THE DATASHEET OF

MC68HC98LJ12CFUE

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

MC68HC908LJ12
Technical Data

M68HC08
Microcontrollers

Rev. 2.1
MC68HC908LJ12/D
August 2, 2005

freescale.com

freescale™
semiconductor

MC68HC908LJ12

Technical Data

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale 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 which may be provided in Freescale 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 does not convey any license under its patent rights nor the rights of others. Freescale 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 product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale 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 was negligent regarding the design or manufacture of the part. Freescale, Inc. is an Equal Opportunity/Affirmative Action Employer.

© Freescale, Inc., 2002

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. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
February 2002	2	First general release.	—
August, 2005	2.1	Updated to meet Freescale identity guidelines.	—

List of Sections

Section 1. General Description 33
Section 2. Memory Map 43
Section 3. Random-Access Memory (RAM) 59
Section 4. FLASH Memory (FLASH) 61
Section 5. Configuration Registers (CONFIG) 71
Section 6. Central Processor Unit (CPU) 77
Section 7. Oscillator (OSC) 95
Section 8. Clock Generator Module (CGM) 101
Section 9. System Integration Module (SIM) 131
Section 10. Monitor ROM (MON) 155
Section 11. Timer Interface Module (TIM) 185
Section 12. Real Time Clock (RTC) 209
Section 13. Infrared Serial Communications Interface Module (IRSCI) 227
Section 14. Serial Peripheral Interface Module (SPI) . . .269
Section 15. Analog-to-Digital Converter (ADC) 301
Section 16. Liquid Crystal Display Driver (LCD) 317
Section 17. Input/Output (I/O) Ports 341
Section 18. External Interrupt (IRQ) 357
Section 19. Keyboard Interrupt Module (KBI) 363

Section 20. Computer Operating Properly (COP) 371
Section 21. Low-Voltage Inhibit (LVI) 377
Section 22. Break Module (BRK) 383
Section 23. Electrical Specifications 391
Section 24. Mechanical Specifications 407
Section 25. Ordering Information 411

Table of Contents

Section 1. General Description

1.1 Contents 33

1.2 Introduction 34

1.3 Features 34

1.4 MCU Block Diagram 36

1.5 Pin Assignments 38

1.6 Pin Functions 40
1.6.1 Power Supply Pins (V_{DD} and V_{SS}) 40
1.6.2 Analog Power Supply Pin (V_{DDA}) 40
1.6.3 Oscillator Pins (OSC1 and OSC2) 41
1.6.4 External Reset Pin ($\overline{RST}$) 41
1.6.5 External Interrupt Pin ($\overline{IRQ}$) 41
1.6.6 External Filter Capacitor Pin (CGMXFC) 41
1.6.7 ADC Voltage High Reference Pin (V_{REFH}) 41
1.6.8 ADC Voltage Low Reference Pin (V_{REFL}) 41
1.6.9 Port A Input/Output (I/O) Pins (PTA7–PTA0) 42
1.6.10 Port B I/O Pins (PTB7–PTB0) 42
1.6.11 Port C I/O Pins (PTC7–PTC0) 42
1.6.12 Port D I/O Pins (PTD7–PTD0) 42
1.6.13 LCD Backplane and Frontplane (BP0–BP2, FP0/BP3, FP1–FP18) 42

Section 2. Memory Map

2.1 Contents 43

2.2 Introduction 43

2.3 Unimplemented Memory Locations 43

2.4 Reserved Memory Locations 44
2.5 Input/Output (I/O) Section 44

Section 3. Random-Access Memory (RAM)

3.1 Contents 59
3.2 Introduction 59
3.3 Functional Description 59

Section 4. FLASH Memory (FLASH)

4.1 Contents 61
4.2 Introduction 61
4.3 Functional Description 62
4.4 FLASH Control Register 63
4.5 FLASH Page Erase Operation 64
4.6 FLASH Mass Erase Operation 65
4.7 FLASH Program Operation 66
4.8 FLASH Protection 68
4.8.1 FLASH Block Protect Register 68

Section 5. Configuration Registers (CONFIG)

5.1 Contents 71
5.2 Introduction 71
5.3 Functional Description 72
5.4 Configuration Register 1 (CONFIG1) 73
5.5 Configuration Register 2 (CONFIG2) 75

Section 6. Central Processor Unit (CPU)

6.1 Contents 77
6.2 Introduction 78
6.3 Features 78
6.4 CPU Registers 79
6.4.1 Accumulator 79
6.4.2 Index Register 80
6.4.3 Stack Pointer 80
6.4.4 Program Counter 81
6.4.5 Condition Code Register 82
6.5 Arithmetic/Logic Unit (ALU) 84
6.6 Low-Power Modes 84
6.6.1 Wait Mode 84
6.6.2 Stop Mode 85
6.7 CPU During Break Interrupts 85
6.8 Instruction Set Summary 85
6.9 Opcode Map 85

Section 7. Oscillator (OSC)

7.1 Contents 95
7.2 Introduction 95
7.3 Internal Oscillator 97
7.4 Crystal (X-tal) Oscillator 97
7.5 I/O Signals 97
7.5.1 Crystal Amplifier Input Pin (OSC1) 98
7.5.2 Crystal Amplifier Output Pin (OSC2) 98
7.5.3 Oscillator Enable Signal (SIMOSCEN) 98
7.5.4 Internal RC Clock (ICLK) 98
7.5.5 CGM Oscillator Clock (CGMXCLK) 98
7.5.6 CGM Reference Clock (CGMRCLK) 98

7.6 Low Power Modes 98
7.6.1 Wait Mode 99
7.6.2 Stop Mode 99

7.7 Oscillator During Break Mode 99

Section 8. Clock Generator Module (CGM)

8.1 Contents 101

8.2 Introduction 102

8.3 Features 103

8.4 Functional Description 103
8.4.1 Oscillator Module 106
8.4.2 Phase-Locked Loop Circuit (PLL) 106
8.4.3 PLL Circuits 106
8.4.4 Acquisition and Tracking Modes 108
8.4.5 Manual and Automatic PLL Bandwidth Modes 108
8.4.6 Programming the PLL 110
8.4.7 Special Programming Exceptions 114
8.4.8 Base Clock Selector Circuit 114
8.4.9 CGM External Connections 115

8.5 I/O Signals 115
8.5.1 External Filter Capacitor Pin (CGMXFC) 116
8.5.2 PLL Analog Power Pin (V_{DDA}) 116
8.5.3 PLL Analog Ground Pin (V_{SSA}) 116
8.5.4 Oscillator Output Frequency Signal (CGMXCLK) 116
8.5.5 CGM Reference Clock (CGMRCLK) 116
8.5.6 CGM VCO Clock Output (CGMVCLK) 117
8.5.7 CGM Base Clock Output (CGMOUT) 117
8.5.8 CGM CPU Interrupt (CGMINT) 117

8.6 CGM Registers 117
8.6.1 PLL Control Register 118
8.6.2 PLL Bandwidth Control Register 120
8.6.3 PLL Multiplier Select Registers 122
8.6.4 PLL VCO Range Select Register 123
8.6.5 PLL Reference Divider Select Register 124

8.7 Interrupts 125

8.8 Special Modes 125
8.8.1 Wait Mode 125
8.8.2 Stop Mode 126
8.8.3 CGM During Break Interrupts 126

8.9 Acquisition/Lock Time Specifications 127
8.9.1 Acquisition/Lock Time Definitions 127
8.9.2 Parametric Influences on Reaction Time 127
8.9.3 Choosing a Filter 129

Section 9. System Integration Module (SIM)

9.1 Contents 131

9.2 Introduction 132

9.3 SIM Bus Clock Control and Generation 134
9.3.1 Bus Timing 135
9.3.2 Clock Start-up from POR or LVI Reset 135
9.3.3 Clocks in Stop Mode and Wait Mode 136

9.4 Reset and System Initialization 136
9.4.1 External Pin Reset 137
9.4.2 Active Resets from Internal Sources 137
9.4.2.1 Power-On Reset 138
9.4.2.2 Computer Operating Properly (COP) Reset 139
9.4.2.3 Illegal Opcode Reset 140
9.4.2.4 Illegal Address Reset 140
9.4.2.5 Low-Voltage Inhibit (LVI) Reset 140
9.4.2.6 Monitor Mode Entry Module Reset (MODRST) 140

9.5 SIM Counter 141
9.5.1 SIM Counter During Power-On Reset 141
9.5.2 SIM Counter During Stop Mode Recovery 141
9.5.3 SIM Counter and Reset States 141

9.6 Exception Control 142
9.6.1 Interrupts 142
9.6.1.1 Hardware Interrupts 144
9.6.1.2 SWI Instruction 145

9.6.1.3 Interrupt Status Registers 145
9.6.1.4 Interrupt Status Register 1 145
9.6.1.5 Interrupt Status Register 2 147
9.6.1.6 Interrupt Status Register 3 147
9.6.2 Reset 148
9.6.3 Break Interrupts 148
9.6.4 Status Flag Protection in Break Mode 148

9.7 Low-Power Modes 149
9.7.1 Wait Mode 149
9.7.2 Stop Mode 150

9.8 SIM Registers 151
9.8.1 SIM Break Status Register 152
9.8.2 SIM Reset Status Register 153
9.8.3 SIM Break Flag Control Register 154

Section 10. Monitor ROM (MON)

10.1 Contents 155

10.2 Introduction 156

10.3 Features 156

10.4 Functional Description 157
10.4.1 Entering Monitor Mode 159
10.4.2 Data Format 163
10.4.3 Break Signal 163
10.4.4 Baud Rate 163
10.4.5 Commands 164

10.5 Security 169

10.6 ROM-Resident Routines 171
10.6.1 PRGRNGE 173
10.6.2 ERARNGE 175
10.6.3 LDRNGE 176
10.6.4 MON_PRGRNGE 177
10.6.5 MON_ERARNGE 178
10.6.6 MON_LDRNGE 179
10.6.7 EE_WRITE 180
10.6.8 EE_READ 183

Section 11. Timer Interface Module (TIM)

11.1 Contents 185
11.2 Introduction 186
11.3 Features 186
11.4 Pin Name Conventions 187
11.5 Functional Description 187
11.5.1 TIM Counter Prescaler 191
11.5.2 Input Capture 191
11.5.3 Output Compare 192
11.5.3.1 Unbuffered Output Compare 192
11.5.3.2 Buffered Output Compare 193
11.5.4 Pulse Width Modulation (PWM) 193
11.5.4.1 Unbuffered PWM Signal Generation 194
11.5.4.2 Buffered PWM Signal Generation 195
11.5.4.3 PWM Initialization 196
11.6 Interrupts 197
11.7 Low-Power Modes 197
11.7.1 Wait Mode 198
11.7.2 Stop Mode 198
11.8 TIM During Break Interrupts 198
11.9 I/O Signals 199
11.10 I/O Registers 199
11.10.1 TIM Status and Control Register 200
11.10.2 TIM Counter Registers 202
11.10.3 TIM Counter Modulo Registers 203
11.10.4 TIM Channel Status and Control Registers 204
11.10.5 TIM Channel Registers 207

Section 12. Real Time Clock (RTC)

12.1 Contents 209
12.2 Introduction 210

12.3 Features 210

12.4 Functional Description 212
12.4.1 Time Functions 214
12.4.2 Calendar Functions 214
12.4.3 Alarm Functions 214
12.4.4 Timebase Interrupts 214
12.4.5 Chronograph Functions 215

12.5 Low-Power Modes 215
12.5.1 Wait Mode 215
12.5.2 Stop Mode 215

12.6 RTC Registers 216
12.6.1 RTC Control Register 1 (RTCCR1) 216
12.6.2 RTC Control Register 2 (RTCCR2) 218
12.6.3 RTC Status Register (RTCSR) 219
12.6.4 Alarm Minute and Hour Registers (ALMR and ALHR) 222
12.6.5 Second Register (SECR) 223
12.6.6 Minute Register (MINR) 223
12.6.7 Hour Register (HRR) 224
12.6.8 Day Register (DAYR) 224
12.6.9 Month Register (MTHR) 225
12.6.10 Year Register (YRR) 225
12.6.11 Day-Of-Week Register (DOWR) 226
12.6.12 Chronograph Data Register (CHRR) 226

Section 13. Infrared Serial Communications Interface Module (IRSCI)

13.1 Contents 227

13.2 Introduction 228

13.3 Features 229

13.4 Pin Name Conventions 231

13.5 IRSCI Module Overview 231

13.6 Infrared Functional Description 232
13.6.1 Infrared Transmit Encoder 233

13.6.2 Infrared Receive Decoder 233

13.7 SCI Functional Description 234
13.7.1 Data Format 235
13.7.2 Transmitter 236
13.7.2.1 Character Length 237
13.7.2.2 Character Transmission 237
13.7.2.3 Break Characters 238
13.7.2.4 Idle Characters 238
13.7.2.5 Transmitter Interrupts 239
13.7.3 Receiver 239
13.7.3.1 Character Length 239
13.7.3.2 Character Reception 241
13.7.3.3 Data Sampling 241
13.7.3.4 Framing Errors 243
13.7.3.5 Baud Rate Tolerance 243
13.7.3.6 Receiver Wakeup 246
13.7.3.7 Receiver Interrupts 247
13.7.3.8 Error Interrupts 247

13.8 Low-Power Modes 248
13.8.1 Wait Mode 248
13.8.2 Stop Mode 248

13.9 SCI During Break Module Interrupts 249

13.10 I/O Signals 249
13.10.1 PTB0/TxD (Transmit Data) 249
13.10.2 PTB1/RxD (Receive Data) 249

13.11 I/O Registers 250
13.11.1 SCI Control Register 1 251
13.11.2 SCI Control Register 2 253
13.11.3 SCI Control Register 3 256
13.11.4 SCI Status Register 1 258
13.11.5 SCI Status Register 2 (SCS2) 262
13.11.6 SCI Data Register (SCDR) 263
13.11.7 SCI Baud Rate Register (SCBR) 264
13.11.8 SCI Infrared Control Register 267

Section 14. Serial Peripheral Interface Module (SPI)

14.1 Contents 269
14.2 Introduction 270
14.3 Features 270
14.4 Pin Name Conventions and I/O Register Addresses 271
14.5 Functional Description 271
14.5.1 Master Mode 273
14.5.2 Slave Mode 274
14.6 Transmission Formats 275
14.6.1 Clock Phase and Polarity Controls 275
14.6.2 Transmission Format When CPHA = 0 276
14.6.3 Transmission Format When CPHA = 1 278
14.6.4 Transmission Initiation Latency 279
14.7 Queuing Transmission Data 281
14.8 Error Conditions 282
14.8.1 Overflow Error 282
14.8.2 Mode Fault Error 284
14.9 Interrupts 286
14.10 Resetting the SPI 288
14.11 Low-Power Modes 289
14.11.1 Wait Mode 289
14.11.2 Stop Mode 289
14.12 SPI During Break Interrupts 290
14.13 I/O Signals 290
14.13.1 MISO (Master In/Slave Out) 291
14.13.2 MOSI (Master Out/Slave In) 291
14.13.3 SPSCK (Serial Clock) 292
14.13.4 $\overline{SS}$ (Slave Select) 292
14.13.5 CGND (Clock Ground) 293
14.14 I/O Registers 294
14.14.1 SPI Control Register 294

14.14.2 SPI Status and Control Register 296
14.14.3 SPI Data Register 299

Section 15. Analog-to-Digital Converter (ADC)

15.1 Contents 301

15.2 Introduction 302

15.3 Features 302

15.4 Functional Description 303
15.4.1 ADC Port I/O Pins 303
15.4.2 Voltage Conversion 305
15.4.3 Conversion Time 305
15.4.4 Continuous Conversion 306
15.4.5 Result Justification 306
15.4.6 Monotonicity 307

15.5 Interrupts 308

15.6 Low-Power Modes 308
15.6.1 Wait Mode 308
15.6.2 Stop Mode 308

15.7 I/O Signals 308
15.7.1 ADC Voltage In (V_{ADIN}) 309
15.7.2 ADC Analog Power Pin (V_{DDA}) 309
15.7.3 ADC Voltage Reference High Pin (V_{REFH}) 309
15.7.4 ADC Voltage Reference Low Pin (V_{REFL}) 309

15.8 I/O Registers 310
15.8.1 ADC Status and Control Register 310
15.8.2 ADC Data Register 312
15.8.3 ADC Clock Control Register 314

Section 16. Liquid Crystal Display Driver (LCD)

16.1 Contents 317

16.2 Introduction 318

16.3 Features 318

16.4 Pin Name Conventions and I/O Register Addresses 318

16.5 Functional Description 320
16.5.1 LCD Duty 321
16.5.2 LCD Voltages (V_{LCD}, V_{LCD1}, V_{LCD2}, V_{LCD3}) 323
16.5.3 LCD Cycle Frame 323
16.5.4 Fast Charge and Low Current 324
16.5.5 Contrast Control 325

16.6 Low-Power Modes 325
16.6.1 Wait Mode 325
16.6.2 Stop Mode 325

16.7 I/O Signals 326
16.7.1 BP0–BP3 (Backplane Drivers) 326
16.7.2 FP0–FP26 (Frontplane Drivers) 328

16.8 Seven Segment Display Connection 332

16.9 I/O Registers 335
16.9.1 LCD Control Register (LCDCR) 335
16.9.2 LCD Clock Register (LCDCLK) 337
16.9.3 LCD Data Registers (LDAT1–LDAT14) 339

Section 17. Input/Output (I/O) Ports

17.1 Contents 341

17.2 Introduction 341

17.3 Port A 344
17.3.1 Port A Data Register (PTA) 344
17.3.2 Data Direction Register A (DDRA) 345

17.4 Port B 347
17.4.1 Port B Data Register (PTB) 347
17.4.2 Data Direction Register B (DDRB) 348
17.4.3 Port B LED Control Register (LEDB) 350

17.5 Port C 351
17.5.1 Port C Data Register (PTC) 351
17.5.2 Data Direction Register C (DDRC) 352

17.6 Port D 354
17.6.1 Port D Data Register (PTD) 354
17.6.2 Data Direction Register D (DDRD) 355

Section 18. External Interrupt (IRQ)

18.1 Contents 357
18.2 Introduction 357
18.3 Features 357
18.4 Functional Description 358
18.4.1 $\overline{\text{IRQ}}$ Pin 360
18.5 IRQ Module During Break Interrupts 361
18.6 IRQ Status and Control Register (INTSCR) 361

Section 19. Keyboard Interrupt Module (KBI)

19.1 Contents 363
19.2 Introduction 363
19.3 Features 364
19.4 I/O Pins 364
19.5 Functional Description 365
19.5.1 Keyboard Initialization 367
19.6 Keyboard Interrupt Registers 367
19.6.1 Keyboard Status and Control Register 368
19.6.2 Keyboard Interrupt Enable Register 369
19.7 Low-Power Modes 369
19.8 Wait Mode 369
19.9 Stop Mode 370
19.10 Keyboard Module During Break Interrupts 370

Section 20. Computer Operating Properly (COP)

20.1 Contents 371
20.2 Introduction 371
20.3 Functional Description 372
20.4 I/O Signals 373
20.4.1 ICLK 373
20.4.2 STOP Instruction 373
20.4.3 COPCTL Write 373
20.4.4 Power-On Reset 373
20.4.5 Internal Reset 374
20.4.6 Reset Vector Fetch 374
20.4.7 COPD (COP Disable) 374
20.4.8 COPRS (COP Rate Select) 374
20.5 COP Control Register 375
20.6 Interrupts 375
20.7 Monitor Mode 375
20.8 Low-Power Modes 375
20.8.1 Wait Mode 376
20.8.2 Stop Mode 376
20.9 COP Module During Break Mode 376

Section 21. Low-Voltage Inhibit (LVI)

21.1 Contents 377
21.2 Introduction 377
21.3 Features 377
21.4 Functional Description 378
21.4.1 Interrupt LVI Operation 380
21.4.2 Forced Reset Operation 380
21.4.3 Voltage Hysteresis Protection 380
21.4.4 LVI Trip Selection 381
21.5 LVI Status Register 381

21.6 Low-Power Modes 382
21.6.1 Wait Mode 382
21.6.2 Stop Mode 382

Section 22. Break Module (BRK)

22.1 Contents 383

22.2 Introduction 383

22.3 Features 384

22.4 Functional Description 384
22.4.1 Flag Protection During Break Interrupts 386
22.4.2 CPU During Break Interrupts 386
22.4.3 TIM1 and TIM2 During Break Interrupts 386
22.4.4 COP During Break Interrupts 386

22.5 Low-Power Modes 386
22.5.1 Wait Mode 386
22.5.2 Stop Mode 387

22.6 Break Module Registers 387
22.6.1 Break Status and Control Register 387
22.6.2 Break Address Registers 388
22.6.3 SIM Break Status Register 388
22.6.4 SIM Break Flag Control Register 390

Section 23. Electrical Specifications

23.1 Contents 391

23.2 Introduction 392

23.3 Absolute Maximum Ratings 392

23.4 Functional Operating Range 393

23.5 Thermal Characteristics 393

23.6 5.0V DC Electrical Characteristics 394

23.7 3.3V DC Electrical Characteristics 396

23.8 5.0V Control Timing 397

23.9 3.3V Control Timing 397
23.10 5.0V Oscillator Characteristics 398
23.11 3.3V Oscillator Characteristics 398
23.12 5.0V ADC Electrical Characteristics 399
23.13 3.3V ADC Electrical Characteristics 400
23.14 Timer Interface Module Characteristics 401
23.15 CGM Electrical Specifications 401
23.16 5.0V SPI Characteristics 402
23.17 3.3V SPI Characteristics 403
23.18 FLASH Memory Characteristics 406

Section 24. Mechanical Specifications

24.1 Contents 407
24.2 Introduction 407
24.3 52-Pin Low-Profile Quad Flat Pack (LQFP) 408
24.4 64-Pin Low-Profile Quad Flat Pack (LQFP) 409
24.5 64-Pin Quad Flat Pack (QFP) 410

Section 25. Ordering Information

25.1 Contents 411
25.2 Introduction 411
25.3 MC Order Numbers 411

List of Figures

Figure	Title	Page
1-1	MC68HC908LJ12 Block Diagram	37
1-2	64-Pin QFP and 64-Pin LQFP Pin Assignment	38
1-3	52-Pin LQFP Pin Assignment	39
1-4	Power Supply Bypassing	40
2-1	Memory Map	45
2-2	Control, Status, and Data Registers	46
4-1	FLASH I/O Register Summary	62
4-2	FLASH Control Register (FLCR)	63
4-3	FLASH Programming Flowchart	67
4-4	FLASH Block Protect Register (FLBPR)	68
4-5	FLASH Block Protect Start Address	68
5-1	CONFIG Registers Summary	72
5-2	Configuration Register 1 (CONFIG1)	73
5-3	Configuration Register 2 (CONFIG2)	75
6-1	CPU Registers	79
6-2	Accumulator (A)	79
6-3	Index Register (H:X)	80
6-4	Stack Pointer (SP)	80
6-5	Program Counter (PC)	81
6-6	Condition Code Register (CCR)	82
7-1	Oscillator Module Block Diagram	96
8-1	CGM Block Diagram	104
8-2	CGM I/O Register Summary	105
8-3	CGM External Connections	115

Figure	Title	Page
8-4	PLL Control Register (PCTL)	118
8-5	PLL Bandwidth Control Register (PBWCR)	121
8-6	PLL Multiplier Select Register High (PMSH)	122
8-7	PLL Multiplier Select Register Low (PMSL)	122
8-8	PLL VCO Range Select Register (PMRS)	123
8-9	PLL Reference Divider Select Register (PMDS)	124
8-10	PLL Filter	129
9-1	SIM Block Diagram	133
9-2	SIM I/O Register Summary	134
9-3	CGM Clock Signals	135
9-4	External Reset Timing	137
9-5	Internal Reset Timing	138
9-6	Sources of Internal Reset	138
9-7	POR Recovery	139
9-8	Interrupt Entry Timing	142
9-9	Interrupt Recovery Timing	142
9-10	Interrupt Processing	143
9-11	Interrupt Recognition Example	144
9-12	Interrupt Status Register 1 (INT1)	145
9-13	Interrupt Status Register 2 (INT2)	147
9-14	Interrupt Status Register 3 (INT3)	147
9-15	Wait Mode Entry Timing	149
9-16	Wait Recovery from Interrupt or Break	150
9-17	Wait Recovery from Internal Reset	150
9-18	Stop Mode Entry Timing	151
9-19	Stop Mode Recovery from Interrupt or Break	151
9-20	SIM Break Status Register (SBSR)	152
9-21	SIM Reset Status Register (SRSR)	153
9-22	SIM Break Flag Control Register (SBFCR)	154
10-1	Monitor Mode Circuit	158
10-2	Low-Voltage Monitor Mode Entry Flowchart	162
10-3	Monitor Data Format	163
10-4	Break Transaction	163
10-5	Read Transaction	165

Figure	Title	Page

10-6 Write Transaction 165
10-7 Stack Pointer at Monitor Mode Entry 168
10-8 Monitor Mode Entry Timing. 169
10-9 Data Block Format for ROM-Resident Routines. 172
10-10 EE_WRITE FLASH Memory Usage 181

11-1 TIM Block Diagram 188
11-2 TIM I/O Register Summary. 189
11-3 PWM Period and Pulse Width 194
11-4 TIM Status and Control Register (TSC) 200
11-5 TIM Counter Registers High (TCNTH) 202
11-6 TIM Counter Registers Low (TCNTL). 202
11-7 TIM Counter Modulo Register High (TMODH) 203
11-8 TIM Counter Modulo Register Low (TMODL). 203
11-9 TIM Channel 0 Status and Control Register (TSC0) 204
11-10 TIM Channel 1 Status and Control Register (TSC1) 204
11-11 CHxMAX Latency 207
11-12 TIM Channel 0 Register High (TCH0H) 208
11-13 TIM Channel 0 Register Low (TCH0L) 208
11-14 TIM Channel 1 Register High (TCH1H) 208
11-15 TIM Channel 1 Register Low (TCH1L) 208

12-1 RTC I/O Register Summary 210
12-2 RTC Block Diagram 213
12-3 RTC Control Register 1 (RTCCR1) 216
12-4 RTC Control Register 2 (RTCCR2) 218
12-5 RTC Status Register (RTCSR). 219
12-6 Alarm Minute Register (ALMR). 222
12-7 Alarm Hour Register (ALHR) 222
12-8 Second Register (SECR) 223
12-9 Minute Register (MINR) 223
12-10 Hour Register (HRR) 224
12-11 Day Register (DAYR) 224
12-12 Month Register (MTHR) 225
12-13 Year Register (YRR). 225
12-14 Day-Of-Week Register (DOWR). 226
12-15 Chronograph Data Register (CHRR) 226

Figure	Title	Page

13-1 IRSCI I/O Registers Summary 230
13-2 IRSCI Block Diagram 231
13-3 Infrared Sub-Module Diagram 232
13-4 Infrared SCI Data Example 233
13-5 SCI Module Block Diagram 234
13-6 SCI Data Formats 235
13-7 SCI Transmitter 236
13-8 SCI Receiver Block Diagram 240
13-9 Receiver Data Sampling 241
13-10 Slow Data 244
13-11 Fast Data 245
13-12 SCI Control Register 1 (SCC1) 251
13-13 SCI Control Register 2 (SCC2) 254
13-14 SCI Control Register 3 (SCC3) 256
13-15 SCI Status Register 1 (SCS1) 258
13-16 Flag Clearing Sequence 261
13-17 SCI Status Register 2 (SCS2) 262
13-18 SCI Data Register (SCDR) 263
13-19 SCI Baud Rate Register (SCBR) 264
13-20 SCI Infrared Control Register (SCIRCR) 267

14-1 SPI I/O Register Summary 271
14-2 SPI Module Block Diagram 272
14-3 Full-Duplex Master-Slave Connections 273
14-4 Transmission Format (CPHA = 0) 277
14-5 CPHA/$\overline{SS}$ Timing 277
14-6 Transmission Format (CPHA = 1) 278
14-7 Transmission Start Delay (Master) 280
14-8 SPRF/SPTE CPU Interrupt Timing 281
14-9 Missed Read of Overflow Condition 283
14-10 Clearing SPRF When OVRF Interrupt Is Not Enabled 284
14-11 SPI Interrupt Request Generation 287
14-12 CPHA/$\overline{SS}$ Timing 292
14-13 SPI Control Register (SPCR) 294
14-14 SPI Status and Control Register (SPSCR) 296
14-15 SPI Data Register (SPDR) 299

Figure	Title	Page

15-1 ADC I/O Register Summary 303
15-2 ADC Block Diagram 304
15-3 8-Bit Truncation Mode Error 307
15-4 ADC Status and Control Register (ADSCR) 310
15-5 ADRH and ADRL in 8-Bit Truncated Mode 312
15-6 ADRH and ADRL in Right Justified Mode 312
15-7 ADRH and ADRL in Left Justified Mode 313
15-8 ADRH and ADRL in Left Justified Sign Data Mode 313
15-9 ADC Clock Control Register (ADICLK) 314

16-1 LCD I/O Register Summary 319
16-2 LCD Block Diagram 321
16-3 Simplified LCD Schematic (1/3 Duty, 1/3 Bias) 322
16-4 Fast Charge Timing 324
16-5 1/3 Duty LCD Backplane Driver Waveforms 326
16-6 Static LCD Backplane Driver Waveform 327
16-7 1/4 Duty LCD Backplane Driver Waveforms 327
16-8 Static LCD Frontplane Driver Waveforms 328
16-9 1/3 Duty LCD Frontplane Driver Waveforms 329
16-10 1/4 Duty LCD Frontplane Driver Waveforms 330
16-11 1/4 Duty LCD Frontplane Driver Waveforms (continued) 331
16-12 7-Segment Display Example 332
16-13 BP0–BP2 and FP0–FP2 Output Waveforms for 7-Segment Display Example 333
16-14 "f" Segment Voltage Waveform 334
16-15 "e" Segment Voltage Waveform 334
16-16 LCD Control Register (LCDCR) 335
16-17 LCD Clock Register (LCDCLK) 337
16-18 LCD Data Registers 1–14 (LDAT1–LDAT14) 339

17-1 I/O Port Register Summary 342
17-2 Port A Data Register (PTA) 344
17-3 Data Direction Register A (DDRA) 345
17-4 Port A I/O Circuit 346
17-5 Port B Data Register (PTB) 347
17-6 Data Direction Register B (DDRB) 349
17-7 Port B I/O Circuit 349

Figure	Title	Page
17-8	Port B LED Control Register (LEDB)	350
17-9	Port C Data Register (PTC)	351
17-10	Data Direction Register B (DDRB)	352
17-11	Port C I/O Circuit.	352
17-12	Port D Data Register (PTD)	354
17-13	Data Direction Register D (DDRD)	355
17-14	Port D I/O Circuit.	356
18-1	IRQ Module Block Diagram	359
18-2	IRQ Status and Control Register (INTSCR)	362
19-1	KBI I/O Register Summary	364
19-2	Keyboard Interrupt Block Diagram	365
19-3	Keyboard Status and Control Register (KBSCR)	368
19-4	Keyboard Interrupt Enable Register (KBIER)	369
20-1	COP Block Diagram	372
20-2	Configuration Register 1 (CONFIG1)	374
20-3	COP Control Register (COPCTL)	375
21-1	LVI I/O Register Summary	378
21-2	LVI Module Block Diagram	378
22-1	Break Module Block Diagram	385
22-2	Break Module I/O Register Summary	385
22-3	Break Status and Control Register (BRKSCR)	387
22-4	Break Address Register High (BRKH)	388
22-5	Break Address Register Low (BRKL)	388
22-6	SIM Break Status Register (SBSR)	389
22-7	SIM Break Flag Control Register (SBFCR)	390
23-1	SPI Master Timing	404
23-2	SPI Slave Timing	405
24-1	52-Pin Low-Profile Quad Flat Pack (Case No. 848D)	408
24-2	64-Pin Low-Profile Quad Flat Pack (Case No. 840F)	409
24-3	64-Pin Quad Flat Pack (Case No. 840B)	410

List of Tables

Table	Title	Page
2-1	Vector Addresses	58
5-1	LVI Trip Point Selection	76
6-1	Instruction Set Summary	86
6-2	Opcode Map	94
8-1	Numeric Examples	113
8-3	VPR1 and VPR0 Programming	120
8-2	PRE 1 and PRE0 Programming	120
9-1	Signal Name Conventions	133
9-2	PIN Bit Set Timing	137
9-3	Vector Addresses	146
10-1	Monitor Mode Signal Requirements and Options	160
10-2	Mode Differences (Vectors)	162
10-3	Monitor Baud Rate Selection	164
10-4	READ (Read Memory) Command	165
10-5	WRITE (Write Memory) Command	166
10-6	IREAD (Indexed Read) Command	166
10-7	IWRITE (Indexed Write) Command	167
10-8	READSP (Read Stack Pointer) Command	167
10-9	RUN (Run User Program) Command	168
10-10	Summary of ROM-Resident Routines	171
10-11	PRGRNGE Routine	173
10-12	ERARNGE Routine	175
10-13	LDRNGE Routine	176
10-14	MON_PRGRNGE Routine	177
10-15	MON_ERARNGE Routine	178

Table	Title	Page
10-16	ICP_LDRNGE Routine	179
10-17	EE_WRITE Routine	180
10-18	EE_READ Routine	183
11-1	Pin Name Conventions	187
11-2	Prescaler Selection	201
11-3	Mode, Edge, and Level Selection	206
12-1	CGMXCLK Frequency for RTC Input Reference	219
13-1	Pin Name Conventions	231
13-2	Start Bit Verification	242
13-3	Data Bit Recovery	242
13-4	Stop Bit Recovery	243
13-5	SCI Pin Functions (Standard and Infrared)	250
13-6	Character Format Selection	253
13-7	SCI Baud Rate Prescaling	264
13-8	SCI Baud Rate Selection	265
13-9	SCI Baud Rate Selection Examples	266
13-10	Infrared Narrow Pulse Selection	267
14-1	Pin Name Conventions	271
14-2	SPI Interrupts	286
14-3	SPI Configuration	293
14-4	SPI Master Baud Rate Selection	298
15-1	MUX Channel Select	311
15-2	ADC Clock Divide Ratio	314
15-3	ADC Mode Select	315
16-1	Pin Name Conventions	318
16-3	LCD Bias Voltage Control	336
16-2	Resistor Ladder Selection	336
16-4	Fast Charge Duty Cycle Selection	337
16-5	LCD Duty Cycle Selection	338
16-6	LCD Waveform Base Clock Selection	338

Table	Title	Page

17-1 Port Control Register Bits Summary 343
17-2 Port A Pin Functions 346
17-3 Port B Pin Functions 350
17-4 Port C Pin Functions 353
17-5 Port D Pin Functions 356

18-1 IRQ I/O Port Register Summary 359

19-1 Pin Name Conventions 364

21-1 LVI Status Register (LVISR) 381
21-2 LVIOUT Bit Indication 381

23-1 Absolute Maximum Ratings 392
23-2 Operating Range 393
23-3 Thermal Characteristics 393
23-4 5.0V DC Electrical Characteristics 394
23-5 3.3V DC Electrical Characteristics 396
23-6 5.0V Control Timing 397
23-7 3.3V Control Timing 397
23-8 5.0V Oscillator Specifications 398
23-9 3.3V Oscillator Specifications 398
23-10 ADC 5.0V Electrical Characteristics 399
23-11 ADC 3.3V Electrical Characteristics 400
23-12 FLASH Memory Electrical Characteristics 406

25-1 MC Order Numbers 411

List of Tables

Section 1. General Description

1.1 Contents

1.2 Introduction 34

1.3 Features 34

1.4 MCU Block Diagram 36

1.5 Pin Assignments 38

1.6 Pin Functions 40
1.6.1 Power Supply Pins (V_{DD} and V_{SS}) 40
1.6.2 Analog Power Supply Pin (V_{DDA}) 40
1.6.3 Oscillator Pins (OSC1 and OSC2) 41
1.6.4 External Reset Pin ($\overline{RST}$) 41
1.6.5 External Interrupt Pin ($\overline{IRQ}$) 41
1.6.6 External Filter Capacitor Pin (CGMXFC) 41
1.6.7 ADC Voltage High Reference Pin (V_{REFH}) 41
1.6.8 ADC Voltage Low Reference Pin (V_{REFL}) 41
1.6.9 Port A Input/Output (I/O) Pins (PTA7–PTA0) 42
1.6.10 Port B I/O Pins (PTB7–PTB0) 42
1.6.11 Port C I/O Pins (PTC7–PTC0) 42
1.6.12 Port D I/O Pins (PTD7–PTD0) 42
1.6.13 LCD Backplane and Frontplane (BP0–BP2, FP0/BP3, FP1–FP18) 42

1.2 Introduction

The MC68HC908LJ12 is a member of the low-cost, high-performance M68HC08 Family of 8-bit microcontroller units (MCUs). The M68HC08 Family is based on the customer-specified integrated circuit (CSIC) design strategy. All MCUs in the family use the enhanced M68HC08 central processor unit (CPU08) and are available with a variety of modules, memory sizes and types, and package types.

1.3 Features

Features of the MC68HC908LJ12 include the following:

- High-performance M68HC08 architecture
- Fully upward-compatible object code with M6805, M146805, and M68HC05 Families
- Maximum internal bus frequency:
  - 8-MHz at 5V operating voltage
  - 4-MHz at 3.3V operating voltage
- 32-kHz crystal oscillator clock input with 32MHz internal phase-lock-loop
- Optional continuous crystal oscillator operation in stop mode
- 12k-bytes user program FLASH memory with security[1] feature
- 512 bytes of on-chip RAM
- Two 16-bit, 2-channel timer interface modules (TIM1 and TIM2) with selectable input capture, output compare, and PWM capability on each channel
- Real time clock (RTC) with clock, calendar, alarm, and chronograph functions. Selectable periodic interrupt requests for seconds, minutes, hours, days, 2-Hz, 4-Hz, and 100-Hz
- Serial communications interface module (SCI) with infrared (IR) encoder/decoder

1. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

- Serial peripheral interface module (SPI)
- $\overline{\text{IRQ}}$ external interrupt pin with integrated pullup
- 8-bit keyboard wakeup port with programmable pullup
- 32 general-purpose input/output (I/O) pins:
  - High current 8-mA sink capability on PTB2–PTB5
  - High current 20-mA sink capability on PTB0–PTB1
- 4/3 backplanes and static with maximum 27 frontplanes liquid crystal display (LCD) driver
- 6-channel, 10-bit successive approximation analog-to-digital converter (ADC)
- Resident routines for in-circuit programming and EEPROM emulation
- Low-power design (fully static with stop and wait modes)
- Master reset pin (with integrated pullup) and power-on reset
- Spike filter protection for EMC performance enhancement
- System protection features
  - Optional computer operating properly (COP) reset, driven by internal 64-kHz RC oscillator
  - Low-voltage detection with optional reset or interrupt
  - Illegal opcode detection with reset
  - Illegal address detection with reset
- 64-pin quad flat pack (QFP), 64-pin low-profile quad flat pack (LQFP), 52-pin low-profile quad flat pack (LQFP), and die form
- Specific features of the MC68HC908LJ12 in 52-pin LQFP are:
  - 20 general-purpose I/Os only
  - High current 8-mA sink capability on PTB2–PTB3 only
  - 4-bit keyboard wakeup port with programmable pullup
  - No serial peripheral interface module (SPI)
  - No TIM2 input capture/output compare pins
  - 4-channel analog-to-digital converter only

Features of the CPU08 include the following:

- Enhanced HC05 programming model
- Extensive loop control functions
- 16 addressing modes (eight more than the HC05)
- 16-bit Index register and stack pointer
- Memory-to-memory data transfers
- Fast 8 × 8 multiply instruction
- Fast 16/8 divide instruction
- Binary-coded decimal (BCD) instructions
- Optimization for controller applications
- Efficient C language support

1.4 MCU Block Diagram

Figure 1-1 shows the structure of the MC68HC908LJ12.

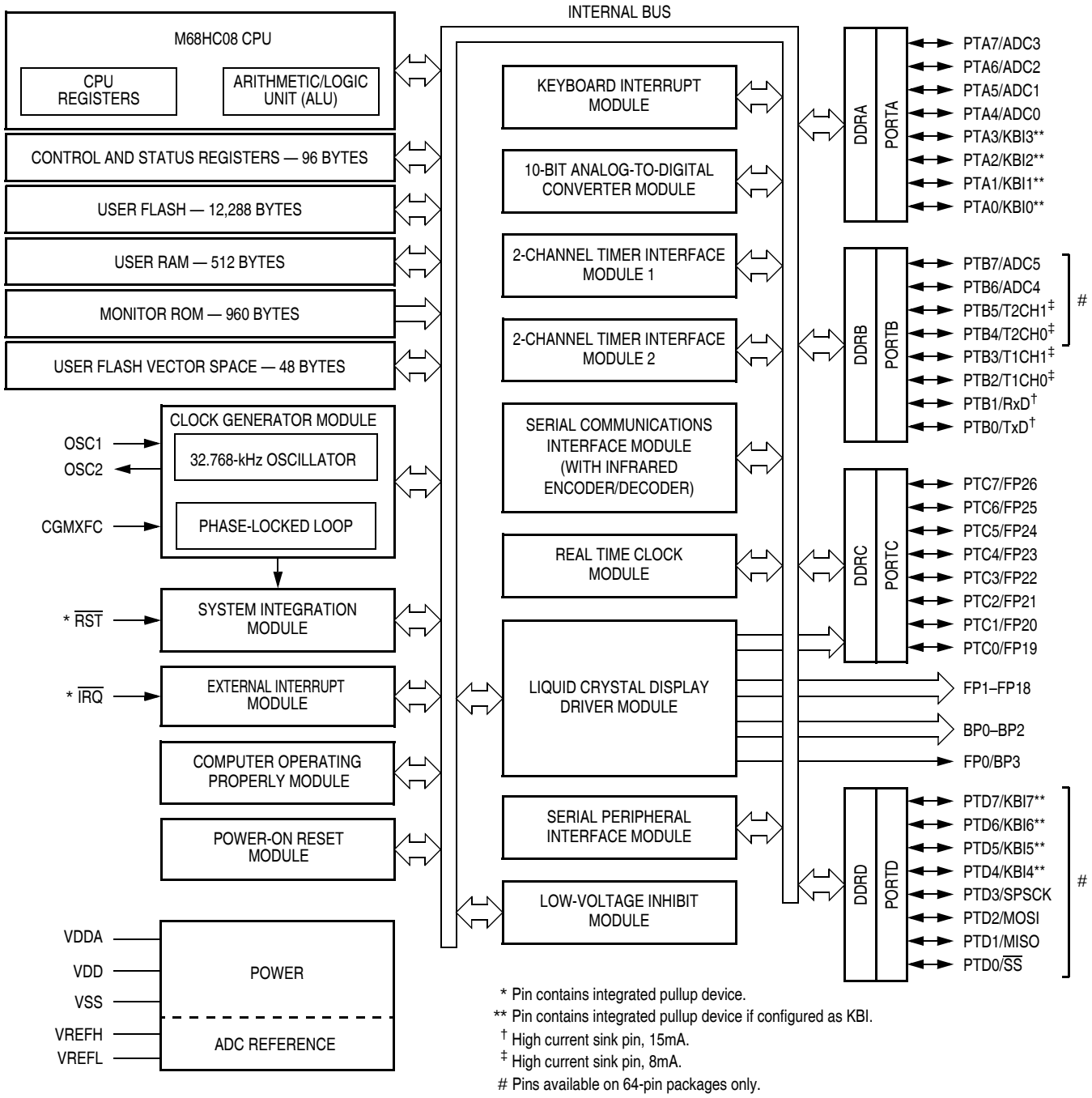


Figure 1-1. MC68HC908LJ12 Block Diagram

1.5 Pin Assignments

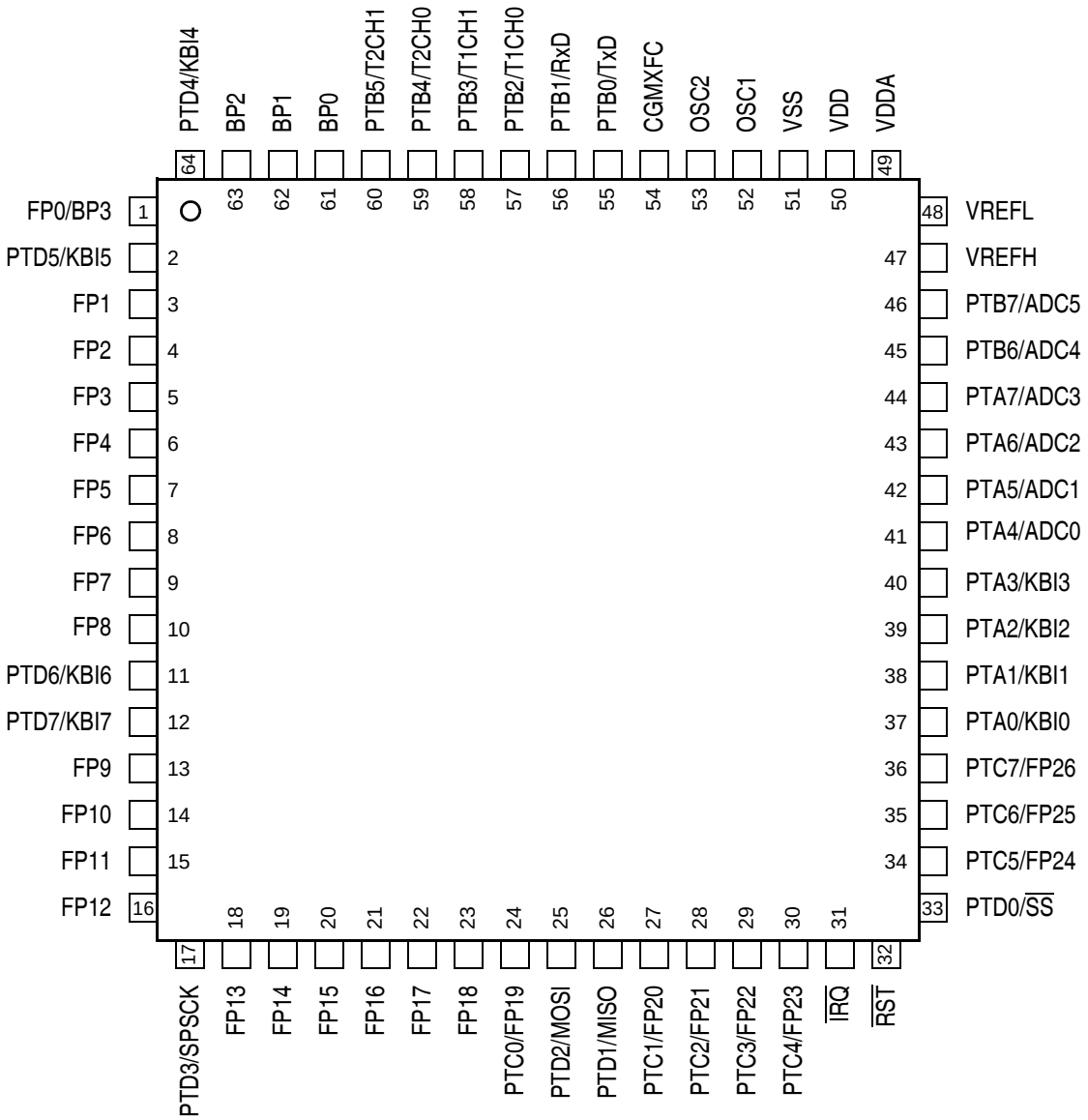
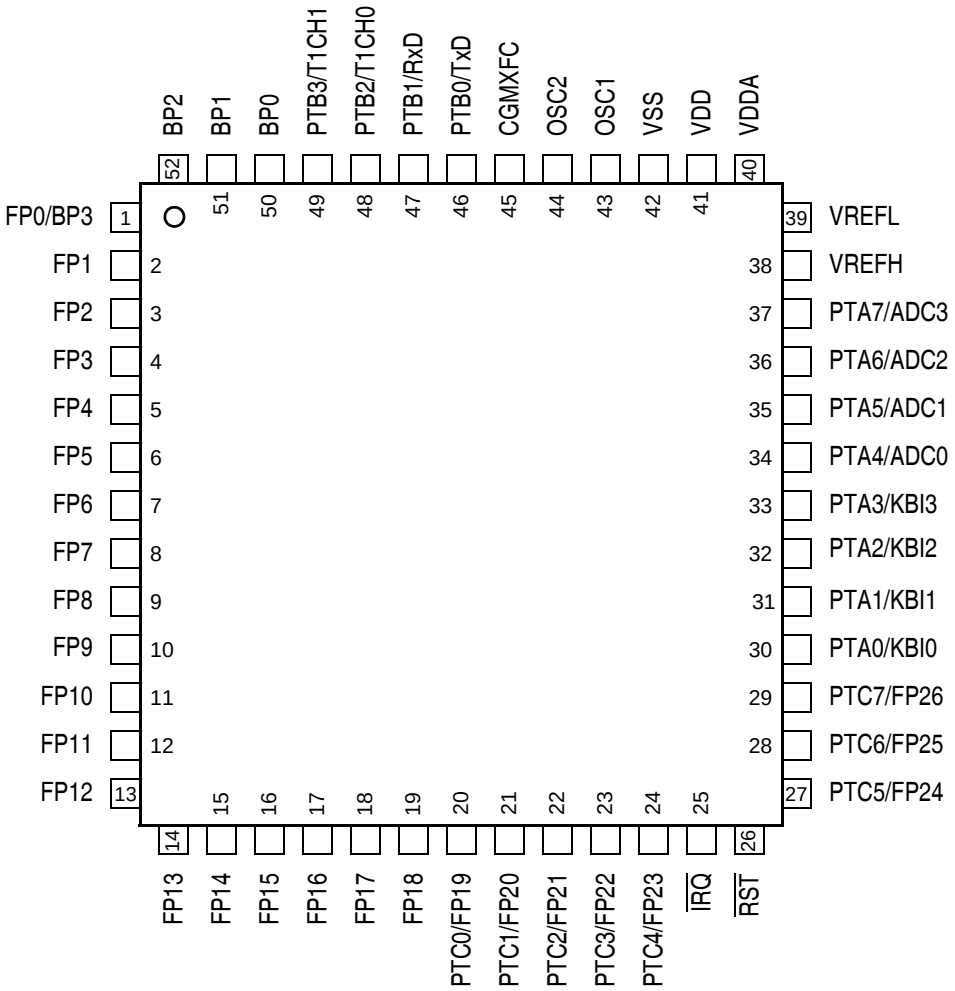


Figure 1-2. 64-Pin QFP and 64-Pin LQFP Pin Assignment



Pins not available on 52-LQFP package:	
PTB7ADC5	PTD7/KBI7
PTB6/ADC4	PTD6/KBI6
PTB5/T2CH1	PTD5/KBI5
PTB4/T2CH0	PTD4/KBI4
	PTD3/SPSCK
	PTD2/MOSI
	PTD1/MISO
	PTD0/$\overline{SS}$
Internal pads are unconnected.	

Figure 1-3. 52-Pin LQFP Pin Assignment

1.6 Pin Functions

Description of pin functions are provided here.

1.6.1 Power Supply Pins (V_{DD} and V_{SS})

V_{DD} and V_{SS} are the power supply and ground pins. The MCU operates from a single power supply.

Fast signal transitions on MCU pins place high, short-duration current demands on the power supply. To prevent noise problems, take special care to provide power supply bypassing at the MCU as **Figure 1-4** shows. Place the C1 bypass capacitor as close to the MCU as possible. Use a high-frequency-response ceramic capacitor for C1. C2 is an optional bulk current bypass capacitor for use in applications that require the port pins to source high current levels. V_{SS} must be grounded for proper MCU operation.

1.6.2 Analog Power Supply Pin (V_{DDA})

V_{DDA} is the voltage supply for the analog parts of the MCU. Connect the V_{DDA} pin to the same voltage potential as V_{DD}. For maximum noise immunity, route V_{DDA} via a separate trace and place bypass capacitors as close as possible to the package (see **Figure 1-4**).

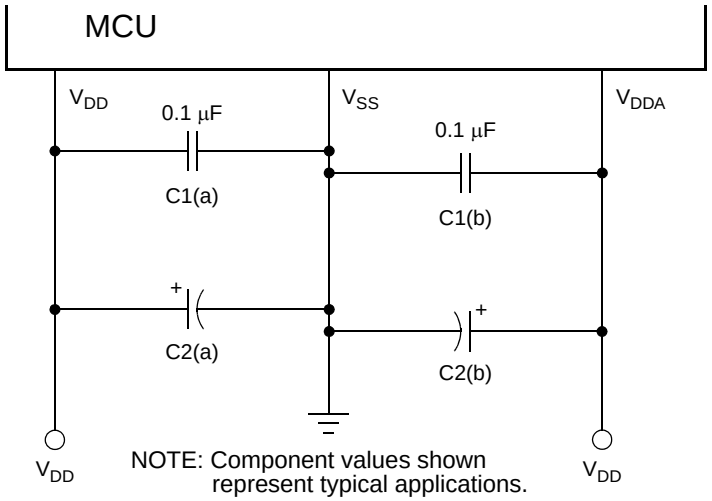


Figure 1-4. Power Supply Bypassing

1.6.3 Oscillator Pins (OSC1 and OSC2)

The OSC1 and OSC2 pins are the connections for the on-chip oscillator circuit. The OSC1 pin contains a schmitt-trigger and a spike filter for improved EMC performance. See **Section 7. Oscillator (OSC)**.

1.6.4 External Reset Pin ($\overline{RST}$)

A logic 0 on the $\overline{RST}$ pin forces the MCU to a known start-up state. $\overline{RST}$ is bidirectional, allowing a reset of the entire system. It is driven low when any internal reset source is asserted. A schmitt-trigger and a spike filter is associated with this pin so that the device is more robust to EMC noise.This pin also contains an internal pullup resistor. See **Section 9. System Integration Module (SIM)**.

1.6.5 External Interrupt Pin ($\overline{IRQ}$)

$\overline{IRQ}$ is an asynchronous external interrupt pin. This pin contains an internal pullup resistor. See **Section 18. External Interrupt (IRQ)**.

1.6.6 External Filter Capacitor Pin (CGMXFC)

CGMXFC is an external filter capacitor connection for the CGM. See **Section 8. Clock Generator Module (CGM)**.

1.6.7 ADC Voltage High Reference Pin (V_{REFH})

V_{REFH} is the voltage input pin for the ADC voltage high reference. See **Section 15. Analog-to-Digital Converter (ADC)**

1.6.8 ADC Voltage Low Reference Pin (V_{REFL})

V_{REFL} is the voltage input pin for the ADC voltage low reference. See **Section 15. Analog-to-Digital Converter (ADC)**

1.6.9 Port A Input/Output (I/O) Pins (PTA7–PTA0)

PTA7–PTA0 are special function, bidirectional port pins (**Section 17.**). PTA7/ADC3–PTA4/ADC0 are shared with the ADC (**Section 15.**), and PTA3/KBI3–PTA0/KBI0 are shared with the KBI module (**Section 19.**).

1.6.10 Port B I/O Pins (PTB7–PTB0)

PTB7–PTB0 are special function, bidirectional port pins (**Section 17.**). PTB0/TxD–PTB1/RxD are shared with the SCI module (**Section 13.**), PTB5/T2CH1–PTB4/T2CH0 are shared with the TIM2 (**Section 11.**), PTB3/T1CH1–PTB2/T1CH0 are shared with the TIM1(**Section 11.**), PTB6/ADC4–PTB7/ADC5 are shared with the ADC (**Section 15.**).

1.6.11 Port C I/O Pins (PTC7–PTC0)

PTC7–PTC0 are special function, bidirectional port pins (**Section 17.**). PTC7/FP26–PTC0/FP19 are shared with the LCD frontplane drivers (**Section 16.**).

1.6.12 Port D I/O Pins (PTD7–PTD0)

PTD7–PTD0 are special function, bidirectional port pins (**Section 17.**). PTD7/KBI7–PTD4/KBI4 are shared with KBI module (**Section 19.**). PTD3/SPSCK–PTD0/$\overline{SS}$ are shared with SPI module (**Section 14.**).

1.6.13 LCD Backplane and Frontplane (BP0–BP2, FP0/BP3, FP1–FP18)

BP0–BP2 are the LCD backplane driver pins and FP1– FP18 are the frontplane driver pins. FP0/BP3 is the shared driver pin between FP0 and BP3 (**Section 16.**).

Section 2. Memory Map

2.1 Contents

2.2 Introduction 43
2.3 Unimplemented Memory Locations 43
2.4 Reserved Memory Locations 44
2.5 Input/Output (I/O) Section 44

2.2 Introduction

The CPU08 can address 64k-bytes of memory space. The memory map, shown in **Figure 2-1**, includes:

- 12,288 bytes of user FLASH memory
- 512 bytes of random-access memory (RAM)
- 48 bytes of user-defined vectors
- 960 bytes of monitor ROM

2.3 Unimplemented Memory Locations

Accessing an unimplemented location can cause an illegal address reset if illegal address resets are enabled. In the memory map (**Figure 2-1**) and in register figures in this document, unimplemented locations are shaded.

2.4 Reserved Memory Locations

Accessing a reserved location can have unpredictable effects on MCU operation. In the **Figure 2-1** and in register figures in this document, reserved locations are marked with the word Reserved or with the letter R.

2.5 Input/Output (I/O) Section

Most of the control, status, and data registers are in the zero page area of $0000–$005F. Additional I/O registers have these addresses:

- $FE00; SIM break status register, SBSR
- $FE01; SIM reset status register, SRSR
- $FE02; Reserved
- $FE03; SIM break flag control register, SBFCR
- $FE04; Interrupt status register 1, INT1
- $FE05; Interrupt status register 2, INT2
- $FE06; Interrupt status register 3, INT3
- $FE07; Reserved
- $FE08; FLASH control register, FLCR
- $FE09; FLASH block protect register, FLBPR
- $FE0A; Reserved
- $FE0B; Reserved
- $FE0C; Break address register high, BRKH
- $FE0D; Break address register low, BRKL
- $FE0E; Break status and control register, BRKSCR
- $FE0F; LVI status register, LVISR
- $FFFF; COP control register, COPCTL

Data registers are shown in **Figure 2-2**. **Table 2-1** is a list of vector locations.

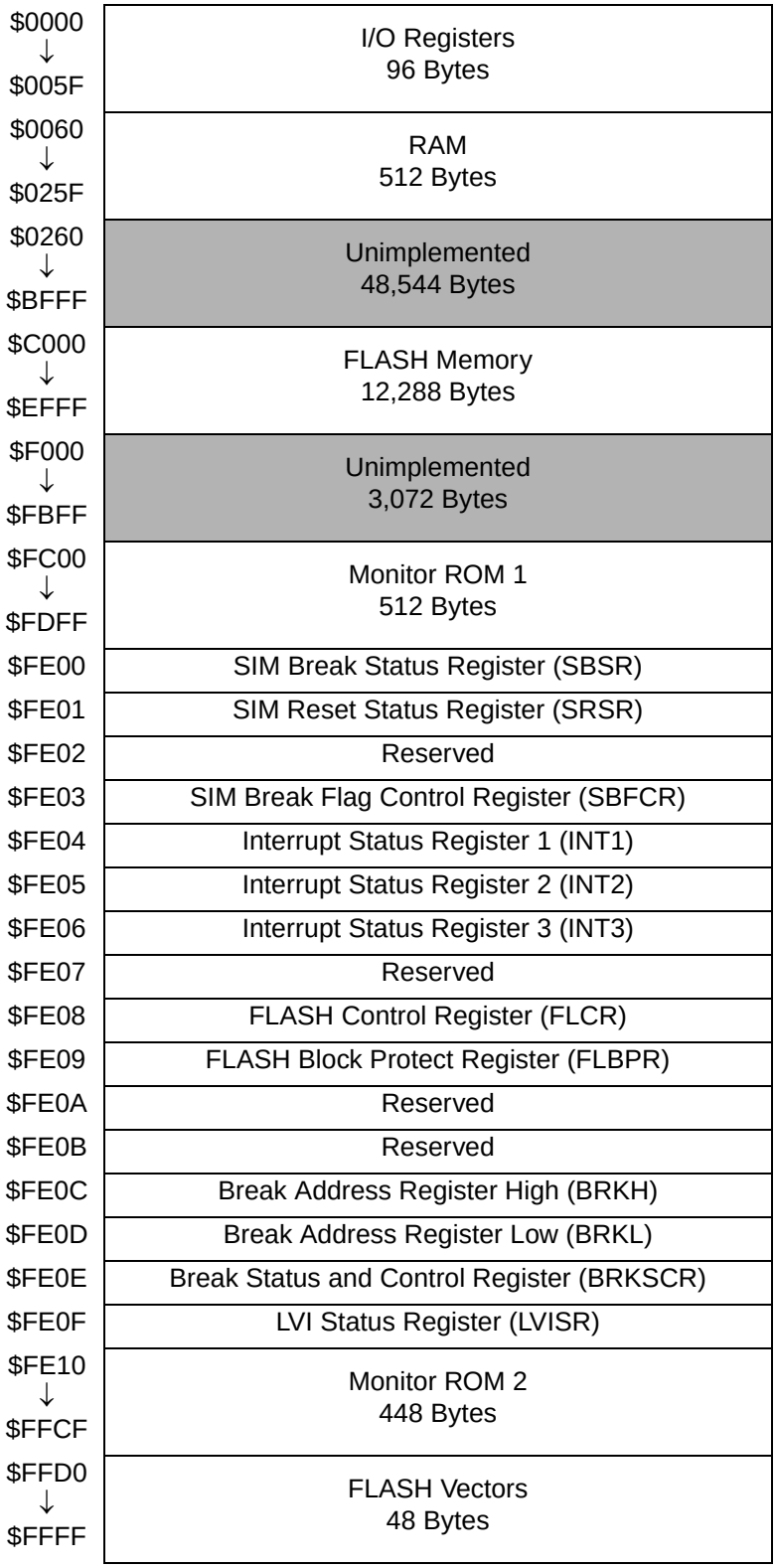


Figure 2-1. Memory Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PTA)	Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
		Reset:	U	U	U	U	U	U	U	U
$0001	Port B Data Register (PTB)	Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Reset:	U	U	U	U	U	U	U	U
$0002	Port C Data Register (PTC)	Read: Write:	PTC7	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
		Reset:	U	U	U	U	U	U	U	U
$0003	Port D Data Register (PTD)	Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Reset:	U	U	U	U	U	U	U	U
$0004	Data Direction Register A (DDRA)	Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B (DDRB)	Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C (DDRC)	Read: Write:	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Reset:	0	0	0	0	0	0	0	0
$0007	Data Direction Register D (DDRD)	Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$0008	Unimplemented	Read: Write:								
		Reset:								
$0009	Unimplemented	Read: Write:								
		Reset:								

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 1 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000A	Unimplemented	Read:								
		Write:								
		Reset:								
$000B	Unimplemented	Read:								
		Write:								
		Reset:								
$000C	Port B LED Control Register (LEDB)	Read:	0	0	LEDB5	LEDB4	LEDB3	LEDB2	LEDB1	LEDB0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000D	Unimplemented	Read:								
		Write:								
		Reset:								
$000E	Unimplemented	Read:								
		Write:								
		Reset:								
$000F	Unimplemented	Read:								
		Write:								
		Reset:								
$0010	SPI Control Register (SPCR)	Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Write:								
		Reset:	0	0	1	0	1	0	0	0
$0011	SPI Status and Control Register (SPSCR)	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	1	0	0	0
$0012	SPI Data Register (SPDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	U	U	U	U	U	U	U	U
$0013	SCI Control Register 1 (SCC1)	Read:	LOOPS	ENSCI	0	M	WAKE	ILTY	PEN	PTY
		Write:								
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 2 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0014	SCI Control Register 2 (SCC2)	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	SCI Control Register 3 (SCC3)	Read:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
		Write:								
		Reset:	U	U	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1)	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2)	Read:	0	0	0	0	0	0	BKF	RPF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0018	SCI Data Register (SCDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	U	U	U	U	U	U	U	U
$0019	SCI Baud Rate Register (SCBR)	Read:	CKS	0	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001A	SCI Infrared Control Register (SCIRCR)	Read:	R	0	0	0	R	TNP1	TNP0	IREN
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001B	Keyboard Status and Control Register (KBSCR)	Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
		Write:						ACKK		
		Reset:	0	0	0	0	0	0	0	0
$001C	Keyboard Interrupt Enable Register (KBIER)	Read:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001D	Configuration Register 2 (CONFIG2)†	Read:	0	STOP_ IRCDIS	STOP_ XCLKEN	DIV2CLK	PCEH	PCEL	LVISEL1	LVISEL0
		Write:								
		Reset:	0	0	0	0	0	0	0††	0††

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 3 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001E	IRQ Status and Control Register (INTSCR)	Read:	0	0	0	0	IRQF	0	IMASK	MODE
		Write:						ACK		
		Reset:	0	0	0	0	0	0	0	0
$001F	Configuration Register 1 (CONFIG1)†	Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	0	SSREC	STOP	COPD
		Write:								
		Reset:	0	0	0	1	0	0	0	0

† One-time writable register after each reset.
†† Reset by POR only.

$0020	Timer 1 Status and Control Register (T1SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0021	Timer 1 Counter Register High (T1CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0022	Timer 1 Counter Register Low (T1CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0023	Timer 1 Counter Modulo Register High (T1MODH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0024	Timer 1 Counter Modulo Register Low (T1MODL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0025	Timer 1 Channel 0 Status and Control Register (T1SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0026	Timer 1 Channel 0 Register High (T1CH0H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	X	X	X	X	X	X	X	X

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 4 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0027	Timer 1 Channel 0 Register Low (T1CH0L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$0028	Timer 1 Channel 1 Status and Control Register (T1SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0029	Timer 1 Channel 1 Register High (T1CH1H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$002A	Timer 1 Channel 1 Register Low (T1CH1L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$002B	Timer 2 Status and Control Register (T2SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$002C	Timer 2 Counter Register High (T2CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002D	Timer 2 Counter Register Low (T2CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002E	Timer 2 Counter Modulo Register High (T2MODH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$002F	Timer 2 Counter Modulo Register Low (T2MODL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0030	Timer 2 Channel 0 Status and Control Register (T2SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 5 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0031	Timer 2 Channel 0 Register High (T2CH0H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$0032	Timer 2 Channel 0 Register Low (T2CH0L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$0033	Timer 2 Channel 1 Status and Control Register (T2SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0034	Timer 2 Channel 1 Register High (T2CH1H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$0035	Timer 2 Channel 1 Register Low (T2CH1L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$0036	PLL Control Register (PTCL)	Read:	PLLIE	PLLF	PLLON	BCS	PRE1	PRE0	VPR1	VPR0
		Write:								
		Reset:	0	0	1	0	0	0	0	0
$0037	PLL Bandwidth Control Register (PBWC)	Read:	AUTO	LOCK	$\overline{\text{ACQ}}$	0	0	0	0	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0038	PLL Multiplier Select Register High (PMSH)	Read:	0	0	0	0	MUL11	MUL10	MUL9	MUL8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0039	PLL Multiplier Select Register Low (PMSL)	Read:	MUL7	MUL6	MUL5	MUL4	MUL3	MUL2	MUL1	MUL0
		Write:								
		Reset:	0	1	0	0	0	0	0	0
$003A	PLL VCO Range Select Register (PMRS)	Read:	VRS7	VRS6	VRS5	VRS4	VRS3	VRS2	VRS1	VRS0
		Write:								
		Reset:	0	1	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 6 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003B	PLL Reference Divider Select Register (PMDS)	Read:	0	0	0	0	RDS3	RDS2	RDS1	RDS0
		Write:								
		Reset:	0	0	0	0	0	0	0	1
$003C	ADC Status and Control Register (ADSCR)	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
$003D	ADC Data Register High (ADRH)	Read:	ADx	ADx	ADx	ADx	ADx	ADx	ADx	ADx
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003E	ADC Data Register Low (ADRL)	Read:	ADx	ADx	ADx	ADx	ADx	ADx	ADx	ADx
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003F	ADC Clock Register (ADCLK)	Read:	ADIV2	ADIV1	ADIV0	ADICLK	MODE1	MODE0	0	0
		Write:								R
		Reset:	0	0	0	0	0	1	0	0
$0040	Unimplemented	Read:								
		Write:								
		Reset:								
$0041	Unimplemented	Read:								
		Write:								
		Reset:								
$0042	RTC Control Register 1 (RTCCR1)	Read:	ALMIE	CHRIE	DAYIE	HRIE	MINIE	SECIE	TB1IE	TB2IE
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0043	RTC Control Register 2 (RTCCR2)	Read:	0	0	CHRE	RTCE	0	XTL2	XTL1	XTL0
		Write:	R	CHRCLR						
		Reset:	0	0	0	0	0	0	0	0
$0044	RTC Status Register (RTCSR)	Read:	ALMF	CHRF	DAYF	HRF	MINF	SECF	TB1F	TB2F
		Write:								
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 7 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0045	Alarm Minute Register (ALMR)	Read:	0	0	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0046	Alarm Hour Register (ALHR)	Read:	0	0	0	AH4	AH3	AH2	AH1	AH0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0047	Second Register (SECR)	Read:	0	0	SEC5	SEC4	SEC3	SEC2	SEC1	SEC0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0048	Minute Register (MINR)	Read:	0	0	MIN5	MIN4	MIN3	MIN2	MIN1	MIN0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0049	Hour Register (HRR)	Read:	0	0	0	HR4	HR3	HR2	HR1	HR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004A	Day Register (DAYR)	Read:	0	0	0	DAY4	DAY3	DAY2	DAY1	DAY0
		Write:								
		Reset:	0	0	0	0	0	0	0	1
$004B	Month Register (MTHR)	Read:	0	0	0	0	MTH3	MTH2	MTH1	MTH0
		Write:								
		Reset:	0	0	0	0	0	0	0	1
$004C	Year Register (YRR)	Read:	YR7	YR6	YR5	YR4	YR3	YR2	YR1	YR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004D	Day-Of-Week Register (DOWR)	Read:	0	0	0	0	0	DOW2	DOW1	DOW0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004E	Chronograph Data Register (CHRR)	Read:	0	CHR6	CHR5	CHR4	CHR3	CHR2	CHR1	CHR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 8 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$004F	LCD Clock Register (LCDCLK)	Read:	0	FCCTL1	FCCTL0	DUTY1	DUTY0	LCLK2	LCLK1	LCLK0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0050	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$0051	LCD Control Register (LCDCR)	Read:	LCDE	0	FC	LC	LCCON3	LCCON2	LCCON1	LCCON0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0052	LCD Data Register 1 (LDAT1)	Read:	F1B3	F1B2	F1B1	F1B0	F0B3	F0B2	F0B1	F0B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0053	LCD Data Register 2 (LDAT2)	Read:	F3B3	F3B2	F3B1	F3B0	F2B3	F2B2	F2B1	F2B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0054	LCD Data Register 3 (LDAT3)	Read:	F5B3	F5B2	F5B1	F5B0	F4B3	F4B2	F4B1	F4B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0055	LCD Data Register 4 (LDAT4)	Read:	F7B3	F7B2	F7B1	F7B0	F6B3	F6B2	F6B1	F6B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0056	LCD Data Register 5 (LDAT5)	Read:	F9B3	F9B2	F9B1	F9B0	F8B3	F8B2	F8B1	F8B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0057	LCD Data Register 6 (LDAT6)	Read:	F11B3	F11B2	F11B1	F11B0	F10B3	F10B2	F10B1	F10B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0058	LCD Data Register 7 (LDAT7)	Read:	F13B3	F13B2	F13B1	F13B0	F12B3	F12B2	F12B1	F12B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 9 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0059	LCD Data Register 8 (LDAT8)	Read: Write:	F15B3	F15B2	F15B1	F15B0	F14B3	F14B2	F14B1	F14B0
		Reset:	U	U	U	U	U	U	U	U
$005A	LCD Data Register 9 (LDAT9)	Read: Write:	F17B3	F17B2	F17B1	F17B0	F16B3	F16B2	F16B1	F16B0
		Reset:	U	U	U	U	U	U	U	U
$005B	LCD Data Register 10 (LDAT10)	Read: Write:	F19B3	F19B2	F19B1	F19B0	F18B3	F18B2	F18B1	F18B0
		Reset:	U	U	U	U	U	U	U	U
$005C	LCD Data Register 11 (LDAT11)	Read: Write:	F21B3	F21B2	F21B1	F21B0	F20B3	F20B2	F20B1	F20B0
		Reset:	U	U	U	U	U	U	U	U
$005D	LCD Data Register 12 (LDAT12)	Read: Write:	F23B3	F23B2	F23B1	F23B0	F22B3	F22B2	F22B1	F22B0
		Reset:	U	U	U	U	U	U	U	U
$005E	LCD Data Register 13 (LDAT13)	Read: Write:	F25B3	F25B2	F25B1	F25B0	F24B3	F24B2	F24B1	F24B0
		Reset:	U	U	U	U	U	U	U	U
$005F	LCD Data Register 14 (LDAT14)	Read: Write:					F26B3	F26B2	F26B1	F26B0
		Reset:	U	U	U	U	U	U	U	U
$FE00	SIM Break Status Register (SBSR)	Read: Write:	R	R	R	R	R	R	SBSW Note	R
		Reset:							0	

Note: Writing a logic 0 clears SBSW.

$FE01	SIM Reset Status Register (SRSR)	Read:	POR	PIN	COP	ILOP	ILAD	0	LVI	0
		Write:								
		POR:	1	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 10 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE02	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$FE03	SIM Break Flag Control Register (SBFCR)	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
$FE04	Interrupt Status Register 1 (INT1)	Read:	IF6	IF5	IF4	IF3	IF2	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE05	Interrupt Status Register 2 (INT2)	Read:	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE06	Interrupt Status Register 3 (INT3)	Read:	0	0	0	0	0	IF17	IF16	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE07	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$FE08	FLASH Control Register (FLCR)	Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE09	FLASH Block Protect Register (FLBPR)	Read:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0A	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$FE0B	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 11 of 12)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE0C	Break Address Register High (BRKH)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$FE0D	Break Address Register Low (BRKL)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$FE0E	Break Status and Control Register (BRKSCR)	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0F	Low-Voltage Inhibit Status Register (LVISR)	Read:	LVIOUT	LVIIE	LVIIF	0	0	0	0	0
		Write:				LVIIAK				
		Reset:	0	0	0	0	0	0	0	0
$FFFF	COP Control Register (COPCTL)	Read:	Low byte of reset vector							
		Write:	Writing clears COP counter (any value)							
		Reset:	Unaffected by reset							

U = Unaffected X = Indeterminate = Unimplemented R = Reserved

Figure 2-2. Control, Status, and Data Registers (Sheet 12 of 12)

Table 2-1. Vector Addresses

Priority	INT Flag	Address	Vector
Lowest	IF17	$FFDA	Real Time Clock Vector (High)
		$FFDB	Real Time Clock Vector (Low)
	IF16	$FFDC	ADC Conversion Complete Vector (High)
		$FFDD	ADC Conversion Complete Vector (Low)
	IF15	$FFDE	Keyboard Vector (High)
		$FFDF	Keyboard Vector (Low)
	IF14	$FFE0	SCI Transmit Vector (High)
		$FFE1	SCI Transmit Vector (Low)
	IF13	$FFE2	SCI Receive Vector (High)
		$FFE3	SCI Receive Vector (Low)
	IF12	$FFE4	SCI Error Vector (High)
		$FFE5	SCI Error Vector (Low)
	IF11	$FFE6	SPI Receive Vector (High)
		$FFE7	SPI Receive Vector (Low)
	IF10	$FFE8	SPI Transmit Vector (High)
		$FFE9	SPI Transmit Vector (Low)
	IF9	$FFEA	TIM2 Overflow Vector (High)
		$FFEB	TIM2 Overflow Vector (Low)
	IF8	$FFEC	TIM2 Channel 1 Vector (High)
		$FFED	TIM2 Channel 1 Vector (Low)
	IF7	$FFEE	TIM2 Channel 0 Vector (High)
		$FFEF	TIM2 Channel 0 Vector (Low)
	IF6	$FFF0	TIM1 Overflow Vector (High)
		$FFF1	TIM1 Overflow Vector (Low)
	IF5	$FFF2	TIM1 Channel 1 Vector (High)
		$FFF3	TIM1 Channel 1 Vector (Low)
	IF4	$FFF4	TIM1 Channel 0 Vector (High)
		$FFF5	TIM1 Channel 0 Vector (Low)
	IF3	$FFF6	PLL Vector (High)
		$FFF7	PLL Vector (Low)
	IF2	$FFF8	LVI Vector (High)
		$FFF9	LVI Vector (Low)
	IF1	$FFFA	$\overline{\text{IRQ}}$ Vector (High)
		$FFFB	$\overline{\text{IRQ}}$ Vector (Low)
	—	$FFFC	SWI Vector (High)
		$FFFD	SWI Vector (Low)
Highest	—	$FFFE	Reset Vector (High)
		$FFFF	Reset Vector (Low)

Section 3. Random-Access Memory (RAM)

3.1 Contents

3.2 Introduction 59
3.3 Functional Description 59

3.2 Introduction

This section describes the 512 bytes of RAM (random-access memory).

3.3 Functional Description

Addresses $0060 through $025F are RAM locations. The location of the stack RAM is programmable. The 16-bit stack pointer allows the stack to be anywhere in the 64k-byte memory space.

NOTE: *For correct operation, the stack pointer must point only to RAM locations.*

Within page zero are 160 bytes of RAM. Because the location of the stack RAM is programmable, all page zero RAM locations can be used for I/O control and user data or code. When the stack pointer is moved from its reset location at $00FF out of page zero, direct addressing mode instructions can efficiently access all page zero RAM locations. Page zero RAM, therefore, provides ideal locations for frequently accessed global variables.

Before processing an interrupt, the CPU uses five bytes of the stack to save the contents of the CPU registers.

NOTE: *For M6805 compatibility, the H register is not stacked.*

During a subroutine call, the CPU uses two bytes of the stack to store the return address. The stack pointer decrements during pushes and increments during pulls.

NOTE: *Be careful when using nested subroutines. The CPU may overwrite data in the RAM during a subroutine or during the interrupt stacking operation.*

Section 4. FLASH Memory (FLASH)

4.1 Contents

4.2 Introduction 61
4.3 Functional Description 62
4.4 FLASH Control Register 63
4.5 FLASH Page Erase Operation 64
4.6 FLASH Mass Erase Operation 65
4.7 FLASH Program Operation. 66
4.8 FLASH Protection 68
4.8.1 FLASH Block Protect Register 68

4.2 Introduction

This section describes the operation of the embedded FLASH memory. This memory can be read, programmed, and erased from a single external supply. The program and erase operations are enabled through the use of an internal charge pump.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE08	FLASH Control Register (FLCR)	Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE09	FLASH Block Protect Register (FLBPR)	Read:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 4-1. FLASH I/O Register Summary

4.3 Functional Description

The FLASH memory consists of an array of 12,288 bytes for user memory plus a block of 48 bytes for user interrupt vectors. *An erased bit reads as logic 1 and a programmed bit reads as a logic 0*. The FLASH memory page size is defined as 128 bytes, and is the minimum size that can be erased in a page erase operation. Program and erase operations are facilitated through control bits in FLASH control register (FLCR). The address ranges for the FLASH memory are:

- $C000–$EFFF; user memory; 12,288 bytes
- $FFD0–$FFFF; user interrupt vectors; 48 bytes

Programming tools are available from Freescale. Contact your local Freescale representative for more information.

NOTE: *A security feature prevents viewing of the FLASH contents.*[1]

1. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

4.4 FLASH Control Register

The FLASH control register (FLCR) controls FLASH program and erase operations.

Address: $FE08

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 4-2. FLASH Control Register (FLCR)

HVEN — High Voltage Enable Bit

This read/write bit enables the charge pump to drive high voltages for program and erase operations in the array. HVEN can only be set if either PGM = 1 or ERASE = 1 and the proper sequence for program or erase is followed.

1 = High voltage enabled to array and charge pump on
0 = High voltage disabled to array and charge pump off

MASS — Mass Erase Control Bit

This read/write bit configures the memory for mass erase operation or block erase operation when the ERASE bit is set.

1 = Mass Erase operation selected
0 = Block Erase operation selected

ERASE — Erase Control Bit

This read/write bit configures the memory for erase operation. ERASE is interlocked with the PGM bit such that both bits cannot be equal to 1 or set to 1 at the same time.

1 = Erase operation selected
0 = Erase operation not selected

PGM — Program Control Bit

This read/write bit configures the memory for program operation. PGM is interlocked with the ERASE bit such that both bits cannot be equal to 1 or set to 1 at the same time.

1 = Program operation selected
0 = Program operation not selected

4.5 FLASH Page Erase Operation

Use the following procedure to erase a page of FLASH memory. A page consists of 128 consecutive bytes starting from addresses $xx00 or $xx80. The 48-byte user interrupt vectors area also forms a page. *The 48-byte user interrupt vectors cannot be erased by the page erase operation because of security reasons. Mass erase is required to erase this page.*

1. Set the ERASE bit and clear the MASS bit in the FLASH control register.
2. Write any data to any FLASH address within the page address range desired.
3. Wait for a time, t_{nvs} (at least 10 μs).
4. Set the HVEN bit.
5. Wait for a time, t_{erase} (1 ms).
6. Clear the ERASE bit.
7. Wait for a time, t_{nvh} (5 μs).
8. Clear the HVEN bit.
9. After time, t_{rcv} (1 μs), the memory can be accessed again in read mode.

NOTE: *Programming and erasing of FLASH locations cannot be performed by executing code from the FLASH memory; the code must be executed from RAM. While these operations must be performed in the order as shown, but other unrelated operations may occur between the steps.*

4.6 FLASH Mass Erase Operation

Use the following procedure to erase the entire FLASH memory to read as logic 1:

1. Set both the ERASE bit and the MASS bit in the FLASH control register.
2. Write any data to any FLASH address within the FLASH memory address range.
3. Wait for a time, t_{nvs} (10 µs).
4. Set the HVEN bit.
5. Wait for a time t_{merase} (4 ms).
6. Clear the ERASE bit.
7. Wait for a time, t_{nvhl} (100 µs).
8. Clear the HVEN bit.
9. After time, t_{rcv} (1 µs), the memory can be accessed again in read mode.

NOTE: *Programming and erasing of FLASH locations cannot be performed by executing code from the FLASH memory; the code must be executed from RAM. While these operations must be performed in the order as shown, but other unrelated operations may occur between the steps.*

4.7 FLASH Program Operation

Programming of the FLASH memory is done on a row basis. A row consists of 64 consecutive bytes starting from addresses $xx00, $xx40, $xx80, or $xxC0. The procedure for programming a row of the FLASH memory is outlined below:

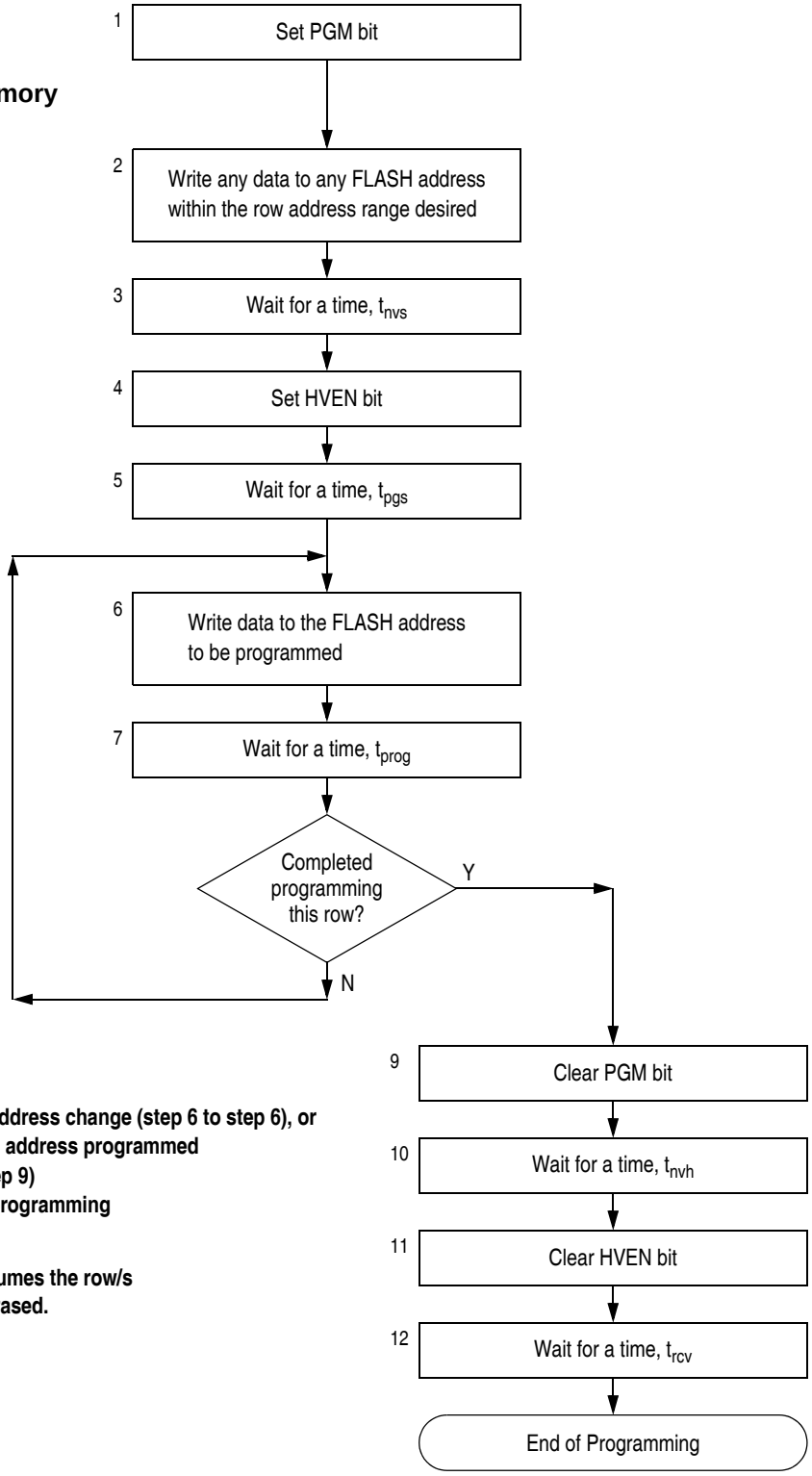
1. Set the PGM bit. This configures the memory for program operation and enables the latching of address and data for programming.
2. Write any data to any FLASH address within the row address range desired.
3. Wait for a time, t_{nvs} (10 µs).
4. Set the HVEN bit.
5. Wait for a time, t_{pgs} (5 µs).
6. Write data to the FLASH address to be programmed.
7. Wait for time, t_{prog} (30 µs).
8. Repeat step 6 and 7 until all the bytes within the row are programmed.
9. Clear the PGM bit.
10. Wait for time, t_{nvh} (5 µs).
11. Clear the HVEN bit.
12. After time, t_{rcv} (1 µs), the memory can be accessed again in read mode.

This program sequence is repeated throughout the memory until all data is programmed.

NOTE: *Programming and erasing of FLASH locations cannot be performed by executing code from the FLASH memory; the code must be executed from RAM. While these operations must be performed in the order as shown, but other unrelated operations may occur between the steps. Do not exceed t_{prog} maximum. See* ***23.18 FLASH Memory Characteristics****.*

Figure 4-3 shows a flowchart representation for programming the FLASH memory.

Algorithm for programming a row (64 bytes) of FLASH memory



NOTE:

The time between each FLASH address change (step 6 to step 6), or the time between the last FLASH address programmed to clearing PGM bit (step 6 to step 9) must not exceed the maximum programming time, t_{PROG} max.

This row program algorithm assumes the row/s to be programmed are initially erased.

Figure 4-3. FLASH Programming Flowchart

4.8 FLASH Protection

Due to the ability of the on-board charge pump to erase and program the FLASH memory in the target application, provision is made to protect pages of memory from unintentional erase or program operations due to system malfunction. This protection is done by use of a FLASH block protect register (FLBPR). The FLBPR determines the range of the FLASH memory which is to be protected. The range of the protected area starts from a location defined by FLBPR and ends to the bottom of the FLASH memory ($FFFF). When the memory is protected, the HVEN bit cannot be set in either erase or program operations.

NOTE: *When the FLBPR is cleared (all 0's), the entire FLASH memory is protected from being programmed and erased. When all the bits are set, the entire FLASH memory is accessible for program and erase.*

4.8.1 FLASH Block Protect Register

The FLASH block protect register is implemented as an 8-bit I/O register. The content of this register determine the starting location of the protected range within the FLASH memory.

Address: $FE09

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
Reset:	0	0	0	0	0	0	0	0

Figure 4-4. FLASH Block Protect Register (FLBPR)

BPR[7:0] — FLASH Block Protect Register Bit 7 to Bit 0

BPR[7:1] represent bits [13:7] of a 16-bit memory address. Bits [15:14] are logic 1's and bits [6:0] are logic 0's.

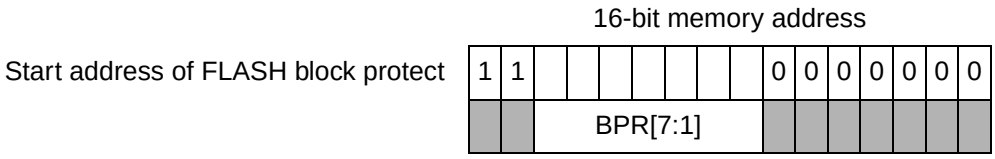


Figure 4-5. FLASH Block Protect Start Address

BPR0 is used only for BPR[7:0] = $FF, for no block protection.

The resultant 16-bit address is used for specifying the start address of the FLASH memory for block protection. The FLASH is protected from this start address to the end of FLASH memory, at $FFFF. With this mechanism, the protect start address can be XX00 or XX80 (at page boundaries — 128 bytes) within the FLASH memory.

Examples of protect start address:

BPR[7:0]	Start of Address of Protect Range
$00 or $01	$C000 (11**00 0000 0**000 0000) The entire FLASH memory is protected.
$02 or $03	$C080 (11**00 0000 1**000 0000)
$04 or $05	$C100 (11**00 0001 0**000 0000)
$06 or $07	$C180 (11**00 0001 1**000 0000)
$08 or $09	$C200 (11**00 0010 0**000 0000)
and so on...	
$F8 or $F9	$FE00 (11**11 1110 0**000 0000)
$FA or $FB	$FE80 (11**11 1110 1**000 0000)
$FC or $FD	$FF00 (11**11 1111 0**000 0000)
$FE	$FF80 (11**11 1111 1**000 0000)
$FF	The entire FLASH memory is not protected.

Note:
The end address of the protected range is always $FFFF.

Section 5. Configuration Registers (CONFIG)

5.1 Contents

5.2 Introduction 71
5.3 Functional Description 72
5.4 Configuration Register 1 (CONFIG1) 73
5.5 Configuration Register 2 (CONFIG2) 75

5.2 Introduction

This section describes the configuration registers, CONFIG1 and CONFIG2. The configuration registers enable or disable these options:

- Computer operating properly module (COP)
- COP timeout period ($2^{18} – 2^4$ or $2^{13} – 2^4$ ICLK cycles)
- Low-voltage inhibit (LVI) module power
- LVI module reset
- LVI module in stop mode
- LVI module voltage trip point selection
- STOP instruction
- Stop mode recovery time (32 ICLK cycles or 4096 ICLK cycles)
- Oscillator during stop mode
- LCD frontplanes FP19–FP26 on port C

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001D	Configuration Register 2 (CONFIG2)†	Read:	0	STOP_ IRCDIS	STOP_ XCLKEN	DIV2CLK	PCEH	PCEL	LVISEL1	LVISEL0
		Write:								
		Reset:	0	0	0	0	0	0	0††	0††
$001F	Configuration Register 1 (CONFIG1)†	Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	0	SSREC	STOP	COPD
		Write:								
		Reset:	0	0	0	1	0	0	0	0

† One-time writable register after each reset.

†† Reset by POR only.

= Unimplemented

Figure 5-1. CONFIG Registers Summary

5.3 Functional Description

The configuration registers are used in the initialization of various options. The configuration registers can be written once after each reset. All of the configuration register bits are cleared during reset. Since the various options affect the operation of the MCU, it is recommended that these registers be written immediately after reset. The configuration registers are located at $001D and $001F. The configuration registers may be read at anytime.

NOTE: *The options except LVISEL[1:0] are one-time writable by the user after each reset. The LVISEL[1:0] bits are one-time writable by the user only after each POR (power-on reset). The CONFIG registers are not in the FLASH memory but are special registers containing one-time writable latches after each reset. Upon a reset, the CONFIG registers default to predetermined settings as shown in* ***Figure 5-2*** *and* ***Figure 5-3****.*

5.4 Configuration Register 1 (CONFIG1)

Address: $001F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	0	SSREC	STOP	COPD
Write:								
Reset:	0	0	0	1	0	0	0	0

= Unimplemented

Figure 5-2. Configuration Register 1 (CONFIG1)

COPRS — COP Rate Select

COPRS selects the COP time-out period. Reset clears COPRS. (See **Section 20. Computer Operating Properly (COP)**.)

1 = COP time out period = $2^{13} – 2^4$ ICLK cycles
0 = COP time out period = $2^{18} – 2^4$ ICLK cycles

LVISTOP — LVI Enable in Stop Mode

When the LVIPWRD bit is clear, setting the LVISTOP bit enables the LVI to operate during stop mode. Reset clears LVISTOP. (See **Section 21. Low-Voltage Inhibit (LVI)**.)

1 = LVI enabled during stop mode
0 = LVI disabled during stop mode

LVIRSTD — LVI Reset Disable

LVIRSTD disables the reset signal from the LVI module. (See **Section 21. Low-Voltage Inhibit (LVI)**.)

1 = LVI module resets disabled
0 = LVI module resets enabled

LVIPWRD — LVI Power Disable Bit

LVIPWRD disables the LVI module. (See **Section 21. Low-Voltage Inhibit (LVI)**.) Reset sets LVIPWRD.

1 = LVI module power disabled
0 = LVI module power enabled

SSREC — Short Stop Recovery

SSREC enables the CPU to exit stop mode with a delay of 32 ICLK cycles instead of a 4096 ICLK cycle delay.

1 = Stop mode recovery after 32 ICLK cycles
0 = Stop mode recovery after 4096 ICLK cycles

NOTE: *Exiting stop mode by pulling reset will result in the long stop recovery.*

If using an external crystal oscillator, do not set the SSREC bit.

NOTE: *When the LVISTOP is enabled, the system stabilization time for power on reset and long stop recovery (both 4096 ICLK cycles) gives a delay longer than the enable time for the LVI. There is no period where the MCU is not protected from a low power condition. However, when using the short stop recovery configuration option, the 32 ICLK delay is less than the LVI's turn-on time and there exists a period in start-up where the LVI is not protecting the MCU.*

STOP — STOP Instruction Enable

STOP enables the STOP instruction.

1 = STOP instruction enabled
0 = STOP instruction treated as illegal opcode

COPD — COP Disable Bit

COPD disables the COP module. (See **Section 20. Computer Operating Properly (COP)**.)

1 = COP module disabled
0 = COP module enabled

5.5 Configuration Register 2 (CONFIG2)

Address: $001D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	STOP_ IRCDIS	STOP_ XCLKEN	DIV2CLK	PCEH	PCEL	LVISEL1	LVISEL0
Write:								
Reset:	0	0	0	0	0	0	0††	0††

= Unimplemented

†† Reset by POR only.

Figure 5-3. Configuration Register 2 (CONFIG2)

STOP_IRCDIS — Internal RC Oscillator Stop Mode Disable

Setting STOP_IRCDIS disables the internal RC oscillator during stop mode. When this bit is cleared, the internal RC oscillator continues to operate in stop mode. Reset clears this bit.

1 = Internal RC oscillator disabled during stop mode
0 = Internal RC oscillator enabled during stop mode

STOP_XCLKEN — Crystal Oscillator Stop Mode Enable

Setting STOP_XCLKEN enables the external crystal (XTAL) oscillator to continue operating during stop mode. This is useful for driving the real time clock module to allow it to generate periodic wake-up while in stop mode. When this bit is cleared, the external XTAL oscillator will be disabled during stop mode. Reset clears this bit.

1 = XTAL oscillator enabled during stop mode
0 = XTAL oscillator disabled during stop mode

DIV2CLK — Divide-by-2 Clock Bypass

When CGMXCLK is selected to drive the system clocks (BCS=0), setting DIV2CLK allows the CGMXCLK to bypass the divide-by-2 divider in the CGM module; CGMOUT will equal CGMXCLK and bus clock will equal CGMXCLK divide-by-2.

DIV2CLK bit has no effect when the BCS=1 in the PLL control register (CGMVCLK selected and divide-by-2 always enabled). Reset clears this bit.

1 = Divide-by-2 divider bypassed;
When BSC=0, CGMOUT equals CGMXCLK
0 = Divide-by-2 divider enabled;
When BSC=0, CGMOUT equals CGMXCLK divide-by-2

PCEH — Port C Enable High Nibble

Setting PCEH configures the PTC4/FP23–PTC7/FP26 pins for LCD frontplane driver use. Reset clears this bit.

1 = PTC4/FP23–PTC7/FP26 pins configured as LCD frontplane driver pins: FP23–FP26

0 = PTC4/FP23–PTC7/FP26 pins configured as standard I/O pins: PTC4–PTC7

PCEL — Port C Enable Low Nibble

Setting PCEL configures the PTC0/FP19–PTC3/FP22 pins for LCD frontplane driver use. Reset clears this bit.

1 = PTC0/FP19–PTC3/FP22 pins configured as LCD frontplane driver pins: FP19–FP22

0 = PTC0/FP19–PTC3/FP22 pins configured as standard I/O pins: PTC0–PTC3

LVISEL[1:0] — LVI Operating Mode Selection

LVISEL[1:0] selects the voltage operating mode of the LVI module. (See **Section 21. Low-Voltage Inhibit (LVI)**.) The voltage mode selected for the LVI should match the operating V_{DD}. See **Section 23. Electrical Specifications** for the LVI voltage trip points for each of the modes.

LVISEL1	LVISEL0	Operating Mode
0	0	Reserved (2.5V)
0	1	3V
1	0	5V
1	1	Reserved

Table 5-1. LVI Trip Point Selection

Section 6. Central Processor Unit (CPU)

6.1 Contents

6.2 Introduction 78

6.3 Features 78

6.4 CPU Registers 79
6.4.1 Accumulator 79
6.4.2 Index Register 80
6.4.3 Stack Pointer 80
6.4.4 Program Counter 81
6.4.5 Condition Code Register 82

6.5 Arithmetic/Logic Unit (ALU) 84

6.6 Low-Power Modes 84
6.6.1 Wait Mode 84
6.6.2 Stop Mode 85

6.7 CPU During Break Interrupts 85

6.8 Instruction Set Summary 85

6.9 Opcode Map 85

6.2 Introduction

The M68HC08 CPU (central processor unit) is an enhanced and fully object-code-compatible version of the M68HC05 CPU. The *CPU08 Reference Manual* (Freescale document order number CPU08RM/AD) contains a description of the CPU instruction set, addressing modes, and architecture.

6.3 Features

Feature of the CPU include:

- Object code fully upward-compatible with M68HC05 Family
- 16-bit stack pointer with stack manipulation instructions
- 16-Bit index register with X-register manipulation instructions
- 8-MHz CPU internal bus frequency
- 64-Kbyte program/data memory space
- 16 addressing modes
- Memory-to-memory data moves without using accumulator
- Fast 8-bit by 8-bit multiply and 16-bit by 8-bit divide instructions
- Enhanced binary-coded decimal (BCD) data handling
- Modular architecture with expandable internal bus definition for extension of addressing range beyond 64-Kbytes
- Low-power stop and wait modes

6.4 CPU Registers

Figure 6-1 shows the five CPU registers. CPU registers are not part of the memory map.

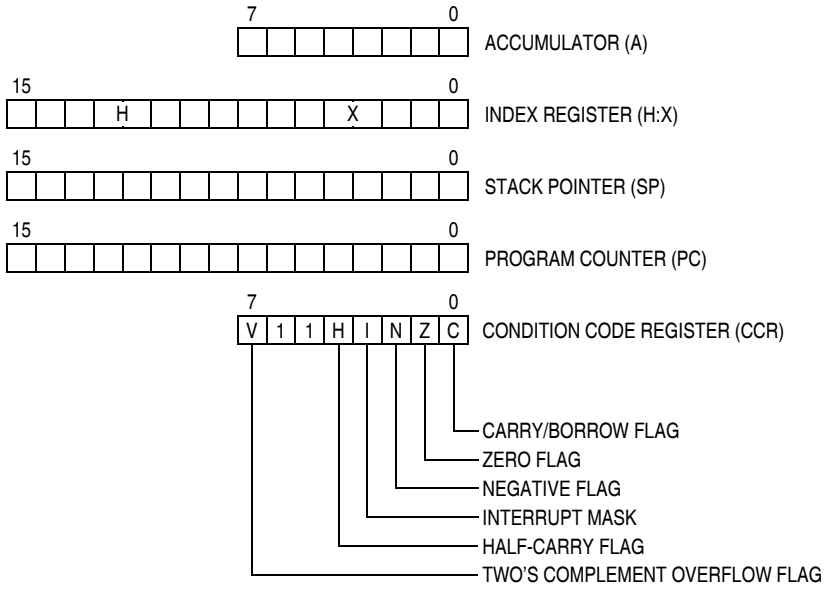


Figure 6-1. CPU Registers

6.4.1 Accumulator

The accumulator is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and the results of arithmetic/logic operations.

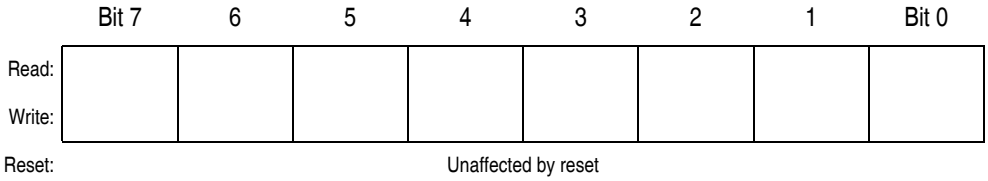


Figure 6-2. Accumulator (A)

6.4.2 Index Register

The 16-bit index register allows indexed addressing of a 64K-byte memory space. H is the upper byte of the index register, and X is the lower byte. H:X is the concatenated 16-bit index register.

In the indexed addressing modes, the CPU uses the contents of the index register to determine the conditional address of the operand.

The index register can serve also as a temporary data storage location.

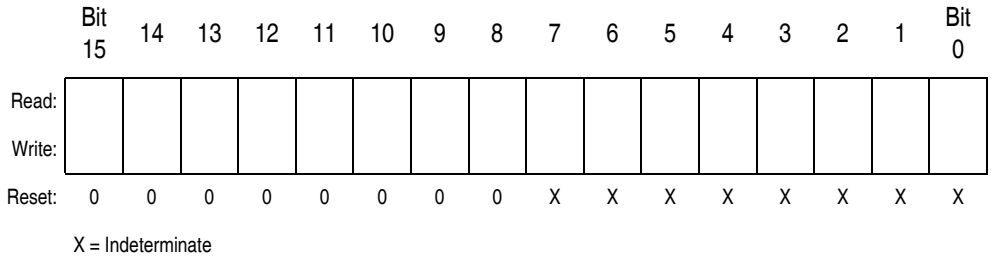


Figure 6-3. Index Register (H:X)

6.4.3 Stack Pointer

The stack pointer is a 16-bit register that contains the address of the next location on the stack. During a reset, the stack pointer is preset to $00FF. The reset stack pointer (RSP) instruction sets the least significant byte to $FF and does not affect the most significant byte. The stack pointer decrements as data is pushed onto the stack and increments as data is pulled from the stack.

In the stack pointer 8-bit offset and 16-bit offset addressing modes, the stack pointer can function as an index register to access data on the stack. The CPU uses the contents of the stack pointer to determine the conditional address of the operand.

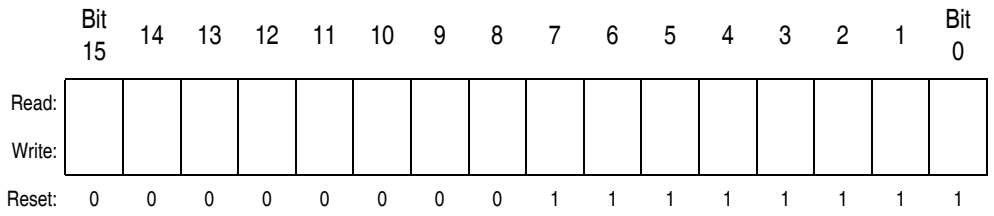


Figure 6-4. Stack Pointer (SP)

NOTE: *The location of the stack is arbitrary and may be relocated anywhere in RAM. Moving the SP out of page 0 ($0000 to $00FF) frees direct address (page 0) space. For correct operation, the stack pointer must point only to RAM locations.*

6.4.4 Program Counter

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

Normally, the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, and interrupt operations load the program counter with an address other than that of the next sequential location.

During reset, the program counter is loaded with the reset vector address located at $FFFE and $FFFF. The vector address is the address of the first instruction to be executed after exiting the reset state.

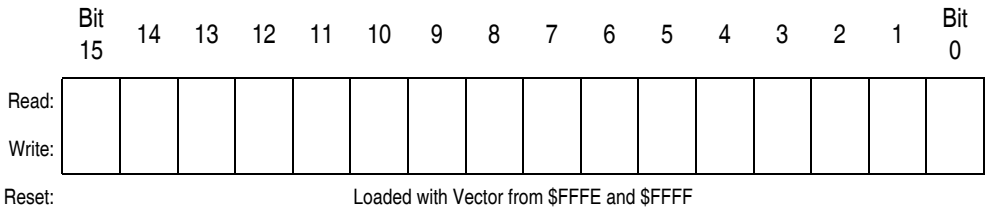


Figure 6-5. Program Counter (PC)

6.4.5 Condition Code Register

The 8-bit condition code register contains the interrupt mask and five flags that indicate the results of the instruction just executed. Bits 6 and 5 are set permanently to logic 1. The following paragraphs describe the functions of the condition code register.

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	V	1	1	H	I	N	Z	C
Reset:	X	1	1	X	1	X	X	X

X = Indeterminate

Figure 6-6. Condition Code Register (CCR)

V — 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 flags to determine the appropriate correction factor.

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

I — Interrupt Mask

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 interrupt vector is fetched.

1 = Interrupts disabled
0 = Interrupts enabled

NOTE: *To maintain M6805 Family compatibility, the upper byte of the index register (H) is not stacked automatically. If the interrupt service routine modifies H, then the user must stack and unstack H using the PSHH and PULH instructions.*

After the I bit is cleared, the highest-priority interrupt request is serviced first.

A return-from-interrupt (RTI) instruction pulls the CPU registers from the stack and restores the interrupt mask from the stack. After any reset, the interrupt mask is set and can be cleared only by the clear interrupt mask software instruction (CLI).

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.

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.

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

6.5 Arithmetic/Logic Unit (ALU)

The ALU performs the arithmetic and logic operations defined by the instruction set.

Refer to the *CPU08 Reference Manual* (Freescale document order number CPU08RM/AD) for a description of the instructions and addressing modes and more detail about the architecture of the CPU.

6.6 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

6.6.1 Wait Mode

The WAIT instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling interrupts. After exit from wait mode by interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock.

6.6.2 Stop Mode

The STOP instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling external interrupts. After exit from stop mode by external interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock.

After exiting stop mode, the CPU clock begins running after the oscillator stabilization delay.

6.7 CPU During Break Interrupts

If the break module is enabled, a break interrupt causes the CPU to execute the software interrupt instruction (SWI) at the completion of the current CPU instruction. (See **Section 22. Break Module (BRK)**.) The program counter vectors to $FFFC–$FFFD ($FEFC–$FEFD in monitor mode).

A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the MCU to normal operation if the break interrupt has been deasserted.

6.8 Instruction Set Summary

Table 6-1 provides a summary of the M68HC08 instruction set.

6.9 Opcode Map

The opcode map is provided in **Table 6-2**.

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

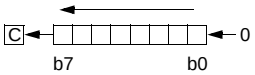
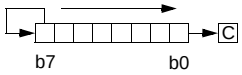
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
ADC #*opr* ADC *opr* ADC *opr* ADC *opr*,X ADC *opr*,X ADC ,X ADC *opr*,SP ADC *opr*,SP	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A9 B9 C9 D9 E9 F9 9EE9 9ED9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

Table 6-1. Instruction Set Summary (Sheet 2 of 8)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + 2 + *rel* ? (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE *opr*	Branch if Greater Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGT *opr*	Branch if Greater Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) | (N ⊕ V) = 0	–	–	–	–	–	–	REL	92	rr	3
BHCC *rel*	Branch if Half Carry Bit Clear	PC ← (PC) + 2 + *rel* ? (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS *rel*	Branch if Half Carry Bit Set	PC ← (PC) + 2 + *rel* ? (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI *rel*	Branch if Higher	PC ← (PC) + 2 + *rel* ? (C) | (Z) = 0	–	–	–	–	–	–	REL	22	rr	3
BHS *rel*	Branch if Higher or Same (Same as BCC)	PC ← (PC) + 2 + *rel* ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH *rel*	Branch if $\overline{\text{IRQ}}$ Pin High	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL *rel*	Branch if $\overline{\text{IRQ}}$ Pin Low	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 0	–	–	–	–	–	–	REL	2E	rr	3
BIT #*opr* BIT *opr* BIT *opr* BIT *opr*,X BIT *opr*,X BIT ,X BIT *opr*,SP BIT *opr*,SP	Bit Test	(A) & (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A5 B5 C5 D5 E5 F5 9EE5 9ED5	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
BLE *opr*	Branch if Less Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) | (N ⊕ V) = 1	–	–	–	–	–	–	REL	93	rr	3
BLO *rel*	Branch if Lower (Same as BCS)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	PC ← (PC) + 2 + *rel* ? (C) | (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT *opr*	Branch if Less Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) =1	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel*	–	–	–	–	–	–	REL	20	rr	3

Table 6-1. Instruction Set Summary (Sheet 3 of 8)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BRCLR *n*,*opr*,*rel*	Branch if Bit *n* in M Clear	PC ← (PC) + 3 + *rel* ? (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	PC ← (PC) + 2	–	–	–	–	–	–	REL	21	rr	3
BRSET *n*,*opr*,*rel*	Branch if Bit *n* in M Set	PC ← (PC) + 3 + *rel* ? (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*,*opr*	Set Bit *n* in M	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	4 4 4 4 4 4 4 4
BSR *rel*	Branch to Subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) – 1; push (PCH) SP ← (SP) – 1 PC ← (PC) + *rel*	–	–	–	–	–	–	REL	AD	rr	4
CBEQ *opr*,*rel* CBEQA #*opr*,*rel* CBEQX #*opr*,*rel* CBEQ *opr*,X+,*rel* CBEQ X+,*rel* CBEQ *opr*,SP,*rel*	Compare and Branch if Equal	PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (X) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 2 + rel ? (A) – (M) = $00 PC ← (PC) + 4 + rel ? (A) – (M) = $00	–	–	–	–	–	–	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 4 6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask	I ← 0	–	–	0	–	–	–	INH	9A		2
CLR *opr* CLRA CLRX CLRH CLR *opr*,X CLR ,X CLR *opr*,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	3 1 1 1 3 2 4

Table 6-1. Instruction Set Summary (Sheet 4 of 8)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
CMP #*opr* CMP *opr* CMP *opr* CMP *opr*,X CMP *opr*,X CMP ,X CMP *opr*,SP CMP *opr*,SP	Compare A with M	(A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A1 B1 C1 D1 E1 F1 9EE1 9ED1	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
COM *opr* COMA COMX COM *opr*,X COM ,X COM *opr*,SP	Complement (One's Complement)	M ← ($\overline{M}$) = $FF – (M) A ← ($\overline{A}$) = $FF – (M) X ← ($\overline{X}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M)	0	–	–	↕	↕	1	DIR INH INH IX1 IX SP1	33 43 53 63 73 9E63	dd ff ff	4 1 1 4 3 5
CPHX #*opr* CPHX *opr*	Compare H:X with M	(H:X) – (M:M + 1)	↕	–	–	↕	↕	↕	IMM DIR	65 75	ii ii+1 dd	3 4
CPX #*opr* CPX *opr* CPX *opr* CPX ,X CPX *opr*,X CPX *opr*,X CPX *opr*,SP CPX *opr*,SP	Compare X with M	(X) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A3 B3 C3 D3 E3 F3 9EE3 9ED3	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
DAA	Decimal Adjust A	$(A)_{10}$	U	–	–	↕	↕	↕	INH	72		2
DBNZ *opr*,*rel* DBNZA *rel* DBNZX *rel* DBNZ *opr*,X,*rel* DBNZ X,*rel* DBNZ *opr*,SP,*rel*	Decrement and Branch if Not Zero	A ← (A) – 1 or M ← (M) – 1 or X ← (X) – 1 PC ← (PC) + 3 + *rel* ? (result) ≠ 0 PC ← (PC) + 2 + *rel* ? (result) ≠ 0 PC ← (PC) + 2 + *rel* ? (result) ≠ 0 PC ← (PC) + 3 + *rel* ? (result) ≠ 0 PC ← (PC) + 2 + *rel* ? (result) ≠ 0 PC ← (PC) + 4 + *rel* ? (result) ≠ 0	–	–	–	–	–	–	DIR INH INH IX1 IX SP1	3B 4B 5B 6B 7B 9E6B	dd rr rr rr ff rr rr ff rr	5 3 3 5 4 6
DEC *opr* DECA DECX DEC *opr*,X DEC ,X DEC *opr*,SP	Decrement	M ← (M) – 1 A ← (A) – 1 X ← (X) – 1 M ← (M) – 1 M ← (M) – 1 M ← (M) – 1	↕	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3A 4A 5A 6A 7A 9E6A	dd ff ff	4 1 1 4 3 5
DIV	Divide	A ← (H:A)/(X) H ← Remainder	–	–	–	–	↕	↕	INH	52		7
EOR #*opr* EOR *opr* EOR *opr* EOR *opr*,X EOR *opr*,X EOR ,X EOR *opr*,SP EOR *opr*,SP	Exclusive OR M with A	A ← (A ⊕ M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A8 B8 C8 D8 E8 F8 9EE8 9ED8	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5

Table 6-1. Instruction Set Summary (Sheet 5 of 8)

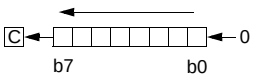
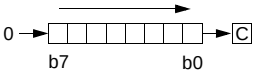
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
INC *opr* INCA INCX INC *opr*,X INC ,X INC *opr*,SP	Increment	M ← (M) + 1 A ← (A) + 1 X ← (X) + 1 M ← (M) + 1 M ← (M) + 1 M ← (M) + 1	↕	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3C 4C 5C 6C 7C 9E6C	dd ff ff	4 1 1 4 3 5
JMP *opr* JMP *opr* JMP *opr*,X JMP *opr*,X JMP ,X	Jump	PC ← Jump Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff 	2 3 4 3 2
JSR *opr* JSR *opr* JSR *opr*,X JSR *opr*,X JSR ,X	Jump to Subroutine	PC ← (PC) + *n* (*n* = 1, 2, or 3) Push (PCL); SP ← (SP) – 1 Push (PCH); SP ← (SP) – 1 PC ← Unconditional Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff 	4 5 6 5 4
LDA #*opr* LDA *opr* LDA *opr* LDA *opr*,X LDA *opr*,X LDA ,X LDA *opr*,SP LDA *opr*,SP	Load A from M	A ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A6 B6 C6 D6 E6 F6 9EE6 9ED6	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
LDHX #*opr* LDHX *opr*	Load H:X from M	H:X ← (M:M + 1)	0	–	–	↕	↕	–	IMM DIR	45 55	ii jj dd	3 4
LDX #*opr* LDX *opr* LDX *opr* LDX *opr*,X LDX *opr*,X LDX ,X LDX *opr*,SP LDX *opr*,SP	Load X from M	X ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	AE BE CE DE EE FE 9EEE 9EDE	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
LSL *opr* LSLA LSLX LSL *opr*,X (Same as ASL) LSL ,X LSL *opr*,SP	Logical Shift Left		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	4 1 1 4 3 5
LSR *opr* LSRA LSR*X* LSR *opr*,X LSR ,X LSR *opr*,SP	Logical Shift Right		↕	–	–	0	↕	↕	DIR INH INH IX1 IX SP1	34 44 54 64 74 9E64	dd ff ff	4 1 1 4 3 5

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

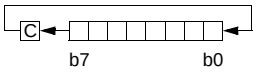
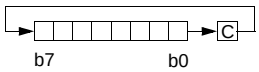
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
MOV *opr,opr* MOV *opr,*X+ MOV *#opr,opr* MOV X+*,opr*	Move	$(M)_{Destination} \leftarrow (M)_{Source}$ H:X ← (H:X) + 1 (IX+D, DIX+)	0	–	–	↕	↕	–	DD DIX+ IMD IX+D	4E 5E 6E 7E	dd dd dd ii dd dd	5 4 4 4
MUL	Unsigned multiply	X:A ← (X) × (A)	–	0	–	–	–	0	INH	42		5
NEG *opr* NEGA NEGX NEG *opr*,X NEG ,X NEG *opr*,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)	↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	30 40 50 60 70 9E60	dd ff ff	4 1 1 4 3 5
NOP	No Operation	None	–	–	–	–	–	–	INH	9D		1
NSA	Nibble Swap A	A ← (A[3:0]:A[7:4])	–	–	–	–	–	–	INH	62		3
ORA *#opr* ORA *opr* ORA *opr* ORA *opr*,X ORA *opr*,X ORA ,X ORA *opr*,SP ORA *opr*,SP	Inclusive OR A and M	A ← (A) | (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	AA BA CA DA EA FA 9EEA 9EDA	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
PSHA	Push A onto Stack	Push (A); SP ← (SP) – 1	–	–	–	–	–	–	INH	87		2
PSHH	Push H onto Stack	Push (H); SP ← (SP) – 1	–	–	–	–	–	–	INH	8B		2
PSHX	Push X onto Stack	Push (X); SP ← (SP) – 1	–	–	–	–	–	–	INH	89		2
PULA	Pull A from Stack	SP ← (SP + 1); Pull (A)	–	–	–	–	–	–	INH	86		2
PULH	Pull H from Stack	SP ← (SP + 1); Pull (H)	–	–	–	–	–	–	INH	8A		2
PULX	Pull X from Stack	SP ← (SP + 1); Pull (X)	–	–	–	–	–	–	INH	88		2
ROL *opr* ROLA ROLX ROL *opr*,X ROL ,X ROL *opr*,SP	Rotate Left through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	39 49 59 69 79 9E69	dd ff ff	4 1 1 4 3 5
ROR *opr* RORA RORX ROR *opr*,X ROR ,X ROR *opr*,SP	Rotate Right through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	4 1 1 4 3 5
RSP	Reset Stack Pointer	SP ← $FF	–	–	–	–	–	–	INH	9C		1

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

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
RTI	Return from Interrupt	SP ← (SP) + 1; Pull (CCR) SP ← (SP) + 1; Pull (A) SP ← (SP) + 1; Pull (X) SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	↕	↕	↕	↕	↕	↕	INH	80		7
RTS	Return from Subroutine	SP ← SP + 1; Pull (PCH) SP ← SP + 1; Pull (PCL)	–	–	–	–	–	–	INH	81		4
SBC #*opr* SBC *opr* SBC *opr* SBC *opr*,X SBC *opr*,X SBC ,X SBC *opr*,SP SBC *opr*,SP	Subtract with Carry	A ← (A) – (M) – (C)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A2 B2 C2 D2 E2 F2 9EE2 9ED2	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask	I ← 1	–	–	1	–	–	–	INH	9B		2
STA *opr* STA *opr* STA *opr*,X STA *opr*,X STA ,X STA *opr*,SP STA *opr*,SP	Store A in M	M ← (A)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	B7 C7 D7 E7 F7 9EE7 9ED7	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
STHX *opr*	Store H:X in M	(M:M + 1) ← (H:X)	0	–	–	↕	↕	–	DIR	35	dd	4
STOP	Enable $\overline{\text{IRQ}}$ Pin; Stop Oscillator	I ← 0; Stop Oscillator	–	–	0	–	–	–	INH	8E		1
STX *opr* STX *opr* STX *opr*,X STX *opr*,X STX ,X STX *opr*,SP STX *opr*,SP	Store X in M	M ← (X)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	BF CF DF EF FF 9EEF 9EDF	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
SUB #*opr* SUB *opr* SUB *opr* SUB *opr*,X SUB *opr*,X SUB ,X SUB *opr*,SP SUB *opr*,SP	Subtract	A ← (A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A0 B0 C0 D0 E0 F0 9EE0 9ED0	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5

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

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
SWI	Software Interrupt	PC ← (PC) + 1; Push (PCL) SP ← (SP) – 1; Push (PCH) SP ← (SP) – 1; Push (X) SP ← (SP) – 1; Push (A) SP ← (SP) – 1; Push (CCR) SP ← (SP) – 1; I ← 1 PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		9
TAP	Transfer A to CCR	CCR ← (A)	↕	↕	↕	↕	↕	↕	INH	84		2
TAX	Transfer A to X	X ← (A)	–	–	–	–	–	–	INH	97		1
TPA	Transfer CCR to A	A ← (CCR)	–	–	–	–	–	–	INH	85		1
TST *opr* TSTA TSTX TST *opr*,X TST ,X TST *opr*,SP	Test for Negative or Zero	(A) – $00 or (X) – $00 or (M) – $00	0	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3D 4D 5D 6D 7D 9E6D	dd ff ff	3 1 1 3 2 4
TSX	Transfer SP to H:X	H:X ← (SP) + 1	–	–	–	–	–	–	INH	95		2
TXA	Transfer X to A	A ← (X)	–	–	–	–	–	–	INH	9F		1
TXS	Transfer H:X to SP	(SP) ← (H:X) – 1	–	–	–	–	–	–	INH	94		2

A	Accumulator
C	Carry/borrow bit
CCR	Condition code register
dd	Direct address of operand
dd rr	Direct address of operand and relative offset of branch instruction
DD	Direct to direct addressing mode
DIR	Direct addressing mode
DIX+	Direct to indexed with post increment addressing mode
ee ff	High and low bytes of offset in indexed, 16-bit offset addressing
EXT	Extended addressing mode
ff	Offset byte in indexed, 8-bit offset addressing
H	Half-carry bit
H	Index register high byte
hh ll	High and low bytes of operand address in extended addressing
I	Interrupt mask
ii	Immediate operand byte
IMD	Immediate source to direct destination addressing mode
IMM	Immediate addressing mode
INH	Inherent addressing mode
IX	Indexed, no offset addressing mode
IX+	Indexed, no offset, post increment addressing mode
IX+D	Indexed with post increment to direct addressing mode
IX1	Indexed, 8-bit offset addressing mode
IX1+	Indexed, 8-bit offset, post increment addressing mode
IX2	Indexed, 16-bit offset addressing mode
M	Memory location
N	Negative bit
n	Any bit
opr	Operand (one or two bytes)
PC	Program counter
PCH	Program counter high byte
PCL	Program counter low byte
REL	Relative addressing mode
rel	Relative program counter offset byte
rr	Relative program counter offset byte
SP1	Stack pointer, 8-bit offset addressing mode
SP2	Stack pointer 16-bit offset addressing mode
SP	Stack pointer
U	Undefined
V	Overflow bit
X	Index register low byte
Z	Zero bit
&	Logical AND
|	Logical OR
⊕	Logical EXCLUSIVE OR
( )	Contents of
–( )	Negation (two's complement)
#	Immediate value
«	Sign extend
←	Loaded with
?	If
:	Concatenated with
↕	Set or cleared
—	Not affected

Table 6-2. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write						Control		Register/Mem				
	DIR	DIR	REL	DIR	INH	INH	IX1	SP1	IX	INH	INH	IMM	DIR	EXT	IX2	SP
MSB / LSB	0	1	2	3	4	5	6	9E6	7	8	9	A	B	C	D	9E
0	5 BRSET0 3 DIR	4 BSET0 2 DIR	3 BRA 2 REL	4 NEG 2 DIR	1 NEGA 1 INH	1 NEGX 1 INH	4 NEG 2 IX1	5 NEG 3 SP1	3 NEG 1 IX	7 RTI 1 INH	3 BGE 2 REL	2 SUB 2 IMM	3 SUB 2 DIR	4 SUB 3 EXT	4 SUB 3 IX2	SU 4
1	5 BRCLR0 3 DIR	4 BCLR0 2 DIR	3 BRN 2 REL	5 CBEQ 3 DIR	4 CBEQA 3 IMM	4 CBEQX 3 IMM	5 CBEQ 3 IX1+	6 CBEQ 4 SP1	4 CBEQ 2 IX+	4 RTS 1 INH	3 BLT 2 REL	2 CMP 2 IMM	3 CMP 2 DIR	4 CMP 3 EXT	4 CMP 3 IX2	CM 4
2	5 BRSET1 3 DIR	4 BSET1 2 DIR	3 BHI 2 REL		5 MUL 1 INH	7 DIV 1 INH	3 NSA 1 INH		2 DAA 1 INH		3 BGT 2 REL	2 SBC 2 IMM	3 SBC 2 DIR	4 SBC 3 EXT	4 SBC 3 IX2	SB 4
3	5 BRCLR1 3 DIR	4 BCLR1 2 DIR	3 BLS 2 REL	4 COM 2 DIR	1 COMA 1 INH	1 COMX 1 INH	4 COM 2 IX1	5 COM 3 SP1	3 COM 1 IX	9 SWI 1 INH	3 BLE 2 REL	2 CPX 2 IMM	3 CPX 2 DIR	4 CPX 3 EXT	4 CPX 3 IX2	CP 4
4	5 BRSET2 3 DIR	4 BSET2 2 DIR	3 BCC 2 REL	4 LSR 2 DIR	1 LSRA 1 INH	1 LSRX 1 INH	4 LSR 2 IX1	5 LSR 3 SP1	3 LSR 1 IX	2 TAP 1 INH	2 TXS 1 INH	2 AND 2 IMM	3 AND 2 DIR	4 AND 3 EXT	4 AND 3 IX2	AN 4
5	5 BRCLR2 3 DIR	4 BCLR2 2 DIR	3 BCS 2 REL	4 STHX 2 DIR	3 LDHX 3 IMM	4 LDHX 2 DIR	3 CPHX 3 IMM		4 CPHX 2 DIR	1 TPA 1 INH	2 TSX 1 INH	2 BIT 2 IMM	3 BIT 2 DIR	4 BIT 3 EXT	4 BIT 3 IX2	BI 4
6	5 BRSET3 3 DIR	4 BSET3 2 DIR	3 BNE 2 REL	4 ROR 2 DIR	1 RORA 1 INH	1 RORX 1 INH	4 ROR 2 IX1	5 ROR 3 SP1	3 ROR 1 IX	2 PULA 1 INH		2 LDA 2 IMM	3 LDA 2 DIR	4 LDA 3 EXT	4 LDA 3 IX2	LD 4
7	5 BRCLR3 3 DIR	4 BCLR3 2 DIR	3 BEQ 2 REL	4 ASR 2 DIR	1 ASRA 1 INH	1 ASRX 1 INH	4 ASR 2 IX1	5 ASR 3 SP1	3 ASR 1 IX	2 PSHA 1 INH	1 TAX 1 INH	2 AIS 2 IMM	3 STA 2 DIR	4 STA 3 EXT	4 STA 3 IX2	ST 4
8	5 BRSET4 3 DIR	4 BSET4 2 DIR	3 BHCC 2 REL	4 LSL 2 DIR	1 LSLA 1 INH	1 LSLX 1 INH	4 LSL 2 IX1	5 LSL 3 SP1	3 LSL 1 IX	2 PULX 1 INH	1 CLC 1 INH	2 EOR 2 IMM	3 EOR 2 DIR	4 EOR 3 EXT	4 EOR 3 IX2	EO 4
9	5 BRCLR4 3 DIR	4 BCLR4 2 DIR	3 BHCS 2 REL	4 ROL 2 DIR	1 ROLA 1 INH	1 ROLX 1 INH	4 ROL 2 IX1	5 ROL 3 SP1	3 ROL 1 IX	2 PSHX 1 INH	1 SEC 1 INH	2 ADC 2 IMM	3 ADC 2 DIR	4 ADC 3 EXT	4 ADC 3 IX2	AD 4
A	5 BRSET5 3 DIR	4 BSET5 2 DIR	3 BPL 2 REL	4 DEC 2 DIR	1 DECA 1 INH	1 DECX 1 INH	4 DEC 2 IX1	5 DEC 3 SP1	3 DEC 1 IX	2 PULH 1 INH	2 CLI 1 INH	2 ORA 2 IMM	3 ORA 2 DIR	4 ORA 3 EXT	4 ORA 3 IX2	OR 4
B	5 BRCLR5 3 DIR	4 BCLR5 2 DIR	3 BMI 2 REL	5 DBNZ 3 DIR	3 DBNZA 2 INH	3 DBNZX 2 INH	5 DBNZ 3 IX1	6 DBNZ 4 SP1	4 DBNZ 2 IX	2 PSHH 1 INH	2 SEI 1 INH	2 ADD 2 IMM	3 ADD 2 DIR	4 ADD 3 EXT	4 ADD 3 IX2	AD 4
C	5 BRSET6 3 DIR	4 BSET6 2 DIR	3 BMC 2 REL	4 INC 2 DIR	1 INCA 1 INH	1 INCX 1 INH	4 INC 2 IX1	5 INC 3 SP1	3 INC 1 IX	1 CLRH 1 INH	1 RSP 1 INH		2 JMP 2 DIR	3 JMP 3 EXT	4 JMP 3 IX2	
D	5 BRCLR6 3 DIR	4 BCLR6 2 DIR	3 BMS 2 REL	3 TST 2 DIR	1 TSTA 1 INH	1 TSTX 1 INH	3 TST 2 IX1	4 TST 3 SP1	2 TST 1 IX		1 NOP 1 INH	4 BSR 2 REL	4 JSR 2 DIR	5 JSR 3 EXT	6 JSR 3 IX2	
E	5 BRSET7 3 DIR	4 BSET7 2 DIR	3 BIL 2 REL		5 MOV 3 DD	4 MOV 2 DIX+	4 MOV 3 IMD		4 MOV 2 IX+D	1 STOP 1 INH	*	2 LDX 2 IMM	3 LDX 2 DIR	4 LDX 3 EXT	4 LDX 3 IX2	LD 4
F	5 BRCLR7 3 DIR	4 BCLR7 2 DIR	3 BIH 2 REL	3 CLR 2 DIR	1 CLRA 1 INH	1 CLRX 1 INH	3 CLR 2 IX1	4 CLR 3 SP1	2 CLR 1 IX	1 WAIT 1 INH	1 TXA 1 INH	2 AIX 2 IMM	3 STX 2 DIR	4 STX 3 EXT	4 STX 3 IX2	ST 4

INH Inherent
IMM Immediate
DIR Direct
EXT Extended
DD Direct-Direct
IX+D Indexed-Direct
REL Relative
IX Indexed, No Offset
IX1 Indexed, 8-Bit Offset
IX2 Indexed, 16-Bit Offset
IMD Immediate-Direct
DIX+ Direct-Indexed
SP1 Stack Pointer, 8-Bit Offset
SP2 Stack Pointer, 16-Bit Offset
IX+ Indexed, No Offset with Post Increment
IX1+ Indexed, 1-Byte Offset with Post Increment
*Pre-byte for stack pointer indexed instructions

Low Byte of Opcode in Hexadecimal

MSB / LSB	0	
0	5 BRSET0 3 DIR	High Byte of O Cycles Opcode Mnem Number of Byt

Section 7. Oscillator (OSC)

7.1 Contents

7.2 Introduction 95
7.3 Internal Oscillator 97
7.4 Crystal (X-tal) Oscillator 97
7.5 I/O Signals 97
7.5.1 Crystal Amplifier Input Pin (OSC1) 98
7.5.2 Crystal Amplifier Output Pin (OSC2) 98
7.5.3 Oscillator Enable Signal (SIMOSCEN) 98
7.5.4 Internal RC Clock (ICLK) 98
7.5.5 CGM Oscillator Clock (CGMXCLK) 98
7.5.6 CGM Reference Clock (CGMRCLK) 98
7.6 Low Power Modes 98
7.6.1 Wait Mode 99
7.6.2 Stop Mode 99
7.7 Oscillator During Break Mode 99

7.2 Introduction

The oscillator module provides the reference clock for the clock generator module (CGM), the real time clock module (RTC), and other MCU sub-systems.

The oscillator module consist of two types of oscillator circuits:

- Internal RC oscillator
- Crystal (x-tal) oscillator

The reference clock for the CGM, real time clock module (RTC), and other MCU sub-systems is driven by the crystal oscillator. The COP module is always driven by internal RC oscillator.

The RC internal oscillator runs continuously after a POR or reset and is always available in run and wait modes. In stop mode, it can be disabled by setting the STOP_IRCDIS bit in CONFIG2 register.
Figure 7-1. shows the block diagram of the oscillator module.

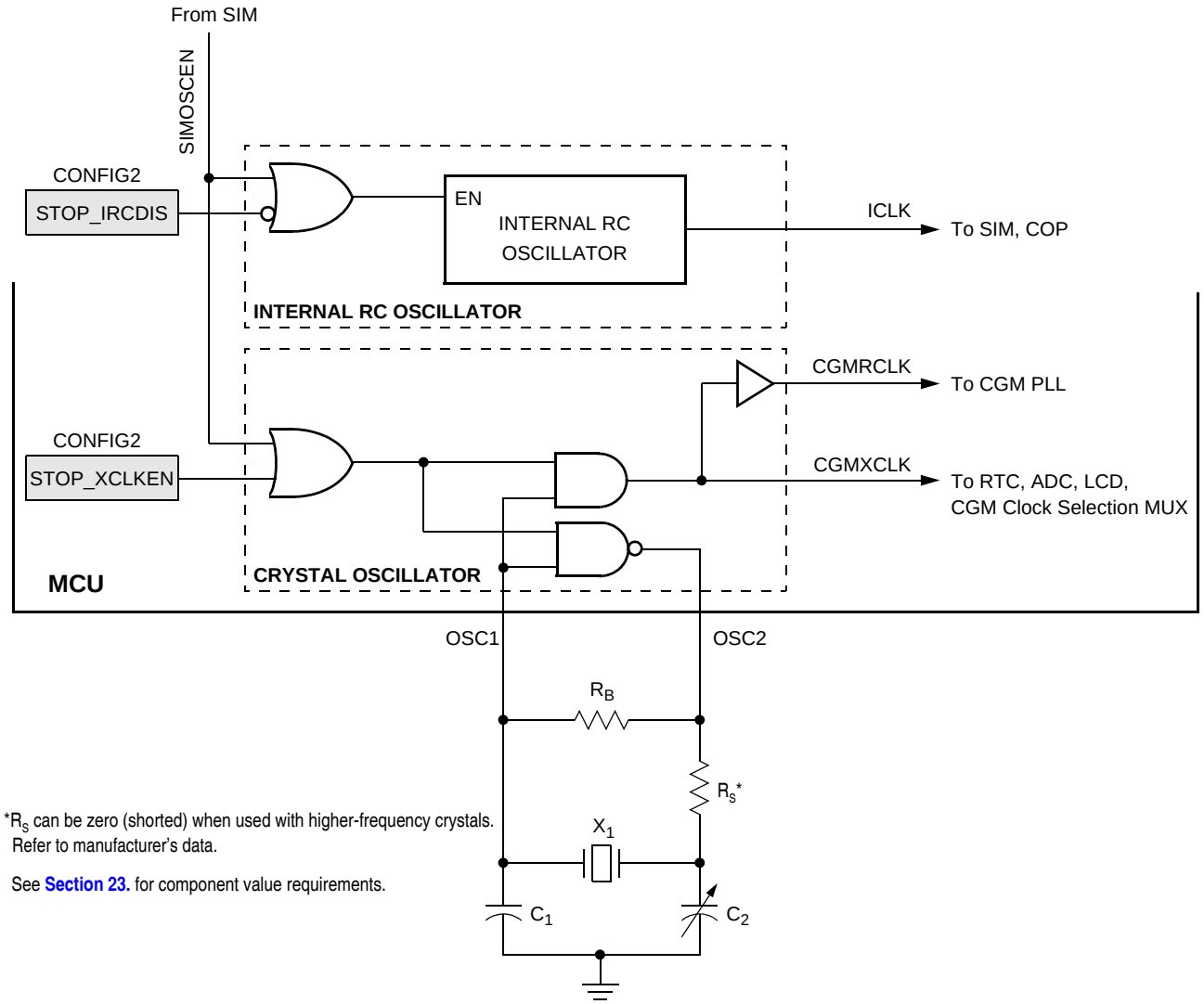


Figure 7-1. Oscillator Module Block Diagram

7.3 Internal Oscillator

The internal RC oscillator clock (ICLK) is a free running 64 kHz clock (at V_{DD} = 5V) that requires no external components. It is the reference clock input to the computer operating properly (COP) module.

The ICLK can be turned off in stop mode by setting the STOP_IRCDIS bit in CONFIG2. After reset, the bit is clear by default and ICLK is enabled during stop mode.

7.4 Crystal (X-tal) Oscillator

The crystal (x-tal) oscillator circuit is designed for use with an external crystal or ceramic resonator to provide an accurate clock source.

In its typical configuration, the X-tal oscillator is connected in a Pierce oscillator configuration, as shown in **Figure 7-1**. This figure shows only the logical representation of the internal components and may not represent actual circuitry. The oscillator configuration uses five components:

- Crystal, X_1
- Fixed capacitor, C_1
- Tuning capacitor, C_2 (can also be a fixed capacitor)
- Feedback resistor, R_B
- Series resistor, R_S (optional)

The series resistor (R_S) is included in the diagram to follow strict Pierce oscillator guidelines and may not be required for all ranges of operation, especially with high frequency crystals. Refer to the crystal manufacturer's data for more information.

7.5 I/O Signals

The following paragraphs describe the oscillator I/O signals.

7.5.1 Crystal Amplifier Input Pin (OSC1)

OSC1 pin is an input to the crystal oscillator amplifier. Schmitt trigger and glitch filter are implemented on this pin to improve EMC performance. See **Section 23. Electrical Specifications** for detail specification of the glitch filter.

7.5.2 Crystal Amplifier Output Pin (OSC2)

OSC2 pin is the output of the crystal oscillator inverting amplifier.

7.5.3 Oscillator Enable Signal (SIMOSCEN)

The SIMOSCEN signal from the system integration module (SIM) enables/disables the internal RC and x-tal oscillator circuits.

7.5.4 Internal RC Clock (ICLK)

The ICLK clock is the output from the internal RC oscillator. This clock drives the SIM and COP modules.

7.5.5 CGM Oscillator Clock (CGMXCLK)

The CGMXCLK clock is the output from the x-tal oscillator. This clock drives to CGM, real time clock module, analog-to-digital converter, liquid crystal display driver module, and other MCU sub-systems.

7.5.6 CGM Reference Clock (CGMRCLK)

This is buffered signal of CGMXCLK, it is used by the CGM as the phase-locked-loop (PLL) reference clock.

7.6 Low Power Modes

The WAIT and STOP instructions put the MCU in low-power consumption standby modes.

7.6.1 Wait Mode

The WAIT instruction has no effect on the oscillator module. CGMXCLK, CGMRCLK, and ICLK continues to drive the MCU modules.

7.6.2 Stop Mode

The STOP instruction clears the SIMOSCEN signal, and hence the CGMXCLK (and CGMRCLK) clock stops running. For continuous CGMXCLK operation in stop mode, set the STOP_XCLKEN to logic 1 before entering stop mode. Continuous CGMXCLK operation in stop mode allows the RTC module to generate interrupts to wake up the CPU.

By default, the internal RC oscillator clock, ICLK, continues to run in stop mode. To disable the ICLK in stop mode, set the STOP_IRCDIS bit to logic 1 before entering stop mode.

7.7 Oscillator During Break Mode

The oscillator circuits continue to drive CGMXCLK, CGMRCLK, and ICLK when the device enters the break state.

Section 8. Clock Generator Module (CGM)

8.1 Contents

8.2 Introduction 102

8.3 Features 103

8.4 Functional Description 103
8.4.1 Oscillator Module 106
8.4.2 Phase-Locked Loop Circuit (PLL) 106
8.4.3 PLL Circuits 106
8.4.4 Acquisition and Tracking Modes 108
8.4.5 Manual and Automatic PLL Bandwidth Modes 108
8.4.6 Programming the PLL 110
8.4.7 Special Programming Exceptions 114
8.4.8 Base Clock Selector Circuit 114
8.4.9 CGM External Connections 115

8.5 I/O Signals 115
8.5.1 External Filter Capacitor Pin (CGMXFC) 116
8.5.2 PLL Analog Power Pin (V_{DDA}) 116
8.5.3 PLL Analog Ground Pin (V_{SSA}) 116
8.5.4 Oscillator Output Frequency Signal (CGMXCLK) 116
8.5.5 CGM Reference Clock (CGMRCLK) 116
8.5.6 CGM VCO Clock Output (CGMVCLK) 117
8.5.7 CGM Base Clock Output (CGMOUT) 117
8.5.8 CGM CPU Interrupt (CGMINT) 117

8.6 CGM Registers 117
8.6.1 PLL Control Register 118
8.6.2 PLL Bandwidth Control Register 120
8.6.3 PLL Multiplier Select Registers 122
8.6.4 PLL VCO Range Select Register 123
8.6.5 PLL Reference Divider Select Register 124

8.7 Interrupts 125

8.8 Special Modes 125
8.8.1 Wait Mode 125
8.8.2 Stop Mode 126
8.8.3 CGM During Break Interrupts 126

8.9 Acquisition/Lock Time Specifications 127
8.9.1 Acquisition/Lock Time Definitions 127
8.9.2 Parametric Influences on Reaction Time 127
8.9.3 Choosing a Filter 129

8.2 Introduction

This section describes the clock generator module (CGM). The CGM generates the base clock signal, CGMOUT, which is based on either the oscillator clock divided by two or the divided phase-locked loop (PLL) clock, CGMPCLK, divided by two. CGMOUT is the clock from which the SIM derives the system clocks, including the bus clock, which is at a frequency of CGMOUT÷2.

The PLL is a frequency generator designed for use with a low frequency crystal (typically 32.768kHz) to generate a base frequency and dividing to a maximum bus frequency of 8MHz.

8.3 Features

Features of the CGM include:

- Phase-locked loop with output frequency in integer multiples of an integer dividend of the crystal reference
- Low-frequency crystal operation with low-power operation and high-output frequency resolution
- Programmable prescaler for power-of-two increases in frequency
- Programmable hardware voltage-controlled oscillator (VCO) for low-jitter operation
- Automatic bandwidth control mode for low-jitter operation
- Automatic frequency lock detector
- CPU interrupt on entry or exit from locked condition
- Configuration register bit to allow oscillator operation during stop mode

8.4 Functional Description

The CGM consists of three major sub-modules:

- Oscillator module — The oscillator module generates the constant reference frequency clock, CGMRCLK (buffered CGMXCLK).
- Phase-locked loop (PLL) — The PLL generates the programmable VCO frequency clock, CGMVCLK, and the divided, CGMPCLK. The CGMPCLK is one of the reference clocks to the base clock selector circuit.
- Base clock selector circuit — This software-controlled circuit selects the one of three clocks as the base clock, CGMOUT: CGMXCLK, CGMXCLK divided by two, or CGMPCLK divided by two.

Figure 8-1 shows the structure of the CGM.

Figure 8-2 is a summary of the CGM registers.

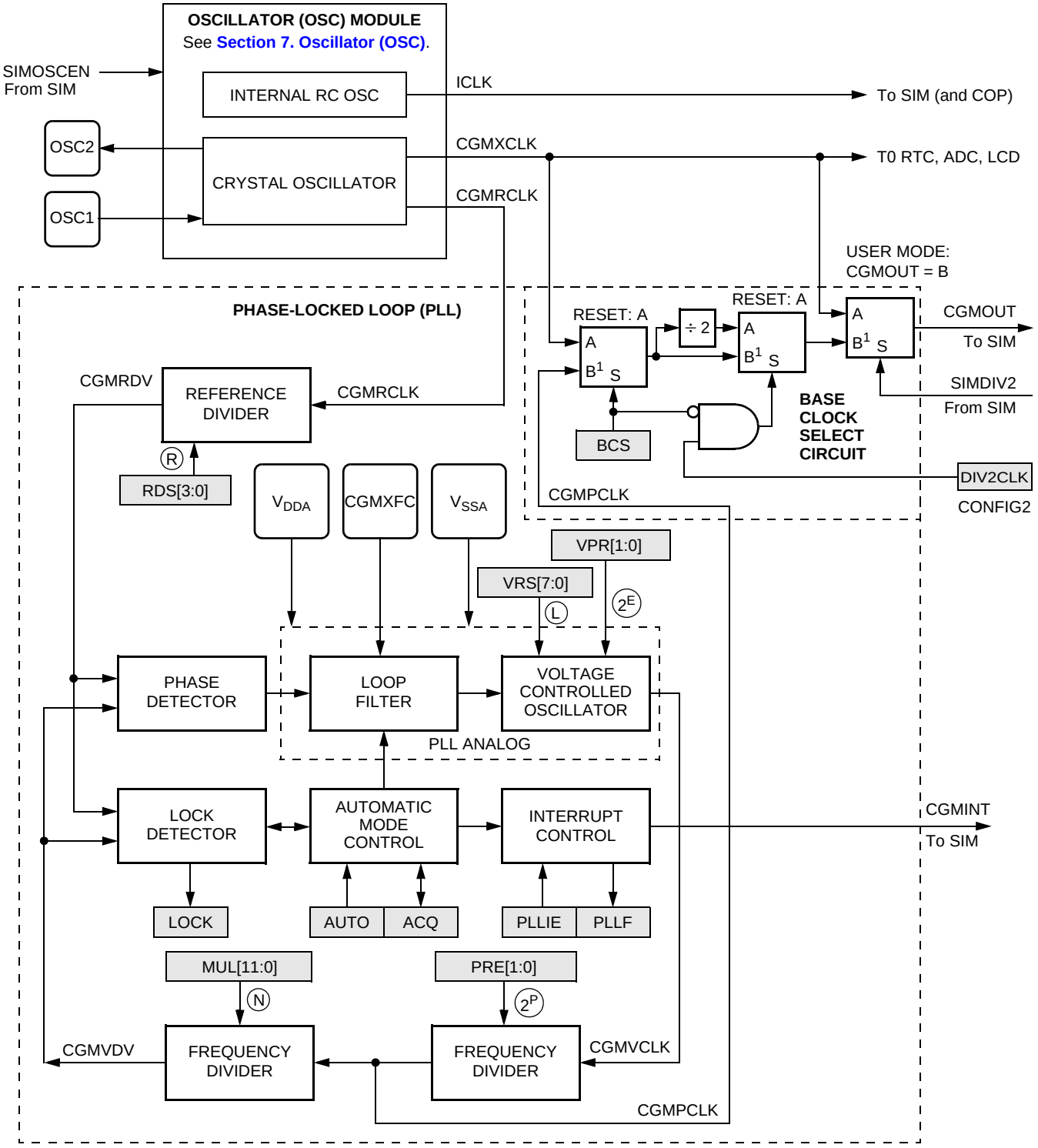


Figure 8-1. CGM Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0036	PLL Control Register (PTCL)	Read:	PLLIE	PLLF	PLLON	BCS	PRE1	PRE0	VPR1	VPR0
		Write:								
		Reset:	0	0	1	0	0	0	0	0
$0037	PLL Bandwidth Control Register (PBWC)	Read:	AUTO	LOCK	$\overline{\text{ACQ}}$	0	0	0	0	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0038	PLL Multiplier Select Register High (PMSH)	Read:	0	0	0	0	MUL11	MUL10	MUL9	MUL8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0039	PLL Multiplier Select Register Low (PMSL)	Read:	MUL7	MUL6	MUL5	MUL4	MUL3	MUL2	MUL1	MUL0
		Write:								
		Reset:	0	1	0	0	0	0	0	0
$003A	PLL VCO Range Select Register (PMRS)	Read:	VRS7	VRS6	VRS5	VRS4	VRS3	VRS2	VRS1	VRS0
		Write:								
		Reset:	0	1	0	0	0	0	0	0
$003B	PLL Reference Divider Select Register (PMDS)	Read:	0	0	0	0	RDS3	RDS2	RDS1	RDS0
		Write:								
		Reset:	0	0	0	0	0	0	0	1

= Unimplemented R = Reserved

NOTES:
1. When AUTO = 0, PLLIE is forced clear and is read-only.
2. When AUTO = 0, PLLF and LOCK read as clear.
3. When AUTO = 1, $\overline{\text{ACQ}}$ is read-only.
4. When PLLON = 0 or VRS7:VRS0 = $0, BCS is forced clear and is read-only.
5. When PLLON = 1, the PLL programming register is read-only.
6. When BCS = 1, PLLON is forced set and is read-only.

Figure 8-2. CGM I/O Register Summary

8.4.1 Oscillator Module

The oscillator module provides two clock outputs CGMXCLK and CGMRCLK to the CGM module. CGMXCLK or CGMXCLK divide-by-two can be selected to drive the SIM module to generate the system bus clocks. CGMRCLK is the reference clock for the phase-lock-loop, to generate a higher frequency clock. The oscillator module also provides the reference clock for the real time clock (RTC) module.
See **Section 7. Oscillator (OSC)** for detailed description on oscillator module. See **Section 12. Real Time Clock (RTC)** for detailed description on RTC.

8.4.2 Phase-Locked Loop Circuit (PLL)

The PLL is a frequency generator that can operate in either acquisition mode or tracking mode, depending on the accuracy of the output frequency. The PLL can change between acquisition and tracking modes either automatically or manually.

8.4.3 PLL Circuits

The PLL consists of these circuits:

- Voltage-controlled oscillator (VCO)
- Reference divider
- Frequency pre-scaler
- Modulo VCO frequency divider
- Phase detector
- Loop filter
- Lock detector

The operating range of the VCO is programmable for a wide range of frequencies and for maximum immunity to external noise, including supply and CGMXFC noise. The VCO frequency is bound to a range from roughly one-half to twice the center-of-range frequency, f_{VRS}. Modulating the voltage on the CGMXFC pin changes the frequency within this range. By design, f_{VRS} is equal to the nominal center-of-range frequency, f_{NOM}, (38.4 kHz) times a linear factor, L, and a power-of-two factor, E, or $(L \times 2^E)f_{NOM}$.

CGMRCLK is the PLL reference clock, a buffered version of CGMXCLK. CGMRCLK runs at a frequency, f_{RCLK}, and is fed to the PLL through a programmable modulo reference divider, which divides f_{RCLK} by a factor, R. The divider's output is the final reference clock, CGMRDV, running at a frequency, $f_{RDV} = f_{RCLK}/R$. With an external crystal (30kHz–100kHz), always set R = 1 for specified performance. With an external high-frequency clock source, use R to divide the external frequency to between 30kHz and 100kHz.

The VCO's output clock, CGMVCLK, running at a frequency, f_{VCLK}, is fed back through a programmable pre-scaler divider and a programmable modulo divider. The pre-scaler divides the VCO clock by a power-of-two factor P (the CGMPCLK) and the modulo divider reduces the VCO clock by a factor, N. The dividers' output is the VCO feedback clock, CGMVDV, running at a frequency, $f_{VDV} = f_{VCLK}/(N \times 2^P)$. (See **8.4.6 Programming the PLL** for more information.)

The phase detector then compares the VCO feedback clock, CGMVDV, with the final reference clock, CGMRDV. A correction pulse is generated based on the phase difference between the two signals. The loop filter then slightly alters the DC voltage on the external capacitor connected to CGMXFC based on the width and direction of the correction pulse. The filter can make fast or slow corrections depending on its mode, described in **8.4.4 Acquisition and Tracking Modes**. The value of the external capacitor and the reference frequency determines the speed of the corrections and the stability of the PLL.

The lock detector compares the frequencies of the VCO feedback clock, CGMVDV, and the final reference clock, CGMRDV. Therefore, the speed of the lock detector is directly proportional to the final reference frequency, f_{RDV}. The circuit determines the mode of the PLL and the lock condition based on this comparison.

8.4.4 Acquisition and Tracking Modes

The PLL filter is manually or automatically configurable into one of two operating modes:

- Acquisition mode — In acquisition mode, the filter can make large frequency corrections to the VCO. This mode is used at PLL start up or when the PLL has suffered a severe noise hit and the VCO frequency is far off the desired frequency. When in acquisition mode, the $\overline{ACQ}$ bit is clear in the PLL bandwidth control register. (See **8.6.2 PLL Bandwidth Control Register**.)
- Tracking mode — In tracking mode, the filter makes only small corrections to the frequency of the VCO. PLL jitter is much lower in tracking mode, but the response to noise is also slower. The PLL enters tracking mode when the VCO frequency is nearly correct, such as when the PLL is selected as the base clock source. (See **8.4.8 Base Clock Selector Circuit**.) The PLL is automatically in tracking mode when not in acquisition mode or when the $\overline{ACQ}$ bit is set.

8.4.5 Manual and Automatic PLL Bandwidth Modes

The PLL can change the bandwidth or operational mode of the loop filter manually or automatically. Automatic mode is recommended for most users.

In automatic bandwidth control mode (AUTO = 1), the lock detector automatically switches between acquisition and tracking modes. Automatic bandwidth control mode also is used to determine when the VCO clock, CGMVCLK, is safe to use as the source for the base clock, CGMOUT. (See **8.6.2 PLL Bandwidth Control Register**.) If PLL interrupts are enabled, the software can wait for a PLL interrupt request and then check the LOCK bit. If interrupts are disabled, software can poll the LOCK bit continuously (during PLL start-up, usually) or at periodic intervals. In either case, when the LOCK bit is set, the VCO clock is safe to use as the source for the base clock. (See **8.4.8 Base Clock Selector Circuit**.) If the VCO is selected as the source for the base clock and the LOCK bit is clear, the PLL has suffered a severe noise hit and the software must take appropriate action, depending on the application. (See **8.7 Interrupts** for information and precautions on using interrupts.)

The following conditions apply when the PLL is in automatic bandwidth control mode:

- The $\overline{\text{ACQ}}$ bit (See **8.6.2 PLL Bandwidth Control Register**.) is a read-only indicator of the mode of the filter. (See **8.4.4 Acquisition and Tracking Modes**.)
- The $\overline{\text{ACQ}}$ bit is set when the VCO frequency is within a certain tolerance and is cleared when the VCO frequency is out of a certain tolerance. (See **8.9 Acquisition/Lock Time Specifications** for more information.)
- The LOCK bit is a read-only indicator of the locked state of the PLL.
- The LOCK bit is set when the VCO frequency is within a certain tolerance and is cleared when the VCO frequency is out of a certain tolerance. (See **8.9 Acquisition/Lock Time Specifications** for more information.)
- CPU interrupts can occur if enabled (PLLIE = 1) when the PLL's lock condition changes, toggling the LOCK bit. (See **8.6.1 PLL Control Register**.)

The PLL also may operate in manual mode (AUTO = 0). Manual mode is used by systems that do not require an indicator of the lock condition for proper operation. Such systems typically operate well below f_{BUSMAX}.

The following conditions apply when in manual mode:

- $\overline{ACQ}$ is a writable control bit that controls the mode of the filter. Before turning on the PLL in manual mode, the $\overline{ACQ}$ bit must be clear.
- Before entering tracking mode ($\overline{ACQ}$ = 1), software must wait a given time, t_{ACQ} (See **8.9 Acquisition/Lock Time Specifications**.), after turning on the PLL by setting PLLON in the PLL control register (PCTL).
- Software must wait a given time, t_{AL}, after entering tracking mode before selecting the PLL as the clock source to CGMOUT (BCS = 1).
- The LOCK bit is disabled.
- CPU interrupts from the CGM are disabled.

8.4.6 Programming the PLL

The following procedure shows how to program the PLL.

NOTE: *The round function in the following equations means that the real number should be rounded to the nearest integer number.*

1. Choose the desired bus frequency, f_{BUSDES}.
2. Calculate the desired VCO frequency, $f_{VCLKDES}$.

$$f_{VCLKDES} = 2^P \times f_{CGMPCLK} = 2^P \times 4 \times f_{BUSDES}$$

   where P is the power of two multiplier, and can be 0, 1, 2, or 3
3. Choose a practical PLL reference frequency, f_{RCLK}, and the reference clock divider, R. Typically, the reference is 32.768 kHz and R = 1.

   Frequency errors to the PLL are corrected at a rate of f_{RCLK}/R. For stability and lock time reduction, this rate must be as fast as possible. The VCO frequency must be an integer multiple of this rate.

The relationship between the VCO frequency, f_{VCLK}, and the reference frequency, f_{RCLK}, is

$$f_{VCLK} = \frac{2^P N}{R}(f_{RCLK})$$

where N is the integer range multiplier, between 1 and 4095.

In cases where desired bus frequency has some tolerance, choose f_{RCLK} to a value determined either by other module requirements (such as modules which are clocked by CGMXCLK), cost requirements, or ideally, as high as the specified range allows. See **Section 23. Electrical Specifications**. Choose the reference divider, R = 1.

When the tolerance on the bus frequency is tight, choose f_{RCLK} to an integer divisor of f_{BUSDES}, and R = 1. If f_{RCLK} cannot meet this requirement, use the following equation to solve for R with practical choices of f_{RCLK}, and choose the f_{RCLK} that gives the lowest R.

$$R = \text{round}\left[R_{MAX} \times \left\{\left(\frac{f_{VCLKDES}}{f_{RCLK}}\right) - \text{integer}\left(\frac{f_{VCLKDES}}{f_{RCLK}}\right)\right\}\right]$$

4. Calculate N:

$$N = \text{round}\left(\frac{R \times f_{VCLKDES}}{f_{RCLK} \times 2^P}\right)$$

5. Calculate and verify the adequacy of the VCO and bus frequencies f_{VCLK} and f_{BUS}.

$$f_{VCLK} = \frac{2^P N}{R}(f_{RCLK})$$

$$f_{BUS} = \frac{f_{VCLK}}{2^P \times 4}$$

6. Select the VCO's power-of-two range multiplier E, according to this table:

Frequency Range	E
$0 < f_{VCLK} < 9{,}830{,}400$	0
$9{,}830{,}400 \leq f_{VCLK} < 19{,}660{,}800$	1
$19{,}660{,}800 \leq f_{VCLK} < 39{,}321{,}600$	2

NOTE: Do not program E to a value of 3.

7. Select a VCO linear range multiplier, L, where f_{NOM} = 38.4 kHz

$$L = \text{round}\left(\frac{f_{VCLK}}{2^E \times f_{NOM}}\right)$$

8. Calculate and verify the adequacy of the VCO programmed center-of-range frequency, f_{VRS}. The center-of-range frequency is the midpoint between the minimum and maximum frequencies attainable by the PLL.

$$f_{VRS} = (L \times 2^E) f_{NOM}$$

For proper operation,

$$|f_{VRS} - f_{VCLK}| \leq \frac{f_{NOM} \times 2^E}{2}$$

9. Verify the choice of P, R, N, E, and L by comparing f_{VCLK} to f_{VRS} and $f_{VCLKDES}$. For proper operation, f_{VCLK} must be within the application's tolerance of $f_{VCLKDES}$, and f_{VRS} must be as close as possible to f_{VCLK}.

NOTE: *Exceeding the recommended maximum bus frequency or VCO frequency can crash the MCU.*

10. Program the PLL registers accordingly:
    a. In the PRE bits of the PLL control register (PCTL), program the binary equivalent of P.
    b. In the VPR bits of the PLL control register (PCTL), program the binary equivalent of E.
    c. In the PLL multiplier select register low (PMSL) and the PLL multiplier select register high (PMSH), program the binary equivalent of N.
    d. In the PLL VCO range select register (PMRS), program the binary coded equivalent of L.
    e. In the PLL reference divider select register (PMDS), program the binary coded equivalent of R.

NOTE: *The values for P, E, N, L, and R can only be programmed when the PLL is off (PLLON = 0).*

Table 8-1 provides numeric examples (numbers are in hexadecimal notation):

Table 8-1. Numeric Examples

CGMVCLK	CGMPCLK	f_{BUS}	f_{RCLK}	R	N	P	E	L
8.0 MHz	8.0 MHz	2.0 MHz	32.768 kHz	1	F5	0	0	D1
9.8304 MHz	9.8304 MHz	2.4576 MHz	32.768 kHz	1	12C	0	1	80
10.0 MHz	10.0 MHz	2.5 MHz	32.768 kHz	1	132	0	1	83
16 MHz	16 MHz	4.0 MHz	32.768 kHz	1	1E9	0	1	D1
19.6608 MHz	19.6608 MHz	4.9152 MHz	32.768 kHz	1	258	0	2	80
20 MHz	20 MHz	5.0 MHz	32.768 kHz	1	263	0	2	82
29.4912 MHz	29.4912 MHz	7.3728 MHz	32.768 kHz	1	384	0	2	C0
32 MHz	32 MHz	8.0 MHz	32.768 kHz	1	3D1	0	2	D0
32 MHz	16 MHz	4.0 MHz	32.768 kHz	1	1E9	1	2	D0
32 MHz	8 MHz	2.0 MHz	32.768 kHz	1	F5	2	2	D0
32 MHz	4 MHz	1.0 MHz	32.768 kHz	1	7B	3	2	D0

8.4.7 Special Programming Exceptions

The programming method described in **8.4.6 Programming the PLL** does not account for three possible exceptions. A value of 0 for R, N, or L is meaningless when used in the equations given. To account for these exceptions:

- A 0 value for R or N is interpreted exactly the same as a value of 1.
- A 0 value for L disables the PLL and prevents its selection as the source for the base clock.

(See **8.4.8 Base Clock Selector Circuit**.)

8.4.8 Base Clock Selector Circuit

This circuit is used to select either the oscillator clock, CGMXCLK, or the divided VCO clock, CGMPCLK, as the source of the base clock, CGMOUT. The two input clocks go through a transition control circuit that waits up to three CGMXCLK cycles and three CGMPCLK cycles to change from one clock source to the other. During this time, CGMOUT is held in stasis. The output of the transition control circuit is then divided by two to correct the duty cycle. Therefore, the bus clock frequency, which is one-half of CGMOUT, is one-fourth the frequency of the selected clock (CGMXCLK or CGMPCLK).

For the CGMXCLK, the divide-by-2 can be by-passed by setting the DIV2CLK bit in the CONFIG2 register. Therefore, the bus clock frequency can be one-half of CGMXCLK.

The BCS bit in the PLL control register (PCTL) selects which clock drives CGMOUT. The divided VCO clock cannot be selected as the base clock source if the PLL is not turned on. The PLL cannot be turned off if the divided VCO clock is selected. The PLL cannot be turned on or off simultaneously with the selection or deselection of the divided VCO clock. The divided VCO clock also cannot be selected as the base clock source if the factor L is programmed to a 0. This value would set up a condition inconsistent with the operation of the PLL, so that the PLL would be disabled and the oscillator clock would be forced as the source of the base clock.

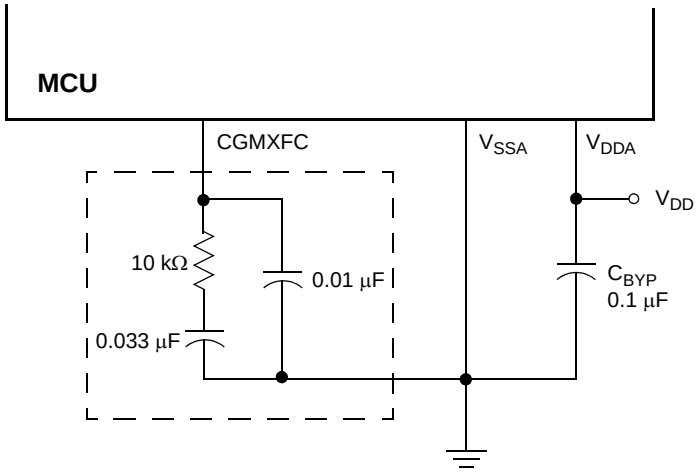
8.4.9 CGM External Connections

In its typical configuration, the CGM requires up to four external components.

Figure 8-3 shows the external components for the PLL:

- Bypass capacitor, C_{BYP}
- Filter network

Care should be taken with PCB routing in order to minimize signal cross talk and noise. (See **8.9 Acquisition/Lock Time Specifications** for routing information, filter network and its effects on PLL performance.)



Note: Filter network in box can be replaced with a 0.47µF capacitor, but will degrade stability.

Figure 8-3. CGM External Connections

8.5 I/O Signals

The following paragraphs describe the CGM I/O signals.

8.5.1 External Filter Capacitor Pin (CGMXFC)

The CGMXFC pin is required by the loop filter to filter out phase corrections. An external filter network is connected to this pin. (See **Figure 8-3**.)

NOTE: *To prevent noise problems, the filter network should be placed as close to the CGMXFC pin as possible, with minimum routing distances and no routing of other signals across the network.*

8.5.2 PLL Analog Power Pin (V_{DDA})

V_{DDA} is a power pin used by the analog portions of the PLL. Connect the V_{DDA} pin to the same voltage potential as the V_{DD} pin.

NOTE: *Route V_{DDA} carefully for maximum noise immunity and place bypass capacitors as close as possible to the package.*

8.5.3 PLL Analog Ground Pin (V_{SSA})

V_{SSA} is a ground pin used by the analog portions of the PLL. Connect the V_{SSA} pin to the same voltage potential as the V_{SS} pin.

NOTE: *Route V_{SSA} carefully for maximum noise immunity and place bypass capacitors as close as possible to the package.*

NOTE: *On this MCU, the V_{SSA} is physically bonded to the V_{SS} pin.*

8.5.4 Oscillator Output Frequency Signal (CGMXCLK)

CGMXCLK is the oscillator output signal. It runs at the full speed of the oscillator, and is generated directly from the crystal oscillator circuit, the RC oscillator circuit, or the internal oscillator circuit.

8.5.5 CGM Reference Clock (CGMRCLK)

CGMRCLK is a buffered version of CGMXCLK, this clock is the reference clock for the phase-locked-loop circuit.

8.5.6 CGM VCO Clock Output (CGMVCLK)

CGMVCLK is the clock output from the VCO.

8.5.7 CGM Base Clock Output (CGMOUT)

CGMOUT is the clock output of the CGM. This signal goes to the SIM, which generates the MCU clocks. CGMOUT is a 50 percent duty cycle clock running at twice the bus frequency. CGMOUT is software programmable to be equal to CGMXCLK, CGMXCLK divided by two, or CGMPCLK divided by two.

8.5.8 CGM CPU Interrupt (CGMINT)

CGMINT is the interrupt signal generated by the PLL lock detector.

8.6 CGM Registers

The following registers control and monitor operation of the CGM:

- PLL control register (PCTL)
  (See **8.6.1 PLL Control Register**.)
- PLL bandwidth control register (PBWC)
  (See **8.6.2 PLL Bandwidth Control Register**.)
- PLL multiplier select registers (PMSH and PMSL)
  (See **8.6.3 PLL Multiplier Select Registers**.)
- PLL VCO range select register (PMRS)
  (See **8.6.4 PLL VCO Range Select Register**.)
- PLL reference divider select register (PMDS)
  (See **8.6.5 PLL Reference Divider Select Register**.)

8.6.1 PLL Control Register

The PLL control register (PCTL) contains the interrupt enable and flag bits, the on/off switch, the base clock selector bit, the prescaler bits, and the VCO power-of-two range selector bits.

Address: $0036

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PLLIE	PLLF	PLLON	BCS	PRE1	PRE0	VPR1	VPR0
Write:								
Reset:	0	0	1	0	0	0	0	0

= Unimplemented

Figure 8-4. PLL Control Register (PCTL)

PLLIE — PLL Interrupt Enable Bit

This read/write bit enables the PLL to generate an interrupt request when the LOCK bit toggles, setting the PLL flag, PLLF. When the AUTO bit in the PLL bandwidth control register (PBWC) is clear, PLLIE cannot be written and reads as logic 0. Reset clears the PLLIE bit.

1 = PLL interrupts enabled
0 = PLL interrupts disabled

PLLF — PLL Interrupt Flag Bit

This read-only bit is set whenever the LOCK bit toggles. PLLF generates an interrupt request if the PLLIE bit also is set. PLLF always reads as logic 0 when the AUTO bit in the PLL bandwidth control register (PBWC) is clear. Clear the PLLF bit by reading the PLL control register. Reset clears the PLLF bit.

1 = Change in lock condition
0 = No change in lock condition

NOTE: *Do not inadvertently clear the PLLF bit. Any read or read-modify-write operation on the PLL control register clears the PLLF bit.*

PLLON — PLL On Bit

This read/write bit activates the PLL and enables the VCO clock, CGMVCLK. PLLON cannot be cleared if the VCO clock is driving the base clock, CGMOUT (BCS = 1). (See **8.4.8 Base Clock Selector Circuit**.) Reset sets this bit so that the loop can stabilize as the MCU is powering up.

1 = PLL on
0 = PLL off

BCS — Base Clock Select Bit

This read/write bit selects either the oscillator output, CGMXCLK, or the divided VCO clock, CGMPCLK, as the source of the CGM output, CGMOUT. CGMOUT frequency is one-half the frequency of the selected clock. BCS cannot be set while the PLLON bit is clear. After toggling BCS, it may take up to three CGMXCLK and three CGMPCLK cycles to complete the transition from one source clock to the other. During the transition, CGMOUT is held in stasis. (See **8.4.8 Base Clock Selector Circuit**.) Reset clears the BCS bit.

1 = CGMPCLK divided by two drives CGMOUT
0 = CGMXCLK divided by two drives CGMOUT

NOTE: *PLLON and BCS have built-in protection that prevents the base clock selector circuit from selecting the VCO clock as the source of the base clock if the PLL is off. Therefore, PLLON cannot be cleared when BCS is set, and BCS cannot be set when PLLON is clear. If the PLL is off (PLLON = 0), selecting CGMPCLK requires two writes to the PLL control register. (See **8.4.8 Base Clock Selector Circuit**.)*

PRE1 and PRE0 — Prescaler Program Bits

These read/write bits control a prescaler that selects the prescaler power-of-two multiplier, P. (See **8.4.3 PLL Circuits** and **8.4.6 Programming the PLL**.) PRE1 and PRE0 cannot be written when the PLLON bit is set. Reset clears these bits.

These prescaler bits affects the relationship between the VCO clock and the final system bus clock.

Table 8-2. PRE 1 and PRE0 Programming

PRE1 and PRE0	P	Prescaler Multiplier
00	0	1
01	1	2
10	2	4
11	3	8

VPR1 and VPR0 — VCO Power-of-Two Range Select Bits

These read/write bits control the VCO's hardware power-of-two range multiplier E that, in conjunction with L (See **8.4.3 PLL Circuits**, **8.4.6 Programming the PLL**, and **8.6.4 PLL VCO Range Select Register**.) controls the hardware center-of-range frequency, f_{VRS}. VPR1:VPR0 cannot be written when the PLLON bit is set. Reset clears these bits.

Table 8-3. VPR1 and VPR0 Programming

VPR1 and VPR0	E	VCO Power-of-Two Range Multiplier
00	0	1
01	1	2
10	2	4

NOTE: Do not program E to a value of 3.

8.6.2 PLL Bandwidth Control Register

The PLL bandwidth control register (PBWC):

- Selects automatic or manual (software-controlled) bandwidth control mode
- Indicates when the PLL is locked
- In automatic bandwidth control mode, indicates when the PLL is in acquisition or tracking mode
- In manual operation, forces the PLL into acquisition or tracking mode

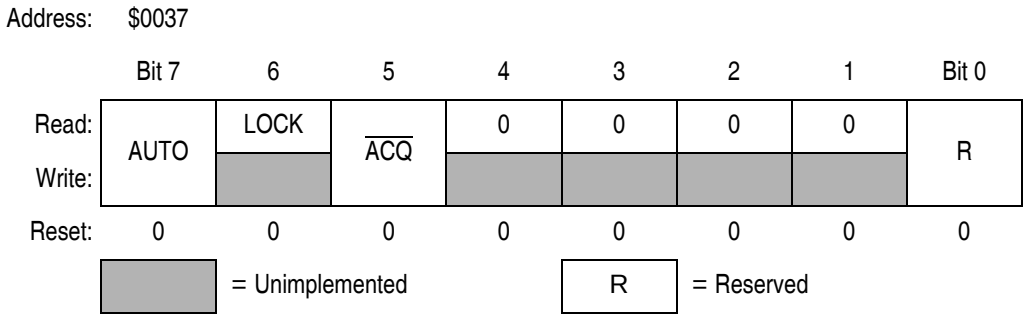


Figure 8-5. PLL Bandwidth Control Register (PBWCR)

AUTO — Automatic Bandwidth Control Bit

This read/write bit selects automatic or manual bandwidth control. When initializing the PLL for manual operation (AUTO = 0), clear the $\overline{ACQ}$ bit before turning on the PLL. Reset clears the AUTO bit.

1 = Automatic bandwidth control
0 = Manual bandwidth control

LOCK — Lock Indicator Bit

When the AUTO bit is set, LOCK is a read-only bit that becomes set when the VCO clock, CGMVCLK, is locked (running at the programmed frequency). When the AUTO bit is clear, LOCK reads as logic 0 and has no meaning. The write one function of this bit is reserved for test, so this bit must *always* be written a 0. Reset clears the LOCK bit.

1 = VCO frequency correct or locked
0 = VCO frequency incorrect or unlocked

$\overline{ACQ}$ — Acquisition Mode Bit

When the AUTO bit is set, $\overline{ACQ}$ is a read-only bit that indicates whether the PLL is in acquisition mode or tracking mode. When the AUTO bit is clear, $\overline{ACQ}$ is a read/write bit that controls whether the PLL is in acquisition or tracking mode.

In automatic bandwidth control mode (AUTO = 1), the last-written value from manual operation is stored in a temporary location and is recovered when manual operation resumes. Reset clears this bit, enabling acquisition mode.

1 = Tracking mode
0 = Acquisition mode

8.6.3 PLL Multiplier Select Registers

The PLL multiplier select registers (PMSH and PMSL) contain the programming information for the modulo feedback divider.

Address: $0038

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	MUL11	MUL10	MUL9	MUL8
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 8-6. PLL Multiplier Select Register High (PMSH)

Address: $0039

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	MUL7	MUL6	MUL5	MUL4	MUL3	MUL2	MUL1	MUL0
Write:								
Reset:	0	1	0	0	0	0	0	0

Figure 8-7. PLL Multiplier Select Register Low (PMSL)

MUL[11:0] — Multiplier Select Bits

These read/write bits control the modulo feedback divider that selects the VCO frequency multiplier N. (See **8.4.3 PLL Circuits** and **8.4.6 Programming the PLL**.) A value of $0000 in the multiplier select registers configure the modulo feedback divider the same as a value of $0001. Reset initializes the registers to $0040 for a default multiply value of 64.

NOTE: *The multiplier select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1).*

8.6.4 PLL VCO Range Select Register

The PLL VCO range select register (PMRS) contains the programming information required for the hardware configuration of the VCO.

Address: $003A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	VRS7	VRS6	VRS5	VRS4	VRS3	VRS2	VRS1	VRS0
Reset:	0	1	0	0	0	0	0	0

Figure 8-8. PLL VCO Range Select Register (PMRS)

VRS[7:0] — VCO Range Select Bits

These read/write bits control the hardware center-of-range linear multiplier L which, in conjunction with E (See **8.4.3 PLL Circuits**, **8.4.6 Programming the PLL**, and **8.6.1 PLL Control Register**.), controls the hardware center-of-range frequency, f_{VRS}. VRS[7:0] cannot be written when the PLLON bit in the PCTL is set. (See **8.4.7 Special Programming Exceptions**.) A value of $00 in the VCO range select register disables the PLL and clears the BCS bit in the PLL control register (PCTL). (See **8.4.8 Base Clock Selector Circuit** and **8.4.7 Special Programming Exceptions**.). Reset initializes the register to $40 for a default range multiply value of 64.

NOTE: *The VCO range select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1) and such that the VCO clock cannot be selected as the source of the base clock (BCS = 1) if the VCO range select bits are all clear.*

The PLL VCO range select register must be programmed correctly. Incorrect programming can result in failure of the PLL to achieve lock.

8.6.5 PLL Reference Divider Select Register

The PLL reference divider select register (PMDS) contains the programming information for the modulo reference divider.

Address: $003B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	RDS3	RDS2	RDS1	RDS0
Write:								
Reset:	0	0	0	0	0	0	0	1

= Unimplemented

Figure 8-9. PLL Reference Divider Select Register (PMDS)

RDS[3:0] — Reference Divider Select Bits

These read/write bits control the modulo reference divider that selects the reference division factor, R. (See **8.4.3 PLL Circuits** and **8.4.6 Programming the PLL**.) RDS[3:0] cannot be written when the PLLON bit in the PCTL is set. A value of $00 in the reference divider select register configures the reference divider the same as a value of $01. (See **8.4.7 Special Programming Exceptions**.) Reset initializes the register to $01 for a default divide value of 1.

NOTE: *The reference divider select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1).*

NOTE: *The default divide value of 1 is recommended for all applications.*

8.7 Interrupts

When the AUTO bit is set in the PLL bandwidth control register (PBWC), the PLL can generate a CPU interrupt request every time the LOCK bit changes state. The PLLIE bit in the PLL control register (PCTL) enables CPU interrupts from the PLL. PLLF, the interrupt flag in the PCTL, becomes set whether interrupts are enabled or not. When the AUTO bit is clear, CPU interrupts from the PLL are disabled and PLLF reads as logic 0.

Software should read the LOCK bit after a PLL interrupt request to see if the request was due to an entry into lock or an exit from lock. When the PLL enters lock, the divided VCO clock, CGMPCLK, divided by two can be selected as the CGMOUT source by setting BCS in the PCTL. When the PLL exits lock, the VCO clock frequency is corrupt, and appropriate precautions should be taken. If the application is not frequency sensitive, interrupts should be disabled to prevent PLL interrupt service routines from impeding software performance or from exceeding stack limitations.

NOTE: *Software can select the CGMPCLK divided by two as the CGMOUT source even if the PLL is not locked (LOCK = 0). Therefore, software should make sure the PLL is locked before setting the BCS bit.*

8.8 Special Modes

The WAIT instruction puts the MCU in low power-consumption standby modes.

8.8.1 Wait Mode

The WAIT instruction does not affect the CGM. Before entering wait mode, software can disengage and turn off the PLL by clearing the BCS and PLLON bits in the PLL control register (PCTL) to save power. Less power-sensitive applications can disengage the PLL without turning it off, so that the PLL clock is immediately available at WAIT exit. This would be the case also when the PLL is to wake the MCU from wait mode, such as when the PLL is first enabled and waiting for LOCK or LOCK is lost.

8.8.2 Stop Mode

If the oscillator stop mode enable bit (STOP_XCLKEN in CONFIG2 register) is configured to disabled the oscillator in stop mode, then the STOP instruction disables the CGM (oscillator and phase locked loop) and holds low all CGM outputs (CGMOUT, CGMVCLK, CGMPCLK, and CGMINT).

If the STOP instruction is executed with the divided VCO clock, CGMPCLK, divided by two driving CGMOUT, the PLL automatically clears the BCS bit in the PLL control register (PCTL), thereby selecting the oscillator clock, CGMXCLK, divided by two as the source of CGMOUT. When the MCU recovers from STOP, the crystal clock divided by two drives CGMOUT and BCS remains clear.

If the oscillator stop mode enable bit is configured for continuous oscillator operation in stop mode, then the phase locked loop is shut off but the CGMXCLK will continue to drive the SIM and other MCU sub-systems.

8.8.3 CGM During Break Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state. (See **9.8.3 SIM Break Flag Control Register**.)

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the PLLF bit during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write the PLL control register during the break state without affecting the PLLF bit.

8.9 Acquisition/Lock Time Specifications

The acquisition and lock times of the PLL are, in many applications, the most critical PLL design parameters. Proper design and use of the PLL ensures the highest stability and lowest acquisition/lock times.

8.9.1 Acquisition/Lock Time Definitions

Typical control systems refer to the acquisition time or lock time as the reaction time, within specified tolerances, of the system to a step input. In a PLL, the step input occurs when the PLL is turned on or when it suffers a noise hit. The tolerance is usually specified as a percent of the step input or when the output settles to the desired value plus or minus a percent of the frequency change. Therefore, the reaction time is constant in this definition, regardless of the size of the step input. For example, consider a system with a 5 percent acquisition time tolerance. If a command instructs the system to change from 0 Hz to 1 MHz, the acquisition time is the time taken for the frequency to reach 1 MHz ±50 kHz. 50 kHz = 5% of the 1 MHz step input. If the system is operating at 1 MHz and suffers a –100 kHz noise hit, the acquisition time is the time taken to return from 900 kHz to 1 MHz ±5 kHz. 5 kHz = 5% of the 100 kHz step input.

Other systems refer to acquisition and lock times as the time the system takes to reduce the error between the actual output and the desired output to within specified tolerances. Therefore, the acquisition or lock time varies according to the original error in the output. Minor errors may not even be registered. Typical PLL applications prefer to use this definition because the system requires the output frequency to be within a certain tolerance of the desired frequency regardless of the size of the initial error.

8.9.2 Parametric Influences on Reaction Time

Acquisition and lock times are designed to be as short as possible while still providing the highest possible stability. These reaction times are not constant, however. Many factors directly and indirectly affect the acquisition time.

The most critical parameter which affects the reaction times of the PLL is the reference frequency, f_{RDV}. This frequency is the input to the phase detector and controls how often the PLL makes corrections. For stability, the corrections must be small compared to the desired frequency, so several corrections are required to reduce the frequency error. Therefore, the slower the reference the longer it takes to make these corrections. This parameter is under user control via the choice of crystal frequency f_{XCLK} and the R value programmed in the reference divider. (See **8.4.3 PLL Circuits**, **8.4.6 Programming the PLL**, and **8.6.5 PLL Reference Divider Select Register**.)

Another critical parameter is the external filter network. The PLL modifies the voltage on the VCO by adding or subtracting charge from capacitors in this network. Therefore, the rate at which the voltage changes for a given frequency error (thus change in charge) is proportional to the capacitance. The size of the capacitor also is related to the stability of the PLL. If the capacitor is too small, the PLL cannot make small enough adjustments to the voltage and the system cannot lock. If the capacitor is too large, the PLL may not be able to adjust the voltage in a reasonable time. (See **8.9.3 Choosing a Filter**.)

Also important is the operating voltage potential applied to V_{DDA}. The power supply potential alters the characteristics of the PLL. A fixed value is best. Variable supplies, such as batteries, are acceptable if they vary within a known range at very slow speeds. Noise on the power supply is not acceptable, because it causes small frequency errors which continually change the acquisition time of the PLL.

Temperature and processing also can affect acquisition time because the electrical characteristics of the PLL change. The part operates as specified as long as these influences stay within the specified limits. External factors, however, can cause drastic changes in the operation of the PLL. These factors include noise injected into the PLL through the filter capacitor, filter capacitor leakage, stray impedances on the circuit board, and even humidity or circuit board contamination.

8.9.3 Choosing a Filter

As described in **8.9.2 Parametric Influences on Reaction Time**, the external filter network is critical to the stability and reaction time of the PLL. The PLL is also dependent on reference frequency and supply voltage.

Either of the filter networks in **Figure 8-10** is recommended when using a 32.768kHz reference clock (CGMRCLK). **Figure 8-10 (a)** is used for applications requiring better stability. **Figure 8-10 (b)** is used in low-cost applications where stability is not critical.

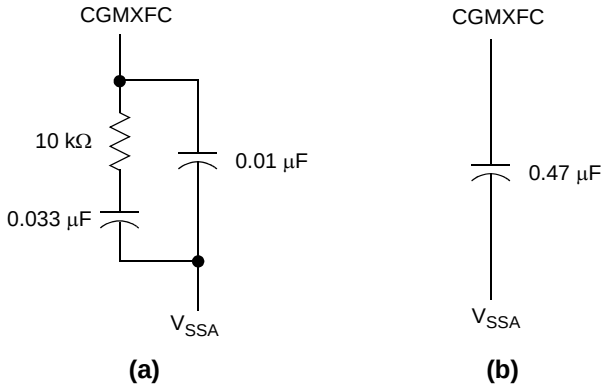


Figure 8-10. PLL Filter

Section 9. System Integration Module (SIM)

9.1 Contents

9.2 Introduction 132

9.3 SIM Bus Clock Control and Generation 134
9.3.1 Bus Timing 135
9.3.2 Clock Start-up from POR or LVI Reset 135
9.3.3 Clocks in Stop Mode and Wait Mode 136

9.4 Reset and System Initialization 136
9.4.1 External Pin Reset 137
9.4.2 Active Resets from Internal Sources 137
9.4.2.1 Power-On Reset 138
9.4.2.2 Computer Operating Properly (COP) Reset 139
9.4.2.3 Illegal Opcode Reset 140
9.4.2.4 Illegal Address Reset 140
9.4.2.5 Low-Voltage Inhibit (LVI) Reset 140
9.4.2.6 Monitor Mode Entry Module Reset (MODRST) 140

9.5 SIM Counter 141
9.5.1 SIM Counter During Power-On Reset 141
9.5.2 SIM Counter During Stop Mode Recovery 141
9.5.3 SIM Counter and Reset States 141

9.6 Exception Control 142
9.6.1 Interrupts 142
9.6.1.1 Hardware Interrupts 144
9.6.1.2 SWI Instruction 145
9.6.1.3 Interrupt Status Registers 145
9.6.1.4 Interrupt Status Register 1 145
9.6.1.5 Interrupt Status Register 2 147
9.6.1.6 Interrupt Status Register 3 147
9.6.2 Reset 148
9.6.3 Break Interrupts 148

9.6.4 Status Flag Protection in Break Mode 148
9.7 Low-Power Modes 149
9.7.1 Wait Mode 149
9.7.2 Stop Mode 150
9.8 SIM Registers 151
9.8.1 SIM Break Status Register 152
9.8.2 SIM Reset Status Register 153
9.8.3 SIM Break Flag Control Register 154

9.2 Introduction

This section describes the system integration module (SIM). Together with the CPU, the SIM controls all MCU activities. A block diagram of the SIM is shown in **Figure 9-1**. **Table 9-1** is a summary of the SIM input/output (I/O) registers. The SIM is a system state controller that coordinates CPU and exception timing. The SIM is responsible for:

- Bus clock generation and control for CPU and peripherals:
  - Stop/wait/reset/break entry and recovery
  - Internal clock control
- Master reset control, including power-on reset (POR) and COP timeout
- Interrupt control:
  - Acknowledge timing
  - Arbitration control timing
  - Vector address generation
- CPU enable/disable timing
- Modular architecture expandable to 128 interrupt sources

Table 9-1 shows the internal signal names used in this section.

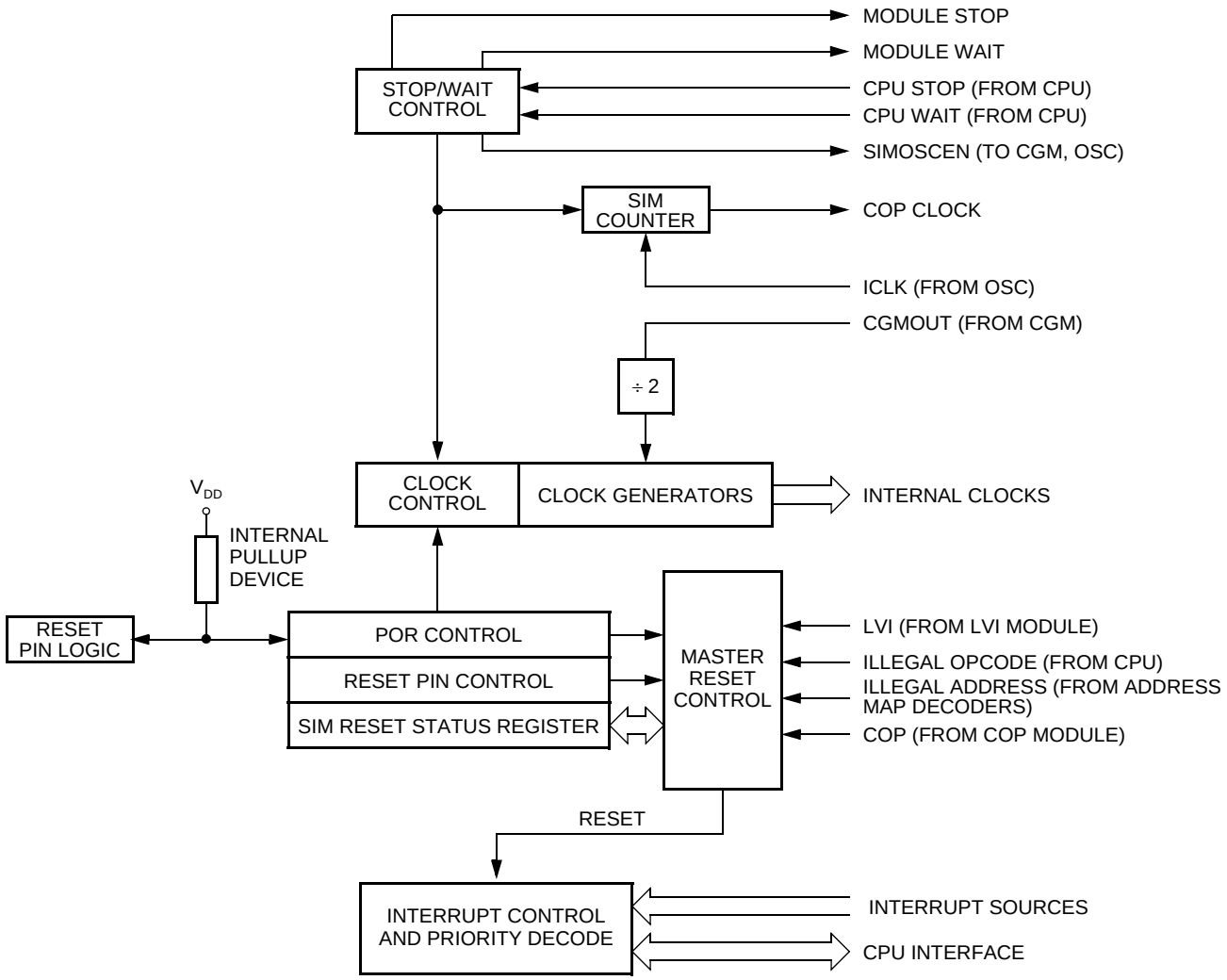


Figure 9-1. SIM Block Diagram

Table 9-1. Signal Name Conventions

Signal Name	Description
ICLK	Internal RC oscillator clock
CGMXCLK	Buffered version of OSC1 from the oscillator module
CGMPCLK	PLL output and the divided PLL output
CGMOUT	PLL-based or oscillator-based clock output from CGM module (Bus clock = CGMOUT ÷ 2)
IAB	Internal address bus
IDB	Internal data bus
PORRST	Signal from the power-on reset module to the SIM
IRST	Internal reset signal
R/$\overline{W}$	Read/write signal

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	SIM Break Status Register (SBSR)	Read:	R	R	R	R	R	R	SBSW	R
		Write:							Note	
		Reset:							0	
Note: Writing a logic 0 clears SBSW.										
$FE01	SIM Reset Status Register (SRSR)	Read:	POR	PIN	COP	ILOP	ILAD	0	LVI	0
		Write:								
		POR:	1	0	0	0	0	0	0	0
$FE03	SIM Break Flag Control Register (SBFCR)	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
$FE04	Interrupt Status Register 1 (INT1)	Read:	IF6	IF5	IF4	IF3	IF2	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE05	Interrupt Status Register 2 (INT2)	Read:	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE06	Interrupt Status Register 3 (INT3)	Read:	0	0	0	0	0	IF17	IF16	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
			= Unimplemented				R	= Reserved		

Figure 9-2. SIM I/O Register Summary

9.3 SIM Bus Clock Control and Generation

The bus clock generator provides system clock signals for the CPU and peripherals on the MCU. The system clocks are generated from an incoming clock, CGMOUT, as shown in **Figure 9-3**. This clock can come from either the oscillator module or from the on-chip PLL. (See **Section 8. Clock Generator Module (CGM)**.)

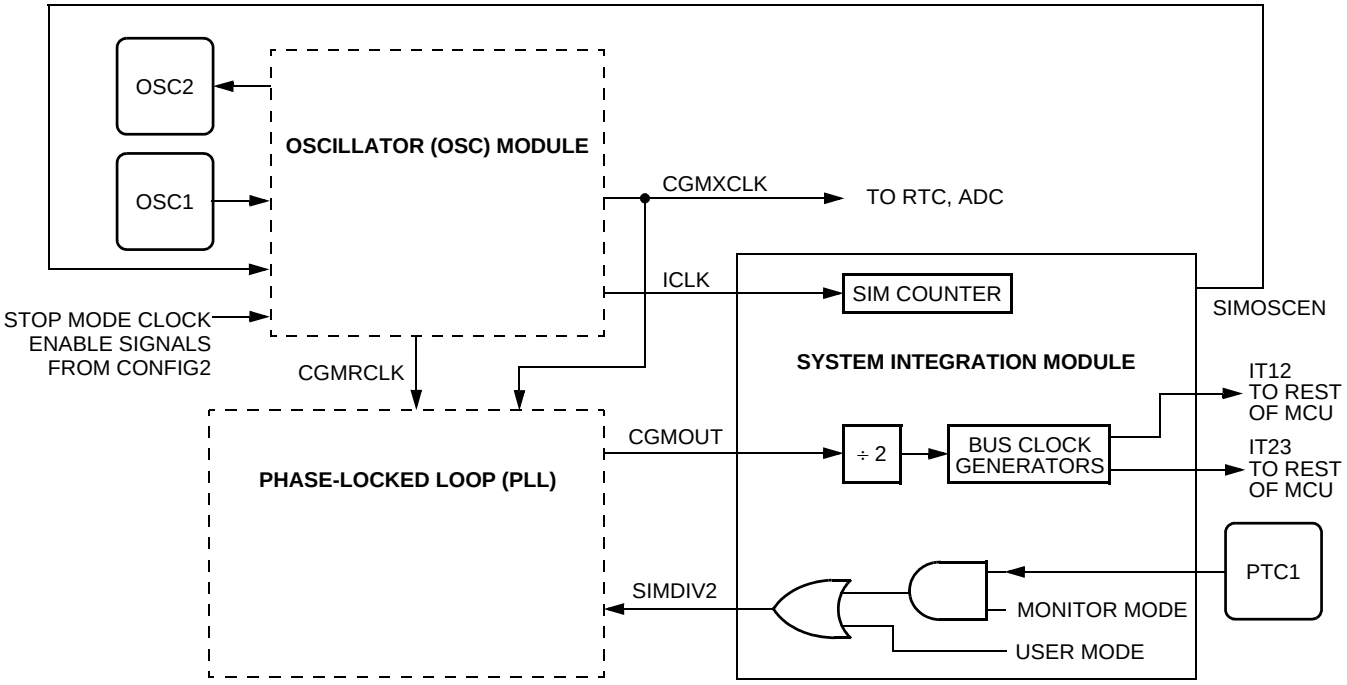


Figure 9-3. CGM Clock Signals

9.3.1 Bus Timing

In user mode, the internal bus frequency is either the oscillator output (CGMXCLK) divided by four, CGMXCLK divided by two, or the PLL output (CGMPCLK) divided by four.

9.3.2 Clock Start-up from POR or LVI Reset

When the power-on reset module or the low-voltage inhibit module generates a reset, the clocks to the CPU and peripherals are inactive and held in an inactive phase until after the 4096 ICLK cycle POR timeout has completed. The $\overline{\text{RST}}$ pin is driven low by the SIM during this entire period. The IBUS clocks start upon completion of the timeout.

9.3.3 Clocks in Stop Mode and Wait Mode

Upon exit from stop mode by an interrupt, break, or reset, the SIM allows ICLK to clock the SIM counter. The CPU and peripheral clocks do not become active until after the stop delay timeout. This timeout is selectable as 4096 or 32 ICLK cycles. (See **9.7.2 Stop Mode**.)

In wait mode, the CPU clocks are inactive. The SIM also produces two sets of clocks for other modules. Refer to the wait mode subsection of each module to see if the module is active or inactive in wait mode. Some modules can be programmed to be active in wait mode.

9.4 Reset and System Initialization

The MCU has these reset sources:

- Power-on reset module (POR)
- External reset pin ($\overline{\text{RST}}$)
- Computer operating properly module (COP)
- Low-voltage inhibit module (LVI)
- Illegal opcode
- Illegal address

All of these resets produce the vector $FFFE:$FFFF ($FEFE:$FEFF in monitor mode) and assert the internal reset signal (IRST). IRST causes all registers to be returned to their default values and all modules to be returned to their reset states.

An internal reset clears the SIM counter (see **9.5 SIM Counter**), but an external reset does not. Each of the resets sets a corresponding bit in the SIM reset status register (SRSR). (See **9.8 SIM Registers**.)

9.4.1 External Pin Reset

The $\overline{\text{RST}}$ pin circuit includes an internal pull-up device. Pulling the asynchronous $\overline{\text{RST}}$ pin low halts all processing. The PIN bit of the SIM reset status register (SRSR) is set as long as $\overline{\text{RST}}$ is held low for a minimum of 67 ICLK cycles, assuming that neither the POR nor the LVI was the source of the reset. See **Table 9-2** for details.
Figure 9-4 shows the relative timing.

Table 9-2. PIN Bit Set Timing

Reset Type	Number of Cycles Required to Set PIN
POR/LVI	4163 (4096 + 64 + 3)
All others	67 (64 + 3)

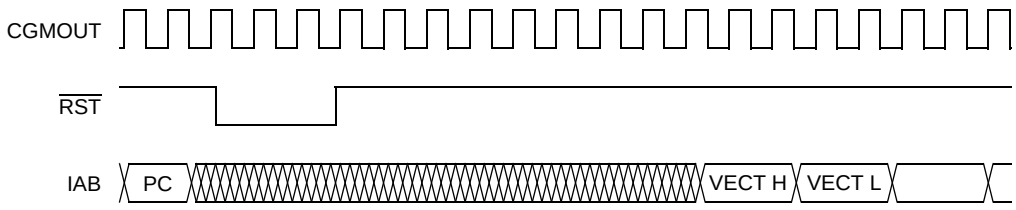


Figure 9-4. External Reset Timing

9.4.2 Active Resets from Internal Sources

All internal reset sources actively pull the $\overline{\text{RST}}$ pin low for 32 ICLK cycles to allow resetting of external peripherals. The internal reset signal IRST continues to be asserted for an additional 32 cycles (see **Figure 9-5**). An internal reset can be caused by an illegal address, illegal opcode, COP timeout, LVI, or POR (see **Figure 9-6**).

NOTE: *For LVI or POR resets, the SIM cycles through 4096 + 32 ICLK cycles during which the SIM forces the $\overline{\text{RST}}$ pin low. The internal reset signal then follows the sequence from the falling edge of $\overline{\text{RST}}$ shown in **Figure 9-5**.*

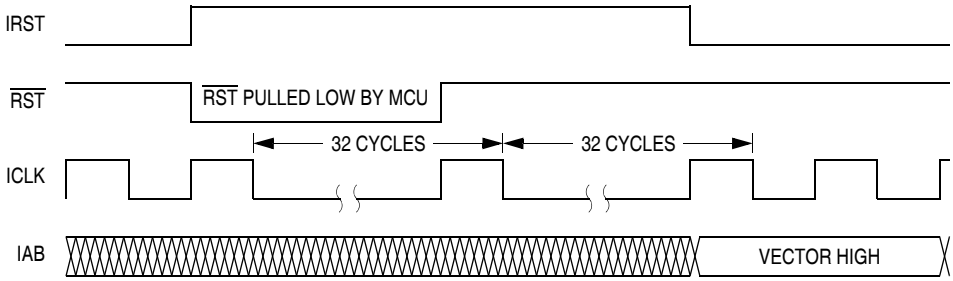


Figure 9-5. Internal Reset Timing

The COP reset is asynchronous to the bus clock.

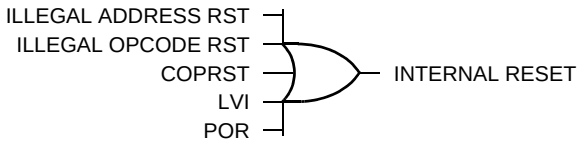


Figure 9-6. Sources of Internal Reset

The active reset feature allows the part to issue a reset to peripherals and other chips within a system built around the MCU.

9.4.2.1 Power-On Reset

When power is first applied to the MCU, the power-on reset module (POR) generates a pulse to indicate that power-on has occurred. The external reset pin ($\overline{RST}$) is held low while the SIM counter counts out 4096 + 32 ICLK cycles. Thirty-two ICLK cycles later, the CPU and memories are released from reset to allow the reset vector sequence to occur.

At power-on, these events occur:

- A POR pulse is generated.
- The internal reset signal is asserted.
- The SIM enables CGMOUT.
- Internal clocks to the CPU and modules are held inactive for 4096 ICLK cycles to allow stabilization of the oscillator.
- The $\overline{RST}$ pin is driven low during the oscillator stabilization time.
- The POR bit of the SIM reset status register (SRSR) is set and all other bits in the register are cleared.

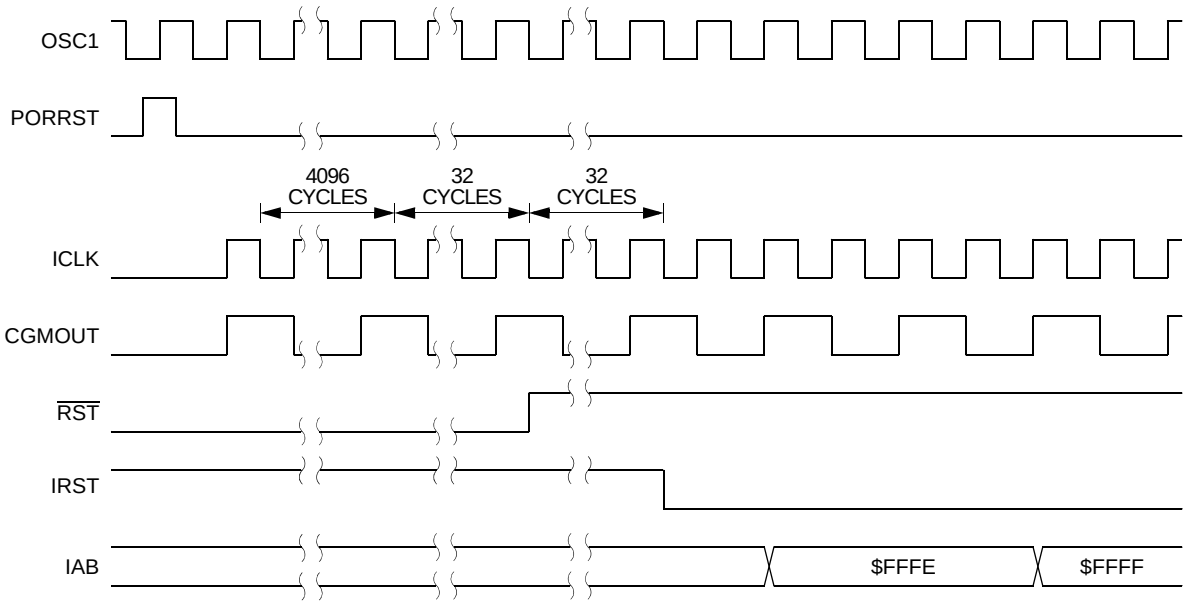


Figure 9-7. POR Recovery

9.4.2.2 Computer Operating Properly (COP) Reset

An input to the SIM is reserved for the COP reset signal. The overflow of the COP counter causes an internal reset and sets the COP bit in the SIM reset status register (SRSR). The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

To prevent a COP module timeout, write any value to location $FFFF. Writing to location $FFFF clears the COP counter and bits 12 through 5 of the SIM counter. The SIM counter output, which occurs at least every $2^{13} – 2^4$ ICLK cycles, drives the COP counter. The COP should be serviced as soon as possible out of reset to guarantee the maximum amount of time before the first timeout.

The COP module is disabled if the $\overline{\text{RST}}$ pin or the $\overline{\text{IRQ}}$ pin is held at V_{TST} while the MCU is in monitor mode. The COP module can be disabled only through combinational logic conditioned with the high voltage signal on the $\overline{\text{RST}}$ or the $\overline{\text{IRQ}}$ pin. This prevents the COP from becoming disabled as a result of external noise. During a break state, V_{TST} on the $\overline{\text{RST}}$ pin disables the COP module.

9.4.2.3 Illegal Opcode Reset

The SIM decodes signals from the CPU to detect illegal instructions. An illegal instruction sets the ILOP bit in the SIM reset status register (SRSR) and causes a reset.

If the stop enable bit, STOP, in the mask option register is logic 0, the SIM treats the STOP instruction as an illegal opcode and causes an illegal opcode reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

9.4.2.4 Illegal Address Reset

An opcode fetch from an unmapped address generates an illegal address reset. The SIM verifies that the CPU is fetching an opcode prior to asserting the ILAD bit in the SIM reset status register (SRSR) and resetting the MCU. A data fetch from an unmapped address does not generate a reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

9.4.2.5 Low-Voltage Inhibit (LVI) Reset

The low-voltage inhibit module (LVI) asserts its output to the SIM when the V_{DD} voltage falls to the LVI trip falling voltage, V_{TRIPF}. The LVI bit in the SIM reset status register (SRSR) is set, and the external reset pin ($\overline{\text{RST}}$) is held low while the SIM counter counts out 4096 + 32 ICLK cycles. Thirty-two ICLK cycles later, the CPU is released from reset to allow the reset vector sequence to occur. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

9.4.2.6 Monitor Mode Entry Module Reset (MODRST)

The monitor mode entry module reset (MODRST) asserts its output to the SIM when monitor mode is entered in the condition where the reset vectors are blank ($FF). (See **Section 10. Monitor ROM (MON)**.) When MODRST gets asserted, an internal reset occurs. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

9.5 SIM Counter

The SIM counter is used by the power-on reset module (POR) and in stop mode recovery to allow the oscillator time to stabilize before enabling the internal bus (IBUS) clocks. The SIM counter also serves as a prescaler for the computer operating properly module (COP). The SIM counter overflow supplies the clock for the COP module. The SIM counter is 12 bits long and is clocked by the falling edge of ICLK.

9.5.1 SIM Counter During Power-On Reset

The power-on reset module (POR) detects power applied to the MCU. At power-on, the POR circuit asserts the signal PORRST. Once the SIM is initialized, it enables the clock generation module (CGM) to drive the bus clock state machine.

9.5.2 SIM Counter During Stop Mode Recovery

The SIM counter also is used for stop mode recovery. The STOP instruction clears the SIM counter. After an interrupt, break, or reset, the SIM senses the state of the short stop recovery bit, SSREC, in the configuration register 1 (CONFIG1). If the SSREC bit is a logic 1, then the stop recovery is reduced from the normal delay of 4096 ICLK cycles down to 32 ICLK cycles. This is ideal for applications using canned oscillators that do not require long start-up times from stop mode. External crystal applications should use the full stop recovery time, that is, with SSREC cleared.

9.5.3 SIM Counter and Reset States

External reset has no effect on the SIM counter. (See **9.7.2 Stop Mode** for details.) The SIM counter is free-running after all reset states. (See **9.4.2 Active Resets from Internal Sources** for counter control and internal reset recovery sequences.)

9.6 Exception Control

Normal, sequential program execution can be changed in three different ways:

- Interrupts:
  - Maskable hardware CPU interrupts
  - Non-maskable software interrupt instruction (SWI)
- Reset
- Break interrupts

9.6.1 Interrupts

At the beginning of an interrupt, the CPU saves the CPU register contents on the stack and sets the interrupt mask (I bit) to prevent additional interrupts. At the end of an interrupt, the RTI instruction recovers the CPU register contents from the stack so that normal processing can resume. **Figure 9-8** shows interrupt entry timing, and **Figure 9-9** shows interrupt recovery timing.

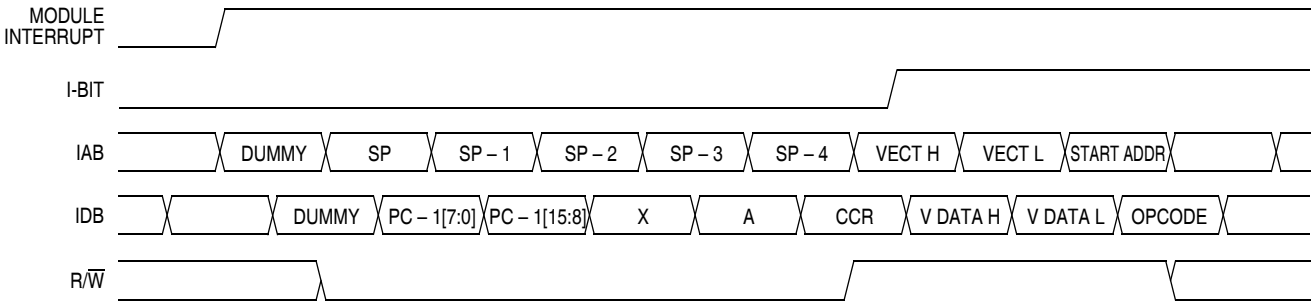


Figure 9-8. Interrupt Entry Timing

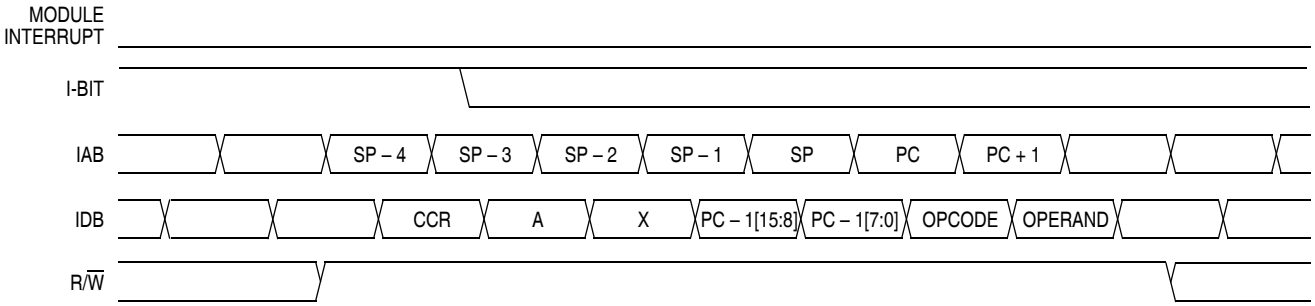


Figure 9-9. Interrupt Recovery Timing

Interrupts are latched, and arbitration is performed in the SIM at the start of interrupt processing. The arbitration result is a constant that the CPU uses to determine which vector to fetch. Once an interrupt is latched by the SIM, no other interrupt can take precedence, regardless of priority, until the latched interrupt is serviced (or the I bit is cleared).
(See **Figure 9-10**.)

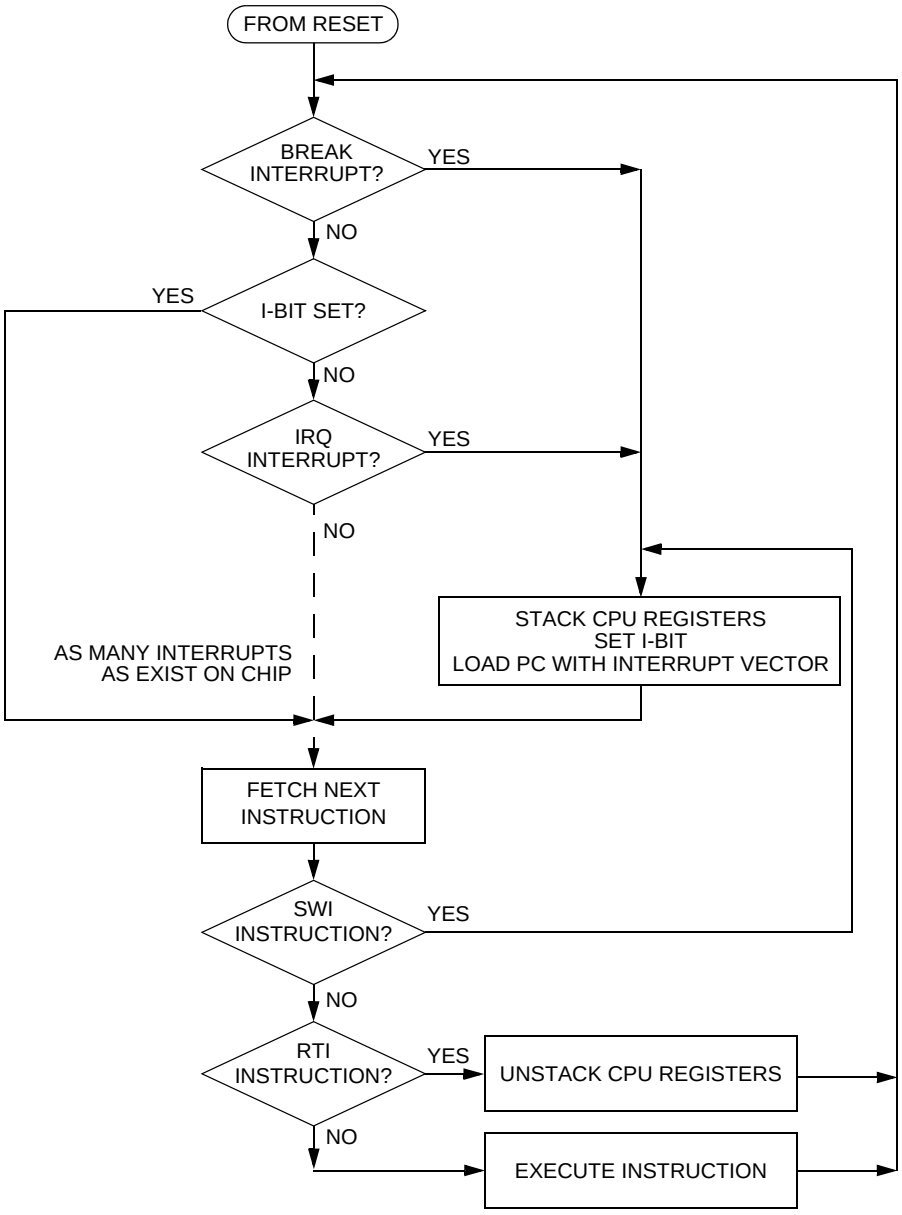


Figure 9-10. Interrupt Processing

9.6.1.1 Hardware Interrupts

A hardware interrupt does not stop the current instruction. Processing of a hardware interrupt begins after completion of the current instruction. When the current instruction is complete, the SIM checks all pending hardware interrupts. If interrupts are not masked (I bit clear in the condition code register) and if the corresponding interrupt enable bit is set, the SIM proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If more than one interrupt is pending at the end of an instruction execution, the highest priority interrupt is serviced first. **Figure 9-11** demonstrates what happens when two interrupts are pending. If an interrupt is pending upon exit from the original interrupt service routine, the pending interrupt is serviced before the LDA instruction is executed.

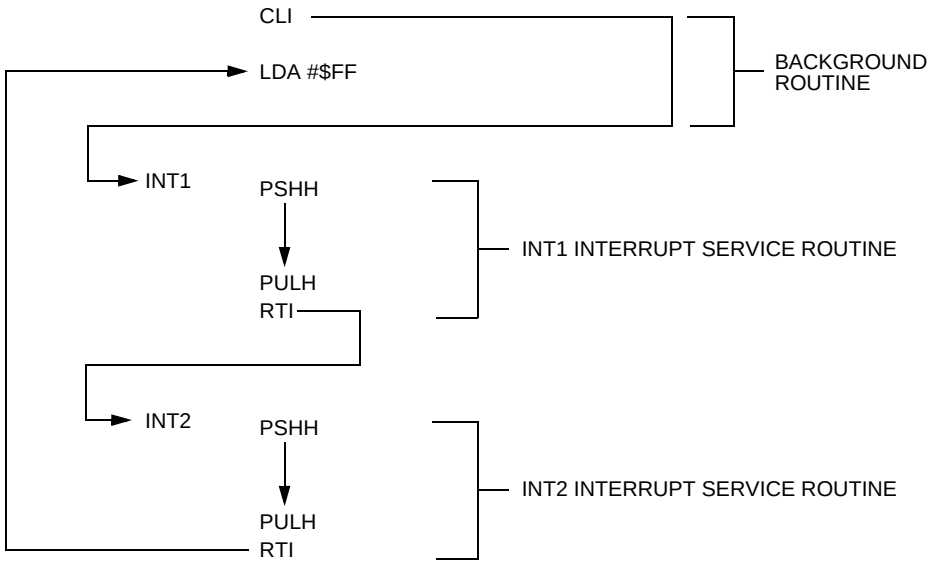


Figure 9-11. Interrupt Recognition Example

The LDA opcode is prefetched by both the INT1 and INT2 RTI instructions. However, in the case of the INT1 RTI prefetch, this is a redundant operation.

NOTE: *To maintain compatibility with the M6805 Family, the H register is not pushed on the stack during interrupt entry. If the interrupt service routine modifies the H register or uses the indexed addressing mode, software should save the H register and then restore it prior to exiting the routine.*

9.6.1.2 SWI Instruction

The SWI instruction is a non-maskable instruction that causes an interrupt regardless of the state of the interrupt mask (I bit) in the condition code register.

NOTE: *A software interrupt pushes PC onto the stack. A software interrupt does not push PC – 1, as a hardware interrupt does.*

9.6.1.3 Interrupt Status Registers

The flags in the interrupt status registers identify maskable interrupt sources. **Table 9-3** summarizes the interrupt sources and the interrupt status register flags that they set. The interrupt status registers can be useful for debugging.

9.6.1.4 Interrupt Status Register 1

Address: $FE04

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	IF6	IF5	IF4	IF3	IF2	IF1	0	0
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 9-12. Interrupt Status Register 1 (INT1)

IF6–IF1 — Interrupt Flags 6–1

These flags indicate the presence of interrupt requests from the sources shown in **Table 9-3**.

1 = Interrupt request present
0 = No interrupt request present

Bit 0 and Bit 1 — Always read 0

Table 9-3. Vector Addresses

Priority	INT Flag	Address	Vector
Lowest	IF17	$FFDA	Real Time Clock Vector (High)
		$FFDB	Real Time Clock Vector (Low)
	IF16	$FFDC	ADC Conversion Complete Vector (High)
		$FFDD	ADC Conversion Complete Vector (Low)
	IF15	$FFDE	Keyboard Vector (High)
		$FFDF	Keyboard Vector (Low)
	IF14	$FFE0	SCI Transmit Vector (High)
		$FFE1	SCI Transmit Vector (Low)
	IF13	$FFE2	SCI Receive Vector (High)
		$FFE3	SCI Receive Vector (Low)
	IF12	$FFE4	SCI Error Vector (High)
		$FFE5	SCI Error Vector (Low)
	IF11	$FFE6	SPI Receive Vector (High)
		$FFE7	SPI Receive Vector (Low)
	IF10	$FFE8	SPI Transmit Vector (High)
		$FFE9	SPI Transmit Vector (Low)
	IF9	$FFEA	TIM2 Overflow Vector (High)
		$FFEB	TIM2 Overflow Vector (Low)
	IF8	$FFEC	TIM2 Channel 1 Vector (High)
		$FFED	TIM2 Channel 1 Vector (Low)
	IF7	$FFEE	TIM2 Channel 0 Vector (High)
		$FFEF	TIM2 Channel 0 Vector (Low)
	IF6	$FFF0	TIM1 Overflow Vector (High)
		$FFF1	TIM1 Overflow Vector (Low)
	IF5	$FFF2	TIM1 Channel 1 Vector (High)
		$FFF3	TIM1 Channel 1 Vector (Low)
	IF4	$FFF4	TIM1 Channel 0 Vector (High)
		$FFF5	TIM1 Channel 0 Vector (Low)
	IF3	$FFF6	PLL Vector (High)
		$FFF7	PLL Vector (Low)
	IF2	$FFF8	LVI Vector (High)
		$FFF9	LVI Vector (Low)
	IF1	$FFFA	$\overline{\text{IRQ}}$ Vector (High)
		$FFFB	$\overline{\text{IRQ}}$ Vector (Low)
	—	$FFFC	SWI Vector (High)
		$FFFD	SWI Vector (Low)
Highest	—	$FFFE	Reset Vector (High)
		$FFFF	Reset Vector (Low)

9.6.1.5 Interrupt Status Register 2

Address: $FE05

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 9-13. Interrupt Status Register 2 (INT2)

IF14–IF7 — Interrupt Flags 14–7

These flags indicate the presence of interrupt requests from the sources shown in **Table 9-3**.

1 = Interrupt request present
0 = No interrupt request present

9.6.1.6 Interrupt Status Register 3

Address: $FE06

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	IF17	IF16	IF15
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 9-14. Interrupt Status Register 3 (INT3)

IF17–IF15 — Interrupt Flags 17–15

These flags indicate the presence of an interrupt request from the source shown in **Table 9-3**.

1 = Interrupt request present
0 = No interrupt request present

9.6.2 Reset

All reset sources always have equal and highest priority and cannot be arbitrated.

9.6.3 Break Interrupts

The break module can stop normal program flow at a software-programmable break point by asserting its break interrupt output. (See **Section 22. Break Module (BRK)**.) The SIM puts the CPU into the break state by forcing it to the SWI vector location. Refer to the break interrupt subsection of each module to see how each module is affected by the break state.

9.6.4 Status Flag Protection in Break Mode

The SIM controls whether status flags contained in other modules can be cleared during break mode. The user can select whether flags are protected from being cleared by properly initializing the break clear flag enable bit (BCFE) in the SIM break flag control register (SBFCR).

Protecting flags in break mode ensures that set flags will not be cleared while in break mode. This protection allows registers to be freely read and written during break mode without losing status flag information.

Setting the BCFE bit enables the clearing mechanisms. Once cleared in break mode, a flag remains cleared even when break mode is exited. Status flags with a 2-step clearing mechanism — for example, a read of one register followed by the read or write of another — are protected, even when the first step is accomplished prior to entering break mode. Upon leaving break mode, execution of the second step will clear the flag as normal.

9.7 Low-Power Modes

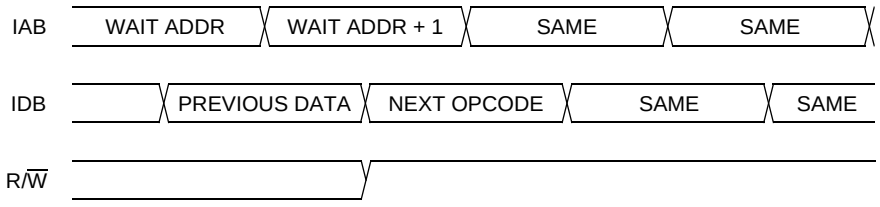
Executing the WAIT or STOP instruction puts the MCU in a low power-consumption mode for standby situations. The SIM holds the CPU in a non-clocked state. The operation of each of these modes is described in the following subsections. Both STOP and WAIT clear the interrupt mask (I) in the condition code register, allowing interrupts to occur.

9.7.1 Wait Mode

In wait mode, the CPU clocks are inactive while the peripheral clocks continue to run. **Figure 9-15** shows the timing for wait mode entry.

A module that is active during wait mode can wake up the CPU with an interrupt if the interrupt is enabled. Stacking for the interrupt begins one cycle after the WAIT instruction during which the interrupt occurred. In wait mode, the CPU clocks are inactive. Refer to the wait mode subsection of each module to see if the module is active or inactive in wait mode. Some modules can be programmed to be active in wait mode.

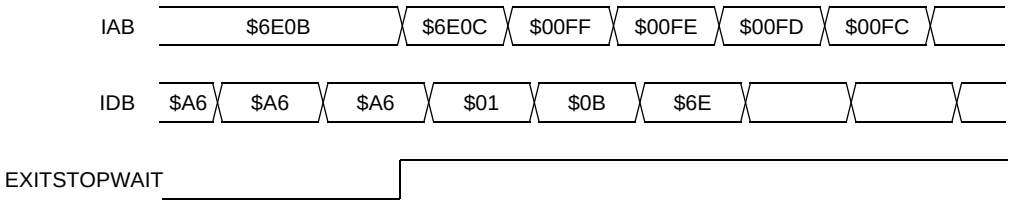
Wait mode also can be exited by a reset or break. A break interrupt during wait mode sets the SIM break stop/wait bit, SBSW, in the SIM break status register (SBSR). If the COP disable bit, COPD, in the mask option register is logic 0, then the computer operating properly module (COP) is enabled and remains active in wait mode.



NOTE: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

Figure 9-15. Wait Mode Entry Timing

Figure 9-16 and **Figure 9-17** show the timing for WAIT recovery.



NOTE: EXITSTOPWAIT = $\overline{\text{RST}}$ pin OR CPU interrupt OR break interrupt

Figure 9-16. Wait Recovery from Interrupt or Break

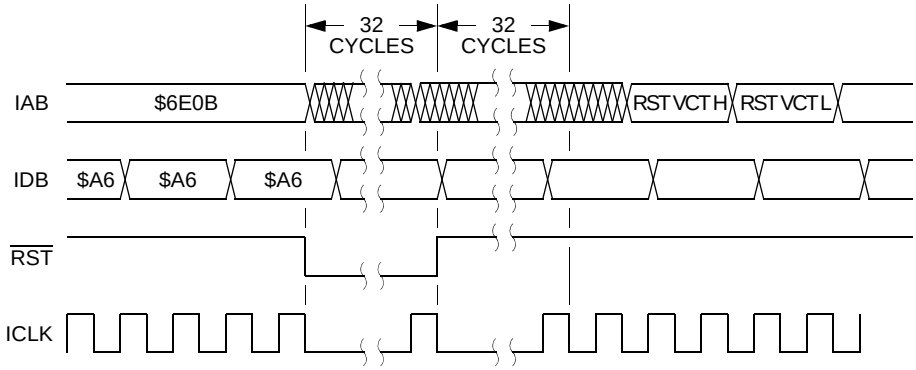


Figure 9-17. Wait Recovery from Internal Reset

9.7.2 Stop Mode

In stop mode, the SIM counter is reset and the system clocks are disabled. An interrupt request from a module can cause an exit from stop mode. Stacking for interrupts begins after the selected stop recovery time has elapsed. Reset or break also causes an exit from stop mode.

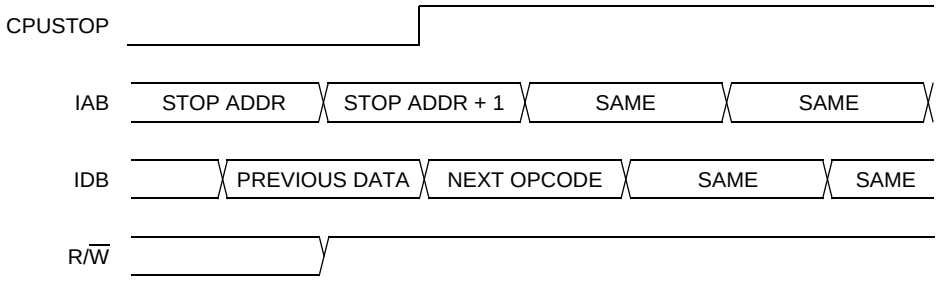
The SIM disables the clock generator module output (CGMOUT) in stop mode, stopping the CPU and peripherals. Stop recovery time is selectable using the SSREC bit in the configuration register 1 (CONFIG1). If SSREC is set, stop recovery is reduced from the normal delay of 4096 ICLK cycles down to 32. This is ideal for applications using canned oscillators that do not require long start-up times from stop mode.

NOTE: *External crystal applications should use the full stop recovery time by clearing the SSREC bit.*

A break interrupt during stop mode sets the SIM break stop/wait bit (SBSW) in the SIM break status register (SBSR).

The SIM counter is held in reset from the execution of the STOP instruction until the beginning of stop recovery. It is then used to time the recovery period. **Figure 9-18** shows stop mode entry timing.

NOTE: *To minimize stop current, all pins configured as inputs should be driven to a logic 1 or logic 0.*



NOTE: Previous data can be operand data or the STOP opcode, depending on the last instruction.

Figure 9-18. Stop Mode Entry Timing

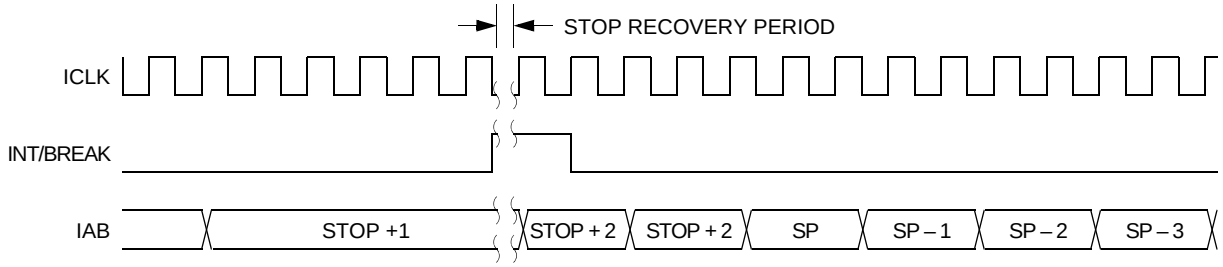


Figure 9-19. Stop Mode Recovery from Interrupt or Break

9.8 SIM Registers

The SIM has three memory-mapped registers:

- SIM Break Status Register (SBSR) — $FE00
- SIM Reset Status Register (SRSR) — $FE01
- SIM Break Flag Control Register (SBFCR) — $FE03

9.8.1 SIM Break Status Register

The SIM break status register (SBSR) contains a flag to indicate that a break caused an exit from stop mode or wait mode.

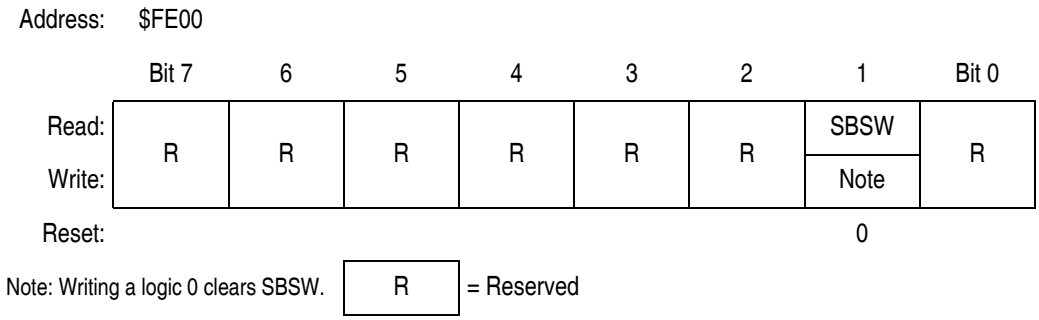


Figure 9-20. SIM Break Status Register (SBSR)

SBSW — Break Wait Bit

This status bit is set when a break interrupt causes an exit from wait mode or stop mode. Clear SBSW by writing a logic 0 to it. Reset clears SBSW.

1 = Stop mode or wait mode was exited by break interrupt
0 = Stop mode or wait mode was not exited by break interrupt

SBSW can be read within the break interrupt routine. The user can modify the return address on the stack by subtracting 1 from it. The following code is an example.

```
This code works if the H register has been pushed onto the stack in the break
service routine software. This code should be executed at the end of the break
service routine software.
HIBYTE   EQU
LOBYTE   EQU
         If not SBSW, do RTI
         BRCLR   SBSW,SBSR, RETURN    ;See if wait mode or stop mode was exited by
                                      ;break.
         TST     LOBYTE,SP            ;If RETURNLO is not zero,
         BNE     DOLO                 ;then just decrement low byte.
         DEC     HIBYTE,SP            ;Else deal with high byte, too.
DOLO     DEC     LOBYTE,SP            ;Point to WAIT/STOP opcode.
RETURN   PULH                         ;Restore H register.
         RTI
```

9.8.2 SIM Reset Status Register

This register contains six flags that show the source of the last reset provided all previous reset status bits have been cleared. Clear the SIM reset status register by reading it. A power-on reset sets the POR bit and clears all other bits in the register.

Address: $FE01

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	POR	PIN	COP	ILOP	ILAD	0	LVI	0
Write:								
Reset:	1	0	0	0	0	0	0	0

= Unimplemented

Figure 9-21. SIM Reset Status Register (SRSR)

POR — Power-On Reset Bit
1 = Last reset caused by POR circuit
0 = Read of SRSR

PIN — External Reset Bit
1 = Last reset caused by external reset pin ($\overline{\text{RST}}$)
0 = POR or read of SRSR

COP — Computer Operating Properly Reset Bit
1 = Last reset caused by COP counter
0 = POR or read of SRSR

ILOP — Illegal Opcode Reset Bit
1 = Last reset caused by an illegal opcode
0 = POR or read of SRSR

ILAD — Illegal Address Reset Bit (opcode fetches only)
1 = Last reset caused by an opcode fetch from an illegal address
0 = POR or read of SRSR

LVI — Low-Voltage Inhibit Reset Bit
1 = Last reset caused by the LVI circuit
0 = POR or read of SRSR

9.8.3 SIM Break Flag Control Register

The SIM break control register contains a bit that enables software to clear status bits while the MCU is in a break state.

Address: $FE03

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

R = Reserved

Figure 9-22. SIM Break Flag Control Register (SBFCR)

BCFE — Break Clear Flag Enable Bit

This read/write bit enables software to clear status bits by accessing status registers while the MCU is in a break state. To clear status bits during the break state, the BCFE bit must be set.

1 = Status bits clearable during break

0 = Status bits not clearable during break

Section 10. Monitor ROM (MON)

10.1 Contents

10.2 Introduction 156
10.3 Features 156
10.4 Functional Description 157
10.4.1 Entering Monitor Mode 159
10.4.2 Data Format 163
10.4.3 Break Signal 163
10.4.4 Baud Rate 163
10.4.5 Commands 164
10.5 Security 169
10.6 ROM-Resident Routines 171
10.6.1 PRGRNGE 173
10.6.2 ERARNGE 175
10.6.3 LDRNGE 176
10.6.4 MON_PRGRNGE 177
10.6.5 MON_ERARNGE 178
10.6.6 MON_LDRNGE 179
10.6.7 EE_WRITE 180
10.6.8 EE_READ 183

10.2 Introduction

This section describes the monitor ROM (MON) and the monitor mode entry methods. The monitor ROM allows complete testing of the MCU through a single-wire interface with a host computer. Monitor mode entry can be achieved without use of the higher test voltage, V_{TST}, as long as vector addresses $FFFE and $FFFF are blank, thus reducing the hardware requirements for in-circuit programming.
In addition, to simply user coding, routines are also stored in the monitor ROM area for FLASH memory program /erase and EEPROM emulation.

10.3 Features

Features of the monitor ROM include:

- Normal user-mode pin functionality
- One pin dedicated to serial communication between monitor ROM and host computer
- Standard mark/space non-return-to-zero (NRZ) communication with host computer
- Execution of code in RAM or FLASH
- FLASH memory security feature[1]
- FLASH memory programming interface
- Enhanced PLL (phase-locked loop) option to allow use of external 32.768-kHz crystal to generate internal frequency of 2.4576 MHz
- 960 bytes monitor ROM code size ($FC00–$FDFF and $FE10–$FFCE)
- Monitor mode entry without high voltage, V_{TST}, if reset vector is blank ($FFFE and $FFFF contain $FF)
- Standard monitor mode entry if high voltage, V_{TST}, is applied to $\overline{IRQ}$
- Resident routines for in-circuit programming and EEPROM emulation

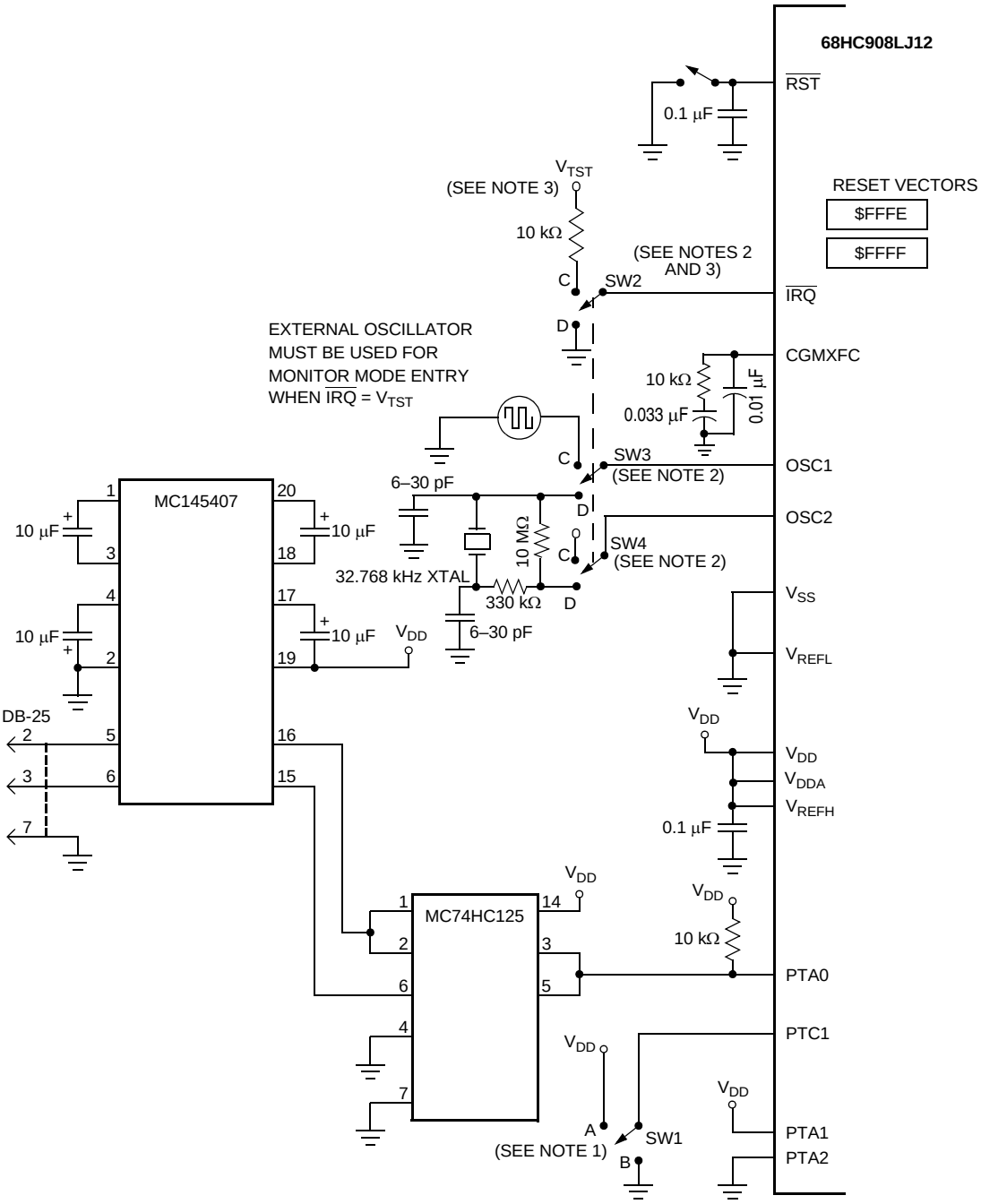
1. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

10.4 Functional Description

The monitor ROM receives and executes commands from a host computer. **Figure 10-1** shows an example circuit used to enter monitor mode and communicate with a host computer via a standard RS-232 interface.

Simple monitor commands can access any memory address. In monitor mode, the MCU can execute code downloaded into RAM by a host computer while most MCU pins retain normal operating mode functions. All communication between the host computer and the MCU is through the PTA0 pin. A level-shifting and multiplexing interface is required between PTA0 and the host computer. PTA0 is used in a wired-OR configuration and requires a pullup resistor.

The monitor code allows enabling the PLL to generate the internal clock, provided the reset vector is blank, when the device is being clocked by a low-frequency crystal. This entry method, which is enabled when $\overline{\text{IRQ}}$ is held low out of reset, is intended to support serial communication/ programming at 9600 baud in monitor mode by stepping up the external frequency (assumed to be 32.768 kHz) by a fixed amount to generate the desired internal frequency (2.4576 MHz). Since this feature is enabled only when $\overline{\text{IRQ}}$ is held low out of reset, it cannot be used when the reset vector is non-zero because entry into monitor mode in this case requires V_{TST} on $\overline{\text{IRQ}}$.



Notes:

1. For monitor mode entry when $\overline{IRQ}$ = V_{TST}:
   SW1: Position A — Bus clock = CGMXCLK ÷ 4 or CGMPCLK ÷ 4
   SW1: Position B — Bus clock = CGMXCLK ÷ 2
2. SW2, SW3, and SW4: Position C — Enter monitor mode using external oscillator.
   SW2, SW3, and SW4: Position D — Enter monitor mode using external XTAL and internal PLL.
3. See **23.6 5.0V DC Electrical Characteristics** for $\overline{IRQ}$ voltage level requirements.

Figure 10-1. Monitor Mode Circuit

10.4.1 Entering Monitor Mode

Table 10-1 shows the pin conditions for entering monitor mode. As specified in the table, monitor mode may be entered after a POR and will allow communication at 9600 baud provided one of the following sets of conditions is met:

1. If $FFFE and $FFFF do not contain $FF (programmed state):
   - The external clock is 4.9152 MHz with PTC1 low or 9.8304 MHz with PTC1 high
   - $\overline{IRQ} = V_{TST}$ (PLL off)
2. If $FFFE and $FFFF both contain $FF (erased state):
   - The external clock is 9.8304 MHz
   - $\overline{IRQ} = V_{DD}$ (this can be implemented through the internal $\overline{IRQ}$ pullup; PLL off)
3. If $FFFE and $FFFF both contain $FF (erased state):
   - The external clock is 32.768 kHz (crystal)
   - $\overline{IRQ} = V_{SS}$ (this setting initiates the PLL to boost the external 32.768 kHz to an internal bus frequency of 2.4576 MHz)

If V_{TST} is applied to $\overline{IRQ}$ and PTC1 is low upon monitor mode entry (above condition set 1), the bus frequency is a divide-by-two of the input clock. If PTC1 is high with V_{TST} applied to $\overline{IRQ}$ upon monitor mode entry, the bus frequency will be a divide-by-four of the input clock. Holding the PTC1 pin low when entering monitor mode causes a bypass of a divide-by-two stage at the oscillator only if V_{TST} is applied to $\overline{IRQ}$. In this event, the CGMOUT frequency is equal to the CGMXCLK frequency, and the OSC1 input directly generates internal bus clocks. In this case, the OSC1 signal must have a 50% duty cycle at maximum bus frequency.

If entering monitor mode without high voltage on $\overline{IRQ}$ (above condition set 2 or 3, where applied voltage is either V_{DD} or V_{SS}), then all port A pin requirements and conditions, including the PTC1 frequency divisor selection, are not in effect. This is to reduce circuit requirements when performing in-circuit programming.

Table 10-1. Monitor Mode Signal Requirements and Options

$\overline{IRQ}$	$\overline{RST}$	Address $FFFE/ $FFFF	PTA2	PTA1	PTA0[1]	PTC1	External Clock[2]	Bus Frequency	PLL	COP	Baud Rate	
X	GND	X	X	X	X	X	X	0	X	Disabled	0	No res
V_{TST}[3]	V_{DD} or V_{TST}	X	0	1	1	0	4.9152 MHz	2.4576 MHz	OFF	Disabled	9600	PT vo req $\overline{IR}$ PT fre
V_{TST}[3]	V_{DD} or V_{TST}	X	0	1	1	1	9.8304 MHz	2.4576 MHz	OFF	Disabled	9600	PT vo req $\overline{IR}$ PT fre
V_{DD}	V_{DD}	Blank "$FFFF"	X	X	1	X	9.8304 MHz	2.4576 MHz	OFF	Disabled	9600	Ex alw
GND	V_{DD}	Blank "$FFFF"	X	X	1	X	32.768 kHz	2.4576 MHz	ON	Disabled	9600	PL (B in
V_{DD} or GND	V_{TST}	Blank "$FFFF"	X	X	X	X	X	—	OFF	Enabled	—	En mo en ad
V_{DD} or GND	V_{DD} or V_{TST}	Not Blank	X	X	X	X	X	—	OFF	Enabled	—	En

Notes:
1. PTA0 = 1 if serial communication; PTA0 = 0 if parallel communication
2. External clock is derived by a 32.768 kHz crystal or a 4.9152/9.8304 MHz off-chip oscillator
3. Monitor mode entry by $\overline{IRQ}$ = V_{TST}, a 4.9152/9.8304 MHz off-chip oscillator must be used. The MCU internal crystal oscillator c

NOTE: *If the reset vector is blank and monitor mode is entered, the chip will see an additional reset cycle after the initial POR reset. Once the part has been programmed, the traditional method of applying a voltage, V_{TST}, to $\overline{IRQ}$ must be used to enter monitor mode.*

The COP module is disabled in monitor mode based on these conditions:

- If monitor mode was entered as a result of the reset vector being blank (above condition set 2 or 3), the COP is always disabled regardless of the state of $\overline{IRQ}$ or $\overline{RST}$.
- If monitor mode was entered with V_{TST} on $\overline{IRQ}$ (condition set 1), then the COP is disabled as long as V_{TST} is applied to either $\overline{IRQ}$ or $\overline{RST}$.

The second condition states that as long as V_{TST} is maintained on the $\overline{IRQ}$ pin after entering monitor mode, or if V_{TST} is applied to $\overline{RST}$ after the initial reset to get into monitor mode (when V_{TST} was applied to $\overline{IRQ}$), then the COP will be disabled. In the latter situation, after V_{TST} is applied to the $\overline{RST}$ pin, V_{TST} can be removed from the $\overline{IRQ}$ pin in the interest of freeing the $\overline{IRQ}$ for normal functionality in monitor mode.

Figure 10-2 shows a simplified diagram of the monitor mode entry when the reset vector is blank and just $1 \times V_{DD}$ voltage is applied to the $\overline{IRQ}$ pin. An external oscillator of 9.8304 MHz is required for a baud rate of 9600, as the internal bus frequency is automatically set to the external frequency divided by four.

Enter monitor mode with pin configuration shown in **Figure 10-1** by pulling $\overline{RST}$ low and then high. The rising edge of $\overline{RST}$ latches monitor mode. Once monitor mode is latched, the values on the specified pins can change.

Once out of reset, the MCU waits for the host to send eight security bytes. (See **10.5 Security**.) After the security bytes, the MCU sends a break signal (10 consecutive logic 0s) to the host, indicating that it is ready to receive a command.

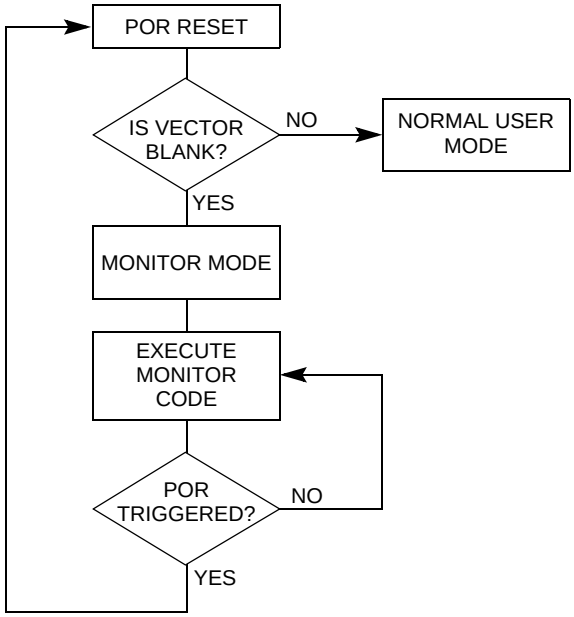


Figure 10-2. Low-Voltage Monitor Mode Entry Flowchart

In monitor mode, the MCU uses different vectors for reset, SWI (software interrupt), and break interrupt than those for user mode. The alternate vectors are in the $FE page instead of the $FF page and allow code execution from the internal monitor firmware instead of user code.

NOTE: *Exiting monitor mode after it has been initiated by having a blank reset vector requires a power-on reset (POR). Pulling $\overline{RST}$ low will not exit monitor mode in this situation.*

Table 10-2 summarizes the differences between user mode and monitor mode vectors.

Table 10-2. Mode Differences (Vectors)

Modes	Functions					
	Reset Vector High	**Reset Vector Low**	**Break Vector High**	**Break Vector Low**	**SWI Vector High**	**SWI Vector Low**
User	$FFFE	$FFFF	$FFFC	$FFFD	$FFFC	$FFFD
Monitor	$FEFE	$FEFF	$FEFC	$FEFD	$FEFC	$FEFD

10.4.2 Data Format

Communication with the monitor ROM is in standard non-return-to-zero (NRZ) mark/space data format. Transmit and receive baud rates must be identical.

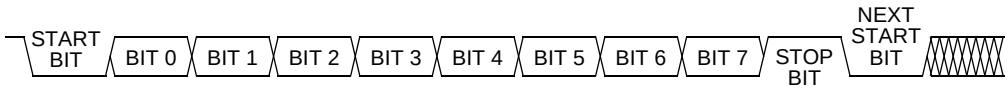


Figure 10-3. Monitor Data Format

10.4.3 Break Signal

A start bit (logic 0) followed by nine logic 0 bits is a break signal. When the monitor receives a break signal, it drives the PTA0 pin high for the duration of two bits and then echoes back the break signal.

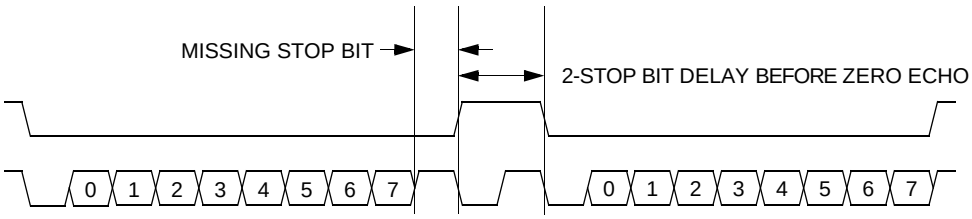


Figure 10-4. Break Transaction

10.4.4 Baud Rate

The communication baud rate is controlled by the crystal frequency and the state of the PTC1 pin (when $\overline{IRQ}$ is set to V_{TST}) upon entry into monitor mode. When PTC1 is high, the divide by ratio is 1024. If the PTC1 pin is at logic 0 upon entry into monitor mode, the divide by ratio is 512.

If monitor mode was entered with V_{DD} on $\overline{IRQ}$, then the divide by ratio is set at 1024, regardless of PTC1. If monitor mode was entered with V_{SS} on $\overline{IRQ}$, then the internal PLL steps up the external frequency, presumed to be 32.768 kHz, to 2.4576 MHz. These latter two conditions for monitor mode entry require that the reset vector is blank.

Table 10-3 lists external frequencies required to achieve a standard baud rate of 9600 BPS. Other standard baud rates can be accomplished using proportionally higher or lower frequency generators. If using a crystal as the clock source, be aware of the upper frequency limit that the internal clock module can handle. See Section 23. Electrical Specifications for this limit.

Table 10-3. Monitor Baud Rate Selection

External Frequency	$\overline{IRQ}$	PTC1	Internal Frequency	Baud Rate (BPS)
4.9152 MHz	V_{TST}	0	2.4576 MHz	9600
9.8304 MHz	V_{TST}	1	2.4576 MHz	9600
9.8304 MHz	V_{DD}	X	2.4576 MHz	9600
32.768 kHz	V_{SS}	X	2.4576 MHz	9600

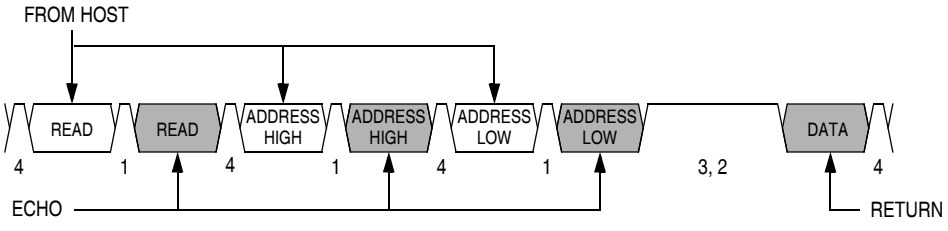
10.4.5 Commands

The monitor ROM firmware uses these commands:

- READ (read memory)
- WRITE (write memory)
- IREAD (indexed read)
- IWRITE (indexed write)
- READSP (read stack pointer)
- RUN (run user program)

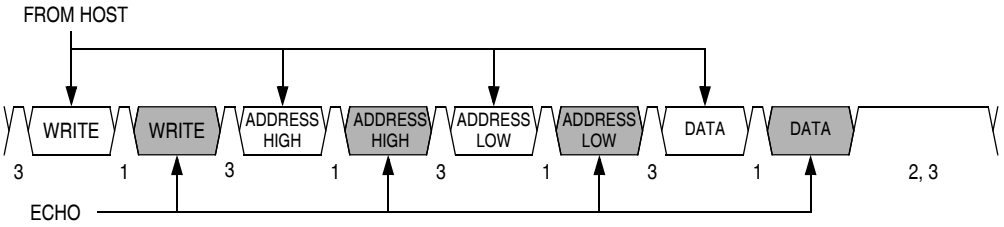
The monitor ROM firmware echoes each received byte back to the PTA0 pin for error checking. An 11-bit delay at the end of each command allows the host to send a break character to cancel the command. A delay of two bit times occurs before each echo and before READ, IREAD, or READSP data is returned. The data returned by a read command appears after the echo of the last byte of the command.

NOTE: *Wait one bit time after each echo before sending the next byte.*



Notes:
1 = Echo delay, 2 bit times
2 = Data return delay, 2 bit times
3 = Cancel command delay, 11 bit times
4 = Wait 1 bit time before sending next byte.

Figure 10-5. Read Transaction



Notes:
1 = Echo delay, 2 bit times
2 = Cancel command delay, 11 bit times
3 = Wait 1 bit time before sending next byte.

Figure 10-6. Write Transaction

A brief description of each monitor mode command is given in **Table 10-4** through **Table 10-9**.

Table 10-4. READ (Read Memory) Command

Description	Read byte from memory
Operand	2-byte address in high-byte:low-byte order
Data Returned	Returns contents of specified address
Opcode	$4A
Command Sequence	

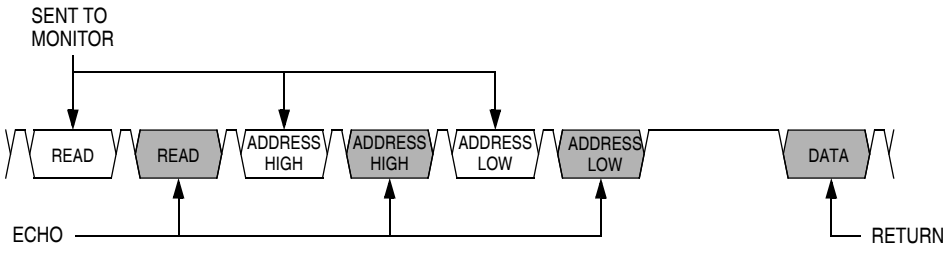


Table 10-5. WRITE (Write Memory) Command

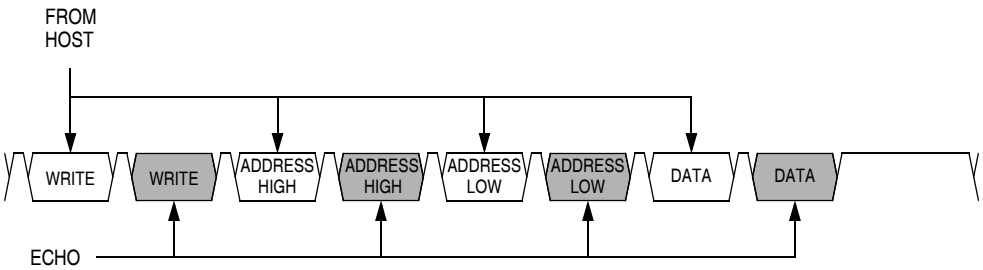
Description	Write byte to memory
Operand	2-byte address in high-byte:low-byte order; low byte followed by data byte
Data Returned	None
Opcode	$49
Command Sequence	

Table 10-6. IREAD (Indexed Read) Command

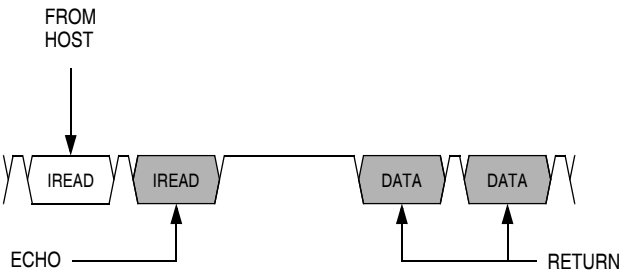
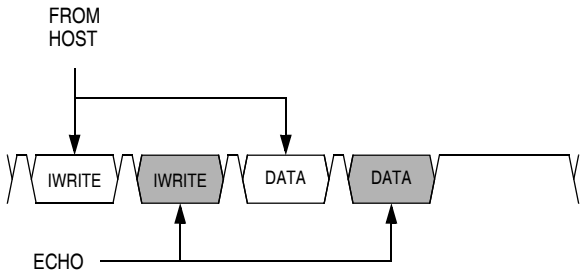
Description	Read next 2 bytes in memory from last address accessed
Operand	2-byte address in high byte:low byte order
Data Returned	Returns contents of next two addresses
Opcode	$1A
Command Sequence	

Table 10-7. IWRITE (Indexed Write) Command

Description	Write to last address accessed + 1
Operand	Single data byte
Data Returned	None
Opcode	$19
Command Sequence	

A sequence of IREAD or IWRITE commands can access a block of memory sequentially over the full 64k-byte memory map.

Table 10-8. READSP (Read Stack Pointer) Command

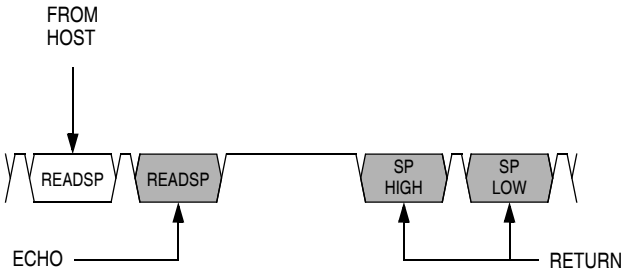
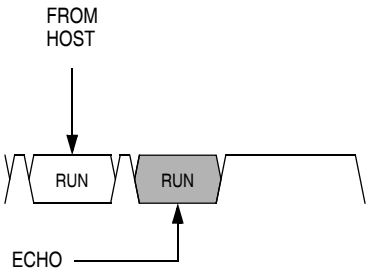
Description	Reads stack pointer
Operand	None
Data Returned	Returns incremented stack pointer value (SP + 1) in high-byte:low-byte order
Opcode	$0C
Command Sequence	

Table 10-9. RUN (Run User Program) Command

Description	Executes PULH and RTI instructions
Operand	None
Data Returned	None
Opcode	$28
Command Sequence	

The MCU executes the SWI and PSHH instructions when it enters monitor mode. The RUN command tells the MCU to execute the PULH and RTI instructions. Before sending the RUN command, the host can modify the stacked CPU registers to prepare to run the host program. The READSP command returns the incremented stack pointer value, SP + 1. The high and low bytes of the program counter are at addresses SP + 5 and SP + 6.

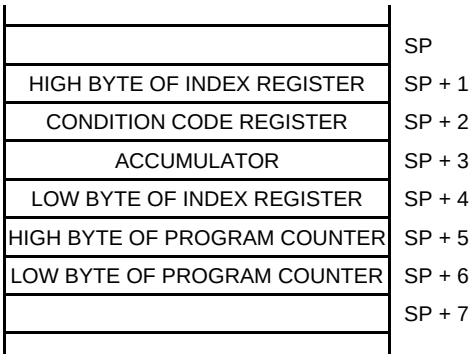


Figure 10-7. Stack Pointer at Monitor Mode Entry

10.5 Security

A security feature discourages unauthorized reading of FLASH locations while in monitor mode. The host can bypass the security feature at monitor mode entry by sending eight security bytes that match the bytes at locations $FFF6–$FFFD. Locations $FFF6–$FFFD contain user-defined data.

NOTE: *Do not leave locations $FFF6–$FFFD blank. For security reasons, program locations $FFF6–$FFFD even if they are not used for vectors.*

During monitor mode entry, the MCU waits after the power-on reset for the host to send the eight security bytes on pin PTA0. If the received bytes match those at locations $FFF6–$FFFD, the host bypasses the security feature and can read all FLASH locations and execute code from FLASH. Security remains bypassed until a power-on reset occurs. If the reset was not a power-on reset, security remains bypassed and security code entry is not required. (See **Figure 10-8**.)

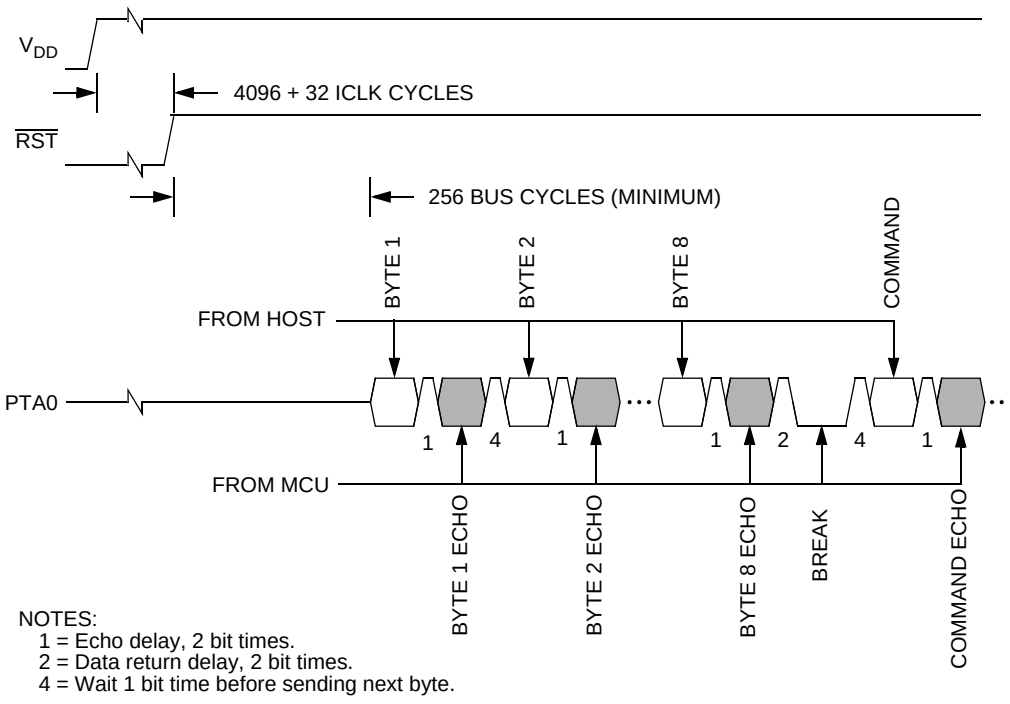


Figure 10-8. Monitor Mode Entry Timing

Upon power-on reset, if the received bytes of the security code do not match the data at locations $FFF6–$FFFD, the host fails to bypass the security feature. The MCU remains in monitor mode, but reading a FLASH location returns an invalid value and trying to execute code from FLASH causes an illegal address reset. After receiving the eight security bytes from the host, the MCU transmits a break character, signifying that it is ready to receive a command.

NOTE: *The MCU does not transmit a break character until after the host sends the eight security bits.*

To determine whether the security code entered is correct, check to see if bit 6 of RAM address $40 is set. If it is, then the correct security code has been entered and FLASH can be accessed.

If the security sequence fails, the device should be reset by a power-on reset and brought up in monitor mode to attempt another entry. After failing the security sequence, the FLASH module can also be mass erased by executing an erase routine that was downloaded into internal RAM. The mass erase operation clears the security code locations so that all eight security bytes become $FF (blank).

10.6 ROM-Resident Routines

Eight routines stored in the monitor ROM area (thus ROM-resident) are provided for FLASH memory manipulation. Six of the eight routines are intended to simply FLASH program, erase, and load operations. The other two routines are intended to simply the use of the FLASH memory as EEPROM. **Table 10-10** shows a summary of the ROM-resident routines.

Table 10-10. Summary of ROM-Resident Routines

Routine Name	Routine Description	Call Address	Stack Used (bytes)
PRGRNGE	Program a range of locations	$FC06	14
ERARNGE	Erase a page or the entire array	$FCBE	9
LDRNGE	Loads data from a range of locations	$FF30	9
MON_PRGRNGE	Program a range of locations in monitor mode	$FF28	16
MON_ERARNGE	Erase a page or the entire array in monitor mode	$FF2C	11
MON_LDRNGE	Loads data from a range of locations in monitor mode	$FF24	11
EE_WRITE	Emulated EEPROM write. Data size ranges from 2 to 15 bytes at a time.	$FC00	17
EE_READ	Emulated EEPROM read. Data size ranges from 2 to 15 bytes at a time.	$FC03	15

The routines are designed to be called as stand-alone subroutines in the user program or monitor mode. The parameters that are passed to a routine are in the form of a contiguous data block, stored in RAM. The index register (H:X) is loaded with the address of the first byte of the data block (acting as a pointer), and the subroutine is called (JSR). Using the start address as a pointer, multiple data blocks can be used, any area of RAM be used. A data block has the control and data bytes in a defined order, as shown in **Figure 10-9**.

During the software execution, it does not consume any dedicated RAM location, the run-time heap will extend the system stack, all other RAM location will not be affected.

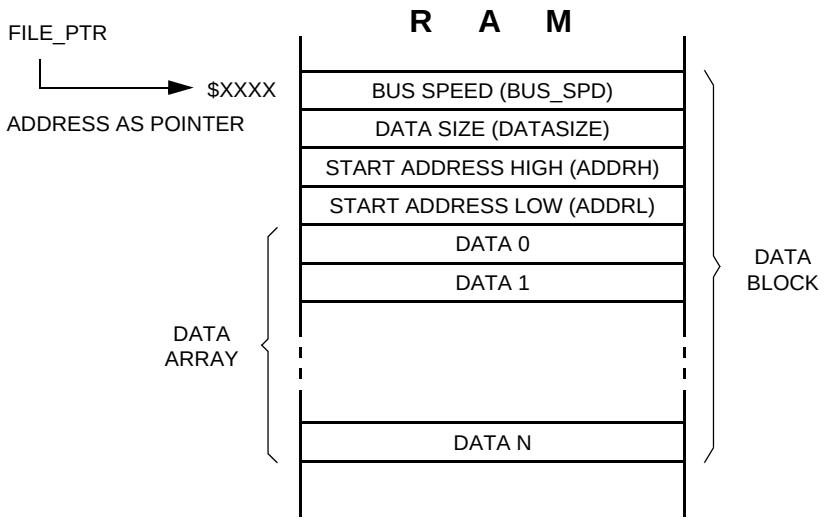


Figure 10-9. Data Block Format for ROM-Resident Routines

The control and data bytes are described below.

- **Bus speed** — This one byte indicates the operating bus speed of the MCU. The value of this byte should be equal to 4 times the bus speed. E.g., for a 4 MHz bus, the value is 16 ($10). This control byte is useful where the MCU clock source is switched between the PLL clock and the crystal clock.
- **Data size** — This one byte indicates the number of bytes in the data array that are to be manipulated. The maximum data array size is 255. Routines EE_WRITE and EE_READ are restricted to manipulate a data array between 2 to 15 bytes. Whereas routines ERARNGE and MON_ERARNGE do not manipulate a data array, thus, this data size byte has no meaning.
- **Start address** — These two bytes, high byte followed by low byte, indicate the start address of the FLASH memory to be manipulated.
- **Data array** — This data array contains data that are to be manipulated. Data in this array are programmed to FLASH memory by the programming routines: PRGRNGE, MON_PRGRNGE, EE_WRITE. For the read routines: LDRNGE, MON_LDRNGE, and EE_READ, data is read from FLASH and stored in this array.

10.6.1 PRGRNGE

PRGRNGE is used to program a range of FLASH locations with data loaded into the data array.

Table 10-11. PRGRNGE Routine

Routine Name	PRGRNGE
Routine Description	Program a range of locations
Calling Address	$FC06
Stack Used	14 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Start address high (ADDRH) Start address (ADDRL) Data 1 (DATA1) : Data N (DATAN)

The start location of the FLASH to be programmed is specified by the address ADDRH:ADDRL and the number of bytes from this location is specified by DATASIZE. The maximum number of bytes that can be programmed in one routine call is 255 bytes (max. DATASIZE is 255).

ADDRH:ADDRL do not need to be at a page boundary, the routine handles any boundary misalignment during programming. A check to see that all bytes in the specified range are erased is not performed by this routine prior programming. Nor does this routine do a verification after programming, so there is no return confirmation that programming was successful. User must assure that the range specified is first erased.

The coding example below is to program 64 bytes of data starting at FLASH location $EF00, with a bus speed of 4.9152 MHz. The coding assumes the data block is already loaded in RAM, with the address pointer, FILE_PTR, pointing to the first byte of the data block.

```
                ORG     RAM
        :
FILE_PTR:
BUS_SPD         DS.B    1       ; Indicates 4x bus frequency
DATASIZE        DS.B    1       ; Data size to be programmed
START_ADDR      DS.W    1       ; FLASH start address
DATAARRAY       DS.B    64      ; Reserved data array

PRGRNGE         EQU     $FC06
FLASH_START     EQU     $EF00

                ORG     FLASH
INITIALISATION:
        MOV     #20,    BUS_SPD
        MOV     #64,    DATASIZE
        LDHX    #FLASH_START
        STHX    START_ADDR
        RTS
MAIN:
        BSR     INITIALISATION
        :
        :
        LDHX    FILE_PTR
        JSR     PRGRNGE
```

10.6.2 ERARNGE

ERARNGE is used to erase a range of locations in FLASH.

Table 10-12. ERARNGE Routine

Routine Name	ERARNGE
Routine Description	Erase a page or the entire array
Calling Address	$FCBE
Stack Used	9 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Starting address (ADDRH) Starting address (ADDRL)

There are two sizes of erase ranges: a page or the entire array. The ERARNGE will erase the page (128 consecutive bytes) in FLASH specified by the address ADDRH:ADDRL. This address can be any address within the page. Calling ERARNGE with ADDRH:ADDRL equal to $FFFF will erase the entire FLASH array (mass erase). Therefore, care must be taken when calling this routine to prevent an accidental mass erase.

The ERARNGE routine do not use a data array. The DATASIZE byte is a dummy byte that is also not used.

The coding example below is to perform a page erase, from $EF00–$EF7F. The Initialization subroutine is the same as the coding example for PRGRNGE (see **10.6.1 PRGRNGE**).

```
ERARNGE         EQU     $FCBE
MAIN:
        BSR     INITIALISATION
        :
        :
        LDHX    FILE_PTR
        JSR     ERARNGE
        :
```

10.6.3 LDRNGE

LDRNGE is used to load the data array in RAM with data from a range of FLASH locations.

Table 10-13. LDRNGE Routine

Routine Name	LDRNGE
Routine Description	Loads data from a range of locations
Calling Address	$FF30
Stack Used	9 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Starting address (ADDRH) Starting address (ADDRL) Data 1 : Data N

The start location of FLASH from where data is retrieved is specified by the address ADDRH:ADDRL and the number of bytes from this location is specified by DATASIZE. The maximum number of bytes that can be retrieved in one routine call is 255 bytes. The data retrieved from FLASH is loaded into the data array in RAM. Previous data in the data array will be overwritten. User can use this routine to retrieve data from FLASH that was previously programmed.

The coding example below is to retrieve 64 bytes of data starting from $EF00 in FLASH. The Initialization subroutine is the same as the coding example for PRGRNGE (see **10.6.1 PRGRNGE**).

```
LDRNGE          EQU     $FF30
MAIN:
        BSR     INITIALIZATION
        :
        :
        LDHX    FILE_PTR
        JSR     LDRNGE
        :
```

10.6.4 MON_PRGRNGE

In monitor mode, MON_PRGRNGE is used to program a range of FLASH locations with data loaded into the data array.

Table 10-14. MON_PRGRNGE Routine

Routine Name	MON_PRGRNGE
Routine Description	Program a range of locations, in monitor mode
Calling Address	$FC28
Stack Used	16 bytes
Data Block Format	Bus speed Data size Starting address (high byte) Starting address (low byte) Data 1 : Data N

The MON_PRGRNGE routine is designed to be used in monitor mode. It performs the same function as the PRGRNGE routine (see **10.6.1 PRGRNGE**), except that MON_PRGRNGE returns to the main program via an SWI instruction. After a MON_PRGRNGE call, the SWI instruction will return the control back to the monitor code.

10.6.5 MON_ERARNGE

In monitor mode, ERARNGE is used to erase a range of locations in FLASH.

Table 10-15. MON_ERARNGE Routine

Routine Name	MON_ERARNGE
Routine Description	Erase a page or the entire array, in monitor mode
Calling Address	$FF2C
Stack Used	11 bytes
Data Block Format	Bus speed Data size Starting address (high byte) Starting address (low byte)

The MON_ERARNGE routine is designed to be used in monitor mode. It performs the same function as the ERARNGE routine (see **10.6.2 ERARNGE**), except that MON_ERARNGE returns to the main program via an SWI instruction. After a MON_ERARNGE call, the SWI instruction will return the control back to the monitor code.

10.6.6 MON_LDRNGE

In monitor mode, LDRNGE is used to load the data array in RAM with data from a range of FLASH locations.

Table 10-16. ICP_LDRNGE Routine

Routine Name	MON_LDRNGE
Routine Description	Loads data from a range of locations, in monitor mode
Calling Address	$FF24
Stack Used	11 bytes
Data Block Format	Bus speed Data size Starting address (high byte) Starting address (low byte) Data 1 : Data N

The MON_LDRNGE routine is designed to be used in monitor mode. It performs the same function as the LDRNGE routine (see **10.6.3 LDRNGE**), except that MON_LDRNGE returns to the main program via an SWI instruction. After a MON_LDRNGE call, the SWI instruction will return the control back to the monitor code.

10.6.7 EE_WRITE

EE_WRITE is used to write a set of data from the data array to FLASH.

Table 10-17. EE_WRITE Routine

Routine Name	EE_WRITE
Routine Description	Emulated EEPROM write. Data size ranges from 2 to 15 bytes at a time.
Calling Address	$FC00
Stack Used	17 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE)(1) Starting address (ADDRH)(2) Starting address (ADDRL)(1) Data 1 : Data N

Notes:
1. The minimum data size is 2 bytes. The maximum data size is 15 bytes.
2. The start address must be a page boundary start address, e.g. $xx00 or $xx80.

The start location of the FLASH to be programmed is specified by the address ADDRH:ADDRL and the number of bytes in the data array is specified by DATASIZE. The minimum number of bytes that can be programmed in one routine call is 2 bytes, the maximum is 15 bytes. ADDRH:ADDRL must always be the start of boundary address (the page start address: $XX00 or $0080) and DATASIZE must be the same size when accessing the same page.

In some applications, the user may want to repeatedly store and read a set of data from an area of non-volatile memory. This is easily possible when using an EEPROM array. As the write and erase operations can be executed on a byte basis. For FLASH memory, the minimum erase size is the page — 128 bytes per page for MC68HC908LJ12. If the data array size is less than the page size, writing and erasing to the same page cannot fully utilize the page. Unused locations in the page will be wasted. The EE_WRITE routine is designed to emulate the properties similar to the EEPROM. Allowing a more efficient use of the FLASH page for data storage.

When the user dedicates a page of FLASH for data storage, and the size of the data array defined, each call of the EE_WRTIE routine will automatically transfer the data in the data array (in RAM) to the next blank block of locations in the FLASH page. Once a page is filled up, the EE_WRITE routine automatically erases the page, and starts reuse the page again. In the 128-byte page, an 8-byte control block is used by the routine to monitor the utilization of the page. In effect, only 120 bytes are used for data storage. (see **Figure 10-10**). The page control operations are transparent to the user.

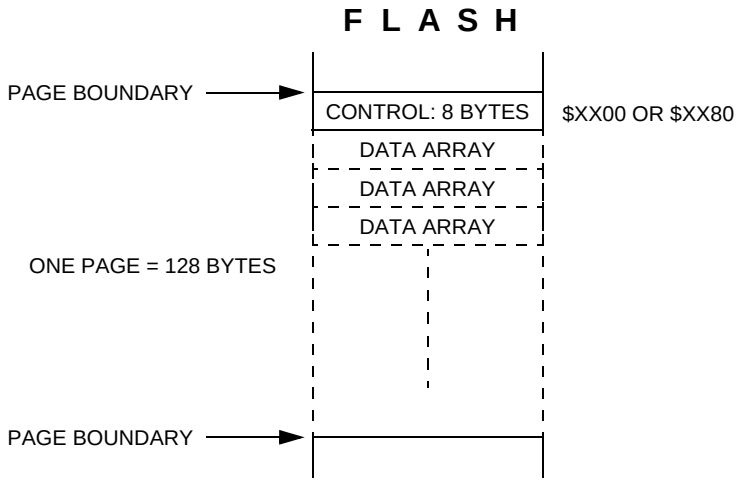


Figure 10-10. EE_WRITE FLASH Memory Usage

When using this routine to store a 2-byte data array, the FLASH page can be programmed 60 times before the an erase is required. In effect, the write/erase endurance is increased by 60 times. When a 15-byte data array is used, the write/erase endurance is increased by 8 times. Due to the FLASH page size limitation, the data array is limited from 2 bytes to 15 bytes.

The coding example below uses the $EF00–$EE7F page for data storage. The data array size is 15 bytes, and the bus speed is 4.9152 MHz. The coding assumes the data block is already loaded in RAM, with the address pointer, FILE_PTR, pointing to the first byte of the data block.

```
                ORG     RAM
        :
FILE_PTR:
BUS_SPD         DS.B    1       ; Indicates 4x bus frequency
DATASIZE        DS.B    1       ; Data size to be programmed
START_ADDR      DS.W    1       ; FLASH starting address
DATAARRAY       DS.B    15      ; Reserved data array

EE_WRITE        EQU     $FC00
FLASH_START     EQU     $EF00

                ORG     FLASH
INITIALISATION:
        MOV     #20,    BUS_SPD
        MOV     #15,    DATASIZE
        LDHX    #FLASH_START
        STHX    START_ADDR
        RTS
MAIN:
        BSR     INITIALISATION
        :
        :
        LHDX    FILE_PTR
        JSR     EE_WRITE
```

NOTE: *The EE_WRITE routine is unable to check for incorrect data blocks, such as the FLASH page boundary address and data size. It is the responsibility of the user to ensure the starting address indicated in the data block is at the FLASH page boundary and the data size is 2 to 15. If the FLASH page is already programmed with a data array with a different size, the EE_WRITE call will be ignored.*

10.6.8 EE_READ

EE_READ is used to load the data array in RAM with a set of data from FLASH.

Table 10-18. EE_READ Routine

Routine Name	EE_READ
Routine Description	Emulated EEPROM read. Data size ranges from 2 to 15 bytes at a time.
Calling Address	$FC03
Stack Used	15 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Starting address (ADDRH)(1) Starting address (ADDRL)(1) Data 1 : Data N

Notes:
1. The start address must be a page boundary start address, e.g. $xx00 or $xx80.

The EE_READ routine reads data stored by the EE_WRITE routine. An EE_READ call will retrieve the last data written to a FLASH page and loaded into the data array in RAM. Same as EE_WRITE, the data size indicated by DATASIZE is 2 to 15, and the start address ADDRH:ADDRL must the FLASH page boundary address.

The coding example below uses the data stored by the EE_WRITE coding example (see **10.6.7 EE_WRITE**). It loads the 15-byte data set stored in the $EF00–$EE7F page to the data array in RAM. The initialization subroutine is the same as the coding example for EE_WRITE (see **10.6.7 EE_WRITE**).

```
EE_READ         EQU     $FC03

MAIN:
        BSR     INITIALIZATION
        :
        :
        LDHX    FILE_PTR
        JSR     EE_READ
        :
```

NOTE: *The EE_READ routine is unable to check for incorrect data blocks, such as the FLASH page boundary address and data size. It is the responsibility of the user to ensure the starting address indicated in the data block is at the FLASH page boundary and the data size is 2 to 15. If the FLASH page is programmed with a data array with a different size, the EE_READ call will be ignored.*

Section 11. Timer Interface Module (TIM)

11.1 Contents

11.2 Introduction 186

11.3 Features 186

11.4 Pin Name Conventions 187

11.5 Functional Description 187
11.5.1 TIM Counter Prescaler 191
11.5.2 Input Capture 191
11.5.3 Output Compare 192
11.5.3.1 Unbuffered Output Compare 192
11.5.3.2 Buffered Output Compare 193
11.5.4 Pulse Width Modulation (PWM) 193
11.5.4.1 Unbuffered PWM Signal Generation 194
11.5.4.2 Buffered PWM Signal Generation 195
11.5.4.3 PWM Initialization 196

11.6 Interrupts 197

11.7 Low-Power Modes 197
11.7.1 Wait Mode 198
11.7.2 Stop Mode 198

11.8 TIM During Break Interrupts 198

11.9 I/O Signals 199

11.10 I/O Registers 199
11.10.1 TIM Status and Control Register 200
11.10.2 TIM Counter Registers 202
11.10.3 TIM Counter Modulo Registers 203
11.10.4 TIM Channel Status and Control Registers 204
11.10.5 TIM Channel Registers 207

11.2 Introduction

This section describes the timer interface (TIM) module. The TIM is a two-channel timer that provides a timing reference with input capture, output compare, and pulse-width-modulation functions. **Figure 11-1** is a block diagram of the TIM.

This particular MCU has two timer interface modules which are denoted as TIM1 and TIM2.

11.3 Features

Features of the TIM include:

- Two input capture/output compare channels:
  - Rising-edge, falling-edge, or any-edge input capture trigger
  - Set, clear, or toggle output compare action
- Buffered and unbuffered pulse-width-modulation (PWM) signal generation
- Programmable TIM clock input with 7-frequency internal bus clock prescaler selection
- Free-running or modulo up-count operation
- Toggle any channel pin on overflow
- TIM counter stop and reset bits

11.4 Pin Name Conventions

The text that follows describes both timers, TIM1 and TIM2. The TIM input/output (I/O) pin names are T[1,2]CH0 (timer channel 0) and T[1,2]CH1 (timer channel 1), where “1” is used to indicate TIM1 and “2” is used to indicate TIM2. The two TIMs share four I/O pins with four I/O port pins. The full names of the TIM I/O pins are listed in **Table 11-1**. The generic pin names appear in the text that follows.

Table 11-1. Pin Name Conventions

TIM Generic Pin Names:		T[1,2]CH0	T[1,2]CH1
Full TIM Pin Names:	TIM1	PTB2/T1CH0	PTB3/T1CH1
	TIM2	PTB4/T2CH0	PTB5/T2CH1

NOTE: *References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TCH0 may refer generically to T1CH0 and T2CH0, and TCH1 may refer to T1CH1 and T2CH1.*

11.5 Functional Description

Figure 11-1 shows the structure of the TIM. The central component of the TIM is the 16-bit TIM counter that can operate as a free-running counter or a modulo up-counter. The TIM counter provides the timing reference for the input capture and output compare functions. The TIM counter modulo registers, TMODH:TMODL, control the modulo value of the TIM counter. Software can read the TIM counter value at any time without affecting the counting sequence.

The two TIM channels (per timer) are programmable independently as input capture or output compare channels.

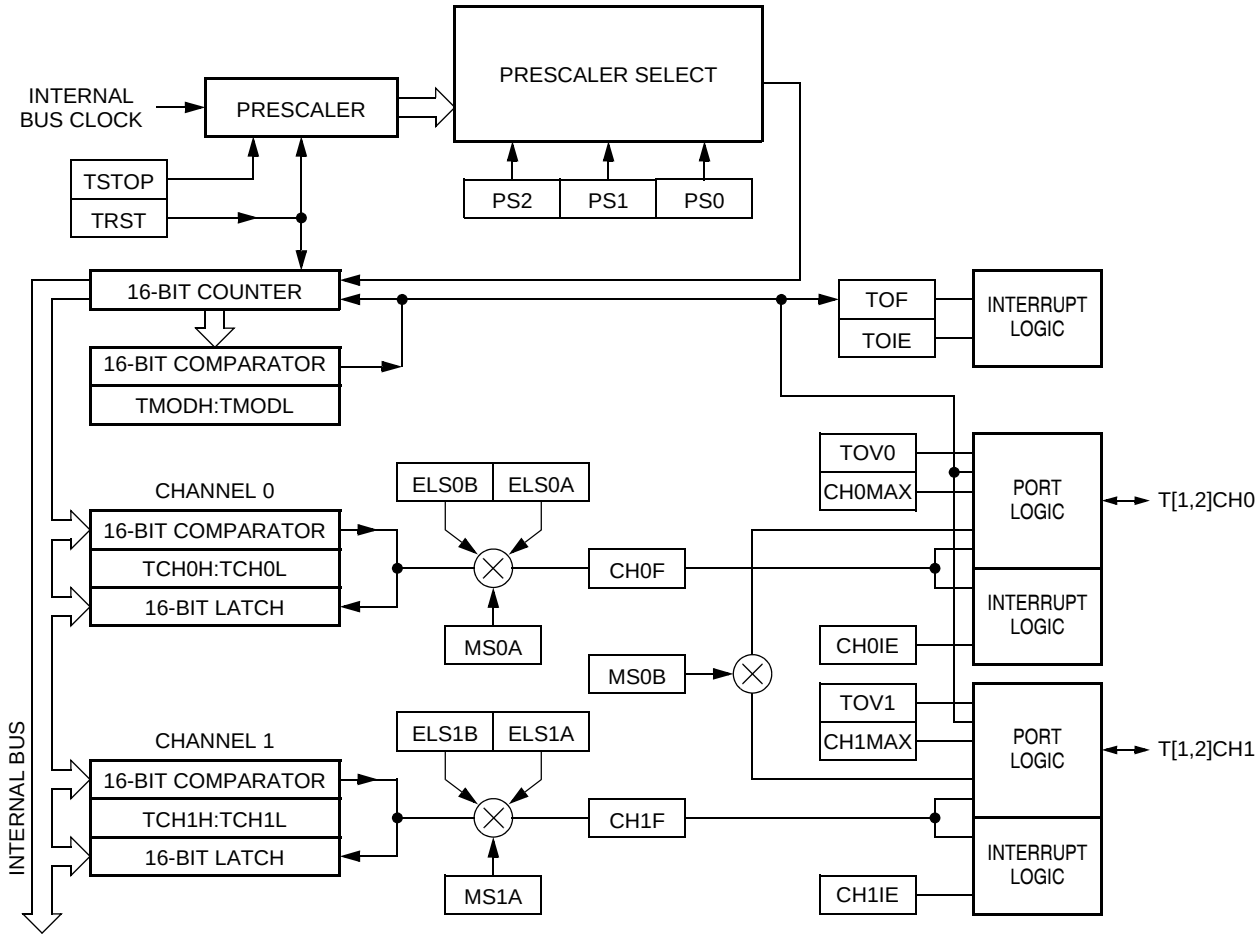


Figure 11-1. TIM Block Diagram

Figure 11-2 summarizes the timer registers.

NOTE: *References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TSC may generically refer to both T1SC and T2SC.*

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0020	Timer 1 Status and Control Register (T1SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0021	Timer 1 Counter Register High (T1CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0022	Timer 1 Counter Register Low (T1CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0023	Timer 1 Counter Modulo Register High (T1MODH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0024	Timer 1 Counter Modulo Register Low (T1MODL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0025	Timer 1 Channel 0 Status and Control Register (T1SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0026	Timer 1 Channel 0 Register High (T1CH0H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0027	Timer 1 Channel 0 Register Low (T1CH0L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0028	Timer 1 Channel 1 Status and Control Register (T1SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 11-2. TIM I/O Register Summary (Sheet 1 of 3)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0029	Timer 1 Channel 1 Register High (T1CH1H)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$002A	Timer 1 Channel 1 Register Low (T1CH1L)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$002B	Timer 2 Status and Control Register (T2SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$002C	Timer 2 Counter Register High (T2CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002D	Timer 2 Counter Register Low (T2CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002E	Timer 2 Counter Modulo Register High (T2MODH)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	1	1	1	1	1	1	1	1
$002F	Timer 2 Counter Modulo Register Low (T2MODL)	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$0030	Timer 2 Channel 0 Status and Control Register (T2SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0031	Timer 2 Channel 0 Register High (T2CH0H)	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							

= Unimplemented

Figure 11-2. TIM I/O Register Summary (Sheet 2 of 3)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0032	Timer 2 Channel 0 Register Low (T2CH0L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0033	Timer 2 Channel 1 Status and Control Register (T2SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0034	Timer 2 Channel 1 Register High (T2CH1H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0035	Timer 2 Channel 1 Register Low (T2CH1L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							

= Unimplemented

Figure 11-2. TIM I/O Register Summary (Sheet 3 of 3)

11.5.1 TIM Counter Prescaler

The TIM clock source can be one of the seven prescaler outputs. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIM status and control register select the TIM clock source.

11.5.2 Input Capture

With the input capture function, the TIM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TIM latches the contents of the TIM counter into the TIM channel registers, TCHxH:TCHxL. The polarity of the active edge is programmable. Input captures can generate TIM CPU interrupt requests.

11.5.3 Output Compare

With the output compare function, the TIM can generate a periodic pulse with a programmable polarity, duration, and frequency. When the counter reaches the value in the registers of an output compare channel, the TIM can set, clear, or toggle the channel pin. Output compares can generate TIM CPU interrupt requests.

11.5.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in **11.5.3 Output Compare**. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIM channel registers.

An unsynchronized write to the TIM channel registers to change an output compare value could cause incorrect operation for up to two counter overflow periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that counter overflow period. Also, using a TIM overflow interrupt routine to write a new, smaller output compare value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the output compare value on channel x:

- When changing to a smaller value, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current output compare pulse. The interrupt routine has until the end of the counter overflow period to write the new value.
- When changing to a larger output compare value, enable TIM overflow interrupts and write the new value in the TIM overflow interrupt routine. The TIM overflow interrupt occurs at the end of the current counter overflow period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same counter overflow period.

11.5.3.2 Buffered Output Compare

Channels 0 and 1 can be linked to form a buffered output compare channel whose output appears on the TCH0 pin. The TIM channel registers of the linked pair alternately control the output.

Setting the MS0B bit in TIM channel 0 status and control register (TSC0) links channel 0 and channel 1. The output compare value in the TIM channel 0 registers initially controls the output on the TCH0 pin. Writing to the TIM channel 1 registers enables the TIM channel 1 registers to synchronously control the output after the TIM overflows. At each subsequent overflow, the TIM channel registers (0 or 1) that control the output are the ones written to last. TSC0 controls and monitors the buffered output compare function, and TIM channel 1 status and control register (TSC1) is unused. While the MS0B bit is set, the channel 1 pin, TCH1, is available as a general-purpose I/O pin.

NOTE: *In buffered output compare operation, do not write new output compare values to the currently active channel registers. User software should track the currently active channel to prevent writing a new value to the active channel. Writing to the active channel registers is the same as generating unbuffered output compares.*

11.5.4 Pulse Width Modulation (PWM)

By using the toggle-on-overflow feature with an output compare channel, the TIM can generate a PWM signal. The value in the TIM counter modulo registers determines the period of the PWM signal. The channel pin toggles when the counter reaches the value in the TIM counter modulo registers. The time between overflows is the period of the PWM signal.

As **Figure 11-3** shows, the output compare value in the TIM channel registers determines the pulse width of the PWM signal. The time between overflow and output compare is the pulse width. Program the TIM to clear the channel pin on output compare if the state of the PWM pulse is logic 1. Program the TIM to set the pin if the state of the PWM pulse is logic 0.

The value in the TIM counter modulo registers and the selected prescaler output determines the frequency of the PWM output. The frequency of an 8-bit PWM signal is variable in 256 increments. Writing $00FF (255) to the TIM counter modulo registers produces a PWM period of 256 times the internal bus clock period if the prescaler select value is $000. See **11.10.1 TIM Status and Control Register**.

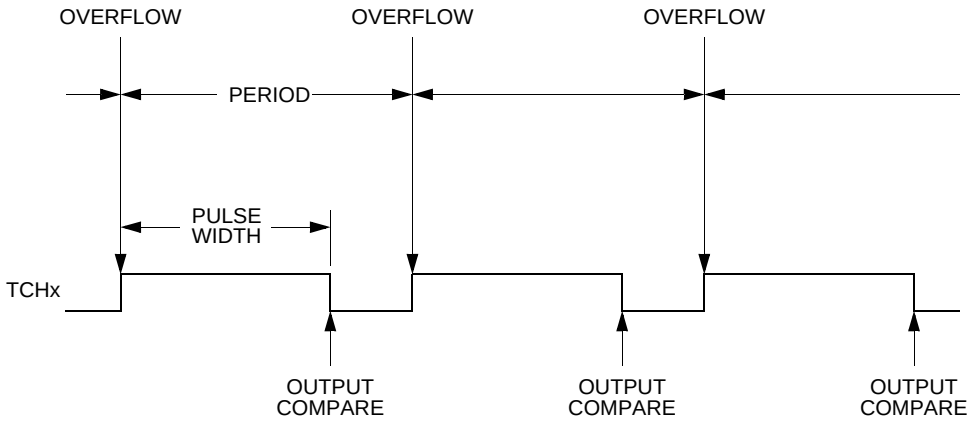


Figure 11-3. PWM Period and Pulse Width

The value in the TIM channel registers determines the pulse width of the PWM output. The pulse width of an 8-bit PWM signal is variable in 256 increments. Writing $0080 (128) to the TIM channel registers produces a duty cycle of 128/256 or 50%.

11.5.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in **11.5.4 Pulse Width Modulation (PWM)**. The pulses are unbuffered because changing the pulse width requires writing the new pulse width value over the old value currently in the TIM channel registers.

An unsynchronized write to the TIM channel registers to change a pulse width value could cause incorrect operation for up to two PWM periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that PWM period. Also, using a TIM overflow interrupt routine to write a new, smaller pulse width value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the PWM pulse width on channel x:

- When changing to a shorter pulse width, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current pulse. The interrupt routine has until the end of the PWM period to write the new value.
- When changing to a longer pulse width, enable TIM overflow interrupts and write the new value in the TIM overflow interrupt routine. The TIM overflow interrupt occurs at the end of the current PWM period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same PWM period.

NOTE: *In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0% duty cycle generation and removes the ability of the channel to self-correct in the event of software error or noise. Toggling on output compare also can cause incorrect PWM signal generation when changing the PWM pulse width to a new, much larger value.*

11.5.4.2 Buffered PWM Signal Generation

Channels 0 and 1 can be linked to form a buffered PWM channel whose output appears on the TCH0 pin. The TIM channel registers of the linked pair alternately control the pulse width of the output.

Setting the MS0B bit in TIM channel 0 status and control register (TSC0) links channel 0 and channel 1. The TIM channel 0 registers initially control the pulse width on the TCH0 pin. Writing to the TIM channel 1 registers enables the TIM channel 1 registers to synchronously control the pulse width at the beginning of the next PWM period. At each subsequent overflow, the TIM channel registers (0 or 1) that control the pulse width are the ones written to last. TSC0 controls and monitors the buffered PWM function, and TIM channel 1 status and control register (TSC1) is unused. While the MS0B bit is set, the channel 1 pin, TCH1, is available as a general-purpose I/O pin.

NOTE: *In buffered PWM signal generation, do not write new pulse width values to the currently active channel registers. User software should track the currently active channel to prevent writing a new value to the active channel. Writing to the active channel registers is the same as generating unbuffered PWM signals.*

11.5.4.3 PWM Initialization

To ensure correct operation when generating unbuffered or buffered PWM signals, use the following initialization procedure:

1. In the TIM status and control register (TSC):
   a. Stop the TIM counter by setting the TIM stop bit, TSTOP.
   b. Reset the TIM counter and prescaler by setting the TIM reset bit, TRST.
2. In the TIM counter modulo registers (TMODH:TMODL), write the value for the required PWM period.
3. In the TIM channel x registers (TCHxH:TCHxL), write the value for the required pulse width.
4. In TIM channel x status and control register (TSCx):
   a. Write 0:1 (for unbuffered output compare or PWM signals) or 1:0 (for buffered output compare or PWM signals) to the mode select bits, MSxB:MSxA. (See **Table 11-3**.)
   b. Write 1 to the toggle-on-overflow bit, TOVx.
   c. Write 1:0 (to clear output on compare) or 1:1 (to set output on compare) to the edge/level select bits, ELSxB:ELSxA. The output action on compare must force the output to the complement of the pulse width level. (See **Table 11-3**.)

NOTE: *In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0% duty cycle generation and removes the ability of the channel to self-correct in the event of software error or noise. Toggling on output compare can also cause incorrect PWM signal generation when changing the PWM pulse width to a new, much larger value.*

5. In the TIM status control register (TSC), clear the TIM stop bit, TSTOP.

Setting MS0B links channels 0 and 1 and configures them for buffered PWM operation. The TIM channel 0 registers (TCH0H:TCH0L) initially control the buffered PWM output. TIM status control register 0 (TSCR0) controls and monitors the PWM signal from the linked channels.

Clearing the toggle-on-overflow bit, TOVx, inhibits output toggles on TIM overflows. Subsequent output compares try to force the output to a state it is already in and have no effect. The result is a 0% duty cycle output.

Setting the channel x maximum duty cycle bit (CHxMAX) and setting the TOVx bit generates a 100% duty cycle output. (See **11.10.4 TIM Channel Status and Control Registers**.)

11.6 Interrupts

The following TIM sources can generate interrupt requests:

- TIM overflow flag (TOF) — The TOF bit is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. The TIM overflow interrupt enable bit, TOIE, enables TIM overflow CPU interrupt requests. TOF and TOIE are in the TIM status and control register.
- TIM channel flags (CH1F:CH0F) — The CHxF bit is set when an input capture or output compare occurs on channel x. Channel x TIM CPU interrupt requests are controlled by the channel x interrupt enable bit, CHxIE. Channel x TIM CPU interrupt requests are enabled when CHxIE = 1. CHxF and CHxIE are in the TIM channel x status and control register.

11.7 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

11.7.1 Wait Mode

The TIM remains active after the execution of a WAIT instruction. In wait mode, the TIM registers are not accessible by the CPU. Any enabled CPU interrupt request from the TIM can bring the MCU out of wait mode.

If TIM functions are not required during wait mode, reduce power consumption by stopping the TIM before executing the WAIT instruction.

11.7.2 Stop Mode

The TIM is inactive after the execution of a STOP instruction. The STOP instruction does not affect register conditions or the state of the TIM counter. TIM operation resumes when the MCU exits stop mode after an external interrupt.

11.8 TIM During Break Interrupts

A break interrupt stops the TIM counter.

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state. (See **9.8.3 SIM Break Flag Control Register**.)

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at logic 0. After the break, doing the second step clears the status bit.

11.9 I/O Signals

Port B shares four of its pins with the TIM. The four TIM channel I/O pins are T1CH0, T1CH1, T2CH0, and T2CH1 as described in **11.4 Pin Name Conventions**.

Each channel I/O pin is programmable independently as an input capture pin or an output compare pin. T1CH0 and T2CH0 can be configured as buffered output compare or buffered PWM pins.

11.10 I/O Registers

NOTE: *References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TSC may generically refer to both T1SC AND T2SC.*

These I/O registers control and monitor operation of the TIM:

- TIM status and control register (TSC)
- TIM counter registers (TCNTH:TCNTL)
- TIM counter modulo registers (TMODH:TMODL)
- TIM channel status and control registers (TSC0, TSC1)
- TIM channel registers (TCH0H:TCH0L, TCH1H:TCH1L)

11.10.1 TIM Status and Control Register

The TIM status and control register (TSC):

- Enables TIM overflow interrupts
- Flags TIM overflows
- Stops the TIM counter
- Resets the TIM counter
- Prescales the TIM counter clock

Address: T1SC, $0020 and T2SC, $002B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
Write:	0			TRST				
Reset:	0	0	1	0	0	0	0	0

= Unimplemented

Figure 11-4. TIM Status and Control Register (TSC)

TOF — TIM Overflow Flag Bit

This read/write flag is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. Clear TOF by reading the TIM status and control register when TOF is set and then writing a logic 0 to TOF. If another TIM overflow occurs before the clearing sequence is complete, then writing logic 0 to TOF has no effect. Therefore, a TOF interrupt request cannot be lost due to inadvertent clearing of TOF. Reset clears the TOF bit. Writing a logic 1 to TOF has no effect.

1 = TIM counter has reached modulo value
0 = TIM counter has not reached modulo value

TOIE — TIM Overflow Interrupt Enable Bit

This read/write bit enables TIM overflow interrupts when the TOF bit becomes set. Reset clears the TOIE bit.

1 = TIM overflow interrupts enabled
0 = TIM overflow interrupts disabled

TSTOP — TIM Stop Bit

This read/write bit stops the TIM counter. Counting resumes when TSTOP is cleared. Reset sets the TSTOP bit, stopping the TIM counter until software clears the TSTOP bit.

1 = TIM counter stopped
0 = TIM counter active

NOTE: *Do not set the TSTOP bit before entering wait mode if the TIM is required to exit wait mode.*

TRST — TIM Reset Bit

Setting this write-only bit resets the TIM counter and the TIM prescaler. Setting TRST has no effect on any other registers. Counting resumes from $0000. TRST is cleared automatically after the TIM counter is reset and always reads as logic 0. Reset clears the TRST bit.

1 = Prescaler and TIM counter cleared
0 = No effect

NOTE: *Setting the TSTOP and TRST bits simultaneously stops the TIM counter at a value of $0000.*

PS[2:0] — Prescaler Select Bits

These read/write bits select one of the seven prescaler outputs as the input to the TIM counter as **Table 11-2** shows. Reset clears the PS[2:0] bits.

Table 11-2. Prescaler Selection

PS2	PS1	PS0	TIM Clock Source
0	0	0	Internal bus clock ÷ 1
0	0	1	Internal bus clock ÷ 2
0	1	0	Internal bus clock ÷ 4
0	1	1	Internal bus clock ÷ 8
1	0	0	Internal bus clock ÷ 16
1	0	1	Internal bus clock ÷ 32
1	1	0	Internal bus clock ÷ 64
1	1	1	Not available

11.10.2 TIM Counter Registers

The two read-only TIM counter registers contain the high and low bytes of the value in the TIM counter. Reading the high byte (TCNTH) latches the contents of the low byte (TCNTL) into a buffer. Subsequent reads of TCNTH do not affect the latched TCNTL value until TCNTL is read. Reset clears the TIM counter registers. Setting the TIM reset bit (TRST) also clears the TIM counter registers.

NOTE: *If you read TCNTH during a break interrupt, be sure to unlatch TCNTL by reading TCNTL before exiting the break interrupt. Otherwise, TCNTL retains the value latched during the break.*

Address: T1CNTH, $0021 and T2CNTH, $002C

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

= Unimplemented

Figure 11-5. TIM Counter Registers High (TCNTH)

Address: T1CNTL, $0022 and T2CNTL, $002D

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

= Unimplemented

Figure 11-6. TIM Counter Registers Low (TCNTL)

11.10.3 TIM Counter Modulo Registers

The read/write TIM modulo registers contain the modulo value for the TIM counter. When the TIM counter reaches the modulo value, the overflow flag (TOF) becomes set, and the TIM counter resumes counting from $0000 at the next timer clock. Writing to the high byte (TMODH) inhibits the TOF bit and overflow interrupts until the low byte (TMODL) is written. Reset sets the TIM counter modulo registers.

Address: T1MODH, $0023 and T2MODH, $002E

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

Figure 11-7. TIM Counter Modulo Register High (TMODH)

Address: T1MODL, $0024 and T2MODL, $002F

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

Figure 11-8. TIM Counter Modulo Register Low (TMODL)

NOTE: *Reset the TIM counter before writing to the TIM counter modulo registers.*

11.10.4 TIM Channel Status and Control Registers

Each of the TIM channel status and control registers:

- Flags input captures and output compares
- Enables input capture and output compare interrupts
- Selects input capture, output compare, or PWM operation
- Selects high, low, or toggling output on output compare
- Selects rising edge, falling edge, or any edge as the active input capture trigger
- Selects output toggling on TIM overflow
- Selects 0% and 100% PWM duty cycle
- Selects buffered or unbuffered output compare/PWM operation

Address: T1SC0, $0025 and T2SC0, $0030

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
Write:	0							
Reset:	0	0	0	0	0	0	0	0

Figure 11-9. TIM Channel 0 Status and Control Register (TSC0)

Address: T1SC1, $0028 and T2SC1, $0033

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
Write:	0							
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 11-10. TIM Channel 1 Status and Control Register (TSC1)

CHxF — Channel x Flag Bit

When channel x is an input capture channel, this read/write bit is set when an active edge occurs on the channel x pin. When channel x is an output compare channel, CHxF is set when the value in the TIM counter registers matches the value in the TIM channel x registers.

When TIM CPU interrupt requests are enabled (CHxIE = 1), clear CHxF by reading TIM channel x status and control register with CHxF set and then writing a logic 0 to CHxF. If another interrupt request occurs before the clearing sequence is complete, then writing logic 0 to CHxF has no effect. Therefore, an interrupt request cannot be lost due to inadvertent clearing of CHxF.

Reset clears the CHxF bit. Writing a logic 1 to CHxF has no effect.

1 = Input capture or output compare on channel x
0 = No input capture or output compare on channel x

CHxIE — Channel x Interrupt Enable Bit

This read/write bit enables TIM CPU interrupt service requests on channel x.

Reset clears the CHxIE bit.

1 = Channel x CPU interrupt requests enabled
0 = Channel x CPU interrupt requests disabled

MSxB — Mode Select Bit B

This read/write bit selects buffered output compare/PWM operation. MSxB exists only in the TIM1 channel 0 and TIM2 channel 0 status and control registers.

Setting MS0B disables the channel 1 status and control register and reverts TCH1 to general-purpose I/O.

Reset clears the MSxB bit.

1 = Buffered output compare/PWM operation enabled
0 = Buffered output compare/PWM operation disabled

MSxA — Mode Select Bit A

When ELSxB:ELSxA ≠ 0:0, this read/write bit selects either input capture operation or unbuffered output compare/PWM operation. See **Table 11-3**.

1 = Unbuffered output compare/PWM operation
0 = Input capture operation

When ELSxB:ELSxA = 0:0, this read/write bit selects the initial output level of the TCHx pin. See **Table 11-3**. Reset clears the MSxA bit.

1 = Initial output level low
0 = Initial output level high

NOTE: *Before changing a channel function by writing to the MSxB or MSxA bit, set the TSTOP and TRST bits in the TIM status and control register (TSC).*

ELSxB and ELSxA — Edge/Level Select Bits

When channel x is an input capture channel, these read/write bits control the active edge-sensing logic on channel x.

When channel x is an output compare channel, ELSxB and ELSxA control the channel x output behavior when an output compare occurs.

When ELSxB and ELSxA are both clear, channel x is not connected to an I/O port, and pin TCHx is available as a general-purpose I/O pin. **Table 11-3** shows how ELSxB and ELSxA work. Reset clears the ELSxB and ELSxA bits.

Table 11-3. Mode, Edge, and Level Selection

MSxB:MSxA	ELSxB:ELSxA	Mode	Configuration
X0	00	Output preset	Pin under port control; initial output level high
X1	00		Pin under port control; initial output level low
00	01	Input capture	Capture on rising edge only
00	10		Capture on falling edge only
00	11		Capture on rising or falling edge
01	01	Output compare or PWM	Toggle output on compare
01	10		Clear output on compare
01	11		Set output on compare
1X	01	Buffered output compare or buffered PWM	Toggle output on compare
1X	10		Clear output on compare
1X	11		Set output on compare

NOTE: *Before enabling a TIM channel register for input capture operation, make sure that the TCHx pin is stable for at least two bus clocks. User software should also clear CHxF before setting CHxIE to avoid any false interrupts.*

TOVx — Toggle On Overflow Bit

When channel x is an output compare channel, this read/write bit controls the behavior of the channel x output when the TIM counter overflows. When channel x is an input capture channel, TOVx has no effect. Reset clears the TOVx bit.

1 = Channel x pin toggles on TIM counter overflow
0 = Channel x pin does not toggle on TIM counter overflow

NOTE: *When TOVx is set, a TIM counter overflow takes precedence over a channel x output compare if both occur at the same time.*

CHxMAX — Channel x Maximum Duty Cycle Bit

When the TOVx bit is at logic 1, setting the CHxMAX bit forces the duty cycle of buffered and unbuffered PWM signals to 100%. As **Figure 11-11** shows, the CHxMAX bit takes effect in the cycle after it is set or cleared. The output stays at the 100% duty cycle level until the cycle after CHxMAX is cleared.

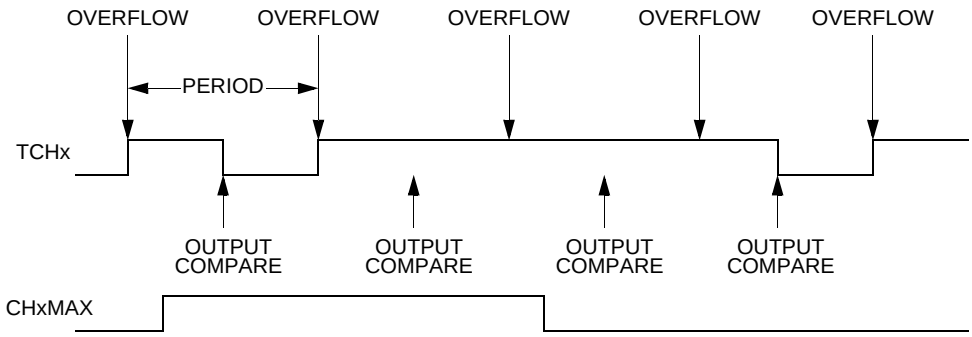


Figure 11-11. CHxMAX Latency

11.10.5 TIM Channel Registers

These read/write registers contain the captured TIM counter value of the input capture function or the output compare value of the output compare function. The state of the TIM channel registers after reset is unknown.

In input capture mode (MSxB:MSxA = 0:0), reading the high byte of the TIM channel x registers (TCHxH) inhibits input captures until the low byte (TCHxL) is read.

In output compare mode (MSxB:MSxA ≠ 0:0), writing to the high byte of the TIM channel x registers (TCHxH) inhibits output compares until the low byte (TCHxL) is written.

Address: T1CH0H, $0026 and T2CH0H, $0031

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

Figure 11-12. TIM Channel 0 Register High (TCH0H)

Address: T1CH0L, $0027 and T2CH0L $0032

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

Figure 11-13. TIM Channel 0 Register Low (TCH0L)

Address: T1CH1H, $0029 and T2CH1H, $0034

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

Figure 11-14. TIM Channel 1 Register High (TCH1H)

Address: T1CH1L, $002A and T2CH1L, $0035

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

Figure 11-15. TIM Channel 1 Register Low (TCH1L)

Section 12. Real Time Clock (RTC)

12.1 Contents

12.2 Introduction 210
12.3 Features 210
12.4 Functional Description 212
12.4.1 Time Functions 214
12.4.2 Calendar Functions 214
12.4.3 Alarm Functions 214
12.4.4 Timebase Interrupts 214
12.4.5 Chronograph Functions 215
12.5 Low-Power Modes 215
12.5.1 Wait Mode 215
12.5.2 Stop Mode 215
12.6 RTC Registers 216
12.6.1 RTC Control Register 1 (RTCCR1) 216
12.6.2 RTC Control Register 2 (RTCCR2) 218
12.6.3 RTC Status Register (RTCSR) 219
12.6.4 Alarm Minute and Hour Registers (ALMR and ALHR) 222
12.6.5 Second Register (SECR) 223
12.6.6 Minute Register (MINR) 223
12.6.7 Hour Register (HRR) 224
12.6.8 Day Register (DAYR) 224
12.6.9 Month Register (MTHR) 225
12.6.10 Year Register (YRR) 225
12.6.11 Day-Of-Week Register (DOWR) 226
12.6.12 Chronograph Data Register (CHRR) 226

12.2 Introduction

This section describes the real time clock (RTC) module. The RTC provides real time clock and calendar functions with automatic leap year adjustments. Other functions include alarm interrupt, periodic interrupts, and a chronograph timer.

12.3 Features

Features of the RTC module include:

- Counter registers for:
  - Second
  - Minute
  - Hour
  - Day
  - Day-of-week
  - Month
  - Year
- Day counter with automatic month and leap year adjustment
- 1/100 seconds chronograph counter
- Seven periodic interrupts
- Alarm interrupt

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0042	RTC Control Register 1 (RTCCR1)	Read: / Write:	ALMIE	CHRIE	DAYIE	HRIE	MINIE	SECIE	TB1IE	TB2IE
		Reset:	0	0	0	0	0	0	0	0
$0043	RTC Control Register 2 (RTCCR2)	Read:	0	0	CHRE	RTCE	0	XTL2	XTL1	XTL0
		Write:	R	CHRCLR						
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-1. RTC I/O Register Summary

$0044	RTC Status Register (RTCSR)	Read:	ALMF	CHRF	DAYF	HRF	MINF	SECF	TB1F	TB2F
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0045	Alarm Minute Register (ALMR)	Read:	0	0	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0046	Alarm Hour Register (ALHR)	Read:	0	0	0	AH4	AH3	AH2	AH1	AH0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0047	Second Register (SECR)	Read:	0	0	SEC5	SEC4	SEC3	SEC2	SEC1	SEC0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0048	Minute Register (MINR)	Read:	0	0	MIN5	MIN4	MIN3	MIN2	MIN1	MIN0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0049	Hour Register (HRR)	Read:	0	0	0	HR4	HR3	HR2	HR1	HR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004A	Day Register (DAYR)	Read:	0	0	0	DAY4	DAY3	DAY2	DAY1	DAY0
		Write:								
		Reset:	0	0	0	0	0	0	0	1
$004B	Month Register (MTHR)	Read:	0	0	0	0	MTH3	MTH2	MTH1	MTH0
		Write:								
		Reset:	0	0	0	0	0	0	0	1
$004C	Year Register (YRR)	Read:	YR7	YR6	YR5	YR4	YR3	YR2	YR1	YR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004D	Day-Of-Week Register (DOWR)	Read:	0	0	0	0	0	DOW2	DOW1	DOW0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$004E	Chronograph Data Register (CHRR)	Read:	0	CHR6	CHR5	CHR4	CHR3	CHR2	CHR1	CHR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-1. RTC I/O Register Summary

12.4 Functional Description

The RTC module provides clock indications in seconds, minutes, and hours; calendar indications in day-of-week, day-of-month, month, and year; with automatic adjustment for month and leap year. Reading the clock and calendar registers return the current time and date. Writing to these registers set the time and date, and the counters will continue to count from the new settings.

The alarm interrupt is set for the hour and minute. When the hour and minute counters matches the time set in the alarm hour and minute registers, the alarm flag is set. The alarm can be configured to generate a CPU interrupt request.

A 1/100 seconds chronograph counter is provided for timing applications. This counter can be independently enabled or disabled, and cleared at any time.

RTC module interrupts include the alarm interrupt and seven periodic interrupts from the clock counters.

For proper RTC module operation, one of the following oscillator frequencies (CGMXCLK) must be used:

- 32.768 kHz
- 32.000 kHz
- 38.400 kHz
- 64.000 kHz
- 76.800 kHz

Configuring the XTL[2:0] bits in the RTC control register 2 selects the appropriate prescalers and dividers to divide CGMXCLK down to the basic 1 Hz clock for driving the clock counters.

Figure 12-2 shows the structure of the RTC module.

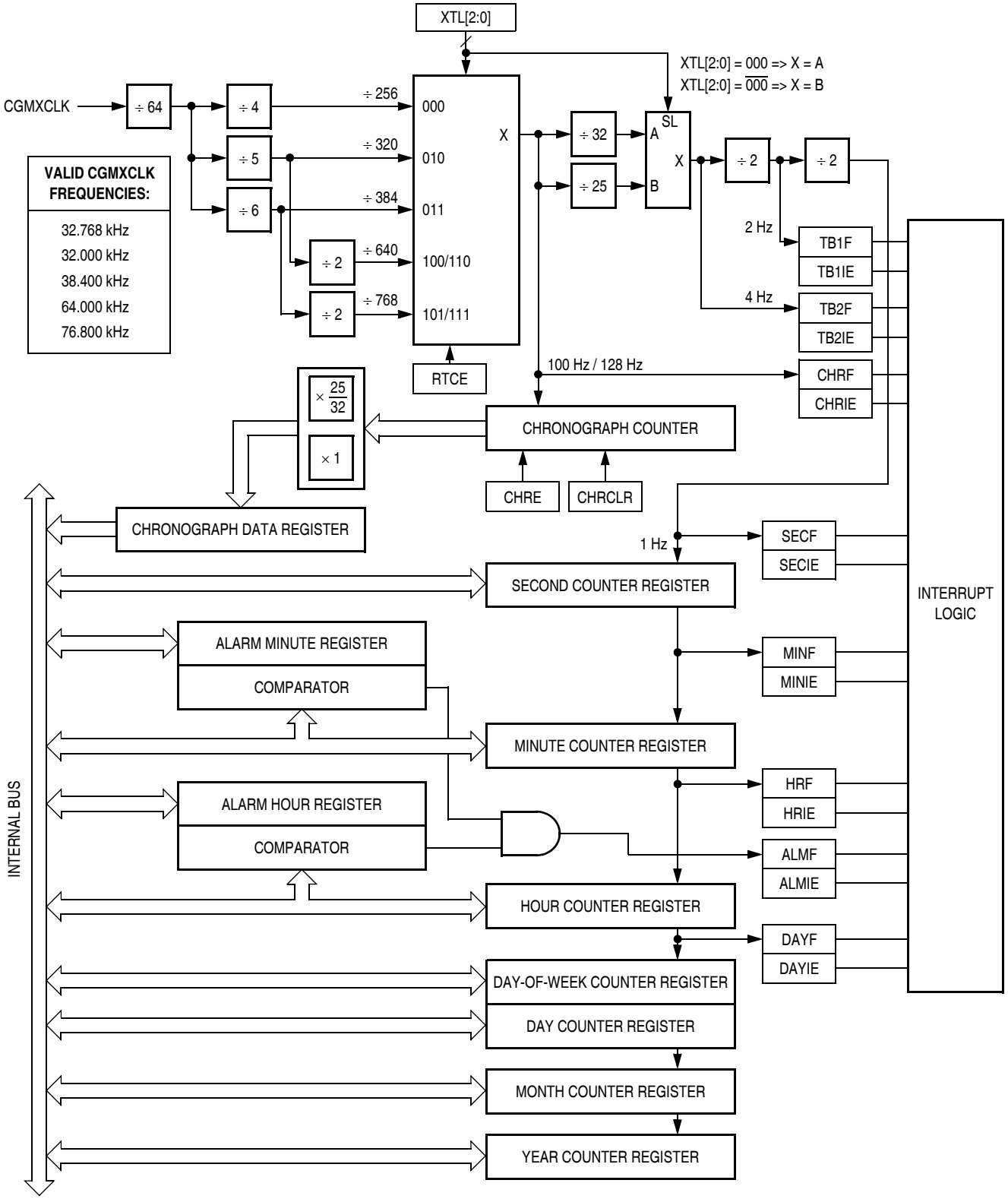


Figure 12-2. RTC Block Diagram

12.4.1 Time Functions

Real time clock functions are provided by the second, minute, and hour counter registers. All three clock counters are able to generate interrupts on every counter increment, providing periodic interrupts for the second (SECF), minute (MINF), and hour (HRF). A CPU interrupt request is generated if the corresponding enable bit (SECIE, MINIE, and HRIE) is also set.

12.4.2 Calendar Functions

Calendar functions are provided by the day, day-of-week, month, and year counter registers. The roll over of the day counter is automatically adjusted for the month and leap years. The setting for the year counter ranges from 1901 to 2099.

The day flag (DAYF) is set on every increment of the day counter. A CPU interrupt request is generated if the day interrupt enable bit (DAYIE) is also set.

12.4.3 Alarm Functions

An alarm function is provided for the minute and hour counters. When minute counter matches the value stored in the alarm minute register, and the hour counter matches the value stored in the alarm hour register, the alarm flag (ALMF) will be set. A CPU interrupt request is generated if the alarm interrupt enable bit (ALMIE) is also set.

12.4.4 Timebase Interrupts

In addition to the second, minute, hour, and day periodic interrupts generated by the clock functions, the divider circuits generates a 2 Hz and a 4 Hz periodic interrupt. These are indicated by the TB1F and TB2F flags. A CPU interrupt request is generated if the corresponding enable bits (TB1IE and TB2IE) is also set.

12.4.5 Chronograph Functions

A 100Hz resolution chronograph counter can be enabled by setting the CHRE bit. The chronograph counter will automatically roll over to zero when the counter reaches 99. If 32.768kHz CGMXCLK is used, the chronograph counter resolution becomes 128Hz. With either 100Hz or 128Hz resolution, the counter value is converted to 100Hz, before it is saved in the chronograph data register. Therefore, each chronograph data register increment represents 10ms.

12.5 Low-Power Modes

The STOP and WAIT instructions put the MCU in low power-consumption standby modes.

12.5.1 Wait Mode

The RTC module continues normal operation in wait mode. Any enabled CPU interrupt request from the RTC can bring the MCU out of wait mode. If the RTC is not required to bring the MCU out of wait mode, power down the RTC by clearing the RTCE bit before executing the WAIT instruction.

12.5.2 Stop Mode

For continuous RTC operation in stop mode, the oscillator stop mode enable bit (STOP_XCLKEN in CONFIG2 register) must be set before executing the STOP instruction. When STOP_XCLKEN is set, CGMXCLK continues to drive the RTC module, and any enabled CPU interrupt request from the RTC can bring the MCU out of stop mode.

If STOP_XCLKEN bit is cleared, the RTC module is inactive after the execution of a STOP instruction. The STOP instruction does not affect RTC register states. RTC module operation resumes after an external interrupt. To further reduce power consumption, the RTC module should be powered-down by clearing the RTCE bit before executing the STOP instruction.

12.6 RTC Registers

The RTC module has thirteen memory-mapped registers:

- RTC control register 1 (RTCCR1)
- RTC control register 2 (RTCCR2)
- RTC status register (RTCSR)
- Alarm minute and hour registers (ALMR and ALHR)
- Second register (SECR)
- Minute register (MINR)
- Hour register (HRR)
- Day register (DAY)
- Month register (MTHR)
- Year register (YRR)
- Day of the week register (DOWR)
- Chronograph data register (CHRR)

12.6.1 RTC Control Register 1 (RTCCR1)

The RTC control register 1 (RTCCR1) contains the eight interrupt enable bits for RTC interrupt functions.

Address: $0042

	ALMIE	CHRIE	DAYIE	HRIE	MINIE	SECIE	TB1IE	TB2IE
Read: Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 12-3. RTC Control Register 1 (RTCCR1)

ALMIE — Alarm Interrupt Enable

This read/write bit enables the alarm flag, ALMF, to generate CPU interrupt requests. Reset clears the ALMIE bit.

1 = ALMF enabled to generate CPU interrupt
0 = ALMF not enabled to generate CPU interrupt

CHRIE — Chronograph Interrupt Enable

This read/write bit enables the chronograph flag, CHRF, to generate CPU interrupt requests. Reset clears the CHRIE bit.

1 = CHRF enabled to generate CPU interrupt
0 = CHRF not enabled to generate CPU interrupt

DAYIE — Day Interrupt Enable

This read/write bit enables the day flag, DAYF, to generate CPU interrupt requests. Reset clears the DAYIE bit.

1 = DAYF enabled to generate CPU interrupt
0 = DAYF not enabled to generate CPU interrupt

HRIE — Hour Interrupt Enable

This read/write bit enables the hour flag, HRF, to generate CPU interrupt requests. Reset clears the HRIE bit.

1 = HRF enabled to generate CPU interrupt
0 = HRF not enabled to generate CPU interrupt

MINIE — Minute Interrupt Enable

This read/write bit enables the minute flag, MINF, to generate CPU interrupt requests. Reset clears the MINIE bit.

1 = MINF enabled to generate CPU interrupt
0 = MINF not enabled to generate CPU interrupt

SECIE — Second Interrupt Enable

This read/write bit enables the second flag, SECF, to generate CPU interrupt requests. Reset clears the SECIE bit.

1 = SECF enabled to generate CPU interrupt
0 = SECF not enabled to generate CPU interrupt

TB1IE — Timebase 1 Interrupt Enable

This read/write bit enables the timebase1 flag, TB1F, to generate CPU interrupt requests. Reset clears the TB1IE bit.

1 = TB1F enabled to generate CPU interrupt
0 = TB1F not enabled to generate CPU interrupt

TB2IE — Timebase 2 Interrupt Enable

This read/write bit enables the timebase2 flag, TB2F, to generate CPU interrupt requests. Reset clears the TB2IE bit.

1 = TB2F enabled to generate CPU interrupt
0 = TB2F not enabled to generate CPU interrupt

12.6.2 RTC Control Register 2 (RTCCR2)

The RTC control register 2 (RTCCR2) contains control and clock selection bits for RTC operation.

Address: $0043

Read:	0	0	CHRE	RTCE	0	XTL2	XTL1	XTL0
Write:	R	CHRCLR						
Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-4. RTC Control Register 2 (RTCCR2)

CHRCLR — Chronograph counter clear

Setting this write-only bit resets the chronograph counter. Setting CHRCLR has no effect on any other registers. Counting resumes from $00. CHRCLR is cleared automatically after the chronograph counter is reset and always reads as logic 0. Reset clears the CHRCLR bit.

1 = Chronograph counter cleared
0 = No effect

CHRE — Chronograph Enable

This read/write bit enables the chronograph counter, the value in the chronograph data register increments by 1 in every 1/100 seconds. When the chronograph counter is disabled (CHRE = 0), the value in the chronograph data register is held at the count value. Reset clears the CHRE bit.

1 = Chronograph counter enabled
0 = Chronograph counter disabled

RTCE — Real Time Clock Enable

This read/write bit enables the entire RTC module, allowing all RTC and chronograph operations. Disabling the RTC module does not affect the contents in the RTC registers. Reset clears the RTCE bit.

1 = RTC module enabled
0 = RTC module disabled

XTL[2:0] — Crystal Frequency Select Bits

These three bits set the prescalers/dividers for proper operation of the RTC module for various crystal (CGMXCLK) input frequencies. The XTL[2:0] bits can only be written once after reset, subsequent writes to these bits will have no effect on its content. **Table 12-1** shows the XTL[2:0] settings for various CGMXCLK frequencies. Reset clear the XTL[2:0] bits.

Table 12-1. CGMXCLK Frequency for RTC Input Reference

CGMXCLK(1)	XTL2	XTL1	XTL0
32.768 kHz	0	0	0
Reserved	0	0	1
32.000 kHz	0	1	0
38.400 kHz	0	1	1
64.000 kHz	1	X	0
76.800 kHz	1	X	1

Notes:
1. Using crystal frequencies other than these specified will cause incorrect timings in the RTC module.

12.6.3 RTC Status Register (RTCSR)

The RTC status register contains eight status flags. When a flag is set and the corresponding interrupt enable bit is also set, a CPU interrupt request is generated.

Address: $0044

Read:	ALMF	CHRF	DAYF	HRF	MINF	SECF	TB1F	TB2F
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-5. RTC Status Register (RTCSR)

ALMF — Alarm Flag

This clearable, read-only bit is set when the value in the RTC hour and minute counters matches the value in the alarm hour and alarm minute registers. When the ALMIE bit in RTCCR1 is set, ALMF generates a CPU interrupt request. In normal operation, clear the ALMF bit by reading RTCSR with ALMF set and then reading the alarm hour register (ALHR). Reset clears ALMF.

1 = RTC hour and minute counters matches the alarm hour and minute registers

0 = No matching between hour and minute counters and alarm hour and minute registers

CHRF — Chronograph Flag

This clearable, read-only bit is set on every tick of the chronograph counter (every counter count). The tick is on every 1/100 or 1/128 seconds (see **12.4.5 Chronograph Functions**). When the CHRIE bit in RTCCR1 is set, CHRF generates a CPU interrupt request. In normal operation, clear the CHRF bit by reading RTCSR with CHRF set and then reading the chronograph data register (CHRR). Reset clears CHRF.

1 = A chronograph counter tick has occurred

0 = No chronograph counter tick has occurred

DAYF — Day Flag

This clearable, read-only bit is set on every increment of the day counter. When the DAYIE bit in RTCCR1 is set, DAYF generates a CPU interrupt request. In normal operation, clear the DAYF bit by reading RTCSR with DAYF set and then reading the day register (DAYR). Reset clears DAYF.

1 = Day counter incremented

0 = No day counter incremented

HRF — Hour Flag

This clearable, read-only bit is set on every increment of the hour counter. When the HRIE bit in RTCCR1 is set, HRF generates a CPU interrupt request. In normal operation, clear the HRF bit by reading RTCSR with HRF set and then reading the hour register (HRR). Reset clears HRF.

1 = Hour counter incremented

0 = No hour counter incremented

MINF — Minute Flag

This clearable, read-only bit is set on every increment of the minute counter. When the MINIE bit in RTCCR1 is set, MINF generates a CPU interrupt request. In normal operation, clear the MINF bit by reading RTCSR with MINF set and then reading the minute register (MINR). Reset clears MINF.

1 = Minute counter incremented
0 = No minute counter incremented

SECF — Second Flag

This clearable, read-only bit is set on every increment of the second counter. When the SECIE bit in RTCCR1 is set, SECF generates a CPU interrupt request. In normal operation, clear the SECF bit by reading RTCSR with SECF set and then reading the second register (SECR). Reset clears SECF.

1 = Second counter incremented
0 = No second counter incremented

TB1F — Timebase 1 Flag

This clearable, read-only bit is set on every tick of the timebase 1 counter (every 0.5 seconds). When the TB1IE bit in RTCCR1 is set, TB1F generates a CPU interrupt request. In normal operation, clear the TB1F bit by reading RTCSR with TB1F set and then reading the chronograph register (CHRR). Reset clears TB1F.

1 = A timebase 1 tick (0.5s) has occurred
0 = No timebase 1 tick has occurred

TB2F — Timebase 2 Flag

This clearable, read-only bit is set on every tick of the timebase 2 counter (every 0.25 seconds). When the TB2IE bit in RTCCR1 is set, TB2F generates a CPU interrupt request. In normal operation, clear the TB2F bit by reading RTCSR with TB2F set and then reading the chronograph register (CHRR). Reset clears TB2F.

1 = A timebase 2 tick (0.25s) has occurred
0 = No timebase 2 tick has occurred

12.6.4 Alarm Minute and Hour Registers (ALMR and ALHR)

These read/write registers contain the alarm minute and hour values for the hour and minute alarm function. When the hour counter matches the value in the alarm hour register (ALHR) and the minute counter matches the value in the alarm minute register (ALMR), the alarm flag, ALMF, is set. When ALMF is set and the alarm interrupt enable bit, ALMIE, is also set, a CPU interrupt request is generated.

Address: $0045

Read:	0	0	AM5	AM4	AM3	AM2	AM1	AM0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-6. Alarm Minute Register (ALMR)

NOTE: *Writing values other than 0 to 59, to ALMR is possible, but the alarm flag will never be set.*

Address: $0046

Read:	0	0	0	AH4	AH3	AH2	AH1	AH0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-7. Alarm Hour Register (ALHR)

NOTE: *Writing values other than 0 to 23, to ALHR is possible, but the alarm flag will never be set.*

12.6.5 Second Register (SECR)

This read/write register contains the current value of the second counter. This register can be read at any time without affecting the counter count. Writing to this register loads the value to the second counter and the counter continues to count from this new value.

The second counter rolls over to 0 ($00) after reaching 59 ($4B). Writing a value other than 0 to 59 to this register has no effect.

Address: $0047

Read:	0	0	SEC5	SEC4	SEC3	SEC2	SEC1	SEC0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-8. Second Register (SECR)

12.6.6 Minute Register (MINR)

This read/write register contains the current value of the minute counter. This register can be read at any time without affecting the counter count. Writing to this register loads the value to the minute counter and the counter continues to count from this new value.

The minute counter rolls over to 0 ($00) after reaching 59 ($4B). Writing a value other than 0 to 59 to this register has no effect.

Address: $0048

Read:	0	0	MIN5	MIN4	MIN3	MIN2	MIN1	MIN0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-9. Minute Register (MINR)

12.6.7 Hour Register (HRR)

This read/write register contains the current value of the hour counter. This register can be read at any time without affecting the counter count. Writing to this register loads the value to the hour counter and the counter continues to count from this new value.

The hour counter rolls over to 0 ($00) after reaching 23 ($17). Writing a value other than 0 to 23 to this register has no effect.

Address: $0049

Read:	0	0	0	HR4	HR3	HR2	HR1	HR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-10. Hour Register (HRR)

12.6.8 Day Register (DAYR)

This read/write register contains the current value of the day-of-month counter. This register can be read at any time without affecting the counter count. Writing to this register loads the value to the day counter and the counter continues to count from this new value.

The day counter rolls over to 1 ($01) after reaching 28 ($1B), 29 ($1C), 30 ($1D), or 31 ($1E), depending on the value in the month and year registers. Writing a value that is not valid for the month and year to this register has no effect.

Address: $004A

Read:	0	0	0	DAY4	DAY3	DAY2	DAY1	DAY0
Write:								
Reset:	0	0	0	0	0	0	0	1

= Unimplemented

Figure 12-11. Day Register (DAYR)

12.6.9 Month Register (MTHR)

This read/write register contains the current value of the month counter. This register can be read at any time without affecting the counter count. Writing to this register loads the value to the month counter and the counter continues to count from this new value.

The month counter rolls over to 1 ($01) after reaching 12 ($0B). Writing a value other than 1 to 12 to this register has no effect.

Address: $004B

Read:	0	0	0	0	MTH3	MTH2	MTH1	MTH0
Write:								
Reset:	0	0	0	0	0	0	0	1

= Unimplemented

Figure 12-12. Month Register (MTHR)

12.6.10 Year Register (YRR)

This read/write register contains the current value of the year counter. This register can be read at any time without affecting the counter count. Writing to this register loads the value to the year counter and the counter continues to count from this new value.

The value stored in this register is a two's complement representation of the year, relative to 2000. For example, the year 2008 is represented by 8 ($08), and the year 1979 is presented by –11 ($F5). The range of this register is only valid for –99 to +99. Writing a value other than –99 to +99 to this register has no effect.

Address: $004C

Read:	YR7	YR6	YR5	YR4	YR3	YR2	YR1	YR0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 12-13. Year Register (YRR)

12.6.11 Day-Of-Week Register (DOWR)

This read/write register contains the current value of the day-of-week counter. This register can be read at any time without affecting the counter count. Writing to this register loads the value to the day-of-week counter and the counter continues to count from this new value.

The day-of-week counter value rolls over to 0 ($00) after reaching 6 ($06). Writing a value other than 0 to 6 to this register has no effect.

Address: $004D

Read:	0	0	0	0	0	DOW2	DOW1	DOW0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-14. Day-Of-Week Register (DOWR)

12.6.12 Chronograph Data Register (CHRR)

This read-only chronograph data register contains the value in the chronograph counter. Reset clears the chronograph data register. Setting the chronograph counter reset bit (CHRCLR) also clears the chronograph data register.

The chronograph data register has a resolution of 1/100 seconds (10ms). The chronograph counter value rolls over to $00 after reaching $63.

Address: $004E

Read:	0	CHR6	CHR5	CHR4	CHR3	CHR2	CHR1	CHR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-15. Chronograph Data Register (CHRR)

Section 13. Infrared Serial Communications Interface Module (IRSCI)

13.1 Contents

13.2 Introduction 228
13.3 Features 229
13.4 Pin Name Conventions 231
13.5 IRSCI Module Overview 231
13.6 Infrared Functional Description 232
13.6.1 Infrared Transmit Encoder 233
13.6.2 Infrared Receive Decoder 233
13.7 SCI Functional Description 234
13.7.1 Data Format 235
13.7.2 Transmitter 236
13.7.2.1 Character Length 237
13.7.2.2 Character Transmission 237
13.7.2.3 Break Characters 238
13.7.2.4 Idle Characters 238
13.7.2.5 Transmitter Interrupts 239
13.7.3 Receiver 239
13.7.3.1 Character Length 239
13.7.3.2 Character Reception 241
13.7.3.3 Data Sampling 241
13.7.3.4 Framing Errors 243
13.7.3.5 Baud Rate Tolerance 243
13.7.3.6 Receiver Wakeup 246
13.7.3.7 Receiver Interrupts 247
13.7.3.8 Error Interrupts 247
13.8 Low-Power Modes 248
13.8.1 Wait Mode 248
13.8.2 Stop Mode 248

13.9 SCI During Break Module Interrupts 249

13.10 I/O Signals 249
13.10.1 PTB0/TxD (Transmit Data) 249
13.10.2 PTB1/RxD (Receive Data) 249

13.11 I/O Registers 250
13.11.1 SCI Control Register 1 251
13.11.2 SCI Control Register 2 253
13.11.3 SCI Control Register 3 256
13.11.4 SCI Status Register 1 258
13.11.5 SCI Status Register 2 (SCS2) 262
13.11.6 SCI Data Register (SCDR) 263
13.11.7 SCI Baud Rate Register (SCBR) 264
13.11.8 SCI Infrared Control Register 267

13.2 Introduction

This section describes the infrared serial communications interface (IRSCI) module which allows high-speed asynchronous communications with peripheral devices and other MCUs. This IRSCI consists of an SCI module for conventional SCI functions and a software programmable infrared encoder/decoder sub-module for encoding/decoding the serial data for connection to infrared LEDs in remote control applications.

NOTE: *References to DMA (direct-memory access) and associated functions are only valid if the MCU has a DMA module. This MCU does not have the DMA function. Any DMA-related register bits should be left in their reset state for normal MCU operation.*

13.3 Features

Features of the SCI module include the following:

- Full duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- Programmable 8-bit or 9-bit character length
- Separately enabled transmitter and receiver
- Separate receiver and transmitter CPU interrupt requests
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Interrupt-driven operation with eight interrupt flags:
  - Transmitter empty
  - Transmission complete
  - Receiver full
  - Idle receiver input
  - Receiver overrun
  - Noise error
  - Framing error
  - Parity error
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection

Features of the infrared (IR) sub-module include the following:

- IR sub-module enable/disable for infrared SCI or conventional SCI on TxD and RxD pins
- Software selectable infrared modulation/demodulation (3/16, 1/16 or 1/32 width pulses)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0013	SCI Control Register 1 (SCC1)	Read:	LOOPS	ENSCI	0	M	WAKE	ILTY	PEN	PTY
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0014	SCI Control Register 2 (SCC2)	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	SCI Control Register 3 (SCC3)	Read:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
		Write:								
		Reset:	U	U	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1)	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:								
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2)	Read:							BKF	RPF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0018	SCI Data Register (SCDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0019	SCI Baud Rate Register (SCBR)	Read:	CKS	0	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001A	SCI Infrared Control Register (SCIRCR)	Read:	R	0	0	0	R	TNP1	TNP0	IREN
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

Figure 13-1. IRSCI I/O Registers Summary

13.4 Pin Name Conventions

The generic names of the IRSCI I/O pins are:

- RxD (receive data)
- TxD (transmit data)

IRSCI I/O (input/output) lines are implemented by sharing parallel I/O port pins. The full name of an IRSCI input or output reflects the name of the shared port pin. **Table 13-1** shows the full names and the generic names of the IRSCI I/O pins.

The generic pin names appear in the text of this section.

Table 13-1. Pin Name Conventions

Generic Pin Names:	RxD	TxD
Full Pin Names:	PTB1/RxD	PTB0/TxD

13.5 IRSCI Module Overview

The IRSCI consists of a serial communications interface (SCI) and a infrared interface sub-module as shown in **Figure 13-2**.

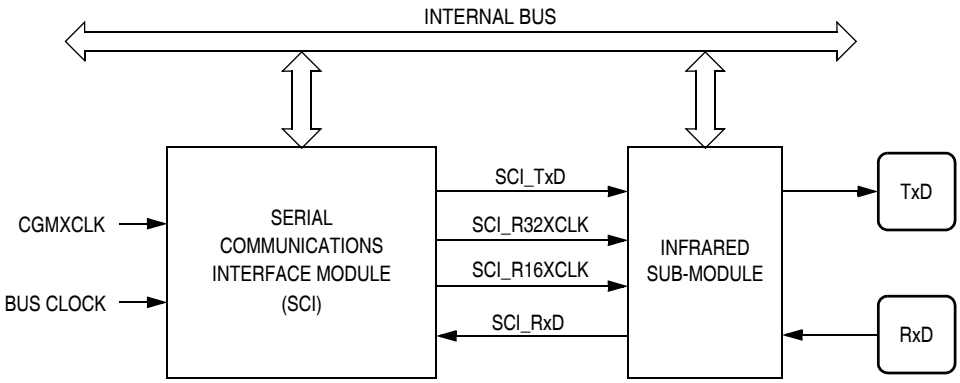


Figure 13-2. IRSCI Block Diagram

The SCI module provides serial data transmission and reception, with a programmable baud rate clock based on the bus clock or the CGMXCLK.

The infrared sub-module receives two clock sources from the SCI module: SCI_R16XCLK and SCI_R32XCLK. Both reference clocks are used to generate the narrow pulses during data transmission.
The SCI_R16XCLK and SCI_R32XCLK are internal clocks with frequencies that are 16 and 32 times the baud rate respectively. Both SCI_R16XCLK and SCI_R32XCLK clocks are used for transmitting data. The SCI_R16XCLK clock is used only for receiving data.

NOTE: *For proper SCI function (transmit or receive), the bus clock MUST be programmed to at least 32 times that of the selected baud rate.*
When the infrared sub-module is disabled, signals on the TxD and RxD pins pass through unchanged to the SCI module.

13.6 Infrared Functional Description

Figure 13-3 shows the structure of the infrared sub-module.

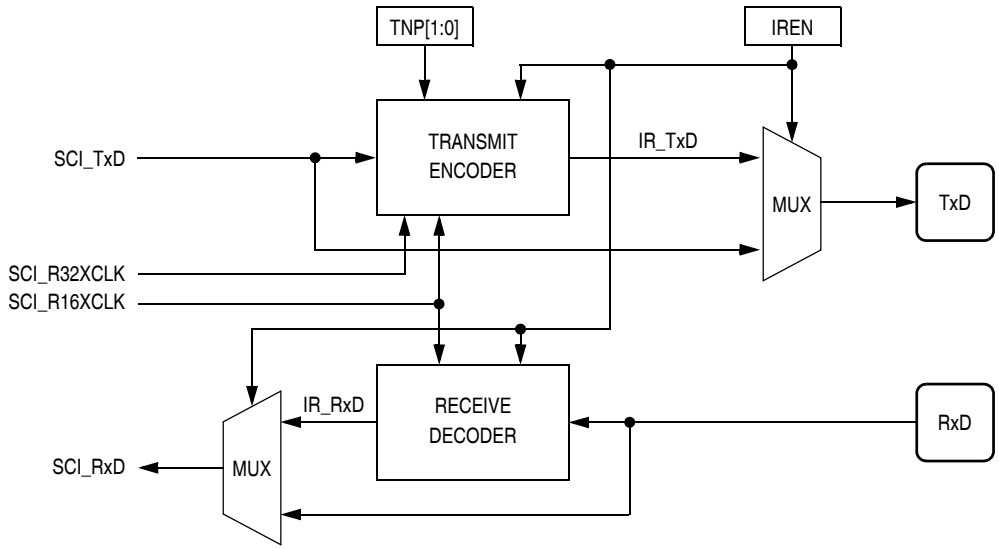


Figure 13-3. Infrared Sub-Module Diagram

The infrared sub-module provides the capability of transmitting narrow pulses to an infrared LED and receiving narrow pulses and transforming them to serial bits, which are sent to the SCI module. The infrared sub-module receives two clocks from the SCI. One of these two clocks is selected as the base clock to generate the 3/16, 1/16, or 1/32 bit width narrow pulses during transmission.

The sub-module consists of two main blocks: the transmit encoder and the receive decoder. When transmitting data, the SCI data stream is encoded by the infrared sub-module. For every "0" bit, a narrow "low" pulse is transmitted; no pulse is transmitted for "1" bits. When receiving data, the infrared pulses should be detected using an infrared photo diode for conversion to CMOS voltage levels before connecting to the RxD pin for the infrared decoder. The SCI data stream is reconstructed by stretching the "0" pulses.

13.6.1 Infrared Transmit Encoder

The infrared transmit encoder converts the "0" bits in the serial data stream from the SCI module to narrow "low" pulses, to the TxD pin. The narrow pulse is sent with a duration of 1/32, 1/16, or 3/16 of a data bit width**.** When two consecutive zeros are sent, the two consecutive narrow pulses will be separated by a time equal to a data bit width.

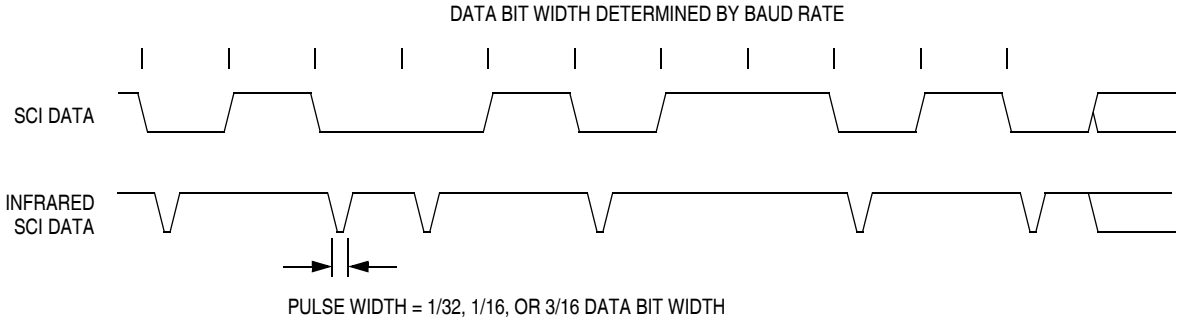


Figure 13-4. Infrared SCI Data Example

13.6.2 Infrared Receive Decoder

The infrared receive decoder converts low narrow pulses from the RxD pin to standard SCI data bits. The reference clock, SCI_R16XCLK, clocks a four bit internal counter which counts from 0 to 15. An incoming pulse starts the internal counter and a "0" is sent out to the IR_RxD output. Subsequent incoming pulses are ignored when the counter count is between 0 and 7; IR_RxD remains "0". Once the counter passes 7, an incoming pulse will reset the counter; IR_RxD remains "0". When the counter reaches 15, the IR_RxD output returns to "1", the counter stops and waits for further pulses. A pulse is interpreted as jitter if it arrives shortly after the counter reaches 15; IR_RxD remains "1".

13.7 SCI Functional Description

Figure 13-5 shows the structure of the SCI.

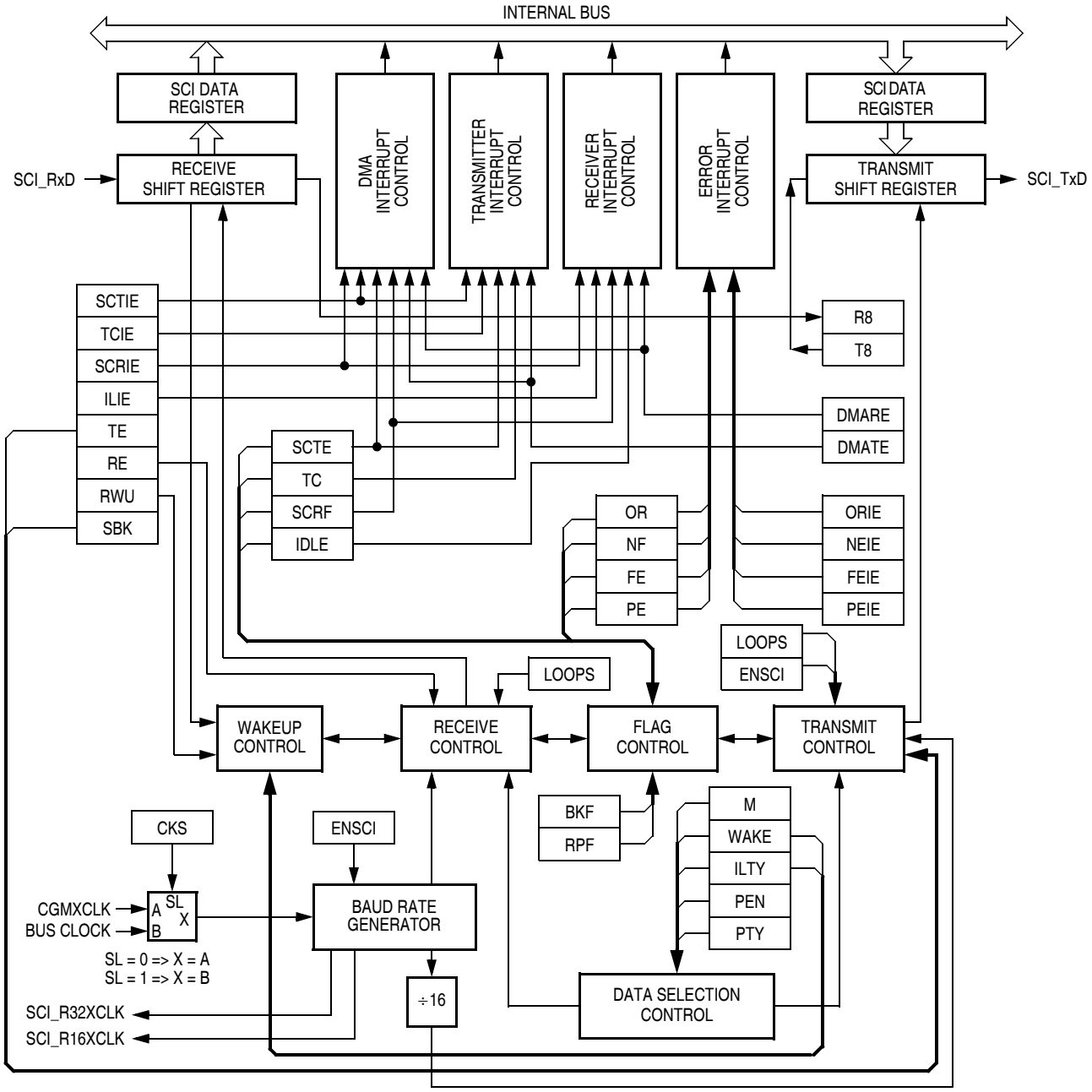


Figure 13-5. SCI Module Block Diagram

The SCI allows full-duplex, asynchronous, NRZ serial communication between the MCU and remote devices, including other MCUs. The transmitter and receiver of the SCI operate independently, although they use the same baud rate generator. During normal operation, the CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

NOTE: *For SCI operations, the IR sub-module is transparent to the SCI module. Data at going out of the SCI transmitter and data going into the SCI receiver is always in SCI format. It makes no difference to the SCI module whether the IR sub-module is enabled or disabled.*

NOTE: *This SCI module is a standard HC08 SCI module with the following modifications:*

- *A control bit, CKS, is added to the SCI baud rate control register to select between two input clocks for baud rate clock generation*
- *The TXINV bit is removed from the SCI control register 1*

13.7.1 Data Format

The SCI uses the standard non-return-to-zero mark/space data format illustrated in **Figure 13-6**.

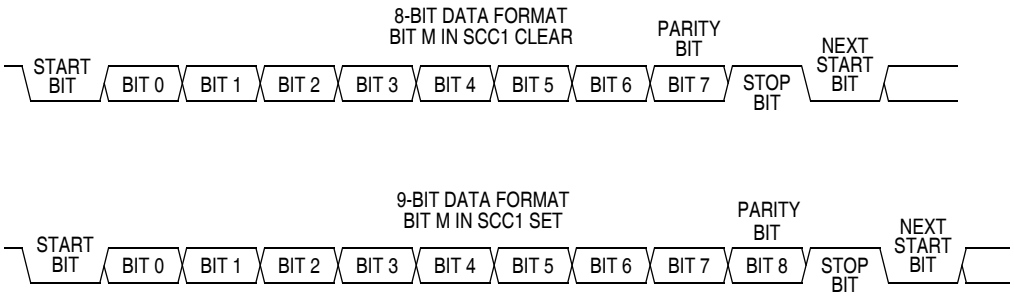


Figure 13-6. SCI Data Formats

13.7.2 Transmitter

Figure 13-7 shows the structure of the SCI transmitter.

The baud rate clock source for the SCI can be selected by the CKS bit, in the SCI baud rate register (see 13.11.7 SCI Baud Rate Register (SCBR)).

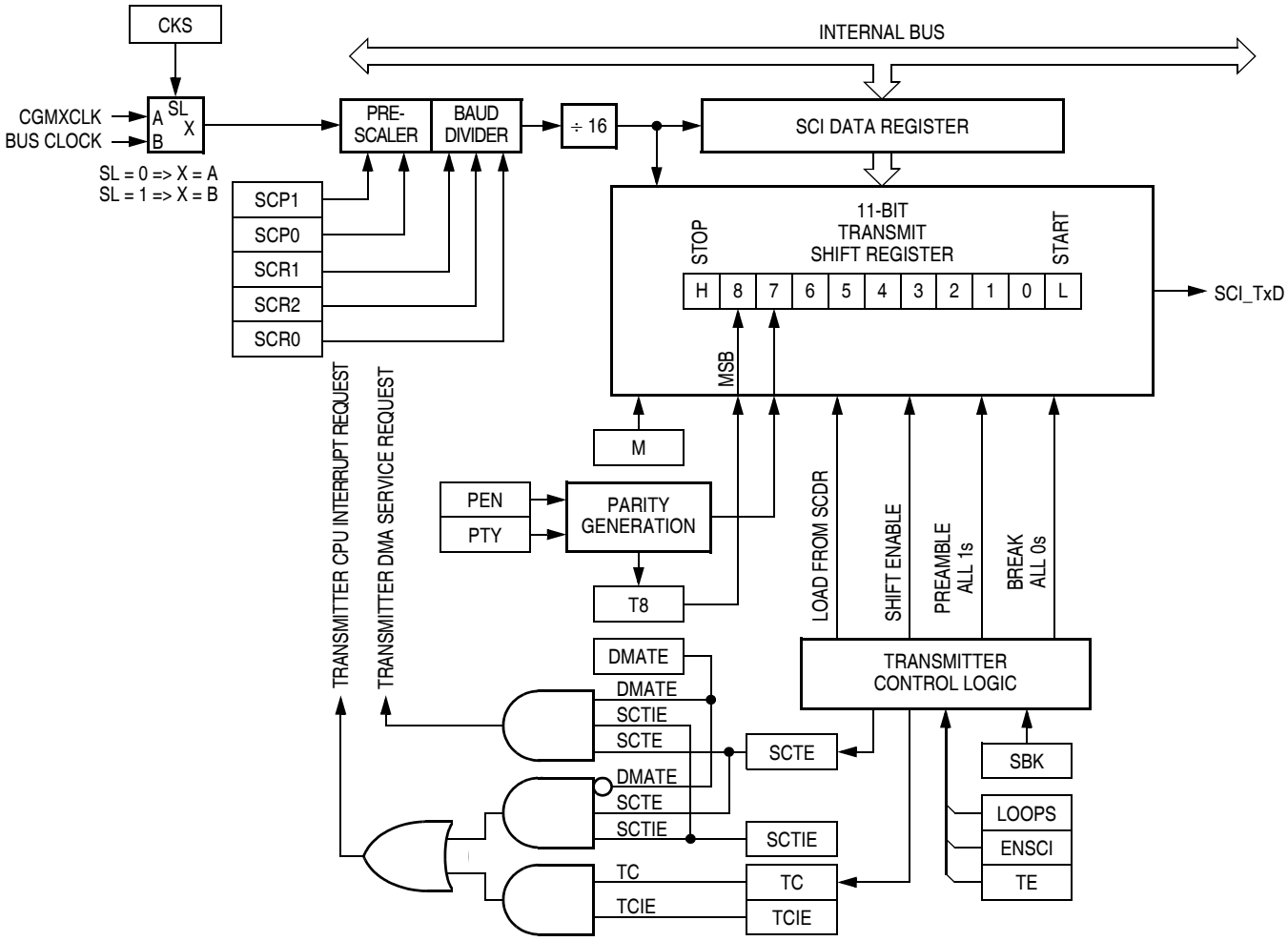


Figure 13-7. SCI Transmitter

13.7.2.1 Character Length

The transmitter can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCC1) determines character length. When transmitting 9-bit data, bit T8 in SCI control register 3 (SCC3) is the ninth bit (bit 8).

13.7.2.2 Character Transmission

During an SCI transmission, the transmit shift register shifts a character out to the TxD pin. The SCI data register (SCDR) is the write-only buffer between the internal data bus and the transmit shift register. To initiate an SCI transmission:

1. Enable the SCI by writing a logic 1 to the enable SCI bit (ENSCI) in SCI control register 1 (SCC1).
2. Enable the transmitter by writing a logic 1 to the transmitter enable bit (TE) in SCI control register 2 (SCC2).
3. Clear the SCI transmitter empty bit by first reading SCI status register 1 (SCS1) and then writing to the SCDR.
4. Repeat step 3 for each subsequent transmission.

At the start of a transmission, transmitter control logic automatically loads the transmit shift register with a preamble of logic 1s. After the preamble shifts out, control logic transfers the SCDR data into the transmit shift register. A logic 0 start bit automatically goes into the least significant bit position of the transmit shift register. A logic 1 stop bit goes into the most significant bit position.

The SCI transmitter empty bit, SCTE, in SCS1 becomes set when the SCDR transfers a byte to the transmit shift register. The SCTE bit indicates that the SCDR can accept new data from the internal data bus. If the SCI transmit interrupt enable bit, SCTIE, in SCC2 is also set, the SCTE bit generates a transmitter interrupt request.

When the transmit shift register is not transmitting a character, the TxD pin goes to the idle condition, logic 1. If at any time software clears the ENSCI bit in SCI control register 1 (SCC1), the transmitter and receiver relinquish control of the port pins.

13.7.2.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCC2 loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCC1. As long as SBK is at logic 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next character.

The SCI recognizes a break character when a start bit is followed by eight or nine logic 0 data bits and a logic 0 where the stop bit should be.

Receiving a break character has the following effects on SCI registers:

- Sets the framing error bit (FE) in SCS1
- Sets the SCI receiver full bit (SCRF) in SCS1
- Clears the SCI data register (SCDR)
- Clears the R8 bit in SCC3
- Sets the break flag bit (BKF) in SCS2
- May set the overrun (OR), noise flag (NF), parity error (PE), or reception in progress flag (RPF) bits

13.7.2.4 Idle Characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in SCC1. The preamble is a synchronizing idle character that begins every transmission.

If the TE bit is cleared during a transmission, the TxD pin becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues an idle character to be sent after the character currently being transmitted.

NOTE: *When queueing an idle character, return the TE bit to logic 1 before the stop bit of the current character shifts out to the TxD pin. Setting TE after the stop bit appears on TxD causes data previously written to the SCDR to be lost.*

Toggle the TE bit for a queued idle character when the SCTE bit becomes set and just before writing the next byte to the SCDR.

13.7.2.5 Transmitter Interrupts

The following conditions can generate CPU interrupt requests from the SCI transmitter:

- SCI transmitter empty (SCTE) — The SCTE bit in SCS1 indicates that the SCDR has transferred a character to the transmit shift register. SCTE can generate a transmitter CPU interrupt request. Setting the SCI transmit interrupt enable bit, SCTIE, in SCC2 enables the SCTE bit to generate transmitter CPU interrupt requests.
- Transmission complete (TC) — The TC bit in SCS1 indicates that the transmit shift register and the SCDR are empty and that no break or idle character has been generated. The transmission complete interrupt enable bit, TCIE, in SCC2 enables the TC bit to generate transmitter CPU interrupt requests.

13.7.3 Receiver

Figure 13-8 shows the structure of the SCI receiver.

13.7.3.1 Character Length

The receiver can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCC1) determines character length. When receiving 9-bit data, bit R8 in SCI control register 2 (SCC2) is the ninth bit (bit 8). When receiving 8-bit data, bit R8 is a copy of the eighth bit (bit 7).

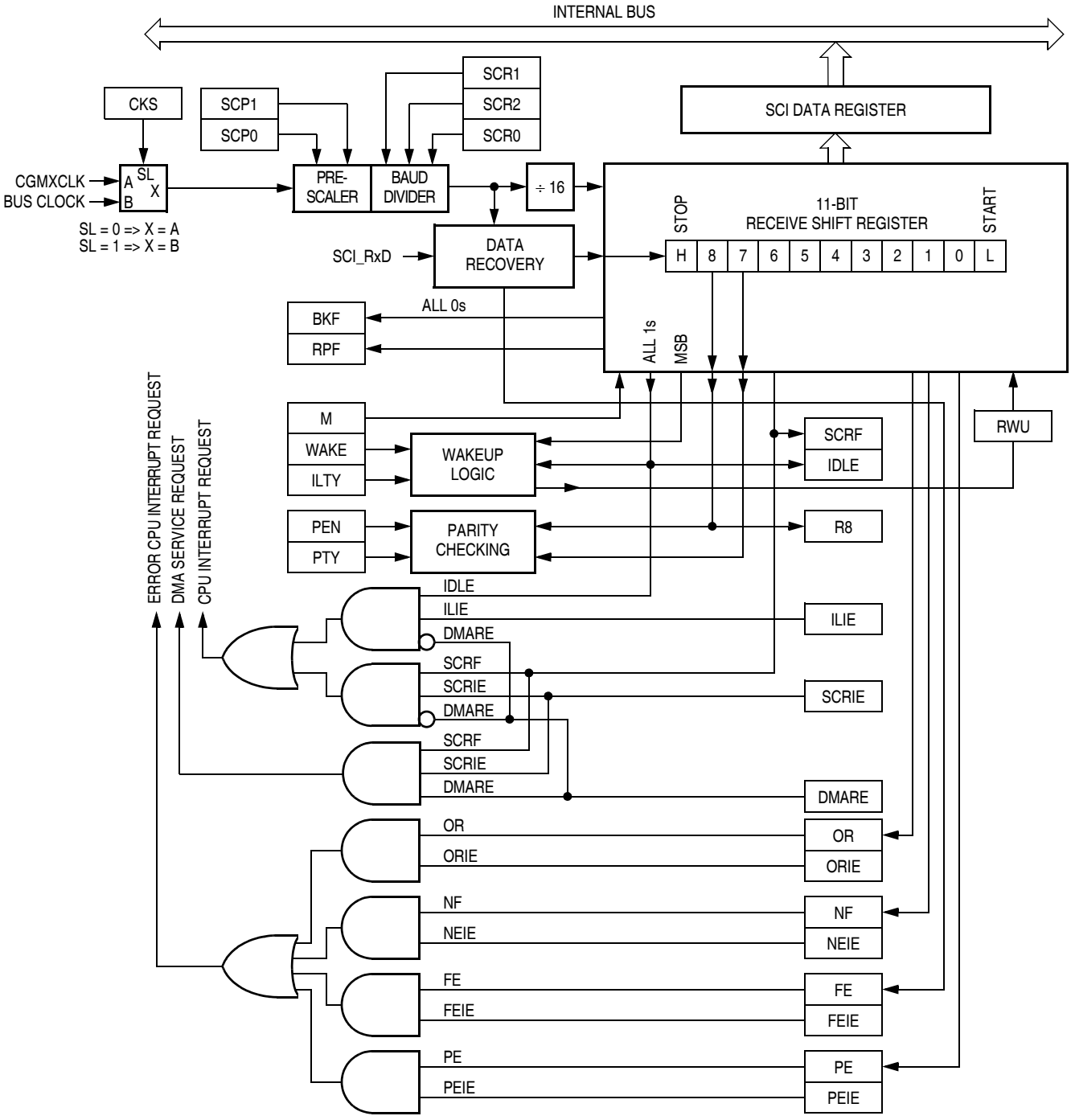


Figure 13-8. SCI Receiver Block Diagram

13.7.3.2 Character Reception

During an SCI reception, the receive shift register shifts characters in from the RxD pin. The SCI data register (SCDR) is the read-only buffer between the internal data bus and the receive shift register.

After a complete character shifts into the receive shift register, the data portion of the character transfers to the SCDR. The SCI receiver full bit, SCRF, in SCI status register 1 (SCS1) becomes set, indicating that the received byte can be read. If the SCI receive interrupt enable bit, SCRIE, in SCC2 is also set, the SCRF bit generates a receiver CPU interrupt request.

13.7.3.3 Data Sampling

The receiver samples the RxD pin at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock is resynchronized at the following times (see **Figure 13-9**):

- After every start bit
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0)

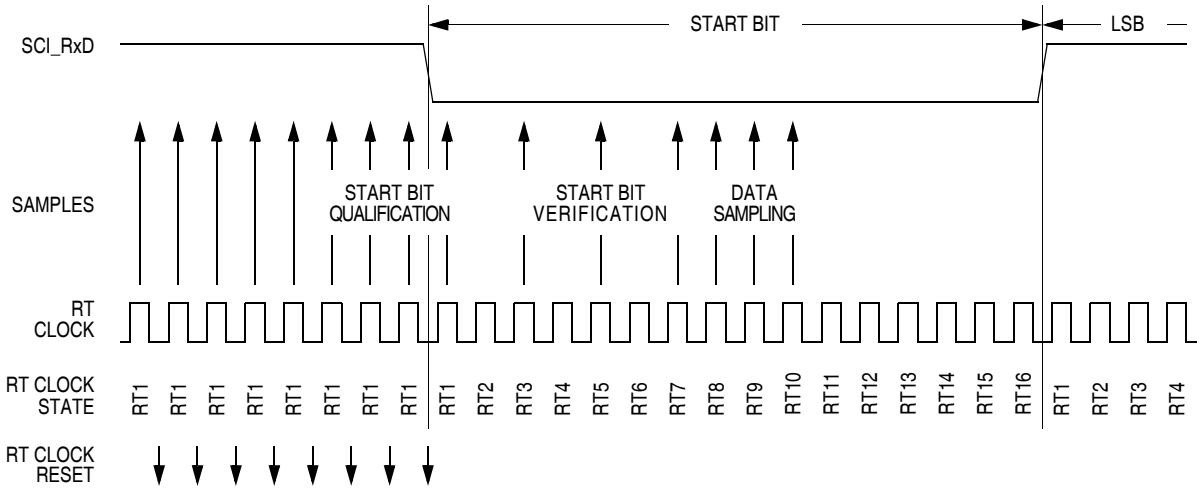


Figure 13-9. Receiver Data Sampling

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. **Table 13-2** summarizes the results of the start bit verification samples.

Table 13-2. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. **Table 13-3** summarizes the results of the data bit samples.

Table 13-3. Data Bit Recovery

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

NOTE: *The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit.*

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. **Table 13-4** summarizes the results of the stop bit samples.

Table 13-4. Stop Bit Recovery

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

13.7.3.4 Framing Errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming character, it sets the framing error bit, FE, in SCS1. The FE flag is set at the same time that the SCRF bit is set. A break character that has no stop bit also sets the FE bit.

13.7.3.5 Baud Rate Tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples to fall outside the actual stop bit. Then a noise error occurs. If more than one of the samples is outside the stop bit, a framing error occurs. In most applications, the baud rate tolerance is much more than the degree of misalignment that is likely to occur.

As the receiver samples an incoming character, it resynchronizes the RT clock on any valid falling edge within the character. Resynchronization within characters corrects misalignments between transmitter bit times and receiver bit times.

Slow Data Tolerance

Figure 13-10 shows how much a slow received character can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

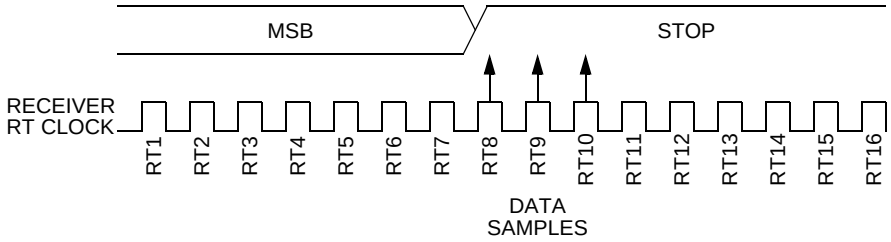


Figure 13-10. Slow Data

For an 8-bit character, data sampling of the stop bit takes the receiver 9 bit times × 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in Figure 13-10, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 9 bit times × 16 RT cycles + 3 RT cycles = 147 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit character with no errors is

$$\left|\frac{154 - 147}{154}\right| \times 100 = 4.54\%$$

For a 9-bit character, data sampling of the stop bit takes the receiver 10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 13-10, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 10 bit times × 16 RT cycles + 3 RT cycles = 163 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is

$$\left|\frac{170 - 163}{170}\right| \times 100 = 4.12\%$$

Fast Data Tolerance

Figure 13-11 shows how much a fast received character can be misaligned without causing a noise error or a framing error. The fast stop bit ends at RT10 instead of RT16 but is still there for the stop bit data samples at RT8, RT9, and RT10.

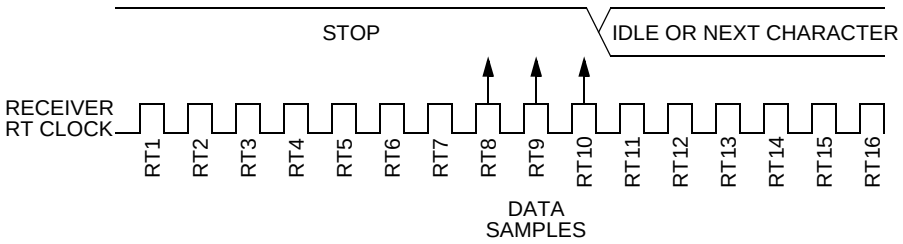


Figure 13-11. Fast Data

For an 8-bit character, data sampling of the stop bit takes the receiver 9 bit times × 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in **Figure 13-11**, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 10 bit times × 16 RT cycles = 160 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is

$$\left|\frac{154 - 160}{154}\right| \times 100 = 3.90\dot{}\%$$

For a 9-bit character, data sampling of the stop bit takes the receiver 10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in **Figure 13-11**, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 11 bit times × 16 RT cycles = 176 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is

$$\left|\frac{170 - 176}{170}\right| \times 100 = 3.53\%$$

13.7.3.6 Receiver Wakeup

So that the MCU can ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in SCC2 puts the receiver into a standby state during which receiver interrupts are disabled.

Depending on the state of the WAKE bit in SCC1, either of two conditions on the RxD pin can bring the receiver out of the standby state:

- Address mark — An address mark is a logic 1 in the most significant bit position of a received character. When the WAKE bit is set, an address mark wakes the receiver from the standby state by clearing the RWU bit. The address mark also sets the SCI receiver full bit, SCRF. Software can then compare the character containing the address mark to the user-defined address of the receiver. If they are the same, the receiver remains awake and processes the characters that follow. If they are not the same, software can set the RWU bit and put the receiver back into the standby state.
- Idle input line condition — When the WAKE bit is clear, an idle character on the RxD pin wakes the receiver from the standby state by clearing the RWU bit. The idle character that wakes the receiver does not set the receiver idle bit, IDLE, or the SCI receiver full bit, SCRF. The idle line type bit, ILTY, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit.

NOTE: *Clearing the WAKE bit after the RxD pin has been idle may cause the receiver to wake up immediately.*

13.7.3.7 Receiver Interrupts

The following sources can generate CPU interrupt requests from the SCI receiver:

- SCI receiver full (SCRF) — The SCRF bit in SCS1 indicates that the receive shift register has transferred a character to the SCDR. SCRF can generate a receiver interrupt request. Setting the SCI receive interrupt enable bit, SCRIE, in SCC2 enables the SCRF bit to generate receiver CPU interrupts.
- Idle input (IDLE) — The IDLE bit in SCS1 indicates that 10 or 11 consecutive logic 1s shifted in from the RxD pin. The idle line interrupt enable bit, ILIE, in SCC2 enables the IDLE bit to generate CPU interrupt requests.

13.7.3.8 Error Interrupts

The following receiver error flags in SCS1 can generate CPU interrupt requests:

- Receiver overrun (OR) — The OR bit indicates that the receive shift register shifted in a new character before the previous character was read from the SCDR. The previous character remains in the SCDR, and the new character is lost. The overrun interrupt enable bit, ORIE, in SCC3 enables OR to generate SCI error CPU interrupt requests.
- Noise flag (NF) — The NF bit is set when the SCI detects noise on incoming data or break characters, including start, data, and stop bits. The noise error interrupt enable bit, NEIE, in SCC3 enables NF to generate SCI error CPU interrupt requests.
- Framing error (FE) — The FE bit in SCS1 is set when a logic 0 occurs where the receiver expects a stop bit. The framing error interrupt enable bit, FEIE, in SCC3 enables FE to generate SCI error CPU interrupt requests.
- Parity error (PE) — The PE bit in SCS1 is set when the SCI detects a parity error in incoming data. The parity error interrupt enable bit, PEIE, in SCC3 enables PE to generate SCI error CPU interrupt requests.

13.8 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

13.8.1 Wait Mode

The SCI module remains active after the execution of a WAIT instruction. In wait mode, the SCI module registers are not accessible by the CPU. Any enabled CPU interrupt request from the SCI module can bring the MCU out of wait mode.

If SCI module functions are not required during wait mode, reduce power consumption by disabling the module before executing the WAIT instruction.

Refer to **9.7 Low-Power Modes** for information on exiting wait mode.

13.8.2 Stop Mode

The SCI module is inactive after the execution of a STOP instruction. The STOP instruction does not affect SCI register states. SCI module operation resumes after an external interrupt.

Because the internal clock is inactive during stop mode, entering stop mode during an SCI transmission or reception results in invalid data.

Refer to **9.7 Low-Power Modes** for information on exiting stop mode.

13.9 SCI During Break Module Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during interrupts generated by the break module. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state.

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a two-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at logic 0. After the break, doing the second step clears the status bit.

13.10 I/O Signals

The two IRSCI I/O pins are:

- PTB0/TxD — Transmit data
- PTB1/RxD — Receive data

13.10.1 PTB0/TxD (Transmit Data)

The PTB0/TxD pin is the serial data (standard or infrared) output from the SCI transmitter. The IRSCI shares the PTB0/TxD pin with port B. When the IRSCI is enabled, the PTB0/TxD pin is an output regardless of the state of the DDRB0 bit in data direction register B (DDRB). TxD pin has high current (15mA) sink capability when the LEDB0 bit is set in the port B LED control register ($000C).

13.10.2 PTB1/RxD (Receive Data)

The PTB1/RxD pin is the serial data input to the IRSCI receiver. The IRSCI shares the PTB1/RxD pin with port B. When the IRSCI is enabled, the PTB1/RxD pin is an input regardless of the state of the DDRB1 bit in data direction register B (DDRB).

Table 13-5 shows a summary of I/O pin functions when the SCI is enabled.

Table 13-5. SCI Pin Functions (Standard and Infrared)

SCC1 [ENSCI]	SCIRCR [IREN]	SCC2 [TE]	SCC2 [RE]	TxD Pin	RxD Pin
1	0	0	0	Hi-Z(1)	Input ignored (terminate externally)
1	0	0	1	Hi-Z(1)	Input sampled, pin should idle high
1	0	1	0	Output SCI (idle high)	Input ignored (terminate externally)
1	0	1	1	Output SCI (idle high)	Input sampled, pin should idle high
1	1	0	0	Hi-Z(1)	Input ignored (terminate externally)
1	1	0	1	Hi-Z(1)	Input sampled, pin should idle high
1	1	1	0	Output IR SCI (idle high)	Input ignored (terminate externally)
1	1	1	1	Output IR SCI (idle high)	Input sampled, pin should idle high
0	X	X	X	Pins under port control (standard I/O port)	

Notes:
1. After completion of transmission in progress.

13.11 I/O Registers

The following I/O registers control and monitor SCI operation:

- SCI control register 1 (SCC1)
- SCI control register 2 (SCC2)
- SCI control register 3 (SCC3)
- SCI status register 1 (SCS1)
- SCI status register 2 (SCS2)
- SCI data register (SCDR)
- SCI baud rate register (SCBR)
- SCI infrared control register (SCIRCR)

13.11.1 SCI Control Register 1

SCI control register:

- Enables loop mode operation
- Enables the SCI
- Controls output polarity
- Controls character length
- Controls SCI wakeup method
- Controls idle character detection
- Enables parity function
- Controls parity type

Address: $0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LOOPS	ENSCI	0	M	WAKE	ILTY	PEN	PTY
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 13-12. SCI Control Register 1 (SCC1)

LOOPS — Loop Mode Select Bit

This read/write bit enables loop mode operation for the SCI only. In loop mode the RxD pin is disconnected from the SCI, and the transmitter output goes into the receiver input. Both the transmitter and the receiver must be enabled to use loop mode. The infrared encoder/decoder is not in the loop. Reset clears the LOOPS bit.

1 = Loop mode enabled
0 = Normal operation enabled

ENSCI — Enable SCI Bit

This read/write bit enables the SCI and the SCI baud rate generator. Clearing ENSCI sets the SCTE and TC bits in SCI status register 1 and disables transmitter interrupts. Reset clears the ENSCI bit.

1 = SCI enabled
0 = SCI disabled

M — Mode (Character Length) Bit

This read/write bit determines whether SCI characters are eight or nine bits long. (See **Table 13-6**.) The ninth bit can serve as an extra stop bit, as a receiver wakeup signal, or as a parity bit. Reset clears the M bit.

1 = 9-bit SCI characters
0 = 8-bit SCI characters

WAKE — Wakeup Condition Bit

This read/write bit determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit position of a received character or an idle condition on the RxD pin. Reset clears the WAKE bit.

1 = Address mark wakeup
0 = Idle line wakeup

ILTY — Idle Line Type Bit

This read/write bit determines when the SCI starts counting logic 1s as idle character bits. The counting begins either after the start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit may cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. Reset clears the ILTY bit.

1 = Idle character bit count begins after stop bit
0 = Idle character bit count begins after start bit

PEN — Parity Enable Bit

This read/write bit enables the SCI parity function. (See **Table 13-6**.) When enabled, the parity function inserts a parity bit in the most significant bit position. (See **Figure 13-6**.) Reset clears the PEN bit.

1 = Parity function enabled
0 = Parity function disabled

PTY — Parity Bit

This read/write bit determines whether the SCI generates and checks for odd parity or even parity. (See **Table 13-6**.) Reset clears the PTY bit.

1 = Odd parity
0 = Even parity

NOTE: *Changing the PTY bit in the middle of a transmission or reception can generate a parity error.*

Table 13-6. Character Format Selection

Control Bits		Character Format				
M	**PEN:PTY**	**Start Bits**	**Data Bits**	**Parity**	**Stop Bits**	**Character Length**
0	0X	1	8	None	1	10 bits
1	0X	1	9	None	1	11 bits
0	10	1	7	Even	1	10 bits
0	11	1	7	Odd	1	10 bits
1	10	1	8	Even	1	11 bits
1	11	1	8	Odd	1	11 bits

13.11.2 SCI Control Register 2

SCI control register 2:

- Enables the following CPU interrupt requests:
  - Enables the SCTE bit to generate transmitter CPU interrupt requests
  - Enables the TC bit to generate transmitter CPU interrupt requests
  - Enables the SCRF bit to generate receiver CPU interrupt requests
  - Enables the IDLE bit to generate receiver CPU interrupt requests
- Enables the transmitter
- Enables the receiver
- Enables SCI wakeup
- Transmits SCI break characters

Address: $0014

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

Figure 13-13. SCI Control Register 2 (SCC2)

SCTIE — SCI Transmit Interrupt Enable Bit

This read/write bit enables the SCTE bit to generate SCI transmitter CPU interrupt requests. Reset clears the SCTIE bit.

1 = SCTE enabled to generate CPU interrupt
0 = SCTE not enabled to generate CPU interrupt

TCIE — Transmission Complete Interrupt Enable Bit

This read/write bit enables the TC bit to generate SCI transmitter CPU interrupt requests. Reset clears the TCIE bit.

1 = TC enabled to generate CPU interrupt requests
0 = TC not enabled to generate CPU interrupt requests

SCRIE — SCI Receive Interrupt Enable Bit

This read/write bit enables the SCRF bit to generate SCI receiver CPU interrupt requests. Reset clears the SCRIE bit.

1 = SCRF enabled to generate CPU interrupt
0 = SCRF not enabled to generate CPU interrupt

ILIE — Idle Line Interrupt Enable Bit

This read/write bit enables the IDLE bit to generate SCI receiver CPU interrupt requests. Reset clears the ILIE bit.

1 = IDLE enabled to generate CPU interrupt requests
0 = IDLE not enabled to generate CPU interrupt requests

TE — Transmitter Enable Bit

Setting this read/write bit begins the transmission by sending a preamble of 10 or 11 logic 1s from the transmit shift register to the TxD pin. If software clears the TE bit, the transmitter completes any transmission in progress before the TxD returns to the idle condition

(logic 1). Clearing and then setting TE during a transmission queues an idle character to be sent after the character currently being transmitted. Reset clears the TE bit.

1 = Transmitter enabled
0 = Transmitter disabled

NOTE: *Writing to the TE bit is not allowed when the enable SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.*

RE — Receiver Enable Bit

Setting this read/write bit enables the receiver. Clearing the RE bit disables the receiver but does not affect receiver interrupt flag bits. Reset clears the RE bit.

1 = Receiver enabled
0 = Receiver disabled

NOTE: *Writing to the RE bit is not allowed when the enable SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.*

RWU — Receiver Wakeup Bit

This read/write bit puts the receiver in a standby state during which receiver interrupts are disabled. The WAKE bit in SCC1 determines whether an idle input or an address mark brings the receiver out of the standby state and clears the RWU bit. Reset clears the RWU bit.

1 = Standby state
0 = Normal operation

SBK — Send Break Bit

Setting and then clearing this read/write bit transmits a break character followed by a logic 1. The logic 1 after the break character guarantees recognition of a valid start bit. If SBK remains set, the transmitter continuously transmits break characters with no logic 1s between them. Reset clears the SBK bit.

1 = Transmit break characters
0 = No break characters being transmitted

NOTE: *Do not toggle the SBK bit immediately after setting the SCTE bit. Toggling SBK before the preamble begins causes the SCI to send a break character instead of a preamble.*

13.11.3 SCI Control Register 3

SCI control register 3:

- Stores the ninth SCI data bit received and the ninth SCI data bit to be transmitted
- Enables the following interrupts:
  - Receiver overrun interrupts
  - Noise error interrupts
  - Framing error interrupts
  - Parity error interrupts

Address: $0015

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

= Unimplemented U = Unaffected

Figure 13-14. SCI Control Register 3 (SCC3)

R8 — Received Bit 8

When the SCI is receiving 9-bit characters, R8 is the read-only ninth bit (bit 8) of the received character. R8 is received at the same time that the SCDR receives the other 8 bits.
When the SCI is receiving 8-bit characters, R8 is a copy of the eighth bit (bit 7). Reset has no effect on the R8 bit.

T8 — Transmitted Bit 8

When the SCI is transmitting 9-bit characters, T8 is the read/write ninth bit (bit 8) of the transmitted character. T8 is loaded into the transmit shift register at the same time that the SCDR is loaded into the transmit shift register. Reset has no effect on the T8 bit.

DMARE — DMA Receive Enable Bit

CAUTION: *The DMA module is not included on this MCU. Writing a logic 1 to DMARE or DMATE may adversely affect MCU performance.*

1 = DMA not enabled to service SCI receiver DMA service requests generated by the SCRF bit (SCI receiver CPU interrupt requests enabled)
0 = DMA not enabled to service SCI receiver DMA service requests generated by the SCRF bit (SCI receiver CPU interrupt requests enabled)

DMATE — DMA Transfer Enable Bit

CAUTION: *The DMA module is not included on this MCU. Writing a logic 1 to DMARE or DMATE may adversely affect MCU performance.*

1 = SCTE DMA service requests enabled; SCTE CPU interrupt requests disabled
0 = SCTE DMA service requests disabled; SCTE CPU interrupt requests enabled

ORIE — Receiver Overrun Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the receiver overrun bit, OR. Reset clears ORIE.
1 = SCI error CPU interrupt requests from OR bit enabled
0 = SCI error CPU interrupt requests from OR bit disabled

NEIE — Receiver Noise Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the noise error bit, NE. Reset clears NEIE.
1 = SCI error CPU interrupt requests from NE bit enabled
0 = SCI error CPU interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the framing error bit, FE. Reset clears FEIE.
1 = SCI error CPU interrupt requests from FE bit enabled
0 = SCI error CPU interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables SCI receiver CPU interrupt requests generated by the parity error bit, PE. (See **13.11.4 SCI Status Register 1**.) Reset clears PEIE.

1 = SCI error CPU interrupt requests from PE bit enabled
0 = SCI error CPU interrupt requests from PE bit disabled

13.11.4 SCI Status Register 1

SCI status register 1 contains flags to signal these conditions:

- Transfer of SCDR data to transmit shift register complete
- Transmission complete
- Transfer of receive shift register data to SCDR complete
- Receiver input idle
- Receiver overrun
- Noisy data
- Framing error
- Parity error

Address: $0016

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
Write:								
Reset:	1	1	0	0	0	0	0	0

= Unimplemented

Figure 13-15. SCI Status Register 1 (SCS1)

SCTE — SCI Transmitter Empty Bit

This clearable, read-only bit is set when the SCDR transfers a character to the transmit shift register. SCTE can generate an SCI transmitter CPU interrupt request. When the SCTIE bit in SCC2 is set, SCTE generates an SCI transmitter CPU interrupt request. In normal

operation, clear the SCTE bit by reading SCS1 with SCTE set and then writing to SCDR. Reset sets the SCTE bit.

1 = SCDR data transferred to transmit shift register
0 = SCDR data not transferred to transmit shift register

TC — Transmission Complete Bit

This read-only bit is set when the SCTE bit is set, and no data, preamble, or break character is being transmitted. TC generates an SCI transmitter CPU interrupt request if the TCIE bit in SCC2 is also set. TC is automatically cleared when data, preamble or break is queued and ready to be sent. There may be up to 1.5 transmitter clocks of latency between queueing data, preamble, and break and the transmission actually starting. Reset sets the TC bit.

1 = No transmission in progress
0 = Transmission in progress

SCRF — SCI Receiver Full Bit

This clearable, read-only bit is set when the data in the receive shift register transfers to the SCI data register. SCRF can generate an SCI receiver CPU interrupt request. When the SCRIE bit in SCC2 is set, SCRF generates a CPU interrupt request. In normal operation, clear the SCRF bit by reading SCS1 with SCRF set and then reading the SCDR. Reset clears SCRF.

1 = Received data available in SCDR
0 = Data not available in SCDR

IDLE — Receiver Idle Bit

This clearable, read-only bit is set when 10 or 11 consecutive logic 1s appear on the receiver input. IDLE generates an SCI error CPU interrupt request if the ILIE bit in SCC2 is also set. Clear the IDLE bit by reading SCS1 with IDLE set and then reading the SCDR. After the receiver is enabled, it must receive a valid character that sets the SCRF bit before an idle condition can set the IDLE bit. Also, after the IDLE bit has been cleared, a valid character must again set the SCRF bit before an idle condition can set the IDLE bit. Reset clears the IDLE bit.

1 = Receiver input idle
0 = Receiver input active (or idle since the IDLE bit was cleared)

OR — Receiver Overrun Bit

This clearable, read-only bit is set when software fails to read the SCDR before the receive shift register receives the next character. The OR bit generates an SCI error CPU interrupt request if the ORIE bit in SCC3 is also set. The data in the shift register is lost, but the data already in the SCDR is not affected. Clear the OR bit by reading SCS1 with OR set and then reading the SCDR. Reset clears the OR bit.

1 = Receive shift register full and SCRF = 1
0 = No receiver overrun

Software latency may allow an overrun to occur between reads of SCS1 and SCDR in the flag-clearing sequence. **Figure 13-16** shows the normal flag-clearing sequence and an example of an overrun caused by a delayed flag-clearing sequence. The delayed read of SCDR does not clear the OR bit because OR was not set when SCS1 was read. Byte 2 caused the overrun and is lost. The next flag-clearing sequence reads byte 3 in the SCDR instead of byte 2.

In applications that are subject to software latency or in which it is important to know which byte is lost due to an overrun, the flag-clearing routine can check the OR bit in a second read of SCS1 after reading the data register.

NF — Receiver Noise Flag Bit

This clearable, read-only bit is set when the SCI detects noise on the RxD pin. NF generates an NF CPU interrupt request if the NEIE bit in SCC3 is also set. Clear the NF bit by reading SCS1 and then reading the SCDR. Reset clears the NF bit.

1 = Noise detected
0 = No noise detected

FE — Receiver Framing Error Bit

This clearable, read-only bit is set when a logic 0 is accepted as the stop bit. FE generates an SCI error CPU interrupt request if the FEIE bit in SCC3 also is set. Clear the FE bit by reading SCS1 with FE set and then reading the SCDR. Reset clears the FE bit.

1 = Framing error detected
0 = No framing error detected

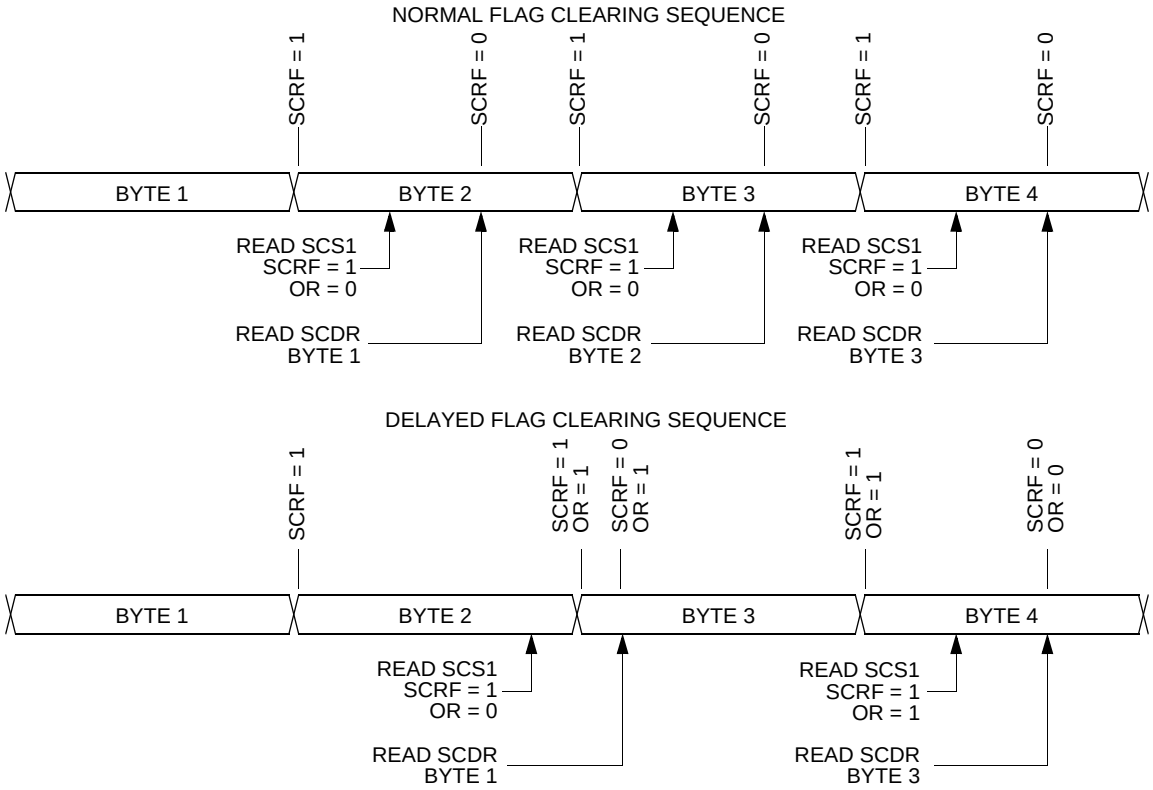


Figure 13-16. Flag Clearing Sequence

PE — Receiver Parity Error Bit

This clearable, read-only bit is set when the SCI detects a parity error in incoming data. PE generates a PE CPU interrupt request if the PEIE bit in SCC3 is also set. Clear the PE bit by reading SCS1 with PE set and then reading the SCDR. Reset clears the PE bit.

1 = Parity error detected

0 = No parity error detected

13.11.5 SCI Status Register 2 (SCS2)

SCI status register 2 contains flags to signal the following conditions:

- Break character detected
- Incoming data

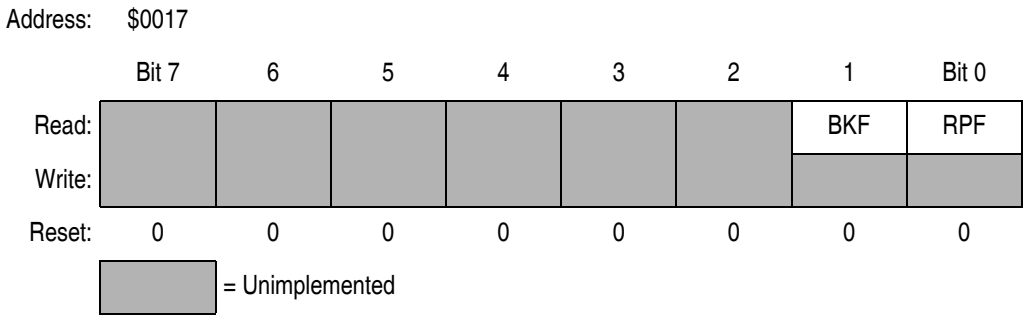


Figure 13-17. SCI Status Register 2 (SCS2)

BKF — Break Flag Bit

This clearable, read-only bit is set when the SCI detects a break character on the RxD pin. In SCS1, the FE and SCRF bits are also set. In 9-bit character transmissions, the R8 bit in SCC3 is cleared. BKF does not generate a CPU interrupt request. Clear BKF by reading SCS2 with BKF set and then reading the SCDR. Once cleared, BKF can become set again only after logic 1s again appear on the RxD pin followed by another break character. Reset clears the BKF bit.

1 = Break character detected
0 = No break character detected

RPF — Reception in Progress Flag Bit

This read-only bit is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RPF does not generate an interrupt request. RPF is reset after the receiver detects false start bits (usually from noise or a baud rate mismatch) or when the receiver detects an idle character. Polling RPF before disabling the SCI module or entering stop mode can show whether a reception is in progress.

1 = Reception in progress
0 = No reception in progress

13.11.6 SCI Data Register (SCDR)

The SCI data register is the buffer between the internal data bus and the receive and transmit shift registers. Reset has no effect on data in the SCI data register.

Address: $0018

	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:	Unaffected by reset							

Figure 13-18. SCI Data Register (SCDR)

R7/T7–R0/T0 — Receive/Transmit Data Bits

Reading the SCDR accesses the read-only received data bits, R7–R0. Writing to the SCDR writes the data to be transmitted, T7–T0. Reset has no effect on SCDR.

NOTE: *Do not use read/modify/write instructions on the SCI data register.*

13.11.7 SCI Baud Rate Register (SCBR)

The baud rate register selects the baud rate for both the receiver and the transmitter.

Address: $0019

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CKS	0	SCP1	SCP0	R	SCR2	SCR1	SCR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 13-19. SCI Baud Rate Register (SCBR)

CKS — Baud Clock Input Select

This read/write bit selects the source clock for the baud rate generator. Reset clears the CKS bit, selecting CGMXCLK.

1 = Bus clock drives the baud rate generator
0 = CGMXCLK drives the baud rate generator

SCP1 and SCP0 — SCI Baud Rate Prescaler Bits

These read/write bits select the baud rate prescaler divisor as shown in **Table 13-7**. Reset clears SCP1 and SCP0.

Table 13-7. SCI Baud Rate Prescaling

SCP1 and SCP0	Prescaler Divisor (PD)
00	1
01	3
10	4
11	13

SCR2–SCR0 — SCI Baud Rate Select Bits

These read/write bits select the SCI baud rate divisor as shown in **Table 13-8**. Reset clears SCR2–SCR0.

Table 13-8. SCI Baud Rate Selection

SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

Use this formula to calculate the SCI baud rate:

$$\text{baud rate} = \frac{\text{SCI clock source}}{16 \times PD \times BD}$$

where:

SCI clock source = f_{BUS} or CGMXCLK
(selected by CKS bit)
PD = prescaler divisor
BD = baud rate divisor

Table 13-9 shows the SCI baud rates that can be generated with a 4.9152-MHz bus clock when f_{BUS} is selected as SCI clock source.

Table 13-9. SCI Baud Rate Selection Examples

SCP1 and SCP0	Prescaler Divisor (PD)	SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)	Baud Rate (f_{BUS} = 4.9152 MHz)
00	1	000	1	—
00	1	001	2	—
00	1	010	4	76800
00	1	011	8	38400
00	1	100	16	19200
00	1	101	32	9600
00	1	110	64	4800
00	1	111	128	2400
01	3	000	1	—
01	3	001	2	51200
01	3	010	4	25600
01	3	011	8	12800
01	3	100	16	6400
01	3	101	32	3200
01	3	110	64	1600
01	3	111	128	800
10	4	000	1	76800
10	4	001	2	38400
10	4	010	4	19200
10	4	011	8	9600
10	4	100	16	4800
10	4	101	32	2400
10	4	110	64	1200
10	4	111	128	600
11	13	000	1	23632
11	13	001	2	11816
11	13	010	4	5908
11	13	011	8	2954
11	13	100	16	1477
11	13	101	32	739
11	13	110	64	369
11	13	111	128	185

13.11.8 SCI Infrared Control Register

The infrared control register contains the control bits for the infrared sub-module.

- Enables the infrared sub-module
- Selects the infrared transmitter narrow pulse width

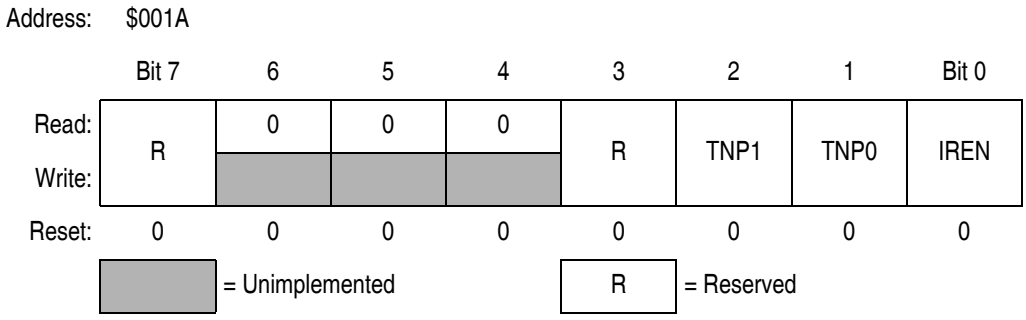


Figure 13-20. SCI Infrared Control Register (SCIRCR)

TNP1 and TNP0 — Transmitter Narrow Pulse Bits

These read/write bits select the infrared transmitter narrow pulse width as shown in **Table 13-10**. Reset clears TNP1 and TNP0.

Table 13-10. Infrared Narrow Pulse Selection

TNP1 and TNP0	Prescaler Divisor (PD)
00	SCI transmits a 3/16 narrow pulse
01	SCI transmits a 1/16 narrow pulse
10	SCI transmits a 1/32 narrow pulse
11	

IREN — Infrared Enable Bit

This read/write bit enables the infrared sub-module for encoding and decoding the SCI data stream. When this bit is clear, the infrared sub-module is disabled. Reset clears the IREN bit.

1 = infrared sub-module enabled
0 = infrared sub-module disabled

Section 14. Serial Peripheral Interface Module (SPI)

14.1 Contents

14.2 Introduction 270
14.3 Features 270
14.4 Pin Name Conventions and I/O Register Addresses 271
14.5 Functional Description 271
14.5.1 Master Mode 273
14.5.2 Slave Mode 274
14.6 Transmission Formats 275
14.6.1 Clock Phase and Polarity Controls 275
14.6.2 Transmission Format When CPHA = 0 276
14.6.3 Transmission Format When CPHA = 1 278
14.6.4 Transmission Initiation Latency 279
14.7 Queuing Transmission Data 281
14.8 Error Conditions 282
14.8.1 Overflow Error 282
14.8.2 Mode Fault Error 284
14.9 Interrupts 286
14.10 Resetting the SPI 288
14.11 Low-Power Modes 289
14.11.1 Wait Mode 289
14.11.2 Stop Mode 289
14.12 SPI During Break Interrupts 290
14.13 I/O Signals 290
14.13.1 MISO (Master In/Slave Out) 291
14.13.2 MOSI (Master Out/Slave In) 291

14.13.3 SPSCK (Serial Clock) 292
14.13.4 $\overline{SS}$ (Slave Select) 292
14.13.5 CGND (Clock Ground) 293

14.14 I/O Registers 294
14.14.1 SPI Control Register 294
14.14.2 SPI Status and Control Register 296
14.14.3 SPI Data Register 299

14.2 Introduction

This section describes the serial peripheral interface (SPI) module, which allows full-duplex, synchronous, serial communications with peripheral devices.

14.3 Features

Features of the SPI module include the following:

- Full-duplex operation
- Master and slave modes
- Double-buffered operation with separate transmit and receive registers
- Four master mode frequencies (maximum = bus frequency ÷ 2)
- Maximum slave mode frequency = bus frequency
- Serial clock with programmable polarity and phase
- Two separately enabled interrupts:
  - SPRF (SPI receiver full)
  - SPTE (SPI transmitter empty)
- Mode fault error flag with CPU interrupt capability
- Overflow error flag with CPU interrupt capability
- Programmable wired-OR mode
- I^2C (inter-integrated circuit) compatibility
- I/O (input/output) port bit(s) software configurable with pullup device(s) if configured as input port bit(s)

14.4 Pin Name Conventions and I/O Register Addresses

The text that follows describes the SPI. The SPI I/O pin names are $\overline{SS}$ (slave select), SPSCK (SPI serial clock), CGND (clock ground), MOSI (master out slave in), and MISO (master in/slave out). The SPI shares four I/O pins with four parallel I/O ports.

The full names of the SPI I/O pins are shown in **Table 14-1**. The generic pin names appear in the text that follows.

Table 14-1. Pin Name Conventions

SPI Generic Pin Names:		MISO	MOSI	$\overline{SS}$	SPSCK	CGND
Full SPI Pin Names:	SPI	PTD1/MISO	PTD2/MOSI	PTD0/$\overline{SS}$	PTD3/SPSCK	V_{SS}

Figure 14-1 summarizes the SPI I/O registers.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0010	SPI Control Register (SPCR)	Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Write:								
		Reset:	0	0	1	0	1	0	0	0
$0011	SPI Status and Control Register (SPSCR)	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	1	0	0	0
$0012	SPI Data Register (SPDR)	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							

= Unimplemented R = Reserved

Figure 14-1. SPI I/O Register Summary

14.5 Functional Description

Figure 14-2 shows the structure of the SPI module.

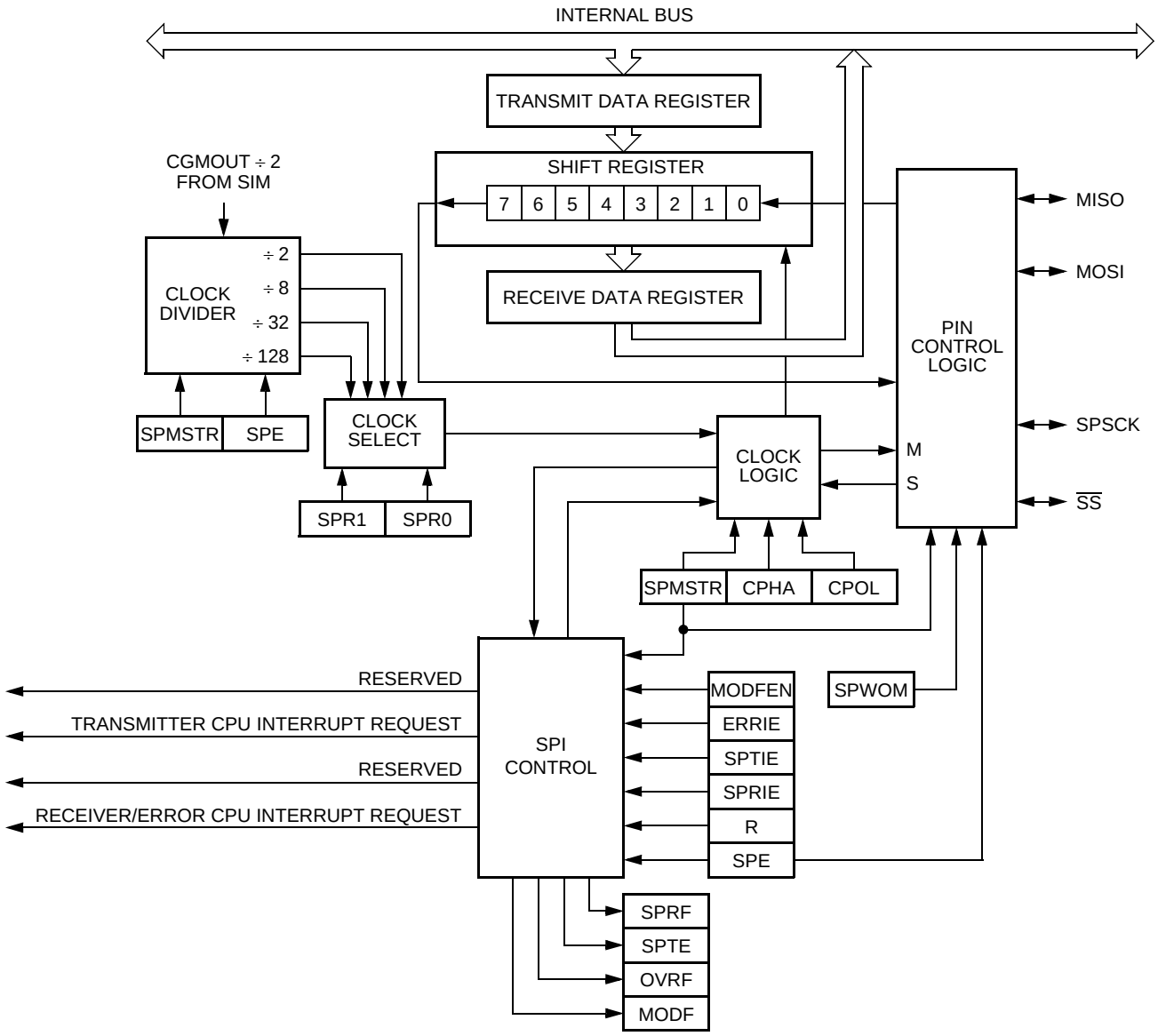


Figure 14-2. SPI Module Block Diagram

The SPI module allows full-duplex, synchronous, serial communication between the MCU and peripheral devices, including other MCUs. Software can poll the SPI status flags or SPI operation can be interrupt-driven.

The following paragraphs describe the operation of the SPI module.

14.5.1 Master Mode

The SPI operates in master mode when the SPI master bit, SPMSTR, is set.

NOTE: *Configure the SPI modules as master or slave before enabling them. Enable the master SPI before enabling the slave SPI. Disable the slave SPI before disabling the master SPI. (See **14.14.1 SPI Control Register**.)*

Only a master SPI module can initiate transmissions. Software begins the transmission from a master SPI module by writing to the transmit data register. If the shift register is empty, the byte immediately transfers to the shift register, setting the SPI transmitter empty bit, SPTE. The byte begins shifting out on the MOSI pin under the control of the serial clock. (See **Figure 14-3**.)

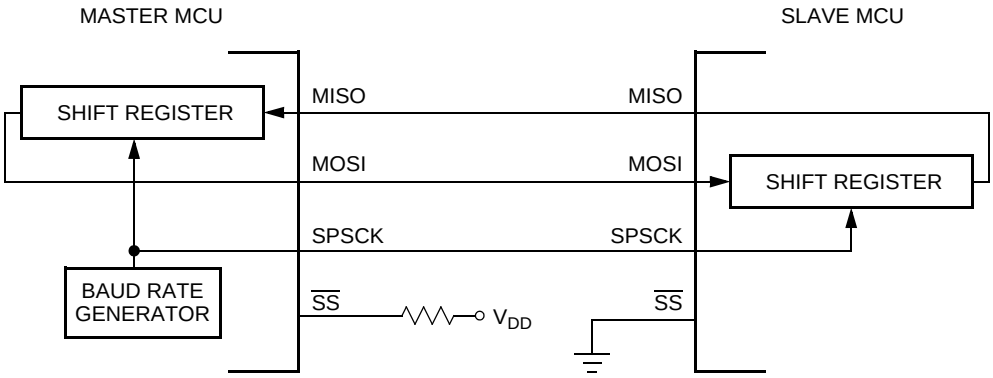


Figure 14-3. Full-Duplex Master-Slave Connections

The SPR1 and SPR0 bits control the baud rate generator and determine the speed of the shift register. (See **14.14.2 SPI Status and Control Register**.) Through the SPSCK pin, the baud rate generator of the master also controls the shift register of the slave peripheral.

As the byte shifts out on the MOSI pin of the master, another byte shifts in from the slave on the master's MISO pin. The transmission ends when the receiver full bit, SPRF, becomes set. At the same time that SPRF becomes set, the byte from the slave transfers to the receive data register. In normal operation, SPRF signals the end of a transmission. Software clears SPRF by reading the SPI status and control register with SPRF set and then reading the SPI data register. Writing to the SPI data register clears the SPTE bit.

14.5.2 Slave Mode

The SPI operates in slave mode when the SPMSTR bit is clear. In slave mode, the SPSCK pin is the input for the serial clock from the master MCU. Before a data transmission occurs, the $\overline{SS}$ pin of the slave SPI must be at logic 0. $\overline{SS}$ must remain low until the transmission is complete. (See **14.8.2 Mode Fault Error**.)

In a slave SPI module, data enters the shift register under the control of the serial clock from the master SPI module. After a byte enters the shift register of a slave SPI, it transfers to the receive data register, and the SPRF bit is set. To prevent an overflow condition, slave software then must read the receive data register before another full byte enters the shift register.

The maximum frequency of the SPSCK for an SPI configured as a slave is the bus clock speed (which is twice as fast as the fastest master SPSCK clock that can be generated). The frequency of the SPSCK for an SPI configured as a slave does not have to correspond to any SPI baud rate. The baud rate only controls the speed of the SPSCK generated by an SPI configured as a master. Therefore, the frequency of the SPSCK for an SPI configured as a slave can be any frequency less than or equal to the bus speed.

When the master SPI starts a transmission, the data in the slave shift register begins shifting out on the MISO pin. The slave can load its shift register with a new byte for the next transmission by writing to its transmit data register. The slave must write to its transmit data register at least one bus cycle before the master starts the next transmission. Otherwise, the byte already in the slave shift register shifts out on the MISO pin. Data written to the slave shift register during a transmission remains in a buffer until the end of the transmission.

When the clock phase bit (CPHA) is set, the first edge of SPSCK starts a transmission. When CPHA is clear, the falling edge of $\overline{SS}$ starts a transmission. (See **14.6 Transmission Formats**.)

NOTE: *SPSCK must be in the proper idle state before the slave is enabled to prevent SPSCK from appearing as a clock edge.*

14.6 Transmission Formats

During an SPI transmission, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). A serial clock synchronizes shifting and sampling on the two serial data lines. A slave select line allows selection of an individual slave SPI device; slave devices that are not selected do not interfere with SPI bus activities. On a master SPI device, the slave select line can optionally be used to indicate multiple-master bus contention.

14.6.1 Clock Phase and Polarity Controls

Software can select any of four combinations of serial clock (SPSCK) phase and polarity using two bits in the SPI control register (SPCR). The clock polarity is specified by the CPOL control bit, which selects an active high or low clock and has no significant effect on the transmission format.

The clock phase (CPHA) control bit selects one of two fundamentally different transmission formats. The clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity are changed between transmissions to allow a master device to communicate with peripheral slaves having different requirements.

NOTE: *Before writing to the CPOL bit or the CPHA bit, disable the SPI by clearing the SPI enable bit (SPE).*

14.6.2 Transmission Format When CPHA = 0

Figure 14-4 shows an SPI transmission in which CPHA is logic 0. The figure should not be used as a replacement for data sheet parametric information.

Two waveforms are shown for SPSCK: one for CPOL = 0 and another for CPOL = 1. The diagram may be interpreted as a master or slave timing diagram since the serial clock (SPSCK), master in/slave out (MISO), and master out/slave in (MOSI) pins are directly connected between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{SS}$ line is the slave select input to the slave. The slave SPI drives its MISO output only when its slave select input ($\overline{SS}$) is at logic 0, so that only the selected slave drives to the master. The $\overline{SS}$ pin of the master is not shown but is assumed to be inactive. The $\overline{SS}$ pin of the master must be high or must be reconfigured as general-purpose I/O not affecting the SPI. (See **14.8.2 Mode Fault Error**.) When CPHA = 0, the first SPSCK edge is the MSB capture strobe. Therefore, the slave must begin driving its data before the first SPSCK edge, and a falling edge on the $\overline{SS}$ pin is used to start the slave data transmission. The slave's $\overline{SS}$ pin must be toggled back to high and then low again between each byte transmitted as shown in **Figure 14-5**.

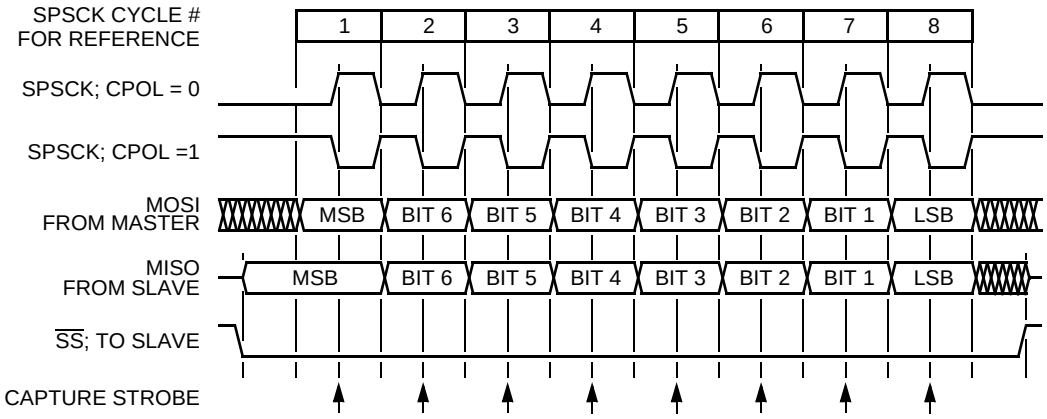


Figure 14-4. Transmission Format (CPHA = 0)

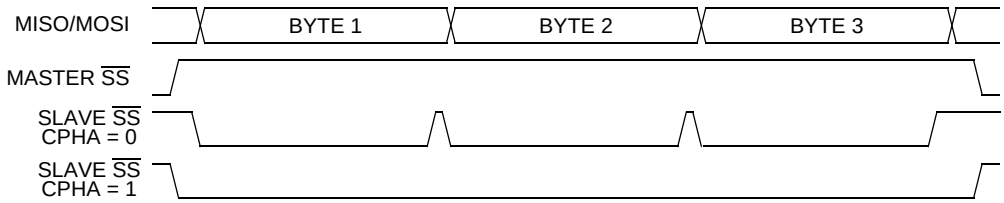


Figure 14-5. CPHA/$\overline{SS}$ Timing

When CPHA = 0 for a slave, the falling edge of $\overline{SS}$ indicates the beginning of the transmission. This causes the SPI to leave its idle state and begin driving the MISO pin with the MSB of its data. Once the transmission begins, no new data is allowed into the shift register from the transmit data register. Therefore, the SPI data register of the slave must be loaded with transmit data before the falling edge of $\overline{SS}$. Any data written after the falling edge is stored in the transmit data register and transferred to the shift register after the current transmission.

14.6.3 Transmission Format When CPHA = 1

Figure 14-6 shows an SPI transmission in which CPHA is logic 1. The figure should not be used as a replacement for data sheet parametric information. Two waveforms are shown for SPSCK: one for CPOL = 0 and another for CPOL = 1. The diagram may be interpreted as a master or slave timing diagram since the serial clock (SPSCK), master in/slave out (MISO), and master out/slave in (MOSI) pins are directly connected between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{SS}$ line is the slave select input to the slave. The slave SPI drives its MISO output only when its slave select input ($\overline{SS}$) is at logic 0, so that only the selected slave drives to the master. The $\overline{SS}$ pin of the master is not shown but is assumed to be inactive. The $\overline{SS}$ pin of the master must be high or must be reconfigured as general-purpose I/O not affecting the SPI. (See **14.8.2 Mode Fault Error**.) When CPHA = 1, the master begins driving its MOSI pin on the first SPSCK edge. Therefore, the slave uses the first SPSCK edge as a start transmission signal. The $\overline{SS}$ pin can remain low between transmissions. This format may be preferable in systems having only one master and only one slave driving the MISO data line.

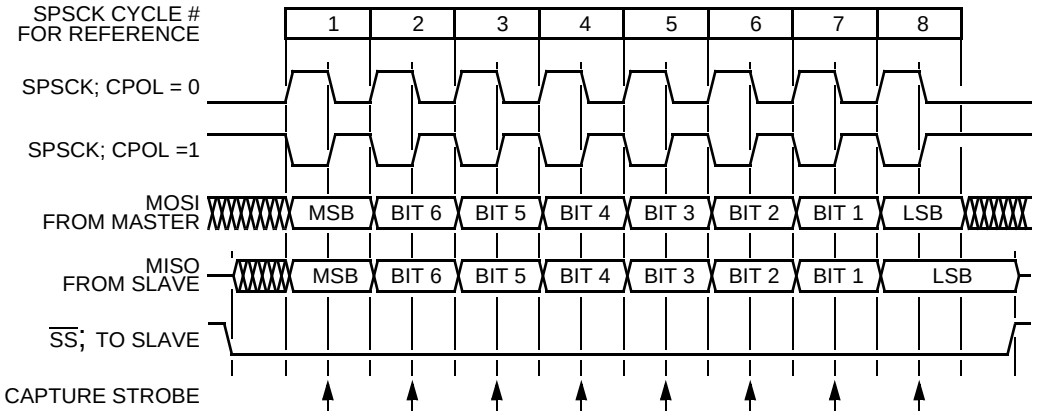


Figure 14-6. Transmission Format (CPHA = 1)

When CPHA = 1 for a slave, the first edge of the SPSCK indicates the beginning of the transmission. This causes the SPI to leave its idle state and begin driving the MISO pin with the MSB of its data. Once the transmission begins, no new data is allowed into the shift register from the transmit data register. Therefore, the SPI data register of the slave must be loaded with transmit data before the first edge of SPSCK. Any data written after the first edge is stored in the transmit data register and transferred to the shift register after the current transmission.

14.6.4 Transmission Initiation Latency

When the SPI is configured as a master (SPMSTR = 1), writing to the SPDR starts a transmission. CPHA has no effect on the delay to the start of the transmission, but it does affect the initial state of the SPSCK signal. When CPHA = 0, the SPSCK signal remains inactive for the first half of the first SPSCK cycle. When CPHA = 1, the first SPSCK cycle begins with an edge on the SPSCK line from its inactive to its active level. The SPI clock rate (selected by SPR1:SPR0) affects the delay from the write to SPDR and the start of the SPI transmission. (See **Figure 14-7**.) The internal SPI clock in the master is a free-running derivative of the internal MCU clock. To conserve power, it is enabled only when both the SPE and SPMSTR bits are set. SPSCK edges occur halfway through the low time of the internal MCU clock. Since the SPI clock is free-running, it is uncertain where the write to the SPDR occurs relative to the slower SPSCK. This uncertainty causes the variation in the initiation delay shown in **Figure 14-7**. This delay is no longer than a single SPI bit time. That is, the maximum delay is two MCU bus cycles for DIV2, eight MCU bus cycles for DIV8, 32 MCU bus cycles for DIV32, and 128 MCU bus cycles for DIV128.

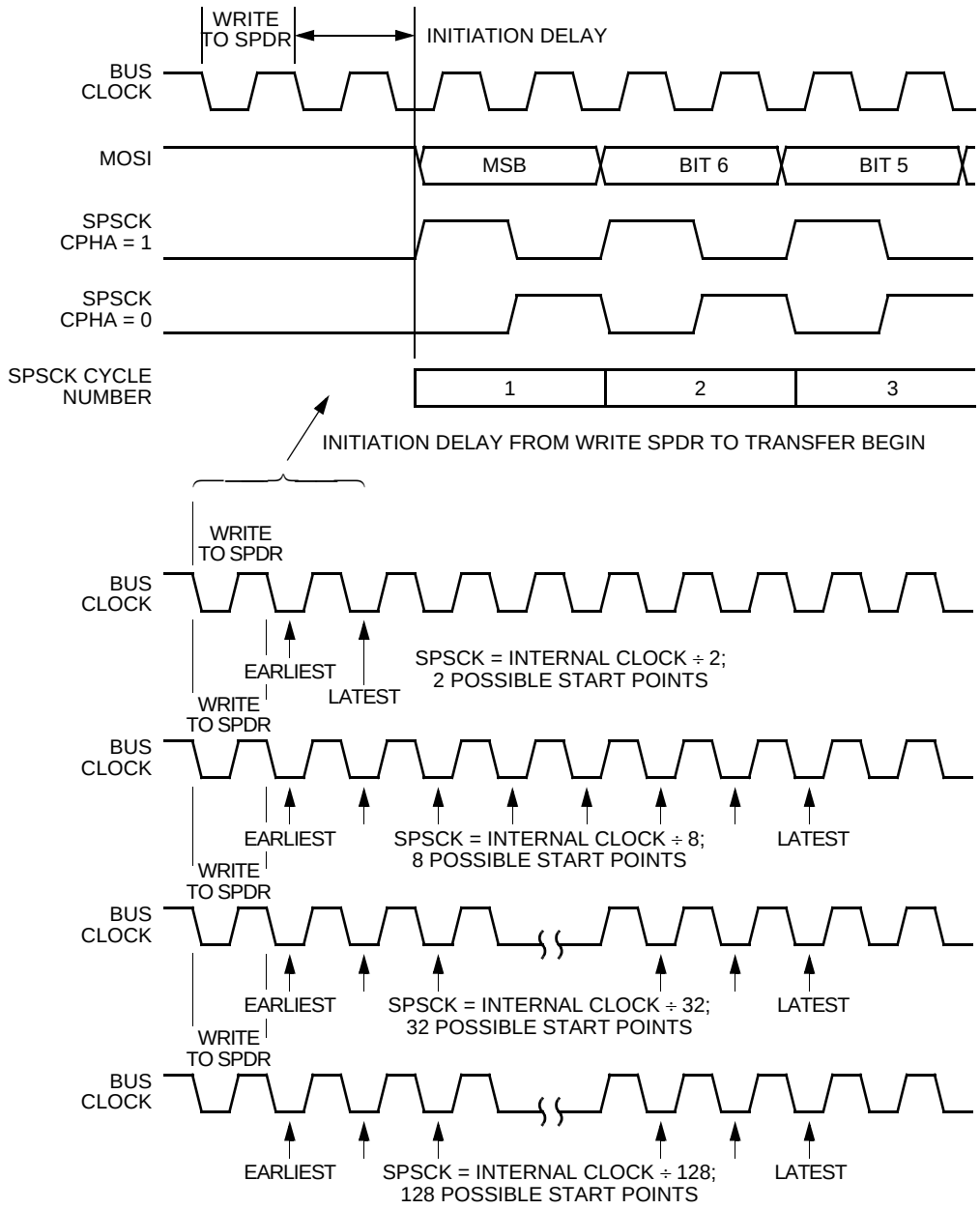


Figure 14-7. Transmission Start Delay (Master)

14.7 Queuing Transmission Data

The double-buffered transmit data register allows a data byte to be queued and transmitted. For an SPI configured as a master, a queued data byte is transmitted immediately after the previous transmission has completed. The SPI transmitter empty flag (SPTE) indicates when the transmit data buffer is ready to accept new data. Write to the transmit data register only when the SPTE bit is high. **Figure 14-8** shows the timing associated with doing back-to-back transmissions with the SPI (SPSCK has CPHA: CPOL = 1:0).

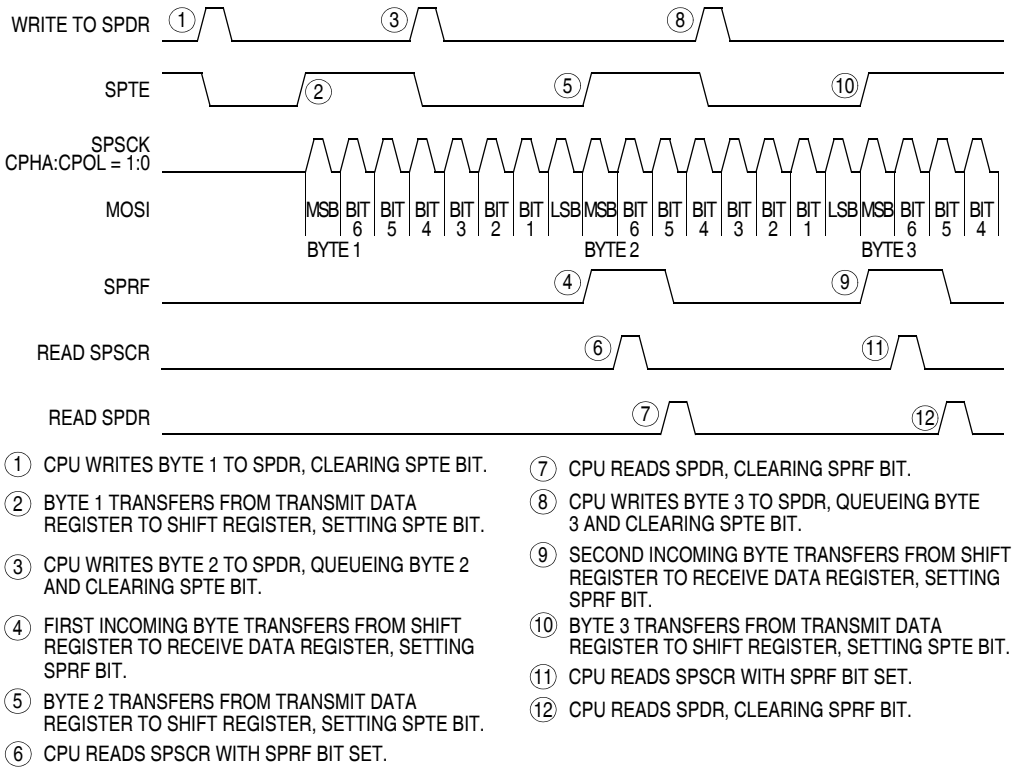


Figure 14-8. SPRF/SPTE CPU Interrupt Timing

The transmit data buffer allows back-to-back transmissions without the slave precisely timing its writes between transmissions as in a system with a single data buffer. Also, if no new data is written to the data buffer, the last value contained in the shift register is the next data word to be transmitted.

For an idle master or idle slave that has no data loaded into its transmit buffer, the SPTE is set again no more than two bus cycles after the transmit buffer empties into the shift register. This allows the user to queue up a 16-bit value to send. For an already active slave, the load of the shift register cannot occur until the transmission is completed. This implies that a back-to-back write to the transmit data register is not possible. The SPTE indicates when the next write can occur.

14.8 Error Conditions

The following flags signal SPI error conditions:

- Overflow (OVRF) — Failing to read the SPI data register before the next full byte enters the shift register sets the OVRF bit. The new byte does not transfer to the receive data register, and the unread byte still can be read. OVRF is in the SPI status and control register.
- Mode fault error (MODF) — The MODF bit indicates that the voltage on the slave select pin ($\overline{SS}$) is inconsistent with the mode of the SPI. MODF is in the SPI status and control register.

14.8.1 Overflow Error

The overflow flag (OVRF) becomes set if the receive data register still has unread data from a previous transmission when the capture strobe of bit 1 of the next transmission occurs. The bit 1 capture strobe occurs in the middle of SPSCK cycle 7. (See **Figure 14-4** and **Figure 14-6**.) If an overflow occurs, all data received after the overflow and before the OVRF bit is cleared does not transfer to the receive data register and does not set the SPI receiver full bit (SPRF). The unread data that transferred to the receive data register before the overflow occurred can still be read. Therefore, an overflow error always indicates the loss of data. Clear the overflow flag by reading the SPI status and control register and then reading the SPI data register.

OVRF generates a receiver/error CPU interrupt request if the error interrupt enable bit (ERRIE) is also set. The SPRF, MODF, and OVRF

interrupts share the same CPU interrupt vector. (See **Figure 14-11**.) It is not possible to enable MODF or OVRF individually to generate a receiver/error CPU interrupt request. However, leaving MODFEN low prevents MODF from being set.

If the CPU SPRF interrupt is enabled and the OVRF interrupt is not, watch for an overflow condition. **Figure 14-9** shows how it is possible to miss an overflow. The first part of **Figure 14-9** shows how it is possible to read the SPSCR and SPDR to clear the SPRF without problems. However, as illustrated by the second transmission example, the OVRF bit can be set in between the time that SPSCR and SPDR are read.

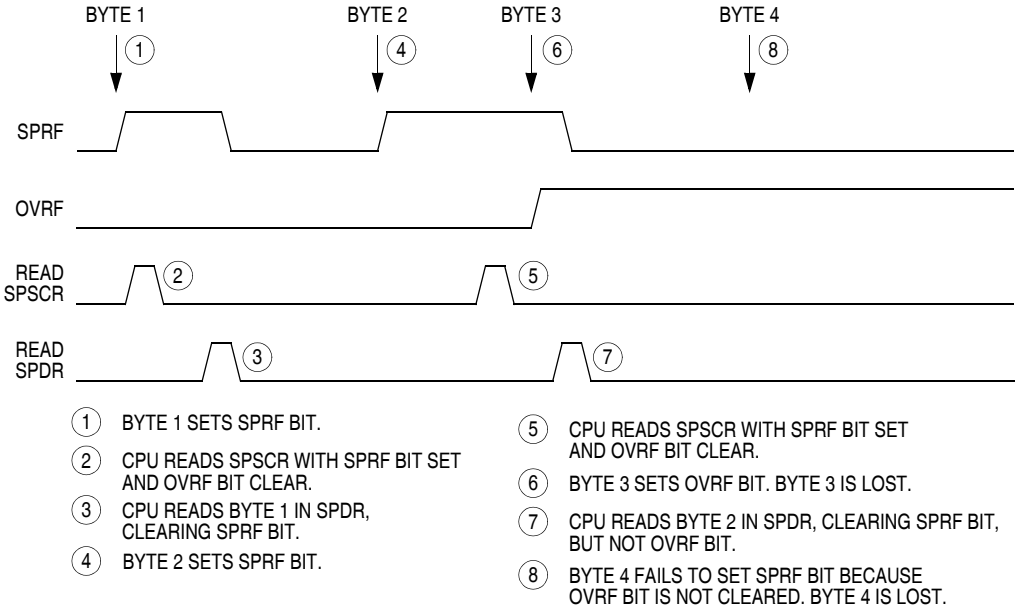


Figure 14-9. Missed Read of Overflow Condition

In this case, an overflow can be missed easily. Since no more SPRF interrupts can be generated until this OVRF is serviced, it is not obvious that bytes are being lost as more transmissions are completed. To prevent this, either enable the OVRF interrupt or do another read of the SPSCR following the read of the SPDR. This ensures that the OVRF was not set before the SPRF was cleared and that future transmissions can set the SPRF bit. **Figure 14-10** illustrates this process. Generally, to avoid this second SPSCR read, enable the OVRF to the CPU by setting the ERRIE bit.

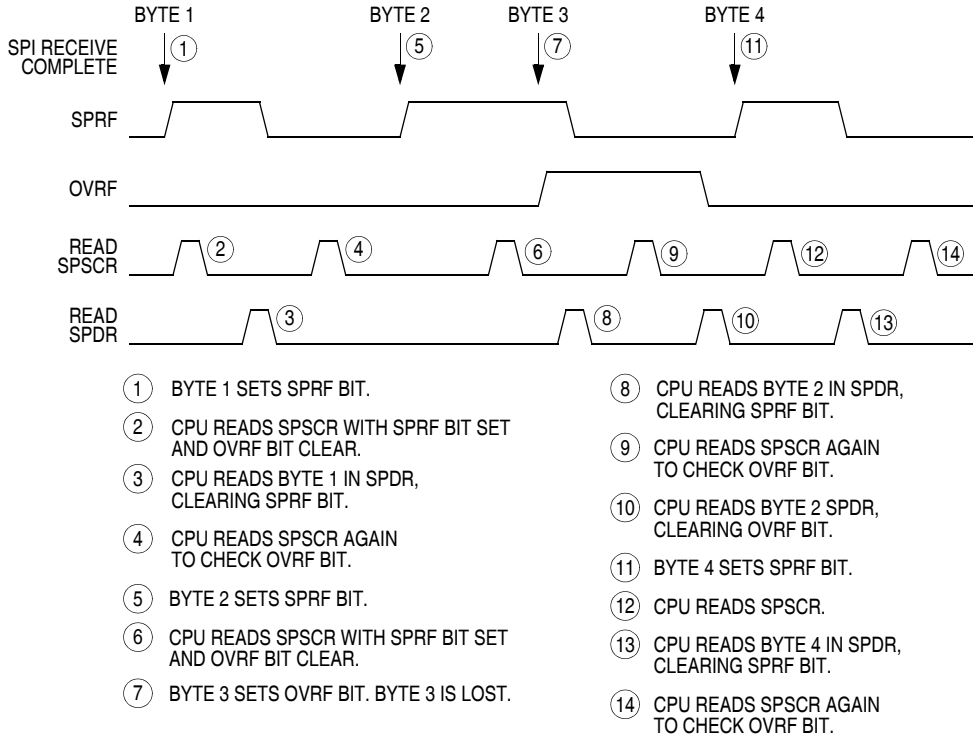


Figure 14-10. Clearing SPRF When OVRF Interrupt Is Not Enabled

14.8.2 Mode Fault Error

Setting the SPMSTR bit selects master mode and configures the SPSCK and MOSI pins as outputs and the MISO pin as an input. Clearing SPMSTR selects slave mode and configures the SPSCK and MOSI pins as inputs and the MISO pin as an output. The mode fault bit, MODF, becomes set any time the state of the slave select pin, $\overline{SS}$, is inconsistent with the mode selected by SPMSTR.

To prevent SPI pin contention and damage to the MCU, a mode fault error occurs if:

- The $\overline{SS}$ pin of a slave SPI goes high during a transmission
- The $\overline{SS}$ pin of a master SPI goes low at any time

For the MODF flag to be set, the mode fault error enable bit (MODFEN) must be set. Clearing the MODFEN bit does not clear the MODF flag but does prevent MODF from being set again after MODF is cleared.

MODF generates a receiver/error CPU interrupt request if the error interrupt enable bit (ERRIE) is also set. The SPRF, MODF, and OVRF interrupts share the same CPU interrupt vector. (See **Figure 14-11**.) It is not possible to enable MODF or OVRF individually to generate a receiver/error CPU interrupt request. However, leaving MODFEN low prevents MODF from being set.

In a master SPI with the mode fault enable bit (MODFEN) set, the mode fault flag (MODF) is set if $\overline{SS}$ goes to logic 0. A mode fault in a master SPI causes the following events to occur:

- If ERRIE = 1, the SPI generates an SPI receiver/error CPU interrupt request.
- The SPE bit is cleared.
- The SPTE bit is set.
- The SPI state counter is cleared.
- The data direction register of the shared I/O port regains control of port drivers.

NOTE: *To prevent bus contention with another master SPI after a mode fault error, clear all SPI bits of the data direction register of the shared I/O port before enabling the SPI.*

When configured as a slave (SPMSTR = 0), the MODF flag is set if $\overline{SS}$ goes high during a transmission. When CPHA = 0, a transmission begins when $\overline{SS}$ goes low and ends once the incoming SPSCK goes back to its idle level following the shift of the eighth data bit. When CPHA = 1, the transmission begins when the SPSCK leaves its idle level and $\overline{SS}$ is already low. The transmission continues until the SPSCK returns to its idle level following the shift of the last data bit. (See **14.6 Transmission Formats**.)

NOTE: *Setting the MODF flag does not clear the SPMSTR bit. The SPMSTR bit has no function when SPE = 0. Reading SPMSTR when MODF = 1 shows the difference between a MODF occurring when the SPI is a master and when it is a slave.*

When CPHA = 0, a MODF occurs if a slave is selected ($\overline{SS}$ is at logic 0) and later unselected ($\overline{SS}$ is at logic 1) even if no SPSCK is sent to that

slave. This happens because $\overline{SS}$ at logic 0 indicates the start of the transmission (MISO driven out with the value of MSB) for CPHA = 0. When CPHA = 1, a slave can be selected and then later unselected with no transmission occurring. Therefore, MODF does not occur since a transmission was never begun.

In a slave SPI (MSTR = 0), the MODF bit generates an SPI receiver/error CPU interrupt request if the ERRIE bit is set. The MODF bit does not clear the SPE bit or reset the SPI in any way. Software can abort the SPI transmission by clearing the SPE bit of the slave.

NOTE: *A logic 1 voltage on the $\overline{SS}$ pin of a slave SPI puts the MISO pin in a high impedance state. Also, the slave SPI ignores all incoming SPSCK clocks, even if it was already in the middle of a transmission.*

To clear the MODF flag, read the SPSCR with the MODF bit set and then write to the SPCR register. This entire clearing mechanism must occur with no MODF condition existing or else the flag is not cleared.

14.9 Interrupts

Four SPI status flags can be enabled to generate CPU interrupt requests.

Table 14-2. SPI Interrupts

Flag	Request
SPTE Transmitter empty	SPI transmitter CPU interrupt request (SPTIE = 1, SPE = 1)
SPRF Receiver full	SPI receiver CPU interrupt request (SPRIE = 1)
OVRF Overflow	SPI receiver/error interrupt request (ERRIE = 1)
MODF Mode fault	SPI receiver/error interrupt request (ERRIE = 1)

Reading the SPI status and control register with SPRF set and then reading the receive data register clears SPRF. The clearing mechanism for the SPTE flag is always just a write to the transmit data register.

The SPI transmitter interrupt enable bit (SPTIE) enables the SPTE flag to generate transmitter CPU interrupt requests, provided that the SPI is enabled (SPE = 1).

The SPI receiver interrupt enable bit (SPRIE) enables the SPRF bit to generate receiver CPU interrupt requests, regardless of the state of the SPE bit. (See **Figure 14-11**.)

The error interrupt enable bit (ERRIE) enables both the MODF and OVRF bits to generate a receiver/error CPU interrupt request.

The mode fault enable bit (MODFEN) can prevent the MODF flag from being set so that only the OVRF bit is enabled by the ERRIE bit to generate receiver/error CPU interrupt requests.

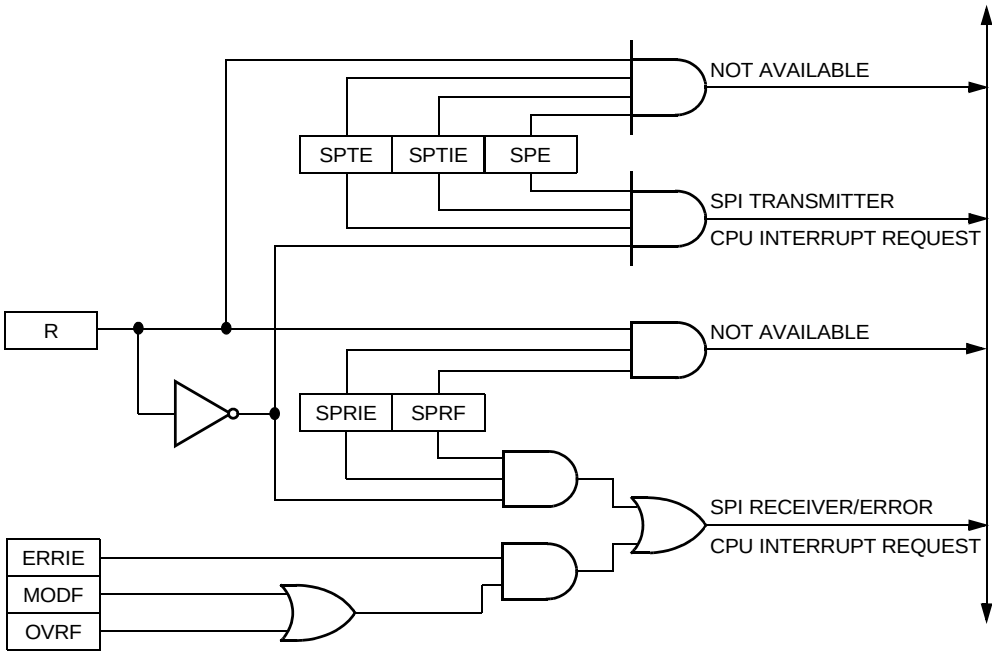


Figure 14-11. SPI Interrupt Request Generation

The following sources in the SPI status and control register can generate CPU interrupt requests:

- SPI receiver full bit (SPRF) — The SPRF bit becomes set every time a byte transfers from the shift register to the receive data register. If the SPI receiver interrupt enable bit, SPRIE, is also set, SPRF generates an SPI receiver/error CPU interrupt request.
- SPI transmitter empty (SPTE) — The SPTE bit becomes set every time a byte transfers from the transmit data register to the shift register. If the SPI transmit interrupt enable bit, SPTIE, is also set, SPTE generates an SPTE CPU interrupt request.

14.10 Resetting the SPI

Any system reset completely resets the SPI. Partial resets occur whenever the SPI enable bit (SPE) is low. Whenever SPE is low, the following occurs:

- The SPTE flag is set.
- Any transmission currently in progress is aborted.
- The shift register is cleared.
- The SPI state counter is cleared, making it ready for a new complete transmission.
- All the SPI port logic is defaulted back to being general-purpose I/O.

These items are reset only by a system reset:

- All control bits in the SPCR register
- All control bits in the SPSCR register (MODFEN, ERRIE, SPR1, and SPR0)
- The status flags SPRF, OVRF, and MODF

By not resetting the control bits when SPE is low, the user can clear SPE between transmissions without having to set all control bits again when SPE is set back high for the next transmission.

By not resetting the SPRF, OVRF, and MODF flags, the user can still service these interrupts after the SPI has been disabled. The user can disable the SPI by writing 0 to the SPE bit. The SPI can also be disabled by a mode fault occurring in an SPI that was configured as a master with the MODFEN bit set.

14.11 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

14.11.1 Wait Mode

The SPI module remains active after the execution of a WAIT instruction. In wait mode the SPI module registers are not accessible by the CPU. Any enabled CPU interrupt request from the SPI module can bring the MCU out of wait mode.

If SPI module functions are not required during wait mode, reduce power consumption by disabling the SPI module before executing the WAIT instruction.

To exit wait mode when an overflow condition occurs, enable the OVRF bit to generate CPU interrupt requests by setting the error interrupt enable bit (ERRIE). (See **14.9 Interrupts**.)

14.11.2 Stop Mode

The SPI module is inactive after the execution of a STOP instruction. The STOP instruction does not affect register conditions. SPI operation resumes after an external interrupt. If stop mode is exited by reset, any transfer in progress is aborted, and the SPI is reset.

14.12 SPI During Break Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state. (See **Section 9. System Integration Module (SIM)**.)

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at logic 0. After the break, doing the second step clears the status bit.

Since the SPTE bit cannot be cleared during a break with the BCFE bit cleared, a write to the transmit data register in break mode does not initiate a transmission nor is this data transferred into the shift register. Therefore, a write to the SPDR in break mode with the BCFE bit cleared has no effect.

14.13 I/O Signals

The SPI module has five I/O pins and shares four of them with a parallel I/O port. They are:

- MISO — Data received
- MOSI — Data transmitted
- SPSCK — Serial clock
- $\overline{SS}$ — Slave select
- CGND — Clock ground (internally connected to V_{SS})

The SPI has limited inter-integrated circuit (I^2C) capability (requiring software support) as a master in a single-master environment. To communicate with I^2C peripherals, MOSI becomes an open-drain output when the SPWOM bit in the SPI control register is set. In I^2C communication, the MOSI and MISO pins are connected to a bidirectional pin from the I^2C peripheral and through a pullup resistor to V_{DD}.

14.13.1 MISO (Master In/Slave Out)

MISO is one of the two SPI module pins that transmits serial data. In full duplex operation, the MISO pin of the master SPI module is connected to the MISO pin of the slave SPI module. The master SPI simultaneously receives data on its MISO pin and transmits data from its MOSI pin.

Slave output data on the MISO pin is enabled only when the SPI is configured as a slave. The SPI is configured as a slave when its SPMSTR bit is logic 0 and its $\overline{SS}$ pin is at logic 0. To support a multiple-slave system, a logic 1 on the $\overline{SS}$ pin puts the MISO pin in a high-impedance state.

When enabled, the SPI controls data direction of the MISO pin regardless of the state of the data direction register of the shared I/O port.

14.13.2 MOSI (Master Out/Slave In)

MOSI is one of the two SPI module pins that transmits serial data. In full-duplex operation, the MOSI pin of the master SPI module is connected to the MOSI pin of the slave SPI module. The master SPI simultaneously transmits data from its MOSI pin and receives data on its MISO pin.

When enabled, the SPI controls data direction of the MOSI pin regardless of the state of the data direction register of the shared I/O port.

14.13.3 SPSCK (Serial Clock)

The serial clock synchronizes data transmission between master and slave devices. In a master MCU, the SPSCK pin is the clock output. In a slave MCU, the SPSCK pin is the clock input. In full-duplex operation, the master and slave MCUs exchange a byte of data in eight serial clock cycles.

When enabled, the SPI controls data direction of the SPSCK pin regardless of the state of the data direction register of the shared I/O port.

14.13.4 $\overline{SS}$ (Slave Select)

The $\overline{SS}$ pin has various functions depending on the current state of the SPI. For an SPI configured as a slave, the $\overline{SS}$ is used to select a slave. For CPHA = 0, the $\overline{SS}$ is used to define the start of a transmission. (See **14.6 Transmission Formats**.) Since it is used to indicate the start of a transmission, the $\overline{SS}$ must be toggled high and low between each byte transmitted for the CPHA = 0 format. However, it can remain low between transmissions for the CPHA = 1 format. See **Figure 14-12**.

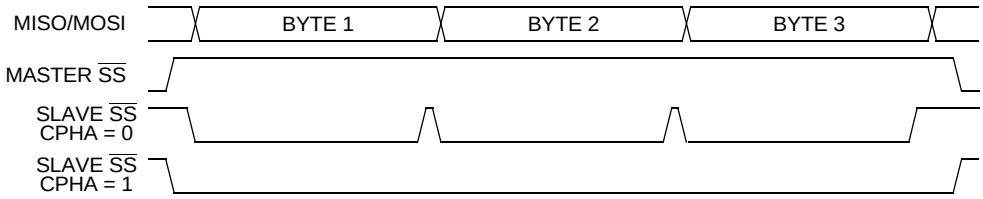


Figure 14-12. CPHA/$\overline{SS}$ Timing

When an SPI is configured as a slave, the $\overline{SS}$ pin is always configured as an input. It cannot be used as a general-purpose I/O regardless of the state of the MODFEN control bit. However, the MODFEN bit can still prevent the state of the $\overline{SS}$ from creating a MODF error. (See **14.14.2 SPI Status and Control Register**.)

NOTE: *A logic 1 voltage on the $\overline{SS}$ pin of a slave SPI puts the MISO pin in a high-impedance state. The slave SPI ignores all incoming SPSCK clocks, even if it was already in the middle of a transmission.*

When an SPI is configured as a master, the $\overline{SS}$ input can be used in conjunction with the MODF flag to prevent multiple masters from driving MOSI and SPSCK. (See **14.8.2 Mode Fault Error**.) For the state of the $\overline{SS}$ pin to set the MODF flag, the MODFEN bit in the SPSCK register must be set. If the MODFEN bit is low for an SPI master, the $\overline{SS}$ pin can be used as a general-purpose I/O under the control of the data direction register of the shared I/O port. With MODFEN high, it is an input-only pin to the SPI regardless of the state of the data direction register of the shared I/O port.

The CPU can always read the state of the $\overline{SS}$ pin by configuring the appropriate pin as an input and reading the port data register. (See **Table 14-3**.)

Table 14-3. SPI Configuration

SPE	SPMSTR	MODFEN	SPI Configuration	State of $\overline{SS}$ Logic
0	X(1)	X	Not enabled	General-purpose I/O; $\overline{SS}$ ignored by SPI
1	0	X	Slave	Input-only to SPI
1	1	0	Master without MODF	General-purpose I/O; $\overline{SS}$ ignored by SPI
1	1	1	Master with MODF	Input-only to SPI

Note 1. X = Don’t care

14.13.5 CGND (Clock Ground)

CGND is the ground return for the serial clock pin, SPSCK, and the ground for the port output buffers. It is internally connected to V_{SS} as shown in **Table 14-1**.

14.14 I/O Registers

Three registers control and monitor SPI operation:

- SPI control register (SPCR)
- SPI status and control register (SPSCR)
- SPI data register (SPDR)

14.14.1 SPI Control Register

The SPI control register:

- Enables SPI module interrupt requests
- Configures the SPI module as master or slave
- Selects serial clock polarity and phase
- Configures the SPSCK, MOSI, and MISO pins as open-drain outputs
- Enables the SPI module

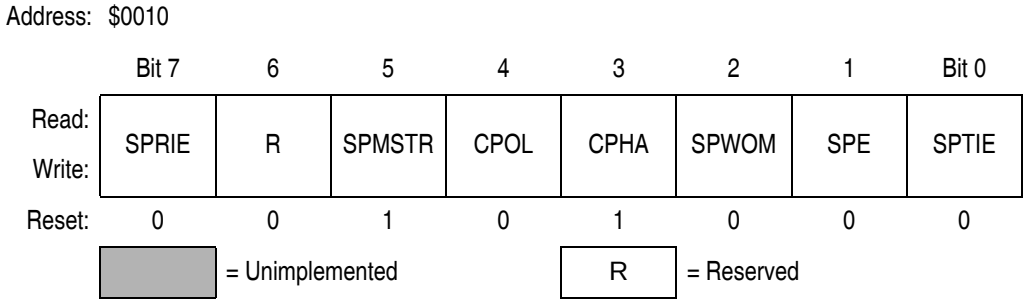


Figure 14-13. SPI Control Register (SPCR)

SPRIE — SPI Receiver Interrupt Enable Bit

This read/write bit enables CPU interrupt requests generated by the SPRF bit. The SPRF bit is set when a byte transfers from the shift register to the receive data register. Reset clears the SPRIE bit.

1 = SPRF CPU interrupt requests enabled
0 = SPRF CPU interrupt requests disabled

SPMSTR — SPI Master Bit

This read/write bit selects master mode operation or slave mode operation. Reset sets the SPMSTR bit.

1 = Master mode
0 = Slave mode

CPOL — Clock Polarity Bit

This read/write bit determines the logic state of the SPSCK pin between transmissions. (See **Figure 14-4** and **Figure 14-6**.) To transmit data between SPI modules, the SPI modules must have identical CPOL values. Reset clears the CPOL bit.

CPHA — Clock Phase Bit

This read/write bit controls the timing relationship between the serial clock and SPI data. (See **Figure 14-4** and **Figure 14-6**.) To transmit data between SPI modules, the SPI modules must have identical CPHA values. When CPHA = 0, the $\overline{SS}$ pin of the slave SPI module must be set to logic 1 between bytes. (See **Figure 14-12**.) Reset sets the CPHA bit.

SPWOM — SPI Wired-OR Mode Bit

This read/write bit disables the pullup devices on pins SPSCK, MOSI, and MISO so that those pins become open-drain outputs.

1 = Wired-OR SPSCK, MOSI, and MISO pins
0 = Normal push-pull SPSCK, MOSI, and MISO pins

SPE — SPI Enable

This read/write bit enables the SPI module. Clearing SPE causes a partial reset of the SPI. (See **14.10 Resetting the SPI**.) Reset clears the SPE bit.

1 = SPI module enabled
0 = SPI module disabled

SPTIE— SPI Transmit Interrupt Enable

This read/write bit enables CPU interrupt requests generated by the SPTE bit. SPTE is set when a byte transfers from the transmit data register to the shift register. Reset clears the SPTIE bit.

1 = SPTE CPU interrupt requests enabled
0 = SPTE CPU interrupt requests disabled

14.14.2 SPI Status and Control Register

The SPI status and control register contains flags to signal these conditions:

- Receive data register full
- Failure to clear SPRF bit before next byte is received (overflow error)
- Inconsistent logic level on $\overline{SS}$ pin (mode fault error)
- Transmit data register empty

The SPI status and control register also contains bits that perform these functions:

- Enable error interrupts
- Enable mode fault error detection
- Select master SPI baud rate

Address: $0011

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
Write:								
Reset:	0	0	0	0	1	0	0	0

= Unimplemented

Figure 14-14. SPI Status and Control Register (SPSCR)

SPRF — SPI Receiver Full Bit

This clearable, read-only flag is set each time a byte transfers from the shift register to the receive data register. SPRF generates a CPU interrupt request if the SPRIE bit in the SPI control register is set also.

During an SPRF CPU interrupt, the CPU clears SPRF by reading the SPI status and control register with SPRF set and then reading the SPI data register. Reset clears the SPRF bit.

1 = Receive data register full
0 = Receive data register not full

ERRIE — Error Interrupt Enable Bit

This read/write bit enables the MODF and OVRF bits to generate CPU interrupt requests. Reset clears the ERRIE bit.

1 = MODF and OVRF can generate CPU interrupt requests
0 = MODF and OVRF cannot generate CPU interrupt requests

OVRF — Overflow Bit

This clearable, read-only flag is set if software does not read the byte in the receive data register before the next full byte enters the shift register. In an overflow condition, the byte already in the receive data register is unaffected, and the byte that shifted in last is lost. Clear the OVRF bit by reading the SPI status and control register with OVRF set and then reading the receive data register. Reset clears the OVRF bit.

1 = Overflow
0 = No overflow

MODF — Mode Fault Bit

This clearable, read-only flag is set in a slave SPI if the $\overline{SS}$ pin goes high during a transmission with the MODFEN bit set. In a master SPI, the MODF flag is set if the $\overline{SS}$ pin goes low at any time with the MODFEN bit set. Clear the MODF bit by reading the SPI status and control register (SPSCR) with MODF set and then writing to the SPI control register (SPCR). Reset clears the MODF bit.

1 = $\overline{SS}$ pin at inappropriate logic level
0 = $\overline{SS}$ pin at appropriate logic level

SPTE — SPI Transmitter Empty Bit

This clearable, read-only flag is set each time the transmit data register transfers a byte into the shift register. SPTE generates an SPTE CPU interrupt request if the SPTIE bit in the SPI control register is set also.

NOTE: *Do not write to the SPI data register unless the SPTE bit is high.*

During an SPTE CPU interrupt, the CPU clears the SPTE bit by writing to the transmit data register.

Reset sets the SPTE bit.

1 = Transmit data register empty
0 = Transmit data register not empty

MODFEN — Mode Fault Enable Bit

This read/write bit, when set to 1, allows the MODF flag to be set. If the MODF flag is set, clearing the MODFEN does not clear the MODF flag. If the SPI is enabled as a master and the MODFEN bit is low, then the $\overline{SS}$ pin is available as a general-purpose I/O.

If the MODFEN bit is set, then this pin is not available as a general-purpose I/O. When the SPI is enabled as a slave, the $\overline{SS}$ pin is not available as a general-purpose I/O regardless of the value of MODFEN. (See **14.13.4 SS (Slave Select)**.)

If the MODFEN bit is low, the level of the $\overline{SS}$ pin does not affect the operation of an enabled SPI configured as a master. For an enabled SPI configured as a slave, having MODFEN low only prevents the MODF flag from being set. It does not affect any other part of SPI operation. (See **14.8.2 Mode Fault Error**.)

SPR1 and SPR0 — SPI Baud Rate Select Bits

In master mode, these read/write bits select one of four baud rates as shown in **Table 14-4**. SPR1 and SPR0 have no effect in slave mode. Reset clears SPR1 and SPR0.

Table 14-4. SPI Master Baud Rate Selection

SPR1 and SPR0	Baud Rate Divisor (BD)
00	2
01	8
10	32
11	128

Use this formula to calculate the SPI baud rate:

$$\text{Baud rate} = \frac{\text{CGMOUT}}{2 \times \text{BD}}$$

where:

CGMOUT = base clock output of the clock generator module (CGM)
BD = baud rate divisor

14.14.3 SPI Data Register

The SPI data register consists of the read-only receive data register and the write-only transmit data register. Writing to the SPI data register writes data into the transmit data register. Reading the SPI data register reads data from the receive data register. The transmit data and receive data registers are separate registers that can contain different values. (See **Figure 14-2**.)

Address: $0012

	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:	Unaffected by reset							

Figure 14-15. SPI Data Register (SPDR)

R7–R0/T7–T0 — Receive/Transmit Data Bits

NOTE: *Do not use read-modify-write instructions on the SPI data register since the register read is not the same as the register written.*

Section 15. Analog-to-Digital Converter (ADC)

15.1 Contents

15.2 Introduction 302

15.3 Features 302

15.4 Functional Description 303
15.4.1 ADC Port I/O Pins 303
15.4.2 Voltage Conversion 305
15.4.3 Conversion Time 305
15.4.4 Continuous Conversion 306
15.4.5 Result Justification 306
15.4.6 Monotonicity 307

15.5 Interrupts 308

15.6 Low-Power Modes 308
15.6.1 Wait Mode 308
15.6.2 Stop Mode 308

15.7 I/O Signals 308
15.7.1 ADC Voltage In (V_{ADIN}) 309
15.7.2 ADC Analog Power Pin (V_{DDA}) 309
15.7.3 ADC Voltage Reference High Pin (V_{REFH}) 309
15.7.4 ADC Voltage Reference Low Pin (V_{REFL}) 309

15.8 I/O Registers 310
15.8.1 ADC Status and Control Register 310
15.8.2 ADC Data Register 312
15.8.3 ADC Clock Control Register 314

15.2 Introduction

This section describes the analog-to-digital convert (ADC). The ADC is a 6-channel 10-bit linear successive approximation ADC.

15.3 Features

Features of the ADC module include:

- Six Channels with Multiplexed Input
- High impedance buffered input
- Linear Successive Approximation with monotonicity
- 10-Bit Resolution
- Single or Continuous Conversion
- Conversion Complete Flag Or Conversion Complete Interrupt
- Selectable ADC Clock
- Conversion result justification
  - 8-bit truncated mode
  - Right justified mode
  - Left justified mode
  - Left justified sign mode

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003C	ADC Status and Control Register (ADSCR)	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
$003D	ADC Data Register High (ADRH)	Read:	ADx	ADx	ADx	ADx	ADx	ADx	ADx	ADx
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003E	ADC Data Register Low (ADRL)	Read:	ADx	ADx	ADx	ADx	ADx	ADx	ADx	ADx
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003F	ADC Clock Register (ADCLK)	Read:	ADIV2	ADIV1	ADIV0	ADICLK	MODE1	MODE0	0	0
		Write:								R
		Reset:	0	0	0	0	0	1	0	0

= Unimplemented R = Reserved

Figure 15-1. ADC I/O Register Summary

15.4 Functional Description

The ADC provides six pins for sampling external sources at pins PTA4/ADC0–PTA7/ADC3 and PTB6/ADC4–PTB7/ADC5. An analog multiplexer allows the single ADC converter to select one of nine ADC channels as ADC voltage in (V_{ADIN}). V_{ADIN} is converted by the successive approximation register-based analog-to-digital converter. When the conversion is completed, ADC places the result in the ADC data register, high and low byte (ADRH and ADRL), and sets a flag or generates an interrupt.

Figure 15-2 shows the structure of the ADC module.

15.4.1 ADC Port I/O Pins

PTA4–PTA7 and PTB6–PTB7 are general-purpose I/O pins that are shared with the ADC channels. The channel select bits, ADCH[4:0], define which ADC channel/port pin will be used as the input signal. The ADC overrides the port I/O logic by forcing that pin as input to the ADC. The remaining ADC channels/port pins are controlled by the port I/O

logic and can be used as general-purpose I/O pins. Writes to the port data register or data direction register will not have any affect on the port pin that is selected by the ADC. Read of a port pin which is in use by the ADC will return the pin condition if the corresponding DDR bit is at logic 0. If the DDR bit is at logic 1, the value in the port data latch is read.

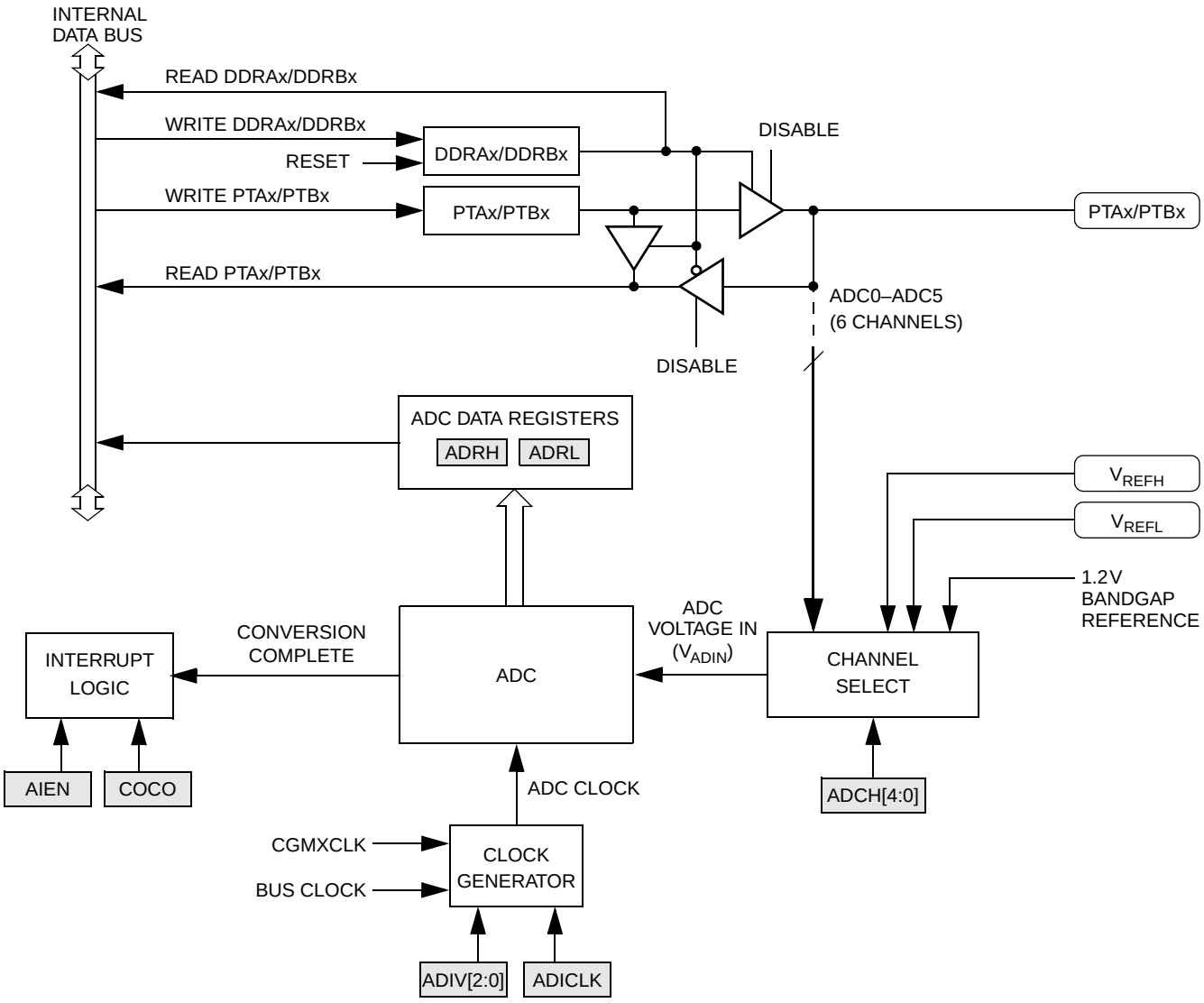


Figure 15-2. ADC Block Diagram

15.4.2 Voltage Conversion

When the input voltage to the ADC equals V_{REFH}, the ADC converts the signal to $3FF (full scale). If the input voltage equals V_{REFL}, the ADC converts it to $000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. All other input voltages will result in $3FF if greater than V_{REFH} and $000 if less than V_{REFL}.

NOTE: *Input voltage should not exceed the analog supply voltages.*

15.4.3 Conversion Time

Conversion starts after a write to the ADSCR. A conversion is between 16 and 17 ADC clock cycles, therefore:

$$\text{Conversion time} = \frac{\text{16 to17 ADC cycles}}{\text{ADC frequency}}$$

Number of bus cycles = conversion time × bus frequency

The ADC conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is either the bus clock or CGMXCLK and is selectable by the ADICLK bit located in the ADC clock register. The divide ratio is selected by the ADIV[2:0] bits.

For example, if a 4MHz CGMXCLK is selected as the ADC input clock source, with a divide-by-2 prescale, and the bus speed is set at 8MHz:

$$\text{Conversion time} = \frac{\text{16 to 17 ADC cycles}}{\text{4MHz} \div 2} = 8 \text{ to } 8.5\ \mu\text{s}$$

Number of bus cycles = 8µs x 8MHz = 64 to 68 cycles

NOTE: *The ADC frequency must be between f_{ADIC} minimum and f_{ADIC} maximum to meet ADC specifications. See* ***23.6 5.0V DC Electrical Characteristics****.*

Since an ADC cycle may be comprised of several bus cycles (eight in the previous example) and the start of a conversion is initiated by a bus cycle write to the ADSCR, from zero to four additional bus cycles may occur before the start of the initial ADC cycle. This results in a fractional ADC cycle and is represented as the 17th cycle.

15.4.4 Continuous Conversion

In the continuous conversion mode, the ADC continuously converts the selected channel, filling the ADC data register (ADRH:ADRL) with new data after each conversion. Data from the previous conversion will be overwritten whether that data has been read or not. Conversions will continue until the ADCO bit is cleared. The COCO bit is set after each conversion and can be cleared by writing to the ADC status and control register or reading of the ADRL data register.

15.4.5 Result Justification

The conversion result may be formatted in four different ways.

- Left justified
- Right justified
- Left justified sign data mode
- 8-bit truncation

All four of these modes are controlled using MODE0 and MODE1 bits located in the ADC clock control register (ADCLK).

Left justification will place the eight most significant bits (MSB) in the ADC data register high (ADRH). This may be useful if the result is to be treated as an 8-bit result where the least significant two bits, located in the ADC data register low (ADRL) can be ignored. However, ADRL must be read after ADRH or else the interlocking will prevent all new conversions from being stored.

Right justification will place only the two MSBs in the corresponding ADC data register high (ADRH) and the eight LSB bits in ADC data register low (ADRL). This mode of operation typically is used when a 10-bit unsigned result is desired.

Left justified sign data mode is similar to left justified mode with one exception. The MSB of the 10-bit result, AD9 located in ADRH is complemented. This mode of operation is useful when a result, represented as a signed magnitude from mid-scale, is needed.

Finally, 8-bit truncation mode will place the eight MSBs in ADC data register low (ADRL). The two LSBs are dropped. This mode of operation is used when compatibility with 8-bit ADC designs are required. No interlocking between ADRH and ADRL is present.

NOTE: *Quantization error is affected when only the most significant eight bits are used as a result. See* ***Figure 15-3****.*

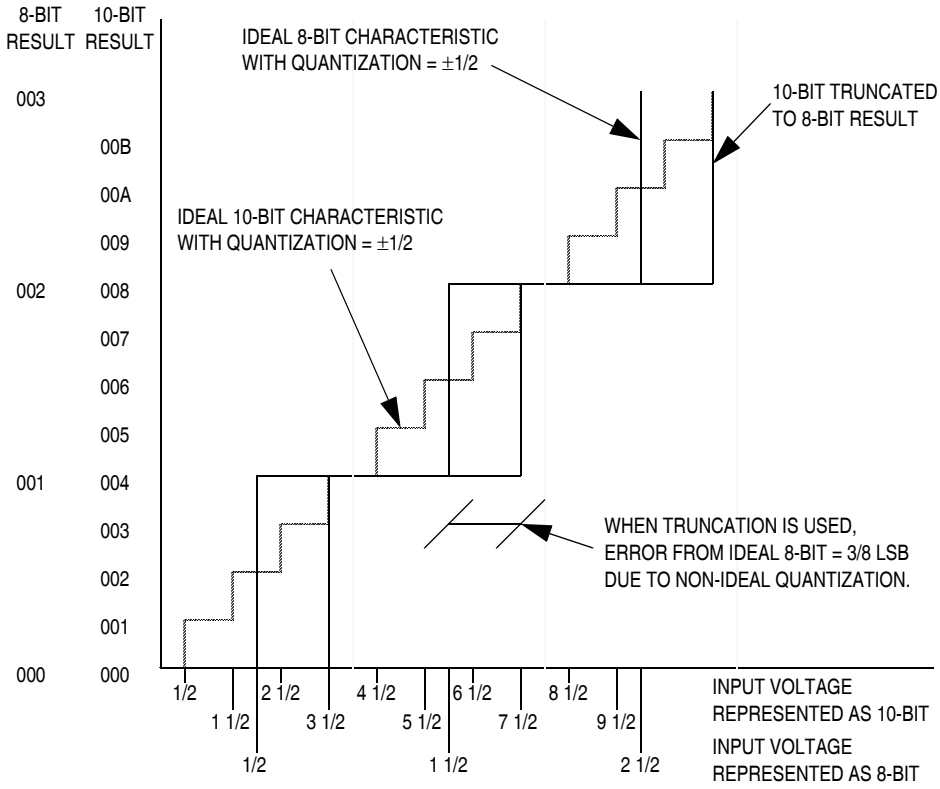


Figure 15-3. 8-Bit Truncation Mode Error

15.4.6 Monotonicity

The conversion process is monotonic and has no missing codes.

15.5 Interrupts

When the AIEN bit is set, the ADC module is capable of generating a CPU interrupt after each ADC conversion. A CPU interrupt is generated if the COCO bit is at logic 0. The COCO bit is not used as a conversion complete flag when interrupts are enabled. The interrupt vector is defined in **Table 2-1 . Vector Addresses**.

15.6 Low-Power Modes

The STOP and WAIT instructions put the MCU in low power-consumption standby modes.

15.6.1 Wait Mode

The ADC continues normal operation in wait mode. Any enabled CPU interrupt request from the ADC can bring the MCU out of wait mode. If the ADC is not required to bring the MCU out of wait mode, power down the ADC by setting the ADCH[4:0] bits to logic 1's before executing the WAIT instruction.

15.6.2 Stop Mode

The ADC module is inactive after the execution of a STOP instruction. Any pending conversion is aborted. ADC conversions resume when the MCU exits stop mode. Allow one conversion cycle to stabilize the analog circuitry before attempting a new ADC conversion after exiting stop mode.

15.7 I/O Signals

The ADC module has nine channels, six channels are shared with port A and port C I/O pins; two channels are the ADC voltage reference inputs, V_{REFH} and V_{REFL}; and one channel is the 1.2V bandgap reference voltage.

15.7.1 ADC Voltage In (V_{ADIN})

V_{ADIN} is the input voltage signal from one of the nine channels to the ADC module.

15.7.2 ADC Analog Power Pin (V_{DDA})

The ADC analog portion uses V_{DDA} as its power pin. Connect the V_{DDA} pin to the same voltage potential as V_{DD}. External filtering may be necessary to ensure clean V_{DDA} for good results.

NOTE: *Route V_{DDA} carefully for maximum noise immunity and place bypass capacitors as close as possible to the package.*

15.7.3 ADC Voltage Reference High Pin (V_{REFH})

V_{REFH} is the power supply for setting the reference voltage V_{REFH}. Connect the V_{REFH} pin to the same voltage potential as V_{DDA}. There will be a finite current associated with V_{REFH} (see **Section 23. Electrical Specifications**).

NOTE: *Route V_{REFH} carefully for maximum noise immunity and place bypass capacitors as close as possible to the package.*

15.7.4 ADC Voltage Reference Low Pin (V_{REFL})

V_{REFL} is the lower reference supply for the ADC. Connect the V_{REFL} pin to the same voltage potential as V_{SSA}. There will be a finite current associated with V_{REFL} (see **Section 23. Electrical Specifications**).

15.8 I/O Registers

These I/O registers control and monitor operation of the ADC:

- ADC status and control register, ADSCR
- ADC data register, ADRH:ADRL
- ADC clock register, ADCLK

15.8.1 ADC Status and Control Register

This section describes the function of the ADC status and control register (ADSCR). Writing ADSCR aborts the current conversion and initiates a new conversion.

Address: $003C

Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
Write:								
Reset:	0	0	0	1	1	1	1	1

= Unimplemented

Figure 15-4. ADC Status and Control Register (ADSCR)

COCO — Conversions Complete Bit

When the AIEN bit is a logic 0, the COCO is a read-only bit which is set each time a conversion is completed. This bit is cleared whenever the ADSCR is written, or whenever the ADC clock control register is written, or whenever the ADC data register low, ADRL, is read.

If the AIEN bit is logic 1, the COCO bit always read as logic 0, CPU to service the ADC interrupt will be generated at the end if an ADC conversion. Reset clears the COCO bit.

1 = Conversion completed (AIEN = 0)
0 = Conversion not completed (AIEN = 0)/CPU interrupt (AIEN=1)

AIEN — ADC Interrupt Enable Bit

When this bit is set, an interrupt is generated at the end of an ADC conversion. The interrupt signal is cleared when the data register, ADR0, is read or the ADSCR is written. Reset clears the AIEN bit.

1 = ADC interrupt enabled
0 = ADC interrupt disabled

ADCO — ADC Continuous Conversion Bit

When set, the ADC will convert samples continuously and update the ADC data register at the end of each conversion. Only one conversion is allowed when this bit is cleared. Reset clears the ADCO bit.

1 = Continuous ADC conversion
0 = One ADC conversion

ADCH[4:0] — ADC Channel Select Bits

ADCH[4:0] form a 5-bit field which is used to select one of the ADC channels when not in auto-scan mode. The five channel select bits are detailed in **Table 15-1**.

NOTE: *Care should be taken when using a port pin as both an analog and a digital input simultaneously to prevent switching noise from corrupting the analog signal.*

NOTE: *Recovery from the disabled state requires one conversion cycle to stabilize.*

Table 15-1. MUX Channel Select

ADCH4	ADCH3	ADCH2	ADCH1	ADCH0	ADC Channel	Input Select
0	0	0	0	0	ADC0	PTA4
0	0	0	0	1	ADC1	PTA5
0	0	0	1	0	ADC2	PTA6
0	0	0	1	1	ADC3	PTA7
0	0	1	0	0	ADC4	PTB6
0	0	1	0	1	ADC5	PTB7
0	0	1	1	0	ADC6	1.2V Bandgap reference
0 ↓ 1	0 ↓ 1	1 ↓ 1	1 ↓ 0	1 ↓ 0	ADC7 ↓ ADC28	Reserved
1	1	1	0	1	ADC29	V_{REFH} (see Note 2)
1	1	1	1	0	ADC30	V_{REFL} (see Note 2)
1	1	1	1	1	ADC powered-off	

NOTES:
1. If any reserved channels are selected, the resulting ADC conversion will be unknown.
2. The voltage levels supplied from internal reference nodes as specified in the table are used to verify the operation of the ADC converter both in production test and for user applications.

15.8.2 ADC Data Register

The ADC data register consist of a pair of 8-bit registers: high byte (ADRH), and low byte (ADRL). This pair form a 16-bit register to store the 10-bit ADC result for the selected ADC result justification mode.

In 8-bit truncated mode, the ADRL holds the eight most significant bits (MSBs) of the 10-bit result. The ADRL is updated each time an ADC conversion completes. In 8-bit truncated mode, ADRL contains no interlocking with ADRH. (See **Figure 15-5 . ADRH and ADRL in 8-Bit Truncated Mode**.)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003D	ADC Data Register High (ADRH)	Read:	0	0	0	0	0	0	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003E	ADC Data Register Low (ADRL)	Read:	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

Figure 15-5. ADRH and ADRL in 8-Bit Truncated Mode

In right justified mode the ADRH holds the two MSBs, and the ADRL holds the eight least significant bits (LSBs), of the 10-bit result. ADRH and ADRL are updated each time a single channel ADC conversion completes. Reading ADRH latches the contents of ADRL. Until ADRL is read all subsequent ADC results will be lost.
(See **Figure 15-6 . ADRH and ADRL in Right Justified Mode**.)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003D	ADC Data Register High (ADRH)	Read:	0	0	0	0	0	0	AD9	AD8
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003E	ADC Data Register Low (ADRL)	Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

Figure 15-6. ADRH and ADRL in Right Justified Mode

In left justified mode the ADRH holds the eight most significant bits (MSBs), and the ADRL holds the two least significant bits (LSBs), of the 10-bit result. The ADRH and ADRL are updated each time a single channel ADC conversion completes. Reading ADRH latches the contents of ADRL. Until ADRL is read all subsequent ADC results will be lost. (See **Figure 15-7 . ADRH and ADRL in Left Justified Mode**.)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003D	ADC Data Register High (ADRH)	Read:	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003E	ADC Data Register Low (ADRL)	Read:	AD1	AD0	0	0	0	0	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

Figure 15-7. ADRH and ADRL in Left Justified Mode

In left justified sign mode the ADRH holds the eight MSBs with the MSB complemented, and the ADRL holds the two least significant bits (LSBs), of the 10-bit result. The ADRH and ADRL are updated each time a single channel ADC conversion completes. Reading ADRH latches the contents of ADRL. Until ADRL is read all subsequent ADC results will be lost. (See **Figure 15-8 . ADRH and ADRL in Left Justified Sign Data Mode**.)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003D	ADC Data Register High (ADRH)	Read:	$\overline{AD9}$	AD8	AD7	AD6	AD5	AD4	AD3	AD2
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003E	ADC Data Register Low (ADRL)	Read:	AD1	AD0	0	0	0	0	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

Figure 15-8. ADRH and ADRL in Left Justified Sign Data Mode

15.8.3 ADC Clock Control Register

The ADC clock control register (ADCLK) selects the clock frequency for the ADC.

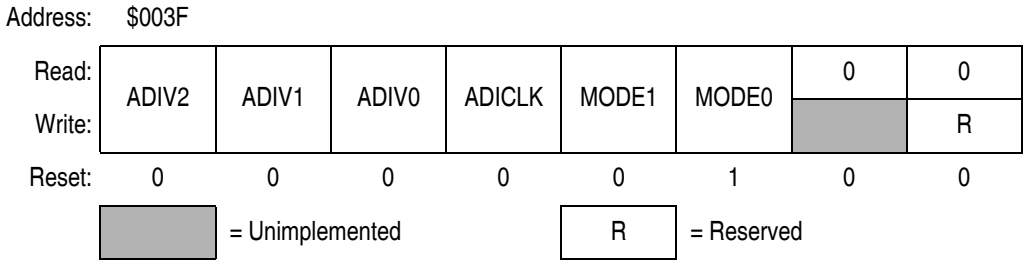


Figure 15-9. ADC Clock Control Register (ADICLK)

ADIV[2:0] — ADC Clock Prescaler Bits

ADIV2, ADIV1, and ADIV0 form a 3-bit field which selects the divide ratio used by the ADC to generate the internal ADC clock. **Table 15-2** shows the available clock configurations. The ADC clock should be set to between 32 kHz and 2 MHz.

Table 15-2. ADC Clock Divide Ratio

ADIV2	ADIV1	ADIV0	ADC Clock Rate
0	0	0	ADC input clock ÷ 1
0	0	1	ADC input clock ÷ 2
0	1	0	ADC input clock ÷ 4
0	1	1	ADC input clock ÷ 8
1	X	X	ADC input clock ÷ 16

X = don’t care

ADICLK — ADC Input Clock Select Bit

ADICLK selects either bus clock or CGMXCLK as the input clock source to generate the internal ADC clock. Reset selects CGMXCLK as the ADC clock source.

If the external clock (CGMXCLK) is equal to or greater than 1MHz, CGMXCLK can be used as the clock source for the ADC. If CGMXCLK is less than 1MHz, use the PLL-generated bus clock as the clock source. As long as the internal ADC clock is at f_{ADIC}, correct operation can be guaranteed.

1 = Internal bus clock

0 = External clock, CGMXCLK

$$f_{ADIC} = \frac{\text{CGMXCLK or bus frequency}}{\text{ADIV[2:0]}}$$

MODE1 and MODE0 — Modes of Result Justification

MODE1 and MODE0 selects between four modes of operation. The manner in which the ADC conversion results will be placed in the ADC data registers is controlled by these modes of operation. Reset returns right-justified mode.

Table 15-3. ADC Mode Select

MODE1	MODE0	ADC Clock Rate
0	0	8-bit truncated mode
0	1	Right justified mode
1	0	Left justified mode
1	1	Left justified sign data mode

Section 16. Liquid Crystal Display Driver (LCD)

16.1 Contents

16.2 Introduction 318
16.3 Features 318
16.4 Pin Name Conventions and I/O Register Addresses 318
16.5 Functional Description 320
16.5.1 LCD Duty 321
16.5.2 LCD Voltages (V_{LCD}, V_{LCD1}, V_{LCD2}, V_{LCD3}) 323
16.5.3 LCD Cycle Frame 323
16.5.4 Fast Charge and Low Current 324
16.5.5 Contrast Control 325
16.6 Low-Power Modes 325
16.6.1 Wait Mode 325
16.6.2 Stop Mode 325
16.7 I/O Signals 326
16.7.1 BP0–BP3 (Backplane Drivers) 326
16.7.2 FP0–FP26 (Frontplane Drivers) 328
16.8 Seven Segment Display Connection 332
16.9 I/O Registers 335
16.9.1 LCD Control Register (LCDCR) 335
16.9.2 LCD Clock Register (LCDCLK) 337
16.9.3 LCD Data Registers (LDAT1–LDAT14) 339

16.2 Introduction

This section describes the liquid crystal display (LCD) driver module. The LCD driver module can drive a maximum of 27 frontplanes and 4 backplanes, depending on the LCD duty selected.

16.3 Features

Features of the LCD driver module include the following:

- Software programmable driver segment configurations:
  - 26 frontplanes × 4 backplanes (104 segments)
  - 27 frontplanes × 3 backplanes (81 segments)
  - 27 frontplanes × 1 backplane (27 segments)
- LCD bias voltages generated by internal resistor ladder
- Software programmable contrast control

16.4 Pin Name Conventions and I/O Register Addresses

Three dedicated I/O pins are for the backplanes, BP0–BP2, eighteen dedicated I/O pins are for the frontplanes, FP1–FP18, and the eight frontplanes, FP19–FP26, are shared with port C pins. FP0 and BP3 shares the same pin and configured by the DUTY[1:0] bits in the LCD clock register.

The full names of the LCD output pins are shown in **Table 16-1**. The generic pin names appear in the text that follows.

Table 16-1. Pin Name Conventions

LCD Generic Pin Name	Full MCU Pin Name	Pin Selected for LCD Function by:
FP0/BP3	FP0/BP3	—
BP0–BP2	BP0–BP2	—
FP1–FP18	FP1–FP18	—
FP19–FP22	PTC0/FP19–PTC3/FP22	PCEL in CONFIG2
FP23–FP26	PTC4/FP23–PTC7/FP26	PCEH in CONFIG2

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$004F	LCD Clock Register (LCDCLK)	Read:	0	FCCTL1	FCCTL0	DUTY1	DUTY0	LCLK2	LCLK1	LCLK0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0051	LCD Control Register (LCDCR)	Read:	LCDE	0	FC	LC	LCCON3	LCCON2	LCCON1	LCCON0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0052	LCD Data Register 1 (LDAT1)	Read:	F1B3	F1B2	F1B1	F1B0	F0B3	F0B2	F0B1	F0B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0053	LCD Data Register 2 (LDAT2)	Read:	F3B3	F3B2	F3B1	F3B0	F2B3	F2B2	F2B1	F2B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0054	LCD Data Register 3 (LDAT3)	Read:	F5B3	F5B2	F5B1	F5B0	F4B3	F4B2	F4B1	F4B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0055	LCD Data Register 4 (LDAT4)	Read:	F7B3	F7B2	F7B1	F7B0	F6B3	F6B2	F6B1	F6B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0056	LCD Data Register 5 (LDAT5)	Read:	F9B3	F9B2	F9B1	F9B0	F8B3	F8B2	F8B1	F8B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0057	LCD Data Register 6 (LDAT6)	Read:	F11B3	F11B2	F11B1	F11B0	F10B3	F10B2	F10B1	F10B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0058	LCD Data Register 7 (LDAT7)	Read:	F13B3	F13B2	F13B1	F13B0	F12B3	F12B2	F12B1	F12B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0059	LCD Data Register 8 (LDAT8)	Read:	F15B3	F15B2	F15B1	F15B0	F14B3	F14B2	F14B1	F14B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U

U = Unaffected = Unimplemented

Figure 16-1. LCD I/O Register Summary

$005A	LCD Data Register 9 (LDAT9)	Read:	F17B3	F17B2	F17B1	F17B0	F16B3	F16B2	F16B1	F16B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$005B	LCD Data Register 10 (LDAT10)	Read:	F19B3	F19B2	F19B1	F19B0	F18B3	F18B2	F18B1	F18B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$005C	LCD Data Register 11 (LDAT11)	Read:	F21B3	F21B2	F21B1	F21B0	F20B3	F20B2	F20B1	F20B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$005D	LCD Data Register 12 (LDAT12)	Read:	F23B3	F23B2	F23B1	F23B0	F22B3	F22B2	F22B1	F22B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$005E	LCD Data Register 13 (LDAT13)	Read:	F25B3	F25B2	F25B1	F25B0	F24B3	F24B2	F24B1	F24B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$005F	LCD Data Register 14 (LDAT14)	Read:					F26B3	F26B2	F26B1	F26B0
		Write:								
		Reset:	U	U	U	U	U	U	U	U

U = Unaffected = Unimplemented

Figure 16-1. LCD I/O Register Summary

16.5 Functional Description

Figure 16-2 shows a block diagram of the LCD driver module, and **Figure 16-3** shows a simplified schematic of the LCD system.

The LCD driver module uses a 1/3 biasing method. The LCD power is supplied by MCU power supply V_{DD}. Voltages V_{LCD1}, V_{LCD2}, and V_{LCD3} are generated by an internal resistor ladder.

The LCD data registers, LDAT1–LDAT14, control the LCD segments' ON/OFF, with each data register controlling two frontplanes. When a logic 1 is written to a FxBx bit in the data register, the corresponding frontplane-backplane segment will turn ON. When a logic 0 is written, the the segment will turn OFF.

When the LCD driver module is disabled (LCDE = 0), the LCD display will be OFF, all backplane and frontplane drivers have the same potential as V_{DD}. The resistor ladder is disconnected from V_{DD} to reduce power consumption.

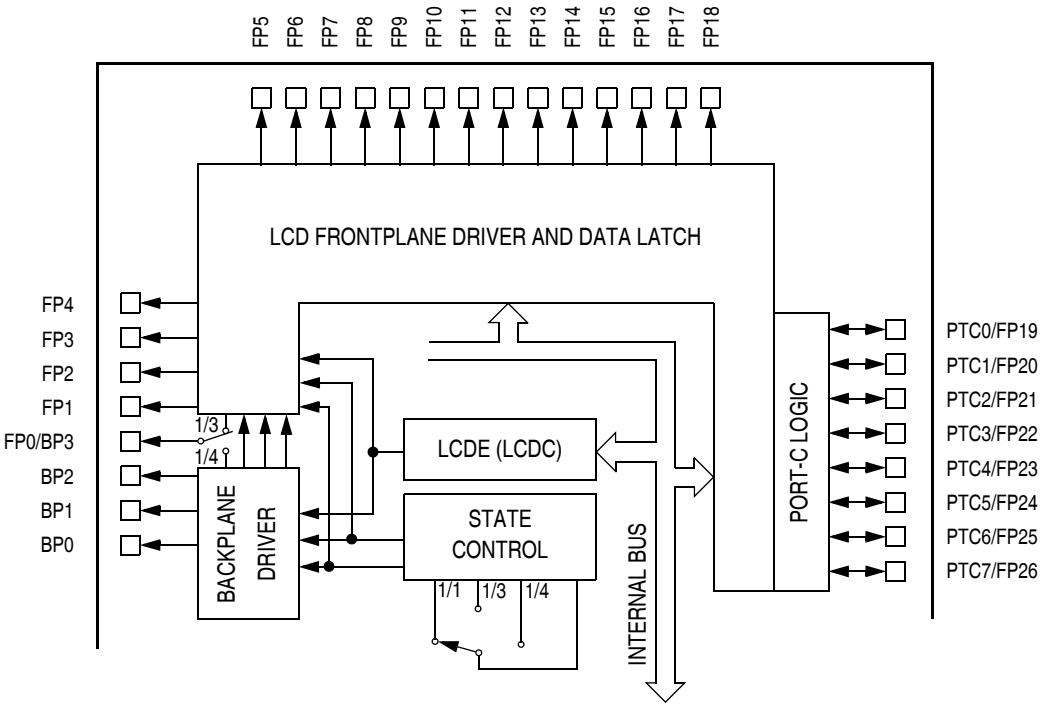


Figure 16-2. LCD Block Diagram

16.5.1 LCD Duty

The setting of the LCD output waveform duty is dependent on the number of backplane drivers required. Three LCD duties are available:

- Static duty — BP0 is used only
- 1/3 duty — BP0, BP1, and BP3 are used
- 1/4 duty — BP0, BP1, BP2, and BP3 are used

When the LCD driver module is enabled the backplane waveforms for the selected duty are driven out of the backplane pins. The backplane waveforms are periodic and are shown are shown in **Figure 16-6**, **Figure 16-5**, and **Figure 16-7**.

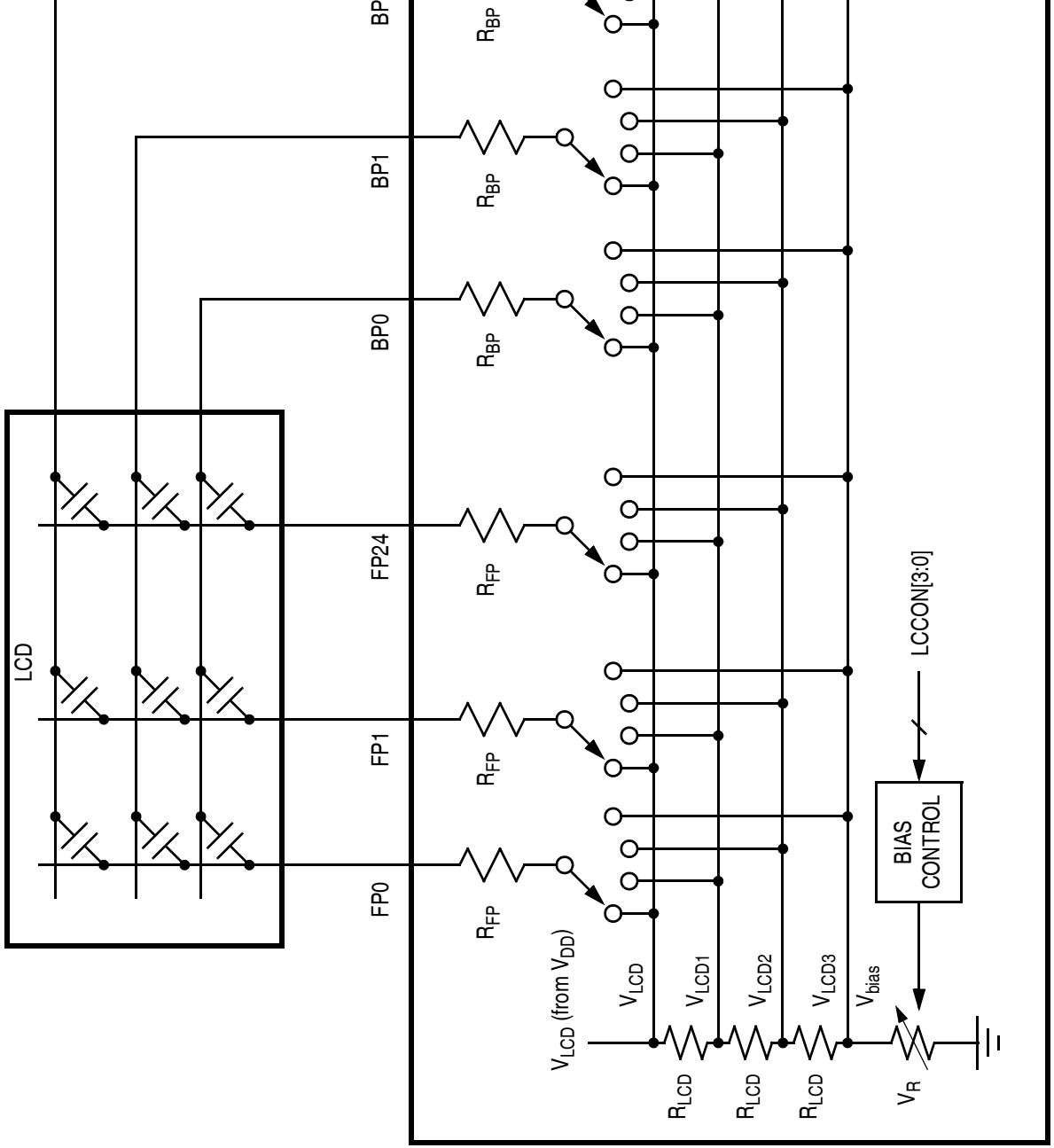


Figure 16-3. Simplified LCD Schematic (1/3 Duty, 1/3 Bias)

16.5.2 LCD Voltages (V_{LCD}, V_{LCD1}, V_{LCD2}, V_{LCD3})

The voltage V_{LCD} is connected directly to V_{DD}. V_{LCD1}, V_{LCD2}, and V_{LCD3} are internal bias voltages for the LCD driver waveforms. They are derived from V_{LCD} using a resistor ladder (see **Figure 16-3**).

The relative potential of the LCD voltages are:

- $V_{LCD} = V_{DD}$
- $V_{LCD1} = 2/3 \times (V_{LCD} - V_{bias})$
- $V_{LCD2} = 1/3 \times (V_{LCD} - V_{bias})$
- $V_{LCD3} = V_{bias}$

The V_{LCD3} bias voltage, V_{bias}, is controlled by the LCD contrast control bits, LCCON[2:0].

16.5.3 LCD Cycle Frame

The LCD driver module uses the CGMXCLK (see **Section 7. Oscillator (OSC)**) as the input reference clock. This clock is divided to produce the LCD waveform base clock, LCDCLK, by configuring the LCLK[2:0] bits in the LCD clock register. The LCDCLK clocks the backplane and the frontplane output waveforms.

The LCD cycle frame is determined by the equation:

$$\text{LCD CYCLE FRAME} = \frac{1}{\text{LCD WAVEFORM BASE CLOCK} \times \text{DUTY}}$$

For example, for 1/3 duty and 256 Hz waveform base clock:

$$\text{LCD CYCLE FRAME} = \frac{1}{256 \times (1/3)}$$

$$= 11.72 \text{ ms}$$

16.5.4 Fast Charge and Low Current

The default value for each of the bias resistors (see **Figure 16-3**), R_{LCD}, in the resistor ladder is approximately 37kΩ at V_{LCD} = 3V. The relatively high current drain through the 37kΩ resistor ladder may not be suitable for some LCD panel connections. Lowering this current is possible by setting the LC bit in the LCD control register, switching the R_{LCD} value to 146kΩ.

Although the lower current drain is desirable, but in some LCD panel connections, the higher current is required to drive the capacitive load of the LCD panel. In most cases, the higher current is only required when the LCD waveforms change state (the rising and falling edges in the LCD output waveforms). The fast charge option is designed to have the high current for the switching and the low current for the steady state. Setting the FC bit in the LCD control register selects the fast charge option. The R_{LCD} value is set to 37kΩ (for high current) for a fraction of time for each LCD waveform switching edge, and then back to 146kΩ for the steady state period. The duration of the fast charge time is set by configuring the FCCTL[1:0] bits in the LCD clock register, and can be LCDCLK/32, LCDCLK/64, or LCDCLK/128. **Figure 16-4** shows the fast charge clock relative to the BP0 waveform.

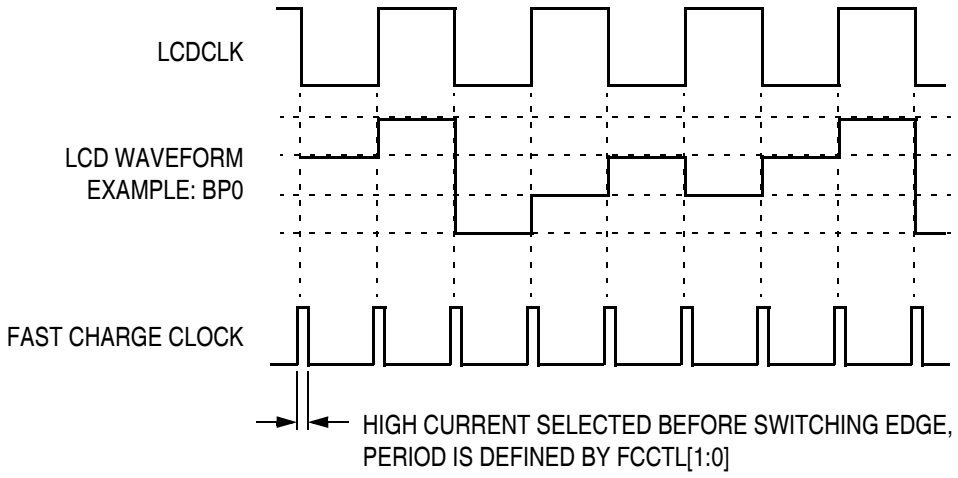


Figure 16-4. Fast Charge Timing

16.5.5 Contrast Control

The contrast of the connected LCD panel can be adjusted by configuring the LCCON[3:0] bits in the LCD control register. The LCCON[3:0] bits provide a 16-step contrast control, which adjusts the bias voltage in the resistor ladder for LCD voltage V_{LCD3}. The relative voltages, V_{LCD1} and V_{LCD2}, are altered according. For example, setting LCCON[3:0] = $F, the relative panel potential voltage ($V_{LCD} - V_{LCD3}$) is reduced from maximum 3.3V to approximate 2.45V.

16.6 Low-Power Modes

The STOP and WAIT instructions put the MCU in low power-consumption standby modes.

16.6.1 Wait Mode

The LCD driver module continues normal operation in wait mode. If the LCD is not required in wait mode, power down the LCD module by clearing the LCDE bit before executing the WAIT instruction.

16.6.2 Stop Mode

For continuous LCD module operation in stop mode, the oscillator stop mode enable bit (STOP_XCLKEN in CONFIG2 register) must be set before executing the STOP instruction. When STOP_XCLKEN is set, CGMXCLK continues to drive the LCD module.

If STOP_XCLKEN bit is cleared, the LCD module is inactive after the execution of a STOP instruction. The STOP instruction does not affect LCD register states. LCD module operation resumes after an external interrupt. To further reduce power consumption, the LCD module should be powered-down by clearing the LCDE bit before executing the STOP instruction.

16.7 I/O Signals

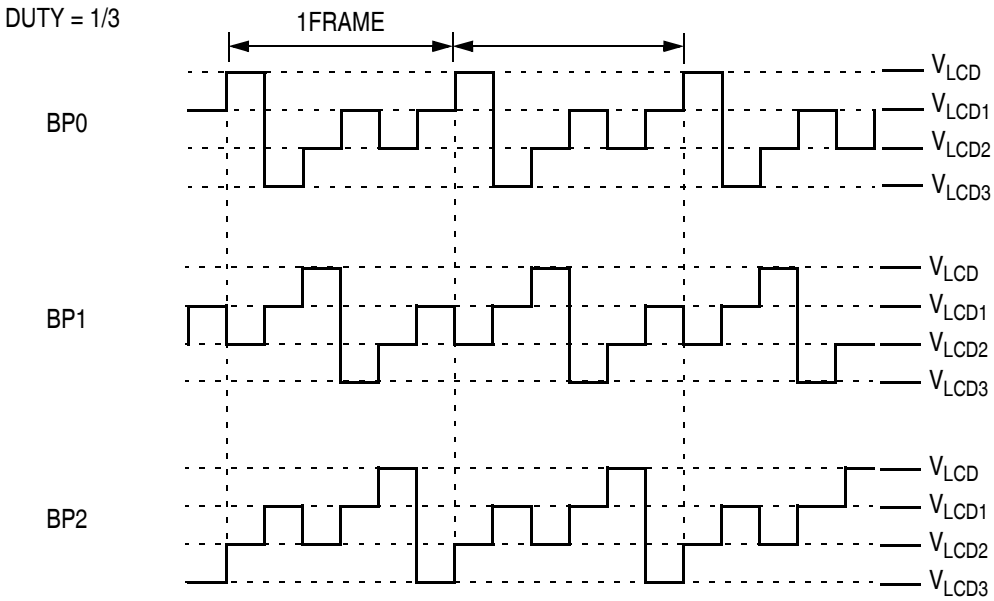
The LCD driver module has thirty (30) output pins and shares eight of them with port C I/O pins.

- FP0/BP3 (multiplexed; selected as FP0 or BP3 by DUTY[1:0])
- BP0–BP2
- FP1–FP26 (FP19–FP26 shared with port C)

16.7.1 BP0–BP3 (Backplane Drivers)

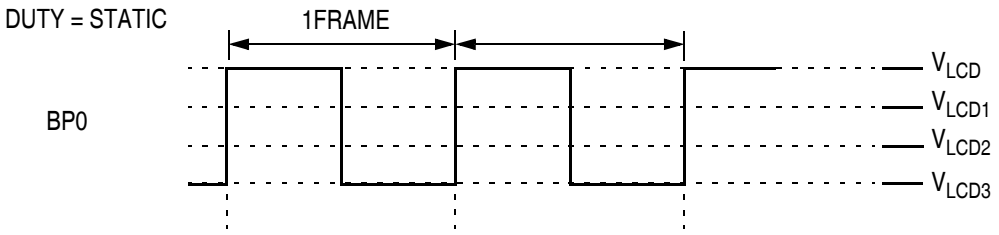
BP0–BP3 are the backplane driver output pins. These are connected to the backplane of the LCD panel. Depending on the LCD duty selected, the voltage waveforms in **Figure 16-6**, **Figure 16-5**, and **Figure 16-7** appear on the backplane pins.

BP3 pin is only used when 1/4 duty is selected. The pin becomes FP0 for static and 1/3 duty operations.



NOTES:
1. BP3 is not used.
2. At 1/3 duty, 1FRAME has three times the cycle of LCD waveform base clock.

Figure 16-5. 1/3 Duty LCD Backplane Driver Waveforms



NOTES:
1. BP1, BP2, and BP3 are not used.
2. At static duty, 1FRAME is equal to the cycle of LCD waveform base clock.

Figure 16-6. Static LCD Backplane Driver Waveform

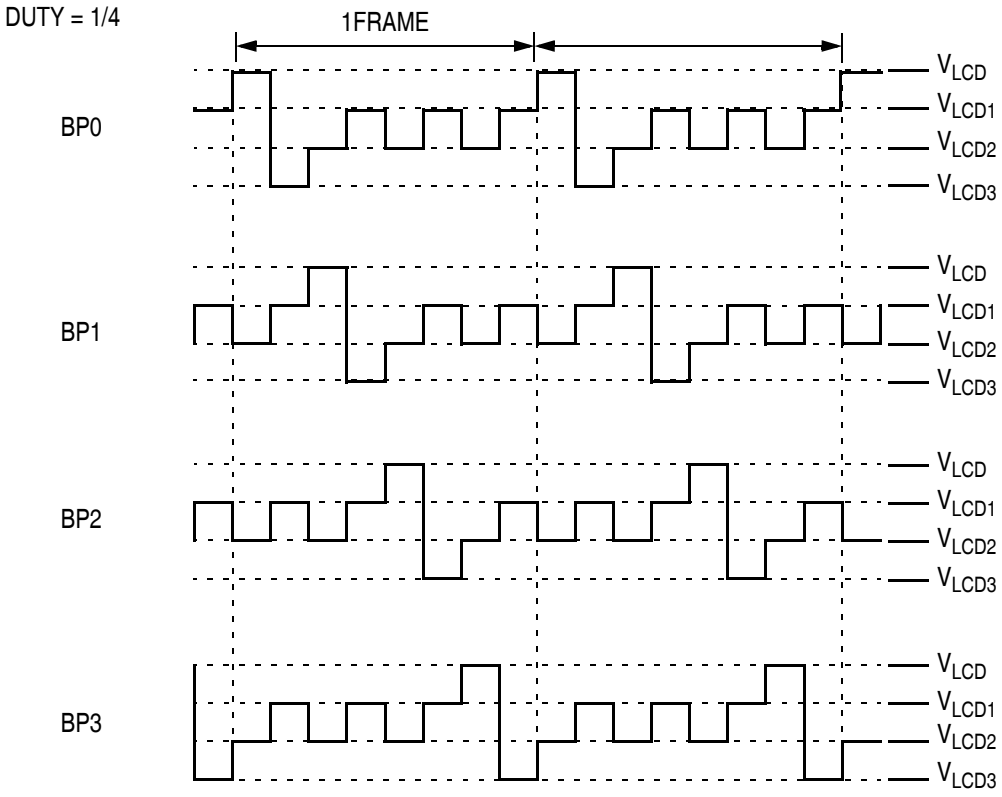


Figure 16-7. 1/4 Duty LCD Backplane Driver Waveforms

16.7.2 FP0–FP26 (Frontplane Drivers)

FP0–FP26 are the frontplane driver output pins. These are connected to the frontplane of the LCD panel. Depending on LCD duty selected and the contents in the LCD data registers, the voltage waveforms in **Figure 16-8**, **Figure 16-9**, and **Figure 16-10** appear on the frontplane pins.

FP19–FP26 are shared with port C I/O pins. These pins are configured for standard I/O or LCD use by the PCEL and PCEH bits in CONFIG2 register.

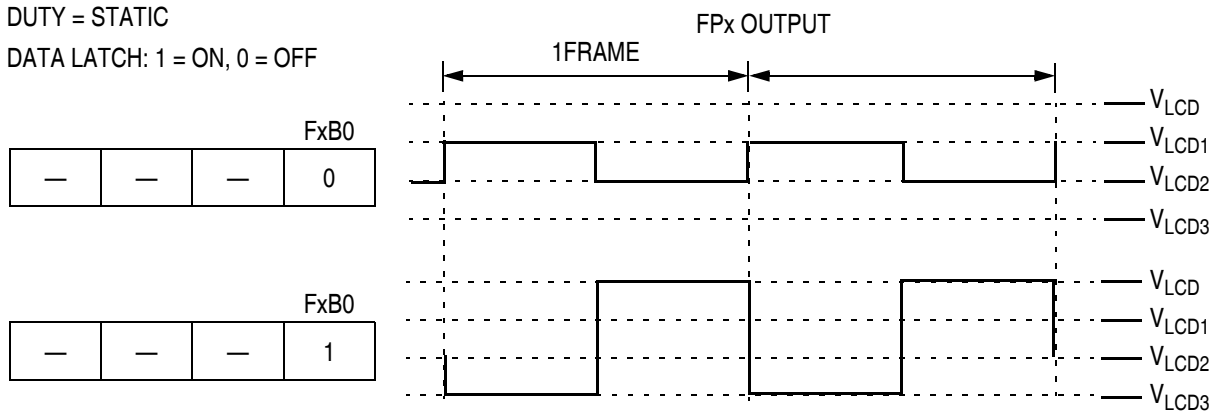


Figure 16-8. Static LCD Frontplane Driver Waveforms

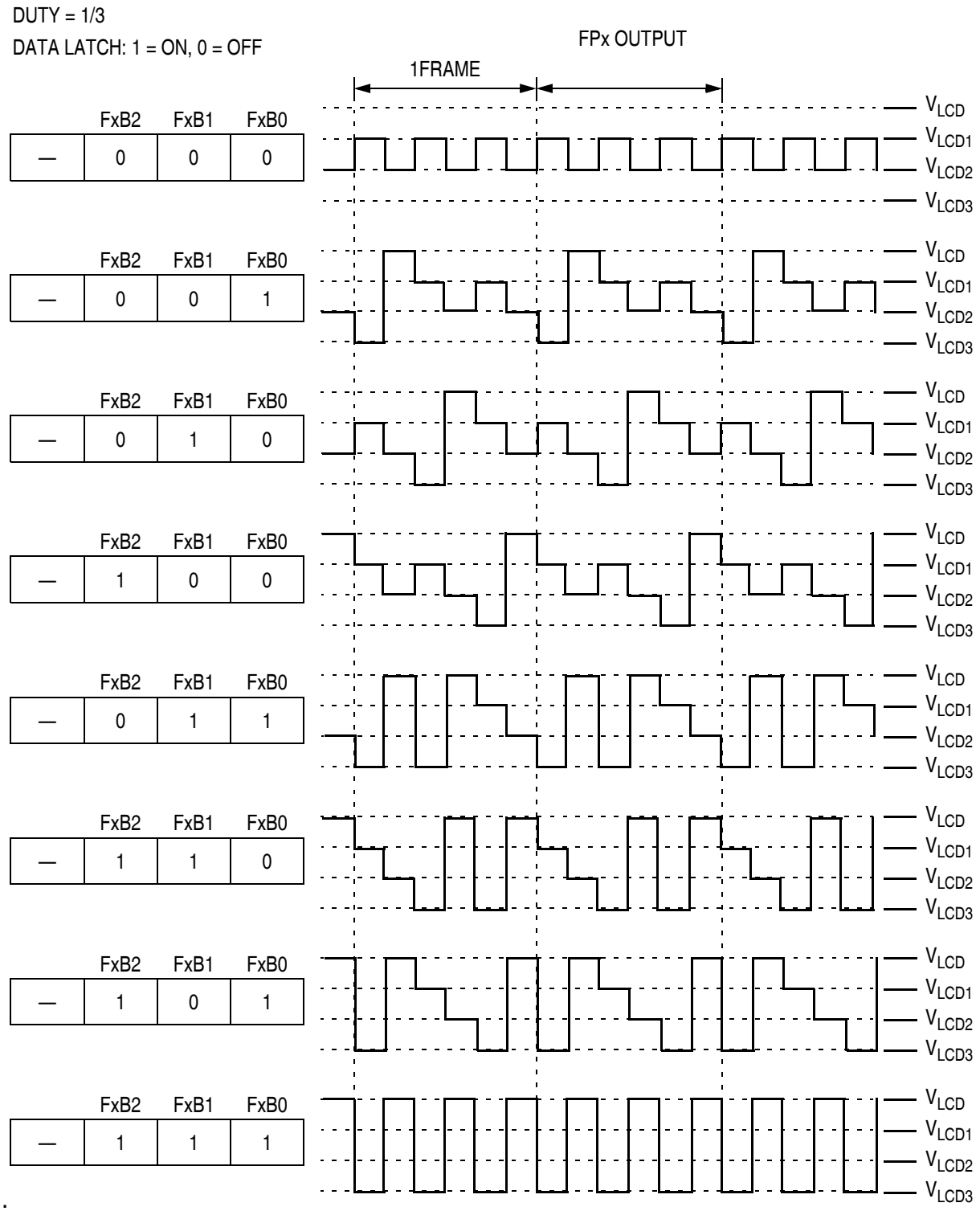


Figure 16-9. 1/3 Duty LCD Frontplane Driver Waveforms

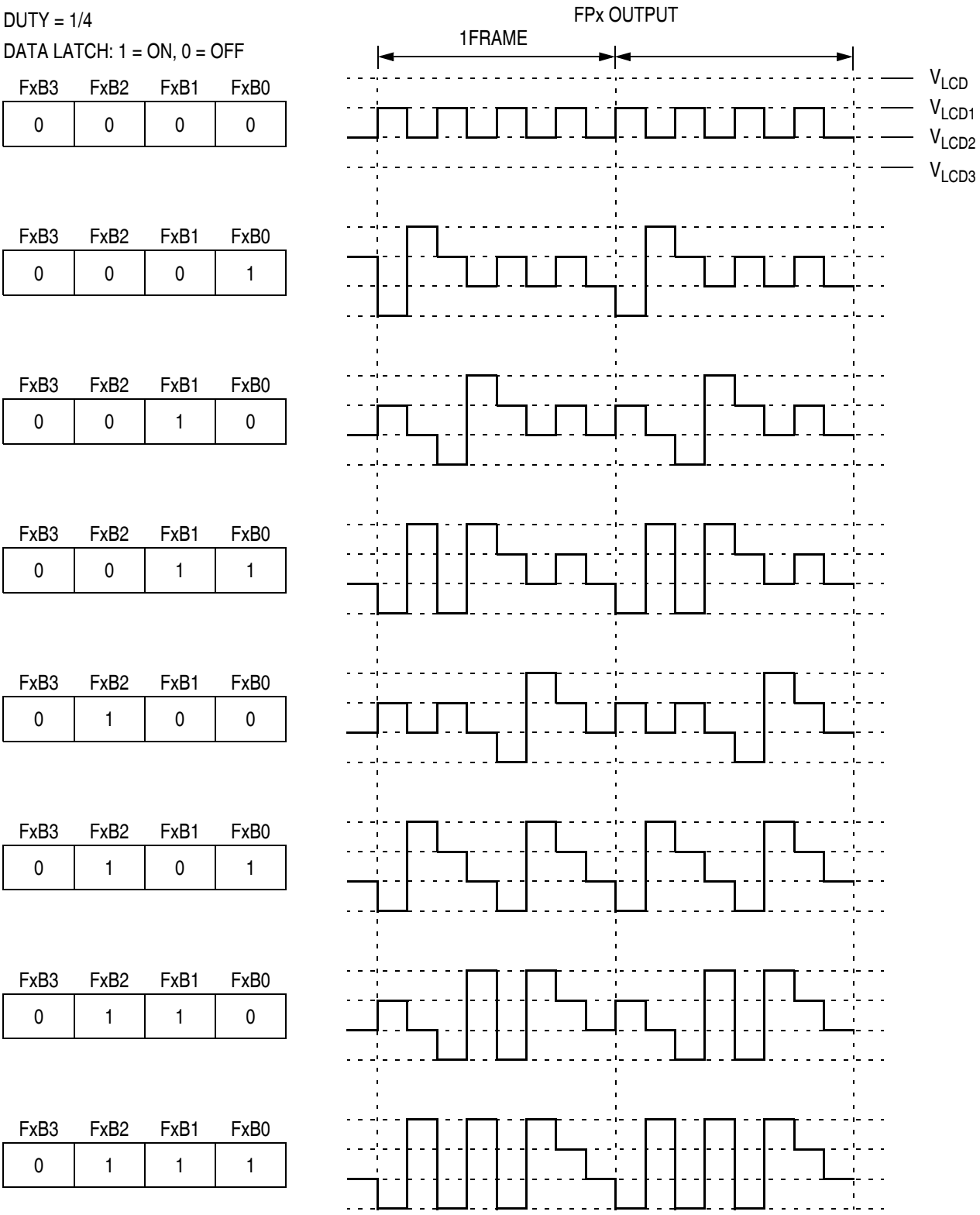


Figure 16-10. 1/4 Duty LCD Frontplane Driver Waveforms

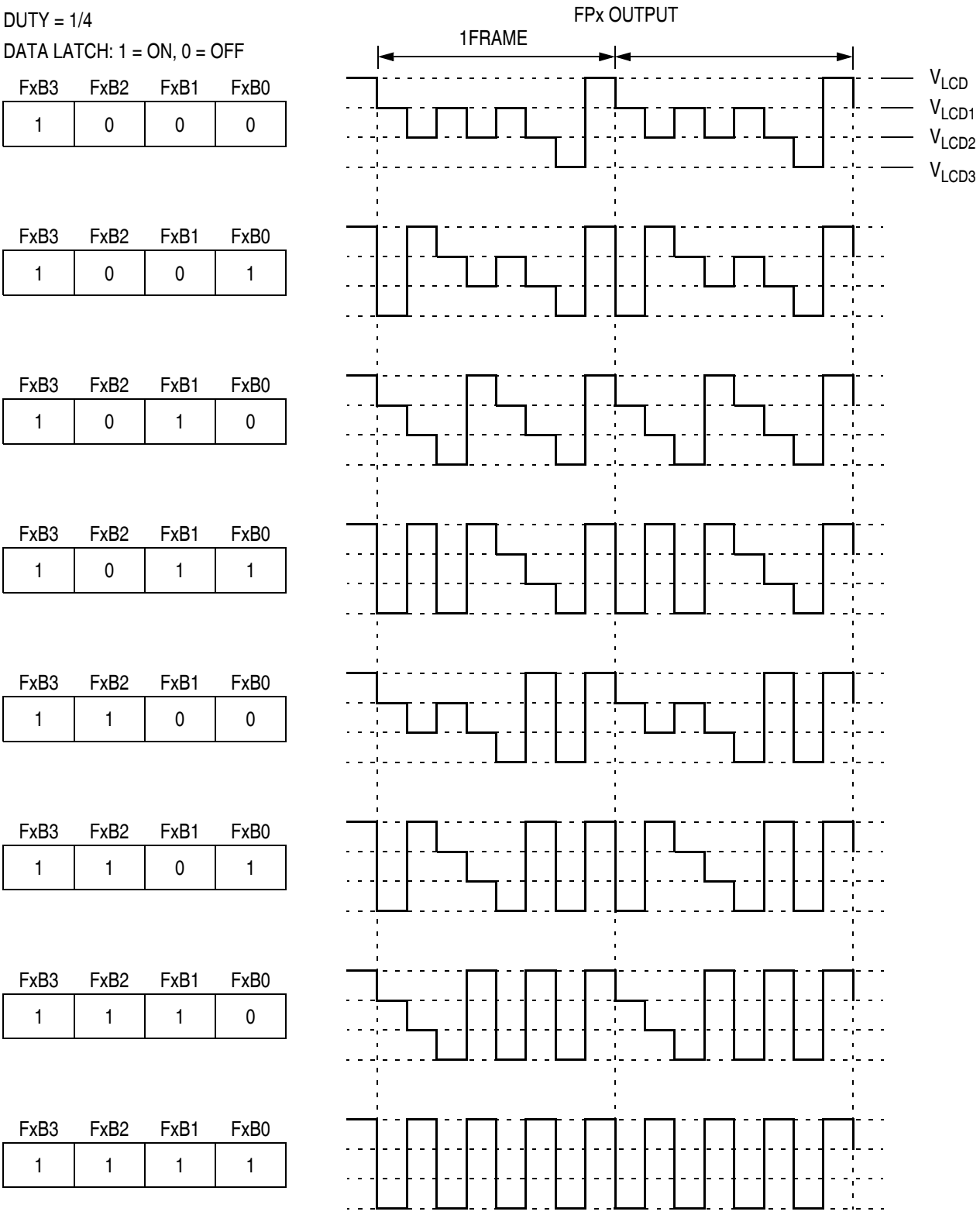


Figure 16-11. 1/4 Duty LCD Frontplane Driver Waveforms (continued)

16.8 Seven Segment Display Connection

The following shows an example for connecting a 7-segment LCD display to the LCD driver.

The example uses 1/3 duty cycle, with pins BP0, BP1, BP2, FP0, FP1, and FP2 connected as shown in **Figure 16-12**. The output waveforms are shown in **Figure 16-13**.

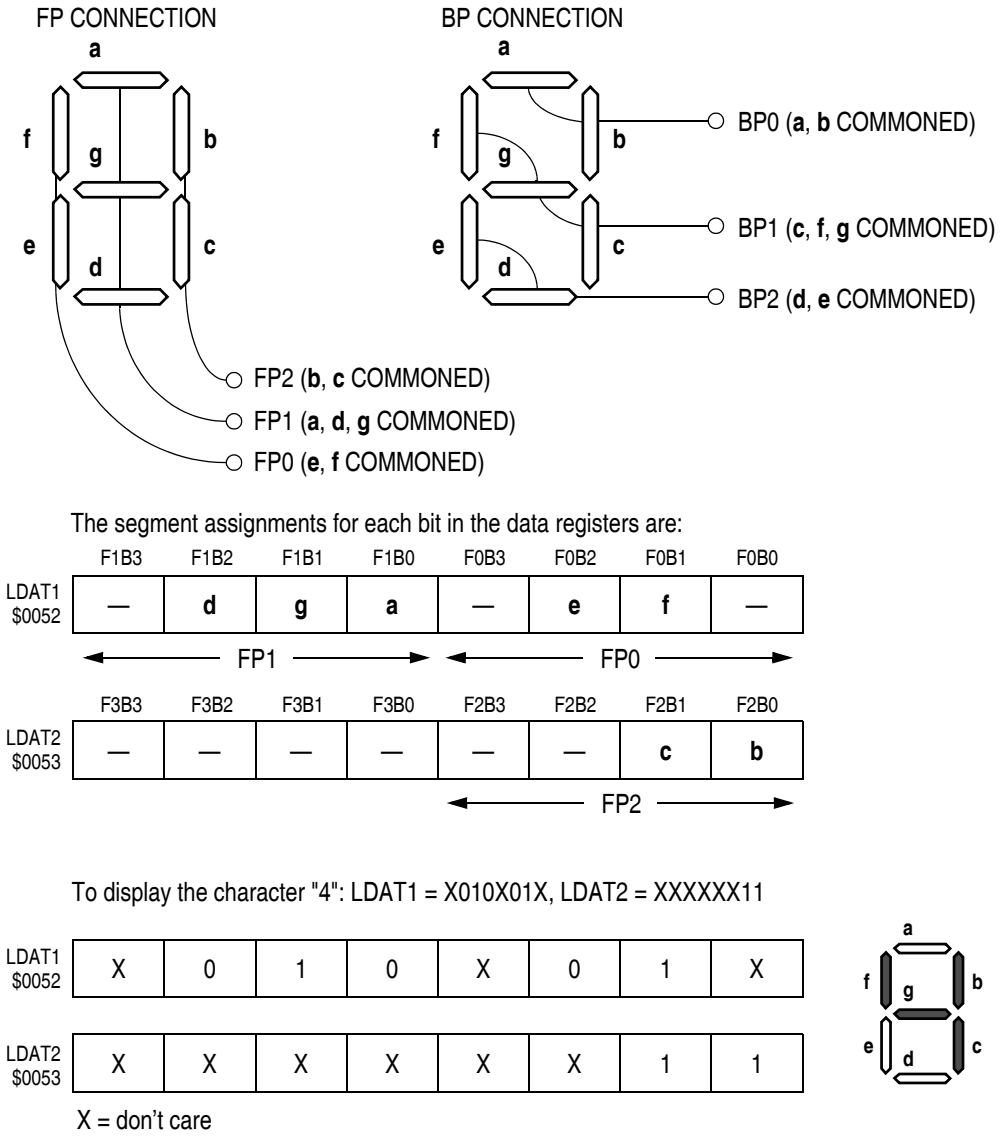


Figure 16-12. 7-Segment Display Example

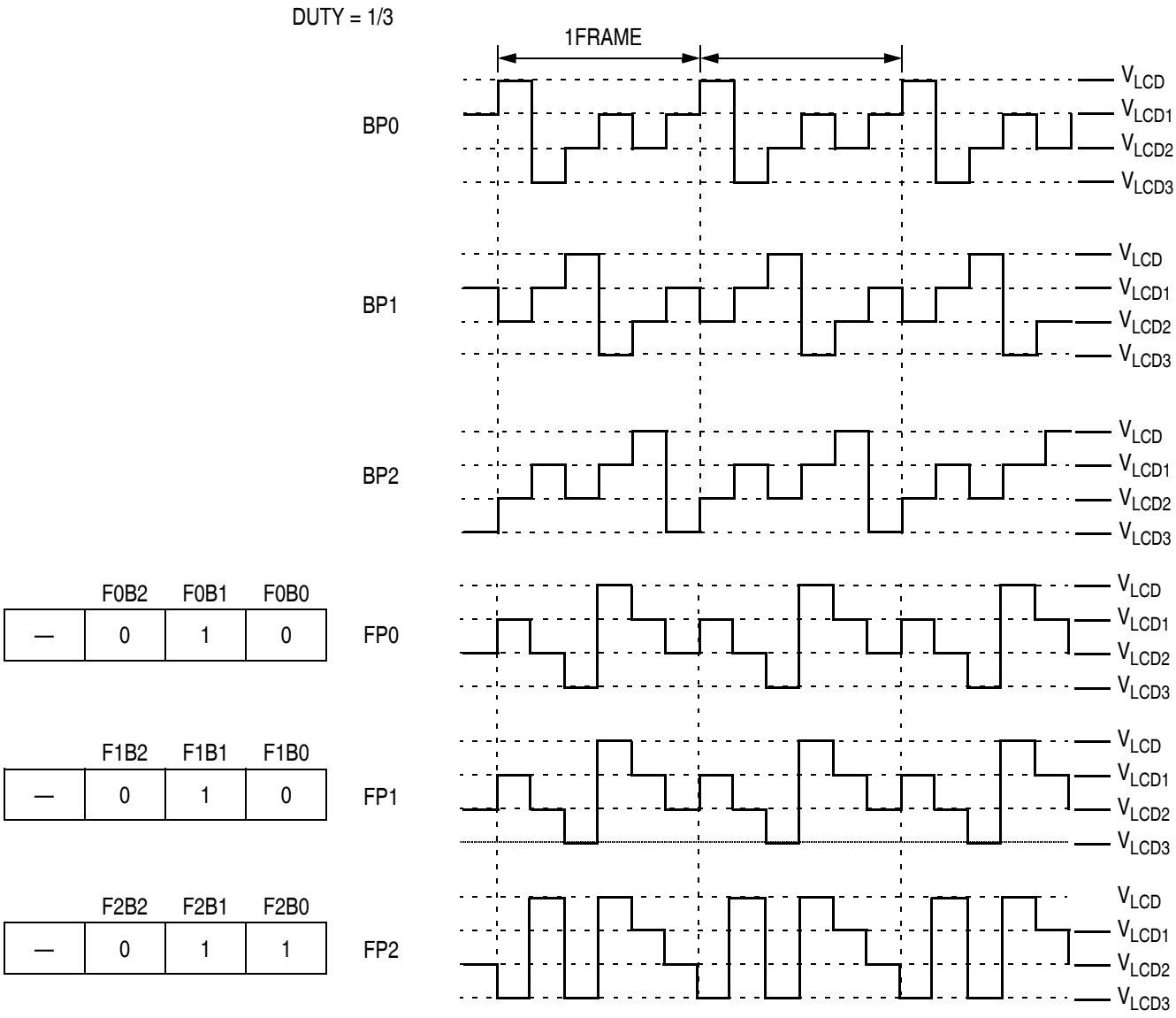


Figure 16-13. BP0–BP2 and FP0–FP2 Output Waveforms for 7-Segment Display Example

The voltage waveform across the "f" segment of the LCD (between BP1 and FP0) is illustrated in **Figure 16-14**. As shown in the waveform, the voltage peaks reach the LCD-ON voltage, V_{LCD}, therefore, the segment will be ON.

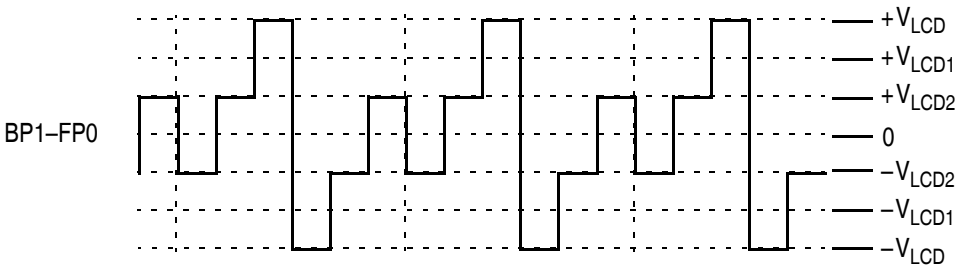


Figure 16-14. "f" Segment Voltage Waveform

The voltage waveform across the "e" segment of the LCD (between BP2 and FP0) is illustrated in **Figure 16-15**. As shown in the waveform, the voltage peaks do not reach the LCD-ON voltage, V_{LCD}, therefore, the segment will be OFF.

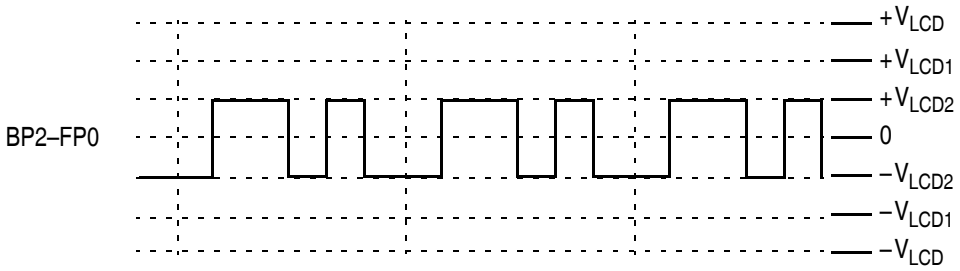


Figure 16-15. "e" Segment Voltage Waveform

16.9 I/O Registers

Sixteen (16) registers control LCD driver module operation:

- LCD control register (LCDCR)
- LCD clock register (LCDCLK)
- LCD data registers (LDAT1–LDAT14)

16.9.1 LCD Control Register (LCDCR)

The LCD control register (LCDCR):

- Enables the LCD driver module
- Selects bias resistor value and fast-charge control
- Selects LCD contrast

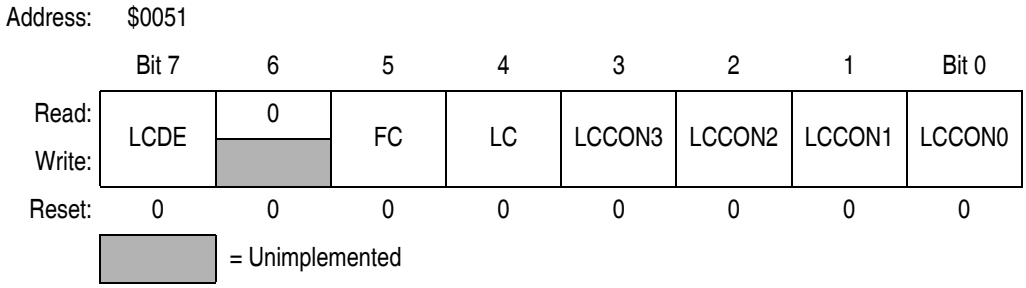


Figure 16-16. LCD Control Register (LCDCR)

LCDE — LCD Enable

This read/write bit enables the LCD driver module; the backplane and frontplane drive LCD waveforms out of BPx and FPx pins. Reset clears the LCDE bit.

1 = LCD driver module enabled
0 = LCD driver module disabled

FC — Fast Charge
LC — Low Current

These read/write bits are used to select the value of the resistors in resistor ladder for LCD voltages. Reset clears the FC and LC bits.

Table 16-2. Resistor Ladder Selection

FC	LC	Action
X	0	Each resistor is approximately 37 kΩ (default)
0	1	Each resistor is approximately 146 kΩ
1	1	Fast charge mode

LCCON[3:0] — LCD Contrast Control

These read/write bits select the bias voltage, V_{bias}. This voltage controls the contrast of the LCD. Maximum contrast is set when LCCON[3:0] = 0000; minimum contrast is when LCCON[3:0] = 1111.

Table 16-3. LCD Bias Voltage Control

LCCON3	LCCON2	LCCON1	LCCON0	Bias Voltage (% of V_{DD})
0	0	0	0	0.6%
0	0	0	1	2.9%
0	0	1	0	5.2%
0	0	1	1	7.4%
0	1	0	0	9.6%
0	1	0	1	11.6%
0	1	1	0	13.5%
0	1	1	1	15.3%
1	0	0	0	17.2%
1	0	0	1	18.8%
1	0	1	0	20.5%
1	0	1	1	22.0%
1	1	0	0	23.6%
1	1	0	1	25.0%
1	1	1	0	26.4%
1	1	1	1	27.7%

16.9.2 LCD Clock Register (LCDCLK)

The LCD clock register (LCDCLK):

- Selects the fast charge duty cycle
- Selects LCD driver duty cycle
- Selects LCD waveform base clock

Address: $004F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	FCCTL1	FCCTL0	DUTY1	DUTY0	LCLK2	LCLK1	LCLK0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-17. LCD Clock Register (LCDCLK)

FCCTL[1:0] — Fast Charge Duty Cycle Select

These read/write bits select the duty cycle of the fast charge duration. Reset clears these bits. (See **16.5.4 Fast Charge and Low Current**)

Table 16-4. Fast Charge Duty Cycle Selection

FCCTL1:FCCTL0	Fast Charge Duty Cycle
00	In each LCDCLK/2 period, each bias resistor is reduced to 37 kΩ for a duration of LCDCLK/32.
01	In each LCDCLK/2 period, each bias resistor is reduced to 37 kΩ for a duration of LCDCLK/64.
10	In each LCDCLK/2 period, each bias resistor is reduced to 37 kΩ for a duration of LCDCLK/128.
11	Not used

DUTY[1:0] — Duty Cycle Select

These read/write bits select the duty cycle of the LCD driver output waveforms. The multiplexed FP0/BP3 pin is controlled by the duty cycle selected. Reset clears these bits.

Table 16-5. LCD Duty Cycle Selection

DUTY1:DUTY0	Description
00	Static selected; FP0/BP3 pin function as FP0.
01	1/3 duty cycle selected; FP0/BP3 pin functions as FP0.
10	1/4 duty cycle selected; FP0/BP3 pin functions as BP3.
11	Not used

LCLK[2:0] — LCD Waveform Base Clock Select

These read/write bits selects the LCD waveform base clock. Reset clears these bits.

Table 16-6. LCD Waveform Base Clock Selection

LCLK2	LCLK1	LCLK0	Divide Ratio	LCD Waveform Base Clock Frequency LCDCLK (Hz)		LCD Frame Rate f_{XTAL} (1) = 32.768 kHz		LCD Frame Rate f_{XTAL} = 4.9152 MHz	
				f_{XTAL} = 32.768 kHz	f_{XTAL} = 4.9152 MHz	1/3 duty	1/4 duty	1/3 duty	1/4 duty
0	0	0	128	256	—	85.3	64	—	—
0	0	1	256	128	—	42.7	32	—	—
0	1	0	512	64	—	21.3	16	—	—
0	1	1	1024	32	—	10.7	8	—	—
1	0	0	16384	—	300	—	—	100	75
1	0	1	32768	—	150	—	—	50	37.5
1	1	0	65536	—	75	—	—	25	18.75
1	1	1	Reserved						

Notes:
1. f_{XTAL} is the same as CGMXCLK (see **Section 7. Oscillator (OSC)**).

16.9.3 LCD Data Registers (LDAT1–LDAT14)

The fourteen (14) LCD data registers enable and disable the drive to the corresponding LCD segments.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0052	LCD Data Register 1 (LDAT1)	Read: Write:	F1B3	F1B2	F1B1	F1B0	F0B3	F0B2	F0B1	F0B0
		Reset:	U	U	U	U	U	U	U	U
$0053	LCD Data Register 2 (LDAT2)	Read: Write:	F3B3	F3B2	F3B1	F3B0	F2B3	F2B2	F2B1	F2B0
		Reset:	U	U	U	U	U	U	U	U
$0054	LCD Data Register 3 (LDAT3)	Read: Write:	F5B3	F5B2	F5B1	F5B0	F4B3	F4B2	F4B1	F4B0
		Reset:	U	U	U	U	U	U	U	U
$0055	LCD Data Register 4 (LDAT4)	Read: Write:	F7B3	F7B2	F7B1	F7B0	F6B3	F6B2	F6B1	F6B0
		Reset:	U	U	U	U	U	U	U	U
$0056	LCD Data Register 5 (LDAT5)	Read: Write:	F9B3	F9B2	F9B1	F9B0	F8B3	F8B2	F8B1	F8B0
		Reset:	U	U	U	U	U	U	U	U
$0057	LCD Data Register 6 (LDAT6)	Read: Write:	F11B3	F11B2	F11B1	F11B0	F10B3	F10B2	F10B1	F10B0
		Reset:	U	U	U	U	U	U	U	U
$0058	LCD Data Register 7 (LDAT7)	Read: Write:	F13B3	F13B2	F13B1	F13B0	F12B3	F12B2	F12B1	F12B0
		Reset:	U	U	U	U	U	U	U	U
$0059	LCD Data Register 8 (LDAT8)	Read: Write:	F15B3	F15B2	F15B1	F15B0	F14B3	F14B2	F14B1	F14B0
		Reset:	U	U	U	U	U	U	U	U
$005A	LCD Data Register 9 (LDAT9)	Read: Write:	F17B3	F17B2	F17B1	F17B0	F16B3	F16B2	F16B1	F16B0
		Reset:	U	U	U	U	U	U	U	U

U = Unaffected = Unimplemented

Figure 16-18. LCD Data Registers 1–14 (LDAT1–LDAT14)

$005B	LCD Data Register 10 (LDAT10)	Read: Write:	F19B3	F19B2	F19B1	F19B0	F18B3	F18B2	F18B1	F18B0
		Reset:	U	U	U	U	U	U	U	U
$005C	LCD Data Register 11 (LDAT11)	Read: Write:	F21B3	F21B2	F21B1	F21B0	F20B3	F20B2	F20B1	F20B0
		Reset:	U	U	U	U	U	U	U	U
$005D	LCD Data Register 12 (LDAT12)	Read: Write:	F23B3	F23B2	F23B1	F23B0	F22B3	F22B2	F22B1	F22B0
		Reset:	U	U	U	U	U	U	U	U
$005E	LCD Data Register 13 (LDAT13)	Read: Write:	F25B3	F25B2	F25B1	F25B0	F24B3	F24B2	F24B1	F24B0
		Reset:	U	U	U	U	U	U	U	U
$005F	LCD Data Register 14 (LDAT14)	Read: Write:					F26B3	F26B2	F26B1	F26B0
		Reset:	U	U	U	U	U	U	U	U

U = Unaffected = Unimplemented

Figure 16-18. LCD Data Registers 1–14 (LDAT1–LDAT14)

Section 17. Input/Output (I/O) Ports

17.1 Contents

17.2 Introduction 341

17.3 Port A 344
17.3.1 Port A Data Register (PTA) 344
17.3.2 Data Direction Register A (DDRA) 345

17.4 Port B 347
17.4.1 Port B Data Register (PTB) 347
17.4.2 Data Direction Register B (DDRB) 348
17.4.3 Port B LED Control Register (LEDB) 350

17.5 Port C 351
17.5.1 Port C Data Register (PTC) 351
17.5.2 Data Direction Register C (DDRC) 352

17.6 Port D 354
17.6.1 Port D Data Register (PTD) 354
17.6.2 Data Direction Register D (DDRD) 355

17.2 Introduction

Thirty-two (32) bidirectional input-output (I/O) pins form four parallel ports. All I/O pins are programmable as inputs or outputs.

NOTE: *Connect any unused I/O pins to an appropriate logic level, either V_{DD} or V_{SS}. Although the I/O ports do not require termination for proper operation, termination reduces excess current consumption and the possibility of electrostatic damage.*

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PTA)	Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
		Reset:	Unaffected by reset							
$0001	Port B Data Register (PTB)	Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Reset:	Unaffected by reset							
$0002	Port C Data Register (PTC)	Read: Write:	PTC7	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
		Reset:	Unaffected by reset							
$0003	Port D Data Register (PTD)	Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Reset:	Unaffected by reset							
$0004	Data Direction Register A (DDRA)	Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B (DDRB)	Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C (DDRC)	Read: Write:	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Reset:	0	0	0	0	0	0	0	0
$0007	Data Direction Register D (DDRD)	Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$000C	Port-B LED Control Register (LEDB)	Read:	0	0	LEDB5	LEDB4	LEDB3	LEDB2	LEDB1	LEDB0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

Figure 17-1. I/O Port Register Summary

Table 17-1. Port Control Register Bits Summary

Port	Bit	DDR	Module Control			Pin
			Module	Register	Control Bit	
A	0	DDRA0	KBI	KBIER ($001C)	KBIE0	PTA0/KBI0
	1	DDRA1			KBIE1	PTA1/KBI1
	2	DDRA2			KBIE2	PTA2/KBI2
	3	DDRA3			KBIE3	PTA3/KBI3
	4	DDRA4	ADC	ADSCR ($003C)	ADCH[4:0]	PTA4/ATD0
	5	DDRA5				PTA5/ATD1
	6	DDRA6				PTA6/ATD2
	7	DDRA7				PTA7/ATD3
B	0	DDRB0	SCI	SCC1 ($0013)	ENSCI	PTB0/TxD
	1	DDRB1				PTB1/RxD
	2	DDRB2	TIM1	T1SC0 ($0025)	ELS0B:ELS0A	PTB2/T1CH0
	3	DDRB3		T1SC1 ($0028)	ELS1B:ELS1A	PTB3/T1CH1
	4	DDRB4	TIM2	T2SC0 ($0030)	ELS0B:ELS0A	PTB4/T2CH0
	5	DDRB5		T2SC1 ($0033)	ELS1B:ELS1A	PTB5/T2CH1
	6	DDRB6	ADC	ADSCR ($003C)	ADCH[4:0]	PTB6/ATD4
	7	DDRB7				PTB7/ATD5
C	0	DDRC0	LCD	CONFIG2 ($001D)	PCEL	PTC0/FP19
	1	DDRC1				PTC1/FP20
	2	DDRC2				PTC2/FP21
	3	DDRC3				PTC3/FP22
	4	DDRC4			PCEH	PTC4/FP23
	5	DDRC5				PTC5/FP24
	6	DDRC6				PTC6/FP25
	7	DDRC7				PTC7/FP26
D	0	DDRD0	SPI	SPCR ($0010)	SPE	PTD0/$\overline{SS}$
	1	DDRD1				PTD1/MISO
	2	DDRD2				PTD2/MOSI
	3	DDRD3				PTD3/SPSCK
	4	DDRD4	KBI	KBIER ($001C)	KBIE4	PTD4/KBI4
	5	DDRD5			KBIE5	PTD5/KBI5
	6	DDRD6			KBIE6	PTD6/KBI6
	7	DDRD7			KBIE7	PTD7/KBI7

17.3 Port A

Port A is an 8-bit special function port that shares four of its port pins with the analog-to-digital converter (ADC) module and four of its port pins with the keyboard interrupt module (KBI).

17.3.1 Port A Data Register (PTA)

The port A data register contains a data latch for each of the eight port A pins.

Address: $0000

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
Reset:	Unaffected by Reset							
Alternative Function:	ADC3	ADC2	ADC1	ADC0	KBI3	KBI2	KBI1	KBI0

Figure 17-2. Port A Data Register (PTA)

PTA[7:0] — Port A Data Bits

These read/write bits are software programmable. Data direction of each port A pin is under the control of the corresponding bit in data direction register A. Reset has no effect on port A data.

KBI[3:0] — Keyboard interrupt channels 0 to 3

KBI[3:0] are input pins to the keyboard interrupt module. The corresponding control bits, KBIE[3:0], in the keyboard interrupt enable register, KBIER, select which port pins will be used as a keyboard interrupt input and overrides any control from the port I/O logic. See **Section 19. Keyboard Interrupt Module (KBI)**.

ADC[3:0] — ADC channels 3 to 0

ADC[3:0] are pins used for the input channels to the analog-to-digital converter module. The channel select bits, ADCH[4:0], in the ADC status and control register define which port pin will be used as an ADC input and overrides any control from the port I/O logic. See **Section 15. Analog-to-Digital Converter (ADC)**.

NOTE: *Care must be taken when reading port A while applying analog voltages to ADC[3:0] pins. If the appropriate ADC channel is not enabled, excessive current drain may occur if analog voltages are applied to the PTAx/ADCx pin, while PTA is read as a digital input. Those ports not selected as analog input channels are considered digital I/O ports.*

17.3.2 Data Direction Register A (DDRA)

Data direction register A determines whether each port A pin is an input or an output. Writing a logic 1 to a DDRA bit enables the output buffer for the corresponding port A pin; a logic 0 disables the output buffer.

Address: $0004

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
Reset:	0	0	0	0	0	0	0	0

Figure 17-3. Data Direction Register A (DDRA)

DDRA[7:0] — Data Direction Register A Bits

These read/write bits control port A data direction. Reset clears DDRA[7:0], configuring all port A pins as inputs.

1 = Corresponding port A pin configured as output
0 = Corresponding port A pin configured as input

NOTE: *Avoid glitches on port A pins by writing to the port A data register before changing data direction register A bits from 0 to 1. **Figure 17-4** shows the port A I/O logic.*

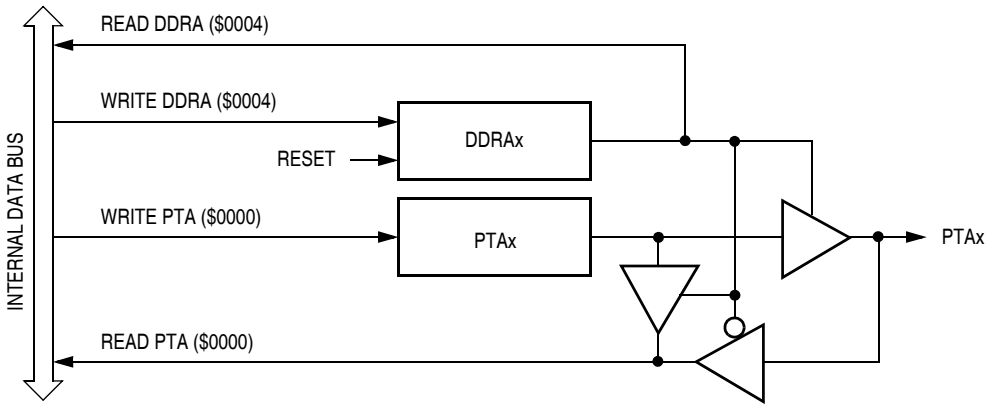


Figure 17-4. Port A I/O Circuit

When DDRAx is a logic 1, reading address $0000 reads the PTAx data latch. When DDRAx is a logic 0, reading address $0000 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit.

Table 17-2 summarizes the operation of the port A pins.

Table 17-2. Port A Pin Functions

DDRA Bit	PTA Bit	I/O Pin Mode	Accesses to DDRA	Accesses to PTA	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRA[7:0]	Pin	PTA[7:0](3)
1	X	Output	DDRA[7:0]	PTA[7:0]	PTA[7:0]

Notes:
1. X = don't care.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect input.

17.4 Port B

Port B is a 8-bit special function port that shares two of its port pins with the infrared serial communication interface (IRSCI) module, two of its port pins with the timer interface module 1 (TIM1) module, two of its port pins with the timer interface module 2 (TIM2), and two of its port pins with the ADC module.

Port pins PTB0–PTB5 can be configured for direct LED drive.

17.4.1 Port B Data Register (PTB)

The port B data register contains a data latch for each of the eight port B pins.

NOTE: *Bit 4–bit 7 of PTB are not available in a 52-pin LQFP.*

Address: $0001

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
Reset:	Unaffected by reset							
Alternative Function:	ADC5	ADC4	T2CH1	T2CH0	T1CH1	T1CH0	RxD	TxD
Additional Function:			LED drive	LED drive	LED drive	LED drive	LED drive	LED drive

Figure 17-5. Port B Data Register (PTB)

PTB[7:0] — Port B Data Bits

These read/write bits are software programmable. Data direction of each port B pin is under the control of the corresponding bit in data direction register B. Reset has no effect on port B data.

TxD, RxD — SCI Data I/O Pins

The TxD and RxD pins are the transmit data output and receive data input for the IRSCI module. The enable SCI bit, ENSCI, in the SCI control register 1 enables the PTB0/TxD and PTB1/RxD pins as SCI TxD and RxD pins and overrides any control from the port I/O. See **Section 13. Infrared Serial Communications Interface Module (IRSCI)**.

T1CH[1:0] — Timer 1 Channel I/O Bits

The T1CH1 and T1CH0 pins are the TIM1 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTB2/T1CH0 and PTB3/T1CH1 pins are timer channel I/O pins or general-purpose I/O pins. See **Section 11. Timer Interface Module (TIM)**.

T2CH[1:0] — Timer 2 Channel I/O Bits

The T2CH1 and T2CH0 pins are the TIM1 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTB4/T2CH0 and PTB5/T2CH1 pins are timer channel I/O pins or general-purpose I/O pins. See **Section 11. Timer Interface Module (TIM)**.

ADC[5:4] — ADC channels 5 and 4

ADC[5:4] are pins used for the input channels to the analog-to-digital converter module. The channel select bits, ADCH[4:0], in the ADC status and control register define which port pin will be used as an ADC input and overrides any control from the port I/O logic. See **Section 15. Analog-to-Digital Converter (ADC)**.

NOTE: *Care must be taken when reading port B while applying analog voltages to ADC[5:4] pins. If the appropriate ADC channel is not enabled, excessive current drain may occur if analog voltages are applied to the PTBx/ADCx pin, while PTB is read as a digital input. Those ports not selected as analog input channels are considered digital I/O ports.*

LED drive — Direct LED Drive Pins

PTB0–PTB5 pins can be configured for direct LED drive. See **17.4.3 Port B LED Control Register (LEDB)**.

17.4.2 Data Direction Register B (DDRB)

Data direction register B determines whether each port B pin is an input or an output. Writing a logic 1 to a DDRB bit enables the output buffer for the corresponding port B pin; a logic 0 disables the output buffer.

NOTE: *For those devices packaged in a 52-pin LQFP, PTB4–PTB7 are not connected. DDRB4–DDRB7 should be set to a 1 to configure PTB4–PTB7 as outputs.*

Address: $0005

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Reset:	0	0	0	0	0	0	0	0

Figure 17-6. Data Direction Register B (DDRB)

DDRB[7:0] — Data Direction Register B Bits

These read/write bits control port B data direction. Reset clears DDRB[7:0], configuring all port B pins as inputs.

1 = Corresponding port B pin configured as output

0 = Corresponding port B pin configured as input

NOTE: *Avoid glitches on port B pins by writing to the port B data register before changing data direction register B bits from 0 to 1. **Figure 17-7** shows the port B I/O logic.*

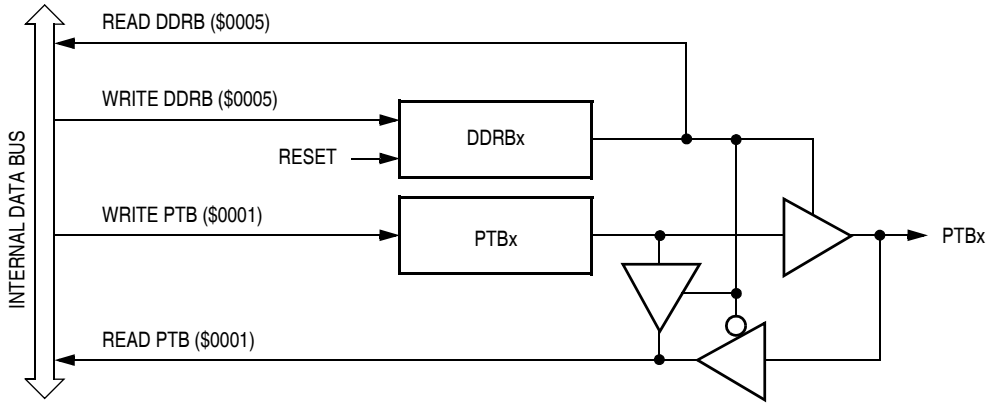


Figure 17-7. Port B I/O Circuit

When DDRBx is a logic 1, reading address $0001 reads the PTBx data latch. When DDRBx is a logic 0, reading address $0001 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit.

Table 17-3 summarizes the operation of the port B pins.

Table 17-3. Port B Pin Functions

DDRB Bit	PTB Bit	I/O Pin Mode	Accesses to DDRB	Accesses to PTB	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRB[7:0]	Pin	PTB[7:0](3)
1	X	Output	DDRB[7:0]	PTB[7:0]	PTB[7:0]

Notes:
1. X = don't care.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect input.

17.4.3 Port B LED Control Register (LEDB)

The port-B LED control register (LEDB) controls the direct LED drive capability on PTB5–PTB0 pins. Each bit is individually configurable and requires that the data direction register, DDRB, bit be configured as an output.

When the IRSCI is enabled, setting the LEDB0 bit also enables high current (15 mA) sink capability for the TxD pin.

Address: $000C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	LEDB5	LEDB4	LEDB3	LEDB2	LEDB1	LEDB0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 17-8. Port B LED Control Register (LEDB)

LEDB[5:0] — Port B LED Drive Enable Bits

These read/write bits are software programmable to enable the direct LED drive on an output port pin.

1 = Corresponding port B pin configured for direct LED drive:
15 mA current sinking capability on PTB[1:0], and
8 mA current sinking capability on PTB[5:2]

0 = Corresponding port B pin configured for standard drive

17.5 Port C

Port C is an 8-bit special function port that shares all of its port pins with the liquid crystal display (LCD) driver module.

17.5.1 Port C Data Register (PTC)

The port C data register contains a data latch for each of the eight port C pins.

Address: $0002

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTC7	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
Reset:	Unaffected by reset							
Alternative Function:	FP26	FP25	FP24	FP23	FP22	FP21	FP20	FP19

Figure 17-9. Port C Data Register (PTC)

PTC[7:0] — Port C Data Bits

These read/write bits are software programmable. Data direction of each port C pin is under the control of the corresponding bit in data direction register C. Reset has no effect on port C data.

FP[26:19] — LCD Driver Frontplanes 26–19

FP[26:19] are pins used for the frontplane output of the LCD driver module. The enable bits, PCEH and PCEL, in the CONFIG2 register, determine whether the PTC7/FP26–PTC4/FP23 and PTC3/FP22–PTC0/FP19 pins are LCD frontplane driver pins or general-purpose I/O pins. See **Section 16. Liquid Crystal Display Driver (LCD)**.

17.5.2 Data Direction Register C (DDRC)

Data direction register C determines whether each port C pin is an input or an output. Writing a logic 1 to a DDRC bit enables the output buffer for the corresponding port C pin; a logic 0 disables the output buffer.

Address: $0006

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
Reset:	0	0	0	0	0	0	0	0

Figure 17-10. Data Direction Register B (DDRB)

DDRC[7:0] — Data Direction Register C Bits

These read/write bits control port C data direction. Reset clears DDRC[7:0], configuring all port C pins as inputs.

1 = Corresponding port C pin configured as output
0 = Corresponding port C pin configured as input

NOTE: *Avoid glitches on port C pins by writing to the port C data register before changing data direction register C bits from 0 to 1.* ***Figure 17-11*** *shows the port C I/O logic.*

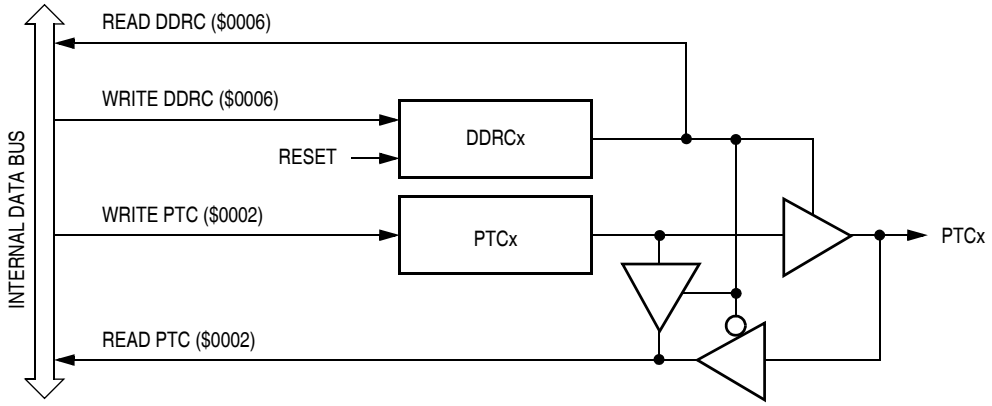


Figure 17-11. Port C I/O Circuit

When DDRCx is a logic 1, reading address $0002 reads the PTCx data latch. When DDRCx is a logic 0, reading address $0002 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit.

Table 17-4 summarizes the operation of the port C pins.

Table 17-4. Port C Pin Functions

DDRC Bit	PTC Bit	I/O Pin Mode	Accesses to DDRC	Accesses to PTC	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRC[7:0]	Pin	PTC[7:0](3)
1	X	Output	DDRC[7:0]	PTC[7:0]	PTC[7:0]

Notes:
1. X = don't care; except PTC2.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect input.

17.6 Port D

Port D is an 8-bit special function port that shares four of its pins with serial peripheral interface (SPI) module and four of its pins with the keyboard interrupt module (KBI).

NOTE: *Port D is not available in a 52-pin LQFP.*

17.6.1 Port D Data Register (PTD)

The port D data register contains a data latch for each of the eight port D pins.

NOTE: *Bit 0–bit 7 of PTD are not available in a 52-pin LQFP.*

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
Reset:	Unaffected by reset							
Alternative Function:	KBI7	KBI6	KBI5	KBI4	SPSCK	MOSI	MISO	$\overline{SS}$

Figure 17-12. Port D Data Register (PTD)

PTD[7:0] — Port D Data Bits

These read/write bits are software programmable. Data direction of each port D pin is under the control of the corresponding bit in data direction register D. Reset has no effect on port D data.

$\overline{SS}$, MISO, MOSI, and SPSCK — SPI functional pins

These are the chip select, master-input-slave-output, master-output-slave-input and clock pins for the SPI module. The SPI enable bit, SPE, in the SPI control register, SPCR, enables these pins as the SPI functional pins and overrides any control from port I/O logic. See **Section 14. Serial Peripheral Interface Module (SPI)**.

KBI[7:4] — Keyboard Interrupt Pins

KBI[7:4] are input pins to the keyboard interrupt module. The corresponding control bits, KBIE[7:4], in the keyboard interrupt enable register, KBIER, select which port pins will be used as a keyboard interrupt input and overrides any control from the port I/O logic. See **Section 19. Keyboard Interrupt Module (KBI)**

17.6.2 Data Direction Register D (DDRD)

Data direction register D determines whether each port D pin is an input or an output. Writing a logic 1 to a DDRD bit enables the output buffer for the corresponding port D pin; a logic 0 disables the output buffer.

NOTE: *For those devices packaged in a 52-pin LQFP, PTD0–PTD7 are not connected. DDRD0–DDRD7 should be set to a 1 to configure PTD0–PTD7 as outputs.*

Address: $0007

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
Reset:	0	0	0	0	0	0	0	0

Figure 17-13. Data Direction Register D (DDRD)

DDRD[7:0] — Data Direction Register D Bits

These read/write bits control port D data direction. Reset clears DDRD[7:0], configuring all port D pins as inputs.

1 = Corresponding port D pin configured as output
0 = Corresponding port D pin configured as input

NOTE: *Avoid glitches on port D pins by writing to the port D data register before changing data direction register D bits from 0 to 1. **Figure 17-14** shows the port D I/O logic.*

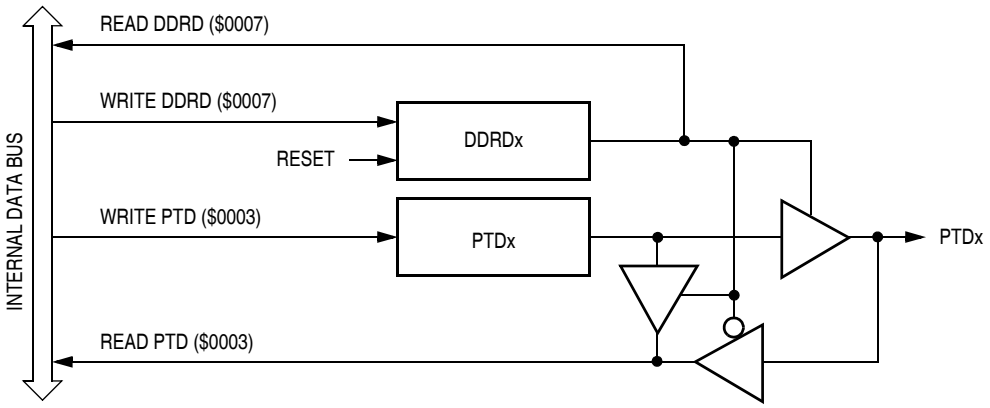


Figure 17-14. Port D I/O Circuit

When DDRDx is a logic 1, reading address $0003 reads the PTDx data latch. When DDRDx is a logic 0, reading address $0003 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit.

Table 17-5 summarizes the operation of the port D pins.

Table 17-5. Port D Pin Functions

DDRD Bit	PTD Bit	I/O Pin Mode	Accesses to DDRD	Accesses to PTD	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRD[7:0]	Pin	PTD[7:0](3)
1	X	Output	DDRD[7:0]	PTD[7:0]	PTD[7:0]

Notes:
1. X = don't care.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect the input.

Section 18. External Interrupt (IRQ)

18.1 Contents

18.2 Introduction 357
18.3 Features 357
18.4 Functional Description 358
18.4.1 $\overline{\text{IRQ}}$ Pin 360
18.5 IRQ Module During Break Interrupts 361
18.6 IRQ Status and Control Register (INTSCR) 361

18.2 Introduction

The external interrupt (IRQ) module provides a maskable interrupt input.

18.3 Features

Features of the IRQ module include the following:

- A dedicated external interrupt pin ($\overline{\text{IRQ}}$)
- IRQ interrupt control bits
- Hysteresis buffer
- Programmable edge-only or edge and level interrupt sensitivity
- Automatic interrupt acknowledge
- Internal pullup resistor

18.4 Functional Description

A logic 0 applied to the external interrupt pin can latch a CPU interrupt request. **Figure 18-1** shows the structure of the IRQ module.

Interrupt signals on the $\overline{\text{IRQ}}$ pin are latched into the IRQ latch. An interrupt latch remains set until one of the following actions occurs:

- Vector fetch — A vector fetch automatically generates an interrupt acknowledge signal that clears the IRQ latch.
- Software clear — Software can clear the interrupt latch by writing to the acknowledge bit in the interrupt status and control register (INTSCR). Writing a logic 1 to the ACK bit clears the IRQ latch.
- Reset — A reset automatically clears the interrupt latch.

The external interrupt pin is falling-edge-triggered and is software-configurable to be either falling-edge or low-level-triggered. The MODE bit in the INTSCR controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin.

When the interrupt pin is edge-triggered only, the CPU interrupt request remains set until a vector fetch, software clear, or reset occurs.

When the interrupt pin is both falling-edge and low-level-triggered, the CPU interrupt request remains set until both of the following occur:

- Vector fetch or software clear
- Return of the interrupt pin to logic 1

The vector fetch or software clear may occur before or after the interrupt pin returns to logic 1. As long as the pin is low, the interrupt request remains pending. A reset will clear the latch and the MODE control bit, thereby clearing the interrupt even if the pin stays low.

When set, the IMASK bit in the INTSCR mask all external interrupt requests. A latched interrupt request is not presented to the interrupt priority logic unless the IMASK bit is clear.

NOTE: *The interrupt mask (I) in the condition code register (CCR) masks all interrupt requests, including external interrupt requests.*

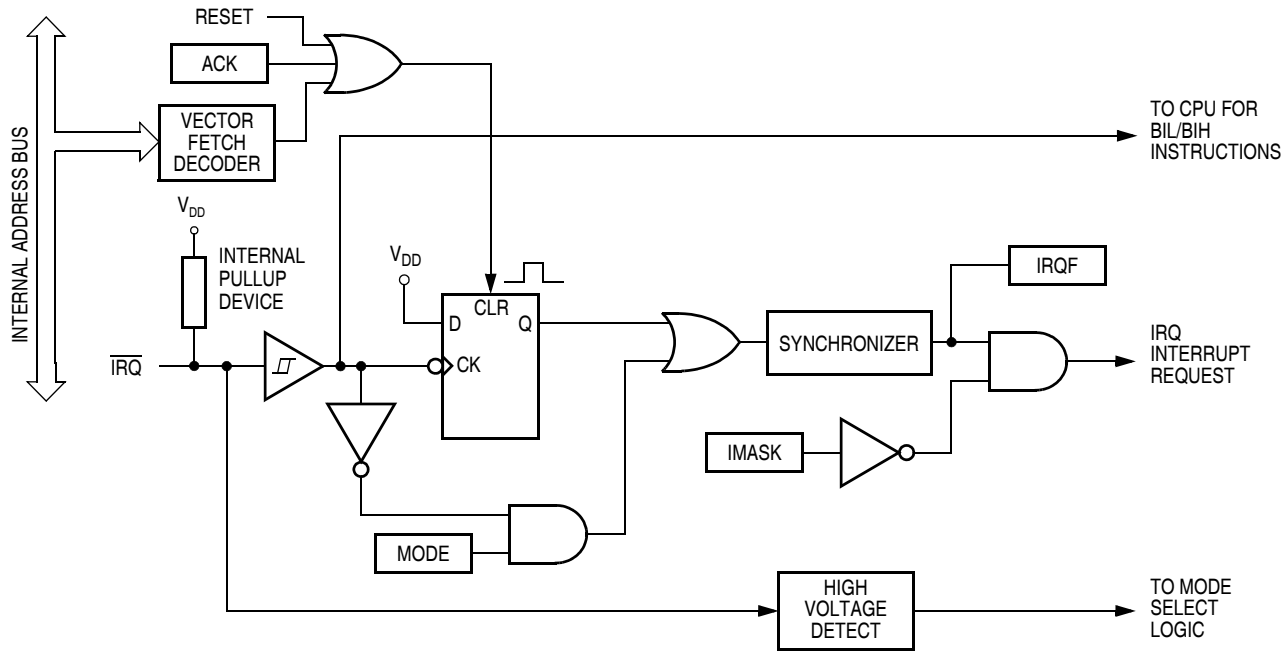


Figure 18-1. IRQ Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001E	IRQ Status and Control Register (INTSCR)	Read:	0	0	0	0	IRQF	0	IMASK	MODE
		Write:						ACK		
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Table 18-1. IRQ I/O Port Register Summary

18.4.1 $\overline{\text{IRQ}}$ Pin

A logic 0 on the $\overline{\text{IRQ}}$ pin can latch an interrupt request into the IRQ latch. A vector fetch, software clear, or reset clears the IRQ latch.

If the MODE bit is set, the $\overline{\text{IRQ}}$ pin is both falling-edge-sensitive and low-level-sensitive. With MODE set, both of the following actions must occur to clear IRQ:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the latch. Software may generate the interrupt acknowledge signal by writing a logic 1 to the ACK bit in the interrupt status and control register (INTSCR). The ACK bit is useful in applications that poll the $\overline{\text{IRQ}}$ pin and require software to clear the IRQ latch. Writing to the ACK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACK does not affect subsequent transitions on the $\overline{\text{IRQ}}$ pin. A falling edge that occurs after writing to the ACK bit latches another interrupt request. If the IRQ mask bit, IMASK, is clear, the CPU loads the program counter with the vector address at locations $FFFA and $FFFB.
- Return of the $\overline{\text{IRQ}}$ pin to logic 1 — As long as the $\overline{\text{IRQ}}$ pin is at logic 0, IRQ remains active.

The vector fetch or software clear and the return of the $\overline{\text{IRQ}}$ pin to logic 1 may occur in any order. The interrupt request remains pending as long as the $\overline{\text{IRQ}}$ pin is at logic 0. A reset will clear the latch and the MODE control bit, thereby clearing the interrupt even if the pin stays low.

If the MODE bit is clear, the $\overline{\text{IRQ}}$ pin is falling-edge-sensitive only. With MODE clear, a vector fetch or software clear immediately clears the IRQ latch.

The IRQF bit in the INTSCR register can be used to check for pending interrupts. The IRQF bit is not affected by the IMASK bit, which makes it useful in applications where polling is preferred.

Use the BIH of BIL instruction to read the logic level on the $\overline{\text{IRQ}}$ pin.

NOTE: *When using the level-sensitive interrupt trigger, avoid false interrupts by masking interrupt requests in the interrupt routine.*

18.5 IRQ Module During Break Interrupts

The system integration module (SIM) controls whether the IRQ latch can be cleared during the break state. The BCFE bit in the break flag control register (BFCR) enables software to clear the latches during the break state. (See **Section 22. Break Module (BRK)**.)

To allow software to clear the IRQ latch during a break interrupt, write a logic 1 to the BCFE bit. If a latch is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the latches during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), writing to the ACK bit in the IRQ status and control register during the break state has no effect on the IRQ latch.

18.6 IRQ Status and Control Register (INTSCR)

The IRQ Status and Control Register (INTSCR) controls and monitors operation of the IRQ module. The INTSCR has the following functions:

- Shows the state of the IRQ flag
- Clears the IRQ latch
- Masks IRQ and interrupt request
- Controls triggering sensitivity of the $\overline{\text{IRQ}}$ interrupt pin

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	IRQF	0	IMASK	MODE
Write:						ACK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 18-2. IRQ Status and Control Register (INTSCR)

IRQF — IRQ Flag Bit

This read-only status bit is high when the IRQ interrupt is pending.

1 = $\overline{\text{IRQ}}$ interrupt pending
0 = $\overline{\text{IRQ}}$ interrupt not pending

ACK — IRQ Interrupt Request Acknowledge Bit

Writing a logic 1 to this write-only bit clears the IRQ latch. ACK always reads as logic 0. Reset clears ACK.

IMASK — IRQ Interrupt Mask Bit

Writing a logic 1 to this read/write bit disables IRQ interrupt requests. Reset clears IMASK.

1 = IRQ interrupt requests disabled
0 = IRQ interrupt requests enabled

MODE — IRQ Edge/Level Select Bit

This read/write bit controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin. Reset clears MODE.

1 = $\overline{\text{IRQ}}$ interrupt requests on falling edges and low levels
0 = $\overline{\text{IRQ}}$ interrupt requests on falling edges only

Section 19. Keyboard Interrupt Module (KBI)

19.1 Contents

19.2 Introduction 363
19.3 Features 364
19.4 I/O Pins 364
19.5 Functional Description 365
19.5.1 Keyboard Initialization 367
19.6 Keyboard Interrupt Registers 367
19.6.1 Keyboard Status and Control Register 368
19.6.2 Keyboard Interrupt Enable Register 369
19.7 Low-Power Modes 369
19.8 Wait Mode 369
19.9 Stop Mode 370
19.10 Keyboard Module During Break Interrupts 370

19.2 Introduction

The keyboard interrupt module (KBI) provides eight independently maskable external interrupts which are accessible via PTA0–PTA3 and PTD4–PTD7. When a port pin is enabled for keyboard interrupt function, an internal 30kΩ pullup device is also enabled on the pin.

19.3 Features

Features of the keyboard interrupt module include the following:

- Eight keyboard interrupt pins with pullup devices
- Separate keyboard interrupt enable bits and one keyboard interrupt mask
- Programmable edge-only or edge- and level- interrupt sensitivity
- Exit from low-lower modes

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001B	Keyboard Status and Control Register (KBSCR)	Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
		Write:						ACKK		
		Reset:	0	0	0	0	0	0	0	0
$001C	Keyboard Interrupt Enable Register (KBIER)	Read:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 19-1. KBI I/O Register Summary

19.4 I/O Pins

The eight keyboard interrupt pins are shared with standard port I/O pins. The full name of the KBI pins are listed in **Table 19-1**. The generic pin name appear in the text that follows.

Table 19-1. Pin Name Conventions

KBI Generic Pin Name	Full MCU Pin Name	Pin Selected for KBI Function by KBIEx Bit in KBIER
KBI0–KBI3	PTA0/KBI0–PTA3/KBI3	KBIE0–KBIE3
KBI4–KBI7	PTD4/KBI4–PTD7/KBI7	KBIE4–KBIE7

19.5 Functional Description

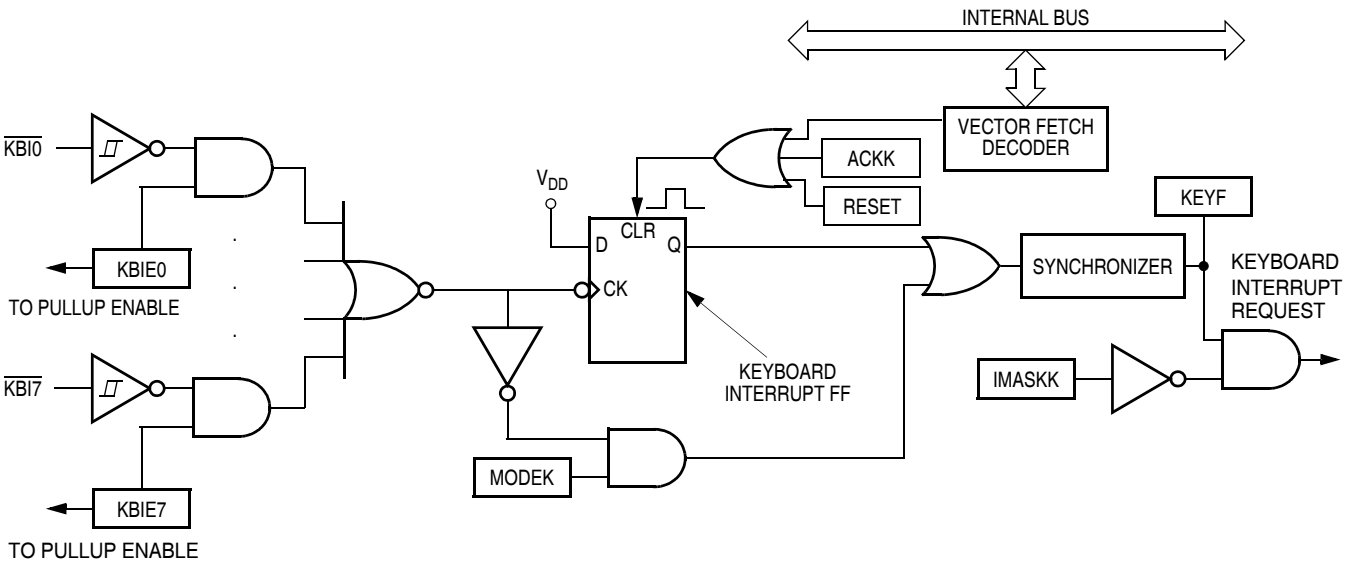


Figure 19-2. Keyboard Interrupt Block Diagram

Writing to the KBIE7–KBIE0 bits in the keyboard interrupt enable register independently enables or disables a port A or port D pin as a keyboard interrupt pin. Enabling a keyboard interrupt pin in port A or port D also enables its internal pull-up device. A logic 0 applied to an enabled keyboard interrupt pin latches a keyboard interrupt request.

A keyboard interrupt is latched when one or more keyboard pins goes low after all were high. The MODEK bit in the keyboard status and control register controls the triggering mode of the keyboard interrupt.

- If the keyboard interrupt is edge-sensitive only, a falling edge on a keyboard pin does not latch an interrupt request if another keyboard pin is already low. To prevent losing an interrupt request on one pin because another pin is still low, software can disable the latter pin while it is low.
- If the keyboard interrupt is falling edge- and low level-sensitive, an interrupt request is present as long as any keyboard pin is low.

If the MODEK bit is set, the keyboard interrupt pins are both falling edge- and low level-sensitive, and both of the following actions must occur to clear a keyboard interrupt request:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the interrupt request. Software may generate the interrupt acknowledge signal by writing a logic 1 to the ACKK bit in the keyboard status and control register KBSCR. The ACKK bit is useful in applications that poll the keyboard interrupt pins and require software to clear the keyboard interrupt request. Writing to the ACKK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACKK does not affect subsequent transitions on the keyboard interrupt pins. A falling edge that occurs after writing to the ACKK bit latches another interrupt request. If the keyboard interrupt mask bit, IMASKK, is clear, the CPU loads the program counter with the vector address at locations $FFDF and $FFDE.
- Return of all enabled keyboard interrupt pins to logic 1 — As long as any enabled keyboard interrupt pin is at logic 0, the keyboard interrupt remains set.

The vector fetch or software clear and the return of all enabled keyboard interrupt pins to logic 1 may occur in any order.

If the MODEK bit is clear, the keyboard interrupt pin is falling-edge-sensitive only. With MODEK clear, a vector fetch or software clear immediately clears the keyboard interrupt request.

Reset clears the keyboard interrupt request and the MODEK bit, clearing the interrupt request even if a keyboard interrupt pin stays at logic 0.

The keyboard flag bit (KEYF) in the keyboard status and control register can be used to see if a pending interrupt exists. The KEYF bit is not affected by the keyboard interrupt mask bit (IMASKK) which makes it useful in applications where polling is preferred.

To determine the logic level on a keyboard interrupt pin, use the data direction register to configure the pin as an input and read the data register.

NOTE: *Setting a keyboard interrupt enable bit (KBIEx) forces the corresponding keyboard interrupt pin to be an input, overriding the data direction register. However, the data direction register bit must be a logic 0 for software to read the pin.*

19.5.1 Keyboard Initialization

When a keyboard interrupt pin is enabled, it takes time for the internal pull-up to reach a logic 1. Therefore a false interrupt can occur as soon as the pin is enabled.

To prevent a false interrupt on keyboard initialization:

1. Mask keyboard interrupts by setting the IMASKK bit in the keyboard status and control register.
2. Enable the KBI pins by setting the appropriate KBIEx bits in the keyboard interrupt enable register.
3. Write to the ACKK bit in the keyboard status and control register to clear any false interrupts.
4. Clear the IMASKK bit.

An interrupt signal on an edge-triggered pin can be acknowledged immediately after enabling the pin. An interrupt signal on an edge- and level-triggered interrupt pin must be acknowledged after a delay that depends on the external load.

Another way to avoid a false interrupt:

1. Configure the keyboard pins as outputs by setting the appropriate DDR bits in data direction register.
2. Write logic 1s to the appropriate data register bits.
3. Enable the KBI pins by setting the appropriate KBIEx bits in the keyboard interrupt enable register.

19.6 Keyboard Interrupt Registers

Two registers control the operation of the keyboard interrupt module:

- Keyboard status and control register
- Keyboard interrupt enable register

19.6.1 Keyboard Status and Control Register

- Flags keyboard interrupt requests
- Acknowledges keyboard interrupt requests
- Masks keyboard interrupt requests
- Controls keyboard interrupt triggering sensitivity

Address: $001B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
Write:						ACKK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 19-3. Keyboard Status and Control Register (KBSCR)

KEYF — Keyboard Flag Bit

This read-only bit is set when a keyboard interrupt is pending. Reset clears the KEYF bit.

1 = Keyboard interrupt pending
0 = No keyboard interrupt pending

ACKK — Keyboard Acknowledge Bit

Writing a logic 1 to this write-only bit clears the keyboard interrupt request. ACKK always reads as logic 0. Reset clears ACKK.

IMASKK — Keyboard Interrupt Mask Bit

Writing a logic 1 to this read/write bit prevents the output of the keyboard interrupt mask from generating interrupt requests. Reset clears the IMASKK bit.

1 = Keyboard interrupt requests masked
0 = Keyboard interrupt requests not masked

MODEK — Keyboard Triggering Sensitivity Bit

This read/write bit controls the triggering sensitivity of the keyboard interrupt pins. Reset clears MODEK.

1 = Keyboard interrupt requests on falling edges and low levels
0 = Keyboard interrupt requests on falling edges only

19.6.2 Keyboard Interrupt Enable Register

The keyboard interrupt enable register individually enables or disables the PTA0/KBI0–PTA3/KBI3 and PTD4/KBI4–PTD7/KBI7 pins to operate as a keyboard interrupt pin.

Address: $001C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
Reset:	0	0	0	0	0	0	0	0

Figure 19-4. Keyboard Interrupt Enable Register (KBIER)

KBIE7–KBIE0 — Keyboard Interrupt Enable Bits

Each of these read/write bits enables the corresponding keyboard interrupt pin to latch interrupt requests. Reset clears the keyboard interrupt enable register.

1 = KBIx pin enabled as keyboard interrupt pin
0 = KBIx pin not enabled as keyboard interrupt pin

19.7 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

19.8 Wait Mode

The keyboard interrupt module remains active in wait mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of wait mode.

19.9 Stop Mode

The keyboard interrupt module remains active in stop mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of stop mode.

19.10 Keyboard Module During Break Interrupts

The system integration module (SIM) controls whether the keyboard interrupt latch can be cleared during the break state. The BCFE bit in the SIM break flag control register (BFCR) enables software to clear status bits during the break state.

To allow software to clear the keyboard interrupt latch during a break interrupt, write a logic 1 to the BCFE bit. If a latch is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the latch during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), writing to the keyboard acknowledge bit (ACKK) in the keyboard status and control register during the break state has no effect.

Section 20. Computer Operating Properly (COP)

20.1 Contents

20.2 Introduction 371
20.3 Functional Description 372
20.4 I/O Signals 373
20.4.1 ICLK 373
20.4.2 STOP Instruction 373
20.4.3 COPCTL Write 373
20.4.4 Power-On Reset 373
20.4.5 Internal Reset 374
20.4.6 Reset Vector Fetch 374
20.4.7 COPD (COP Disable) 374
20.4.8 COPRS (COP Rate Select) 374
20.5 COP Control Register 375
20.6 Interrupts 375
20.7 Monitor Mode 375
20.8 Low-Power Modes 375
20.8.1 Wait Mode 376
20.8.2 Stop Mode 376
20.9 COP Module During Break Mode 376

20.2 Introduction

The computer operating properly (COP) module contains a free-running counter that generates a reset if allowed to overflow. The COP module helps software recover from runaway code. Prevent a COP reset by clearing the COP counter periodically. The COP module can be disabled through the COPD bit in the configuration register 1 (CONFIG1).

20.3 Functional Description

Figure 20-1 shows the structure of the COP module.

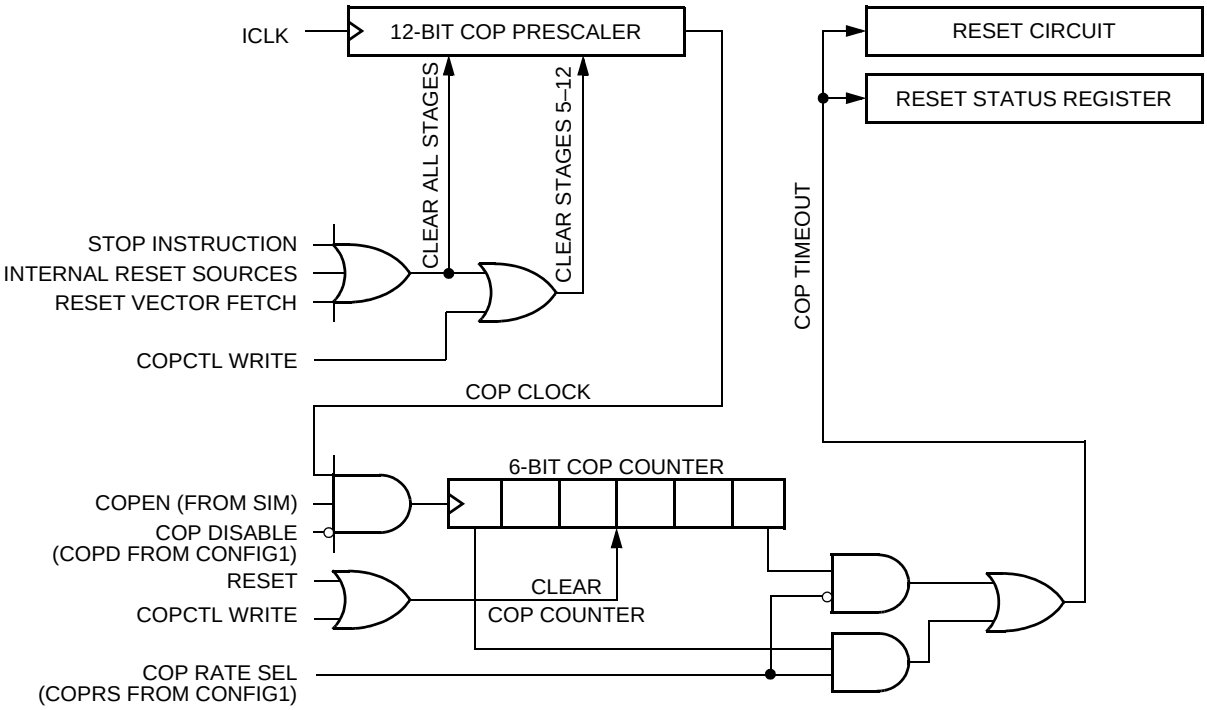


Figure 20-1. COP Block Diagram

The COP counter is a free-running 6-bit counter preceded by a 12-bit prescaler counter. If not cleared by software, the COP counter overflows and generates an asynchronous reset after $2^{18} - 2^4$ or $2^{13} - 2^4$ ICLK cycles, depending on the state of the COP rate select bit, COPRS, in the CONFIG1 register. With a $2^{13} - 2^4$ ICLK cycle overflow option, a 47-kHz ICLK gives a COP timeout period of 174 ms. Writing any value to location $FFFF before an overflow occurs prevents a COP reset by clearing the COP counter and stages 12 through 5 of the prescaler.

NOTE: *Service the COP immediately after reset and before entering or after exiting STOP Mode to guarantee the maximum time before the first COP counter overflow.*

A COP reset pulls the $\overline{\text{RST}}$ pin low for 32 ICLK cycles and sets the COP bit in the SIM reset status register (SRSR).

In monitor mode, the COP is disabled if the $\overline{\text{RST}}$ pin or the $\overline{\text{IRQ1}}$ is held at V_{TST}. During the break state, V_{TST} on the $\overline{\text{RST}}$ pin disables the COP.

NOTE: *Place COP clearing instructions in the main program and not in an interrupt subroutine. Such an interrupt subroutine could keep the COP from generating a reset even while the main program is not working properly.*

20.4 I/O Signals

The following paragraphs describe the signals shown in **Figure 20-1**.

20.4.1 ICLK

ICLK is the internal oscillator output signal. ICLK frequency is approximately equal to 47-kHz. See **Section 23. Electrical Specifications** for ICLK parameters.

20.4.2 STOP Instruction

The STOP instruction clears the COP prescaler.

20.4.3 COPCTL Write

Writing any value to the COP control register (COPCTL) (see **20.5 COP Control Register**) clears the COP counter and clears bits 12 through 5 of the prescaler. Reading the COP control register returns the low byte of the reset vector.

20.4.4 Power-On Reset

The power-on reset (POR) circuit clears the COP prescaler 4096 ICLK cycles after power-up.

20.4.5 Internal Reset

An internal reset clears the COP prescaler and the COP counter.

20.4.6 Reset Vector Fetch

A reset vector fetch occurs when the vector address appears on the data bus. A reset vector fetch clears the COP prescaler.

20.4.7 COPD (COP Disable)

The COPD signal reflects the state of the COP disable bit (COPD) in the CONFIG1 register. (See **Figure 20-2** and **Section 5. Configuration Registers (CONFIG)**.)

20.4.8 COPRS (COP Rate Select)

The COPRS signal reflects the state of the COP rate select bit (COPRS) in the CONFIG1 register.

Address: $001F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	0	SSREC	STOP	COPD
Write:								
Reset:	0	0	0	1	0	0	0	0

= Unimplemented

Figure 20-2. Configuration Register 1 (CONFIG1)

COPRS — COP Rate Select

COPRS selects the COP time-out period. Reset clears COPRS.

1 = COP time out period = $2^{13} – 2^4$ ICLK cycles
0 = COP time out period = $2^{18} – 2^4$ ICLK cycles

COPD — COP Disable Bit

COPD disables the COP module.

1 = COP module disabled
0 = COP module enabled

20.5 COP Control Register

The COP control register is located at address $FFFF and overlaps the reset vector. Writing any value to $FFFF clears the COP counter and starts a new timeout period. Reading location $FFFF returns the low byte of the reset vector.

Address: $FFFF

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Low byte of reset vector							
Write:	Clear COP counter							
Reset:	Unaffected by reset							

Figure 20-3. COP Control Register (COPCTL)

20.6 Interrupts

The COP does not generate CPU interrupt requests.

20.7 Monitor Mode

When monitor mode is entered with V_{TST} on the $\overline{IRQ}$ pin, the COP is disabled as long as V_{TST} remains on the $\overline{IRQ}$ pin or the $\overline{RST}$ pin. When monitor mode is entered by having blank reset vectors and not having V_{TST} on the $\overline{IRQ}$ pin, the COP is automatically disabled until a POR occurs.

20.8 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

20.8.1 Wait Mode

The COP remains active during wait mode. To prevent a COP reset during wait mode, periodically clear the COP counter in a CPU interrupt routine.

20.8.2 Stop Mode

Stop mode turns off the ICLK input to the COP and clears the COP prescaler. Service the COP immediately before entering or after exiting stop mode to ensure a full COP timeout period after entering or exiting stop mode.

To prevent inadvertently turning off the COP with a STOP instruction, a configuration option is available that disables the STOP instruction. When the STOP bit in the configuration register has the STOP instruction is disabled, execution of a STOP instruction results in an illegal opcode reset.

20.9 COP Module During Break Mode

The COP is disabled during a break interrupt when V_{TST} is present on the $\overline{RST}$ pin.

Section 21. Low-Voltage Inhibit (LVI)

21.1 Contents

21.2 Introduction 377
21.3 Features 377
21.4 Functional Description 378
21.4.1 Interrupt LVI Operation 380
21.4.2 Forced Reset Operation 380
21.4.3 Voltage Hysteresis Protection 380
21.4.4 LVI Trip Selection 381
21.5 LVI Status Register 381
21.6 Low-Power Modes 382
21.6.1 Wait Mode 382
21.6.2 Stop Mode 382

21.2 Introduction

This section describes the low-voltage inhibit (LVI) module, which monitors the voltage on the V_{DD} pin and can force a reset when the V_{DD} voltage falls below the LVI trip falling voltage, V_{TRIPF}.

21.3 Features

Features of the LVI module include:

- Programmable LVI interrupt and reset
- Selectable LVI trip voltage
- Programmable stop mode operation

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE0F	Low-Voltage Inhibit Status Register (LVISR)	Read:	LVIOUT	LVIIE	LVIIF	0	0	0	0	0
		Write:				LVIIAK				
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-1. LVI I/O Register Summary

21.4 Functional Description

Figure 21-2 shows the structure of the LVI module.

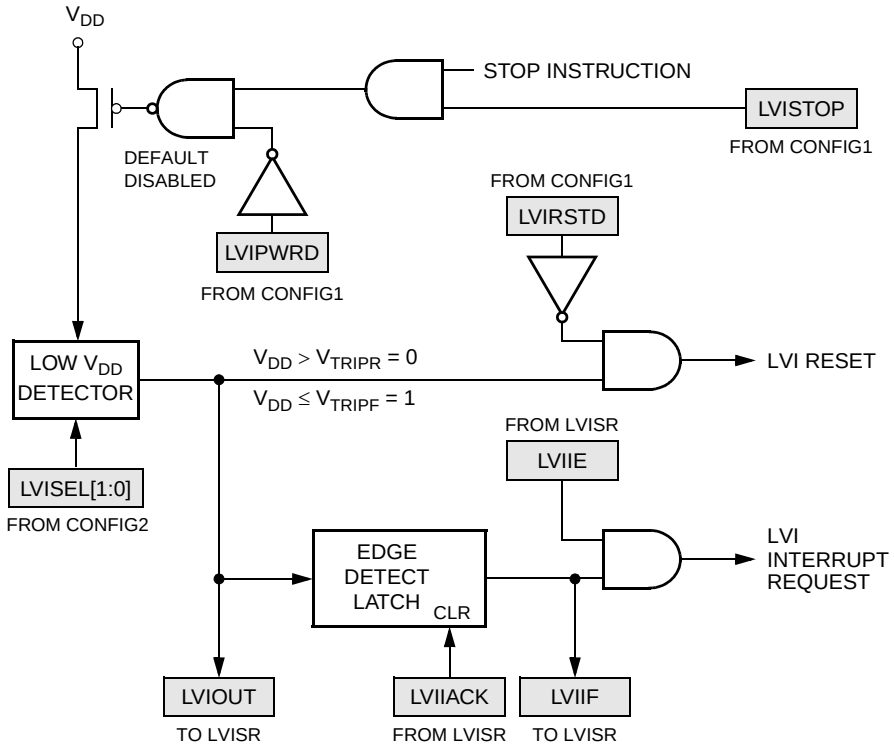


Figure 21-2. LVI Module Block Diagram

The LVI is disabled out of reset. The LVI module contains a bandgap reference circuit and comparator. Clearing the LVI power disable bit, LVIPWRD, enables the LVI to monitor V_{DD} voltage. Clearing the LVI reset disable bit, LVIRSTD, enables the LVI module to generate a reset when V_{DD} falls below a voltage, V_{TRIPF}. Setting the LVI enable in stop mode bit, LVISTOP, enables the LVI to operate in stop mode.

The LVI trip point selection bits, LVISEL[1:0], select the trip point voltage, V_{TRIPF}, to be configured for 5V or 3.3V operation. The actual trip points are shown in **Section 23. Electrical Specifications**.

Setting LVI interrupt enable bit, LVIIE, enables LVI interrupts whenever the LVIOUT bit toggles (from logic 0 to logic 1, or from logic 1 to logic 0).

NOTE: *After a power-on reset (POR) the user must configure the LVISEL[1:0} bits for 3.3V or 5V operation before enabling the LVI module (by clearing the LVIPWRD bit in CONFIG1 register).*

NOTE: *If the user requires 3.3V mode and enables the LVI module after configuring the LVISEL[1;0] bits to 3.3V operation mode while the V_{DD} supply is not above the V_{TRIPF} for 3.3V mode, the MCU will immediately go into reset. The LVI in this case will hold the MCU in reset until either V_{DD} goes above the rising 3.3V trip point, V_{TRIPR}, which will release reset or V_{DD} decreases to approximately 0V which will re-trigger the power-on reset.*

LVISTOP, LVIPWRD, LVIRSTD, and LVISEL[1:0] are in the configuration registers. See **Section 5. Configuration Registers (CONFIG)** for details of the LVI's configuration bits. Once an LVI reset occurs, the MCU remains in reset until V_{DD} rises above a voltage, V_{TRIPR}, which causes the MCU to exit reset. See **9.4.2.5 Low-Voltage Inhibit (LVI) Reset** for details of the interaction between the SIM and the LVI. The output of the comparator controls the state of the LVIOUT flag in the LVI status register (LVISR). The LVIIE, LVIIF, and LVIIAK bits in the LVISR control LVI interrupt functions.

An LVI reset also drives the $\overline{\text{RST}}$ pin low to provide low-voltage protection to external peripheral devices.

21.4.1 Interrupt LVI Operation

In applications that can operate at V_{DD} levels below the V_{TRIPF} level, software can monitor V_{DD} by polling the LVIOUT bit, or by setting the LVI interrupt enable bit, LVIIE, to enable interrupt requests. In the configuration register 1 (CONFIG1), the LVIPWRD bit must be at logic 0 to enable the LVI module, and the LVIRSTD bit must be at logic 1 to disable LVI resets.

The LVI interrupt flag, LVIIF, is set whenever the LVIOUT bit changes state (toggles). When LVIF is set, a CPU interrupt request is generated if the LVIIE is also set. In the LVI interrupt service subroutine, LVIIF bit can be cleared by writing a logic 1 to the LVI interrupt acknowledge bit, LVIIAK.

21.4.2 Forced Reset Operation

In applications that require V_{DD} to remain above the V_{TRIPF} level, enabling LVI resets allows the LVI module to reset the MCU when V_{DD} falls below the V_{TRIPF} level. In the configuration register 1 (CONFIG1), the LVIPWRD and LVIRSTD bits must be at logic 0 to enable the LVI module and to enable LVI resets.

If LVIIE is set to enable LVI interrupts when LVIRSTD is cleared, LVI reset has a higher priority over LVI interrupt. In this case, when V_{DD} falls below the V_{TRIPF} level, an LVI reset will occur, and the LVIIE bit will be cleared.

21.4.3 Voltage Hysteresis Protection

Once the LVI has triggered (by having V_{DD} fall below V_{TRIPF}), the LVI will maintain a reset condition until V_{DD} rises above the rising trip point voltage, V_{TRIPR}. This prevents a condition in which the MCU is continually entering and exiting reset if V_{DD} is approximately equal to V_{TRIPF}. V_{TRIPR} is greater than V_{TRIPF} by the hysteresis voltage, V_{HYS}.

21.4.4 LVI Trip Selection

The trip point selection bits, LVISEL[1:0], in the CONFIG2 register select whether the LVI is configured for 5 V or 3.3 V operation. (See **Section 5. Configuration Registers (CONFIG)**.)

NOTE: *The MCU is guaranteed to operate at a minimum supply voltage. The trip point (V_{TRIPF} [5 V] or V_{TRIPF} [3.3 V]) may be lower than this. (See **Section 23. Electrical Specifications** for the actual trip point voltages.)*

21.5 LVI Status Register

The LVI status register (LVISR) controls LVI interrupt functions and indicates if the V_{DD} voltage was detected below the V_{TRIPF} level.

Address: $FE0F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LVIOUT	LVIIE	LVIIF	0	0	0	0	0
Write:				LVIIAK				
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Table 21-1. LVI Status Register (LVISR)

LVIOUT — LVI Output Bit

This read-only flag becomes set when the V_{DD} voltage falls below the V_{TRIPF} trip voltage (see **Table 21-2**). Reset clears the LVIOUT bit.

Table 21-2. LVIOUT Bit Indication

V_{DD}	LVIOUT
$V_{DD} > V_{TRIPR}$	0
$V_{DD} < V_{TRIPF}$	1
$V_{TRIPF} < V_{DD} < V_{TRIPR}$	Previous value

LVIIE — LVI Interrupt Enable Bit

This read/write bit enables the LVIIF bit to generate CPU interrupt requests. Reset clears the LVIIE bit.

1 = LVIIF can generate CPU interrupt requests
0 = LVIIF cannot generate CPU interrupt requests

LVIIF — LVI Interrupt Flag

This clearable, read-only flag is set whenever the LVIOUT bit toggles. Reset clears the LVIIF bit.

1 = LVIOUT has toggled
0 = LVIOUT has not toggled

LVIIAK — LVI Interrupt Acknowledge Bit

Writing a logic 1 to this write-only bit clears the LVI interrupt flag, LVIIF. LVIIAK always reads as logic 0.

1 = Clears LVIIF bit
0 = No effect

21.6 Low-Power Modes

The STOP and WAIT instructions put the MCU in low power-consumption standby modes.

21.6.1 Wait Mode

If enabled, the LVI module remains active in wait mode. If enabled to generate resets or interrupts, the LVI module can generate a reset or an interrupt and bring the MCU out of wait mode.

21.6.2 Stop Mode

If enabled in stop mode (LVISTOP = 1), the LVI module remains active in stop mode. If enabled to generate resets or interrupts, the LVI module can generate a reset or an interrupt and bring the MCU out of stop mode.

NOTE: *If enabled to generate both resets and interrupts, there will be no LVI interrupts, as resets have a higher priority.*

Section 22. Break Module (BRK)

22.1 Contents

22.2 Introduction 383
22.3 Features 384
22.4 Functional Description 384
22.4.1 Flag Protection During Break Interrupts 386
22.4.2 CPU During Break Interrupts 386
22.4.3 TIM1 and TIM2 During Break Interrupts 386
22.4.4 COP During Break Interrupts 386
22.5 Low-Power Modes 386
22.5.1 Wait Mode 386
22.5.2 Stop Mode 387
22.6 Break Module Registers 387
22.6.1 Break Status and Control Register 387
22.6.2 Break Address Registers 388
22.6.3 SIM Break Status Register 388
22.6.4 SIM Break Flag Control Register 390

22.2 Introduction

This section describes the break module. The break module can generate a break interrupt that stops normal program flow at a defined address to enter a background program.

22.3 Features

Features of the break module include:

- Accessible input/output (I/O) registers during the break interrupt
- CPU-generated break interrupts
- Software-generated break interrupts
- COP disabling during break interrupts

22.4 Functional Description

When the internal address bus matches the value written in the break address registers, the break module issues a breakpoint signal to the CPU. The CPU then loads the instruction register with a software interrupt instruction (SWI) after completion of the current CPU instruction. The program counter vectors to $FFFC and $FFFD ($FEFC and $FEFD in monitor mode).

The following events can cause a break interrupt to occur:

- A CPU-generated address (the address in the program counter) matches the contents of the break address registers.
- Software writes a logic 1 to the BRKA bit in the break status and control register.

When a CPU-generated address matches the contents of the break address registers, the break interrupt begins after the CPU completes its current instruction. A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the MCU to normal operation. **Figure 22-1** shows the structure of the break module.

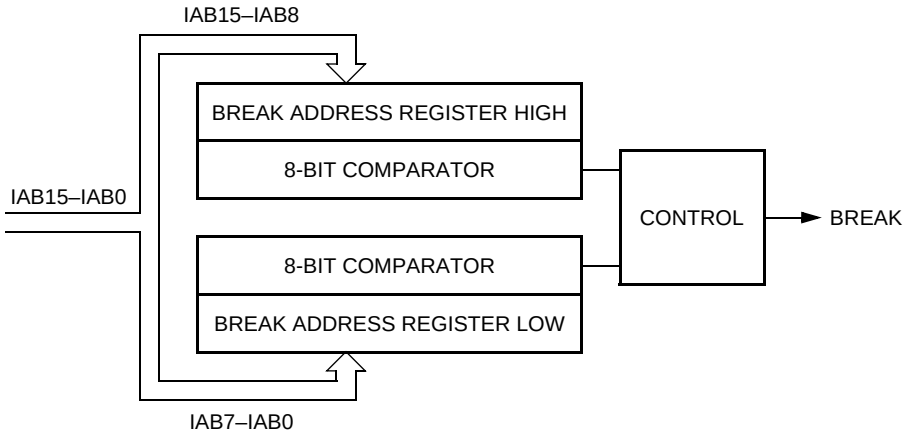


Figure 22-1. Break Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	SIM Break Status Register (SBSR)	Read:	R	R	R	R	R	R	SBSW	R
		Write:							Note	
		Reset:							0	
$FE03	SIM Break Flag Control Register (SBFCR)	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
$FE0C	Break Address Register High (BRKH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0D	Break Address Register Low (BRKL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0E	Break Status and Control Register (BRKSCR)	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

Note: Writing a logic 0 clears SBSW.

= Unimplemented R = Reserved

Figure 22-2. Break Module I/O Register Summary

22.4.1 Flag Protection During Break Interrupts

The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state.

22.4.2 CPU During Break Interrupts

The CPU starts a break interrupt by:

- Loading the instruction register with the SWI instruction
- Loading the program counter with $FFFC and $FFFD ($FEFC and $FEFD in monitor mode)

The break interrupt begins after completion of the CPU instruction in progress. If the break address register match occurs on the last cycle of a CPU instruction, the break interrupt begins immediately.

22.4.3 TIM1 and TIM2 During Break Interrupts

A break interrupt stops the timer counters.

22.4.4 COP During Break Interrupts

The COP is disabled during a break interrupt when V_{TST} is present on the $\overline{RST}$ pin.

22.5 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

22.5.1 Wait Mode

If enabled, the break module is active in wait mode. In the break routine, the user can subtract one from the return address on the stack if SBSW is set (see **Section 9. System Integration Module (SIM)**). Clear the SBSW bit by writing logic 0 to it.

22.5.2 Stop Mode

A break interrupt causes exit from stop mode and sets the SBSW bit in the break status register.

22.6 Break Module Registers

These registers control and monitor operation of the break module:

- Break status and control register (BRKSCR)
- Break address register high (BRKH)
- Break address register low (BRKL)
- SIM break status register (SBSR)
- SIM break flag control register (SBFCR)

22.6.1 Break Status and Control Register

The break status and control register (BRKSCR) contains break module enable and status bits.

Address: $FE0E

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

= Unimplemented

Figure 22-3. Break Status and Control Register (BRKSCR)

BRKE — Break Enable Bit

This read/write bit enables breaks on break address register matches. Clear BRKE by writing a logic 0 to bit 7. Reset clears the BRKE bit.

1 = Breaks enabled on 16-bit address match
0 = Breaks disabled on 16-bit address match

BRKA — Break Active Bit

This read/write status and control bit is set when a break address match occurs. Writing a logic 1 to BRKA generates a break interrupt. Clear BRKA by writing a logic 0 to it before exiting the break routine. Reset clears the BRKA bit.

1 = (When read) Break address match

0 = (When read) No break address match

22.6.2 Break Address Registers

The break address registers (BRKH and BRKL) contain the high and low bytes of the desired breakpoint address. Reset clears the break address registers.

Address: $FE0C

	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 22-4. Break Address Register High (BRKH)

Address: $FE0D

	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 22-5. Break Address Register Low (BRKL)

22.6.3 SIM Break Status Register

The SIM break status register (SBSR) contains a flag to indicate that a break caused an exit from wait mode. The flag is useful in applications requiring a return to wait mode after exiting from a break interrupt.

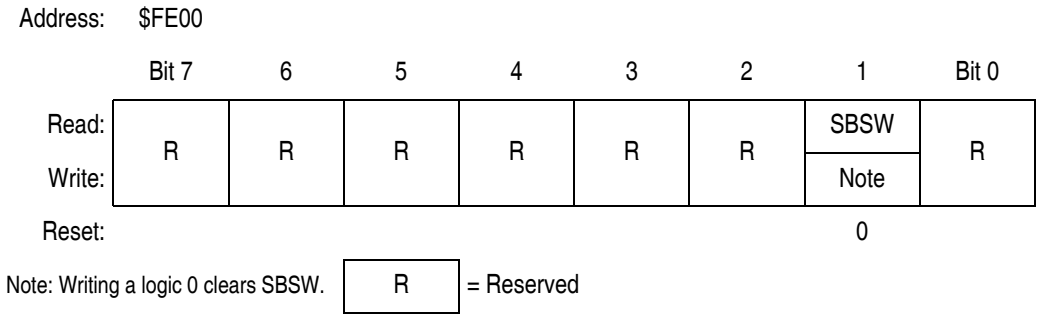


Figure 22-6. SIM Break Status Register (SBSR)

SBSW — Break Wait Bit

This status bit is set when a break interrupt causes an exit from wait mode or stop mode. Clear SBSW by writing a logic 0 to it. Reset clears SBSW.

1 = Stop mode or wait mode was exited by break interrupt
0 = Stop mode or wait mode was not exited by break interrupt

SBSW can be read within the break interrupt routine. The user can modify the return address on the stack by subtracting 1 from it. The following code is an example.

```
;This code works if the H register has been pushed onto the stack in the break
;service routine software. This code should be executed at the end of the break
;service routine software.

 HIBYTE   EQU     5

 LOBYTE   EQU     6

;         If not SBSW, do RTI

          BRCLR   SBSW,SBSR, RETURN    ;See if wait mode or stop mode was exited by
                                       ;break.

          TST     LOBYTE,SP            ;If RETURNLO is not zero,

          BNE     DOLO                 ;then just decrement low byte.

          DEC     HIBYTE,SP            ;Else deal with high byte, too.

 DOLO     DEC     LOBYTE,SP            ;Point to WAIT/STOP opcode.

 RETURN   PULH                         ;Restore H register.
          RTI
```

22.6.4 SIM Break Flag Control Register

The SIM break flag control register (SBFCR) contains a bit that enables software to clear status bits while the MCU is in a break state.

Address: $FE03

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

R = Reserved

Figure 22-7. SIM Break Flag Control Register (SBFCR)

BCFE — Break Clear Flag Enable Bit

This read/write bit enables software to clear status bits by accessing status registers while the MCU is in a break state. To clear status bits during the break state, the BCFE bit must be set.

1 = Status bits clearable during break

0 = Status bits not clearable during break

Section 23. Electrical Specifications

23.1 Contents

23.2 Introduction 392
23.3 Absolute Maximum Ratings 392
23.4 Functional Operating Range 393
23.5 Thermal Characteristics 393
23.6 5.0V DC Electrical Characteristics 394
23.7 3.3V DC Electrical Characteristics 396
23.8 5.0V Control Timing 397
23.9 3.3V Control Timing 397
23.10 5.0V Oscillator Characteristics 398
23.11 3.3V Oscillator Characteristics 398
23.12 5.0V ADC Electrical Characteristics 399
23.13 3.3V ADC Electrical Characteristics 400
23.14 Timer Interface Module Characteristics 401
23.15 CGM Electrical Specifications 401
23.16 5.0V SPI Characteristics 402
23.17 3.3V SPI Characteristics 403
23.18 FLASH Memory Characteristics 406

23.2 Introduction

This section contains electrical and timing specifications.

23.3 Absolute Maximum Ratings

Maximum ratings are the extreme limits to which the MCU can be exposed without permanently damaging it.

NOTE: *This device is not guaranteed to operate properly at the maximum ratings. Refer to* ***23.6 5.0V DC Electrical Characteristics*** *for guaranteed operating conditions.*

Table 23-1. Absolute Maximum Ratings(1)

Characteristic	Symbol	Value	Unit
Supply voltage	V_{DD}	–0.3 to +6.0	V
Input voltage All pins (except $\overline{IRQ}$) $\overline{IRQ}$ pin	V_{IN}	V_{SS}–0.3 to V_{DD} +0.3 V_{SS}–0.3 to 8.5	V
Maximum current per pin excluding V_{DD} and V_{SS}	I	±25	mA
Maximum current out of V_{SS}	I_{MVSS}	100	mA
Maximum current into V_{DD}	I_{MVDD}	100	mA
Storage temperature	T_{STG}	–55 to +150	°C

Notes:
1. Voltages referenced to V_{SS}.

NOTE: *This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{IN} and V_{OUT} be constrained to the range $V_{SS} \le (V_{IN}$ or $V_{OUT}) \le V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (for example, either V_{SS} or V_{DD}.)*

23.4 Functional Operating Range

Table 23-2. Operating Range

Characteristic	Symbol	Value	Unit
Operating temperature range	T_A	–40 to +85	°C
Operating voltage range	V_{DD}	3.3V ± 10% 5.0V ± 10%	V

23.5 Thermal Characteristics

Table 23-3. Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance 52-pin LQFP 64-pin LQFP 64-pin QFP	θ_{JA}	 85 80 70	°C/W
I/O pin power dissipation	$P_{I/O}$	User determined	W
Power dissipation(1)	P_D	$P_D = (I_{DD} \times V_{DD}) + P_{I/O} = K/(T_J + 273\ °C)$	W
Constant(2)	K	$P_D \times (T_A + 273\ °C) + P_D^2 \times \theta_{JA}$	W/°C
Average junction temperature	T_J	$T_A + (P_D \times \theta_{JA})$	°C

Notes:
1. Power dissipation is a function of temperature.
2. K constant unique to the device. K can be determined for a known T_A and measured P_D. With this value of K, P_D and T_J can be determined for any value of T_A.

23.6 5.0V DC Electrical Characteristics

Table 23-4. 5.0V DC Electrical Characteristics

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Output high voltage (I_{LOAD} = –2.0 mA) All ports	V_{OH}	V_{DD}–0.8	—	—	V
Output low voltage (I_{LOAD} = 1.6mA) All ports (I_{LOAD} = 8.0 mA) PTB2–PTB5 (I_{LOAD} = 15.0 mA) PTB0/TxD–PTB1	V_{OL}	—	—	0.4	V
Input high voltage All ports, $\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input low voltage All ports, $\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IL}	V_{SS}	—	0.3 × V_{DD}	V
V_{DD} supply current Run[3], f_{OP} = 8 MHz with all modules on with ADC on with ADC off Wait[4], f_{OP} = 8 MHz (all modules off) Stop, f_{OP} = 8 kHz[5] 25°C (with OSC, RTC, LCD[6], LVI on) 25°C (with OSC, RTC, LCD[6] on) 25°C (with OSC, RTC on) 25°C (all modules off)	I_{DD}	 — — — — — — — —	 — — — — — — — —	 18 15 12 10 350 50 30 1	 mA mA mA mA µA µA µA µA
Digital I/O ports Hi-Z leakage current All ports, $\overline{RST}$	I_{IL}	—	—	± 10	µA
Input current $\overline{IRQ}$	I_{IN}	—	—	± 1	µA
Capacitance Ports (as input or output)	C_{OUT} C_{IN}	— —	— —	12 8	pF
POR re-arm voltage[7]	V_{POR}	0	—	100	mV
POR rise-time ramp rate[8]	R_{POR}	0.035	—	—	V/ms
Monitor mode entry voltage (at $\overline{IRQ}$ pin)	V_{TST}	1.5 × V_{DD}	—	8	V
Pullup resistors[9] PTA0–PTA3, PTD4–PTD7 configured as KBI0–KBI7 $\overline{RST}$, $\overline{IRQ}$	 R_{PU1} R_{PU2}	 — —	 28 28	 — —	 kΩ kΩ
Low-voltage inhibit, trip falling voltage	V_{TRIPF}	4.00	4.32	4.70	V
Low-voltage inhibit, trip rising voltage	V_{TRIPR}	4.00	4.32	4.70	V

Notes:

1. V_{DD} = 4.5 to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted.
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. Run (operating) I_{DD} measured using external square wave clock source. All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects run I_{DD}.
4. Wait I_{DD} measured using external square wave clock source. All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects wait I_{DD}.
5. The 8kHz clock is from a 32kHz clock input at OSC1, for the driving the RTC.
6. LCD driver configured for high current mode.
7. Maximum is highest voltage that POR is guaranteed.
8. If minimum V_{DD} is not reached before the internal POR reset is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached.
9. R_{PU1} and R_{PU2} are measured at V_{DD} = 5.0V

23.7 3.3V DC Electrical Characteristics

Table 23-5. 3.3V DC Electrical Characteristics

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Output high voltage (I_{LOAD} = –1.0 mA) All ports	V_{OH}	V_{DD}–0.4	—	—	V
Output low voltage (I_{LOAD} = 0.8mA) All ports (I_{LOAD} = 4.0 mA) PTB2–PTB5 (I_{LOAD} = 10.0 mA) PTB0/TxD–PTB1	V_{OL}	—	—	0.4	V
Input high voltage All ports, $\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input low voltage All ports, $\overline{RST}$, $\overline{IRQ}$, OSC1	V_{IL}	V_{SS}	—	0.3 × V_{DD}	V
V_{DD} supply current					
Run[3], f_{OP} = 4 MHz					
with all modules on	I_{DD}	—	—	8	mA
with ADC on		—	—	6	mA
with ADC off		—	—	5	mA
Wait[4], f_{OP} = 4 MHz (all modules off)		—	—	3.5	mA
Stop, f_{OP} = 8 kHz[5]					
25°C (with OSC, RTC, LCD[6], LVI on)		—	—	280	µA
25°C (with OSC, RTC, LCD[6] on)		—	—	38	µA
25°C (with OSC, RTC on)		—	—	25	µA
25°C (all modules off)		—	—	1	µA
Digital I/O ports Hi-Z leakage current All ports, $\overline{RST}$	I_{IL}	—	—	± 10	µA
Input current $\overline{IRQ}$	I_{IN}	—	—	± 1	µA
Capacitance Ports (as input or output)	C_{OUT} C_{IN}	— —	— —	12 8	pF
POR re-arm voltage[7]	V_{POR}	0	—	100	mV
POR rise-time ramp rate[8]	R_{POR}	0.02	—	—	V/ms
Monitor mode entry voltage (at $\overline{IRQ}$ pin)	V_{HI}	1.5 × V_{DD}	—	2 × V_{DD}	V
Pullup resistors[9]					
PTA0–PTA3, PTD4–PTD7 configured as KBI0–KBI7	R_{PU1}	—	26	—	kΩ
$\overline{RST}$, $\overline{IRQ}$	R_{PU2}	—	28	—	kΩ
Low-voltage inhibit, trip falling voltage	V_{TRIPF}	2.40	2.57	2.88	V
Low-voltage inhibit, trip rising voltage	V_{TRIPR}	2.46	2.63	2.97	V

Notes:
1. V_{DD} = 3.0 to 3.6 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted.
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. Run (operating) I_{DD} measured using external square wave clock source. All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects run I_{DD}.
4. Wait I_{DD} measured using external square wave clock source. All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects wait I_{DD}.
5. The 8kHz clock is from a 32kHz clock input at OSC1, for the driving the RTC.
6. LCD driver configured for high current mode.
7. Maximum is highest voltage that POR is guaranteed.
8. If minimum V_{DD} is not reached before the internal POR reset is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached.
9. R_{PU1} and R_{PU2} are measured at V_{DD} = 3.3V.

23.8 5.0V Control Timing

Table 23-6. 5.0V Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Internal operating frequency[2]	f_{OP}	—	8	MHz
$\overline{RST}$ input pulse width low[3]	t_{IRL}	750	—	ns

Notes:
1. V_{SS} = 0 Vdc; timing shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.
2. Some modules may require a minimum frequency greater than dc for proper operation; see appropriate table for this information.
3. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.

23.9 3.3V Control Timing

Table 23-7. 3.3V Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Internal operating frequency[2]	f_{OP}	—	4	MHz
$\overline{RST}$ input pulse width low[3]	t_{IRL}	1.5	—	µs

Notes:
1. V_{SS} = 0 Vdc; timing shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.
2. Some modules may require a minimum frequency greater than dc for proper operation; see appropriate table for this information.
3. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.

23.10 5.0V Oscillator Characteristics

Table 23-8. 5.0V Oscillator Specifications

Characteristic	Symbol	Min	Typ	Max	Unit
Internal oscillator clock frequency	f_{ICLK}	46k	47k(1)	48k	Hz
External reference clock to OSC1(2)	f_{OSC}	dc	—	20M	Hz
Crystal reference frequency(3)	f_{XCLK}		32.768k	4.9152M	Hz
Crystal load capacitance(4)	C_L	—	—	—	
Crystal fixed capacitance	C_1	—	$2 \times C_L$ (25p)	—	F
Crystal tuning capacitance	C_2	—	$2 \times C_L$ (25p)	—	F
Feedback bias resistor	R_B	—	10M	—	Ω
Series resistor(5)	R_S	—	100k	—	Ω

Notes:
1. Typical value reflect average measurements at midpoint of voltage range, 25 °C only.
2. No more than 10% duty cycle deviation from 50%.
3. Fundamental mode crystals only.
4. Consult crystal manufacturer's data.
5. Not Required for high frequency crystals.

23.11 3.3V Oscillator Characteristics

Table 23-9. 3.3V Oscillator Specifications

Characteristic	Symbol	Min	Typ	Max	Unit
Internal oscillator clock frequency	f_{ICLK}	42.8k	43.4k(1)	44k	Hz
External reference clock to OSC1(2)	f_{OSC}	dc	—	16M	Hz
Crystal reference frequency(3)	f_{XCLK}		32.768k	4.9152M	Hz
Crystal load capacitance(4)	C_L	—	—	—	
Crystal fixed capacitance	C_1	—	$2 \times C_L$ (25p)	—	F
Crystal tuning capacitance	C_2	—	$2 \times C_L$ (25p)	—	F
Feedback bias resistor	R_B	—	10M	—	Ω
Series resistor(5)	R_S	—	100k	—	Ω

Notes:
1. Typical value reflect average measurements at midpoint of voltage range, 25 °C only.
2. No more than 10% duty cycle deviation from 50%.
3. Fundamental mode crystals only.
4. Consult crystal manufacturer's data.
5. Not Required for high frequency crystals.

23.12 5.0V ADC Electrical Characteristics

Table 23-10. ADC 5.0V Electrical Characteristics

Characteristic	Symbol	Min	Max	Unit	Notes
Supply voltage	V_{DDA}	4.5	5.5	V	V_{DDA} is an dedicated pin and should be tied to V_{DD} on the PCB with proper decoupling.
Input range	V_{ADIN}	0	V_{DDA}	V	$V_{ADIN} \leq V_{DDA}$
Resolution	B_{AD}	10	10	bits	
Absolute accuracy	A_{AD}	—	± 1.5	LSB	Includes quantization. ±0.5 LSB = ±1 ADC count.
ADC internal clock	f_{ADIC}	32 k	2 M	Hz	$t_{ADIC} = 1/f_{ADIC}$
Conversion range	R_{AD}	V_{REFL}	V_{REFH}	V	
ADC voltage reference high	V_{REFH}	—	V_{DDA} + 0.1	V	
ADC voltage reference low	V_{REFL}	V_{SSA} – 0.1	—	V	V_{SSA} is tied to V_{SS} internally.
Conversion time	t_{ADC}	16	17	t_{ADIC} cycles	
Sample time	t_{ADS}	5	—	t_{ADIC} cycles	
Monotonically	M_{AD}	Guaranteed			
Zero input reading	Z_{ADI}	000	001	HEX	$V_{ADIN} = V_{REFL}$
Full-scale reading	F_{ADI}	3FC	3FF	HEX	$V_{ADIN} = V_{REFH}$
Input capacitance	C_{ADI}	—	20	pF	Not tested.
Input impedance	R_{ADI}	20M	—	Ω	
V_{REFH}/V_{REFL}	I_{VREF}	—	1.6	mA	Not tested.

23.13 3.3V ADC Electrical Characteristics

Table 23-11. ADC 3.3V Electrical Characteristics

Characteristic	Symbol	Min	Max	Unit	Notes
Supply voltage	V_{DDA}	3.0	3.6	V	V_{DDA} is an dedicated pin and should be tied to V_{DD} on the PCB with proper decoupling.
Input range	V_{ADIN}	0	V_{DDA}	V	$V_{ADIN} \leq V_{DDA}$
Resolution	B_{AD}	10	10	bits	
Absolute accuracy	A_{AD}	—	± 1.5	LSB	Includes quantization. ±0.5 LSB = ±1 ADC count.
ADC internal clock	f_{ADIC}	32 k	2 M	Hz	$t_{ADIC} = 1/f_{ADIC}$
Conversion range	R_{AD}	V_{REFL}	V_{REFH}	V	
ADC voltage reference high	V_{REFH}	—	V_{DDA} + 0.1	V	
ADC voltage reference low	V_{REFL}	V_{SSA} – 0.1	—	V	V_{SSA} is tied to V_{SS} internally.
Conversion time	t_{ADC}	16	17	t_{ADIC} cycles	
Sample time	t_{ADS}	5	—	t_{ADIC} cycles	
Monotonically	M_{AD}	Guaranteed			
Zero input reading	Z_{ADI}	000	001	HEX	$V_{ADIN} = V_{REFL}$
Full-scale reading	F_{ADI}	3FC	3FF	HEX	$V_{ADIN} = V_{REFH}$
Input capacitance	C_{ADI}	—	20	pF	Not tested.
Input impedance	R_{ADI}	20M	—	Ω	Measured at 5V
V_{REFH}/V_{REFL}	I_{VREF}	—	1.6	mA	Not tested.

23.14 Timer Interface Module Characteristics

Characteristic	Symbol	Min	Max	Unit
Input capture pulse width	t_{TIH}, t_{TIL}	1	—	t_{CYC}

23.15 CGM Electrical Specifications

Characteristic	Symbol	Min	Typ	Max	Unit
Reference frequency	f_{RDV}	30	32.768	100	kHz
Range nominal multiplies	f_{NOM}	—	38.4	—	kHz
VCO center-of-range frequency	f_{VRS}	38.4k	—	40.0M	Hz
VCO range linear range multiplier	L	1	—	255	
VCO power-of-two-range multiplier	2^E	1	—	4	
VCO multiply factor	N	1	—	4095	
VCO prescale multiplier	2^P	1	—	8	
Reference divider factor	R	1	1	15	
VCO operating frequency	f_{VCLK}	38.4k	—	40.0M	Hz
Manual acquisition time	t_{LOCK}	—	—	50	ms
Automatic lock time	t_{LOCK}	—	—	50	ms
PLL jitter[1]	f_J	0	—	$f_{RCLK} \times 0.025\% \times 2^P N/4$	Hz

Notes:
1. Deviation of average bus frequency over 2ms. N = VCO multiplier.

23.16 5.0V SPI Characteristics

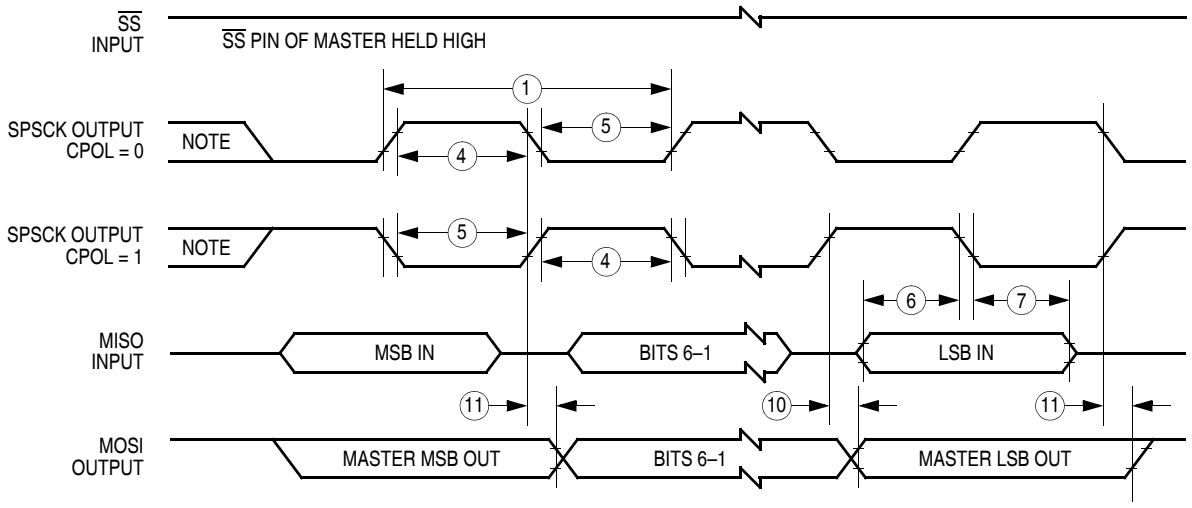
Diagram Number[1]	Characteristic[2]	Symbol	Min	Max	Unit
	Operating frequency Master Slave	 $f_{OP(M)}$ $f_{OP(S)}$	 $f_{OP}/128$ dc	 $f_{OP}/2$ f_{OP}	 MHz MHz
1	Cycle time Master Slave	 $t_{CYC(M)}$ $t_{CYC(S)}$	 2 1	 128 —	 t_{CYC} t_{CYC}
2	Enable lead time	$t_{Lead(S)}$	1	—	t_{CYC}
3	Enable lag time	$t_{Lag(S)}$	1	—	t_{CYC}
4	Clock (SPSCK) high time Master Slave	 $t_{SCKH(M)}$ $t_{SCKH(S)}$	 t_{CYC} –25 1/2 t_{CYC} –25	 64 t_{CYC} —	 ns ns
5	Clock (SPSCK) low time Master Slave	 $t_{SCKL(M)}$ $t_{SCKL(S)}$	 t_{CYC} –25 1/2 t_{CYC} –25	 64 t_{CYC} —	 ns ns
6	Data setup time (inputs) Master Slave	 $t_{SU(M)}$ $t_{SU(S)}$	 30 30	 — —	 ns ns
7	Data hold time (inputs) Master Slave	 $t_{H(M)}$ $t_{H(S)}$	 30 30	 — —	 ns ns
8	Access time, slave[3] CPHA = 0 CPHA = 1	 $t_{A(CP0)}$ $t_{A(CP1)}$	 0 0	 40 40	 ns ns
9	Disable time, slave[4]	$t_{DIS(S)}$	—	40	ns
10	Data valid time, after enable edge Master Slave[5]	 $t_{V(M)}$ $t_{V(S)}$	 — —	 50 50	 ns ns
11	Data hold time, outputs, after enable edge Master Slave	 $t_{HO(M)}$ $t_{HO(S)}$	 0 0	 — —	 ns ns

Notes:
1. Numbers refer to dimensions in **Figure 23-1** and **Figure 23-2**.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless noted; 100 pF load on all SPI pins.
3. Time to data active from high-impedance state
4. Hold time to high-impedance state
5. With 100 pF on all SPI pins

23.17 3.3V SPI Characteristics

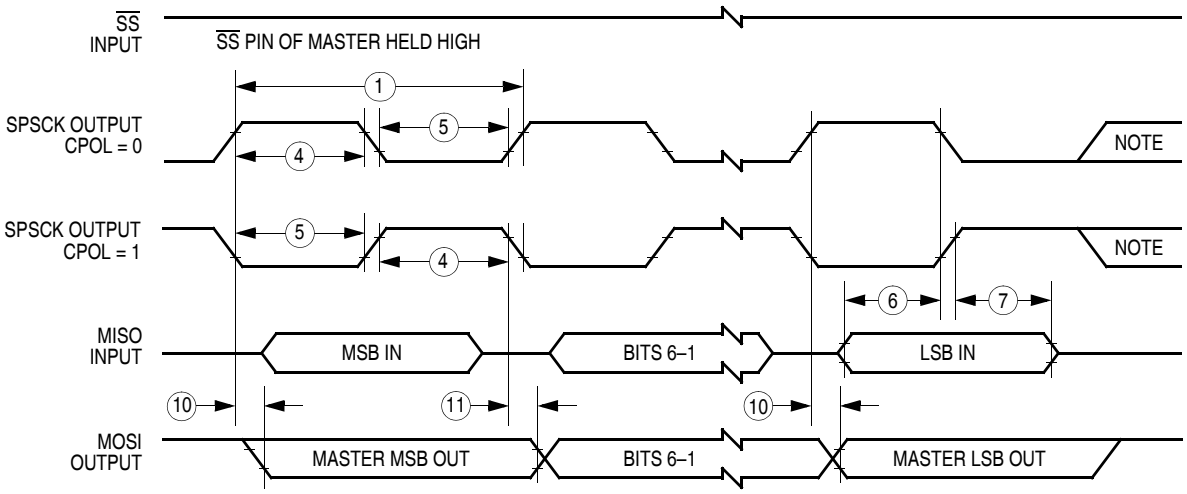
Diagram Number[1]	Characteristic[2]	Symbol	Min	Max	Unit
	Operating frequency Master Slave	$f_{OP(M)}$ $f_{OP(S)}$	$f_{OP}/128$ dc	$f_{OP}/2$ f_{OP}	MHz MHz
1	Cycle time Master Slave	$t_{CYC(M)}$ $t_{CYC(S)}$	2 1	128 —	t_{CYC} t_{CYC}
2	Enable lead time	$t_{Lead(s)}$	1	—	t_{CYC}
3	Enable lag time	$t_{Lag(s)}$	1	—	t_{CYC}
4	Clock (SPSCK) high time Master Slave	$t_{SCKH(M)}$ $t_{SCKH(S)}$	t_{CYC} –35 1/2 t_{CYC} –35	64 t_{CYC} —	ns ns
5	Clock (SPSCK) low time Master Slave	$t_{SCKL(M)}$ $t_{SCKL(S)}$	t_{CYC} –35 1/2 t_{CYC} –35	64 t_{CYC} —	ns ns
6	Data setup time (inputs) Master Slave	$t_{SU(M)}$ $t_{SU(S)}$	40 40	— —	ns ns
7	Data hold time (inputs) Master Slave	$t_{H(M)}$ $t_{H(S)}$	40 40	— —	ns ns
8	Access time, slave[3] CPHA = 0 CPHA = 1	$t_{A(CP0)}$ $t_{A(CP1)}$	0 0	50 50	ns ns
9	Disable time, slave[4]	$t_{DIS(S)}$	—	50	ns
10	Data valid time, after enable edge Master Slave[5]	$t_{V(M)}$ $t_{V(S)}$	— —	60 60	ns ns
11	Data hold time, outputs, after enable edge Master Slave	$t_{HO(M)}$ $t_{HO(S)}$	0 0	— —	ns ns

Notes:
1. Numbers refer to dimensions in **Figure 23-1** and **Figure 23-2.**
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless noted; 100 pF load on all SPI pins.
3. Time to data active from high-impedance state
4. Hold time to high-impedance state
5. With 100 pF on all SPI pins



Note: This first clock edge is generated internally, but is not seen at the SPSCK pin.

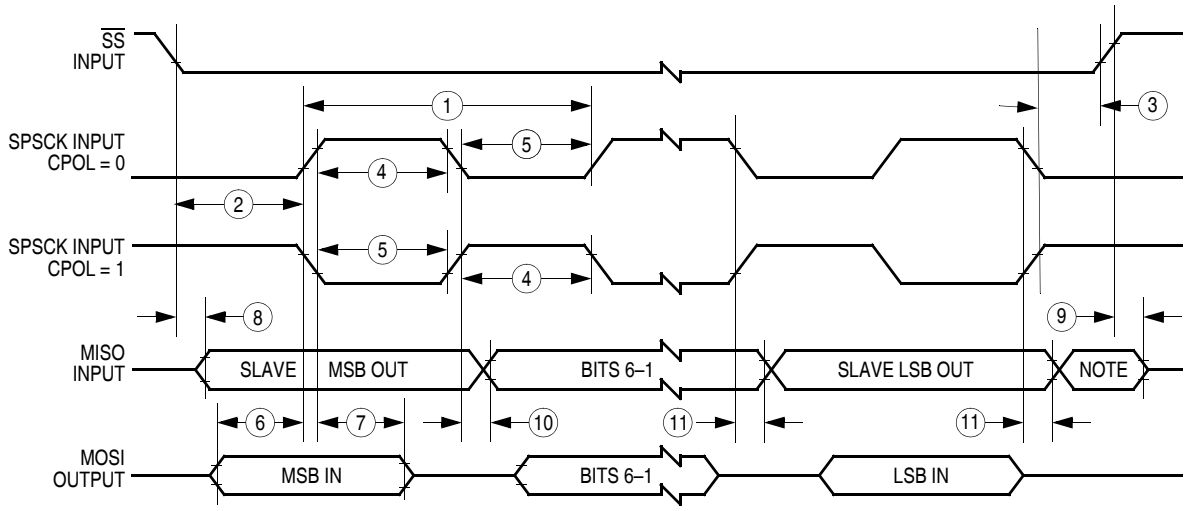
a) SPI Master Timing (CPHA = 0)



Note: This last clock edge is generated internally, but is not seen at the SPSCK pin.

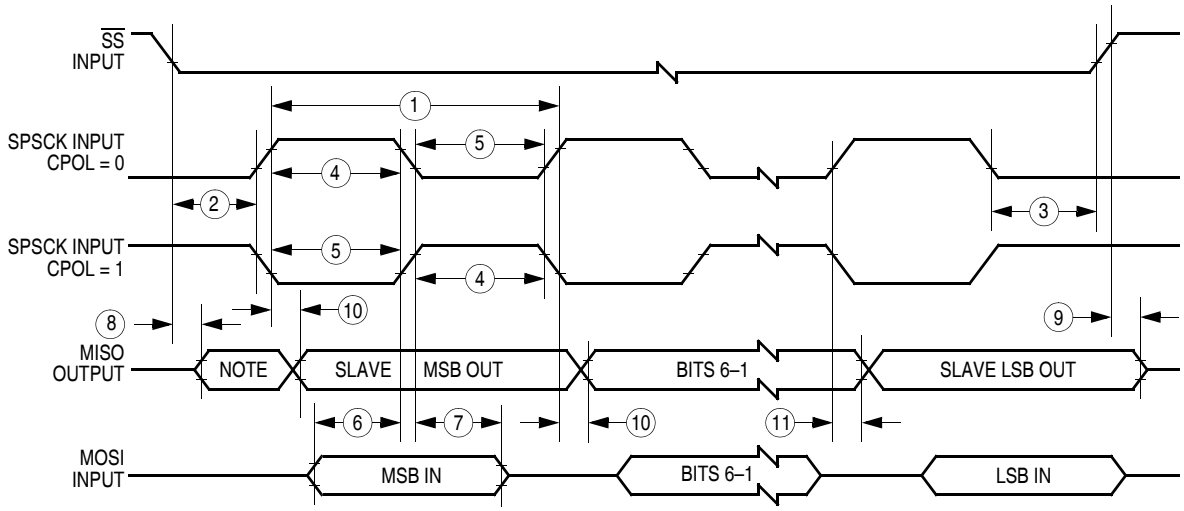
b) SPI Master Timing (CPHA = 1)

Figure 23-1. SPI Master Timing



Note: Not defined but normally MSB of character just received

a) SPI Slave Timing (CPHA = 0)



Note: Not defined but normally LSB of character previously transmitted

b) SPI Slave Timing (CPHA = 1)

Figure 23-2. SPI Slave Timing

23.18 FLASH Memory Characteristics

Table 23-12. FLASH Memory Electrical Characteristics

Characteristic	Symbol	Min.	Max.	Unit
Data retention voltage	V_{RDR}	1.3	—	V
Number of rows per page		2		Rows
Number of bytes per page		128		Bytes
Read bus clock frequency	f_{Read} (1)	32k	8M	Hz
Page erase time	t_{Erase} (2)	1	—	ms
Mass erase time	t_{MErase} (3)	4	—	ms
PGM/ERASE to HVEN setup time	t_{nvs}	10	—	µs
High-voltage hold time	t_{nvh}	5	—	µs
High-voltage hold time (mass erase)	t_{nvhl}	100	—	µs
Program hold time	t_{pgs}	5	—	µs
Program time	t_{Prog}	30	40	µs
Address/data setup time	t_{ads}	—	30	ns
Address/data hold time	t_{adh}	—	30	ns
Recovery time	t_{rcv} (4)	1	—	µs
Cumulative HV period	t_{hv} (5)	—	25	ms
Row erase endurance (6)	—	10k	—	Cycles
Row program endurance (7)	—	10k	—	Cycles
Data retention time (8)	—	10	—	Years

Notes:
1. f_{Read} is defined as the frequency range for which the FLASH memory can be read.
2. If the page erase time is longer than t_{Erase} (Min.), there is no erase-disturb, but it reduces the endurance of the FLASH memory.
3. If the mass erase time is longer than t_{MErase} (Min.), there is no erase-disturb, but is reduces the endurance of the FLASH memory.
4. It is defined as the time it needs before the FLASH can be read after turning off the high voltage charge pump, by clearing HVEN to logic 0.
5. t_{hv} is the cumulative high voltage programming time to the same row before next erase, and the same address can not be programmed twice before next erase.
6. The minimum row endurance value specifies each row of the FLASH memory is guaranteed to work for at least this many erase/program cycles.
7. The minimum row endurance value specifies each row of the FLASH memory is guaranteed to work for at least this many erase/program cycle.
8. The FLASH is guaranteed to retain data over the entire operating temperature range for at least the minimum time specified.

Section 24. Mechanical Specifications

24.1 Contents

24.2 Introduction 407
24.3 52-Pin Low-Profile Quad Flat Pack (LQFP) 408
24.4 64-Pin Low-Profile Quad Flat Pack (LQFP) 409
24.5 64-Pin Quad Flat Pack (QFP) 410

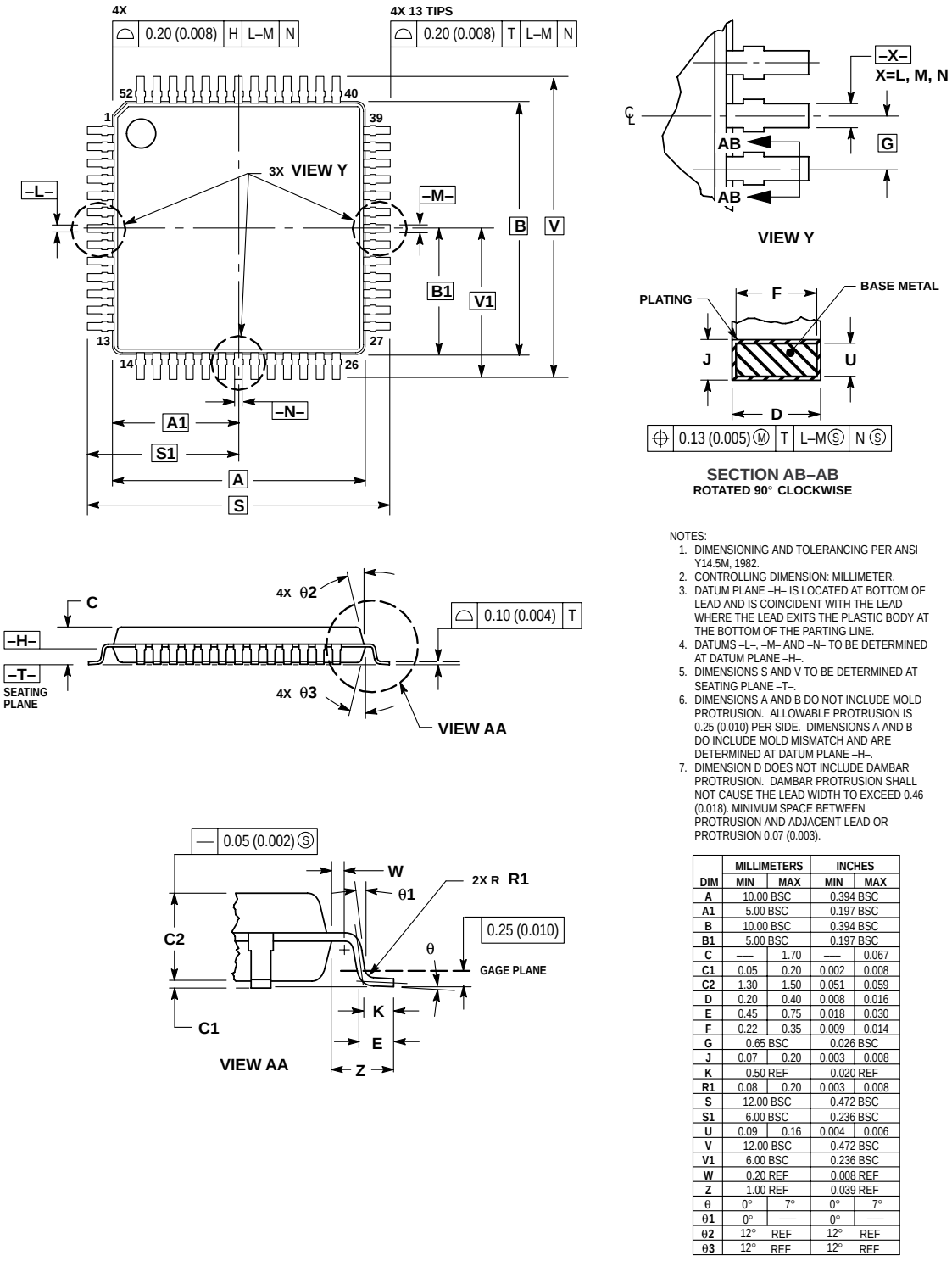
24.2 Introduction

This section gives the dimensions for:

- 52-pin low-profile quad flat pack (case no. 848D)
- 64-pin low-profile quad flat pack (case no. 840F)
- 64-pin quad flat pack (case no. 840B)

The following figures show the latest package drawings at the time of this publication. To make sure that you have the latest package specifications, please visit the Freescale website at http://freescale.com. Follow the World Wide Web on-line instructions to retrieve the current mechanical specifications.

24.3 52-Pin Low-Profile Quad Flat Pack (LQFP)

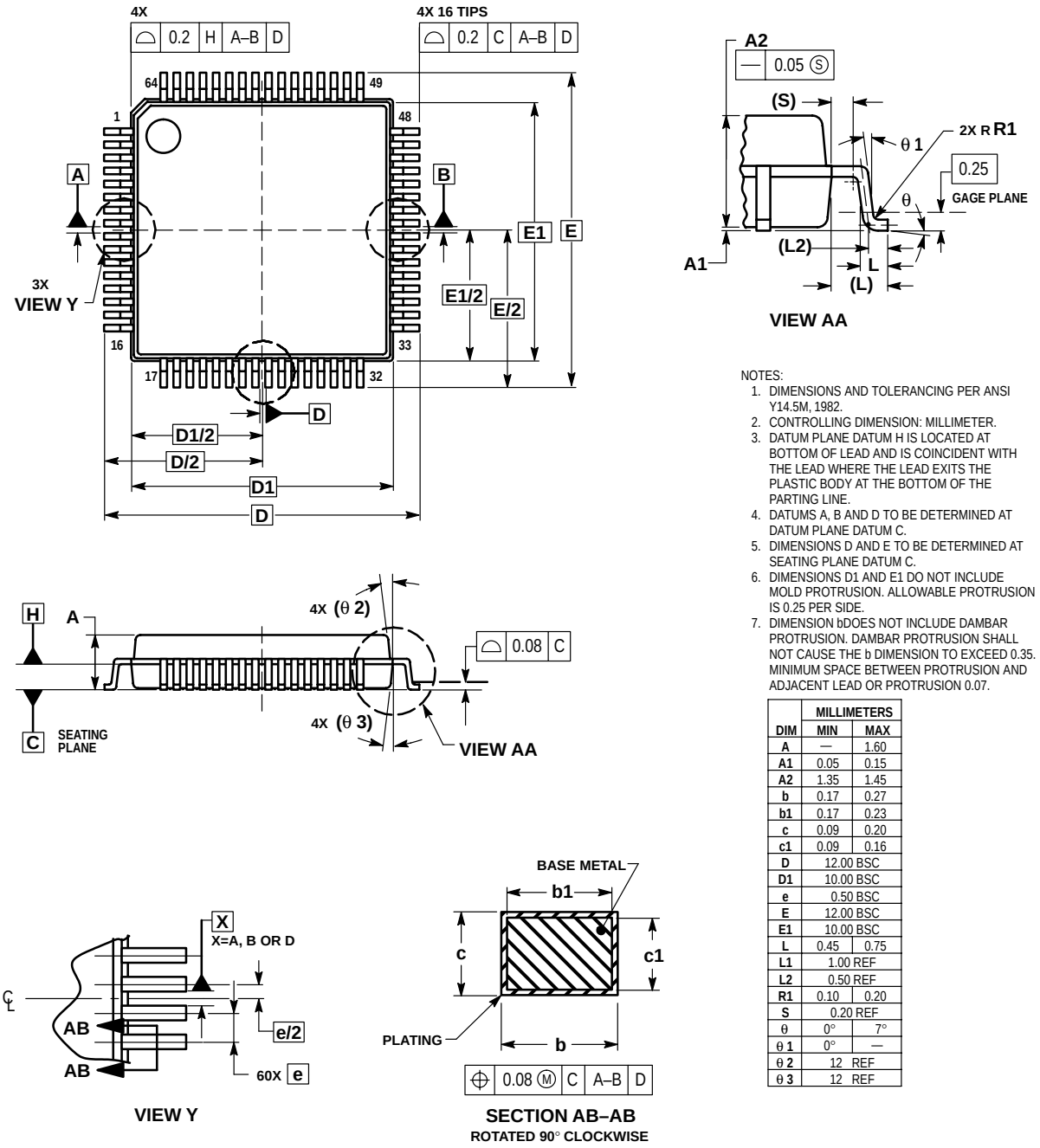


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE –H– IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS –L–, –M– AND –N– TO BE DETERMINED AT DATUM PLANE –H–.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE –T–.
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. DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.46 (0.018). MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07 (0.003).

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	10.00 BSC		0.394 BSC	
A1	5.00 BSC		0.197 BSC	
B	10.00 BSC		0.394 BSC	
B1	5.00 BSC		0.197 BSC	
C	—	1.70	—	0.067
C1	0.05	0.20	0.002	0.008
C2	1.30	1.50	0.051	0.059
D	0.20	0.40	0.008	0.016
E	0.45	0.75	0.018	0.030
F	0.22	0.35	0.009	0.014
G	0.65 BSC		0.026 BSC	
J	0.07	0.20	0.003	0.008
K	0.50 REF		0.020 REF	
R1	0.08	0.20	0.003	0.008
S	12.00 BSC		0.472 BSC	
S1	6.00 BSC		0.236 BSC	
U	0.09	0.16	0.004	0.006
V	12.00 BSC		0.472 BSC	
V1	6.00 BSC		0.236 BSC	
W	0.20 REF		0.008 REF	
Z	1.00 REF		0.039 REF	
θ	0°	7°	0°	7°
θ1	0°	—	0°	—
θ2	12°	REF	12°	REF
θ3	12°	REF	12°	REF

Figure 24-1. 52-Pin Low-Profile Quad Flat Pack (Case No. 848D)

24.4 64-Pin Low-Profile Quad Flat Pack (LQFP)

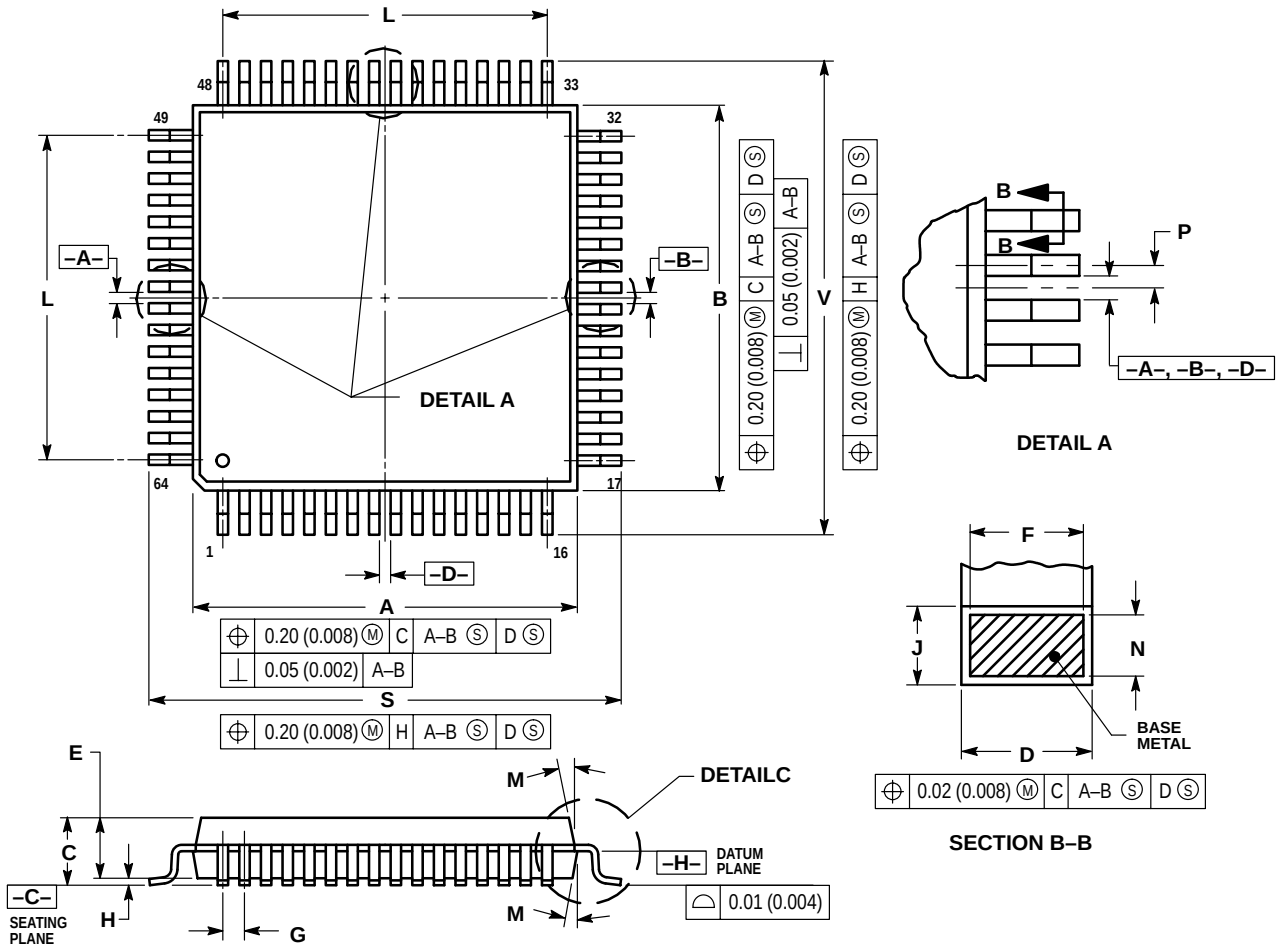


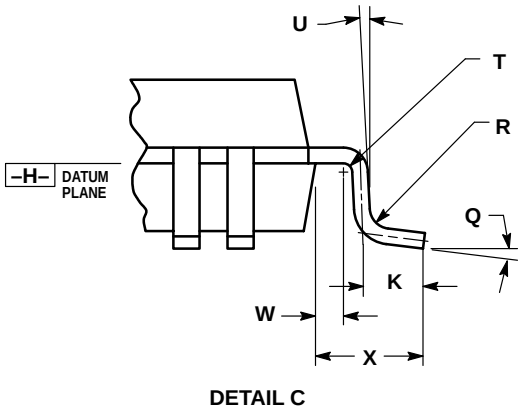
NOTES:
1. DIMENSIONS AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE DATUM H IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS A, B AND D TO BE DETERMINED AT DATUM 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 bDOES 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	
θ	0°	7°
θ 1	0°	—
θ 2	12 REF	
θ 3	12 REF	

Figure 24-2. 64-Pin Low-Profile Quad Flat Pack (Case No. 840F)

24.5 64-Pin Quad Flat Pack (QFP)





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) PER SIDE. 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	13.90	14.10	0.547	0.555
B	13.90	14.10	0.547	0.555
C	2.15	2.45	0.085	0.096
D	0.30	0.45	0.012	0.018
E	2.00	2.40	0.079	0.094
F	0.30	0.40	0.012	0.016
G	0.80 BSC		0.031 BSC	
H	—	0.25	—	0.010
J	0.13	0.23	0.005	0.009
K	0.65	0.95	0.026	0.037
L	12.00 REF		0.472 REF	
M	5°	10°	5°	10°
N	0.13	0.17	0.005	0.007
P	0.40 BSC		0.016 BSC	
Q	0°	7°	0°	7°
R	0.13	0.30	0.005	0.012
S	16.95	17.45	0.667	0.687
T	0.13	—	0.005	—
U	0°	—	0°	—
V	16.95	17.45	0.667	0.687
W	0.35	0.45	0.014	0.018
X	1.6 REF		0.063 REF	

Figure 24-3. 64-Pin Quad Flat Pack (Case No. 840B)

Section 25. Ordering Information

25.1 Contents

25.2 Introduction 411
25.3 MC Order Numbers 411

25.2 Introduction

This section contains ordering numbers for the MC68HC908LJ12.

25.3 MC Order Numbers

Table 25-1. MC Order Numbers

MC Order Number	Package	Operating Temperature Range
MC68HC908LJ12CFB	52-pin LQFP	–40 °C to +85 °C
MC68HC908LJ12CPB	64-pin LQFP	–40 °C to +85 °C
MC68HC908LJ12CFU	64-pin QFP	–40 °C to +85 °C

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

Rev. 2.1
MC68HC908LJ12/D
August 2, 2005

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. The ARM POWERED logo is a registered trademark of ARM Limited. ARM7TDMI-S is a trademark of ARM Limited. Java and all other Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. The Bluetooth trademarks are owned by their proprietor and used by Freescale Semiconductor, Inc. under license.

© Freescale Semiconductor, Inc. 2005. All rights reserved.

freescale™
semiconductor

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View MC68HC98LJ12CFUE 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