Selects the ICGOUT/2 (bus clock) as the LCD clock source.
6 DIV16	**LCD Clock Prescaler Enable**— Enable prescaler by 16. 0 LCD clock prescaler is disabled. 1 LCD clock prescaler is enabled.
5:0 CLKADJ[5:0]	**LCD Clock Source Divider**— The LCD module is designed to operate using a 32.768 kHz clock source for reduced power consumption (LCDCLK = 32.768 kHz). This bit field is used as a clock divider to adjust the LCD clock source to be approximately 32.768 kHz. $\text{LCDCLK} = \text{LCD clock source} / (16^{\text{DIV16}} \times (\text{CLKADJ[5:0]} + 1))$ ***Eqn. 9-2***

9.3.6 LCD Voltage Supply Register (LCDSUPPLY)

	7	6	5	4	3	2	1	0
R W	LCDCPEN	LCDCPMS	CPCADJ1	CPCADJ0	HDRVBUF	BBYPASS	VSUPPLY1	VSUPPLY0
Reset	0	0	1	1	0	1	0	1

Unimplemented or Reserved

Figure 9-7. LCD Voltage Supply Register (LCDSUPPLY)

Read: anytime

Write: anytime. It is recommended that VSUPPLY[1:0] must not be modified while the LCDEN bit is asserted. In addition, VSUPPLY[1:0] must be configured according to the external hardware power supply configuration.

Table 9-10. LCDSUPPLY Field Descriptions

Field	Description
7 LCDCPEN	**LCD Module Charge Pump Enable**— Enables LCD module charge pump for 1/3 bias. 0 LCD charge pump is disabled. (An external bias is required.) 1 LCD charge pump is enabled. (The internal 1/3-bias is forced.)
6 LCDCPMS	**LCD Module Charge Pump Mode Select**— This configuration depends on whether the LCD panel operating voltage is specified as 3 V or 5 V. LCDCPMS configures the internal charge pump to be a voltage doubler (recommended for use with 3-V LCD glass) or a voltage tripler (recommended for use with 5-V LCD glass). 0 Selects voltage doubler. 1 Selects voltage tripler.

Table 9-10. LCDSUPPLY Field Descriptions (continued)

Field	Description
5:4 CPCADJ[1:0]	**LCD Module Charge Pump Clock Adjust-** Adjust the clock source for the charge pump **Charge Pump Clock Rate = LCDCLK / (6 × $2^{(CPADJ[1:0]+1)}$)** ***Eqn. 9-3*** 00 Configures for 2728 Hz charge pump frequency (LCDCLK = 32.768khz) 01 Configures for 1364 Hz charge pump frequency (LCDCLK = 32.768khz) 10 Configures for 682 Hz charge pump frequency (LCDCLK = 32.768khz) 11 Configures for 341 Hz charge pump frequency (LCDCLK = 32.768khz)
3 HDRVBUF	**High Drive Buffer Mode Select** — This bit enhances the VLCD buffer drive active high buffer drive for larger capacitance LCD glass. (See Figure 9-17 for details.) 0 Normal buffer drive. (Ideal for 2000 pF LCD glass.) 1 High buffer drive. (Ideal for 4000 pF LCD glass.)
2 BBYPASS	**Op Amp Control**— Determines whether the internal LCD op amp buffer is bypassed. (See Figure 9-17 for details) 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. It is recommended that this bit field not be modified while the LCD module is enabled (e.g., LCDEN = 1). See Figure 9-17 for more detail.

9.3.7 LCD Blink Control Register (LCDBCTL)

	7	6	5	4	3	2	1	0
R	BLINK	0	0	0	BLKMODE	BRATE2	BRATE1	BRATE0
W								
Reset	0	0	0	0	0	0	0	0

Unimplemented or Reserved

Figure 9-8. LCD Blink Control Register (LCDBCTL)

Read: anytime

Write: anytime. It is recommended that BRATE[1:0] and BLKMODE, must not be modified while BLINK is asserted.

Table 9-11. LCDBCTL Field Descriptions

Field	Description
7 BLINK	**Blink Command** — Starts or stops LCD module blinking.The blink command takes effect at the beginning of the next LCD frame cycle. 0 Disables blinking. 1 Starts blinking at blinking frequency specified by LCD blink rate calculation (see Equation 9-4).
3 BLKMODE	**Blink Mode Select** — Configures whether to blink either individual LCD segments or the entire LCD panel depending on the state of the BLKMODE bit.BLINK must be enabled; if BLINK = 0, this bit has no effect. 0 Blink only individual LCD segments as specified by the LCDRAM register banks. 1 Blink all LCD segments regardless of contents of the LCDRAM register banks.
2:0 BRATE[2:0]	**Blink Rate Configuration**— Selects the frequency at which the LCD display blinks when the BLINK is asserted. Equation 9-4 shows how BRATE[2:0] bit field is used in the LCD blink rate calculation. Equation 9-4 provides an expression for the LCD waveform base clock ***Eqn. 9-4*** $\text{LCD module blink rate} = \frac{\text{LCD waveform base clock}}{2^{(5+\text{ BRATE[2:0]})}}$ LCD module blink rate calculations are provided in 9.4.3.2/p.150**.**

9.3.8 LCD Command and Status Register (LCDCMD)

	7	6	5	4	3	2	1	0
R	LCDIF	0	0	0	LCDDRMS	0	0	BLANK
W							LCDCLR	
Reset	0	0	0	0	0	0	0	0

Unimplemented or Reserved

Figure 9-9. LCD Command Register (LCDCMD)

Read: anytime. The LCDCLR bit always reads zero.

Write: anytime.

Table 9-12. LCDCMD Field Descriptions

Field	Description
7 LCDIF	**LCD Interrupt Flag** — LCDIF indicates that an interrupt condition has occurred. To clear the interrupt, read LCDCMD register and then write a 1 to LCDIF. 0 interrupt condition has not occurred. 1 interrupt condition has occurred.
3 LCDDRMS	**LCD Module Data Register Mode Select** — The LCDRAM registers provide access to two different register groups. Access to each register group is controlled by the state of the LCDDRMS bit. 0 Selects the LCDRAM registers to access registers that control the on/off state for LCD segments. 1 Selects the LCDRAM registers to access registers that control the blink enable/disable state for LCD segments.

Table 9-12. LCDCMD Field Descriptions

Field	Description
1 LCDCLR	**LCD Data Register Clear Command** — Deasserts all accessible bits in the LCDRAM registers. To clear all LCD segment blink enables in the LCDRAM registers, the LCDCLR bit must be asserted only while LCDDRMS = 1.To clear the entire LCD display, the LCDCLR bit must be asserted only while LCDDRMS = 0. 0 Contents of LCD data register are not deasserted by hardware. 1 Deasserts all accessible bits in the LCDRAM registers. The LCDDLR bit clears after all accessible bits in the LCDRAM registers are set to 0.
0 BLANK	**LCD Display Blank Command** — Asserting this bit clears all segments in the LCD display regardless of the contents of the LCDRAM registers or the state of the LCDDRMS bit. BLANK does not disable the LCD timing generator. 0 LCD segments are displayed or cleared depending on the contents of the LCDRAM registers when the LCDDRMS bit is clear. 1 LCD segments are cleared regardless of the contents of the LCDRAM registers or the state of the LCDDRMS bit. The content of the LCDRAM registers is unchanged by the BLANK bit.

9.4 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 LCDCR0 register, it is recommended that the LCD module be configured 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
- Frame frequency interrupt enable
- Blinking frequency and options

The LCD module also provides a frontplane enable control. Setting the frontplane enable bit, FP[n]EN, for a particular pin enables the LCD module functionality of that pin when the LCDEN bit is set. When both the LCDEN and required FP[n]EN bits are set, the LCDRAM can then be used to activate (display) the corresponding LCD segments on an LCD panel.

The LCDRAM registers control the on/off state for the FP and BP segments of the LCD when the LCDDRMS bit in the LCDCMD is cleared.If LCDDRMS = 0 when a 1 is written to the FP[n]BP[x] bit, the corresponding connected segment turns on.When a 0 is written, the segment is turned off. For a detailed description of LCD module operation for a basic seven-segment LCD display, see Section 9.6.1, “LCD Seven Segment Example Description”.

9.4.1 LCD Driver Description

The LCD module driver has three modes of operation:

- 1/2 duty (2 backplanes), 1/3 bias (4 voltage levels)
- 1/3 duty (3 backplanes), 1/3 bias (4 voltage levels)
- 1/4 duty (4 backplanes), 1/3 bias (4 voltage levels)

Note all modes are 1/3 bias. These modes of operation are described in more detail in the following sections.

9.4.1.1 LCD Duty Cycle

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 either static or multiplex.

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 frontplace driver can drive only one LCD segment, static driving limits the number of LCD segments that can be driven. In static mode, only one backplane is required.

In multiplex 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. Multiplex mode is effective in reducing the number of driver circuits and the number of 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. Therefore, the available duty cycle options for the LCD module are 1/2, 1/3, and 1/4. The duty cycle is configured using the DUTY[1:0] bit field in the LCDCR0 register as shown in Table 9-13.

Table 9-13. LCD Module Duty Cycle Modes

Duty	LCDCR0 Register		Backplanes			
	DUTY1	DUTY0	BP3	BP2	BP1	BP0
1/2	0	1	OFF	OFF	BP1	BP0
1/3	1	0	OFF	BP2	BP1	BP0
1/4	1	1	BP3	BP2	BP1	BP0
Reserved	0	0	N/A	N/A	N/A	N/A

9.4.1.2 LCD Bias

Because a single frontplane driver is configured to drive more and more individual LCD segments, more 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.

Defined by Equation 9-5, the bias indicates the number of voltage levels used to power the LCD display.

1 / (voltage level – 1) ***Eqn. 9-5***

9.4.1.3 LCD Module Waveform Base Clock and Frame Frequency

The LCD module is designed to operate using a 32.768-kHz clock input. Two possible clock sources are available to the LCD module has which are selectable by configuration of the SOURCE bit in the LCDCLKS register. The two clock sources include:

- External crystal (SOURCE = 0)
- Internal ICG (SOURCE = 1)

Figure 9-10 shows the LCD clock tree. The clock tree shows the two possible clock sources and shows the LCD frame frequency and blink frequency clock source. The LCD blink frequency is discussed in Section 9.4.3.2, “Blink Frequency.”

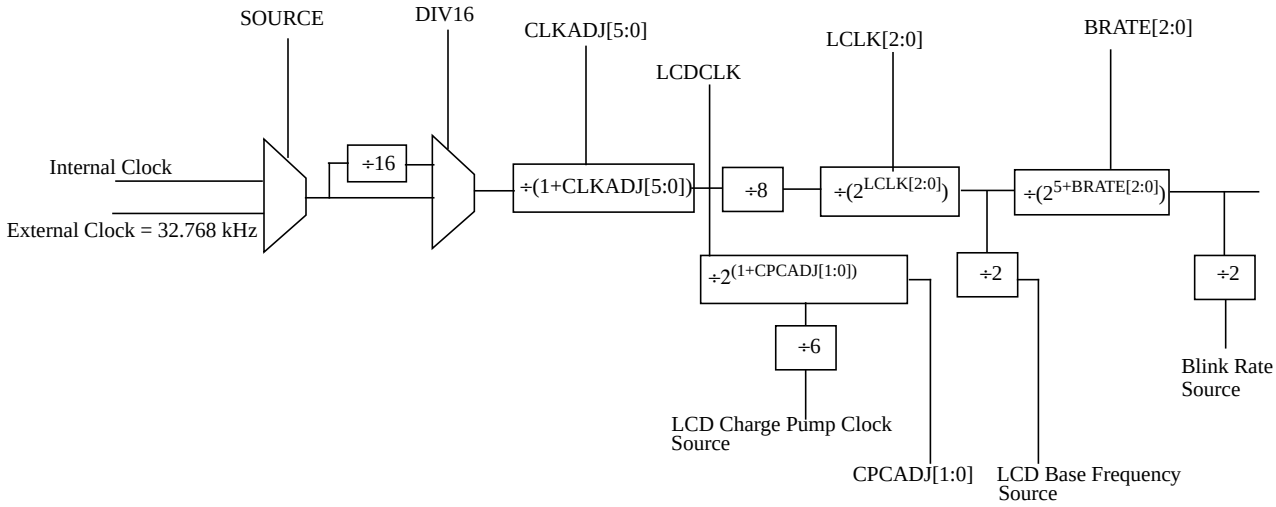


Figure 9-10. LCD Clock Tree

Because the clock sources may not be approximately 32.768 kHz, the DIV16 bit and CLKADJ[5:0] bit field are provided as a clock divider mechanism that can be used to make LCD module clock source adjustments in order to achieve the required 32.768-kHz LCD module clock input, LCDCLK. Table 9-14 provides calculations of LCDCLK using different values of DIV16 and CLKADJ[5:0] for a range of clock inputs frequencies. Using an external 32.768-kHz clock input is required for reduced power consumption applications.

Table 9-14. LCDCLK Calculations (only bold/unshaded values are valid)

CLKADJ[5:0] + 1	Selected Clock Frequencies in kHz (NOTE: DIV16 = 0)						Selected Clock Frequencies in kHz (NOTE: DIV16 = 1)					
	32.8	**100**	**200**	**300**	**400**	**500**	**2000**	**4000**	**9980**	**16000**	**18886**	**20000**
1	**32.8**	100.0	200.0	300.0	400.0	500.0	62.5	250.0	624	1000	1180	1250
2	16.4	50.0	100.0	150.0	200.0	250.0	62.5	125.0	312	500	590	625
3	10.9	**33.3**	66.7	100.0	133.3	166.7	41.7	83.3	207.9	333.3	393.5	416.7
4	8.2	25.0	50.0	75.0	100.0	125.0	**31.3**	62.5	155.9	250.0	295.1	312.5
5	6.6	20.0	40.0	60.0	80.0	100.0	25.0	50.0	124.8	200.0	236.1	250.0
6	5.5	16.7	**33.3**	50.0	66.7	83.3	20.8	41.7	104.0	166.7	196.7	208.3
7	4.7	14.3	28.6	42.9	57.1	71.4	17.9	35.7	89.1	142.9	168.6	178.6
8	4.1	12.5	25.0	37.5	50.0	62.5	15.6	**31.3**	78.0	125.0	147.5	156.3
9	3.6	11.1	22.2	**33.3**	44.4	55.6	13.9	27.8	69.3	111.1	131.2	138.9
10	3.3	10.0	20.0	30.0	40.0	50.0	12.5	25.0	62.4	100.0	118.0	125.0
11	3.0	9.1	18.2	27.3	36.4	45.5	11.4	22.7	56.7	90.9	107.3	113.6
12	2.7	8.3	16.7	25.0	**33.3**	41.7	10.4	20.8	52.0	83.3	98.4	104.2
13	2.5	7.7	15.4	23.1	30.8	38.5	9.6	19.2	48.0	76.9	90.8	96.2
14	2.3	7.1	14.3	21.4	28.6	35.7	8.9	17.9	44.6	71.4	84.3	89.3
15	2.2	6.7	13.3	20.0	26.7	**33.3**	8.3	16.7	41.6	66.7	78.7	83.3
16	2.0	6.3	12.5	18.8	25.0	**31.3**	7.8	15.6	39.0	62.5	73.8	78.1
17	1.9	5.9	11.8	17.6	23.5	29.4	7.4	14.7	36.7	58.8	69.4	73.5
18	1.8	5.6	11.1	16.7	22.2	27.8	6.9	13.9	34.7	55.6	65.6	69.4
19	1.7	5.3	10.5	15.8	21.1	26.3	6.6	13.2	**32.8**	52.6	62.1	65.8
20	1.6	5.0	10.0	15.0	20.0	25.0	6.3	12.5	**31.2**	50.0	59.0	62.5
21	1.6	4.8	9.5	14.3	19.0	23.8	6.0	11.9	29.7	47.6	56.2	59.5
22	1.5	4.5	9.1	13.6	18.2	22.7	5.7	11.4	28.4	45.5	53.7	56.8
23	1.4	4.3	8.7	13.0	17.4	21.7	5.4	10.9	27.1	43.5	51.3	54.3
24	1.4	4.2	8.3	12.5	16.7	20.8	5.2	10.4	26.0	41.7	49.2	52.1
25	1.3	4.0	8.0	12.0	16.0	20.0	5.0	10.0	25.0	40.0	47.2	50.0
26	1.3	3.8	7.7	11.5	15.4	19.2	4.8	9.6	24.0	38.5	45.4	48.1
27	1.2	3.7	7.4	11.1	14.8	18.5	4.6	9.3	23.1	37.0	43.7	46.3
28	1.2	3.6	7.1	10.7	14.3	17.9	4.5	8.9	22.3	35.7	42.2	44.6
29	1.1	3.4	6.9	10.3	13.8	17.2	4.3	8.6	21.5	34.5	40.7	43.1
30	1.1	3.3	6.7	10.0	13.3	16.7	4.2	8.3	20.8	**33.3**	39.3	41.7
31	1.1	3.2	6.5	9.7	12.9	16.1	4.0	8.1	20.1	**32.3**	38.1	40.3
32	1.0	3.1	6.3	9.4	12.5	15.6	3.9	7.8	19.5	31.3	36.9	39.1
33	1.0	3.0	6.1	9.1	12.1	15.2	3.8	7.6	18.9	30.3	35.8	37.9
34	1.0	2.9	5.9	8.8	11.8	14.7	3.7	7.4	18.3	29.4	34.7	36.8
35	0.9	2.9	5.7	8.6	11.4	14.3	3.6	7.1	17.8	28.6	**33.7**	35.7
36	0.9	2.8	5.6	8.3	11.1	13.9	3.5	6.9	17.3	27.8	**32.8**	34.7
37	0.9	2.7	5.4	8.1	10.8	13.5	3.4	6.8	16.9	27.0	**31.9**	33.8
38	0.9	2.6	5.3	7.9	10.5	13.2	3.3	6.6	16.4	26.3	31.1	**32.9**
39	0.8	2.6	5.1	7.7	10.3	12.8	3.2	6.4	16.0	25.6	30.3	**32.1**
40	0.8	2.5	5.0	7.5	10.0	12.5	3.1	6.3	15.6	25.0	29.5	**31.3**
41	0.8	2.4	4.9	7.3	9.8	12.2	3.0	6.1	15.2	24.4	28.8	30.5
42	0.8	2.4	4.8	7.1	9.5	11.9	3.0	6.0	14.9	23.8	28.1	29.8

The value of LCDCLK is important because it is used to generate the LCD module waveform base clock frequency. Equation 9-1 provides an expression for the LCD module waveform base clock frequency calculation. Equation 9-1 illustrates that the LCD module waveform base clock frequency also depends on the LCLK[2:0] bit field.

The LCD module waveform base clock is the basis for the calculation of the LCD module frame frequency. The LCD module frame frequency is a function of LCD module duty cycle as shown in Equation 9-6. Table 9-15 shows LCD module frame frequency calculations considering several possible LCD module configurations of LCLK[2:0] and DUTY[1:0]. The using LCD module frame frequency calculations a based on LCD module clock input value approximately 32 kHz.

$$\text{LCD module frame frequency} = (\text{LCD module waveform base clock}) \times (\text{duty cycle}) \qquad \textit{Eqn. 9-6}$$

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 a LCD module frame frequency in the range of 30 to 100 Hz is required. LCD module frame frequency less that 30 Hz or greater that 100 Hz are out of specification and invalid. Selecting lower values for the LCD base and frame frequency results in lower current consumption for the LCD module.

Table 9-15. LCD Module Frame Frequency Calculations[1], [2]

LCLK[2:0]	LCD Clock Input Divider ($16 \times 2^{LCLK[2:0]}$)	LCD Base Frequency (Hz)	LCD Frame Frequency (Hz)		
0	16	2049.3	1024	683	512.32
1	32	1024.7	512.32	341.55	256.16
2	64	512.3	256.16	170.77	128.08
3	128	256.2	128.08	**85.38**	**64.04**
4	256	128.1	**64.04**	**42.69**	**32.02**
5	512	64.0	**32.02**	21.34	16.01
6	1024	32.0	16.01	10.67	8.00
7	2048	16.0	8.00	5.33	4.00
			Duty Cycle		
			1/2/	1/3	1/4

1 LCD clock input ~ 32.768 kHz

2 Shaded table entries are out of specification and are invalid.

9.4.1.4 LCD Waveform Examples

This section shows the timing examples of the LCD output waveforms for the available modes of operation. As shown in Table 9-16, all examples use 1/3 bias mode.

Table 9-16. Configurations for Example LCD Waveforms

	Bias Mode	DUTY[1:0]	Duty Cycle	LPWAVE bit
Example 1	1/3	01	1/2	0
Example 2		01	1/2	1
Example 3		10	1/3	0
Example 4		10	1/3	1
Example 5		11	1/4	0
Example 6		11	1/4	1

9.4.1.4.1 Example 1: 1/2 Duty Multiplexed with 1/3 Bias Mode (Normal Waveform)

Duty=1/2: DUTY[1:0] = 01 (BP2 and BP3 are not used)

Normal Waveform selected: LPWAVE = 0

LCDRAM = XX10

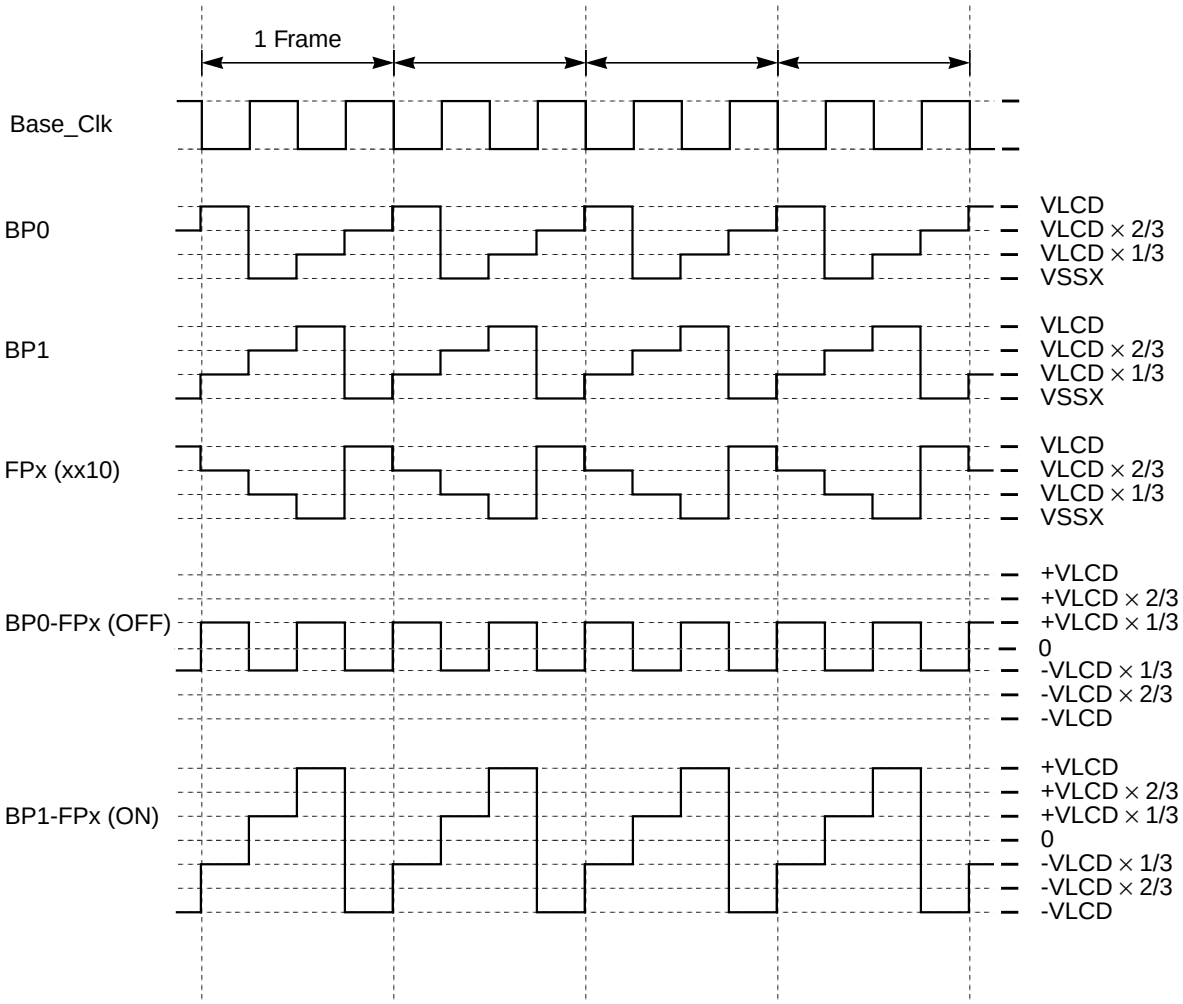


Figure 9-11. 1/2 Duty and 1/3 Bias (Normal Waveform)

9.4.1.4.2 Example 2: 1/2 Duty Multiplexed with 1/3 Bias Mode (Low-power Waveform)

Duty=1/2: DUTY[1:0] = 01 (BP2 and BP3 are not used)

Low-Power Waveform selected: LPWAVE = 1

LCDRAM = XX10

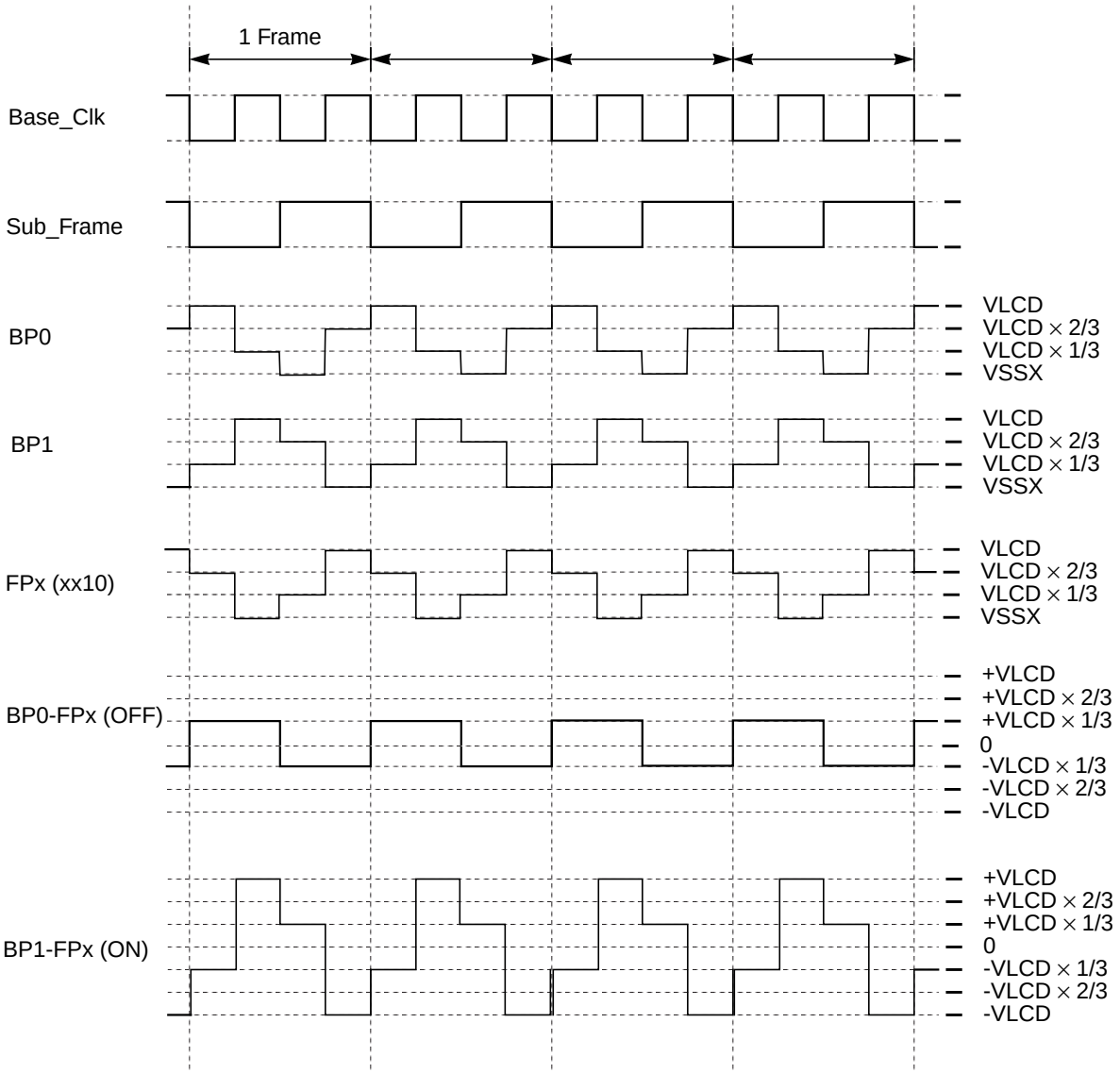


Figure 9-12. 1/2 Duty and 1/3 Bias (Low-power Waveform)

9.4.1.4.3 Example 3: 1/3 Duty Multiplexed with 1/3 Bias Mode (Normal Waveform)

Duty=1/3: DUTY[1:0] = 10 (BP3 are not used)

Normal Waveform selected: LPWAVE = 0

LCDRAM = X010

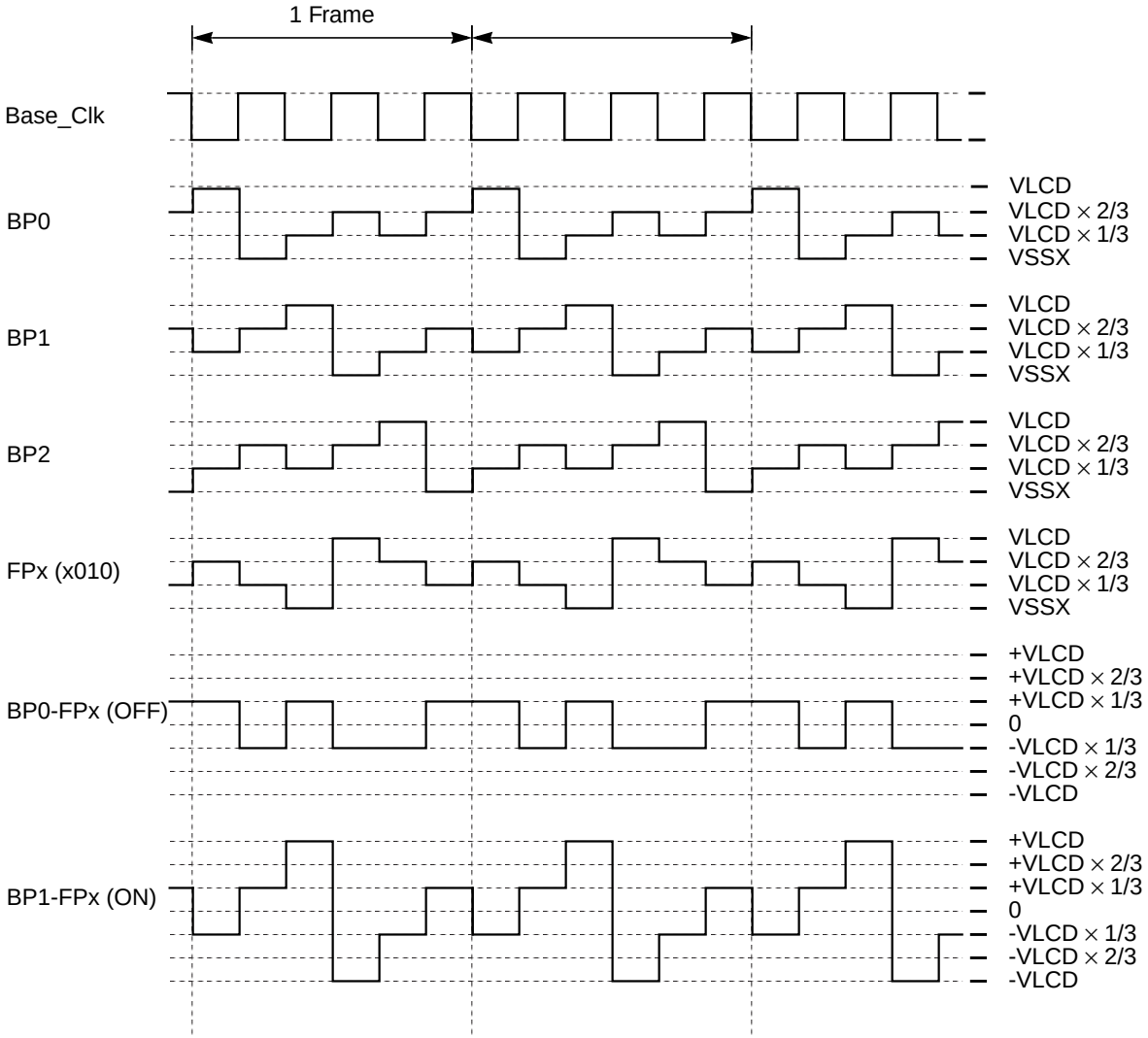


Figure 9-13. 1/3 Duty and 1/3 Bias (Normal Waveform)

9.4.1.4.4 Example 4: 1/3 Duty Multiplexed with 1/3 Bias Mode (Low-power Waveform)

Duty=1/3: DUTY[1:0] = 10 (BP3 are not used)

Low-power Waveform selected: LPWAVE = 1

LCDRAM = X010

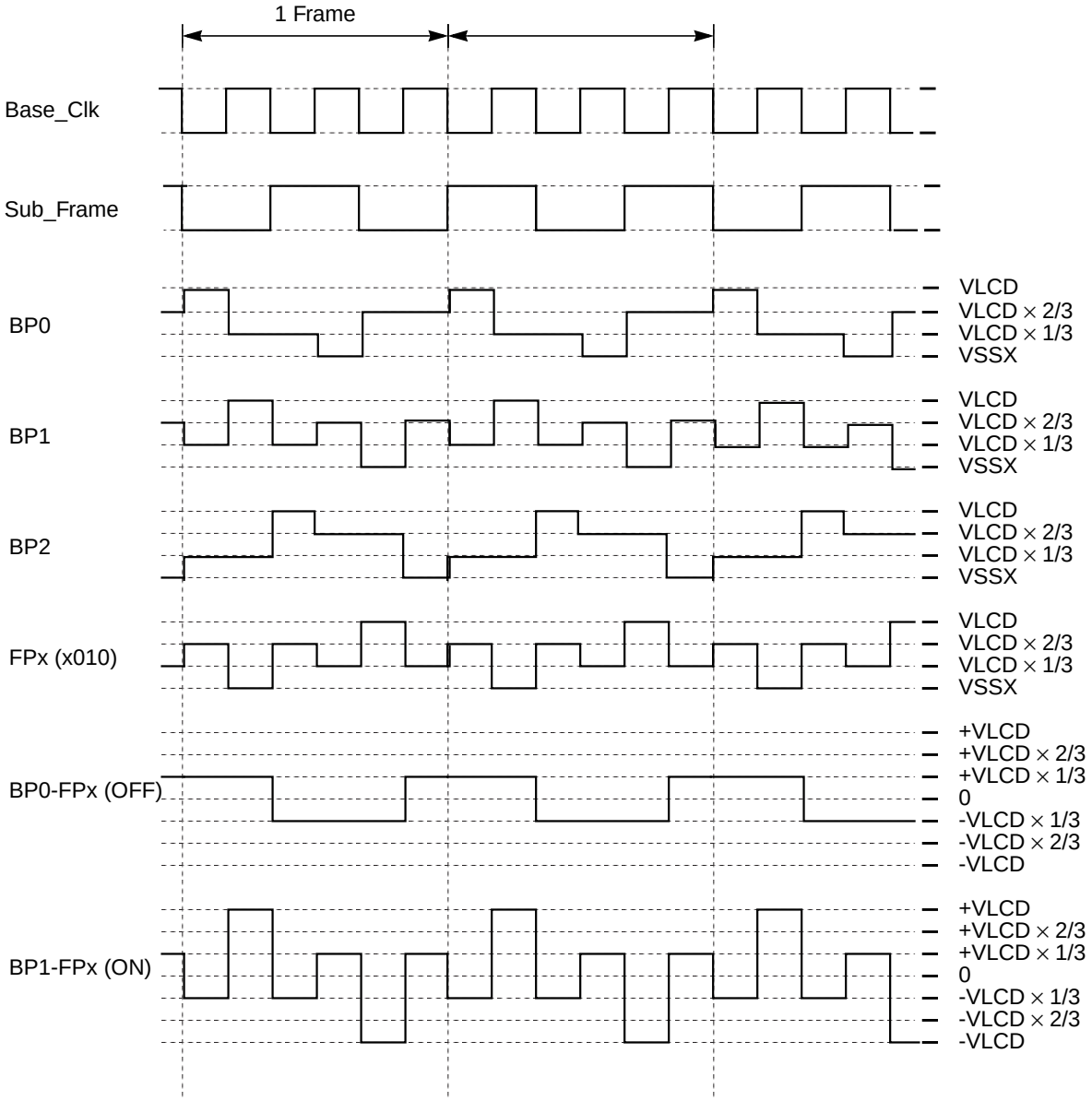


Figure 9-14. 1/3 Duty and 1/3 Bias (Low-power Waveform)

9.4.1.4.5 Example 5: 1/4 Duty Multiplexed with 1/3 Bias Mode (Normal Waveform)

Duty=1/4: DUTY[1:0] = 11 (All available backplanes used)

Normal Waveform selected: LPWAVE = 0

LCDRAM = 1001

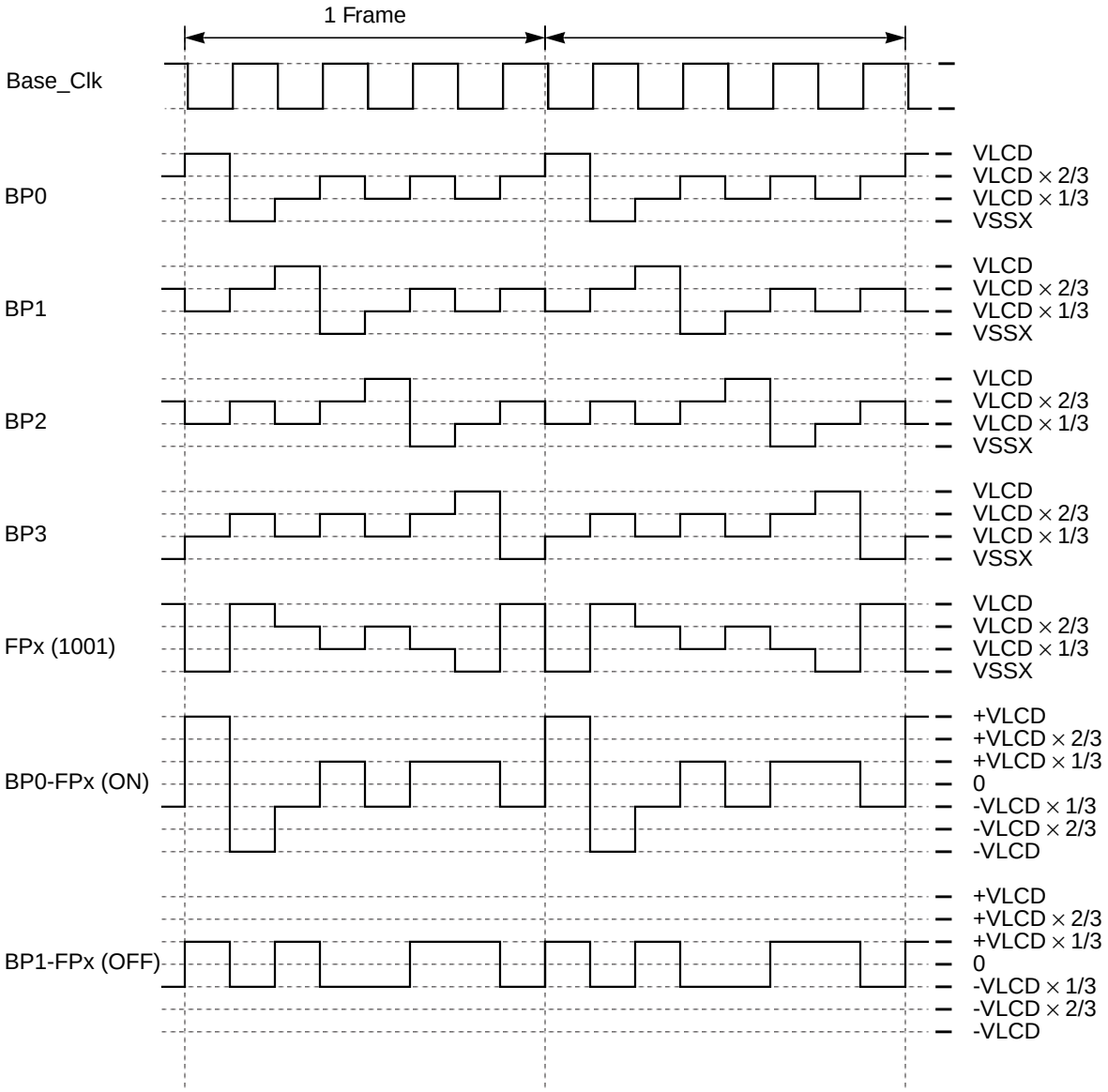


Figure 9-15. 1/4 Duty and 1/3 Bias (Normal Waveform)

9.4.1.4.6 Example 6: 1/4 Duty Multiplexed with 1/3 Bias Mode (Low-power Waveform)

Duty=1/4: DUTY[1:0] = 11 (All available backplanes used)

Low-power Waveform selected: LPWAVE = 1

LCDRAM = 1001

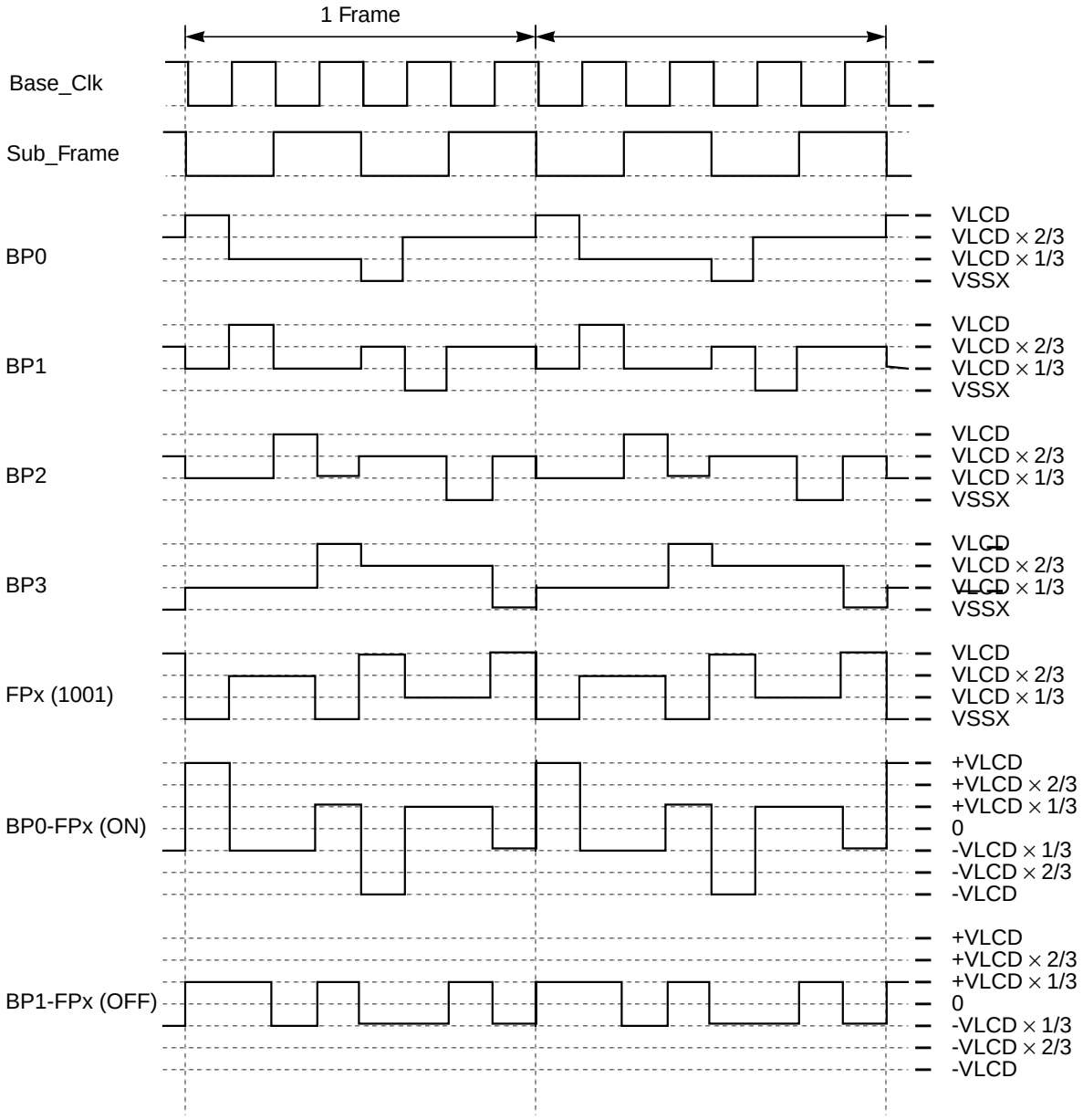


Figure 9-16. 1/4 Duty and 1/3 Bias (Low-power Waveform)

9.4.2 LCDRAM Registers

For a segment on the LCD panel to be displayed, data must be written to the LCDRAM registers. Each bit in the LCDRAM registers correspond to a segment on the LCD panel.

The LCDRAM registers provide access to two different register groups depending on the state of the LCDDRMS bit in the LCDCMD register. If LCDDRMS = 0, the LCDRAM register accesses a register bank that controls the on/off state for frontplane drivers. If LCDDRMS = 1, the LCDRAM register bank accesses a register bank that enables the blink mode for each individual LCD segment.

If LCDDRMS = 0, when the LCDEN bit is set and the corresponding FP[31:0]EN bit is set, writing a 1 to a given LCDRAM location will result in the corresponding display segment being driven with a differential root mean square (RMS) voltage necessary to turn the segment on.Writing a 0 to a given location will result in the corresponding display segment being driven with a differential RMS voltage necessary to turn the segment off.

The LCDRAM is a dual port RAM that interfaces with the internal address and data buses of the MCU. When LCDEN = 0, the LCDRAM registers can be used as on-chip RAM. Writing or reading of the LCDEN bit does not change the contents of the LCDRAM registers.After a reset, the LCDRAM contents will be indeterminate.

9.4.2.1 LCDRAM Data Clear Command

The LCD module data register clear command deasserts all accessible bits in the LCDRAM registers. LCDDRMS bit in the LCDCMD register determines which LCDRAM registers bits are accessible.To clear all LCD segment blink enables in the LCDRAM registers, the LCDCLR bit must be asserted only when the LCDDRMS bit is asserted. To clear the entire LCD display, the LCDCLR bit must be asserted only when the LCDDRMS bit is deasserted.

9.4.2.2 LCDRAM Data Blank Command

The LCD module display blank command clears all segments in the LCD display regardless of the contents of the LCDRAM registers or the state of the LCDDRMS bit. This bit does not disable the LCD module timing generator. Writing or reading of the BLANK bit does not change the contents of the LCDRAM registers.

9.4.3 LCD Blinking

The LCD module LCD panel blink capabilities are very flexible. The LCD module can be configured to blink either individual LCD segments or the entire LCD panel. The blink rate frequency is configured using the BRATE[2:0] bit field.

The LCD will blink at the configured frequency while the BLINK bit in the LCDBCTL register is set to 1. When the BLINK bit is modified to start or stop the LCD display blinking, the BLINK command change takes place at the beginning of the next LCD frame cycle.

9.4.3.1 LCD Segment Blinking

To configure all LCD segments to blink regardless of the contents of the LCDRAM bits while LCDDRMS = 1, the BLKMODE bit in the LCDBCTL control register must to set to 1. To configure individual LCD segments to blink, the BLKMODE bit in the LCDBCTL control register must be deasserted.

If BLKMODE = 0, asserting the LCDRAM FP[n]BP[x] bits while LCDDRMS = 0 and LCDDRMS = 1 enables the LCD segment connected between FP[n] and BP[x] to blink when BLINK = 1. Each LCDRAM register controls two frontplane drivers.

9.4.3.2 Blink Frequency

The LCD module waveform base clock is the basis for the calculation of the LCD module blink frequency. The LCD module blink frequency is equal to the LCD module waveform base clock divided by the BRATE[2:0] divider. Table 9-17 shows LCD module blink frequency calculations for all values of BRATE[2:0] and LCLK[2:0].

Table 9-17. Blink Frequency Calculations
(Blink Rate = LCD Base (Hz) ÷ Blink Divider)

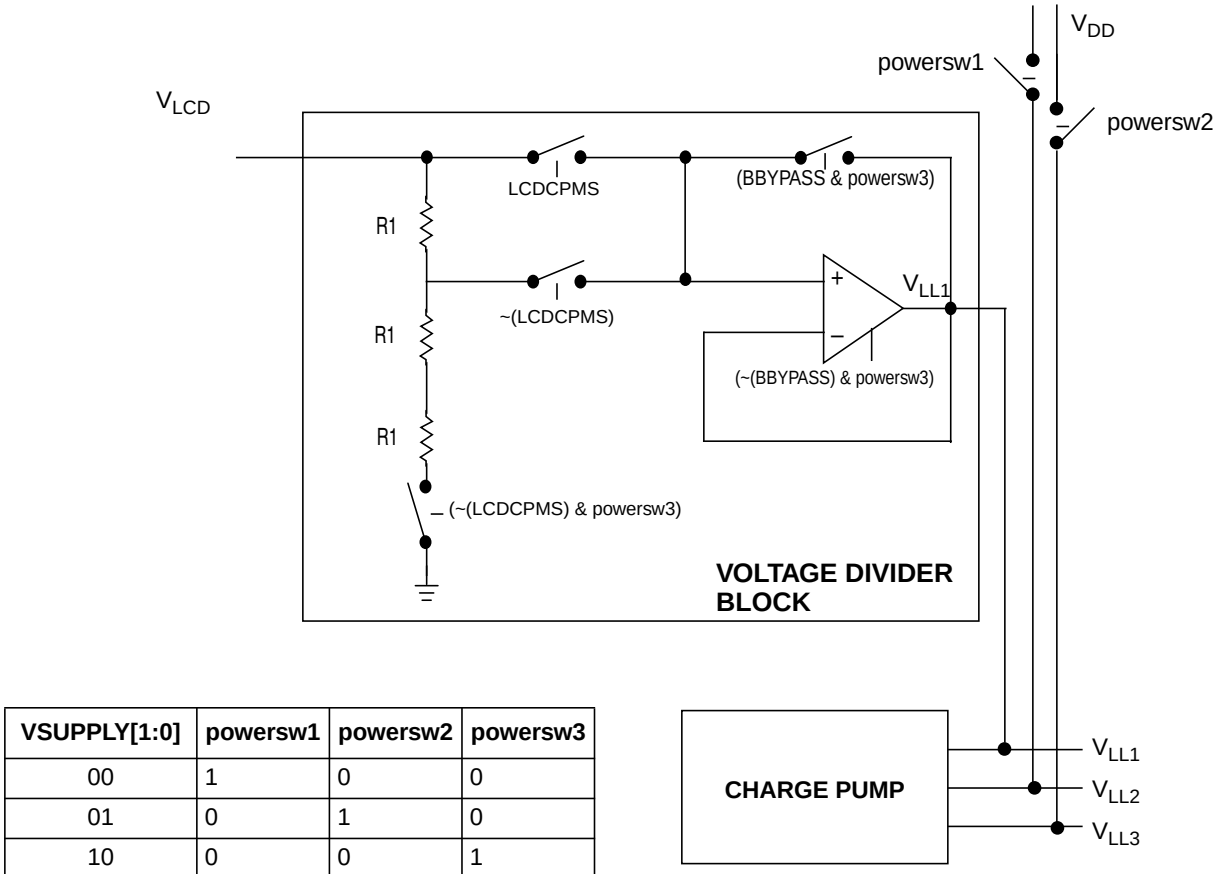
LCLK[2:0]	LCD Base Frequency (Hz)	Blink Frequency							
0	2049.3	64.0	32.0	16.0	8.00	4.00	2.00	1.00	0.50
1	1024.7	32.0	16.0	8.00	4.00	2.00	1.00	0.50	0.25
2	512.3	16.0	8.00	4.00	2.00	1.00	0.50	0.25	0.13
3	256.2	8.00	4.00	2.00	1.00	0.50	0.25	0.13	0.06
4	128.1	4.00	2.00	1.00	0.50	0.25	0.13	0.06	0.03
5	64.0	2.00	1.00	0.50	0.25	0.13	0.06	0.03	0.02
6	32.0	1.00	0.50	0.25	0.13	0.06	0.03	0.02	0.01
7	16.0	0.50	0.25	0.13	0.06	0.03	0.02	0.01	0.00
		Blink Divider = $2^{(5+ BRATE[2:0])}$							
		32	**64**	**128**	**256**	**512**	**1024**	**2048**	**4096**

1 Shaded table entries are out of specification and are not valid

9.4.4 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 9-17 provides a block diagram for the LCD charge pump and a V_{LCD} voltage divider.

The VSUPPLY[1:0] bit field in the LCDSUPPLY register is used to configure the LCD module power supply source. VSUPPLY[1:0] indicates the state of internal signals used to configure power switches as shown in the Table in Figure 9-17. The block diagram in Figure 9-17 illustrates several potential operational modes for the LCD module including configuration of the LCD module power supply source using internal V_{DD} or an external supply.



VSUPPLY[1:0]	powersw1	powersw2	powersw3
00	1	0	0
01	0	1	0
10	0	0	1
11	0	0	0

Figure 9-17. LCD Charge Pump and V_{LCD} Voltage Divider Block Diagram

Figure 9-17 also illustrates a buffer, a voltage follower with an ideal op amp. The buffer, if enabled, gives VIn = VOut, and, because the input impedance of the op amp is very high, VIn is isolated from VOut This isolation can protect VIn from things like current draw from VOut; however, if the buffer is disabled ((~BBYPASS & powersw3) = 0), the output and input will be configured in a tri-state condition; that is, 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 state of the LCDEN bit.

The charge pump requires external capacitance for is operation. To provide this external capacitance, the V_{cap1} and V_{cap2} external pins are provided. It is recommended that a low equivalent series resistance (ESR) capacitor be used. Proper orientation is imperative when using a polarized capacitor. The recommended value for the external capacitor is 0.1 µF.

9.4.4.1 LCD Charge Pump and Voltage Divider

The LCD charge pump is a voltage tripler. Using the voltage divider and the charge pump, the LCD module can effectively double or triple $V_{LCD.}$ This LCD module configurability makes the LCD module compatible with both 3-V or 5-V LCD glass.

The LCD module charge pump mode select bit (LCDCPMS) in the LCDSUPPLY register configures the LCD module operational mode as a voltage doubler or a voltage tripler. In Figure 9-17, LCDCPMS 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 LCDCPMS = 0, the LCD module is configured as a voltage doubler (recommended to V_{LCD} be used with 3-V LCD glass) by enabling the voltage divider. If LCDCPMS = 1, the LCD module is configured as a voltage tripler (recommended to be used with 5-V LCD glass) by shorting the voltage divider. The LCDCPMS configuration depends on the LCD panel operating voltage specification in the design application.

9.4.4.1.1 LCD Charge Pump Enabled

The LCDCPEN bit in the LCDSUPPLY register enables the charge pump. When charge pump is enabled (LCDCPEN = 1), V_{LL1}, V_{LL2}, and V_{LL3} are will be generated internally.

9.4.4.1.2 LCD Charge Pump Disabled

When charge pump is disabled (LCDCPEN = 0), V_{LL1}, V_{LL2}, and V_{LL3} are not generated internally. Instead, V_{LL1}, V_{LL2}, and V_{LL3} must be provided by external hardware.

9.4.4.2 LCD Power Supply and Voltage Buffer Configuration

The LCD power supply can be internally derived from V_{DD} or it can be externally derived from a voltage source in the range between 0.9 to 1.8 Volts that is applied to the V_{LCD} pin. The Table below provides a more detailed description of the power state of the LCD module which depends on the configuration of the VSUPPLY[1:0], LCDCPMS, BBYPASS, and LCDCPEN bits.

Table 9-18. V_{DD} Switch Option

VSUPPLY[1:0]	LCDCPMS	BBYPASS	LCDCPEN	LCD Power Supply Configuration	LCD Operational State
00	X	X	0	Initial V_{LL2} voltage to V_{DD} level.	LCD disabled
			1	Internal power supply. V_{LL2} is generated from V_{DD}.	LCD operational
01	X	X	0	Initial V_{LL3} voltage to V_{DD} level.	LCD disabled
			1	Internal power supply. V_{LL3} is generated from V_{DD}.	LCD operational Minimum current consumption
10	x	x	0	Bias voltages not generated.	LCD disabled
	0	0	1	External power supply for V_{LCD}. Buffered doubler mode.	LCD operational Maximum current consumption
	0	1	1	External power supply for V_{LCD}. Un-buffered doubler mode.	LCD operational
	1	0	1	External power supply for V_{LCD}. Buffered tripler mode.	LCD operational
	1	1	1	External power supply for V_{LCD}. Un-buffered tripler mode.	LCD operational Minimum current consumption
11	X	X	0	External power supply for V_{LL1},V_{LL2}, and VLL3 required. V_{LCD} pin floating.	LCD Operational
			1	V_{LCD} pin floating.	Invalid LCD power configuration

Figure 9-18 shows that if VSUPPLY[1:0] = 10 or 11, the LCD module is configured for an external power source. If VSUPPLY[1:0] = 00 or 01, the LCD power supply is configured to be internally derived from V_{DD}.

9.4.4.2.1 LCD External Power Supply, VSPUPPLY[1:0] = 10

When VSPUPPLY[1:0] = 10, only the powersw3 signal is asserted and the LCD module is configured to be powered via an external voltage input on V_{LCD} (Recall V_{LCD} is specified to be in the range from 0.9 V to 1.8 V). The figure above shows that V_{LCD} is an input to the voltage divider block and is related to V_{LL1}. The voltage divider block uses the states of LCDCPMS, BBYPASS, and powersw3 to derive a state for V_{LL1}.

The output of the voltage divider block is $V_{LL1.}$ V_{LL1} is connected to the internal charge pump via the $V_{ref.}$ Using the charge pump, the value of V_{LL1} is tripled and outputted as $V_{LL3.}$ V_{LL3}, a 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 LCDCPMS bit configuration, V_{LL3} will be equal to $3*V_{LCD}$ or $3*(2/3*V_{LCD})$ (see Section 9.4.4, “LCD Charge Pump, Voltage Divider, and Power Supply Operation). Table 9-19 shows the selected V_{LL1} and V_{LL3} values based on the input value of V_{LCD}.

Table 9-19. V_{LL1} Typical Values

V_{LCD}	LCDCPMS = 0 Voltage Doubler		LCDCPMS = 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	$(2/3) \times 1.4$ V	2.8 V	1.4 V	4.2 V
1.5 V	$(2/3) \times 1.5$ V	3.0 V	1.5 V	4.5 V
1.7 V	$(2/3) \times 1.7$ V	3.3 V	1.67 V	5.0 V
1.8 V	$(2/3) \times 1.8$ 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 (LCDCPEN = 1). For a typical LCD panel, the bias voltages in 1/3 bias mode would be:

- $V_3 = V_{LL3} = V_{LCDON} = 3 \times V_{ref}$
- $V_2 = V_{LL2} = 2 \times V_{ref}$
- $V_1 = V_{LL1} = V_{ref}$
- $V_0 = V_{SS}$

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.

9.4.4.2.2 LCD External Power Supply, VSPUPPLY[1:0] = 11

When VSPUPPLY[1:0] = 11, powersw1, powersw2, and powersw3 are deasserted. Moreover 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 either $V_{LL1,}$ $V_{LL2,}$ or V_{LL3}.

9.4.4.2.3 LCD Internal Power Supply, VSUPPLY[1:0] = 00 or 01

V_{DD} is specified to be from 1.8 V to 3.6 V. V_{DD} is used as the LCD module power supply when VSUPPLY[1:0] = 00 or 01(see Table 9-18). When powering the LCD module using V_{DD}, the charge pump must be enabled (LCDCPEN = 1). Table 9-20 provides recommendations regarding configuration of the VSUPPLY[1:0] bit field when using both 3-V and 5-V LCD panels.

Table 9-20. 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} = 1v • V_{DD} = V_{LL2} = 2v • V_{LL3} = 3v	• V_{LL1} = 1.67v • V_{DD} = V_{LL2} = 3.3v • V_{LL3} = 5v
01	V_{LL3} is generated from V_{DD}	• V_{LL1} = 1v • V_{LL2} = 2v • V_{DD} = V_{LL3} = 3v	Invalid LCD power configuration

9.4.5 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
- All frontplane enable bits, FP[n]EN, are cleared
- LCLK[2:0], VSUPPLY[2:0], BBYPASS, and BRATE[2:0] revert to their reset values

9.4.6 Interrupts

When an LCD module frame (LPWAVE = 0) or sub-frame (LPWAVE = 1) frequency interrupt event occurs, the LCDIF bit in the LCDCMD register is asserted. The LCDIF bit remains asserted until the LCD module frame frequency interrupt is cleared by software. The interrupt can be cleared by software by writing a 1 to the LCDIF bit.

If a both the LCDIF bit in the LCDCMD register and the LCDIEN bit in the LCDCR1 register are set, an LCD interrupt signal asserts.

For both normal waveform and low-power waveform, configured for the same frequency with the same clock configuration. The low-power waveform splits a frame into two subframes with equal duration. See Figure 9-11 through Figure 9-16.

When an LCD module frame(LPWAVE=0) or sub-frame(LPWAVE=1) frequency interrupt event occurs, the LCDIF bit in the LCDCMD register is asserted. The LCDIF bit remains asserted until the LCD module frame frequency interrupt is cleared by software. The interrupt can be cleared by software by writing a 1 to the LCDIF bit.

9.5 Initialization Section

This section provides a recommended initialization sequence for the LCD module and also includes initialization examples for several possible LCD application scenarios.

9.5.1 Initialization Sequence

The list below provides a recommended initialization sequence for the LCD module.

1. LCDCLKS register
   a) Configure LCD clock source (SOURCE bit)
   b) Adjust the clock source to achieve a value for LCDCLK of ~ 32 kHz (CLKADJ[5:0] & DIV16)
2. LCDSUPPLY register
   a) Enable charge pump (LCDCPEN bit)
   b) Configure the LCD module for doubler or tripler mode (LCDCPMS bit)
   c) Configure charge pump clock (CPCADJ[1:0])
   d) Configure HDRVBUF
   e) Configure op amp switch (BBYPASS bit)
   f) Configure LCD power supply (VSUPPLY[1:0])
3. LCDCR1 register
   a) Configure the LCD frame frequency interrupt (LCDIEN bit)
   b) Configure LCD behavior in low power mode (LCDWAI and LCDSTP3 bits)
4. LCDCR0 register
   a) Configure LCD duty cycle (DUTY[1:0])
   b) Configure LPWAVE
   c) Select and configure the LCD frame frequency (LCLK[2:0])
5. LCDBCTL register
   a) Configure the blink mode to blink individual or blink all segments (BLKMODE bit)
   b) Configure the blink frequency (BRATE[2:0])
6. FPENR[5:0] register
   a) Enable the LCD module frontplane waveform output (FP[40:0]EN bits)
7. LCDCR0 register
   a) Enable the LCD module (LCDEN bit)

9.5.2 Initialization Examples

This section provides initialization information for configuration of the LCD. Each example details the register and bit field values required in order to achieve the appropriate LCD configuration for a given LCD application scenario. Table 9-21 lists each example and the setup requirements.

Table 9-21. LCD Application Scenarios

Example	Operating Voltage, V_{DD}	LCD Clock Source	LCD Glass Operating Voltage	Required LCD segments	LCD Frame Rate	Blinking Mode/Rate	Behavior in WAIT/STO3 modes	LCD Power Input
1	1.8-V	External 32.768 kHz	3-V	128	30 Hz	None	WAIT: on STOP3: on	Power via V_{LCD}
2	3.6-V	Internal 100 kHz	3-V	99	80 Hz	Individual segment 0.5 Hz	WAIT: on STOP3: off	Power via V_{DD}
3	3.6-V	Internal 18886 kHz	5-V	160	60 Hz	Individual segment 2.0 Hz	WAIT: off STOP3: on	Power via V_{DD}
4	1.8-V	External 32.768 kHz	5-V	123	30 Hz	all segment 2.0 Hz	WAIT: off STOP3: off	Power via V_{LCD}

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

9.5.2.1 Initialization Example 1

Example 1 LCD setup requirements are reiterated in the following table:

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
1	1.8-V	External 32.768 kHz	3-V	128	30 Hz	No Blinking	WAIT: on STOP3: on	Power via V_{LCD}

Table 9-22 lists the required setup values required to initialize the LCD as specified by Example 1:

Table 9-22. Initialization Register Values for Example 1

Register	bit or bit field	Binary Value	Comment
LCDCLKS 00000000	SOURCE	0	Selects the external clock reference as the LCD clock input External clock reference = 0; Bus clock = 1
	DIV16	0	Adjusts the LCD clock input (see table 9-12)
	CLKADJ[5:0]	000000	Adjusts the LCD clock input (see table 9-12)
LCDSUPPLY 10XXXX10	LCDCPEN	1	Enable the charge pump
	LCDCPMS	0	For 3-V LCD glass, select doubler mode; Doubler mode = 0; Tripler mode = 1
	HDRVBUF	X	High drive buffer
	CPCADJ[1:0]	XX	Configure LCD charge pump clock source
	BBYPASS	X	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 9-16). For 3-V glass, the nominal value of V_{LCD} should be 1.5-V.
LCDCR1 XXXXXX00	LCDIEN	X	LCD Interrupt Enable
	LCDWAI	0	LCD is “on” in WAIT mode
	LCDSTP3	0	LCD is “on” in STOP3 mode
LCDCR0 0X100X00	LCLK[2:0]	100	For 1/4 duty cycle, select closest value to the desired 30 Hz LCD frame frequency (see table 9-13)
	LPWAVE	X	Low power waveform
	DUTY[1:0]	11	For 128 segments (4x32), select 1/4 duty cycle (see table 9-11)
LCDBCTL 0XXXXXXX	BLKMODE	X	N/A; Blink Segments = 0; Blink All = 1
	BRATE[2:0]	XXX	N/A
FPENR[5:0]	FPENR0 FPENR1 FPENR2 FPENR3 FPENR4 FPENR5	11111111 11111111 11111111 11111111 00000000 XXXXXXX0	Only 32 Frontplanes need to be enabled.

9.5.2.2 Initialization Example 2

Example 2 LCD setup requirements are reiterated in the following table:

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 100 kHz	3-V	99	80 Hz	Individual segment 0.5 Hz	WAIT: on STOP3: off	Power via V_{DD}

Table 9-23 lists the required setup values required to initialize the LCD as specified by Example 2:

Table 9-23. Initialization Register Values for Example 2

Register	Bit/bit field	Binary Value	Comment
LCDCLKS 10000010	SOURCE	1	Selects the bus clock as the LCD clock input External clock reference = 0; Bus clock = 1
	DIV16	0	Adjusts the LCD clock input (see table 9-12)
	CLKADJ[5:0]	000010	Adjusts the LCD clock input (see table 9-12)
LCDSUPPLY 1XXXXX01	LCDCPEN	1	Enable the charge pump
	LCDCPMS	X	Don't care since power is from internal V_{DD} Doubler mode = 0; Tripler mode = 1
	HDRVBUF	X	High drive buffer
	CPCADJ[1:0]	XX	Configure LCD charge pump clock source
	BBYPASS	X	Buffer Bypass; Buffer mode = 0; Unbuffered mode = 1
	VSUPPLY[1:0]	01	Power LCD via V_{DD} internal power (see table 9-16). When VSUPPLY[1:0] = 01, V_{LL3} is generated from V_{DD} .
LCDCR1 XXXXXX01	LCDWAI	0	LCD is "on" in WAIT mode
	LCDSTP3	1	LCD is "off" in STOP3 mode
LCDCR0 0X011X11	LCLK[2:0]	011	For 1/3 duty cycle, select closest value to the desired 80 Hz LCD frame frequency (see table 9-13). Note the LCD base frequency - 256.2 Hz
	LPWAVE	X	Low power waveform
	DUTY[1:0]	10	For 99 segments (3x33), select 1/3 duty cycle (see table 9-11)

Table 9-23. Initialization Register Values for Example 2 (continued)

Register	Bit/bit field	Binary Value	Comment
LCDBCTL 0XXX0100	BLKMODE	0	Blink individual segments; Blink Segments = 0; Blink All = 1
	BRATE[2:0]	100	Using the LCD base frequency for the selected LCD frame frequency, select 0.5 Hz blink frequency (see table 9-15).
FPENR[5:0]	FPENR0 FPENR1 FPENR2 FPENR3 FPENR4 FPENR5	11111111 11111111 11111111 11111111 00000001 XXXXXXX0	Only 33 Frontplanes need to be enabled. Optionally, if required, 1/4 duty cycle could be used. This option would only require 25 frontplane pins to be enabled.

9.5.2.3 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	3.6-V	Internal 18886 kHz	5-V	160	60 Hz	Individual segment 2.0 Hz	WAIT: off STOP3: on	Power via V_{DD}

Table 9-24 lists the required setup values required to initialize the LCD as specified by Example 3:

Table 9-24. Initialization Register Values for Example 3

Register	Bit/bit field	Binary Value	Comment
LCDCLKS 11100011	SOURCE	1	Selects the bus clock as the LCD clock input External clock reference = 0; Bus clock = 1
	DIV16	1	Adjusts the LCD clock input (see table 9-12)
	CLKADJ[5:0]	100011	Adjusts the LCD clock input (see table 9-12)
LCDSUPPLY 1XXXXX00	LCDCPEN	1	Enable the charge pump
	LCDCPMS	X	Don't care since power is from internal V_{DD} Doubler mode = 0; Tripler mode = 1
	HDRVBUF	X	High drive buffer
	CPCADJ[1:0]	XX	Configure LCD charge pump clock source
	BBYPASS	X	Buffer Bypass; Buffer mode = 0; Unbuffered mode = 1
	VSUPPLY[1:0]	00	Power LCD via V_{DD} internal power (see table 9-16). When VSUPPLY[1:0] = 00, V_{LL2} is generated from V_{DD} .
LCDCR1 XXXXXX10	LCDWAI	1	LCD is "off" in WAIT mode
	LCDSTP3	0	LCD is "on" in STOP3 mode
LCDCR0 0X011X00	LCLK[2:0]	011	For 1/4 duty cycle, select closest value to the desired 60 Hz LCD frame frequency (see table 9-13). Note the LCD base frequency - 256.2 Hz
	LPWAVE	X	Low power waveform
	DUTY[1:0]	11	For 160 segments (4x40), select 1/4 duty cycle (see table 9-11)
LCDBCTL 0XXX0010	BLKMODE	0	Blink individual segments; Blink Segments = 0; Blink All = 1
	BRATE[2:0]	010	Using the LCD base frequency for the selected LCD frame frequency, select 2.0 Hz blink frequency (see table 9-15).
FPENR[5:0]	FPENR0 FPENR1 FPENR2 FPENR3 FPENR4 FPENR5	11111111 11111111 11111111 11111111 11111111 XXXXXXX0	40 LCD frontplanes need to be enabled.

9.5.2.4 Initialization Example 4

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
4	1.8-V	External 32.768 kHz	5-V	123	30 Hz	all segment 2.0 Hz	WAIT: off STOP3: off	Power via V_{LCD}

Table 9-25 lists the required setup values required to initialize the LCD as specified by Example 4:

Table 9-25. Initialization Register Values for Example 4

Register	Bit/bit field	Binary Value	Comment
LCDCLKS 00000000	SOURCE	0	Selects the external clock reference as the LCD clock input External clock reference = 0; Bus clock = 1
	DIV16	0	Adjusts the LCD clock input (see table 9-12)
	CLKADJ[5:0]	000000	Adjusts the LCD clock input (see table 9-12)
LCDSUPPLY 1100XX10	LCDCPEN	1	Enable the charge pump
	LCDCPMS	1	For 3-V LCD glass, select tripler mode Doubler mode = 0; Tripler mode = 1
	HDRVBUF	X	High drive buffer
	CPCADJ[1:0]	00	Configure LCD charge pump clock source
	BBYPASS	X	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 9-16). For 5-V glass, the nominal value of V_{LCD} should be 1.67-V.
LCDCR1 XXXXXX00	LCDWAI	0	LCD is “off” in WAIT mode
	LCDSTP3	0	LCD is “off” in STOP3 mode
LCDCR0 0X010X11	LCLK[2:0]	010	For 1/3 duty cycle, select the closest value to the desired 30 Hz LCD frame frequency (see table 9-13). Note the LCD base frequency - 128.1 Hz
	LPWAVE	X	Low power waveform
	DUTY[1:0]	10	For 123 segments (3x41), select 1/3 duty cycle (see table 9-11)
LCDBCTL 0XXX1001	BLKMODE	1	Blink all segments; Blink Segments = 0; Blink All = 1
	BRATE[2:0]	001	Using the LCD base frequency for the selected LCD frame frequency, select 2.0 Hz blink frequency (see table 9-15).
FPENR[5:0]	FPENR0 FPENR1 FPENR2 FPENR3 FPENR4 FPENR5	11111111 11111111 11111111 11111111 11111111 XXXXXXX1	All 41 LCD frontplanes need to be enabled.

9.6 Application Information

Figure 9-18 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 very high level illustration of the LCD module showing the module's functional hierarchy including initialization and runtime control.

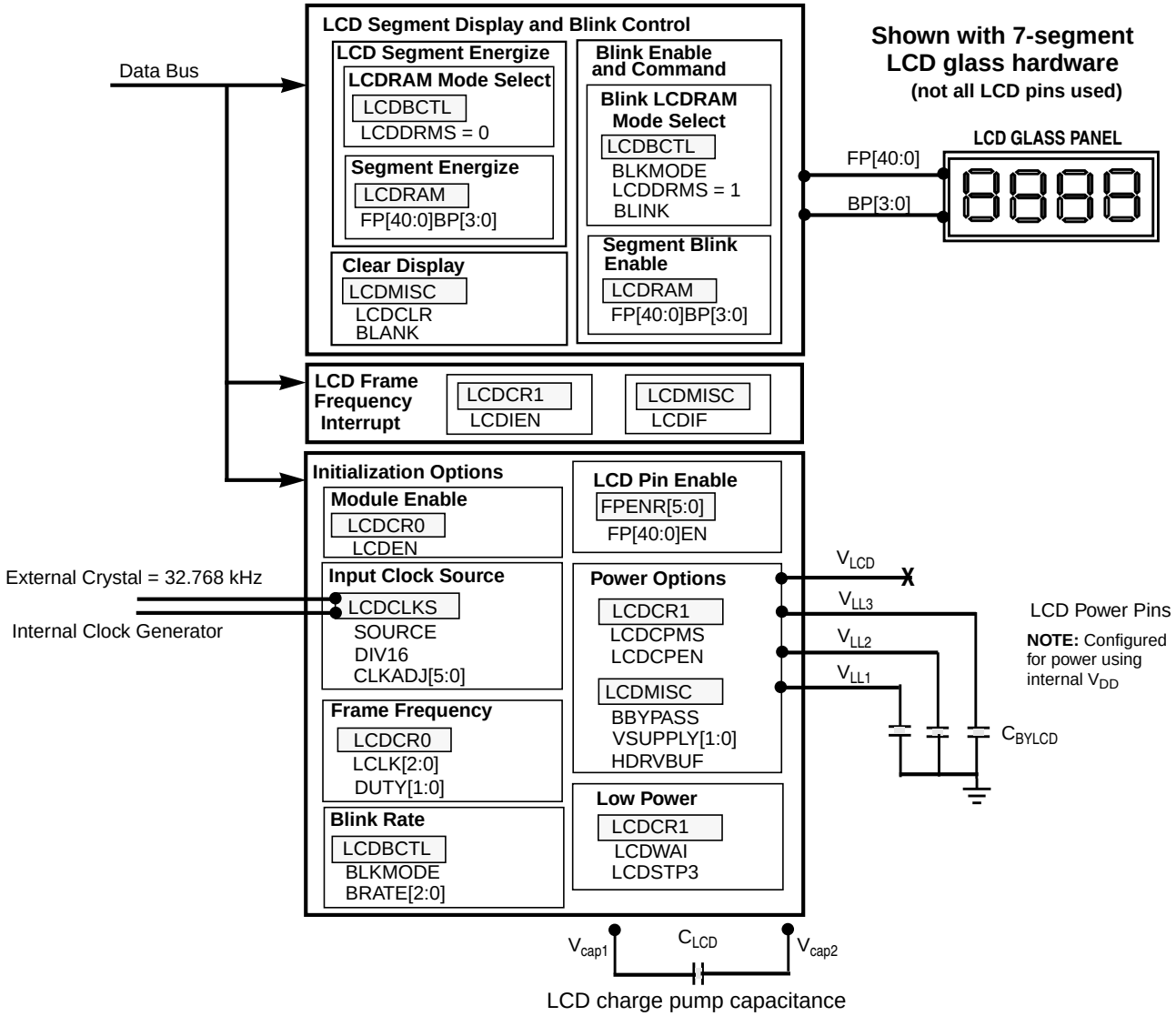
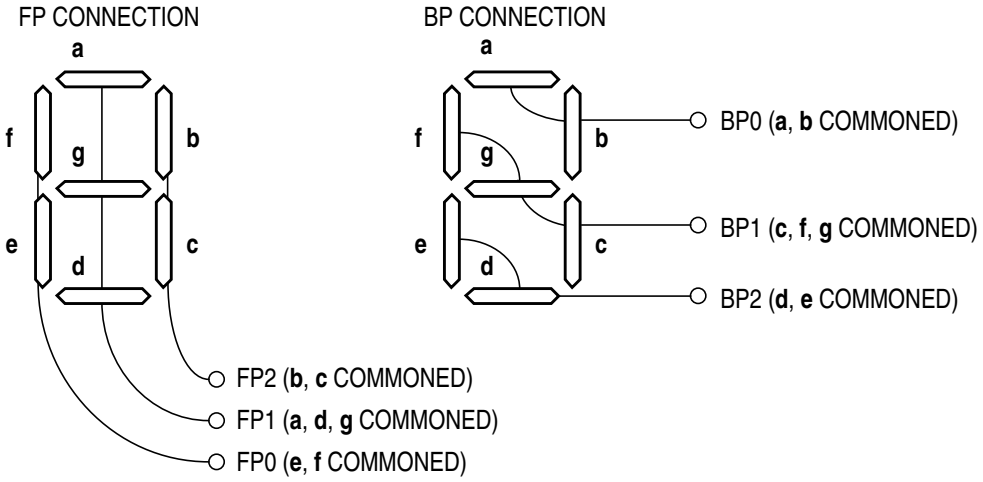


Figure 9-18. LCD Programmer's Model Diagram

9.6.1 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 LPWAVE = 0 and 1/3 duty cycle LCD implementation. The example use backplane pins (BP0, BP1, BP2) and frontplace pins (FP0, FP1, and FP2). LCDRAM contents and output waveforms are also shown. Output waveforms are illustrated in Figure 9-19 and Figure 9-20.



The segment assignments for each bit in the data registers are:

	F1B3	F1B2	F1B1	F1B0	F0B3	F0B2	F0B1	F0B0
LDAT1 0x0052	—	d	g	a	—	e	f	—
	← FP1 →				← FP0 →			

	F3B3	F3B2	F3B1	F3B0	F2B3	F2B2	F2B1	F2B0
LDAT2 0x0053	—	—	—	—	—	—	c	b
					← FP2 →			

To display the character “4”: LDAT1 = X010X01X, LDAT2 = XXXXXX11

LDAT1 0x0052	X	0	1	0	X	0	1	X
LDAT2 0x0053	X	X	X	X	X	X	1	1

X = don’t care

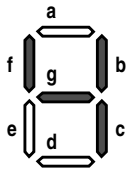


Figure 9-19. Waveform Output from LCDRAM Registers

9.6.1.1 LCD Module Waveforms

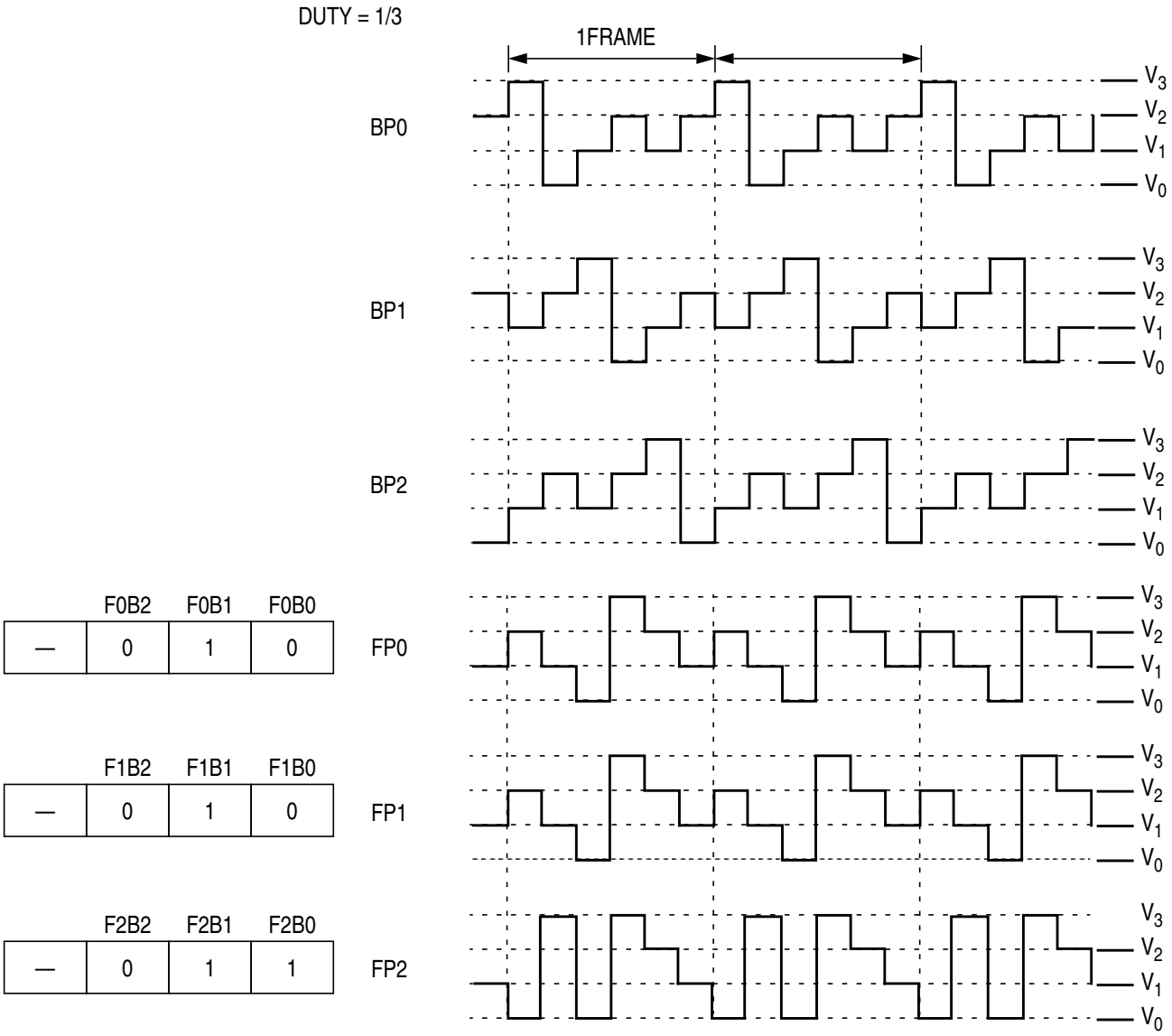


Figure 9-20. LCD Waveforms (LPWAVE = 0)

9.6.1.2 Segment On Driving Waveform

The voltage waveform across the “f” segment of the LCD (between BP1 and FP0) is illustrated in Figure 9-21. As shown in the waveform, the voltage level reaches the value V_3 therefore the segment will be on.

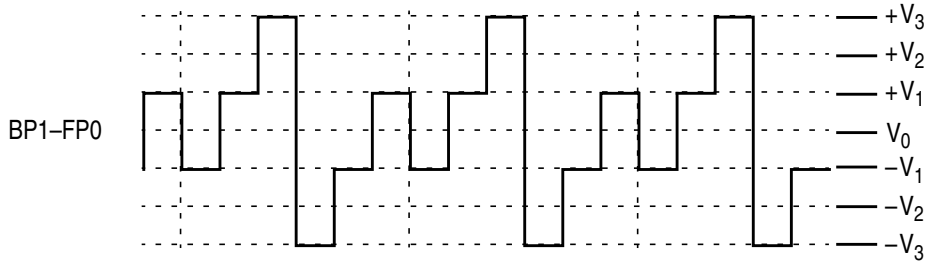


Figure 9-21. “f” Segment Voltage Waveform

9.6.1.3 Segment Off Driving Waveform

The voltage waveform across the “e” segment of the LCD (between BP2 and FP0) is illustrated in Figure 9-22. As shown in the waveform, the voltage does not reach the voltage V_3 threshold therefore the segment will be off.

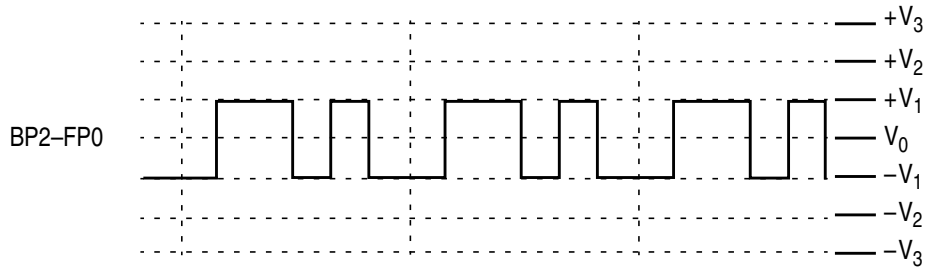


Figure 9-22. “e” Segment Voltage Waveform

9.6.2 LCD Contrast Control

Contrast control for the LCD module is achieved when 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, the LCD threshold voltage is 5-V. By increasing the value of the LCD threshold voltage, the energized segments on the LCD glass will become more opaque. Decreasing the value of the LCD threshold 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 9-23 shows two circuits that could be used to implement contract control.

NOTE:
Contrast control configuration when LCD is powered using external V_{LCD}

This is the recommended configuration for contrast control.

V_{LCD} specified between 0.9 and 1.8 volts.

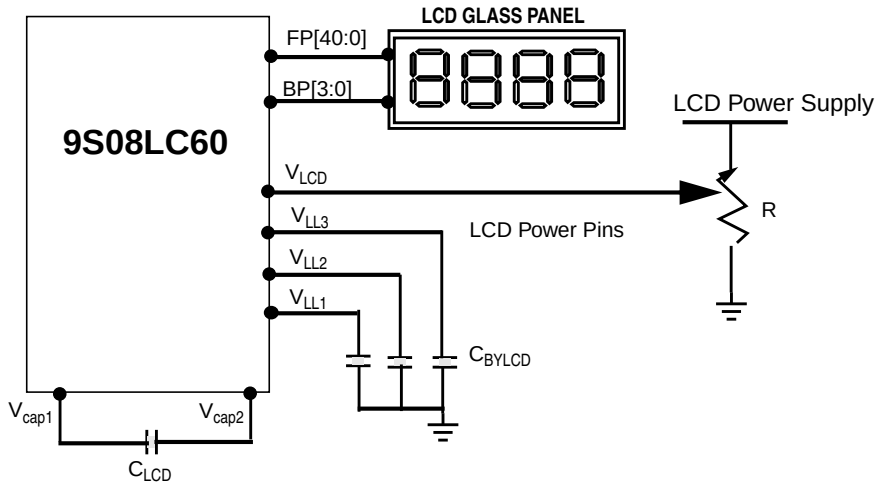


Figure 9-23. Power Connections for Contrast Control

9.6.3 LCD Power Consumption

The following tables show relative power consumption from V_{DD} for the different modes available for the LCD module.

Table 26. Relative Power Consumption, 5 V

Mode	Typical Voltage	LCD Display Voltage Range	V_{DD} Supply Range	V_{DD} Current Consumption Level
V_{LL2} connect to V_{DD}	V_{DD} = 3.333 V	V_{DD} = 3 V ~ 3.6 V	3 V ~ 3.6 V	low
Tripler Buffered Mode	V_{LCD} = 1.6667 V	V_{LCD} = 1.5 V ~ 1.8 V	2.4 V ~ 3.6 V	high
Tripler Bypassed Mode	V_{LCD} = 1.6667 V	V_{LCD} = 1.5 V ~ 1.8 V	1.8 V ~ 3.6 V	lowest

Note: Current consumption data based on using the external 32-kHz oscillator with LCD configured using the low-power wave forms option, a 1/4 duty, and a 32-Hz frame frequency. $C_{LCD} = C_{BYLCD}$ = 100 nF; 160 segment 2000 pF LCD panel.

Table 27. Relative Power Consumption, 3 V

Mode	Typical Voltage	LCD Display Voltage Range	V_{DD} Supply Range	V_{DD} Current Consumption Level
V_{LL2} connect to V_{DD}	V_{DD} = 2 V	V_{DD} = 1.8 V ~ 2.2 V	1.8 V ~ 2.2 V	high
Tripler Buffered Mode	V_{LCD} = 1 V	V_{LCD} = 0.9 V ~ 1.1 V	2.0 V ~ 3.6 V	highest
Tripler Bypassed Mode	V_{LCD} = 1 V	V_{LCD} = 0.9 V ~ 1.1 V	1.8 V ~ 3.6 V	lowest
V_{LL3} connect to V_{DD}	V_{DD} = 3 V	V_{DD} = 2.7 V ~ 3.3 V	2.7 V ~ 3.3 V	low
Doubler Buffered Mode	V_{LCD} = 1.5 V	V_{LCD} = 1.35 V ~ 1.65 V	2.4 V ~ 3.6 V	highest

Note: Current consumption data based on using the external 32-kHz oscillator with LCD configured using the low-power wave forms option, a 1/4 duty, and a 32-Hz frame frequency. $C_{LCD} = C_{BYLCD}$ = 100 nF; 160 segment 2000 pF LCD panel.

Chapter 10
Internal Clock Generator (S08ICGV4)

10.1 Introduction

The ICG module is used to generate the system clocks for the MC9S08LC60/36/20 MCU. Figure 10-1 shows the clock distribution for the MC9S08LC60/36/20 MCU. Electrical parametric data for the ICG may be found in Appendix.

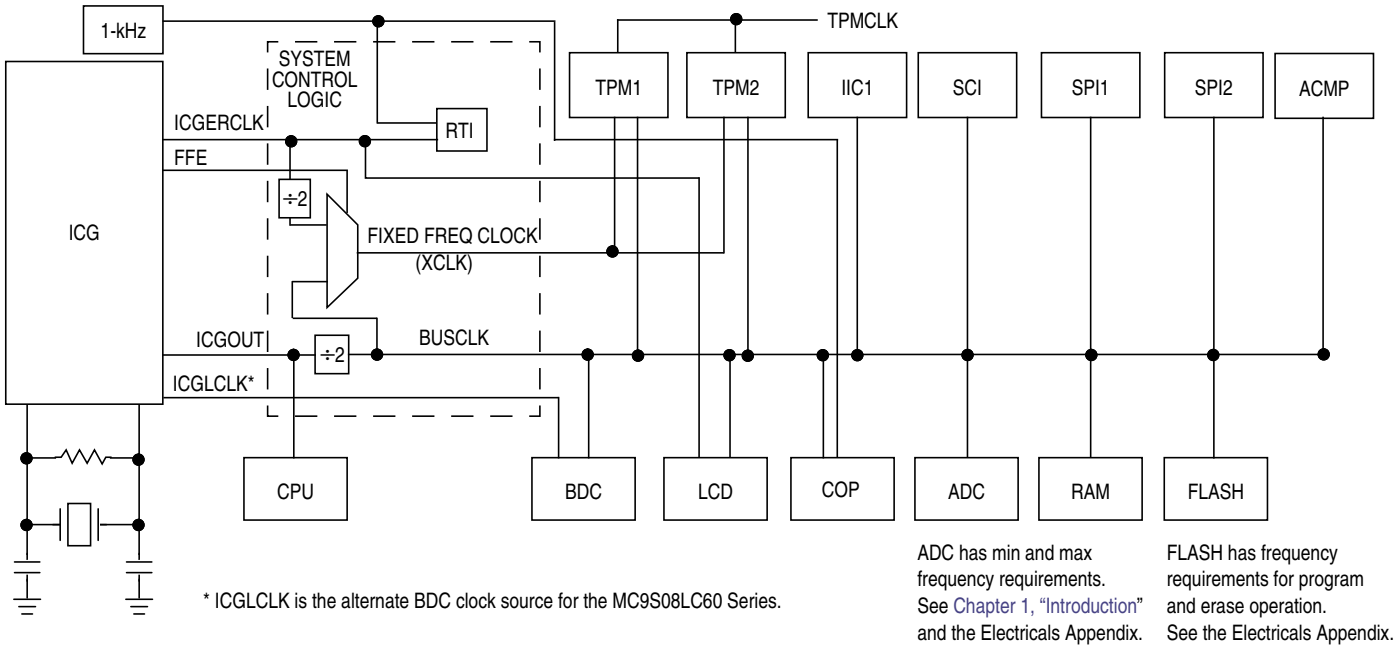
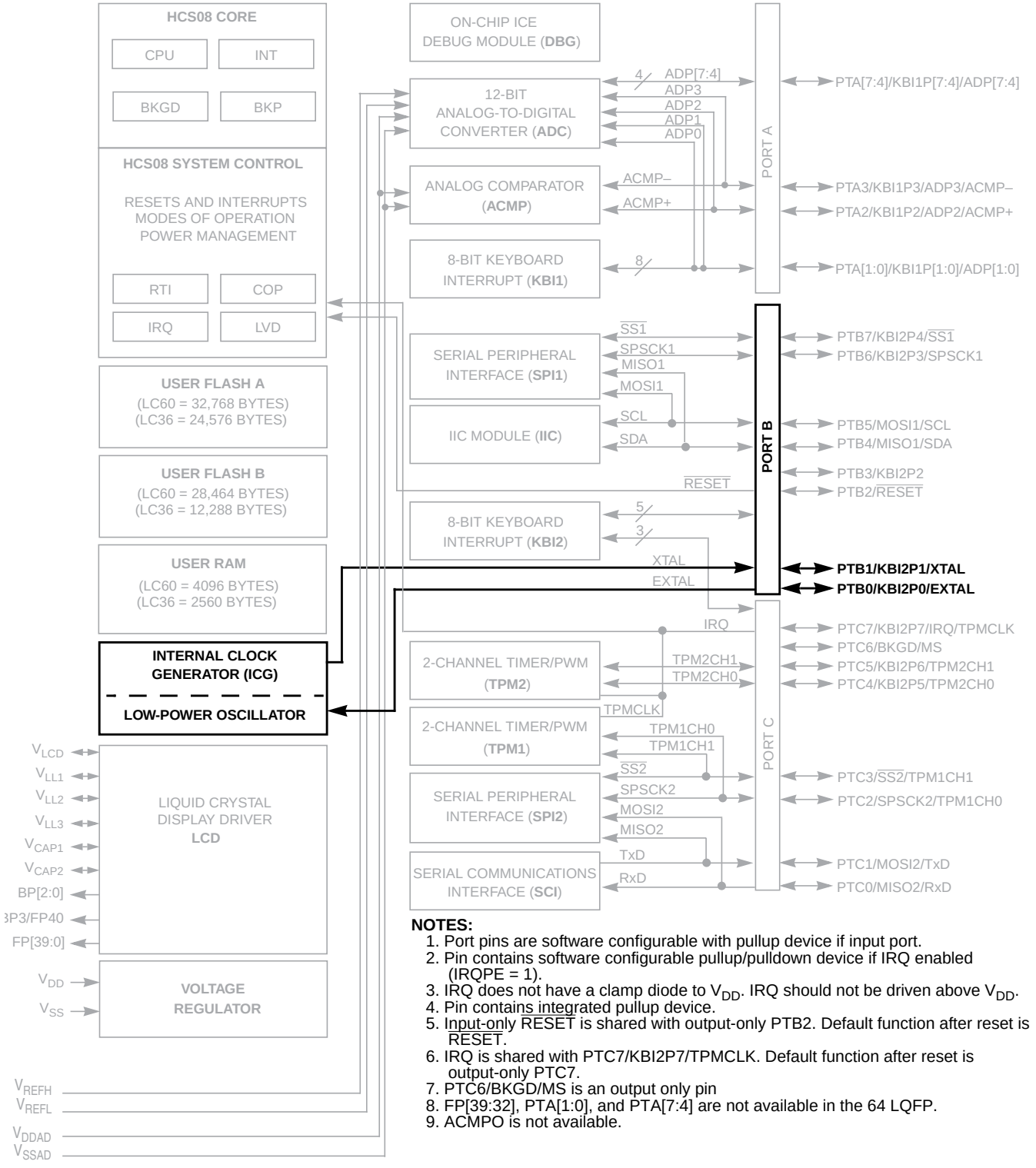


Figure 10-1. System Clock Distribution Diagram

NOTE

Freescale Semiconductor programs a factory trim value for ICGTRM into the FLASH location $FFBE (NVICGTRM). Leaving this address for the ICGTRM value also allows debugger and programmer vendors to perform a manual trim operation and store the resultant ICGTRM value into NVICGTRM for users to access at a later time. The value in NVICGTRM is not automatically loaded and therefore must be copied into ICGTRM by user code.

Figure 10-2 shows the MC9S08LC60 Series block diagram with the ICG highlighted.



NOTES:
1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 10-2. MC9S08LC60 Series Block Diagram Highlighting the ICGBlock and Pins

10.2 Introduction

The ICG provides multiple options for clock sources. This offers a user great flexibility when making choices between cost, precision, current draw, and performance. As seen in Figure 10-3, the ICG consists of four functional blocks. Each of these is briefly described here and then in more detail in a later section.

- **Oscillator block** — The oscillator block provides means for connecting an external crystal or resonator. Two frequency ranges are software selectable to allow optimal startup and stability. Alternatively, the oscillator block can be used to route an external square wave to the system clock. External sources can provide a very precise clock source. The oscillator is capable of being configured for low power mode or high amplitude mode as selected by HGO.
- **Internal reference generator** — The internal reference generator consists of two controlled clock sources. One is designed to be approximately 8 MHz and can be selected as a local clock for the background debug controller. The other internal reference clock source is typically 243 kHz and can be trimmed for finer accuracy via software when a precise timed event is input to the MCU. This provides a highly reliable, low-cost clock source.
- **Frequency-locked loop** — A frequency-locked loop (FLL) stage takes either the internal or external clock source and multiplies it to a higher frequency. Status bits provide information when the circuit has achieved lock and when it falls out of lock. Additionally, this block can monitor the external reference clock and signals whether the clock is valid or not.
- **Clock select block** — The clock select block provides several switch options for connecting different clock sources to the system clock tree. ICGDCLK is the multiplied clock frequency out of the FLL, ICGERCLK is the reference clock frequency from the crystal or external clock source, and FFE (fixed frequency enable) is a control signal used to control the system fixed frequency clock (XCLK). ICGLCLK is the clock source for the background debug controller (BDC).

10.2.1 Features

The module is intended to be very user friendly with many of the features occurring automatically without user intervention. To quickly configure the module, go to Section 10.6, “Initialization/Application Information” and pick an example that best suits the application needs.

Features of the ICG and clock distribution system:

- Several options for the primary clock source allow a wide range of cost, frequency, and precision choices:
  - 32 kHz–100 kHz crystal or resonator
  - 1 MHz–16 MHz crystal or resonator
  - External clock
  - Internal reference generator
- Defaults to self-clocked mode to minimize startup delays
- Frequency-locked loop (FLL) generates 8 MHz to 40 MHz (for bus rates up to 20 MHz)
  - Uses external or internal clock as reference frequency
- Automatic lockout of non-running clock sources
- Reset or interrupt on loss of clock or loss of FLL lock

- Digitally-controlled oscillator (DCO) preserves previous frequency settings, allowing fast frequency lock when recovering from stop3 mode
- DCO will maintain operating frequency during a loss or removal of reference clock
- Post-FLL divider selects 1 of 8 bus rate divisors (/1 through /128)
- Separate self-clocked source for real-time interrupt
- Trimmable internal clock source supports SCI communications without additional external components
- Automatic FLL engagement after lock is acquired
- External oscillator selectable for low power or high gain

10.2.2 Modes of Operation

This is a high-level description only. Detailed descriptions of operating modes are contained in Section 10.5, "Functional Description."

- Mode 1 — Off

  The output clock, ICGOUT, is static. This mode may be entered when the STOP instruction is executed.
- Mode 2 — Self-clocked (SCM)

  Default mode of operation that is entered immediately after reset. The ICG's FLL is open loop and the digitally controlled oscillator (DCO) is free running at a frequency set by the filter bits.
- Mode 3 — FLL engaged internal (FEI)

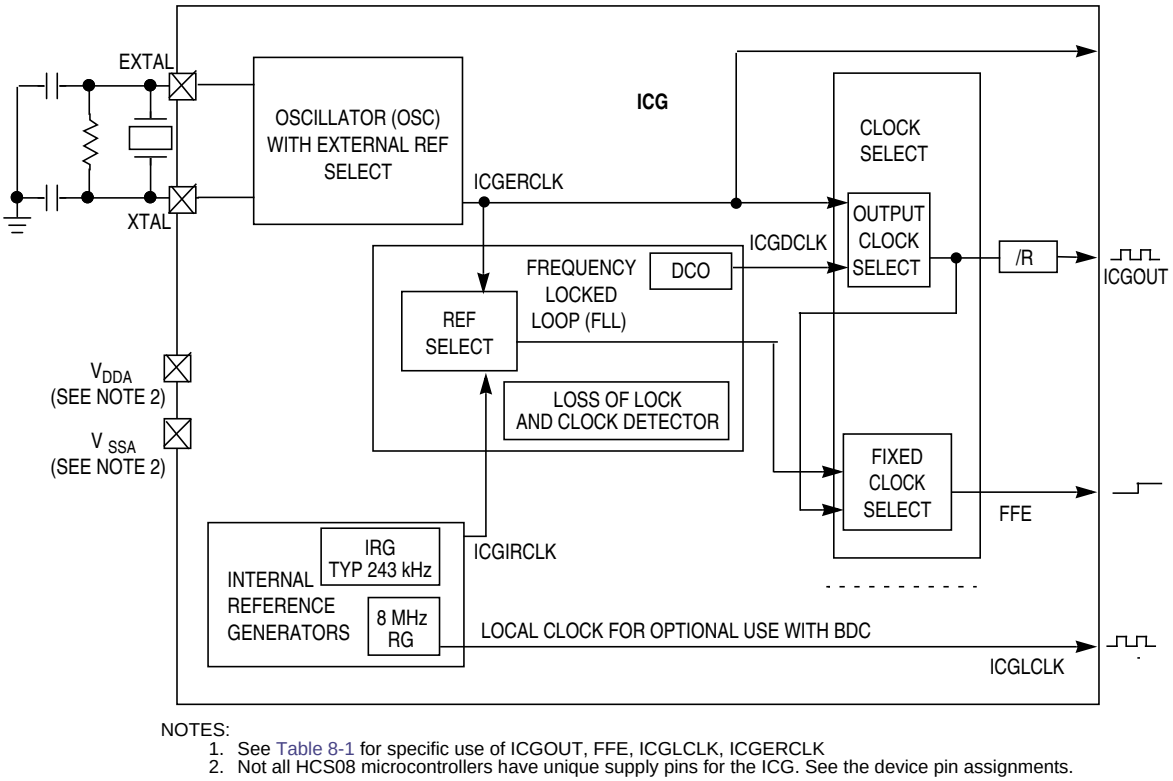
  In this mode, the ICG's FLL is used to create frequencies that are programmable multiples of the internal reference clock.
  - FLL engaged internal unlocked is a transition state that occurs while the FLL is attempting to lock. The FLL DCO frequency is off target and the FLL is adjusting the DCO to match the target frequency.
  - FLL engaged internal locked is a state that occurs when the FLL detects that the DCO is locked to a multiple of the internal reference.
- Mode 4 — FLL bypassed external (FBE)

  In this mode, the ICG is configured to bypass the FLL and use an external clock as the clock source.
- Mode 5 — FLL engaged external (FEE)

  The ICG's FLL is used to generate frequencies that are programmable multiples of the external clock reference.
  - FLL engaged external unlocked is a transition state that occurs while the FLL is attempting to lock. The FLL DCO frequency is off target and the FLL is adjusting the DCO to match the target frequency.
  - FLL engaged external locked is a state which occurs when the FLL detects that the DCO is locked to a multiple of the internal reference.

10.2.3 Block Diagram

Figure 10-3 is a top-level diagram that shows the functional organization of the internal clock generation (ICG) module. This section includes a general description and a feature list.



NOTES:
1. See Table 8-1 for specific use of ICGOUT, FFE, ICGLCLK, ICGERCLK
2. Not all HCS08 microcontrollers have unique supply pins for the ICG. See the device pin assignments.

Figure 10-3. ICG Block Diagram

10.3 External Signal Description

The oscillator pins are used to provide an external clock source for the MCU. The oscillator pins are gain controlled in low-power mode (default). Oscillator amplitudes in low-power mode are limited to approximately 1 V, peak-to-peak.

10.3.1 EXTAL — External Reference Clock / Oscillator Input

If upon the first write to ICGC1, either the FEE mode or FBE mode is selected, this pin functions as either the external clock input or the input of the oscillator circuit as determined by REFS. If upon the first write to ICGC1, either the FEI mode or SCM mode is selected, this pin is not used by the ICG.

10.3.2 XTAL — Oscillator Output

If upon the first write to ICGC1, either the FEE mode or FBE mode is selected, this pin functions as the output of the oscillator circuit. If upon the first write to ICGC1, either the FEI mode or SCM mode is

selected, this pin is not used by the ICG. The oscillator is capable of being configured to provide a higher amplitude output for improved noise immunity. This mode of operation is selected by HGO = 1.

10.3.3 External Clock Connections

If an external clock is used, then the pins are connected as shown Figure 10-4.

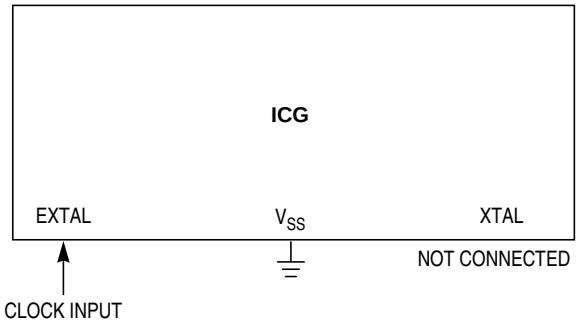


Figure 10-4. External Clock Connections

10.3.4 External Crystal/Resonator Connections

If an external crystal/resonator frequency reference is used, then the pins are connected as shown in Figure 10-5. Recommended component values are listed in the Electrical Characteristics chapter.

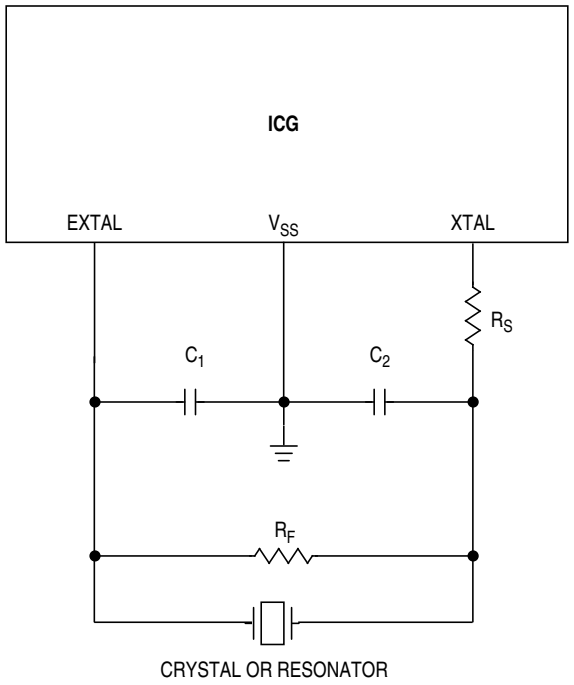


Figure 10-5. External Frequency Reference Connection

10.4 Register Definition

Refer to the direct-page register summary in the Memory chapter of this data sheet for the absolute address assignments for all ICG 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.

10.4.1 ICG Control Register 1 (ICGC1)

	7	6	5	4	3	2	1	0
R	HGO[1]	RANGE	REFS	CLKS		OSCSTEN	LOCD	0
W								
Reset	0	1	0	0	0	1	0	0

= Unimplemented or Reserved

Figure 10-6. ICG Control Register 1 (ICGC1)

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

Table 10-1. ICGC1 Register Field Descriptions

Field	Description
7 HGO	**High Gain Oscillator Select** — The HGO bit is used to select between low power operation and high gain operation for improved noise immunity. This bit is write-once after reset. 0 Oscillator configured for low power operation. 1 Oscillator configured for high gain operation.
6 RANGE	**Frequency Range Select** — The RANGE bit controls the oscillator, reference divider, and FLL loop prescaler multiplication factor (P). It selects one of two reference frequency ranges for the ICG. The RANGE bit is write-once after a reset. The RANGE bit only has an effect in FLL engaged external and FLL bypassed external modes. 0 Oscillator configured for low frequency range. FLL loop prescale factor P is 64. 1 Oscillator configured for high frequency range. FLL loop prescale factor P is 1.
5 REFS	**External Reference Select** — The REFS bit controls the external reference clock source for ICGERCLK. The REFS bit is write-once after a reset. 0 External clock requested. 1 Oscillator using crystal or resonator requested.
4:3 CLKS	**Clock Mode Select** — The CLKS bits control the clock mode as described below. If FLL bypassed external is requested, it will not be selected until ERCS = 1. If the ICG enters off mode, the CLKS bits will remain unchanged. Writes to the CLKS bits will not take effect if a previous write is not complete. 00 Self-clocked 01 FLL engaged, internal reference 10 FLL bypassed, external reference 11 FLL engaged, external reference The CLKS bits are writable at any time, unless the first write after a reset was CLKS = 0X, the CLKS bits cannot be written to 1X until after the next reset (because the EXTAL pin was not reserved).

Table 10-1. ICGC1 Register Field Descriptions (continued)

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

10.4.2 ICG Control Register 2 (ICGC2)

	7	6	5	4	3	2	1	0
R W	LOLRE	MFD			LOCRE	RFD		
Reset	0	0	0	0	0	0	0	0

Figure 10-7. ICG Control Register 2 (ICGC2)

Table 10-2. ICGC2 Register Field Descriptions

Field	Description
7 LOLRE	**Loss of Lock Reset Enable** — The LOLRE bit determines what type of request is made by the ICG following a loss of lock indication. The LOLRE bit only has an effect when LOLS is set. 0 Generate an interrupt request on loss of lock. 1 Generate a reset request on loss of lock.
6:4 MFD	**Multiplication Factor** — The MFD bits control the programmable multiplication factor in the FLL loop. The value specified by the MFD bits establishes the multiplication factor (N) applied to the reference frequency. Writes to the MFD bits will not take effect if a previous write is not complete. Select a low enough value for N such that $f_{ICGDCLK}$ does not exceed its maximum specified value. 000 Multiplication factor = 4 001 Multiplication factor = 6 010 Multiplication factor = 8 011 Multiplication factor = 10 100 Multiplication factor = 12 101 Multiplication factor = 14 110 Multiplication factor = 16 111 Multiplication factor = 18
3 LOCRE	**Loss of Clock Reset Enable** — The LOCRE bit determines how the system manages a loss of clock condition. 0 Generate an interrupt request on loss of clock. 1 Generate a reset request on loss of clock.
2:0 RFD	**Reduced Frequency Divider** — The RFD bits control the value of the divider following the clock select circuitry. The value specified by the RFD bits establishes the division factor (R) applied to the selected output clock source. Writes to the RFD bits will not take effect if a previous write is not complete. 000 Division factor = 1 001 Division factor = 2 010 Division factor = 4 011 Division factor = 8 100 Division factor = 16 101 Division factor = 32 110 Division factor = 64 111 Division factor = 128

10.4.3 ICG Status Register 1 (ICGS1)

	7	6	5	4	3	2	1	0
R	CLKST		REFST	LOLS	LOCK	LOCS	ERCS	ICGIF
W								1
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-8. ICG Status Register 1 (ICGS1)

Table 10-3. ICGS1 Register Field Descriptions

Field	Description
7:6 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 Self-clocked 01 FLL engaged, internal reference 10 FLL bypassed, external reference 11 FLL engaged, external reference
5 REFST	**Reference Clock Status** — The REFST bit indicates which clock reference is currently selected by the Reference Select circuit. 0 External Clock selected. 1 Crystal/Resonator selected.
4 LOLS	**FLL Loss of Lock Status** — The LOLS bit is a sticky indication of FLL lock status. 0 FLL has not unexpectedly lost lock since LOLS was last cleared. 1 FLL has unexpectedly lost lock since LOLS was last cleared, LOLRE determines action taken.FLL has unexpectedly lost lock since LOLS was last cleared, LOLRE determines action taken.
3 LOCK	**FLL Lock Status** — The LOCK bit indicates whether the FLL has acquired lock. The LOCK bit is cleared in off, self-clocked, and FLL bypassed modes. 0 FLL is currently unlocked. 1 FLL is currently locked.
2 LOCS	**Loss Of Clock Status** — The LOCS bit is an indication of ICG loss of clock status. 0 ICG has not lost clock since LOCS was last cleared. 1 ICG has lost clock since LOCS was last cleared, LOCRE determines action taken.
1 ERCS	**External Reference Clock Status** — The ERCS bit is an indication of whether or not the external reference clock (ICGERCLK) meets the minimum frequency requirement. 0 External reference clock is not stable, frequency requirement is not met. 1 External reference clock is stable, frequency requirement is met.
0 ICGIF	**ICG Interrupt Flag** — The ICGIF read/write flag is set when an ICG interrupt request is pending. It is cleared by a reset or by reading the ICG status register when ICGIF is set and then writing a logic 1 to ICGIF. If another ICG interrupt occurs before the clearing sequence is complete, the sequence is reset so ICGIF would remain set after the clear sequence was completed for the earlier interrupt. Writing a logic 0 to ICGIF has no effect. 0 No ICG interrupt request is pending. 1 An ICG interrupt request is pending.

10.4.4 ICG Status Register 2 (ICGS2)

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

= Unimplemented or Reserved

Figure 10-9. ICG Status Register 2 (ICGS2)

Table 10-4. ICGS2 Register Field Descriptions

Field	Description
0 DCOS	**DCO Clock Stable** — The DCOS bit is set when the DCO clock (ICG2DCLK) is stable, meaning the count error has not changed by more than n_{unlock} for two consecutive samples and the DCO clock is not static. This bit is used when exiting off state if CLKS = X1 to determine when to switch to the requested clock mode. It is also used in self-clocked mode to determine when to start monitoring the DCO clock. This bit is cleared upon entering the off state. 0 DCO clock is unstable. 1 DCO clock is stable.

10.4.5 ICG Filter Registers (ICGFLTU, ICGFLTL)

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

= Unimplemented or Reserved

Figure 10-10. ICG Upper Filter Register (ICGFLTU)

Table 10-5. ICGFLTU Register Field Descriptions

Field	Description
3:0 FLT	**Filter Value** — The FLT bits indicate the current filter value, which controls the DCO frequency. The FLT bits are read only except when the CLKS bits are programmed to self-clocked mode (CLKS = 00). In self-clocked mode, any write to ICGFLTU updates the current 12-bit filter value. Writes to the ICGFLTU register will not affect FLT if a previous latch sequence is not complete.

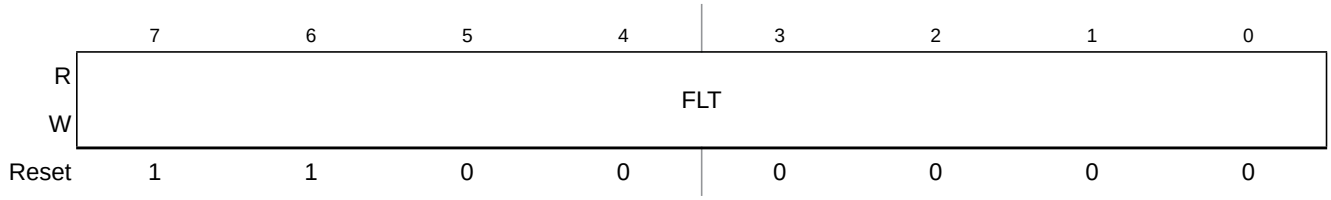


Figure 10-11. ICG Lower Filter Register (ICGFLTL)

Table 10-6. ICGFLTL Register Field Descriptions

Field	Description
7:0 FLT	**Filter Value** — The FLT bits indicate the current filter value, which controls the DCO frequency. The FLT bits are read only except when the CLKS bits are programmed to self-clocked mode (CLKS = 00). In self-clocked mode, any write to ICGFLTU updates the current 12-bit filter value. Writes to the ICGFLTU register will not affect FLT if a previous latch sequence is not complete. The filter registers show the filter value (FLT).

10.4.6 ICG Trim Register (ICGTRM)

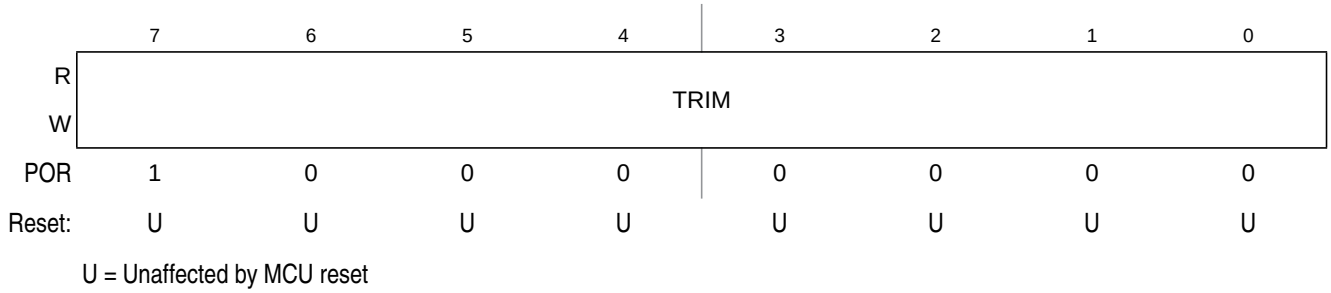


Figure 10-12. ICG Trim Register (ICGTRM)

Table 10-7. ICGTRM Register Field Descriptions

Field	Description
7 TRIM	**ICG Trim Setting** — The TRIM bits control the internal reference generator frequency. They allow a ±25% adjustment of the nominal (POR) period. The bit's effect on period is binary weighted (i.e., bit 1 will adjust twice as much as changing bit 0). Increasing the binary value in TRIM will increase the period and decreasing the value will decrease the period.

10.5 Functional Description

This section provides a functional description of each of the five operating modes of the ICG. Also discussed are the loss of clock and loss of lock errors and requirements for entry into each mode. The ICG is very flexible, and in some configurations, it is possible to exceed certain clock specifications. When using the FLL, configure the ICG so that the frequency of ICGDCLK does not exceed its maximum value to ensure proper MCU operation.

10.5.1 Off Mode (Off)

Normally when the CPU enters stop mode, the ICG will cease all clock activity and is in the off state. However there are two cases to consider when clock activity continues while the CPU is in stop mode,

10.5.1.1 BDM Active

When the BDM is enabled, the ICG continues activity as originally programmed. This allows access to memory and control registers via the BDC controller.

10.5.1.2 OSCSTEN Bit Set

When the oscillator is enabled in stop mode (OSCSTEN = 1), the individual clock generators are enabled but the clock feed to the rest of the MCU is turned off. This option is provided to avoid long oscillator startup times if necessary, or to run the RTI from the oscillator during stop3.

10.5.1.3 Stop/Off Mode Recovery

Upon the CPU exiting stop mode due to an interrupt, the previously set control bits are valid and the system clock feed resumes. If FEE is selected, the ICG will source the internal reference until the external clock is stable. If FBE is selected, the ICG will wait for the external clock to stabilize before enabling ICGOUT.

Upon the CPU exiting stop mode due to a reset, the previously set ICG control bits are ignored and the default reset values applied. Therefore the ICG will exit stop in SCM mode configured for an approximately 8 MHz DCO output (4 MHz bus clock) with trim value maintained. If using a crystal, 4096 clocks are detected prior to engaging ICGERCLK. This is incorporated in crystal start-up time.

10.5.2 Self-Clocked Mode (SCM)

Self-clocked mode (SCM) is the default mode of operation and is entered when any of the following conditions occur:

- After any reset.
- Exiting from off mode when CLKS does not equal 10. If CLKS = X1, the ICG enters this state temporarily until the DCO is stable (DCOS = 1).
- CLKS bits are written from X1 to 00.
- CLKS = 1X and ICGERCLK is not detected (both ERCS = 0 and LOCS = 1).

In this state, the FLL loop is open. The DCO is on, and the output clock signal ICGOUT frequency is given by $f_{ICGDCLK}$ / R. The ICGDCLK frequency can be varied from 8 MHz to 40 MHz by writing a new value into the filter registers (ICGFLTH and ICGFLTL). This is the only mode in which the filter registers can be written.

If this mode is entered due to a reset, $f_{ICGDCLK}$ will default to f_{Self_reset} which is nominally 8 MHz. If this mode is entered from FLL engaged internal, $f_{ICGDCLK}$ will maintain the previous frequency.If this mode is entered from FLL engaged external (either by programming CLKS or due to a loss of external reference clock), $f_{ICGDCLK}$ will maintain the previous frequency, but ICGOUT will double if the FLL was unlocked. If this mode is entered from off mode, $f_{ICGDCLK}$ will be equal to the frequency of ICGDCLK before

entering off mode. If CLKS bits are set to 01 or 11 coming out of the Off state, the ICG enters this mode until ICGDCLK is stable as determined by the DCOS bit. After ICGDCLK is considered stable, the ICG automatically closes the loop by switching to FLL engaged (internal or external) as selected by the CLKS bits.

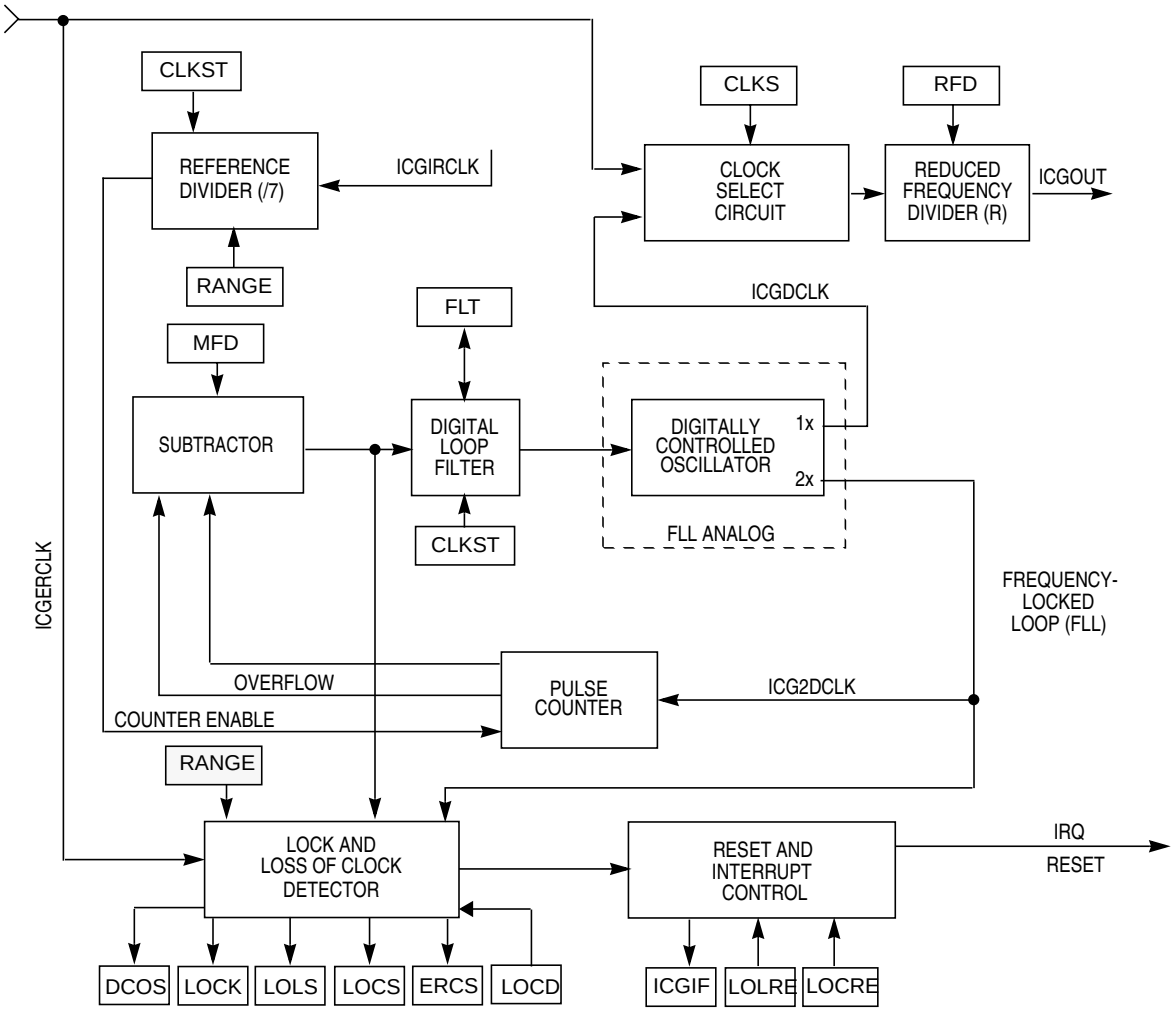


Figure 10-13. Detailed Frequency-Locked Loop Block Diagram

10.5.3 FLL Engaged, Internal Clock (FEI) Mode

FLL engaged internal (FEI) is entered when any of the following conditions occur:

- CLKS bits are written to 01
- The DCO clock stabilizes (DCOS = 1) while in SCM upon exiting the off state with CLKS = 01

In FLL engaged internal mode, the reference clock is derived from the internal reference clock ICGIRCLK, and the FLL loop will attempt to lock the ICGDCLK frequency to the desired value, as selected by the MFD bits.

10.5.4 FLL Engaged Internal Unlocked

FEI unlocked is a temporary state that is entered when FEI is entered and the count error (Δn) output from the subtractor is greater than the maximum n_{unlock} or less than the minimum n_{unlock}, as required by the lock detector to detect the unlock condition.

The ICG will remain in this state while the count error (Δn) is greater than the maximum n_{lock} or less than the minimum n_{lock}, as required by the lock detector to detect the lock condition.

In this state the output clock signal ICGOUT frequency is given by $f_{ICGDCLK}$ / R.

10.5.5 FLL Engaged Internal Locked

FLL engaged internal locked is entered from FEI unlocked when the count error (Δn), which comes from the subtractor, is less than n_{lock} (max) and greater than n_{lock} (min) for a given number of samples, as required by the lock detector to detect the lock condition. The output clock signal ICGOUT frequency is given by $f_{ICGDCLK}$ / R. In FEI locked, the filter value is updated only once every four comparison cycles. The update made is an average of the error measurements taken in the four previous comparisons.

10.5.6 FLL Bypassed, External Clock (FBE) Mode

FLL bypassed external (FBE) is entered when any of the following conditions occur:

- From SCM when CLKS = 10 and ERCS is high
- When CLKS = 10, ERCS = 1 upon entering off mode, and off is then exited
- From FLL engaged external mode if a loss of DCO clock occurs and the external reference remains valid (both LOCS = 1 and ERCS = 1)

In this state, the DCO and IRG are off and the reference clock is derived from the external reference clock, ICGERCLK. The output clock signal ICGOUT frequency is given by $f_{ICGERCLK}$ / R. If an external clock source is used (REFS = 0), then the input frequency on the EXTAL pin can be anywhere in the range 0 MHz to 40 MHz. If a crystal or resonator is used (REFS = 1), then frequency range is either low for RANGE = 0 or high for RANGE = 1.

10.5.7 FLL Engaged, External Clock (FEE) Mode

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

- CLKS = 11 and ERCS and DCOS are both high.
- The DCO stabilizes (DCOS = 1) while in SCM upon exiting the off state with CLKS = 11.

In FEE mode, the reference clock is derived from the external reference clock ICGERCLK, and the FLL loop will attempt to lock the ICGDCLK frequency to the desired value, as selected by the MFD bits. To run in FEE mode, there must be a working 32 kHz–100 kHz or 2 MHz–10 MHz external clock source. The maximum external clock frequency is limited to 10 MHz in FEE mode to prevent over-clocking the DCO. The minimum multiplier for the FLL, from Table 10-12 is 4. Because 4 X 10 MHz is 40MHz, which is the operational limit of the DCO, the reference clock cannot be any faster than 10 MHz.

10.5.7.1 FLL Engaged External Unlocked

FEE unlocked is entered when FEE is entered and the count error (Δn) output from the subtractor is greater than the maximum n_{unlock} or less than the minimum n_{unlock}, as required by the lock detector to detect the unlock condition.

The ICG will remain in this state while the count error (Δn) is greater than the maximum n_{lock} or less than the minimum n_{lock}, as required by the lock detector to detect the lock condition.

In this state, the pulse counter, subtractor, digital loop filter, and DCO form a closed loop and attempt to lock it according to their operational descriptions later in this section. Upon entering this state and until the FLL becomes locked, the output clock signal ICGOUT frequency is given by $f_{ICGDCLK}$ / (2×R) This extra divide by two prevents frequency overshoots during the initial locking process from exceeding chip-level maximum frequency specifications. After the FLL has locked, if an unexpected loss of lock causes it to re-enter the unlocked state while the ICG remains in FEE mode, the output clock signal ICGOUT frequency is given by $f_{ICGDCLK}$ / R.

10.5.7.2 FLL Engaged External Locked

FEE locked is entered from FEE unlocked when the count error (Δn) is less than n_{lock} (max) and greater than n_{lock} (min) for a given number of samples, as required by the lock detector to detect the lock condition. The output clock signal ICGOUT frequency is given by $f_{ICGDCLK}$/R. In FLL engaged external locked, the filter value is updated only once every four comparison cycles. The update made is an average of the error measurements taken in the four previous comparisons.

10.5.8 FLL Lock and Loss-of-Lock Detection

To determine the FLL locked and loss-of-lock conditions, the pulse counter counts the pulses of the DCO for one comparison cycle (see Table 10-9 for explanation of a comparison cycle) and passes this number to the subtractor. The subtractor compares this value to the value in MFD and produces a count error, Δn. To achieve locked status, Δn must be between n_{lock} (min) and n_{lock} (max). After the FLL has locked, Δn must stay between n_{unlock} (min) and n_{unlock} (max) to remain locked. If Δn goes outside this range unexpectedly, the LOLS status bit is set and remains set until cleared by software or until the MCU is reset. LOLS is cleared by reading ICGS1 then writing 1 to ICGIF (LOLRE = 0), or by a loss-of-lock induced reset (LOLRE = 1), or by any MCU reset.

If the ICG enters the off state due to stop mode when ENBDM = OSCSTEN = 0, the FLL loses locked status (LOCK is cleared), but LOLS remains unchanged because this is not an unexpected loss-of-lock condition. Though it would be unusual, if ENBDM is cleared to 0 while the MCU is in stop, the ICG enters the off state. Because this is an unexpected stopping of clocks, LOLS will be set when the MCU wakes up from stop.

Expected loss of lock occurs when the MFD or CLKS bits are changed or in FEI mode only, when the TRIM bits are changed. In these cases, the LOCK bit will be cleared until the FLL regains lock, but the LOLS will not be set.

10.5.9 FLL Loss-of-Clock Detection

The reference clock and the DCO clock are monitored under different conditions (see Table 10-8). Provided the reference frequency is being monitored, ERCS = 1 indicates that the reference clock meets minimum frequency requirements. When the reference and/or DCO clock(s) are being monitored, if either one falls below a certain frequency, f_{LOR} and f_{LOD}, respectively, the LOCS status bit will be set to indicate the error. LOCS will remain set until it is acknowledged or until the MCU is reset. LOCS is cleared by reading ICGS1 then writing 1 to ICGIF (LOCRE = 0), or by a loss-of-clock induced reset (LOCRE = 1), or by any MCU reset.

If the ICG is in FEE, a loss of reference clock causes the ICG to enter SCM, and a loss of DCO clock causes the ICG to enter FBE mode. If the ICG is in FBE mode, a loss of reference clock will cause the ICG to enter SCM. In each case, the CLKST and CLKS bits will be automatically changed to reflect the new state.

If the ICG is in FEE mode when a loss of clock occurs and the ERCS is still set to 1, then the CLKST bits are set to 10 and the ICG reverts to FBE mode.

A loss of clock will also cause a loss of lock when in FEE or FEI modes. Because the method of clearing the LOCS and LOLS bits is the same, this would only be an issue in the unlikely case that LOLRE = 1 and LOCRE = 0. In this case, the interrupt would be overridden by the reset for the loss of lock.

Table 10-8. Clock Monitoring (When LOCD = 0)

Mode	CLKS	REFST	ERCS	External Reference Clock Monitored?	DCO Clock Monitored?
Off	0X or 11	X	Forced Low	No	No
	10	0	Forced Low	No	No
	10	1	Real-Time[1]	Yes(1)	No
SCM (CLKST = 00)	0X	X	Forced Low	No	Yes[2]
	10	0	Forced High	No	Yes(2)
	10	1	Real-Time	Yes	Yes(2)
	11	X	Real-Time	Yes	Yes(2)
FEI (CLKST = 01)	0X	X	Forced Low	No	Yes
	11	X	Real-Time	Yes	Yes
FBE (CLKST = 10)	10	0	Forced High	No	No
	10	1	Real-Time	Yes	No
FEE (CLKST = 11)	11	X	Real-Time	Yes	Yes

1 If ENABLE is high (waiting for external crystal start-up after exiting stop).

2 DCO clock will not be monitored until DCOS = 1 upon entering SCM from off or FLL bypassed external mode.

10.5.10 Clock Mode Requirements

A clock mode is requested by writing to CLKS1:CLKS0 and the actual clock mode is indicated by CLKST1:CLKST0. Provided minimum conditions are met, the status shown in CLKST1:CLKST0 should be the same as the requested mode in CLKS1:CLKS0. Table 10-9 shows the relationship between CLKS, CLKST, and ICGOUT. It also shows the conditions for CLKS = CLKST or the reason CLKS ≠ CLKST.

NOTE

If a crystal will be used before the next reset, then be sure to set REFS = 1 and CLKS = 1x on the first write to the ICGC1 register. Failure to do so will result in "locking" REFS = 0 which will prevent the oscillator amplifier from being enabled until the next reset occurs.

Table 10-9. ICG State Table

Actual Mode (CLKST)	Desired Mode (CLKS)	Range	Reference Frequency ($f_{REFERENCE}$)	Comparison Cycle Time	ICGOUT	Conditions[1] for CLKS = CLKST	Reason CLKS1 ≠ CLKST
Off (XX)	Off (XX)	X	0	—	0	—	—
	FBE (10)	X	0	—	0	—	ERCS = 0
SCM (00)	SCM (00)	X	$f_{ICGIRCLK}/7$[2]	$8/f_{ICGIRCLK}$	ICGDCLK/R	Not switching from FBE to SCM	—
	FEI (01)	0	$f_{ICGIRCLK}/7$(1)	$8/f_{ICGIRCLK}$	ICGDCLK/R	—	DCOS = 0
	FBE (10)	X	$f_{ICGIRCLK}/7$(1)	$8/f_{ICGIRCLK}$	ICGDCLK/R	—	ERCS = 0
	FEE (11)	X	$f_{ICGIRCLK}/7$(1)	$8/f_{ICGIRCLK}$	ICGDCLK/R	—	DCOS = 0 or ERCS = 0
FEI (01)	FEI (01)	0	$f_{ICGIRCLK}/7$	$8/f_{ICGIRCLK}$	ICGDCLK/R	DCOS = 1	—
	FEE (11)	X	$f_{ICGIRCLK}/7$	$8/f_{ICGIRCLK}$	ICGDCLK/R	—	ERCS = 0
FBE (10)	FBE (10)	X	0	—	ICGERCLK/R	ERCS = 1	—
	FEE (11)	X	0	—	ICGERCLK/R	—	LOCS = 1 & ERCS = 1
FEE (11)	FEE (11)	0	$f_{ICGERCLK}$	$2/f_{ICGERCLK}$	ICGDCLK/R[3]	ERCS = 1 and DCOS = 1	—
		1	$f_{ICGERCLK}$	$128/f_{ICGERCLK}$	ICGDCLK/R(2)	ERCS = 1 and DCOS = 1	—

1 CLKST will not update immediately after a write to CLKS. Several bus cycles are required before CLKST updates to the new value.

2 The reference frequency has no effect on ICGOUT in SCM, but the reference frequency is still used in making the comparisons that determine the DCOS bit

3 After initial LOCK; will be ICGDCLK/2R during initial locking process and while FLL is re-locking after the MFD bits are changed.

10.5.11 Fixed Frequency Clock

The ICG provides a fixed frequency clock output, XCLK, for use by on-chip peripherals. This output is equal to the internal bus clock, BUSCLK, in all modes except FEE. In FEE mode, XCLK is equal to ICGERCLK ÷ 2 when the following conditions are met:

- $(P \times N) \div R \geq 4$ where P is determined by RANGE (see Table 10-11), N and R are determined by MFD and RFD respectively (see Table 10-12).
- LOCK = 1.

If the above conditions are not true, then XCLK is equal to BUSCLK.

When the ICG is in either FEI or SCM mode, XCLK is turned off. Any peripherals which can use XCLK as a clock source must not do so when the ICG is in FEI or SCM mode.

10.5.12 High Gain Oscillator

The oscillator has the option of running in a high gain oscillator (HGO) mode, which improves the oscillator's resistance to EMC noise when running in FBE or FEE modes. This option is selected by writing a 1 to the HGO bit in the ICGC1 register. HGO is used with both the high and low range oscillators but is only valid when REFS = 1 in the ICGC1 register. When HGO = 0, the standard low-power oscillator is selected. This bit is writable only once after any reset.

10.6 Initialization/Application Information

10.6.1 Introduction

The section is intended to give some basic direction on which configuration a user would want to select when initializing the ICG. For some applications, the serial communication link may dictate the accuracy of the clock reference. For other applications, lowest power consumption may be the chief clock consideration. Still others may have lowest cost as the primary goal. The ICG allows great flexibility in choosing which is best for any application.

Table 10-10. ICG Configuration Consideration

	Clock Reference Source = Internal	Clock Reference Source = External
FLL Engaged	**FEI** 4 MHz < f_{Bus} < 20 MHz. Medium power (will be less than FEE if oscillator range = high) Good clock accuracy (After IRG is trimmed) **Lowest system cost** (no external components required) IRG is on. DCO is on. [1]	**FEE** 4 MHz < f_{Bus} < 20 MHz Medium power (will be less than FEI if oscillator range = low) High clock accuracy Medium/High system cost (crystal, resonator or external clock source required) IRG is off. DCO is on.
FLL Bypassed	**SCM** This mode is mainly provided for quick and reliable system startup. 3 MHz < f_{Bus} < 5 MHz (default). 3 MHz < f_{Bus} < 20 MHz (via filter bits). Medium power Poor accuracy. IRG is off. DCO is on and open loop.	**FBE** f_{Bus} range ≤ 8 MHz when crystal or resonator is used. **Lowest power** Highest clock accuracy Medium/High system cost (Crystal, resonator or external clock source required) IRG is off. DCO is off.

1 The IRG typically consumes 100 µA. The FLL and DCO typically consumes 0.5 to 2.5 mA, depending upon output frequency. For minimum power consumption and minimum jitter, choose N and R to be as small as possible.

The following sections contain initialization examples for various configurations.

NOTE

Hexadecimal values designated by a preceding $, binary values designated by a preceding %, and decimal values have no preceding character.

Important configuration information is repeated here for reference.

Table 10-11. ICGOUT Frequency Calculation Options

Clock Scheme	f_{ICGOUT} [1]	P	Note
SCM — self-clocked mode (FLL bypassed internal)	$f_{ICGDCLK}$ / R	NA	Typical f_{ICGOUT} = 8 MHz immediately after reset
FBE — FLL bypassed external	f_{ext} / R	NA	
FEI — FLL engaged internal	(f_{IRG} / 7)* 64 * N / R	64	Typical f_{IRG} = 243 kHz
FEE — FLL engaged external	f_{ext} * P * N / R	Range = 0 ; P = 64 Range = 1; P = 1	

1 Ensure that $f_{ICGDCLK}$, which is equal to f_{ICGOUT} * R, does not exceed $f_{ICGDCLKmax}$.

Table 10-12. MFD and RFD Decode Table

MFD Value	Multiplication Factor (N)
000	4
001	6
010	8
011	10
100	12

RFD	Division Factor (R)
000	÷1
001	÷2
010	÷4
011	÷8
100	÷16

Table 10-12. MFD and RFD Decode Table

101	14
110	16
111	18

101	÷32
110	÷64
111	÷128

10.6.2 Example #1: External Crystal = 32 kHz, Bus Frequency = 4.19 MHz

In this example, the FLL will be used (in FEE mode) to multiply the external 32 kHz oscillator up to 8.38 MHz to achieve 4.19 MHz bus frequency.

After the MCU is released from reset, the ICG is in self-clocked mode (SCM) and supplies approximately 8 MHz on ICGOUT, which corresponds to a 4 MHz bus frequency (f_{Bus}).

The clock scheme will be FLL engaged, external (FEE). So

$$f_{ICGOUT} = f_{ext} * P * N / R ; P = 64, f_{ext} = 32 \text{ kHz} \quad \textit{Eqn. 10-1}$$

Solving for N / R gives:

$$N / R = 8.38 \text{ MHz} / (32 \text{ kHz} * 64) = 4 ; \text{we can choose N = 4 and R =1} \quad \textit{Eqn. 10-2}$$

The values needed in each register to set up the desired operation are:

ICGC1 = $38 (%00111000)

Bit 7	HGO	0	Configures oscillator for low power
Bit 6	RANGE	0	Configures oscillator for low-frequency range; FLL prescale factor is 64
Bit 5	REFS	1	Oscillator using crystal or resonator is requested
Bits 4:3	CLKS	11	FLL engaged, external reference clock mode
Bit 2	OSCSTEN	0	Oscillator disabled
Bit 1	LOCD	0	Loss-of-clock detection enabled
Bit 0		0	Unimplemented or reserved, always reads zero

ICGC2 = $00 (%00000000)

Bit 7	LOLRE	0	Generates an interrupt request on loss of lock
Bits 6:4	MFD	000	Sets the MFD multiplication factor to 4
Bit 3	LOCRE	0	Generates an interrupt request on loss of clock
Bits 2:0	RFD	000	Sets the RFD division factor to ÷1

ICGS1 = $xx

This is read only except for clearing interrupt flag

ICGS2 = $xx

This is read only; should read DCOS = 1 before performing any time critical tasks

ICGFLTLU/L = $xx

Only needed in self-clocked mode; FLT will be adjusted by loop to give 8.38 MHz DCO clock
Bits 15:12 unused 0000

Bits 11:0 FLT No need for user initialization

ICGTRM = $xx

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

Figure 10-14 shows flow charts for three conditions requiring ICG initialization.

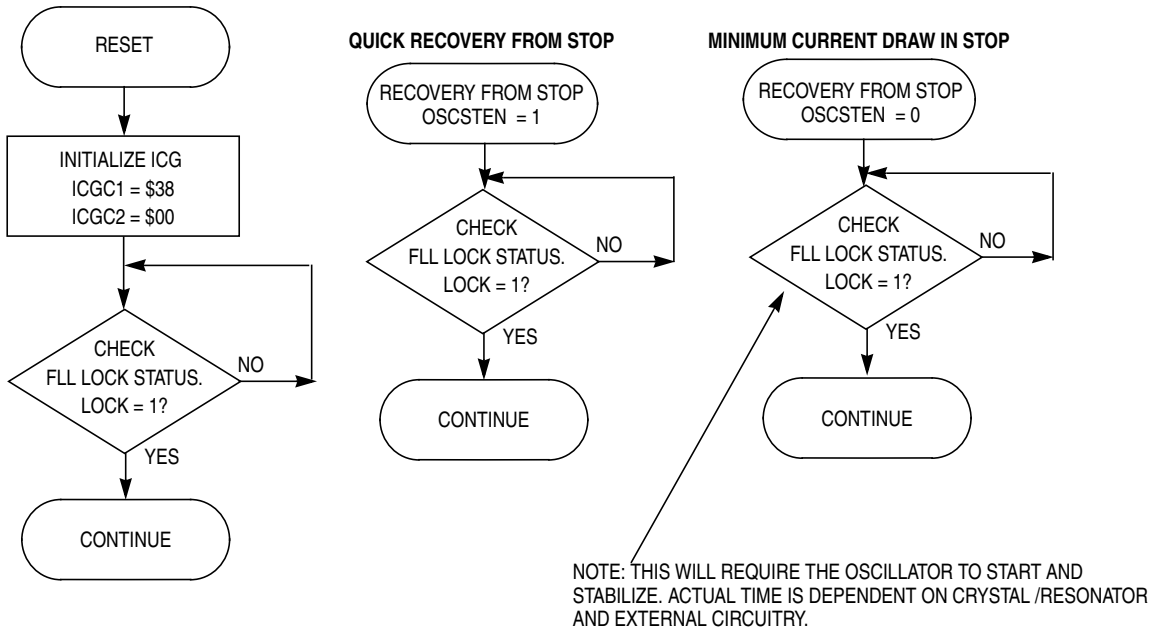


Figure 10-14. ICG Initialization for FEE in Example #1

10.6.3 Example #2: External Crystal = 4 MHz, Bus Frequency = 20 MHz

In this example, the FLL will be used (in FEE mode) to multiply the external 4 MHz oscillator up to 40-MHz to achieve 20 MHz bus frequency.

After the MCU is released from reset, the ICG is in self-clocked mode (SCM) and supplies approximately 8 MHz on ICGOUT which corresponds to a 4 MHz bus frequency (f_{Bus}).

During reset initialization software, the clock scheme will be set to FLL engaged, external (FEE). So

$$f_{ICGOUT} = f_{ext} * P * N / R ; P = 1, f_{ext} = 4.00 \text{ MHz} \qquad \textit{Eqn. 10-3}$$

Solving for N / R gives:

$$N / R = 40 \text{ MHz} / (4 \text{ MHz} * 1) = 10 ; \text{We can choose N = 10 and R = 1} \qquad \textit{Eqn. 10-4}$$

The values needed in each register to set up the desired operation are:

ICGC1 = $78 (%01111000)

Bit 7	HGO	0	Configures oscillator for low power
Bit 6	RANGE	1	Configures oscillator for high-frequency range; FLL prescale factor is 1
Bit 5	REFS	1	Requests an oscillator
Bits 4:3	CLKS	11	FLL engaged, external reference clock mode
Bit 2	OSCSTEN	0	Disables the oscillator
Bit 1	LOCD	0	Loss-of-clock detection enabled
Bit 0		0	Unimplemented or reserved, always reads zero

ICGC2 = $30 (%00110000)

Bit 7	LOLRE	0	Generates an interrupt request on loss of lock
Bit 6:4	MFD	011	Sets the MFD multiplication factor to 10
Bit 3	LOCRE	0	Generates an interrupt request on loss of clock
Bit 2:0	RFD	000	Sets the RFD division factor to ÷1

ICGS1 = $xx

This is read only except for clearing interrupt flag

ICGS2 = $xx

This is read only. Should read DCOS before performing any time critical tasks

ICGFLTLU/L = $xx

Not used in this example

ICGTRM

Not used in this example

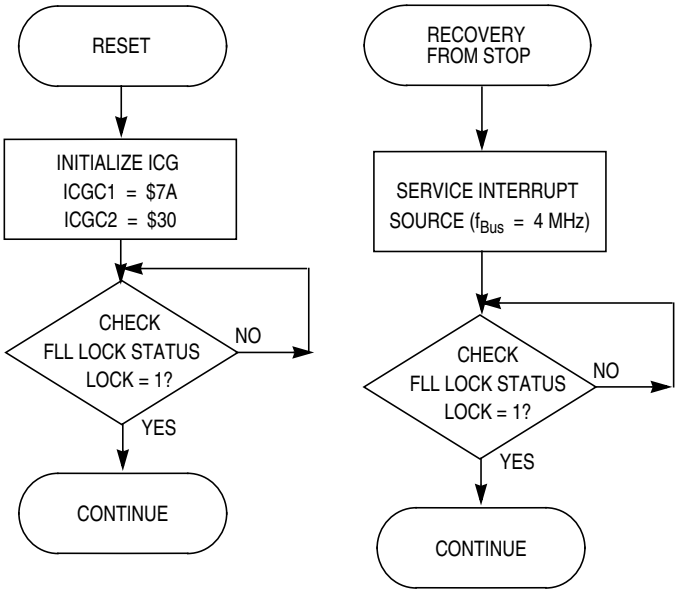


Figure 10-15. ICG Initialization and Stop Recovery for Example #2

10.6.4 Example #3: No External Crystal Connection, 5.4 MHz Bus Frequency

In this example, the FLL will be used (in FEI mode) to multiply the internal 243 kHz (approximate) reference clock up to 10.8 MHz to achieve 5.4 MHz bus frequency. This system will also use the trim function to fine tune the frequency based on an external reference signal.

After the MCU is released from reset, the ICG is in self-clocked mode (SCM) and supplies approximately 8 MHz on ICGOUT which corresponds to a 4 MHz bus frequency (f_{Bus}).

The clock scheme will be FLL engaged, internal (FEI). So

$$f_{ICGOUT} = (f_{IRG} / 7) * P * N / R ; P = 64,\ f_{IRG} = 243\ kHz \qquad \textit{Eqn. 10-5}$$

Solving for N / R gives:

$$N / R = 10.8\ MHz /(243/7\ kHz * 64) = 4.86 ;\ \text{We can choose N = 10 and R = 2.} \qquad \textit{Eqn. 10-6}$$

A trim procedure will be required to hone the frequency to exactly 5.4 MHz. An example of the trim procedure is shown in example #4.

The values needed in each register to set up the desired operation are:

ICGC1 = $28 (%00101000)

Bit 7	HGO	0	Configures oscillator for low power
Bit 6	RANGE	0	Configures oscillator for low-frequency range; FLL prescale factor is 64
Bit 5	REFS	1	Oscillator using crystal or resonator requested (bit is really a don’t care)
Bits 4:3	CLKS	01	FLL engaged, internal reference clock mode
Bit 2	OSCSTEN	0	Disables the oscillator
Bit 1	LOCD	0	Loss-of-clock enabled
Bit 0		0	Unimplemented or reserved, always reads zero

ICGC2 = $31 (%00110001)

Bit 7	LOLRE	0	Generates an interrupt request on loss of lock
Bit 6:4	MFD	011	Sets the MFD multiplication factor to 10
Bit 3	LOCRE	0	Generates an interrupt request on loss of clock
Bit 2:0	RFD	001	Sets the RFD division factor to ÷2

ICGS1 = $xx

This is read only except for clearing interrupt flag

ICGS2 = $xx

This is read only; good idea to read this before performing time critical operations

ICGFLTLU/L = $xx

Not used in this example

ICGTRM = $xx

Bit 7:0 TRIM — Only need to write when trimming internal oscillator; done in separate operation (see example #4)

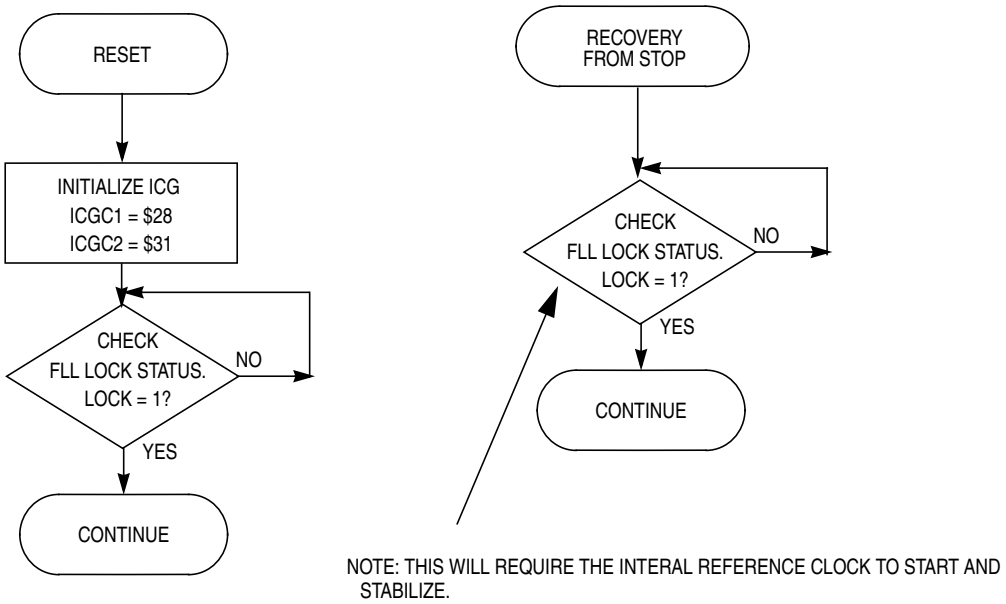


Figure 10-16. ICG Initialization and Stop Recovery for Example #3

10.6.5 Example #4: Internal Clock Generator Trim

The internally generated clock source is guaranteed to have a period ± 25% of the nominal value. In some cases, this may be sufficient accuracy. For other applications that require a tight frequency tolerance, a trimming procedure is provided that will allow a very accurate source. This section outlines one example of trimming the internal oscillator. Many other possible trimming procedures are valid and can be used.

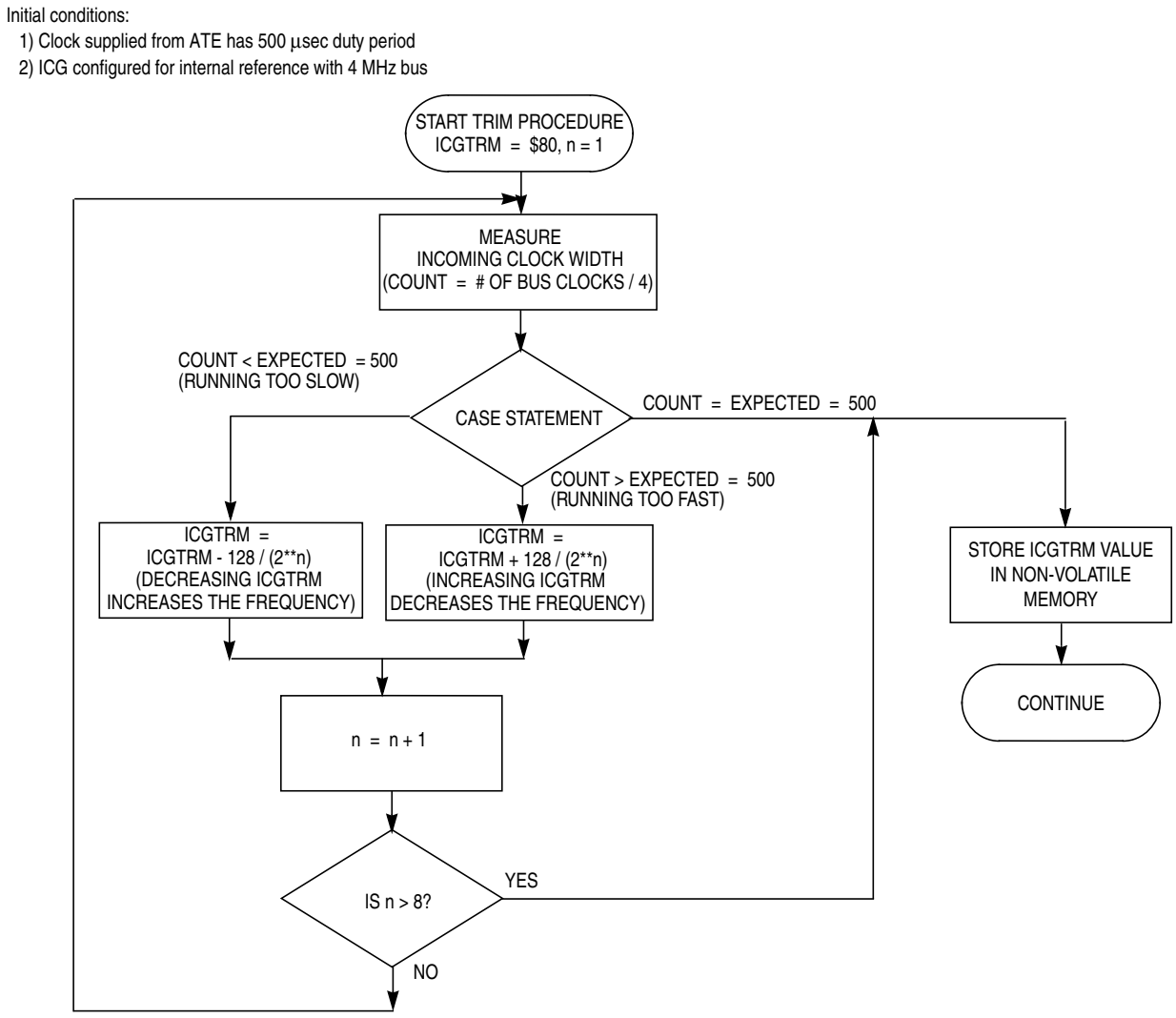


Figure 10-17. Trim Procedure

In this particular case, the MCU has been attached to a PCB and the entire assembly is undergoing final test with automated test equipment. A separate signal or message is provided to the MCU operating under user provided software control. The MCU initiates a trim procedure as outlined in Figure 10-17 while the tester supplies a precision reference signal.

If the intended bus frequency is near the maximum allowed for the device, it is recommended to trim using a reduction divisor (R) twice the final value. After the trim procedure is complete, the reduction divisor can be restored. This will prevent accidental overshoot of the maximum clock frequency.

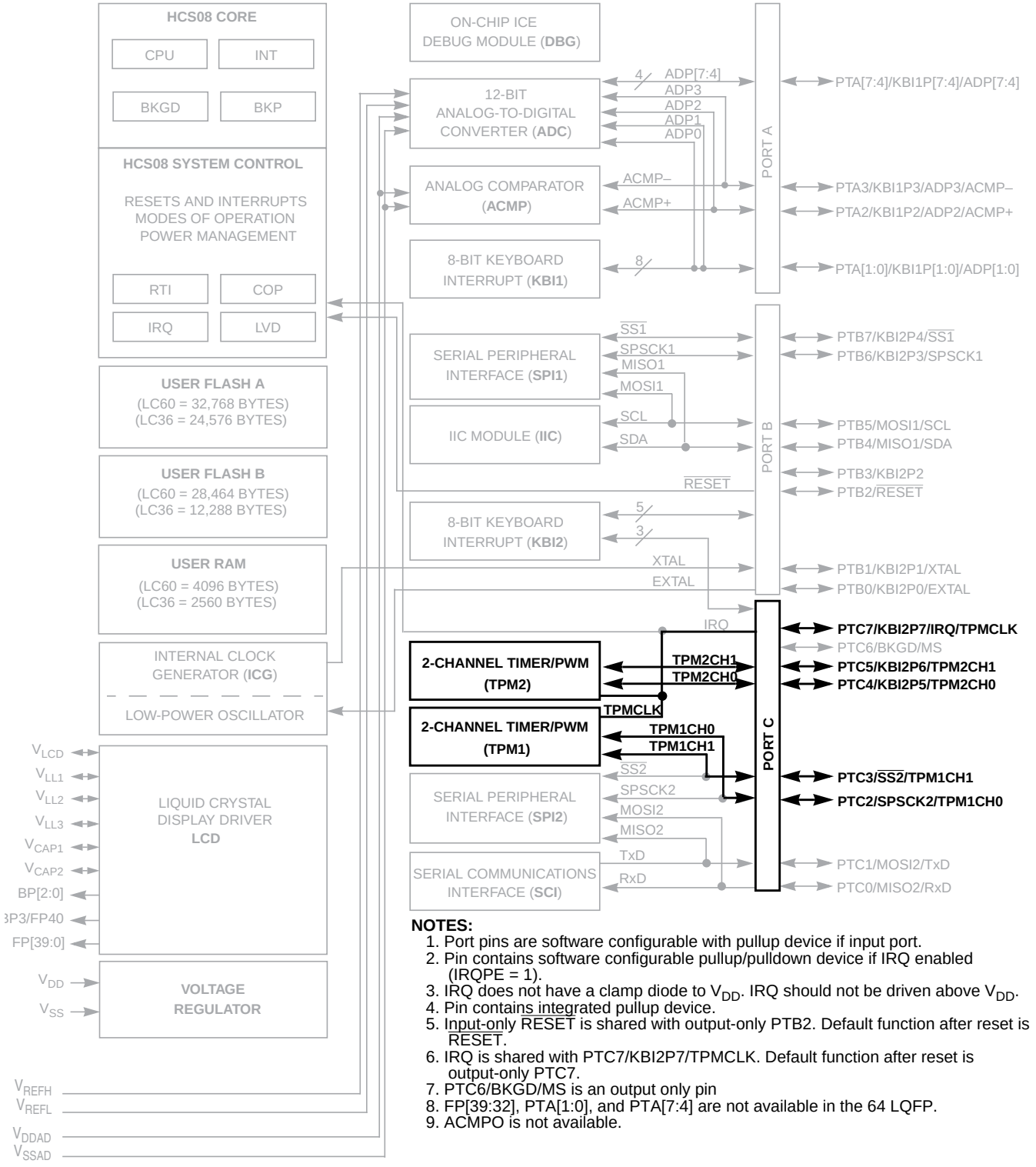
Chapter 11
Timer Pulse-Width Modulator (S08TPMV2)

11.1 Introduction

The TPM uses one input/output (I/O) pin per channel, TPMxCHn where x is the TPM number (for example, 1 or 2) and n is the channel number (for example, 0–4). The TPM shares its I/O pins with general-purpose I/O port pins (refer to the Pins and Connections chapter for more information).

The MC9S08LC60 Series has two TPM modules

Figure 11-1 shows the MC9S08LC60 Series block diagram with the TPMs highlighted.



NOTES:

1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 11-1. MC9S08LC60 Series Block Diagram Highlighting TPM Block and Pins

11.1.1 Features

The TPM has the following features:

- Each TPM may be configured for buffered, center-aligned pulse-width modulation (CPWM) on all channels
- Clock sources independently selectable per TPM (multiple TPMs device)
- Selectable clock sources (device dependent): bus clock, fixed system clock, external pin
- Clock prescaler taps for divide by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit free-running or up/down (CPWM) count operation
- 16-bit modulus register to control counter range
- Timer system enable
- One interrupt per channel plus a terminal count interrupt for each TPM module (multiple TPMs device)
- Channel features:
  - Each channel may be input capture, output compare, or buffered 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

11.1.2 Block Diagram

Figure 11-2 shows the structure of a TPM. Some MCUs include more than one TPM, with various numbers of channels.

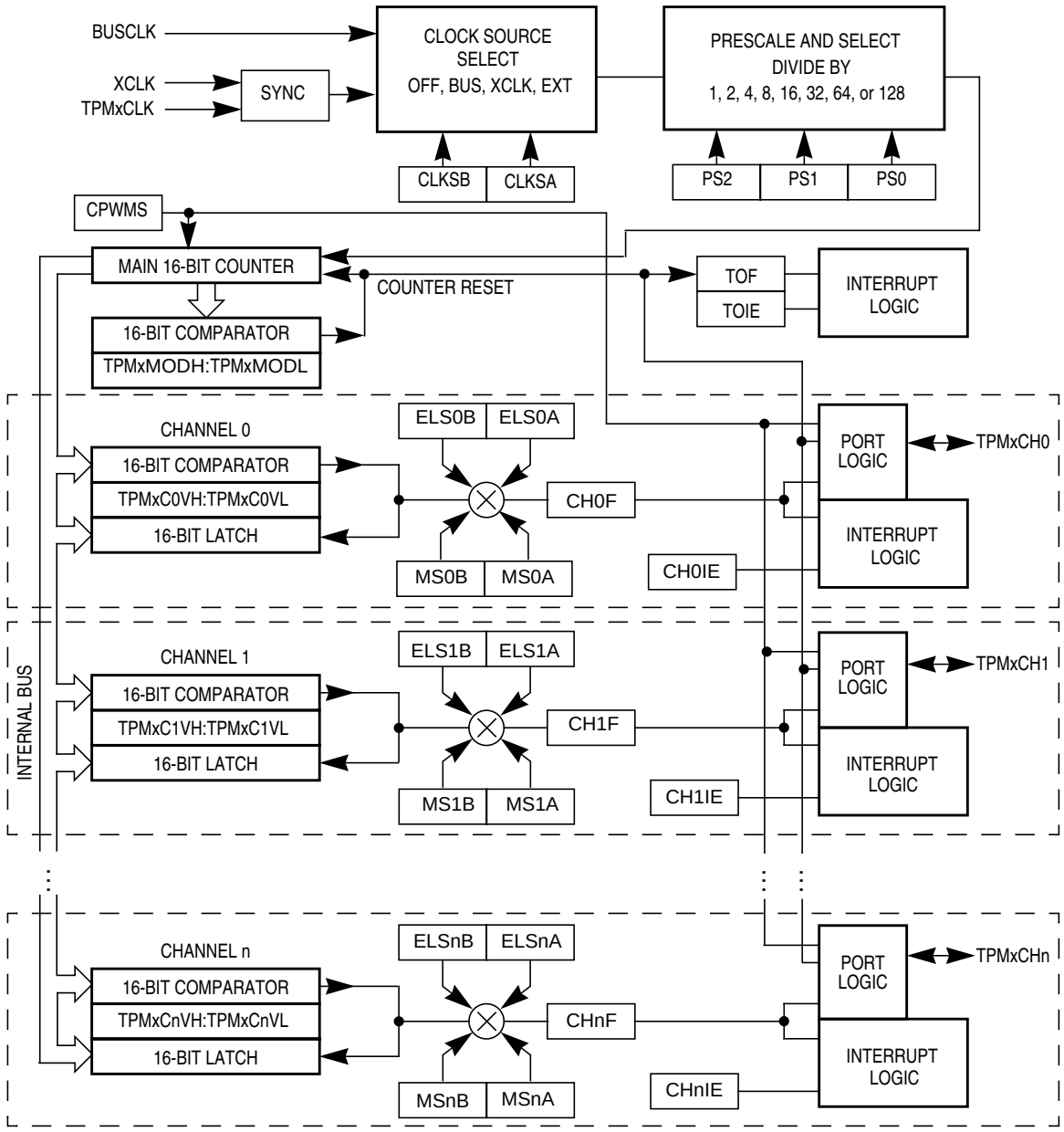


Figure 11-2. TPM Block Diagram

The central component of the TPM is the 16-bit counter that can operate as a free-running counter, a modulo counter, or an up-/down-counter when the TPM is configured for center-aligned PWM. 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 byte of the TPMxCNT counter resets the counter regardless of the data value written.

All TPM channels are programmable independently as input capture, output compare, or buffered edge-aligned PWM channels.

11.2 External Signal Description

When any pin associated with the timer is configured as a timer input, a passive pullup can be enabled. After reset, the TPM modules are disabled and all pins default to general-purpose inputs with the passive pullups disabled.

11.2.1 External TPM Clock Sources

When control bits CLKSB:CLKSA in the timer status and control register are set to 1:1, the prescaler and consequently the 16-bit counter for TPMx are driven by an external clock source, TPMxCLK, connected to an I/O pin. A synchronizer is needed between the external clock and the rest of the TPM. This synchronizer is clocked by the bus clock so the frequency of the external source must be less than one-half the frequency of the bus rate clock. The upper frequency limit for this external clock source is specified to be one-fourth the bus frequency to conservatively accommodate duty cycle and phase-locked loop (PLL) or frequency-locked loop (FLL) frequency jitter effects.

On some devices the external clock input is shared with one of the TPM channels. When a TPM channel is shared as the external clock input, the associated TPM channel cannot use the pin. (The channel can still be used in output compare mode as a software timer.) Also, if one of the TPM channels is used as the external clock input, the corresponding ELSnB:ELSnA control bits must be set to 0:0 so the channel is not trying to use the same pin.

11.2.2 TPMxCHn — TPMx Channel n I/O Pins

Each TPM channel is associated with an I/O pin on the MCU. The function of this pin depends on the configuration of the channel. In some cases, no pin function is needed so the pin reverts to being controlled by general-purpose I/O controls. When a timer has control of a port pin, the port data and data direction registers do not affect the related pin(s). See the Pins and Connections chapter for additional information about shared pin functions.

11.3 Register Definition

The TPM includes:

- An 8-bit status and control register (TPMxSC)
- A 16-bit counter (TPMxCNTH:TPMxCNTL)
- A 16-bit modulo register (TPMxMODH:TPMxMODL)

Each timer channel has:

- An 8-bit status and control register (TPMxCnSC)
- A 16-bit channel value register (TPMxCnVH:TPMxCnVL)

Refer to the direct-page register summary in the Memory chapter of this data sheet for the absolute address assignments for all TPM 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 MCU systems have more than one TPM, so register names include placeholder characters to identify which TPM and which channel is being referenced. For example, TPMxCnSC refers to timer (TPM) x, channel n and TPM1C2SC is the status and control register for timer 1, channel 2.

11.3.1 Timer x Status and Control Register (TPMxSC)

TPMxSC contains the overflow status flag and control bits that are used to configure the interrupt enable, TPM configuration, clock source, and prescale divisor. 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								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-3. Timer x Status and Control Register (TPMxSC)

Table 11-1. TPMxSC Register Field Descriptions

Field	Description
7 TOF	**Timer Overflow Flag** — This flag is set when the TPM counter changes to 0x0000 after reaching the modulo value programmed in the TPM counter modulo registers. When the TPM is configured for CPWM, TOF is set after the counter has reached the value in the modulo register, at the transition to the next lower count value. Clear TOF by reading the TPM status and control register when TOF is set and then writing a 0 to TOF. If another TPM overflow occurs before the clearing sequence is complete, the sequence is reset so TOF would remain set after the clear sequence was completed for the earlier TOF. Reset clears TOF. Writing a 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 1. Reset clears TOIE. 0 TOF interrupts inhibited (use software polling) 1 TOF interrupts enabled
5 CPWMS	**Center-Aligned PWM Select** — This read/write bit selects CPWM operating mode. Reset clears this bit so 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 TPMx 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 TPMx channels operate in center-aligned PWM mode
4:3 CLKS[B:A]	**Clock Source Select** — As shown in Table 11-2, this 2-bit field is used to disable the TPM system or select one of three clock sources to drive the counter prescaler. The external source and the XCLK are synchronized to the bus clock by an on-chip synchronization circuit.
2:0 PS[2:0]	**Prescale Divisor Select** — This 3-bit field selects one of eight divisors for the TPM clock input as shown in Table 11-3. This prescaler is located after any clock source synchronization or clock source selection, so it affects whatever clock source is selected to drive the TPM system.

Table 11-2. TPM Clock Source Selection

CLKSB:CLKSA	TPM Clock Source to Prescaler Input
0:0	No clock selected (TPMx disabled)
0:1	Bus rate clock (BUSCLK)
1:0	Fixed system clock (XCLK)
1:1	External source (TPMxCLK)[1,2]

1 The maximum frequency that is allowed as an external clock is one-fourth of the bus frequency.

2 If the external clock input is shared with channel n and is selected as the TPM clock source, the corresponding ELSnB:ELSnA control bits should be set to 0:0 so channel n does not try to use the same pin for a conflicting function.

Table 11-3. Prescale Divisor Selection

PS2:PS1:PS0	TPM Clock Source Divided-By
0:0:0	1
0:0:1	2
0:1:0	4
0:1:1	8
1:0:0	16
1:0:1	32
1:1:0	64
1:1:1	128

11.3.2 Timer x 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 byte is read. This allows coherent 16-bit reads in either order. The coherency mechanism is automatically restarted by an MCU reset, a write of any value to TPMxCNTH or TPMxCNTL, or any write to the timer status/control register (TPMxSC).

Reset clears the TPM counter registers.

	7	6	5	4	3	2	1	0
R	Bit 15	14	13	12	11	10	9	Bit 8
W	Any write to TPMxCNTH clears the 16-bit counter.							
Reset	0	0	0	0	0	0	0	0

Figure 11-4. Timer x Counter Register High (TPMxCNTH)

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W	Any write to TPMxCNTL clears the 16-bit counter.							
Reset	0	0	0	0	0	0	0	0

Figure 11-5. Timer x Counter Register Low (TPMxCNTL)

When background mode is active, the timer counter and the coherency mechanism are frozen such that the buffer latches remain in the state they were in when the background mode became active even if one or both bytes of the counter are read while background mode is active.

11.3.3 Timer x 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 (CPWMS = 0) or starts counting down (CPWMS = 1), and the overflow flag (TOF) becomes set. Writing to TPMxMODH or TPMxMODL inhibits TOF and overflow interrupts until the other byte is written. Reset sets the TPM counter modulo registers to 0x0000, which results in a free-running timer counter (modulo disabled).

	7	6	5	4	3	2	1	0
R W	Bit 15	14	13	12	11	10	9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 11-6. Timer x Counter Modulo Register High (TPMxMODH)

	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 11-7. Timer x Counter Modulo Register Low (TPMxMODL)

It is good practice to wait for an overflow interrupt so both bytes of the modulo register can be written well before a new overflow. An alternative approach is to reset the TPM counter before writing to the TPM modulo registers to avoid confusion about when the first counter overflow will occur.

11.3.4 Timer x Channel n Status and Control Register (TPMxCnSC)

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

Figure 11-8. Timer x Channel n Status and Control Register (TPMxCnSC)

Table 11-4. TPMxCnSC Register Field Descriptions

Field	Description
7 CHnF	**Channel n Flag** — When channel n is configured for input capture, this flag bit is set when an active edge occurs on the channel n pin. When channel n is an output compare or edge-aligned PWM channel, CHnF is set when the value in the TPM counter registers matches the value in the TPM channel n value registers. This flag is seldom used with center-aligned PWMs because it is set every time the counter matches the channel value register, which correspond to both edges of the active duty cycle period. A corresponding interrupt is requested when CHnF is set and interrupts are enabled (CHnIE = 1). Clear CHnF by reading TPMxCnSC while CHnF is set and then writing a 0 to CHnF. If another interrupt request occurs before the clearing sequence is complete, the sequence is reset so CHnF would remain set after the clear sequence was completed for the earlier CHnF. This is done so a CHnF interrupt request cannot be lost by clearing a previous CHnF. Reset clears CHnF. Writing a 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 occurred on channel n
6 CHnIE	**Channel n Interrupt Enable** — This read/write bit enables interrupts from channel n. Reset clears CHnIE. 0 Channel n interrupt requests disabled (use software polling) 1 Channel n interrupt requests enabled
5 MSnB	**Mode Select B for TPM Channel n** — When CPWMS = 0, MSnB = 1 configures TPM channel n for edge-aligned PWM mode. For a summary of channel mode and setup controls, refer to Table 11-5.
4 MSnA	**Mode Select A for TPM Channel n** — When CPWMS = 0 and MSnB = 0, MSnA configures TPM channel n for input capture mode or output compare mode. Refer to Table 11-5 for a summary of channel mode and setup controls.
3:2 ELSn[B:A]	**Edge/Level Select Bits** — Depending on the operating mode for the timer channel as set by CPWMS:MSnB:MSnA and shown in Table 11-5, these bits select the polarity of the input edge that triggers an input capture event, select the level that will be driven in response to an output compare match, or select the polarity of the PWM output. Setting ELSnB:ELSnA to 0:0 configures the related timer pin as a general-purpose I/O pin unrelated to any timer channel functions. This function is typically used to temporarily disable an input capture channel or to make the timer pin available as a general-purpose I/O pin when the associated timer channel is set up as a software timer that does not require the use of a pin.

Table 11-5. Mode, Edge, and Level Selection

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	XX	00	Pin not used for TPM channel; use as an external clock for the TPM or revert to general-purpose I/O	
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 compare
		10		Clear output on compare
		11		Set output on compare
	1X	10	Edge-aligned PWM	High-true pulses (clear output on compare)
		X1		Low-true pulses (set output on compare)
1	XX	10	Center-aligned PWM	High-true pulses (clear output on compare-up)
		X1		Low-true pulses (set output on compare-up)

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. Typically, a program would clear status flags after changing channel configuration bits and before enabling channel interrupts or using the status flags to avoid any unexpected behavior.

11.3.5 Timer x 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 value registers are cleared by reset.

	7	6	5	4	3	2	1	0
R W	Bit 15	14	13	12	11	10	9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 11-9. Timer x Channel Value Register High (TPMxCnVH)

	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 11-10. Timer 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 byte is read. This latching mechanism also resets (becomes unlatched) when the TPMxCnSC register is written.

In output compare or PWM modes, writing to either byte (TPMxCnVH or TPMxCnVL) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the timer channel value registers. This latching mechanism may be manually reset by writing to the TPMxCnSC register.

This latching mechanism allows coherent 16-bit writes in either order, which is friendly to various compiler implementations.

11.4 Functional Description

All TPM functions are associated with a main 16-bit counter that allows flexible selection of the clock source and prescale divisor. A 16-bit modulo register also is associated with the main 16-bit counter in the TPM. Each TPM channel is optionally associated with an MCU pin and a maskable interrupt function.

The TPM has center-aligned PWM capabilities controlled by the CPWMS control bit in TPMxSC. When CPWMS is set to 1, timer counter TPMxCNT changes to an up-/down-counter and all channels in the associated TPM act as center-aligned PWM channels. When CPWMS = 0, each channel can independently be configured to operate in input capture, output compare, or buffered edge-aligned PWM mode.

The following sections describe the main 16-bit 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 on the operating mode, these topics are covered in the associated mode sections.

11.4.1 Counter

All timer functions are based on the main 16-bit counter (TPMxCNTH:TPMxCNTL). This section discusses selection of the clock source, up-counting vs. up-/down-counting, end-of-count overflow, and manual counter reset.

After any MCU reset, CLKSB:CLKSA = 0:0 so no clock source is selected and the TPM is inactive. Normally, CLKSB:CLKSA would be set to 0:1 so the bus clock drives the timer counter. The clock source for each of the TPM can be independently selected to be off, the bus clock (BUSCLK), the fixed system clock (XCLK), or an external input. The maximum frequency allowed for the external clock option is one-fourth the bus rate. Refer to Section 11.3.1, “Timer x Status and Control Register (TPMxSC)” and Table 11-2 for more information about clock source selection.

When the microcontroller is in active background mode, the TPM temporarily suspends all counting until the microcontroller returns to normal user operating mode. During stop mode, all TPM clocks are stopped; therefore, the TPM is effectively disabled until clocks resume. During wait mode, the TPM continues to operate normally.

The main 16-bit 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 main 16-bit counter counts from 0x0000 through its terminal count and then 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 upward from 0x0000 through its terminal count and then counts downward to 0x0000 where it returns to up-counting. Both 0x0000 and the terminal count value (value in TPMxMODH:TPMxMODL) are normal length counts (one timer clock period long).

An interrupt flag and enable are associated with the main 16-bit counter. The timer overflow flag (TOF) is a software-accessible indication that the timer counter has overflowed. The enable signal selects between software polling (TOIE = 0) where no hardware interrupt is generated, or interrupt-driven operation (TOIE = 1) where a static hardware interrupt is automatically generated whenever the TOF flag is 1.

The conditions that cause TOF to become set depend on the counting mode (up or up/down). In up-counting mode, the main 16-bit counter counts from 0x0000 through 0xFFFF and overflows to 0x0000 on the next counting clock. TOF becomes set at the transition from 0xFFFF to 0x0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to 0x0000. When the main 16-bit counter is operating in up-/down-counting mode, the TOF flag gets set as the counter changes direction at the transition from the value set in the modulus register and 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.)

Because the HCS08 MCU is an 8-bit architecture, a coherency mechanism is built into the timer counter for read operations. Whenever either byte of the counter is read (TPMxCNTH or TPMxCNTL), both bytes are captured into a buffer so when the other byte is read, the value will represent the other byte of the count at the time the first byte was read. The counter continues to count normally, but no new value can be read from either byte until both bytes of the old count have been read.

The main timer counter can be reset manually at any time by writing any value to either byte of the timer count TPMxCNTH or TPMxCNTL. Resetting the counter in this manner also resets the coherency mechanism in case only one byte of the counter was read before resetting the count.

11.4.2 Channel Mode Selection

Provided CPWMS = 0 (center-aligned PWM operation is not specified), the MSnB and MSnA control bits in the channel n status and control registers determine the basic mode of operation for the corresponding channel. Choices include input capture, output compare, and buffered edge-aligned PWM.

11.4.2.1 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 may be chosen as the active edge that triggers an input capture.

When either byte of the 16-bit capture register is read, both bytes are latched into a buffer to support coherent 16-bit accesses regardless of order. The coherency sequence can be manually reset by writing to the channel status/control register (TPMxCnSC).

An input capture event sets a flag bit (CHnF) that can optionally generate a CPU interrupt request.

11.4.2.2 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 the channel value registers of an output compare channel, the TPM can set, clear, or toggle the channel pin.

In output compare mode, values are transferred to the corresponding timer channel value registers only after both 8-bit bytes of a 16-bit register have been written. This 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 can optionally generate a CPU interrupt request.

11.4.2.3 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 setting in the modulus register (TPMxMODH:TPMxMODL). The duty cycle is determined by the setting in the timer channel value register (TPMxCnVH:TPMxCnVL). The polarity of this PWM signal is determined by the setting in the ELSnA control bit. Duty cycle cases of 0 percent and 100 percent are possible.

As Figure 11-11 shows, the output compare value in the TPM channel registers determines the pulse width (duty cycle) of the PWM signal. The time between the modulus overflow and the output compare is the pulse width. If ELSnA = 0, the counter overflow forces the PWM signal high and the output compare forces the PWM signal low. If ELSnA = 1, the counter overflow forces the PWM signal low and the output compare forces the PWM signal high.

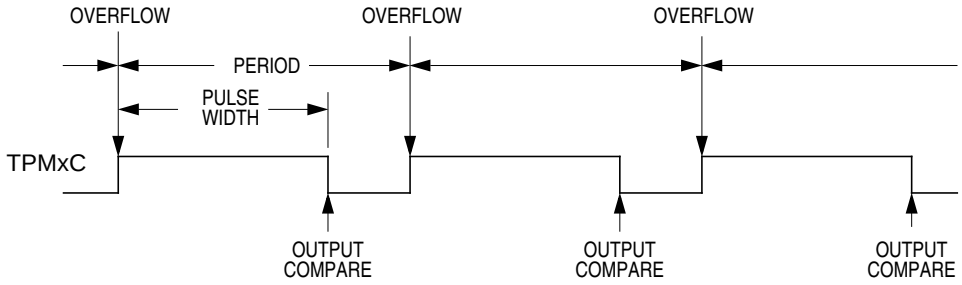


Figure 11-11. PWM Period and Pulse Width (ELSnA = 0)

When the channel value register is set to 0x0000, the duty cycle is 0 percent. By setting the timer channel value register (TPMxCnVH:TPMxCnVL) to a value greater than the modulus setting, 100% duty cycle can be achieved. This implies that the modulus setting must be less than 0xFFFF to get 100% duty cycle.

Because the HCS08 is a family of 8-bit MCUs, the settings in the timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to either register, TPMxCnVH or TPMxCnVL, write to buffer registers. In edge-PWM mode, values are transferred to the corresponding timer channel registers only after both 8-bit bytes of a 16-bit register have been written and the value in the TPMxCNTH:TPMxCNTL counter is 0x0000. (The new duty cycle does not take effect until the next full period.)

11.4.3 Center-Aligned PWM Mode

This type of PWM output uses the up-/down-counting mode of the timer counter (CPWMS = 1). The output compare value in TPMxCnVH:TPMxCnVL determines the pulse width (duty cycle) of the PWM signal and the period is determined by the value in TPMxMODH:TPMxMODL. TPMxMODH:TPMxMODL should be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results. ELSnA will determine the polarity of the CPWM output.

pulse width = 2 x (TPMxCnVH:TPMxCnVL) ***Eqn. 11-1***

period = 2 x (TPMxMODH:TPMxMODL);
for TPMxMODH:TPMxMODL = 0x0001–0x7FFF ***Eqn. 11-2***

If the channel value register TPMxCnVH:TPMxCnVL is zero or negative (bit 15 set), the duty cycle will be 0%. If TPMxCnVH:TPMxCnVL is a positive value (bit 15 clear) and is greater than the (nonzero) modulus setting, the duty cycle will be 100% because the duty cycle compare will never occur. This implies the usable range of periods set by the modulus register is 0x0001 through 0x7FFE (0x7FFF if generation of 100% duty cycle is not necessary). This is not a significant limitation because the resulting period is much longer than required for normal applications.

TPMxMODH:TPMxMODL = 0x0000 is a special case that should not be used with center-aligned PWM mode. When CPWMS = 0, this case corresponds to the counter running free from 0x0000 through 0xFFFF, but when CPWMS = 1 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.

Figure 11-12 shows the output compare value in the TPM channel registers (multiplied by 2), which determines the pulse width (duty cycle) of the CPWM signal. If ELSnA = 0, the compare match while counting up forces the CPWM output signal low and a compare match while counting down forces the output high. 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.

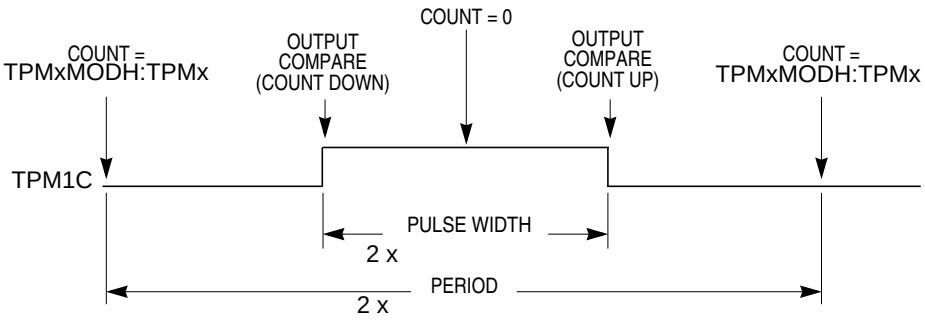


Figure 11-12. CPWM Period and Pulse Width (ELSnA = 0)

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.

Because the HCS08 is a family of 8-bit MCUs, the settings in 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, TPMxMODH, TPMxMODL, TPMxCnVH, and TPMxCnVL, actually write to buffer registers. Values are

transferred to the corresponding timer channel registers only after both 8-bit bytes of a 16-bit register have been written and the timer counter overflows (reverses direction from up-counting to down-counting at the end of the terminal count in the modulus register). This TPMxCNT overflow requirement only applies to PWM channels, not output compares.

Optionally, when TPMxCNTH:TPMxCNTL = TPMxMODH:TPMxMODL, the TPM can generate a TOF interrupt at the end of this count. The user can choose to reload any number of the PWM buffers, and they will all update simultaneously at the start of a new period.

Writing to TPMxSC cancels any values written to TPMxMODH and/or TPMxMODL and resets the coherency mechanism for the modulo registers. Writing to TPMxCnSC cancels any values written to the channel value registers and resets the coherency mechanism for TPMxCnVH:TPMxCnVL.

11.5 TPM Interrupts

The TPM generates an optional interrupt for the main counter overflow and an interrupt for each channel. The meaning of channel interrupts depends on the mode of operation for each channel. 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. See the Resets, Interrupts, and System Configuration chapter for absolute interrupt vector addresses, priority, and local interrupt mask control bits.

For each interrupt source in the TPM, a flag bit is set on recognition of the interrupt condition such as timer overflow, channel input capture, or output compare events. This flag may be read (polled) by software to verify that the action has occurred, or an associated enable bit (TOIE or CHnIE) can be set to enable hardware interrupt generation. While the interrupt enable bit is set, a static interrupt will be generated whenever the associated interrupt flag equals 1. It is the responsibility of user software to perform a sequence of steps to clear the interrupt flag before returning from the interrupt service routine.

11.5.1 Clearing Timer Interrupt Flags

TPM interrupt flags are cleared by a 2-step process that includes a read of the flag bit while it is set (1) followed by a write of 0 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.

11.5.2 Timer Overflow Interrupt Description

The conditions that cause TOF to become set depend on the counting mode (up or up/down). In up-counting mode, the 16-bit timer counter counts from 0x0000 through 0xFFFF and overflows to 0x0000 on the next counting clock. TOF becomes set at the transition from 0xFFFF to 0x0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to 0x0000. When the counter is operating in up-/down-counting mode, the TOF flag gets set as the counter changes direction at the transition from the value set in the modulus register and 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.)

11.5.3 Channel Event Interrupt Description

The meaning of channel interrupts depends on the current mode of the channel (input capture, output compare, edge-aligned PWM, or center-aligned PWM).

When a channel is configured as an input capture channel, the ELSnB:ELSnA control bits select rising edges, falling edges, any edge, or no edge (off) 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 2-step sequence described in Section 11.5.1, “Clearing Timer Interrupt Flags.”

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 2-step sequence described in Section 11.5.1, “Clearing Timer Interrupt Flags.”

11.5.4 PWM End-of-Duty-Cycle Events

For channels that are configured for PWM operation, there are two possibilities:

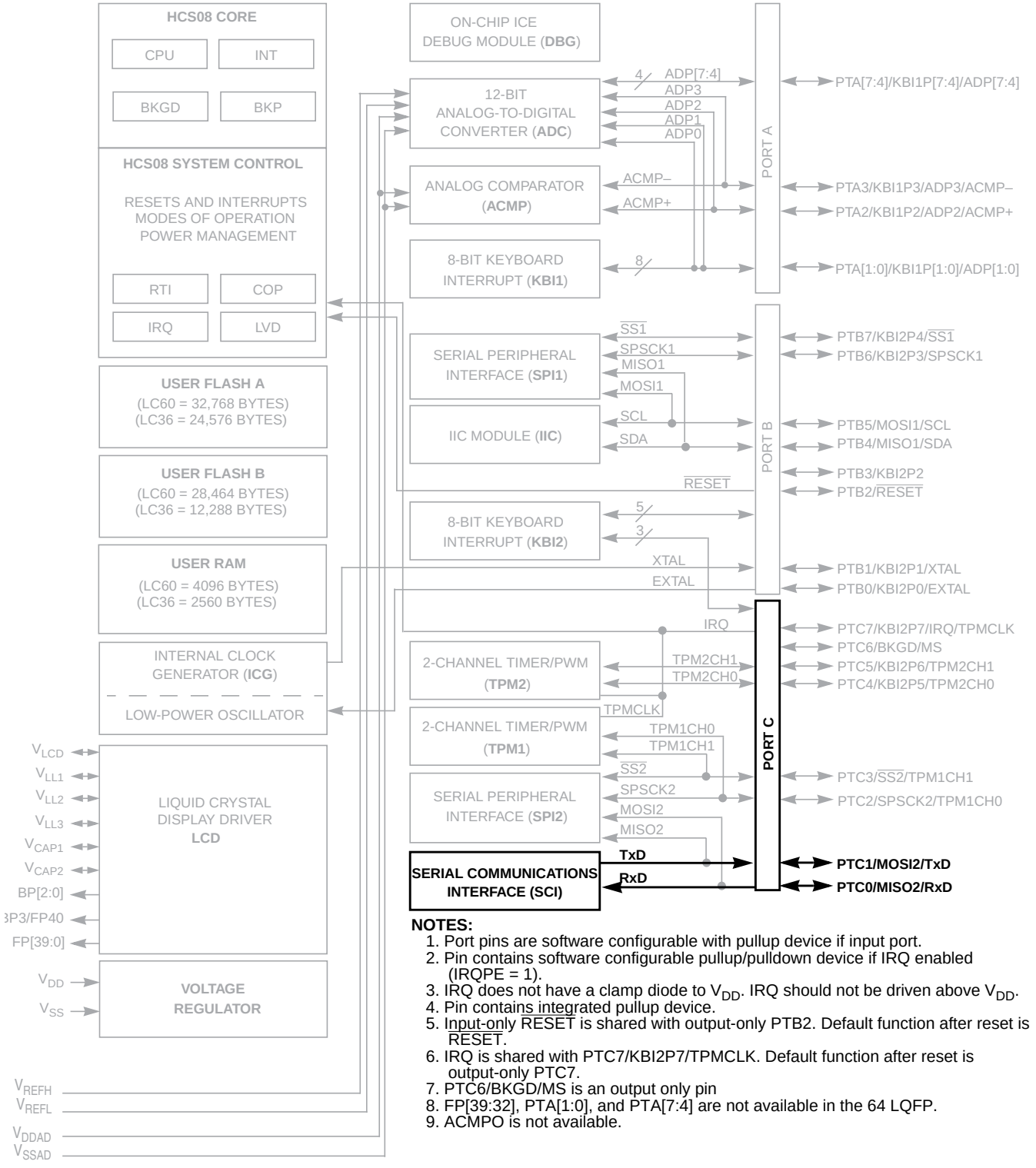
- When the channel is configured for edge-aligned PWM, the channel flag is set when the timer counter matches the channel value register that marks the end of the active duty cycle period.
- When the channel is configured for center-aligned PWM, the timer count matches the channel value register twice 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, which are the times when the timer counter matches the channel value register.

The flag is cleared by the 2-step sequence described in Section 11.5.1, “Clearing Timer Interrupt Flags.”

Chapter 12
Serial Communications Interface (S08SCIV3)

12.1 Introduction

Figure 12-1 shows the MC9S08LC60 Series block diagram with the SCI highlighted.



NOTES:

1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 12-1. MC9S08LC60 Series Block Diagram Highlighting SCI Block and Pins

Module Initialization:

Write:	SCIBDH:SCIBDL	to set	baud rate
Write:	SCFC1	to configure	1-wire/2-wire, 9/8-bit data, wakeup, and parity, if used.
Write;	SCIC2	to configure	interrupts, enable Rx and Tx, RWU Enable Rx wakeup, SBK sends break character
Write:	SCIC3	to enable	Rx error interrupt sources. Also controls pin direction in 1-wire modes. R8 and T8 only used in 9-bit data modes.

Module Use:

Wait for TDRE, then write data to SCID

Wait for RDRF, then read data from SCID

A small number of applications will use RWU to manage automatic receiver wakeup, SBK to send break characters, and R8 and T8 for 9-bit data.

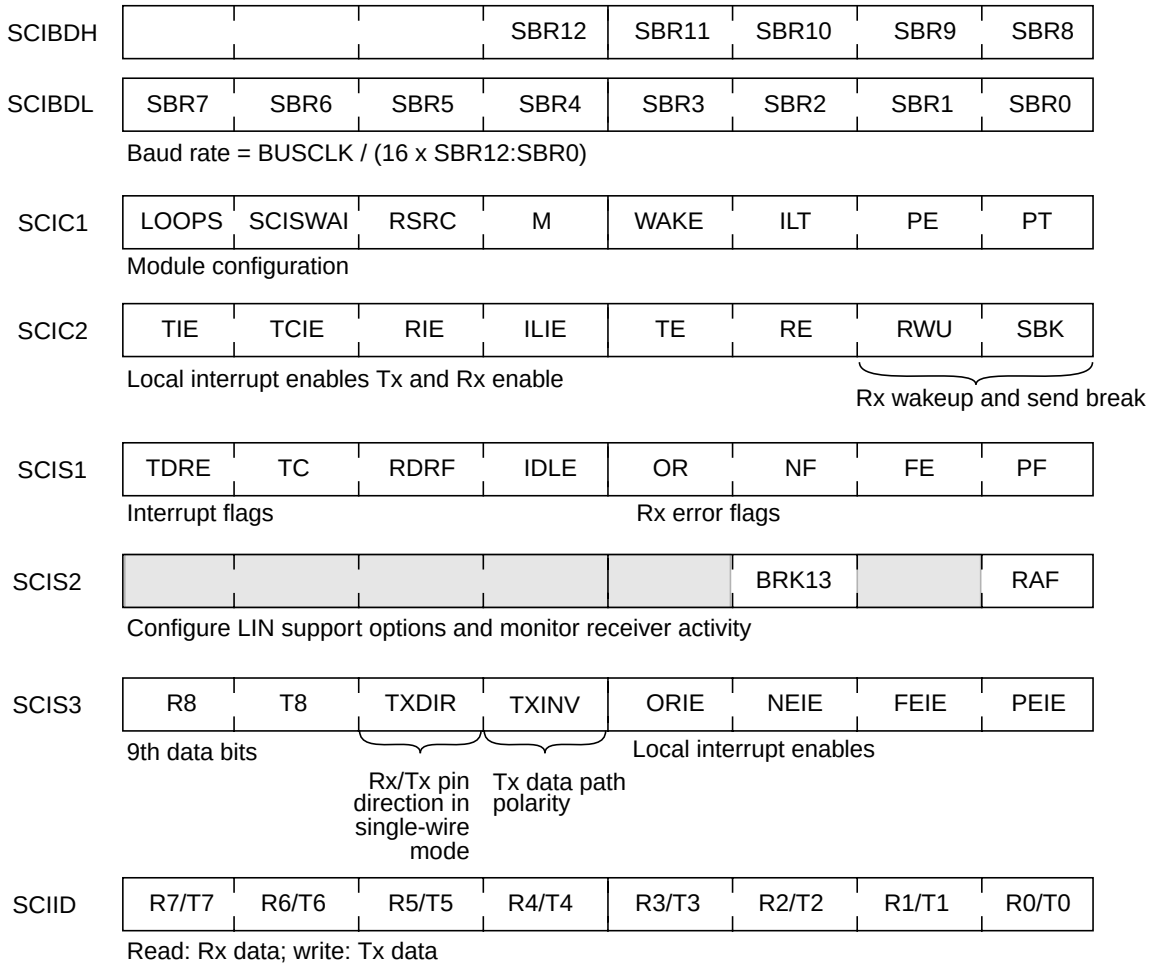


Figure 12-2. SCI Module Quick Start

12.1.1 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
- 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
- Selectable transmitter output polarity

12.1.2 Modes of Operation

See Section 12.3, “Functional Description,” for a detailed description of SCI operation in the different modes.

- 8- and 9- bit data modes
- Stop modes — SCI is halted during all stop modes
- Loop mode
- Single-wire mode

12.1.3 Block Diagram

Figure 12-3 shows the transmitter portion of the SCI. (Figure 12-4 shows the receiver portion of the SCI.)

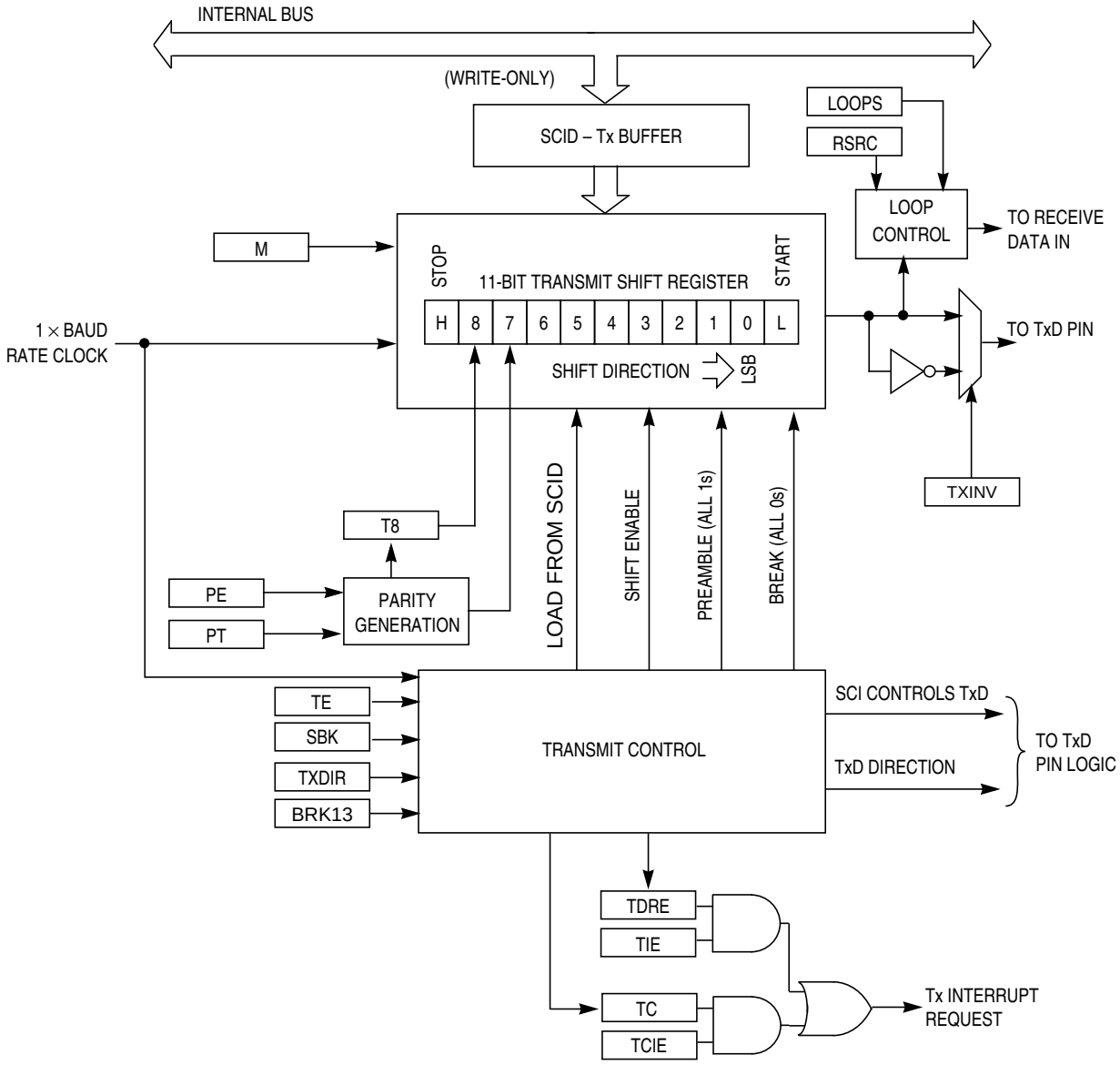


Figure 12-3. SCI Transmitter Block Diagram

Figure 12-4 shows the receiver portion of the SCI.

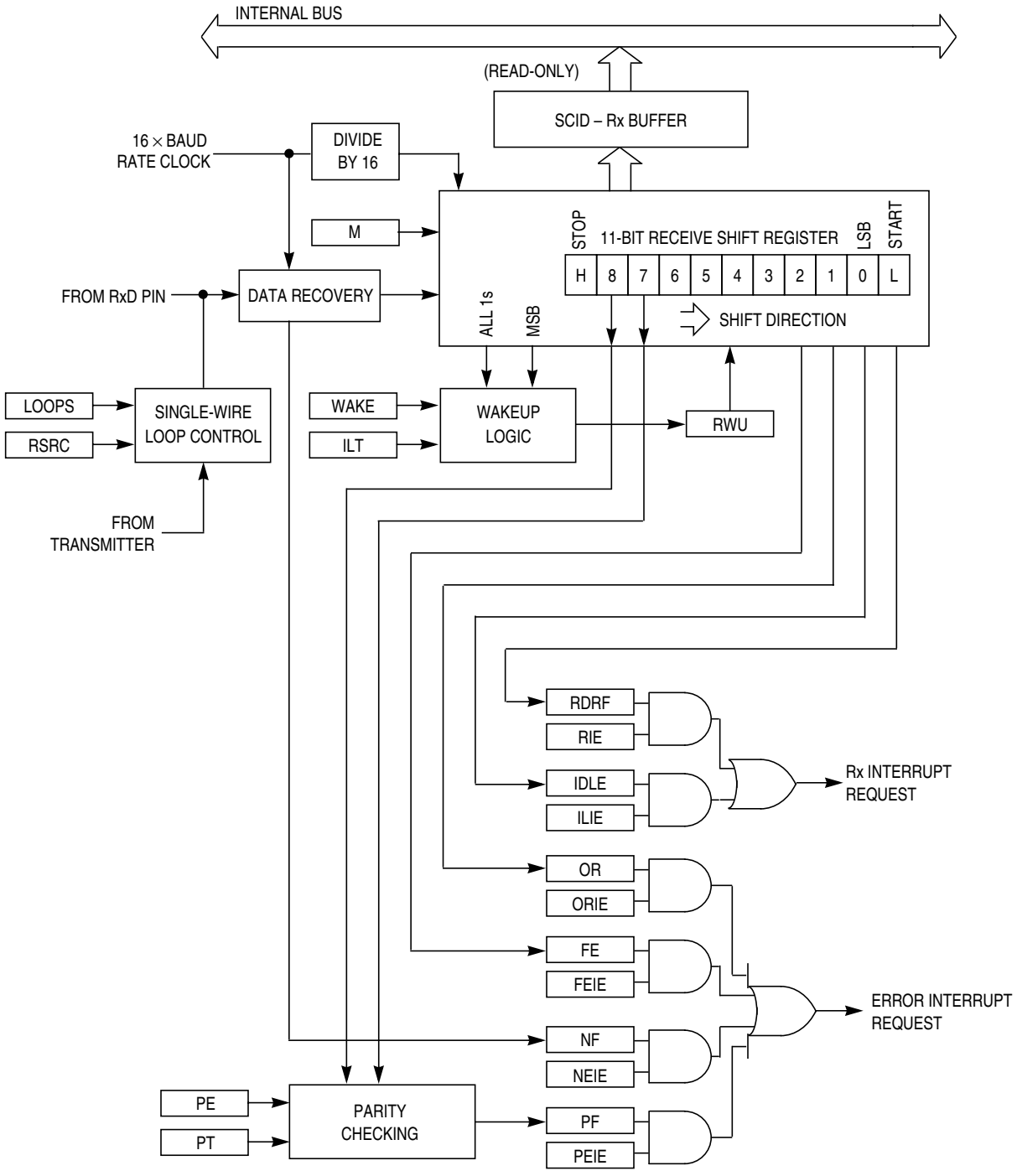


Figure 12-4. SCI Receiver Block Diagram

12.2 Register Definition

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

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

12.2.1 SCI Baud Rate Registers (SCIBDH, SCIBHL)

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

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

	7	6	5	4	3	2	1	0
R	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-5. SCI Baud Rate Register (SCIBDH)

Table 12-1. SCIBDH Register Field Descriptions

Field	Description
4:0 SBR[12:8]	**Baud Rate Modulo Divisor** — These 13 bits are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR = 0, the SCI baud rate generator is disabled to reduce supply current. When BR = 1 to 8191, the SCI baud rate = BUSCLK/(16×BR). See also BR bits in Table 12-2.

	7	6	5	4	3	2	1	0
R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
W								
Reset	0	0	0	0	0	1	0	0

Figure 12-6. SCI Baud Rate Register (SCIBDL)

Table 12-2. SCIBDL Register Field Descriptions

Field	Description
7:0 SBR[7:0]	**Baud Rate Modulo Divisor** — These 13 bits are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR = 0, the SCI baud rate generator is disabled to reduce supply current. When BR = 1 to 8191, the SCI baud rate = BUSCLK/(16×BR). See also BR bits in Table 12-1.

12.2.2 SCI Control Register 1 (SCIC1)

This read/write register is used to control 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 12-7. SCI Control Register 1 (SCIC1)

Table 12-3. SCIC1 Register Field Descriptions

Field	Description
7 LOOPS	**Loop Mode Select** — Selects between loop back modes and normal 2-pin full-duplex modes. When LOOPS = 1, 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 = 1, 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 = 1, RSRC = 0 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 12.3.3.2, “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 the logic high level by the idle line detection logic. Refer to Section 12.3.3.2.1, “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 12-3. SCIC1 Register 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.

12.2.3 SCI Control Register 2 (SCIC2)

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 12-8. SCI Control Register 2 (SCIC2)

Table 12-4. SCIC2 Register 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.
3 TE	**Transmitter Enable** 0 Transmitter off. 1 Transmitter on. TE must be 1 in order to use the SCI transmitter. When TE = 1, 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 also can be used to queue an idle character by writing TE = 0 then TE = 1 while a transmission is in progress. Refer to Section 12.3.2.1, “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.

Table 12-4. SCIC2 Register Field Descriptions (continued)

Field	Description
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 = 1, the RxD pin reverts to being a general-purpose I/O pin even if RE = 1. 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 either 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 12.3.3.2, “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 bit times of logic 0 are queued as long as SBK = 1. 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 12.3.2.1, “Send Break and Queued Idle,” for more details. 0 Normal transmitter operation. 1 Queue break character(s) to be sent.

12.2.4 SCI Status Register 1 (SCIS1)

This register has eight read-only status flags. Writes have no effect. Special software sequences (which do not involve writing to this register) are used to 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

= Unimplemented or Reserved

Figure 12-9. SCI Status Register 1 (SCIS1)

Table 12-5. SCIS1 Register 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 SCIS1 with TDRE = 1 and then write to the SCI data register (SCID). 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 = 1 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 SCIS1 with TC = 1 and then doing one of the following three things: • Write to the SCI data register (SCID) to transmit new data • Queue a preamble by changing TE from 0 to 1 • Queue a break character by writing 1 to SBK in SCIC2

Table 12-5. SCIS1 Register Field Descriptions (continued)

Field	Description
5 RDRF	**Receive Data Register Full Flag** — RDRF becomes set when a character transfers from the receive shifter into the receive data register (SCID). To clear RDRF, read SCIS1 with RDRF = 1 and then read the SCI data register (SCID). 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 = 0, the receiver starts counting idle bit times after the start bit. So 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 = 1, the receiver doesn't start counting idle bit times until after the stop bit. So 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 SCIS1 with IDLE = 1 and then read the SCI data register (SCID). After IDLE has been cleared, it cannot become set again until after a new character has been received and RDRF has been set. IDLE will get 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 SCID yet. In this case, the new character (and all associated error information) is lost because there is no room to move it into SCID. To clear OR, read SCIS1 with OR = 1 and then read the SCI data register (SCID). 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 will be set at the same time as the flag RDRF gets set for the character. To clear NF, read SCIS1 and then read the SCI data register (SCID). 0 No noise detected. 1 Noise detected in the received character in SCID.
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 SCIS1 with FE = 1 and then read the SCI data register (SCID). 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 SCIS1 and then read the SCI data register (SCID). 0 No parity error. 1 Parity error.

12.2.5 SCI Status Register 2 (SCIS2)

This register has one read-only status flag. Writes have no effect.

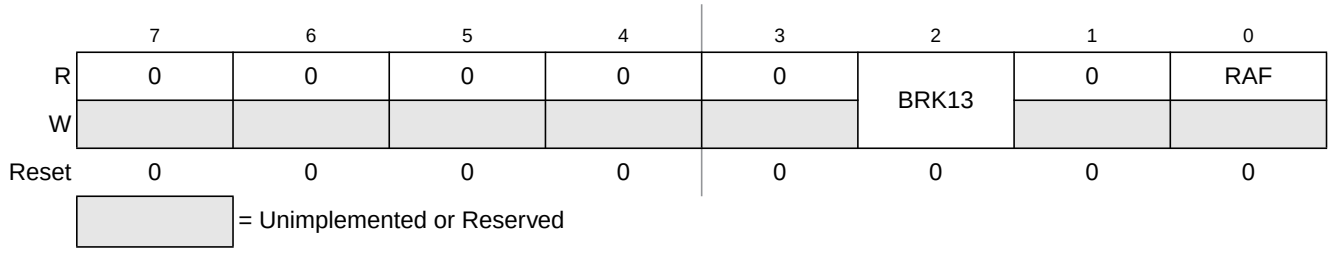


Figure 12-10. SCI Status Register 2 (SCIS2)

Table 12-6. SCIS2 Register Field Descriptions

Field	Description
2 BRK13	**Break Character Length** — BRK13 is used to select a longer break character length. Detection of a framing error is not affected by the state of this bit. 0 Break character is 10 bit times (11 if M = 1) 1 Break character is 13 bit times (14 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).

12.2.6 SCI Control Register 3 (SCIC3)

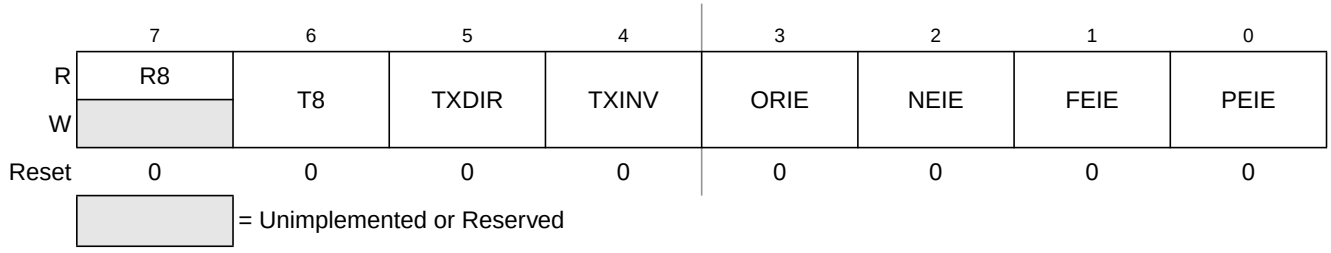


Figure 12-11. SCI Control Register 3 (SCIC3)

Table 12-7. SCIC3 Register 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 SCID register. When reading 9-bit data, read R8 before reading SCID because reading SCID completes automatic flag clearing sequences which could allow R8 and SCID 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 SCID register. When writing 9-bit data, the entire 9-bit value is transferred to the SCI shift register after SCID is written so T8 should be written (if it needs to change from its previous value) before SCID 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 SCID is written.

Table 12-7. SCIC3 Register Field Descriptions (continued)

Field	Description
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.
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 = 1.
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 = 1.
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 = 1.
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 = 1.

[1] Setting TXINV inverts the TxD output for all cases: data bits, start and stop bits, break, and idle.

12.2.7 SCI Data Register (SCID)

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 12-12. SCI Data Register (SCID)

12.3 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.

12.3.1 Baud Rate Generation

As shown in Figure 12-13, the clock source for the SCI baud rate generator is the bus-rate clock.

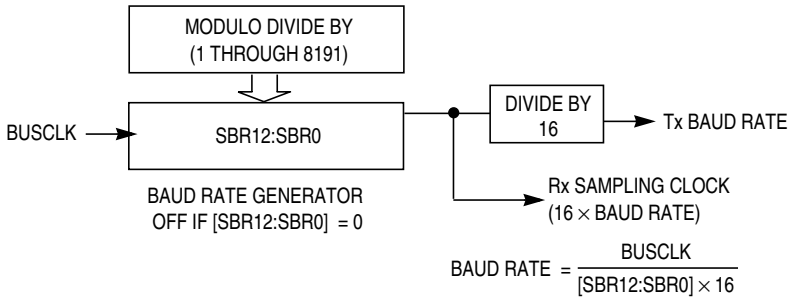


Figure 12-13. SCI Baud Rate Generation

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, but 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.

12.3.2 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 12-3.

The transmitter output (TxD) idle state defaults to logic high (TXINV = 0 following reset). The transmitter output is inverted by setting TXINV = 1. The transmitter is enabled by setting the TE bit in SCIC2. 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 (SCID).

The central element of the SCI transmitter is the transmit shift register that is either 10 or 11 bits long depending on the setting in the M control bit. For the remainder of this section, we will assume M = 0,

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 SCID.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the TxD1 pin, the transmitter sets the transmit complete flag and enters an idle mode, with TxD1 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 that is in progress must first be completed. This includes data characters in progress, queued idle characters, and queued break characters.

12.3.2.1 Send Break and Queued Idle

The SBK control bit in SCIC2 is used to send break characters which were 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 = 1. 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 is still 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 will be 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 = 0, the SCI transmitter never actually releases control of the TxD1 pin. If there is a possibility of the shifter finishing while TE = 0, set the general-purpose I/O controls so the pin that is shared with TxD1 is an output driving a logic 1. This ensures that the TxD1 line will look 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 12-8. 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

12.3.3 Receiver Functional Description

In this section, the receiver block diagram (Figure 12-4) is used as 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 is enabled by setting the RE bit in SCIC2. 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 12.4.1, “8- and 9-Bit Data Modes.” For the remainder of this discussion, we assume the SCI is configured for normal 8-bit data mode.

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

When a program detects that the receive data register is full (RDRF = 1), it gets the data from the receive data register by reading SCID. The RDRF flag is cleared automatically by a 2-step sequence which is normally satisfied in the course of the user’s program that handles receive data. Refer to Section 12.3.4, “Interrupts and Status Flags,” for more details about flag clearing.

12.3.3.1 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 RxD1 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 is used to divide the bit time into 16 segments labeled RT1 through RT16. When a falling edge is located, three more samples are taken at RT3, RT5, and RT7 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a receive character.

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

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

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

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

12.3.3.2 Receiver Wakeup Operation

Receiver wakeup is a hardware mechanism that allows an SCI receiver to ignore the characters in a message that is intended for a different SCI receiver. In such a system, all receivers evaluate the first 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 SCIC2. When RWU = 1, it inhibits setting of the status flags associated with the receiver, 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.

12.3.3.2.1 Idle-Line Wakeup

When WAKE = 0, 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 the RWU bit is set, the idle character that wakes a receiver does not set the receiver idle bit, IDLE, or the receive data register full flag, RDRF. It therefore will not generate an interrupt when this idle character occurs. The receiver will wake up and wait for the next data transmission which will set RDRF and generate an interrupt if enabled.

The idle-line type (ILT) control bit selects one of two ways to detect an idle line. When ILT = 0, 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 = 1, 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.

12.3.3.2.2 Address-Mark Wakeup

When WAKE = 1, 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 in M = 0 mode and ninth bit in M = 1 mode).

Address-mark wakeup allows messages to contain idle characters but requires that the MSB be reserved for use in address frames. The logic 1 MSB of an address frame clears the receivers RWU bit before the stop bit is received and sets the RDRF flag.

12.3.4 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 and IDLE events, and a third vector is used for OR, NF, FE, and PF error conditions. Each of these eight interrupt sources can be separately

masked by local interrupt enable masks. The flags can still 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 optionally can 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 SCID. If the transmit interrupt enable (TIE) bit is set, a hardware interrupt will be requested whenever TDRE = 1. Transmit complete (TC) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TxD1 high. 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 will be requested whenever TC = 1. 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 0s.

When a program detects that the receive data register is full (RDRF = 1), it gets the data from the receive data register by reading SCID. The RDRF flag is cleared by reading SCIS1 while RDRF = 1 and then reading SCID.

When polling is used, this sequence is naturally satisfied in the normal course of the user program. If hardware interrupts are used, SCIS1 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 RxD1 line remains idle for an extended period of time. IDLE is cleared by reading SCIS1 while IDLE = 1 and then reading SCID. 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) — get 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 gets set instead and the data and any associated NF, FE, or PF condition is lost.

12.4 Additional SCI Functions

The following sections describe additional SCI functions.

12.4.1 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 SCIC1. 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 SCIC3. For the receiver, the ninth bit is held in R8 in SCIC3.

For coherent writes to the transmit data buffer, write to the T8 bit before writing to SCID.

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 SCID to the shifter.

9-bit data mode typically is used in conjunction 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.

12.4.2 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.

Note, because the clocks are halted, the SCI module will resume 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.

12.4.3 Loop Mode

When LOOPS = 1, 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 RxD1 pin is not used by the SCI, so it reverts to a general-purpose port I/O pin.

12.4.4 Single-Wire Operation

When LOOPS = 1, the RSRC bit in the same register chooses between loop mode (RSRC = 0) or single-wire mode (RSRC = 1). Single-wire mode is used to implement a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TxD1 pin. The RxD1 pin is not used and reverts to a general-purpose port I/O pin.

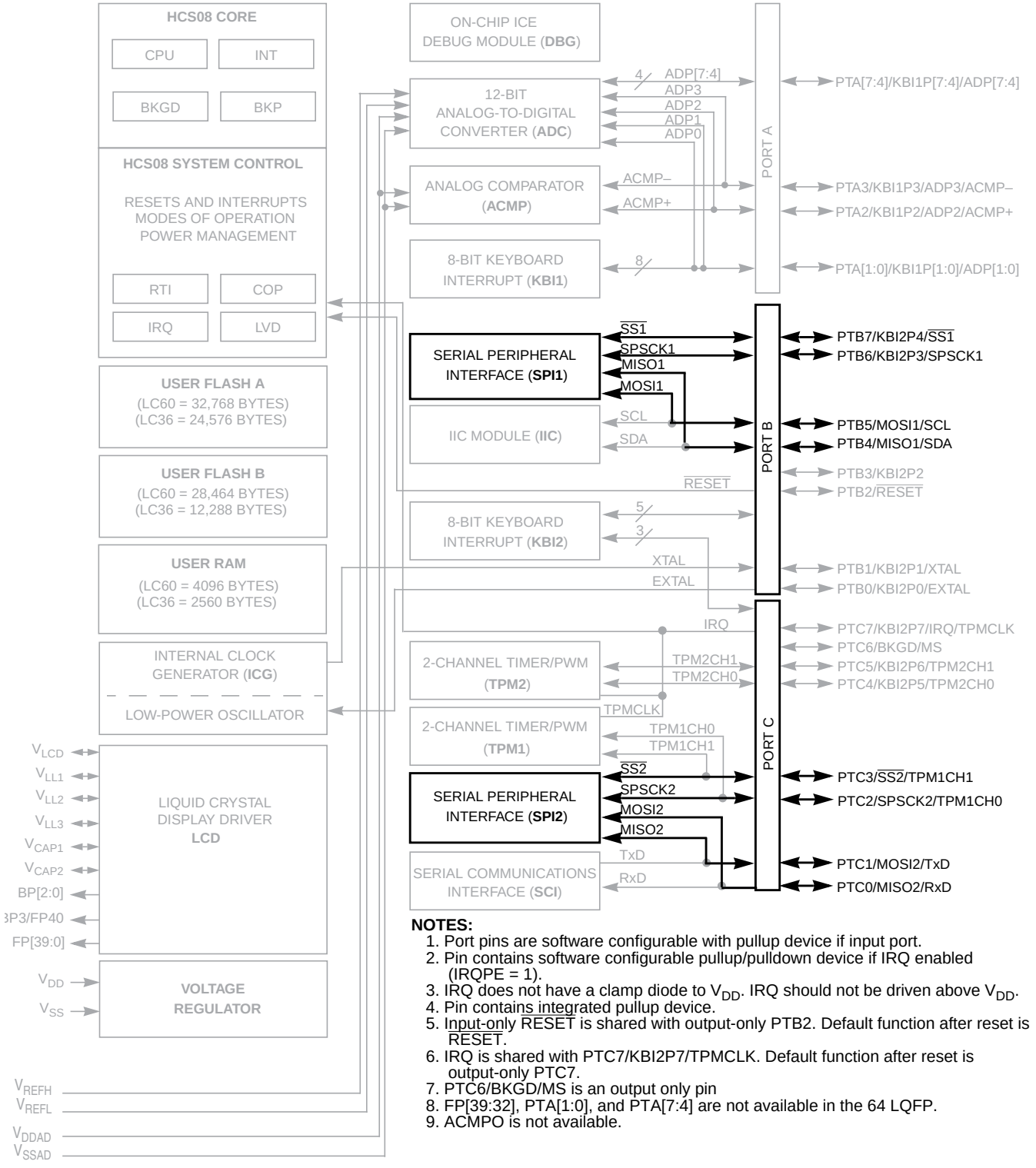
In single-wire mode, the TXDIR bit in SCIC3 controls the direction of serial data on the TxD1 pin. When TXDIR = 0, the TxD1 pin is an input to the SCI receiver and the transmitter is temporarily disconnected from the TxD1 pin so an external device can send serial data to the receiver. When TXDIR = 1, the TxD1 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 13
Serial Peripheral Interface (S08SPIV3)

13.1 Introduction

The MC9S08LC60 Series contains two SPI modules.

Figure 13-1 shows the MC9S08LC60 Series block diagram with the SPIs highlighted.



NOTES:
1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 13-1. MC9S08LC60 Series Block Diagram Highlighting SPI Blocks and Pins

13.1.1 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

13.1.2 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.

13.1.2.1 SPI System Block Diagram

Figure 13-2 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.

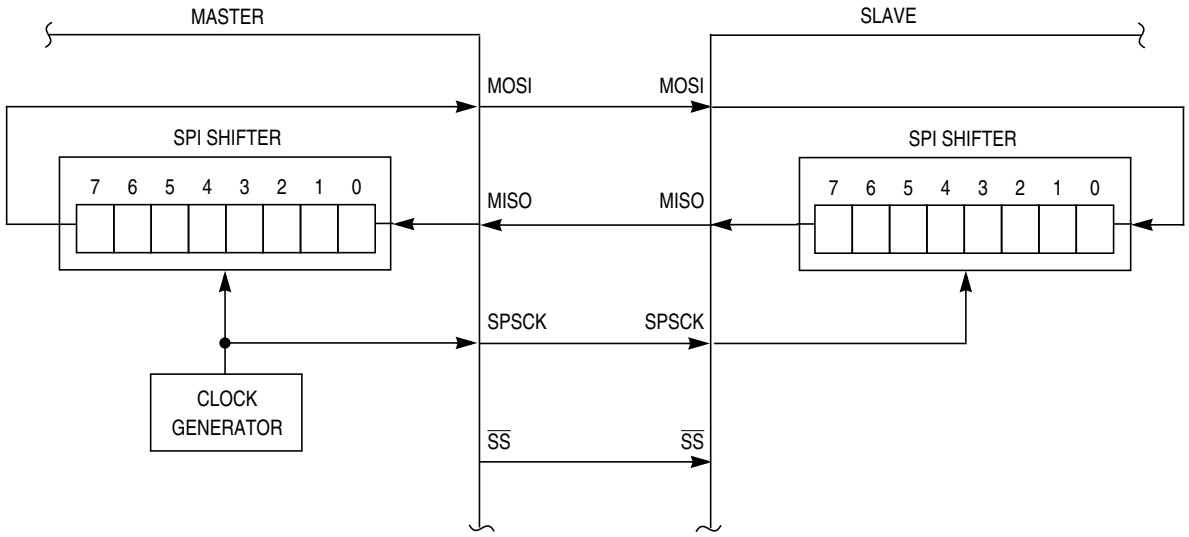


Figure 13-2. SPI System Connections

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 13-2 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.

13.1.2.2 SPI Module Block Diagram

Figure 13-3 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 SPIxD) 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 SPIxD). 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.

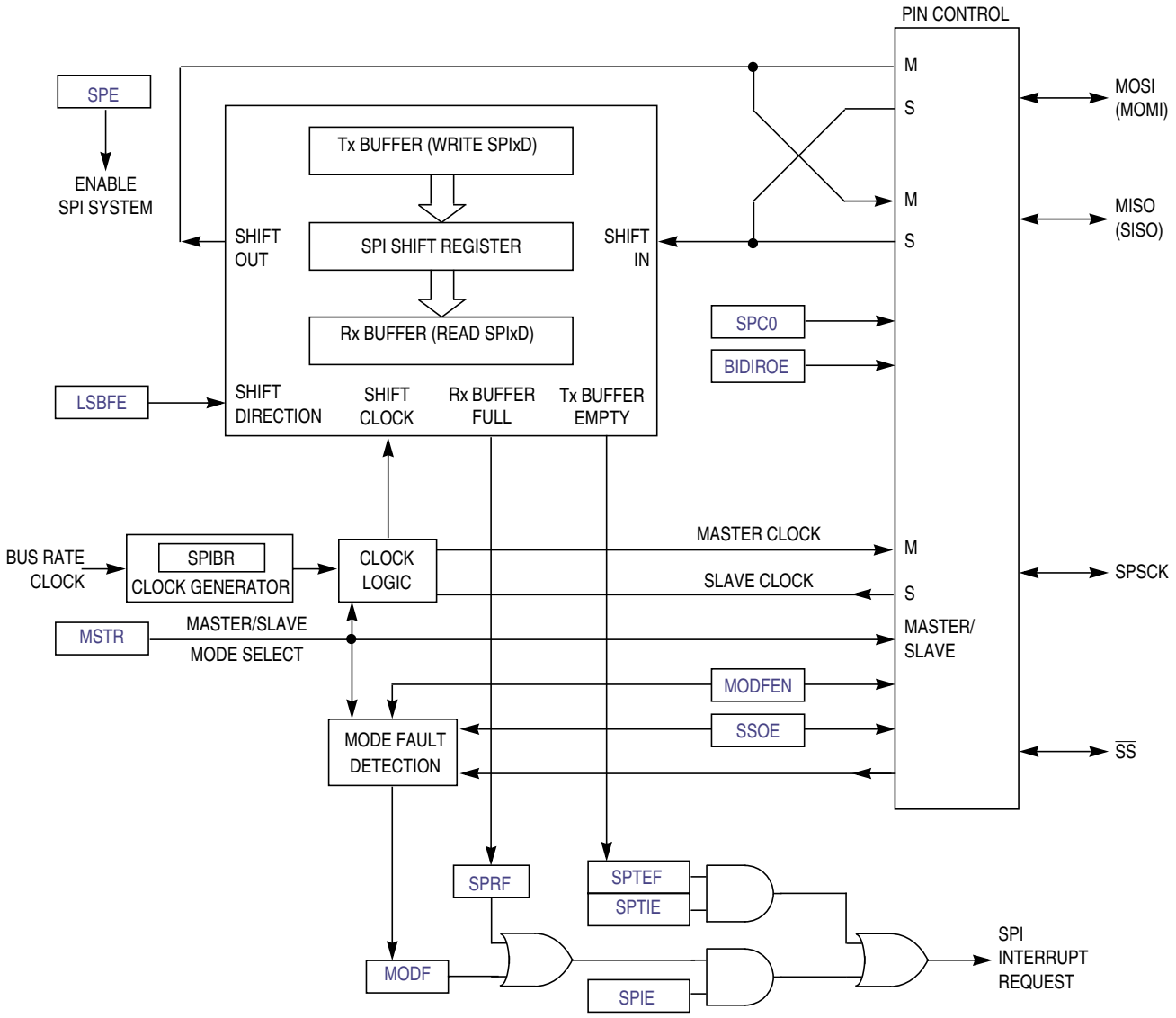


Figure 13-3. SPI Module Block Diagram

13.1.3 SPI Baud Rate Generation

As shown in Figure 13-4, 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 (SPR2:SPR1:SPR0) divide the output of the prescaler stage by 2, 4, 8, 16, 32, 64, 128, or 256 to get the internal SPI master mode bit-rate clock.

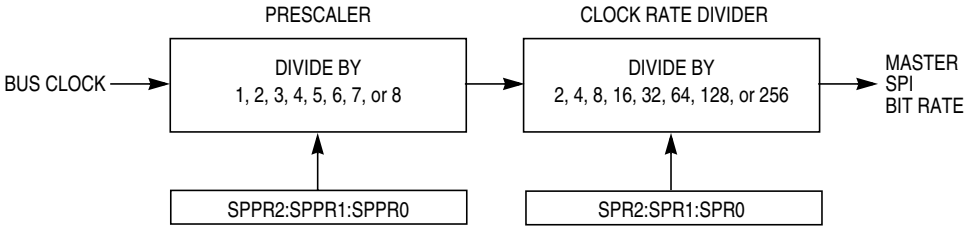


Figure 13-4. SPI Baud Rate Generation

13.2 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.

13.2.1 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.

13.2.2 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.

13.2.3 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.

13.2.4 $\overline{SS}$ — 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).

13.3 Modes of Operation

13.3.1 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.

13.4 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.

13.4.1 SPI Control Register 1 (SPIxC1)

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 13-5. SPI Control Register 1 (SPIxC1)

Table 13-1. SPIxC1 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 13-1. SPIxC1 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 13.5.1, "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 13.5.1, "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 13-2.
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 13-2. $\overline{SS}$ 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.

13.4.2 SPI Control Register 2 (SPIxC2)

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 13-6. SPI Control Register 2 (SPIxC2)

Table 13-3. SPIxC2 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 13-2 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

13.4.3 SPI Baud Rate Register (SPIxBR)

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.

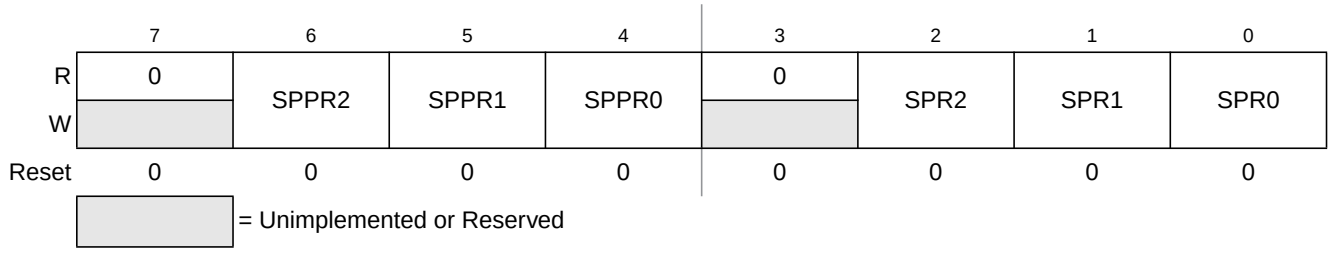


Figure 13-7. SPI Baud Rate Register (SPIxBR)

Table 13-4. SPIxBR 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 13-5. 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 13-4).
2:0 SPR[2:0]	**SPI Baud Rate Divisor** — This 3-bit field selects one of eight divisors for the SPI baud rate divider as shown in Table 13-6. The input to this divider comes from the SPI baud rate prescaler (see Figure 13-4). The output of this divider is the SPI bit rate clock for master mode.

Table 13-5. 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 13-6. SPI Baud Rate Divisor

SPR2:SPR1:SPR0	Rate Divisor
0:0:0	2
0:0:1	4
0:1:0	8
0:1:1	16
1:0:0	32
1:0:1	64
1:1:0	128
1:1:1	256

13.4.4 SPI Status Register (SPIxS)

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 13-8. SPI Status Register (SPIxS)

Table 13-7. SPIxS 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 (SPIxD). 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 SPIxS with SPTEF set, followed by writing a data value to the transmit buffer at SPIxD. SPIxS must be read with SPTEF = 1 before writing data to SPIxD or the SPIxD write will be ignored. SPTEF generates an SPTEF CPU interrupt request if the SPTIE bit in the SPIxC1 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 SPIxD 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 (SPIxC1). 0 No mode fault error 1 Mode fault error detected

13.4.5 SPI Data Register (SPIxD)

	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 13-9. SPI Data Register (SPIxD)

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 SPIxD 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.

13.5 Functional Description

An SPI transfer is initiated by checking for the SPI transmit buffer empty flag (SPTEF = 1) and then writing a byte of data to the SPI data register (SPIxD) in the master SPI device. When the SPI shift register is available, this byte of data is moved from the transmit data buffer to the shifter, SPTEF is set to indicate there is room in the buffer to queue another transmit character if desired, and the SPI serial transfer starts.

During the SPI transfer, data is sampled (read) on the MISO pin at one SPSCK edge and shifted, changing the bit value on the MOSI pin, one-half SPSCK cycle later. After eight SPSCK cycles, the data that was in the shift register of the master has been shifted out the MOSI pin to the slave while eight bits of data were shifted in the MISO pin into the master's shift register. At the end of this transfer, the received data byte is moved from the shifter into the receive data buffer and SPRF is set to indicate the data can be read by reading SPIxD. If another byte of data is waiting in the transmit buffer at the end of a transfer, it is moved into the shifter, SPTEF is set, and a new transfer is started.

Normally, SPI data is transferred most significant bit (MSB) first. If the least significant bit first enable (LSBFE) bit is set, SPI data is shifted LSB first.

When the SPI is configured as a slave, its $\overline{SS}$ pin must be driven low before a transfer starts and $\overline{SS}$ must stay low throughout the transfer. If a clock format where CPHA = 0 is selected, $\overline{SS}$ must be driven to a logic 1 between successive transfers. If CPHA = 1, $\overline{SS}$ may remain low between successive transfers. See Section 13.5.1, "SPI Clock Formats" for more details.

Because the transmitter and receiver are double buffered, a second byte, in addition to the byte currently being shifted out, can be queued into the transmit data buffer, and a previously received character can be in the receive data buffer while a new character is being shifted in. The SPTEF flag indicates when the transmit buffer has room for a new character. The SPRF flag indicates when a received character is available in the receive data buffer. The received character must be read out of the receive buffer (read SPIxD) before the next transfer is finished or a receive overrun error results.

In the case of a receive overrun, the new data is lost because the receive buffer still held the previous character and was not ready to accept the new data. There is no indication for such an overrun condition so the application system designer must ensure that previous data has been read from the receive buffer before a new transfer is initiated.

13.5.1 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 13-10 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.

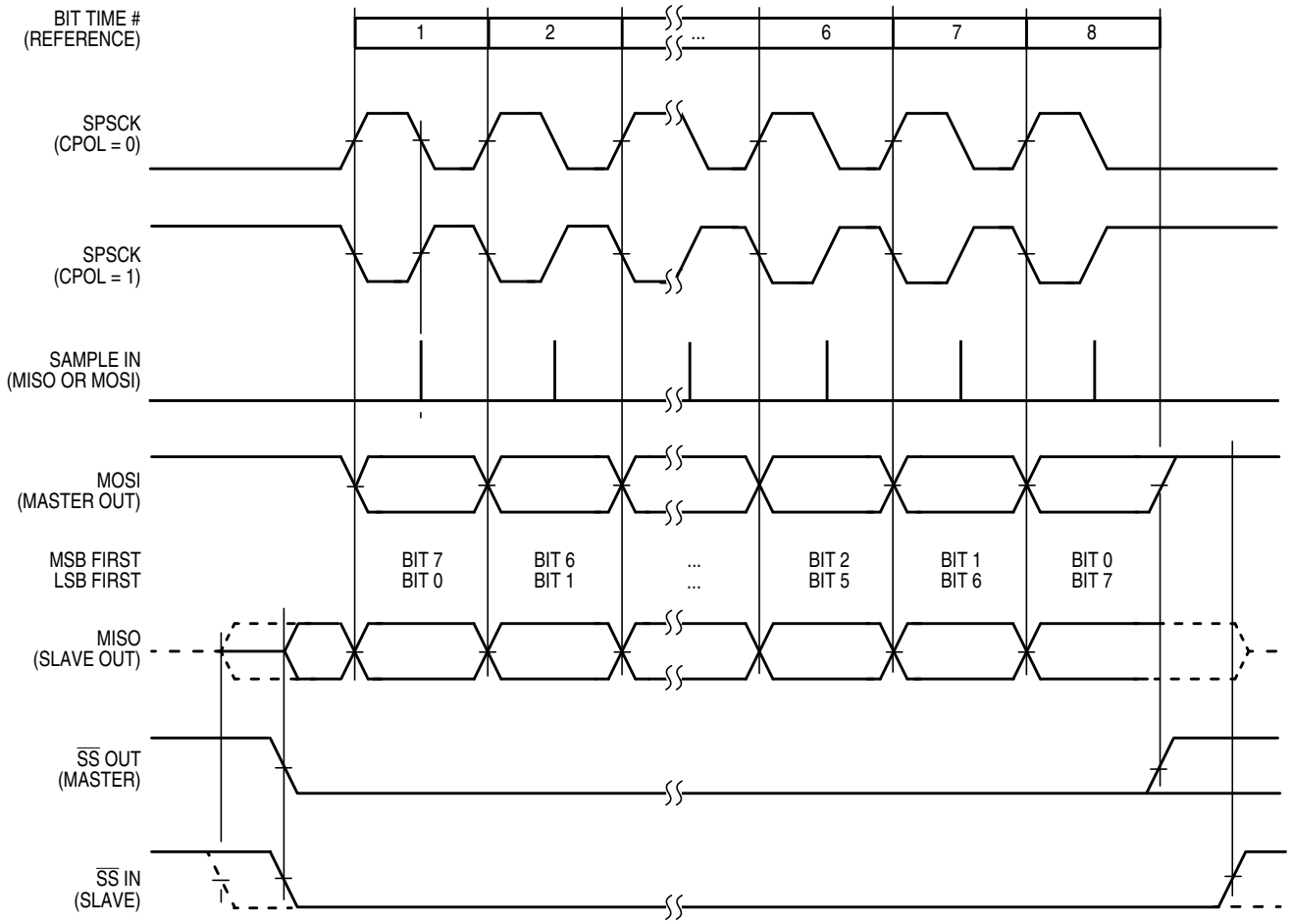


Figure 13-10. SPI Clock Formats (CPHA = 1)

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 13-11 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.

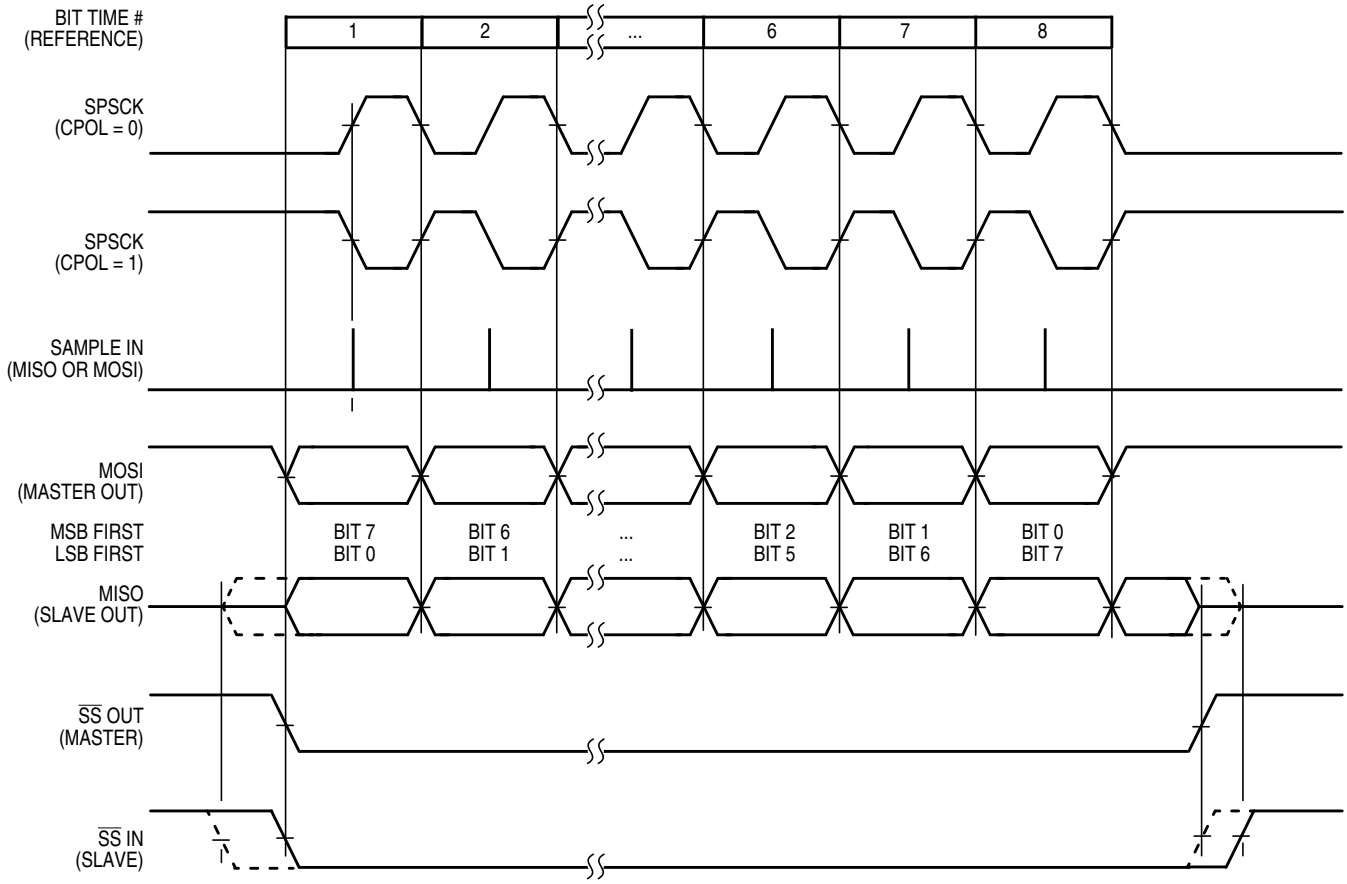


Figure 13-11. SPI Clock Formats (CPHA = 0)

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.

13.5.2 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).

13.5.3 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.

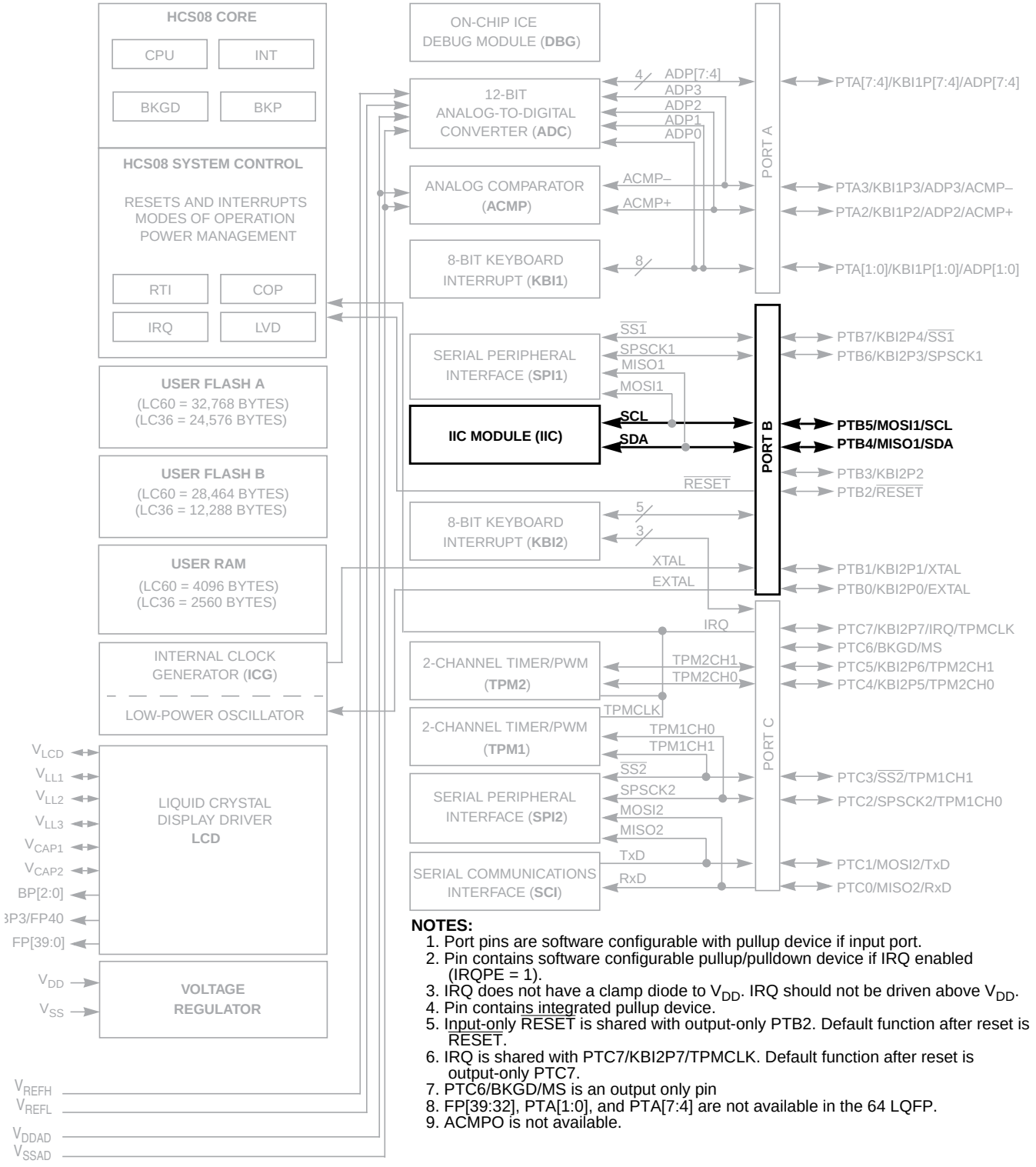
MODF is cleared by reading it while it is set, then writing to the SPI control register 1 (SPIxC1). User software should verify the error condition has been corrected before changing the SPI back to master mode.

Chapter 14
Inter-Integrated Circuit (S08IICV1)

14.1 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.

Figure 14-1 is the MC9S08LC60 Series block diagram with the IIC block highlighted.



NOTES:
1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only RESET is shared with output-only PTB2. Default function after reset is RESET.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 14-1. MC9S08LC60 Series Block Diagram Highlighting IIC Block and Pins

14.1.1 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

14.1.2 Modes of Operation

The IIC functions the same in normal and monitor modes. 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 will continue 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 and stop1 will reset the register contents.

14.1.3 Block Diagram

Figure 14-2 is a block diagram of the IIC.

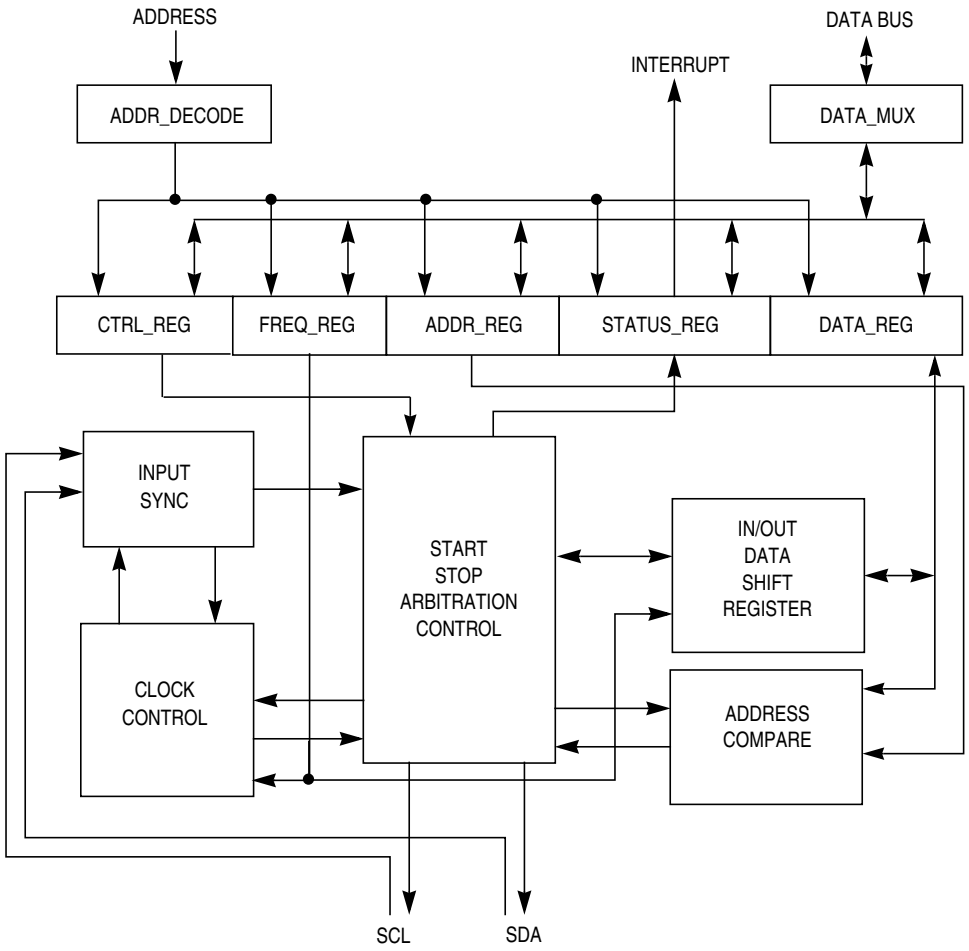


Figure 14-2. IIC Functional Block Diagram

14.2 External Signal Description

This section describes each user-accessible pin signal.

14.2.1 SCL — Serial Clock Line

The bidirectional SCL is the serial clock line of the IIC system.

14.2.2 SDA — Serial Data Line

The bidirectional SDA is the serial data line of the IIC system.

14.3 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 data sheet 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.

14.3.1 IIC Address Register (IICA)

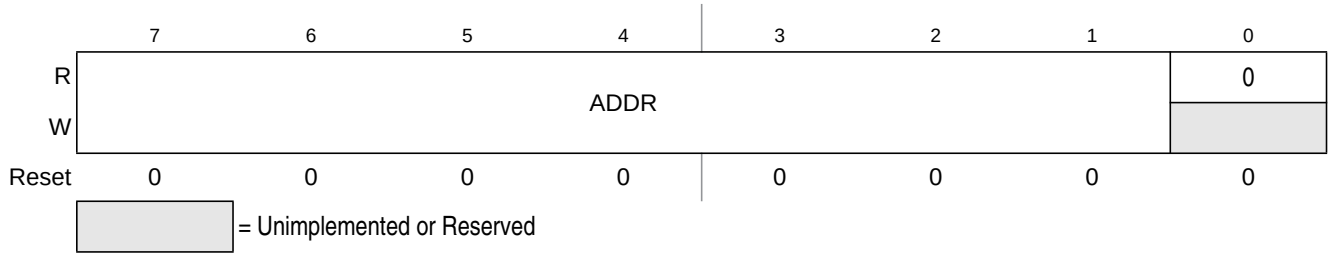


Figure 14-3. IIC Address Register (IICA)

Table 14-1. IICA Register Field Descriptions

Field	Description
7:1 ADDR[7:1]	**IIC Address Register** — The ADDR contains the specific slave address to be used by the IIC module. This is the address the module will respond to when addressed as a slave.

14.3.2 IIC Frequency Divider Register (IICF)

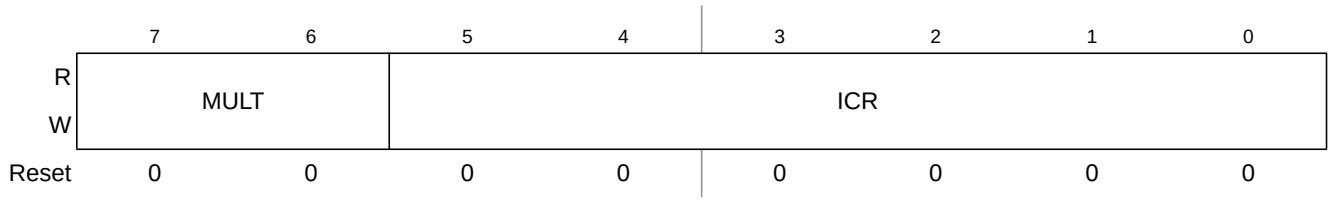


Figure 14-4. IIC Frequency Divider Register (IICF)

Table 14-2. IICA Register Field Descriptions

Field	Description
7:6 MULT	**IIC Multiplier Factor** — The MULT bits define the multiplier factor mul. This factor is used along with the SCL divider to generate 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 are used to define the SCL divider and the SDA hold value. The SCL divider multiplied by the value provided by the MULT register (multiplier factor mul) is used to generate IIC baud rate. IIC baud rate = bus speed (Hz)/(mul * SCL divider) SDA hold time is the delay from the falling edge of the SCL (IIC clock) to the changing of SDA (IIC data). The ICR is used to determine the SDA hold value. SDA hold time = bus period (s) * SDA hold value Table 14-3 provides the SCL divider and SDA hold values for corresponding values of the ICR. These values can be used to set IIC baud rate and SDA hold time. For example: Bus speed = 8 MHz MULT is set to 01 (mul = 2) Desired IIC baud rate = 100 kbps IIC baud rate = bus speed (Hz)/(mul * SCL divider) 100000 = 8000000/(2*SCL divider) SCL divider = 40 Table 14-3 shows that ICR must be set to 0B to provide an SCL divider of 40 and that this will result in an SDA hold value of 9. SDA hold time = bus period (s) * SDA hold value SDA hold time = 1/8000000 * 9 = 1.125 µs If the generated SDA hold value is not acceptable, the MULT bits can be used to change the ICR. This will result in a different SDA hold value.

Table 14-3. IIC Divider and Hold Values

ICR (hex)	SCL Divider	SDA Hold Value
00	20	7
01	22	7
02	24	8
03	26	8
04	28	9
05	30	9
06	34	10
07	40	10
08	28	7
09	32	7
0A	36	9
0B	40	9
0C	44	11
0D	48	11
0E	56	13
0F	68	13
10	48	9
11	56	9
12	64	13
13	72	13
14	80	17
15	88	17
16	104	21
17	128	21
18	80	9
19	96	9
1A	112	17
1B	128	17
1C	144	25
1D	160	25
1E	192	33
1F	240	33

ICR (hex)	SCL Divider	SDA Hold Value
20	160	17
21	192	17
22	224	33
23	256	33
24	288	49
25	320	49
26	384	65
27	480	65
28	320	33
29	384	33
2A	448	65
2B	512	65
2C	576	97
2D	640	97
2E	768	129
2F	960	129
30	640	65
31	768	65
32	896	129
33	1024	129
34	1152	193
35	1280	193
36	1536	257
37	1920	257
38	1280	129
39	1536	129
3A	1792	257
3B	2048	257
3C	2304	385
3D	2560	385
3E	3072	513
3F	3840	513

14.3.3 IIC Control Register (IICC)

	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 14-5. IIC Control Register (IICC)

Table 14-4. IICC Register 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 is changed 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 will always be 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 both master and slave receivers. 0 An acknowledge signal will be sent out to the bus after receiving one data byte. 1 No acknowledge signal response is sent.
2 RSTA	**Repeat START** — Writing a one to this bit will generate a repeated START condition provided it is the current master. This bit will always be read as a low. Attempting a repeat at the wrong time will result in loss of arbitration.

14.3.4 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 14-6. IIC Status Register (IICS)

Table 14-5. IICS Register Field Descriptions

Field	Description
7 TCF	**Transfer Complete Flag** — This bit is set on the completion of a byte transfer. Note that 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. 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 one 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 one 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.

14.3.5 IIC Data I/O Register (IICD)

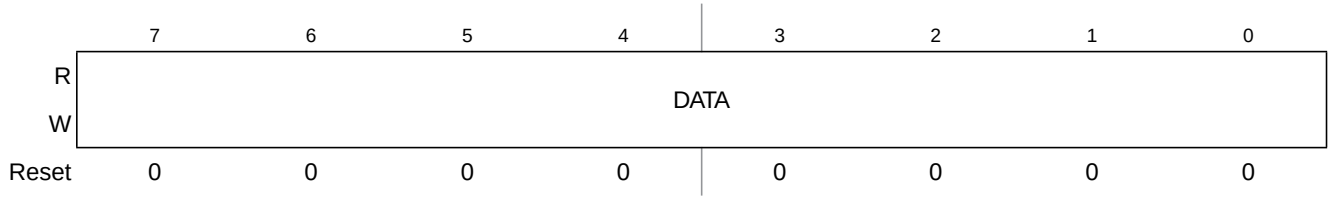


Figure 14-7. IIC Data I/O Register (IICD)

Table 14-6. IICD Register 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 transmitting 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.

Note that 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, then reading the IICD will not initiate the receive.

Reading the IICD will return the last byte received while the IIC is configured in either master receive or slave receive modes. The IICD does not reflect every byte that is 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–bit 1) concatenated with the required R/W bit (in position bit 0).

14.4 Functional Description

This section provides a complete functional description of the IIC module.

14.4.1 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 14-8.

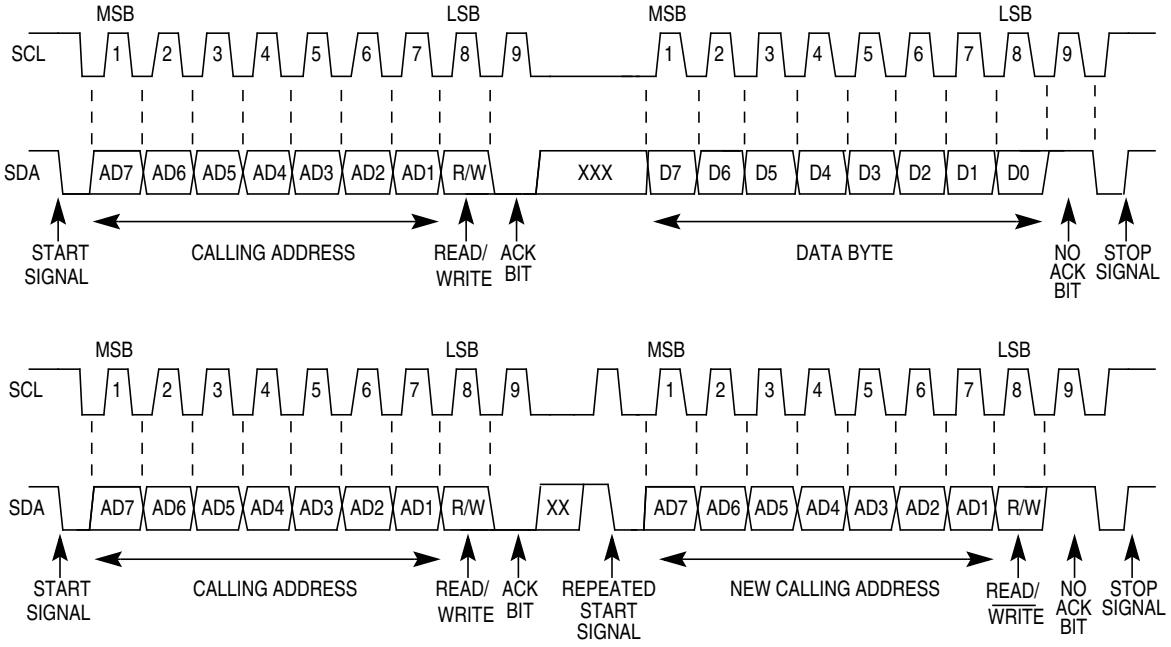


Figure 14-8. IIC Bus Transmission Signals

14.4.1.1 START Signal

When the bus is free; i.e., no master device is engaging the bus (both SCL and SDA lines are at logical high), a master may initiate communication by sending a START signal. As shown in Figure 14-8, 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.

14.4.1.2 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/W bit. The R/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 will respond by sending back an acknowledge bit. This is done by pulling the SDA low at the 9th clock (see Figure 14-8).

No two slaves in the system may have the same address. If the IIC module is the master, it must not transmit an address that is 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 will revert to slave mode and operate correctly even if it is being addressed by another master.

14.4.1.3 Data Transfer

Before successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the R/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 14-8. 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 9th 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.

14.4.1.4 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 14-8).

The master can generate a STOP even if the slave has generated an acknowledge at which point the slave must release the bus.

14.4.1.5 Repeated START Signal

As shown in Figure 14-8, 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.

14.4.1.6 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.

14.4.1.7 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 14-9). 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.

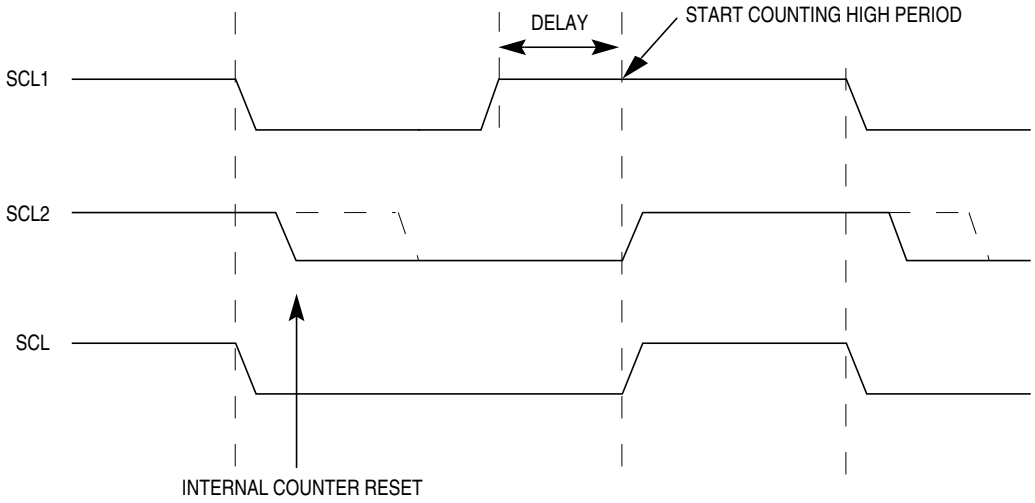


Figure 14-9. IIC Clock Synchronization

14.4.1.8 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 case, it halts the bus clock and forces the master clock into wait states until the slave releases the SCL line.

14.4.1.9 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.

14.5 Resets

The IIC is disabled after reset. The IIC cannot cause an MCU reset.

14.6 Interrupts

The IIC generates a single interrupt.

An interrupt from the IIC is generated when any of the events in Table 14-7 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 one to it in the interrupt routine. The user can determine the interrupt type by reading the status register.

Table 14-7. 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

14.6.1 Byte Transfer Interrupt

The TCF (transfer complete flag) bit is set at the falling edge of the 9th clock to indicate the completion of byte transfer.

14.6.2 Address Detect Interrupt

When the calling address matches the programmed slave address (IIC address register), 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.

14.6.3 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 by writing a one to it.

14.7 Initialization/Application Information

Module Initialization (Slave)

1. Write: IICA
   — to set the slave address
2. Write: IICC
   — 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 14-11

Module Initialization (Master)

1. Write: IICF
   — to set the IIC baud rate (example provided in this chapter)
2. Write: IICC
   — 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 14-11
5. Write: IICC
   — to enable TX
6. Write: IICC
   — to enable MST (master mode)
7. Write: IICD
   — with the address of the target slave. (The LSB of this byte will determine whether the communication is master receive or transmit.)

Module Use

The routine shown in Figure 14-11 can handle both master and slave IIC operations. For slave operation, an incoming IIC message that contains the proper address will begin IIC communication. For master operation, communication must be initiated by writing to the IICD register.

Register Model

IICA	ADDR							0
	Address to which the module will respond when addressed as a slave (in slave mode)							
IICF	MULT		ICR					
	Baud rate = BUSCLK / (2 x MULT x (SCL DIVIDER))							
IICC	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							

Figure 14-10. IIC Module Quick Start

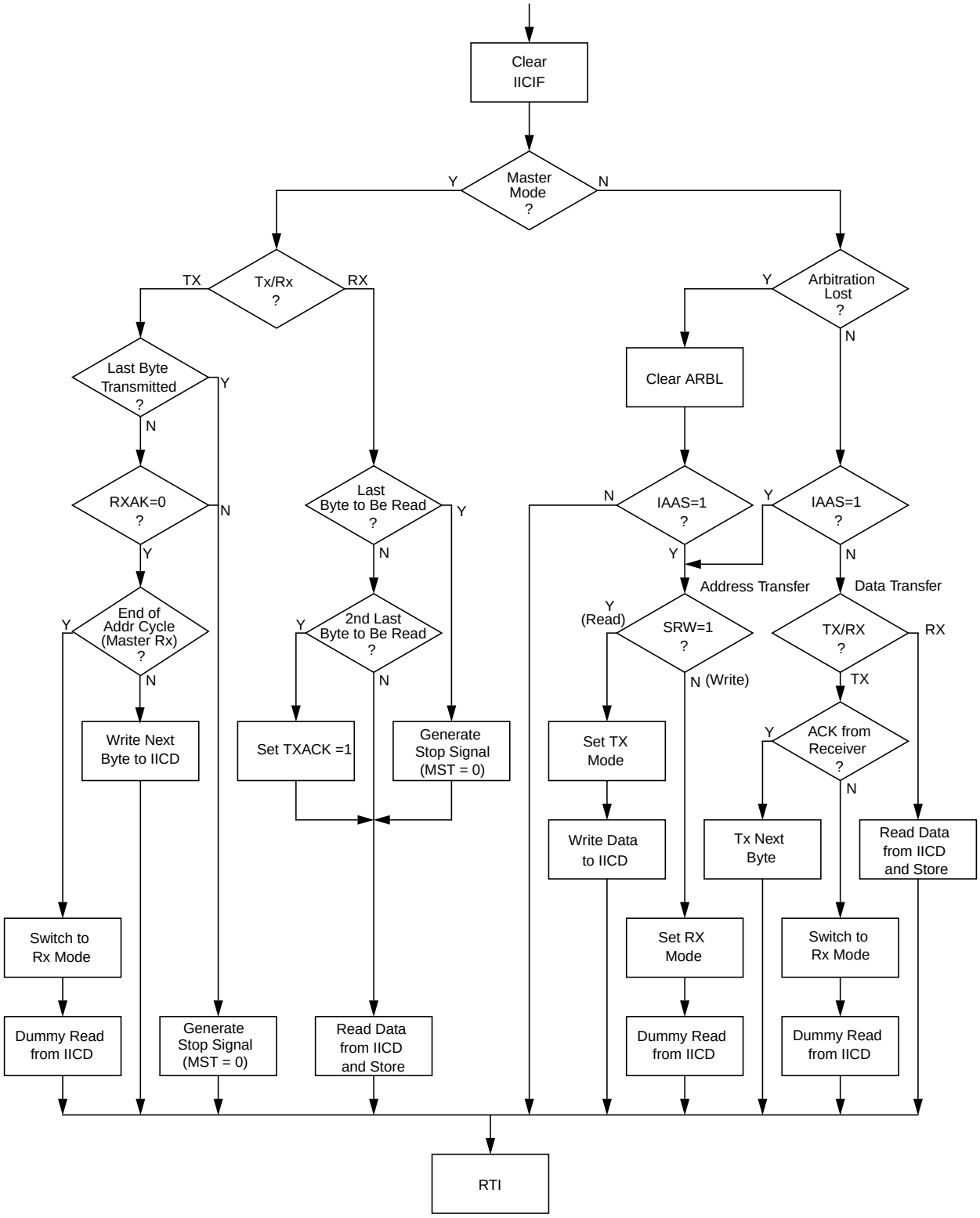


Figure 14-11. Typical IIC Interrupt Routine

Chapter 15
Analog-to-Digital Converter (S08ADC12V1)

15.1 Introduction

The 12-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

15.1.1 ADC Configuration Information

The ADC channel assignments, alternate clock function, and hardware trigger function are configured as described in this section for the MC9S08LC60/36/20 Family of devices.

15.1.1.1 Channel Assignments

The ADC channel assignments for the MC9S08LC60/36/20 devices are shown in Table 15-1. Reserved channels convert to an unknown value.

Table 15-1. ADC Channel Assignment

ADCH	Channel	Input	Pin Control
00000	AD0	PTA0/ADP0	ADPC0
00001	AD1	PTA1/ADP1	ADPC1
00010	AD2	PTA2/ADP2	ADPC2
00011	AD3	PTA3/ADP3	ADPC3
00100	AD4	PTA4/ADP4	ADPC4
00101	AD5	PTA5/ADP5	ADPC5
00110	AD6	PTA6/ADP6	ADPC6
00111	AD7	PTA7/ADP7	ADPC7
01000	AD8	V_{REFL}	N/A
01001	AD9	V_{REFL}	N/A
01010	AD10	V_{REFL}	N/A
01011	AD11	V_{REFL}	N/A
01100	AD12	V_{REFL}	N/A
01101	AD13	V_{REFL}	N/A
01110	AD14	V_{REFL}	N/A
01111	AD15	V_{REFL}	N/A

ADCH	Channel	Input	Pin Control
10000	AD16	V_{REFL}	N/A
10001	AD17	V_{REFL}	N/A
10010	AD18	V_{REFL}	N/A
10011	AD19	V_{SS}	N/A
10100	AD20	V_{LCD}	N/A
10101	AD21	V_{LL1}	N/A
10110	AD22	V_{DDASW}	N/A
10111	AD23	V_{DDA}	N/A
11000	AD24	V_{DDSW}	N/A
11001	AD25	V_{DD}	N/A
11010	AD26	Temperature Sensor	N/A
11011	AD27	Internal Bandgap[1]	N/A
11100	V_{REFH}	V_{REFH}	N/A
11101	V_{REFH}	V_{REFH}	N/A
11110	V_{REFL}	V_{REFL}	N/A
11111	Module Disabled	None	N/A

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

15.1.1.2 Alternate Clock

The ADC is capable of performing conversions using the MCU bus clock, the bus clock divided by two, or the local asynchronous clock (ADACK) within the module. The alternate clock, ALTCLK, input for the MC9S08LC60/36/20 MCU devices is not implemented.

15.1.1.3 Hardware Trigger

The ADC hardware trigger, ADHWT, is output from the real-time interrupt (RTI) counter. The RTI counter can be clocked by either ICGERCLK or a nominal 1-kHz clock source within the RTI block.

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

The RTI can be configured to cause a hardware trigger in MCU run, wait, and stop3.

15.1.1.4 Analog Pin Enables

The ADC on MC9S08LC60/36/20 MCU devices contains only one analog pin enable registers, APCTL1.

15.1.1.5 Temperature Sensor

To use the on-chip temperature sensor, the user must perform the following:

1. Configure ADC for long sample with a maximum of 1-MHz clock.
2. Convert the bandgap voltage reference channel (AD27).

   By converting the digital value of the bandgap voltage reference channel using the value of VBG, the user can determine V_{DD}. For value of bandgap voltage, see Section A.5, “DC Characteristics”.
3. Convert the temperature sensor channel (AD26).

   By using the calculated value of V_{DD}, convert the digital value of AD26 into a voltage, V_{temp} Equation 15-1 provides an approximate transfer function of the on-chip temperature sensor for: V_{DD} = 3.0 V, Temp = 25˚C, using the ADC at f_{ADCK} = 1.0 MHz, and configured for long sample.

   $$\text{TempC} = (V_{temp} - 0.7013) \div (0.0017) \qquad \textit{Eqn. 15-1}$$

   0.0017 is the uncalibrated voltage versus temperature slope in V/˚C. Uncalibrated accuracy of the temperature sensor is approximately ± 12˚C, using Equation 15-1.
4. To improve accuracy, the user should calibrate the bandgap voltage reference and temperature sensor.
   - — Calibrating at 25˚C will improve accuracy to 4.5˚C.
   - — Calibration at three temperature points (–40˚C, 25˚C, and 125˚C) will improve accuracy to ± 2.5˚C. After calibration has been completed, the user must calculate the slope for both hot and cold. In application code, the user would then calculate the temperature using Equation 15-1 and then determine whether the temperature is above or below 25˚C. After determining whether the temperature is above or below 25˚C, the user can recalculate the temperature using either the hot or cold slope value obtained during calibration.

15.1.1.6 Low-Power Mode Operation

The ADC is capable of running in stop3 mode but requires LVDSE in SPMSC1 to be set.

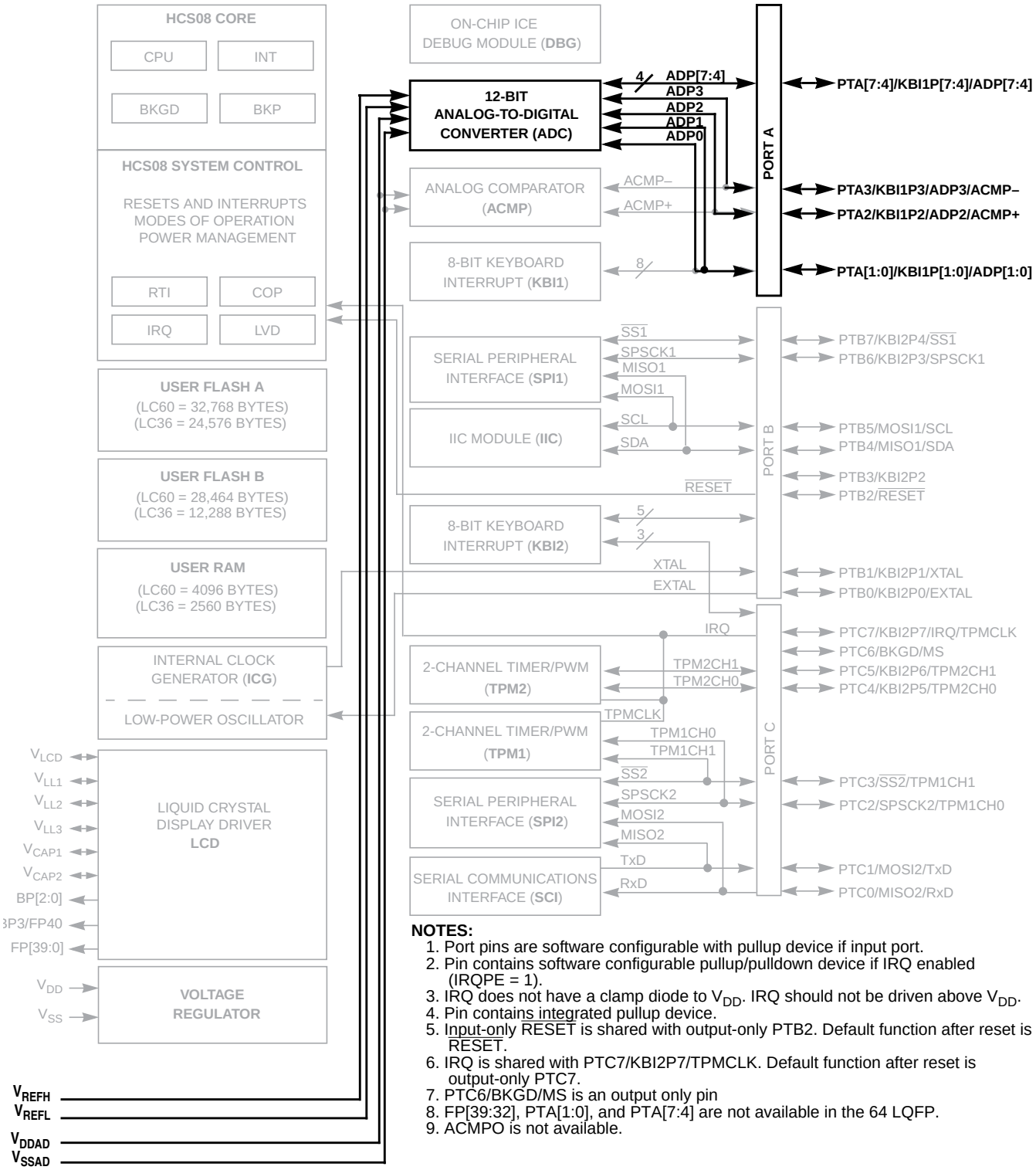


Figure 15-1. MC9S08LC60 Series Block Diagram Highlighting ADC Block and Pins

15.1.2 Features

Features of the ADC module include:

- Linear successive approximation algorithm with 12 bits resolution
- Up to 28 analog inputs
- Output formatted in 12-, 10- or 8-bit right-justified format
- Single or continuous conversion (automatic return to idle after single conversion)
- Configurable sample time and conversion speed/power
- Conversion 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
- Selectable asynchronous hardware conversion trigger
- Automatic compare with interrupt for less-than, or greater-than or equal-to, programmable value

15.1.3 Block Diagram

Figure 15-2 provides a block diagram of the ADC module.

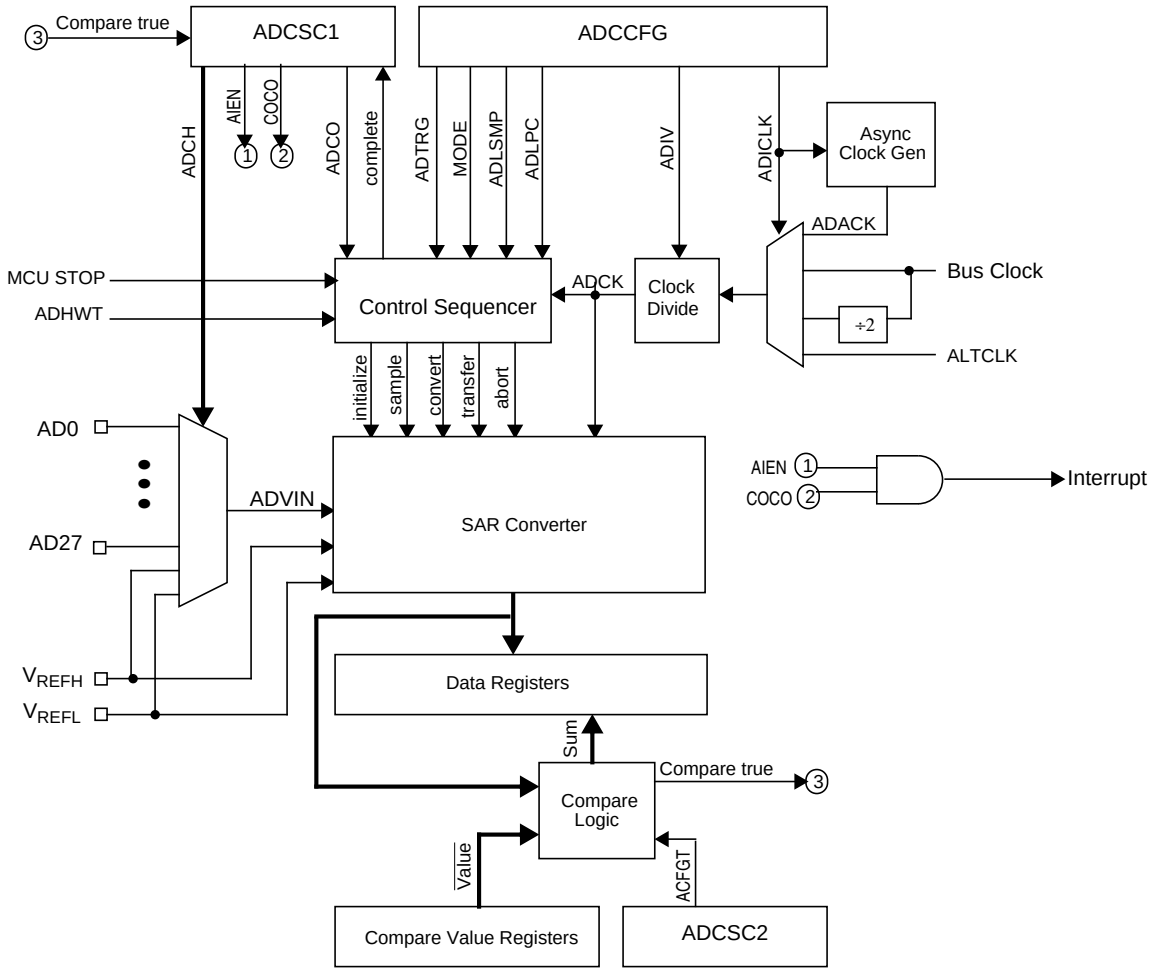


Figure 15-2. ADC Block Diagram

15.2 External Signal Description

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

Table 15-2. Signal Properties

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

15.2.1 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.

15.2.2 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}.

15.2.3 Voltage Reference High (V_{REFH})

V_{REFH} is the high reference voltage for the converter. In some packages, V_{REFH} is connected internally to V_{DDAD}. If externally available, V_{REFH} may be connected to the same potential as V_{DDAD}, or may be driven by an external source that is between the minimum V_{DDAD} spec and the V_{DDAD} potential (V_{REFH} must never exceed V_{DDAD}).

15.2.4 Voltage Reference Low (V_{REFL})

V_{REFL} is the low reference voltage for the converter. In some packages, V_{REFL} is connected internally to V_{SSAD}. If externally available, connect the V_{REFL} pin to the same voltage potential as V_{SSAD}.

15.2.5 Analog Channel Inputs (ADx)

The ADC module supports up to 28 separate analog inputs. An input is selected for conversion through the ADCH channel select bits.

15.3 Register Definition

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

- Status and control register, ADCSC1
- Status and control register, ADCSC2
- Data result registers, ADCRH and ADCRL
- Compare value registers, ADCCVH and ADCCVL
- Configuration register, ADCCFG
- Pin enable registers, APCTL1, APCTL2, APCTL3

15.3.1 Status and Control Register 1 (ADCSC1)

This section describes the function of the ADC status and control register (ADCSC1). Writing ADCSC1 aborts the current conversion and initiates a new conversion (if the ADCH bits are equal to a value other than all 1s).

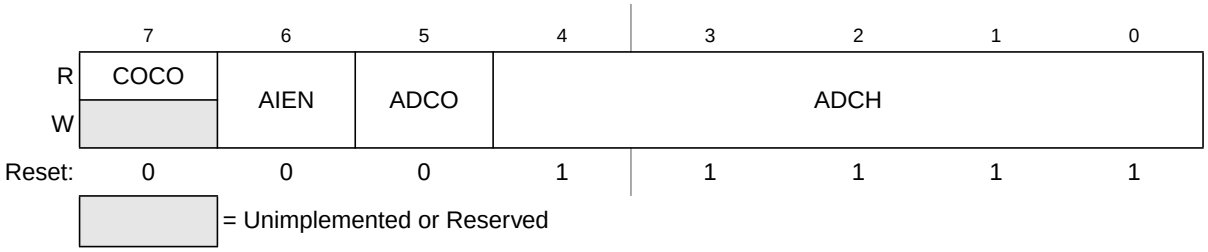


Figure 15-3. Status and Control Register (ADCSC1)

Table 15-3. ADCSC1 Register Field Descriptions

Field	Description
7 COCO	**Conversion Complete Flag** — The COCO flag is a read-only bit which is set each time a conversion is completed when the compare function is disabled (ACFE = 0). When the compare function is enabled (ACFE = 1) the COCO flag is set upon completion of a conversion only if the compare result is true. This bit is cleared whenever ADCSC1 is written or whenever ADCRL is read. 0 Conversion not completed 1 Conversion completed
6 AIEN	**Interrupt Enable** — AIEN is used to enable conversion complete interrupts. When COCO becomes set while AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt disabled 1 Conversion complete interrupt enabled
5 ADCO	**Continuous Conversion Enable** — ADCO is used to enable continuous conversions. 0 One conversion following a write to the ADCSC1 when software triggered operation is selected, or one conversion following assertion of ADHWT when hardware triggered operation is selected. 1 Continuous conversions initiated following a write to ADCSC1 when software triggered operation is selected. Continuous conversions are initiated by an ADHWT event when hardware triggered operation is selected.
4:0 ADCH	**Input Channel Select** — The ADCH bits form a 5-bit field which is used to select one of the input channels. The input channels are detailed in Figure 15-4. The successive approximation converter subsystem is turned off when the channel select bits are all set to 1. This feature allows for explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way will prevent an additional, single conversion from being performed. It is not necessary to set the channel select bits to all 1s 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.

Figure 15-4. Input Channel Select

ADCH	Input Select
00000	AD0
00001	AD1
00010	AD2
00011	AD3
00100	AD4
00101	AD5
00110	AD6
00111	AD7

ADCH	Input Select
10000	AD16
10001	AD17
10010	AD18
10011	AD19
10100	AD20
10101	AD21
10110	AD22
10111	AD23

Figure 15-4. Input Channel Select (continued)

ADCH	Input Select
01000	AD8
01001	AD9
01010	AD10
01011	AD11
01100	AD12
01101	AD13
01110	AD14
01111	AD15

ADCH	Input Select
11000	AD24
11001	AD25
11010	AD26
11011	AD27
11100	Reserved
11101	V_{REFH}
11110	V_{REFL}
11111	Module disabled

15.3.2 Status and Control Register 2 (ADCSC2)

The ADCSC2 register is used to control the compare function, conversion trigger and conversion active of the ADC module.

	7	6	5	4	3	2	1	0
R	ADACT	ADTRG	ACFE	ACFGT	0	0	R[1]	R[1]
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

[1] Bits 1 and 0 are reserved bits that must always be written to 0.

Figure 15-5. Status and Control Register 2 (ADCSC2)

Table 15-4. ADCSC2 Register Field Descriptions

Field	Description
7 ADACT	**Conversion Active** — ADACT indicates that a conversion 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 is used to select the type of trigger to be 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 ADCSC1. When hardware trigger is selected, a conversion is initiated following the assertion of the ADHWT input. 0 Software trigger selected 1 Hardware trigger selected

Table 15-4. ADCSC2 Register Field Descriptions (continued)

Field	Description
5 ACFE	**Compare Function Enable** — ACFE is used to enable the compare function. 0 Compare function disabled 1 Compare function enabled
4 ACFGT	**Compare Function Greater Than Enable** — ACFGT is used to configure the compare function to trigger when the result of the conversion of the input being monitored is greater than or equal to the compare value. The compare function defaults to triggering when the result of the compare of the input being monitored is less than the compare value. 0 Compare triggers when input is less than compare level 1 Compare triggers when input is greater than or equal to compare level

15.3.3 Data Result High Register (ADCRH)

In 12-bit operation, ADCRH contains the upper four bits of the result of a 12-bit conversion.

	7	6	5	4	3	2	1	0
R	0	0	0	0	ADR11	ADR10	ADR9	ADR8
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-6. Data Result High Register (ADCRH)

In 10-bit mode, ADCRH contains the upper two bits of the result of a 10-bit conversion. When configured for 10-bit mode, ADR11 – ADR10 are equal to zero. When configured for 8-bit mode, ADR11 – ADR8 are equal to zero.

In both 12-bit and 10-bit mode, ADCRH is updated each time a conversion completes except when automatic compare is enabled and the compare condition is not met. In 12-bit and 10-bit mode, reading ADCRH prevents the ADC from transferring subsequent conversion results into the result registers until ADCRL is read. If ADCRL is not read until after the next conversion is completed, then the intermediate conversion result is lost. In 8-bit mode there is no interlocking with ADCRL.

In the case that the MODE bits are changed, any data in ADCRH becomes invalid.

15.3.4 Data Result Low Register (ADCRL)

ADCRL contains the lower eight bits of the result of a 12-bit or 10-bit conversion, and all eight bits of an 8-bit conversion. This register is updated each time a conversion completes except when automatic compare is enabled and the compare condition is not met. In 12-bit and 10-bit mode, reading ADCRH prevents the ADC from transferring subsequent conversion results into the result registers until ADCRL is read. If ADCRL is not read until the after next conversion is completed, then the intermediate conversion results will be lost. In 8-bit mode, there is no interlocking with ADCRH. In the case that the MODE bits are changed, any data in ADCRL becomes invalid.

	7	6	5	4	3	2	1	0
R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-7. Data Result Low Register (ADCRL)

15.3.5 Compare Value High Register (ADCCVH)

In 12-bit mode, the ADCCVH register holds the upper four bits of the 12-bit compare value. These bits are compared to the upper four bits of the result following a conversion in 12-bit mode when the compare function is enabled.

	7	6	5	4	3	2	1	0
R	0	0	0	0	ADCV11	ADCV10	ADCV9	ADCV8
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-8. Compare Value High Register (ADCCVH)

In 10-bit mode, the ADCCVH register holds the upper two bits of the 10-bit compare value (ADCV9 – ADCV8). These bits are compared to the upper two bits of the result following a conversion in 10-bit mode when the compare function is enabled.

In 8-bit mode, ADCCVH is not used during compare.

15.3.6 Compare Value Low Register (ADCCVL)

This register holds the lower 8 bits of the 12-bit or 10-bit compare value, or all 8 bits of the 8-bit compare value. Bits ADCV7:ADCV0 are compared to the lower 8 bits of the result following a conversion in 12-bit, 10-bit or 8-bit mode.

	7	6	5	4	3	2	1	0
R	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 15-9. Compare Value Low Register(ADCCVL)

15.3.7 Configuration Register (ADCCFG)

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

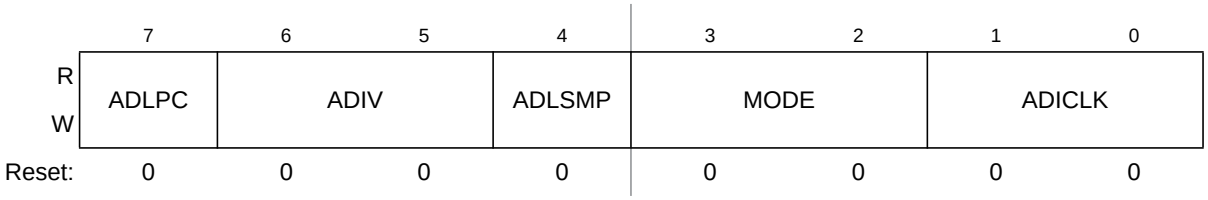


Figure 15-10. Configuration Register (ADCCFG)

Table 15-5. ADCCFG Register Field Descriptions

Field	Description
7 ADLPC	**Low Power Configuration** — ADLPC controls the speed and power configuration of the successive approximation converter. This is used to optimize power consumption when higher sample rates are not required. 0 High speed configuration 1 Low power configuration: {FC31}The power is reduced at the expense of maximum clock speed.
6:5 ADIV	**Clock Divide Select** — ADIV select the divide ratio used by the ADC to generate the internal clock ADCK. Table 15-6 shows the available clock configurations.
4 ADLSMP	**Long Sample Time Configuration** — ADLSMP selects between long and short sample time. This 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. 0 Short sample time 1 Long sample time
3:2 MODE	**Conversion Mode Selection** — MODE bits are used to select between 12-, 10- or 8-bit operation. See Table 15-7.
1:0 ADICLK	**Input Clock Select** — ADICLK bits select the input clock source to generate the internal clock ADCK. See Table 15-8.

Table 15-6. 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 15-7. Conversion Modes

MODE	Mode Description
00	8-bit conversion (N=8)
01	12-bit conversion (N=12)
10	10-bit conversion (N=10)
11	Reserved

Table 15-8. Input Clock Select

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

15.3.8 Pin Control 1 Register (APCTL1)

The pin control registers are used to 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	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
Reset:	0	0	0	0	0	0	0	0

Figure 15-11. Pin Control 1 Register (APCTL1)

Table 15-9. APCTL1 Register Field Descriptions

Field	Description
7 ADPC7	**ADC Pin Control 7** — ADPC7 is used to control 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 is used to control 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 is used to control 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 is used to control 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 is used to control 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 is used to control the pin associated with channel AD2. 0 AD2 pin I/O control enabled 1 AD2 pin I/O control disabled

Table 15-9. APCTL1 Register Field Descriptions (continued)

Field	Description
1 ADPC1	**ADC Pin Control 1** — ADPC1 is used to control 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 is used to control the pin associated with channel AD0. 0 AD0 pin I/O control enabled 1 AD0 pin I/O control disabled

15.3.9 Pin Control 2 Register (APCTL2)

APCTL2 is used to control channels 8–15 of the ADC module.

	7	6	5	4	3	2	1	0
R W	ADPC15	ADPC14	ADPC13	ADPC12	ADPC11	ADPC10	ADPC9	ADPC8
Reset:	0	0	0	0	0	0	0	0

Figure 15-12. Pin Control 2 Register (APCTL2)

Table 15-10. APCTL2 Register Field Descriptions

Field	Description
7 ADPC15	**ADC Pin Control 15** — ADPC15 is used to control 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 is used to control 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 is used to control 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 is used to control 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 is used to control 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 is used to control the pin associated with channel AD10. 0 AD10 pin I/O control enabled 1 AD10 pin I/O control disabled

Table 15-10. APCTL2 Register Field Descriptions (continued)

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

15.3.10 Pin Control 3 Register (APCTL3)

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

	7	6	5	4	3	2	1	0
R W	ADPC23	ADPC22	ADPC21	ADPC20	ADPC19	ADPC18	ADPC17	ADPC16
Reset:	0	0	0	0	0	0	0	0

Figure 15-13. Pin Control 3 Register (APCTL3)

Table 15-11. APCTL3 Register Field Descriptions

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

Table 15-11. APCTL3 Register Field Descriptions (continued)

Field	Description
1 ADPC17	**ADC Pin Control 17** — ADPC17 is used to control 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 is used to control the pin associated with channel AD16. 0 AD16 pin I/O control enabled 1 AD16 pin I/O control disabled

15.4 Functional Description

The ADC module is disabled during reset or when the ADCH bits are all high. The module is idle when a conversion has completed and another conversion has not been initiated. When idle, the module is in its lowest power state.

The ADC can perform an analog-to-digital conversion on any of the software selectable channels. In 12-bit and 10-bit mode, the selected channel voltage is converted by a successive approximation algorithm into a 12-bit digital result. In 8-bit mode, the selected channel voltage is converted by a successive approximation algorithm into a 9-bit digital result.

When the conversion is completed, the result is placed in the data registers (ADCRH and ADCRL). In 10-bit mode, the result is rounded to 10 bits and placed in the data registers (ADCRH and ADCRL). In 8-bit mode, the result is rounded to 8 bits and placed in ADCRL. The conversion complete flag (COCO) is then set and an interrupt is generated if the conversion complete interrupt has been enabled (AIEN = 1).

The ADC module has the capability of automatically comparing the result of a conversion with the contents of its compare registers. The compare function is enabled by setting the ACFE bit and operates in conjunction with any of the conversion modes and configurations.

15.4.1 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 2. For higher bus clock rates, this allows a maximum divide by 16 of the bus clock.
- 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. When selected as the clock source this clock remains active while the MCU is in wait or stop3 mode and allows conversions in these modes for lower noise operation.

Whichever clock is selected, its frequency must fall within the specified frequency range for ADCK. If the available clocks are too slow, the ADC will not perform according to specifications. If the available clocks

are too fast, then 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.

15.4.2 Input Select and Pin Control

The pin control registers (APCTL3, APCTL2, and APCTL1) are used to 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.

15.4.3 Hardware Trigger

The ADC module has a selectable asynchronous hardware conversion trigger, ADHWT, that is enabled when the ADTRG bit is set. This source is not available on all MCUs. Consult the module introduction for information on the ADHWT source specific to this MCU.

When ADHWT source is available and hardware trigger is enabled (ADTRG=1), a conversion is initiated on the rising edge of ADHWT. If a conversion is in progress when a rising edge occurs, the rising edge is ignored. In continuous convert configuration, only the initial rising edge to launch continuous conversions is observed. The hardware trigger function operates in conjunction with any of the conversion modes and configurations.

15.4.4 Conversion Control

Conversions can be performed in 12-bit mode, 10-bit mode or 8-bit mode as determined by the MODE bits. Conversions can be initiated by either a software or hardware trigger. In addition, the ADC module can be configured for low power operation, long sample time, continuous conversion, and automatic compare of the conversion result to a software determined compare value.

15.4.4.1 Initiating Conversions

A conversion is initiated:

- Following a write to ADCSC1 (with ADCH bits not all 1s) if software triggered operation is selected.
- Following a hardware trigger (ADHWT) event if hardware triggered operation is selected.
- Following the transfer of the result to the data registers when continuous conversion is enabled.

If continuous conversions are enabled a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation, continuous conversions begin after ADCSC1 is written and continue until aborted. In hardware triggered operation, continuous conversions begin after a hardware trigger event and continue until aborted.

15.4.4.2 Completing Conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, ADCRH and ADCRL. This is indicated by the setting of COCO. An interrupt is generated if AIEN is high at the time that COCO is set.

A blocking mechanism prevents a new result from overwriting previous data in ADCRH and ADCRL if the previous data is in the process of being read while in 12-bit or 10-bit MODE (the ADCRH register has been read but the ADCRL register has not). When blocking is active, the data transfer is blocked, COCO 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 data transfer is blocked, another conversion is initiated regardless of the state of ADCO (single or continuous conversions enabled).

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.

15.4.4.3 Aborting Conversions

Any conversion in progress will be aborted when:

- A write to ADCSC1 occurs (the current conversion will be aborted and a new conversion will be initiated, if ADCH are not all 1s).
- A write to ADCSC2, ADCCFG, ADCCVH, or ADCCVL occurs. This indicates a mode of operation change has occurred and the current conversion is therefore invalid.
- The MCU is reset.
- The MCU enters stop mode with ADACK not enabled.

When a conversion is aborted, the contents of the data registers, ADCRH and ADCRL, are not altered but continue to be the values transferred after the completion of the last successful conversion. In the case that the conversion was aborted by a reset, ADCRH and ADCRL return to their reset states.

15.4.4.4 Power Control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source, the ADACK clock generator is also enabled.

Power consumption when active can be reduced by setting ADLPC. This results in a lower maximum value for f_{ADCK} (see the electrical specifications).

15.4.4.5 Sample Time and Total Conversion Time

The total conversion time depends on the sample time (as determined by ADLSMP), the MCU bus frequency, the conversion mode (8-bit, 10-bit or 12-bit), and the frequency of the conversion clock (f_{ADCK}). After the module becomes active, sampling of the input begins. ADLSMP is used to select between short (3.5 ADCK cycles) and long (23.5 ADCK cycles) sample times.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 ADCRH and ADCRL 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). If the bus frequency is less than 1/11th of the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when long sample is enabled (ADLSMP=1).

The maximum total conversion time for different conditions is summarized in Table 15-12.

Table 15-12. Total Conversion Time vs. Control Conditions

Conversion Type	ADICLK	ADLSMP	Max Total Conversion Time
Single or first continuous 8-bit	0x, 10	0	20 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	0x, 10	0	23 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	0x, 10	1	40 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	0x, 10	1	43 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	11	0	5 µs + 20 ADCK + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	11	0	5 µs + 23 ADCK + 5 bus clock cycles
Single or first continuous 8-bit	11	1	5 µs + 40 ADCK + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	11	1	5 µs + 43 ADCK + 5 bus clock cycles
Subsequent continuous 8-bit; $f_{BUS} \geq f_{ADCK}$	xx	0	17 ADCK cycles
Subsequent continuous 10-bit or 12-bit; $f_{BUS} \geq f_{ADCK}$	xx	0	20 ADCK cycles
Subsequent continuous 8-bit; $f_{BUS} \geq f_{ADCK}/11$	xx	1	37 ADCK cycles
Subsequent continuous 10-bit or 12-bit; $f_{BUS} \geq f_{ADCK}/11$	xx	1	40 ADCK cycles

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. For example, in 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, then the conversion time for a single conversion is:

$$\text{Conversion time} = \frac{23 \text{ ADCK cyc}}{8 \text{ MHz}/1} + \frac{5 \text{ bus cyc}}{8 \text{ MHz}} = 3.5 \text{ µs}$$

$$\text{Number of bus cycles} = 3.5 \text{ µs} \times 8 \text{ MHz} = 28 \text{ cycles}$$

NOTE

The ADCK frequency must be between f_{ADCK} minimum and f_{ADCK} maximum to meet ADC specifications.

15.4.5 Automatic Compare Function

The compare function can be configured to check for either an upper limit or lower limit. After the input is sampled and converted, the result is added to the two's complement of the compare value (ADCCVH and ADCCVL). When comparing to an upper limit (ACFGT = 1), if the result is greater-than or equal-to the compare value, COCO is set. When comparing to a lower limit (ACFGT = 0), if the result is less than the compare value, COCO is set. The value generated by the addition of the conversion result and the two's complement of the compare value is transferred to ADCRH and ADCRL.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, COCO is not set and no data is transferred to the result registers. An ADC interrupt is generated upon the setting of COCO if the ADC interrupt is enabled (AIEN = 1).

NOTE

The compare function can be used to monitor the voltage on a channel while the MCU is in either wait or stop3 mode. The ADC interrupt will wake the MCU when the compare condition is met.

15.4.6 MCU Wait Mode Operation

The WAIT instruction puts the MCU in a lower power-consumption standby mode from which recovery is very 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.

A conversion complete event sets the COCO and generates an ADC interrupt to wake the MCU from wait mode if the ADC interrupt is enabled (AIEN = 1).

15.4.7 MCU Stop3 Mode Operation

The STOP instruction is used to put the MCU in a low power-consumption standby mode during which most or all clock sources on the MCU are disabled.

15.4.7.1 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 ADCRH and ADCRL are unaffected by stop3 mode. After exiting from stop3 mode, a software or hardware trigger is required to resume conversions.

15.4.7.2 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.

A conversion complete event sets the COCO and generates an ADC interrupt to wake the MCU from stop3 mode if the ADC interrupt is enabled (AIEN = 1).

NOTE

It is possible for the ADC module to 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 that the data transfer blocking mechanism (discussed in Section 15.4.4.2, "Completing Conversions) is cleared when entering stop3 and continuing ADC conversions.

15.4.8 MCU Stop1 and Stop2 Mode Operation

The ADC module is automatically disabled when the MCU enters either stop1 or stop2 mode. All module registers contain their reset values following exit from stop1 or stop2. Therefore the module must be re-enabled and re-configured following exit from stop1 or stop2.

15.5 Initialization Information

This section gives an example which provides some basic direction on how a user would initialize and configure the ADC module. The user has the flexibility of choosing between configuring the module for 8-, 10-, or 12-bit resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. Refer to Table 15-6, Table 15-7, and Table 15-8 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.

15.5.1 ADC Module Initialization Example

15.5.1.1 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. 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.

2. Update status and control register 2 (ADCSC2) to select the conversion trigger (hardware or software) and compare function options, if enabled.
3. Update status and control register 1 (ADCSC1) to select whether conversions will be continuous or completed only once, and to enable or disable conversion complete interrupts. The input channel on which conversions will be performed is also selected here.

15.5.1.2 Pseudo — Code Example

In this example, the ADC module will be 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 will be 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	Unimplemented or reserved, always reads zero
Bit 1:0		00	Reserved for Freescale's internal use; always write zero

ADCSC1 = 0x41 (%01000001)

Bit 7	COCO	0	Read-only flag which is set when a conversion completes
Bit 6	AIEN	1	Conversion complete interrupt enabled
Bit 5	ADCO	0	One conversion only (continuous conversions disabled)
Bit 4:0	ADCH	00001	Input channel 1 selected as ADC input channel

ADCRH/L = 0xxx

Holds results of conversion. Read high byte (ADCRH) before low byte (ADCRL) 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

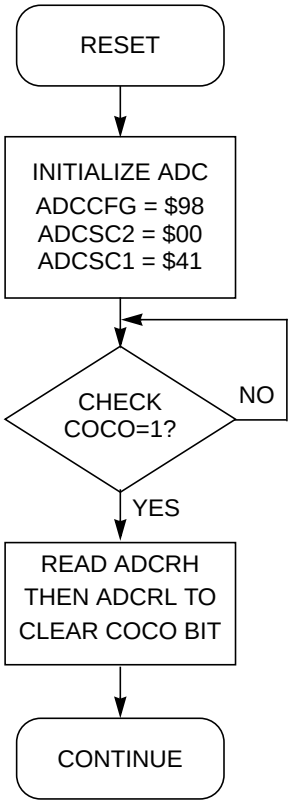


Figure 15-14. Initialization Flowchart for Example

15.6 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.

15.6.1 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.

15.6.1.1 Analog Supply Pins

The ADC module has analog power and ground supplies (V_{DDAD} and V_{SSAD}) which are available as separate pins on some devices. On other devices, V_{SSAD} is shared on the same pin as the MCU digital V_{SS}, and on others, both V_{SSAD} and V_{DDAD} are shared with the MCU digital supply pins. In these cases, there are separate pads for the analog supplies which are 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.

In cases where 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.

15.6.1.2 Analog Reference Pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs. The high reference is V_{REFH}, which may be shared on the same pin as V_{DDAD} on some devices. The low reference is V_{REFL}, which may be shared on the same pin as V_{SSAD} on some devices.

When available on a separate pin, V_{REFH} may be connected to the same potential as V_{DDAD}, or may be driven by an external source that is between the minimum V_{DDAD} spec and the V_{DDAD} potential (V_{REFH} must never exceed V_{DDAD}). When available on a separate pin, V_{REFL} must be connected to the same voltage potential as V_{SSAD}. Both V_{REFH} and V_{REFL} 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 will cause a voltage drop which could result in conversion errors. Inductance in this path must be minimum (parasitic only).

15.6.1.3 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 will be in its high impedance state and the pullup is disabled. Also, the input buffer draws DC current when its input is not at either 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 $FFF (full scale 12-bit representation), $3FF (full scale 10-bit representation) or $FF (full scale 8-bit representation). If the input is equal to or less than V_{REFL}, the converter circuit converts it to $000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. There will be a brief current associated with V_{REFL} when the sampling capacitor is charging. The input is sampled for 3.5 cycles of the ADCK source when ADLSMP is low, or 23.5 cycles when ADLSMP is high.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins should not be transitioning during conversions.

15.6.2 Sources of Error

Several sources of error exist for A/D conversions. These are discussed in the following sections.

15.6.2.1 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 (to increase the sample window to 23.5 cycles) or decreasing ADCK frequency to increase sample time.

15.6.2.2 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).

15.6.2.3 Noise-Induced Errors

System noise which 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 the 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 will improve noise issues but will affect 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.

15.6.2.4 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:

$$1_{LSB} = (V_{REFH} - V_{REFL}) / 2^N \qquad \textit{Eqn. 15-2}$$

There is an inherent quantization error due to the digitization of the result. For 8-bit or 10-bit conversions the code will transition 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 ± 1/2LSB in 8- or 10-bit mode. As a consequence, however, the code width of the first ($000) conversion is only 1/2LSB and the code width of the last ($FF or $3FF) is 1.5LSB.

For 12-bit conversions the code transitions only after the full code width is present, so the quantization error is -1LSB to 0LSB and the code width of each step is 1LSB.

15.6.2.5 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 (1/2LSB in 8-bit or 10-bit modes and 1LSB in 12-bit mode). Note, if the first conversion is $001, then the difference between the actual $001 code width and its ideal (1LSB) 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 (1.5LSB in 8-bit or 10-bit modes and 1LSB in 12-bit mode). Note, if the last conversion is $3FE, then the difference between the actual $3FE 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 therefore includes all forms of error.

15.6.2.6 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 very 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 1/2LSB in 8-bit or 10-bit mode, or around 2 LSB in 12-bit mode, and will increase with noise.

This error may be reduced by repeatedly sampling the input and averaging the result. Additionally the techniques discussed in Section 15.6.2.3 will reduce 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 which are never converted for any input value.

In 8-bit or 10-bit mode, the ADC is guaranteed to be monotonic and to have no missing codes.

Chapter 16
Analog Comparator (S08ACMPV2)

16.1 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).

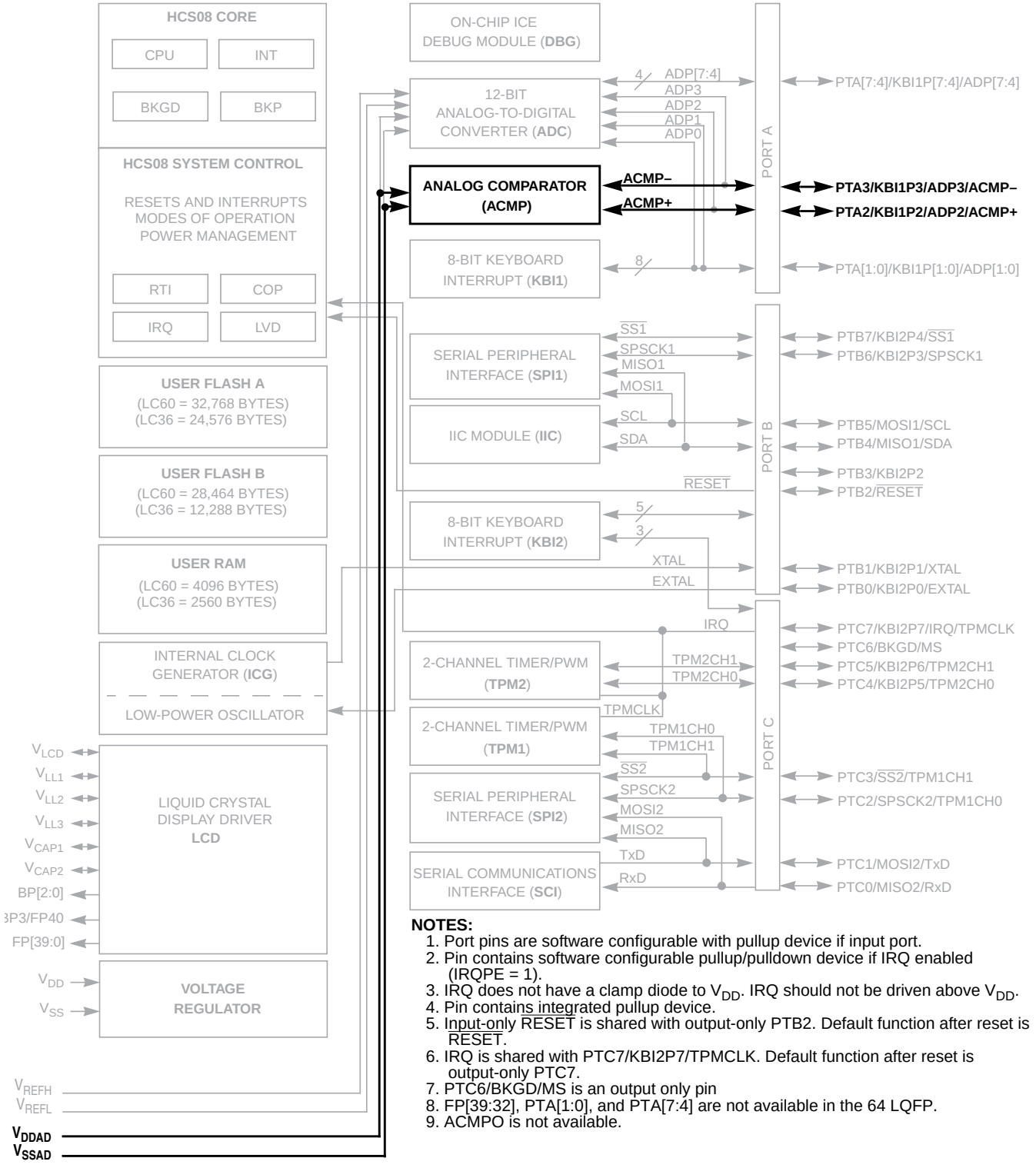
16.1.1 ACMP/TPM1 Configuration Information

The ACMP module can be configured to connect the output of the analog comparator to TPM1 input capture channel 0 by setting ACIC in SOPT2. With ACIC set, the TPM1CH0 pin is not available externally regardless of the configuration of the TPM1 module

Figure 16-1 shows the MC9S08LC60 Series block diagram with the ACMP highlighted.

16.1.2 AMCPO Availability

For the MC9S08LC60 Series, the AMCPO pin is not available, so the ACOPE bit in the ACMPSC register is reserved and does not have any effect.



NOTES:

1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. Input-only $\overline{\text{RESET}}$ is shared with output-only PTB2. Default function after reset is $\overline{\text{RESET}}$.
6. IRQ is shared with PTC7/KBI2P7/TPMCLK. Default function after reset is output-only PTC7.
7. PTC6/BKGD/MS is an output only pin
8. FP[39:32], PTA[1:0], and PTA[7:4] are not available in the 64 LQFP.
9. ACMPO is not available.

Figure 16-1. MC9S08LC60 Series Block Diagram Highlighting ACMP Block and Pins

16.1.3 Features

The ACMP has the following features:

- Full rail-to-rail supply operation.
- Less than 40 mV of input offset.
- Less than 15 mV of hysteresis.
- Selectable interrupt on rising edge, falling edge, or either rising or falling edges of comparator output.
- Option to compare to fixed internal bandgap reference voltage.
- Option to allow comparator output to be visible on a pin, ACMPO.

16.1.4 Modes of Operation

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

16.1.4.1 ACMP in Wait Mode

The ACMP continues to run in wait mode if enabled before executing the WAIT instruction. Therefore, the ACMP can be used to bring the MCU out of wait mode if the ACMP interrupt, ACIE, is enabled. For lowest possible current consumption, the ACMP should be disabled by software if not required as an interrupt source during wait mode.

16.1.4.2 ACMP in Stop Modes

The ACMP is disabled in all stop modes, regardless of the settings before executing the STOP instruction. Therefore, the ACMP cannot be used as a wake up source from stop modes.

During either stop1 or stop2 mode, the ACMP module will be fully powered down. Upon wake-up from stop1 or stop2 mode, the ACMP module will be in the reset state.

During stop3 mode, clocks to the ACMP module are halted. No registers are affected. In addition, the ACMP comparator circuit will enter a low power state. No compare operation will occur while in stop3.

If stop3 is exited with a reset, the ACMP will be put into its reset state. If stop3 is exited with an interrupt, the ACMP continues from the state it was in when stop3 was entered.

16.1.4.3 ACMP in Active Background Mode

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

16.1.5 Block Diagram

The block diagram for the analog comparator module is shown Figure 16-2.

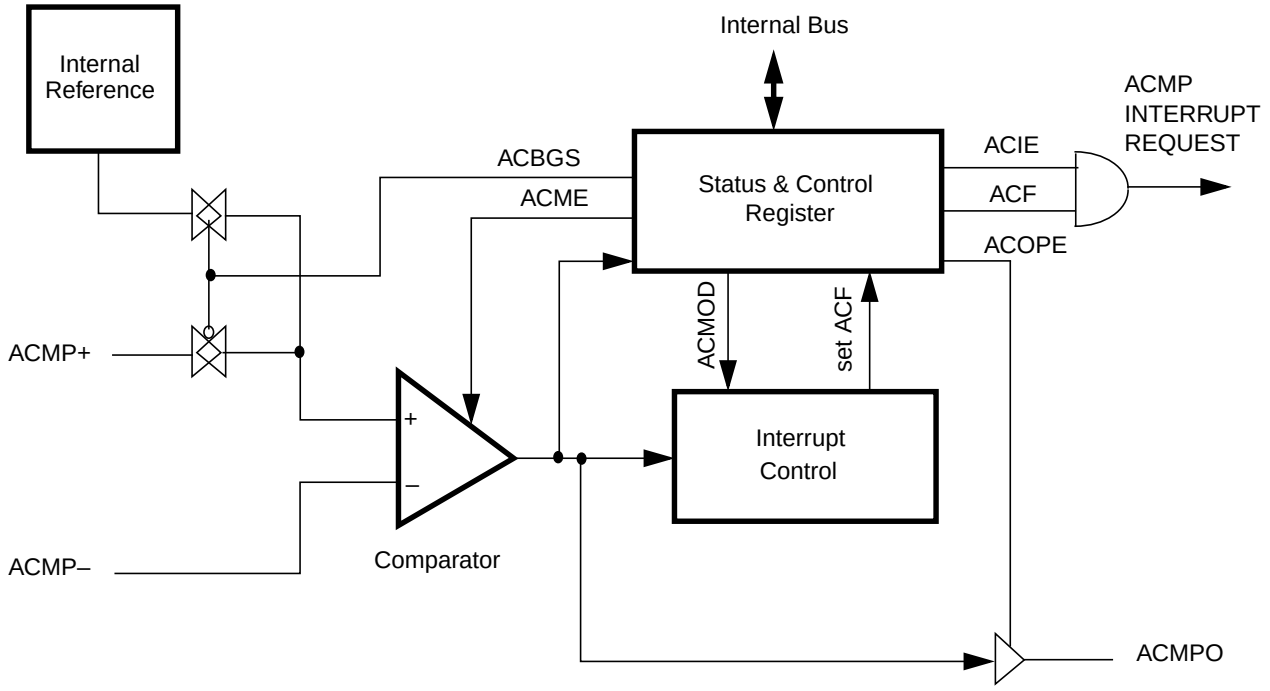


Figure 16-2. Analog Comparator (ACMP) Block Diagram

16.2 External Signal Description

The ACMP has two analog input pins, ACMP+ and ACMP– and one digital output pin ACMPO. Each of these pins can accept an input voltage that varies across the full operating voltage range of the MCU. As shown in Figure 16-2, the ACMP– pin is connected to the inverting input of the comparator, and the ACMP+ pin is connected to the comparator non-inverting input if ACBGS is a 0. As shown in Figure 16-2, the ACMPO pin can be enabled to drive an external pin.

The signal properties of ACMP are shown in Table 16-1.

Table 16-1. Signal Properties

Signal	Function	I/O
ACMP–	Inverting analog input to the ACMP. (Minus input)	I
ACMP+	Non-inverting analog input to the ACMP. (Positive input)	I
ACMPO	Digital output of the ACMP.	O

16.3 Register Definition

The ACMP includes one register:

- An 8-bit status and control register

Refer to the direct-page register summary in the memory section of this data sheet for the absolute address assignments for all ACMP registers.This section refers to registers and control bits only by their names and relative address offsets.

Some MCUs may have more than one ACMP, so register names include placeholder characters to identify which ACMP is being referenced.

16.3.1 ACMP Status and Control Register (ACMPSC)

ACMPSC contains the status flag and control bits which are used to enable and configure the ACMP.

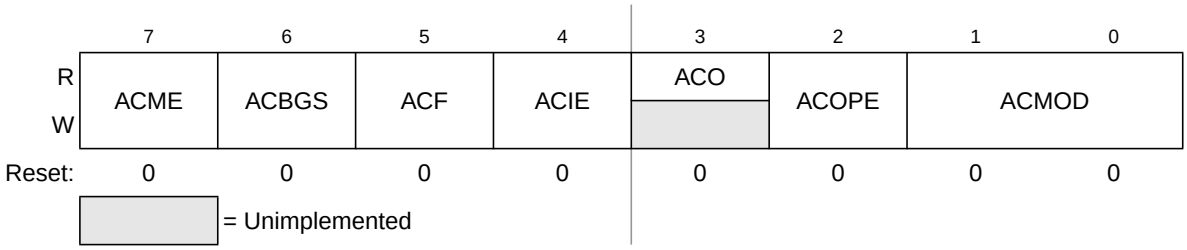


Figure 16-3. ACMP Status and Control Register

Table 16-2. ACMP Status and Control Register Field Descriptions

Field	Description
7 ACME	**Analog Comparator Module Enable** — ACME enables the ACMP module. 0 ACMP not enabled 1 ACMP is enabled
6 ACBGS	**Analog Comparator Bandgap Select** — ACBGS is used to select between the bandgap reference voltage or the ACMP+ pin as the input to the non-inverting input of the analog comparatorr. 0 External pin ACMP+ selected as non-inverting input to comparator 1 Internal reference select as non-inverting input to comparator
5 ACF	**Analog Comparator Flag** — ACF is set when a compare event occurs. Compare events are defined by ACMOD. ACF is cleared by writing a one to ACF. 0 Compare event has not occurred 1 Compare event has occurred
4 ACIE	**Analog Comparator Interrupt Enable** — ACIE enables the interrupt from the ACMP. When ACIE is set, an interrupt will be asserted when ACF is set. 0 Interrupt disabled 1 Interrupt enabled
3 ACO	**Analog Comparator Output** — Reading ACO will return the current value of the analog comparator output. ACO is reset to a 0 and will read as a 0 when the ACMP is disabled (ACME = 0).
2 ACOPE	**Analog Comparator Output Pin Enable** — ACOPE is used to enable 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 Mode** — ACMOD selects the type of compare event which sets ACF. 00 Encoding 0 — Comparator output falling edge 01 Encoding 1 — Comparator output rising edge 10 Encoding 2 — Comparator output falling edge 11 Encoding 3 — Comparator output rising or falling edge

16.4 Functional Description

The analog comparator can be used to compare two analog input voltages applied to ACMP+ and ACMP–; or it can be used to compare an analog input voltage applied to ACMP– with an internal bandgap reference voltage. ACBGS is used to select between the bandgap reference voltage or the ACMP+ pin as the input to the non-inverting input of the analog comparator. The comparator output is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. ACMOD is used to select the condition which will cause ACF to be set. ACF can be set on a rising edge of the comparator output, a falling edge of the comparator output, or either a rising or a falling edge (toggle). The comparator output can be read directly through ACO. The comparator output can be driven onto the ACMPO pin using ACOPE.

Chapter 17
Development Support

17.1 Introduction

This chapter describes the single-wire background debug mode (BDM), which uses the on-chip background debug controller (BDC) module, and the independent on-chip real-time in-circuit emulation (ICE) system, which uses the on-chip debug (DBG) module.

17.1.1 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

Features of the ICE system include:

- Two trigger comparators: Two address + read/write (R/W) or one full address + data + R/W
- Flexible 8-word by 16-bit FIFO (first-in, first-out) buffer for capture information:
  - Change-of-flow addresses or
  - Event-only data
- Two types of breakpoints:
  - Tag breakpoints for instruction opcodes
  - Force breakpoints for any address access
- Nine trigger modes:
  - Basic: A-only, A OR B
  - Sequence: A then B
  - Full: A AND B data, A AND NOT B data
  - Event (store data): Event-only B, A then event-only B
  - Range: Inside range ($A \leq address \leq B$), outside range ($address < A$ or $address > B$)

17.2 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.

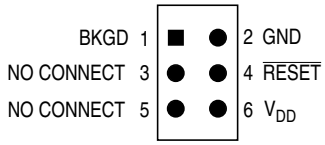


Figure 17-1. BDM Tool Connector

17.2.1 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 17.2.2, “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 17.2.2, “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.

17.2.2 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 17-2 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.

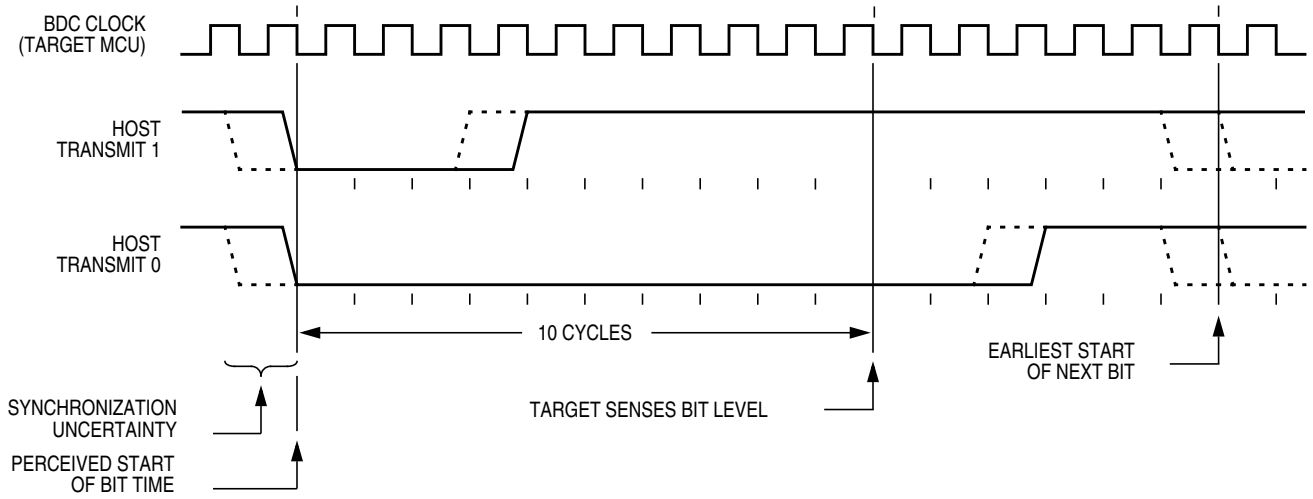


Figure 17-2. BDC Host-to-Target Serial Bit Timing

Figure 17-3 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.

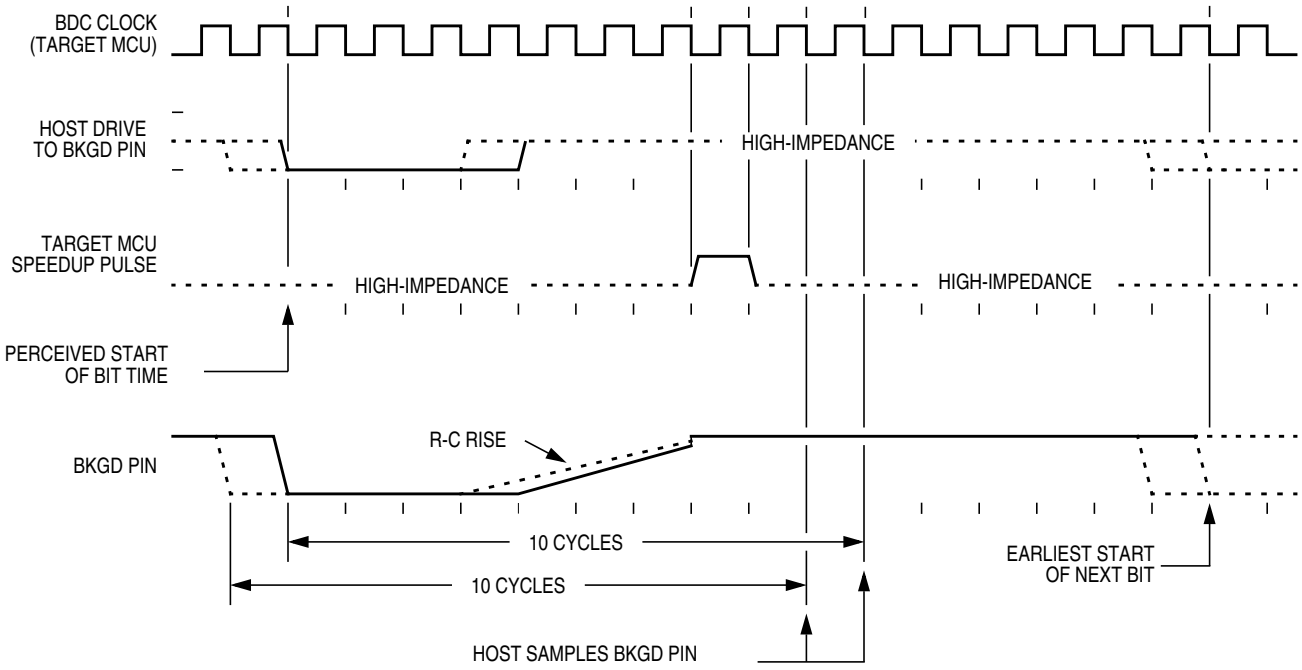


Figure 17-3. BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 17-4 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.

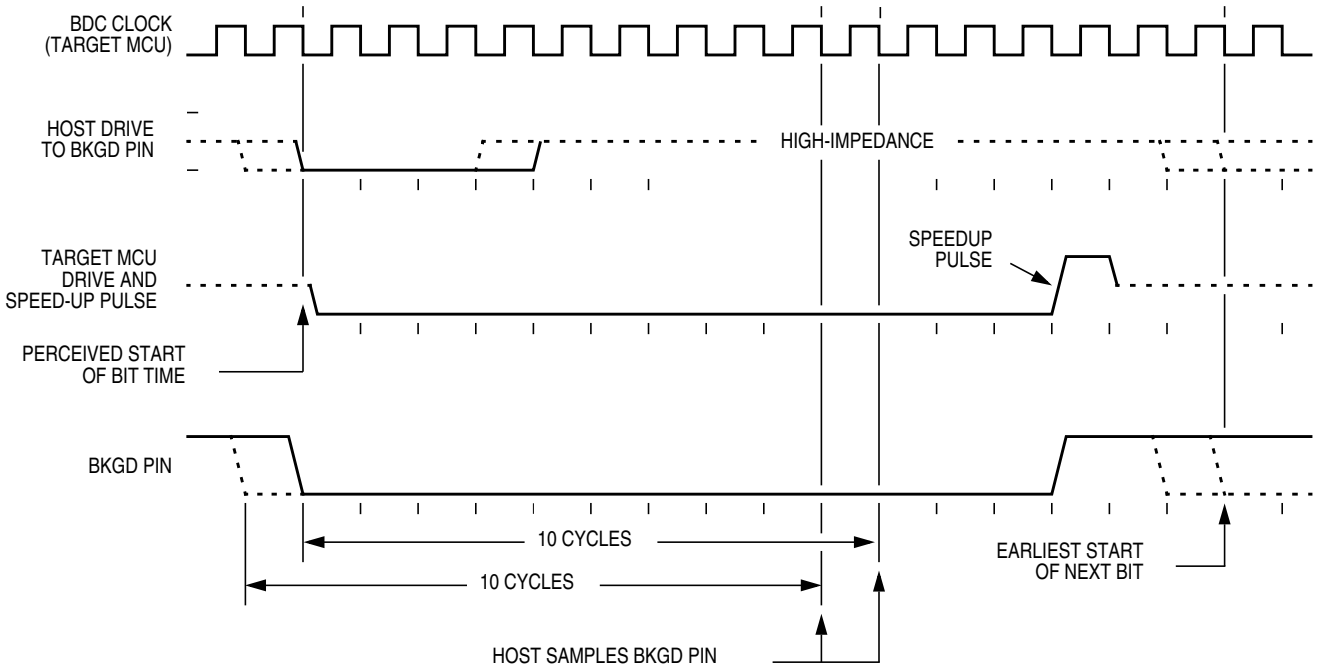


Figure 17-4. BDM Target-to-Host Serial Bit Timing (Logic 0)

17.2.3 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 17-1 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 17-1 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 17-1. 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
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.

17.2.4 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.

The on-chip debug module (DBG) includes circuitry for two additional hardware breakpoints that are more flexible than the simple breakpoint in the BDC module.

17.3 On-Chip Debug System (DBG)

Because HCS08 devices do not have external address and data buses, the most important functions of an in-circuit emulator have been built onto the chip with the MCU. The debug system consists of an 8-stage FIFO that can store address or data bus information, and a flexible trigger system to decide when to capture bus information and what information to capture. The system relies on the single-wire background debug system to access debug control registers and to read results out of the eight stage FIFO.

The debug module includes control and status registers that are accessible in the user's memory map. These registers are located in the high register space to avoid using valuable direct page memory space.

Most of the debug module's functions are used during development, and user programs rarely access any of the control and status registers for the debug module. The one exception is that the debug system can provide the means to implement a form of ROM patching. This topic is discussed in greater detail in Section 17.3.6, "Hardware Breakpoints."

17.3.1 Comparators A and B

Two 16-bit comparators (A and B) can optionally be qualified with the R/W signal and an opcode tracking circuit. Separate control bits allow you to ignore R/W for each comparator. The opcode tracking circuitry optionally allows you to specify that a trigger will occur only if the opcode at the specified address is actually executed as opposed to only being read from memory into the instruction queue. The comparators are also capable of magnitude comparisons to support the inside range and outside range trigger modes. Comparators are disabled temporarily during all BDC accesses.

The A comparator is always associated with the 16-bit CPU address. The B comparator compares to the CPU address or the 8-bit CPU data bus, depending on the trigger mode selected. Because the CPU data bus is separated into a read data bus and a write data bus, the RWAEN and RWA control bits have an additional purpose, in full address plus data comparisons they are used to decide which of these buses to use in the comparator B data bus comparisons. If RWAEN = 1 (enabled) and RWA = 0 (write), the CPU's write data bus is used. Otherwise, the CPU's read data bus is used.

The currently selected trigger mode determines what the debugger logic does when a comparator detects a qualified match condition. A match can cause:

- Generation of a breakpoint to the CPU
- Storage of data bus values into the FIFO
- Starting to store change-of-flow addresses into the FIFO (begin type trace)
- Stopping the storage of change-of-flow addresses into the FIFO (end type trace)

17.3.2 Bus Capture Information and FIFO Operation

The usual way to use the FIFO is to setup the trigger mode and other control options, then arm the debugger. When the FIFO has filled or the debugger has stopped storing data into the FIFO, you would read the information out of it in the order it was stored into the FIFO. Status bits indicate the number of words of valid information that are in the FIFO as data is stored into it. If a trace run is manually halted by writing 0 to ARM before the FIFO is full (CNT = 1:0:0:0), the information is shifted by one position and

the host must perform ((8 – CNT) – 1) dummy reads of the FIFO to advance it to the first significant entry in the FIFO.

In most trigger modes, the information stored in the FIFO consists of 16-bit change-of-flow addresses. In these cases, read DBGFH then DBGFL to get one coherent word of information out of the FIFO. Reading DBGFL (the low-order byte of the FIFO data port) causes the FIFO to shift so the next word of information is available at the FIFO data port. In the event-only trigger modes (see Section 17.3.5, “Trigger Modes”), 8-bit data information is stored into the FIFO. In these cases, the high-order half of the FIFO (DBGFH) is not used and data is read out of the FIFO by simply reading DBGFL. Each time DBGFL is read, the FIFO is shifted so the next data value is available through the FIFO data port at DBGFL.

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

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

17.3.3 Change-of-Flow Information

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

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

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

17.3.4 Tag vs. Force Breakpoints and Triggers

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

A force-type breakpoint waits for the current instruction to finish and then acts upon the breakpoint request. The usual action in response to a breakpoint is to go to active background mode rather than continuing to the next instruction in the user application program.

The tag vs. force terminology is used in two contexts within the debug module. The first context refers to breakpoint requests from the debug module to the CPU. The second refers to match signals from the comparators to the debugger control logic. When a tag-type break request is sent to the CPU, a signal is entered into the instruction queue along with the opcode so that if/when this opcode ever executes, the CPU will effectively replace the tagged opcode with a BGND opcode so the CPU goes to active background mode rather than executing the tagged instruction. When the TRGSEL control bit in the DBGT register is set to select tag-type operation, the output from comparator A or B is qualified by a block of logic in the debug module that tracks opcodes and only produces a trigger to the debugger if the opcode at the compare address is actually executed. There is separate opcode tracking logic for each comparator so more than one compare event can be tracked through the instruction queue at a time.

17.3.5 Trigger Modes

The trigger mode controls the overall behavior of a debug run. The 4-bit TRG field in the DBGT register selects one of nine trigger modes. When TRGSEL = 1 in the DBGT register, the output of the comparator must propagate through an opcode tracking circuit before triggering FIFO actions. The BEGIN bit in DBGT chooses whether the FIFO begins storing data when the qualified trigger is detected (begin trace), or the FIFO stores data in a circular fashion from the time it is armed until the qualified trigger is detected (end trigger).

A debug run is started by writing a 1 to the ARM bit in the DBGC register, which sets the ARMF flag and clears the AF and BF flags and the CNT bits in DBGS. A begin-trace debug run ends when the FIFO gets full. An end-trace run ends when the selected trigger event occurs. Any debug run can be stopped manually by writing a 0 to ARM or DBGEN in DBGC.

In all trigger modes except event-only modes, the FIFO stores change-of-flow addresses. In event-only trigger modes, the FIFO stores data in the low-order eight bits of the FIFO.

The BEGIN control bit is ignored in event-only trigger modes and all such debug runs are begin type traces. When TRGSEL = 1 to select opcode fetch triggers, it is not necessary to use R/W in comparisons because opcode tags would only apply to opcode fetches that are always read cycles. It would also be unusual to specify TRGSEL = 1 while using a full mode trigger because the opcode value is normally known at a particular address.

The following trigger mode descriptions only state the primary comparator conditions that lead to a trigger. Either comparator can usually be further qualified with R/W by setting RWAEN (RWBEN) and the corresponding RWA (RWB) value to be matched against R/W. The signal from the comparator with optional R/W qualification is used to request a CPU breakpoint if BRKEN = 1 and TAG determines whether the CPU request will be a tag request or a force request.

A-Only — Trigger when the address matches the value in comparator A

A OR B — Trigger when the address matches either the value in comparator A or the value in comparator B

A Then B — Trigger when the address matches the value in comparator B but only after the address for another cycle matched the value in comparator A. There can be any number of cycles after the A match and before the B match.

A AND B Data (Full Mode) — This is called a full mode because address, data, and R/W (optionally) must match within the same bus cycle to cause a trigger event. Comparator A checks address, the low byte of comparator B checks data, and R/W is checked against RWA if RWAEN = 1. The high-order half of comparator B is not used.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

A AND NOT B Data (Full Mode) — Address must match comparator A, data must not match the low half of comparator B, and R/W must match RWA if RWAEN = 1. All three conditions must be met within the same bus cycle to cause a trigger.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

Event-Only B (Store Data) — Trigger events occur each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

A Then Event-Only B (Store Data) — After the address has matched the value in comparator A, a trigger event occurs each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

Inside Range (A ≤ Address ≤ B) — A trigger occurs when the address is greater than or equal to the value in comparator A and less than or equal to the value in comparator B at the same time.

Outside Range (Address < A or Address > B) — A trigger occurs when the address is either less than the value in comparator A or greater than the value in comparator B.

17.3.6 Hardware Breakpoints

The BRKEN control bit in the DBGC register may be set to 1 to allow any of the trigger conditions described in Section 17.3.5, “Trigger Modes,” to be used to generate a hardware breakpoint request to the CPU. TAG in DBGC controls whether the breakpoint request will be treated as a tag-type breakpoint or a force-type breakpoint. A tag breakpoint causes the current opcode to be marked as it enters the instruction queue. If a tagged opcode reaches the end of the pipe, the CPU executes a BGND instruction to go to active background mode rather than executing the tagged opcode. A force-type breakpoint causes the CPU to finish the current instruction and then go to active background mode.

If the background mode has not been enabled (ENBDM = 1) by a serial WRITE_CONTROL command through the BKGD pin, the CPU will execute an SWI instruction instead of going to active background mode.

17.4 Register Definition

This section contains the descriptions of the BDC and DBG registers and control bits.

Refer to the high-page register summary in the device overview chapter of this data sheet for the absolute address assignments for all DBG 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.

17.4.1 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.

17.4.1.1 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 17-5. BDC Status and Control Register (BDCSCR)

Table 17-2. 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
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

Table 17-2. BDCSCR Register Field Descriptions (continued)

Field	Description
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 MC9S08LC60 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

17.4.1.2 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 17.2.4, “BDC Hardware Breakpoint.”

17.4.2 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 17-6. System Background Debug Force Reset Register (SBDFR)

Table 17-3. 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.

17.4.3 DBG Registers and Control Bits

The debug module includes nine bytes of register space for three 16-bit registers and three 8-bit control and status registers. These registers are located in the high register space of the normal memory map so they are accessible to normal application programs. These registers are rarely if ever accessed by normal user application programs with the possible exception of a ROM patching mechanism that uses the breakpoint logic.

17.4.3.1 Debug Comparator A High Register (DBGCAH)

This register contains compare value bits for the high-order eight bits of comparator A. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

17.4.3.2 Debug Comparator A Low Register (DBGCAL)

This register contains compare value bits for the low-order eight bits of comparator A. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

17.4.3.3 Debug Comparator B High Register (DBGCBH)

This register contains compare value bits for the high-order eight bits of comparator B. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

17.4.3.4 Debug Comparator B Low Register (DBGCBL)

This register contains compare value bits for the low-order eight bits of comparator B. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

17.4.3.5 Debug FIFO High Register (DBGFH)

This register provides read-only access to the high-order eight bits of the FIFO. Writes to this register have no meaning or effect. In the event-only trigger modes, the FIFO only stores data into the low-order byte of each FIFO word, so this register is not used and will read 0x00.

Reading DBGFH does not cause the FIFO to shift to the next word. When reading 16-bit words out of the FIFO, read DBGFH before reading DBGFL because reading DBGFL causes the FIFO to advance to the next word of information.

17.4.3.6 Debug FIFO Low Register (DBGFL)

This register provides read-only access to the low-order eight bits of the FIFO. Writes to this register have no meaning or effect.

Reading DBGFL causes the FIFO to shift to the next available word of information. When the debug module is operating in event-only modes, only 8-bit data is stored into the FIFO (high-order half of each FIFO word is unused). When reading 8-bit words out of the FIFO, simply read DBGFL repeatedly to get successive bytes of data from the FIFO. It isn't necessary to read DBGFH in this case.

Do not attempt to read data from the FIFO while it is still armed (after arming but before the FIFO is filled or ARMF is cleared) because the FIFO is prevented from advancing during reads of DBGFL. This can interfere with normal sequencing of reads from the FIFO.

Reading DBGFL while the debugger is not armed causes the address of the most-recently fetched opcode to be stored to the last location in the FIFO. By reading DBGFH then DBGFL periodically, external host software can develop a profile of program execution. After eight reads from the FIFO, the ninth read will return the information that was stored as a result of the first read. To use the profiling feature, read the FIFO eight times without using the data to prime the sequence and then begin using the data to get a delayed picture of what addresses were being executed. The information stored into the FIFO on reads of DBGFL (while the FIFO is not armed) is the address of the most-recently fetched opcode.

17.4.3.7 Debug Control Register (DBGC)

This register can be read or written at any time.

	7	6	5	4	3	2	1	0
R W	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
Reset	0	0	0	0	0	0	0	0

Figure 17-7. Debug Control Register (DBGC)

Table 17-4. DBGC Register Field Descriptions

Field	Description
7 DBGEN	**Debug Module Enable** — Used to enable the debug module. DBGEN cannot be set to 1 if the MCU is secure. 0 DBG disabled 1 DBG enabled
6 ARM	**Arm Control** — Controls whether the debugger is comparing and storing information in the FIFO. A write is used to set this bit (and ARMF) and completion of a debug run automatically clears it. Any debug run can be manually stopped by writing 0 to ARM or to DBGEN. 0 Debugger not armed 1 Debugger armed
5 TAG	**Tag/Force Select** — Controls whether break requests to the CPU will be tag or force type requests. If BRKEN = 0, this bit has no meaning or effect. 0 CPU breaks requested as force type requests 1 CPU breaks requested as tag type requests
4 BRKEN	**Break Enable** — Controls whether a trigger event will generate a break request to the CPU. Trigger events can cause information to be stored in the FIFO without generating a break request to the CPU. For an end trace, CPU break requests are issued to the CPU when the comparator(s) and R/W meet the trigger requirements. For a begin trace, CPU break requests are issued when the FIFO becomes full. TRGSEL does not affect the timing of CPU break requests. 0 CPU break requests not enabled 1 Triggers cause a break request to the CPU
3 RWA	**R/W Comparison Value for Comparator A** — When RWAEN = 1, this bit determines whether a read or a write access qualifies comparator A. When RWAEN = 0, RWA and the R/W signal do not affect comparator A. 0 Comparator A can only match on a write cycle 1 Comparator A can only match on a read cycle
2 RWAEN	**Enable R/W for Comparator A** — Controls whether the level of R/W is considered for a comparator A match. 0 R/W is not used in comparison A 1 R/W is used in comparison A
1 RWB	**R/W Comparison Value for Comparator B** — When RWBEN = 1, this bit determines whether a read or a write access qualifies comparator B. When RWBEN = 0, RWB and the R/W signal do not affect comparator B. 0 Comparator B can match only on a write cycle 1 Comparator B can match only on a read cycle
0 RWBEN	**Enable R/W for Comparator B** — Controls whether the level of R/W is considered for a comparator B match. 0 R/W is not used in comparison B 1 R/W is used in comparison B

17.4.3.8 Debug Trigger Register (DBGT)

This register can be read any time, but may be written only if ARM = 0, except bits 4 and 5 are hard-wired to 0s.

	7	6	5	4	3	2	1	0
R	TRGSEL	BEGIN	0	0	TRG3	TRG2	TRG1	TRG0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 17-8. Debug Trigger Register (DBGT)

Table 17-5. DBGT Register Field Descriptions

Field	Description
7 TRGSEL	**Trigger Type** — Controls whether the match outputs from comparators A and B are qualified with the opcode tracking logic in the debug module. If TRGSEL is set, a match signal from comparator A or B must propagate through the opcode tracking logic and a trigger event is only signalled to the FIFO logic if the opcode at the match address is actually executed. 0 Trigger on access to compare address (force) 1 Trigger if opcode at compare address is executed (tag)
6 BEGIN	**Begin/End Trigger Select** — Controls whether the FIFO starts filling at a trigger or fills in a circular manner until a trigger ends the capture of information. In event-only trigger modes, this bit is ignored and all debug runs are assumed to be begin traces. 0 Data stored in FIFO until trigger (end trace) 1 Trigger initiates data storage (begin trace)
3:0 TRG[3:0]	**Select Trigger Mode** — Selects one of nine triggering modes, as described below. 0000 A-only 0001 A OR B 0010 A Then B 0011 Event-only B (store data) 0100 A then event-only B (store data) 0101 A AND B data (full mode) 0110 A AND NOT B data (full mode) 0111 Inside range: A ≤ address ≤ B 1000 Outside range: address < A or address > B 1001 – 1111 (No trigger)

17.4.3.9 Debug Status Register (DBGS)

This is a read-only status register.

	7	6	5	4	3	2	1	0
R	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 17-9. Debug Status Register (DBGS)

Table 17-6. DBGS Register Field Descriptions

Field	Description
7 AF	**Trigger Match A Flag** — AF is cleared at the start of a debug run and indicates whether a trigger match A condition was met since arming. 0 Comparator A has not matched 1 Comparator A match
6 BF	**Trigger Match B Flag** — BF is cleared at the start of a debug run and indicates whether a trigger match B condition was met since arming. 0 Comparator B has not matched 1 Comparator B match
5 ARMF	**Arm Flag** — While DBGEN = 1, this status bit is a read-only image of ARM in DBGC. This bit is set by writing 1 to the ARM control bit in DBGC (while DBGEN = 1) and is automatically cleared at the end of a debug run. A debug run is completed when the FIFO is full (begin trace) or when a trigger event is detected (end trace). A debug run can also be ended manually by writing 0 to ARM or DBGEN in DBGC. 0 Debugger not armed 1 Debugger armed
3:0 CNT[3:0]	**FIFO Valid Count** — These bits are cleared at the start of a debug run and indicate the number of words of valid data in the FIFO at the end of a debug run. The value in CNT does not decrement as data is read out of the FIFO. The external debug host is responsible for keeping track of the count as information is read out of the FIFO. 0000 Number of valid words in FIFO = No valid data 0001 Number of valid words in FIFO = 1 0010 Number of valid words in FIFO = 2 0011 Number of valid words in FIFO = 3 0100 Number of valid words in FIFO = 4 0101 Number of valid words in FIFO = 5 0110 Number of valid words in FIFO = 6 0111 Number of valid words in FIFO = 7 1000 Number of valid words in FIFO = 8

Appendix A
Electrical Characteristics

A.1 Introduction

This section contains electrical and timing specifications.

A.2 Absolute Maximum Ratings

Absolute maximum ratings are stress ratings only, and functional operation at the maxima is not guaranteed. Stress beyond the limits specified in Table A-1 may affect device reliability or cause permanent damage to the device. For functional operating conditions, refer to the remaining tables in this section.

This device contains circuitry protecting against damage due to high static voltage or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (for instance, either V_{SS} or V_{DD}) or the programmable pull-up resistor associated with the pin is enabled.

Table A-1. Absolute Maximum Ratings

Rating	Symbol	Value	Unit
Supply voltage	V_{DD}	–0.3 to +3.8	V
Maximum current into V_{DD}	I_{DD}	120	mA
Digital input voltage	V_{In}	–0.3 to V_{DD} + 0.3	V
Instantaneous maximum current Single pin limit (applies to all port pins)(1), (2), (3)	I_D	± 25	mA
Storage temperature range	T_{stg}	–55 to 150	°C

1 Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive (V_{DD}) and negative (V_{SS}) clamp voltages, then use the larger of the two resistance values.

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

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

A.3 Thermal Characteristics

This section provides information about operating temperature range, power dissipation, and package thermal resistance. Power dissipation on I/O pins is usually small compared to the power dissipation in on-chip logic and voltage regulator circuits and it is user-determined rather than being controlled by the MCU design. In order to take $P_{I/O}$ into account in power calculations, determine the difference between actual pin voltage and V_{SS} or V_{DD} and multiply by the pin current for each I/O pin. Except in cases of unusually high pin current (heavy loads), the difference between pin voltage and V_{SS} or V_{DD} will be very small.

Table A-2. Thermal Characteristics

Rating	Symbol	Value	Unit	
Operating temperature range (packaged)	T_A	-40 to 85	°C	
Thermal resistance 80-pin LQFP 1s 2s2p 64-pin LQFP 1s 2s2p	θ_{JA}[1], [2], [3], [4]	 64 49 66 47	°C/W	

1 Junction temperature is a function of die size, on-chip power dissipation, package thermal resistance, mounting site (board) temperature, ambient temperature, airflow, power dissipation of other components on the board, and board thermal resistance.

2 Junction to Ambient Natural Convection

3 1s - Single Layer Board, one signal layer

4 2s2p - Four Layer Board, 2 signal and 2 power layers

The average chip-junction temperature (T_J) in °C can be obtained from:

$$T_J = T_A + (P_D \times \theta_{JA}) \qquad \textit{Eqn. A-1}$$

where:

T_A = Ambient temperature, °C
θ_{JA} = Package thermal resistance, junction-to-ambient, °C/W
$P_D = P_{int} + P_{I/O}$
$P_{int} = I_{DD} \times V_{DD}$, Watts — chip internal power
$P_{I/O}$ = Power dissipation on input and output pins — user determined

For most applications, $P_{I/O} << P_{int}$ and can be neglected. An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

$$P_D = K \div (T_J + 273°C) \qquad \textit{Eqn. A-2}$$

Solving Equation A-1 and Equation A-2 for K gives:

$$K = P_D \times (T_A + 273°C) + \theta_{JA} \times (P_D)^2 \qquad \textit{Eqn. A-3}$$

where K is a constant pertaining to the particular part. K can be determined from Equation A-3 by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations 1 and 2 iteratively for any value of T_A.

A.4 Electrostatic Discharge (ESD) Protection Characteristics

Although damage from static discharge is much less common on these devices than on early CMOS circuits, normal handling precautions should be used to avoid exposure to static discharge. Qualification tests are performed to ensure that these devices can withstand exposure to reasonable levels of static without suffering any permanent damage. All ESD testing is in conformity with CDF-AEC-Q00 Stress Test Qualification for Automotive Grade Integrated Circuits. (http://www.aecouncil.com/) This device was qualified to AEC-Q100 Rev E. A device is considered to have failed if, after exposure to ESD pulses, the device no longer meets the device specification requirements. Complete dc parametric and functional testing is performed per the applicable device specification at room temperature followed by hot temperature, unless specified otherwise in the device specification.

Table A-3. ESD and Latch-up Test Conditions

Model	Description	Symbol	Value	Unit
Human Body	Series resistance	R1	1500	Ω
	Storage capacitance	C	100	pF
	Number of pulses per pin	—	3	
Machine	Series resistance	R1	0	Ω
	Storage capacitance	C	200	pF
	Number of pulses per pin	—	3	
Latch-up	Minimum input voltage limit		1.8	V
	Maximum input voltage limit		3.6	V

Table A-4. ESD and Latch-Up Protection Characteristics

No.	Rating[1]	Symbol	Min	Max	Unit
1	Human body model (HBM)	V_{HBM}	± 2000	—	V
2	Machine model (MM)	V_{MM}	± 200	—	V
3	Charge device model (CDM)	V_{CDM}	± 500	—	V
4	Latch-up current at T_A = 85°C	I_{LAT}	± 100	—	mA

1 Parameter is achieved by design characterization on a small sample size from typical devices under typical conditions unless otherwise noted.

A.5 DC Characteristics

This section includes information about power supply requirements, I/O pin characteristics, and power supply current in various operating modes.

Table A-5. DC Characteristics (Sheet 1 of 2)
(Temperature Range = -40 to 85°C Ambient)

Parameter	Symbol	Min	Typical[1]	Max	Unit
Supply voltage (run, wait and stop modes.)	V_{DD}	1.8	—	3.6	V
Minimum RAM retention supply voltage applied to V_{DD}	V_{RAM}	1.0[2]	—	—	V
Low-voltage detection threshold — high range (V_{DD} falling) (V_{DD} rising)	V_{LVDH}	 2.02 2.07	 2.15 2.23	 2.3 2.33	V
Low-voltage detection threshold — low range (V_{DD} falling) (V_{DD} rising)	V_{LVDL}	 1.76 1.8	 1.88 1.93	 1.98 2.04	V
Low-voltage warning threshold — high range (V_{DD} falling) (V_{DD} rising)	V_{LVWH}	 2.32 2.32	 2.45 2.48	 2.6 2.6	V
Low-voltage warning threshold — low range (V_{DD} falling) (V_{DD} rising)	V_{LVWL}	 2.02 2.02	 2.15 2.21	 2.3 2.33	V
Power on reset (POR) re-arm voltage[2] Mode = stop Mode = run and Wait	V_{Rearm}	 0.20 0.50	 0.30 0.80	 0.40 1.2	V
Bandgap voltage reference,	V_{BG}	1.18	1.20	1.21	V
Input high voltage (V_{DD} > 2.3 V) (all digital inputs)	V_{IH}	$0.70 \times V_{DD}$		—	V
Input high voltage (1.8 V ≤ V_{DD} ≤ 2.3 V) (all digital inputs)	V_{IH}	$0.85 \times V_{DD}$		—	V
Input low voltage (V_{DD} > 2.3 V) (all digital inputs)	V_{IL}	—		$0.35 \times V_{DD}$	V
Input low voltage (1.8 V ≤ V_{DD} ≤ 2.3 V) (all digital inputs)	V_{IL}	—		$0.30 \times V_{DD}$	V
Input hysteresis (all digital inputs)	V_{hys}	$0.06 \times V_{DD}$		—	V
Input leakage current (per pin) V_{In} = V_{DD} or $V_{SS,}$ all input only pins	$\lvert I_{In} \rvert$	—	0.025	1	µA
High impedance (off-state) leakage current (per pin) V_{In} = V_{DD} or V_{SS}, all input/output (all except PTC7) V_{In} = V_{DD} or V_{SS}, all input/output (PTC7 only)	$\lvert I_{OZ} \rvert$	—	 0.025 0.025	 1 2	µA
Internal pullup and pulldown resistors[3] (all port pins and IRQ)	R_{PU}	17.5		52.5	kΩ
Internal pulldown resistors (all port pins and IRQ)	R_{PD}	17.5		52.5	kΩ

Table A-5. DC Characteristics (Sheet 2 of 2)
(Temperature Range = -40 to 85°C Ambient)

Parameter	Symbol	Min	Typical[1]	Max	Unit
Output high voltage ($V_{DD} \geq 1.8$ V) I_{OH} = –2 mA (ports A, B, and C)	V_{OH}	V_{DD} – 0.5		—	V
Output high voltage (all port pins) I_{OH} = –10 mA ($V_{DD} \geq 2.7$ V) I_{OH} = –6 mA ($V_{DD} \geq 2.3$ V) I_{OH} = –3 mA ($V_{DD} \geq 1.8$ V)		V_{DD} – 0.5		— — —	
Maximum total I_{OH} for all port pins	$\lvert I_{OHT} \rvert$	—		60	mA
Output low voltage ($V_{DD} \geq 1.8$ V) I_{OL} = 2.0 mA (ports A, B, and C)	V_{OL}	—		0.5	V
Output low voltage (all port pins) I_{OL} = 10.0 mA ($V_{DD} \geq 2.7$ V) I_{OL} = 6 mA ($V_{DD} \geq 2.3$ V) I_{OL} = 3 mA ($V_{DD} \geq 1.8$ V)		— — —		0.5 0.5 0.5	
Maximum total I_{OL} for all port pins	I_{OLT}	—		60	mA
dc injection current[4], [5], [6], [7], [8] $V_{IN} < V_{SS}$, $V_{IN} > V_{DD}$ Single pin limit Total MCU limit, includes sum of all stressed pins	$\lvert I_{IC} \rvert$	 — —		 0.2 5	 mA mA
Input capacitance (all non-supply pins)[2]	C_{In}	—		7	pF

1 Typicals are measured at 25°C.

2 This parameter is characterized and not tested on each device.

3 Measurement condition for pull resistors: $V_{In} = V_{SS}$ for pullup and $V_{In} = V_{DD}$ for pulldown.

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

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

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

7 This parameter is characterized and not tested on each device.

8 IRQ does not have a clamp diode to V_{DD}. Do not drive IRQ above V_{DD}.

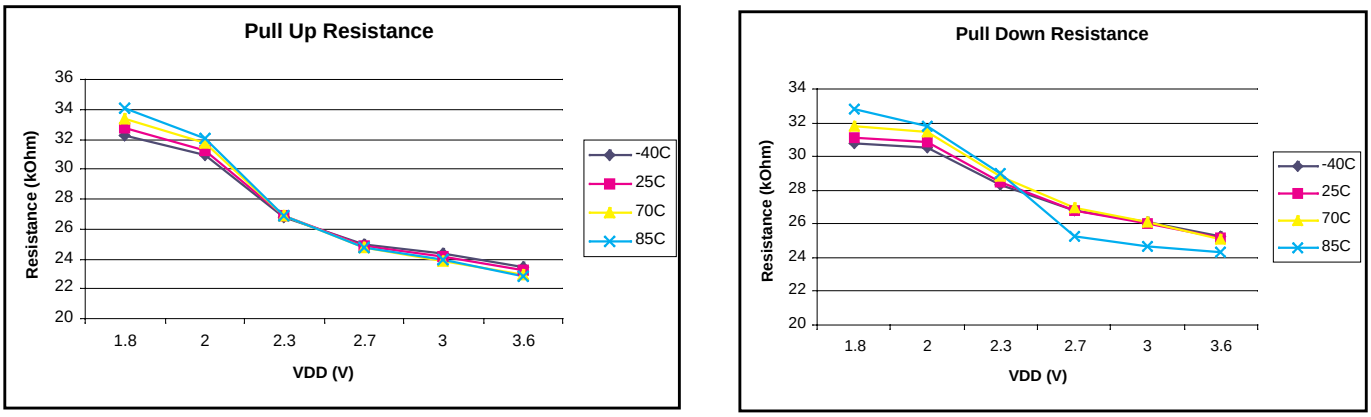


Figure A-1. Pullup and Pulldown Typical Resistor Values (V_{DD} = 3.0 V)

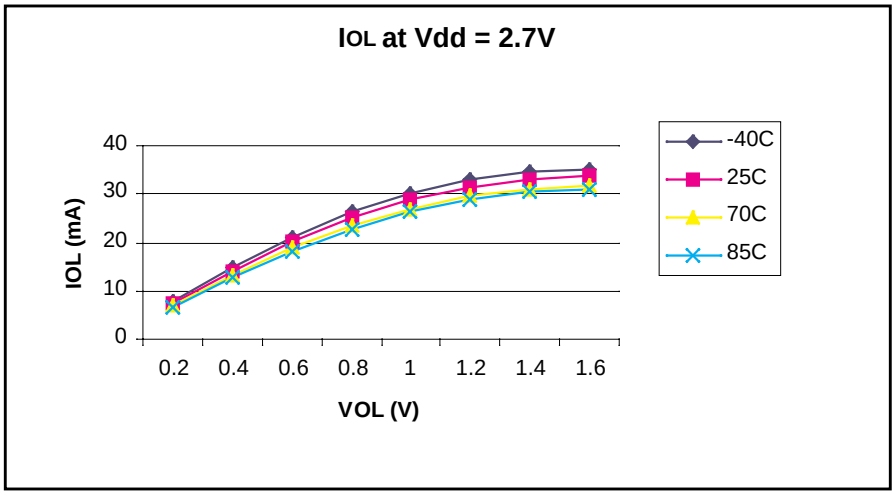


Figure A-2. Typical Low-Side Driver (Sink) Characteristics (Ports A, B, and C)
Typical I_{OL} vs. V_{OL} at V_{DD} = 2.7 V

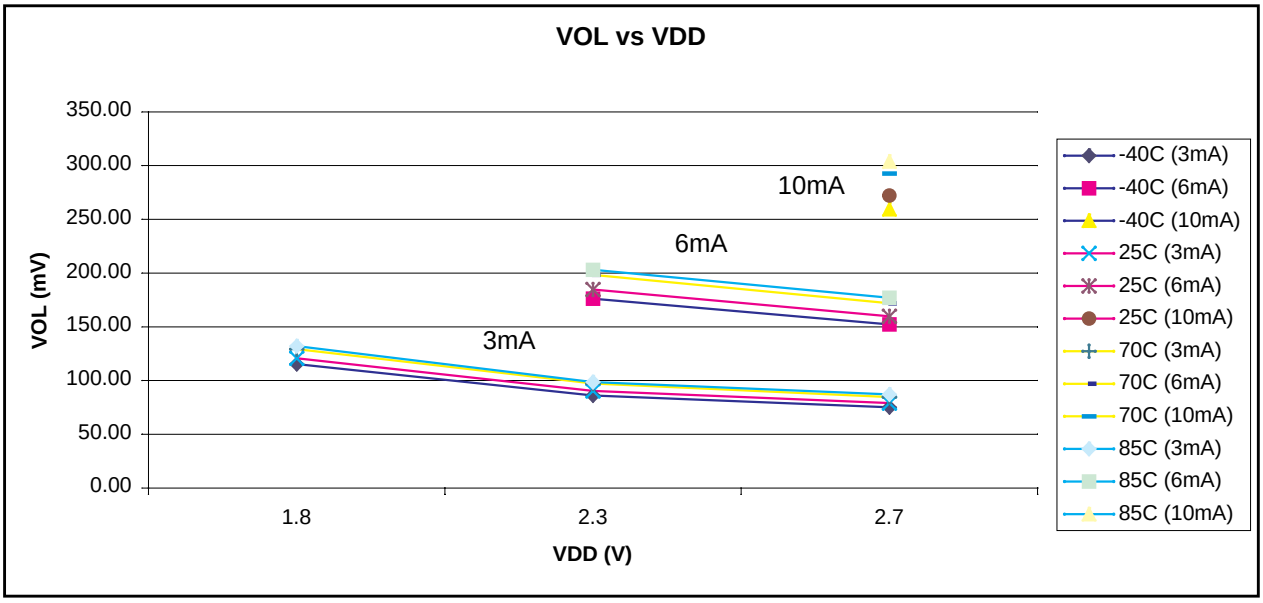


Figure A-3. Typical Low-Side Driver (Sink) Characteristics (Ports A, B, and C)
Typical V_{OL} vs. V_{DD}

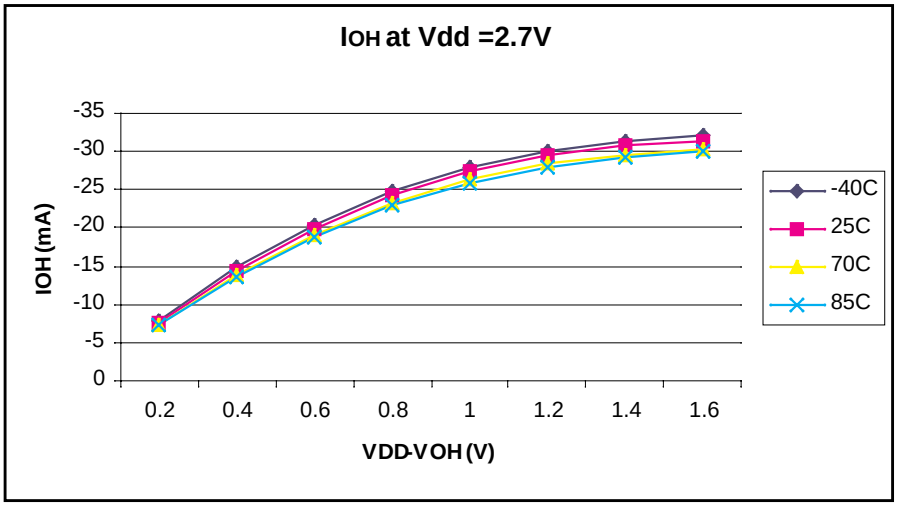


Figure A-4. Typical Low-Side Driver (Source) Characteristics (Ports A, B, and C)
Typical I_{OH} vs. $V_{DD} - V_{OH}$ at V_{DD} = 2.7 V

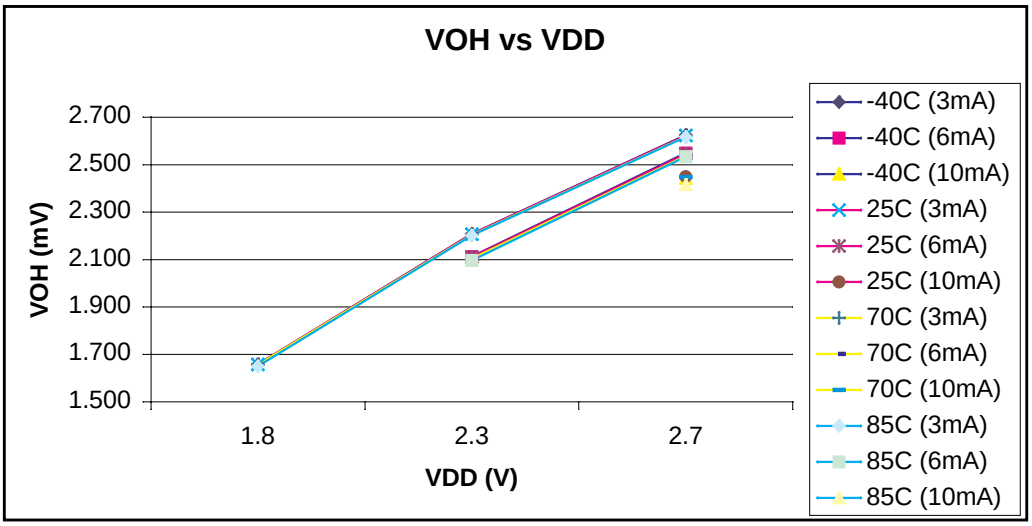


Figure A-5. Typical Low-Side Driver (Sink) Characteristics (Ports A, B, and C)
Typical V_{OH} vs. V_{DD} at Spec I_{OH}

A.6 Supply Current Characteristics

Table A-6. Supply Current Characteristics

Parameter	Symbol	V_{DD} (V)	Typical[1]	Max[2]	Temp. (°C)
Run supply current[3] measured at (CPU clock = 2 MHz, f_{Bus} = 1 MHz)	RI_{DD}	3	800 µA	1.7 mA	85
		2	600 µA	1.3 mA	85
Run supply current [3] measured at (CPU clock = 16 MHz, f_{Bus} = 8 MHz)	RI_{DD}	3	5 mA	8.7 mA	85
		2	4 mA	6.3 mA	85
Run supply current [3] measured at (CPU clock = 40 MHz, f_{Bus} = 20 MHz)	RI_{DD}	3	12 mA	17.2 mA	85
		2	10 mA	15.5 mA	85
Stop1 mode supply current	$S1I_{DD}$	3	770 nA	10 µA	85
		2	600 nA	10 µA	85
Stop2 mode supply current	$S2I_{DD}$	3	770 nA	12 µA	85
		2	600 nA	12 µA	85
Stop3 mode supply current[4]	$S3I_{DD}$	3	840 nA	20 µA	85
		2	660 nA	20 µA	85
Internal RTI [clock?] adder to stop2 or stop3[5]		3	350 nA		85
		2	350 nA		85
LVI adder to stop3 (LVDSE = LVDE = 1)		3	75 µA		85
		2	70 µA		85

Table A-6. Supply Current Characteristics (continued)

Parameter	Symbol	V_{DD} (V)	Typical(1)	Max(2)	Temp. (°C)
Adder to stop3 for oscillator enabled(6) (OSCSTEN =1) (32 kHz)		3	4 µA		85
		2	3.5 µA		85
Adder for loss-of-clock detection (LOCD = 0)		3	9 µA		85

1 Typicals are measured at 25°C. See Table A-6 through Table A-9 for typical curves across voltage/temperature.

2 Values given here are preliminary estimates prior to completing characterization.

3 All modules except ADC active, ICG configured for FBE, and does not include any dc loads on port pins.

4 With LCD and external clock module disabled.

5 Most customers are expected to find that auto-wakeup from stop2 or stop3 can be used instead of the higher current wait mode. Wait mode typical is 560 µA at 3 V and 422 µA at 2V with f_{Bus} = 1 MHz.

6 Values given under the following conditions: low range operation (RANGE = 0), low power mode (HGO = 0), clock monitor disabled (LOCD = 1).

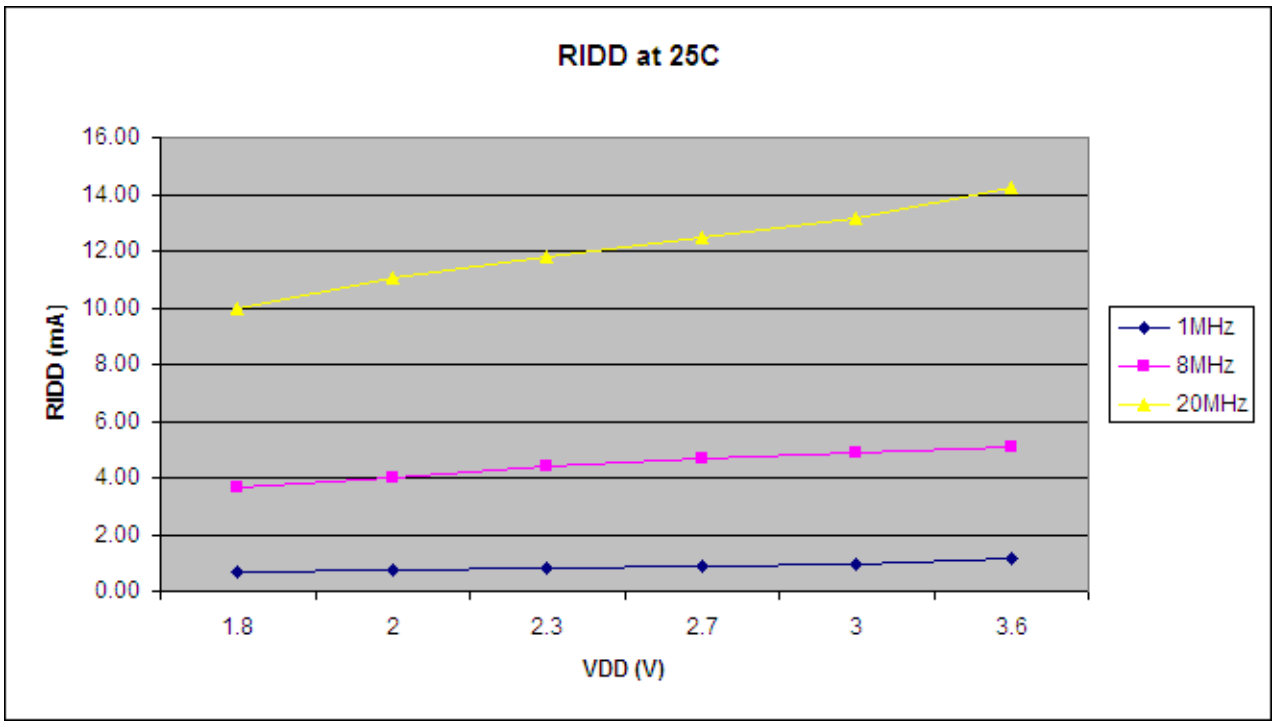


Figure A-6. Typical Run I_{DD} for FBE FEE Mode, I_{DD} vs. V_{DD}

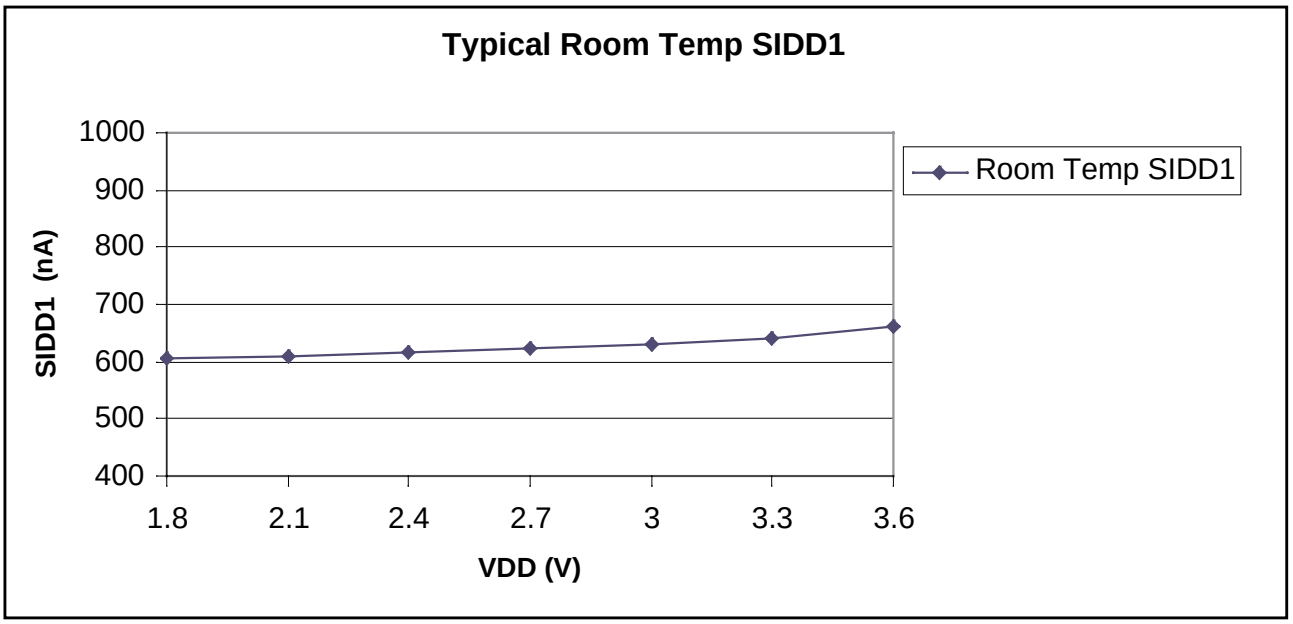


Figure A-7. Typical Stop1 I_{DD}

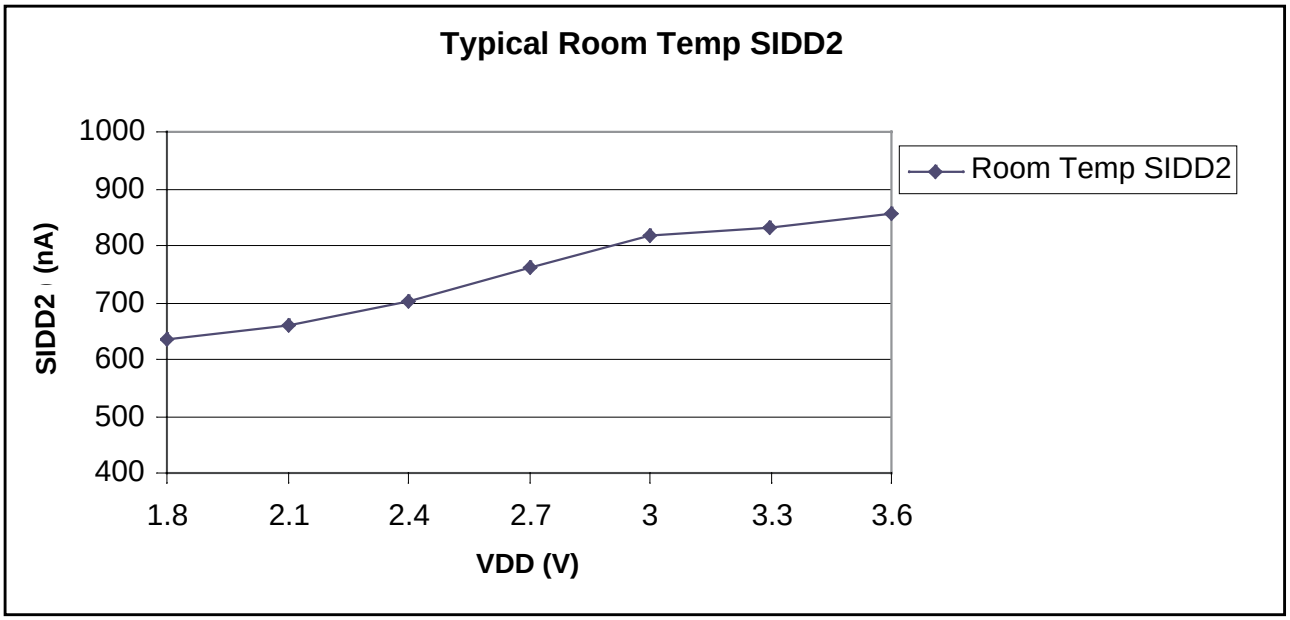


Figure A-8. Typical Stop 2 I_{DD}

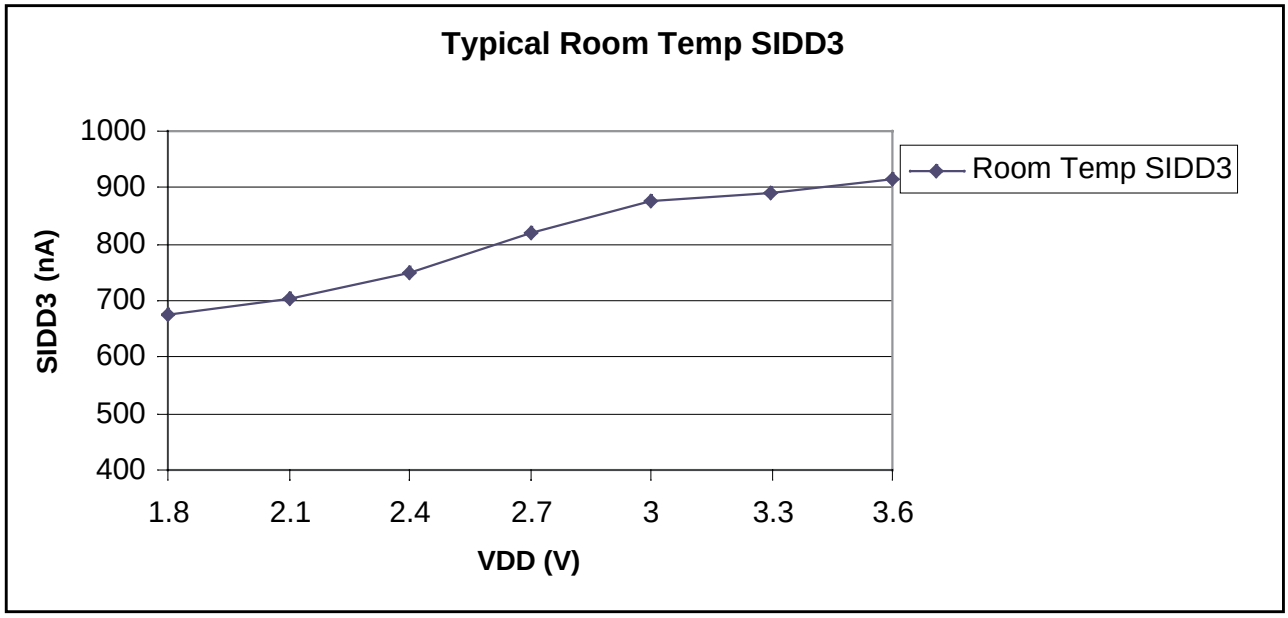


Figure A-9. Typical Stop3 I_{DD}

A.7 ADC Characteristics

Table A-7. 3 Volt 12-bit ADC Operating Conditions

Characteristic	Conditions	Symbol	Min	Typ[1]	Max	Unit	Comment
Supply voltage	Absolute	V_{DDAD}	1.8	—	3.6	V	
	Delta to V_{DD} (V_{DD}-V_{DDAD})[2]	ΔV_{DDAD}	-100	0	+100	mV	
Ground voltage	Delta to V_{SS} (V_{SS}-V_{SSAD})[2]	ΔV_{SSAD}	-100	0	+100	mV	
Ref Voltage High (80-pin package only)		V_{REFH}	1.8	V_{DDAD}	V_{DDAD}	V	
Ref Voltage Low (80-pin package only)		V_{REFL}	V_{SSAD}	V_{SSAD}	V_{SSAD}	V	
Supply Current	Stop, Reset, Module Off	I_{DDAD}	—	0.007	0.8	µA	
Input Voltage		V_{ADIN}	V_{REFL}	—	V_{REFH}	V	
Input Capacitance		C_{ADIN}	—	4.5	5.5	pF	
Input Resistance		R_{ADIN}	—	5	7	kΩ	

Table A-7. 3 Volt 12-bit ADC Operating Conditions

Characteristic	Conditions	Symbol	Min	Typ[1]	Max	Unit	Comment
Analog Source Resistance	12 bit mode f_{ADCK} > 4MHz f_{ADCK} < 4MHz	R_{AS}	 — —	 — —	 2 5	kΩ	External to MCU
	10 bit mode f_{ADCK} > 4MHz f_{ADCK} < 4MHz		 — —	 — —	 5 10		
	8 bit mode (all valid f_{ADCK})		—	—	10		
ADC Conversion Clock Freq.	High Speed (ADLPC=0)	f_{ADCK}	0.4	—	8.0	MHz	
	Low Power (ADLPC=1)		0.4	—	4.0		

1 Typical values assume V_{DDAD} = 3.0V, Temp = 25°C, f_{ADCK}=1.0MHz unless otherwise stated. Typical values are for reference only and are not tested in production.

2 DC potential difference.

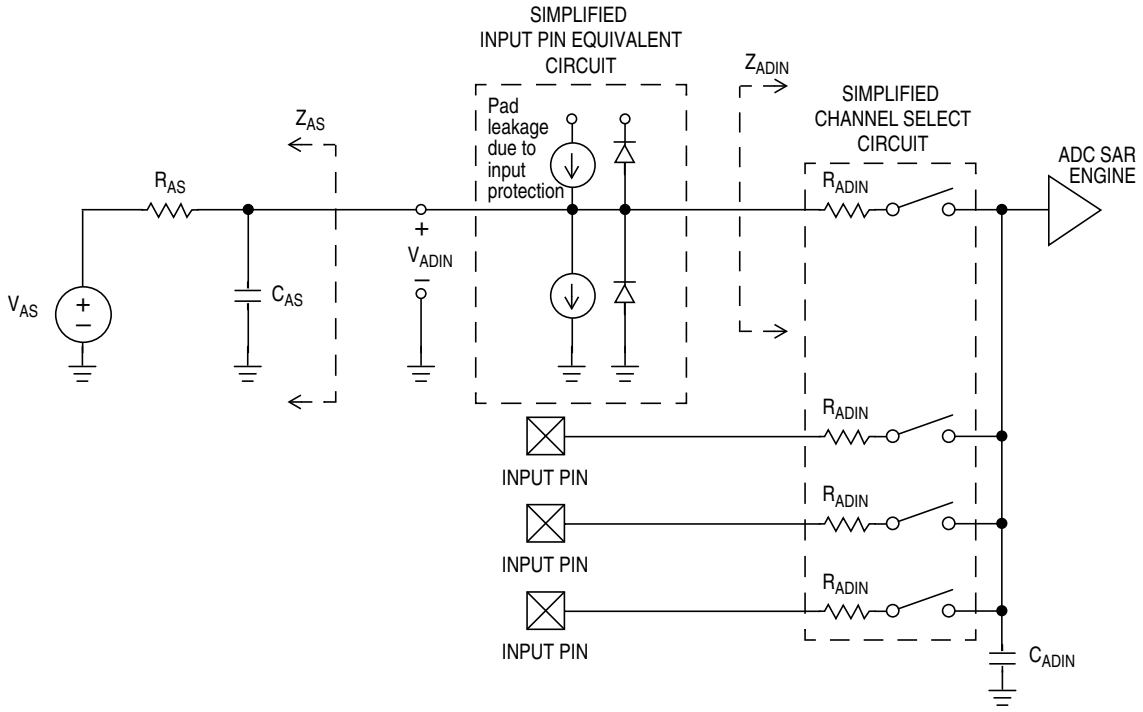


Figure A-10. ADC Input Impedance Equivalency Diagram

Table A-8. 3 Volt 12-bit ADC Characteristics (V_{REFH} = V_{DDAD}, V_{REFL} = V_{SSAD})

Characteristic	Conditions	C	Symbol	Min	Typ[1]	Max	Unit	Comment
Supply Current ADLPC=1 ADLSMP=1 ADCO=1		T	I_{DDAD}	—	120	—	μA	

Table A-8. 3 Volt 12-bit ADC Characteristics ($V_{REFH} = V_{DDAD}$, $V_{REFL} = V_{SSAD}$) (continued)

Characteristic	Conditions	C	Symbol	Min	Typ[1]	Max	Unit	Comment
Supply Current ADLPC=1 ADLSMP=0 ADCO=1		T	I_{DDAD}	—	202	—	µA	
Supply Current ADLPC=0 ADLSMP=1 ADCO=1		T	I_{DDAD}	—	288	—	µA	
Supply Current ADLPC=0 ADLSMP=0 ADCO=1		T	I_{DDAD}	—	532	—	µA	
	$V_{DDAD} \leq 3.6V$	P		—	—	1	mA	
ADC Asynchronous Clock Source	High Speed (ADLPC=0)	P	f_{ADACK}	2	3.3	5	MHz	$t_{ADACK} = 1/f_{ADACK}$
	Low Power (ADLPC=1)			1.25	2	3.3		
Conversion Time (Including sample time)	Short Sample (ADLSMP=0)	P	t_{ADC}	—	20	—	ADCK cycles	See Table A-7 for conversion time variances
	Long Sample (ADLSMP=1)			—	40	—		
Sample Time	Short Sample (ADLSMP=0)	P	t_{ADS}	—	3.5	—	ADCK cycles	
	Long Sample (ADLSMP=1)			—	23.5	—		
Total Unadjusted Error (80-pin package only)	12 bit mode	T	E_{TUE}	—	±3.0	—	LSB^2	Includes quantization
	10 bit mode	P		—	±1	±2.5		
	8 bit mode	P		—	±0.5	±1.0		
Total Unadjusted Error (64-pin package only)	12 bit mode	T	E_{TUE}				LSB^2	Includes quantization
	10 bit mode	P		—	±1.5	±3.5		
	8 bit mode	P		—	±0.7	±1.5		
Differential Non-Linearity	12 bit mode	T	DNL	—	±1.75	—	LSB[2]	
	10 bit mode[3]	P		—	±0.5	±1.0		
	8 bit mode[3]	P		—	±0.3	±0.5		
Integral Non-Linearity	12 bit mode	T	INL	—	±3	—	LSB^2	
	10 bit mode	T		—	±2.5	±3.5		
	8 bit mode	T		—	±1.5	±2		
Zero-Scale Error (80-pin package only)	12 bit mode	T	E_{ZS}	—	±1.5	—	LSB^2	$V_{ADIN} = V_{SSAD}$
	10 bit mode	P		—	±0.5	±1.5		
	8 bit mode	P		—	±0.5	±0.5		

Table A-8. 3 Volt 12-bit ADC Characteristics (V_{REFH} = V_{DDAD}, V_{REFL} = V_{SSAD}) (continued)

Characteristic	Conditions	C	Symbol	Min	Typ(1)	Max	Unit	Comment
Zero-Scale Error (64-pin package only)	12 bit mode	T	E_{ZS}				LSB^2	V_{ADIN} = V_{SSAD}
	10 bit mode	P		—	±1.5	±2.1		
	8 bit mode	P		—	±0.5	±0.7		
Full-Scale Error (80-pin package only)	12 bit mode	T	E_{FS}	—	±1.0	—	LSB^2	V_{ADIN} = V_{DDAD}
	10 bit mode	P		—	±0.5	±1		
	8 bit mode	P		—	±0.5	±0.5		
Full-Scale Error (64-pin package only)	12 bit mode	T	E_{FS}				LSB^2	V_{ADIN} = V_{DDAD}
	10 bit mode	P		—	±1	±1.5		
	8 bit mode	P		—	±0.5	±0.5		
Quantization Error	12 bit mode	D	E_Q	—	±0.5	—	LSB^2	
	10 bit mode			—	—	±0.5		
	8 bit mode			—	—	±0.5		
Input Leakage Error	12 bit mode	D	E_{IL}	—	±2	—	LSB^2	Pad leakage(4) * R_{AS}
	10 bit mode			—	±0.2	±4		
	8 bit mode			—	±0.1	±1.2		
Temp Sensor Slope	-40°C– 25°C	D	m	—	1.646	—	mV/°C	
	25°C– 125°C			—	1.769	—		
Temp Sensor Voltage	25°C	D	V_{TEMP25}	—	701.2	—	mV	

1 Typical values assume V_{DDAD} = 3.0V, Temp = 25°C, f_{ADCK}=1.0MHz unless otherwise stated. Typical values are for reference only and are not tested in production.

2 1 LSB = $(V_{REFH} - V_{REFL})/2^N$

3 Monotonicity and No-Missing-Codes guaranteed in 10 bit and 8 bit modes

4 Based on input pad leakage current. Refer to pad electricals.

A.8 LCD Characteristics

Table A-9. LCD Electricals, 3-V Glass

Characteristic	Symbol	Min	Typ	Max	Unit
LCD Supply Voltage	V_{LCD}	0.9	-	1.8	V
LCD Frame Frequency	f_{Rame}	25	30	100	Hz

Note: Current consumption data based on using the external 32-kHz oscillator with LCD configured using the low-power wave forms option, a 1/4 duty, and a 32-Hz frame frequency. C_{LCD} = C_{BYLCD} = 100 nF; 160 segment 2000 pF LCD panel.

Table A-9. LCD Electricals, 3-V Glass (continued)

Characteristic	Symbol	Min	Typ	Max	Unit
LCD Charge Pump Capacitance	C_{LCD}	100	100	433	nF
LCD Bypass Capacitance	C_{BYLCD}	100	100	433	nF
LCD Current Consumption **V_{LL2} connect to V_{DD}; V_{DD} = 2 V**					
all segments off, all FP enabled	I_{segoff}	—	0.35	—	µA
half of segments on	$I_{seghalf}$	—	1	—	µA
all segments on	$I_{segallon}$	—	1	—	µA
LCD Current Consumption **Trippler Buffered Mode; V_{DD} = 3 V, V_{LCD} = 1.0 V**					
all segments off, all FP enabled	I_{segoff}	—	2.65	—	µA
half of segments on	$I_{seghalf}$	—	3.8	—	µA
all segments on	$I_{segallon}$	—	3.6	—	µA
LCD Current Consumption **Trippler Bypassed Mode; V_{DD} = 3 V, V_{LCD} = 1.0 V**					
all segments off, all FP enabled	I_{segoff}	—	0.2	—	µA
half of segments on	$I_{seghalf}$	—	0.2	—	µA
all segments on	$I_{segallon}$	—	0.2	—	µA
LCD Current Consumption **V_{LL3} connect to V_{DD}; V_{DD} = 3 V**					
all segments off, all FP enabled	I_{segoff}	—	0.15	-	µA
half of segments on	$I_{seghalf}$	-	0.75	-	µA
all segments on	$I_{segallon}$	-	0.7	-	µA
LCD Current Consumption **Doubler Buffered Mode; V_{LCD} = 1.5 V**					
all segments off, all FP enabled	I_{segoff}	-	2.7	-	µA
half of segments on	$I_{seghalf}$	-	3.85	-	µA
all segments on	$I_{segallon}$	-	3.6	-	µA

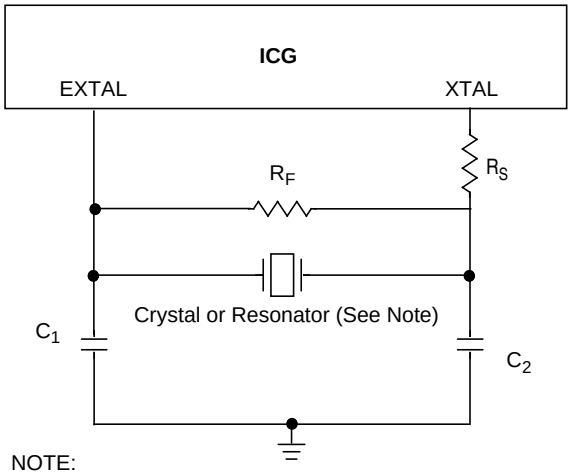
Note: Current consumption data based on using the external 32-kHz oscillator with LCD configured using the low-power wave forms option, a 1/4 duty, and a 32-Hz frame frequency. C_{LCD} = C_{BYLCD} = 100 nF; 160 segment 2000 pF LCD panel.

Table A-10. LCD Electricals, 5 V Glass

Characteristic	Symbol	Min	Typ	Max	Unit
LCD Supply Voltage	V_{LCD}	0.9	—	1.8	V
LCD Frame Frequency	f_{Rame}	25	30	100	Hz
LCD Charge Pump Capacitance	C_{LCD}	100	100	433	nF
LCD Bypass Capacitance	C_{BYLCD}	100	100	433	nF
LCD Current Consumption **V_{LL2} connect to V_{DD}; V_{DD} = 3.3 V**					
all segments off, all FP enabled	I_{segoff}	—	0.2	—	µA
half of segments on	$I_{seghalf}$	—	0.95	—	µA
all segments on	$I_{segallon}$	—	0.67	—	µA
LCD Current Consumption **Trippler Buffered Mode; V_{DD} = 3 V, V_{LCD} = 1.667 V**					
all segments off, all FP enabled	I_{segoff}	—	3.1	—	µA
half of segments on	$I_{seghalf}$	—	4.2	—	µA
all segments on	$I_{segallon}$	—	3.6	—	µA
LCD Current Consumption **Trippler Bypassed Mode; V_{DD} = 3 V, V_{LCD} = 1.667 V**					
all segments off, all FP enabled	I_{segoff}	—	0.1	—	µA
half of segments on	$I_{seghalf}$	—	0.1	—	µA
all segments on	$I_{segallon}$	—	0.1	—	µA

Note: Current consumption data based on using the external 32-kHz oscillator with LCD configured using the low-power wave forms option, a 1/4 duty, and a 32-Hz frame frequency. C_{LCD} = C_{BYLCD} = 100 nF; 160 segment 2000 pF LCD panel.

A.9 Internal Clock Generation Module Characteristics



NOTE:
Use fundamental mode crystal or ceramic resonator only.

Table A-11. ICG DC Electrical Specifications (Temperature Range = 0 to 70°C Ambient)

Characteristic	Symbol	Min	Typ(1)	Max	Unit
Load capacitors	C_1 C_2	See Note (2)			
Feedback resistor Low range (32k to 100 kHz) High range (1M – 16 MHz)	R_F		 10 1		 MΩ MΩ
Series resistor Low range (32 kHz to 100 kHz) High range (1 MHz to 16 MHz)	R_S	 0 0	 0 0	 10 0	 kΩ

1 Data in Typical column was characterized at 3.0 V, 25°C or is typical recommended value.

2 See crystal or resonator manufacturer's recommendation.

A.9.1 ICG Frequency Specifications

Table A-12. ICG Frequency Specifications
(V_{DDA} = V_{DDA} (min) to V_{DDA} (max), Temperature Range = 0 to 70°C Ambient)

Characteristic	Symbol	Min	Typical	Max	Unit
Oscillator crystal or resonator (REFS = 1) (Fundamental mode crystal or ceramic resonator) Low range High range, FLL bypassed external (CLKS = 10) High range, FLL engaged external (CLKS = 11)	 f_{lo} f_{hi_byp} f_{hi_eng}	 32 1 2	 — — —	 100 16 10	 kHz MHz MHz
Input clock frequency (CLKS = 11, REFS = 0) Low range High range	 f_{lo} f_{hi_eng}	 32 2	 — —	 100 10	 kHz MHz
Input clock frequency (CLKS = 10, REFS = 0)	f_{Extal}	0	—	40	MHz
Internal reference frequency (untrimmed)	$f_{ICGIRCLK}$	182.25	243	303.75	kHz

Table A-12. ICG Frequency Specifications
(V_{DDA} = V_{DDA} (min) to V_{DDA} (max), Temperature Range = 0 to 70°C Ambient)

Characteristic	Symbol	Min	Typical	Max	Unit
Duty cycle of input clock [4] (REFS = 0)	t_{dc}	40	—	60	%
Output clock ICGOUT frequency CLKS = 10, REFS = 0 All other cases	f_{ICGOUT}	f_{Extal} (min) f_{lo} (min)		f_{Extal} (max) $f_{ICGDCLKmax}$ (max)	MHz
Minimum DCO clock (ICGDCLK) frequency	$f_{ICGDCLKmin}$	8	—		MHz
Maximum DCO clock (ICGDCLK) frequency	$f_{ICGDCLKmax}$		—	40	MHz
Self-clock mode (ICGOUT) frequency [1]	f_{Self}	$f_{ICGDCLKmin}$		$f_{ICGDCLKmax}$	MHz
Self-clock mode reset (ICGOUT) frequency	f_{Self_reset}	5.5	8	10.5	MHz
Loss of reference frequency [2] Low range High range	f_{LOR}	5 50		25 500	kHz
Loss of DCO frequency [3]	f_{LOD}	0.5		1.5	MHz
Crystal start-up time [4], [5] Low range High range	t_{CSTL} t_{CSTH}	— —	430 4	— —	ms
FLL lock time [4, 6] Low range High range	t_{Lockl} t_{Lockh}	— —		5 5	ms
FLL frequency unlock range	n_{Unlock}	–4*N		4*N	counts
FLL frequency lock range	n_{Lock}	–2*N		2*N	counts
ICGOUT period jitter, [4, 7] measured at f_{ICGOUT} Max Long term jitter (averaged over 2 ms interval)	C_{Jitter}	—		0.2	% f_{ICG}
Internal oscillator deviation from trimmed frequency[8] V_{DD} = 1.8 – 3.6 V, (constant temperature) V_{DD} = 3.0 V ±10%, –40° C to 85° C	ACC_{int}	— —	±0.5 ±0.5	±2 ±2	%

1 Self-clocked mode frequency is the frequency that the DCO generates when the FLL is open-loop.

2 Loss of reference frequency is the reference frequency detected internally, which transitions the ICG into self-clocked mode if it is not in the desired range.

3 Loss of DCO frequency is the DCO frequency detected internally, which transitions the ICG into FLL bypassed external mode (if an external reference exists) if it is not in the desired range.

4 This parameter is characterized before qualification rather than 100% tested.

5 Proper PC board layout procedures must be followed to achieve specifications.

6 This specification applies to the period of time required for the FLL to lock after entering FLL engaged internal or external modes. If a crystal/resonator is being used as the reference, this specification assumes it is already running.

7 Jitter is the average deviation from the programmed frequency measured over the specified interval at maximum f_{ICGOUT}. Measurements are made with the device powered by filtered supplies and clocked by a stable external clock signal. Noise injected into the FLL circuitry via V_{DDA} and V_{SSA} and variation in crystal oscillator frequency increase the C_{Jitter} percentage for a given interval.

8 See Figure A-10

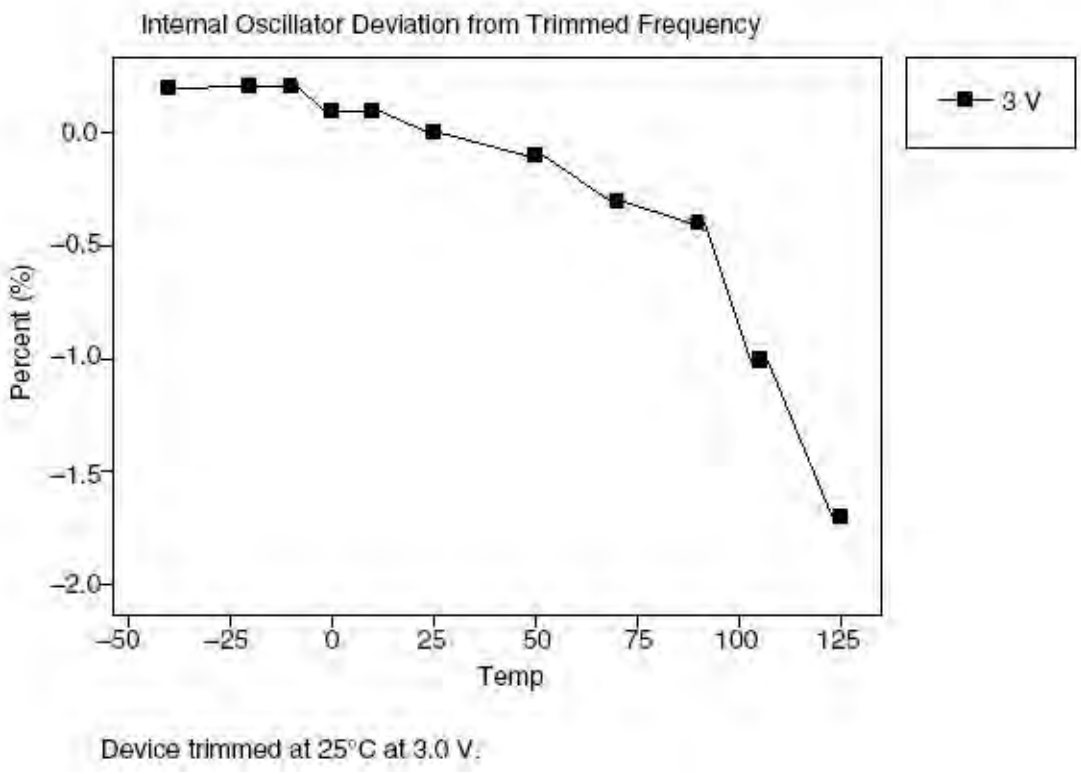


Figure A-11. Internal Oscillator Deviation from Trimmed Frequency

A.10 AC Characteristics

This section describes ac timing characteristics for each peripheral system. For detailed information about how clocks for the bus are generated, see Chapter 7, “Internal Clock Generator (ICG) Module.”

A.10.1 Control Timing

Table A-13. Control Timing

Parameter	Symbol	Min	Typical	Max	Unit
Bus frequency (t_{cyc} = 1/f_{Bus})	f_{Bus}	dc	—	20	MHz
Real-time interrupt internal oscillator period	t_{RTI}	700		1300	µs
External reset pulse width[1]	t_{extrst}	1.5 x f_{Self_reset}		—	ns
Reset low drive[2]	t_{rstdrv}	34 x f_{Self_reset}		—	ns
Active background debug mode latch setup time	t_{MSSU}	25		—	ns
Active background debug mode latch hold time	t_{MSH}	25		—	ns
IRQ pulse width[3]	t_{ILIH}	1.5 x t_{cyc}		—	ns
Port rise and fall time (load = 50 pF)[4] Slew rate control disabled Slew rate control enabled	t_{Rise}, t_{Fall}	 — —	 3 30		ns

1 This is the shortest pulse that is guaranteed to be recognized as a reset pin request. Shorter pulses are not guaranteed to override reset requests from internal sources.

2 When any reset is initiated, internal circuitry drives the reset pin low for about 34 cycles of f_{Self_reset} and then samples the level on the reset pin about 38 cycles later to distinguish external reset requests from internal requests.

3 This is the minimum pulse width that is guaranteed to pass through the pin synchronization circuitry. Shorter pulses may or may not be recognized. In stop mode, the synchronizer is bypassed so shorter pulses can be recognized in that case.

4 Timing is shown with respect to 20% V_{DD} and 80% V_{DD} levels. Temperature range –40°C to 85°C.

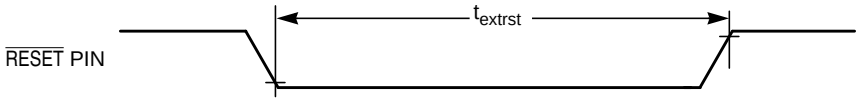


Figure A-12. Reset Timing

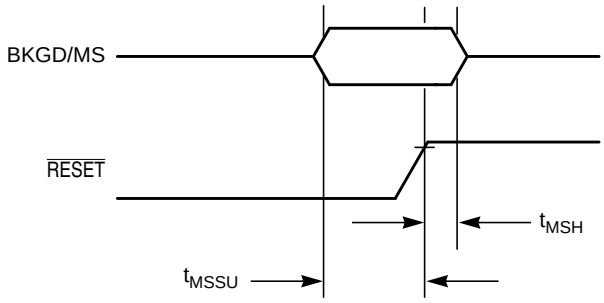


Figure A-13. Active Background Debug Mode Latch Timing

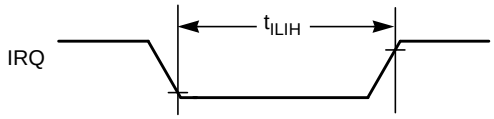


Figure A-14. IRQ Timing

A.10.2 Timer/PWM (TPM) Module Timing

Synchronizer circuits determine the shortest input pulses that can be recognized or the fastest clock that can be used as the optional external source to the timer counter. These synchronizers operate from the current bus rate clock.

Table A-14. TPM Input Timing

Function	Symbol	Min	Max	Unit
External clock frequency	f_{TPMext}	dc	$f_{Bus}/4$	MHz
External clock period	t_{TPMext}	4	—	t_{cyc}
External clock high time	t_{clkh}	1.5	—	t_{cyc}
External clock low time	t_{clkl}	1.5	—	t_{cyc}
Input capture pulse width	t_{ICPW}	1.5	—	t_{cyc}

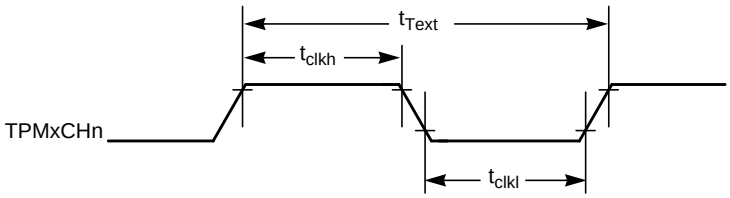


Figure A-15. Timer External Clock

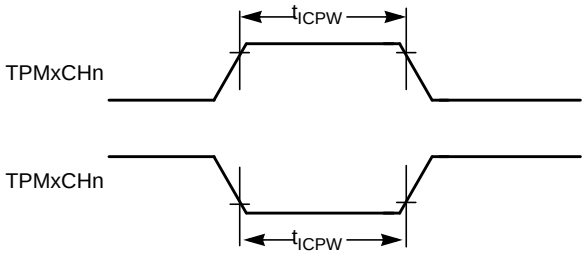


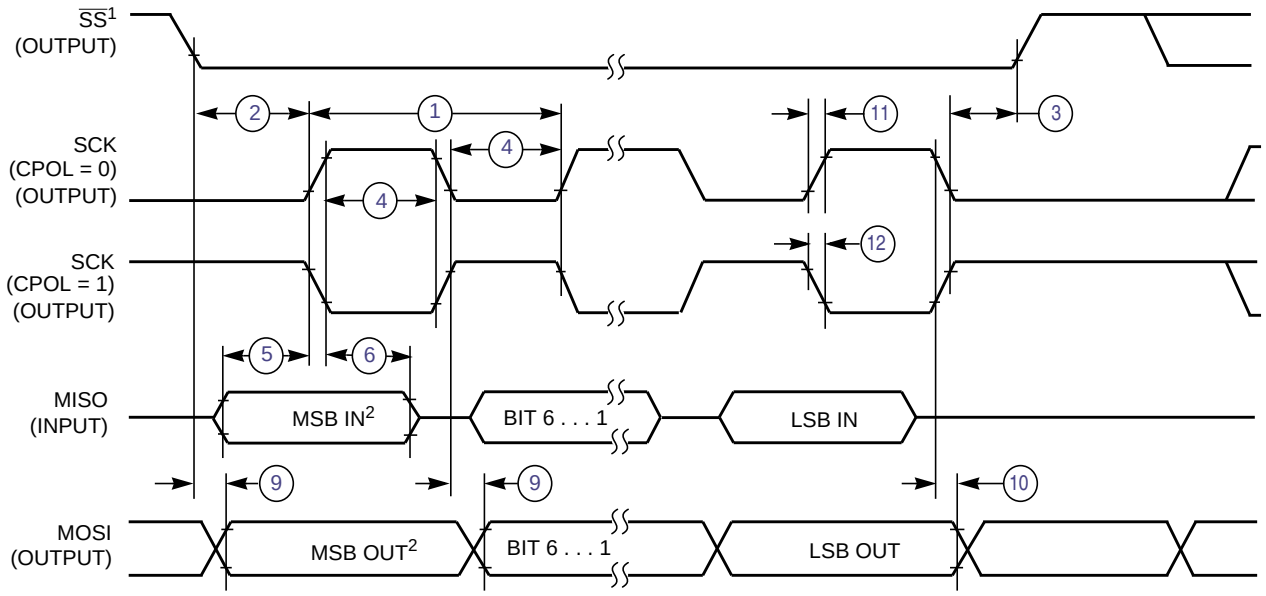
Figure A-16. Timer Input Capture Pulse

A.10.3 SPI Timing

Table A-12 and Figure A-16 through Figure A-19 describe the timing requirements for the SPI system.

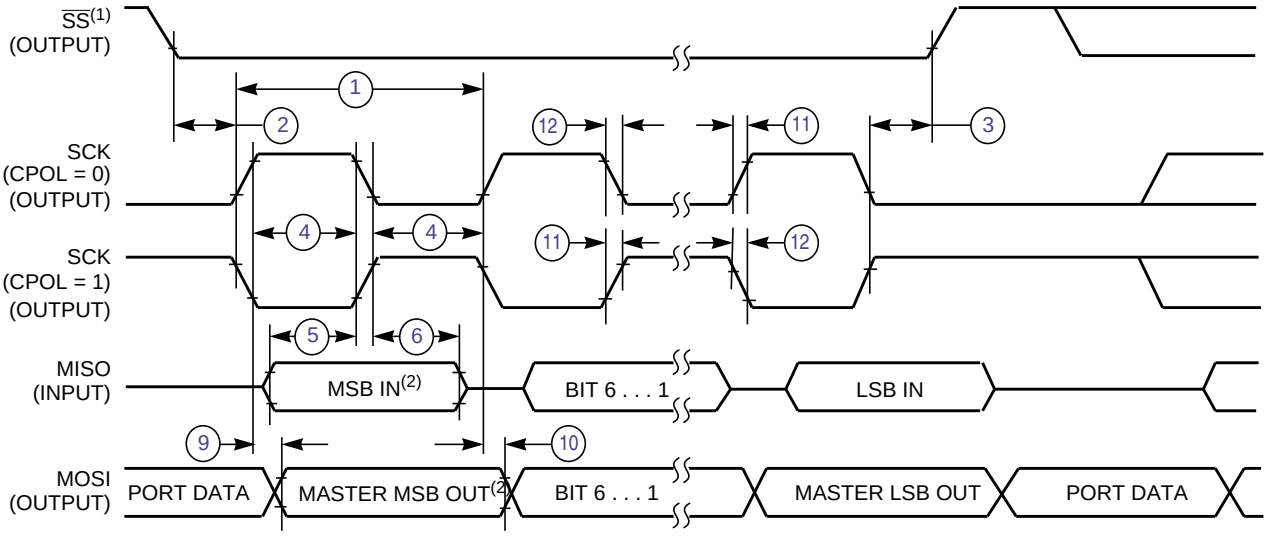
Table A-15. SPI Timing

No.	Function	Symbol	Min	Max	Unit
	Operating frequency Master Slave	f_{op}	$f_{Bus}/2048$ dc	$f_{Bus}/2$ $f_{Bus}/4$	Hz
1	SCK period Master Slave	t_{SCK}	2 4	2048 —	t_{cyc} t_{cyc}
2	Enable lead time Master Slave	t_{Lead}	1/2 1	— —	t_{SCK} t_{cyc}
3	Enable lag time Master Slave	t_{Lag}	1/2 1	— —	t_{SCK} t_{cyc}
4	Clock (SCK) high or low time Master Slave	t_{WSCK}	$t_{cyc} - 30$ $t_{cyc} - 30$	$1024\ t_{cyc}$ —	ns ns
5	Data setup time (inputs) Master Slave	t_{SU}	15 15	— —	ns ns
6	Data hold time (inputs) Master Slave	t_{HI}	0 25	— —	ns ns
7	Slave access time	t_a	—	1	t_{cyc}
8	Slave MISO disable time	t_{dis}	—	1	t_{cyc}
9	Data valid (after SCK edge) Master Slave	t_v	— —	25 25	ns ns
10	Data hold time (outputs) Master Slave	t_{HO}	0 0	— —	ns ns
11	Rise time Input Output	t_{RI} t_{RO}	— —	$t_{cyc} - 25$ 25	ns ns
12	Fall time Input Output	t_{FI} t_{FO}	— —	$t_{cyc} - 25$ 25	ns ns



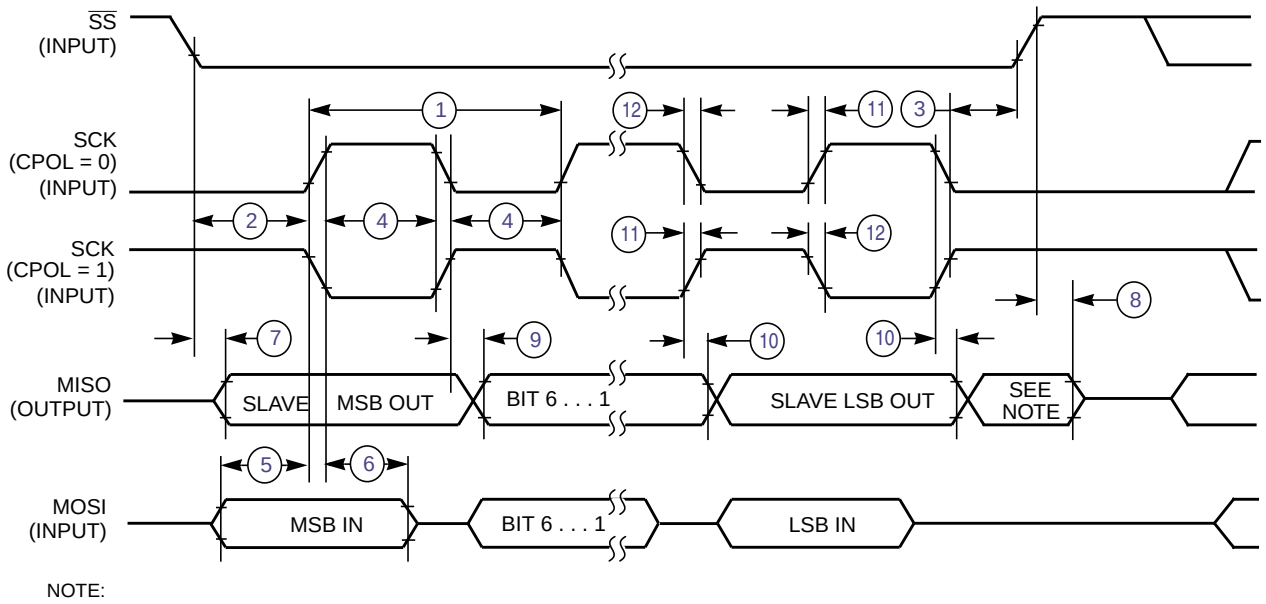
NOTES:
1. $\overline{SS}$ output mode (DDS7 = 1, SSOE = 1).
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure A-17. SPI Master Timing (CPHA = 0)



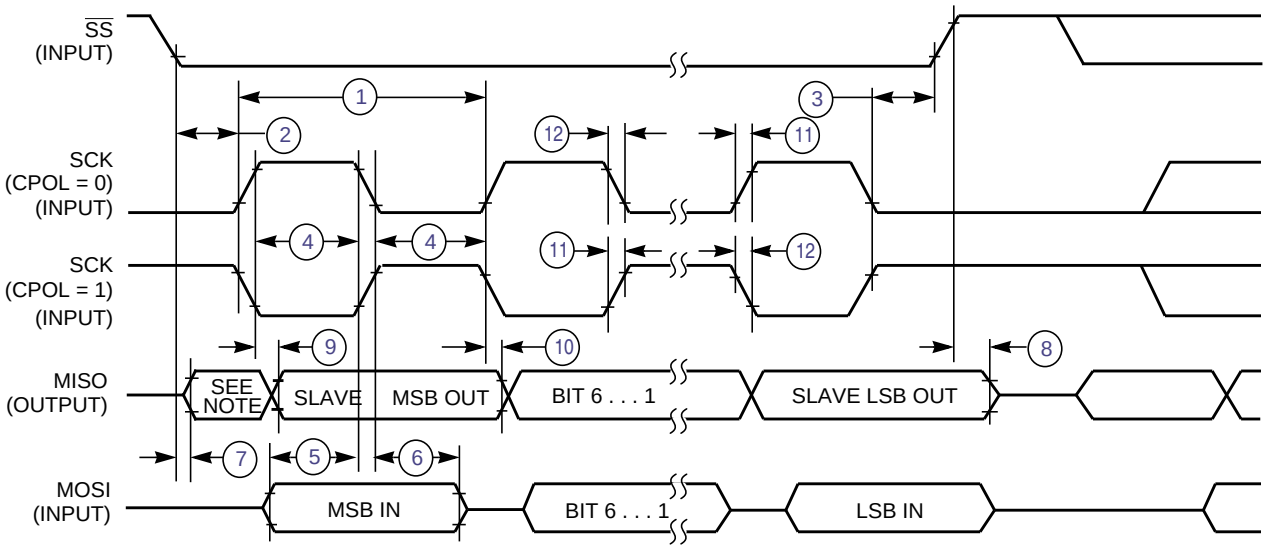
NOTES:
1. $\overline{SS}$ output mode (DDS7 = 1, SSOE = 1).
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure A-18. SPI Master Timing (CPHA = 1)



NOTE:
1. Not defined but normally MSB of character just received

Figure A-19. SPI Slave Timing (CPHA = 0)



NOTE:
1. Not defined but normally LSB of character just received

Figure A-20. SPI Slave Timing (CPHA = 1)

A.11 FLASH Specifications

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

Program and erase operations do not require any special power sources other than the normal V_{DD} supply. For more detailed information about program/erase operations, see Chapter 4, "Memory."

Table A-16. FLASH Characteristics

Characteristic	Symbol	Min	Typical	Max	Unit
Supply voltage for program/erase	$V_{prog/erase}$	1.8		3.6	V
Supply voltage for read operation $0 < f_{Bus} < 8$ MHz $0 < f_{Bus} < 20$ MHz	V_{Read}	 1.8 2.08		 3.6 3.6	V
Internal FCLK frequency[1]	f_{FCLK}	150		200	kHz
Internal FCLK period (1/FCLK)	t_{Fcyc}	5		6.67	µs
Byte program time (random location)[2]	t_{prog}	9			t_{Fcyc}
Byte program time (burst mode)[2]	t_{Burst}	4			t_{Fcyc}
Page erase time[2]	t_{Page}	4000			t_{Fcyc}
Mass erase time[2]	t_{Mass}	20,000			t_{Fcyc}
Program/erase endurance[3] T_L to T_H = –40°C to + 85°C T = 25°C		10,000	 100,000	 —	cycles
Data retention[4]	t_{D_ret}	15	100	—	years

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

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

3 **Typical endurance for FLASH** was evaluated for this product family on the 9S12Dx64. For additional information on how Freescale Semiconductor defines typical endurance, please refer to Engineering Bulletin EB619/D, *Typical Endurance for Nonvolatile Memory*.

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

A.12 EMC Performance

Electromagnetic compatibility (EMC) performance is highly dependant on the environment in which the MCU resides. Board design and layout, circuit topology choices, location and characteristics of external components as well as MCU software operation all play a significant role in EMC performance. The system designer should consult Freescale applications notes such as AN2321, AN1050, AN1263, AN2764, and AN1259 for advice and guidance specifically targeted at optimizing EMC performance.

A.12.1 Radiated Emissions

Microcontroller radiated RF emissions are measured from 150 kHz to 1 GHz using the TEM/GTEM Cell method in accordance with the IEC 61967-2 and SAE J1752/3 standards. The measurement is performed with the microcontroller installed on a custom EMC evaluation board while running specialized EMC test software. The radiated emissions from the microcontroller are measured in all four package orientations (North, South, East and West). For more detailed information concerning the evaluation results, conditions and setup, please refer to the Radiated RF Emissions test report for this device.

The maximum radiated RF emissions of the tested configuration in all orientations are less than or equal to the reported emissions levels.

Table 17-7. Radiated RF Emissions Characteristics[1]

Package	Supply Voltage [V]	Ambient Temp. [°C]	Oscillator Source & Frequency	System Bus Frequency	Test Frequency Range	Emissions Level [Typical][2]	Unit
80 LQFP	3.3	25	Internal	20 MHz	0.150 - 50 MHz	15	dBuV
					50 - 150 MHz	17	
					150 - 500 MHz	3	
					500 - 1000 MHz	3	
					IEC Level[3]	L	-
					SAE Level[4]	2	-
			External crystal, 16 MHz	8 MHz	0.150 - 50 MHz	10	dBuV
					50 - 150 MHz	12	
					150 - 500 MHz	-3	
					500 - 1000 MHz	-3	
					IEC Level[3]	L	-
					SAE Level[4]	2	-

NOTES:
1. This data based on qualification test results. Not tested in production.
2. The reported emission level is the value of the maximum emission, rounded up to the next whole dB, from among the mesured orienations in each frequency range.
3. IEC levels are as specified in IEC 61967-1 and IEC 61967-2.
4. SAE levels are as specified in SAE J1752/1 and SAE J1752/3.

A.12.2 Conducted Transient Susceptibility

Microcontroller transient conducted susceptibility is measured in accordance with an internal Freescale test method. The measurement is performed with the microcontroller installed on a custom EMC evaluation board and running specialized EMC test software designed in compliance with the test method. The conducted susceptibility is determined by injecting the transient signal on each pin of the microcontroller. The transient waveform and injection methodology is in accordance with IEC 61000-4-4 for the electrical fast transient/burst (EFT/B). The transient voltage required to cause performance degradation on any pin in the tested configuration is greater than or equal to the reported levels unless otherwise indicated by footnotes below the table.

Table A-17. Conducted Transient Susceptibility Characteristics[1]

Package	Supply Voltage [V]	Ambient Temp. [^{o}C]	Oscillator Source & Frequency	System Bus Frequency	Result	Amplitude Level [Typical][2]	Unit
80LQFP	2.2	25	External crystal, 32 kHz	20 MHz	A	+/- 4	kV
					B	none	
					C	none	
					D	none	
					E	none	

NOTES:
1. This data based on qualification test results. Not tested in production.
2. The reported transient immunity voltage ;levels indicate the minimum voltage range for each result type.

The susceptibility performance classification is described in Table A-18.

Table A-18. Susceptibility Performance Classification

Result	Performance Criteria	
A	No failure	The MCU performs as designed during and after exposure.
B	Self-recovering failure	The MCU does not perform as designed during exposure. The MCU returns automatically to normal operation after exposure is removed.
C	Soft failure	The MCU does not perform as designed during exposure. The MCU does not return to normal operation until exposure is removed and the RESET pin is asserted.
D	Hard failure	The MCU does not perform as designed during exposure. The MCU does not return to normal operation until exposure is removed and the power to the MCU is cycled.
E	Damage	The MCU does not perform as designed during and after exposure. The MCU cannot be returned to proper operation due to physical damage or other permanent performance degradation.

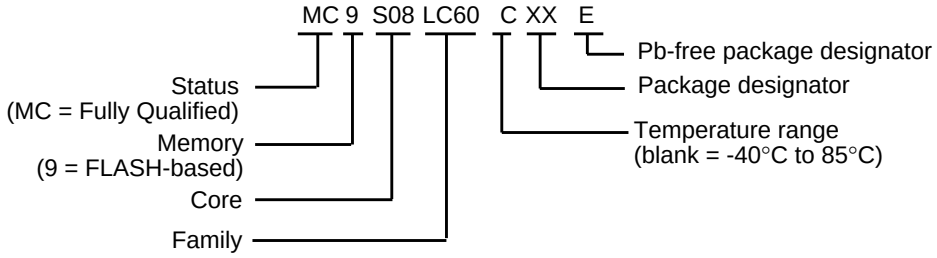
Appendix B
Ordering Information and Mechanical Drawings

B.1 Ordering Information

This section contains ordering numbers for MC9S08LC60 and MC9S08LC36 devices. See below for an example of the device numbering system.

Table B-1. Device Numbering System

Device Number	Memory		Package	
	FLASH	RAM	Type	Designator
MC9S08LC60	60K	4K	80 LQFP	LK
MC9S08LC36	36K	2K	80 LQFP	LK
			64 LQFP	LH



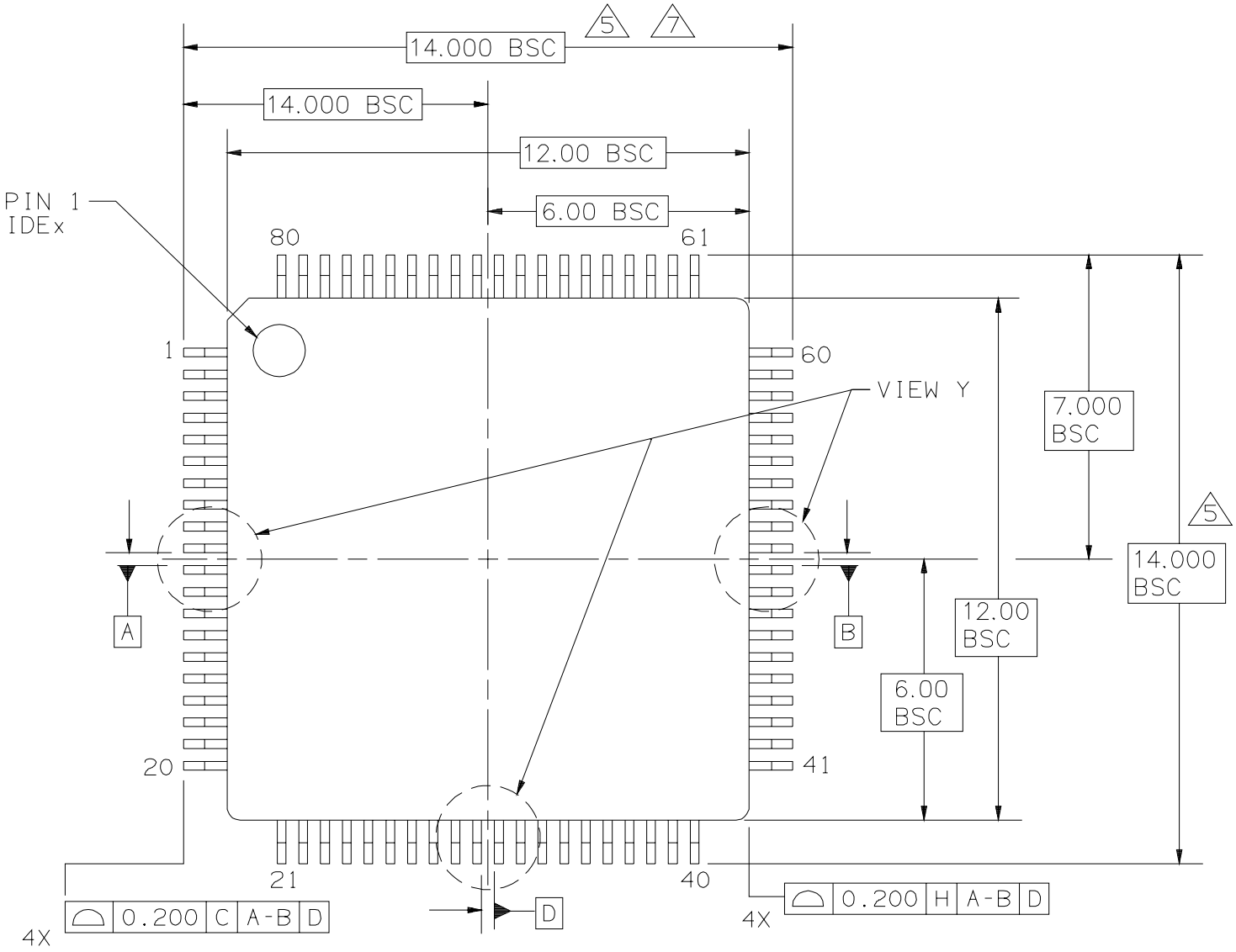
B.2 Mechanical Drawings

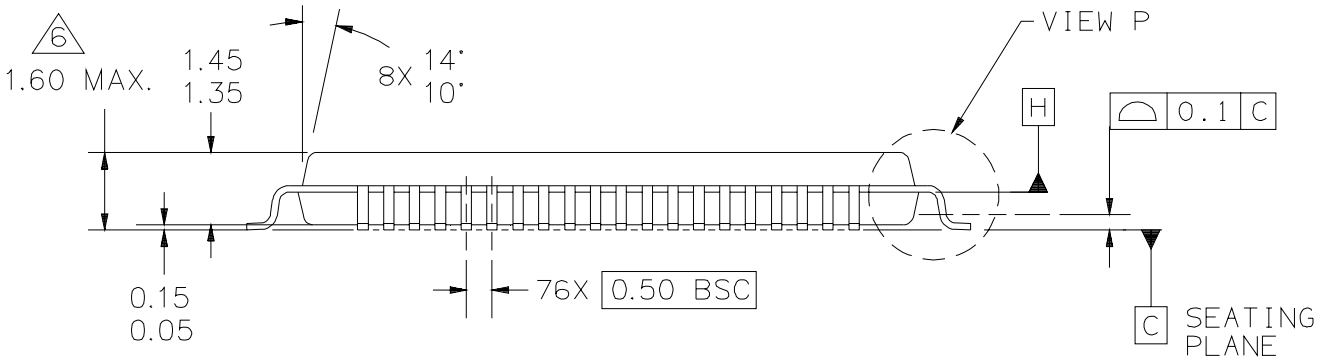
The following pages provide mechanical drawings packages available for MC9S08LC60 Series:

- 80 LQFP
- 64 LQFP

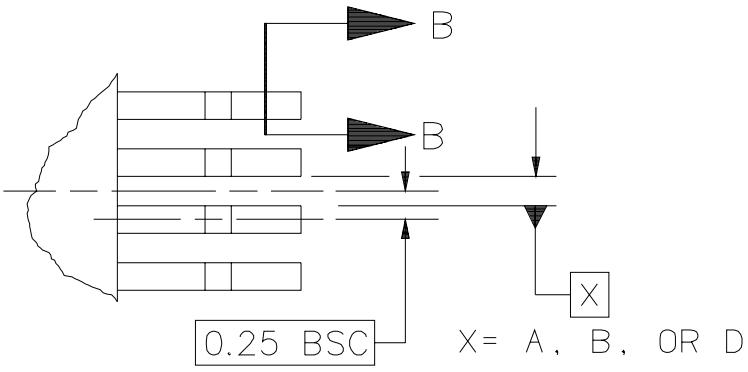
Table B-2. Package Information

Pin Count	Type	Designator	Document No.
80	LQFP	LK	98ASS23174W
64	LQFP	LH	98ASS23234W

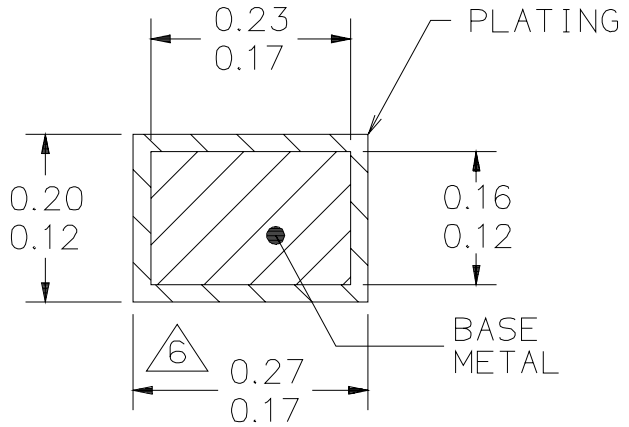




© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE
TITLE: 80 LD FQFP (12 X 12)	DOCUMENT NO: 98ASS23174W	REV: D
	CASE NUMBER: 917-01	25 MAY 2005
	STANDARD: NON-JEDEC	

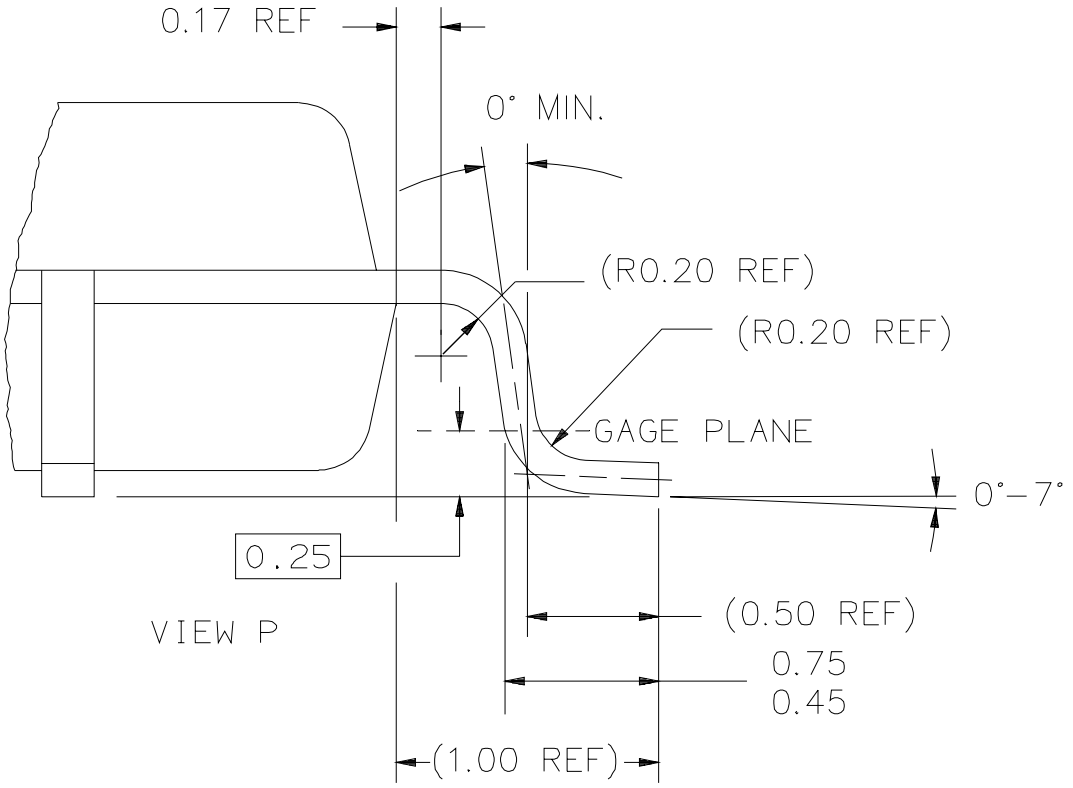


VIEW Y
3 PLACES



⌖	0.08 Ⓜ	C	A-B	D

SECTION B-B
80 PLACES
ROTATED 90° CW



VIEW P

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE
TITLE: 80 LD FQFP (12 X 12)	DOCUMENT NO: 98ASS23174W	REV: D
	CASE NUMBER: 917-01	25 MAY 2005
	STANDARD: NON-JEDEC	

NOTES

1. ALL DIMENSIONS AND TOLERANCES CONFORM TO ASME Y1415M-1994.

2. CONTROLLING DIMENSION: MILLIMETER.

3. DATUM PLANE H IS COINCIDENT WITH THE BOTTOM OF THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.

4. DATUMS A, B AND D TO BE DETERMINED AT DATUM PLANE H.

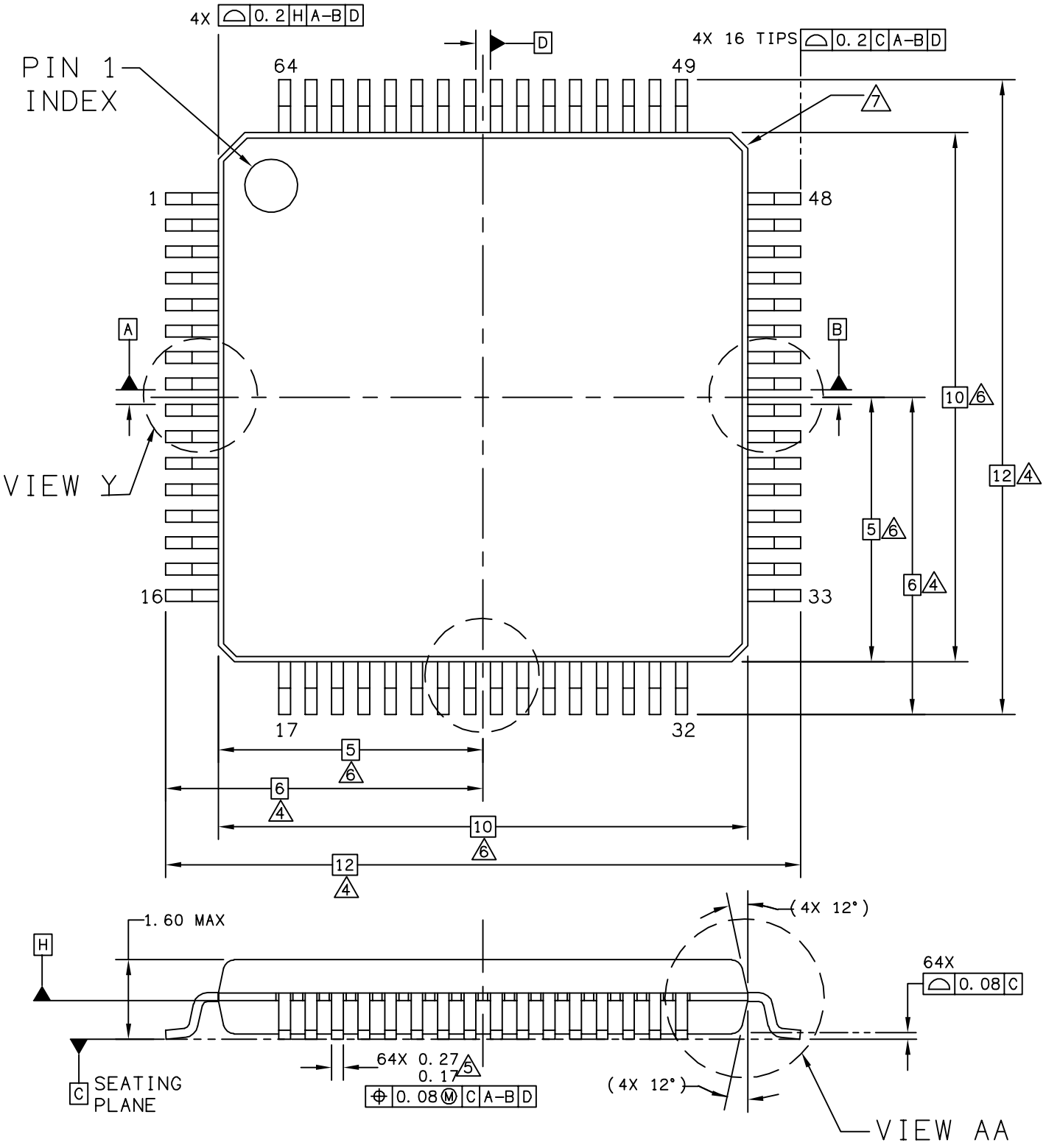
5. DIMENSIONS TO BE DETERMINED AT SEATING PLANE DATUM C.

6. DIMENSIONS DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 PER SIDE. DIMENSIONS DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE H.

7. DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. THE DAMBAR PROTRUSION SHALL NOT CAUSE THE D DIMENSION TO EXCEED 0.35.

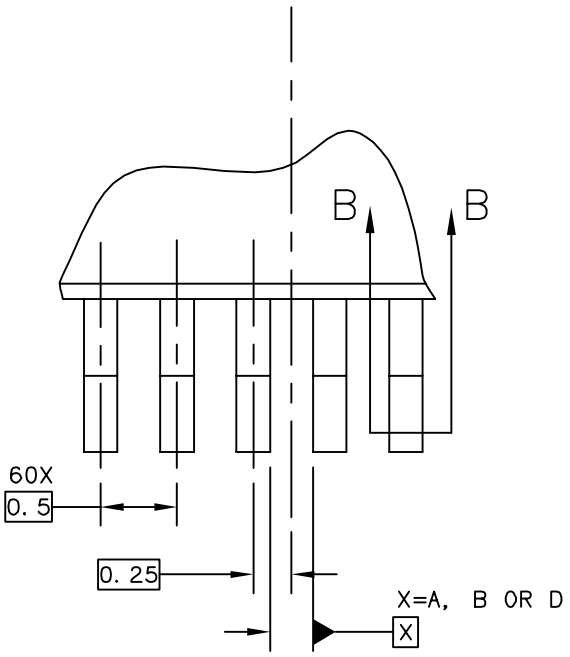
MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE:	DOCUMENT NO: 98ASS23174W	REV: D
80 LD FQFP (12 X 12)	CASE NUMBER: 917-01	25 MAY 2005
	STANDARD: NON-JEDEC	

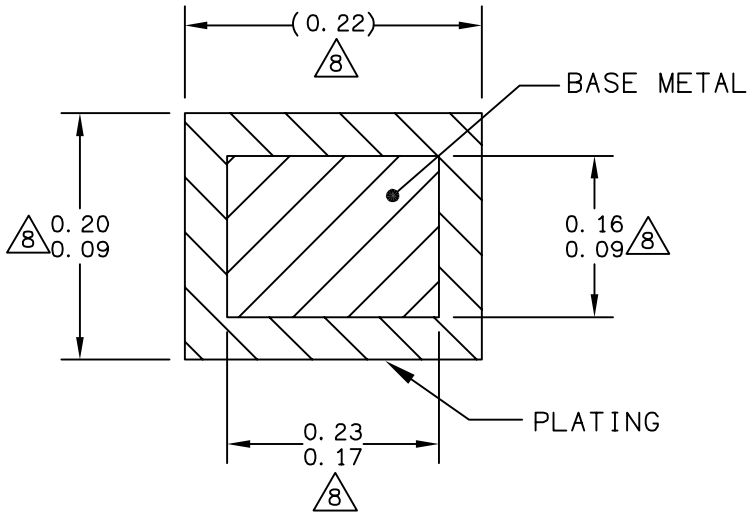


MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

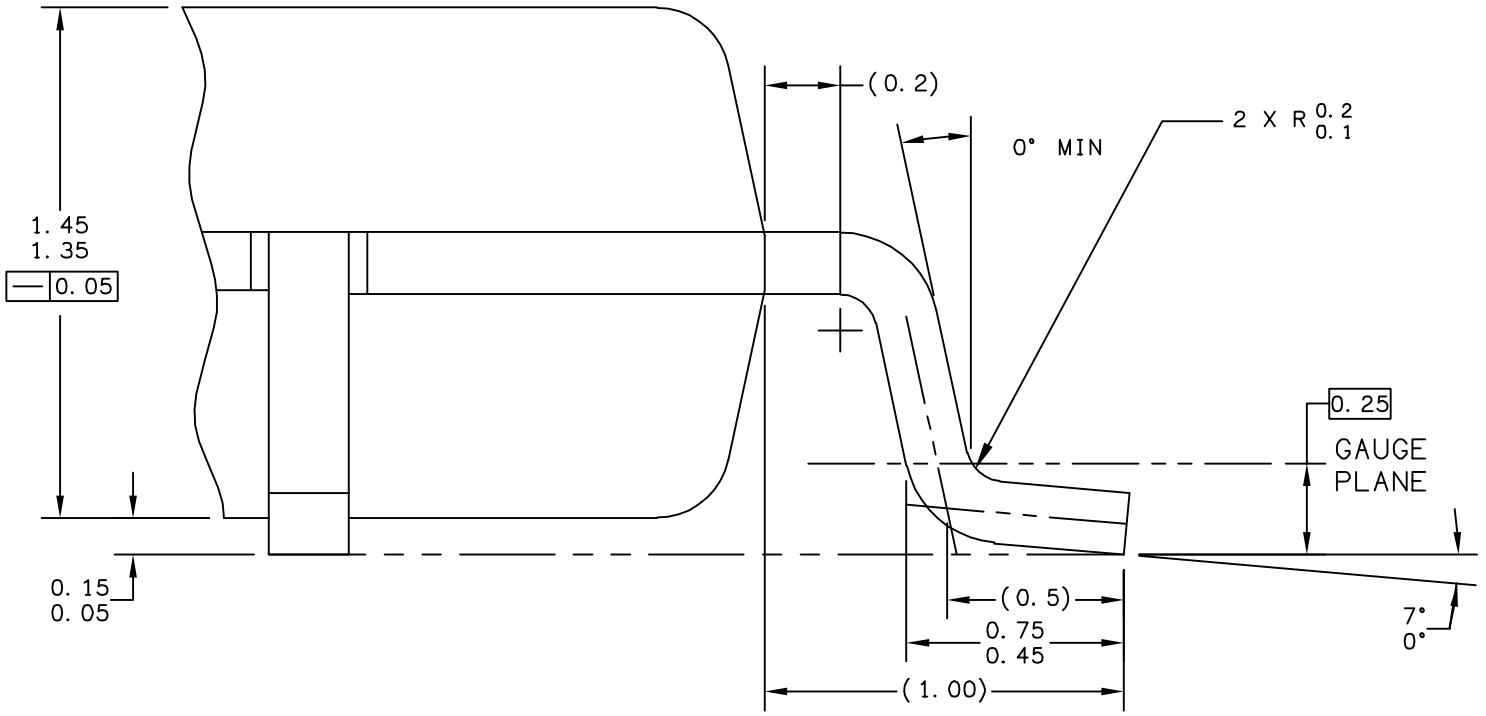
TITLE: 64LD LQFP, 10 X 10 X 1.4 PKG, 0.5 PITCH, CASE OUTLINE	DOCUMENT NO: 98ASS23234W	REV: E
	CASE NUMBER: 840F-02	11 AUG 2006
	STANDARD: JEDEC MS-026 BCD	



VIEW Y



SECTION B–B



VIEW AA

	MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE
TITLE: 64LD LQFP, 10 X 10 X 1.4 PKG, 0.5 PITCH, CASE OUTLINE	DOCUMENT NO: 98ASS23234W	REV: E
	CASE NUMBER: 840F-02	11 AUG 2006
	STANDARD: JEDEC MS-026 BCD	

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M-1994.

3. DATUMS A, B AND D TO BE DETERMINED AT DATUM PLANE H.

4. DIMENSIONS TO BE DETERMINED AT SEATING PLANE C.

5. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE UPPER LIMIT BY MORE THAN 0.08 mm AT MAXIMUM MATERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD SHALL NOT BE LESS THAN 0.07 mm.

6. THIS DIMENSION DOES NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 mm PER SIDE. THIS DIMENSION IS MAXIMUM PLASTIC BODY SIZE DIMENSION INCLUDING MOLD MISMATCH.

7. EXACT SHAPE OF EACH CORNER IS OPTIONAL.

8. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.1 mm AND 0.25 mm FROM THE LEAD TIP.

MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE: 64LD LQFP, 10 X 10 X 1.4 PKG, 0.5 PITCH, CASE OUTLINE	DOCUMENT NO: 98ASS23234W	REV: E
	CASE NUMBER: 840F-02	11 AUG 2006
	STANDARD: JEDEC MS-026 BCD	

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

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

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

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

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

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

MC9S08LC60
Rev. 4, 07/2007

freescale™
semiconductor