THE DATASHEET OF

MC9S08GB32CFUER

www.win-source.net 0086-755-83957316

MC9S08GB60
MC9S08GB32
MC9S08GT60
MC9S08GT32
MC9S08GT16

Data Sheet

HCS08
Microcontrollers

MC9S08GB60/D
Rev. 2.3
12/2004

freescale.com

freescale™
semiconductor

MC9S08GB/GT Data Sheet

Covers: MC9S08GB60
MC9S08GB32
MC9S08GT60
MC9S08GT32
MC9S08GT16

MC9S08GB60
Rev. 2.3
12/2004

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
1.0	4/25/2003	Initial release
1.1		Electricals change, appendix A only
1.2		Electricals change, appendix A only
1.3	10/2/2003	Added module version table; clarifications
1.4	10/29/2003	Fixed typos and made corrections and clarifications
1.5	11/12/2003	Added 1-MHz I_{DD} values to Electricals, appendix A
2	2/10/2004	Changed format of register names to enable reuse of code (from SCIBD to SCI1BD, even when only one instance of a module on a chip) Added new device: MC9S08GT16 to book. Added new 48-pin QFN package to book. BKGDPE description in Section 5 — changed PTD0 to PTG0. Changed typo in CPU section that listed MOV instruction as being 6 cycles instead of 5 (Table 8-2).
2.2	9/2/2004	Format to Freescale look-and-feel; Clarified RTI clock sources and other changes in Chapter 5; updated ICG initialization examples; expanded descriptions of LOLS and LOCS bits in ICGS1; updated ICG electricals Table A-9 and added a figure
2.3	12/01/2004	Minor changes to Table 7-4, Table 7-5, Table A-9; Clarifications in Section 11.10.6, “SCI x Control Register 3 (SCIxC3)”, Section 11.7, “Interrupts and Status Flags”, Section 11.8.1, “8- and 9-Bit Data Modes”, PTG availability in 48-pin package (see Table 2-2)

This product incorporates SuperFlash® technology licensed from SST.

Freescale, and the Freescale logo are trademarks of Freescale Semiconductor, Inc.
© Freescale Semiconductor, Inc., 2004. All rights reserved.

List of Chapters

Chapter 1 Introduction 17
Chapter 2 Pins and Connections 23
Chapter 3 Modes of Operation 33
Chapter 4 Memory 39
Chapter 5 Resets, Interrupts, and System Configuration 61
Chapter 6 Parallel Input/Output 77
Chapter 7 Internal Clock Generator (ICG) Module 97
Chapter 8 Central Processor Unit (CPU) 125
Chapter 9 Keyboard Interrupt (KBI) Module 145
Chapter 10 Timer/PWM (TPM) Module 151
Chapter 11 Serial Communications Interface (SCI) Module 167
Chapter 12 Serial Peripheral Interface (SPI) Module 187
Chapter 13 Inter-Integrated Circuit (IIC) Module 203
Chapter 14 Analog-to-Digital Converter (ATD) Module 219
Chapter 15 Development Support 235
Appendix A Electrical Characteristics 259
Appendix B Ordering Information and Mechanical Drawings 281

Contents

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

Chapter 1
Introduction

1.1 Overview 17
1.2 Features 17
1.2.1 Standard Features of the HCS08 Family 17
1.2.2 Features of MC9S08GB/GT Series of MCUs 17
1.2.3 Devices in the MC9S08GB/GT Series 18
1.3 MCU Block Diagrams 19
1.4 System Clock Distribution 21

Chapter 2
Pins and Connections

2.1 Introduction 23
2.2 Device Pin Assignment 23
2.3 Recommended System Connections 26
2.3.1 Power 28
2.3.2 Oscillator 28
2.3.3 Reset 28
2.3.4 Background / Mode Select (PTG0/BKGD/MS) 29
2.3.5 General-Purpose I/O and Peripheral Ports 29
2.3.6 Signal Properties Summary 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 34
3.6.1 Stop1 Mode 35
3.6.2 Stop2 Mode 35
3.6.3 Stop3 Mode 36
3.6.4 Active BDM Enabled in Stop Mode 36
3.6.5 LVD Enabled in Stop Mode 37
3.6.6 On-Chip Peripheral Modules in Stop Modes 37

Chapter 4
Memory

4.1 MC9S08GB/GT Memory Map 39
4.1.1 Reset and Interrupt Vector Assignments 39
4.2 Register Addresses and Bit Assignments 41
4.3 RAM 46
4.4 FLASH 46
4.4.1 Features 47
4.4.2 Program and Erase Times 47
4.4.3 Program and Erase Command Execution 48
4.4.4 Burst Program Execution 49
4.4.5 Access Errors 50
4.4.6 FLASH Block Protection 51
4.4.7 Vector Redirection 52
4.5 Security 52
4.6 FLASH Registers and Control Bits 53
4.6.1 FLASH Clock Divider Register (FCDIV) 54
4.6.2 FLASH Options Register (FOPT and NVOPT) 55
4.6.3 FLASH Configuration Register (FCNFG) 56
4.6.4 FLASH Protection Register (FPROT and NVPROT) 56
4.6.5 FLASH Status Register (FSTAT) 58
4.6.6 FLASH Command Register (FCMD) 59

Chapter 5
Resets, Interrupts, and System Configuration

5.1 Introduction 61
5.2 Features 61
5.3 MCU Reset 61
5.4 Computer Operating Properly (COP) Watchdog 62
5.5 Interrupts 62
5.5.1 Interrupt Stack Frame 63
5.5.2 External Interrupt Request (IRQ) Pin 64
5.5.2.1 Pin Configuration Options 64
5.5.2.2 Edge and Level Sensitivity 65
5.5.3 Interrupt Vectors, Sources, and Local Masks 65
5.6 Low-Voltage Detect (LVD) System 67
5.6.1 Power-On Reset Operation 67
5.6.2 LVD Reset Operation 67
5.6.3 LVD Interrupt Operation 67
5.6.4 Low-Voltage Warning (LVW) 67
5.7 Real-Time Interrupt (RTI) 67

5.8 Reset, Interrupt, and System Control Registers and Control Bits 68
5.8.1 Interrupt Pin Request Status and Control Register (IRQSC) 68
5.8.2 System Reset Status Register (SRS) 69
5.8.3 System Background Debug Force Reset Register (SBDFR) 71
5.8.4 System Options Register (SOPT) 71
5.8.5 System Device Identification Register (SDIDH, SDIDL) 72
5.8.6 System Real-Time Interrupt Status and Control Register (SRTISC) 73
5.8.7 System Power Management Status and Control 1 Register (SPMSC1) 74
5.8.8 System Power Management Status and Control 2 Register (SPMSC2) 75

Chapter 6
Parallel Input/Output

6.1 Introduction 77
6.2 Features 79
6.3 Pin Descriptions 79
6.3.1 Port A and Keyboard Interrupts 79
6.3.2 Port B and Analog to Digital Converter Inputs 80
6.3.3 Port C and SCI2, IIC, and High-Current Drivers 80
6.3.4 Port D, TPM1 and TPM2 81
6.3.5 Port E, SCI1, and SPI 81
6.3.6 Port F and High-Current Drivers 82
6.3.7 Port G, BKGD/MS, and Oscillator 82
6.4 Parallel I/O Controls 82
6.4.1 Data Direction Control 83
6.4.2 Internal Pullup Control 83
6.4.3 Slew Rate Control 83
6.5 Stop Modes 84
6.6 Parallel I/O Registers and Control Bits 84
6.6.1 Port A Registers (PTAD, PTAPE, PTASE, and PTADD) 84
6.6.2 Port B Registers (PTBD, PTBPE, PTBSE, and PTBDD) 86
6.6.3 Port C Registers (PTCD, PTCPE, PTCSE, and PTCDD) 87
6.6.4 Port D Registers (PTDD, PTDPE, PTDSE, and PTDDD) 89
6.6.5 Port E Registers (PTED, PTEPE, PTESE, and PTEDD) 90
6.6.6 Port F Registers (PTFD, PTFPE, PTFSE, and PTFDD) 92
6.6.7 Port G Registers (PTGD, PTGPE, PTGSE, and PTGDD) 93

Chapter 7
Internal Clock Generator (ICG) Module

7.1 Introduction 99
7.1.1 Features 100
7.1.2 Modes of Operation 101
7.2 External Signal Description 101
7.2.1 Overview 101
7.2.2 Detailed Signal Descriptions 102
7.2.2.1 EXTAL— External Reference Clock / Oscillator Input 102
7.2.2.2 XTAL— Oscillator Output 102
7.2.3 External Clock Connections 102
7.2.4 External Crystal/Resonator Connections 102
7.3 Functional Description 103
7.3.1 Off Mode (Off) 103
7.3.1.1 BDM Active 103
7.3.1.2 OSCSTEN Bit Set 103
7.3.1.3 Stop/Off Mode Recovery 104
7.3.2 Self-Clocked Mode (SCM) 104
7.3.3 FLL Engaged, Internal Clock (FEI) Mode 105
7.3.3.1 FLL Engaged Internal Unlocked 105
7.3.3.2 FLL Engaged Internal Locked 106
7.3.4 FLL Bypassed, External Clock (FBE) Mode 106
7.3.5 FLL Engaged, External Clock (FEE) Mode 106
7.3.5.1 FLL Engaged External Unlocked 106
7.3.5.2 FLL Engaged External Locked 107
7.3.6 FLL Lock and Loss-of-Lock Detection 107
7.3.7 FLL Loss-of-Clock Detection 107
7.3.8 Clock Mode Requirements 108
7.3.9 Fixed Frequency Clock 109
7.4 Initialization/Application Information 110
7.4.1 Introduction 110
7.4.2 Example #1: External Crystal = 32 kHz, Bus Frequency = 4.19 MHz 112
7.4.3 Example #2: External Crystal = 4 MHz, Bus Frequency = 20 MHz 113
7.4.4 Example #3: No External Crystal Connection, 5.4 MHz Bus Frequency 114
7.4.5 Example #4: Internal Clock Generator Trim 116
7.5 ICG Registers and Control Bits 117
7.5.1 ICG Control Register 1 (ICGC1) 118
7.5.2 ICG Control Register 2 (ICGC2) 119
7.5.3 ICG Status Register 1 (ICGS1) 120
7.5.4 ICG Status Register 2 (ICGS2) 122
7.5.5 ICG Filter Registers (ICGFLTU, ICGFLTL) 122
7.5.6 ICG Trim Register (ICGTRM) 123

Chapter 8
Central Processor Unit (CPU)

8.1 Introduction 125
8.2 Features 126
8.3 Programmer's Model and CPU Registers 126
8.3.1 Accumulator (A) 127
8.3.2 Index Register (H:X) 127
8.3.3 Stack Pointer (SP) 128
8.3.4 Program Counter (PC) 128
8.3.5 Condition Code Register (CCR) 128
8.4 Addressing Modes 130
8.4.1 Inherent Addressing Mode (INH) 130
8.4.2 Relative Addressing Mode (REL) 130
8.4.3 Immediate Addressing Mode (IMM) 130
8.4.4 Direct Addressing Mode (DIR) 130
8.4.5 Extended Addressing Mode (EXT) 131
8.4.6 Indexed Addressing Mode 131
8.4.6.1 Indexed, No Offset (IX) 131
8.4.6.2 Indexed, No Offset with Post Increment (IX+) 131
8.4.6.3 Indexed, 8-Bit Offset (IX1) 131
8.4.6.4 Indexed, 8-Bit Offset with Post Increment (IX1+) 131
8.4.6.5 Indexed, 16-Bit Offset (IX2) 131
8.4.6.6 SP-Relative, 8-Bit Offset (SP1) 131
8.4.6.7 SP-Relative, 16-Bit Offset (SP2) 132
8.5 Special Operations 132
8.5.1 Reset Sequence 132
8.5.2 Interrupt Sequence 132
8.5.3 Wait Mode Operation 133
8.5.4 Stop Mode Operation 133
8.5.5 BGND Instruction 134
8.6 HCS08 Instruction Set Summary 134

Chapter 9
Keyboard Interrupt (KBI) Module

9.1 Introduction 145
9.1.1 Port A and Keyboard Interrupt Pins 145
9.2 Features 145
9.3 KBI Block Diagram 147
9.4 Keyboard Interrupt (KBI) Module 147
9.4.1 Pin Enables 147
9.4.2 Edge and Level Sensitivity 147
9.4.3 KBI Interrupt Controls 148
9.5 KBI Registers and Control Bits 148
9.5.1 KBI Status and Control Register (KBI1SC) 148
9.5.2 KBI Pin Enable Register (KBI1PE) 150

Chapter 10
Timer/PWM (TPM) Module

10.1 Introduction 151
10.2 Features 151
10.3 TPM Block Diagram 153
10.4 Pin Descriptions 154
10.4.1 External TPM Clock Sources 154
10.4.2 TPMxCHn — TPMx Channel n I/O Pins 154
10.5 Functional Description 154
10.5.1 Counter 155
10.5.2 Channel Mode Selection 156
10.5.2.1 Input Capture Mode 156
10.5.2.2 Output Compare Mode 156
10.5.2.3 Edge-Aligned PWM Mode 156
10.5.3 Center-Aligned PWM Mode 157
10.6 TPM Interrupts 159
10.6.1 Clearing Timer Interrupt Flags 159
10.6.2 Timer Overflow Interrupt Description 159
10.6.3 Channel Event Interrupt Description 159
10.6.4 PWM End-of-Duty-Cycle Events 160
10.7 TPM Registers and Control Bits 160
10.7.1 Timer x Status and Control Register (TPMxSC) 160
10.7.2 Timer x Counter Registers (TPMxCNTH:TPMxCNTL) 162
10.7.3 Timer x Counter Modulo Registers (TPMxMODH:TPMxMODL) 163
10.7.4 Timer x Channel n Status and Control Register (TPMxCnSC) 163
10.7.5 Timer x Channel Value Registers (TPMxCnVH:TPMxCnVL) 165

Chapter 11
Serial Communications Interface (SCI) Module

11.1 Introduction 167
11.2 Features 169
11.3 SCI System Description 169
11.4 Baud Rate Generation 169
11.5 Transmitter Functional Description 170
11.5.1 Transmitter Block Diagram 170
11.5.2 Send Break and Queued Idle 172
11.6 Receiver Functional Description 172
11.6.1 Receiver Block Diagram 172
11.6.2 Data Sampling Technique 174
11.6.3 Receiver Wakeup Operation 174
11.6.3.1 Idle-Line Wakeup 175
11.6.3.2 Address-Mark Wakeup 175
11.7 Interrupts and Status Flags 175

11.8 Additional SCI Functions 176
11.8.1 8- and 9-Bit Data Modes 176
11.9 Stop Mode Operation 176
11.9.1 Loop Mode 177
11.9.2 Single-Wire Operation 177
11.10 SCI Registers and Control Bits 177
11.10.1 SCI x Baud Rate Registers (SCIxBDH, SCIxBDL) 177
11.10.2 SCI x Control Register 1 (SCIxC1) 178
11.10.3 SCI x Control Register 2 (SCIxC2) 180
11.10.4 SCI x Status Register 1 (SCIxS1) 181
11.10.5 SCI x Status Register 2 (SCIxS2) 183
11.10.6 SCI x Control Register 3 (SCIxC3) 184
11.10.7 SCI x Data Register (SCIxD) 185

Chapter 12
Serial Peripheral Interface (SPI) Module

12.1 Features 189
12.2 Block Diagrams 189
12.2.1 SPI System Block Diagram 189
12.2.2 SPI Module Block Diagram 190
12.2.3 SPI Baud Rate Generation 192
12.3 Functional Description 192
12.3.1 SPI Clock Formats 193
12.3.2 SPI Pin Controls 195
12.3.2.1 SPSCK1 — SPI Serial Clock 195
12.3.2.2 MOSI1 — Master Data Out, Slave Data In 195
12.3.2.3 MISO1 — Master Data In, Slave Data Out 195
12.3.2.4 $\overline{SS1}$ — Slave Select 195
12.3.3 SPI Interrupts 196
12.3.4 Mode Fault Detection 196
12.4 SPI Registers and Control Bits 196
12.4.1 SPI Control Register 1 (SPI1C1) 197
12.4.2 SPI Control Register 2 (SPI1C2) 198
12.4.3 SPI Baud Rate Register (SPI1BR) 199
12.4.4 SPI Status Register (SPI1S) 201
12.4.5 SPI Data Register (SPI1D) 202

Chapter 13
Inter-Integrated Circuit (IIC) Module

13.1 Introduction 205
13.1.1 Features 205
13.1.2 Modes of Operation 205
13.1.3 Block Diagram 206
13.1.4 Detailed Signal Descriptions 206
13.1.4.1 SCL1 — Serial Clock Line 206
13.1.4.2 SDA1 — Serial Data Line 206
13.2 Functional Description 207
13.2.1 IIC Protocol 207
13.2.1.1 START Signal 208
13.2.1.2 Slave Address Transmission 208
13.2.1.3 Data Transfer 208
13.2.1.4 STOP Signal 209
13.2.1.5 Repeated START Signal 209
13.2.1.6 Arbitration Procedure 209
13.2.1.7 Clock Synchronization 209
13.2.1.8 Handshaking 210
13.2.1.9 Clock Stretching 210
13.3 Resets 210
13.4 Interrupts 211
13.4.1 Byte Transfer Interrupt 211
13.4.2 Address Detect Interrupt 211
13.4.3 Arbitration Lost Interrupt 211
13.5 IIC Registers and Control Bits 212
13.5.1 IIC Address Register (IIC1A) 212
13.5.2 IIC Frequency Divider Register (IIC1F) 212
13.5.3 IIC Control Register (IIC1C) 215
13.5.4 IIC Status Register (IIC1S) 216
13.5.5 IIC Data I/O Register (IIC1D) 217

Chapter 14
Analog-to-Digital Converter (ATD) Module

14.1 Introduction 221
14.1.1 Features 221
14.1.2 Modes of Operation 221
14.1.2.1 Stop Mode 221
14.1.2.2 Power Down Mode 221
14.1.3 Block Diagram 221
14.2 Signal Description 222
14.2.1 Overview 222
14.2.1.1 Channel Input Pins — AD1P7–AD1P0 223

14.2.1.2 ATD Reference Pins — V_{REFH}, V_{REFL} 223
14.2.1.3 ATD Supply Pins — V_{DDAD}, V_{SSAD} 223
14.3 Functional Description 223
14.3.1 Mode Control 223
14.3.2 Sample and Hold 224
14.3.3 Analog Input Multiplexer 226
14.3.4 ATD Module Accuracy Definitions 226
14.4 Resets 229
14.5 Interrupts 229
14.6 ATD Registers and Control Bits 229
14.6.1 ATD Control (ATDC) 230
14.6.2 ATD Status and Control (ATD1SC) 232
14.6.3 ATD Result Data (ATD1RH, ATD1RL) 234
14.6.4 ATD Pin Enable (ATD1PE) 234

Chapter 15
Development Support

15.1 Introduction 235
15.2 Features 236
15.3 Background Debug Controller (BDC) 237
15.3.1 BKGD Pin Description 237
15.3.2 Communication Details 238
15.3.3 BDC Commands 242
15.3.4 BDC Hardware Breakpoint 244
15.4 On-Chip Debug System (DBG) 245
15.4.1 Comparators A and B 245
15.4.2 Bus Capture Information and FIFO Operation 245
15.4.3 Change-of-Flow Information 246
15.4.4 Tag vs. Force Breakpoints and Triggers 246
15.4.5 Trigger Modes 247
15.4.6 Hardware Breakpoints 249
15.5 Registers and Control Bits 249
15.5.1 BDC Registers and Control Bits 249
15.5.1.1 BDC Status and Control Register (BDCSCR) 250
15.5.1.2 BDC Breakpoint Match Register (BDCBKPT) 251
15.5.2 System Background Debug Force Reset Register (SBDFR) 251
15.5.3 DBG Registers and Control Bits 252
15.5.3.1 Debug Comparator A High Register (DBGCAH) 252
15.5.3.2 Debug Comparator A Low Register (DBGCAL) 252
15.5.3.3 Debug Comparator B High Register (DBGCBH) 252
15.5.3.4 Debug Comparator B Low Register (DBGCBL) 252

15.5.3.5 Debug FIFO High Register (DBGFH) 253
15.5.3.6 Debug FIFO Low Register (DBGFL) 253
15.5.3.7 Debug Control Register (DBGC) 254
15.5.3.8 Debug Trigger Register (DBGT) 255
15.5.3.9 Debug Status Register (DBGS) 256

Appendix A
Electrical Characteristics

A.1 Introduction 259
A.2 Absolute Maximum Ratings 259
A.3 Thermal Characteristics 260
A.4 Electrostatic Discharge (ESD) Protection Characteristics 261
A.5 DC Characteristics 261
A.6 Supply Current Characteristics 265
A.7 ATD Characteristics 269
A.8 Internal Clock Generation Module Characteristics 271
A.8.1 ICG Frequency Specifications 271
A.9 AC Characteristics 273
A.9.1 Control Timing 273
A.9.2 Timer/PWM (TPM) Module Timing 274
A.9.3 SPI Timing 275
A.10 FLASH Specifications 279

Appendix B
Ordering Information and Mechanical Drawings

B.1 Ordering Information 281
B.2 Mechanical Drawings 281
B.3 64-Pin LQFP Package Drawing 282
B.4 48-Pin QFN Package Drawing 283
B.5 44-Pin QFP Package Drawing 284
B.6 42-Pin SDIP Package Drawing 285

Chapter 1 Introduction

1.1 Overview

The MC9S08GB/GT 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 Features

Features have been organized to reflect:

- Standard features of the HCS08 Family
- Features of the MC9S08GB/GT MCU

1.2.1 Standard Features of the HCS08 Family

- 40-MHz HCS08 CPU (central processor unit)
- HC08 instruction set with added BGND instruction
- Background debugging system (see also Chapter 15, “Development Support”)
- Breakpoint capability to allow single breakpoint setting during in-circuit debugging (plus two more breakpoints in on-chip debug module)
- Debug module containing two comparators and nine trigger modes. Eight deep FIFO for storing change-of-flow addresses and event-only data. Debug module supports both tag and force breakpoints.
- Support for up to 32 interrupt/reset sources
- Power-saving modes: wait plus three stops
- System protection features:
  - Optional computer operating properly (COP) reset
  - Low-voltage detection with reset or interrupt
  - Illegal opcode detection with reset
  - Illegal address detection with reset (some devices don’t have illegal addresses)

1.2.2 Features of MC9S08GB/GT Series of MCUs

- On-chip in-circuit programmable FLASH memory with block protection and security options (see Table 1-1 for device specific information)
- On-chip random-access memory (RAM) (see Table 1-1 for device specific information)
- 8-channel, 10-bit analog-to-digital converter (ATD)
- Two serial communications interface modules (SCI)
- Serial peripheral interface module (SPI)

- Multiple clock source options:
  - Internally generated clock with ±0.2% trimming resolution and ±0.5% deviation across voltage.
  - Crystal
  - Resonator, or
  - External clock
- Inter-integrated circuit bus module to operate up to 100 kbps (IIC)
- One 3-channel and one 5-channel 16-bit timer/pulse width modulator (TPM) modules 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 (TPMx).
- 8-pin keyboard interrupt module (KBI)
- 16 high-current pins (limited by package dissipation)
- Software selectable pullups on ports when used as input. Selection is on an individual port bit basis. During output mode, pullups are disengaged.
- Internal pullup on $\overline{\text{RESET}}$ and IRQ pin to reduce customer system cost
- Up to 56 general-purpose input/output (I/O) pins, depending on package selection
- 64-pin low-profile quad flat package (LQFP) — MC9S08GBxx
- 48-pin quad flat package, no lead (QFN) — MC9S08GTxx
- 44-pin quad flat package (QFP) — MC9S08GTxx
- 42-pin shrink dual in-line package (SDIP) — MC9S08GTxx

1.2.3 Devices in the MC9S08GB/GT Series

Table 1-1 lists the devices available in the MC9S08GB/GT series and summarizes the differences among them.

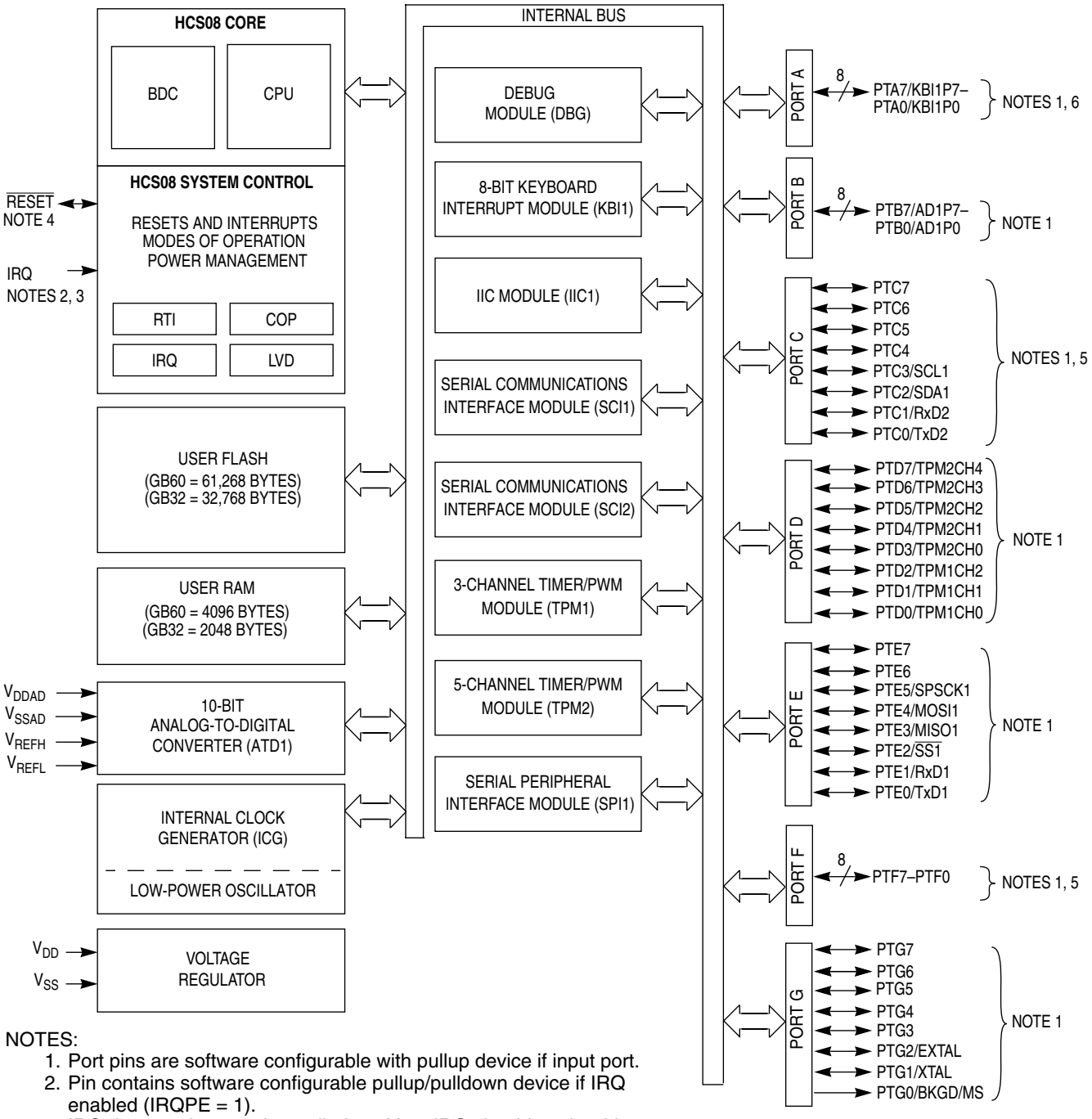
Table 1-1. Devices in the MC9S08GB/GT Series

Device	FLASH	RAM	TPM	I/O	Packages
MC9S08GB60	60K	4K	One 3-channel and one 5-channel, 16-bit timer	56	64 LQFP
MC9S08GB32	32K	2K	One 3-channel and one 5-channel, 16-bit timer	56	64 LQFP
MC9S08GT60	60K	4K	Two 2-channel, 16-bit timers	39 36 34	48 QFN[1] 44 QFP 42 SDIP
MC9S08GT32	32K	2K	Two 2-channel, 16-bit timers	39 36 34	48 QFN[(1)] 44 QFP 42 SDIP
MC9S08GT16	16K	1K	Two 2-channel, 16-bit timers	39 36 34	48 QFN[(1)] 44 QFP 42 SDIP

1 The 48-pin QFN package has one 3-channel and one 2-channel 16-bit TPM.

1.3 MCU Block Diagrams

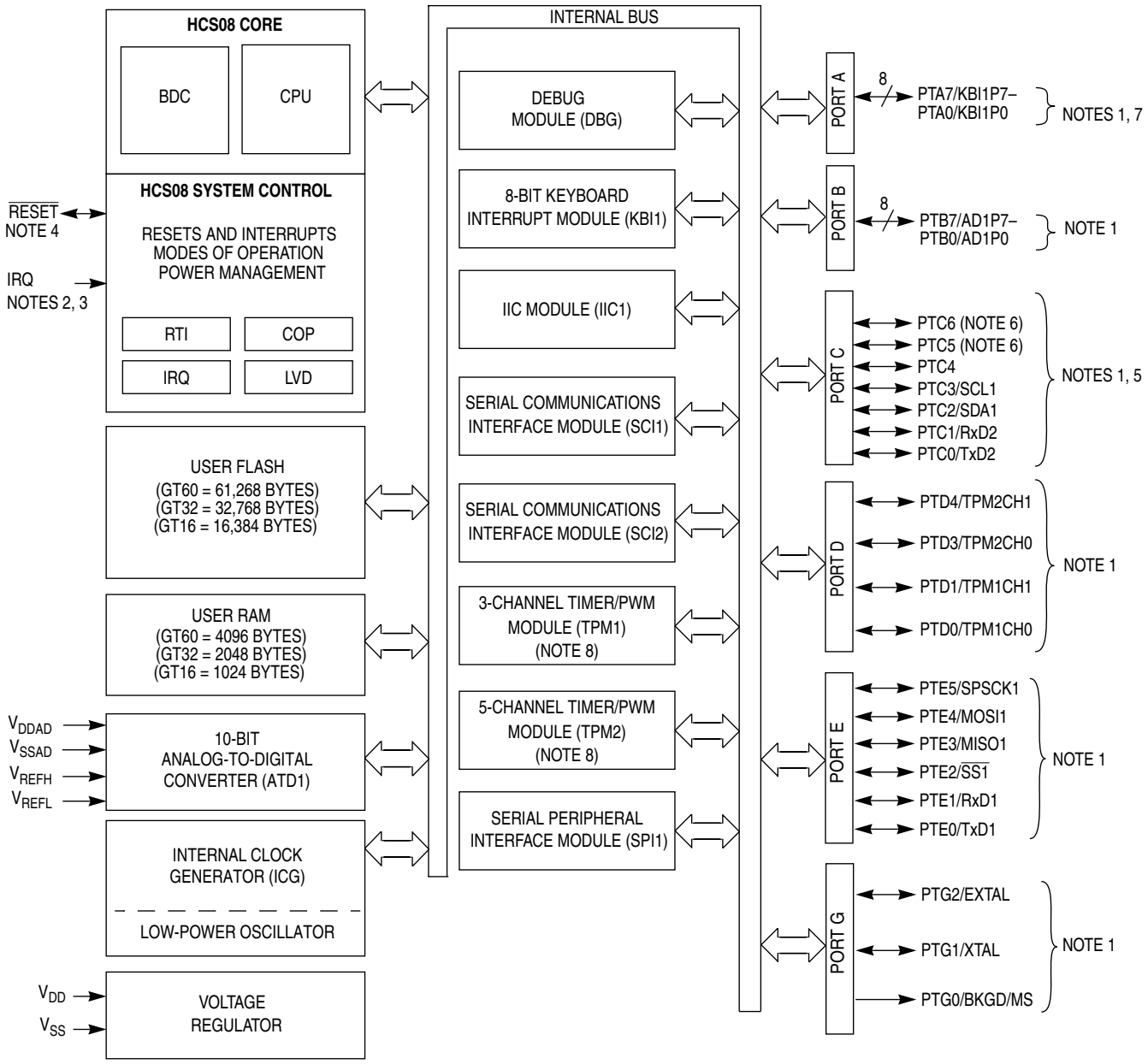
These block diagrams show the structure of the MC9S08GB/GT MCUs.



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. High current drive
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).

Figure 1-1. MC9S08GBxx Block Diagram



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. High current drive
6. PTC[6:5] are not available on the 42-pin SDIP package.
7. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).
8. Only two timer channels per TPM are bonded out. All channels are available for use as software compare.

Figure 1-2. MC9S08GTxx Block Diagram

Table 1-2 lists the functional versions of the on-chip modules.

Table 1-2. Block Versions

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

1.4 System Clock Distribution

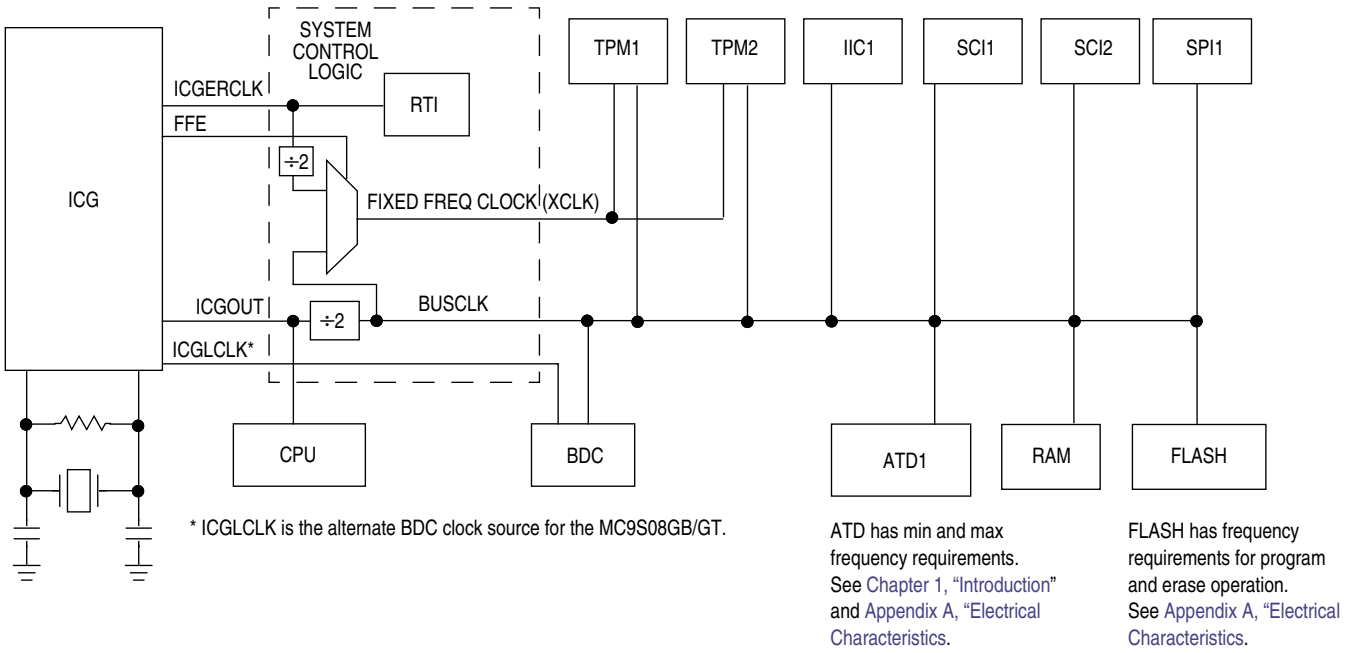


Figure 1-3. System Clock Distribution Diagram

Some of the modules inside the MCU have clock source choices. Figure 1-3 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

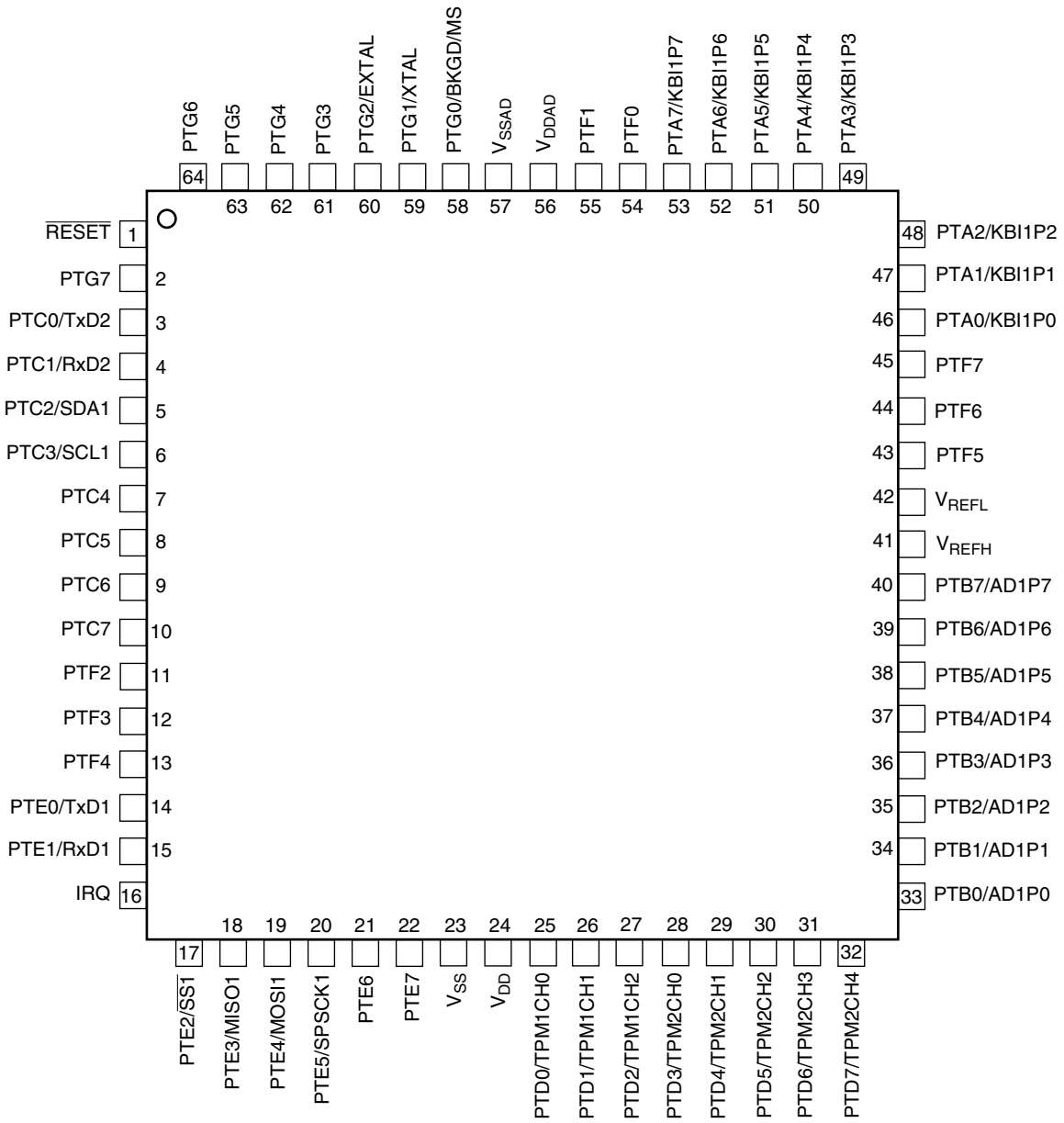


Figure 2-1. MC9S08GBxx in 64-Pin LQFP Package

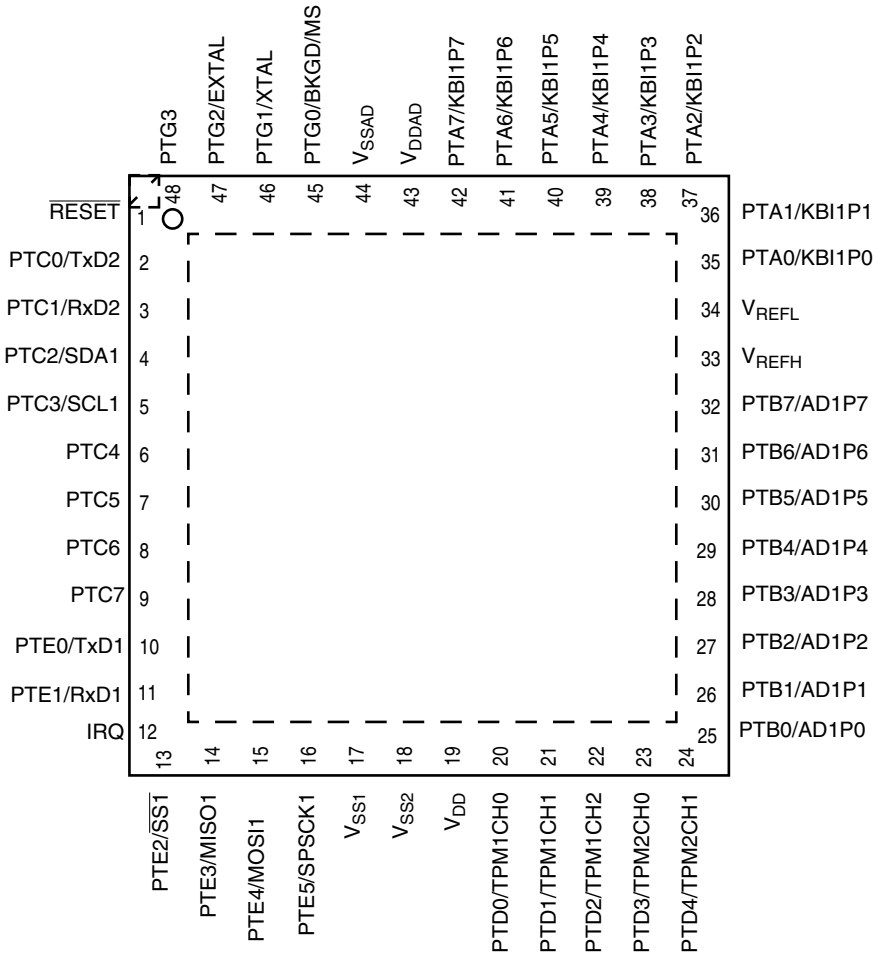


Figure 2-2. MC9S08GTxx in 48-Pin QFN Package

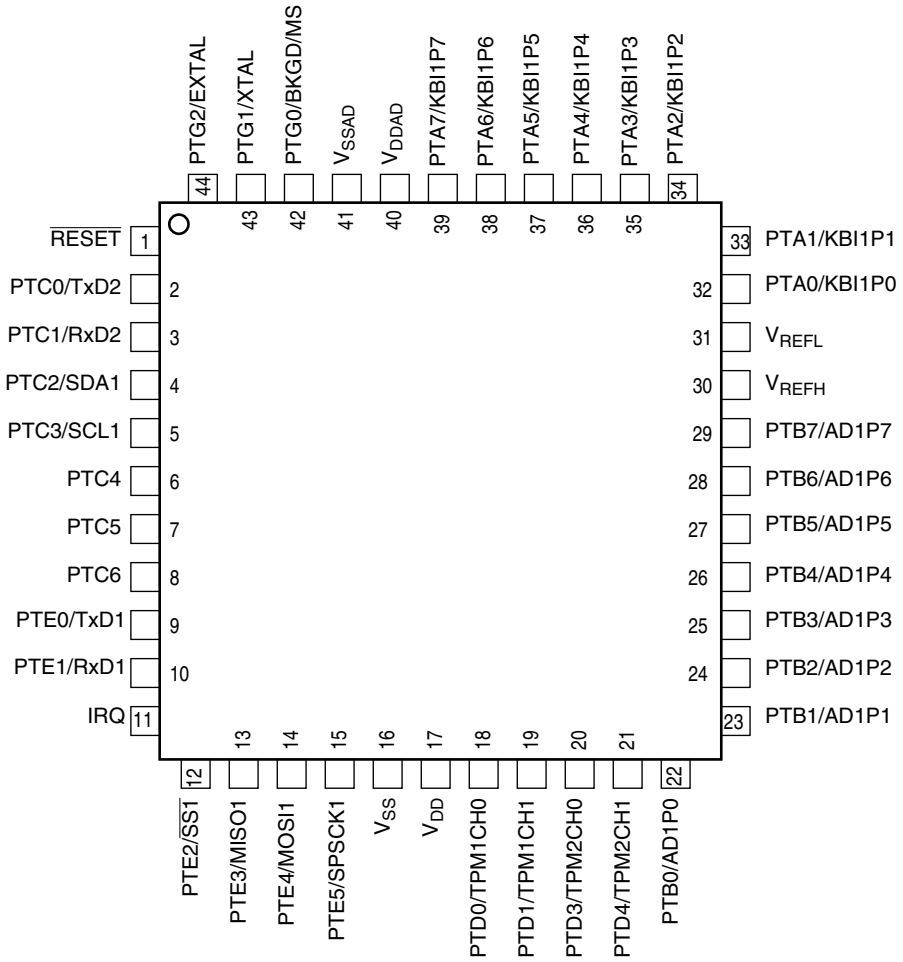


Figure 2-3. MC9S08GTxx in 44-Pin QFP Package

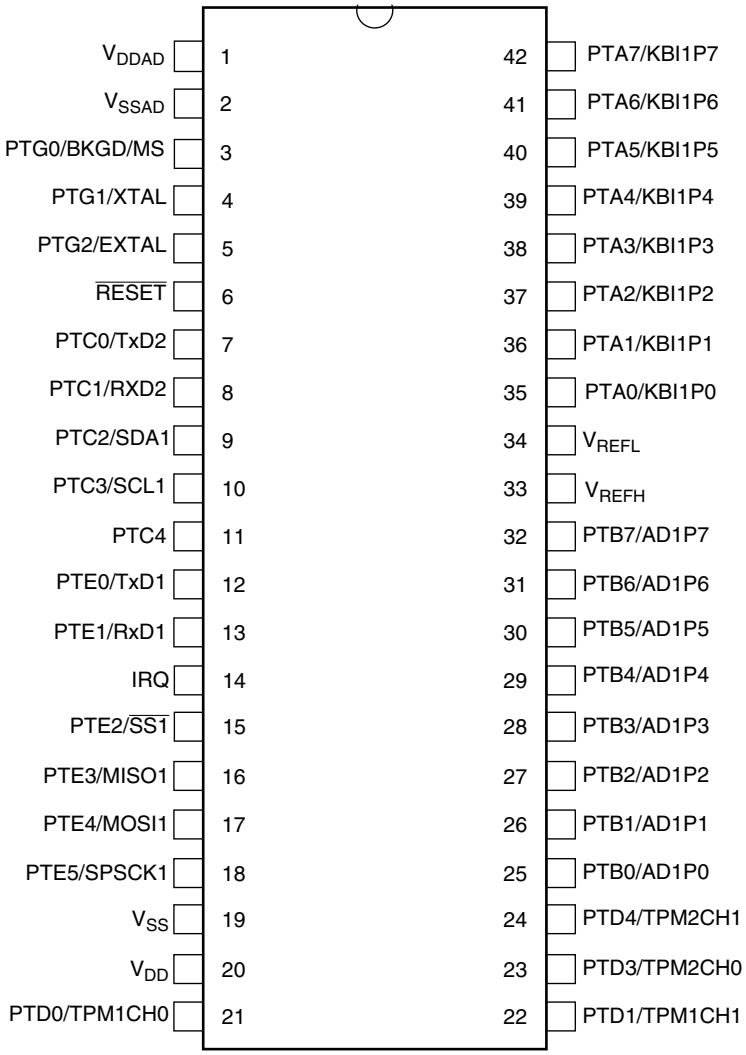
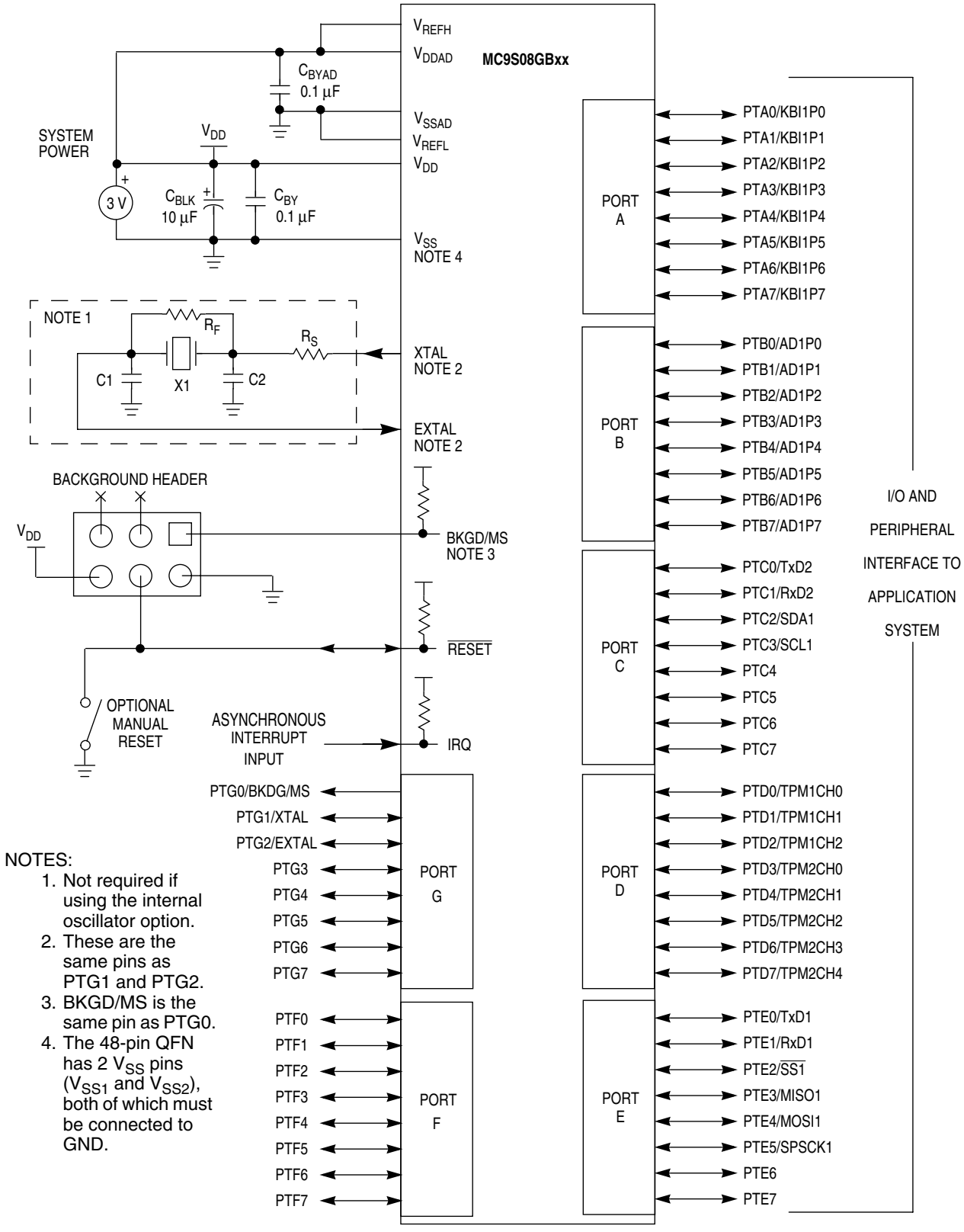


Figure 2-4. MC9S08GTxx in 42-Pin SDIP Package

2.3 Recommended System Connections

Figure 2-5 shows pin connections that are common to almost all MC9S08GB60 application systems. MC9S08GTxx connections will be similar 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. These are the same pins as PTG1 and PTG2.
3. BKGD/MS is the same pin as PTG0.
4. The 48-pin QFN has 2 V_{SS} pins (V_{SS1} and V_{SS2}), both of which must be connected to GND.

Figure 2-5. Basic System Connections

2.3.1 Power

V_{DD} and V_{SS} are the primary power supply pins for the MCU. This voltage source supplies power to all I/O buffer circuitry and to an internal voltage regulator. The internal voltage regulator provides 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 ATD. 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 Oscillator

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 7, “Internal Clock Generator (ICG) Module.”

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 2-5 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).

2.3.3 Reset

$\overline{\text{RESET}}$ is a dedicated pin with a pullup device built in. 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. 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, 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. If the pin is still low at this sample point, the reset is assumed to be from an external source. The reset circuitry decodes the cause of reset and records it by setting a corresponding bit in the system control reset status register (SRS).

Never connect any significant capacitance to the reset pin because that would interfere with the circuit and sequence that detects the source of reset. If an external capacitance prevents the reset pin from rising to a valid logic 1 before the reset sample point, all resets will appear to be external resets.

2.3.4 Background / Mode Select (PTG0/BKGD/MS)

The background/mode select (BKGD/MS) shares its function with an I/O port pin. 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, 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 (PTG0) 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.

2.3.5 General-Purpose I/O and Peripheral Ports

The remaining 55 pins are shared among general-purpose I/O and on-chip peripheral functions such as timers and serial I/O systems. (Sixteen of these pins are not bonded out on the 48-pin package, twenty of these pins are not bonded out on the 44-pin package, and twenty-two are not bonded out on the 42-pin package.) Immediately after reset, all 55 of these pins are configured as high-impedance general-purpose inputs with internal pullup devices disabled.

NOTE

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

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

Table 2-1. Pin Sharing References

Port Pins	Alternate Function	Reference[1]
PTA7–PTA0	KBI1P7–KBI1P0	Chapter 2, "Pins and Connections"
PTB7–PTB0	AD1P7–AD1P0	Chapter 14, "Analog-to-Digital Converter (ATD) Module"
PTC7–PTC4	—	Chapter 6, "Parallel Input/Output"
PTC3–PTC2	SCL1–SDA1	Chapter 13, "Inter-Integrated Circuit (IIC) Module"
PTC1–PTC0	RxD2–TxD2	Chapter 11, "Serial Communications Interface (SCI) Module"
PTD7–PTD3	TPM2CH4– TPM2CH0	Chapter 10, "Timer/PWM (TPM) Module"
PTD2–PTD0	TPM1CH2– TPM1CH0	Chapter 10, "Timer/PWM (TPM) Module"
PTE7–PTE6	—	Chapter 6, "Parallel Input/Output"
PTE5 PTE4 PTE3 PTE2	SPSCK1 MISO1 MOSI1 $\overline{SS1}$	Chapter 12, "Serial Peripheral Interface (SPI) Module"
PTE1–PTE0	RxD1–TxD1	Chapter 11, "Serial Communications Interface (SCI) Module"
PTF7–PTF0	—	Chapter 6, "Parallel Input/Output"
PTG7–PTG3	—	Chapter 6, "Parallel Input/Output"
PTG2–PTG1	EXTAL–XTAL	Chapter 7, "Internal Clock Generator (ICG) Module"
PTG0	BKGD/MS	Chapter 15, "Development Support"

1 See this section for information about modules that share these pins.

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

Pullup enable bits for each input pin control whether on-chip pullup devices are enabled whenever the pin is acting as an input even if it is being controlled by an on-chip peripheral module. When the PTA7–PTA4 pins are controlled by the KBI module and are configured for rising-edge/high-level sensitivity, the pullup enable control bits enable pulldown devices rather than pullup devices. Similarly, when IRQ is configured

as the IRQ input and is set to detect rising edges, the pullup enable control bit enables a pulldown device rather than a pullup device.

2.3.6 Signal Properties Summary

Table 2-2 summarizes I/O pin characteristics. These characteristics are determined by the way the common pin interfaces are hardwired to internal circuits.

Table 2-2. Signal Properties

Pin Name	Dir	High Current Pin	Output Slew [1]	Pull-Up[2]	Comments
V_{DD}		—	—	—	
V_{SS}		—	—	—	The 48-pin QFN package has two V_{SS} pins — V_{SS1} and V_{SS2}.
V_{DDAD}		—	—	—	
V_{SSAD}		—	—	—	
V_{REFH}		—	—	—	
V_{REFL}		—	—	—	
$\overline{RESET}$	I/O	Y	N	Y	Pin contains integrated pullup.
IRQ	I	—	—	Y	IRQPE must be set to enable IRQ function. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}. Pullup/pulldown active when IRQ pin function enabled. Pullup forced on when IRQ enabled for falling edges; pulldown forced on when IRQ enabled for rising edges.
PTA0/KBI1P0	I/O	N	SWC	SWC	
PTA1/KBI1P1	I/O	N	SWC	SWC	
PTA2/KBI1P2	I/O	N	SWC	SWC	
PTA3/KBI1P3	I/O	N	SWC	SWC	
PTA4/KBI1P4	I/O	N	SWC	SWC	Pullup/pulldown active when KBI pin function enabled. Pullup forced on when KBI1Px enabled for falling edges; pulldown forced on when KBI1Px enabled for rising edges.
PTA5/KBI1P5	I/O	N	SWC	SWC	
PTA6/KBI1P6	I/O	N	SWC	SWC	
PTA7/KBI1P7	I/O	N	SWC	SWC	
PTB0/AD1P0	I/O	N	SWC	SWC	
PTB1/AD1P1	I/O	N	SWC	SWC	
PTB2/AD1P2	I/O	N	SWC	SWC	
PTB3/AD1P3	I/O	N	SWC	SWC	
PTB4/AD1P4	I/O	N	SWC	SWC	
PTB5/AD1P5	I/O	N	SWC	SWC	
PTB6/AD1P6	I/O	N	SWC	SWC	
PTB7/AD1P7	I/O	N	SWC	SWC	
PTC0/TxD2	I/O	Y	SWC	SWC	When pin is configured for SCI function, pin is configured for partial output drive.
PTC1/RxD2	I/O	Y	SWC	SWC	
PTC2/SDA1	I/O	Y	SWC	SWC	
PTC3/SCL1	I/O	Y	SWC	SWC	

Table 2-2. Signal Properties (continued)

Pin Name	Dir	High Current Pin	Output Slew [1]	Pull-Up[2]	Comments
PTC4	I/O	Y	SWC	SWC	
PTC5	I/O	Y	SWC	SWC	Not available on 42-pin pkg
PTC6	I/O	Y	SWC	SWC	Not available on 42-pin pkg
PTC7	I/O	Y	SWC	SWC	Not available on 42- or 44-pin pkg
PTD0/TPM1CH0	I/O	N	SWC	SWC	
PTD1/TPM1CH1	I/O	N	SWC	SWC	
PTD2/TPM1CH2	I/O	N	SWC	SWC	Not available on 42- or 44-pin pkg
PTD3/TPM2CH0	I/O	N	SWC	SWC	
PTD4/TPM2CH1	I/O	N	SWC	SWC	
PTD5/TPM2CH2	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTD6/TPM2CH3	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTD7/TPM2CH4	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTE0/TxD1	I/O	N	SWC	SWC	
PTE1/RxD1	I/O	N	SWC	SWC	
PTE2/$\overline{SS1}$	I/O	N	SWC	SWC	
PTE3/MISO1	I/O	N	SWC	SWC	
PTE4/MOSI1	I/O	N	SWC	SWC	
PTE5/SPSCK1	I/O	N	SWC	SWC	
PTE6	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTE7	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF0	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF1	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF2	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF3	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF4	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF5	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF6	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTF7	I/O	Y	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTG0/BKGD/MS	O	N	SWC	SWC	Pullup enabled and slew rate disabled when BDM function enabled.
PTG1/XTAL	I/O	N	SWC	SWC	Pullup and slew rate disabled when XTAL pin function.
PTG2/EXTAL	I/O	N	SWC	SWC	Pullup and slew rate disabled when EXTAL pin function.
PTG3	I/O	N	SWC	SWC	Not available on 42-, or 44-pin pkg
PTG4	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTG5	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTG6	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg
PTG7	I/O	N	SWC	SWC	Not available on 42-, 44-, or 48-pin pkg

1 SWC is software controlled slew rate, the register is associated with the respective port.

2 SWC is software controlled pullup resistor, the register is associated with the respective port.

Chapter 3 Modes of Operation

3.1 Introduction

The operating modes of the MC9S08GB/GT 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 shuts down to conserve power
  - System clocks running
  - Full voltage regulation maintained
- Stop modes:
  - System clocks stopped; voltage regulator in standby
  - 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

3.3 Run Mode

This is the normal operating mode for the MC9S08GB/GT. This mode is selected when the BKGD/MS pin is high at the rising edge of reset. In this mode, the CPU executes code from internal memory with execution beginning at the address fetched from memory at $FFFE:$FFFF 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), provide 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 rising edge of 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 while 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 be executed only 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 the MC9S08GB/GT is shipped from the Freescale Semiconductor 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 Chapter 15, "Development Support."

3.5 Wait Mode

Wait mode is entered by executing a WAIT instruction. Upon execution of the WAIT instruction, the CPU enters a low-power state in which it is not clocked. The I bit in CCR is cleared when the CPU enters the wait mode, enabling interrupts. When an interrupt request occurs, the CPU exits the wait mode and resumes processing, beginning with the stacking operations 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 when the MCU is in wait mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from wait mode and enter active background mode.

3.6 Stop Modes

One of three stop modes is entered upon execution of a STOP instruction when the STOPE bit in the system option register is set. In all stop modes, all internal clocks are halted. If the STOPE bit is not set when the CPU executes a STOP instruction, the MCU will not enter any of the stop modes and an illegal opcode reset is forced. The stop modes are selected by setting the appropriate bits in SPMSC2.

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

Table 3-1. Stop Mode Behavior

Mode	PDC	PPDC	CPU, Digital Peripherals, FLASH	RAM	ICG	ATD	Regulator	I/O Pins	RTI
Stop1	1	0	Off	Off	Off	Disabled[1]	Off	Reset	Off
Stop2	1	1	Off	Standby	Off	Disabled	Standby	States held	Optionally on
Stop3	0	Don't care	Standby	Standby	Off[2]	Disabled	Standby	States held	Optionally on

1 Either ATD stop mode or power-down mode depending on the state of ATDPU.

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

3.6.1 Stop1 Mode

The stop1 mode provides the lowest possible standby power consumption by causing the internal circuitry of the MCU to be powered down. Stop1 can be entered only if the LVD circuit is not enabled in stop modes (either LVDE or LVDSE not set).

When the MCU is in stop1 mode, all internal circuits that are powered from the voltage regulator are turned off. The voltage regulator is in a low-power standby state, as is the ATD.

Exit from stop1 is performed by asserting either of the wake-up pins on the MCU: $\overline{\text{RESET}}$ or IRQ. IRQ is always an active low input when the MCU is in stop1, regardless of how it was configured before entering stop1.

Entering stop1 mode automatically asserts LVD. Stop1 cannot be exited until $V_{DD} > V_{LVDH/L}$ rising (V_{DD} must rise above the LVI rearm voltage).

Upon wake-up from stop1 mode, the MCU will start up as from a power-on reset (POR). The CPU will take the reset vector.

3.6.2 Stop2 Mode

The stop2 mode provides very low standby power consumption and maintains the contents of RAM and the current state of all of the I/O pins. Stop2 can be entered only if the LVD circuit is not enabled in stop modes (either LVDE or LVDSE not set).

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

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

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

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

After waking up from stop2, the PPDF bit in SPMSC2 is set. This flag may be 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 state for pins that were configured as general-purpose I/O, 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 register bits will assume their reset states when the I/O pin latches are opened and the I/O pins will switch to their reset states.

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 Stop3 Mode

Upon entering the stop3 mode, all of the clocks in the MCU, including the oscillator itself, are halted. The ICG is turned off, the ATD is disabled, and the voltage regulator is put in standby. The states of all of the internal registers and logic, as well as the RAM content, are maintained. The I/O pin states are not latched at the pin as in stop2. Instead they are maintained by virtue of the states of the internal logic driving the pins being maintained.

Exit from stop3 is performed by asserting $\overline{\text{RESET}}$, an asynchronous interrupt pin, or through the real-time interrupt. The asynchronous interrupt pins are the IRQ or KBI pins.

If stop3 is exited by means of the $\overline{\text{RESET}}$ pin, then the MCU will be reset and operation will resume after taking the reset vector. Exit by means of an asynchronous interrupt or the real-time interrupt will result 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.

3.6.4 Active BDM Enabled in Stop Mode

Entry into the active background mode from run mode is enabled if the ENBDM bit in BDCSCR is set. This register is described in the Chapter 15, “Development Support,” section 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 so background debug communication is still possible. In addition,

the voltage regulator does not enter its low-power standby state but maintains full internal regulation. If the user attempts to enter either stop1 or stop2 with ENBDM set, the MCU will instead enter stop3.

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

Table 3-2. BDM Enabled Stop Mode Behavior

Mode	PDC	PPDC	CPU, Digital Peripherals, FLASH	RAM	ICG	ATD	Regulator	I/O Pins	RTI
Stop3	Don't care	Don't care	Standby	Standby	Active	Disabled[1]	Active	States held	Optionally on

1 Either ATD stop mode or power-down mode depending on the state of ATDPU.

3.6.5 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 by setting the LVDE and the LVDSE bits in SPMSC1 when the CPU executes a STOP instruction, then the voltage regulator remains active during stop mode. If the user attempts to enter either stop1 or stop2 with the LVD enabled for stop (LVDSE = 1), the MCU will instead enter stop3. The table below summarizes the behavior of the MCU in stop when the LVD is enabled.

Table 3-3. LVD Enabled Stop Mode Behavior

Mode	PDC	PPDC	CPU, Digital Peripherals, FLASH	RAM	ICG	ATD	Regulator	I/O Pins	RTI
Stop3	Don't care	Don't care	Standby	Standby	Standby	Disabled[1]	Active	States held	Optionally on

1 Either ATD stop mode or power-down mode depending on the state of ATDPU.

3.6.6 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 3.6.1, "Stop1 Mode," Section 3.6.2, "Stop2 Mode," and Section 3.6.3, "Stop3 Mode," for specific information on system behavior in stop modes.

I/O Pins

- All I/O pin states remain unchanged when the MCU enters stop3 mode.
- If the MCU is configured to go into stop2 mode, all I/O pins states are latched before entering stop.
- If the MCU is configured to go into stop1 mode, all I/O pins are forced to their default reset state upon entry into stop.

Memory

- All RAM and register contents are preserved while the MCU is in stop3 mode.
- All registers will be reset upon wake-up from stop2, but the contents of RAM are preserved and pin states remain latched until the PPDACK bit is written. The user may save any memory-mapped register data into RAM before entering stop2 and restore the data upon exit from stop2.
- All registers will be reset upon wake-up from stop1 and the contents of RAM are not preserved. The MCU must be initialized as upon reset. The contents of the FLASH memory are nonvolatile and are preserved in any of the stop modes.

ICG — In stop3 mode, the ICG enters its low-power standby state. Either the oscillator or the internal reference may be kept running when the ICG is in standby by setting the appropriate control bit. In both stop2 and stop1 modes, the ICG is turned off. Neither the oscillator nor the internal reference can be kept running in stop2 or stop1, even if enabled within the ICG module.

TPM — When the MCU enters stop mode, the clock to the TPM1 and TPM2 modules stop. The modules halt operation. If the MCU is configured to go into stop2 or stop1 mode, the TPM modules will be reset upon wake-up from stop and must be reinitialized.

ATD — When the MCU enters stop mode, the ATD will enter a low-power standby state**.** No conversion operation will occur while in stop. If the MCU is configured to go into stop2 or stop1 mode, the ATD will be reset upon wake-up from stop and must be reinitialized.

KBI — During stop3, the KBI pins that are enabled continue to function as interrupt sources that are capable of waking the MCU from stop3. The KBI is disabled in stop1 and stop2 and must be reinitialized after waking up from either of these modes.

SCI — When the MCU enters stop mode, the clocks to the SCI1 and SCI2 modules stop. The modules halt operation. If the MCU is configured to go into stop2 or stop1 mode, the SCI modules will be reset upon wake-up from stop and must be reinitialized.

SPI — When the MCU enters stop mode, the clocks to the SPI module stop. The module halts operation. If the MCU is configured to go into stop2 or stop1 mode, the SPI module will be reset upon wake-up from stop and must be reinitialized.

IIC — When the MCU enters stop mode, the clocks to the IIC module stops. The module halts operation. If the MCU is configured to go into stop2 or stop1 mode, the IIC module will be reset upon wake-up from stop and must be reinitialized.

Voltage Regulator — The voltage regulator enters a low-power standby state when the MCU enters any of the stop modes unless the LVD is enabled in stop mode or BDM is enabled.

Chapter 4 Memory

4.1 MC9S08GB/GT Memory Map

As shown in Figure 4-1, on-chip memory in the MC9S08GB/GT series of MCUs 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 ($0000 through $007F)
- High-page registers ($1800 through $182B)
- Nonvolatile registers ($FFB0 through $FFBF)

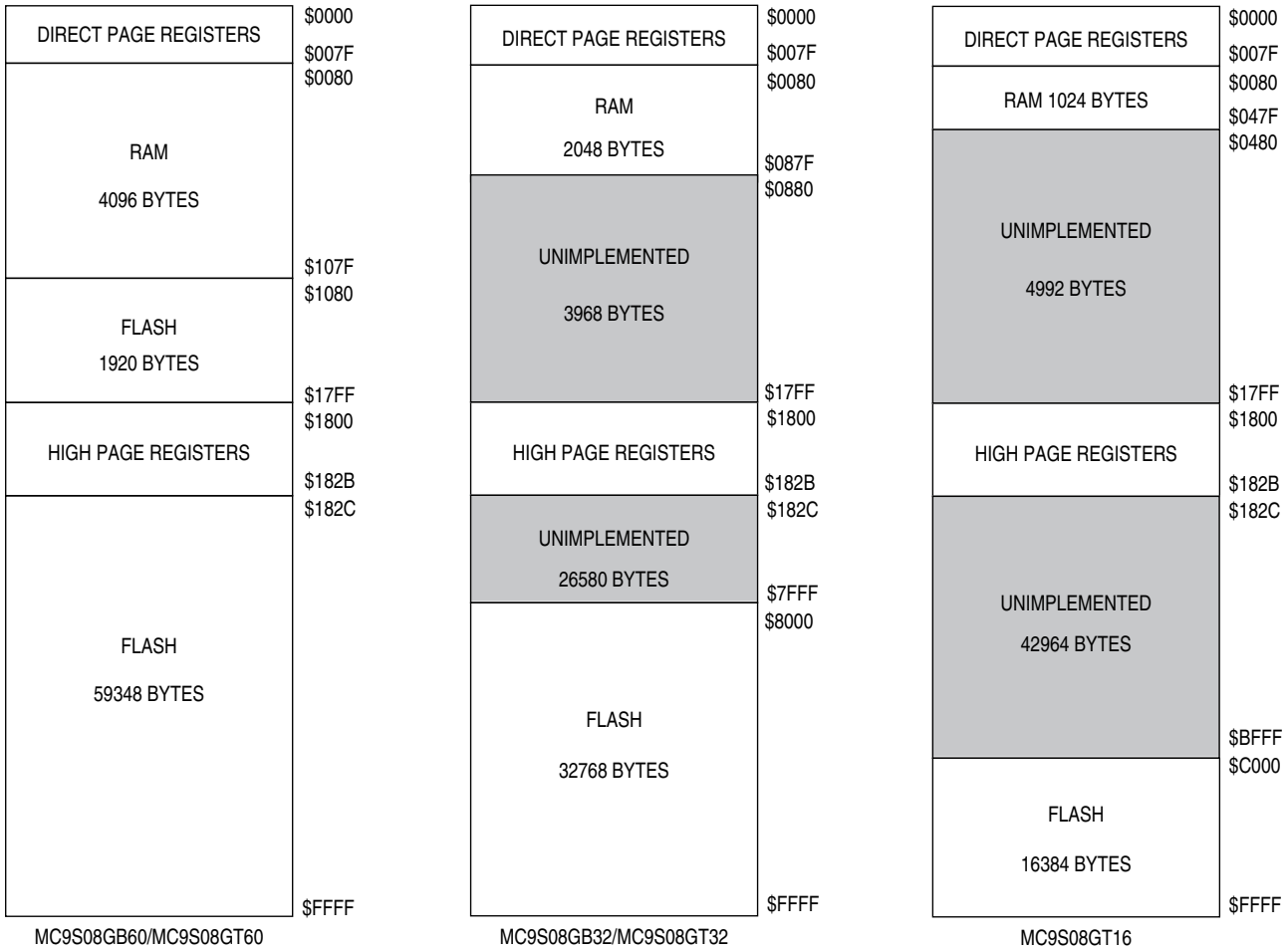


Figure 4-1. MC9S08GB/GT Memory Map

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 MC9S08GB/GT. 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
$FFC0:FFC1 ⇕ $FFCA:FFCB	Unused Vector Space (available for user program)	
$FFCC:FFCD	RTI	Vrti
$FFCE:FFCF	IIC	Viic1
$FFD0:FFD1	ATD Conversion	Vatd1
$FFD2:FFD3	Keyboard	Vkeyboard1
$FFD4:FFD5	SCI2 Transmit	Vsci2tx
$FFD6:FFD7	SCI2 Receive	Vsci2rx
$FFD8:FFD9	SCI2 Error	Vsci2err
$FFDA:FFDB	SCI1 Transmit	Vsci1tx
$FFDC:FFDD	SCI1 Receive	Vsci1rx
$FFDE:FFDF	SCI1 Error	Vsci1err
$FFE0:FFE1	SPI	Vspi1
$FFE2:FFE3	TPM2 Overflow	Vtpm2ovf
$FFE4:FFE5	TPM2 Channel 4	Vtpm2ch4
$FFE6:FFE7	TPM2 Channel 3	Vtpm2ch3
$FFE8:FFE9	TPM2 Channel 2	Vtpm2ch2
$FFEA:FFEB	TPM2 Channel 1	Vtpm2ch1
$FFEC:FFED	TPM2 Channel 0	Vtpm2ch0
$FFEE:FFEF	TPM1 Overflow	Vtpm1ovf
$FFF0:FFF1	TPM1 Channel 2	Vtpm1ch2
$FFF2:FFF3	TPM1 Channel 1	Vtpm1ch1
$FFF4:FFF5	TPM1 Channel 0	Vtpm1ch0
$FFF6:FFF7	ICG	Vicg
$FFF8:FFF9	Low Voltage Detect	Vlvd
$FFFA:FFFB	IRQ	Virq
$FFFC:FFFD	SWI	Vswi
$FFFE:FFFF	Reset	Vreset

4.2 Register Addresses and Bit Assignments

The registers in the MC9S08GB/GT 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 $1800 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 $FFB0–$FFBF.

  Nonvolatile register locations include:

  — Three values which 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
$0000	PTAD	PTAD7	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
$0001	PTAPE	PTAPE7	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
$0002	PTASE	PTASE7	PTASE6	PTASE5	PTASE4	PTASE3	PTASE2	PTASE1	PTASE0
$0003	PTADD	PTADD7	PTADD6	PTADD5	PTADD4	PTADD3	PTADD2	PTADD1	PTADD0
$0004	PTBD	PTBD7	PTBD6	PTBD5	PTBD4	PTBD3	PTBD2	PTBD1	PTBD0
$0005	PTBPE	PTBPE7	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
$0006	PTBSE	PTBSE7	PTBSE6	PTBSE5	PTBSE4	PTBSE3	PTBSE2	PTBSE1	PTBSE0
$0007	PTBDD	PTBDD7	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2	PTBDD1	PTBDD0
$0008	PTCD	PTCD7	PTCD6	PTCD5	PTCD4	PTCD3	PTCD2	PTCD1	PTCD0
$0009	PTCPE	PTCPE7	PTCPE6	PTCPE5	PTCPE4	PTCPE3	PTCPE2	PTCPE1	PTCPE0
$000A	PTCSE	PTCSE7	PTCSE6	PTCSE5	PTCSE4	PTCSE3	PTCSE2	PTCSE1	PTCSE0
$000B	PTCDD	PTCDD7	PTCDD6	PTCDD5	PTCDD4	PTCDD3	PTCDD2	PTCDD1	PTCDD0
$000C	PTDD	PTDD7	PTDD6	PTDD5	PTDD4	PTDD3	PTDD2	PTDD1	PTDD0
$000D	PTDPE	PTDPE7	PTDPE6	PTDPE5	PTDPE4	PTDPE3	PTDPE2	PTDPE1	PTDPE0
$000E	PTDSE	PTDSE7	PTDSE6	PTDSE5	PTDSE4	PTDSE3	PTDSE2	PTDSE1	PTDSE0
$000F	PTDDD	PTDDD7	PTDDD6	PTDDD5	PTDDD4	PTDDD3	PTDDD2	PTDDD1	PTDDD0
$0010	PTED	PTED7	PTED6	PTED5	PTED4	PTED3	PTED2	PTED1	PTED0
$0011	PTEPE	PTEPE7	PTEPE6	PTEPE5	PTEPE4	PTEPE3	PTEPE2	PTEPE1	PTEPE0
$0012	PTESE	PTESE7	PTESE6	PTESE5	PTESE4	PTESE3	PTESE2	PTESE1	PTESE0
$0013	PTEDD	PTEDD7	PTEDD6	PTEDD5	PTEDD4	PTEDD3	PTEDD2	PTEDD1	PTEDD0
$0014	IRQSC	0	0	IRQEDG	IRQPE	IRQF	IRQACK	IRQIE	IRQMOD
$0015	Reserved	—	—	—	—	—	—	—	—
$0016	KBI1SC	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBF	KBACK	KBIE	KBIMOD
$0017	KBI1PE	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
$0018	SCI1BDH	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
$0019	SCI1BDL	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
$001A	SCI1C1	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
$001B	SCI1C2	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
$001C	SCI1S1	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
$001D	SCI1S2	0	0	0	0	0	0	0	RAF
$001E	SCI1C3	R8	T8	TXDIR	0	ORIE	NEIE	FEIE	PEIE
$001F	SCI1D	Bit 7	6	5	4	3	2	1	Bit 0
$0020	SCI2BDH	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
$0021	SCI2BDL	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
$0022	SCI2C1	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
$0023	SCI2C2	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
$0024	SCI2S1	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
$0025	SCI2S2	0	0	0	0	0	0	0	RAF
$0026	SCI2C3	R8	T8	TXDIR	0	ORIE	NEIE	FEIE	PEIE
$0027	SCI2D	Bit 7	6	5	4	3	2	1	Bit 0

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

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$0028	SPI1C1	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
$0029	SPI1C2	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
$002A	SPI1BR	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
$002B	SPI1S	SPRF	0	SPTEF	MODF	0	0	0	0
$002C	Reserved	0	0	0	0	0	0	0	0
$002D	SPI1D	Bit 7	6	5	4	3	2	1	Bit 0
$002E	Reserved	0	0	0	0	0	0	0	0
$002F	Reserved	0	0	0	0	0	0	0	0
$0030	TPM1SC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
$0031	TPM1CNTH	Bit 15	14	13	12	11	10	9	Bit 8
$0032	TPM1CNTL	Bit 7	6	5	4	3	2	1	Bit 0
$0033	TPM1MODH	Bit 15	14	13	12	11	10	9	Bit 8
$0034	TPM1MODL	Bit 7	6	5	4	3	2	1	Bit 0
$0035	TPM1C0SC	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
$0036	TPM1C0VH	Bit 15	14	13	12	11	10	9	Bit 8
$0037	TPM1C0VL	Bit 7	6	5	4	3	2	1	Bit 0
$0038	TPM1C1SC	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0
$0039	TPM1C1VH	Bit 15	14	13	12	11	10	9	Bit 8
$003A	TPM1C1VL	Bit 7	6	5	4	3	2	1	Bit 0
$003B	TPM1C2SC	CH2F	CH2IE	MS2B	MS2A	ELS2B	ELS2A	0	0
$003C	TPM1C2VH	Bit 15	14	13	12	11	10	9	Bit 8
$003D	TPM1C2VL	Bit 7	6	5	4	3	2	1	Bit 0
$003E–$003F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$0040	PTFD	PTFD7	PTFD6	PTFD5	PTFD4	PTFD3	PTFD2	PTFD1	PTFD0
$0041	PTFPE	PTFPE7	PTFPE6	PTFPE5	PTFPE4	PTFPE3	PTFPE2	PTFPE1	PTFPE0
$0042	PTFSE	PTFSE7	PTFSE6	PTFSE5	PTFSE4	PTFSE3	PTFSE2	PTFSE1	PTFSE0
$0043	PTFDD	PTFDD7	PTFDD6	PTFDD5	PTFDD4	PTFDD3	PTFDD2	PTFDD1	PTFDD0
$0044	PTGD	PTGD7	PTGD6	PTGD5	PTGD4	PTGD3	PTGD2	PTGD1	PTGD0
$0045	PTGPE	PTGPE7	PTGPE6	PTGPE5	PTGPE4	PTGPE3	PTGPE2	PTGPE1	PTGPE0
$0046	PTGSE	PTGSE7	PTGSE6	PTGSE5	PTGSE4	PTGSE3	PTGSE2	PTGSE1	PTGSE0
$0047	PTGDD	PTGDD7	PTGDD6	PTGDD5	PTGDD4	PTGDD3	PTGDD2	PTGDD1	PTGDD0
$0048	ICGC1	0	RANGE	REFS	CLKS		OSCSTEN	0*	0
$0049	ICGC2	LOLRE	MFD			LOCRE	RFD		
$004A	ICGS1	CLKST		REFST	LOLS	LOCK	LOCS	ERCS	ICGIF
$004B	ICGS2	0	0	0	0	0	0	0	DCOS
$004C	ICGFLTU	0	0	0	0	FLT			
$004D	ICGFLTL	FLT							
$004E	ICGTRM	TRIM							

* This bit is reserved for Freescale Semiconductor internal use only. Always write a 0 to this bit.

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

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$004F	Reserved	0	0	0	0	0	0	0	0
$0050	ATD1C	ATDPU	DJM	RES8	SGN	PRS			
$0051	ATD1SC	CCF	ATDIE	ATDCO	ATDCH				
$0052	ATD1RH	Bit 7	6	5	4	3	2	1	Bit 0
$0053	ATD1RL	Bit 7	6	5	4	3	2	1	Bit 0
$0054	ATD1PE	ATDPE7	ATDPE6	ATDPE5	ATDPE4	ATDPE3	ATDPE2	ATDPE1	ATDPE0
$0055–$0057	Reserved	—	—	—	—	—	—	—	—
$0058	IIC1A	ADDR							0
$0059	IIC1F	MULT		ICR					
$005A	IIC1C	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
$005B	IIC1S	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
$005C	IIC1D	DATA							
$005D–$005F	Reserved	—	—	—	—	—	—	—	—
$0060	TPM2SC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
$0061	TPM2CNTH	Bit 15	14	13	12	11	10	9	Bit 8
$0062	TPM2CNTL	Bit 7	6	5	4	3	2	1	Bit 0
$0063	TPM2MODH	Bit 15	14	13	12	11	10	9	Bit 8
$0064	TPM2MODL	Bit 7	6	5	4	3	2	1	Bit 0
$0065	TPM2C0SC	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
$0066	TPM2C0VH	Bit 15	14	13	12	11	10	9	Bit 8
$0067	TPM2C0VL	Bit 7	6	5	4	3	2	1	Bit 0
$0068	TPM2C1SC	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0
$0069	TPM2C1VH	Bit 15	14	13	12	11	10	9	Bit 8
$006A	TPM2C1VL	Bit 7	6	5	4	3	2	1	Bit 0
$006B	TPM2C2SC	CH2F	CH2IE	MS2B	MS2A	ELS2B	ELS2A	0	0
$006C	TPM2C2VH	Bit 15	14	13	12	11	10	9	Bit 8
$006D	TPM2C2VL	Bit 7	6	5	4	3	2	1	Bit 0
$006E	TPM2C3SC	CH3F	CH3IE	MS3B	MS3A	ELS3B	ELS3A	0	0
$006F	TPM2C3VH	Bit 15	14	13	12	11	10	9	Bit 8
$0070	TPM2C3VL	Bit 7	6	5	4	3	2	1	Bit 0
$0071	TPM2C4SC	CH4F	CH4IE	MS4B	MS4A	ELS4B	ELS4A	0	0
$0072	TPM2C4VH	Bit 15	14	13	12	11	10	9	Bit 8
$0073	TPM2C4VL	Bit 7	6	5	4	3	2	1	Bit 0
$0074–$007F	Reserved	—	—	—	—	—	—	—	—

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 $1800.

Table 4-3. High-Page Register Summary

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$1800	SRS	POR	PIN	COP	ILOP	0	ICG	LVD	0
$1801	SBDFR	0	0	0	0	0	0	0	BDFR
$1802	SOPT	COPE	COPT	STOPE	—	0	0	BKGDPE	—
$1803 – $1805	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$1806	SDIDH	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
$1807	SDIDL	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
$1808	SRTISC	RTIF	RTIACK	RTICLKS	RTIE	0	RTIS2	RTIS1	RTIS0
$1809	SPMSC1	LVDF	LVDACK	LVDIE	LVDRE	LVDSE	LVDE	0	0
$180A	SPMSC2	LVWF	LVWACK	LVDV	LVWV	PPDF	PPDACK	PDC	PPDC
$180B– $180F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$1810	DBGCAH	Bit 15	14	13	12	11	10	9	Bit 8
$1811	DBGCAL	Bit 7	6	5	4	3	2	1	Bit 0
$1812	DBGCBH	Bit 15	14	13	12	11	10	9	Bit 8
$1813	DBGCBL	Bit 7	6	5	4	3	2	1	Bit 0
$1814	DBGFH	Bit 15	14	13	12	11	10	9	Bit 8
$1815	DBGFL	Bit 7	6	5	4	3	2	1	Bit 0
$1816	DBGC	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
$1817	DBGT	TRGSEL	BEGIN	0	0	TRG3	TRG2	TRG1	TRG0
$1818	DBGS	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0
$1819– $181F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$1820	FCDIV	DIVLD	PRDIV8	DIV5	DIV4	DIV3	DIV2	DIV1	DIV0
$1821	FOPT	KEYEN	FNORED	0	0	0	0	SEC01	SEC00
$1822	Reserved	—	—	—	—	—	—	—	—
$1823	FCNFG	0	0	KEYACC	0	0	0	0	0
$1824	FPROT	FPOPEN	FPDIS	FPS2	FPS1	FPS0	0	0	0
$1825	FSTAT	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
$1826	FCMD	FCMD7	FCMD6	FCMD5	FCMD4	FCMD3	FCMD2	FCMD1	FCMD0
$1827– $182B	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
$FFB0 – $FFB7	NVBACKKEY	8-Byte Comparison Key							
$FFB8 – $FFBC	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$FFBD	NVPROT	FPOPEN	FPDIS	FPS2	FPS1	FPS0	0	0	0
$FFBE	Reserved[1]	—	—	—	—	—	—	—	—
$FFBF	NVOPT	KEYEN	FNORED	0	0	0	0	SEC01	SEC00

[1] This location is used to store the factory trim value for the ICG.

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 MC9S08GB/GT includes static RAM. The locations in RAM below $0100 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 $00FF. In the MC9S08GB/GT, 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.

4.4.1 Features

Features of the FLASH memory include:

- FLASH Size
  - MC9S08GB60/MC9S08GT60 — 61268 bytes (120 pages of 512 bytes each)
  - MC9S08GB32/MC9S08GT32— 32768 bytes (64 pages of 512 bytes each)
  - MC9S08GT16 — 16384 bytes (32 pages of 512 bytes each)
- 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

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 Table 4.6.1). 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 blocks of FLASH that may be erased. In the 60K version, there are two instances where the size of a block that is accessible to the user is less than 512 bytes: the first page following RAM, and the first page following the high page registers. These pages are overlapped by the RAM and high page registers, respectively.

   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 ($05), byte program ($20), burst program ($25), page erase ($40), and mass erase ($41). 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 adhered to, or the command will not be accepted. This minimizes the possibility of any unintended change 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. This must be done only once following a reset.

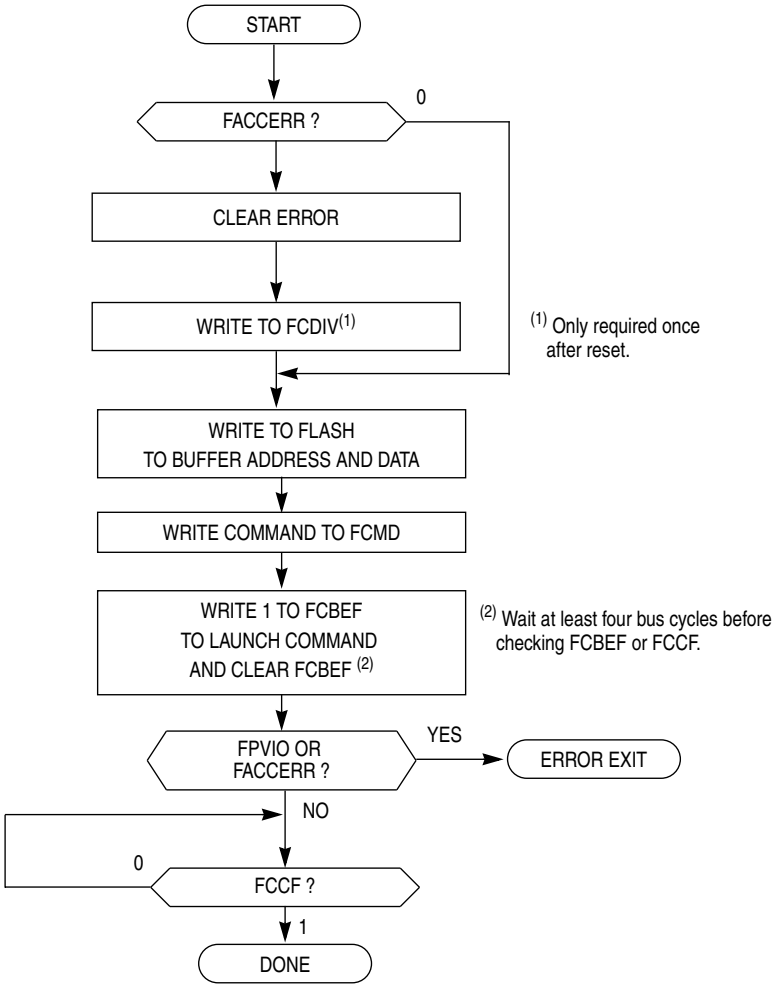


Figure 4-2. FLASH Program and Erase Flowchart

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. 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 the following two conditions are met:

1. The next burst program command has been queued before the current program operation has completed.
2. 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.

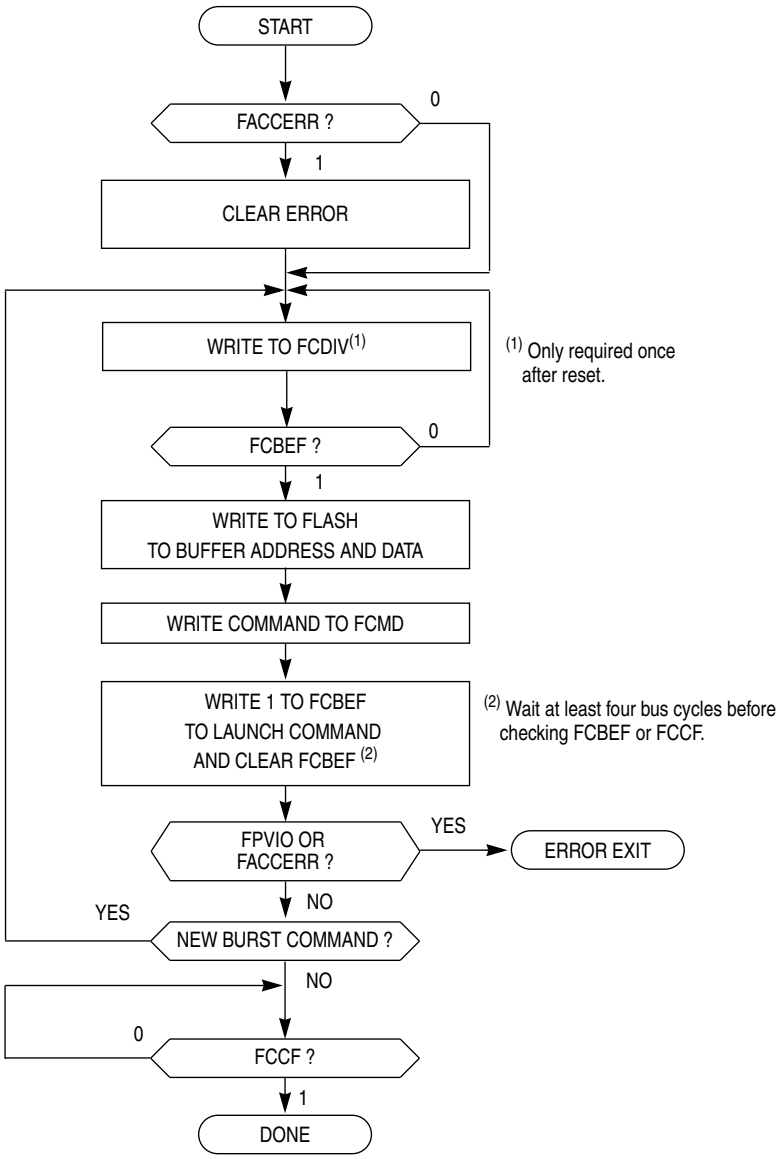


Figure 4-3. FLASH Burst Program Flowchart

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 ($05, $20, $25, $40, or $41) 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 ($20, $25, or $40) with a background debug command while the MCU is secured (The background debug controller can only do blank check and mass erase commands when the MCU is secure.)
- Writing 0 to FCBEF to cancel a partial command

4.4.6 FLASH Block Protection

Block protection prevents program or erase changes for FLASH memory locations in a designated address range. Mass erase is disabled when any block of FLASH is protected. The MC9S08GB/GT allows a block of memory at the end of FLASH, and/or the entire FLASH memory to be block protected. A disable control bit and a 3-bit control field, allows the user to set the size of this block. A separate control bit allows block protection of the entire FLASH memory array. All seven of these control bits are located in the FPROT register (see Section 4.6.4, “FLASH Protection Register (FPROT and NVPROT)”).

At reset, the high-page register (FPROT) is loaded with the contents of the NVPROT location which is in the nonvolatile register block of the FLASH memory. The value in FPROT cannot be changed directly from application software so a runaway program cannot alter the block protection settings. If the last 512 bytes of FLASH which includes the NVPROT register is protected, the application program cannot alter the block protection settings (intentionally or unintentionally). The FPROT control bits can be written by background debug commands to allow a way to erase a protected FLASH memory.

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

4.4.7 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 $FFBF 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 $FFBD. All of the interrupt vectors (memory locations $FFC0–$FFFD) are redirected, while the reset vector ($FFFE:FFFF) is not. When more than 32K of memory is protected, vector redirection must not be enabled.

For example, if 512 bytes of FLASH are protected, the protected address region is from $FE00 through $FFFF. The interrupt vectors ($FFC0–$FFFD) are redirected to the locations $FDC0–$FDFD. Now, if an SPI interrupt is taken for instance, the values in the locations $FDE0:FDE1 are used for the vector instead of the values in the locations $FFE0:FFE1. This allows the user to reprogram the unprotected portion of the FLASH with new program code including new interrupt vector values while leaving the protected area, which includes the default vector locations, unchanged.

4.5 Security

The MC9S08GB/GT 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 RAM, so it cannot be entered through background commands without the cooperation of a secure user program. The FLASH memory cannot be accessed by read operations while KEYACC is set.

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 has nine 8-bit registers in the high-page register space, three locations in the nonvolatile register space in FLASH memory that are copied into three 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.

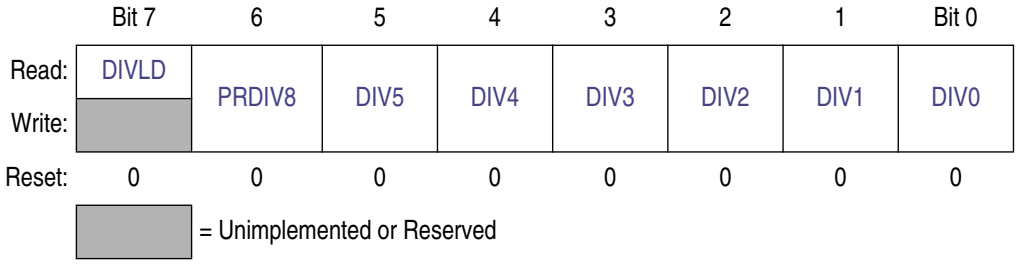


Figure 4-4. FLASH Clock Divider Register (FCDIV)

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.

1 = FCDIV has been written since reset; erase and program operations enabled for FLASH.
0 = FCDIV has not been written since reset; erase and program operations disabled for FLASH.

PRDIV8 — Prescale (Divide) FLASH Clock by 8

1 = Clock input to the FLASH clock divider is the bus rate clock divided by 8.
0 = Clock input to the FLASH clock divider is the bus rate clock.

DIV5:DIV0 — 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.

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

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

Table 4-6 shows the appropriate values for PRDIV8 and DIV5:DIV0 for selected bus frequencies.

Table 4-6. 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.

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

= Unimplemented or Reserved

Figure 4-5. FLASH Options Register (FOPT)

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."

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.

0 = No backdoor key access allowed.

FNORED — Vector Redirection Disable

When this bit is 1, vector redirection is disabled.

1 = Vector redirection disabled.

0 = Vector redirection enabled.

SEC01:SEC00 — Security State Code

This 2-bit field determines the security state of the MCU as shown in Table 4-7. 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 4-7. Security States

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

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

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.

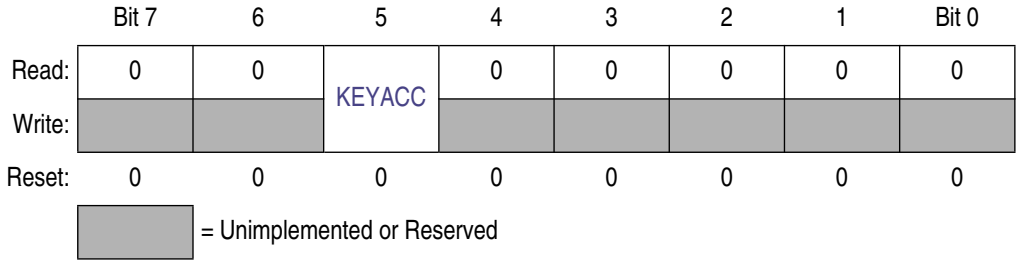


Figure 4-6. FLASH Configuration Register (FCNFG)

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.”

1 = Writes to NVBACKKEY ($FFB0–$FFB7) are interpreted as comparison key writes. Reads of the FLASH return invalid data.

0 = Writes to $FFB0–$FFB7 are interpreted as the start of a FLASH programming or erase command.

4.6.4 FLASH Protection Register (FPROT and NVPROT)

During reset, the contents of the nonvolatile location NVPROT is copied from FLASH into FPROT. Bits 0, 1, and 2 are not used and each always reads as 0. This register may be read at any time, but user program writes have no meaning or effect. Background debug commands can write to FPROT.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	FPOPEN	FPDIS	FPS2	FPS1	FPS0	0	0	0
Write:	1	(1)	(1)	(1)	(1)			
Reset:	This register is loaded from nonvolatile location NVPROT during reset.							

= Unimplemented or Reserved

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

Figure 4-7. FLASH Protection Register (FPROT)

FPOPEN — Open Unprotected FLASH for Program/Erase

1 = Any FLASH location, not otherwise block protected or secured, may be erased or programmed.
0 = Entire FLASH memory is block protected (no program or erase allowed).

FPDIS — FLASH Protection Disable

1 = No FLASH block is protected.
0 = FLASH block specified by FPS2:FPS0 is block protected (program and erase not allowed).

FPS2:FPS1:FPS0 — FLASH Protect Size Selects

When FPDIS = 0, this 3-bit field determines the size of a protected block of FLASH locations at the high address end of the FLASH (see Table 4-8). Protected FLASH locations cannot be erased or programmed.

Table 4-8. High Address Protected Block

FPS2:FPS1:FPS0	Protected Address Range	Protected Block Size	Redirected Vectors[1]
0:0:0	$FE00–$FFFF	512 bytes	$FDC0–$FDFD[2]
0:0:1	$FC00–$FFFF	1024 bytes	$FBC0–$FBFD
0:1:0	$F800–$FFFF	2048 bytes	$F7C0–$F7FD
0:1:1	$F000–$FFFF	4096 bytes	$EFC0–$EFFD
1:0:0	$E000–$FFFF	8192 bytes	$DFC0–$DFFD
1:0:1	$C000–$FFFF	16384 bytes	$BFC0–$BFFD[3]
1:1:0	$8000–$FFFF	32768 bytes	$7FC0–$7FFD[4]
1:1:1	$8000–$FFFF	32768 bytes	$7FC0–$7FFD[4]

1 No redirection if FPOPEN = 0, or FNORED = 1.
2 Reset vector is not redirected.
3 32K and 60K devices only.
4 60K devices only.

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.

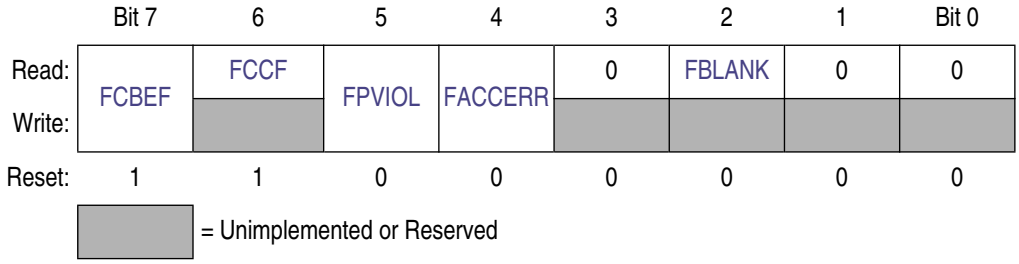


Figure 4-8. FLASH Status Register (FSTAT)

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.

1 = A new burst program command may be written to the command buffer.
0 = Command buffer is full (not ready for additional commands).

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.

1 = All commands complete
0 = Command in progress

FPVIOL — Protection Violation Flag

FPVIOL is set automatically when FCBEF is cleared to register a command that attempts to erase or program a location in a protected block (the erroneous command is ignored). FPVIOL is cleared by writing a 1 to FPVIOL.

1 = An attempt was made to erase or program a protected location.
0 = No protection violation.

FACCERR — 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.

1 = An access error has occurred.
0 = No access error has occurred.

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.

1 = After a blank check command is completed and FCCF = 1, FBLANK = 1 indicates the FLASH array is completely erased (all $FF).

0 = After a blank check command is completed and FCCF = 1, FBLANK = 0 indicates the FLASH array is not completely erased.

4.6.6 FLASH Command Register (FCMD)

Only five command codes are recognized in normal user modes as shown in Table 4-9. Refer to Section 4.4.3, “Program and Erase Command Execution” for a detailed discussion of FLASH programming and erase operations.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	0
Write:	FCMD7	FCMD6	FCMD5	FCMD4	FCMD3	FCMD2	FCMD1	FCMD0
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-9. FLASH Command Register (FCMD)

Table 4-9. FLASH Commands

Command	FCMD	Equate File Label
Blank check	$05	mBlank
Byte program	$20	mByteProg
Byte program — burst mode	$25	mBurstProg
Page erase (512 bytes/page)	$40	mPageErase
Mass erase (all FLASH)	$41	mMassErase

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

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

Chapter 5 Resets, Interrupts, and System Configuration

5.1 Introduction

This section discusses basic reset and interrupt mechanisms and the various sources of reset and interrupts in the MC9S08GB/GT. 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 5-1)

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 ($FFFE:$FFFF). 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 $00FF at reset.

The MC9S08GB/GT 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
- The reset pin ($\overline{\text{RESET}}$)
- 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. The COP watchdog is enabled by the COPE bit in SOPT (see Section 5.8.4, “System Options Register (SOPT)” for additional information). The COP timer 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 timer.

After any reset, the COP timer is enabled. This provides a reliable way to detect code that is not executing as intended. If the COP watchdog is not used in an application, it can be disabled by clearing the COPE bit in the write-once SOPT register. Also, the COPT bit can be used to choose one of two timeout periods (2^{18} or 2^{13} cycles of the bus rate clock). Even if the application will use the reset default settings in COPE and COPT, the user should still write to write-once SOPT during reset initialization to lock in the settings. That way, they cannot be changed accidentally if the application program gets lost.

The write to SRS that services (clears) the COP timer should 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 MCU is in active background mode, the COP timer is temporarily disabled.

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 5-1).

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.

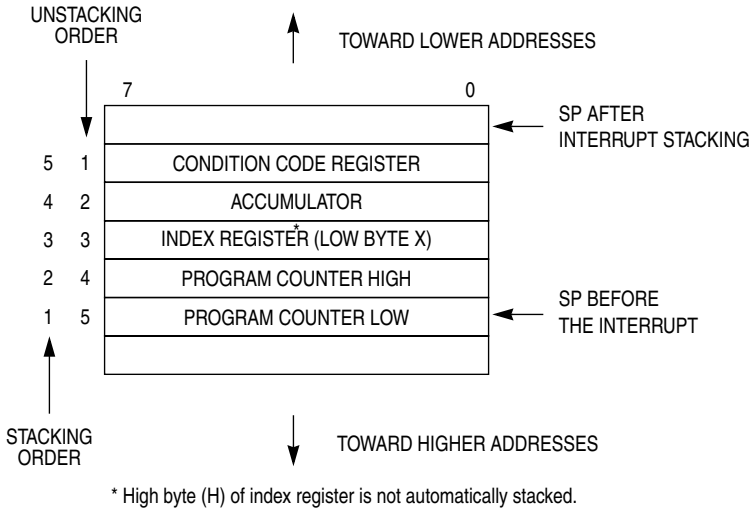


Figure 5-1. Interrupt Stack Frame

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 5-1 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 5-1. Vector Summary

Vector Priority	Vector Number	Address (High/Low)	Vector Name	Module	Source	Enable	Description
Lower	26 through 31	$FFC0/FFC1 through $FFCA/FFCB	Unused Vector Space (available for user program)				
	25	$FFCC/FFCD	Vrti	System control	RTIF	RTIE	Real-time interrupt
	24	$FFCE/FFCF	Viic1	IIC	IICIS	IICIE	IIC control
	23	$FFD0/FFD1	Vatd1	ATD	COCO	AIEN	AD conversion complete
	22	$FFD2/FFD3	Vkeyboard1	KBI	KBF	KBIE	Keyboard pins
	21	$FFD4/FFD5	Vsci2tx	SCI2	TDRE TC	TIE TCIE	SCI2 transmit
	20	$FFD6/FFD7	Vsci2rx	SCI2	IDLE RDRF	ILIE RIE	SCI2 receive
	19	$FFD8/FFD9	Vsci2err	SCI2	OR NF FE PF	ORIE NFIE FEIE PFIE	SCI2 error
	18	$FFDA/FFDB	Vsci1tx	SCI1	TDRE TC	TIE TCIE	SCI1 transmit
	17	$FFDC/FFDD	Vsci1rx	SCI1	IDLE RDRF	ILIE RIE	SCI1 receive
	16	$FFDE/FFDF	Vsci1err	SCI1	OR NF FE PF	ORIE NFIE FEIE PFIE	SCI1 error
	15	$FFE0/FFE1	Vspi1	SPI	SPIF MODF SPTEF	SPIE SPIE SPTIE	SPI
	14	$FFE2/FFE3	Vtpm2ovf	TPM2	TOF	TOIE	TPM2 overflow
	13	$FFE4/FFE5	Vtpm2ch4	TPM2	CH4F	CH4IE	TPM2 channel 4
	12	$FFE6/FFE7	Vtpm2ch3	TPM2	CH3F	CH3IE	TPM2 channel 3
	11	$FFE8/FFE9	Vtpm2ch2	TPM2	CH2F	CH2IE	TPM2 channel 2
	10	$FFEA/FFEB	Vtpm2ch1	TPM2	CH1F	CH1IE	TPM2 channel 1
	9	$FFEC/FFED	Vtpm2ch0	TPM2	CH0F	CH0IE	TPM2 channel 0
	8	$FFEE/FFEF	Vtpm1ovf	TPM1	TOF	TOIE	TPM1 overflow
	7	$FFF0/FFF1	Vtpm1ch2	TPM1	CH2F	CH2IE	TPM1 channel 2
	6	$FFF2/FFF3	Vtpm1ch1	TPM1	CH1F	CH1IE	TPM1 channel 1
	5	$FFF4/FFF5	Vtpm1ch0	TPM1	CH0F	CH0IE	TPM1 channel 0
	4	$FFF6/FFF7	Vicg	ICG	ICGIF (LOLS/LOCS)	LOLRE/LOCRE	ICG
	3	$FFF8/FFF9	Vlvd	System control	LVDF	LVDIE	Low-voltage detect
	2	$FFFA/FFFB	Virq	IRQ	IRQF	IRQIE	IRQ pin
	1	$FFFC/FFFD	Vswi	Core	SWI Instruction	—	Software interrupt
Higher	0	$FFFE/FFFF	Vreset	System control	COP LVD $\overline{\text{RESET}}$ pin Illegal opcode	COPE LVDRE — —	Watchdog timer Low-voltage detect External pin Illegal opcode

5.6 Low-Voltage Detect (LVD) System

The MC9S08GB/GT 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 SPMSC2.

5.7 Real-Time Interrupt (RTI)

The real-time interrupt function can be used to generate periodic interrupts based on a multiple of the source clock's period. The RTI has two source clock choices, the external clock input (ICGERCLK) to the ICG or the RTI's own internal clock. The RTI can be used in run, wait, stop2 and stop3 modes. It is not available in stop1 mode.

In run and wait modes, only the external clock can be used as the RTI's clock source. In stop2 mode, only the internal RTI clock can be used. In stop3, either the external clock or internal RTI clock can be used.

When using the external oscillator in stop3 mode, it must be enabled in stop (OSCSTEN = 1) and configured for low bandwidth operation (RANGE = 0).

The SRTISC register includes a read-only status flag, a write-only acknowledge bit, and a 3-bit control value (RTIS2:RTIS1:RTIS0) used to select one of seven RTI periods. The RTI has a local interrupt enable, RTIE, to allow masking of the real-time interrupt. The module can be disabled by writing 0:0:0 to RTIS2:RTIS1:RTIS0 in which case the clock source input is disabled and no interrupts will be generated. See Section 5.8.6, "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 SOPT and SPMSC2 registers are related to modes of operation. Although brief descriptions of these bits are provided here, the related functions are discussed in greater detail in Chapter 3, "Modes of Operation."

5.8.1 Interrupt Pin Request Status and Control Register (IRQSC)

This direct page register includes two unimplemented bits which always read 0, four read/write bits, one read-only status bit, and one write-only bit. These bits are used to configure the IRQ function, report status, and acknowledge IRQ events.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	IRQEDG	IRQPE	IRQF	0	IRQIE	IRQMOD
Write:						IRQACK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

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

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.

1 = IRQ is rising edge or rising edge/high-level sensitive.
0 = IRQ is falling edge or falling edge/low-level sensitive.

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.

1 = IRQ pin function is enabled.
0 = IRQ pin function is disabled.

IRQF — IRQ Flag

This read-only status bit indicates when an interrupt request event has occurred.

1 = IRQ event detected.
0 = No IRQ request.

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.

IRQIE — IRQ Interrupt Enable

This read/write control bit determines whether IRQ events generate a hardware interrupt request.

1 = Hardware interrupt requested whenever IRQF = 1.
0 = Hardware interrupt requests from IRQF disabled (use polling).

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.

1 = IRQ event on falling edges and low levels or on rising edges and high levels.
0 = IRQ event on falling edges or rising edges only.

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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	POR	PIN	COP	ILOP	0	ICG	LVD	0
Write:	Writing any value to SIMRS address clears COP watchdog timer.							
Power-on reset:	1	0	0	0	0	0	1	0
Low-voltage reset:	U	0	0	0	0	0	1	0
Any other reset:	0	1	(1)	(1)	0	(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)

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.

1 = POR caused reset.
0 = Reset not caused by POR.

PIN — External Reset Pin

Reset was caused by an active-low level on the external reset pin.

1 = Reset came from external reset pin.
0 = Reset not caused by external reset pin.

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.

1 = Reset caused by COP timeout.
0 = Reset not caused by COP timeout.

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.

1 = Reset caused by an illegal opcode.
0 = Reset not caused by an illegal opcode.

ICG — Internal Clock Generation Module Reset

Reset was caused by an ICG module reset.

1 = Reset caused by ICG module.
0 = Reset not caused by ICG module.

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.

1 = Reset caused by LVD trip or POR.

0 = Reset not 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 $00.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	0
Write:								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)

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 (SOPT)

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 SOPT (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT 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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COPE	COPT	STOPE		0	0	BKGDPE	
Write:								
Reset:	1	1	0	1	0	0	1	1

= Unimplemented or Reserved

Figure 5-5. System Options Register (SOPT)

COPE — COP Watchdog Enable

This write-once bit defaults to 1 after reset.
1 = COP watchdog timer enabled (force reset on timeout).
0 = COP watchdog timer disabled.

COPT — COP Watchdog Timeout

This write-once bit defaults to 1 after reset.
1 = Long timeout period selected (2^{18} cycles of BUSCLK).
0 = Short timeout period selected (2^{13} cycles of BUSCLK).

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.
1 = Stop mode enabled.
0 = Stop mode disabled.

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.
1 = BKGD pin enabled.
0 = BKGD pin disabled.

5.8.5 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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
Reset:	0[1]	0[(1)]	0[(1)]	0[(1)]	0	0	0	0
Read:	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
Reset:	0	0	0	0	0	0	1	0

= Unimplemented or Reserved

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

Figure 5-6. System Device Identification Register (SDIDH, SDIDL)

REV[3:0] — Revision Number

The high-order 4 bits of address $1806 are hard coded to reflect the current mask set revision number (0–F).

ID[11:0] — Part Identification Number

Each derivative in the HCS08 Family has a unique identification number. The MC9S08GB/GT is hard coded to the value $002.

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

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

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RTIF	0	RTICLKS	RTIE	0	RTIS2	RTIS1	RTIS0
Write:		RTIACK						
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-7. System RTI Status and Control Register (SRTISC)

RTIF — Real-Time Interrupt Flag

This read-only status bit indicates the periodic wakeup timer has timed out.
1 = Periodic wakeup timer timed out.
0 = Periodic wakeup timer not timed out.

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.

RTICLKS — Real-Time Interrupt Clock Select

This read/write bit selects the clock source for the real-time interrupt.
1 = Real-time interrupt request clock source is external clock.
0 = Real-time interrupt request clock source is internal oscillator.

RTIE — Real-Time Interrupt Enable

This read-write bit enables real-time interrupts.
1 = Real-time interrupts enabled.
0 = Real-time interrupts disabled.

RTIS2:RTIS1:RTIS0 — Real-Time Interrupt Period Selects

These read/write bits select the wakeup period for the RTI. The clock source for the real-time interrupt is its own clock source, which oscillates with a period of approximately $1/f_{ext}$, and it is independent of other MCU clock sources. Using an external clock source, the delays will be crystal frequency divided by value in RTIS2:RTIS1:RTIS0.

Table 5-2. Real-Time Interrupt Period

RTIS2:RTIS1:RTIS0	Internal Clock Source [1] (t_{RTI} = 1 ms, Nominal)	External Clock Source [2] Period = t_{ext}
0:0:0	Disable periodic wakeup timer	Disable periodic wakeup timer
0:0:1	8 ms	t_{ext} x 256
0:1:0	32 ms	t_{ex} x 1024
0:1:1	64 ms	t_{ex} x 2048
1:0:0	128 ms	t_{ex} 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_{ex} x 32768

1 See Table A-10 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-9 for details.

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

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LVDF	0	LVDIE	LVDRE[1]	LVDSE(1)	LVDE(1)	0	0
Write:		LVDACK						
Reset:	0	0	0	1	1	1	0	0

= Unimplemented or Reserved

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

Figure 5-8. System Power Management Status and Control 1 Register (SPMSC1)

LVDF — Low-Voltage Detect Flag

Provided LVDE = 1, this read-only status bit indicates a low-voltage detect event.

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.

LVDIE — Low-Voltage Detect Interrupt Enable

This read/write bit enables hardware interrupt requests for LVDF.
1 = Request a hardware interrupt when LVDF = 1.
0 = Hardware interrupt disabled (use polling).

LVDRE — Low-Voltage Detect Reset Enable

This read/write bit enables LVDF events to generate a hardware reset (provided LVDE = 1).
1 = Force an MCU reset when LVDF = 1.
0 = LVDF does not generate hardware resets.

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.
1 = Low-voltage detect enabled during stop mode.
0 = Low-voltage detect disabled during stop mode.

LVDE — Low-Voltage Detect Enable

This read/write bit enables low-voltage detect logic and qualifies the operation of other bits in this register.
1 = LVD logic enabled.
0 = LVD logic disabled.

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

This register is used to report the status of the low voltage warning function, and to configure the stop mode behavior of the MCU.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LVWF	0	LVDV	LVWV	PPDF	0	PDC	PPDC
Write:		LVWACK				PPDACK		
Power-on reset:	0[1]	0	0	0	0	0	0	0
LVD reset:	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 5-9. System Power Management Status and Control 2 Register (SPMSC2)

LVWF — Low-Voltage Warning Flag

The LVWF bit indicates the low-voltage warning status.
1 = Low voltage warning is present or was present.
0 = Low voltage warning **not** present.

LVWACK — Low-Voltage Warning Acknowledge

The LVWACK bit indicates the low-voltage warning acknowledge.

Writing a 1 to LVWACK clears LVWF to 0 if a low voltage warning is not present.

LVDV — Low-Voltage Detect Voltage Select

The LVDV bit selects the LVD trip point voltage (V_{LVD}).

1 = High trip point selected ($V_{LVD} = V_{LVDH}$).

0 = Low trip point selected ($V_{LVD} = V_{LVDL}$).

LVWV — Low-Voltage Warning Voltage Select

The LVWV bit selects the LVW trip point voltage (V_{LVW}).

1 = High trip point selected ($V_{LVW} = V_{LVWH}$).

0 = Low trip point selected ($V_{LVW} = V_{LVWL}$).

PPDF — Partial Power Down Flag

The PPDF bit indicates that the MCU has exited the stop2 mode.

1 = Stop2 mode recovery.

0 = Not stop2 mode recovery.

PPDACK — Partial Power Down Acknowledge

Writing a 1 to PPDACK clears the PPDF bit.

PDC — Power Down Control

The write-once PDC bit controls entry into the power down (stop2 and stop1) modes.

1 = Power down modes are enabled.

0 = Power down modes are disabled.

PPDC — Partial Power Down Control

The write-once PPDC bit controls which power down mode, stop1 or stop2, is selected.

1 = Stop2, partial power down, mode enabled if PDC set.

0 = Stop1, full power down, mode enabled if PDC set.

Chapter 6 Parallel Input/Output

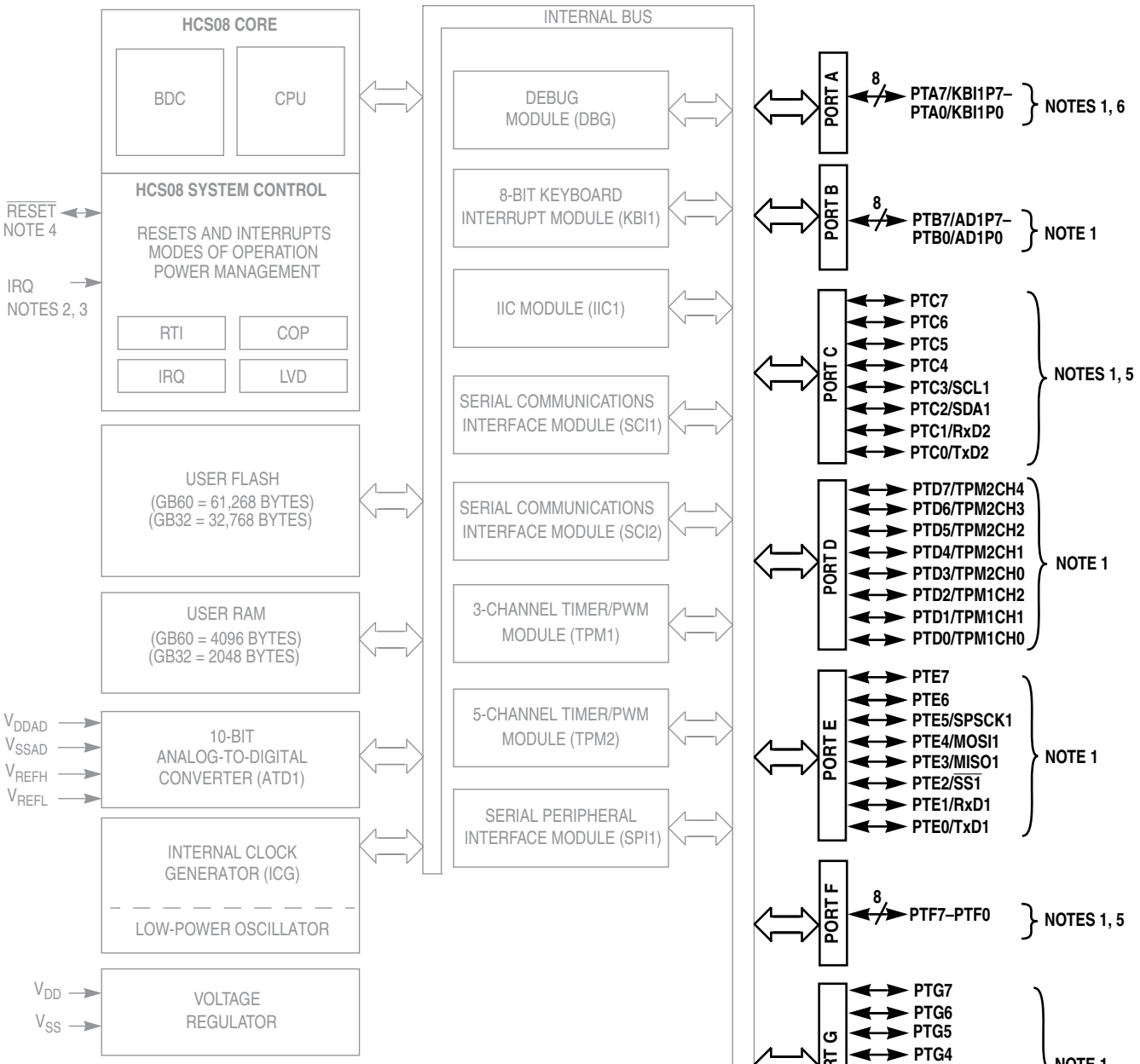
6.1 Introduction

This section explains software controls related to parallel input/output (I/O). The MC9S08GBxx has seven I/O ports which include a total of 56 general-purpose I/O pins (one of these pins is output only). The MC9S08GTxx has six I/O ports which include a total of up to 39 general-purpose I/O pins, depending on the package (one pin, PTG0, is output only). 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, external interrupts, or keyboard interrupts. When these other modules are not controlling the port pins, they revert to general-purpose I/O control. For each I/O pin, 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.



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. **High current drive**
6. **Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).**

Figure 6-1. Block Diagram Highlighting Parallel Input/Output Pins

6.2 Features

Parallel I/O features, depending on package choice, include:

- A total of 56 general-purpose I/O pins in seven ports (PTG0 is output only)
- High-current drivers on port C and port F pins
- Hysteresis input buffers
- Software-controlled pullups on each input pin
- Software-controlled slew rate output buffers
- Eight port A pins shared with KBI1
- Eight port B pins shared with ATD1
- Eight high-current port C pins shared with SCI2 and IIC1
- Eight port D pins shared with TPM1 and TPM2
- Eight port E pins shared with SCI1 and SPI1
- Eight high-current port F pins
- Eight port G pins shared with EXTAL, XTAL, and BKGD/MS

6.3 Pin Descriptions

The MC9S08GB/GT has a total of 56 parallel I/O pins (one is output only) in seven 8-bit ports (PTA–PTG). Not all pins are bonded out in all packages. Consult the pin assignment in Chapter 2, “Pins and Connections,” for available parallel I/O pins. All of these pins are available for general-purpose I/O when they are not used by other on-chip peripheral systems.

After reset, BKGD/MS is enabled and therefore is not usable as an output pin until BKGDPE in SOPT is cleared. The rest of the peripheral functions are disabled. After reset, all data direction and pullup enable controls are set to 0s. These pins default to being high-impedance inputs with on-chip pullup devices disabled.

The following paragraphs discuss each port and the software controls that determine each pin’s use.

6.3.1 Port A and Keyboard Interrupts

Port A		Bit 7	6	5	4	3	2	1	Bit 0
	MCU Pin:	PTA7/ KBI1P7	PTA6/ KBI1P6	PTA5/ KBI1P5	PTA4/ KBI1P4	PTA3/ KBI1P3	PTA2/ KBI1P2	PTA1/ KBI1P1	PTA0/ KBI1P0

Figure 6-2. Port A Pin Names

Port A is an 8-bit port shared among the KBI keyboard interrupt inputs and general-purpose I/O. Any pins enabled as KBI inputs will be forced to act as inputs.

Port A pins are available as general-purpose I/O pins controlled by the port A data (PTAD), data direction (PTADD), pullup enable (PTAPE), and slew rate control (PTASE) registers. Refer to Section 6.4, “Parallel I/O Controls,” for more information about general-purpose I/O control.

Port A can be configured to be keyboard interrupt input pins. Refer to Chapter 9, “Keyboard Interrupt (KBI) Module,” for more information about using port A pins as keyboard interrupts pins.

6.3.2 Port B and Analog to Digital Converter Inputs

Port B		Bit 7	6	5	4	3	2	1	Bit 0
	MCU Pin:	PTB7/ AD1P7	PTB6/ AD1P6	PTB5/ AD1P5	PTB4/ AD1P4	PTB3/ AD1P3	PTB2/ AD1P2	PTB1/ AD1P1	PTB0/ AD1P0

Figure 6-3. Port B Pin Names

Port B is an 8-bit port shared among the ATD inputs and general-purpose I/O. Any pin enabled as an ATD input will be forced to act as an input.

Port B pins are available as general-purpose I/O pins controlled by the port B data (PTBD), data direction (PTBDD), pullup enable (PTBPE), and slew rate control (PTBSE) registers. Refer to Section 6.4, “Parallel I/O Controls,” for more information about general-purpose I/O control.

When the ATD module is enabled, analog pin enables are used to specify which pins on port B will be used as ATD inputs. Refer to Chapter 14, “Analog-to-Digital Converter (ATD) Module,” for more information about using port B pins as ATD pins.

6.3.3 Port C and SCI2, IIC, and High-Current Drivers

Port C		Bit 7	6	5	3	3	2	1	Bit 0
	MCU Pin:	PTC7	PTC6	PTC5	PTC4	PTC3/ SCL1	PTC2/ SDA1	PTC1/ RxD2	PTC0/ TxD2

Figure 6-4. Port C Pin Names

Port C is an 8-bit port which is shared among the SCI2 and IIC1 modules, and general-purpose I/O. When SCI2 or IIC1 modules are enabled, the pin direction will be controlled by the module or function. Port C has high current output drivers.

Port C pins are available as general-purpose I/O pins controlled by the port C data (PTCD), data direction (PTCDD), pullup enable (PTCPE), and slew rate control (PTCSE) registers. Refer to Section 6.4, “Parallel I/O Controls,” for more information about general-purpose I/O control.

When the SCI2 module is enabled, PTC0 serves as the SCI2 module’s transmit pin (TxD2) and PTC1 serves as the receive pin (RxD2). Refer to Chapter 11, “Serial Communications Interface (SCI) Module,” for more information about using PTC0 and PTC1 as SCI pins

When the IIC module is enabled, PTC2 serves as the IIC modules’s serial data input/output pin (SDA1) and PTC3 serves as the clock pin (SCL1). Refer to Chapter 13, “Inter-Integrated Circuit (IIC) Module,” for more information about using PTC2 and PTC3 as IIC pins.

6.3.4 Port D, TPM1 and TPM2

Port D		Bit 7	6	5	4	3	2	1	Bit 0
	MCU Pin:	PTD7/ TPM2CH4	PTD6/ TPM2CH3	PTD5/ TPM2CH2	PTD4/ TPM2CH1	PTD3/ TPM2CH0	PTD2/ TPM1CH2	PTD1/ TPM1CH1	PTD0/ TPM1CH0

Figure 6-5. Port D Pin Names

Port D is an 8-bit port shared with the two TPM modules, TPM1 and TPM2, and general-purpose I/O. When the TPM1 or TPM2 modules are enabled in output compare or input capture modes of operation, the pin direction will be controlled by the module function.

Port D pins are available as general-purpose I/O pins controlled by the port D data (PTDD), data direction (PTDDD), pullup enable (PTDPE), and slew rate control (PTDSE) registers. Refer to Section 6.4, “Parallel I/O Controls” for more information about general-purpose I/O control.

The TPM2 module can be configured to use PTD7–PTD3 as either input capture, output compare, PWM, or external clock input pins (PTD3 only). Refer to Chapter 10, “Timer/PWM (TPM) Module” for more information about using PTD7–PTD3 as timer pins.

The TPM1 module can be configured to use PTD2–PTD0 as either input capture, output compare, PWM, or external clock input pins (PTD0 only). Refer to Chapter 10, “Timer/PWM (TPM) Module” for more information about using PTD2–PTD0 as timer pins.

6.3.5 Port E, SCI1, and SPI

Port E		Bit 7	6	5	4	3	2	1	Bit 0
	MCU Pin:	PTE7	PTE6	PTE5/ SPSCK1	PTE4/ MOSI1	PTE3/ MISO1	PTE2/ $\overline{SS1}$	PTE1/ RxD1	PTE0/ TxD1

Figure 6-6. Port E Pin Names

Port E is an 8-bit port shared with the SCI1 module, SPI1 module, and general-purpose I/O. When the SCI or SPI modules are enabled, the pin direction will be controlled by the module function.

Port E pins are available as general-purpose I/O pins controlled by the port E data (PTED), data direction (PTEDD), pullup enable (PTEPE), and slew rate control (PTESE) registers. Refer to Section 6.4, “Parallel I/O Controls” for more information about general-purpose I/O control.

When the SCI1 module is enabled, PTE0 serves as the SCI1 module’s transmit pin (TxD1) and PTE1 serves as the receive pin (RxD1). Refer to Chapter 11, “Serial Communications Interface (SCI) Module” for more information about using PTE0 and PTE1 as SCI pins.

When the SPI module is enabled, PTE2 serves as the SPI module’s slave select pin ($\overline{SS1}$), PTE3 serves as the master-in slave-out pin (MISO1), PTE4 serves as the master-out slave-in pin (MOSI1), and PTE5 serves as the SPI clock pin (SPSCK1). Refer to Chapter 12, “Serial Peripheral Interface (SPI) Module for more information about using PTE5–PTE2 as SPI pins.

6.3.6 Port F and High-Current Drivers

Port F		Bit 7	6	5	4	3	2	1	Bit 0
	MCU Pin:	PTF7	PTF6	PTF5	PTF4	PTF3	PTF2	PTF1	PTF0

Figure 6-7. Port F Pin Names

Port F is an 8-bit port general-purpose I/O that is not shared with any peripheral module. Port F has high current output drivers.

Port F pins are available as general-purpose I/O pins controlled by the port F data (PTFD), data direction (PTFDD), pullup enable (PTFPE), and slew rate control (PTFSE) registers. Refer to Section 6.4, “Parallel I/O Controls” for more information about general-purpose I/O control.

6.3.7 Port G, BKGD/MS, and Oscillator

Port G		Bit 7	6	5	4	3	2	1	Bit 0
	MCU Pin:	PTG7	PTG6	PTG5	PTG4	PTG3	PTG2/ EXTAL	PTG1/ XTAL	PTG0/ BKGD/MS

Figure 6-8. Port G Pin Names

Port G is an 8-bit port which is shared among the background/mode select function, oscillator, and general-purpose I/O. When the background/mode select function or oscillator is enabled, the pin direction will be controlled by the module function.

Port G pins are available as general-purpose I/O pins controlled by the port G data (PTGD), data direction (PTGDD), pullup enable (PTGPE), and slew rate control (PTGSE) registers. Refer to Section 6.4, “Parallel I/O Controls” for more information about general-purpose I/O control.

The internal pullup for PTG0 is enabled when the background/mode select function is enabled, regardless of the state of PTGPE0. During reset, the BKGD/MS pin functions as a mode select pin. After the MCU is out of reset, the BKGD/MS pin becomes the background communications input/output pin. The PTG0 can be configured to be a general-purpose output pin. Refer to Chapter 3, “Modes of Operation”, Chapter 5, “Resets, Interrupts, and System Configuration”, and Chapter 15, “Development Support” for more information about using this pin.

The ICG module can be configured to use PTG2–PTG1 ports as crystal oscillator or external clock pins.

Refer to Chapter 13, “Inter-Integrated Circuit (IIC) Module” for more information about using these pins as oscillator pins.

6.4 Parallel I/O Controls

Provided no on-chip peripheral is controlling a port pin, the pins operate as general-purpose I/O pins that are accessed and controlled by a data register (PTxD), a data direction register (PTxDD), a pullup enable register (PTxPE), and a slew rate control register (PTxSE) where x is A, B, C, D, E, F, or G.

Reads of the data register return the pin value (if PTxDDn = 0) or the contents of the port data register (if PTxDDn = 1). Writes to the port data register are latched into the port register whether the pin is controlled by an on-chip peripheral or the pin is configured as an input. If the corresponding pin is not controlled by a peripheral and is configured as an output, this level will be driven out the port pin.

6.4.1 Data Direction Control

The data direction control bits determine whether the pin output driver is enabled, and they control what is read for port data register reads. Each port pin has a data direction control bit. When PTxDDn = 0, the corresponding pin is an input and reads of PTxD return the pin value. When PTxDDn = 1, the corresponding pin is an output and reads of PTxD return the last value written to the port data register. When a peripheral module or system function is in control of a port pin, the data direction control still controls what is returned for reads of the port data register, even though the peripheral system has overriding control of the actual pin direction.

For the MC9S08GB/GT MCU, reads of PTG0/BKGD/MS will return the value on the output 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.

6.4.2 Internal Pullup Control

An internal pullup device can be enabled for each port pin that is configured as an input (PTxDDn = 0). The pullup device is available for a peripheral module to use, provided the peripheral is enabled and is an input function as long as the PTxDDn = 0.

For the four configurable KBI module inputs on PTA7–PTA4, when a pin is configured to detect rising edges, the port pullup enable associated with the pin (PTAPEn) selects a pulldown rather than a pullup device.

6.4.3 Slew Rate Control

Slew rate control can be enabled for each port pin that is configured as an output (PTxDDn = 1) or if a peripheral module is enabled and its function is an output. Not all peripheral modules' outputs have slew rate control; refer to Chapter 2, "Pins and Connections" for more information about which pins have slew rate control.

6.5 Stop Modes

Depending on the stop mode, I/O functions differently as the result of executing a STOP instruction. An explanation of I/O behavior for the various stop modes follows:

- When the MCU enters stop1 mode, all internal registers including general-purpose I/O control and data registers are powered down. All of the general-purpose I/O pins assume their reset state: output buffers and pullups turned off. Upon exit from stop1, all I/O must be initialized as if the MCU had been reset.
- When the MCU enters stop2 mode, the internal registers are powered down as in stop1 but the I/O pin states are latched and held. For example, a port pin that is an output driving low continues to function as an output driving low even though its associated data direction and output data registers are powered down internally. Upon exit from stop2, the pins continue to hold their states until a 1 is written to the PPDACK bit. To avoid discontinuity in the pin state following exit from stop2, the user must restore the port control and data registers to the values they held before entering stop2. These values can be stored in RAM before entering stop2 because the RAM is maintained during stop2.
- In stop3 mode, all I/O is maintained because internal logic circuity stays powered up. Upon recovery, normal I/O function is available to the user.

6.6 Parallel I/O Registers and Control Bits

This section provides information about all registers and control bits associated with the parallel I/O ports.

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.

6.6.1 Port A Registers (PTAD, PTAPE, PTASE, and PTADD)

Port A includes eight pins shared between general-purpose I/O and the KBI module. Port A pins used as general-purpose I/O pins are controlled by the port A data (PTAD), data direction (PTADD), pullup enable (PTAPE), and slew rate control (PTASE) registers.

If the KBI takes control of a port A pin, the corresponding PTASE bit is ignored since the pin functions as an input. As long as PTADD is 0, the PTAPE controls the pullup enable for the KBI function. Reads of PTAD will return the logic value of the corresponding pin, provided PTADD is 0.

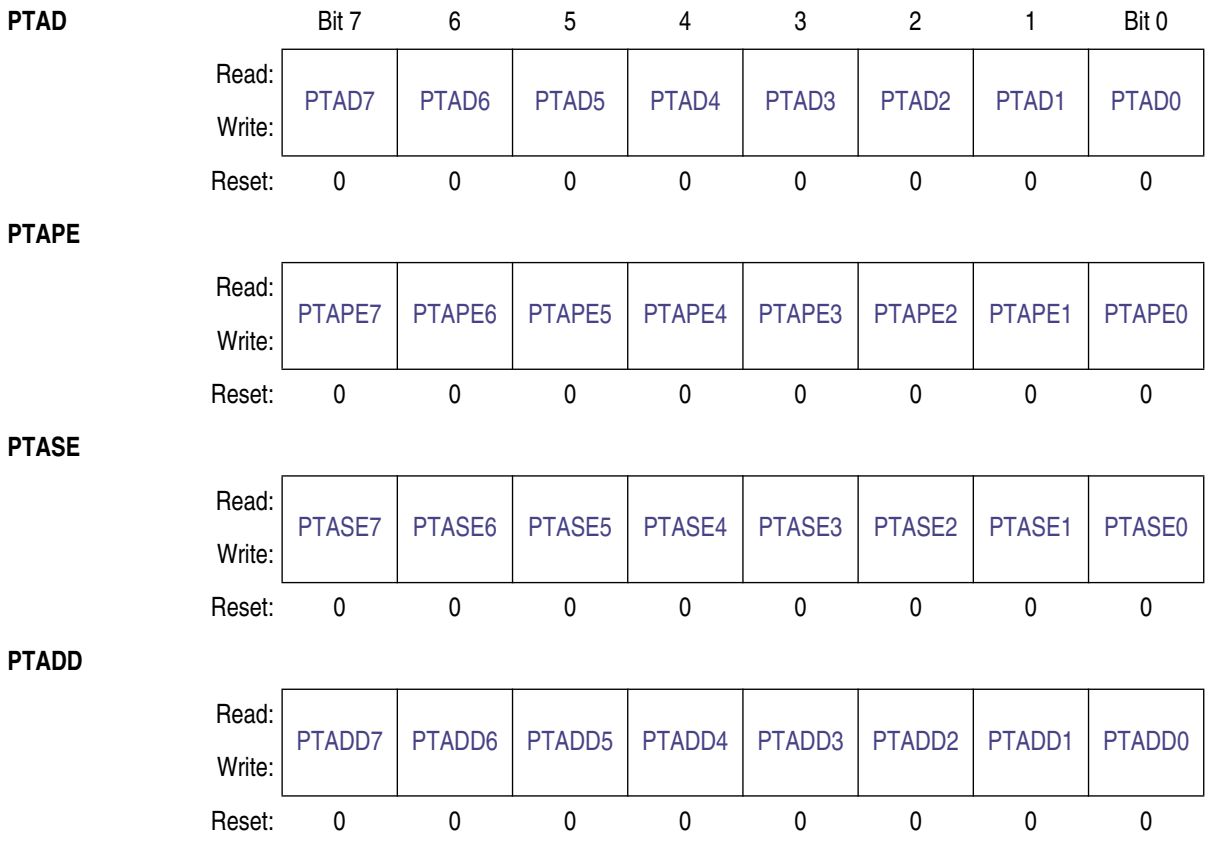


Figure 6-9. Port A Registers

PTADn — Port A Data Register Bit n (n = 0–7)

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.

PTAPEn — Pullup Enable for Port A Bit n (n = 0–7)

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 4 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.

1 = Internal pullup device enabled.

0 = Internal pullup device disabled.

PTASEn — Slew Rate Control Enable for Port A Bit n (n = 0–7)

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.

1 = Slew rate control enabled.

0 = Slew rate control disabled.

PTADDn — Data Direction for Port A Bit n (n = 0–7)

These read/write bits control the direction of port A pins and what is read for PTAD reads.

1 = Output driver enabled for port A bit n and PTAD reads return the contents of PTADn.

0 = Input (output driver disabled) and reads return the pin value.

6.6.2 Port B Registers (PTBD, PTBPE, PTBSE, and PTBDD)

Port B includes eight general-purpose I/O pins that share with the ATD function. Port B pins used as general-purpose I/O pins are controlled by the port B data (PTBD), data direction (PTBDD), pullup enable (PTBPE), and slew rate control (PTBSE) registers.

If the ATD takes control of a port B pin, the corresponding PTBDD, PTBSE, and PTBPE bits are ignored. When a port B pin is being used as an ATD pin, reads of PTBD will return a 0 of the corresponding pin, provided PTBDD is 0.

PTBD

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTBD7	PTBD6	PTBD5	PTBD4	PTBD3	PTBD2	PTBD1	PTBD0
Reset:	0	0	0	0	0	0	0	0

PTBPE

Read: Write:	PTBPE7	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
Reset:	0	0	0	0	0	0	0	0

PTBSE

Read: Write:	PTBSE7	PTBSE6	PTBSE5	PTBSE4	PTBSE3	PTBSE2	PTBSE1	PTBSE0
Reset:	0	0	0	0	0	0	0	0

PTBDD

Read: Write:	PTBDD7	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2	PTBDD1	PTBDD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-10. Port B Registers

PTBDn — Port B Data Register Bit n (n = 0–7)

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 on the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups disabled.

PTBPEn — Pullup Enable for Port B Bit n (n = 0–7)

For port B pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled. For port B pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled.

1 = Internal pullup device enabled.
0 = Internal pullup device disabled.

PTBSEn — Slew Rate Control Enable for Port B Bit n (n = 0–7)

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.

1 = Slew rate control enabled.
0 = Slew rate control disabled.

PTBDDn — Data Direction for Port B Bit n (n = 0–7)

These read/write bits control the direction of port B pins and what is read for PTBD reads.

1 = Output driver enabled for port B bit n and PTBD reads return the contents of PTBDn.
0 = Input (output driver disabled) and reads return the pin value.

6.6.3 Port C Registers (PTCD, PTCPE, PTCSE, and PTCDD)

Port C includes eight general-purpose I/O pins that share with the SCI2 and IIC modules. Port C pins used as general-purpose I/O pins are controlled by the port C data (PTCD), data direction (PTCDD), pullup enable (PTCPE), and slew rate control (PTCSE) registers.

If the SCI2 takes control of a port C pin, the corresponding PTCDD bit is ignored. PTCSE can be used to provide slew rate on the SCI2 transmit pin, TxD2. PTCPE can be used, provided the corresponding PTCDD bit is 0, to provide a pullup device on the SCI2 receive pin, RxD2.

If the IIC takes control of a port C pin, the corresponding PTCDD bit is ignored. PTCSE can be used to provide slew rate on the IIC serial data pin (SDA1), when in output mode and the IIC clock pin (SCL1). PTCPE can be used, provided the corresponding PTCDD bit is 0, to provide a pullup device on the IIC serial data pin, when in receive mode.

Reads of PTCD will return the logic value of the corresponding pin, provided PTCDD is 0.

		Bit 7	6	5	4	3	2	1	Bit 0
PTCD	Read: Write:	PTCD7	PTCD6	PTCD5	PTCD4	PTCD3	PTCD2	PTCD1	PTCD0
	Reset:	0	0	0	0	0	0	0	0
PTCPE	Read: Write:	PTCPE7	PTCPE6	PTCPE5	PTCPE4	PTCPE3	PTCPE2	PTCPE1	PTCPE0
	Reset:	0	0	0	0	0	0	0	0
PTCSE	Read: Write:	PTCSE7	PTCSE6	PTCSE5	PTCSE4	PTCSE3	PTCSE2	PTCSE1	PTCSE0
	Reset:	0	0	0	0	0	0	0	0
PTCDD	Read: Write:	PTCDD7	PTCDD6	PTCDD5	PTCDD4	PTCDD3	PTCDD2	PTCDD1	PTCDD0
	Reset:	0	0	0	0	0	0	0	0

Figure 6-11. Port C Registers

PTCDn — Port C Data Register Bit n (n = 0–7)

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.

PTCPEn — Pullup Enable for Port C Bit n (n = 0–7)

For port C pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled. For port C pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled.

1 = Internal pullup device enabled.
0 = Internal pullup device disabled.

PTCSEn — Slew Rate Control Enable for Port C Bit n (n = 0–7)

For port C 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.

1 = Slew rate control enabled.
0 = Slew rate control disabled.

PTCDDn — Data Direction for Port C Bit n (n = 0–7)

These read/write bits control the direction of port C pins and what is read for PTCD reads.
1 = Output driver enabled for port C bit n and PTCD reads return the contents of PTCDn.
0 = Input (output driver disabled) and reads return the pin value.

6.6.4 Port D Registers (PTDD, PTDPE, PTDSE, and PTDDD)

Port D includes eight pins shared between general-purpose I/O, TPM1, and TPM2. Port D pins used as general-purpose I/O pins are controlled by the port D data (PTDD), data direction (PTDDD), pullup enable (PTDPE), and slew rate control (PTDSE) registers.

If a TPM takes control of a port D pin, the corresponding PTDDD bit is ignored. When the TPM is in output compare mode, the corresponding PTDSE can be used to provide slew rate on the pin. When the TPM is in input capture mode, the corresponding PTDPE can be used, provided the corresponding PTDDD bit is 0, to provide a pullup device on the pin.

Reads of PTDD will return the logic value of the corresponding pin, provided PTDDD is 0.

PTDD

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTDD7	PTDD6	PTDD5	PTDD4	PTDD3	PTDD2	PTDD1	PTDD0
Reset:	0	0	0	0	0	0	0	0

PTDPE

Read: Write:	PTDPE7	PTDPE6	PTDPE5	PTDPE4	PTDPE3	PTDPE2	PTDPE1	PTDPE0
Reset:	0	0	0	0	0	0	0	0

PTDSE

Read: Write:	PTDSE7	PTDSE6	PTDSE5	PTDSE4	PTDSE3	PTDSE2	PTDSE1	PTDSE0
Reset:	0	0	0	0	0	0	0	0

PTDDD

Read: Write:	PTDDD7	PTDDD6	PTDDD5	PTDDD4	PTDDD3	PTDDD2	PTDDD1	PTDDD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-12. Port D Registers

PTDDn — Port D Data Register Bit n (n = 0–7)

For port D pins that are inputs, reads return the logic level on the pin. For port D pins that are configured as outputs, reads return the last value written to this register.

Writes are latched into all bits of this register. For port D pins that are configured as outputs, the logic level is driven out the corresponding MCU pin.

Reset forces PTDD to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups disabled.

PTDPEn — Pullup Enable for Port D Bit n (n = 0–7)

For port D pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled. For port D pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled.

1 = Internal pullup device enabled.
0 = Internal pullup device disabled.

PTDSEn — Slew Rate Control Enable for Port D Bit n (n = 0–7)

For port D pins that are outputs, these read/write control bits determine whether the slew rate controlled outputs are enabled. For port D pins that are configured as inputs, these bits are ignored.

1 = Slew rate control enabled.
0 = Slew rate control disabled.

PTDDDn — Data Direction for Port D Bit n (n = 0–7)

These read/write bits control the direction of port D pins and what is read for PTDD reads.

1 = Output driver enabled for port D bit n and PTDD reads return the contents of PTDDn.
0 = Input (output driver disabled) and reads return the pin value.

6.6.5 Port E Registers (PTED, PTEPE, PTESE, and PTEDD)

Port E includes eight general-purpose I/O pins that share with the SCI1 and SPI modules. Port E pins used as general-purpose I/O pins are controlled by the port E data (PTED), data direction (PTEDD), pullup enable (PTEPE), and slew rate control (PTESE) registers.

If the SCI1 takes control of a port E pin, the corresponding PTEDD bit is ignored. PTESE can be used to provide slew rate on the SCI1 transmit pin, TxD1. PTEPE can be used, provided the corresponding PTEDD bit is 0, to provide a pullup device on the SCI1 receive pin, RxD1.

If the SPI takes control of a port E pin, the corresponding PTEDD bit is ignored. PTESE can be used to provide slew rate on the SPI serial output pin (MOSI1 or MISO1) and serial clock pin (SPSCK1) depending on the SPI operational mode. PTEPE can be used, provided the corresponding PTEDD bit is 0, to provide a pullup device on the SPI serial input pins (MOSI1 or MISO1) and slave select pin ($\overline{SS1}$) depending on the SPI operational mode.

Reads of PTED will return the logic value of the corresponding pin, provided PTEDD is 0.

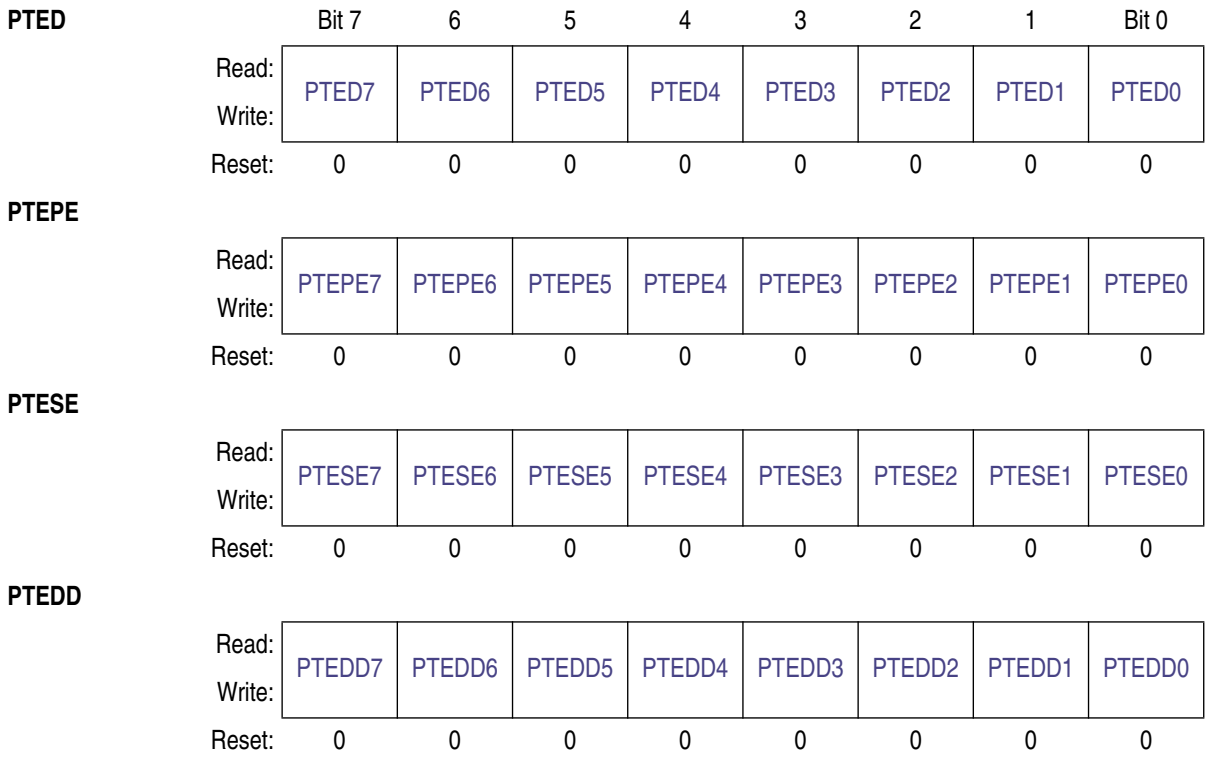


Figure 6-13. Port E Registers

PTEDn — Port E Data Register Bit n (n = 0–7)

For port E pins that are inputs, reads return the logic level on the pin. For port E pins that are configured as outputs, reads return the last value written to this register.

Writes are latched into all bits in this register. For port E pins that are configured as outputs, the logic level is driven out the corresponding MCU pin.

Reset forces PTED to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups disabled.

PTEPEn — Pullup Enable for Port E Bit n (n = 0–7)

For port E pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled. For port E pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled.

1 = Internal pullup device enabled.
0 = Internal pullup device disabled.

PTESEn — Slew Rate Control Enable for Port E Bit n (n = 0–7)

For port E pins that are outputs, these read/write control bits determine whether the slew rate controlled outputs are enabled. For port E pins that are configured as inputs, these bits are ignored.

1 = Slew rate control enabled.
0 = Slew rate control disabled.

PTEDDn — Data Direction for Port E Bit n (n = 0–7)

These read/write bits control the direction of port E pins and what is read for PTED reads.
1 = Output driver enabled for port E bit n and PTED reads return the contents of PTEDn.
0 = Input (output driver disabled) and reads return the pin value.

6.6.6 Port F Registers (PTFD, PTFPE, PTFSE, and PTFDD)

Port F includes eight general-purpose I/O pins that are not shared with any peripheral module. Port F pins used as general-purpose I/O pins are controlled by the port F data (PTFD), data direction (PTFDD), pullup enable (PTFPE), and slew rate control (PTFSE) registers.

PTFD

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTFD7	PTFD6	PTFD5	PTFD4	PTFD3	PTFD2	PTFD1	PTFD0
Reset:	0	0	0	0	0	0	0	0

PTFPE

Read: Write:	PTFPE7	PTFPE6	PTFPE5	PTFPE4	PTFPE3	PTFPE2	PTFPE1	PTFPE0
Reset:	0	0	0	0	0	0	0	0

PTFSE

Read: Write:	PTFSE7	PTFSE6	PTFSE5	PTFSE4	PTFSE3	PTFSE2	PTFSE1	PTFSE0
Reset:	0	0	0	0	0	0	0	0

PTFDD

Read: Write:	PTFDD7	PTFDD6	PTFDD5	PTFDD4	PTFDD3	PTFDD2	PTFDD1	PTFDD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-14. Port F Registers

PTFDn — Port PTF Data Register Bit n (n = 0–7)

For port F pins that are inputs, reads return the logic level on the pin. For port F 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 F pins that are configured as outputs, the logic level is driven out the corresponding MCU pin.

Reset forces PTFD 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.

PTFPEn — Pullup Enable for Port F Bit n (n = 0–7)

For port F pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled. For port F pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled.

1 = Internal pullup device enabled.
0 = Internal pullup device disabled.

PTFSEn — Slew Rate Control Enable for Port F Bit n (n = 0–7)

For port F pins that are outputs, these read/write control bits determine whether the slew rate controlled outputs are enabled. For port F pins that are configured as inputs, these bits are ignored.

1 = Slew rate control enabled.
0 = Slew rate control disabled.

PTFDDn — Data Direction for Port F Bit n (n = 0–7)

These read/write bits control the direction of port F pins and what is read for PTFD reads.

1 = Output driver enabled for port F bit n and PTFD reads return the contents of PTFDn.
0 = Input (output driver disabled) and reads return the pin value.

6.6.7 Port G Registers (PTGD, PTGPE, PTGSE, and PTGDD)

Port G includes eight general-purpose I/O pins that are shared with BKGD/MS function and the oscillator or external clock pins. Port G pins used as general-purpose I/O pins are controlled by the port G data (PTGD), data direction (PTGDD), pullup enable (PTGPE), and slew rate control (PTGSE) registers.

Port pin PTG0, while in reset, defaults to the BKGD/MS pin. After the MCU is out of reset, PTG0 can be configured to be a general-purpose output pin. When BKGD/MS takes control of PTG0, the corresponding PTGDD, PTGPE, and PTGPSE bits are ignored.

Port pins PTG1 and PTG2 can be configured to be oscillator or external clock pins. When the oscillator takes control of a port G pin, the corresponding PTGD, PTGDD, PTGSE, and PTGPE bits are ignored.

Reads of PTGD will return the logic value of the corresponding pin, provided PTGDD is 0.

		Bit 7	6	5	4	3	2	1	Bit 0
PTGD	Read: Write:	PTGD7	PTGD6	PTGD5	PTGD4	PTGD3	PTGD2	PTGD1	PTGD0
	Reset:	0	0	0	0	0	0	0	0
PTGPE	Read: Write:	PTGPE7	PTGPE6	PTGPE5	PTGPE4	PTGPE3	PTGPE2	PTGPE1	PTGPE0
	Reset:	0	0	0	0	0	0	0	0
PTGSE	Read: Write:	PTGSE7	PTGSE6	PTGSE5	PTGSE4	PTGSE3	PTGSE2	PTGSE1	PTGSE0
	Reset:	0	0	0	0	0	0	0	0
PTGDD	Read: Write:	PTGDD7	PTGDD6	PTGDD5	PTGDD4	PTGDD3	PTGDD2	PTGDD1	PTGDD0
	Reset:	0	0	0	0	0	0	0	0

Figure 6-15. Port G Registers

PTGDn — Port PTG Data Register Bit n (n = 0–7)

For port G pins that are inputs, reads return the logic level on the pin. For port G 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 G pins that are configured as outputs, the logic level is driven out the corresponding MCU pin.

Reset forces PTGD 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.

PTGPEn — Pullup Enable for Port G Bit n (n = 0–7)

For port G pins that are inputs, these read/write control bits determine whether internal pullup devices are enabled. For port G pins that are configured as outputs, these bits are ignored and the internal pullup devices are disabled.

1 = Internal pullup device enabled.
0 = Internal pullup device disabled.

PTGSEn — Slew Rate Control Enable for Port G Bit n (n = 0–7)

For port G pins that are outputs, these read/write control bits determine whether the slew rate controlled outputs are enabled. For port G pins that are configured as inputs, these bits are ignored.

1 = Slew rate control enabled.
0 = Slew rate control disabled.

PTGDDn — Data Direction for Port G Bit n (n = 0–7)

These read/write bits control the direction of port G pins and what is read for PTGD reads.

1 = Output driver enabled for port G bit n and PTGD reads return the contents of PTGDn.

0 = Input (output driver disabled) and reads return the pin value.

Chapter 7 Internal Clock Generator (ICG) Module

The MC9S08GB/GT microcontroller provides one internal clock generation (ICG) module to create the system bus frequency. All functions described in this section are available on the MC9S08GB/GT microcontroller. The EXTAL and XTAL pins share port G bits 2 and 1, respectively. Analog supply lines V_{DDA} and V_{SSA} are internally derived from the MCU's V_{DD} and V_{SS} pins. Electrical parametric data for the ICG may be found in Appendix A, "Electrical Characteristics."

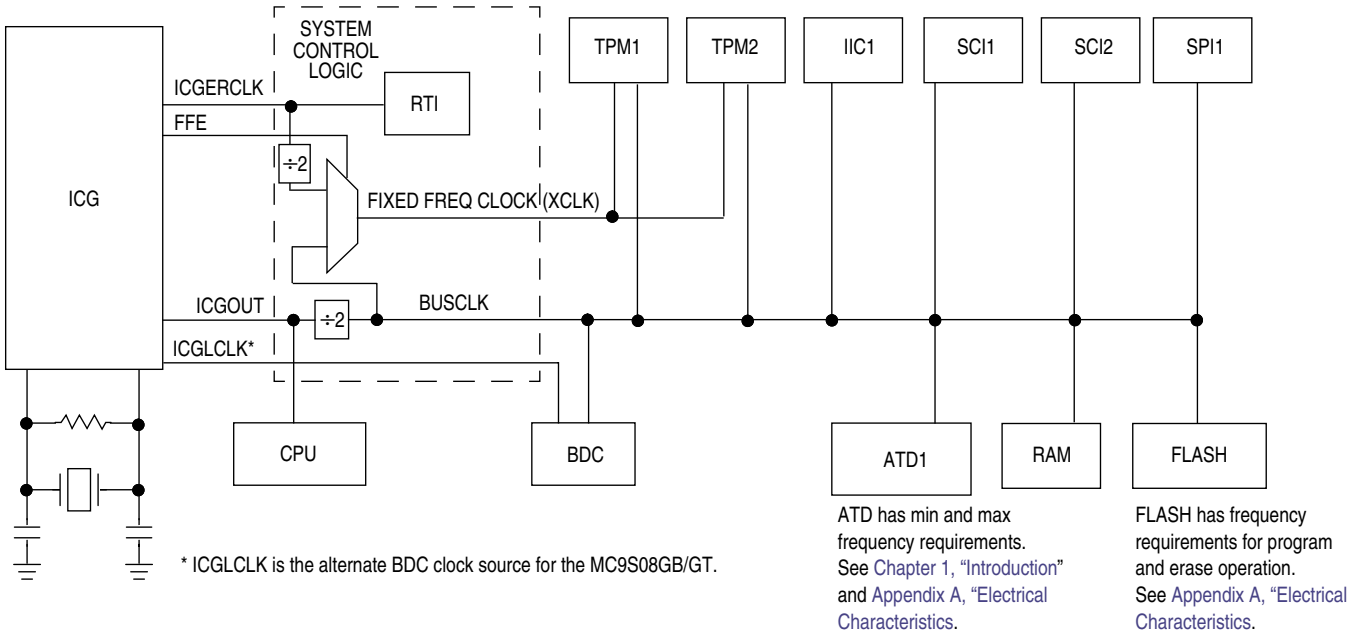
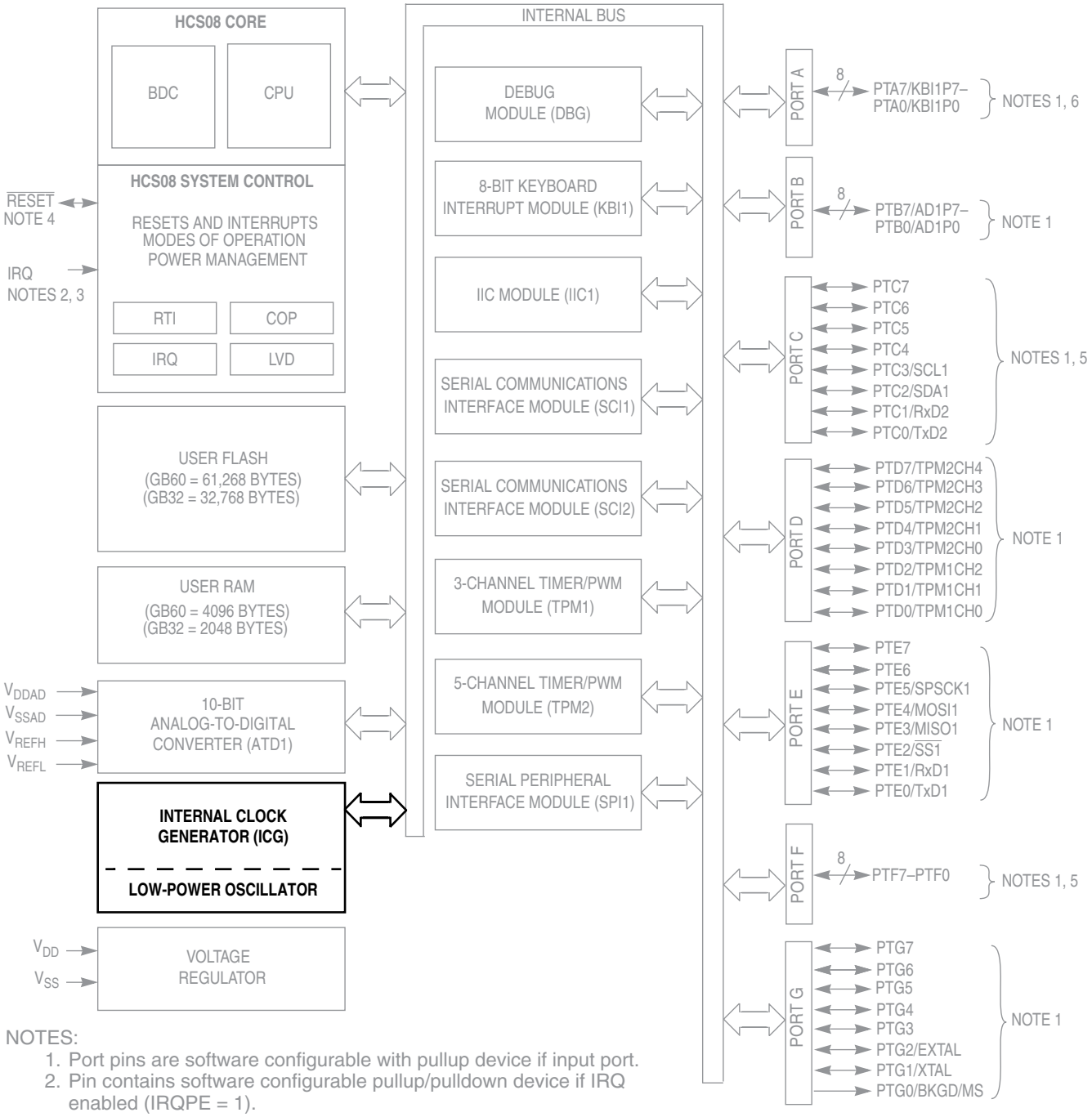


Figure 7-1. System Clock Distribution Diagram

NOTE

Freescale Semiconductor recommends that FLASH location $FFBE be reserved to store a nonvolatile version of ICGTRM. This will allow debugger and programmer vendors to perform a manual trim operation and store the resultant ICGTRM value for users to access at a later time.

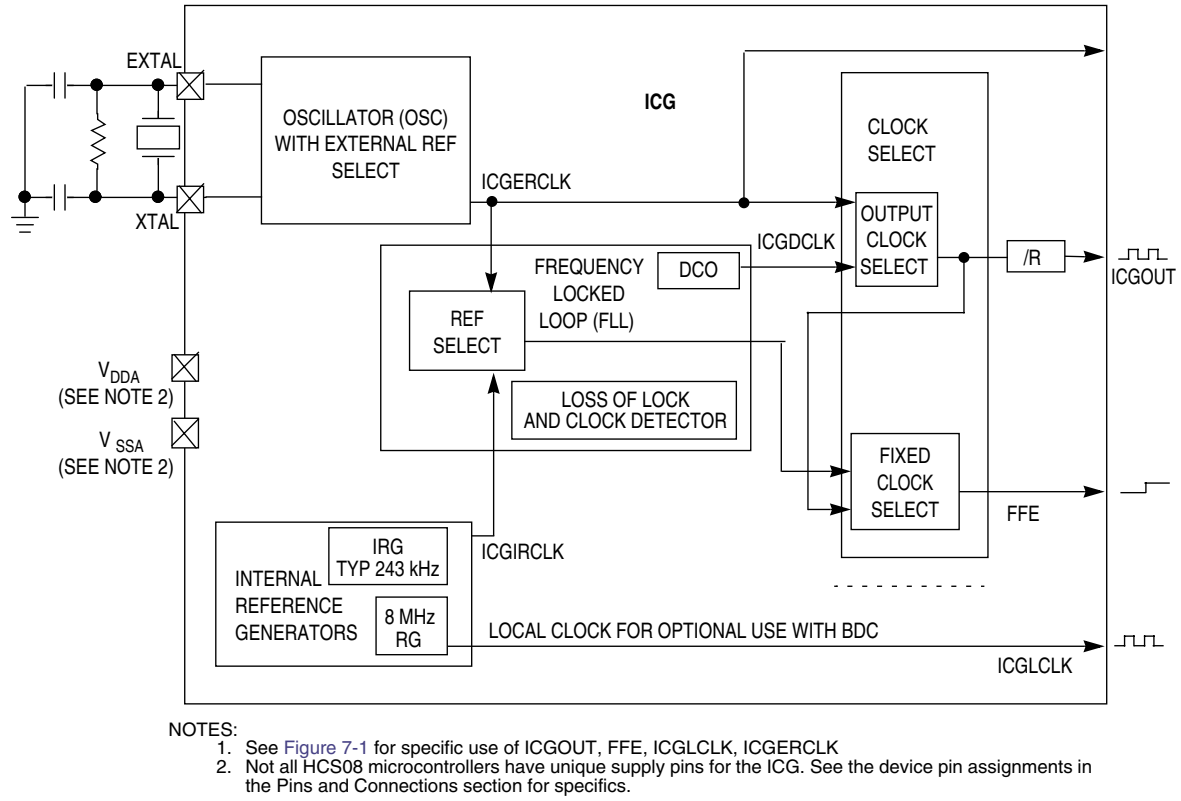


NOTES:
1. Port pins are software configurable with pullup device if input port.
2. Pin contains software configurable pullup/pulldown device if IRQ enabled (IRQPE = 1).
3. IRQ does not have a clamp diode to V_{DD}. IRQ should not be driven above V_{DD}.
4. Pin contains integrated pullup device.
5. High current drive
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).

Figure 7-2. Block Diagram Highlighting ICG Module

7.1 Introduction

Figure 7-3 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 Figure 7-1 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 in the Pins and Connections section for specifics.

Figure 7-3. ICG Block Diagram

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 7-3, 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.
- **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).

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 7.4, "Initialization/Application Information," and pick an example that best suits the application needs.

7.1.1 Features

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

7.1.2 Modes of Operation

This is a high-level description only. Detailed descriptions of operating modes are contained in Section 7.3, “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 out of 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 which 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 which 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 which 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.

7.2 External Signal Description

7.2.1 Overview

Table 7-1 shows the user-accessible signals available for the ICG.

Table 7-1. Signal Properties

Name	Function	Reset State
EXTAL	External clock/oscillator input	Analog input
XTAL	Oscillator output	Analog output

7.2.2 Detailed Signal Descriptions

This section describes each pin signal in detail.

7.2.2.1 EXTAL— External Reference Clock / Oscillator Input

If the first write to the ICG control register 1 selected FLL engaged external or FLL bypassed modes, this signal is the analog external/reference clock or the input of the oscillator circuit. If the first write to the ICG control register 1 selected FLL engaged internal or self-clocked modes, this signal has no effect on the ICG.

7.2.2.2 XTAL— Oscillator Output

If the first write to the ICG control register 1 selected FLL engaged external or FLL bypassed modes using a crystal/resonator reference, this signal is the analog output of the oscillator amplifier circuit. In all other cases, this signal has no effect on the ICG.

7.2.3 External Clock Connections

If an external clock is used, then the pins are connected as shown below.

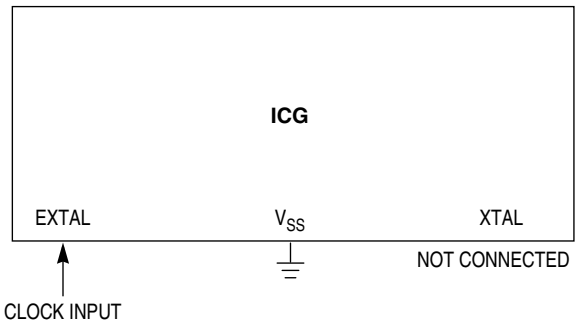


Figure 7-4. External Clock Connections

7.2.4 External Crystal/Resonator Connections

If an external crystal/resonator frequency reference is used, then the pins are connected as shown below. Recommended component values are listed in Appendix A, “Electrical Characteristics.”

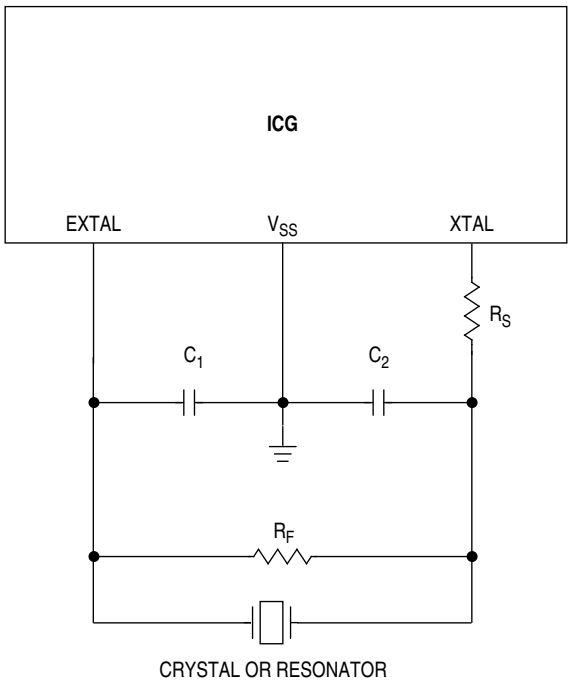


Figure 7-5. External Frequency Reference Connection

7.3 Functional Description

This section provides a functional description of each of the five operating modes of the ICG. Also covered 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.

7.3.1 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.

7.3.1.1 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.

7.3.1.2 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.

7.3.1.3 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.

7.3.2 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. Once ICGDCLK is considered stable, the ICG automatically closes the loop by switching to FLL engaged (internal or external) as selected by the CLKS bits.

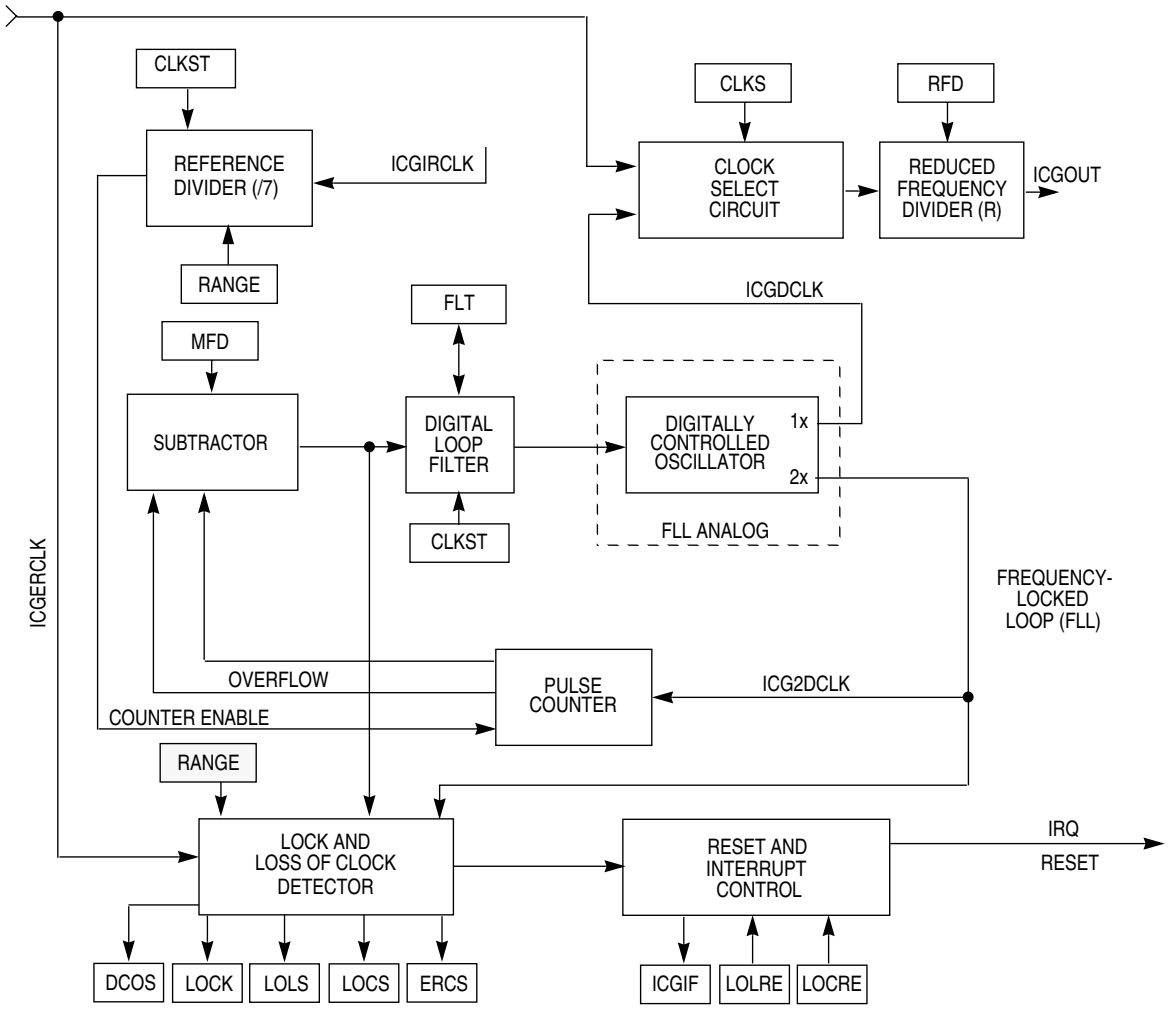


Figure 7-6. Detailed Frequency-Locked Loop Block Diagram

7.3.3 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.

7.3.3.1 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.

7.3.3.2 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 only updated once every four comparison cycles. The update made is an average of the error measurements taken in the four previous comparisons.

7.3.4 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 is still 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.

7.3.5 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 7-7 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.

7.3.5.1 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. Once 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.

7.3.5.2 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 only updated once every four comparison cycles. The update made is an average of the error measurements taken in the four previous comparisons.

7.3.6 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 7-3 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). Once 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.

7.3.7 FLL Loss-of-Clock Detection

The reference clock and the DCO clock are monitored under different conditions (see Table 7-2). 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 cleared by software 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.

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 7-2. Clock Monitoring

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.

7.3.8 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 7-3 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 7-3. ICG State Table

Actual Mode (CLKST)	Desired Mode (CLKS)	Range	Reference Frequency ($f_{REFERENCE}$)	Comparison Cycle Time	ICGOUT	Conditions[1] for CLKS = CLKST	Reason CLKS ≠ 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.

7.3.9 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 FBE mode. 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 7-5), N and R are determined by MFD and RFD, respectively (see Table 7-6).
- 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.

7.4 Initialization/Application Information

7.4.1 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 7-4. 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 sake.

Table 7-5. 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 out of 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 7-6. MFD and RFD Decode Table

MFD Value	Multiplication Factor (N)
000	4
001	6
010	8
011	10
100	12
101	14
110	16
111	18

RFD	Division Factor (R)
000	÷1
001	÷2
010	÷4
011	÷8
100	÷16
101	÷32
110	÷64
111	÷128

Register	Bit 7	6	5	4	3	2	1	Bit 0
ICGC1	0	RANGE	REFS	CLKS		OSCSTEN	0(1)	0
ICGC2	LOLRE	MFD			LOCRE	RFD		
ICGS1	CLKST		REFST	LOLS	LOCK	LOCS	ERCS	ICGIF
ICGS2	0	0	0	0	0	0	0	DCOS
ICGFLTU	0	0	0	0	FLT			
ICGFLTL	FLT							
ICGTRM	TRIM							

= Unimplemented or Reserved

1. This bit is reserved for Freescale Semiconductor internal use only. Any write operations to this register should write a 0 to this bit.

Figure 7-7. ICG Register Set

7.4.2 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

$$f_{ICGOUT} = f_{ext} * P * N / R ; P = 64, f_{ext} = 32 \text{ kHz} \quad \textit{Eqn. 7-1}$$

Solving for N / R gives:

$$N / R = 8.38 \text{ MHz} /(32 \text{ kHz} * 64) = 4 ; \text{we can choose N = 4 and R =1} \quad \textit{Eqn. 7-2}$$

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

ICGC1 = $38 (%00111000)

Bit 7		0	Unimplemented or reserved, always reads zero
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 in stop modes
Bit 1		0	Reserved for Freescale Semiconductor internal use; always write zero
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 7-8 shows flow charts for three conditions requiring ICG initialization.

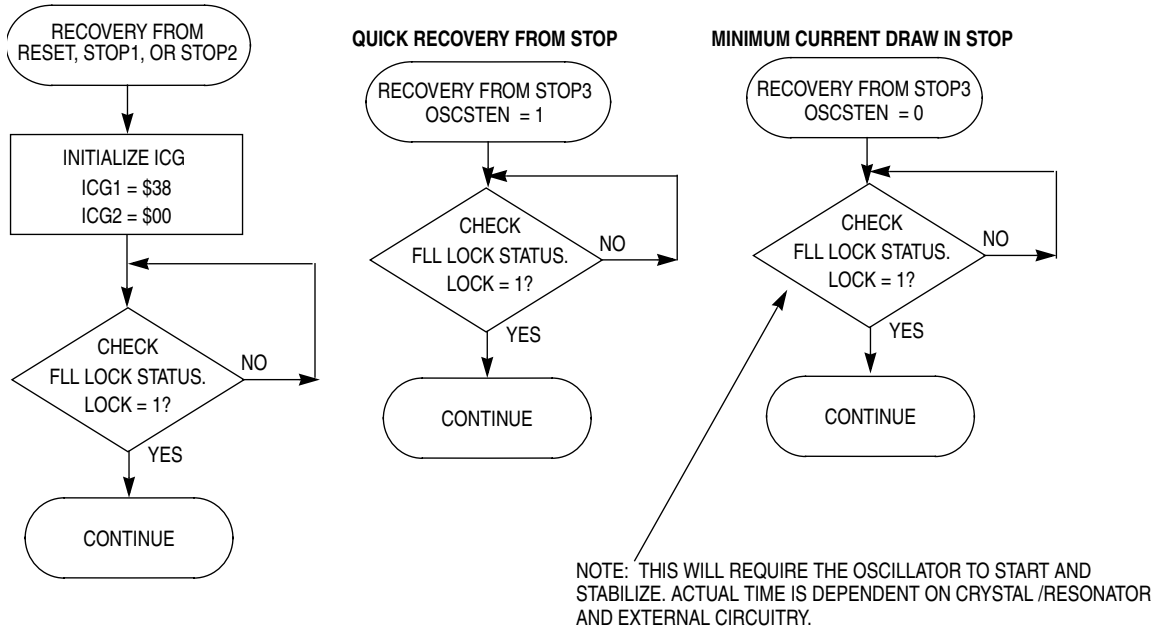


Figure 7-8. ICG Initialization for FEE in Example #1

7.4.3 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

$$f_{ICGOUT} = f_{ext} * P * N / R \; ; \; P = 1, f_{ext} = 4.00 \text{ MHz} \qquad \textit{Eqn. 7-3}$$

Solving for N / R gives:

$$N / R = 40 \text{ MHz} / (4 \text{ MHz} * 1) = 10 \; ; \text{ We can choose N = 10 and R = 1} \qquad \textit{Eqn. 7-4}$$

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

ICGC1 = $78 (%01111000)

Bit 7		0	Unimplemented or reserved, always reads zero
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 in stop modes
Bit 1		0	Reserved for Freescale Semiconductor internal use; always write zero
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

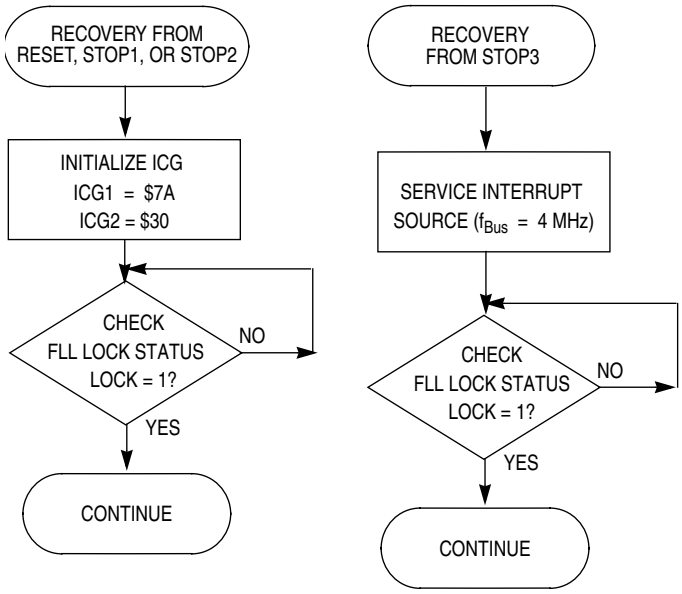


Figure 7-9. ICG Initialization and Stop Recovery for Example #2

7.4.4 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

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

Solving for N / R gives:

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

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		0	Unimplemented or reserved, always reads zero
Bit 6	RANGE	0	Configures oscillator for low-frequency range; (don't care)
Bit 5	REFS	1	Oscillator using crystal or resonator requested (don't care)
Bits 4:3	CLKS	01	FLL engaged, internal reference clock mode
Bit 2	OSCSTEN	0	Disables the oscillator in stop modes
Bit 1		0	Reserved for Freescale Semiconductor internal use; always write zero
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)

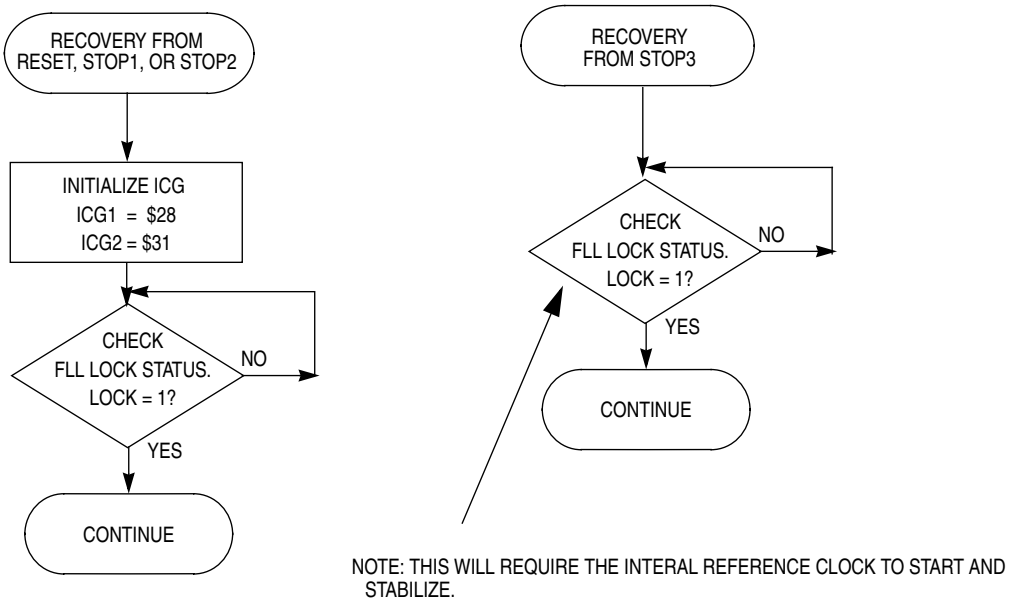


Figure 7-10. ICG Initialization and Stop Recovery for Example #3

7.4.5 Example #4: Internal Clock Generator Trim

The internally generated clock source is guaranteed to have a period ± 25% of the nominal value. In some case 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.

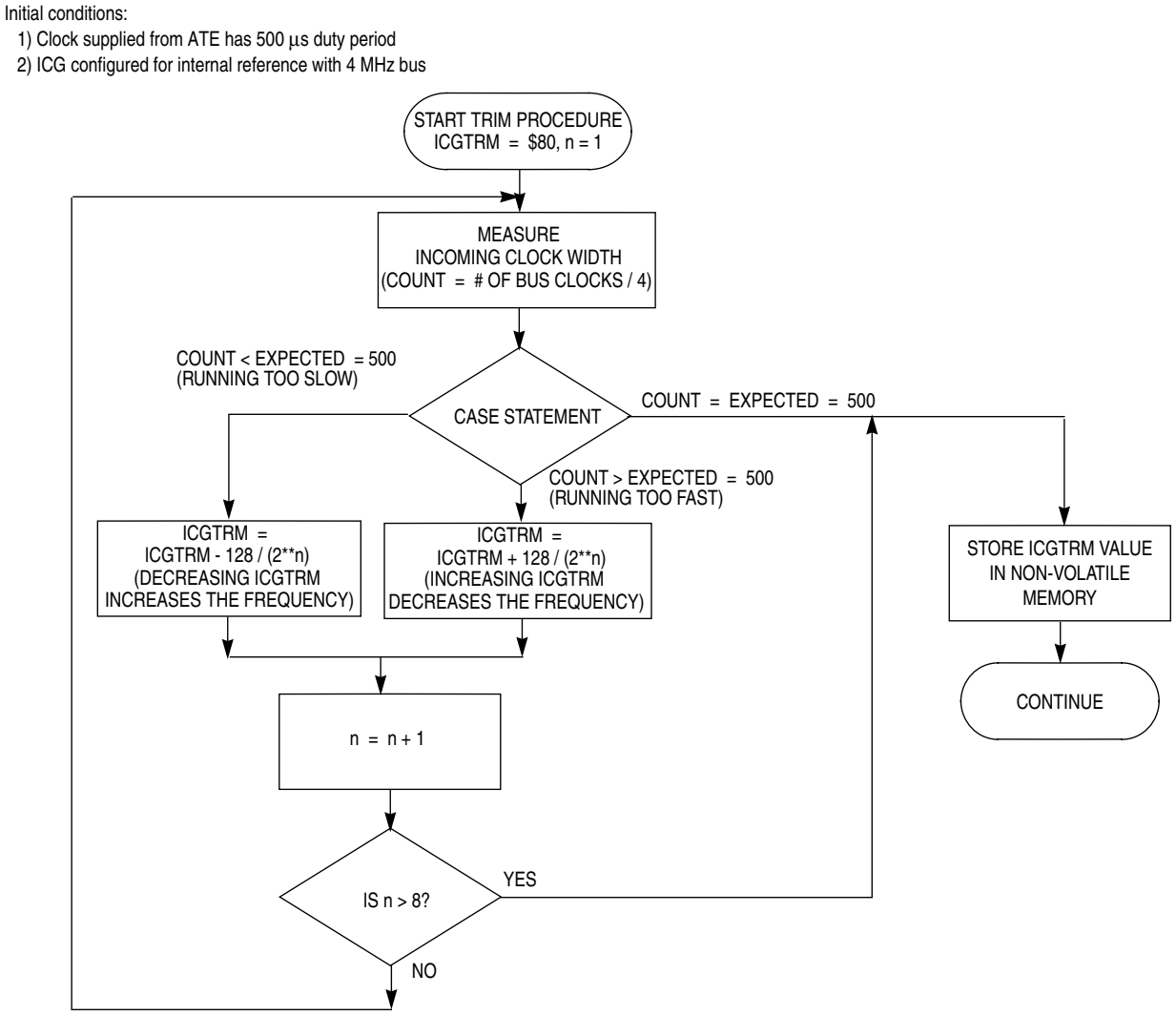


Figure 7-11. Trim Procedure

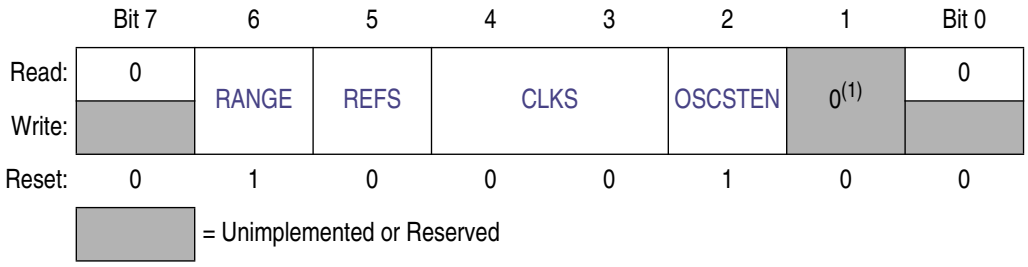
In this particular case, the MCU has been attached to a PCB and the entire assembly is undergoing final test with automated test equipment. A separate signal or message is provided to the MCU operating under user provided software control. The MCU initiates a trim procedure as outlined in Figure 7-11 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. Once the trim procedure is complete, the reduction divisor can be restored. This will prevent accidental overshoot of the maximum clock frequency.

7.5 ICG Registers and Control Bits

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.

7.5.1 ICG Control Register 1 (ICGC1)



1. This bit is reserved for Freescale Semiconductor internal use only. Any write operations to this register should write a 0 to this bit.

Figure 7-12. ICG Control Register 1 (ICGC1)

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.

1 = Oscillator configured for high frequency range. FLL loop prescale factor P is 1.
0 = Oscillator configured for low frequency range. FLL loop prescale factor P is 64.

REFS — External Reference Select

The REFS bit controls the external reference clock source for ICGERCLK. The REFS bit is write-once after a reset.

1 = Oscillator using crystal or resonator requested.
0 = External clock requested.

CLKS — Clock Mode Select

The CLKS bits control the clock mode according to Figure 7-13. 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.

Figure 7-13. CLKS Clock Select

CLKS[1:0]	Clock Mode
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).

OSCSTEN — Enable Oscillator in Off Mode

The OSCSTEN bit controls whether or not the oscillator circuit remains enabled when the ICG enters off mode.

1 = Oscillator enabled when ICG is in off mode, CLKS = 1X and REFST = 1.

0 = Oscillator disabled when ICG is in off mode unless ENABLE is high, CLKS = 10, and REFST = 1.

7.5.2 ICG Control Register 2 (ICGC2)

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	LOLRE	MFD			LOCRE	RFD		
Reset:	0	0	0	0	0	0	0	0

Figure 7-14. ICG Control Register 2 (ICGC2)

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.

1 = Generate a reset request on loss of lock.

0 = Generate an interrupt request on loss of lock.

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.

Table 7-7. MFD Multiplication Factor Select

MFD Value	Multiplication Factor (N)
000	4
001	6
010	8
011	10
100	12
101	14
110	16
111	18

LOCRE — Loss of Clock Reset Enable

The LOCRE bit determines how the system handles a loss of clock condition.

1 = Generate a reset request on loss of clock.

0 = Generate an interrupt request on loss of clock.

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.

Table 7-8. RFD Reduced Frequency Divider Select

RFD	Division Factor (R)
000	÷1
001	÷2
010	÷4
011	÷8
100	÷16
101	÷32
110	÷64
111	÷128

7.5.3 ICG Status Register 1 (ICGS1)

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CLKST		REFST	LOLS	LOCK	LOCS	ERCS	ICGIF
Write:								1
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-15. ICG Status Register 1 (ICGS1)

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.

Table 7-9. CLKST Clock Mode Status

CLKST[1:0]	Clock Status
00	Self-clocked
01	FLL engaged, internal reference
10	FLL bypassed, external reference
11	FLL engaged, external reference

REFST — Reference Clock Status

The REFST bit indicates which clock reference is currently selected by the Reference Select circuit.

1 = Crystal/Resonator selected.
0 = External Clock selected.

LOLS — FLL Loss of Lock Status

The LOLS bit is an indication of FLL lock status. If LOLS is set, it remains set until cleared by software or an MCU reset.

1 = FLL has unexpectedly lost lock since LOLS was last cleared, LOLRE determines action taken.
0 = FLL has not unexpectedly lost lock since LOLS was last cleared.

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.

1 = FLL is currently locked.
0 = FLL is currently unlocked.

LOCS — Loss Of Clock Status

The LOCS bit is an indication of ICG loss of clock status. If LOCS is set, it remains set until cleared by software on an MCU reset.

1 = ICG has lost clock since LOCS was last cleared, LOCRE determines action taken.
0 = ICG has not lost clock since LOCS was last cleared.

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.

1 = External reference clock is stable, frequency requirement is met.
0 = External reference clock is not stable, frequency requirement is not met.

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 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 0 to ICGIF has no effect.

1 = An ICG interrupt request is pending.

0 = No ICG interrupt request is pending.

7.5.4 ICG Status Register 2 (ICGS2)

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	DCOS
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-16. ICG Status Register 2 (ICGS2)

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.

1 = DCO clock is stable.

0 = DCO clock is unstable.

7.5.5 ICG Filter Registers (ICGFLTU, ICGFLTL)

	Bit 15	14	13	12	11	10	9	Bit 8
Read:	0	0	0	0	FLT			
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-17. ICG Upper Filter Register (ICGFLTU)

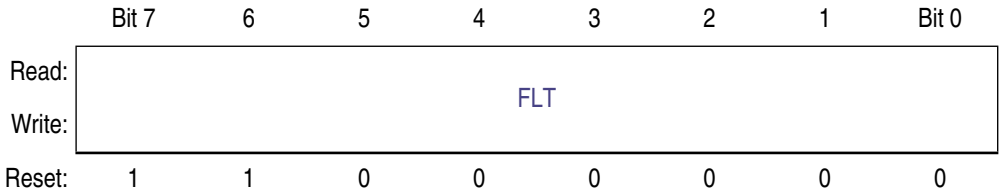


Figure 7-18. ICG Lower Filter Register (ICGFLTL)

The filter registers show the filter value (FLT).

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.

7.5.6 ICG Trim Register (ICGTRM)

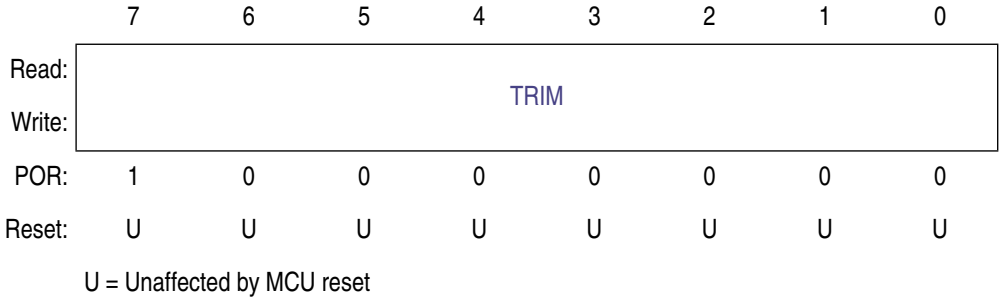


Figure 7-19. ICG Trim Register (ICGTRM)

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.

Chapter 8 Central Processor Unit (CPU)

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).

8.2 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 $0000–$00FF
  - 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

8.3 Programmer's Model and CPU Registers

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

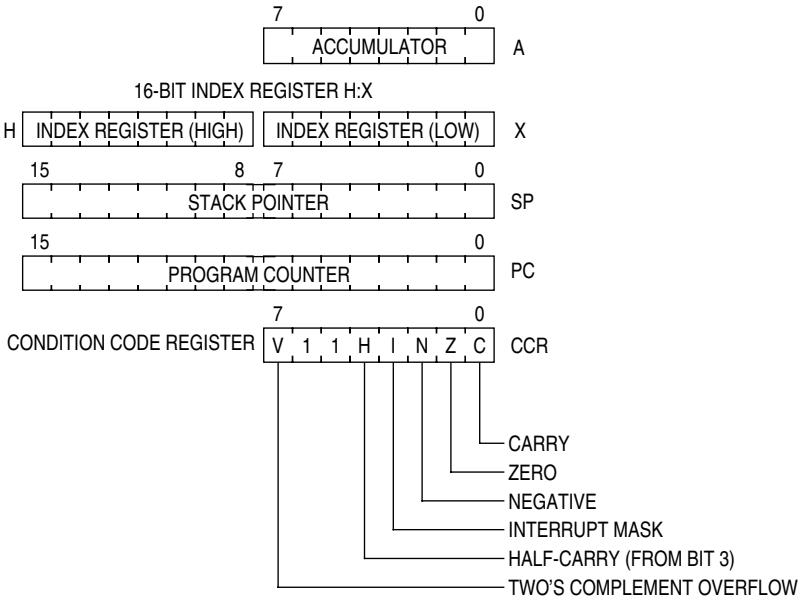


Figure 8-1. CPU Registers

8.3.1 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.

8.3.2 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 $00 during reset. Reset has no effect on the contents of X.

8.3.3 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 $00FF 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 $00FF).

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.

8.3.4 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 $FFFE and $FFFF. The vector stored there is the address of the first instruction that will be executed after exiting the reset state.

8.3.5 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/D.

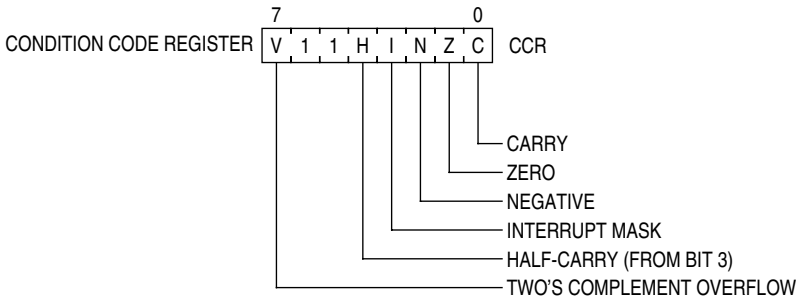


Figure 8-2. Condition Code Register

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.

1 = Overflow
0 = No overflow

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.

1 = Carry between bits 3 and 4
0 = No carry between bits 3 and 4

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.

1 = Interrupts disabled
0 = Interrupts enabled

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.

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.

1 = Negative result
0 = Non-negative result

Z — Zero Flag

The CPU sets the zero flag when an arithmetic operation, logic operation, or data manipulation produces a result of $00 or $0000. 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.

1 = Zero result
0 = Non-zero result

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.

1 = Carry out of bit 7
0 = No carry out of bit 7

8.4 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.

8.4.1 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.

8.4.2 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.

8.4.3 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.

8.4.4 Direct Addressing Mode (DIR)

In direct addressing mode, the instruction includes the low-order eight bits of an address in the direct page ($0000–$00FF). During execution a 16-bit address is formed by concatenating an implied $00 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.

8.4.5 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).

8.4.6 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.

8.4.6.1 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.

8.4.6.2 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 + $0001) after the operand has been fetched. This addressing mode is only used for MOV and CBEQ instructions.

8.4.6.3 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.

8.4.6.4 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 + $0001) after the operand has been fetched. This addressing mode is used only for the CBEQ instruction.

8.4.6.5 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.

8.4.6.6 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.

8.4.6.7 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.

8.5 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.

8.5.1 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 $FFFE and $FFFF and to fill the instruction queue in preparation for execution of the first program instruction.

8.5.2 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.

8.5.3 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.

8.5.4 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.

8.5.5 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.

8.6 HCS08 Instruction Set Summary

Instruction Set Summary Nomenclature

The nomenclature listed here is used in the instruction descriptions in Table 8-1.

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 + $0001	=	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 $0000–$00FF (high byte assumed to be $00)
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 ($00xx).
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 8-1. 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

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

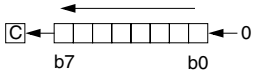
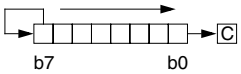
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
ASL *opr8a* ASLA ASLX ASL *oprx8*,X ASL ,X ASL *oprx8*,SP	Arithmetic Shift Left (Same as LSL)			–	–				DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	5 1 1 5 4 6
ASR *opr8a* ASRA ASRX 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
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 ⊕ 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) | (N ⊕ 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) | (N ⊕ 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 ⊕ V) = 1	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	Branch if (I) = 0	–	–	–	–	–	–	REL	2C	rr	3

Table 8-1. 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				
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
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) + $0002 push (PCL); SP ← (SP) – $0001 push (PCH); SP ← (SP) – $0001 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 ← $00 A ← $00 X ← $00 H ← $00 M ← $00 M ← $00 M ← $00	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

Table 8-1. 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				
COM *oprx8*,SP	Complement (One's Complement)	M ← (M̄)= $FF – (M)	0	–	–			1	SP1	9E63	ff	6
CPHX *opr16a*	Compare Index Register (H:X) with Memory	(H:X) – (M:M + $0001) (CCR Updated But Operands Not Changed)		–	–				EXT	3E	hh ll	6
CPHX #*opr16i*									IMM	65	jj kk	3
CPHX *opr8a*									DIR	75	dd	5
CPHX *oprx8*,SP									SP1	9EF3	ff	6
CPX #*opr8i*	Compare X (Index Register Low) with Memory	(X) – (M) (CCR Updated But Operands Not Changed)		–	–				IMM	A3	ii	2
CPX *opr8a*									DIR	B3	dd	3
CPX *opr16a*									EXT	C3	hh ll	4
CPX *oprx16*,X									IX2	D3	ee ff	4
CPX *oprx8*,X									IX1	E3	ff	3
CPX ,X									IX	F3		3
CPX *oprx16*,SP									SP2	9ED3	ee ff	5
CPX *oprx8*,SP									SP1	9EE3	ff	4
DAA	Decimal Adjust Accumulator After ADD or ADC of BCD Values	$(A)_{10}$	U	–	–				INH	72		1
DBNZ *oprx8*,*rel*	Decrement and Branch if Not Zero	Decrement A, X, or M Branch if (result) ≠ 0 DBNZX Affects X Not H	–	–	–	–	–	–	DIR	3B	dd rr	7
DBNZA *rel*									INH	4B	rr	4
DBNZX *rel*									INH	5B	rr	4
DBNZ *oprx8*,X,*rel*									IX1	6B	ff rr	7
DBNZ ,X,*rel*									IX	7B	rr	6
DBNZ *oprx8*,SP,*rel*									SP1	9E6B	ff rr	8
DEC *oprx8*,SP	Decrement	M ← (M) – $01		–	–			–	SP1	9E6A	ff	6
DIV	Divide	A ← (H:A)÷(X) H ← Remainder	–	–	–	–			INH	52		6
EOR #*opr8i*	Exclusive OR Memory with Accumulator	A ← (A ⊕ M)	0	–	–			–	IMM	A8	ii	2
EOR *opr8a*									DIR	B8	dd	3
EOR *opr16a*									EXT	C8	hh ll	4
EOR *oprx16*,X									IX2	D8	ee ff	4
EOR *oprx8*,X									IX1	E8	ff	3
EOR ,X									IX	F8		3
EOR *oprx16*,SP									SP2	9ED8	ee ff	5
EOR *oprx8*,SP									SP1	9EE8	ff	4
INC *oprx8*,SP	Increment	M ← (M) + $01		–	–			–	SP1	9E6C	ff	6
JMP *oprx8*,SP	Jump	PC ← Jump Address	–	–	–	–	–	–	SP1	9EEC	ff	6
JSR *oprx8*,SP	Jump to Subroutine	PC ← (PC) + *n* (*n* = 1, 2, or 3) Push (PCL); SP ← (SP) – $0001 Push (PCH); SP ← (SP) – $0001 PC ← Unconditional Address	–	–	–	–	–	–	SP1	9EED	ff	6

Table 8-1. HCS08 Instruction Set Summary (Sheet 5 of 7)

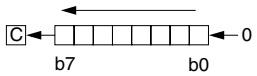
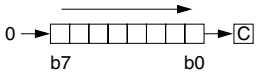
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
LDA #*oprx8i* LDA *oprx8a* LDA *oprx16a* LDA *oprx16*,X LDA *oprx8*,X LDA ,X LDA *oprx16*,SP LDA *oprx8*,SP	Load Accumulator from Memory	A ← (M)	0	–	–			–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A6 B6 C6 D6 E6 F6 9ED6 9EE6	ii dd hh ll ee ff ff	2 3 4 4 3 3 5 4
LDHX #*oprx16i* LDHX *oprx8a* LDHX *oprx16a* LDHX ,X LDHX *oprx16*,X LDHX *oprx8*,X LDHX *oprx8*,SP	Load Index Register (H:X) from Memory	H:X ← (M:M + $0001)	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
LDX #*oprx8i* LDX *oprx8a* LDX *oprx16a* 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 ff ee ff ff	2 3 4 4 3 3 5 4
LSL *oprx8a* 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 *oprx8a* 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 *oprx8a*,*oprx8a* MOV *oprx8a*,X+ MOV #*oprx8i*,*oprx8a* MOV ,X+,*oprx8a*	Move	$(M)_{destination}$ ← $(M)_{source}$ H:X ← (H:X) + $0001 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 *oprx8a* NEGA NEGX NEG *oprx8*,X NEG ,X NEG *oprx8*,SP	Negate (Two's Complement)	M ← – (M) = $00 – (M) A ← – (A) = $00 – (A) X ← – (X) = $00 – (X) M ← – (M) = $00 – (M) M ← – (M) = $00 – (M) M ← – (M) = $00 – (M)		–	–				DIR INH INH IX1 IX SP1	30 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 #*oprx8i* ORA *oprx8a* ORA *oprx16a* 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) – $0001	–	–	–	–	–	–	INH	87		2
PSHH	Push H (Index Register High) onto Stack	Push (H); SP ← (SP) – $0001	–	–	–	–	–	–	INH	8B		2
PSHX	Push X (Index Register Low) onto Stack	Push (X); SP ← (SP) – $0001	–	–	–	–	–	–	INH	89		2

Table 8-1. HCS08 Instruction Set Summary (Sheet 6 of 7)

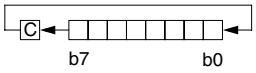
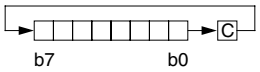
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
PULA	Pull Accumulator from Stack	SP ← (SP + $0001); Pull (A)	–	–	–	–	–	–	INH	86		3
PULH	Pull H (Index Register High) from Stack	SP ← (SP + $0001); Pull (H)	–	–	–	–	–	–	INH	8A		3
PULX	Pull X (Index Register Low) from Stack	SP ← (SP + $0001); 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
ROR *opr8a* RORA RORX 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 ← $FF (High Byte Not Affected)	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + $0001; Pull (CCR) SP ← (SP) + $0001; Pull (A) SP ← (SP) + $0001; Pull (X) SP ← (SP) + $0001; Pull (PCH) SP ← (SP) + $0001; Pull (PCL)							INH	80		9
RTS	Return from Subroutine	SP ← SP + $0001; Pull (PCH) SP ← SP + $0001; 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 + $0001) ← (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

Table 8-1. 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				
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) + $0001 Push (PCL); SP ← (SP) – $0001 Push (PCH); SP ← (SP) – $0001 Push (X); SP ← (SP) – $0001 Push (A); SP ← (SP) – $0001 Push (CCR); SP ← (SP) – $0001 I ← 1; PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		11
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) – $00 (A) – $00 (X) – $00 (M) – $00 (M) – $00 (M) – $00	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) + $0001	–	–	–	–	–	–	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) – $0001	–	–	–	–	–	–	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 8-2. 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 8-2. Opcode Map (Sheet 2 of 2)

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

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

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

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

9E60 6
NEG
3 SP1

HCS08 Cycles
Instruction Mnemonic
Addressing Mode

Chapter 9 Keyboard Interrupt (KBI) Module

9.1 Introduction

The MC9S08GB/GT has one KBI module with eight keyboard interrupt inputs that share port A pins. See Chapter 2, “Pins and Connections” for more information about the logic and hardware aspects of these pins.

9.1.1 Port A and Keyboard Interrupt Pins

MCU Pin:	PTA7/ KBI1P7	PTA6/ KBI1P6	PTA5/ KBI1P5	PTA4/ KBI1P4	PTA3/ KBI1P3	PTA2/ KBI1P2	PTA1/ KBI1P1	PTA0/ KBI1P0

Figure 9-1. Port A Pin Names

The following paragraphs discuss controlling the keyboard interrupt pins.

Port A is an 8-bit port which is shared among the KBI keyboard interrupt inputs and general-purpose I/O. The eight KBIPEn control bits in the KBIPE register allow selection of any combination of port A pins to be assigned as KBI inputs. Any pins which are enabled as KBI inputs will be forced to act as inputs and the remaining port A pins are available as general-purpose I/O pins controlled by the port A data (PTAD), data direction (PTADD), and pullup enable (PTAPE) registers.

KBI inputs can be configured for edge-only sensitivity or edge-and-level sensitivity. Bits 3 through 0 of port A are falling-edge/low-level sensitive while bits 7 through 4 can be configured for rising-edge/high-level or for falling-edge/low-level sensitivity.

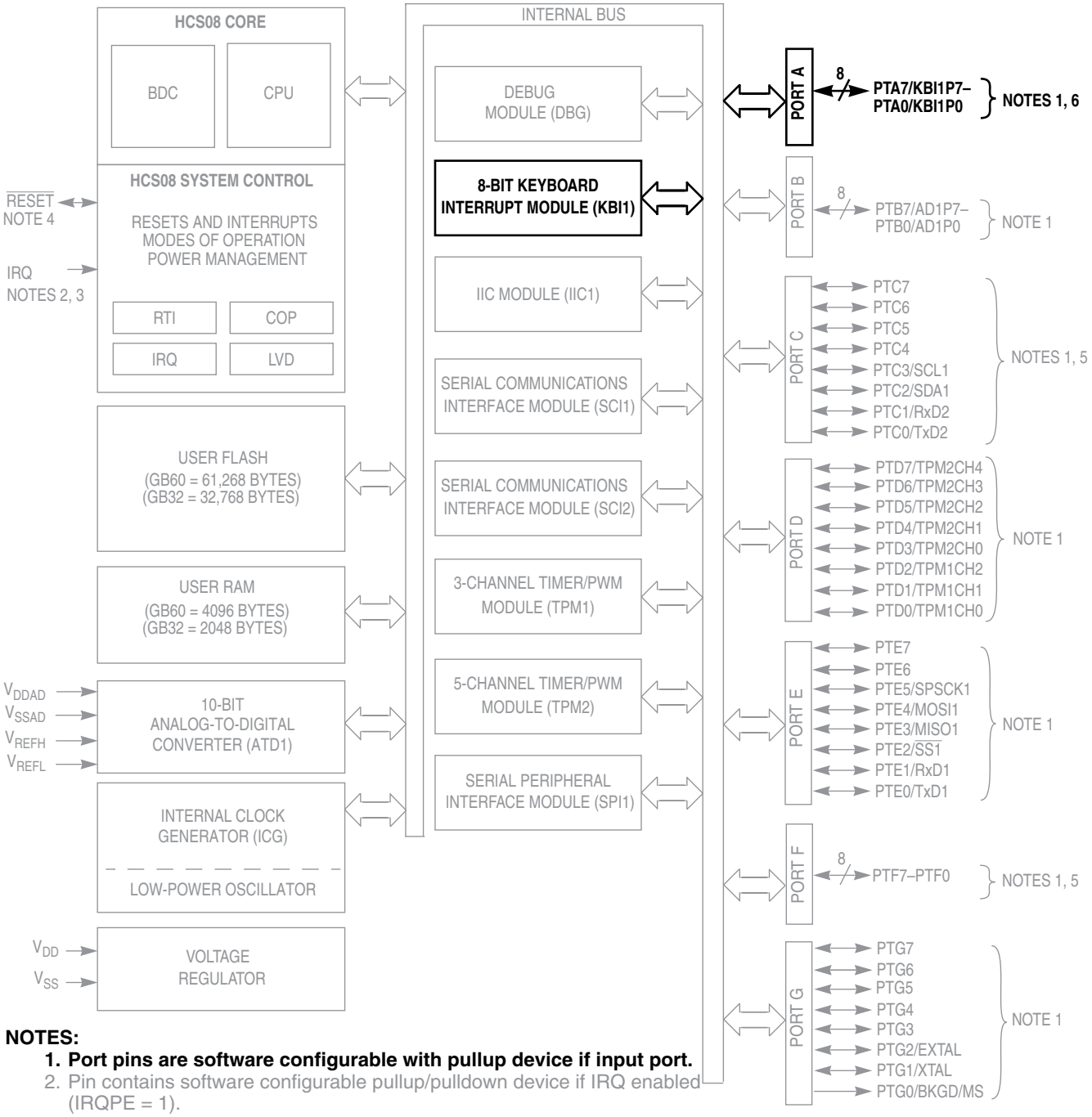
The eight PTAPEn control bits in the PTAPE register allow you to select whether an internal pullup device is enabled on each port A pin that is configured as an input. When any of bits 7 through 4 of port A are enabled as KBI inputs and are configured to detect rising edges/high levels, the pullup enable bits enable pulldown rather than pullup devices.

An enabled keyboard interrupt can be used to wake the MCU from wait or standby (stop3).

9.2 Features

The keyboard interrupt (KBI) module features include:

- Keyboard interrupts selectable on eight port pins:
  - Four falling-edge/low-level sensitive
  - Four falling-edge/low-level or rising-edge/high-level sensitive
  - Choice of edge-only or edge-and-level sensitivity
  - Common interrupt flag and interrupt enable control
  - Capable of waking up the MCU from stop3 or wait mode



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. High current drive
6. **Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).**

Figure 9-2. Block Diagram Highlighting KBI Module

9.3 KBI Block Diagram

Figure 9-3 shows the block diagram for a KBI module.

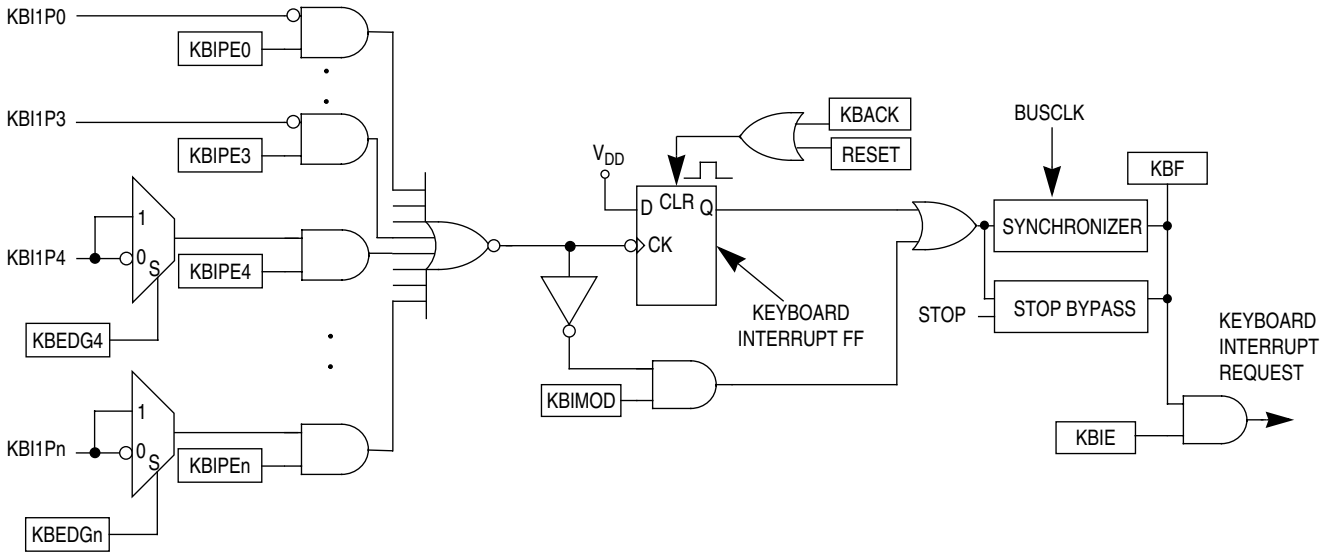


Figure 9-3. KBI Block Diagram

The KBI module allows up to eight pins to act as additional interrupt sources. Four of these pins allow falling-edge sensing while the other four 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.

9.4 Keyboard Interrupt (KBI) Module

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.

9.4.1 Pin Enables

The KBIPEn control bits in the KBI1PE register allow a user to enable (KBIPEn = 1) any combination of KBI-related port pins to be connected to the KBI module. Pins corresponding to 0s in KBI1PE are general-purpose I/O pins that are not associated with the KBI module.

9.4.2 Edge and Level Sensitivity

Synchronous logic is used to detect edges. Prior to detecting an edge, enabled keyboard inputs in a KBI module 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.

The KBIMOD control bit can be set to reconfigure the detection logic so that it detects edges and levels. In KBIMOD = 1 mode, the KBF status flag becomes set when an edge is detected (when one or more enabled pins change from the deasserted to the asserted level while all other enabled pins remain at their deasserted levels), but the flag is continuously set (and cannot be cleared) as long as any enabled keyboard input pin remains at the asserted level. When the MCU enters stop mode, the synchronous edge-detection logic is bypassed (because clocks are stopped). In stop mode, KBI inputs act as asynchronous level-sensitive inputs so they can wake the MCU from stop mode.

9.4.3 KBI Interrupt Controls

The KBF status flag becomes set (1) when an edge event has been detected on any KBI input pin. If KBIE = 1 in the KBI1SC register, a hardware interrupt will be requested whenever KBF = 1. The KBF flag is cleared by writing a 1 to the keyboard acknowledge (KBACK) bit.

When KBIMOD = 0 (selecting edge-only operation), KBF is always cleared by writing 1 to KBACK. When KBIMOD = 1 (selecting edge-and-level operation), KBF cannot be cleared as long as any keyboard input is at its asserted level.

9.5 KBI Registers and Control Bits

This section provides information about all registers and control bits associated with the KBI modules.

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

9.5.1 KBI Status and Control Register (KBI1SC)

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBF	0	KBIE	KBIMOD
Write:						KBACK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-4. KBI Status and Control Register (KBI1SC)

KBEDGn — Keyboard Edge Select for KBI Port Bit n (n = 7–4)

Each of these read/write bits selects the polarity of the edges and/or levels that are recognized as trigger events on the corresponding KBI port pin when it is configured as a keyboard interrupt input (KBIPEn = 1). Also see the KBIMOD control bit, which determines whether the pin is sensitive to edges-only or edges and levels.

1 = Rising edges/high levels.
0 = Falling edges/low levels.

KBF — Keyboard Interrupt Flag

This read-only status flag is set whenever the selected edge event has been detected on any of the enabled KBI port pins. This flag is cleared by writing a 1 to the KBACK control bit. The flag will remain set if KBIMOD = 1 to select edge-and-level operation and any enabled KBI port pin remains at the asserted level.

1 = KBI interrupt pending.
0 = No KBI interrupt pending.

KBF can be used as a software pollable flag (KBIE = 0) or it can generate a hardware interrupt request to the CPU (KBIE = 1).

KBACK — Keyboard Interrupt Acknowledge

This write-only bit (reads always return 0) is used to clear the KBF status flag by writing a 1 to KBACK. When KBIMOD = 1 to select edge-and-level operation and any enabled KBI port pin remains at the asserted level, KBF is being continuously set so writing 1 to KBACK does not clear the KBF flag.

KBIE — Keyboard Interrupt Enable

This read/write control bit determines whether hardware interrupts are generated when the KBF status flag equals 1. When KBIE = 0, no hardware interrupts are generated, but KBF can still be used for software polling.

1 = KBI hardware interrupt requested when KBF = 1.
0 = KBF does not generate hardware interrupts (use polling).

KBIMOD — Keyboard Detection Mode

This read/write control bit selects either edge-only detection or edge-and-level detection. KBI port bits 3 through 0 can detect falling edges-only or falling edges and low levels.

KBI port bits 7 through 4 can be configured to detect either:

- Rising edges-only or rising edges and high levels (KBEDGn = 1)
- Falling edges-only or falling edges and low levels (KBEDGn = 0)

1 = Edge-and-level detection.
0 = Edge-only detection.

9.5.2 KBI Pin Enable Register (KBI1PE)

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
Reset:	0	0	0	0	0	0	0	0

Figure 9-5. KBI Pin Enable Register (KBI1PE)

KBIPEn — Keyboard Pin Enable for KBI Port Bit n (n = 7–0)

Each of these read/write bits selects whether the associated KBI port pin is enabled as a keyboard interrupt input or functions as a general-purpose I/O pin.

1 = Bit n of KBI port enabled as a keyboard interrupt input

0 = Bit n of KBI port is a general-purpose I/O pin not associated with the KBI.

Chapter 10 Timer/PWM (TPM) Module

10.1 Introduction

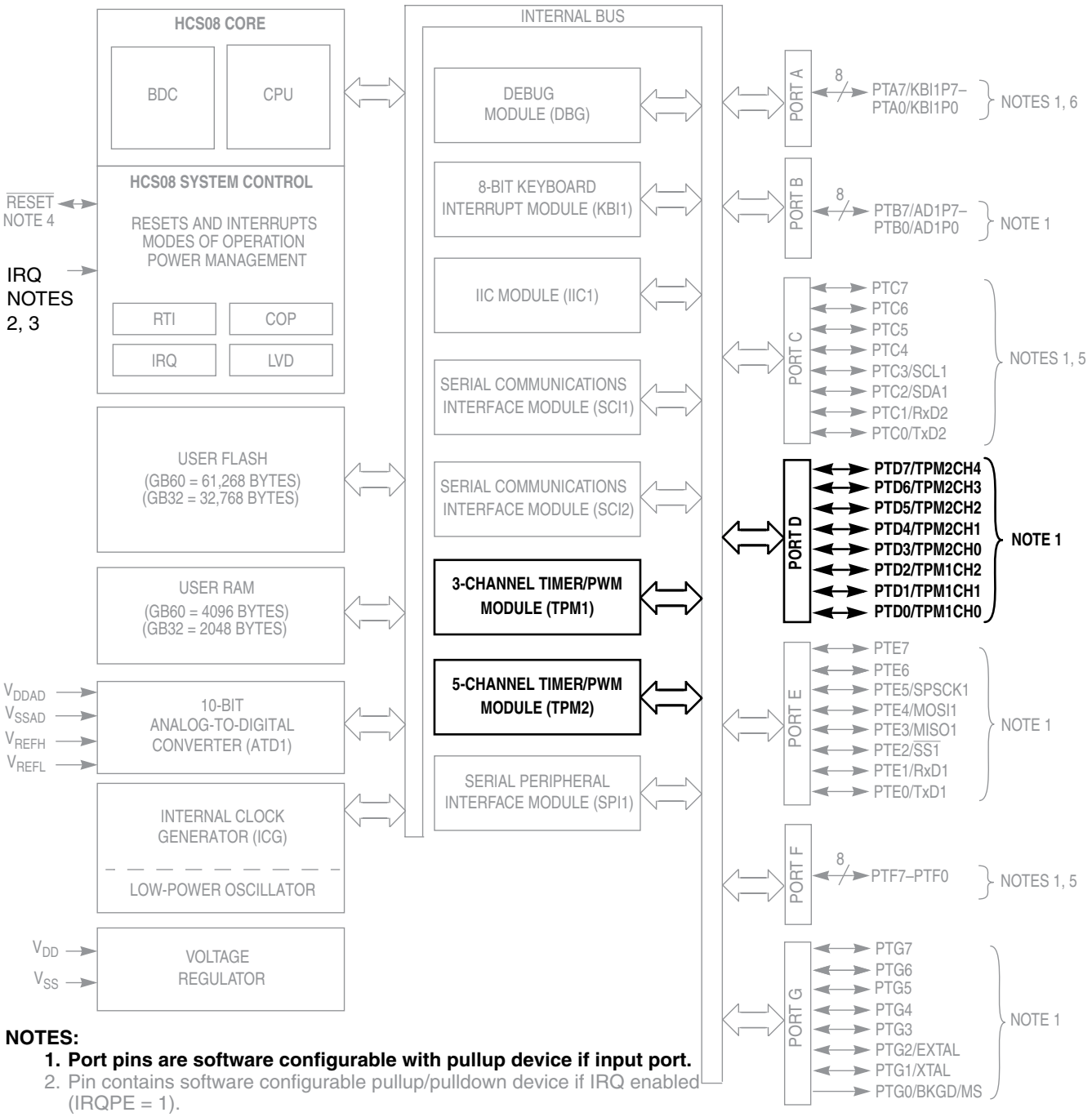
The MC9S08GB/GT includes two independent timer/PWM (TPM) modules which support traditional input capture, output compare, or buffered edge-aligned pulse-width modulation (PWM) on each channel. A control bit in each TPM configures all channels in that timer to operate as center-aligned PWM functions. In each of these two TPMs, timing functions are based on a separate 16-bit counter with prescaler and modulo features to control frequency and range (period between overflows) of the time reference. This timing system is ideally suited for a wide range of control applications, and the center-aligned PWM capability on the 3-channel TPM extends the field of applications to motor control in small appliances.

The use of the fixed system clock, XCLK, as the clock source for either of the TPM modules allows the TPM prescaler to run using the oscillator rate divided by two (ICGERCLK/2). This clock source must be selected only if the ICG is configured in either FBE or FEE mode. In FBE mode, this selection is redundant because the BUSCLK frequency is the same as XCLK. In FEE mode, the proper conditions must be met for XCLK to equal ICGERCLK/2 (see Section 7.3.9, “Fixed Frequency Clock”). Selecting XCLK as the clock source with the ICG in either FEI or SCM mode will result in the TPM being non-functional.

10.2 Features

The timer system in the MC9S08GB60 includes a 3-channel TPM1 and a separate 5-channel TPM2; the timer system in the MC9S08GB32 includes two 2-channel modules, TPM1 and TPM2. Timer system features include:

- A total of eight channels:
  - 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
- Each TPM may be configured for buffered, center-aligned pulse-width modulation (CPWM) on all channels
- Clock source to prescaler for each TPM is independently selectable as bus clock, fixed system clock, or an external pin
- Prescale 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 terminal count interrupt



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. High current drive
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).

Figure 10-1. Block Diagram Highlighting the TPM Modules

10.3 TPM Block Diagram

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). Figure 10-2 shows the structure of a TPM. Some MCUs include more than one TPM, with various numbers of channels.

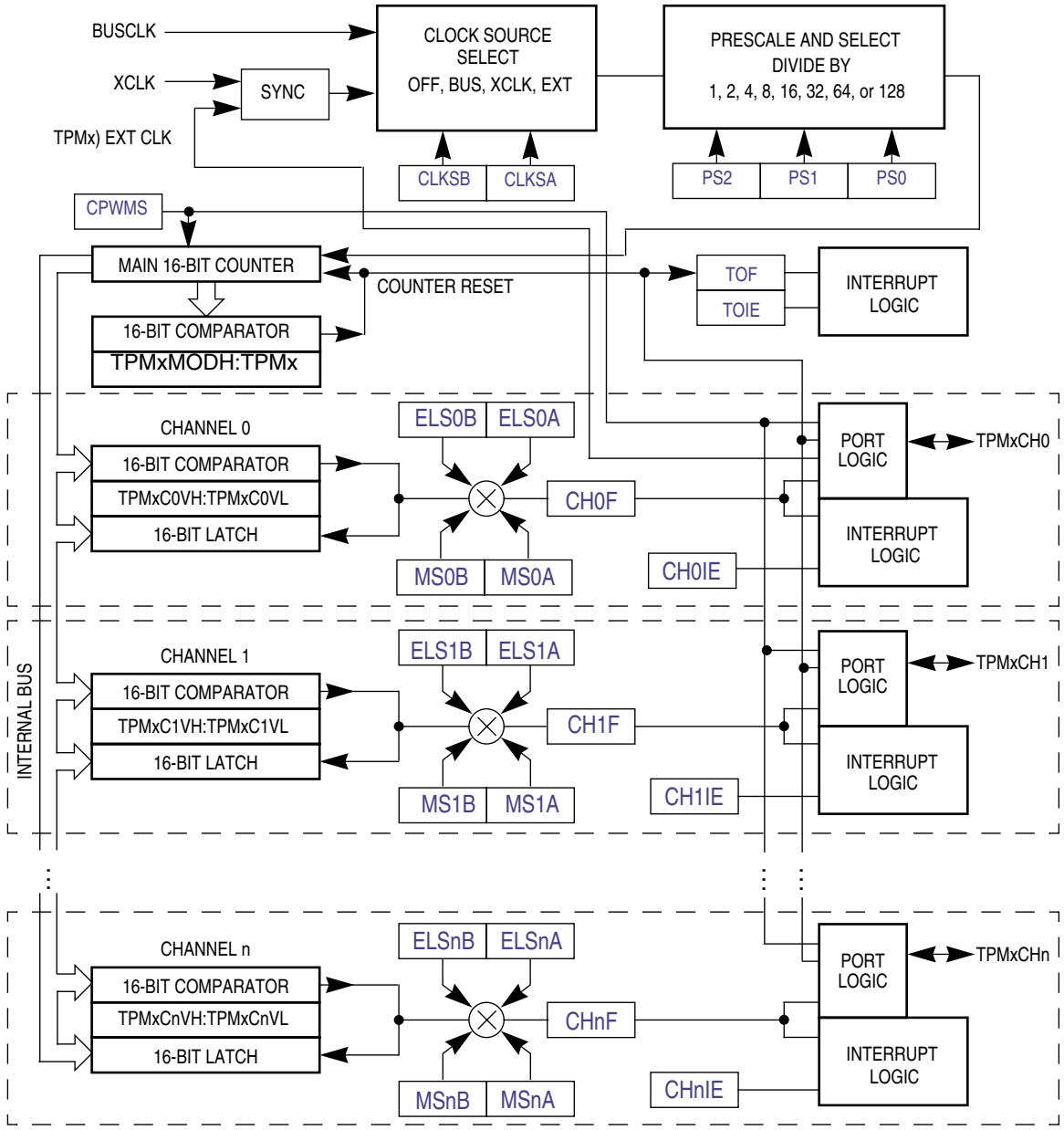


Figure 10-2. TPM Block Diagram

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 $0000 or $FFFF 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.

10.4 Pin Descriptions

Table 10-1 shows the MCU pins related to the TPM modules. When TPMxCH0 is used as an external clock input, the associated TPM channel 0 can not use the pin. (Channel 0 can still be used in output compare mode as a software timer.) When any of the pins 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.

10.4.1 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 connected to the TPMxCH0 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.

When the TPM is using the channel 0 pin for an external clock, the corresponding ELS0B:ELS0A control bits should be set to 0:0 so channel 0 is not trying to use the same pin.

10.4.2 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.

10.5 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.

10.5.1 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 through the TPMxCH0 pin. The maximum frequency allowed for the external clock option is one-fourth the bus rate. Refer to Section 10.7.1, “Timer x Status and Control Register (TPMxSC),” and Table 10-1 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 $0000 through its terminal count and then continues with $0000. The terminal count is $FFFF or a modulus value in TPMxMODH:TPMxMODL.

When center-aligned PWM operation is specified, the counter counts upward from $0000 through its terminal count and then counts downward to $0000 where it returns to up-counting. Both $0000 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 $0000 through $FFFF and overflows to $0000 on the next counting clock. TOF becomes set at the transition from $FFFF to $0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to $0000. 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 $0000 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.

10.5.2 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.

10.5.2.1 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.

10.5.2.2 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.

10.5.2.3 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 10-3 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.

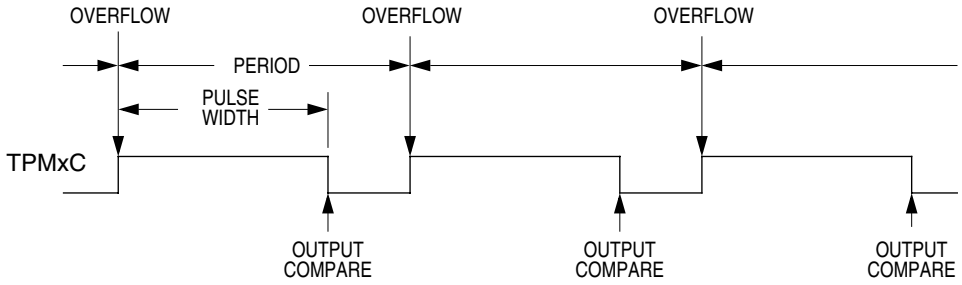


Figure 10-3. PWM Period and Pulse Width (ELSnA = 0)

When the channel value register is set to $0000, the duty cycle is 0 percent. By setting the timer channel value register (TPMxCnVH:TPMxCnVL) to a value greater than the modulus setting, 100 percent duty cycle can be achieved. This implies that the modulus setting must be less than $FFFF to get 100 percent 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 $0000. (The new duty cycle does not take effect until the next full period.)

10.5.3 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 $0001 to $7FFF 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. 10-1***

period = 2 x (TPMxMODH:TPMxMODL);
for TPMxMODH:TPMxMODL = $0001–$7FFF ***Eqn. 10-2***

If the channel value register TPMxCnVH:TPMxCnVL is zero or negative (bit 15 set), the duty cycle will be 0 percent. If TPMxCnVH:TPMxCnVL is a positive value (bit 15 clear) and is greater than the (nonzero) modulus setting, the duty cycle will be 100 percent because the duty cycle compare will never occur. This implies the usable range of periods set by the modulus register is $0001 through $7FFE ($7FFF if

generation of 100 percent 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 = $0000 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 $0000 through $FFFF, but when CPWMS = 1 the counter needs a valid match to the modulus register somewhere other than at $0000 in order to change directions from up-counting to down-counting.

Figure 10-4 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.

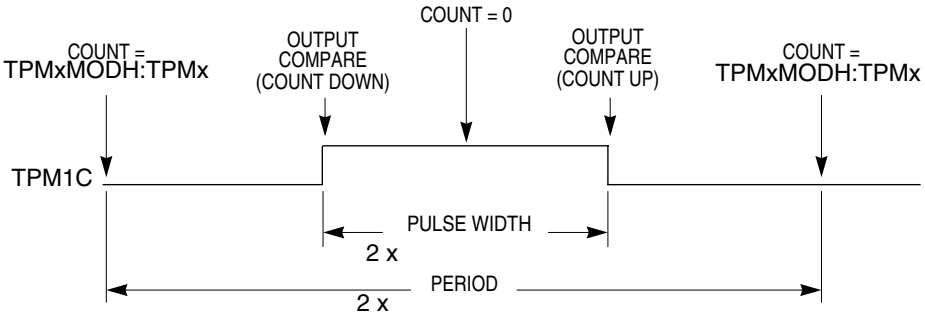


Figure 10-4. CPWM Period and Pulse Width (ELSnA = 0)

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.

10.6 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.

10.6.1 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.

10.6.2 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 $0000 through $FFFF and overflows to $0000 on the next counting clock. TOF becomes set at the transition from $FFFF to $0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to $0000. 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 $0000 count value corresponds to the center of a period.)

10.6.3 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 10.6.1, “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 10.6.1, “Clearing Timer Interrupt Flags.”

10.6.4 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 10.6.1, “Clearing Timer Interrupt Flags.”

10.7 TPM Registers and Control Bits

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.

10.7.1 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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-5. Timer x Status and Control Register (TPMxSC)

TOF — Timer Overflow Flag

This flag is set when the TPM counter changes to $0000 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 the TOF bit. Writing a 1 to TOF has no effect.

1 = TPM counter has overflowed.

0 = TPM counter has not reached modulo value or overflow.

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.

1 = TOF interrupts enabled.

0 = TOF interrupts inhibited (use software polling).

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 the CPWMS bit.

1 = All TPMx channels operate in center-aligned PWM mode.

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.

CLKSB:CLKSA — Clock Source Select

As shown in Table 10-1, 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.

Table 10-1. TPM Clock Source Selection

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

1. The maximum frequency that is allowed as an external clock is one-fourth of the bus frequency.
2. When the TPMxCH0 pin is selected as the TPM clock source, the corresponding ELS0B:ELS0A control bits should be set to 0:0 so channel 0 does not try to use the same pin for a conflicting function.

PS2:PS1:PS0 — Prescale Divisor Select

This 3-bit field selects one of eight divisors for the TPM clock input as shown in Table 10-2. 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 10-2. 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

10.7.2 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.

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

Figure 10-6. Timer x Counter Register High (TPMxCNTH)

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

Figure 10-7. 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.

10.7.3 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 $0000 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 the TOF bit and overflow interrupts until the other byte is written. Reset sets the TPM counter modulo registers to $0000, which results in a free-running timer counter (modulo disabled).

	Bit 7	6	5	4	3	3	2	Bit 0
Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
Reset:	0	0	0	0	0	0	0	0

Figure 10-8. Timer x Counter Modulo Register High (TPMxMODH)

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 10-9. 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.

10.7.4 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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	0	0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-10. Timer x Channel n Status and Control Register (TPMxCnSC)

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 the CHnF bit. Writing a 1 to CHnF has no effect.

1 = Input capture or output compare event occurred on channel n.

0 = No input capture or output compare event occurred on channel n.

CHnIE — Channel n Interrupt Enable

This read/write bit enables interrupts from channel n. Reset clears the CHnIE bit.

1 = Channel n interrupt requests enabled.

0 = Channel n interrupt requests disabled (use software polling).

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 10-3.

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 10-3 for a summary of channel mode and setup controls.

Table 10-3. 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.

ELSnB:ELSnA — Edge/Level Select Bits

Depending on the operating mode for the timer channel as set by CPWMS:MSnB:MSnA and shown in Table 10-3, 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. This is also the setting required for channel 0 when the TPMxCH0 pin is used as an external clock input.

10.7.5 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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
Reset:	0	0	0	0	0	0	0	0

Figure 10-11. Timer x Channel Value Register High (TPMxCnVH)

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 10-12. Timer x 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.

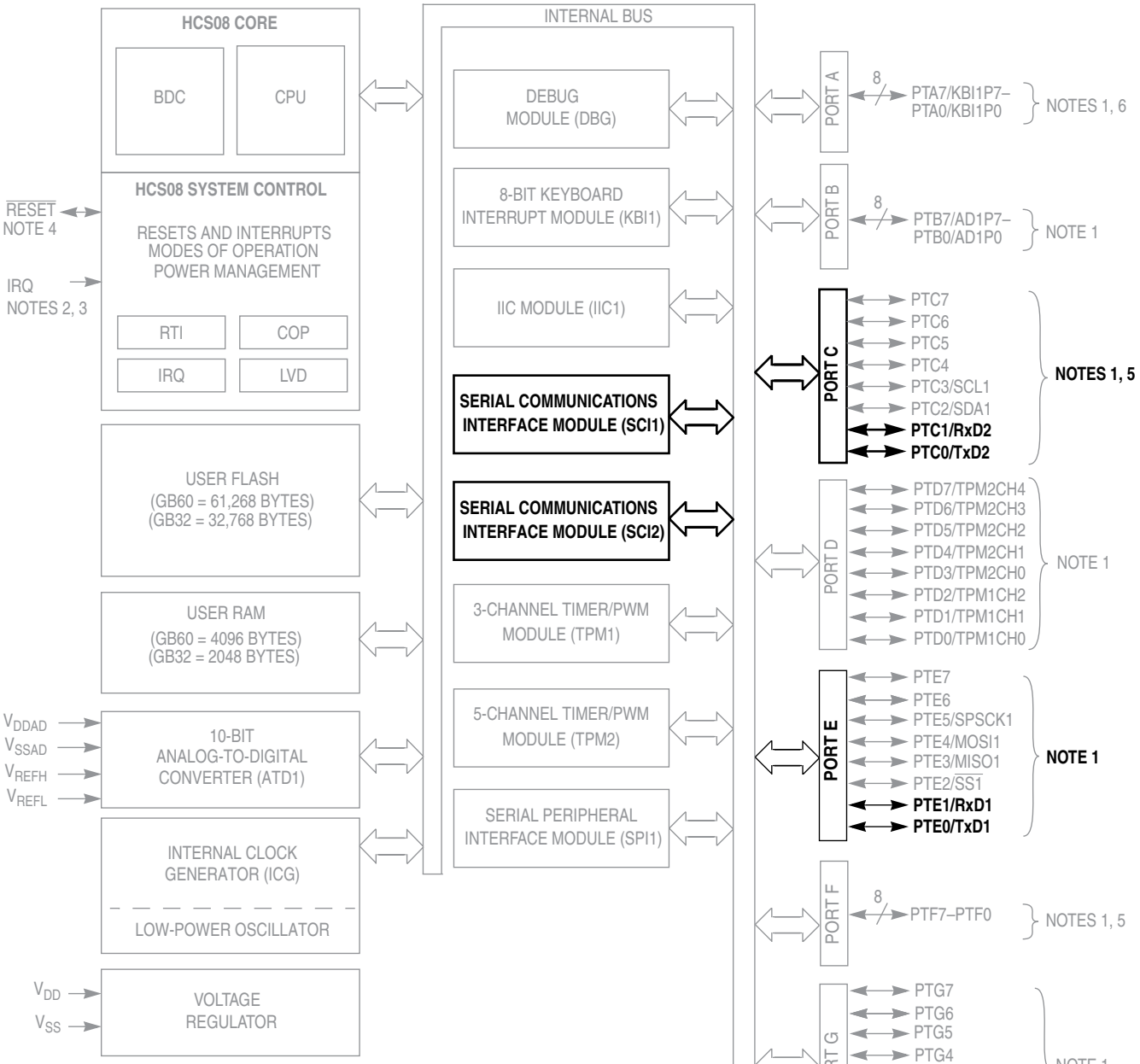
Chapter 11 Serial Communications Interface (SCI) Module

11.1 Introduction

The MC9S08GB/GT includes two independent serial communications interface (SCI) modules — sometimes called universal asynchronous receiver/transmitters (UARTs). Typically, these systems are used to connect to the RS232 serial input/output (I/O) port of a personal computer or workstation, and they can also be used to communicate with other embedded controllers.

A flexible, 13-bit, modulo-based baud rate generator supports a broad range of standard baud rates beyond 115.2 kbaud. Transmit and receive within the same SCI use a common baud rate, and each SCI module has a separate baud rate generator.

This SCI system offers many advanced features not commonly found on other asynchronous serial I/O peripherals on other embedded controllers. The receiver employs an advanced data sampling technique that ensures reliable communication and noise detection. Hardware parity, receiver wakeup, and double buffering on transmit and receive are also included.



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. **High current drive**
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).

Figure 11-1. Block Diagram Highlighting the SCI Modules

11.2 Features

Features of SCI module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Double-buffered transmitter and receiver with separate enables
- Programmable baud rates (13-bit modulo divider)
- Interrupt-driven or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
- Hardware parity generation and checking
- Programmable 8-bit or 9-bit character length
- Receiver wakeup by idle-line or address-mark

11.3 SCI System 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.

11.4 Baud Rate Generation

As shown in Figure 11-2, the clock source for the SCI baud rate generator is the bus-rate clock.

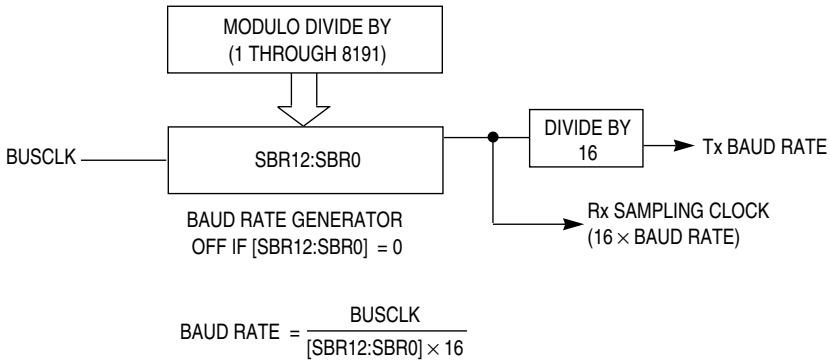


Figure 11-2. SCI Baud Rate Generation

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 MC9S08GB/GT re-synchronizes 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.

11.5 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.

11.5.1 Transmitter Block Diagram

Figure 11-3 shows the transmitter portion of the SCI.

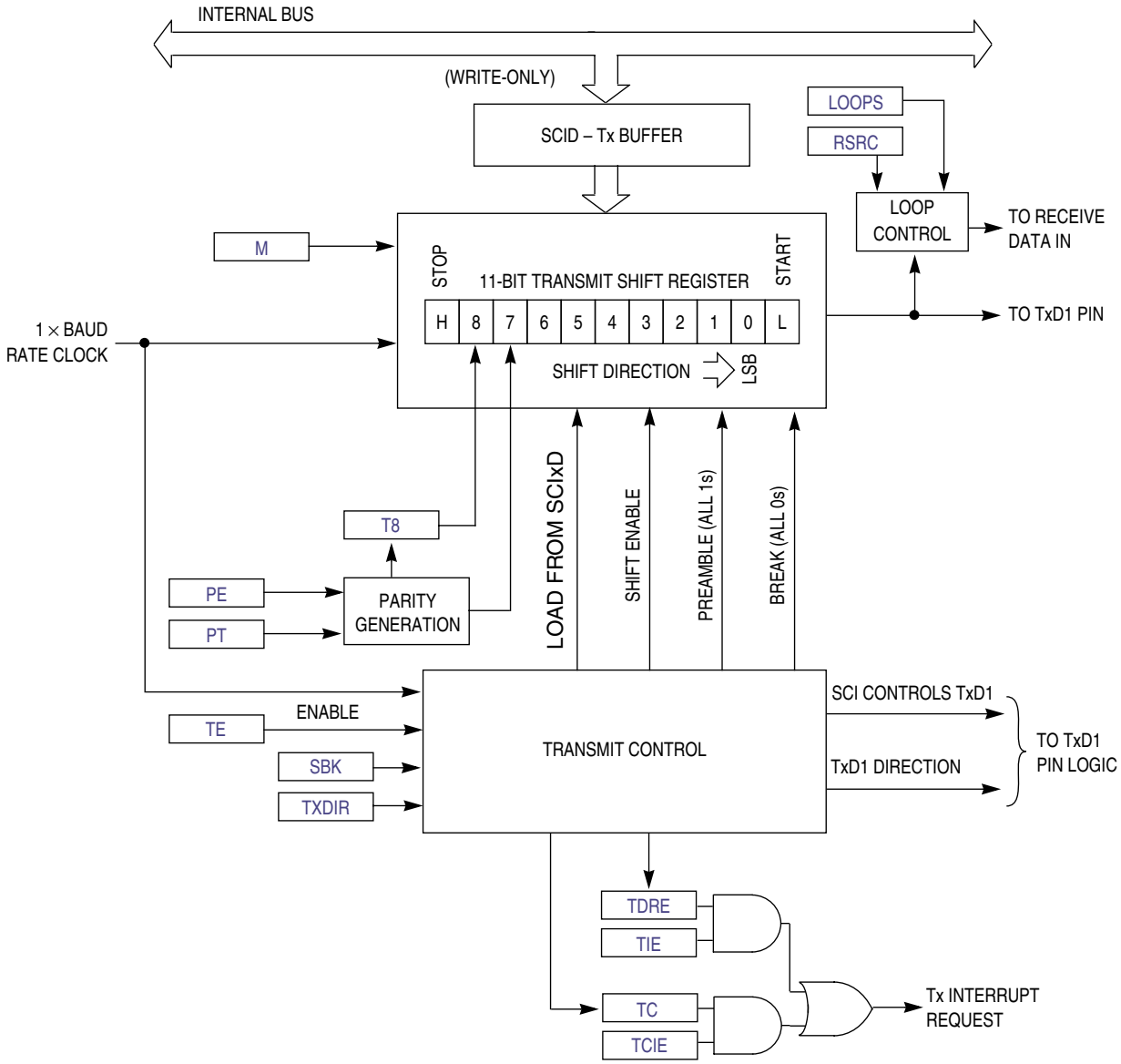


Figure 11-3. SCI Transmitter Block Diagram

The transmitter is enabled by setting the TE bit in SCIxC2. This queues a preamble character that is one full character frame of logic high. The transmitter then remains idle (TxD1 pin remains high) until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the SCI data register (SCIxD).

The central element of the SCI transmitter is the transmit shift register that is 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 SCIxD.

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.

11.5.2 Send Break and Queued Idle

The SBK control bit in SCIxC2 is used to send break characters that were originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0 (including a 0 where the stop bit would be normally). 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 (or nine) data bits and a framing error (FE = 1).

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.

11.6 Receiver Functional Description

In this section, the receiver block diagram (Figure 11-4) 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.

11.6.1 Receiver Block Diagram

Figure 11-4 shows the receiver portion of the SCI.

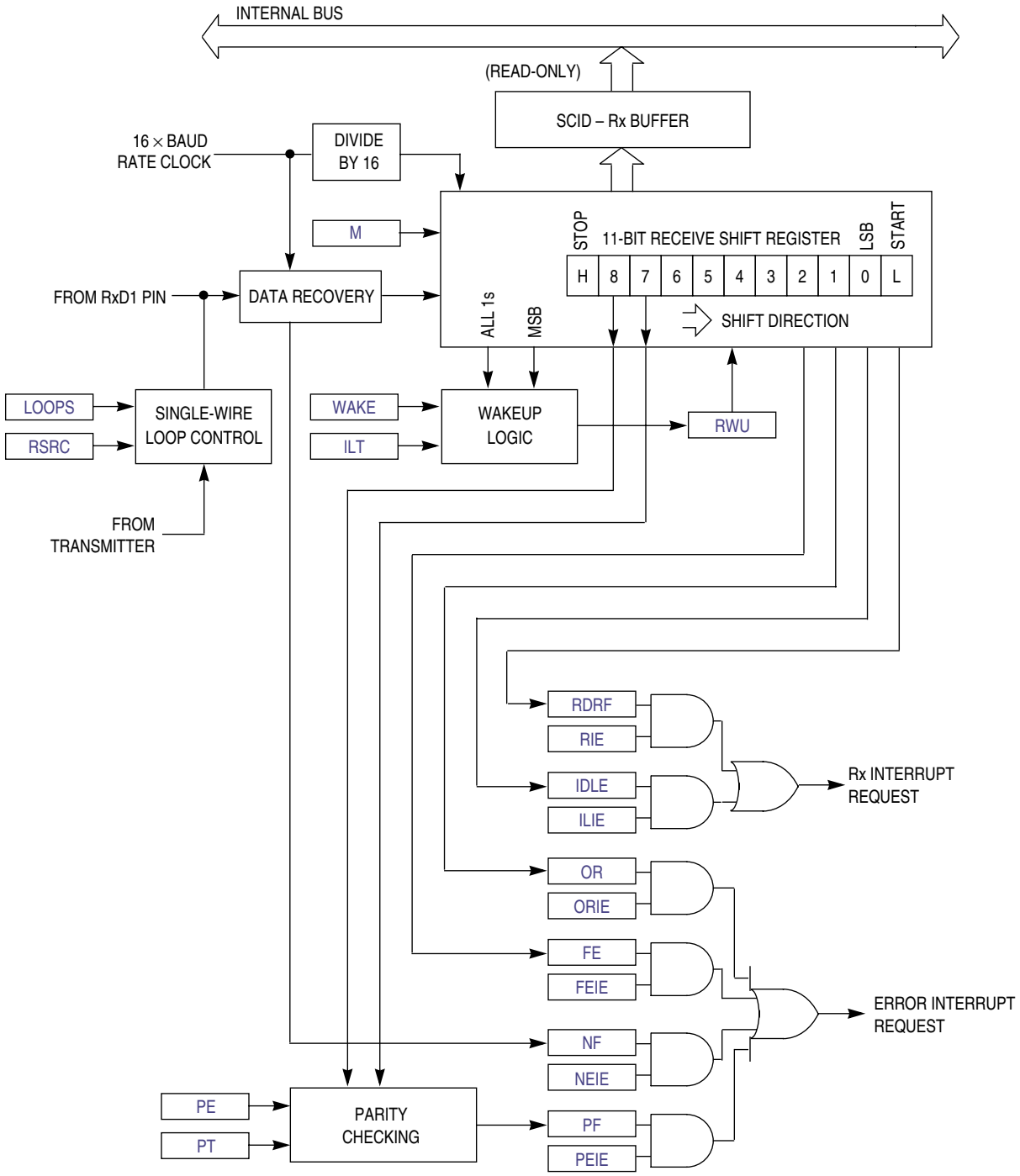


Figure 11-4. SCI Receiver Block Diagram

The receiver is enabled by setting the RE bit in SCIxC2. Character frames consist of a start bit of logic 0, eight (or nine) data bits (LSB first), and a stop bit of logic 1. For information about 9-bit data mode, refer to Section 11.8.1, "8- and 9-Bit Data Modes." For the remainder of this discussion, we assume the SCI is configured for normal 8-bit data mode.

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

When a program detects that the receive data register is full (RDRF = 1), it gets the data from the receive data register by reading SCIxD. The RDRF flag is cleared automatically by a 2-step sequence (which is normally satisfied in the course of the user's program that handles receive data). Refer to Section 11.7, "Interrupts and Status Flags," for more details about flag clearing.

11.6.2 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 re-synchronized to bit times. This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

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

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

11.6.3 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 SCIxC2. 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.

11.6.3.1 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). 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 doesn't 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.

11.6.3.2 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).

11.7 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 SCIxD. 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 SCIxD. The RDRF flag is cleared by reading SCIxS1 while RDRF = 1 and then reading SCIxD. If the SCI is configured to operate in 9-bit mode, an additional read to the SCIxC3 register is required to clear RDRF

When polling is used, this sequence is naturally satisfied in the normal course of the user program. If hardware interrupts are used, SCIxS1 must be read in the interrupt service routine (ISR). Normally, this is done in the ISR anyway to check for receive errors, so the sequence is automatically satisfied.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the RxD1 line remains idle for an extended period of time. IDLE is cleared by reading SCIxS1 while IDLE = 1 and then reading SCIxD. After IDLE has been cleared, it cannot become set again until the receiver has received at least one new character and has set RDRF.

If the associated error was detected in the received character that caused RDRF to be set, the error flags — noise flag (NF), framing error (FE), and parity error flag (PF) — 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.

11.8 Additional SCI Functions

The following sections describe additional SCI functions.

11.8.1 8- and 9-Bit Data Modes

The SCI system (transmitter and receiver) can be configured to operate in 9-bit data mode by setting the M control bit in SCIxC1. In 9-bit mode, there is a ninth data bit to the left of the MSB of the SCI data register. For the transmit data buffer, this bit is stored in T8 in SCIxC3. For the receiver, the ninth bit is held in R8 in SCIxC3.

When transmitting 9-bit data, write to the T8 bit before writing to SCIxD for coherent writes to the transmit data buffer. If the bit value to be transmitted as the ninth bit of a new character is the same as for the previous character, it is not necessary to write to T8 again. When data is transferred from the transmit data buffer to the transmit shifter, the value in T8 is copied at the same time data is transferred from SCIxD to the shifter.

When receiving 9-bit data, clear the RDRF bit by reading both R8 and SCIxD. R8 and SCIxD can be read in either order.

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.

11.9 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.

11.9.1 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.

11.9.2 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 SCIxC3 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.

11.10 SCI Registers and Control Bits

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 section 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.

Some MCU systems have more than one SCI, so register names include placeholder characters to identify which SCI is being referenced. For example, SCIxC1 refers to the SCIx control register 1 and SCI2C1 is the control register 1 for SCI2.

11.10.1 SCI x Baud Rate Registers (SCIxBDH, SCIxBDL)

This pair of registers controls the prescale divisor for SCI baud rate generation. To update the 13-bit baud rate setting [SBR12:SBR0], first write to SCIxBDH to buffer the high half of the new value and then write to SCIxBDL. The working value in SCIxBDH does not change until SCIxBDL is written.

SCIxBDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled (RE or TE bits in SCIxC2 are written to 1).

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-5. SCI Baud Rate Register (SCIxBDH)

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
Write:								
Reset:	0	0	0	0	0	1	0	0

= Unimplemented or Reserved

Figure 11-6. SCI x Baud Rate Register (SCIxBDL)

SBR12:SBR0 — 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).

11.10.2 SCI x Control Register 1 (SCIxC1)

This read/write register is used to control various optional features of the SCI system.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 11-7. SCI x Control Register 1 (SCIxC1)

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.

1 = Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input. (See RSRC bit.) RxD1 pin is not used by SCI.

0 = Normal operation — RxD1 and TxD1 use separate pins.

SCISWAI — SCI Stops in Wait Mode

1 = SCI clocks freeze while CPU is 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.

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 TxD1 pin and RSRC determines whether this connection is also connected to the transmitter output.

1 = Single-wire SCI mode where the TxD1 pin is connected to the transmitter output and receiver input.

0 = Provided LOOPS = 1, RSRC = 0 selects internal loop back mode and the SCI does not use the RxD1 or TxD1 pins.

M — 9-Bit or 8-Bit Mode Select

1 = Receiver and transmitter use 9-bit data characters start + 8 data bits (LSB first) + 9th data bit + stop.

0 = Normal — start + 8 data bits (LSB first) + stop.

WAKE — Receiver Wakeup Method Select

Refer to Section 11.6.3, “Receiver Wakeup Operation,” for more information.

1 = Address-mark wakeup.

0 = Idle-line wakeup.

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 11.6.3.1, “Idle-Line Wakeup,” for more information.

1 = Idle character bit count starts after stop bit.

0 = Idle character bit count starts after start bit.

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.

1 = Parity enabled.

0 = No hardware parity generation or checking.

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.

1 = Odd parity.

0 = Even parity.

11.10.3 SCI x Control Register 2 (SCIxC2)

This register can be read or written at any time.

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset:	0	0	0	0	0	0	0	0

Figure 11-8. SCI x Control Register 2 (SCIxC2)

TIE — Transmit Interrupt Enable (for TDRE)

1 = Hardware interrupt requested when TDRE flag is 1.
0 = Hardware interrupts from TDRE disabled (use polling).

TCIE — Transmission Complete Interrupt Enable (for TC)

1 = Hardware interrupt requested when TC flag is 1.
0 = Hardware interrupts from TC disabled (use polling).

RIE — Receiver Interrupt Enable (for RDRF)

1 = Hardware interrupt requested when RDRF flag is 1.
0 = Hardware interrupts from RDRF disabled (use polling).

ILIE — Idle Line Interrupt Enable (for IDLE)

1 = Hardware interrupt requested when IDLE flag is 1.
0 = Hardware interrupts from IDLE disabled (use polling).

TE — Transmitter Enable

1 = Transmitter on.
0 = Transmitter off.

TE must be 1 in order to use the SCI transmitter. Normally, when TE = 1, the SCI forces the TxD1 pin to act as an output for the SCI system. If LOOPS = 1 and RSRC = 0, the TxD1 pin reverts to being a port B general-purpose I/O pin even if TE = 1.

When the SCI is configured for single-wire operation (LOOPS = RSRC = 1), TXDIR controls the direction of traffic on the single SCI communication line (TxD1 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 11.5.2, “Send Break and Queued Idle,” for more details.

When TE is written to 0, the transmitter keeps control of the port TxD1 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.

RE — Receiver Enable

When the SCI receiver is off, the RxD1 pin reverts to being a general-purpose port I/O pin.
1 = Receiver on.
0 = Receiver off.

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 11.6.3, “Receiver Wakeup Operation,” for more details.
1 = SCI receiver in standby waiting for wakeup condition.
0 = Normal SCI receiver operation.

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 11.5.2, “Send Break and Queued Idle,” for more details.
1 = Queue break character(s) to be sent.
0 = Normal transmitter operation.

11.10.4 SCI x Status Register 1 (SCIxS1)

This register has eight read-only status flags. Writes have no effect. Special software sequences (that do not involve writing to this register) are used to clear these status flags.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write:								
Reset:	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-9. SCI x Status Register 1 (SCIxS1)

TDRE — Transmit Data Register Empty Flag

TDRE is set out of reset and when a transmit data value transfers from the transmit data buffer to the transmit shifter, leaving room for a new character in the buffer. To clear TDRE, read SCIxS1 with TDRE = 1 and then write to the SCI data register (SCIxD).
1 = Transmit data register (buffer) empty.
0 = Transmit data register (buffer) full.

TC — Transmission Complete Flag

TC is set out of reset and when TDRE = 1 and no data, preamble, or break character is being transmitted.

1 = Transmitter idle (transmission activity complete).
0 = Transmitter active (sending data, a preamble, or a break).

TC is cleared automatically by reading SCIxS1 with TC = 1 and then doing one of the following:

— Write to the SCI data register (SCIxD) to transmit new data
— Queue a preamble by changing TE from 0 to 1
— Queue a break character by writing 1 to SBK in SCIxC2

RDRF — Receive Data Register Full Flag

RDRF becomes set when a character transfers from the receive shifter into the receive data register (SCIxD). In 8-bit mode, to clear RDRF, read SCIxS1 with RDRF = 1 and then read the SCI data register (SCIxD). In 9-bit mode, to clear RDRF, read SCIxS1 with RDRF = 1 and then read SCIxD and the SCI control 3 register (SCIxC3). SCIxD and SCIxC3 can be read in any order, but the flag is cleared only after both data registers are read.

1 = Receive data register full.
0 = Receive data register empty.

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 SCIxS1 with IDLE = 1 and then read the SCI data register (SCIxD). After IDLE has been cleared, it cannot become set again until after a new character has been received and RDRF has been set. IDLE will be set only once even if the receive line remains idle for an extended period.

1 = Idle line was detected.
0 = No idle line detected.

OR — Receiver Overrun Flag

OR is set when a new serial character is ready to be transferred to the receive data register (buffer), but the previously received character has not been read from SCIxD yet. In this case, the new character (and all associated error information) is lost because there is no room to move it into SCIxD. To clear OR, read SCIxS1 with OR = 1 and then read the SCI data register (SCIxD).

1 = Receive overrun (new SCI data lost).
0 = No overrun.

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 SCIxS1 and then read the SCI data register (SCIxD).

1 = Noise detected in the received character in SCIxD.
0 = No noise detected.

FE — Framing Error Flag

FE is set at the same time as RDRF when the receiver detects a logic 0 where the stop bit was expected. This suggests the receiver was not properly aligned to a character frame. To clear FE, read SCIxS1 with FE = 1 and then read the SCI data register (SCIxD).

1 = Framing error.
0 = No framing error detected. This does not guarantee the framing is correct.

PF — Parity Error Flag

PF is set at the same time as RDRF when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, read SCIxS1 and then read the SCI data register (SCIxD).

1 = Parity error.
0 = No parity error.

11.10.5 SCI x Status Register 2 (SCIxS2)

This register has one read-only status flag. Writes have no effect.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	RAF
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-10. SCI x Status Register 2 (SCIxS2)

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.

1 = SCI receiver active (RxD1 input not idle).

0 = SCI receiver idle waiting for a start bit.

11.10.6 SCI x Control Register 3 (SCIxC3)

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R8	T8	TXDIR	0	ORIE	NEIE	FEIE	PEIE
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-11. SCI x Control Register 3 (SCIxC3)

R8 — Ninth Data Bit for Receiver

When the SCI is configured for 9-bit data (M = 1), R8 can be thought of as a ninth receive data bit to the left of the MSB of the buffered data in the SCIxD register. When reading 9-bit data, both R8 and SCIxD must be read to complete the automatic RDRF clearing sequence.

T8 — Ninth Data Bit for Transmitter

When the SCI is configured for 9-bit data (M = 1), T8 may be thought of as a ninth transmit data bit to the left of the MSB of the data in the SCIxD register. When writing 9-bit data, the entire 9-bit value is transferred to the SCI shift register after SCIxD is written so T8 should be written (if it needs to change from its previous value) before SCIxD is written. If T8 does not need to change in the new value (such as when it is used to generate mark or space parity), it need not be written each time SCIxD is written.

TXDIR — TxD1 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 TxD1 pin.

1 = TxD1 pin is an output in single-wire mode.

0 = TxD1 pin is an input in single-wire mode.

ORIE — Overrun Interrupt Enable

This bit enables the overrun flag (OR) to generate hardware interrupt requests.

1 = Hardware interrupt requested when OR = 1.

0 = OR interrupts disabled (use polling).

NEIE — Noise Error Interrupt Enable

This bit enables the noise flag (NF) to generate hardware interrupt requests.
1 = Hardware interrupt requested when NF = 1.
0 = NF interrupts disabled (use polling).

FEIE — Framing Error Interrupt Enable

This bit enables the framing error flag (FE) to generate hardware interrupt requests.
1 = Hardware interrupt requested when FE = 1.
0 = FE interrupts disabled (use polling).

PEIE — Parity Error Interrupt Enable

This bit enables the parity error flag (PF) to generate hardware interrupt requests.
1 = Hardware interrupt requested when PF = 1.
0 = PF interrupts disabled (use polling).

11.10.7 SCI x Data Register (SCIxD)

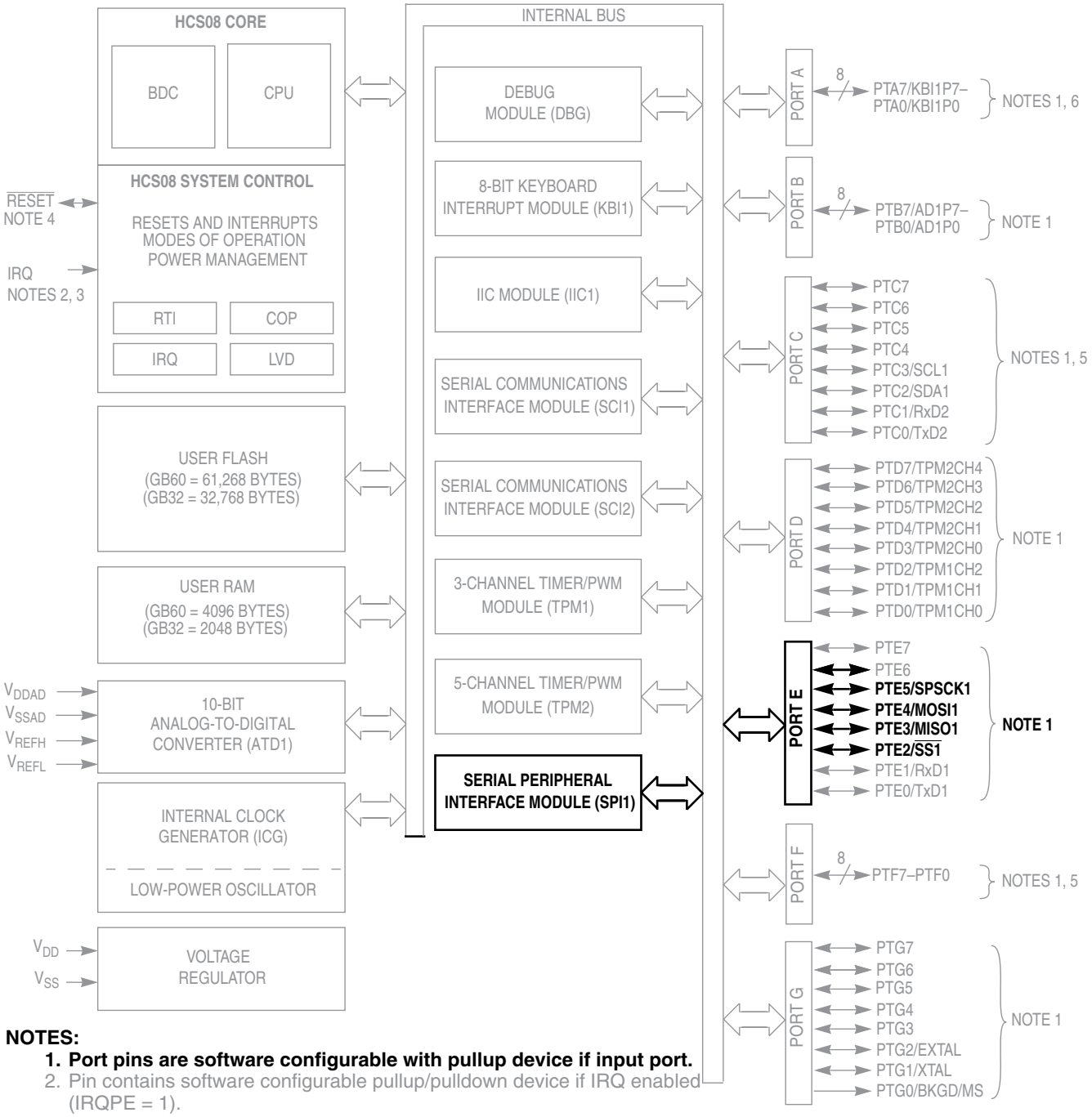
This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for the SCI status flags.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R7	R6	R5	R4	R3	R2	R1	R0
Write:	T7	T6	T5	T4	T3	T2	T1	T0
Reset:	0	0	0	0	0	0	0	0

Figure 11-12. SCI x Data Register (SCIxD)

Chapter 12 Serial Peripheral Interface (SPI) Module

The MC9S08GB/GT provides one serial peripheral interface (SPI) module. The four pins associated with SPI functionality are shared with port E pins 2–5. See the Appendix A, “Electrical Characteristics,” appendix for SPI electrical parametric information. When the SPI is enabled, the direction of pins is controlled by module configuration. If the SPI is disabled, all four pins can be used as general-purpose I/O.



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. High current drive
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).

Figure 12-1. Block Diagram Highlighting the SPI Module

12.1 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

12.2 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.

12.2.1 SPI System Block Diagram

Figure 12-2 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 MOSI1 pin) to the slave while simultaneously shifting data in (on the MISO1 pin) from the slave. The transfer effectively exchanges the data that was in the SPI shift registers of the two SPI systems. The SPSCK1 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 ($\overline{SS1}$ pin). In this system, the master device has configured its $\overline{SS1}$ pin as an optional slave select output.

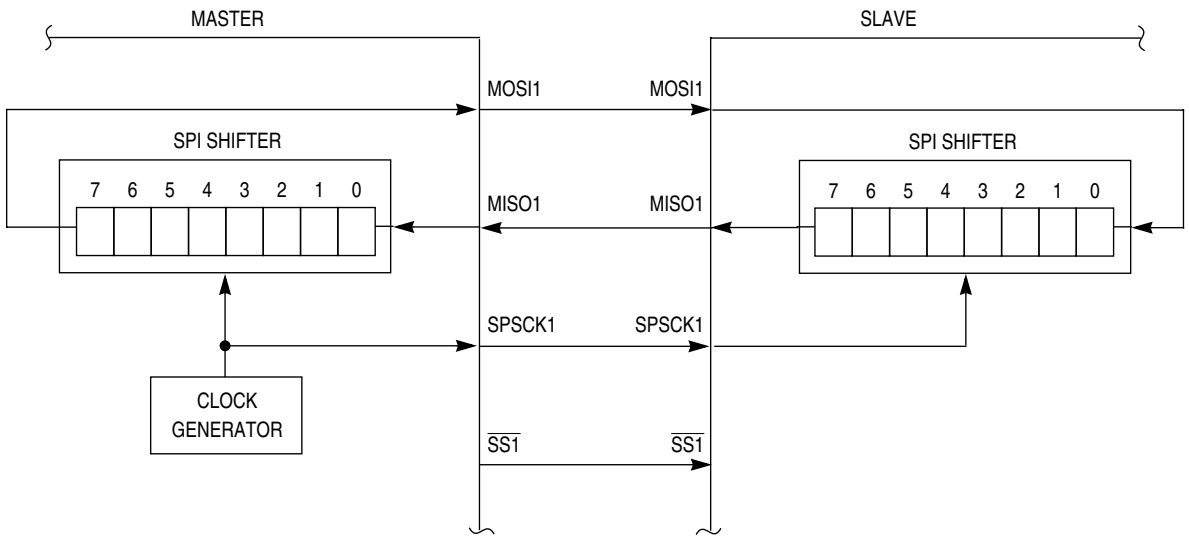


Figure 12-2. SPI System Connections

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 12-2 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.

12.2.2 SPI Module Block Diagram

Figure 12-3 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 SPI1D) 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 SPI1D). 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 SPSCK1 pin, the shifter output is routed to MOSI1, and the shifter input is routed from the MISO1 pin.

When the SPI is configured as a slave, the SPSCK1 pin is routed to the clock input of the SPI, the shifter output is routed to MISO1, and the shifter input is routed from the MOSI1 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.

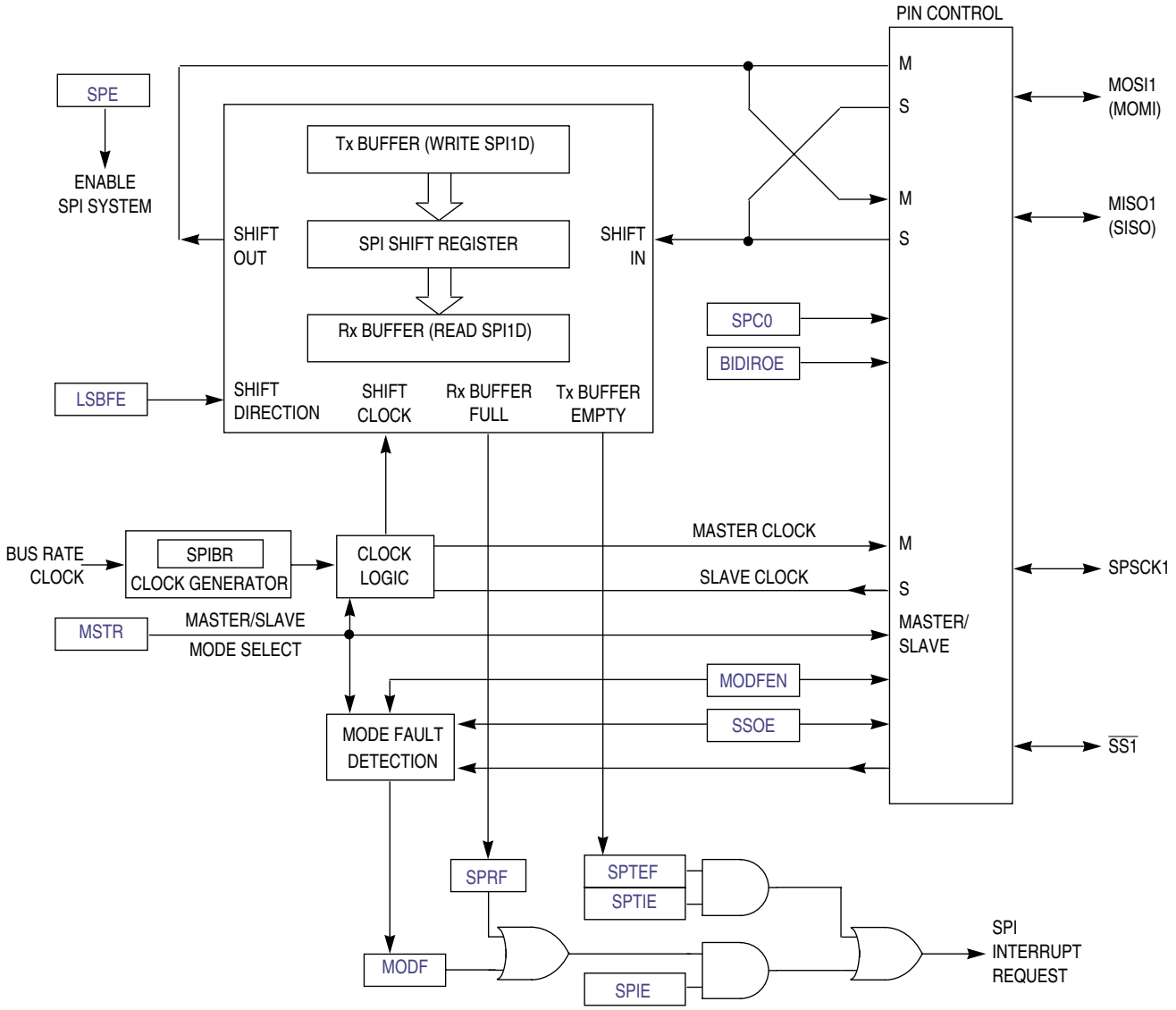


Figure 12-3. SPI Module Block Diagram

12.2.3 SPI Baud Rate Generation

As shown in Figure 12-4, 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.

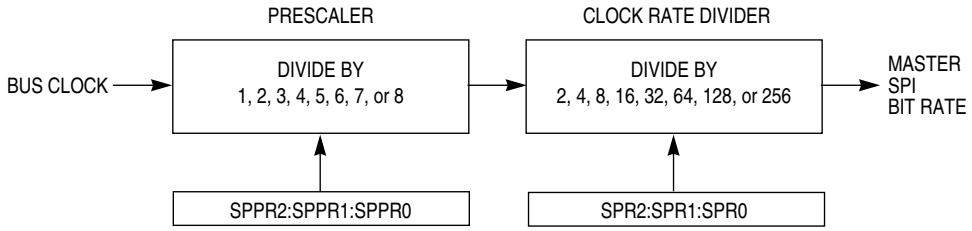


Figure 12-4. SPI Baud Rate Generation

12.3 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 (SPI1D) 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 MISO1 pin at one SPSCK edge and shifted, changing the bit value on the MOSI1 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 MOSI1 pin to the slave while eight bits of data were shifted in the MISO1 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 SPI1D. 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 $\overline{SS1}$ pin must be driven low before a transfer starts and $\overline{SS1}$ must stay low throughout the transfer. If a clock format where CPHA = 0 is selected, $\overline{SS1}$ must be driven to a logic 1 between successive transfers. If CPHA = 1, $\overline{SS1}$ may remain low between successive transfers. See Section 12.3.1, "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 SPI1D) 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.

12.3.1 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 12-5 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.

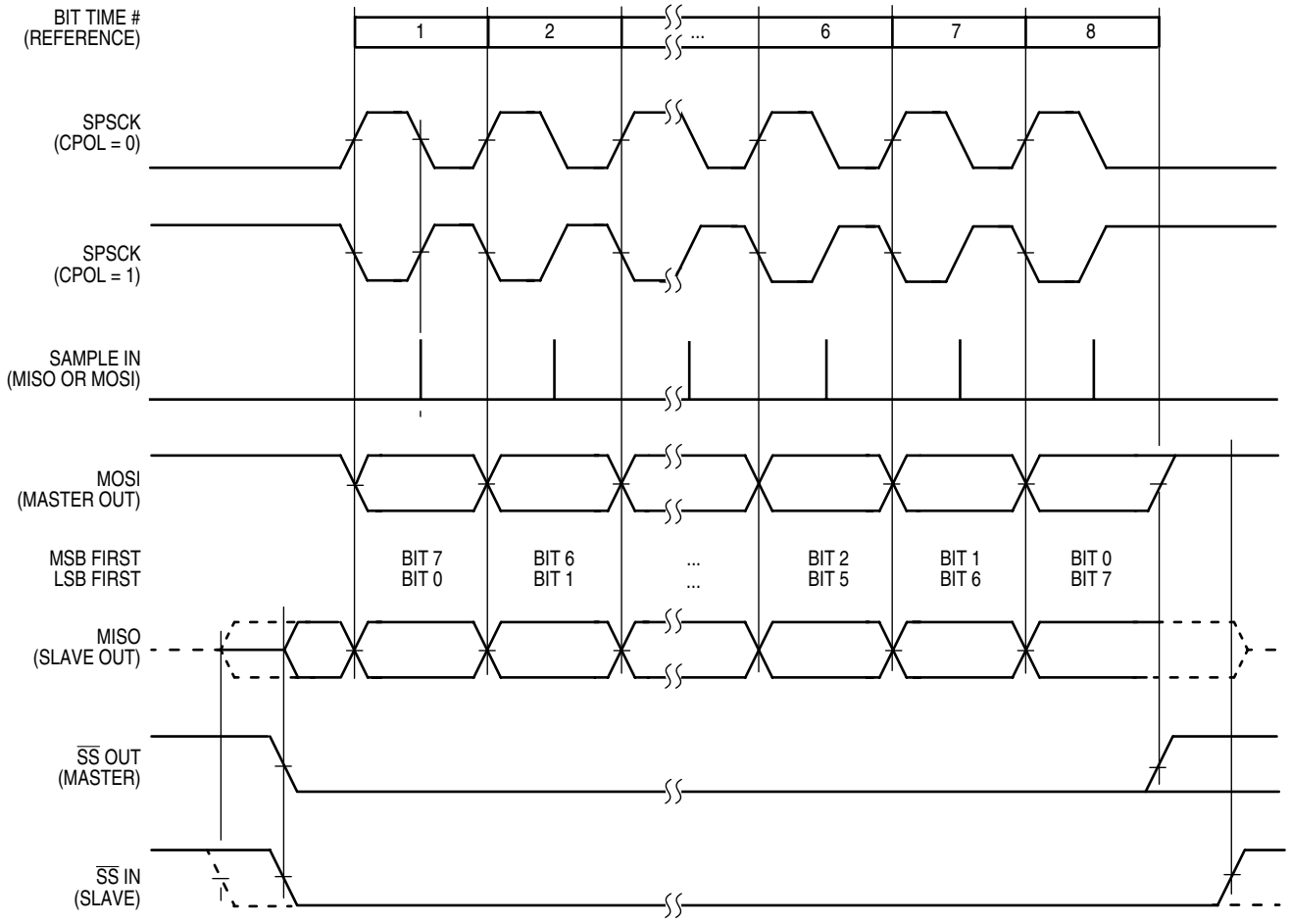


Figure 12-5. SPI Clock Formats (CPHA = 1)

When CPHA = 1, the slave begins to drive its MISO output when $\overline{SS1}$ 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 12-6 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.

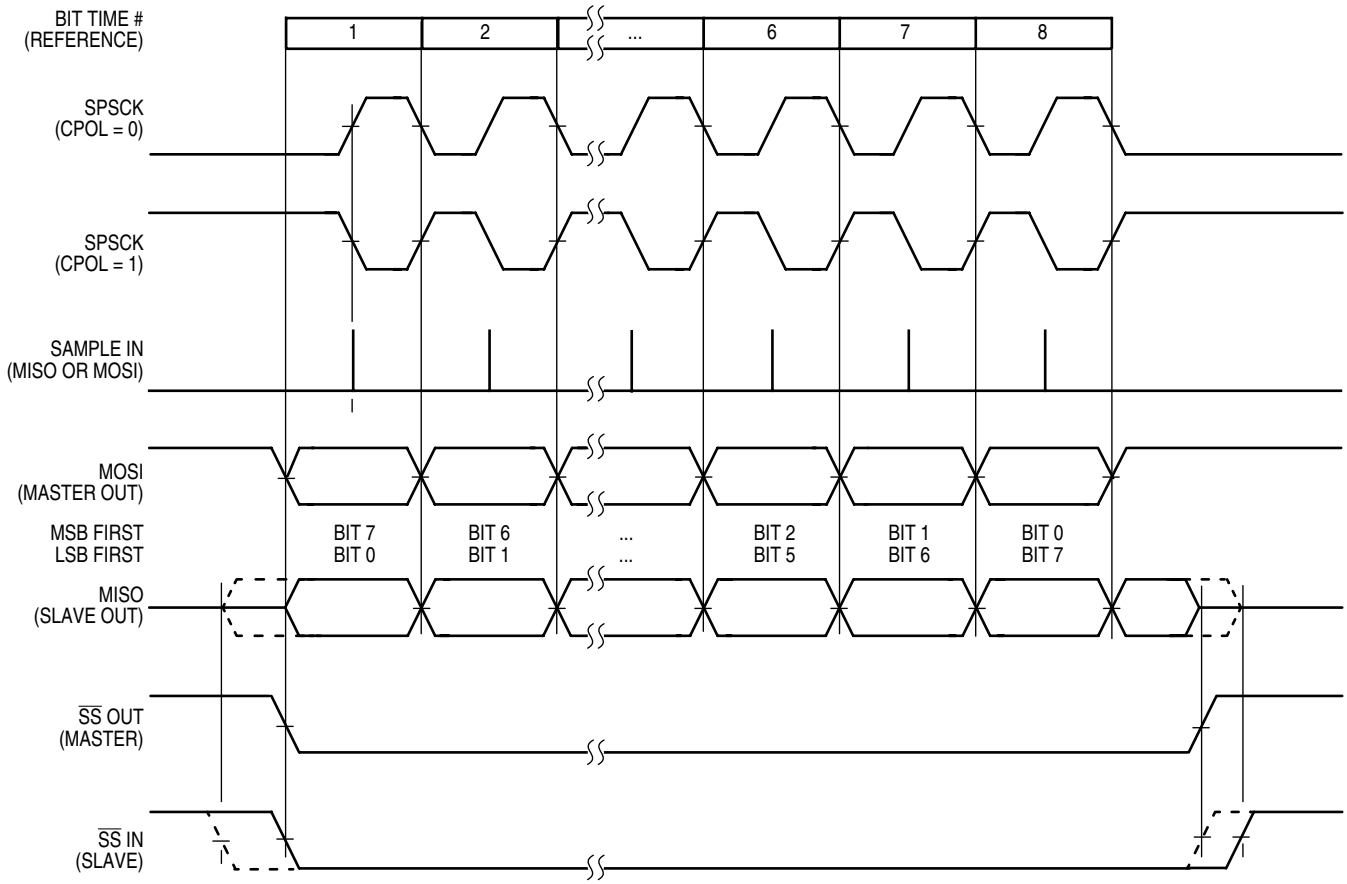


Figure 12-6. SPI Clock Formats (CPHA = 0)

When CPHA = 0, the slave begins to drive its MISO output with the first data bit value (MSB or LSB depending on LSBFE) when $\overline{SS1}$ 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.

12.3.2 SPI Pin Controls

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.

12.3.2.1 SPSCK1 — 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.

12.3.2.2 MOSI1 — 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.

12.3.2.3 MISO1 — 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.

12.3.2.4 $\overline{SS1}$ — 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).

12.3.3 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).

12.3.4 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 $\overline{SS1}$ pin (provided the $\overline{SS1}$ pin is configured as the mode fault input signal). The $\overline{SS1}$ 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 $\overline{SS1}$ 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 SPSCK1, MOSI1, and MISO1 (if not bidirectional mode) are disabled.

MODF is cleared by reading it while it is set, then writing to the SPI control register 1 (SPI1C1). User software should verify the error condition has been corrected before changing the SPI back to master mode.

12.4 SPI Registers and Control Bits

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.

12.4.1 SPI Control Register 1 (SPI1C1)

This read/write register includes the SPI enable control, interrupt enables, and configuration options.

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
Reset:	0	0	0	0	0	1	0	0

Figure 12-7. SPI Control Register 1 (SPI1C1)

SPIE — SPI Interrupt Enable (for SPRF and MODF)

This is the interrupt enable for SPI receive buffer full (SPRF) and mode fault (MODF) events.
1 = When SPRF or MODF is 1, request a hardware interrupt.
0 = Interrupts from SPRF and MODF inhibited (use polling).

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.
1 = SPI system enabled.
0 = SPI system inactive.

SPTIE — SPI Transmit Interrupt Enable

This is the interrupt enable bit for SPI transmit buffer empty (SPTEF).
1 = When SPTEF is 1, hardware interrupt requested.
0 = Interrupts from SPTEF inhibited (use polling).

MSTR — Master/Slave Mode Select
1 = SPI module configured as a master SPI device.
0 = SPI module configured as a slave SPI device.

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 12.3.1, “SPI Clock Formats,” for more details.
1 = Active-low SPI clock (idles high).
0 = Active-high SPI clock (idles low).

CPHA — Clock Phase

This bit selects one of two clock formats for different kinds of synchronous serial peripheral devices. Refer to Section 12.3.1, “SPI Clock Formats,” for more details.
1 = First edge on SPSCK occurs at the start of the first cycle of an 8-cycle data transfer.
0 = First edge on SPSCK occurs at the middle of the first cycle of an 8-cycle data transfer.

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 $\overline{SS1}$ pin as shown in Table 12-1.

Table 12-1. $\overline{SS1}$ 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

LSBFE — LSB First (Shifter Direction)

1 = SPI serial data transfers start with least significant bit.
0 = SPI serial data transfers start with most significant bit.

12.4.2 SPI Control Register 2 (SPI1C2)

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.

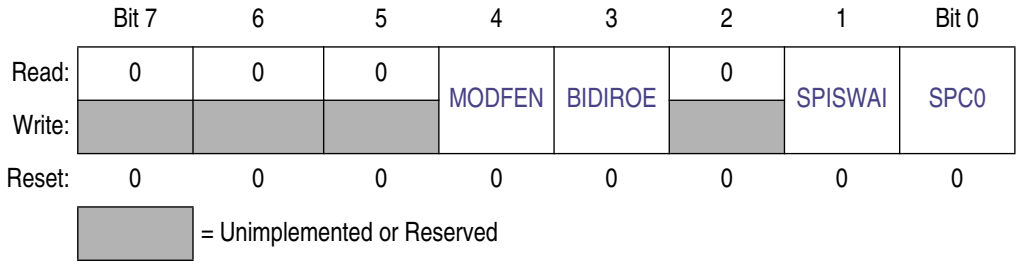


Figure 12-8. SPI Control Register 2 (SPI1C2)

MODFEN — Master Mode-Fault Function Enable

When the SPI is configured for slave mode, this bit has no meaning or effect. (The $\overline{SS1}$ pin is the slave select input.) In master mode, this bit determines how the $\overline{SS1}$ pin is used (refer to Table 12-1 for more details).

1 = Mode fault function enabled, master $\overline{SS1}$ pin acts as the mode fault input or the slave select output.
0 = Mode fault function disabled, master $\overline{SS1}$ pin reverts to general-purpose I/O not controlled by SPI.

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 MOSI1 (MOMI) or MISO1 (SISO) pin, respectively, as the single SPI data I/O pin. When SPC0 = 0, BIDIROE has no meaning or effect.

1 = SPI I/O pin enabled as an output.
0 = Output driver disabled so SPI data I/O pin acts as an input.

SPISWAI — SPI Stop in Wait Mode

1 = SPI clocks stop when the MCU enters wait mode.
0 = SPI clocks continue to operate in wait mode.

SPC0 — SPI Pin Control 0

The SPC0 bit chooses single-wire bidirectional mode. If MSTR = 0 (slave mode), the SPI uses the MISO1 (SISO) pin for bidirectional SPI data transfers. If MSTR = 1 (master mode), the SPI uses the MOSI1 (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.

1 = SPI configured for single-wire bidirectional operation.
0 = SPI uses separate pins for data input and data output.

12.4.3 SPI Baud Rate Register (SPI1BR)

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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-9. SPI Baud Rate Register (SPI1BR)

SPPR2:SPPR1:SPPR0 — SPI Baud Rate Prescale Divisor

This 3-bit field selects one of eight divisors for the SPI baud rate prescaler as shown in Table 12-2. 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 12-4).

Table 12-2. 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

SPR2:SPR1:SPR0 — SPI Baud Rate Divisor

This 3-bit field selects one of eight divisors for the SPI baud rate divider as shown in Figure 12-3. The input to this divider comes from the SPI baud rate prescaler (see Figure 12-4). The output of this divider is the SPI bit rate clock for master mode.

Table 12-3. 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

12.4.4 SPI Status Register (SPI1S)

This register has three read-only status bits. Bits 6, 3, 2, 1, and 0 are not implemented and always read 0s. Writes have no meaning or effect.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SPRF	0	SPTEF	MODF	0	0	0	0
Write:								
Reset:	0	0	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-10. SPI Status Register (SPI1S)

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 (SPI1D). SPRF is cleared by reading SPRF while it is set, then reading the SPI data register.

1 = Data available in the receive data buffer.
0 = No data available in the receive data buffer.

SPTEF — SPI Transmit Buffer Empty Flag

This bit is set when there is room in the transmit data buffer. It is cleared by reading SPI1S with SPTEF set, followed by writing a data value to the transmit buffer at SPI1D. SPI1S must be read with SPTEF = 1 before writing data to SPI1D or the SPI1D write will be ignored. SPTEF generates an SPTEF CPU interrupt request if the SPTIE bit in the SPI1C1 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 SPI1D 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.

1 = SPI transmit buffer empty.
0 = SPI transmit buffer not empty.

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 $\overline{SS1}$ 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 (SPI1C1).

1 = Mode fault error detected.
0 = No mode fault error.

12.4.5 SPI Data Register (SPI1D)

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 12-11. SPI Data Register (SPI1D)

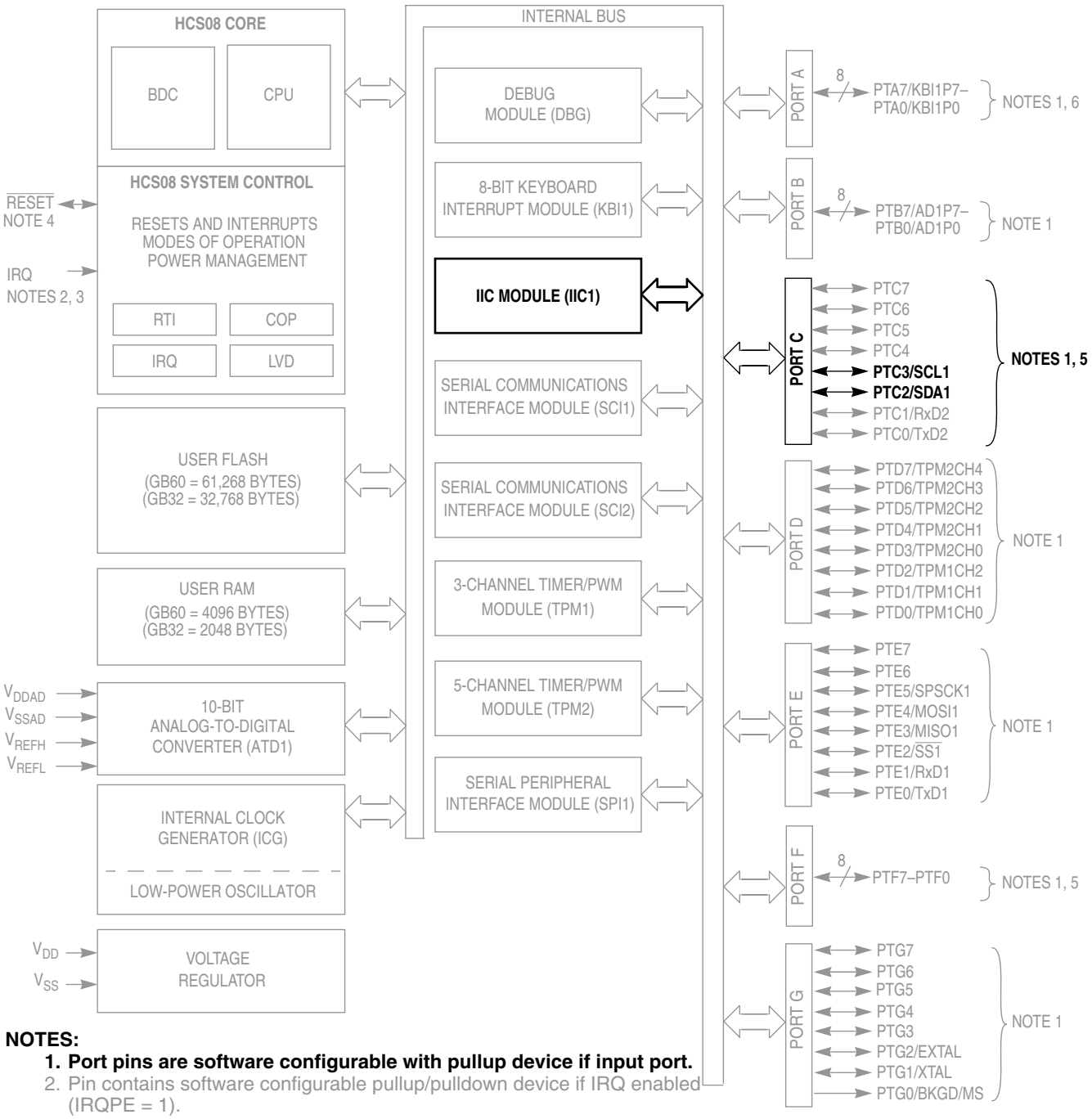
Reads of this register return the data read from the receive data buffer. Writes to this register write data to the transmit data buffer. When the SPI is configured as a master, writing data to the transmit data buffer initiates an SPI transfer.

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 SPI1D 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.

Chapter 13 Inter-Integrated Circuit (IIC) Module

The MC9S08GB/GT series of microcontrollers provides one inter-integrated circuit (IIC) module for communication with other integrated circuits. The two pins associated with this module, SDA1 and SCL1 share port C pins 2 and 3, respectively. All functionality as described in this section is available on MC9S08GB/GT. When the IIC is enabled, the direction of pins is controlled by module configuration. If the IIC is disabled, both pins can be used as general-purpose I/O.



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. **High current drive**
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).

Figure 13-1. Block Diagram Highlighting the IIC Module

13.1 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.

For additional detail, please refer to volume 1 of the *HCS08 Reference Manual,* (Freescale Semiconductor document order number HCS08RMv1/D).

13.1.1 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

13.1.2 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. Stop1 and stop2 will reset the register contents.

13.1.3 Block Diagram

Figure 13-2 is a block diagram of the IIC.

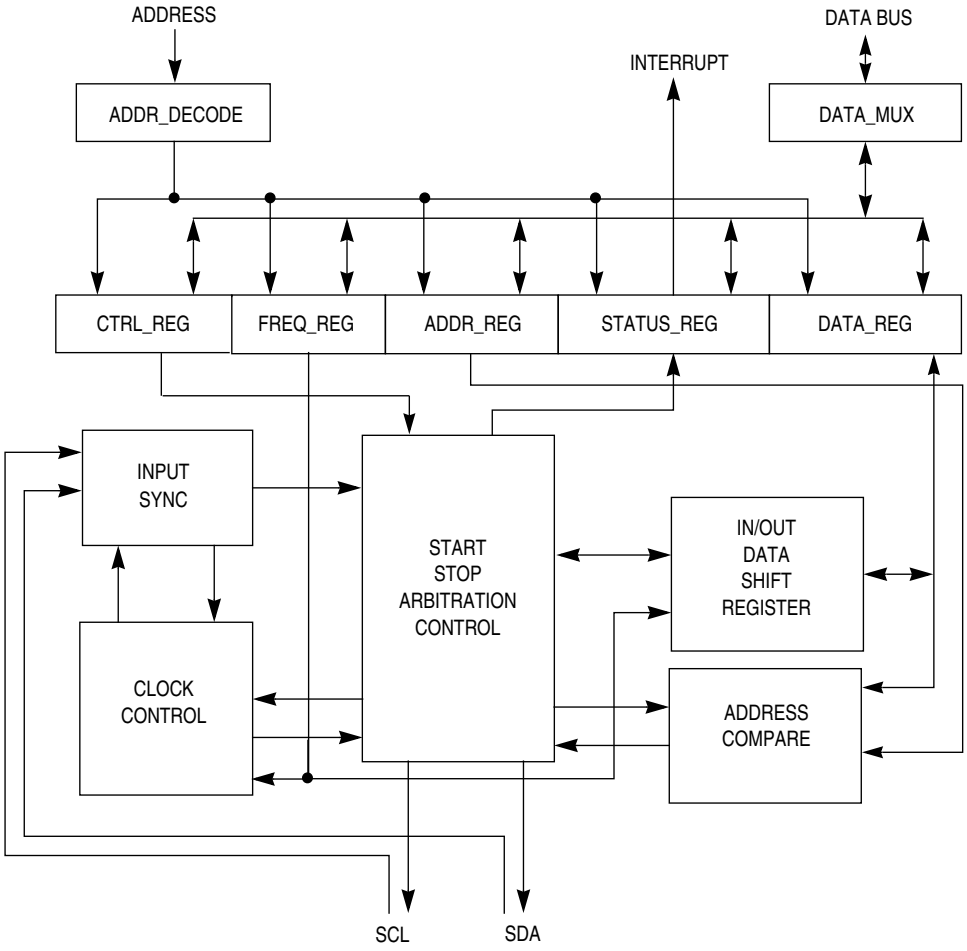


Figure 13-2. IIC Functional Block Diagram

13.1.4 Detailed Signal Descriptions

This section describes each user-accessible pin signal.

13.1.4.1 SCL1 — Serial Clock Line

The bidirectional SCL1 is the serial clock line of the IIC system.

13.1.4.2 SDA1 — Serial Data Line

The bidirectional SDA1 is the serial data line of the IIC system

13.2 Functional Description

This section provides a complete functional description of the IIC module.

13.2.1 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 13-3.

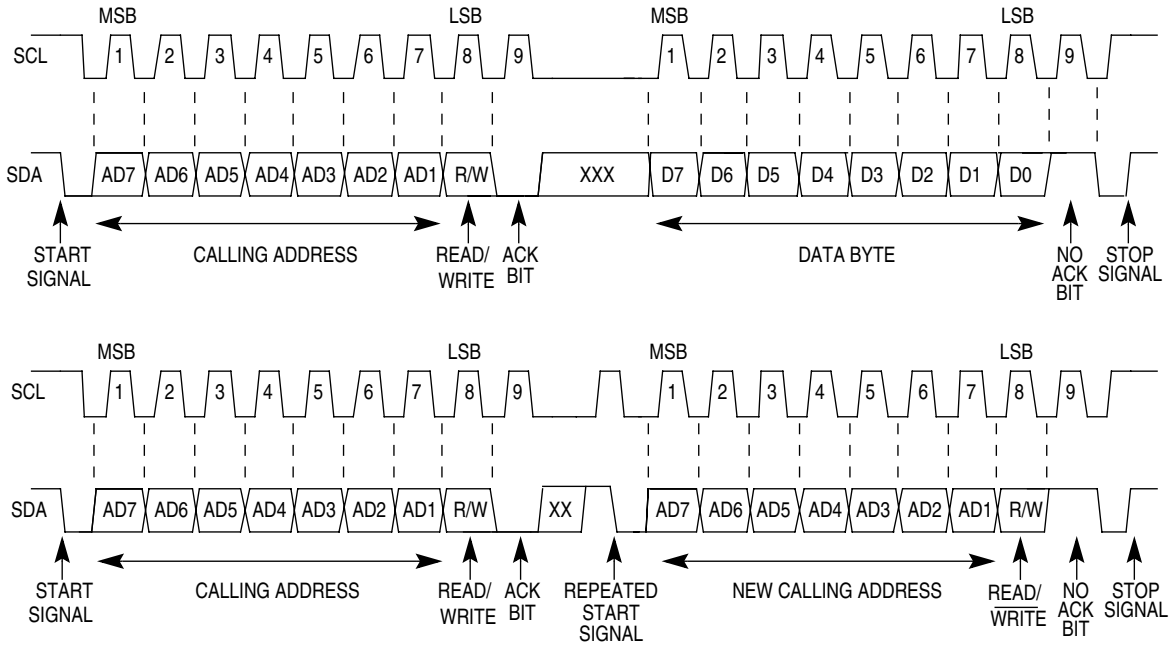


Figure 13-3. IIC Bus Transmission Signals

13.2.1.1 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 13-3, 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.

13.2.1.2 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 13-3).

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.

13.2.1.3 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 13-3. 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.

13.2.1.4 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 13-3).

The master can generate a STOP even if the slave has generated an acknowledge at which point the slave must release the bus.

13.2.1.5 Repeated START Signal

As shown in Figure 13-3, 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.

13.2.1.6 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.

13.2.1.7 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 13-4). 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.

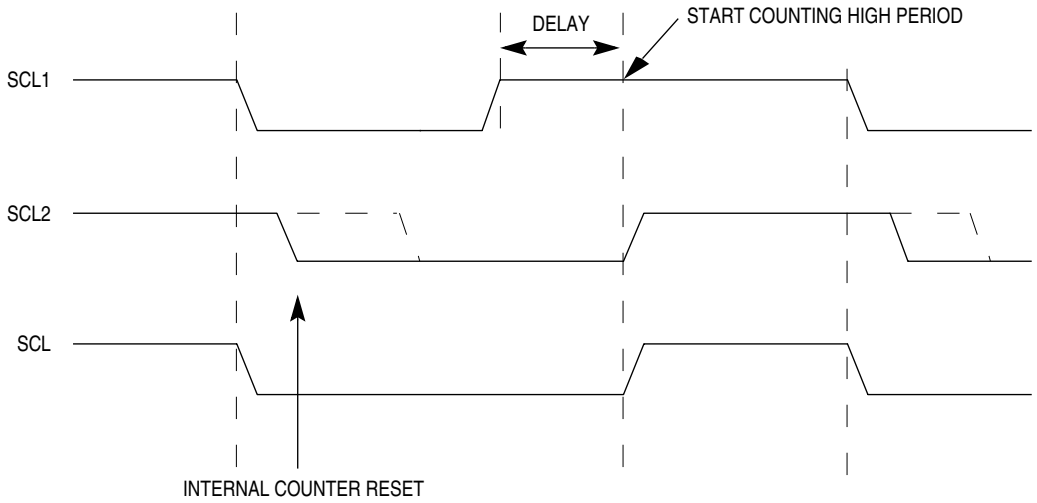


Figure 13-4. IIC Clock Synchronization

13.2.1.8 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.

13.2.1.9 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.

13.3 Resets

The IIC is disabled after reset. The IIC cannot cause an MCU reset.

13.4 Interrupts

The IIC generates a single interrupt.

An interrupt from the IIC is generated when any of the events in Table 13-1 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 13-1. 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

13.4.1 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.

13.4.2 Address Detect Interrupt

When its own specific address (IIC address register) is matched with the calling address, the IAAS bit in 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.

13.4.3 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.

13.5 IIC Registers and Control Bits

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.

13.5.1 IIC Address Register (IIC1A)

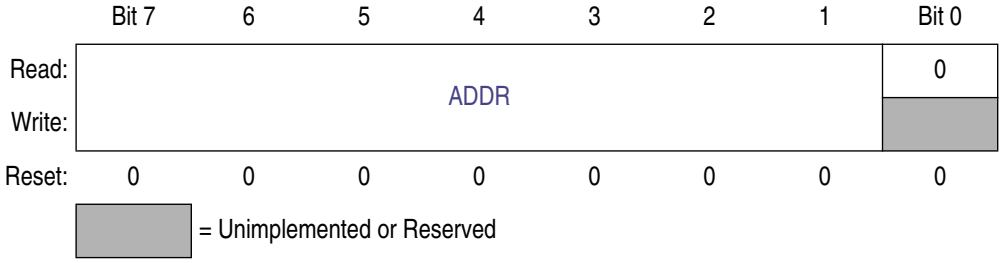


Figure 13-5. IIC Address Register (IIC1A)

ADDR — 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.

13.5.2 IIC Frequency Divider Register (IIC1F)

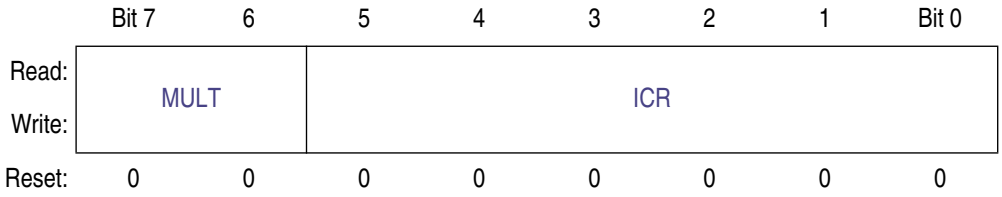


Figure 13-6. IIC Frequency Divider Register (IIC1F)

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. Table 13-2 provides the multiplier factor mul as defined by the MULT bits.

Table 13-2. Multiplier Factor

MULT	mul
00	01
01	02
10	04
11	Reserved

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 13-3 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 13-3 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 13-3. 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

13.5.3 IIC Control Register (IIC1C)

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	IICEN	IICIE	MST	TX	TXAK	0	0	0
Write:						RSTA		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 13-7. IIC Control Register (IIC1C)

IICEN — IIC Enable

The IICEN bit determines whether the IIC module is enabled.

1 = IIC is enabled.
0 = IIC is not enabled.

IICIE — IIC Interrupt Enable

The IICIE bit determines whether an IIC interrupt is requested.

1 = IIC interrupt request enabled.
0 = IIC interrupt request not enabled.

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.

1 = Master Mode.
0 = Slave Mode.

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.

1 = Transmit.
0 = Receive.

TXAK — Transmit Acknowledge Enable

This bit specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers.

1 = No acknowledge signal response is sent.
0 = An acknowledge signal will be sent out to the bus after receiving one data byte.

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.

13.5.4 IIC Status Register (IIC1S)

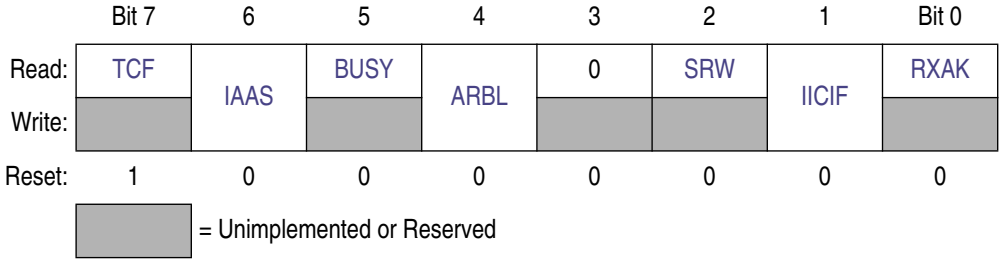


Figure 13-8. IIC Status Register (IIC1S)

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 IIC1D register in receive mode or writing to the IIC1D in transmit mode.

1 = Transfer complete.
0 = Transfer in progress.

IAAS — Addressed as a Slave

The IAAS bit is set when its own specific address is matched with the calling address. Writing the IIC1C register clears this bit.

1 = Addressed as a slave.
0 = Not addressed.

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.

1 = Bus is busy.
0 = Bus is idle.

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.

1 = Loss of arbitration.
0 = Standard bus operation.

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.

1 = Slave transmit, master reading from slave.
0 = Slave receive, master writing to slave.

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

1 = Interrupt pending.
0 = No interrupt pending.

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.

1 = No acknowledge received.
0 = Acknowledge received.

13.5.5 IIC Data I/O Register (IIC1D)

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DATA							
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 13-9. IIC Data I/O Register (IIC1D)

DATA — Data

In master transmit mode, when data is written to the IIC1D, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates receiving of the next byte of data.

NOTE

When transitioning out of master receive mode, the IIC mode should be switched before reading the IIC1D 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.

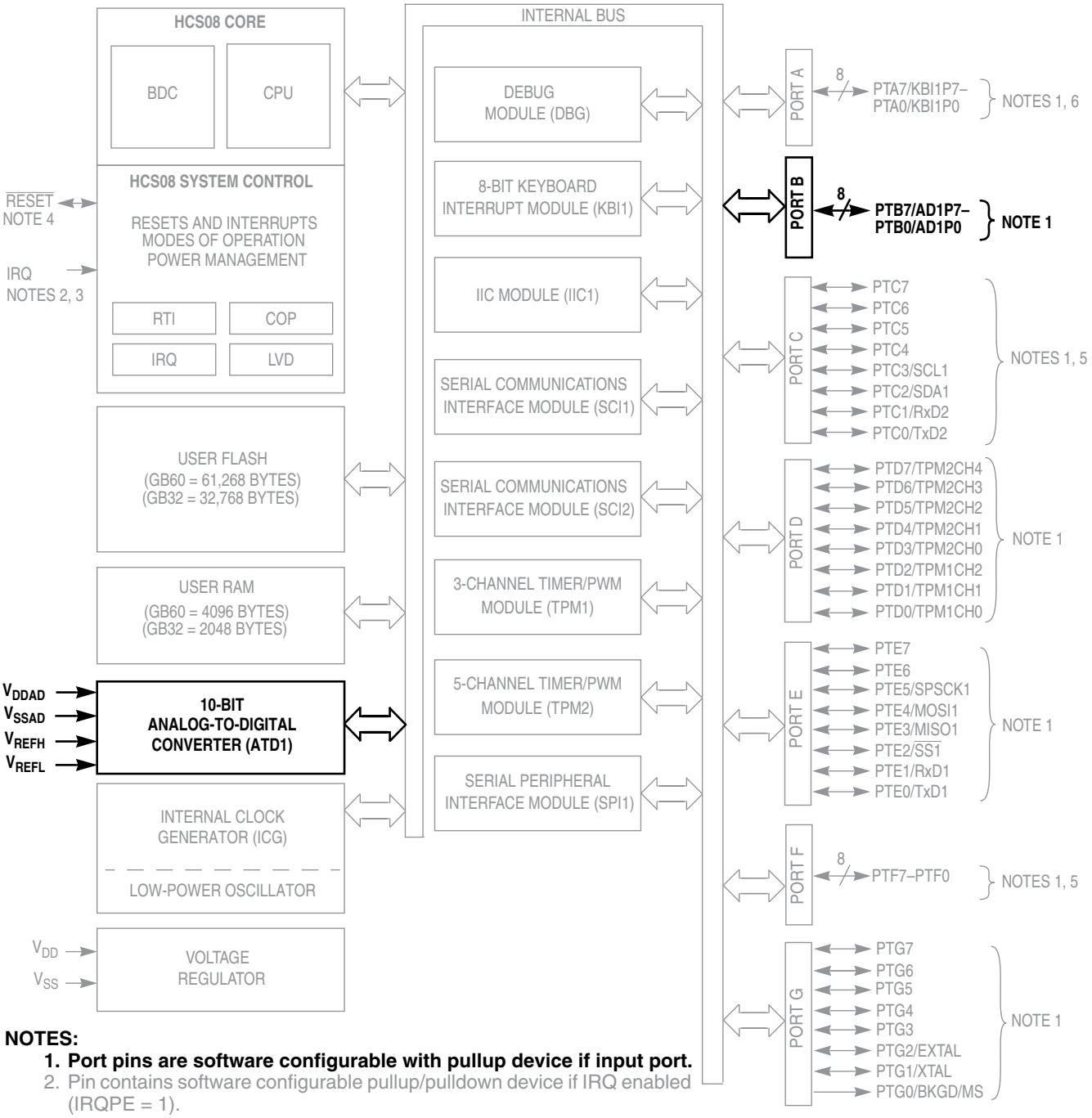
Note that the TX bit in IIC1C must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For instance, if the IIC is configured for master transmit but a master receive is desired, then reading the IIC1D will not initiate the receive.

Reading the IIC1D will return the last byte received while the IIC is configured in either master receive or slave receive modes. The IIC1D does not reflect every byte that is transmitted on the IIC bus, nor can software verify that a byte has been written to the IIC1D correctly by reading it back.

In master transmit mode, the first byte of data written to IIC1D 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).

Chapter 14 Analog-to-Digital Converter (ATD) Module

The MC9S08GB/GT provides one 8-channel analog-to-digital (ATD) module. The eight ATD channels share port B. Each channel individually can be configured for general-purpose I/O or for ATD functionality. All features of the ATD module as described in this section are available on the MC9S08GB/GT. Electrical parametric information for the ATD may be found in Appendix A, “Electrical Characteristics.”



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. High current drive
6. Pins PTA[7:4] contain both pullup and pulldown devices. Pulldown available when KBI enabled (KBIPn = 1).

Figure 14-1. MC9S08GBxx Block Diagram Highlighting ATD Block and Pins

14.1 Introduction

The ATD module is an analog-to-digital converter with a successive approximation register (SAR) architecture with sample and hold.

14.1.1 Features

- 8-/10-bit resolution
- 14.0 µsec, 10-bit single conversion time at a conversion frequency of 2 MHz
- Left-/right-justified result data
- Left-justified signed data mode
- Conversion complete flag or conversion complete interrupt generation
- Analog input multiplexer for up to eight analog input channels
- Single or continuous conversion mode

14.1.2 Modes of Operation

The ATD has two modes for low power

- Stop mode
- Power-down mode

14.1.2.1 Stop Mode

When the MCU goes into stop mode, the MCU stops the clocks and the ATD analog circuitry is turned off, placing the module into a low-power state. Once in stop mode, the ATD module aborts any single or continuous conversion in progress. Upon exiting stop mode, no conversions occur and the registers have their previous values. As long as the ATDPU bit is set prior to entering stop mode, the module is reactivated coming out of stop.

14.1.2.2 Power Down Mode

Clearing the ATDPU bit in register ATD1C also places the ATD module in a low-power state. The ATD conversion clock is disabled and the analog circuitry is turned off, placing the module in power-down mode. (This mode does not remove power to the ATD module.) Once in power-down mode, the ATD module aborts any conversion in progress. Upon setting the ATDPU bit, the module is reactivated. During power-down mode, the ATD registers are still accessible.

Note that the reset state of the ATDPU bit is zero. Therefore, the module is reset into the power-down state.

14.1.3 Block Diagram

Figure 14-2 illustrates the functional structure of the ATD module.

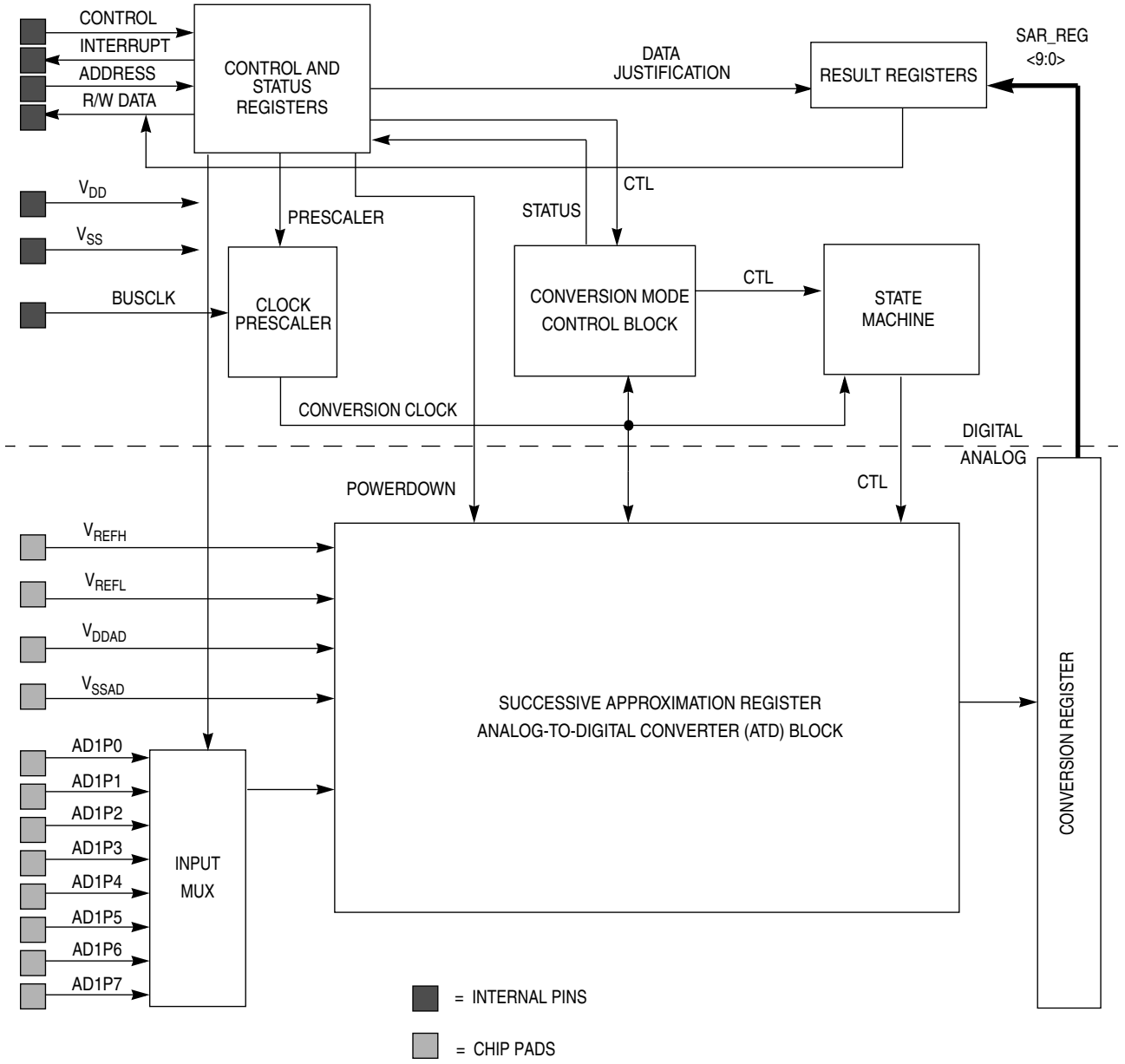


Figure 14-2. ATD Block Diagram

14.2 Signal Description

14.2.1 Overview

The ATD supports eight input channels and requires 4 supply/reference/ground pins. These pins are listed in Table 14-1.

Table 14-1. Signal Properties

Name	Function
AD7–AD0	Channel input pins
V_{REFH}	High reference voltage for ATD converter
V_{REFL}	Low reference voltage for ATD converter
V_{DDAD}	ATD power supply voltage
V_{SSAD}	ATD ground supply voltage

14.2.1.1 Channel Input Pins — AD1P7–AD1P0

The channel pins are used as the analog input pins of the ATD. Each pin is connected to an analog switch which serves as the signal gate into the sample submodule.

14.2.1.2 ATD Reference Pins — V_{REFH}, V_{REFL}

These pins serve as the source for the high and low reference potentials for the converter. Separation from the power supply pins accommodates the filtering necessary to achieve the accuracy of which the system is capable.

14.2.1.3 ATD Supply Pins — V_{DDAD}, V_{SSAD}

These two pins are used to supply power and ground to the analog section of the ATD. Dedicated power is required to isolate the sensitive analog circuitry from the normal levels of noise present on digital power supplies.

NOTE

V_{DDAD1} and V_{DD} must be at the same potential. Likewise, V_{SSAD1} and V_{SS} must be at the same potential.

14.3 Functional Description

The ATD uses a successive approximation register (SAR) architecture. The ATD contains all the necessary elements to perform a single analog-to-digital conversion.

A write to the ATD1SC register initiates a new conversion. A write to the ATD1C register will interrupt the current conversion but it will not initiate a new conversion. A write to the ATD1PE register will also abort the current conversion but will not initiate a new conversion. If a conversion is already running when a write to the ATD1SC register is made, it will be aborted and a new one will be started.

14.3.1 Mode Control

The ATD has a mode control unit to communicate with the sample and hold (S/H) machine and the SAR machine when necessary to collect samples and perform conversions. The mode control unit signals the S/H machine to begin collecting a sample and for the SAR machine to begin receiving a sample. At the end of the sample period, the S/H machine signals the SAR machine to begin the analog-to-digital conversion process. The conversion process is terminated when the SAR machine signals the end of

conversion to the mode control unit. For V_{REFL} and V_{REFH}, the SAR machine uses the reference potentials to set the sampled signal level within itself without relying on the S/H machine to deliver them.

The mode control unit organizes the conversion, specifies the input sample channel, and moves the digital output data from the SAR register to the result register. The result register consists of a dual-port register. The SAR register writes data into the register through one port while the module data bus reads data out of the register through the other port.

14.3.2 Sample and Hold

The S/H machine accepts analog signals and stores them as capacitor charge on a storage node located in the SAR machine. Only one sample can be held at a time so the S/H machine and the SAR machine can not run concurrently even though they are independent machines. Figure 14-3 shows the placement of the various resistors and capacitors.

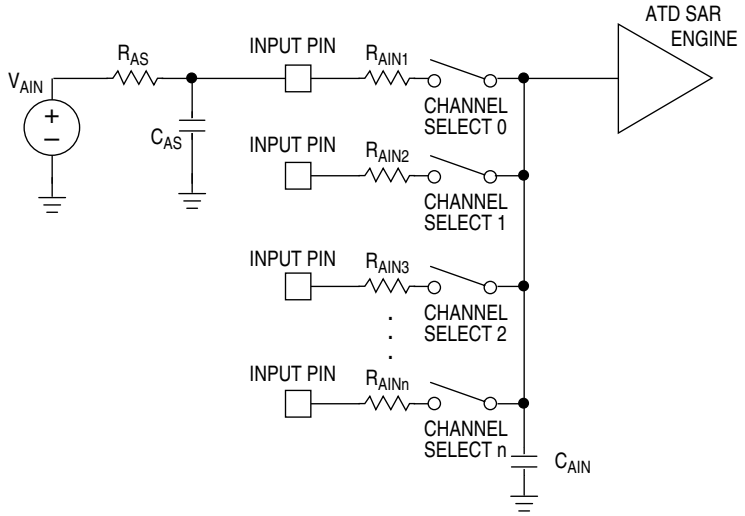


Figure 14-3. Resistor and Capacitor Placement

When the S/H machine is not sampling, it disables its own internal clocks.The input analog signals are unipolar. The signals must fall within the potential range of V_{SSAD} to V_{DDAD}. The S/H machine is not required to perform special conversions (i.e., convert V_{REFL} and V_{REFH}).

Proper sampling is dependent on the following factors:

- Analog source impedance (the real portion, R_{AS} – see Appendix A, “Electrical Characteristics”) — This is the resistive (or real, in the case of high frequencies) portion of the network driving the analog input voltage V_{AIN}.
- Analog source capacitance (C_{AS}) — This is the filtering capacitance on the analog input, which (if large enough) may help the analog source network charge the ATD input in the case of high R_{AS}.
- ATD input resistance (R_{AIN} – maximum value 7 kΩ) — This is the internal resistance of the ATD circuit in the path between the external ATD input and the ATD sample and hold circuit. This resistance varies with temperature, voltage, and process variation but a worst case number is necessary to compute worst case sample error.

- ATD input capacitance (C_{AIN} – maximum value 50 pF) — This is the internal capacitance of the ATD sample and hold circuit. This capacitance varies with temperature, voltage, and process variation but a worst case number is necessary to compute worst case sample error.
- ATD conversion clock frequency (f_{ATDCLK} – maximum value 2 MHz) — This is the frequency of the clock input to the ATD and is dependent on the bus clock frequency and the ATD prescaler. This frequency determines the width of the sample window, which is 14 ATDCLK cycles.
- Input sample frequency (f_{SAMP} – see Appendix A, "Electrical Characteristics") — This is the frequency that a given input is sampled.
- Delta-input sample voltage (ΔV_{SAMP}) — This is the difference between the current input voltage (intended for conversion) and the previously sampled voltage (which may be from a different channel). In non-continuous convert mode, this is assumed to be the greater of ($V_{REFH} - V_{AIN}$) and ($V_{AIN} - V_{REFL}$). In continuous convert mode, 5 LSB should be added to the known difference to account for leakage and other losses.
- Delta-analog input voltage (ΔV_{AIN}) — This is the difference between the current input voltage and the input voltage during the last conversion on a given channel. This is based on the slew rate of the input.

In cases where there is no external filtering capacitance, the sampling error is determined by the number of time constants of charging and the change in input voltage relative to the resolution of the ATD:

$$\text{\# of time constants } (\tau) = (14 / f_{ATDCLK}) / ((R_{AS} + R_{AIN}) * C_{AIN}) \qquad \textit{Eqn. 14-1}$$

$$\text{sampling error in LSB } (E_S) = 2^N * (\Delta V_{SAMP} / (V_{REFH} - V_{REFL})) * e^{-\tau}$$

The maximum sampling error (assuming maximum change on the input voltage) will be:

$$E_S = (3.6/3.6) * e^{-(14/((7\text{ k} + 10\text{ k}) * 50\text{ p} * 2\text{ M}))} * 1024 = 0.271 \text{ LSB} \qquad \textit{Eqn. 14-2}$$

In the case where an external filtering capacitance is applied, the sampling error can be reduced based on the size of the source capacitor (C_{AS}) relative to the analog input capacitance (C_{AIN}). Ignoring the analog source impedance (R_{AS}), C_{AS} will charge C_{AIN} to a value of:

$$E_S = 2^N * (\Delta V_{SAMP} / (V_{REFH} - V_{REFL})) * (C_{AIN} / (C_{AIN} + C_{AS})) \qquad \textit{Eqn. 14-3}$$

In the case of a 0.1 µF C_{AS}, a worst case sampling error of 0.5 LSB is achieved regardless of R_{AS}. However, in the case of repeated conversions at a rate of f_{SAMP}, R_{AS} must re-charge C_{AS}. This recharge is continuous and controlled only by R_{AS} (not R_{AIN}), and reduces the overall sampling error to:

$$E_S = 2^N * \{(\Delta V_{AIN} / (V_{REFH} - V_{REFL})) * e^{-(1 / (f_{SAMP} * R_{AS} * C_{AS}))} + (\Delta V_{SAMP} / (V_{REFH} - V_{REFL})) * \text{Min}[(C_{AIN} / (C_{AIN} + C_{AS})), e^{-(1 / (f_{ATDCLK} * (R_{AS} + R_{AIN}) * C_{AIN}))}]\} \qquad \textit{Eqn. 14-4}$$

This is a worst case sampling error which does not account for R_{AS} recharging the combination of C_{AS} and C_{AIN} during the sample window. It does illustrate that high values of R_{AS} (>10 kΩ) are possible if a large C_{AS} is used and sufficient time to recharge C_{AS} is provided between samples. In order to achieve accuracy specified under the worst case conditions of maximum ΔV_{SAMP} and minimum C_{AS}, R_{AS} must be less than the maximum value of 10 kΩ. The maximum value of 10 kΩ for R_{AS} is to ensure low sampling error in the worst case condition of maximum ΔV_{SAMP} and minimum C_{AS}.

14.3.3 Analog Input Multiplexer

The analog input multiplexer selects one of the eight external analog input channels to generate an analog sample. The analog input multiplexer includes negative stress protection circuitry which prevents cross-talk between channels when the applied input potentials are within specification. Only analog input signals within the potential range of V_{REFL} to V_{REFH} (ATD reference potentials) will result in valid ATD conversions.

14.3.4 ATD Module Accuracy Definitions

Figure 14-4 illustrates an ideal ATD transfer function. The horizontal axis represents the ATD input voltage in millivolts. The vertical axis the conversion result code. The ATD is specified with the following figures of merit:

- Number of bits (N) — The number of bits in the digitized output
- Resolution (LSB) — The resolution of the ATD is the step size of the ideal transfer function. This is also referred to as the ideal code width, or the difference between the transition voltages to a given code and to the next code. This unit, known as 1LSB, is equal to

$$1LSB = (V_{REFH} - V_{REFL}) / 2^N \qquad \textit{Eqn. 14-5}$$

- Inherent quantization error (E_Q) — This is the error caused by the division of the perfect ideal straight-line transfer function into the quantized ideal transfer function with 2^N steps. This error is ± 1/2 LSB.
- Differential non-linearity (DNL) — This is the difference between the current code width and the ideal code width (1LSB). The current code width is the difference in the transition voltages to the current code and to the next code. A negative DNL means the transfer function spends less time at the current code than ideal; a positive DNL, more. The DNL cannot be less than –1.0; a DNL of greater than 1.0 reduces the effective number of bits by 1.
- Integral non-linearity (INL) — This is the difference between the transition voltage to the current code and the transition to the corresponding code on the adjusted transfer curve. INL is a measure of how straight the line is (how far it deviates from a straight line). The adjusted ideal transition voltage is:

Eqn. 14-6

$$\text{Adjusted Ideal Trans. V} = \frac{(\text{Current Code} - 1/2)}{2^N} * ((V_{REFH} + E_{FS}) - (V_{REFL} + E_{ZS}))$$

- Zero scale error (E_{ZS}) — This is the difference between the transition voltage to the first valid code and the ideal transition to that code. Normally, it is defined as the difference between the actual and ideal transition to code $001, but in some cases the first transition may be to a higher code. The ideal transition to any code is:

Eqn. 14-7

$$\text{Ideal Transition V} = \frac{(\text{Current Code} - 1/2)}{2^N} *(V_{REFH} - V_{REFL})$$

- Full scale error (E_{FS}) — This is the difference between the transition voltage to the last valid code and the ideal transition to that code. Normally, it is defined as the difference between the actual and ideal transition to code $3FF, but in some cases the last transition may be to a lower code. The ideal transition to any code is:

Eqn. 14-8

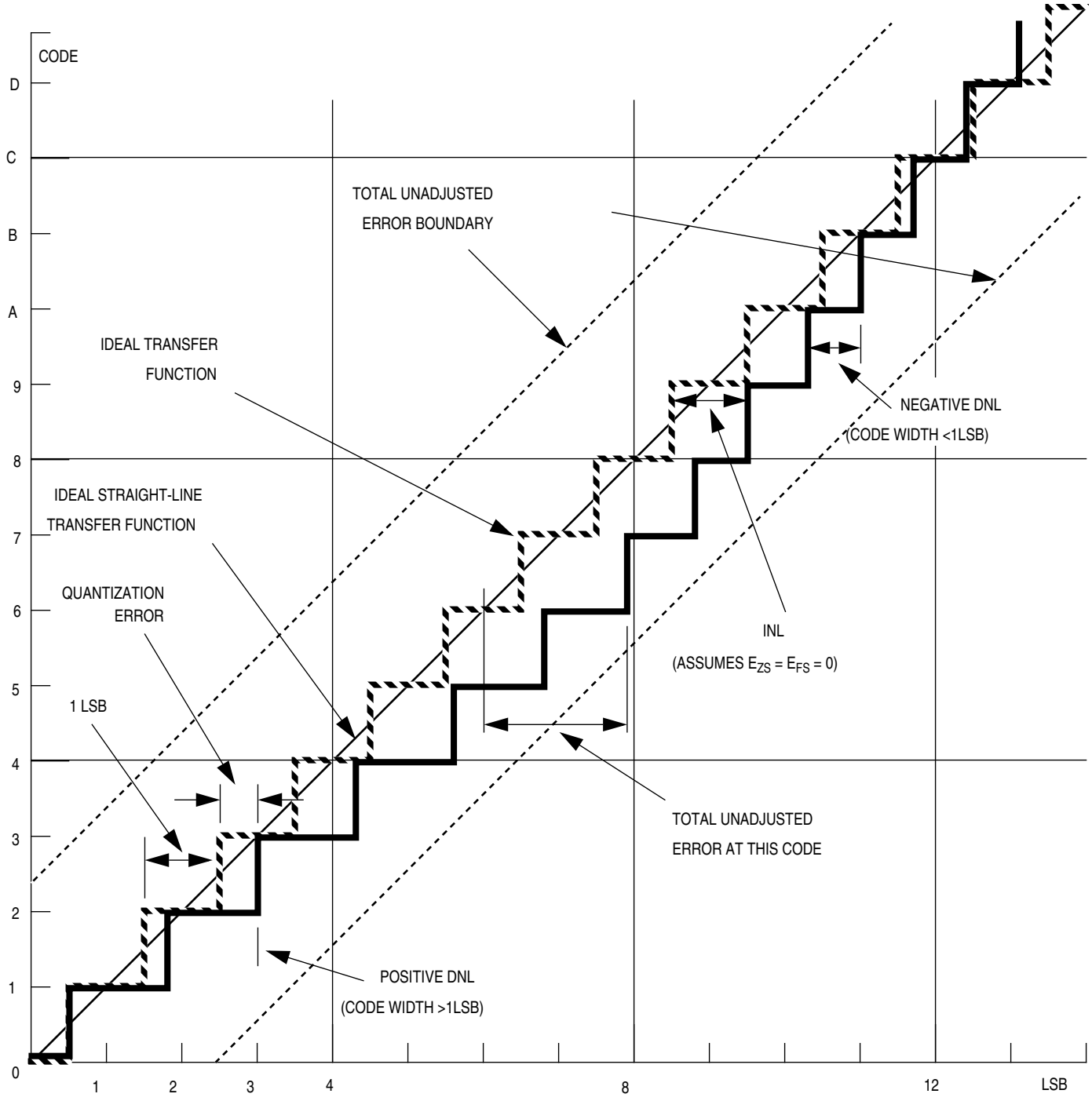
$$\text{Ideal Transition V} = \frac{(\text{Current Code} - 1/2)}{2^N} * (V_{REFH} - V_{REFL})$$

- Total unadjusted error (E_{TU}) — This is the difference between the transition voltage to a given code and the ideal straight-line transfer function. An alternate definition (with the same result) is the difference between the actual transfer function and the ideal straight-line transfer function. This measure of error includes inherent quantization error and all forms of circuit error (INL, DNL, zero-scale, and full-scale) except input leakage error, which is not due to the ATD.
- Input leakage error (E_{IL}) — This is the error between the transition voltage to the current code and the ideal transition to that code that is the result of input leakage across the real portion of the impedance of the network that drives the analog input. This error is a system-observable error which is not inherent to the ATD, so it is not added to total error. This error is:

$$E_{IL} \text{ (in V)} = \text{input leakage} * R_{AS} \qquad \textit{Eqn. 14-9}$$

There are two other forms of error which are not specified which can also affect ATD accuracy. These are:

- Sampling error (E_S) — The error due to inadequate time to charge the ATD circuitry
- Noise error (E_N) — The error due to noise on V_{AIN}, V_{REFH}, or V_{REFL} due to either direct coupling (noise source capacitively coupled directly on the signal) or power supply (V_{DDAD}, V_{SSAD}, V_{DD}, and V_{SS}) noise interfering with the ATD's ability to resolve the input accurately. The error due to internal sources can be reduced (and specified operation achieved) by operating the ATD conversion in wait mode and ceasing all IO activity. Reducing the error due to external sources is dependent on system activity and board layout.



NOTES: Graph is for example only and may not represent actual performance

Figure 14-4. ATD Accuracy Definitions

14.4 Resets

The ATD module is reset on system reset. If the system reset signal is activated, the ATD registers are initialized back to their reset state and the ATD module is powered down. This occurs as a function of the register file initialization; the reset definition of the ATDPU bit (power down bit) is zero or disabled.

The MCU places the module back into an initialized state. If the module is performing a conversion, the current conversion is terminated, the conversion complete flag is cleared, and the SAR register bits are cleared. Any pending interrupts are also cancelled. Note that the control, test, and status registers are initialized on reset; the initialized register state is defined in the register description section of this specification.

Enabling the module (using the ATDPU bit) does not cause the module to reset since the register file is not initialized. Finally, writing to control register ATD1C does not cause the module to reset; the current conversion will be terminated.

14.5 Interrupts

The ATD module originates interrupt requests and the MCU handles or services these requests. Details on how the ATD interrupt requests are handled can be found in Chapter 5, “Resets, Interrupts, and System Configuration”.

The ATD interrupt function is enabled by setting the ATDIE bit in the ATD1SC register. When the ATDIE bit is set, an interrupt is generated at the end of an ATD conversion and the ATD result registers (ATD1RH and ATD1RL) contain the result data generated by the conversion. If the interrupt function is disabled (ATDIE = 0), then the CCF flag must be polled to determine when a conversion is complete.

The interrupt will remain pending as long as the CCF flag is set. The CCF bit is cleared whenever the ATD status and control (ATD1SC) register is written. The CCF bit is also cleared whenever the ATD result registers (ATD1RH or ATD1RL) are read.

Table 14-2. Interrupt Summary

Interrupt	Local Enable	Description
CCF	ATDIE	Conversion complete

14.6 ATD Registers and Control Bits

The ATD has seven registers which control ATD functions.

Refer to the direct-page register summary in Chapter 4, “Memory” of this data sheet for the absolute address assignments for all ATD 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.

14.6.1 ATD Control (ATDC)

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	ATDPU	DJM	RES8	SGN	PRS			
Reset:	0	0	0	0	0	0	0	0

Figure 14-5. ATD Control Register (ATD1C)

Writes to the ATD control register will abort the current conversion, but will not start a new conversion.

ATDPU — ATD Power Up

This bit provides program on/off control over the ATD, reducing power consumption when the ATD is not being used. When cleared, the ATDPU bit aborts any conversion in progress.

1 = ATD functionality.

0 = Disable the ATD and enter a low-power state.

DJM — Data Justification Mode

This bit determines how the 10-bit conversion result data maps onto the ATD result register bits. When RES8 is set, bit DJM has no effect and the 8-bit result is always located in ATD1RH.

For left-justified mode, result data bits 9–2 map onto bits 7–0 of ATD1RH, result data bits 1 and 0 map onto ATD1RL bits 7 and 6, where bit 7 of ATD1RH is the most significant bit (MSB).

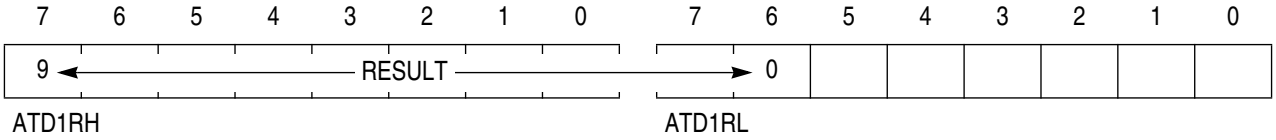


Figure 14-6. Left-Justified Mode

For right-justified mode, result data bits 9 and 8 map onto bits 1 and 0 of ATD1RH, result data bits 7–0 map onto ATD1RL bits 7–0, where bit 1 of ATD1RH is the most significant bit (MSB).

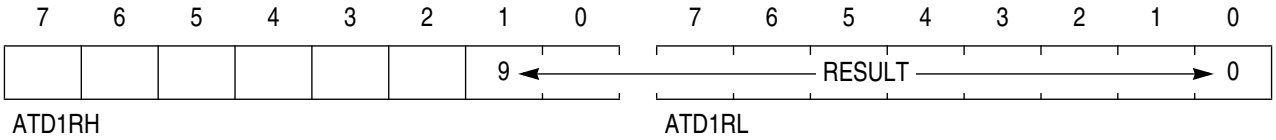


Figure 14-7. Right-Justified Mode

The effect of the DJM bit on the result is shown in Table 14-3.

1 = Result register data is right justified.

0 = Result register data is left justified.

RES8 — ATD Resolution Select

This bit determines the resolution of the ATD converter, 8-bits or 10-bits. The ATD converter has the accuracy of a 10-bit converter. However, if 8-bit compatibility is required, selecting 8-bit resolution will map result data bits 9-2 onto ATD1RH bits 7-0.

The effect of the RES8 bit on the result is shown in Table 14-3.

1 = 8-bit resolution selected.
0 = 10-bit resolution selected.

SGN — Signed Result Select

This bit determines whether the result will be signed or unsigned data. Signed data is represented as 2's complement data and is achieved by complementing the MSB of the result. Signed data mode can be used only when the result is left justified (DJM = 0) and is not available for right-justified mode (DJM = 1). When a signed result is selected, the range for conversions becomes –512 ($200) to 511 ($1FF) for 10-bit resolution and –128 ($80) to 127 ($7F) for 8-bit resolution.

The effect of the SGN bit on the result is shown in Table 14-3.

1 = Left justified result data is signed.
0 = Left justified result data is unsigned.

Table 14-3. Available Result Data Formats

RES8	DJM	SGN	Data Formats of Result	Analog Input $V_{REFH} = V_{DDA}$, $V_{REFL} = V_{SSA}$ ATD1RH:ATD1RL	
				V_{DDA}	V_{SSA}
1	0	0	8-bit : left justified : unsigned	$FF:$00	$00:$00
1	0	1	8-bit : left justified : signed	$7F:$00	$80:$00
1	1	X[1]	8-bit : left justified[2] : unsigned	$FF:$00	$00:$00
0	0	0	10-bit : left justified : unsigned	$FF:$C0	$00:$00
0	0	1	10-bit : left justified : signed	$7F:$C0	$80:$00
0	1	X[1]	10-bit : right justified : unsigned	$03:$FF	$00:$00

1 The SGN bit is only effective when DJM = 0. When DJM = 1, SGN is ignored.
2 8-bit results are always in ATD1RH.

PRS — Prescaler Rate Select

This field of bits determines the prescaled factor for the ATD conversion clock. Table 14-4 illustrates the divide-by operation and the appropriate range of bus clock frequencies.

Table 14-4. Clock Prescaler Values

PRS	Factor = (PRS +1) × 2	Max Bus Clock MHz (2 MHz max ATD Clock)[1]	Max Bus Clock MHz (1 MHz max ATD Clock)[2]	Min Bus Clock[3] MHz (500 kHz min ATD Clock)
0000	2	4	2	1
0001	4	8	4	2
0010	6	12	6	3
0011	8	16	8	4
0100	10	20	10	5
0101	12	20	12	6
0110	14	20	14	7
0111	16	20	16	8
1000	18	20	18	9
1001	20	20	20	10
1010	22	20	20	11
1011	24	20	20	12
1100	26	20	20	13
1101	28	20	20	14
1110	30	20	20	15
1111	32	20	20	16

1 Maximum ATD conversion clock frequency is 2 MHz. The max bus clock frequency is computed from the max ATD conversion clock frequency times the indicated prescaler setting; i.e., for a PRS of 0, max bus clock = 2 (max ATD conversion clock frequency) × 2 (Factor) = 4 MHz.

2 Use these settings if the maximum desired ATD conversion clock frequency is 1 MHz. The max bus clock frequency is computed from the max ATD conversion clock frequency times the indicated prescaler setting; i.e., for a PRS of 0, max bus clock = 1 (max ATD conversion clock frequency) × 2 (Factor) = 2 MHz.

3 Minimum ATD conversion clock frequency is 500 kHz. The min bus clock frequency is computed from the min ATD conversion clock frequency times the indicated prescaler setting; i.e., for a PRS of 1, min bus clock = 0.5 (min ATD conversion clock frequency) × 2 (Factor) = 1 MHz.

14.6.2 ATD Status and Control (ATD1SC)

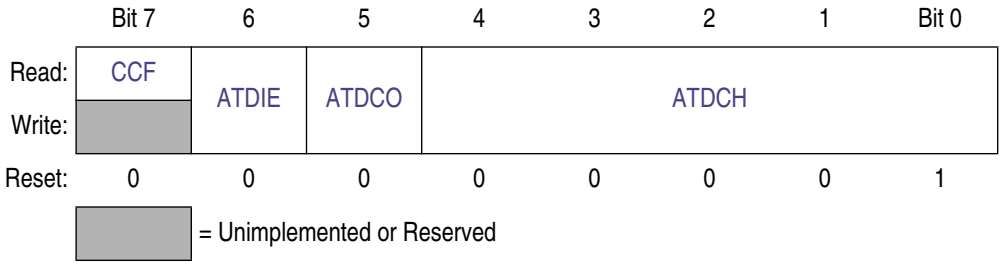


Figure 14-8. ATD Status and Control Register (ATD1SC)

Writes to the ATD status and control register clears the CCF flag, cancels any pending interrupts, and initiates a new conversion.

CCF — Conversion Complete Flag

The CCF is a read-only bit which is set each time a conversion is complete. The CCF bit is cleared whenever the ATD1SC register is written. It is also cleared whenever the result registers, ATD1RH or ATD1RL, are read.

1 = Current conversion is complete.
0 = Current conversion is not complete.

ATDIE — ATD Interrupt Enabled

When this bit is set, an interrupt is generated upon completion of an ATD conversion. At this time, the result registers contain the result data generated by the conversion. The interrupt will remain pending as long as the conversion complete flag CCF is set. If the ATDIE bit is cleared, then the CCF bit must be polled to determine when the conversion is complete. Note that system reset clears pending interrupts.

1 = ATD interrupt enabled.
0 = ATD interrupt disabled.

ATDCO — ATD Continuous Conversion

When this bit is set, the ATD will convert samples continuously and update the result registers at the end of each conversion. When this bit is cleared, only one conversion is completed between writes to the ATD1SC register.

1 = Continuous conversion mode.
0 = Single conversion mode.

ATDCH — Analog Input Channel Select

This field of bits selects the analog input channel whose signal is sampled and converted to digital codes. Table 14-5 lists the coding used to select the various analog input channels.

Table 14-5. Analog Input Channel Select Coding

ATDCH	Analog Input Channel
00	AD0
01	AD1
02	AD2
03	AD3
04	AD4
05	AD5
06	AD6
07	AD7
08–1D	Reserved (default to V_{REFL})
1E	V_{REFH}
1F	V_{REFL}

14.6.3 ATD Result Data (ATD1RH, ATD1RL)

For left-justified mode, result data bits 9–2 map onto bits 7–0 of ATD1RH, result data bits 1 and 0 map onto ATD1RL bits 7 and 6, where bit 7 of ATD1RH is the most significant bit (MSB).

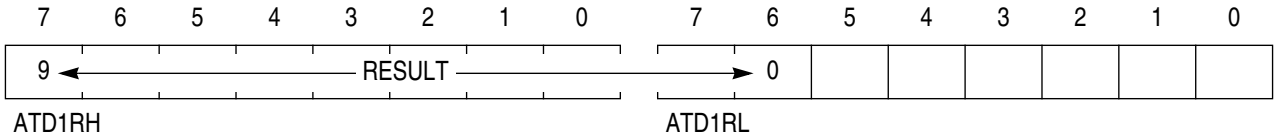


Figure 14-9. Left-Justified Mode

For right-justified mode, result data bits 9 and 8 map onto bits 1 and 0 of ATD1RH, result data bits 7–0 map onto ATD1RL bits 7–0, where bit 1 of ATD1RH is the most significant bit (MSB).

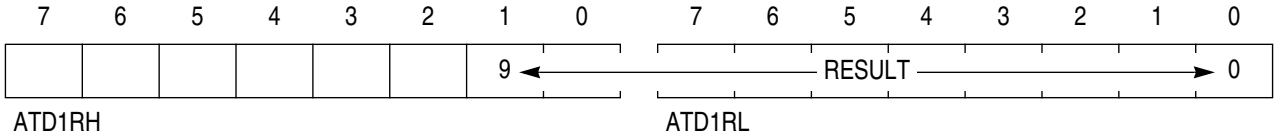


Figure 14-10. Right-Justified Mode

The ATD 10-bit conversion results are stored in two 8-bit result registers, ATD1RH and ATD1RL. The result data is formatted either left or right justified where the format is selected using the DJM control bit in the ATD1C register. The 10-bit result data is mapped either between ATD1RH bits 7–0 and ATD1RL bits 7–6 (left justified), or ATD1RH bits 1–0 and ATD1RL bits 7–0 (right justified).

For 8-bit conversions, the 8-bit result is always located in ATD1RH bits 7–0, and the ATD1RL bits read 0. For 10-bit conversions, the six unused bits always read 0.

The ATD1RH and ATD1RL registers are read-only.

14.6.4 ATD Pin Enable (ATD1PE)

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	ATDPE7	ATDPE6	ATDPE5	ATDPE4	ATDPE3	ATDPE2	ATDPE1	ATDPE0
Reset:	0	0	0	0	0	0	0	0

Figure 14-11. ATD Pin Enable Register (ATD1PE)

The ATD pin enable register allows the pins dedicated to the ATD module to be configured for ATD usage. A write to this register will abort the current conversion but will not initiate a new conversion. If the ATDPEx bit is 0 (disabled for ATD usage) but the corresponding analog input channel is selected via the ATDCH bits, the ATD will not convert the analog input but will instead convert V_{REFL} placing zeroes in the ATD result registers.

ATDPE7:ATDPE0 — ATD Pin 7–0 Enables

1 = Pin enabled for ATD usage.
0 = Pin disabled for ATD usage.

Chapter 15 Development Support

15.1 Introduction

Development support systems in the HCS08 include the background debug controller (BDC) and the on-chip debug module (DBG). The BDC provides a single-wire debug interface to the target MCU that provides a convenient interface for programming the on-chip FLASH and other nonvolatile memories. The BDC is also the primary debug interface for development and allows non-intrusive access to memory data and traditional debug features such as CPU register modify, breakpoints, and single instruction trace commands.

In the HCS08 Family, address and data bus signals are not available on external pins (not even in test modes). Debug is done through commands fed into the target MCU via the single-wire background debug interface. The debug module provides a means to selectively trigger and capture bus information so an external development system can reconstruct what happened inside the MCU on a cycle-by-cycle basis without having external access to the address and data signals.

The alternate BDC clock source for MC9S08GB/GT is the ICGLCLK. See Chapter 7, “Internal Clock Generator (ICG) Module,” for more information about ICGCLK and how to select clock sources.

15.2 Features

Features of the background debug controller (BDC) 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 debug module (DBG) 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:
  - A-only
  - A OR B
  - A then B
  - A AND B data (full mode)
  - A AND NOT B data (full mode)
  - Event-only B (store data)
  - A then event-only B (store data)
  - Inside range ($A \leq address \leq B$)
  - Outside range ($address < A$ or $address > B$)

15.3 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.

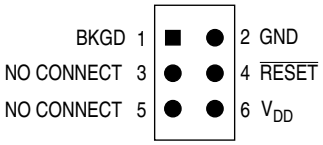


Figure 15-1. BDM Tool Connector

15.3.1 BKGD Pin Description

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

BDC serial communications use a custom serial protocol first introduced on the M68HC12 Family of microcontrollers. This protocol assumes the host knows the communication clock rate that is determined by the target BDC clock rate. All communication is initiated and controlled by the host that drives a high-to-low edge to signal the beginning of each bit time. Commands and data are sent most significant bit

first (MSB first). For a detailed description of the communications protocol, refer to Section 15.3.2, "Communication Details."

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

BKGD is a pseudo-open-drain pin and there is an on-chip pullup so no external pullup resistor is required. Unlike typical open-drain pins, the external RC time constant on this pin, which is influenced by external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speedup pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Section 15.3.2, "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 development system is connected, it can pull both BKGD and $\overline{\text{RESET}}$ low, release $\overline{\text{RESET}}$ to select active background mode rather than normal operating mode, then release BKGD. It is not necessary to reset the target MCU to communicate with it through the background debug interface.

15.3.2 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 15-2 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.

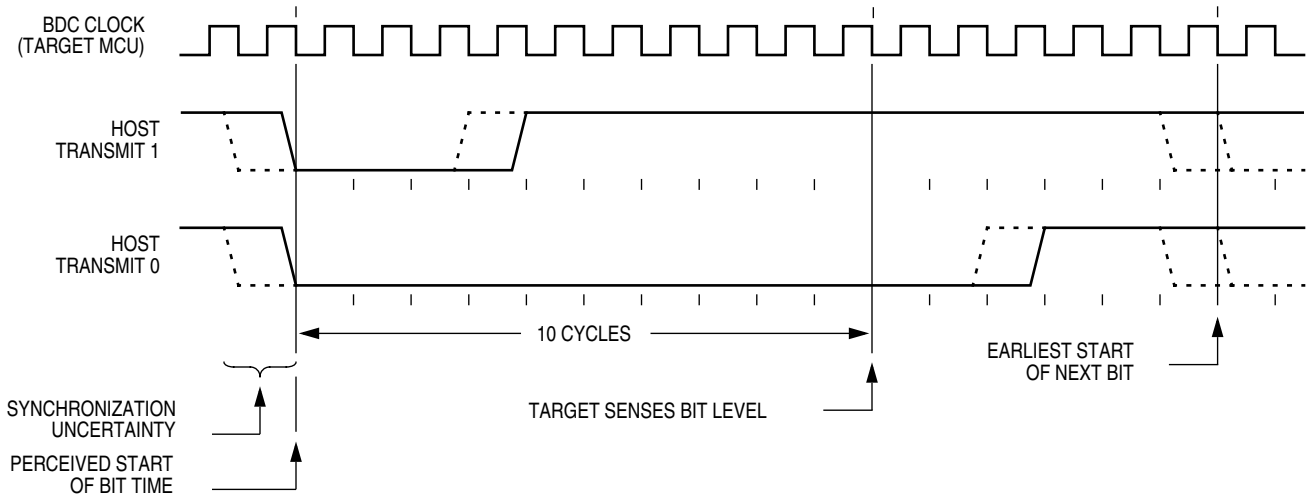


Figure 15-2. BDC Host-to-Target Serial Bit Timing

Figure 15-3 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.

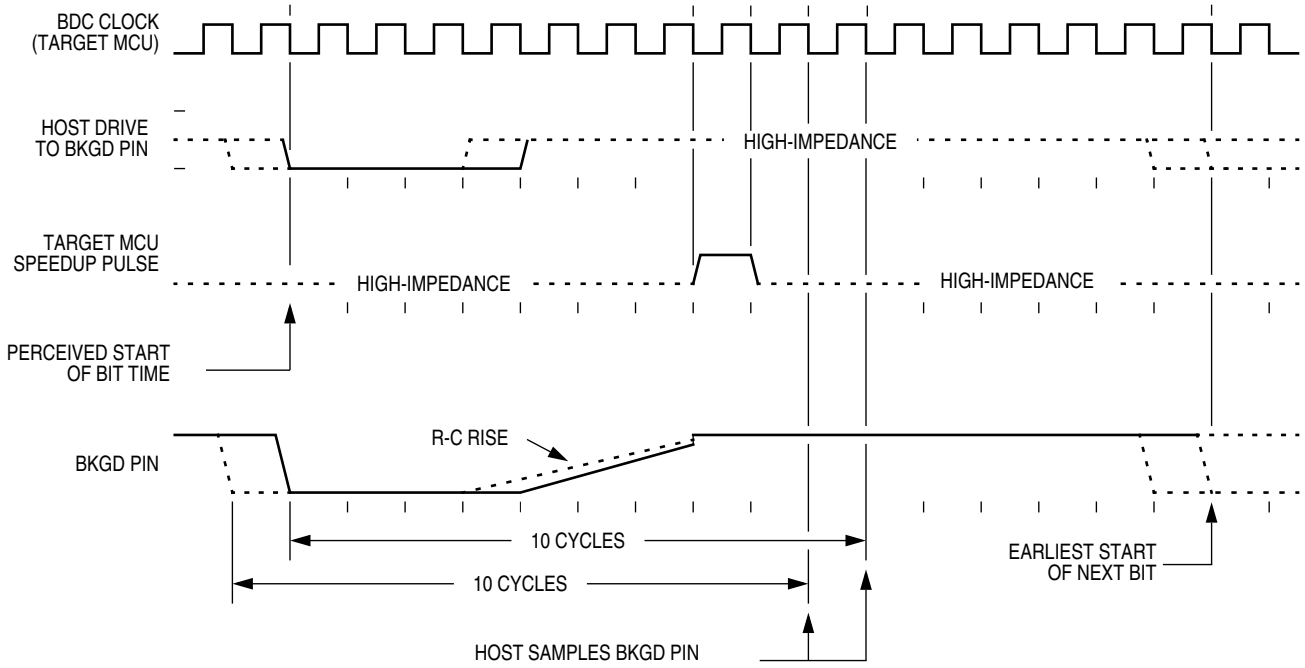


Figure 15-3. BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 15-4 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.

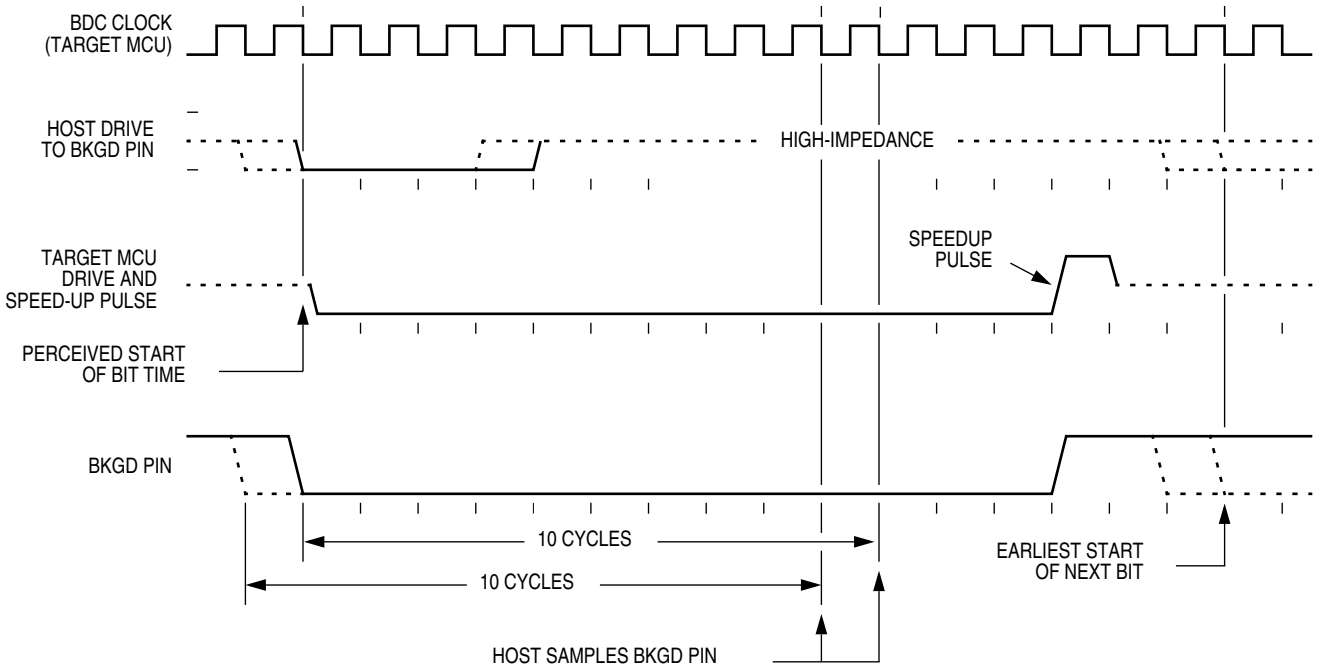


Figure 15-4. BDM Target-to-Host Serial Bit Timing (Logic 0)

15.3.3 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 15-1 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 15-1 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 15-1. 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.

15.3.4 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.

15.4 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 15.4.6, "Hardware Breakpoints."

15.4.1 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)

15.4.2 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 15.4.5, “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.

15.4.3 Change-of-Flow Information

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

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

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

15.4.4 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.

15.4.5 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 the ARM bit or DBGEN bit 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.

15.4.6 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 15.4.5, “Trigger Modes,” to be used to generate a hardware breakpoint request to the CPU. The TAG bit 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.

15.5 Registers and Control Bits

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.

15.5.1 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.

15.5.1.1 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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ENBDM	BDMACT	BKPTEN	FTS	CLKSW	WS	WSF	DVF
Write:								
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 15-5. BDC Status and Control Register (BDCSCR)

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.

1 = BDM can be made active to allow active background mode commands.
0 = BDM cannot be made active (non-intrusive commands still allowed).

BDMACT — Background Mode Active Status

This is a read-only status bit.

1 = BDM active and waiting for serial commands.
0 = BDM not active (user application program running).

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.

1 = BDC breakpoint enabled.
0 = BDC breakpoint disabled.

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.

1 = Breakpoint match forces active background mode at next instruction boundary (address need not be an opcode).
0 = Tag opcode at breakpoint address and enter active background mode if CPU attempts to execute that instruction.

CLKSW — Select Source for BDC Communications Clock

CLKSW defaults to 0, which selects the alternate BDC clock source.
1 = MCU bus clock.
0 = Alternate BDC clock source.

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.
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.
0 = Target CPU is running user application code or in active background mode (was not in wait or stop mode when background became active).

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.)
1 = Memory access command failed because the CPU entered wait or stop mode.
0 = Memory access did not conflict with a wait or stop instruction.

DVF — Data Valid Failure Status

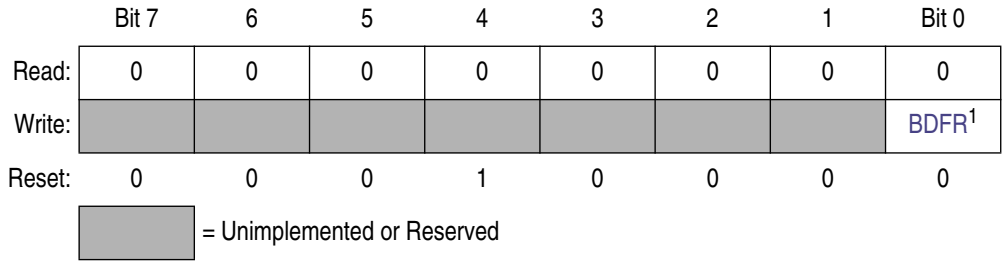
This status bit is not used in the MC9S08GB/GT because it does not have any slow access memory.
1 = Memory access command failed because CPU was not finished with a slow memory access.
0 = Memory access did not conflict with a slow memory access.

15.5.1.2 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 15.3.4, “BDC Hardware Breakpoint.”

15.5.2 System Background Debug Force Reset Register (SBDFR)

This register contains a single write-only control bit. A serial active 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 $00.



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

Figure 15-6. System Background Debug Force Reset Register (SBDFR)

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.

15.5.3 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.

15.5.3.1 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 $00 at reset and can be read at any time or written at any time unless ARM = 1.

15.5.3.2 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 $00 at reset and can be read at any time or written at any time unless ARM = 1.

15.5.3.3 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 $00 at reset and can be read at any time or written at any time unless ARM = 1.

15.5.3.4 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 $00 at reset and can be read at any time or written at any time unless ARM = 1.

15.5.3.5 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 $00.

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.

15.5.3.6 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.

15.5.3.7 Debug Control Register (DBGC)

This register can be read or written at any time.

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
Reset:	0	0	0	0	0	0	0	0

Figure 15-7. Debug Control Register (DBGC)

DBGEN — Debug Module Enable

Used to enable the debug module. DBGEN cannot be set to 1 if the MCU is secure.
1 = DBG enabled.
0 = DBG disabled.

ARM — Arm Control

Controls whether the debugger is comparing and storing information in the FIFO. A write is used to set this bit (and the ARMF bit) and completion of a debug run automatically clears it. Any debug run can be manually stopped by writing 0 to ARM or to DBGEN.
1 = Debugger armed.
0 = Debugger not armed.

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.
1 = CPU breaks requested as tag type requests.
0 = CPU breaks requested as force type requests.

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.
1 = Triggers cause a break request to the CPU.
0 = CPU break requests not enabled.

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.
1 = Comparator A can only match on a read cycle.
0 = Comparator A can only match on a write cycle.

RWAEN — Enable R/W for Comparator A

Controls whether the level of R/W is considered for a comparator A match.
1 = R/W is used in comparison A.
0 = R/W is not used in comparison A.

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.
1 = Comparator B can match only on a read cycle.
0 = Comparator B can match only on a write cycle.

RWBEN — Enable R/W for Comparator B

Controls whether the level of R/W is considered for a comparator B match.
1 = R/W is used in comparison B.
0 = R/W is not used in comparison B.

15.5.3.8 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.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TRGSEL	BEGIN	0	0	TRG3	TRG2	TRG1	TRG0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-8. Debug Trigger Register (DBGT)

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.
1 = Trigger if opcode at compare address is executed (tag).
0 = Trigger on access to compare address (force).

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.
1 = Trigger initiates data storage (begin trace).
0 = Data stored in FIFO until trigger (end trace).

TRG3:TRG2:TRG1:TRG0 — Select Trigger Mode

Selects one of nine triggering modes

Table 15-2. Trigger Mode Selection

TRG[3:0]	Triggering Mode
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

15.5.3.9 Debug Status Register (DBGS)

This is a read-only status register.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-9. Debug Status Register (DBGS)

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.

1 = Comparator A match.
0 = Comparator A has not matched.

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.

1 = Comparator B match.
0 = Comparator B has not matched.

ARMF — Arm Flag

While DBGEN = 1, this status bit is a read-only image of the ARM bit 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 the ARM or DBGEN bits in DBGC.

1 = Debugger armed.
0 = Debugger not armed.

CNT3:CNT2:CNT1:CNT0 — 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.

Table 15-3. CNT Status Bits

CNT[3:0]	Valid Words in FIFO
0000	No valid data
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	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	Temp. Code
Operating temperature range (packaged)	T_A	–40 to 85	°C	C
Thermal resistance 64-pin LQFP (GB60) 48-pin QFN (GT60) 42-pin SDIP (GT60) 44-pin QFP (GT60)	θ_{JA} [1, 2]	 65 82 57 118	°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 Per SEMI G38-87 and JEDEC JESD51-2 with the single layer board horizontal. Single layer board is designed per JEDEC JESD51-3.

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 equations 1 and 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 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 Protection Characteristics

Parameter	Symbol	Value	Unit
ESD Target for Machine Model (MM) MM circuit description	V_{THMM}	200	V
ESD Target for Human Body Model (HBM) HBM circuit description	V_{THHBM}	2000	V

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-4. 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.) $0 < f_{Bus} < 8$ MHz $0 < f_{Bus} < 20$ MHz	V_{DD}	1.8 2.08		3.6 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}	2.08 2.16	2.1 2.19	2.2 2.27	V
Low-voltage detection threshold — low range (V_{DD} falling) (V_{DD} rising)	V_{LVDL}	1.80 1.88	1.82 1.90	1.91 1.99	V
Low-voltage warning threshold — high range (V_{DD} falling) (V_{DD} rising)	V_{LVWH}	2.35 2.35	2.40 2.40	2.5	V
Low-voltage warning threshold — low range (V_{DD} falling) (V_{DD} rising)	V_{LVWL}	2.08 2.16	2.1 2.19	2.2 2.27	V

Table A-4. DC Characteristics (Sheet 2 of 2)
(Temperature Range = –40 to 85°C Ambient)

Parameter	Symbol	Min	Typical[1]	Max	Unit
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
Input high voltage (V_{DD} > 2.3 V) (all digital inputs)	V_{IH}	0.70 × V_{DD}		—	V
Input high voltage (1.8 V ≤ V_{DD} ≤ 2.3 V) (all digital inputs)	V_{IH}	0.85 × V_{DD}		—	V
Input low voltage (V_{DD} > 2.3 V) (all digital inputs)	V_{IL}	—		0.35 × V_{DD}	V
Input low voltage (1.8 V ≤ V_{DD} ≤ 2.3 V) (all digital inputs)	V_{IL}	—		0.30 × V_{DD}	V
Input hysteresis (all digital inputs)	V_{hys}	0.06 × 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	1.0	µA
High impedance (off-state) leakage current (per pin) V_{In} = V_{DD} or V_{SS}, all input/output	$\lvert I_{OZ}\rvert$	—	0.025	1.0	µA
Internal pullup and pulldown resistors[3] (all port pins and IRQ)	R_{PU}	17.5		52.5	kW
Internal pulldown resistors (Port A4–A7 and IRQ)	R_{PD}	17.5		52.5	kW
Output high voltage (V_{DD} ≥ 1.8 V) I_{OH} = –2 mA (ports A, B, D, E, and G)	V_{OH}	V_{DD} – 0.5		—	V
Output high voltage (ports C and F) I_{OH} = –10 mA (V_{DD} ≥ 2.7 V) I_{OH} = –6 mA (V_{DD} ≥ 2.3 V) I_{OH} = –3 mA (V_{DD} ≥ 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} ≥ 1.8 V) I_{OL} = 2.0 mA (ports A, B, D, E, and G)	V_{OL}	—		0.5	V
Output low voltage (ports C and F) I_{OL} = 10.0 mA (V_{DD} ≥ 2.7 V) I_{OL} = 6 mA (V_{DD} ≥ 2.3 V) I_{OL} = 3 mA (V_{DD} ≥ 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}.

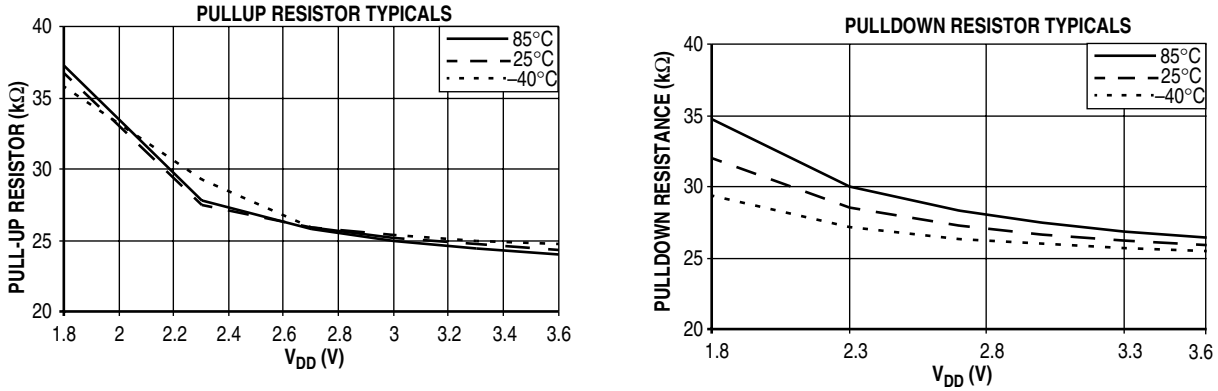


Figure A-1. Pullup and Pulldown Typical Resistor Values (V_{DD} = 3.0 V)

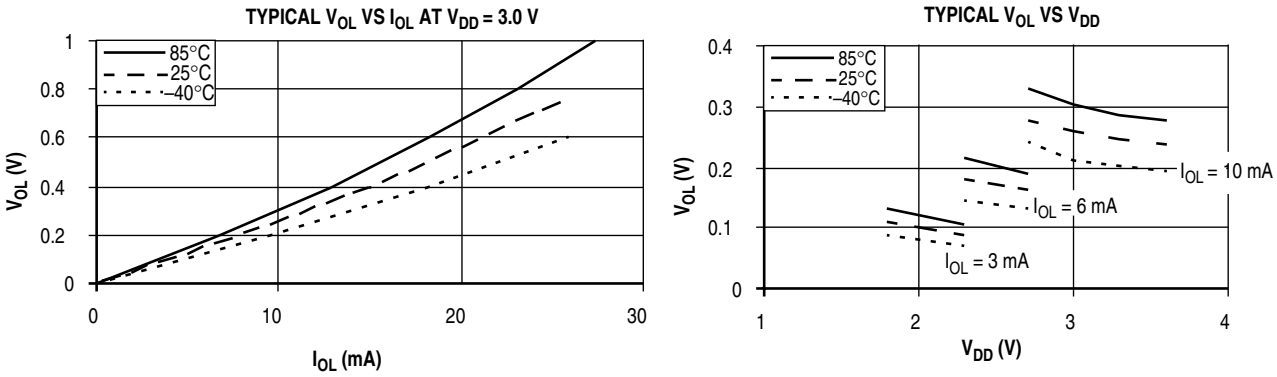


Figure A-2. Typical Low-Side Driver (Sink) Characteristics (Ports C and F)

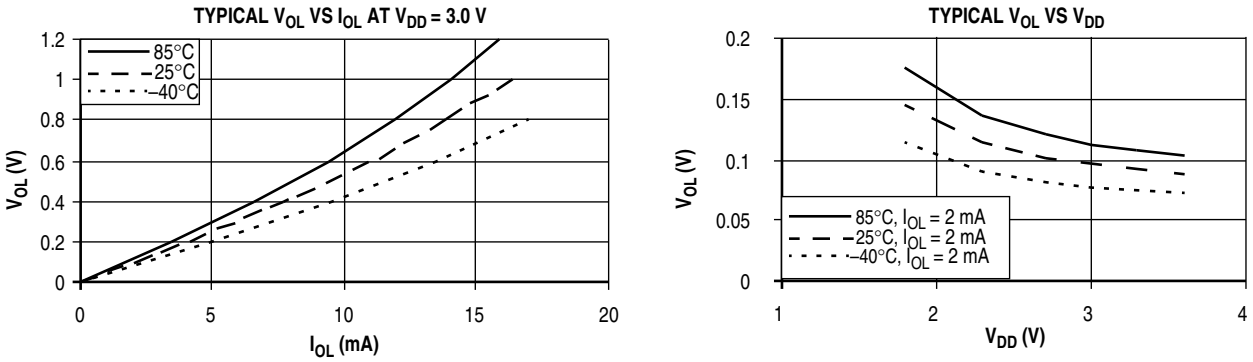


Figure A-3. Typical Low-Side Driver (Sink) Characteristics (Ports A, B, D, E, and G)

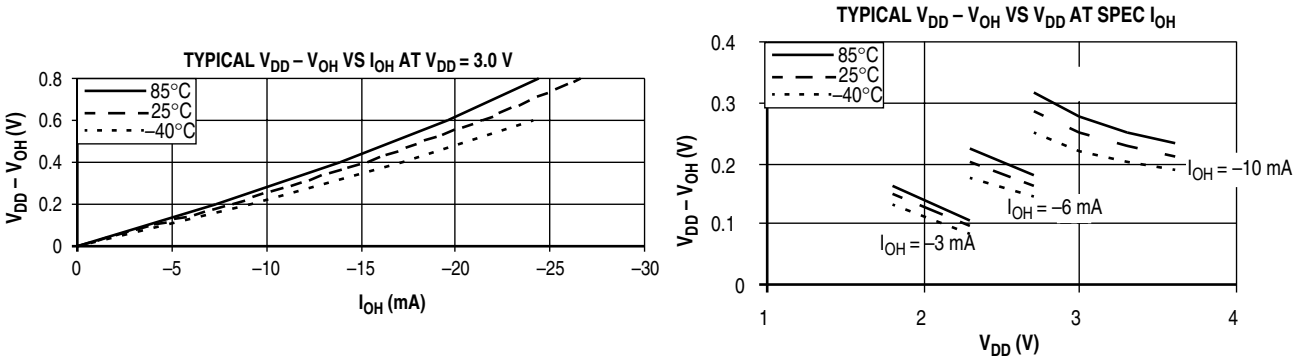


Figure A-4. Typical High-Side Driver (Source) Characteristics (Ports C and F)

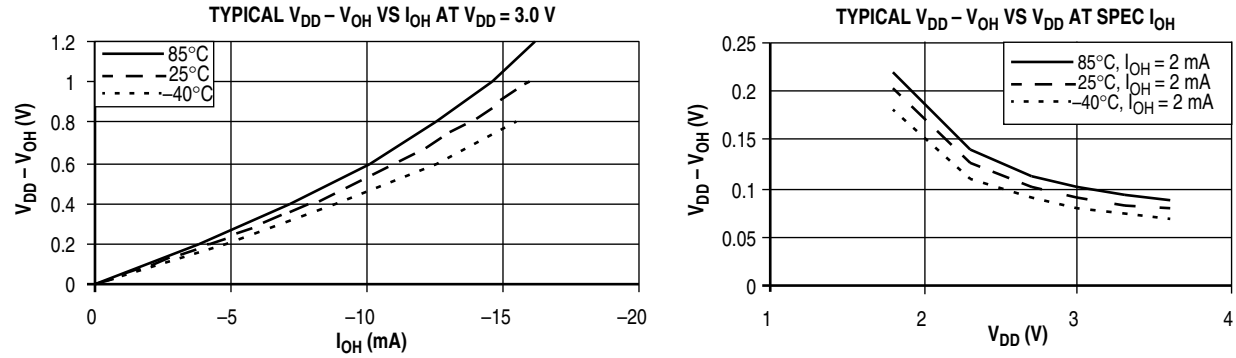


Figure A-5. Typical High-Side (Source) Characteristics (Ports A, B, D, E, and G)

A.6 Supply Current Characteristics

Table A-5. 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	1.1 mA	2.1 mA[4] 2.1 mA(4) 2.1 mA(4)	55 70 85
		2	0.8 mA	1.8 mA(4) 1.8 mA(4) 1.8 mA(4)	55 70 85
Run supply current (3) measured at (CPU clock = 16 MHz, f_{Bus} = 8 MHz)	RI_{DD}	3	6.5 mA	7.5 mA(4) 7.5 mA(4) 7.5 mA[5]	55 70 85
		2	4.8 mA	5.8 mA(4) 5.8 mA(4) 5.8 mA(4)	55 70 85
Stop1 mode supply current	$S1I_{DD}$	3	25 nA	0.6 µA(4) 1.8 µA(4) 4.0 µA(5)	55 70 85
		2	20 nA	500 nA(4) 1.5 µA(4) 3.3 µA(4)	55 70 85
Stop2 mode supply current	$S2I_{DD}$	3	550 nA	3.0 µA(4) 5.5 µA(4) 11 µA(5)	55 70 85
		2	400 nA	2.4 µA(4) 5.0 µA(4) 9.5 µA(4)	55 70 85
Stop3 mode supply current	$S3I_{DD}$	3	675 nA	4.3 µA(4) 7.2 µA(4) 17.0 µA(5)	55 70 85
		2	500 nA	3.5 µA(4) 6.2 µA(4) 15.0 µA(4)	55 70 85
RTI adder to stop2 or stop3[6]		3	300 nA		55 70 85
		2	300 nA		55 70 85

Table A-5. Supply Current Characteristics (continued)

Parameter	Symbol	V_{DD} (V)	Typical[1]	Max[2]	Temp. (°C)
LVI adder to stop3 (LVDSE = LVDE = 1)		3	70 µA		55 70 85
		2	60 µA		55 70 85

1 Typicals are measured at 25°C. See Figure A-6 through Figure A-9 for typical curves across voltage/temperature.

2 Values given here are preliminary estimates prior to completing characterization.

3 All modules except ATD active, ICG configured for FBE, and does not include any dc loads on port pins

4 Values are characterized but not tested on every part.

5 Every unit tested to this parameter. All other values in the Max column are guaranteed by characterization.

6 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.

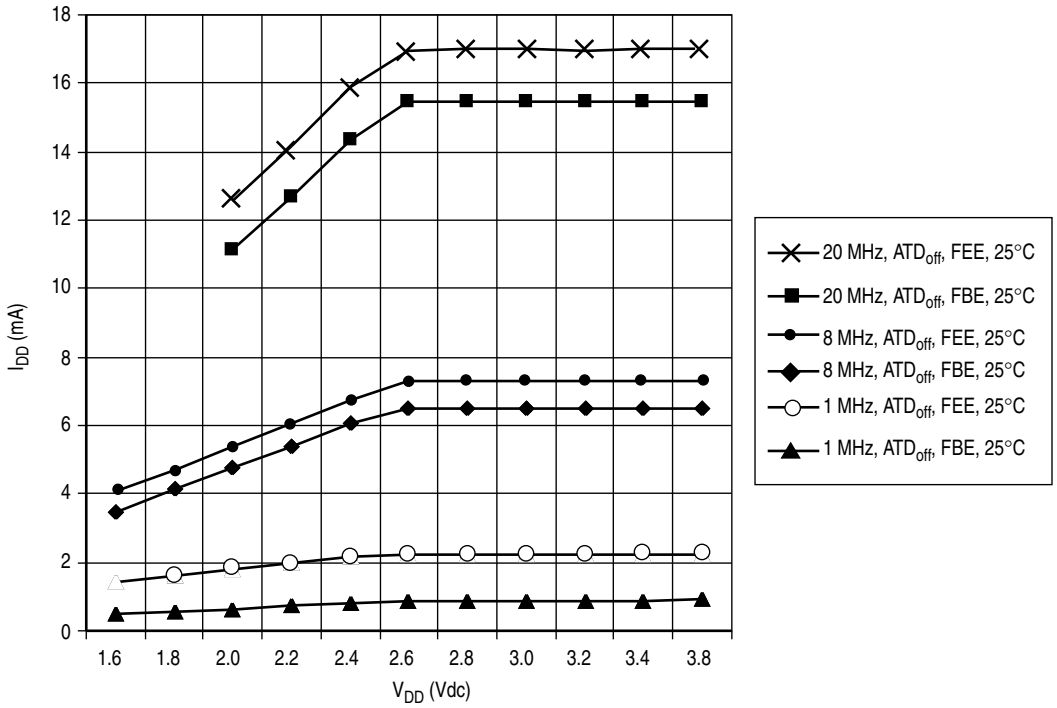
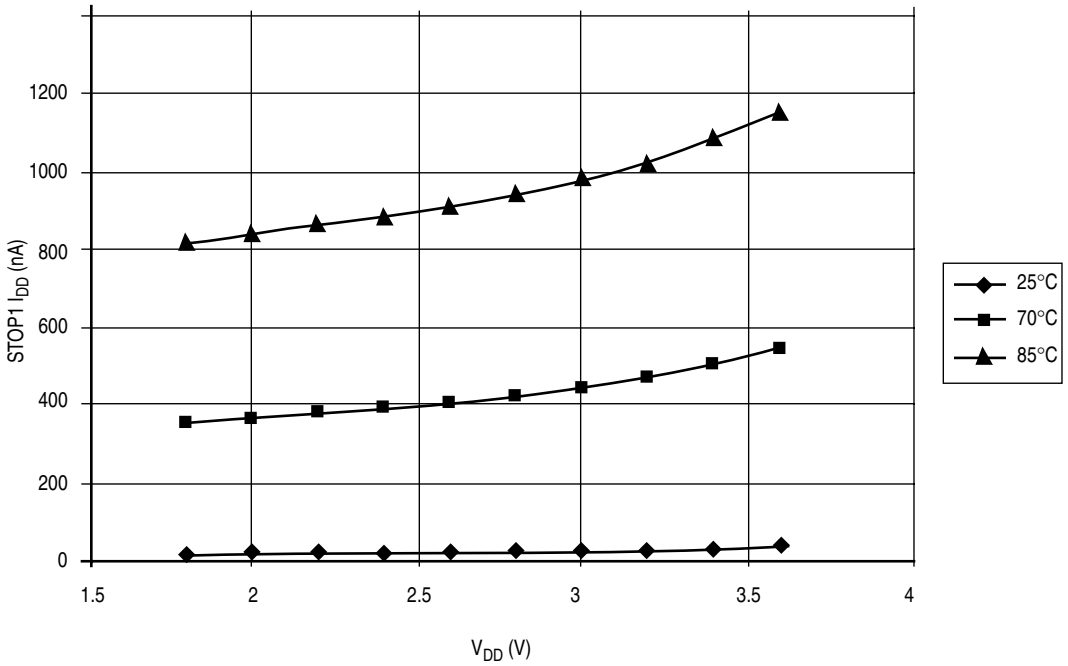
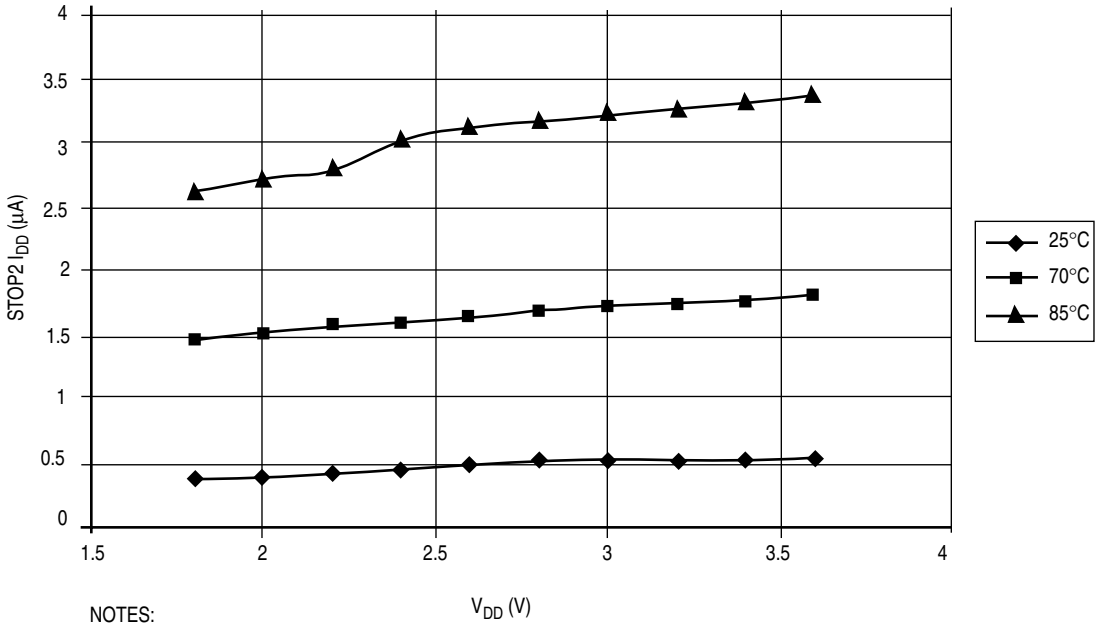


Figure A-6. Typical Run I_{DD} for FBE and FEE Modes, I_{DD} vs V_{DD}



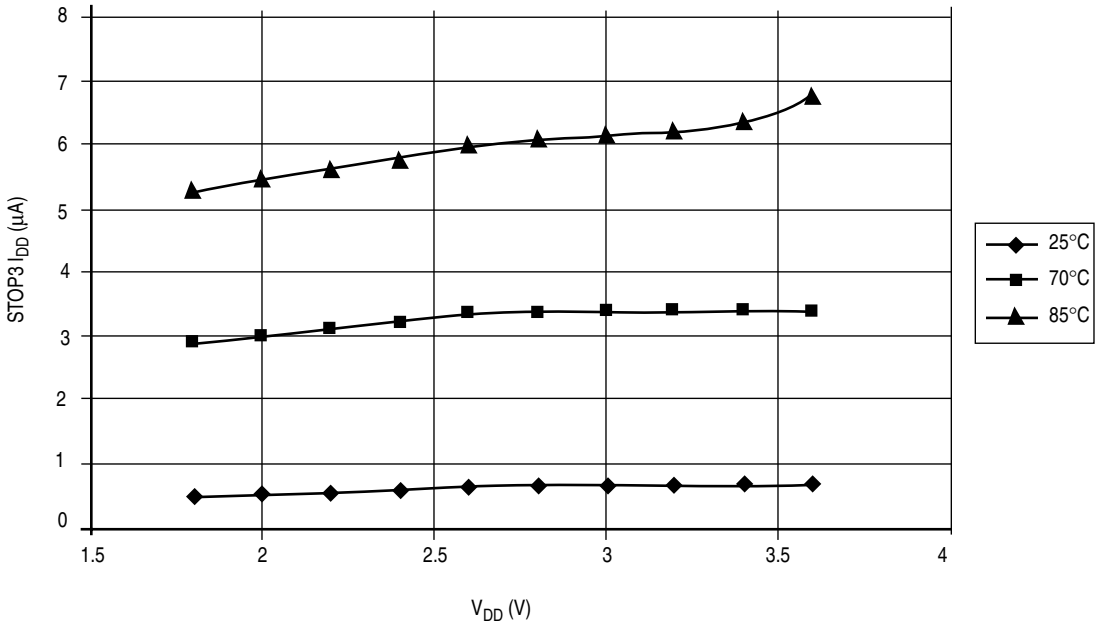
NOTES:
1. Clock sources and LVD are all disabled (OSCSTEN = LVDSE = 0).
2. All I/O are set as outputs and driven to V_{SS} with no load.

Figure A-7. Typical Stop1 I_{DD}



NOTES:
1. Clock sources and LVD are all disabled (OSCSTEN = LVDSE = 0).
2. All I/O are set as outputs and driven to V_{SS} with no load.

Figure A-8. Typical Stop 2 I_{DD}



NOTES:
1. Clock sources and LVD are all disabled (OSCSTEN = LVDSE = 0).
2. All I/O are set as outputs and driven to V_{SS} with no load.

Figure A-9. Typical Stop3 I_{DD}

A.7 ATD Characteristics

Table A-6. ATD Electrical Characteristics (Operating)

Num	Characteristic	Condition	Symbol	Min	Typical	Max	Unit
1	ATD supply[1]		V_{DDAD}	1.80	—	3.6	V
2	ATD supply current	Enabled	$I_{DDADrun}$	—	0.7	1.2	mA
		Disabled (ATDPU = 0 or STOP)	$I_{DDADstop}$	—	0.02	0.6	µA
3	Differential supply voltage	V_{DD}–V_{DDAD}	$\|V_{DDLT}\|$	—	—	100	mV
4	Differential ground voltage	V_{SS}–V_{SSAD}	$\|V_{SDLT}\|$	—	—	100	mV
5	Reference potential, low		$\|V_{REFL}\|$	—	—	V_{SSAD}	V
	Reference potential, high	2.08V ≤ V_{DDAD} ≤ 3.6V	V_{REFH}	2.08	—	V_{DDAD}	V
		1.80V ≤ V_{DDAD} < 2.08V		V_{DDAD}	—	V_{DDAD}	
6	Reference supply current (V_{REFH} to V_{REFL})	Enabled	I_{REF}	—	200	300	µA
		Disabled (ATDPU = 0 or STOP)	I_{REF}	—	<0.01	0.02	
7	Analog input voltage[2]		V_{INDC}	V_{SSAD} – 0.3	—	V_{DDAD} + 0.3	V

1 V_{DDAD} must be at same potential as V_{DD}.
2 Maximum electrical operating range, not valid conversion range.

Table A-7. ATD Timing/Performance Characteristics[1]

Num	Characteristic	Symbol	Condition	Min	Typ	Max	Unit
1	ATD conversion clock frequency	f_{ATDCLK}	$2.08V \le V_{DDAD} \le 3.6V$	0.5	—	2.0	MHz
			$1.80V \le V_{DDAD} < 2.08V$	0.5	—	1.0	
2	Conversion cycles (continuous convert)[2]	CC		28	28	<30	ATDCLK cycles
3	Conversion time	T_{conv}	$2.08V \le V_{DDAD} \le 3.6V$	14.0	—	60.0	µS
			$1.80V \le V_{DDAD} < 2.08V$	28.0	—	60.0	
4	Source impedance at input[3]	R_{AS}		—	—	10	kΩ
5	Analog Input Voltage[4]	V_{AIN}		V_{REFL}		V_{REFH}	V
6	Ideal resolution (1 LSB)[5]	RES	$2.08V \le V_{DDAD} \le 3.6V$	2.031	—	3.516	mV
			$1.80V \le V_{DDAD} < 2.08V$	1.758	—	2.031	
7	Differential non-linearity[6]	DNL	$1.80V \le V_{DDAD} \le 3.6V$	—	±0.5	±1.0	LSB
8	Integral non-linearity[7]	INL	$1.80\ V \le V_{DDAD} \le 3.6V$	—	±0.5	±1.0	LSB
9	Zero-scale error[8]	E_{ZS}	$1.80V \le V_{DDAD} \le 3.6V$	—	±0.4	±1.0	LSB
10	Full-scale error[9]	E_{FS}	$1.80V \le V_{DDAD} \le 3.6V$	—	±0.4	±1.0	LSB
11	Input leakage error [10]	E_{IL}	$1.80V \le V_{DDAD} \le 3.6V$	—	±0.05	±5	LSB
12	Total unadjusted error[11]	E_{TU}	$1.80V \le V_{DDAD} \le 3.6V$	—	±1.1	±2.5	LSB

1 All ACCURACY numbers are based on processor and system being in WAIT state (very little activity and no IO switching) and that adequate low-pass filtering is present on analog input pins (filter with 0.01 µF to 0.1 µF capacitor between analog input and V_{REFL}). Failure to observe these guidelines may result in system or microcontroller noise causing accuracy errors which will vary based on board layout and the type and magnitude of the activity.

2 This is the conversion time for subsequent conversions in continuous convert mode. Actual conversion time for single conversions or the first conversion in continuous mode is extended by one ATD clock cycle and 2 bus cycles due to starting the conversion and setting the CCF flag. The total conversion time in Bus Cycles for a conversion is:

SC Bus Cycles = ((PRS+1)*2) * (28+1) + 2 CC Bus Cycles = ((PRS+1)*2) * (28)

3 R_{AS} is the real portion of the impedance of the network driving the analog input pin. Values greater than this amount may not fully charge the input circuitry of the ATD resulting in accuracy error.

4 Analog input must be between V_{REFL} and V_{REFH} for valid conversion. Values greater than V_{REFH} will convert to $3FF less the full scale error (E_{FS}).

5 The resolution is the ideal step size or 1LSB = $(V_{REFH}–V_{REFL})/1024$

6 Differential non-linearity is the difference between the current code width and the ideal code width (1LSB). The current code width is the difference in the transition voltages to and from the current code.

7 Integral non-linearity is the difference between the transition voltage to the current code and the adjusted ideal transition voltage for the current code. The adjusted ideal transition voltage is $(Current\ Code–1/2)*(1/((V_{REFH}+E_{FS})–(V_{REFL}+E_{ZS})))$.

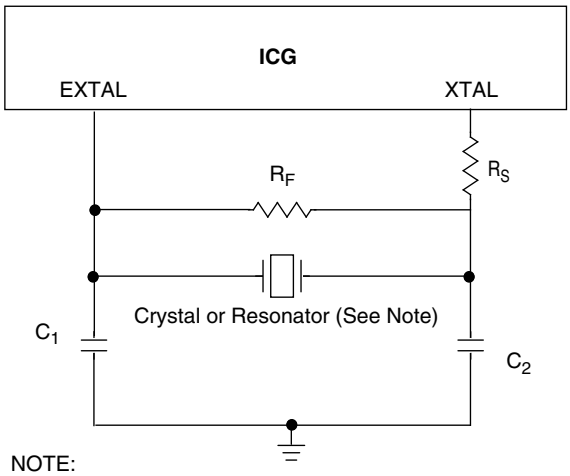
8 Zero-scale error is the difference between the transition to the first valid code and the ideal transition to that code. The Ideal transition voltage to a given code is $(Code–1/2)*(1/(V_{REFH}–V_{REFL}))$.

9 Full-scale error is the difference between the transition to the last valid code and the ideal transition to that code. The ideal transition voltage to a given code is $(Code–1/2)*(1/(V_{REFH}–V_{REFL}))$.

10 Input leakage error is error due to input leakage across the real portion of the impedance of the network driving the analog pin. Reducing the impedance of the network reduces this error.

11 Total unadjusted error is the difference between the transition voltage to the current code and the ideal straight-line transfer function. This measure of error includes inherent quantization error (1/2LSB) and circuit error (differential, integral, zero-scale, and full-scale) error. The specified value of E_T assumes zero E_{IL} (no leakage or zero real source impedance).

A.8 Internal Clock Generation Module Characteristics



NOTE:
Use fundamental mode crystal or ceramic resonator only.

Table A-8. ICG DC Electrical Specifications (Temperature Range = –40 to 85°C Ambient)

Characteristic	Symbol	Min	Typ[1]	Max	Unit
Load capacitors	C_1 C_2	[2]			
Feedback resistor Low range (32k to 100 kHz) High range (1M – 16 MHz)	R_F		 10 1		 MΩ MΩ
Series Resistor	R_S		0		Ω

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.8.1 ICG Frequency Specifications

Table A-9. ICG Frequency Specifications
(V_{DDA} = V_{DDA} (min) to V_{DDA} (max), Temperature Range = –40 to 85°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 2 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
Duty cycle of input clock [4] (REFS = 0)	t_{dc}	40	—	60	%

Table A-9. ICG Frequency Specifications (continued)
(V_{DDA} = V_{DDA} (min) to V_{DDA} (max), Temperature Range = –40 to 85°C Ambient)

Characteristic	Symbol	Min	Typical	Max	Unit
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}	— —		2 2	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

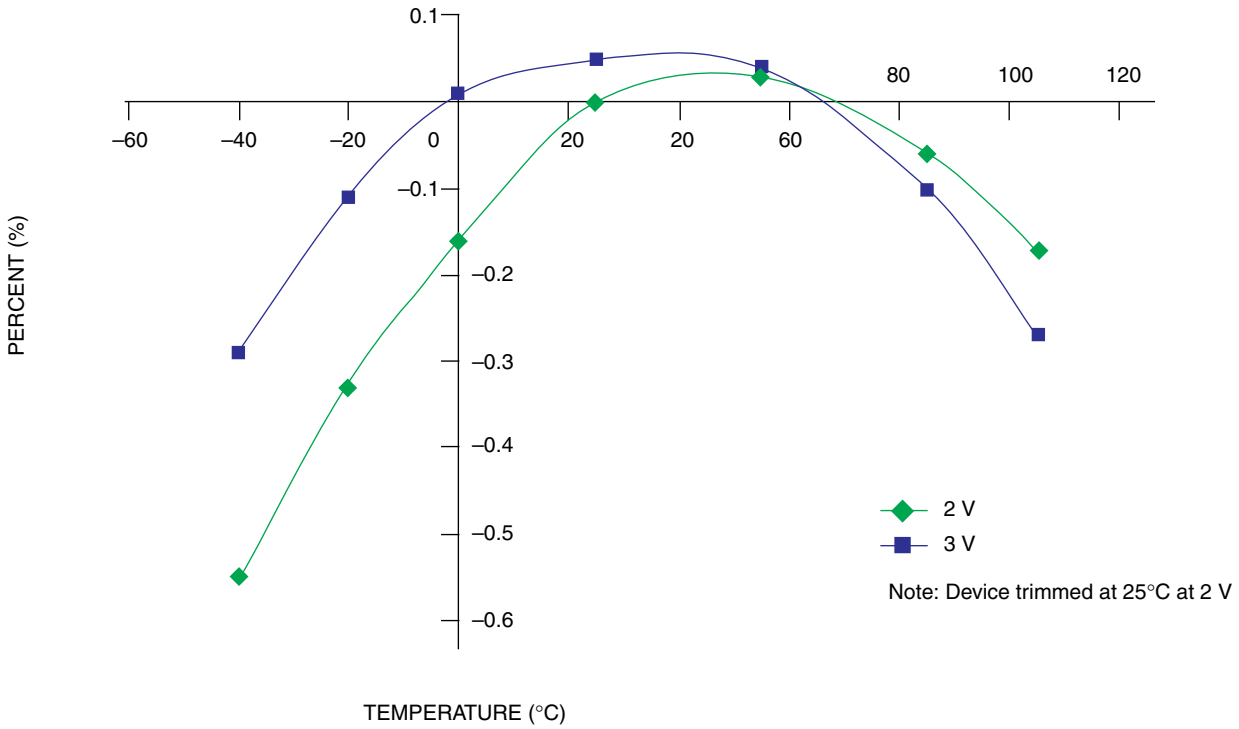


Figure A-10. Internal Oscillator Deviation from Trimmed Frequency

A.9 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.9.1 Control Timing

Table A-10. 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.

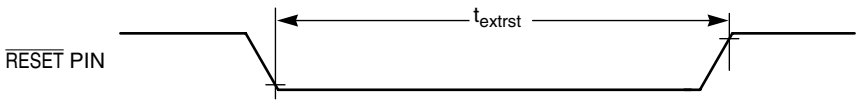


Figure A-11. Reset Timing

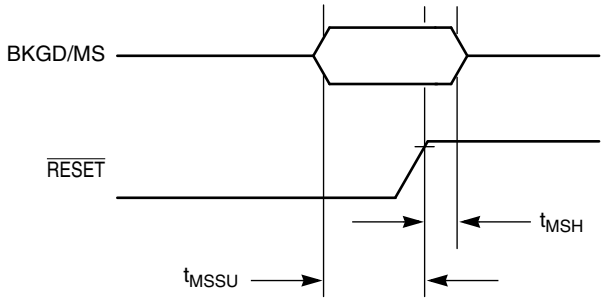


Figure A-12. Active Background Debug Mode Latch Timing

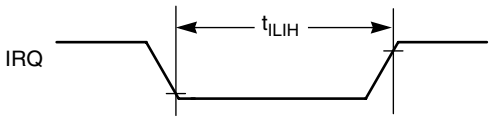


Figure A-13. IRQ Timing

A.9.2 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-11. 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}

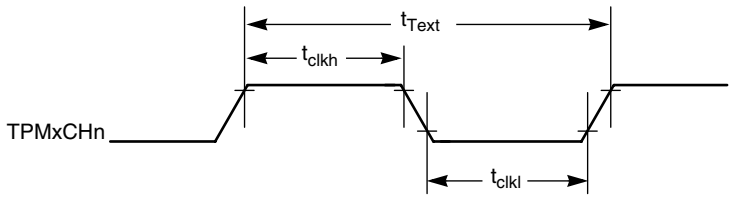


Figure A-14. Timer External Clock

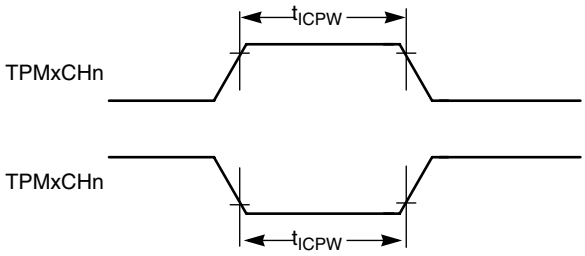


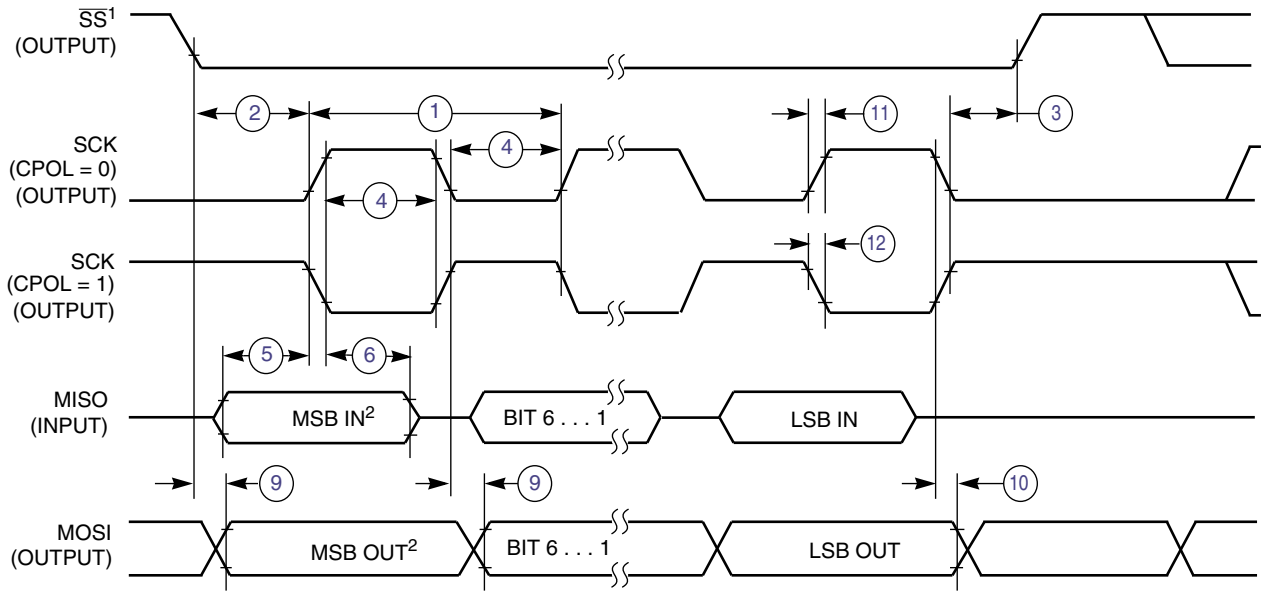
Figure A-15. Timer Input Capture Pulse

A.9.3 SPI Timing

Table A-12 and Figure A-16 through Figure A-19 describe the timing requirements for the SPI system.

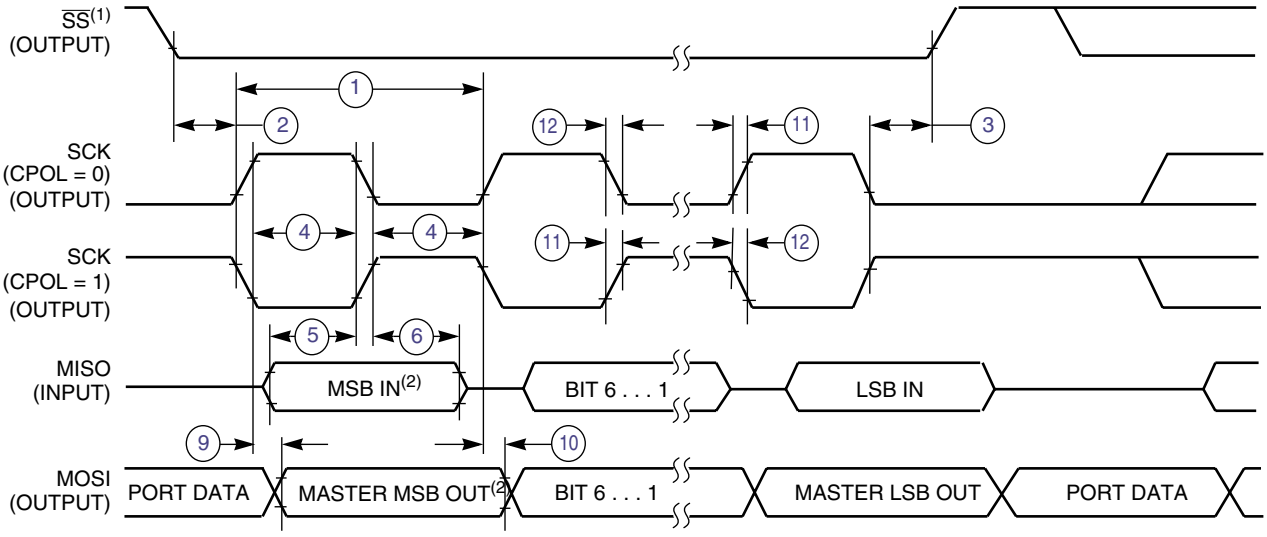
Table A-12. SPI Timing

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



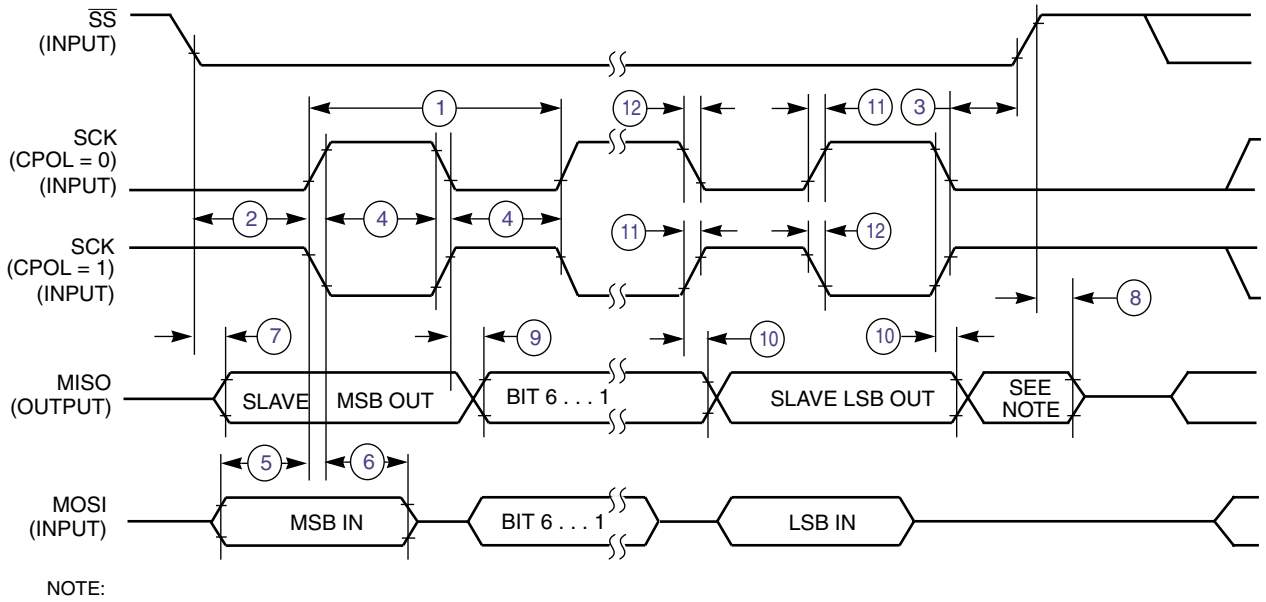
NOTES:
1. $\overline{SS}$ output mode (DDS7 = 1, SSOE = 1).
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure A-16. SPI Master Timing (CPHA = 0)



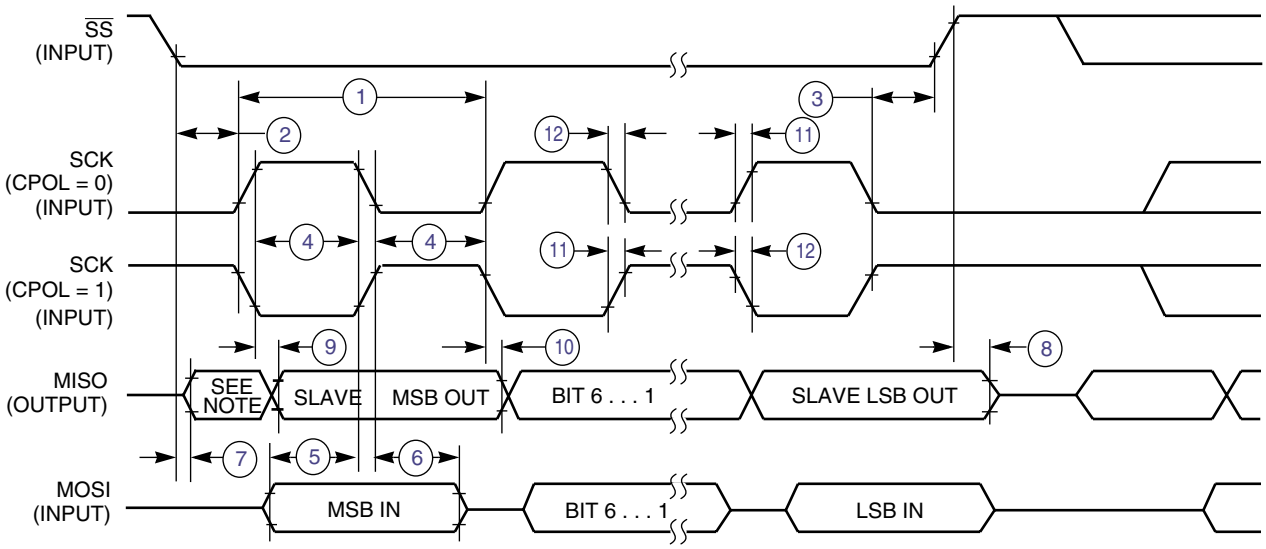
NOTES:
1. $\overline{SS}$ output mode (DDS7 = 1, SSOE = 1).
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure A-17. SPI Master Timing (CPHA = 1)



NOTE:
1. Not defined but normally MSB of character just received

Figure A-18. SPI Slave Timing (CPHA = 0)



NOTE:
1. Not defined but normally LSB of character just received

Figure A-19. SPI Slave Timing (CPHA = 1)

A.10 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-13. FLASH Characteristics

Characteristic	Symbol	Min	Typical	Max	Unit
Supply voltage for program/erase	$V_{prog/erase}$	2.1		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.*

Appendix B Ordering Information and Mechanical Drawings

B.1 Ordering Information

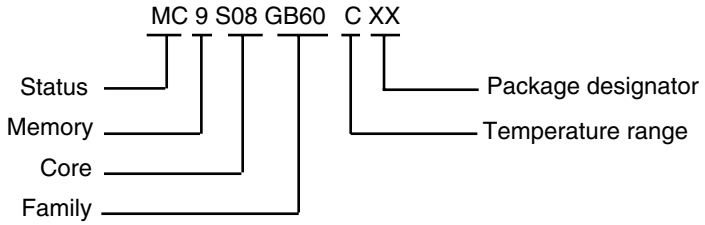
This section contains ordering numbers for MC9S08GB60, MC9S08GB32, MC9S08GT60, MC9S08GT32, and MC9S08GT16 devices. See below for an example of the device numbering system.

Table B-1. Device Numbering System

MC Order Number	FLASH Memory	RAM	TPM	Available Package Type (Part Number Suffix)
MC9S08GB60	60K	4K	One 3-channel and one 5-channel 16-bit timer	64 LQFP (FU)
MC9S08GB32	32K	2K	One 3-channel and one 5-channel 16-bit timer	64 LQFP (FU)
MC9S08GT60	60K	4K	Two 2-channel/16-bit timers	48 QFN (FD)[1] 44 QFP (FB) 42 SDIP (B)
MC9S08GT32	32K	2K	Two 2-channel/16-bit timers	48 QFN (FD)(1) 44 QFP (FB) 42 SDIP (B)
MC9S08GT16	16K	1K	Two 2-channel/16-bit timers	48 QFN (FD)(1) 44 QFP (FB) 42 SDIP (B)

[1] The 48-pin QFN package has one 3-channel and one 2-channel 16-bit TPM.

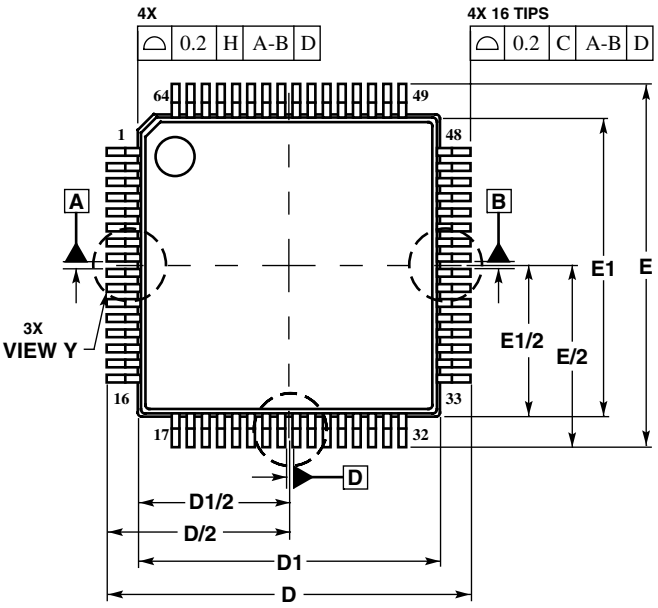
Temperature and package designators:
C = –40°C to 85°C
FU = 64-pin Low Quad Flat Package (LQFP)
FD = 48-pin Quad Flat Package, No Leads
FB = 44-pin Quad Flat Package (QFP)
B = 42-pin Skinny Dual In-Line Package (SDIP)
MC = Fully qualified

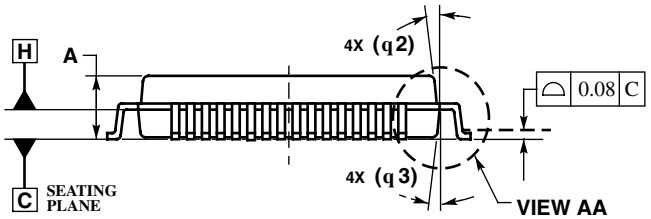


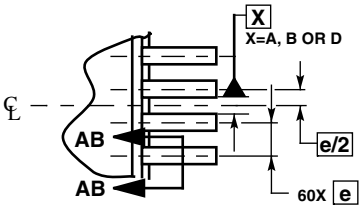
B.2 Mechanical Drawings

This appendix contains mechanical specifications for MC9S08GB/GT MCU.

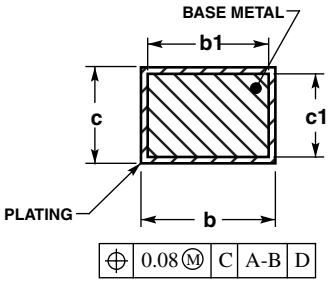
B.3 64-Pin LQFP Package Drawing



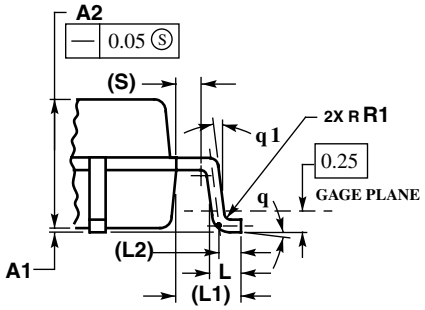




VIEW Y



SECTION AB-AB
ROTATED 90° CLOCKWISE



VIEW AA

NOTES:
1. DIMENSIONS AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE DATUM H IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH 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 DATUM C.
5. DIMENSIONS D AND E TO BE DETERMINED AT SEATING PLANE DATUM C.
6. DIMENSIONS D1 AND E1 DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 PER SIDE.
7. DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION. DAMBAR PROTRUSION SHALL NOT CAUSE THE b DIMENSION TO EXCEED 0.35. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07.

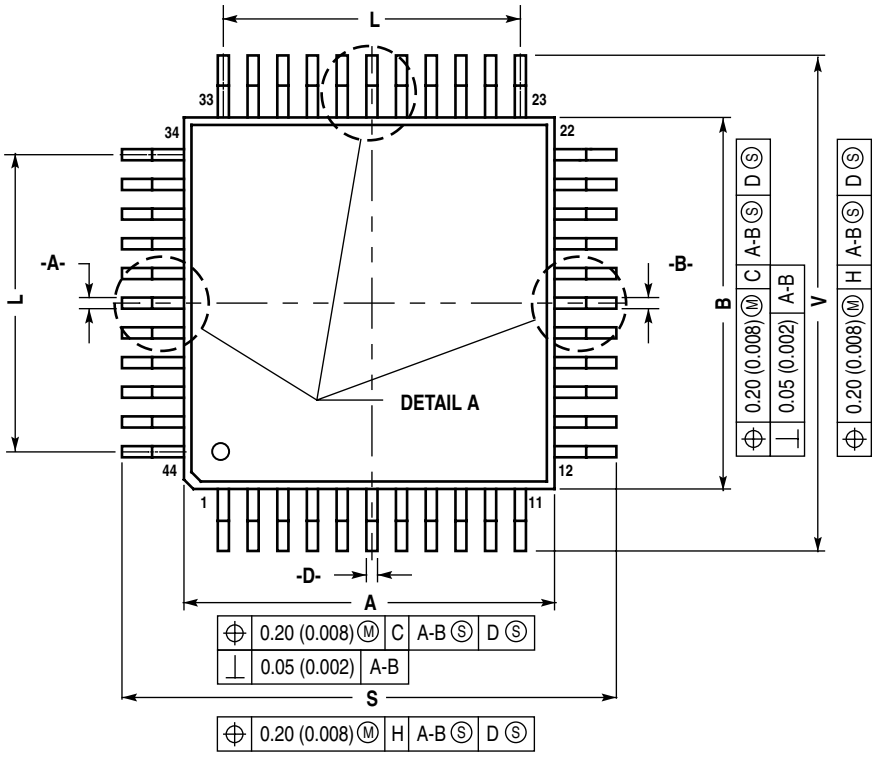
	MILLIMETERS	
DIM	MIN	MAX
A	---	1.60
A1	0.05	0.15
A2	1.35	1.45
b	0.17	0.27
b1	0.17	0.23
c	0.09	0.20
c1	0.09	0.16
D	12.00 BSC	
D1	10.00 BSC	
e	0.50 BSC	
E	12.00 BSC	
E1	10.00 BSC	
L	0.45	0.75
L1	1.00 REF	
L2	0.50 REF	
R1	0.10	0.20
S	0.20 REF	
q	0°	7°
q1	0°	---
q2	12° REF	
q3	12° REF	

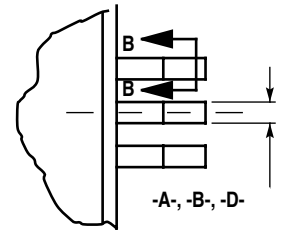
CASE 840F-02
ISSUE B

DATE 09/16/98

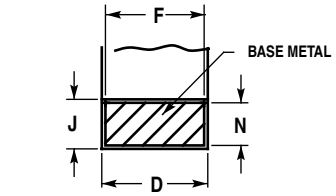
B.4 48-Pin QFN Package Drawing

B.5 44-Pin QFP Package Drawing

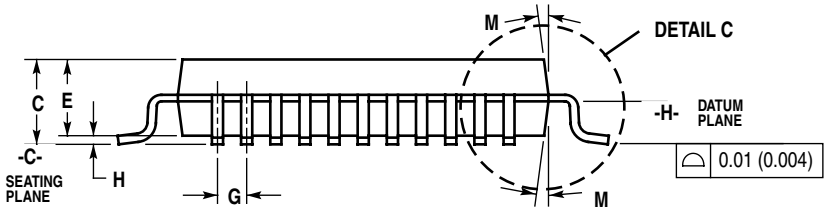


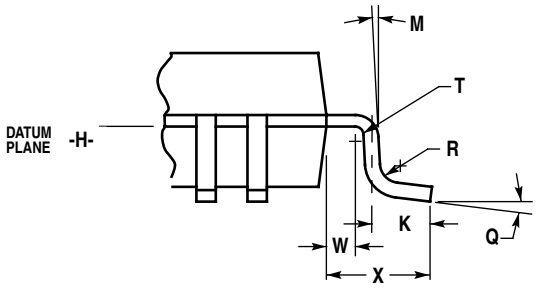


DETAIL A



SECTION B-B





DETAIL C

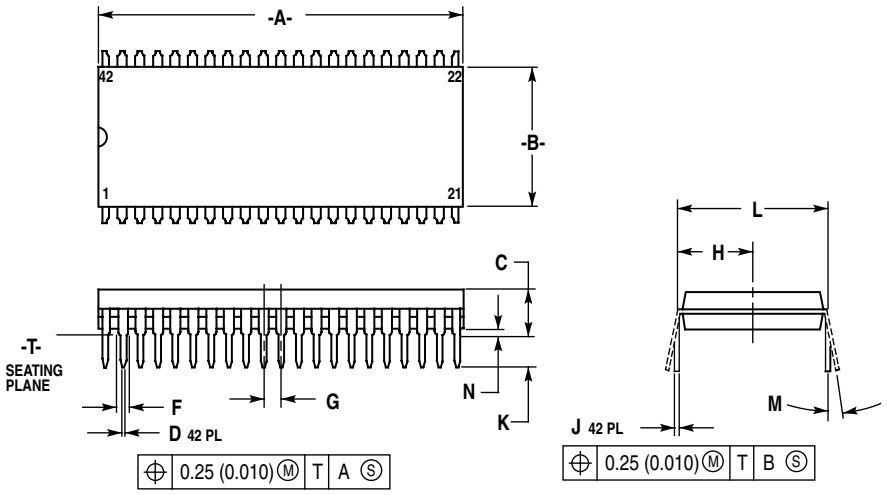
NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE -H- IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS -A-, -B- AND -D- TO BE DETERMINED AT DATUM PLANE -H-.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE -C-.
6. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 (0.010) PER SIDE. DIMENSIONS A AND B DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE -H-.
7. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08 (0.003) TOTAL IN EXCESS OF THE D DIMENSION AT MAXIMUM MATERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	9.90	10.10	0.390	0.398
B	9.90	10.10	0.390	0.398
C	2.10	2.45	0.083	0.096
D	0.30	0.45	0.012	0.018
E	2.00	2.10	0.079	0.083
F	0.30	0.40	0.012	0.016
G	0.80 BSC		0.031 BSC	
H	---	0.25	---	0.010
J	0.013	0.23	0.005	0.009
K	0.65	0.95	0.026	0.037
L	8.00 REF		0.315 REF	
M	5°	10°	5°	10°
N	0.13	0.17	0.005	0.007
Q	0°	7°	0°	7°
R	0.13	0.30	0.005	0.012
S	12.95	13.45	0.510	0.530
T	0.13	---	0.005	---
U	0°	---	0°	---
V	12.95	13.45	0.510	0.530
W	0.40	---	0.016	---
X	1.6 REF		0.063 REF	

CASE 824A-01
ISSUE O

DATE 11/19/90

B.6 42-Pin SDIP Package Drawing



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION L TO CENTER OF LEAD WHEN FORMED PARALLEL.
4. DIMENSIONS A AND B DO NOT INCLUDE MOLD FLASH. MAXIMUM MOLD FLASH 0.25 (0.010).

	INCHES		MILLIMETERS	
DIM	MIN	MAX	MIN	MAX
A	1.435	1.465	36.45	37.21
B	0.540	0.560	13.72	14.22
C	0.155	0.200	3.94	5.08
D	0.014	0.022	0.36	0.56
F	0.032	0.046	0.81	1.17
G	0.070 BSC		1.778 BSC	
H	0.300 BSC		7.62 BSC	
J	0.008	0.015	0.20	0.38
K	0.115	0.135	2.92	3.43
L	0.600 BSC		15.24 BSC	
M	0	15	0	15
N	0.020	0.040	0.51	1.02

CASE 858-01
ISSUE O

DATE 02/27/90

Data Sheet End Sheet

FINAL PAGE OF
288
PAGES

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

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

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

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

© Freescale Semiconductor, Inc. 2004. All rights reserved.

MC9S08GB60
Rev. 2.3, 12/2004

freescale™
semiconductor

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View MC9S08GB32CFUER on WIN SOURCE
- Freescale Semiconductor - NXP Information

Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management