THE DATASHEET OF

MC9S12NE64CPVE

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

MC9S12NE64

Data Sheet

HCS12
Microcontrollers

MC9S12NE64V1
Rev. 1.1
06/2006

freescale.com

freescale™
semiconductor

MC9S12NE64 Data Sheet

MC9S12NE64V1
Rev. 1.1
06/2006

freescale™
semiconductor

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

http://freescale.com/

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

Revision History

Date	Revision Level	Description
September, 2004	1.0	Initial external release.
June 27, 2006	1.1	Fixed labels for addresses $0167-$0169 on Detailed Register map. Updated PHY Rx and Tx ESD protection characteristics on Table A-3.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc.
This product incorporates SuperFlash® technology licensed from SST.

© Freescale Semiconductor, Inc., 2006. All rights reserved.

LIST OF CHAPTERS

Chapter 1 MC9S12NE64 Device Overview 19
Chapter 2 64 Kbyte Flash Module (S12FTS64KV3) 67
Chapter 3 Port Integration Module (PIM9NE64V1) 105
Chapter 4 Clocks and Reset Generator (CRGV4) 141
Chapter 5 Oscillator (OSCV2) 177
Chapter 6 Timer Module (TIM16B4CV1) 181
Chapter 7 Analog-to-Digital Converter (ATD10B8CV3) 205
Chapter 8 Serial Communication Interface (SCIV3) 229
Chapter 9 Serial Peripheral Interface (SPIV3) 261
Chapter 10 Inter-Integrated Circuit (IICV2) 283
Chapter 11 Ethernet Media Access Controller (EMACV1) 307
Chapter 12 Ethernet Physical Transceiver (EPHYV2) 347
Chapter 13 Penta Output Voltage Regulator (VREGPHYV1) 379
Chapter 14 Interrupt (INTV1) 387
Chapter 15 Multiplexed External Bus Interface (MEBIV3) 395
Chapter 16 Module Mapping Control (MMCV4) 423
Chapter 17 Background Debug Module (BDMV4) 443
Chapter 18 Debug Module (DBGV1) 469

TABLE OF CONTENTS

Chapter 1
MC9S12NE64 Device Overview

1.1 Introduction . . . 19
1.1.1 Features . . . 19
1.1.2 Modes of Operation . . . 21
1.1.3 Block Diagram . . . 22
1.1.4 Device Memory Map . . . 23
1.1.5 Detailed Register Map . . . 24
1.1.6 Part ID Assignments . . . 39
1.2 Signal Description . . . 40
1.2.1 Device Pinout . . . 41
1.2.2 Signal Properties Summary . . . 43
1.2.3 Detailed Signal Descriptions . . . 47
1.2.4 Power Supply Pins . . . 57
1.3 System Clock Description . . . 59
1.4 Modes of Operation . . . 59
1.4.1 Chip Configuration Summary . . . 59
1.4.2 Security . . . 60
1.5 Low-Power Modes . . . 61
1.5.1 Stop . . . 61
1.5.2 Pseudo Stop . . . 61
1.5.3 Wait . . . 62
1.5.4 Run . . . 62
1.6 Resets and Interrupts . . . 62
1.6.1 Vectors . . . 62
1.6.2 Resets . . . 64
1.7 Block Configuration for MC9S12NE64 . . . 64
1.7.1 V_{DDR}/V_{REGEN} . . . 64
1.7.2 V_{DD1}, V_{DD2}, V_{SS1}, V_{SS2} . . . 64
1.7.3 Clock Reset Generator (CRG) . . . 65
1.7.4 Oscillator (OSC) . . . 65
1.7.5 Ethernet Media Access Controller (EMAC) . . . 65
1.7.6 Ethernet Physical Transceiver (EPHY) . . . 66
1.7.7 RAM 8K Block Description . . . 66

Chapter 2
64 Kbyte Flash Module (S12FTS64KV3)

2.1 Introduction . . . 67
2.1.1 Glossary . . . 67
2.1.2 Features . . . 67
2.1.3 Modes of Operation . . . 67
2.1.4 Block Diagram . . . 68
2.2 External Signal Description . . . 68
2.3 Memory Map and Register Definition . . . 68

2.3.1 Module Memory Map . . . 68
2.3.2 Register Descriptions . . . 73
2.4 Functional Description . . . 86
2.4.1 Flash Command Operations . . . 86
2.5 Operating Modes . . . 100
2.5.1 Wait Mode . . . 100
2.5.2 Stop Mode . . . 100
2.5.3 Background Debug Mode . . . 100
2.6 Flash Module Security . . . 100
2.6.1 Unsecuring the MCU using Backdoor Key Access . . . 100
2.6.2 Unsecuring the Flash Module in Special Single-Chip Mode using BDM . . . 102
2.7 Resets . . . 102
2.7.1 Flash Reset Sequence . . . 102
2.7.2 Reset While Flash Command Active . . . 102
2.8 Interrupts . . . 102
2.8.1 Description of Flash Interrupt Operation . . . 103

Chapter 3
Port Integration Module (PIM9NE64V1)

3.1 Introduction . . . 105
3.1.1 Features . . . 106
3.2 External Signal Description . . . 107
3.3 Memory Map and Register Descriptions . . . 110
3.3.1 Module Memory Map . . . 110
3.3.2 Register Descriptions . . . 112
3.4 Functional Description . . . 134
3.4.1 I/O Register . . . 134
3.4.2 Input Register . . . 134
3.4.3 Reduced Drive Register . . . 135
3.4.4 Pull Device Enable Register . . . 135
3.4.5 Polarity Select Register . . . 135
3.4.6 Port T . . . 135
3.4.7 Port S . . . 136
3.4.8 Port G . . . 136
3.4.9 Port H . . . 137
3.4.10 Port J . . . 137
3.4.11 Port L . . . 138
3.4.12 Port A, B, E and BKGD Pin . . . 138
3.4.13 External Pin Descriptions . . . 138
3.4.14 Low Power Options . . . 138
3.5 Initialization/Application Information . . . 138
3.5.1 Reset Initialization . . . 138
3.6 Interrupts . . . 139
3.6.1 Interrupt Sources . . . 139
3.6.2 Recovery from Stop . . . 139

Chapter 4
Clocks and Reset Generator (CRGV4)

4.1 Introduction . . . 141
4.1.1 Features . . . 141
4.1.2 Modes of Operation . . . 142
4.1.3 Block Diagram . . . 142
4.2 External Signal Description . . . 143
4.2.1 V_{DDPLL}, V_{SSPLL} — PLL Operating Voltage, PLL Ground . . . 143
4.2.2 XFC — PLL Loop Filter Pin . . . 143
4.2.3 $\overline{\text{RESET}}$ — Reset Pin . . . 144
4.3 Memory Map and Register Definition . . . 144
4.3.1 Module Memory Map . . . 144
4.3.2 Register Descriptions . . . 145
4.4 Functional Description . . . 156
4.4.1 Phase Locked Loop (PLL) . . . 156
4.4.2 System Clocks Generator . . . 159
4.4.3 Clock Monitor (CM) . . . 160
4.4.4 Clock Quality Checker . . . 160
4.4.5 Computer Operating Properly Watchdog (COP) . . . 162
4.4.6 Real-Time Interrupt (RTI) . . . 163
4.4.7 Modes of Operation . . . 163
4.4.8 Low-Power Operation in Run Mode . . . 164
4.4.9 Low-Power Operation in Wait Mode . . . 164
4.4.10 Low-Power Operation in Stop Mode . . . 168
4.5 Resets . . . 172
4.5.1 Clock Monitor Reset . . . 174
4.5.2 Computer Operating Properly Watchdog (COP) Reset . . . 174
4.5.3 Power-On Reset, Low Voltage Reset . . . 175
4.6 Interrupts . . . 176
4.6.1 Real-Time Interrupt . . . 176
4.6.2 PLL Lock Interrupt . . . 176
4.6.3 Self-Clock Mode Interrupt . . . 176

Chapter 5
Oscillator (OSCV2)

5.1 Introduction . . . 177
5.1.1 Features . . . 177
5.1.2 Modes of Operation . . . 177
5.2 External Signal Description . . . 177
5.2.1 V_{DDPLL} and V_{SSPLL} — PLL Operating Voltage, PLL Ground . . . 178
5.2.2 EXTAL and XTAL — Clock/Crystal Source Pins . . . 178
5.2.3 XCLKS — Colpitts/Pierce Oscillator Selection Signal . . . 179
5.3 Memory Map and Register Definition . . . 180
5.4 Functional Description . . . 180

5.4.1 Amplitude Limitation Control (ALC) . . . 180
5.4.2 Clock Monitor (CM) . . . 180
5.5 Interrupts . . . 180

Chapter 6
Timer Module (TIM16B4CV1)

6.1 Introduction . . . 181
6.1.1 Features . . . 181
6.1.2 Modes of Operation . . . 181
6.1.3 Block Diagrams . . . 182
6.2 External Signal Description . . . 184
6.2.1 IOC7 — Input Capture and Output Compare Channel 7 Pin . . . 184
6.2.2 IOC6 — Input Capture and Output Compare Channel 6 Pin . . . 184
6.2.3 IOC5 — Input Capture and Output Compare Channel 5 Pin . . . 184
6.2.4 IOC4 — Input Capture and Output Compare Channel 4 Pin . . . 184
6.3 Memory Map and Register Definition . . . 185
6.3.1 Module Memory Map . . . 185
6.3.2 Register Descriptions . . . 186
6.4 Functional Description . . . 201
6.4.1 Prescaler . . . 202
6.4.2 Input Capture . . . 202
6.4.3 Output Compare . . . 202
6.4.4 Pulse Accumulator . . . 202
6.4.5 Event Counter Mode . . . 203
6.4.6 Gated Time Accumulation Mode . . . 203
6.5 Resets . . . 203
6.6 Interrupts . . . 204
6.6.1 Channel [7:4] Interrupt (C[7:4]F) . . . 204
6.6.2 Pulse Accumulator Input Interrupt (PAOVI) . . . 204
6.6.3 Pulse Accumulator Overflow Interrupt (PAOVF) . . . 204
6.6.4 Timer Overflow Interrupt (TOF) . . . 204

Chapter 7
Analog-to-Digital Converter (ATD10B8CV3)

7.1 Introduction . . . 205
7.1.1 Features . . . 205
7.1.2 Modes of Operation . . . 205
7.1.3 Block Diagram . . . 206
7.2 External Signal Description . . . 206
7.2.1 ANx (x = 7, 6, 5, 4, 3, 2, 1, 0) — Analog Input Pin . . . 206
7.2.2 ETRIG3, ETRIG2, ETRIG1, and ETRIG0 — External Trigger Pins . . . 206
7.2.3 $V_{RH\ and}$ V_{RL} — High and Low Reference Voltage Pins . . . 206
7.2.4 V_{DDA} and V_{SSA} — Power Supply Pins . . . 206
7.3 Memory Map and Register Definition . . . 208

7.3.1 Module Memory Map . . . 208
7.3.2 Register Descriptions . . . 208
7.4 Functional Description . . . 226
7.4.1 Analog Sub-Block . . . 226
7.4.2 Digital Sub-Block . . . 227
7.5 Resets . . . 228
7.6 Interrupts . . . 228

Chapter 8
Serial Communication Interface (SCIV3)

8.1 Introduction . . . 229
8.1.1 Glossary . . . 229
8.1.2 Features . . . 229
8.1.3 Modes of Operation . . . 230
8.1.4 Block Diagram . . . 231
8.2 External Signal Descriptions . . . 232
8.2.1 TXD — SCI Transmit Pin . . . 232
8.2.2 RXD — SCI Receive Pin . . . 232
8.3 Memory Map and Register Definition . . . 232
8.3.1 Module Memory Map . . . 232
8.3.2 Register Descriptions . . . 232
8.4 Functional Description . . . 242
8.4.1 Infrared Interface Submodule . . . 243
8.4.2 Data Format . . . 243
8.4.3 Baud Rate Generation . . . 245
8.4.4 Transmitter . . . 246
8.4.5 Receiver . . . 249
8.4.6 Single-Wire Operation . . . 257
8.4.7 Loop Operation . . . 258
8.5 Interrupts . . . 258
8.5.1 Description of Interrupt Operation . . . 258
8.5.2 Recovery from Wait Mode . . . 259

Chapter 9
Serial Peripheral Interface (SPIV3)

9.1 Introduction . . . 261
9.1.1 Features . . . 261
9.1.2 Modes of Operation . . . 261
9.1.3 Block Diagram . . . 262
9.2 External Signal Description . . . 262
9.2.1 MOSI — Master Out/Slave In Pin . . . 262
9.2.2 MISO — Master In/Slave Out Pin . . . 263
9.2.3 $\overline{SS}$ — Slave Select Pin . . . 263
9.2.4 SCK — Serial Clock Pin . . . 263

9.3 Memory Map and Register Definition . . . 263
9.3.1 Module Memory Map . . . 263
9.3.2 Register Descriptions . . . 264
9.4 Functional Description . . . 271
9.4.1 Master Mode . . . 272
9.4.2 Slave Mode . . . 273
9.4.3 Transmission Formats . . . 274
9.4.4 SPI Baud Rate Generation . . . 277
9.4.5 Special Features . . . 278
9.4.6 Error Conditions . . . 279
9.4.7 Operation in Run Mode . . . 280
9.4.8 Operation in Wait Mode . . . 280
9.4.9 Operation in Stop Mode . . . 280
9.5 Reset . . . 281
9.6 Interrupts . . . 281
9.6.1 MODF . . . 281
9.6.2 SPIF . . . 281
9.6.3 SPTEF . . . 281

Chapter 10
Inter-Integrated Circuit (IICV2)

10.1 Introduction . . . 283
10.1.1 Features . . . 283
10.1.2 Modes of Operation . . . 284
10.1.3 Block Diagram . . . 284
10.2 External Signal Description . . . 285
10.2.1 IIC_SCL — Serial Clock Line Pin . . . 285
10.2.2 IIC_SDA — Serial Data Line Pin . . . 285
10.3 Memory Map and Register Definition . . . 285
10.3.1 Module Memory Map . . . 285
10.3.2 Register Descriptions . . . 286
10.4 Functional Description . . . 297
10.4.1 I-Bus Protocol . . . 297
10.4.2 Operation in Run Mode . . . 300
10.4.3 Operation in Wait Mode . . . 300
10.4.4 Operation in Stop Mode . . . 300
10.5 Resets . . . 301
10.6 Interrupts . . . 301
10.7 Initialization/Application Information . . . 301
10.7.1 IIC Programming Examples . . . 301

Chapter 11
Ethernet Media Access Controller (EMACV1)

11.1 Introduction . . . 307
11.1.1 Features . . . 307
11.1.2 Block Diagram . . . 308
11.2 External Signal Description . . . 308
11.2.1 MII_TXCLK — MII Transmit Clock . . . 309
11.2.2 MII_TXD[3:0] — MII Transmit Data . . . 309
11.2.3 MII_TXEN — MII Transmit Enable . . . 309
11.2.4 MII_TXER — MII Transmit Coding Error . . . 309
11.2.5 MII_RXCLK — MII Receive Clock . . . 309
11.2.6 MII_RXD[3:0] — MII Receive Data . . . 310
11.2.7 MII_RXDV — MII Receive Data Valid . . . 310
11.2.8 MII_RXER — MII Receive Error . . . 310
11.2.9 MII_CRS — MII Carrier Sense . . . 310
11.2.10 MII_COL — MII Collision . . . 310
11.2.11 MII_MDC — MII Management Data Clock . . . 311
11.2.12 MII_MDIO — MII Management Data Input/Output . . . 311
11.3 Memory Map and Register Descriptions . . . 311
11.3.1 Module Memory Map . . . 311
11.3.2 Register Descriptions . . . 312
11.4 Functional Description . . . 330
11.4.1 Ethernet Frame . . . 330
11.4.2 Receiver . . . 333
11.4.3 Transmitter . . . 338
11.4.4 Ethernet Buffers . . . 340
11.4.5 Full-Duplex Operation . . . 341
11.4.6 MII Management . . . 342
11.4.7 Loopback . . . 344
11.4.8 Software Reset . . . 345
11.4.9 Interrupts . . . 345
11.4.10 Debug and Stop . . . 345

Chapter 12
Ethernet Physical Transceiver (EPHYV2)

12.1 Introduction . . . 347
12.1.1 Features . . . 347
12.1.2 Block Diagram . . . 348
12.2 External Signal Descriptions . . . 349
12.2.1 PHY_TXP — EPHY Twisted Pair Output + . . . 349
12.2.2 PHY_TXN — EPHY Twisted Pair Output – . . . 349
12.2.3 PHY_RXP — EPHY Twisted Pair Input + . . . 349
12.2.4 PHY_RXN — EPHY Twisted Pair Input – . . . 350
12.2.5 PHY_RBIAS — EPHY Bias Control Resistor . . . 350

12.2.6 PHY_VDDRX, PHY_VSSRX — Power Supply Pins for EPHY Receiver 350
12.2.7 PHY_VDDTX, PHY_VSSTX — Power Supply Pins for EPHY Transmitter 350
12.2.8 PHY_VDDA, PHY_VSSA — Power Supply Pins for EPHY Analog 350
12.2.9 COLLED — Collision LED 350
12.2.10DUPLED — Duplex LED 350
12.2.11SPDLED — Speed LED 350
12.2.12LNKLED — Link LED 350
12.2.13ACTLEC — Activity LED 351
12.3 Memory Map and Register Descriptions 351
12.3.1 Module Memory Map 351
12.3.2 Register Descriptions 351
12.3.3 MII Registers 354
12.3.4 PHY-Specific Registers 364
12.4 Functional Description 367
12.4.1 Power Down/Initialization 368
12.4.2 Auto-Negotiation 370
12.4.3 10BASE-T 371
12.4.4 100BASE-TX 373
12.4.5 Low Power Modes 376

Chapter 13
Penta Output Voltage Regulator (VREGPHYV1)

13.1 Introduction 379
13.1.1 Overview 379
13.1.2 Features 379
13.1.3 Modes of Operation 379
13.1.4 Block Diagram 380
13.2 Signal Description 381
13.2.1 Overview 381
13.2.2 Detailed Signal Descriptions 381
13.3 Memory Map and Registers 383
13.3.1 Overview 383
13.4 Functional Description 383
13.4.1 General 383
13.4.2 REG - Regulator Core 383
13.4.3 POR - Power-On Reset 384
13.4.4 LVR - Low Voltage Reset 384
13.4.5 CTRL - Regulator Control 384
13.5 Resets 384
13.5.1 General 384
13.5.2 Description of Reset Operation 384
13.6 Interrupts 385
13.6.1 General 385

Chapter 14
Interrupt (INTV1)

14.1 Introduction . . . 387
14.1.1 Features . . . 388
14.1.2 Modes of Operation . . . 388
14.2 External Signal Description . . . 388
14.3 Memory Map and Register Definition . . . 389
14.3.1 Module Memory Map . . . 389
14.3.2 Register Descriptions . . . 389
14.4 Functional Description . . . 391
14.4.1 Low-Power Modes . . . 391
14.5 Resets . . . 391
14.6 Interrupts . . . 391
14.6.1 Interrupt Registers . . . 392
14.6.2 Highest Priority I-Bit Maskable Interrupt . . . 392
14.6.3 Interrupt Priority Decoder . . . 392
14.7 Exception Priority . . . 392

Chapter 15
Multiplexed External Bus Interface (MEBIV3)

15.1 Introduction . . . 395
15.1.1 Features . . . 395
15.1.2 Modes of Operation . . . 397
15.2 External Signal Description . . . 397
15.3 Memory Map and Register Definition . . . 399
15.3.1 Module Memory Map . . . 400
15.3.2 Register Descriptions . . . 400
15.4 Functional Description . . . 416
15.4.1 Detecting Access Type from External Signals . . . 416
15.4.2 Stretched Bus Cycles . . . 417
15.4.3 Modes of Operation . . . 417
15.4.4 Internal Visibility . . . 422
15.4.5 Low-Power Options . . . 422

Chapter 16
Module Mapping Control (MMCV4)

16.1 Introduction . . . 423
16.1.1 Features . . . 423
16.1.2 Modes of Operation . . . 424
16.2 External Signal Description . . . 424
16.3 Memory Map and Register Definition . . . 424
16.3.1 Module Memory Map . . . 424
16.3.2 Register Descriptions . . . 426
16.4 Functional Description . . . 435

16.4.1 Bus Control 435
16.4.2 Address Decoding 436
16.4.3 Memory Expansion 437

Chapter 17
Background Debug Module (BDMV4)

17.1 Introduction 443
17.1.1 Features 443
17.1.2 Modes of Operation 444
17.2 External Signal Description 444
17.2.1 BKGD — Background Interface Pin 445
17.2.2 $\overline{\text{TAGHI}}$ — High Byte Instruction Tagging Pin 445
17.2.3 $\overline{\text{TAGLO}}$ — Low Byte Instruction Tagging Pin 445
17.3 Memory Map and Register Definition 446
17.3.1 Module Memory Map 446
17.3.2 Register Descriptions 447
17.4 Functional Description 452
17.4.1 Security 452
17.4.2 Enabling and Activating BDM 452
17.4.3 BDM Hardware Commands 453
17.4.4 Standard BDM Firmware Commands 454
17.4.5 BDM Command Structure 455
17.4.6 BDM Serial Interface 457
17.4.7 Serial Interface Hardware Handshake Protocol 460
17.4.8 Hardware Handshake Abort Procedure 462
17.4.9 SYNC — Request Timed Reference Pulse 465
17.4.10 Instruction Tracing 465
17.4.11 Instruction Tagging 466
17.4.12 Serial Communication Time-Out 466
17.4.13 Operation in Wait Mode 467
17.4.14 Operation in Stop Mode 467

Chapter 18
Debug Module (DBGV1)

18.1 Introduction 469
18.1.1 Features 469
18.1.2 Modes of Operation 471
18.1.3 Block Diagram 471
18.2 External Signal Description 473
18.3 Memory Map and Register Definition 474
18.3.1 Module Memory Map 474
18.3.2 Register Descriptions 474
18.4 Functional Description 489
18.4.1 DBG Operating in BKP Mode 489

18.4.2 DBG Operating in DBG Mode . . . 491
18.4.3 Breakpoints . . . 498
18.5 Resets . . . 499
18.6 Interrupts . . . 499

Appendix A
Electrical Characteristics

A.1 Parameter Classification . . . 501
A.2 Power Supply . . . 501
A.3 Pins . . . 502
A.3.1 3.3 V I/O Pins . . . 502
A.3.2 Analog Reference, Special Function Analog . . . 502
A.3.3 Oscillator . . . 503
A.3.4 TEST . . . 503
A.4 Current Injection . . . 503
A.5 Absolute Maximum Ratings . . . 503
A.6 ESD Protection and Latch-Up Immunity . . . 504
A.7 Operating Conditions . . . 506
A.8 Power Dissipation and Thermal Characteristics . . . 506
A.9 I/O Characteristics . . . 508
A.10 Supply Currents . . . 510
A.10.1 Measurement Conditions . . . 510
A.10.2 Additional Remarks . . . 510
A.11 ATD Electrical Characteristics . . . 512
A.11.1 ATD Operating Characteristics — 3.3 V Range . . . 512
A.11.2 Factors Influencing Accuracy . . . 512
A.11.3 ATD Accuracy — 3.3 V Range . . . 513
A.12 Reset, Oscillator, and PLL Electrical Characteristics . . . 516
A.12.1 Startup . . . 516
A.12.2 Oscillator . . . 517
A.12.3 Phase-Locked Loop . . . 518
A.13 EMAC Electrical Characteristics . . . 523
A.13.1 MII Timing . . . 523
A.14 EPHY Electrical Characteristics . . . 527
A.14.1 10BASE-T Jab and Unjab Timing . . . 527
A.14.2 Auto Negotiation . . . 528
A.15 FLASH NVM Electrical Characteristics . . . 532
A.15.1 NVM timing . . . 532
A.15.2 NVM Reliability . . . 533
A.16 SPI Electrical Characteristics . . . 535
A.16.1 Master Mode . . . 535
A.16.2 Slave Mode . . . 536
A.17 Voltage Regulator Operating Characteristics . . . 539
A.17.1 MCU Power-Up and LVR Graphical Explanation . . . 539
A.17.2 Output Loads . . . 540

A.18 External Bus Timing 541

Appendix B
Schematic and PCB Layout Design Recommendations

B.1 Introduction 543
B.1.1 Schematic Designing with the MC9S12NE64 and Adding an Ethernet Interface 543
B.1.2 Power Supply Notes 545
B.1.3 Clocking Notes 545
B.1.4 EPHY Notes 545
B.1.5 EPHY LED Indicator Notes 545
B.2 PCB Design Recommendation 546
B.2.1 General PCB Design Recommendations 546
B.2.2 Ethernet PCB Design Recommendations 546

Appendix C
Package Information

C.1 112-Pin LQFP Package 549
C.2 80-Pin TQFP-EP Package 550

Chapter 1
MC9S12NE64 Device Overview

1.1 Introduction

The MC9S12NE64 is a 112-/80-pin cost-effective, low-end connectivity applications MCU family. The MC9S12NE64 is composed of standard on-chip peripherals including a 16-bit central processing unit (HCS12 CPU), 64K bytes of FLASH EEPROM, 8K bytes of RAM, Ethernet media access controller (EMAC) with integrated 10/100 Mbps Ethernet physical transceiver (EPHY), two asynchronous serial communications interface modules (SCI), a serial peripheral interface (SPI), one inter-IC bus (IIC), a 4-channel/16-bit timer module (TIM), an 8-channel/10-bit analog-to-digital converter (ATD), up to 21 pins available as keypad wakeup inputs (KWU), and two additional external asynchronous interrupts. The inclusion of a PLL circuit allows power consumption and performance to be adjusted to suit operational requirements. Furthermore, an on-chip bandgap-based voltage regulator (VREG_PHY) generates the internal digital supply voltage of 2.5 V (VDD) from a 3.15 V to 3.45 V external supply range. The MC9S12NE64 has full 16-bit data paths throughout. The 112-pin package version has a total of 70 I/O port pins and 10 input-only pins available. The 80-pin package version has a total of 38 I/O port pins and 10 input-only pins available.

1.1.1 Features

- 16-bit HCS12 core
  - HCS12 CPU
    - Upward compatible with M68HC11 instruction set
    - Interrupt stacking and programmer's model identical to M68HC11
    - Instruction queue
    - Enhanced indexed addressing
  - Memory map and interface (MMC)
  - Interrupt control (INT)
  - Background debug mode (BDM)
  - Enhanced debug12 module, including breakpoints and change-of-flow trace buffer (DBG)
  - Multiplexed expansion bus interface (MEBI) — available only in 112-pin package version
- Wakeup interrupt inputs
  - Up to 21 port bits available for wakeup interrupt function with digital filtering
- Memory
  - 64K bytes of FLASH EEPROM
  - 8K bytes of RAM
- Analog-to-digital converter (ATD)
  - One 8-channel module with 10-bit resolution
  - External conversion trigger capability

- Timer module (TIM)
  - 4-channel timer
  - Each channel configurable as either input capture or output compare
  - Simple PWM mode
  - Modulo reset of timer counter
  - 16-bit pulse accumulator
  - External event counting
  - Gated time accumulation
- Serial interfaces
  - Two asynchronous serial communications interface (SCI)
  - One synchronous serial peripheral interface (SPI)
  - One inter-IC bus (IIC)
- Ethernet Media access controller (EMAC)
  - IEEE 802.3 compliant
  - Medium-independent interface (MII)
  - Full-duplex and half-duplex modes
  - Flow control using pause frames
  - MII management function
  - Address recognition
    - Frames with broadcast address are always accepted or always rejected
    - Exact match for single 48-bit individual (unicast) address
    - Hash (64-bit hash) check of group (multicast) addresses
    - Promiscuous mode
- Ethertype filter
- Loopback mode
- Two receive and one transmit Ethernet buffer interfaces
- Ethernet 10/100 Mbps transceiver (EPHY)
  - IEEE 802.3 compliant
  - Digital adaptive equalization
  - Half-duplex and full-duplex
  - Auto-negotiation next page ability
  - Baseline wander (BLW) correction
  - 125-MHz clock generator and timing recovery
  - Integrated wave-shaping circuitry
  - Loopback modes
- CRG (clock and reset generator module)
  - Windowed COP watchdog
  - Real-time interrupt

  - Clock monitor
  - Pierce oscillator
  - Phase-locked loop clock frequency multiplier
  - Limp home mode in absence of external clock
  - 25-MHz crystal oscillator reference clock
- Operating frequency
  - 50 MHz equivalent to 25 MHz bus speed for single chip
  - 32 MHz equivalent to 16 MHz bus speed in expanded bus modes
- Internal 2.5-V regulator
  - Supports an input voltage range from 3.3 V ± 5%
  - Low-power mode capability
  - Includes low-voltage reset (LVR) circuitry
- 80-pin TQFP-EP or 112-pin LQFP package
  - Up to 70 I/O pins with 3.3 V input and drive capability (112-pin package)
  - Up to two dedicated 3.3 V input only lines ($\overline{\text{IRQ}}$, $\overline{\text{XIRQ}}$)
- Development support
  - Single-wire background debug™ mode (BDM)
  - On-chip hardware breakpoints
  - Enhanced DBG debug features

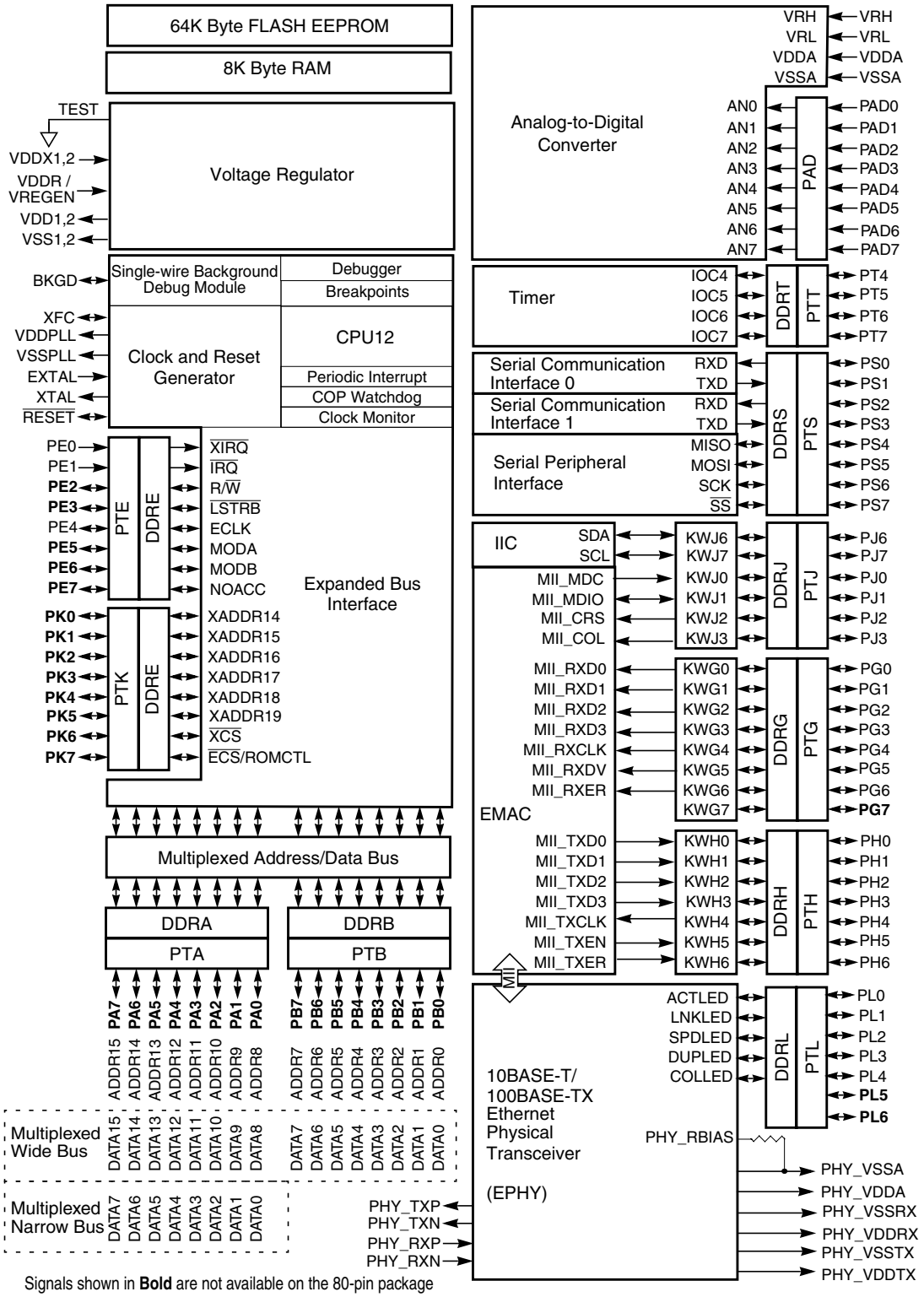
1.1.2 Modes of Operation

- Normal modes
  - Normal single-chip mode
  - Normal expanded wide mode[1]
  - Normal expanded narrow mode[1]
  - Emulation expanded wide mode[1]
  - Emulation expanded narrow mode[1]
- Special operating modes
  - Special single-chip mode with active background debug mode
- Each of the above modes of operation can be configured for three low-power submodes
  - Stop mode
  - Pseudo stop mode
  - Wait mode
- Secure operation, preventing the unauthorized read and write of the memory contents[2]

1.MEBI is available only in the 112-pin package and specified at a maximum speed of 16 MHz. If using MEBI from 2.5 MHz to 16 MHz, only 10BASE-T communication is available.

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

1.1.3 Block Diagram



Signals shown in **Bold** are not available on the 80-pin package

Figure 1-1. MC9S12NE64 Block Diagram

1.1.4 Device Memory Map

Table 1-1 shows the device register map of the MC9S12NE64 after reset. Figure 1-1 illustrates the full device memory map with FLASH and RAM.

Table 1-1. Device Register Map Overview

Address	Module[1]	Size (in Bytes)
$0000 – $0017	CORE (Ports A, B, E, Modes, Inits — MMC, INT, MEBI)	24
$0018 – $0019	Reserved	2
$001A – $001B	Device ID register (PARTID)	2
$001C – $001F	CORE (MEMSIZ, $\overline{\text{IRQ}}$, HPRIO — INT, MMC)	4
$0020 – $002F	CORE (DBG)	16
$0030 – $0033	CORE (PPAGE, Port K — MEBI, MMC)	4
$0034 – $003F	Clock and Reset Generator (PLL, RTI, COP)	12
$0040 – $006F	Standard Timer 16-bit 4 channels (TIM)	48
$0070 – $007F	Reserved	16
$0080 – $009F	Analog-to-Digital Converter 10-bit, 8-channel (ATD)	32
$00A0 – $00C7	Reserved	40
$00C8 – $00CF	Serial Communications Interface 0 (SCI0)	8
$00D0 – $00D7	Serial Communications Interface 1 (SCI1)	8
$00D8 – $00DF	Serial Peripheral Interface (SPI)	8
$00E0 – $00E7	Inter IC Bus (IIC)	8
$00E8 – $00FF	Reserved	24
$0100 – $010F	FLASH Control Register	16
$0110 – $011F	Reserved	16
$0120 – $0123	Ethernet Physical Interface (EPHY)	4
$0124 – $013F	Reserved	28
$0140 – $016F	Ethernet Media Access Controller (EMAC)	48
$0170 – $023F	Reserved	208
$0240 – $026F	Port Integration Module (PIM)	48
$0270 – $03FF	Reserved	400

1 Information about the HCS12 core can be found in the MMC, INT, MEBI, BDM, and DBG block description chapters in this data sheet, and also in the HCS12 CPU Reference Manual, S12CPUV2/D.

This figure shows a suggested map, which is not the map out of reset. After reset the map is:

$0000 – $03FF: register space
$0000 – $1FFF: 7K RAM (1K RAM hidden behind register space)

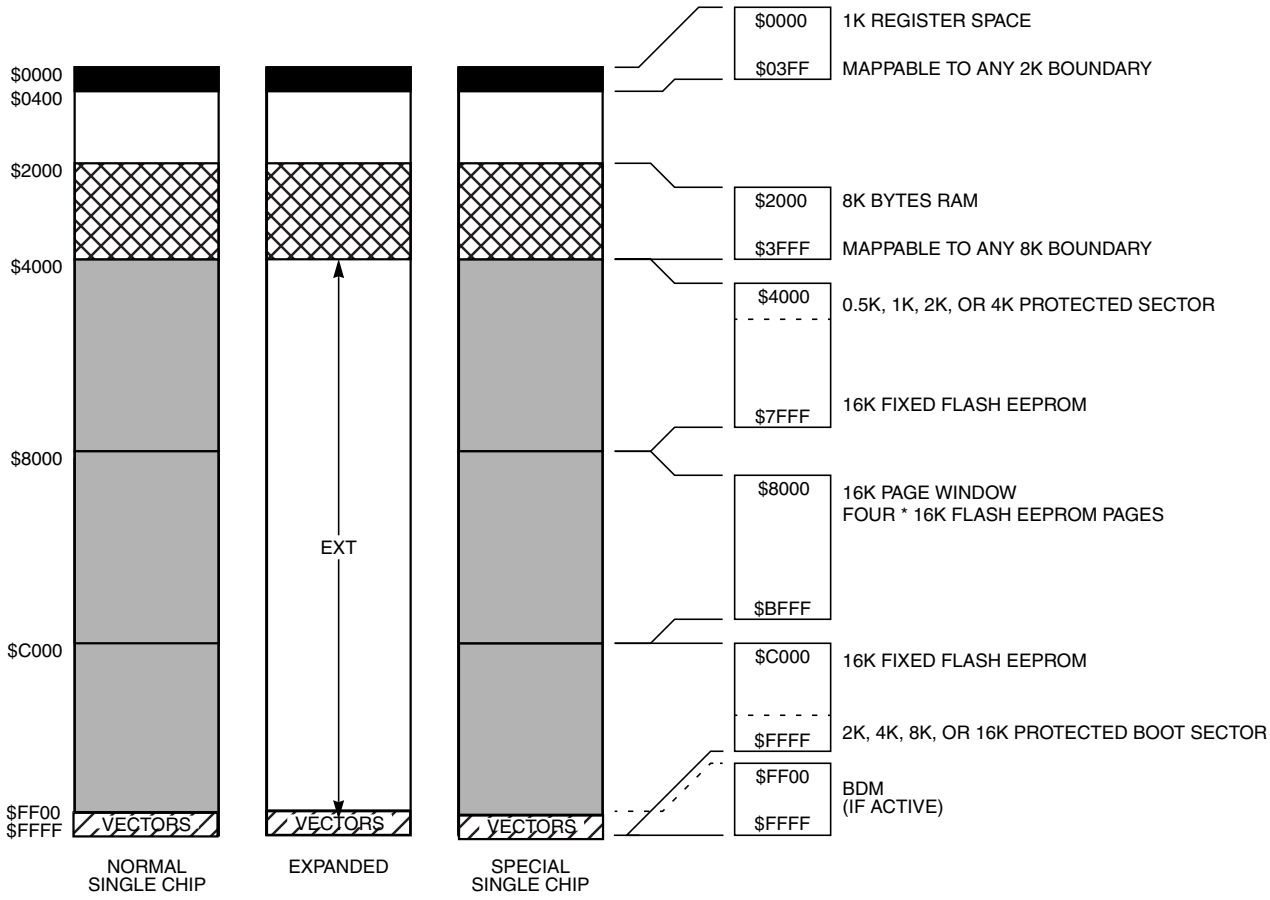


Figure 1-2. MC9S12NE64 User Configurable Memory Map

1.1.5 Detailed Register Map

The following tables show the register maps of the MC9S12NE64. For detailed information about register functions, please see the appropriate block description chapter.

$0000 - $000F Multiplexed External Bus Interface Module (MEBI) Map 1 of 3

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0000	PORTA	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0001	PORTB	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0002	DDRA	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0003	DDRB	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0004	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0005 –$0007	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0008	PORTE	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0009	DDRE	Read:	Bit 7	6	5	4	3	2	0	0
		Write:								
$000A	PEAR	Read:	NOACCE	0	PIPOE	NECLK	LSTRE	RDWE	0	0
		Write:								
$000B	MODE	Read:	MODC	MODB	MODA	0	IVIS	0	EMK	EME
		Write:								
$000C	PUCR	Read:	PUPKE	0	0	PUPEE	0	0	PUPBE	PUPAE
		Write:								
$000D	RDRIV	Read:	RDPK	0	0	RDPE	0	0	RDPB	RDPA
		Write:								
$000E	EBICTL	Read:	0	0	0	0	0	0	0	ESTR
		Write:								
$000F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0010 - $0014 Module Mapping Control Module (MMC) Map 1 of 4

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0010	INITRM	Read:	RAM15	RAM14	RAM13	RAM12	RAM11	0	0	RAMHAL
		Write:								
$0011	INITRG	Read:	0	REG14	REG13	REG12	REG11	0	0	0
		Write:								
$0012	INITEE	Read:	EE15	EE14	EE13	EE12	EE11	0	0	EEON
		Write:								
$0013	MISC	Read:	0	0	0	0	EXSTR1	EXSTR0	ROMHM	ROMON
		Write:								
$0014	MTST0	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

$0015 - $0016 Interrupt Module (INT) Map 1 of 2

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0015	ITCR	Read:	0	0	0	WRTINT	ADR3	ADR2	ADR1	ADR0
		Write:								
$0016	ITEST	Read:	INTE	INTC	INTA	INT8	INT6	INT4	INT2	INT0
		Write:								

$0017 - $0017 Module Mapping Control Module (MMC) Map 2 of 4

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0017	MTST1	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

$0018 - $0019 Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0018 – $0019	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$001A - $001B Miscellaneous Peripherals

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$001A	PARTIDH	Read:	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
		Write:								
$001B	PARTIDL	Read:	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
		Write:								

$001C - $001D Module Mapping Control Module (MMC) Map 3 of 4

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$001C	MEMSIZ0	Read:	REG_SW0	0	EEP_SW1	EEP_SW0	0	RAM_SW2	RAM_SW1	RAM_SW0
		Write:								
$001D	MEMSIZ1	Read:	ROM_SW1	ROM_SW0	0	0	0	0	PAG_SW1	PAG_SW0
		Write:								

$001E - $001E Multiplexed External Bus Interface Module (MEBI) Map 2 of 3

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$001E	IRQCR	Read:	IRQE	IRQEN	0	0	0	0	0	0
		Write:								

$001F - $001F Interrupt Module (INT) Map 2 of 2

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$001F	HPRIO	Read:	PSEL7	PSEL6	PSEL5	PSEL4	PSEL3	PSEL2	PSEL1	0
		Write:								

$0020 - $002F Debug Module (DBG) Including BKP Map 1 of 1

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0020	DBGC1	Read:	DBGEN	ARM	TRGSEL	BEGIN	DBGBRK	0	CAPMOD	
		Write:								
$0021	DBGSC	Read:	AF	BF	CF	0	TRG			
		Write:								
$0022	DBGTBH	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
$0023	DBGTBL	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
$0024	DBGCNT	Read:	TBF	0	CNT					
		Write:								
$0025	DBGCCX	Read:	PAGSEL		EXTCMP					
		Write:								
$0026	DBGCCH	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$0027	DBGCCL	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0028	DBGC2 (BKPCT0)[1]	Read:	BKABEN	FULL	BDM	TAGAB	BKCEN	TAGC	RWCEN	RWC
		Write:								
$0029	DBGC3 (BKPCT1)[1]	Read:	BKAMBH	BKAMBL	BKBMBH	BKBMBL	RWAEN	RWA	RWBEN	RWB
		Write:								
$002A	DBGCAX (BKP0X)[1]	Read:	PAGSEL		EXTCMP					
		Write:								
$002B	DBGCAH (BKP0H)[1]	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$002C	DBGCAL (BKP0L)[1]	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$002D	DBGCBX (BKP1X)[1]	Read:	PAGSEL		EXTCMP					
		Write:								
$002E	DBGCBH (BKP1H)[1]	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$002F	DBGCBL (BKP1L)[1]	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

[1]Legacy HCS12 MCUs used this name for this register.

$0030 - $0031 Module Mapping Control Module (MMC) Map 4 of 4

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0030	PPAGE	Read:	0	0	PIX5	PIX4	PIX3	PIX2	PIX1	PIX0
		Write:								
$0031	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0032 - $0033 Multiplexed External Bus Interface Module (MEBI) Map 3 of 3

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0032	PORTK	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0033	DDRK	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

$0034 - $003F Clock and Reset Generator (CRG)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0034	SYNR	Read:	0	0	SYN5	SYN4	SYN3	SYN2	SYN1	SYN0
		Write:								
$0035	REFDV	Read:	0	0	0	0	REFDV3	REFDV2	REFDV1	REFDV0
		Write:								
$0036	CTFLG Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0037	CRGFLG	Read:	RTIF	PORF	LVRF	LOCKIF	LOCK	TRACK	SCMIF	SCM
		Write:								
$0038	CRGINT	Read:	RTIE	0	0	LOCKIE	0	0	SCMIE	0
		Write:								
$0039	CLKSEL	Read:	PLLSEL	PSTP	SYSWAI	ROAWAI	PLLWAI	CWAI	RTIWAI	COPWAI
		Write:								
$003A	PLLCTL	Read:	CME	PLLON	AUTO	ACQ	0	PRE	PCE	SCME
		Write:								
$003B	RTICTL	Read:	0	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
		Write:								
$003C	COPCTL	Read:	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
		Write:								
$003D	FORBYP Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$003E	CTCTL Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$003F	ARMCOP	Read:	0	0	0	0	0	0	0	0
		Write:	Bit 7	6	5	4	3	2	1	Bit 0

$0040 - $006F 16-Bit, 4-Channel Timer Module (TIM) (Sheet 1 of 2)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0040	TIOS	Read:	IOS7	IOS6	IOS5	IOS4	0	0	0	0
		Write:								
$0041	CFORC	Read:	0	0	0	0	0	0	0	0
		Write:	FOC7	FOC6	FOC5	FOC4				
$0042	OC7M	Read:	OC7M7	OC7M6	OC7M5	OC7M4	0	0	0	0
		Write:								
$0043	OC7D	Read:	OC7D7	OC7D6	OC7D5	OC7D4	0	0	0	0
		Write:								
$0044	TCNT (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$0045	TCNT (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0046	TSCR1	Read:	TEN	TSWAI	TSFRZ	TFFCA	0	0	0	0
		Write:								
$0047	TTOV	Read:	TOV7	TOV6	TOV5	TOV4	0	0	0	0
		Write:								
$0048	TCTL1	Read:	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
		Write:								
$0049	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$004A	TCTL3	Read:	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
		Write:								
$004B	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$004C	TIE	Read:	C7I	C6I	C5I	C4I	0	0	0	0
		Write:								
$004D	TSCR2	Read:	TOI	0	0	0	TCRE	PR2	PR1	PR0
		Write:								
$004E	TFLG1	Read:	C7F	C6F	C5F	C4F	0	0	0	0
		Write:								
$004F	TFLG2	Read:	TOF	0	0	0	0	0	0	0
		Write:								
$0050 – $0057	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0058	TC4 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$0059	TC4 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$005A	TC5 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$005B	TC5 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$005C	TC6 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								

$0040 - $006F 16-Bit, 4-Channel Timer Module (TIM) (Sheet 2 of 2)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$005D	TC6 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$005E	TC7 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$005F	TC7 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0060	PACTL	Read:	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
		Write:								
$0061	PAFLG	Read:	0	0	0	0	0	0	PAOVF	PAIF
		Write:								
$0062	PACNT (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$0063	PACNT (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0064 – $006F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0080 - $009F 10-Bit, 8-Channel Analog-to-Digital Converter Module (ATD)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0080	ATDCTL0	Read:	0	0	0	0	0	WRAP2	WRAP1	WRAP0
		Write:								
$0081	ATDCTL1	Read:	ETRIG SEL	0	0	0	0	ETRIG CH2	ETRIG CH1	ETRIG CH0
		Write:								
$0082	ATDCTL2	Read:	ADPU	AFFC	AWAI	ETRIGLE	ETRIGP	ETRIGE	ASCIE	ASCIF
		Write:								
$0083	ATDCTL3	Read:	0	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
		Write:								
$0084	ATDCTL4	Read:	SRES8	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
		Write:								
$0085	ATDCTL5	Read:	DJM	DSGN	SCAN	MULT	0	CC	CB	CA
		Write:								
$0086	ATDSTAT0	Read:	SCF	0	ETORF	FIFOR	0	CC2	CC1	CC0
		Write:								
$0087	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0088	ATDTEST0 Reserved	Read:	U	U	U	U	U	U	U	U
		Write:								
$0089	ATDTEST1	Read:	U	U	0	0	0	0	0	SC
		Write:								
$008A	Unimplemented	Read:	U	U	U	U	U	U	U	U
		Write:								
$008B	ATDSTAT1	Read:	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
		Write:								

$0080 - $009F 10-Bit, 8-Channel Analog-to-Digital Converter Module (ATD)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$008C	Unimplemented	Read:	U	U	U	U	U	U	U	U
		Write:								
$008D	ATDDIEN	Read:	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0
		Write:								
$008E	Unimplemented	Read:	U	U	U	U	U	U	U	U
		Write:								
$008F	PORTAD	Read:	PTAD7	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
		Write:								
$0090	ATDDR0H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$0091	ATDDR0L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
$0092	ATDDR1H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$0093	ATDDR1L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
$0094	ATDDR2H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$0095	ATDDR2L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
$0096	ATDDR3H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$0097	ATDDR3L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
$0098	ATDDR4H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$0099	ATDDR4L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
$009A	ATDDR5H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$009B	ATDDR5L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
$009C	ATDDR6H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$009D	ATDDR6L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
$009E	ATDDR7H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
$009F	ATDDR7L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								

$00A0 - $00C7 Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00A0 – $00C7	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$00C8 - $00CF Asynchronous Serial Communications Interface Module (SCI0)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00C8	SCIBDH	Read:	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
		Write:								
$00C9	SCIBDL	Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Write:								
$00CA	SCICR1	Read:	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		Write:								
$00CB	SCICR2	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
$00CC	SCISR1	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
$00CD	SCISR2	Read:	0	0	0	0	0	BRK13	TXDIR	RAF
		Write:								
$00CE	SCIDRH	Read:	R8	T8	0	0	0	0	0	0
		Write:								
$00CF	SCIDRL	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0

$00D0 - $00D7 Asynchronous Serial Communications Interface Module (SCI1)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00D0	SCIBDH	Read:	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
		Write:								
$00D1	SCIBDL	Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Write:								
$00D2	SCICR1	Read:	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		Write:								
$00D3	SCICR2	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
$00D4	SCISR1	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
$00D5	SCISR2	Read:	0	0	0	0	0	BRK13	TXDIR	RAF
		Write:								
$00D6	SCIDRH	Read:	R8	T8	0	0	0	0	0	0
		Write:								
$00D7	SCIDRL	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0

$00D8 - $00DF Serial Peripheral Interface Module (SPI)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00D8	SPICR1	Read:	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		Write:								
$00D9	SPICR2	Read:	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		Write:								
$00DA	SPIBR	Read:	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		Write:								
$00DB	SPISR	Read:	SPIF	0	SPTEF	MODF	0	0	0	0
		Write:								
$00DC	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$00DD	SPIDR	Read:	Bit7	6	5	4	3	2	1	Bit0
		Write:								
$00DE	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$00DF	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$00E0 - $00E7 Inter-IC Bus Module (IIC)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00E0	IBAD	Read:	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
		Write:								
$00E1	IBFD	Read:	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
		Write:								
$00E2	IBCR	Read:	IBEN	IBIE	MS/$\overline{SL}$	Tx/$\overline{Rx}$	TXAK	0	0	IBSWAI
		Write:						RSTA		
$00E3	IBSR	Read:	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
		Write:								
$00E4	IBDR	Read:	D7	D6	D5	D4	D3	D2	D1	D0
		Write:								
$00E5 – $00E7	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$00E8 - $00FF Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00E8- $00FF	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0100 - $010F FLASH Control Register (fts64k)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0100	FCLKDIV	Read:	FDIVLD	PRDIV8	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
		Write:								
$0101	FSEC	Read:	KEYEN	NV6	NV5	NV4	NV3	NV2	SEC1	SEC0
		Write:								
$0102	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0103	FCNFG	Read:	CBEIE	CCIE	KEYACC	0	0	0	0	0
		Write:								
$0104	FPROT	Read:	FPOPEN	NV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
		Write:								
$0105	FSTAT	Read:	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
		Write:								
$0106	FCMD	Read:	0	CMDB6	CMDB5	0	0	CMDB2	0	CMDB0
		Write:								
$0107 – $010F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0110 - $011F Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0110 – $011F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0120 - $0123 Ethernet Physical Transceiver Module (EPHY)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0120	EPHYCTL0	Read:	EPHYEN	ANDIS	DIS100	DIS10	LEDEN	EPHYWAI	0	EPHYIEN
		Write:								
$0121	EPHYCTL1	Read:	0	0	0	PHYADD4	PHYADD3	PHYADD2	PHYADD1	PHYADD0
		Write:								
$0122	EPHYSR	Read:	0	0	100DIS	10DIS	0	0	0	EPHYIF
		Write:								
$0123	EPHYTST Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0124 - $013F Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0124 – $013F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0140 - $016F Ethernet Media Access Controller (EMAC)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0140	NETCTL	Read:	EMACE	0	0	ESWAI	EXTPHY	MLB	FDX	0
		Write:								
$0141	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0142	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0143	RXCTS	Read:	RXACT	0	0	RFCE	0	PROM	CONMC	BCREJ
		Write:								
$0144	TXCTS	Read:	TXACT	0	CSLF	PTRC	SSB	0	0	0
		Write:							TCMD	
$0145	ETCTL	Read:	FPET	0	0	FEMW	FIPV6	FARP	FIPV4	FIEEE
		Write:								
$0146	ETYPE	Read:	ETYPE[15:8]							
		Write:								
$0147	ETYPE	Read:	ETYPE[7:0]							
		Write:								
$0148	PTIME	Read:	PTIME[15:8]							
		Write:								
$0149	PTIME	Read:	PTIME[7:0]							
		Write:								
$014A	IEVENT [15:8]	Read:	RFCIF	0	BREIF	RXEIF	RXAOIF	RXBOIF	RXACIF	RXBCIF
		Write:								
$014B	IEVENT [7:0]	Read:	MMCIF	0	LCIF	ECIF	0	0	TXCIF	0
		Write:								
$0141C	IMASK [15:8]	Read:	RFCIE	0	BREIE	RXEIE	RXAOIE	RXBOIE	RXACIE	RXBCIE
		Write:								
$014D	IMASK [7:0]	Read:	MMCIE	0	LCIE	ECIE	0	0	TXCIE	0
		Write:								
$014E	SWRST	Read:	0	0	0	0	0	0	0	0
		Write:	MACRST							
$014F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0150	MPADR	Read:	0	0	0	PADDR				
		Write:								
$0151	MRADR	Read:	0	0	0	RADDR				
		Write:								
$0152	MWDATA	Read:	WDATA[15:8]							
		Write:								
$0123	MWDATA	Read:	WDATA[7:0]							
		Write:								
$0154	MRDATA	Read:	RDATA[15:8]							
		Write:								
$0155	MRDATA	Read:	RDATA[7:0]							
		Write:								

$0140 - $016F Ethernet Media Access Controller (EMAC) (Continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0156	MCMST	Read:	0	0	BUSY	NOPRE	MDCSEL			
		Write:	OP							
$0157	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0158	BUFCFG [15:8]	Read:	0	BUFMAP			0	MAXFL[10:8]		
		Write:								
$0159	BUFCFG [7:0]	Read:	MAXFL[7:0]							
		Write:								
$015A	RXAEFP [15:8]	Read:	0	0	0	0	0	RXAEFP[10:8]		
		Write:								
$015B	RXAEFP [7:0]	Read:	RXAEFP[7:0]							
		Write:								
$015C	RXBEFP [15:8]	Read:	0	0	0	0	0	RXBEFP[10:8]		
		Write:								
$015D	RXBEFP [7:0]	Read:	RXBEFP[7:0]							
		Write:								
$015E	TXEFP [15:8]	Read:	0	0	0	0	0	TXEFP[10:8]		
		Write:								
$015F	TXEFP	Read:	TXEFP[7:0]							
		Write:								
$0160	MCHASH	Read:	MCHASH[63:56]							
		Write:								
$0161	MCHASH	Read:	MCHASH[55:48]							
		Write:								
$0162	MCHASH	Read:	MCHASH[47:40]							
		Write:								
$0163	MCHASH	Read:	MCHASH[39:32]							
		Write:								
$0164	MCHASH	Read:	MCHASH[31:24]							
		Write:								
$0165	MCHASH	Read:	MCHASH[23:16]							
		Write:								
$0166	MCHASH	Read:	MCHASH[15:8]							
		Write:								
$0167	MCHASH	Read:	MCHASH[7:0]							
		Write:								
$0168	MACAD	Read:	MACAD[47:40]							
		Write:								
$0169	MACAD	Read:	MACAD[39:32]							
		Write:								
$016A	MACAD	Read:	MACAD[31:24]							
		Write:								
$016B	MACAD	Read:	MACAD[23:16]							
		Write:								
$016C	MACAD	Read:	MACAD[15:8]							
		Write:								

$0140 - $016F Ethernet Media Access Controller (EMAC) (Continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$016D	MACAD	Read:	MACAD[7:0]							
		Write:								
$016E	EMISC	Read:	INDEX			0	0	MISC[10:8]		
		Write:								
$016F	EMISC	Read:	MISC[7:0]							
		Write:								

$0170 - $023F Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0170 - $023F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0240 - $026F Port Integration Module (PIM) (Sheet 1 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0240	PTT	Read:	PTT7	PTT6	PTT5	PTT4	0	0	0	0
		Write:								
$0241	PTIT	Read:	PTIT7	PTIT6	PTIT5	PTIT4	0	0	0	0
		Write:								
$0242	DDRT	Read:	DDRT7	DDRT6	DDRT5	DDRT4	0	0	0	0
		Write:								
$0243	RDRT	Read:	RDRT7	RDRT6	RDRT5	RDRT4	0	0	0	0
		Write:								
$0244	PERT	Read:	PERT7	PERT6	PERT5	PERT4	0	0	0	0
		Write:								
$0245	PPST	Read:	PPST7	PPST6	PPST5	PPST4	0	0	0	0
		Write:								
$0246	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0247	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0248	PTS	Read:	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
		Write:								
$0249	PTIS	Read:	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
		Write:								
$024A	DDRS	Read:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		Write:								
$024B	RDRS	Read:	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
		Write:								
$024C	PERS	Read:	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
		Write:								
$024D	PPSS	Read:	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
		Write:								

$0240 - $026F Port Integration Module (PIM) (Sheet 2 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$024E	WOMS	Read:	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
		Write:								
$024F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0250	PTG	Read:	PTG7	PTG6	PTG5	PTG4	PTG3	PTG2	PTG1	PTG0
		Write:								
$0251	PTIG	Read:	PTIG7	PTIG6	PTIG5	PTIG4	PTIG3	PTIG2	PTIG1	PTIG0
		Write:								
$0252	DDRG	Read:	DDRG7	DDRG6	DDRG5	DDRG4	DDRG3	DDRG2	DDRG1	DDRG0
		Write:								
$0253	RDRG	Read:	RDRG7	RDRG6	RDRG5	RDRG4	RDRG3	RDRG2	RDRG1	RDRG0
		Write:								
$0254	PERG	Read:	PERG7	PERG6	PERG5	PERG4	PERG3	PERG2	PERG1	PERG0
		Write:								
$0255	PPSG	Read:	PPSG7	PPSG6	PPSG5	PPSG4	PPSG3	PPSG2	PPSG1	PPSG0
		Write:								
$0256	PIEG	Read:	PIEG7	PIEG6	PIEG5	PIEG4	PIEG3	PIEG2	PIEG1	PIEG0
		Write:								
$0257	PIFG	Read:	PIFG7	PIFG6	PIFG5	PIFG4	PIFG3	PIFG2	PIFG1	PIFG0
		Write:								
$0258	PTH	Read:	0	PTH6	PTH5	PTH4	PTH3	PTH2	PTH1	PTH0
		Write:								
$0259	PTIH	Read:	0	PTIH6	PTIH5	PTIH4	PTIH3	PTIH2	PTIH1	PTIH0
		Write:								
$025A	DDRH	Read:	0	DDRH6	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
		Write:								
$025B	RDRH	Read:	0	RDRH6	RDRH5	RDRH4	RDRH3	RDRH2	RDRH1	RDRH0
		Write:								
$025C	PERH	Read:	0	PERH6	PERH5	PERH4	PERH3	PERH2	PERH1	PERH0
		Write:								
$025D	PPSH	Read:	0	PPSH6	PPSH5	PPSH4	PPSH3	PPSH2	PPSH1	PPSH0
		Write:								
$025E	PIEH	Read:	0	PIEH6	PIEH5	PIEH4	PIEH3	PIEH2	PIEH1	PIEH0
		Write:								
$025F	PIFH	Read:	0	PIFH6	PIFH5	PIFH4	PIFH3	PIFH2	PIFH1	PIFH0
		Write:								
$0260	PTJ	Read:	PTJ7	PTJ6	0	0	PTJ3	PTJ2	PTJ1	PTJ0
		Write:								
$0262	PTIJ	Read:	PTIJ7	PTIJ6	0	0	PTIJ3	PTIJ2	PTIJ1	PTIJ0
		Write:								
$0262	DDRJ	Read:	DDRJ7	DDRJ6	0	0	DDRJ3	DDRJ2	DDRJ1	DDRJ0
		Write:								
$0263	RDRJ	Read:	RDRJ7	RDRJ6	0	0	RDRJ3	RDRJ2	RDRJ1	RDRJ0
		Write:								
$0264	PERJ	Read:	PERJ7	PERJ6	0	0	PERJ3	PERJ2	PERJ1	PERJ0
		Write:								

$0240 - $026F Port Integration Module (PIM) (Sheet 3 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0265	PPSJ	Read:	PPSJ7	PPSJ6	0	0	PPSJ3	PPSJ2	PPSJ1	PPSJ0
		Write:								
$0266	PIEJ	Read:	PIEJ7	PIEJ6	0	0	PIEJ3	PIEJ2	PIEJ1	PIEJ0
		Write:								
$0267	PIFJ	Read:	PIFJ7	PIFJ6	0	0	PIFJ3	PIFJ2	PIFJ1	PIFJ0
		Write:								
$0268	PTL	Read:	0	PTL6	PTL5	PTL4	PTL3	PTL2	PTL1	PTL0
		Write:								
$0269	PTIL	Read:	0	PTIL6	PTIL5	PTIL4	PTIL3	PTIL2	PTIL1	PTIL0
		Write:								
$026A	DDRL	Read:	0	DDRL6	DDRL5	DDRL4	DDRL3	DDRL2	DDRL1	DDRL0
		Write:								
$026B	RDRL	Read:	0	RDRL6	RDRL5	RDRL4	RDRL3	RDRL2	RDRL1	RDRL0
		Write:								
$026C	PERL	Read:	0	PERL6	PERL5	PERL4	PERL3	PERL2	PERL1	PERL0
		Write:								
$026D	PPSL	Read:	0	PPSL6	PPSL5	PPSL4	PPSL3	PPSL2	PPSL1	PPSL0
		Write:								
$026E	WOML	Read:	0	WOML6	WOML5	WOML4	WOML3	WOML2	WOML1	WOML0
		Write:								
$026F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

$0270 - $03FF Reserved Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0270 – $3FF	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

1.1.6 Part ID Assignments

The part ID is located in two 8-bit registers PARTIDH and PARTIDL (addresses $001A and $001B after reset). The read-only value is a unique part ID for each revision of the MCU. Table 1-2 shows the assigned part ID number.

Table 1-2. Assigned Part ID Numbers

Device	Mask Set Number	Part ID[1]
MC9S12NE64	0L19S	$8200
MC9S12NE64	1L19S	$8201

[1] The coding is as follows:
Bit 15-12: Major family identifier
Bit 11-8: Minor family identifier
Bit 7-4: Major mask set revision number including FAB transfers
Bit 3-0: Minor (or non full) mask set revision

The PRTIDH register is constructed of four hexadecimal digits (0xABCD) as follows:

Digit “A” = Family ID
Digit “B” = Memory ID (flash size)
Digit “C” = Major mask revision
Digit “D” = Minor mask revision

Currently, family IDs are:

0x0 = D family
0x1 = H family
0x2 = B family
0x3 = C family
0x4 = T family
0x5 = E family
0x6 = reserved
0x7 = reserved
0x8 = NE family

Current memory IDs are:

0x0 = 256K
0x1 = 128K
0x2 = 64K
0x3 = 32K
0x4 = 512K

The major and minor mask revision increments from 0x0 as follows:

- Major mask increments on a complete (full/all layer) mask change.
- Minor mask increments on a single or smaller than full mask change.

The device memory sizes are located in two 8-bit registers MEMSIZ0 and MEMSIZ1 (addresses $001C and $001D after reset). Table 1-3 shows the read-only values of these registers. See the module mapping and control (MMC) block description chapter for further details.

Table 1-3. Memory Size Registers

	Register Name	Value
MC9S12NE64	MEMSIZ0	$03
MC9S12NE64	MEMSIZ1	$80

1.2 Signal Description

This section describes signals that connect off-chip. It includes a pinout diagram, a table of signal properties, and detailed discussion of signals.

1.2.1 Device Pinout

The MC9S12NE64 is available in a 112-pin low-profile quad flat pack (LQFP) and in an 80-pin quad flat pack (TQFP-EP). Most pins perform two or more functions, as described in this section. Figure 1-3 and Figure 1-4 show the pin assignments.

1.2.1.1 112-Pin LQFP

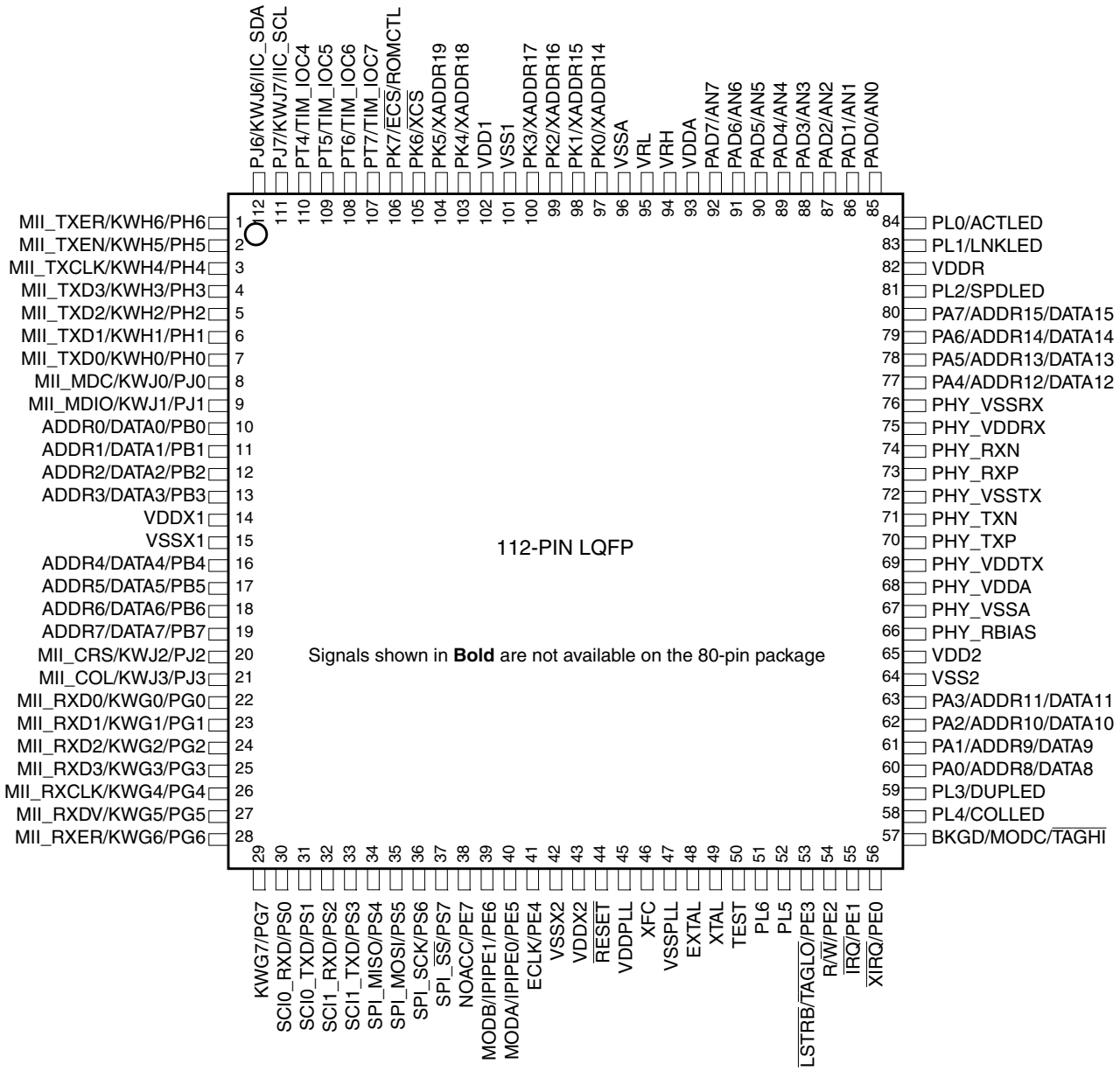


Figure 1-3. Pin Assignments in 112-Pin LQFP for MC9S12NE64

1.2.1.2 80-Pin TQFP-EP

The MEBI is not available in the 80-pin package. The 80-pin package features an exposed tab that is used for enhanced thermal management. The exposed tab requires special PCB layout considerations as described in Appendix B, “Schematic and PCB Layout Design Recommendations.”

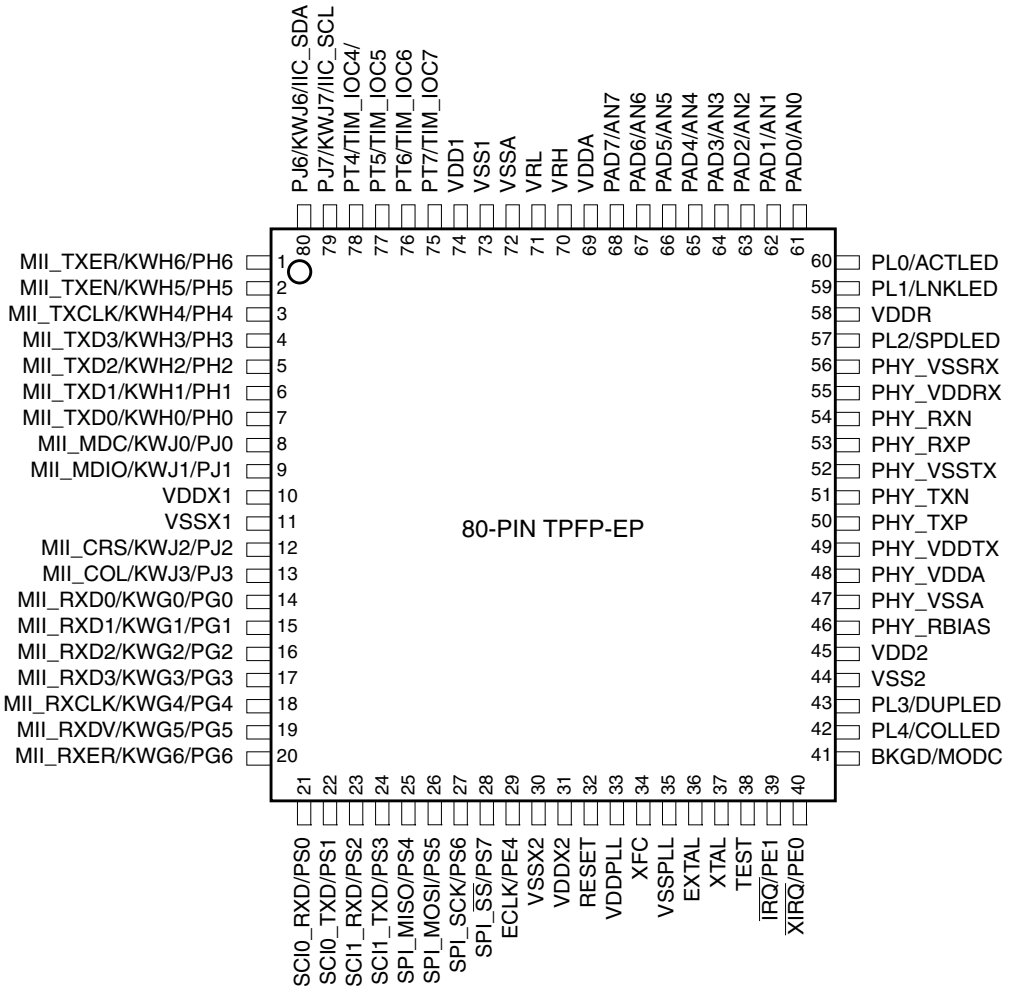


Figure 1-4. Pin Assignments in 80-Pin TQFP-EP for MC9S12NE64

1.2.2 Signal Properties Summary

Table 1-4. Signal Properties (Sheet 1 of 4)

80 Pin No.	112 Pin No.	Pin Name Function 1	Pin Name Function 2	Pin Name Function 3	Power Domain	Internal Pull Resistor		Description	Reset State
						CTRL	Reset State		
1	1	PH6	KWH6	MII_TXER	VDDX	PERH/ PPSH	Disabled	Port H I/O pin; EMAC MII transmit error; interrupt	Input
2	2	PH5	KWH5	MII_TXEN	VDDX	PERH/ PPSH	Disabled	Port H I/O pin; EMAC MII transmit enable; interrupt	Input
3	3	PH4	KWH4	MII_TXCLK	VDDX	PERH/ PPSH	Disabled	Port H I/O pin; EMAC MII transmit clock; interrupt	Input
4	4	PH3	KWH3	MII_TXD3	VDDX	PERH/ PPSH	Disabled	Port H I/O pin; EMAC MII transmit data; interrupt	Input
5	5	PH2	KWH2	MII_TXD2	VDDX	PERH/ PPSH	Disabled	Port H I/O pin; EMAC MII transmit data; interrupt	Input
6	6	PH1	KWH1	MII_TXD1	VDDX	PERH/ PPSH	Disabled	Port H I/O pin; EMAC MII transmit data; interrupt	Input
7	7	PH0	KWH0	MII_TXD0	VDDX	PERH/ PPSH	Disabled	Port H I/O pin; EMAC MII transmit data; interrupt	Input
8	8	PJ0	KWJ0	MII_MDC	VDDX	PERJ/ PPSJ	Disabled	Port J I/O pin; EMAC MII management data clock; interrupt	Input
9	9	PJ1	KWJ1	MII_MDIO	VDDX	PERJ/ PPSJ	Disabled	Port J I/O pin; EMAC MII management data I/O; interrupt	Input
—	**10–13 16–19**	**PB[7:0]**	**ADDR[7:0] / DATA[7:0]**	**—**	**VDDX**	**PUCR**	**Disabled**	**Port B I/O pin; multiplexed address/data**	**Input**
10	14	VDDX1	—	—				See Table 1-5	
11	15	VSSX1	—	—				See Table 1-5	
12	20	PJ2	KWJ2	MII_CRS	VDDX	PERJ/ PPSJ	Disabled	Port J I/O pin; EMAC MII carrier sense; interrupt	Input
13	21	PJ3	KWJ3	MII_COL	VDDX	PERJ/ PPSJ	Disabled	Port J I/O pin; EMAC MII collision; interrupt	Input
14	22	PG0	KWG0	MII_RXD0	VDDX	PERG/ PPSG	Disabled	Port G I/O pin; EMAC MII receive data; interrupt	Input
15	23	PG1	KWG1	MII_RXD1	VDDX	PERG/ PPSG	Disabled	Port G I/O pin; EMAC MII receive data; interrupt	Input

Table 1-4. Signal Properties (Sheet 2 of 4)

80 Pin No.	112 Pin No.	Pin Name Function 1	Pin Name Function 2	Pin Name Function 3	Power Domain	Internal Pull Resistor		Description	Reset State
						CTRL	Reset State		
16	24	PG2	KWG2	MII_RXD2	VDDX	PERG/ PPSG	Disabled	Port G I/O pin; EMAC MII receive data; interrupt	Input
17	25	PG3	KWG3	MII_RXD3	VDDX	PERG/ PPSG	Disabled	Port G I/O pin; EMAC MII receive data; interrupt	Input
18	26	PG4	KWG4	MII_RXCLK	VDDX	PERG/ PPSG	Disabled	Port G I/O pin; EMAC MII receive clock; interrupt	Input
19	27	PG5	KWG5	MII_RXDV	VDDX	PERG/ PPSG	Disabled	Port G I/O pin; EMAC MII receive data valid; interrupt	Input
20	28	PG6	KWG6	MII_RXER	VDDX	PERG/ PPSG	Disabled	Port G I/O pin; EMAC MII receive error; interrupt	Input
—	**29**	**PG7**	**KWG7**	**—**	**VDDX**	**PERG/ PPSG**	**Disabled**	**Port G I/O pin; interrupt**	**Input**
21	30	PS0	SCI0_RXD	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SCI0 receive signal	Input
22	31	PS1	SCI0_TXD	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SCI0 transmit signal	Input
23	32	PS2	SCI1_RXD	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SCI1 receive signal	Input
24	33	PS3	SCI1_TXD	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SCI1 transmit signal	Input
25	34	PS4	SPI_MISO	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SPI MISO signal	Input
26	35	PS5	SPI_MOSI	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SPI MOSI signal	Input
27	36	PS6	SPI_SCK	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SPI SCK signal	Input
28	37	PS7	SPI_$\overline{SS}$	—	VDDX	PERS/ PPSS	Disabled	Port S I/O pin; SPI $\overline{SS}$ signal	Input
—	**38**	**PE7**	**NOACC**	**—**	**VDDX**	**PUCR**	**Up**	**Port E I/O pin; access**	**Input**
—	**39**	**PE6**	**IPIPE1**	**MODB**	**VDDX**	**While $\overline{RESET}$ pin is low: Down**		**Port E I/O pin; pipe status; mode selection**	**Input**
—	**40**	**PE5**	**IPIPE0**	**MODA**	**VDDX**	**While $\overline{RESET}$ pin is low: Down**		**Port E I/O pin; pipe status; mode selection**	**Input**
29	41	PE4	ECLK	—	VDDX	PUCR	Up	Port E I/O pin; bus clock output	Input
30	42	VSSX2	—	—				See Table 1-5	
31	43	VDDX2	—	—				See Table 1-5	

Table 1-4. Signal Properties (Sheet 3 of 4)

80 Pin No.	112 Pin No.	Pin Name Function 1	Pin Name Function 2	Pin Name Function 3	Power Domain	Internal Pull Resistor		Description	Reset State
						CTRL	Reset State		
32	44	$\overline{\text{RESET}}$	—	—	VDDX	None	None	External reset pin	Input
33	45	VDDPLL	—	—				See Table 1-5	
34	46	XFC	—	—	VDDPLL	NA	NA	PLL filter pin	
35	47	VSSPLL	—	—				See Table 1-5	
36	48	EXTAL	—	—	VDDPLL	NA	NA	Oscillator pins	Input
37	49	XTAL	—	—	VDDPLL	NA	NA		Output
38	50	TEST	—	—	VDDX	None	None	Must be grounded	Input
—	**51**	**PL6**	**—**	**—**	**VDDX**	**PERL/ PPSL**	**Disabled**	**Port L I/O pin**	**Input**
—	**52**	**PL5**	**—**	**—**	**VDDX**	**PERL/ PPSL**	**Disabled**	**Port L I/O pin**	**Input**
—	**53**	**PE3**	**$\overline{\text{TAGLO}}$**	**$\overline{\text{LSTRB}}$**	**VDDX**	**PUCR**	**Up**	**Port E I/O pin; low strobe; tag signal low**	**Input**
—	**54**	**PE2**	**R/$\overline{\text{W}}$**	**—**	**VDDX**	**PUCR**	**Up**	**Port E I/O pin; R/$\overline{\text{W}}$ in expanded modes**	**Input**
39	55	PE1	$\overline{\text{IRQ}}$	—	VDDX	PUCR	Up	Port E input; external interrupt pin	Input
40	56	PE0	$\overline{\text{XIRQ}}$	—	VDDX	PUCR	Up	Port E input; non-maskable interrupt pin	Input
41	57	BKGD	MODC	$\overline{\text{TAGHI}}$	VDDX	None	Up	Background debug; mode pin; tag signal high	Input
42	58	PL4	COLLED	—	VDDX	PERL/ PPSL	Disabled	Port L I/O pin; EPHY collision LED	Input
43	59	PL3	DUPLED	—	VDDX	PERL/ PPSL	Disabled	Port L I/O pin; EPHY full duplex LED	Input
—	**60–63 77–80**	**PA[7:0]**	**ADDR[15:8]/ DATA[15:8]**	**—**	**VDDX**	**PUCR**	**Disabled**	**Port A I/O pin; multiplexed address/data**	**Input**
44	64	VSS2	—	—				See Table 1-5	
45	65	VDD2	—	—				See Table 1-5	
46	66	PHY_RBIAS	—	—	PHY_ VSSA	NA	NA	Bias control:1.0% external resistor (see the Electricals Chapter for R_{Bias})	Analog Input
47	67	PHY_VSSA	—	—				See Table 1-5	
48	68	PHY_VDDA	—	—				See Table 1-5	
49	69	PHY_VDDTX	—	—				See Table 1-5	
50	70	PHY_TXP	—	—	PHY_ VDDTX	NA	NA	Twisted pair output +	Analog Output

Table 1-4. Signal Properties (Sheet 4 of 4)

80 Pin No.	112 Pin No.	Pin Name Function 1	Pin Name Function 2	Pin Name Function 3	Power Domain	Internal Pull Resistor		Description	Reset State
						CTRL	Reset State		
51	71	PHY_TXN	—	—	PHY_ VDDTX	NA	NA	Twisted pair output –	Analog Output
52	72	PHY_VSSTX	—	—				See Table 1-5	
53	73	PHY_RXP	—	—	PHY_ VDDRX	NA	NA	Twisted pair input +	Analog Input
54	74	PHY_RXN	—	—	PHY_ VDDRX	NA	NA	Twisted pair input –	Analog Input
55	75	PHY_VDDRX	—	—				See Table 1-5	
56	76	PHY_VSSRX	—	—				See Table 1-5	
57	81	PL2	SPDLED	—	VDDX	PERL/ PPSL	Disabled	Port L I/O pin; EPHY 100 Mbps LED	Input
58	82	VDDR/ VREGEN	—	—				See Table 1-5	
59	83	PL1	LNKLED	—	VDDX	PERL/ PPSL	Disabled	Port L I/O pin; EPHY valid link LED	Input
60	84	PL0	ACTLED	—	VDDX	PERL/ PPSL	Disabled	Port L I/O pin; EPHY transmit or receive LED	Input
61–68	85–92	PAD[7:0]	AN[7:0]	—	VDDA	None	None	Port AD input pins; ATD inputs	Input
69	93	VDDA	—	—				See Table 1-5	
70	94	VRH	—	—				See Table 1-5	
71	95	VRL	—	—				See Table 1-5	
72	96	VSSA	—	—				See Table 1-5	
—	**97–100 103–104**	**PK[5:0]**	**XADDR [19:14]**	**—**	**VDDX**	**PUCR**	**Up**	**Port K I/O pins; extended addresses**	**Input**
73	101	VSS1	—	—				See Table 1-5	
74	102	VDD1	—	—				See Table 1-5	
—	**105**	**PK[6]**	**$\overline{\text{XCS}}$**	**—**	**VDDX**	**PUCR**	**Up**	**Port K I/O pin; external chip select**	**Input**
—	**106**	**PK[7]**	**$\overline{\text{ECS}}$**	**ROMCTL**	**VDDX**	**PUCR**	**Up**	**Port K I/O pin; emulation chip select;**	**Input**
75–78	107–110	PT[7:4]	TIM_IOC [7:4]	—	VDDX	PERT/ PPST	Disabled	Port T I/O pins; timer TIM input cap. output compare	Input
79	111	PJ7	KWJ7	IIC_SCL	VDDX	PERJ/ PPSJ	Disabled	Port J I/O pin; IIC SCL; interrupt	Input
80	112	PJ6	KWJ6	IIC_SDA	VDDX	PERJ/ PPSJ	Disabled	Port J I/O pin; IIC SDA; interrupt	Input

NOTE

Signals shown in bold are not available in the 80-pin package.

NOTE

If the port pins are not bonded out in the chosen package, the user must initialize the registers to be inputs with enabled pull resistance to avoid excess current consumption. This applies to the following pins:

(80-Pin TQFP-EP): Port A[7:0], Port B[7:0], Port E[7,6,5,3,2], Port K[7:0]; Port G[7]; Port L[6:5]

1.2.3 Detailed Signal Descriptions

1.2.3.1 EXTAL, XTAL — Oscillator Pins

EXTAL and XTAL are the external clock and crystal driver pins. Upon reset, all the device clocks are derived from the EXTAL input frequency. XTAL is the crystal output.

1.2.3.2 $\overline{\text{RESET}}$ — External Reset Pin

$\overline{\text{RESET}}$ is an active-low bidirectional control signal that acts as an input to initialize the MCU to a known start-up state. It also acts as an open-drain output to indicate that an internal failure has been detected in either the clock monitor or COP watchdog circuit. External circuitry connected to the $\overline{\text{RESET}}$ pin must not include a large capacitance that would interfere with the ability of this signal to rise to a valid logic one within 32 ECLK cycles after the low drive is released. Upon detection of any reset, an internal circuit drives the $\overline{\text{RESET}}$ pin low and a clocked reset sequence controls when the MCU can begin normal processing. The $\overline{\text{RESET}}$ pin includes an internal pull-up device.

1.2.3.3 XFC — PLL Loop Filter Pin

Dedicated pin used to create the PLL filter. See A.12.3.1, “XFC Component Selection,” and the CRG block description chapter for more detailed information.

1.2.3.4 BKGD / MODC / $\overline{\text{TAGHI}}$ — Background Debug / Tag High / Mode Pin

The BKGD / MODC / $\overline{\text{TAGHI}}$ pin is used as a pseudo-open-drain pin for background debug communication. It is used as an MCU operating mode select pin during reset. The state of this pin is latched to the MODC bit at the rising edge of $\overline{\text{RESET}}$. In MCU expanded modes of operation, while instruction tagging is on, an input low on this pin during the falling edge of E-clock tags the high half of the instruction word being read into the instruction queue. This pin always has an internal pull-up.

1.2.3.5 PA[7:0] / ADDR[15:8] / DATA[15:8] — Port A I/O Pins

PA[7:0] are general-purpose I/O pins. In MCU expanded modes of operation, these pins are used for the multiplexed external address and data bus. PA[7:0] pins are not available in the 80-pin package version.

1.2.3.6 PB[7:0] / ADDR[7:0] / DATA[7:0] — Port B I/O Pins

PB[7:0] are general-purpose I/O pins. In MCU expanded modes of operation, these pins are used for the multiplexed external address and data bus. PB[7:0] pins are not available in the 80-pin package version.

1.2.3.7 PE7 / NOACC — Port E I/O Pin 7

PE7 is a general-purpose I/O pin. During MCU expanded modes of operation, the NOACC signal, while enabled, is used to indicate that the current bus cycle is an unused or free cycle. This signal will assert when the CPU is not using the bus.

1.2.3.8 PE6 / IPIPE1/ MODB — Port E I/O Pin 6

PE6 is a general-purpose I/O pin. It is used as an MCU operating mode select pin during reset. The state of this pin is latched to the MODB bit at the rising edge of $\overline{\text{RESET}}$. This pin is shared with the instruction queue tracking signal IPIPE1. PE6 is an input with a pulldown device that is active only while $\overline{\text{RESET}}$ is low. PE6 is not available in the 80-pin package version.

1.2.3.9 PE5 / IPIPE0 / MODA — Port E I/O Pin 5

PE5 is a general-purpose I/O pin. It is used as an MCU operating mode select pin during reset. The state of this pin is latched to the MODA bit at the rising edge of $\overline{\text{RESET}}$. This pin is shared with the instruction queue tracking signal IPIPE0. This pin is an input with a pull-down device that is only active while $\overline{\text{RESET}}$ is low. PE5 is not available in the 80-pin package version.

1.2.3.10 PE4 / ECLK— Port E I/O Pin 4 / E-Clock Output

PE4 is a general-purpose I/O pin. In normal single chip mode, PE4 is configured with an active pull-up while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPEE in the PUCR register. In all modes except normal single chip mode, the PE4 pin is initially configured as the output connection for the internal bus clock (ECLK). ECLK is used as a timing reference and to demultiplex the address and data in expanded modes. The ECLK frequency is equal to 1/2 the crystal frequency out of reset. The ECLK output function depends upon the settings of the NECLK bit in the PEAR register, the IVIS bit in the MODE register, and the ESTR bit in the EBICTL register. All clocks, including the ECLK, are halted while the MCU is in stop mode. It is possible to configure the MCU to interface to slow external memory. ECLK can be stretched for such accesses. The PE4 pin is initially configured as ECLK output with stretch in all expanded modes. See the MISC register (EXSTR[1:0] bits) for more information. In normal expanded narrow mode, the ECLK is available for use in external select decode logic or as a constant speed clock for use in the external application system.

1.2.3.11 PE3 / $\overline{\text{TAGLO}}$ / $\overline{\text{LSTRB}}$ — Port E I/O Pin 3 / Low-Byte Strobe ($\overline{\text{LSTRB}}$)

PE3 can be used as a general-purpose I/O in all modes and is an input with an active pull-up out of reset. The pull-up can be turned off by clearing PUPEE in the PUCR register. PE3 can also be configured as a Low-Byte Strobe ($\overline{\text{LSTRB}}$). The $\overline{\text{LSTRB}}$ signal is used in write operations, so external low byte writes will not be possible until this function is enabled. $\overline{\text{LSTRB}}$ can be enabled by setting the LSTRE bit in the PEAR register. In expanded wide and emulation narrow modes, and while BDM tagging is enabled, the $\overline{\text{LSTRB}}$ function is multiplexed with the $\overline{\text{TAGLO}}$ function. While enabled, a logic zero on the $\overline{\text{TAGLO}}$ pin at the falling edge of ECLK will tag the low byte of an instruction word being read into the instruction queue. PE3 is not available in the 80-pin package version.

1.2.3.12 PE2 / R/$\overline{\text{W}}$ — Port E I/O Pin 2 / Read/$\overline{\text{Write}}$

PE2 can be used as a general-purpose I/O in all modes and is configured as an input with an active pull-up out of reset. The pull-up can be turned off by clearing PUPEE in the PUCR register. If the read/write function is required, it must be enabled by setting the RDWE bit in the PEAR register. External writes will not be possible until the read/write function is enabled. The PE2 pin is not available in the 80-pin package version.

1.2.3.13 PE1 / $\overline{\text{IRQ}}$ — Port E Input Pin 1 / Maskable Interrupt Pin

PE1 is always an input and can be read anytime. The PE1 pin is also the $\overline{\text{IRQ}}$ input used for requesting an asynchronous interrupt to the MCU. During reset, the I bit in the condition code register (CCR) is set and any $\overline{\text{IRQ}}$ interrupt is masked until the I bit is cleared. The $\overline{\text{IRQ}}$ is software programmable to either falling-edge-sensitive triggering or level-sensitive triggering based on the setting of the IRQE bit in the IRQCR register. The $\overline{\text{IRQ}}$ is always enabled and configured to level-sensitive triggering out of reset. It can be disabled by clearing IRQEN bit in the IRQCR register. There is an active pull-up on this pin while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPEE in the PUCR register.

1.2.3.14 PE0 / $\overline{\text{XIRQ}}$ — Port E input Pin 0 / Non-Maskable Interrupt Pin

PE0 is always an input and can be read anytime. The PE0 pin is also the $\overline{\text{XIRQ}}$ input for requesting a non-maskable asynchronous interrupt to the MCU. During reset, the X bit in the condition code register (CCR) is set and any $\overline{\text{XIRQ}}$ interrupt is masked until the X bit is cleared. Because the $\overline{\text{XIRQ}}$ input is level sensitive triggered, it can be connected to a multiple-source wired-OR network. There is an active pull-up on this pin while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPEE in the PUCR register.

1.2.3.15 PK7 / $\overline{\text{ECS}}$ / ROMCTL — Port K I/O Pin 7

PK7 is a general-purpose I/O pin. During MCU expanded modes of operation, while the EMK bit in the MODE register is set to 1, this pin is used as the emulation chip select output ($\overline{\text{ECS}}$). In expanded modes, the PK7 pin can be used to determine the reset state of the ROMON bit in the MISC register. At the rising edge of $\overline{\text{RESET}}$, the state of the PK7 pin is latched to the ROMON bit. There is an active pull-up on this pin while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPKE in the PUCR register. PK7 is not available in the 80-pin package version.

1.2.3.16 PK6 / $\overline{\text{XCS}}$ — Port K I/O Pin 6

PK6 is a general-purpose I/O pin. During MCU expanded modes of operation, while the EMK bit in the MODE register is set to 1, this pin is used as an external chip select signal for most external accesses that are not selected by ECS. There is an active pull-up on this pin while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPKE in the PUCR register. See the multiplexed external bus interface (MEBI) block description chapter for further details. PK6 is not available in the 80-pin package version.

1.2.3.17 PK[5:0] / XADDR[19:14] — Port K I/O Pins [5:0]

PK[5:0] are general-purpose I/O pins. In MCU expanded modes of operation, when the EMK bit in the MODE register is set to 1, PK[5:0] provide the expanded address XADDR[19:14] for the external bus. There are active pull-ups on PK[5:0] pins while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPKE in the PUCR register. See multiplexed external bus interface (MEBI) block description chapter for further details. PK[5:0] are not available in the 80-pin package version.

1.2.3.18 PAD[7:0] / AN[7:0] — Port AD Input Pins [7:0]

PAD[7:0] are the analog inputs for the analog-to-digital converter (ATD). They can also be configured as general-purpose digital input. See the port integration module (PIM) PIM_9NE64 block description chapter and the ATD_10B8C block description chapter for information about pin configurations.

1.2.3.19 PG7 / KWG7 — Port G I/O Pin 7

PG7 is a general-purpose I/O pin. It can be configured to generate an interrupt (KWG7) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG7 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter for information about pin configurations.

1.2.3.20 PG6 / KWG6 / MII_RXER — Port G I/O Pin 6

PG6 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the receive error (MII_RXER) signal. It can be configured to generate an interrupt (KWG6) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG6 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.21 PG5 / KWG5 / MII_RXDV — Port G I/O Pin 5

PG5 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the receive data valid (MII_RXDV) signal. It can be configured to generate an interrupt (KWG5) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG5 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.22 PG4 / KWG4 / MII_RXCLK — Port G I/O Pin 4

PG4 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the receive clock (MII_RXCLK) signal. It can be configured to generate an interrupt (KWG4) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG4 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.23 PG3 / KWG3 / MII_RXD3 — Port G I/O Pin 3

PG3 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the receive data (MII_RXD3) signal. It can be configured to generate an interrupt (KWG3) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG3 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.24 PG2 / KWG2 / MII_RXD2 — Port G I/O Pin 2

PG2 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the receive data (MII_RXD2) signal. It can be configured to generate an interrupt (KWG2) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG2 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.25 PG1 / KWG1 / MII_RXD1 — Port G I/O Pin 1

PG1 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the receive data (MII_RXD1) signal. It can be configured to generate an interrupt (KWG1) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG1 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.26 PG0 / KWG0 / MII_RXD0 — Port G I/O Pin 0

PG0 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the receive data (MII_RXD0) signal. It can be configured to generate an interrupt (KWG0) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PG0 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.27 PH6 / KWH6 / MII_TXER — Port H I/O Pin 6

PH6 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the transmit error (MII_TXER) signal. It can be configured to generate an interrupt (KWH6) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PH6 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.28 PH5 / KWH5 / MII_TXEN — Port H I/O Pin 5

PH5 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the transmit enabled (MII_TXEN) signal. It can be configured to generate an interrupt (KWH5) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PH5 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.29 PH4 / KWH4 / MII_TXCLK — Port H I/O Pin 4

PH4 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the transmit Clock (MII_TXCLK) signal. It can be configured to generate an interrupt (KWH4) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PH4 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.30 PH3 / KWH3 / MII_TXD3 — Port H I/O Pin 3

PH3 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the transmit data (MII_TXD3) signal. It can be configured to generate an interrupt (KWH3) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PH3 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.31 PH2 / KWH2 / MII_TXD2 — Port H I/O Pin 2

PH2 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the transmit data (MII_TXD2) signal. It can be configured to generate an interrupt (KWH2) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PH2 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.32 PH1 / KWH1 / MII_TXD1 — Port H I/O Pin 1

PH1 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the transmit data (MII_TXD1) signal. It can be configured to generate an interrupt (KWH1) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PH1 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.33 PH0 / KWH0 / MII_TXD0 — Port H I/O Pin 0

PH0 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the transmit data (MII_TXD0) signal. It can be configured to generate an interrupt (KWH0) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PH0 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.34 PJ7 / KWJ7 / IIC_SCL — Port J I/O Pin 7

PJ7 is a general-purpose I/O pin. When the IIC module is enabled, it becomes the serial clock line (IIC_SCL) for the IIC module (IIC). It can be configured to generate an interrupt (KWJ7) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PJ7 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the IIC block description chapter for information about pin configurations.

1.2.3.35 PJ6 / KWJ6 / IIC_SDA — Port J I/O Pin 6

PJ6 is a general-purpose I/O pin. When the IIC module is enabled, it becomes the serial data line (IIC_SDL) for the IIC module (IIC). It can be configured to generate an interrupt (KWJ6) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PJ6 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the IIC block description chapter for information about pin configurations.

1.2.3.36 PJ3 / KWJ3 / MII_COL — Port J I/O Pin 3

PJ3 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the collision (MII_COL) signal. It can be configured to generate an interrupt (KWJ3) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PJ3 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.37 PJ2 / KWJ2 / MII_CRS /— Port J I/O Pin 2

PJ2 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the carrier sense (MII_CRS) signal. It can be configured to generate an interrupt (KWJ2) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PJ2 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.38 PJ1 / KWJ1 / MII_MDIO — Port J I/O Pin 1

PJ1 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the Management Data I/O (MII_MDIO) signal. It can be configured to generate an interrupt (KWH1) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PJ1 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.39 PJ0 / KWJ0 / MII_MDC — Port J I/O Pin 0

PJ0 is a general-purpose I/O pin. When the EMAC MII external interface is enabled, it becomes the management data clock (MII_MDC) signal. It can be configured to generate an interrupt (KWJ0) causing the MCU to exit stop or wait mode. While in reset and immediately out of reset, the PJ0 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EMAC block description chapter for information about pin configurations.

1.2.3.40 PL6 — Port L I/O Pin 6

PL6 is a general-purpose I/O pin. While in reset and immediately out of reset, the PL6 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter for information about pin configurations.

1.2.3.41 PL5 — Port L I/O Pin 5

PL5 is a general-purpose I/O pin. While in reset and immediately out of reset, the PL5 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter for information about pin configurations.

1.2.3.42 PL4 / COLLED — Port L I/O Pin 4

PL4 is a general-purpose I/O pin. When the internal Ethernet physical transceiver (EPHY) is enabled with the EPHYCTL0 LEDEN bit set, it becomes the collision status signal (COLLED). While in reset and immediately out of reset the PL4 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EPHY block description chapter for information about pin configurations.

1.2.3.43 PL3 / DUPLED — Port L I/O Pin 3

PL3 is a general-purpose I/O pin. When the internal Ethernet physical transceiver (EPHY) is enabled with the EPHYCTL0 LEDEN bit set, it becomes the duplex status signal (DUPLED). While in reset and immediately out of reset, the PL3 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EPHY block description chapter for information about pin configurations.

1.2.3.44 PL2 / SPDLED — Port L I/O Pin 2

PL2 is a general-purpose I/O pin. When the internal Ethernet physical transceiver (EPHY) is enabled with the EPHYCTL0 LEDEN bit set, it becomes the speed status signal (SPDLED). While in reset and immediately out of reset, the PL2 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EPHY block description chapter for information about pin configurations.

1.2.3.45 PL1 / LNKLED — Port L I/O Pin 1

PL1 is a general-purpose I/O pin. When the internal Ethernet physical transceiver (EPHY) is enabled with the EPHYCTL0 LEDEN bit set, it becomes the link status signal (LNKLED). While in reset and immediately out of reset, the PL1 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EPHY block description chapter for information about pin configurations.

1.2.3.46 PL0 / ACTLED — Port L I/O Pin 0

PL0 is a general-purpose I/O pin. When the internal Ethernet physical transceiver (EPHY) is enabled with the EPHYCTL0 LEDEN bit set, it becomes the active status signal (ACTLED). While in reset and immediately out of reset, the PL0 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the EPHY block description chapter for information about pin configurations.

1.2.3.47 PS7 / SPI_$\overline{SS}$ — Port S I/O Pin 7

PS7 is a general-purpose I/O. When the serial peripheral interface (SPI) is enabled, PS7 becomes the slave select pin $\overline{SS}$. While in reset and immediately out of reset, the PS7 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SPI block description chapter for information about pin configurations.

1.2.3.48 PS6 / SPI_SCK — Port S I/O Pin 6

PS6 is a general-purpose I/O pin. When the serial peripheral interface (SPI) is enabled, PS6 becomes the serial clock pin, SCK. While in reset and immediately out of reset, the PS6 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SPI block description chapter for information about pin configurations.

1.2.3.49 PS5 / SPI_MOSI — Port S I/O Pin 5

PS5 is a general-purpose I/O pin. When the serial peripheral interface (SPI) is enabled, PS5 becomes the master output (during master mode) or slave input (during slave mode) pin. While in reset and immediately out of reset, the PS5 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SPI block description chapter for information about pin configurations.

1.2.3.50 PS4 / SPI_MISO — Port S I/O Pin 4

PS4 is a general-purpose I/O pin. When the serial peripheral interface (SPI) is enabled, PS4 becomes the master input (during master mode) or slave output (during slave mode) pin. While in reset and immediately out of reset, the PS4 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SPI block description chapter for information about pin configurations.

1.2.3.51 PS3 / SCI1_TXD — Port S I/O Pin 3

PS3 is a general-purpose I/O. When the serial communications interface 1 (SCI1) transmitter is enabled, PS3 becomes the transmit pin, TXD, of SCI1. While in reset and immediately out of reset, the PS3 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SCI block description chapter for information about pin configurations.

1.2.3.52 PS2 / SCI1_RXD — Port S I/O Pin 2

PS2 is a general-purpose I/O. When the serial communications interface 1 (SCI1) receiver is enabled, PS2 becomes the receive pin RXD of SCI1. While in reset and immediately out of reset, the PS2 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SCI block description chapter for information about pin configurations.

1.2.3.53 PS1 / SCI0_TXD — Port S I/O Pin 1

PS1 is a general-purpose I/O. When the serial communications interface 0 (SCI0) transmitter is enabled, PS1 becomes the transmit pin, TXD, of SCI0. While in reset and immediately out of reset, the PS1 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SCI block description chapter for information about pin configurations.

1.2.3.54 PS0 / SCI0_RXD — Port S I/O Pin 0

PS0 is a general-purpose I/O. When the serial communications interface 0 (SCI0) receiver is enabled, PS0 becomes the receive pin RXD0 of SCI0. While in reset and immediately out of reset, the PS0 pin is configured as a high-impedance input pin. See the port integration module (PIM) PIM_9NE64 block description chapter and the SCI block description chapter for information about pin configurations.

1.2.3.55 PT[7:4] / IOC1[7:4] — Port T I/O Pins [7:4]

PT[7:4] are general-purpose I/O pins. While the timer system 1 (TIM1) is enabled, these pins can also be configured as the TIM1 input capture or output compare pins IOC1[7-4]. While in reset and immediately out of reset, the PT[7:4] pins are configured as a high-impedance input pins. See the port integration module (PIM) PIM_9NE64 block description chapter and the TIM_16B4C block description chapter for information about pin configurations.

1.2.3.56 PHY_TXP — EPHY Twisted Pair Output +

Ethernet twisted pair output pin. This pin is hi-z out of reset.

1.2.3.57 PHY_TXN — EPHY Twisted Pair Output –

Ethernet twisted pair output pin. This pin is hi-z out of reset.

1.2.3.58 PHY_RXP — EPHY Twisted Pair Input +

Ethernet twisted pair input pin. This pin is hi-z out of reset.

1.2.3.59 PHY_RXN — EPHY Twisted Pair Input –

Ethernet twisted pair input pin. This pin is hi-z out of reset.

1.2.3.60 PHY_RBIAS — EPHY Bias Control Resistor

Connect a 1.0% external resistor, RBIAS, between PHY_RBIAS pin and PHY_VSSA. This resistor must be placed as near as possible to the chip pin. Stray capacitance must be kept to less than 10 pF (> 50 pF may cause instability). No high-speed signals are allowed in the region of RBIAS.

1.2.4 Power Supply Pins

1.2.4.1 V_{DDX1}, V_{DDX2}, V_{SSX1}, V_{SSX2} — Power & Ground Pins for I/O & Internal Voltage Regulator

External power and ground for I/O drivers. Bypass requirements depend on how heavily the MCU pins are loaded.

1.2.4.2 V_{DDR}/V_{REGEN} — Power Pin for Internal Voltage Regulator

External power for internal voltage regulator.

1.2.4.3 V_{DD1}, V_{DD2}, V_{SS1}, V_{SS2} — Core Power Pins

Power is supplied to the MCU through VDD and VSS. This 2.5V supply is derived from the internal voltage regulator. No static load is allowed on these pins. The internal voltage regulator is turned off, if VDDR/VREGEN is tied to ground.

1.2.4.4 V_{DDA}, V_{SSA} — Power Supply Pins for ATD and VREG_PHY

VDDA and VSSA are the power supply and ground input pins for the voltage regulator and the analog-to-digital converter.

1.2.4.5 PHY_VDDA, PHY_VSSA — Power Supply Pins for EPHY Analog

Power is supplied to the Ethernet physical transceiver (EPHY) PLLs through PHY_VDDA and PHY_VSSA. This 2.5V supply is derived from the internal voltage regulator. No static load is allowed on these pins. The internal voltage regulator is turned off, if VDDR/VREGEN is tied to ground.

1.2.4.6 PHY_VDDRX, PHY_VSSRX — Power Supply Pins for EPHY Receiver

Power is supplied to the Ethernet physical transceiver (EPHY) receiver through PHY_VDDRX and PHY_VSSRX. This 2.5V supply is derived from the internal voltage regulator. No static load is allowed on these pins. The internal voltage regulator is turned off, if VDDR/VREGEN is tied to ground.

1.2.4.7 PHY_VDDTX, PHY_VSSTX — Power Supply Pins for EPHY Transmitter

External power is supplied to the Ethernet physical transceiver (EPHY) transmitter through PHY_VDDTX and PHY_VSSTX. This 2.5 V supply is derived from the internal voltage regulator. No static load is allowed on these pins. The internal voltage regulator is turned off, if V_{DDR}/V_{REGEN} is tied to ground.

1.2.4.8 V_{RH}, V_{RL} — ATD Reference Voltage Input Pins

V_{RH} and V_{RL} are the reference voltage input pins for the analog-to-digital converter.

1.2.4.9 V_{DDPLL}, V_{SSPLL} — Power Supply Pins for PLL

Provides operating voltage and ground for the oscillator and the phase-locked loop. This allows the supply voltage to the oscillator and PLL to be bypassed independently. This 2.5 V voltage is generated by the internal voltage regulator. The internal voltage regulator is turned off, if V_{DDR}/V_{REGEN} is tied to ground.

Table 1-5. MC9S12NE64 Power and Ground Connection Summary

Mnemonic	Nominal Voltage	Description
V_{DDR}/V_{REGEN}	3.3 V	External power and ground, supply to internal voltage regulator. To disable voltage regulator attach V_{REGEN} to V_{SSX}.
V_{DDX1} V_{DDX2}	3.3 V	External power and ground, supply to pin drivers.
V_{SSX1} V_{SSX2}	0 V	
V_{DDA}	3.3 V	Operating voltage and ground for the analog-to-digital converter, the reference for the internal voltage regulator and the digital-to-analog converters, allows the supply voltage to the A/D to be bypassed independently.
V_{SSA}	0 V	
V_{RH}	3.3 V	Reference voltage high for the analog-to-digital converter.
V_{RL}	0 V	Reference voltage low for the analog-to-digital converter.
PHY_VDDTX PHY_VDDRX PHY_VDDA	2.5 V	Internal power and ground generated by internal regulator for internal Ethernet Physical Transceiver (EPHY). These also allow an external source to supply the EPHY voltages and bypass the internal voltage regulator.
PHY_VSSTX PHY_VSSRX PHY_VSSA	0V	
V_{DD1} V_{DD2}	2.5 V	Internal power and ground generated by internal regulator. These also allow an external source to supply the core V_{DD}/V_{SS} voltages and bypass the internal voltage regulator.
V_{SS1} V_{SS2}	0 V	
V_{DDPLL}	2.5 V	Provide operating voltage and ground for the phase-locked loop. This allows the supply voltage to the PLL to be bypassed independently. Internal power and ground generated by internal regulator.
V_{SSPLL}	0 V	

NOTE

All V_{SS} pins must be connected together in the application. Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as near to the MCU as possible. Bypass requirements depend on MCU pin load.

1.3 System Clock Description

The clock and reset generator provides the internal clock signals for the core and all peripheral modules. Figure 1-5 shows the clock connections from the CRG to all modules. See the CRG block description chapter for details on clock generation.

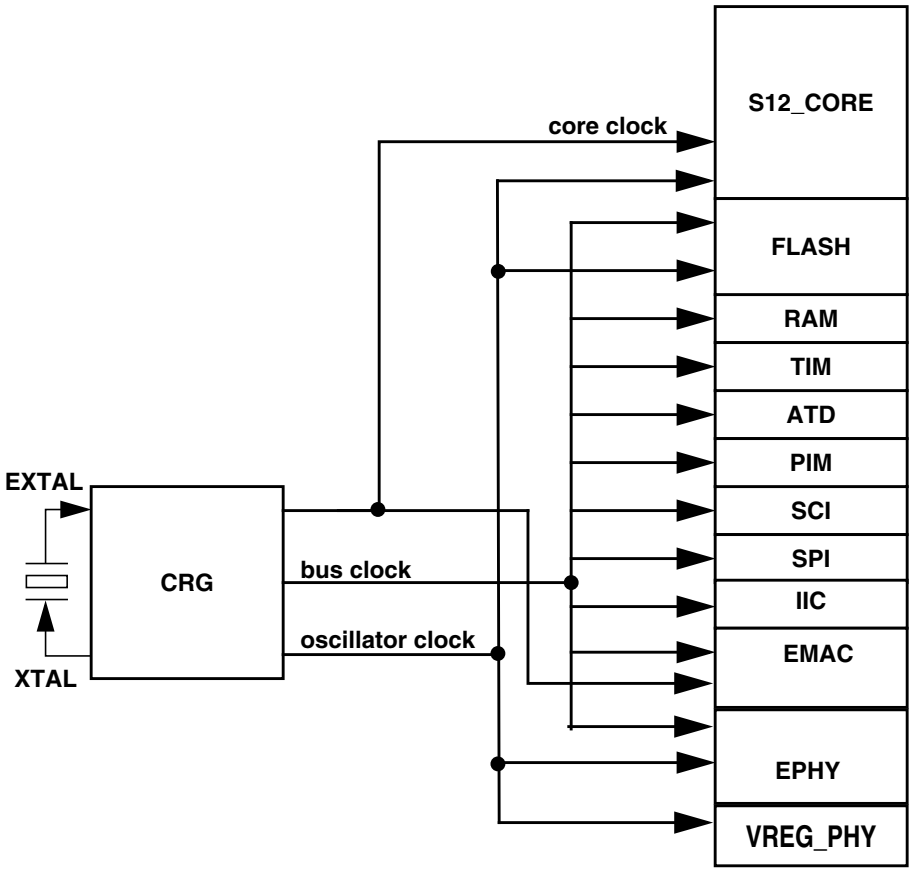


Figure 1-5. Clock Connections

1.4 Modes of Operation

There are eight possible modes of operation available on the MC9S12NE64. Each mode has an associated default memory map and external bus configuration.

1.4.1 Chip Configuration Summary

The operating mode out of reset is determined by the states of the MODC, MODB, and MODA pins during reset. The MODC, MODB, and MODA bits in the MODE register show the current operating mode and provide limited mode switching during operation. The states of the MODC, MODB, and MODA pins are latched into these bits on the rising edge of the $\overline{\text{RESET}}$ signal. The ROMCTL signal allows the setting of the ROMON bit in the MISC register thus controlling whether the internal FLASH is visible in the memory map. ROMON = 1 means the FLASH is visible in the memory map. The state of the ROMCTL pin is latched into the ROMON bit in the MISC register on the rising edge of the $\overline{\text{RESET}}$ signal.

Table 1-6. Mode Selection

BKGD = MODC	PE6 = MODB	PE5 = MODA	PP6 = ROMCTL	ROMON Bit	Mode Description
0	0	0	X	1	Special Single Chip, BDM allowed and ACTIVE. BDM is allowed in all other modes but a serial command is required to make BDM active.
0	0	1	0	1	Emulation Expanded Narrow, BDM allowed.
			1	0	
0	1	0	X	0	Special Test (Expanded Wide), BDM allowed.
0	1	1	0	1	Emulation Expanded Wide, BDM allowed.
			1	0	
1	0	0	X	1	Normal Single Chip, BDM allowed.
1	0	1	0	0	Normal Expanded Narrow, BDM allowed.
			1	1	
1	1	0	X	1	Peripheral; BDM allowed but bus operations would cause bus conflicts (must not be used).
1	1	1	0	0	Normal Expanded Wide, BDM allowed.
			1	1	

For further explanation on the modes, see the MEBI block description chapter.

1.4.2 Security

The MC9S12NE64 provides a security feature that prevents the unauthorized read and write of the memory contents[1]. This feature allows:

- Protection of the contents of FLASH
- Operation in single-chip mode
- Operation from external memory with internal FLASH disabled

On-chip security can be compromised by user code. An extreme example would be user code that dumps the contents of the internal program. This code would defeat the purpose of security. At the same time the user may also wish to put a back door in the user program. An example of this would be the user downloading a key through the SCI, which would allow access to a programming routine that could update parameters.

1.4.2.1 Securing the Microcontroller

After the user has programmed the FLASH, the MCU can be secured by programming the security bits located in the FLASH module. These nonvolatile bits will keep the MCU secured through resetting the MCU and through powering down the MCU.

The security byte resides in a portion of the FLASH array.

See the FLASH block description chapter for more details on the security configuration.

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

1.4.2.2 Operation of the Secured Microcontroller

1.4.2.2.1 Normal Single Chip Mode

This will be the most common usage of the secured MCU. Everything will appear the same as if the MCU were not secured, with the exception of BDM operation. The BDM operation will be blocked.

1.4.2.2.2 Executing from External Memory

The user may wish to execute from external memory with a secured microcontroller. This is accomplished by resetting directly into expanded mode. The internal FLASH will be disabled. BDM operations will be blocked.

1.4.2.3 Unsecuring the Microcontroller

In order to unsecure the microcontroller, the internal FLASH must be erased. This can be performed through an external program in expanded mode.

After the user has erased the FLASH, the MCU can be reset into special single chip mode. This invokes a program that verifies the erasure of the internal FLASH. After this program completes, the user can erase and program the FLASH security bits to the unsecured state. This is generally performed through the BDM, but the user could also change to expanded mode (by writing the mode bits through the BDM) and jumping to an external program (again through BDM commands). Note that if the MCU goes through a reset before the security bits are reprogrammed to the unsecure state, the MCU will be secured again.

1.5 Low-Power Modes

The microcontroller features three main low-power modes. See the respective block description chapter for information on the module behavior in stop, pseudo stop, and wait mode. An important source of information about the clock system is the clock and reset generator (CRG) block description chapter.

1.5.1 Stop

Executing the CPU STOP instruction stops all clocks and the oscillator thus putting the chip in fully static mode. Wakeup from this mode can be performed via reset or external interrupts.

1.5.2 Pseudo Stop

This mode is entered by executing the CPU STOP instruction. In this mode, the oscillator stays running and the real-time interrupt (RTI) or watchdog (COP) sub module can stay active. Other peripherals are turned off. This mode consumes more current than the full stop mode, but the wakeup time from this mode is significantly shorter.

1.5.3 Wait

This mode is entered by executing the CPU WAI instruction. In this mode, the CPU will not execute instructions. The internal CPU signals (address and databus) will be fully static. All peripherals stay active. For further power consumption, the peripherals can individually turn off their local clocks.

1.5.4 Run

Although this is not a low-power mode, unused peripheral modules must not be enabled in order to save power.

1.6 Resets and Interrupts

See the exception processing section of the CPU12 reference manual for information on resets and interrupts. System resets can be generated through external control of the $\overline{\text{RESET}}$ pin, through the clock and reset generator module (CRG), or through the low-voltage reset (LVR) generator of the voltage regulator module. See the CRG and VREG_PHY block description sections for detailed information on reset generation.

1.6.1 Vectors

Table 1-7 lists interrupt sources and vectors in default order of priority.

Table 1-7. Interrupt Vector Locations

Vector No.	Vector Address	Vector Name	Interrupt Source	CCR Mask	Local Enable	HPRIO Value to Elevate
0	$FFFE, $FFFF	Vreset	External reset, power on reset or low voltage reset (see CRG flags register to determine reset source)	None	None	—
1	$FFFC, $FFFD	Vclkmon	Clock monitor fail reset	None	COPCTL (CME, FCME)	—
2	$FFFA, $FFFB	Vcop	COP failure reset	None	COP rate select	—
3	$FFF8, $FFF9	Vtrap	Unimplemented instruction trap	None	None	—
4	$FFF6, $FFF7	Vswi	SWI	None	None	—
5	$FFF4, $FFF5	Vxirq	$\overline{\text{XIRQ}}$	X-Bit	None	—
6	$FFF2, $FFF3	Virq	$\overline{\text{IRQ}}$	I-Bit	INTCR (IRQEN)	$F2
7	$FFF0, $FFF1	Vrti	Real-time interrupt	I-Bit	CRGINT (RTIE)	$F0
8 through 11	$FFE8 to $FFEF	Reserved				
12	$FFE6, $FFE7	Vtimch4	Standard timer channel 4	I-Bit	T0IE (T0C4I)	$E6
13	$FFE4, $FFE5	Vtimch5	Standard timer channel 5	I-Bit	T0IE (T0C5I)	$E4
14	$FFE2, $FFE3	Vtimch6	Standard timer channel 6	I-Bit	T0IE (T0C6I)	$E2
15	$FFE0, $FFE1	Vtimch7	Standard timer channel 7	I-Bit	T0IE (T0C7I)	$E0

Table 1-7. Interrupt Vector Locations (Continued)

Vector No.	Vector Address	Vector Name	Interrupt Source	CCR Mask	Local Enable	HPRIO Value to Elevate
16	$FFDE, $FFDF	Vtimovf	Standard timer overflow	I-Bit	T0MSK2 (T0OI)	$DE
17	$FFDC, $FFDD	Vtimpaovf	Pulse accumulator overflow	I-Bit	PACTL0 (PAOVI0)	$DC
18	$FFDA, $FFDB	Vtimpaie	Pulse accumulator input edge	I-Bit	PACTL0 (PAI0)	$DA
19	$FFD8, $FFD9	Vspi	SPI	I-Bit	SPCR1 (SPIE, SPTIE)	$D8
20	$FFD6, $FFD7	Vsci0	SCI0	I-Bit	SC0CR2 (TIE, TCIE, RIE, ILIE)	$D6
21	$FFD4, $FFD5	Vsci1	SCI1	I-Bit	SC1CR2 (TIE, TCIE, RIE, ILIE)	$D4
22	$FFD2, $FFD3	Vatd	ATD	I-Bit	ATDCTL2 (ASCIE)	$D2
23	$FFD0, $FFD1	Reserved				
24	$FFCE, $FFCF	Vportj	Port J	I-Bit	PTJIF (PTJIE)	$CE
25	$FFCC, $FFCD	Vporth	Port H	I-Bit	PTHIF (PTHIE)	$CC
26	$FFCA, $FFCB	Vportg	Port G	I-Bit	PTGIF (PTGIE)	$CA
27	$FFC8, $FFC9	Reserved				
28	$FFC6, $FFC7	Vcrgpllck	CRG PLL lock	I-Bit	PLLCR (LOCKIE)	$C6
29	$FFC4, $FFC5	Vcrgscm	CRG self clock mode	I-Bit	PLLCR (SCMIE)	$C4
30	$FFC2, $FFC3	Reserved				
31	$FFC0, $FFC1	Viic	IIC bus	I-Bit	IBCR (IBIE)	$C0
32 through 34	$FFBA to $FFBF	Reserved				
35	$FFB8, $FFB9	Vflash	FLASH	I-Bit	FCNFG (CCIE, CBEIE)	$B8
36	$FFB6, $FFB7	Vephy	EPHY interrupt	I-Bit	EPHYCTL0 (EPHYIE)	$B6
37	$FFB4, $FFB5	Vemacrxbac	EMAC receive buffer A complete	I-Bit	IMASK (RXACIE)	$B4
38	$FFB2, $FFB3	Vemacrxbbc	EMAC receive buffer B complete	I-Bit	IMASK (RXBCIE)	$B2
39	$FFB0, $FFB1	Vemactxc	EMAC frame transmission complete	I-Bit	IMASK (TXCIE)	$B0
40	$FFAE, $FFAF	Vemacrxfc	EMAC receive flow control	I-Bit	IMASK (RFCIE)	$AE
41	$FFAC, $FFAD	Vemacmii	EMAC MII management transfer complete	I-Bit	IMASK (MMCIE)	$AC
42	$FFAA, $FFAB	Vemacrxerr	EMAC receive error	I-Bit	IMASK (RXAIE)	$AA
43	$FFA8, $FFA9	Vemacrxbao	EMAC receive buffer A overrun	I-Bit	IMASK (RXAOIE)	$A8
44	$FFA6, $FFA7	Vemacrxbbo	EMAC receive buffer B overrun	I-Bit	IMASK (RXBOIE)	$A6
45	$FFA4, $FFA5	Vemacbrxerr	EMAC babbling receive error	I-Bit	IMASK (BREIE)	$A4
46	$FFA2, $FFA3	Vemaclc	EMAC late collision	I-Bit	IMASK (LCIE)	$A2
47	$FFA0, $FFA1	Vemacec	EMAC excessive collision	I-Bit	IMASK (ECIE)	$A0
48 through 63	$FF80 to $FF9F	Reserved				

1.6.2 Resets

Resets are a subset of the interrupts featured inTable 1-7. The different sources capable of generating a system reset are summarized in Table 1-8.

1.6.2.1 Reset Summary Table

Table 1-8. Reset Summary

Reset	Priority	Source	Vector
Power-on reset	1	CRG module	$FFFE, $FFFF
External reset	1	$\overline{\text{RESET}}$ pin	$FFFE, $FFFF
Low-voltage reset	1	VREG_PHY module	$FFFE, $FFFF
Clock monitor reset	2	CRG module	$FFFC, $FFFD
COP watchdog reset	3	CRG module	$FFFA, $FFFB

1.6.2.2 Effects of Reset

When a reset occurs, MCU registers and control bits are changed to known start-up states. See the respective module block description chapter for register reset states. See the MEBI block description chapter for mode-dependent pin configuration of port A, B, E, and K out of reset.

See the PIM block description chapter for reset configurations of all peripheral module ports.

See Table 1-1 for locations of the memories depending on the operating mode after reset.

The RAM array is not automatically initialized out of reset.

1.7 Block Configuration for MC9S12NE64

This section contains information regarding how the modules are implemented on the MC9S12NE64 device.

1.7.1 V_{DDR}/V_{REGEN}

On the MC9S12NE64, the V_{DDR}/V_{REGEN} pin is used to enable or disable the internal voltage 3.3V to 2.5V regulator. If this pin is tied low, then V_{DD1}, V_{DD2}, V_{DDPLL}, PHY_VDDRX, PHY_VDDTX, and PHY_VDDA must be supplied externally.

1.7.2 V_{DD1}, V_{DD2}, V_{SS1}, V_{SS2}

In both the 112-pin LQFP and the 80-pin TQFP-EP package versions, both internal V_{DD} and V_{SS} of the 2.5 V domain are bonded out on two sides of the device as two pin pairs (V_{DD1}/V_{SS1} and V_{DD2}/V_{SS2}). V_{DD1} and V_{DD2} are connected together internally. V_{SS1} and V_{SS2} are connected together internally. This allows systems to employ better supply routing and further decoupling.

1.7.3 Clock Reset Generator (CRG)

See the CRG chapter for information about the clock and reset generator module. For the MC9S12NE64, only the Pierce circuitry is available for the oscillator.

The low-voltage reset feature uses the low-voltage reset signal from the VREG_PHY module as an input to the CRG module. When the regulator output voltage supply to the internal chip logic falls below a specified threshold, the LVR signal from the VREG_PHY module causes the CRG module to generate a reset. See the VREG_PHY block description chapter for voltage level specifications.

1.7.4 Oscillator (OSC)

See the OSC chapter for information about the oscillator module. The XCLKS input signal is not available on the MC9S12NE64. The signal is internally tied low to select the Pierce oscillator or external clock configuration.

1.7.5 Ethernet Media Access Controller (EMAC)

See the EMAC chapter for information about the Ethernet media access controller module. The EMAC is part of the IPBus domain.

1.7.5.1 EMAC MII External Pin Configuration

When the EMAC is configured for and external Ethernet physical transceiver internal pull-ups and pull-downs are not automatically configured on the MII inputs. Any internal pull-up or pull-down resistors, which may be required, must be configured by setting the appropriate pull control registers in the port integration module (PIM). This implementation allows the use of external pull-up and pull-down resistors.

1.7.5.2 EMAC Internal PHY Configuration

When the EXTPHY bit (in the EMAC NETCTL register) is set to 1, the EMAC is configured to work with the internal EPHY block. Please see 1.7.6, “Ethernet Physical Transceiver (EPHY),” for more information regarding the EPHY block.

1.7.5.3 Low-Power Operation

Special care must be taken when executing STOP and WAIT instructions while using the EMAC, or undesired operation may result.

1.7.5.3.1 Wait

Transmit and receive operations are not possible in wait mode if the CWAI bit is set in the CLKSEL register because the clocks to the transmit and receive buffers are stopped. It is recommended that the EMAC ESWAI bit be set if wait mode is entered with the CWAI set.

1.7.5.3.2 Stop

During system low-power stop mode, the EMAC is immediately disabled. Any receive in progress is dropped and any PAUSE time-out is cleared. The user must not to enter low-power stop mode when TXACT or BUSY are set.

1.7.6 Ethernet Physical Transceiver (EPHY)

See the EPHY chapter for information about the Ethernet physical transceiver module. The EPHY also has MII register space which is not part of the MCU address space and not accessible via the IP bus. The MII registers can be accessed using the MDIO functions of the EMAC when the EMAC is configured for internal PHY operation. The MII pins of the EPHY are not externally accessible. All communication and management of the EPHY must be performed using the EMAC.

The organization unique identifier (OUI) for the MC9S12NE64 is 00-60-11 (hex).

1.7.6.1 Low-Power Operation

Special care must be taken when executing STOP and WAIT instructions while using the EPHY or undesired operation may result.

1.7.6.1.1 Wait

Transmit and receive operations are not possible in wait mode if the CWAI bit is set in the CLKSEL register because the clocks to the internal MII interface are stopped.

1.7.6.1.2 Stop

During system low-power stop mode, the EPHY is immediately reset and powered down. Upon exiting stop mode, the a start-up delay is required prior to initiating MDIO communications with the EPHY. See A.14, “EPHY Electrical Characteristics” for details. It is not possible to use an EPHY interrupt to wake the system from stop mode.

1.7.7 RAM 8K Block Description

This module supports single-cycle misaligned word accesses without wait states.

In addition to operating as the CPU storage, the 8K system RAM also functions as the Ethernet buffer while the EMAC module is enabled. While the EMAC is enabled, the Ethernet buffer will occupy 0.375K to 4.5K of RAM with physical addresses starting at $0000 and ending at $017F up to $11FF, depending on the setting of the BUFMAP bits in the EMAC Ethernet buffer configuration register (BUFCFG). The relative RAM address, which are controlled by settings in the internal RAM position register (INTRM), must be tracked in software.

The Ethernet buffer operation of the RAM is independent of the CPU and allows same cycle read/write access from the CPU and the EMAC. No hardware blocking mechanism is implemented to prevent the CPU from accessing the Ethernet RAM space, so care must be taken to ensure that the CPU does not corrupt the RAM Ethernet contents.

Chapter 2
64 Kbyte Flash Module (S12FTS64KV3)

2.1 Introduction

This document describes the FTS64K module that includes a 64Kbyte Flash (nonvolatile) memory. The Flash memory may be read as either bytes, aligned words, or misaligned words. Read access time is one bus cycle for bytes and aligned words, and two bus cycles for misaligned words.

The Flash memory is ideal for program and data storage for single-supply applications allowing for field reprogramming without requiring external voltage sources for program or erase. Program and erase functions are controlled by a command driven interface. The Flash module supports both block erase and sector erase. An erased bit reads 1 and a programmed bit reads 0. The high voltage required to program and erase the Flash memory is generated internally. It is not possible to read from a Flash block while it is being erased or programmed.

CAUTION

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

2.1.1 Glossary

Command Write Sequence — A three-step MCU instruction sequence to execute built-in algorithms (including program and erase) on the Flash memory.

2.1.2 Features

- 64 Kbytes of Flash memory comprised of one 64 Kbyte block divided into 128 sectors of 512 bytes
- Automated program and erase algorithm
- Interrupts on Flash command completion, command buffer empty
- Fast sector erase and word program operation
- 2-stage command pipeline for faster multi-word program times
- Sector erase abort feature for critical interrupt response
- Flexible protection scheme to prevent accidental program or erase
- Single power supply for all Flash operations including program and erase
- Security feature to prevent unauthorized access to the Flash memory
- Code integrity check using built-in data compression

2.1.3 Modes of Operation

Program, erase, erase verify, and data compress operations (please refer to Section 2.4.1 for details).

2.1.4 Block Diagram

A block diagram of the Flash module is shown in Figure 2-1.

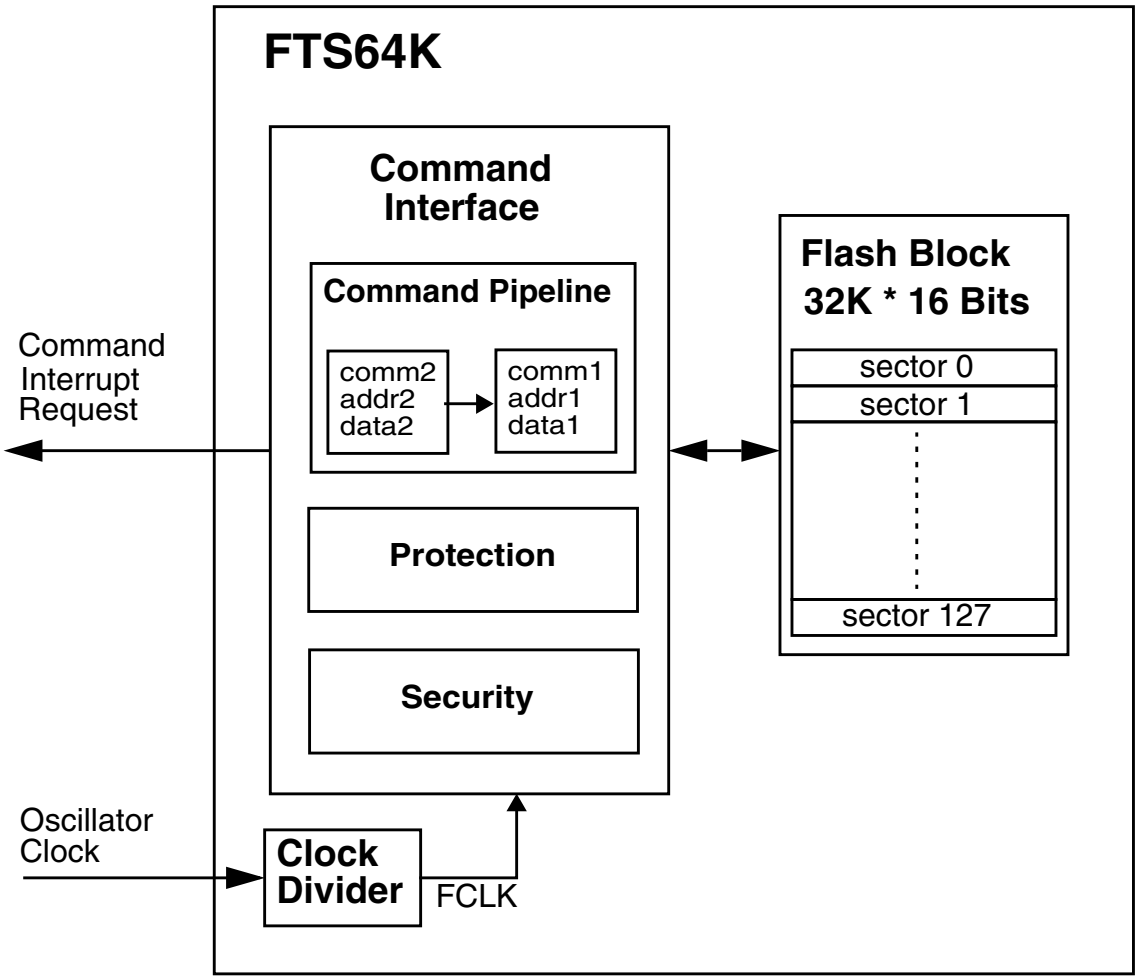


Figure 2-1. FTS64K Block Diagram

2.2 External Signal Description

The Flash module contains no signals that connect off-chip.

2.3 Memory Map and Register Definition

This subsection describes the memory map and registers for the Flash module.

2.3.1 Module Memory Map

The Flash memory map is shown in Figure 2-2. The HCS12 architecture places the Flash memory addresses between 0x4000 and 0xFFFF which corresponds to three 16-Kbyte pages. The content of the

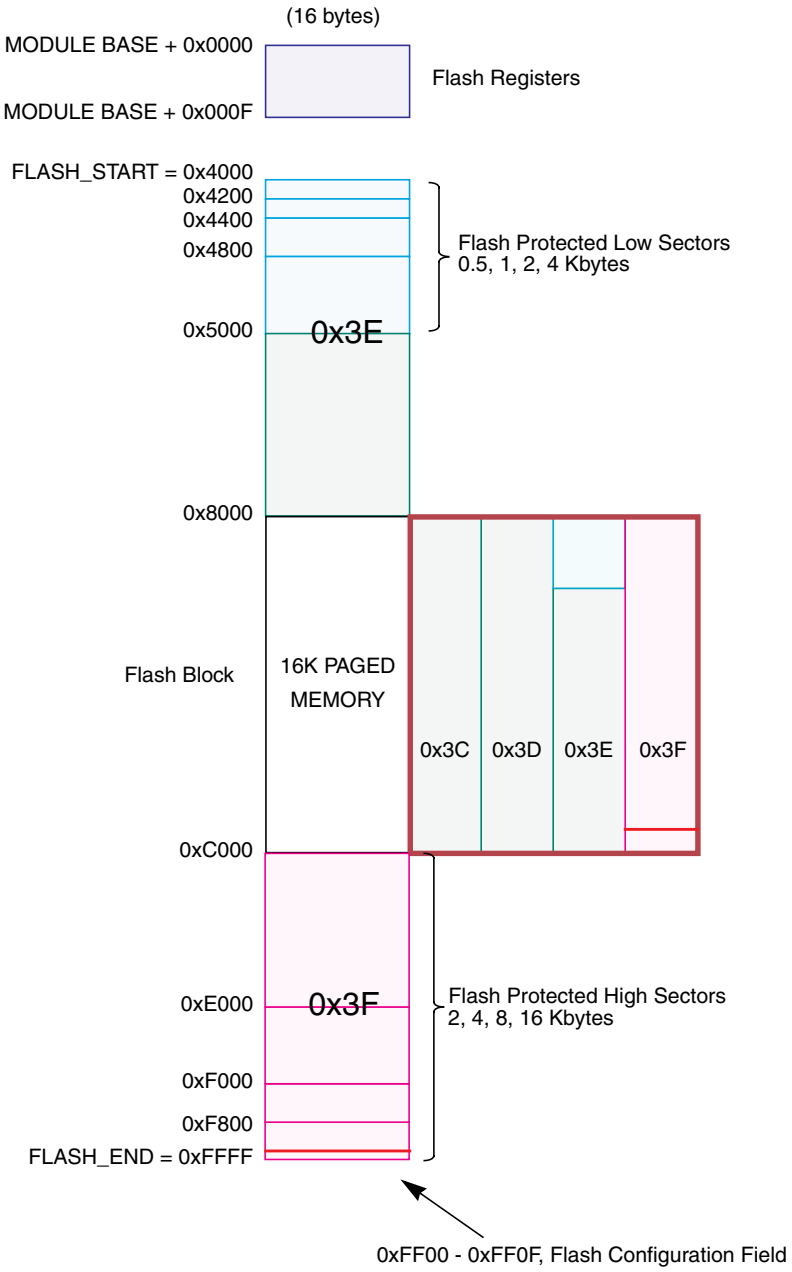
HCS12 core PPAGE register is used to map the logical middle page ranging from address 0x8000 to 0xBFFF to any physical 16 Kbyte page in the Flash memory. By placing 0x3E or 0x3F in the HCS12 Core PPAGE register, the associated 16 Kbyte pages appear twice in the MCU memory map.

The FPROT register, described in Section 2.3.2.5, “Flash Protection Register (FPROT)”, can be set to globally protect a Flash block. However, three separate memory regions, one growing upward from the first address in the next-to-last page in the Flash block (called the lower region), one growing downward from the last address in the last page in the Flash block (called the higher region), and the remaining addresses in the Flash block, can be activated for protection. The Flash locations of these protectable regions are shown in Table 2-2. The higher address region is mainly targeted to hold the boot loader code because it covers the vector space. The lower address region can be used for EEPROM emulation in an MCU without an EEPROM module because it can remain unprotected while the remaining addresses are protected from program or erase.

Security information that allows the MCU to restrict access to the Flash module is stored in the Flash configuration field, described in Table 2-1.

Table 2-1. Flash Configuration Field

Unpaged Flash Address	Paged Flash Address (PPAGE 0x3F)	Size (bytes)	Description
0xFF00 - 0xFF07	0xBF00-0xBF07	8	Backdoor Comparison Key Refer to Section Section 2.6.1, “Unsecuring the MCU using Backdoor Key Access”
0xFF08 - 0xFF0C	0xBF08-0xBF0C	5	Reserved
0xFF0D	0xBF0D	1	Flash Protection byte Refer to Section 2.3.2.5, “Flash Protection Register (FPROT)”
0xFF0E	0xBF0E	1	Flash Nonvolatile byte Refer to Section 2.3.2.9, “Flash Control Register (FCTL)”
0xFF0F	0xBF0F	1	Flash Security byte Refer to Section 2.3.2.2, “Flash Security Register (FSEC)”



Note: 0x3C-0x3F correspond to the PPAGE register content

Figure 2-2. Flash Memory Map

Table 2-2. Detailed Flash Memory Map Summary

MCU Address Range	PPAGE	Protectable Lower Range	Protectable Higher Range	Block Relative Address[1]
0x4000-0x7FFF	Unpaged (0x3E)	0x4000-0x41FF 0x4000-0x43FF 0x4000-0x47FF 0x4000-0x4FFF	N.A.	0x8000-0xBFFF
0x8000-0xBFFF	0x3C	N.A.	N.A.	0x0000-0x3FFF
	0x3D	N.A.	N.A.	0x4000-0x7FFF
	0x3E	0x8000-0x81FF 0x8000-0x83FF 0x8000-0x87FF 0x8000-0x8FFF	N.A.	0x8000-0xBFFF
	0x3F	N.A.	0xB800-0xBFFF 0xB000-0xBFFF 0xA000-0xBFFF 0x8000-0xBFFF	0xC000-0xFFFF
0xC000-0xFFFF	Unpaged (0x3F)	N.A.	0xF800-0xFFFF 0xF000-0xFFFF 0xE000-0xFFFF 0xC000-0xFFFF	0xC000-0xFFFF

1 Block Relative Address for 64 Kbyte Flash block consists of 16 address bits.

The Flash module also contains a set of 16 control and status registers located in address space module base + 0x0000 to module base + 0x000F. A summary of these registers is given in Table 2-3 while their accessibility in normal and special modes is detailed in Section 2.3.2, “Register Descriptions”.

Table 2-3. Flash Register Map

Module Base +	Register Name	Normal Mode Access
0x0000	Flash Clock Divider Register (FCLKDIV)	R/W
0x0001	Flash Security Register (FSEC)	R
0x0002	RESERVED1[1]	R
0x0003	Flash Configuration Register (FCNFG)	R/W
0x0004	Flash Protection Register (FPROT)	R/W
0x0005	Flash Status Register (FSTAT)	R/W
0x0006	Flash Command Register (FCMD)	R/W
0x0007	Flash Control Register (FCTL)	R
0x0008	Flash High Address Register (FADDRHI)[1]	R
0x0009	Flash Low Address Register (FADDRLO)[1]	R

Table 2-3. Flash Register Map

0x000A	Flash High Data Register (FDATAHI)	R
0x000B	Flash Low Data Register (FDATALO)	R
0x000C	RESERVED2[1]	R
0x000D	RESERVED3[1]	R
0x000E	RESERVED4[1]	R
0x000F	RESERVED5[1]	R

1 Intended for factory test purposes only.

2.3.2 Register Descriptions

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
FCLKDIV	R	FDIVLD	PRDIV8	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
	W								
FSEC	R	KEYEN		RNV5	RNV4	RNV3	RNV2	SEC	
	W								
FTSTMOD	R	0	0	0	0	0	0	0	0
	W								
FCNFG	R	CBEIE	CCIE	KEYACC	0	0	0	0	0
	W								
FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS		FPLDIS	FPLS	
	W								
FSTAT	R	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
	W								
FCMD	R	0	CMDB						
	W								
FCTL	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
	W								
FADDRHI	R	0	FADDRHI						
	W								
FADDRLO	R	FADDRLO							
	W								
FDATAHI	R	FDATAHI							
	W								
FDATALO	R	FDATALO							
	W								
RESERVED1	R	0	0	0	0	0	0	0	0
	W								

= Unimplemented or Reserved

Figure 2-3. FTS64K Register Summary

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
RESERVED2	R	0	0	0	0	0	0	0	0
	W								
RESERVED3	R	0	0	0	0	0	0	0	0
	W								
RESERVED4	R	0	0	0	0	0	0	0	0
	W								

= Unimplemented or Reserved

Figure 2-3. FTS64K Register Summary (continued)

2.3.2.1 Flash Clock Divider Register (FCLKDIV)

The FCLKDIV register is used to control timed events in program and erase algorithms.

	7	6	5	4	3	2	1	0
R	FDIVLD	PRDIV8	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-4. Flash Clock Divider Register (FCLKDIV)

All bits in the FCLKDIV register are readable, bits 6-0 are write once and bit 7 is not writable.

Table 2-4. FCLKDIV Field Descriptions

Field	Description
7 FDIVLD	**Clock Divider Loaded.** 0 Register has not been written. 1 Register has been written to since the last reset.
6 PRDIV8	**Enable Prescalar by 8.** 0 The oscillator clock is directly fed into the clock divider. 1 The oscillator clock is divided by 8 before feeding into the clock divider.
5-0 FDIV[5:0]	**Clock Divider Bits** — The combination of PRDIV8 and FDIV[5:0] must divide the oscillator clock down to a frequency of 150 kHz–200 kHz. The maximum divide ratio is 512. Please refer to Section 2.4.1.1, “Writing the FCLKDIV Register” for more information.

2.3.2.2 Flash Security Register (FSEC)

The FSEC register holds all bits associated with the security of the MCU and Flash module.

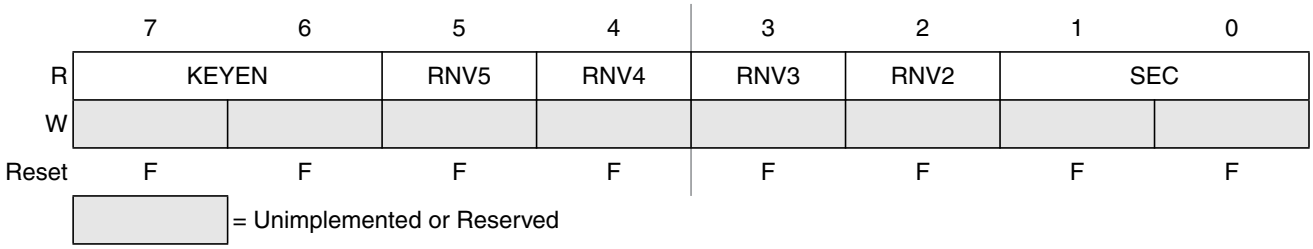


Figure 2-5. Flash Security Register (FSEC)

All bits in the FSEC register are readable but are not writable.

The FSEC register is loaded from the Flash Configuration Field at address $FF0F during the reset sequence, indicated by F in Figure 2-5.

Table 2-5. FSEC Field Descriptions

Field	Description
1-0 KEYEN[1:0]	**Backdoor Key Security Enable Bits** —The KEYEN[1:0] bits define the enabling of backdoor key access to the Flash module as shown in Table 2-6.
5-2 RNV[5:2]	**Reserved Nonvolatile Bits** — The RNV[5:2] bits must remain in the erased 1 state for future enhancements.
1-0 SEC[1:0]	**Flash Security Bits** — The SEC[1:0] bits define the security state of the MCU as shown in Table 2-7. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10.

Table 2-6. Flash KEYEN States

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01[1]	DISABLED
10	**ENABLED**
11	DISABLED

1 Preferred KEYEN state to disable Backdoor Key Access.

Table 2-7. Flash Security States

SEC[1:0]	Status of Security
00	SECURED
01[1]	SECURED
10	**UNSECURED**
11	SECURED

1 Preferred SEC state to set MCU to secured state.

The security function in the Flash module is described in Section 2.6, “Flash Module Security”.

2.3.2.3 RESERVED1

This register is reserved for factory testing and is not accessible.

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

= Unimplemented or Reserved

Figure 2-6. RESERVED1

All bits read 0 and are not writable in normal mode.

2.3.2.4 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash interrupts and gates the security backdoor writes.

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

= Unimplemented or Reserved

Figure 2-7. Flash Configuration Register (FCNFG)

CBEIE, CCIE and KEYACC bits are readable and writable while all remaining bits read 0 and are not writable. KEYACC is only writable if KEYEN (see Section 2.3.2.2) is set to the enabled state.

Table 2-8. FCNFG Field Descriptions

Field	Description
7 CBEIE	**Command Buffer Empty Interrupt Enable** — The CBEIE bit enables an interrupt in case of an empty command buffer in the Flash module. 0 Command buffer empty interrupt disabled. 1 An interrupt will be requested whenever the CBEIF flag (see Section 2.3.2.7, “Flash Status Register (FSTAT)”) is set.

Table 2-8. FCNFG Field Descriptions (continued)

Field	Description
6 CCIE	**Command Complete Interrupt Enable** — The CCIE bit enables an interrupt in case all commands have been completed in the Flash module. 0 Command complete interrupt disabled. 1 An interrupt will be requested whenever the CCIF flag (see Section 2.3.2.7, “Flash Status Register (FSTAT)”) is set.
5 KEYACC	**Enable Security Key Writing** 0 Flash writes are interpreted as the start of a command write sequence. 1 Writes to Flash array are interpreted as keys to open the backdoor. Reads of the Flash array return invalid data.

2.3.2.5 Flash Protection Register (FPROT)

The FPROT register defines which Flash sectors are protected against program or erase operations.

All bits in the FPROT register are readable and writable with restrictions except for RNV[6] which is only readable (see Section 2.3.2.6, “Flash Protection Restrictions”).

During reset, the FPROT register is loaded from the Flash Configuration Field at address 0xFF0D. To change the Flash protection that will be loaded during the reset sequence, the upper sector of the Flash memory must be unprotected, then the Flash Protect/Security byte located as described in Table 2-1 must be reprogrammed.

Trying to alter data in any of the protected areas in the Flash block will result in a protection violation error and the PVIOL flag will be set in the FSTAT register. A mass erase of the Flash block is not possible if any of the contained Flash sectors are protected.

Table 2-9. FPROT Field Descriptions

Field	Description
7 FPOPEN	**Protection Function Bit** — The FPOPEN bit determines the protection function for program or erase as shown in Table 2-10. 0 FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS[1:0] and FPLS[1:0] bits. For an MCU without an EEPROM module, the FPOPEN clear state allows the main part of the Flash block to be protected while a small address range can remain unprotected for EEPROM emulation. 1 FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS[1:0] and FPLS[1:0] bits.
6 RNV[6]	**Reserved Nonvolatile Bit** — The RNV[6] bit must remain in the erased state 1 for future enhancements.
5 FPHDIS	**Flash Protection Higher Address Range Disable** — The FPHDIS bit determines whether there is a protected/unprotected area in the higher address space of the Flash block. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
4:3 FPHS[1:0]	**Flash Protection Higher Address Size** — The FPHS[1:0] bits determine the size of the protected/unprotected area as shown in Table 2-11. The FPHS[1:0] bits can only be written to while the FPHDIS bit is set.

Table 2-9. FPROT Field Descriptions

Field	Description
2 FPLDIS	**Flash Protection Lower address range Disable** — The FPLDIS bit determines whether there is a protected/unprotected area in the lower address space of the Flash block. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
1:0 FPLS[1:0]	**Flash Protection Lower Address Size** — The FPLS[1:0] bits determine the size of the protected/unprotected area as shown in Table 2-12. The FPLS[1:0] bits can only be written to while the FPLDIS bit is set.

Table 2-10. Flash Protection Function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No Protection
1	1	0	Protected Low Range
1	0	1	Protected High Range
1	0	0	Protected High and Low Ranges
0	1	1	Full Block Protected
0	1	0	Unprotected Low Range
0	0	1	Unprotected High Range
0	0	0	Unprotected High and Low Ranges

1 For range sizes, refer to and .

Table 2-11. Flash Protection Higher Address Range

FPHS[1:0]	Unpaged Address Range	Paged Address Range	Protected Size
00	0xF800-0xFFFF	0x3F: 0xC800-0xCFFF	2 Kbytes
01	0xF000-0xFFFF	0x3F: 0xC000-0xCFFF	4 Kbytes
10	0xE000-0xFFFF	0x3F: 0xB000-0xCFFF	8 Kbytes
11	0xC000-0xFFFF	0x3F: 0x8000-0xCFFF	16 Kbytes

Table 2-12. Flash Protection Lower Address Range

FPLS[1:0]	Unpaged Address Range	Paged Address Range	Protected Size
00	0x4000-0x41FF	0x3E: 0x8000-0x81FF	512 bytes
01	0x4000-0x43FF	0x3E: 0x8000-0x83FF	1 Kbyte
10	0x4000-0x47FF	0x3E: 0x8000-0x87FF	2 Kbytes
11	0x4000-0x4FFF	0x3E: 0x8000-0x8FFF	4 Kbytes

All possible Flash protection scenarios are illustrated in Figure 2-8. Although the protection scheme is loaded from the Flash array after reset, it can be changed by the user. This protection scheme can be used by applications requiring re-programming in single-chip mode while providing as much protection as possible if re-programming is not required.

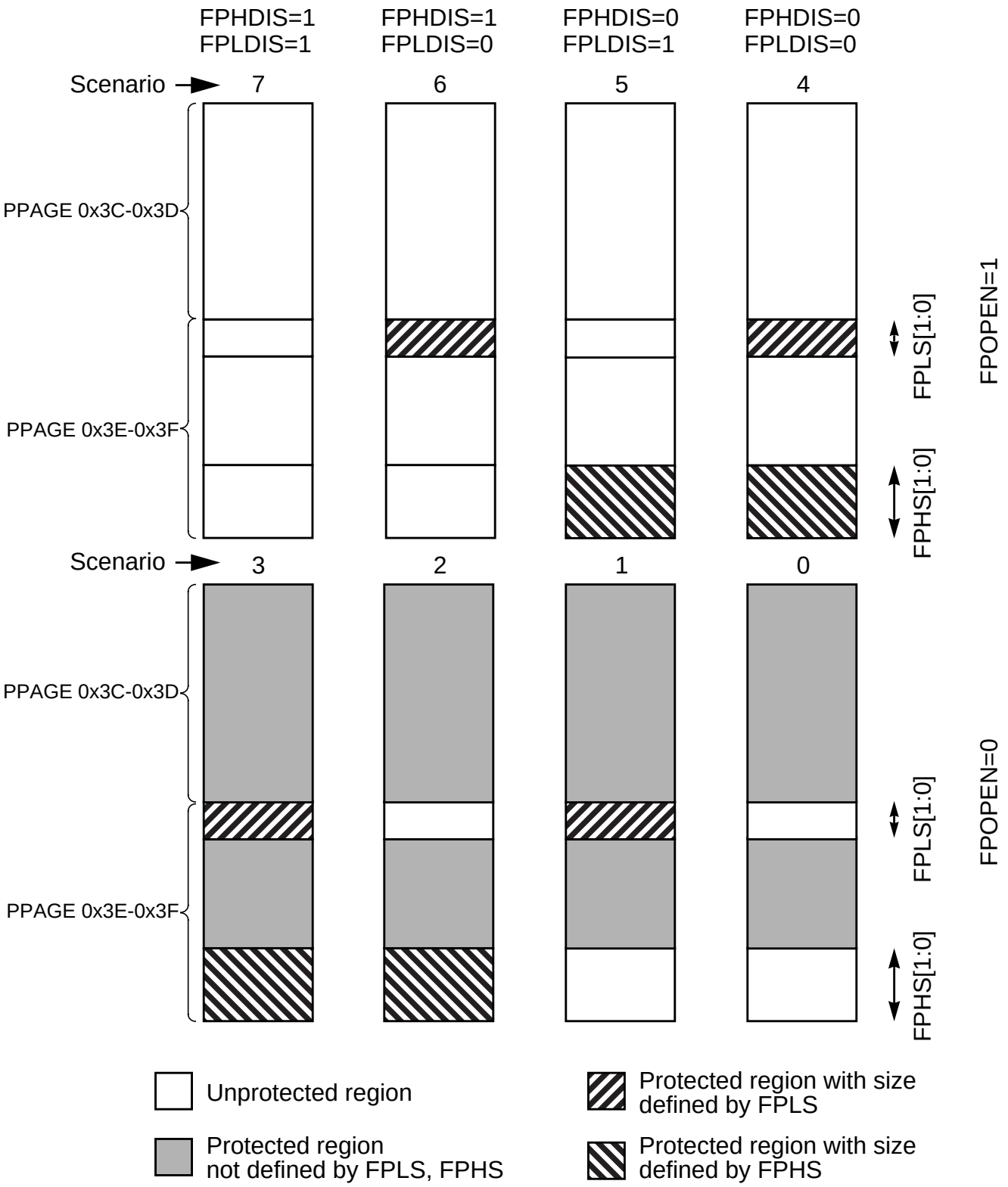


Figure 2-8. Flash Protection Scenarios

2.3.2.6 Flash Protection Restrictions

The general guideline is that Flash protection can only be added and not removed. Table 2-13 specifies all valid transitions between Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored and the FPROT register will remain unchanged. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS descriptions for additional restrictions.

Table 2-13. Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario[1]							
	0	1	2	3	4	5	6	7
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		
6		X		X	X		X	
7	X	X	X	X	X	X	X	X

1 Allowed transitions marked with X.

2.3.2.7 Flash Status Register (FSTAT)

The FSTAT register defines the operational status of the module.

	7	6	5	4	3	2	1	0
R	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
W								
Reset	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-9. Flash Status Register (FSTAT - Normal Mode)

	7	6	5	4	3	2	1	0
R	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	FAIL	0
W								
Reset	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-10. Flash Status Register (FSTAT - Special Mode)

CBEIF, PVIOL, and ACCERR are readable and writable, CCIF and BLANK are readable and not writable, remaining bits read 0and are not writable in normal mode. FAIL is readable and writable in special mode. FAIL must be clear when starting a command write sequence.

Table 2-14. FSTAT Field Descriptions

Field	Description
7 CBEIF	**Command Buffer Empty Interrupt Flag** — The CBEIF flag indicates that the address, data and command buffers are empty so that a new command write sequence can be started. The CBEIF flag is cleared by writing a 1 to CBEIF. Writing a 0 to the CBEIF flag has no effect on CBEIF. Writing a 0 to CBEIF after writing an aligned word to the Flash address space but before CBEIF is cleared will abort a command write sequence and cause the ACCERR flag to be set. Writing a 0 to CBEIF outside of a command write sequence will not set the ACCERR flag. The CBEIF flag is used together with the CBEIE bit in the FCNFG register to generate an interrupt request (see Figure 2-28). 0 Buffers are full. 1 Buffers are ready to accept a new command.
6 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that there are no more commands pending. The CCIF flag is cleared when CBEIF is clear and sets automatically upon completion of all active and pending commands. The CCIF flag does not set when an active commands completes and a pending command is fetched from the command buffer. Writing to the CCIF flag has no effect on CCIF. The CCIF flag is used together with the CCIE bit in the FCNFG register to generate an interrupt request (see Figure 2-28). 0 Command in progress. 1 All commands are completed.
5 PVIOL	**Protection Violation Flag** — The PVIOL flag indicates an attempt was made to program or erase an address in a protected area of the Flash block during a command write sequence. The PVIOL flag is cleared by writing a 1 to PVIOL. Writing a 0 to the PVIOL flag has no effect on PVIOL. While PVIOL is set, it is not possible to launch a command or start a command write sequence. 0 No failure. 1 A protection violation has occurred.
4 ACCERR	**Access Error Flag** — The ACCERR flag indicates an illegal access to the Flash array caused by either a violation of the command write sequence, issuing an illegal command (illegal combination of the CMDBx bits in the FCMD register), launching the sector erase abort command terminating a sector erase operation early or the execution of a CPU STOP instruction while a command is executing (CCIF = 0). The ACCERR flag is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR flag has no effect on ACCERR. While ACCERR is set, it is not possible to launch a command or start a command write sequence. If ACCERR is set by an erase verify operation or a data compress operation, any buffered command will not launch. 0 No access error detected. 1 Access error has occurred.
2 BLANK	**Erase Verify Operation Status Flag** — When the CCIF flag is set after completion of an erase verify command, the BLANK flag indicates the result of the erase verify operation. The BLANK flag is cleared by the Flash module when CBEIF is cleared as part of a new valid command write sequence. Writing to the BLANK flag has no effect on BLANK. 0 Flash block verified as not erased. 1 Flash block verified as erased.
1 FAIL	**Flag Indicating a Failed Flash Operation** — The FAIL flag will set if the erase verify operation fails (Flash block verified as not erased). The FAIL flag is cleared by writing a 1 to FAIL. Writing a 0 to the FAIL flag has no effect on FAIL. 0 Flash operation completed without error. 1 Flash operation failed.

2.3.2.8 Flash Command Register (FCMD)

The FCMD register is the Flash command register.

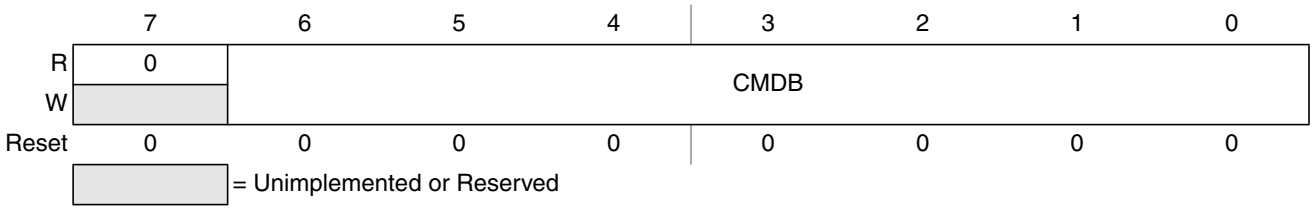


Figure 2-11. Flash Command Register (FCMD - NVM User Mode)

All CMDB bits are readable and writable during a command write sequence while bit 7 reads 0 and is not writable.

Table 2-15. FCMD Field Descriptions

Field	Description
6-0 CMDB[6:0]	**Flash Command** — Valid Flash commands are shown in Table 2-16. Writing any command other than those listed in Table 2-16 sets the ACCERR flag in the FSTAT register.

Table 2-16. Valid Flash Command List

CMDB[6:0]	NVM Command
0x05	Erase Verify
0x06	Data Compress
0x20	Word Program
0x40	Sector Erase
0x41	Mass Erase
0x47	Sector Erase Abort

2.3.2.9 Flash Control Register (FCTL)

The FCTL register is the Flash control register.

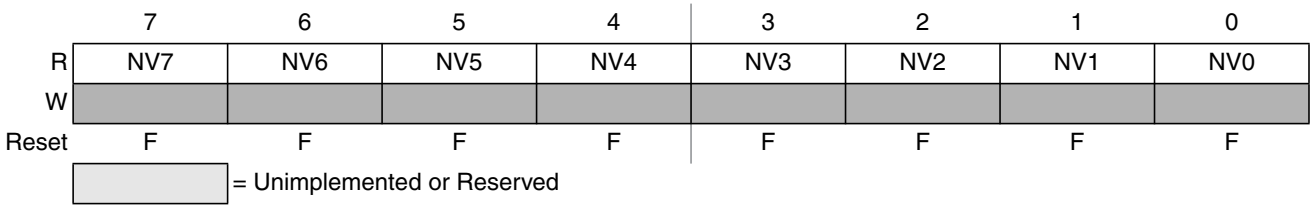


Figure 2-12. Flash Control Register (FCTL)

All bits in the FCTL register are readable but are not writable.

The FCTL register is loaded from the Flash Configuration Field byte at $FF0E during the reset sequence, indicated by F in Figure 2-12.

Table 2-17. FCTL Field Descriptions

Field	Description
7-0 NV[7:0]	**Nonvolatile Bits** — The NV[7:0] bits are available as nonvolatile bits. Refer to the Device User Guide for proper use of the NV bits.

2.3.2.10 Flash Address Registers (FADDR)

The FADDRHI and FADDRLO registers are the Flash address registers.

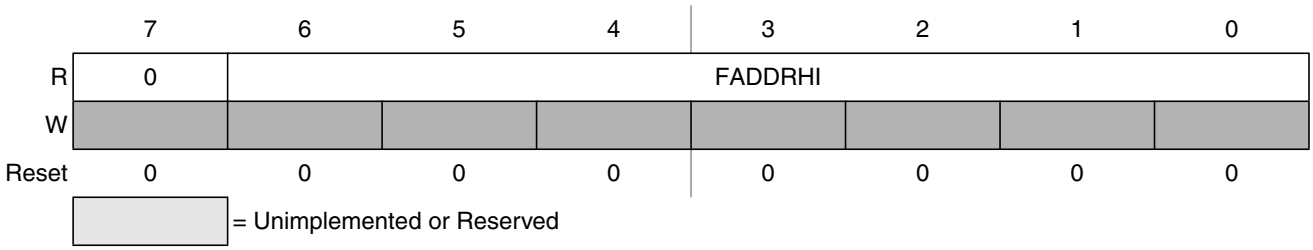


Figure 2-13. Flash Address High Register (FADDRHI)

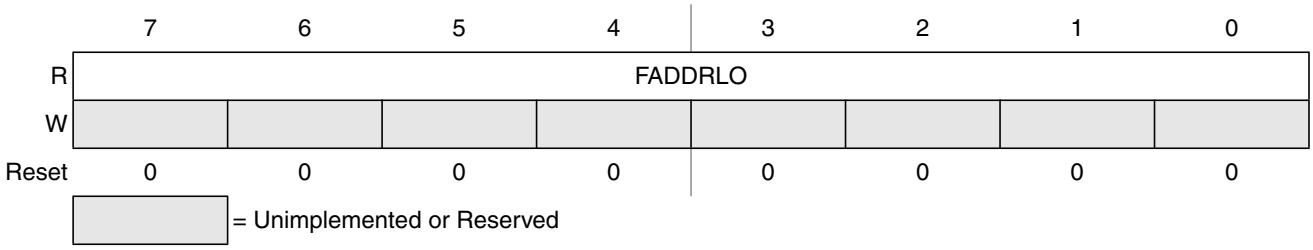


Figure 2-14. Flash Address Low Register (FADDRLO)

All FADDRHI and FADDRLO bits are readable but are not writable. After an array write as part of a command write sequence, the FADDR registers will contain the mapped MCU address written.

2.3.2.11 Flash Data Registers (FDATA)

The FDATAHI and FDATALO registers are the Flash data registers.

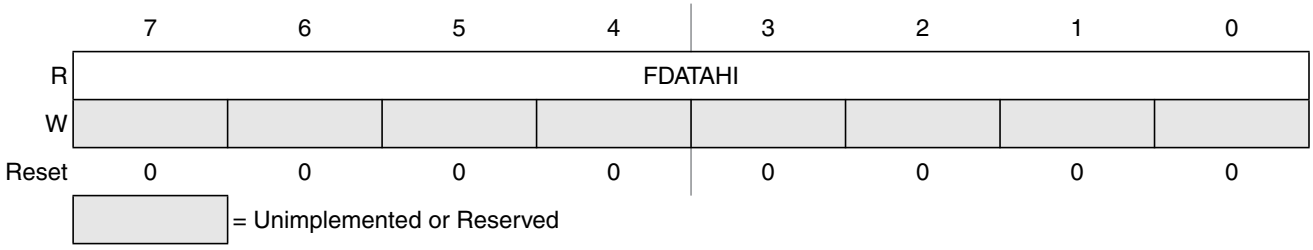


Figure 2-15. Flash Data High Register (FDATAHI)

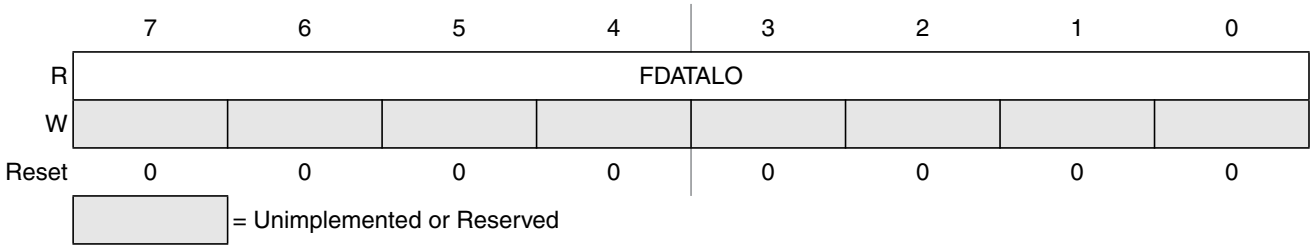


Figure 2-16. Flash Data Low Register (FDATALO)

All FDATAHI and FDATALO bits are readable but are not writable. After an array write as part of a command write sequence, the FDATA registers will contain the data written. At the completion of a data compress operation, the resulting 16-bit signature is stored in the FDATA registers. The data compression signature is readable in the FDATA registers until a new command write sequence is started.

2.3.2.12 RESERVED2

This register is reserved for factory testing and is not accessible.

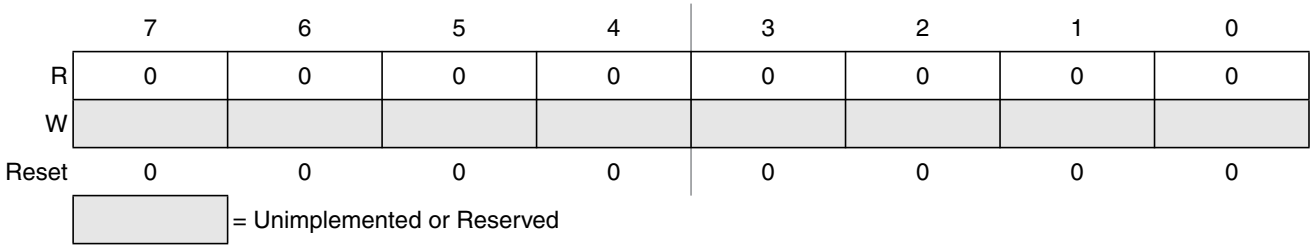


Figure 2-17. RESERVED2

All bits read 0 and are not writable.

2.3.2.13 RESERVED3

This register is reserved for factory testing and is not accessible.

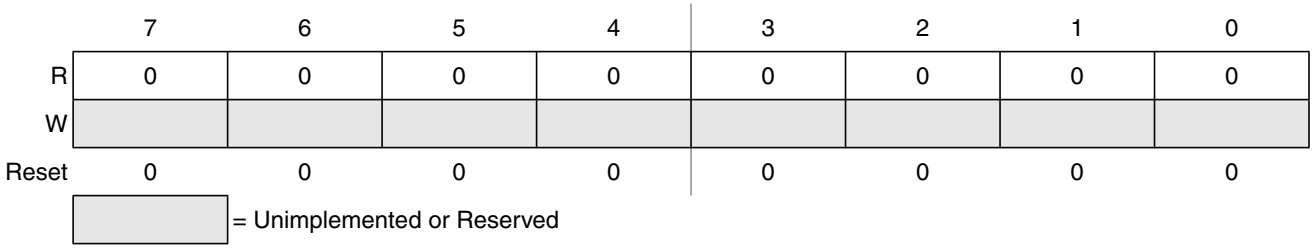


Figure 2-18. RESERVED3

All bits read 0 and are not writable.

2.3.2.14 RESERVED4

This register is reserved for factory testing and is not accessible.

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

= Unimplemented or Reserved

Figure 2-19. RESERVED4

All bits read 0 and are not writable.

2.3.2.15 RESERVED5

This register is reserved for factory testing and is not accessible.

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

= Unimplemented or Reserved

Figure 2-20. RESERVED5

All bits read 0 and are not writable.

2.4 Functional Description

2.4.1 Flash Command Operations

Write and read operations are both used for the program, erase, erase verify, and data compress algorithms described in this subsection. The program and erase algorithms are time controlled by a state machine whose timebase, FCLK, is derived from the oscillator clock via a programmable divider. The command register as well as the associated address and data registers operate as a buffer and a register (2-stage FIFO) so that a second command along with the necessary data and address can be stored to the buffer while the first command remains in progress. This pipelined operation allows a time optimization when programming more than one word on a specific row in the Flash block as the high voltage generation can be kept active in between two programming commands. The pipelined operation also allows a simplification of command launching. Buffer empty as well as command completion are signalled by flags in the Flash status register with interrupts generated, if enabled.

The next paragraphs describe:

1. How to write the FCLKDIV register.
2. Command write sequences used to program, erase, and verify the Flash memory.
3. Valid Flash commands.
4. Effects resulting from illegal Flash command write sequences or aborting Flash operations.

2.4.1.1 Writing the FCLKDIV Register

Prior to issuing any program, erase, erase verify, or data compress command, it is first necessary to write the FCLKDIV register to divide the oscillator clock down to within the 150 kHz to 200 kHz range. Because the program and erase timings are also a function of the bus clock, the FCLKDIV determination must take this information into account.

If we define:

- FCLK as the clock of the Flash timing control block,
- Tbus as the period of the bus clock, and
- INT(x) as taking the integer part of x (e.g. INT(4.323)=4).

Then, FCLKDIV register bits PRDIV8 and FDIV[5:0] are to be set as described in Figure 2-21.

For example, if the oscillator clock frequency is 950 kHz and the bus clock frequency is 10 MHz, FCLKDIV bits FDIV[5:0] must be set to 4 (000100) and bit PRDIV8 set to 0. The resulting FCLK frequency is then 190 kHz. As a result, the Flash program and erase algorithm timings are increased over the optimum target by:

$$(200 - 190)/200 \times 100 \; = \; 5\%$$

CAUTION

Program and erase command execution time will increase proportionally with the period of FCLK. Because of the impact of clock synchronization on the accuracy of the functional timings, programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 1 MHz. Programming or erasing the Flash memory with FCLK < 150 kHz must be avoided. Setting FCLKDIV to a value such that FCLK < 150 kHz can destroy the Flash memory due to overstress. Setting FCLKDIV to a value such that (1/FCLK + Tbus) < 5µs can result in incomplete programming or erasure of the Flash memory cells.

If the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. Flash commands will not be executed if the FCLKDIV register has not been written to.

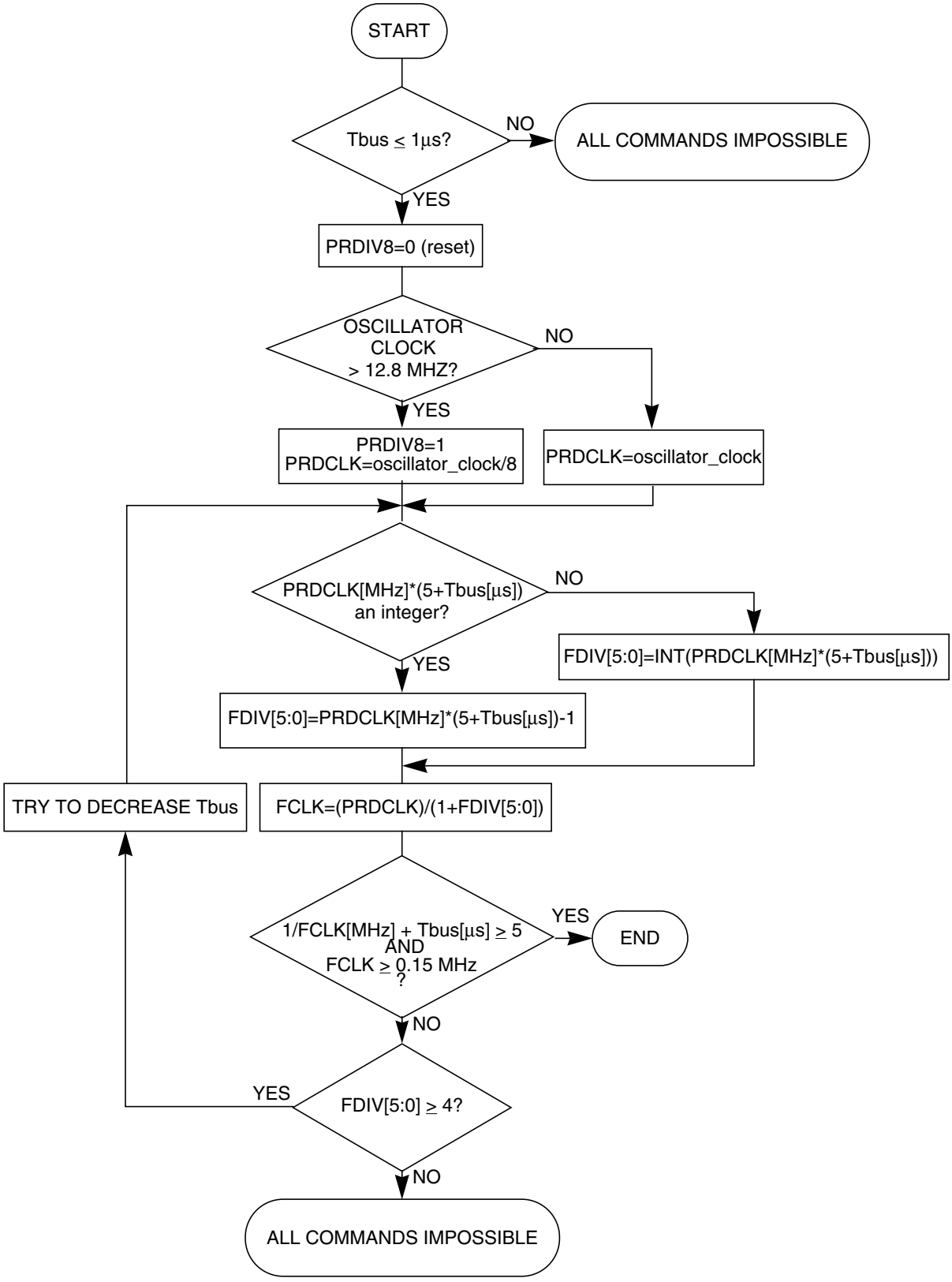


Figure 2-21. Determination Procedure for PRDIV8 and FDIV Bits

2.4.1.2 Command Write Sequence

The Flash command controller is used to supervise the command write sequence to execute program, erase, erase verify, and data compress algorithms.

Before starting a command write sequence, the ACCERR and PVIOL flags in the FSTAT register must be clear (see Section 2.3.2.7, “Flash Status Register (FSTAT)”) and the CBEIF flag must be tested to determine the state of the address, data, and command buffers. If the CBEIF flag is set, indicating the buffers are empty, a new command write sequence can be started. If the CBEIF flag is clear, indicating the buffers are not available, a new command write sequence will overwrite the contents of the address, data, and command buffers.

A command write sequence consists of three steps which must be strictly adhered to with writes to the Flash module not permitted between the steps. However, Flash register and array reads are allowed during a command write sequence. A command write sequence consists of the following steps:

1. Write an aligned data word to a valid Flash array address. The address and data will be stored in the address and data buffers, respectively. If the CBEIF flag is clear when the Flash array write occurs, the contents of the address and data buffers will be overwritten and the CBEIF flag will be set.
2. Write a valid command to the FCMD register.
   a) For the erase verify command (see Section 2.4.1.3.1, “Erase Verify Command”), the contents of the data buffer are ignored and all address bits in the address buffer are ignored.
   b) For the data compress command (see Section 2.4.1.3.2, “Data Compress Command”), the contents of the data buffer represents the number of consecutive words to read for data compression and the contents of the address buffer represents the starting address.
   c) For the program command (see Section 2.4.1.3.3, “Program Command”), the contents of the data buffer will be programmed to the address specified in the address buffer with all address bits valid.
   d) For the sector erase command (see Section 2.4.1.3.4, “Sector Erase Command”), the contents of the data buffer are ignored and address bits [9:0] contained in the address buffer are ignored.
   e) For the mass erase command (see Section 2.4.1.3.5, “Mass Erase Command”), the contents of the data buffer and address buffer are ignored.
   f) For the sector erase abort command (see Section 2.4.1.3.6, “Sector Erase Abort Command”), the contents of the data buffer and address buffer are ignored.
3. Clear the CBEIF flag by writing a 1 to CBEIF to launch the command. When the CBEIF flag is cleared, the CCIF flag is cleared on the same bus cycle by internal hardware indicating that the command was successfully launched. For all command write sequences except data compress and sector erase abort, the CBEIF flag will set four bus cycles after the CCIF flag is cleared indicating that the address, data, and command buffers are ready for a new command write sequence to begin. For data compress and sector erase abort operations, the CBEIF flag will remain clear until the operation completes.

A command write sequence can be aborted prior to clearing the CBEIF flag by writing a 0 to the CBEIF flag and will result in the ACCERR flag being set.

Except for the sector erase abort command, a buffered command will wait for the active operation to be completed before being launched. The sector erase abort command is launched when the CBEIF flag is cleared as part of a sector erase abort command write sequence. After a command is launched, the completion of the command operation is indicated by the setting of the CCIF flag. The CCIF flag only sets when all active and buffered commands have been completed.

2.4.1.3 Valid Flash Commands

Table 2-18 summarizes the valid Flash commands along with the effects of the commands on the Flash block.

Table 2-18. Valid Flash Command Description

FCMDB	NVM Command	Function on Flash Memory
0x05	Erase Verify	Verify all memory bytes in the Flash block are erased. If the Flash block is erased, the BLANK flag in the FSTAT register will set upon command completion.
0x06	Data Compress	Compress data from a selected portion of the Flash block. The resulting signature is stored in the FDATA register.
0x20	Program	Program a word (two bytes) in the Flash block.
0x40	Sector Erase	Erase all memory bytes in a sector of the Flash block.
0x41	Mass Erase	Erase all memory bytes in the Flash block. A mass erase of the full Flash block is only possible when FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register are set prior to launching the command.
0x47	Sector Erase Abort	Abort the sector erase operation. The sector erase operation will terminate according to a set procedure. The Flash sector must not be considered erased if the ACCERR flag is set upon command completion.

CAUTION

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

2.4.1.3.1 Erase Verify Command

The erase verify operation is used to confirm that a Flash block is erased. After launching the erase verify command, the CCIF flag in the FSTAT register will set after the operation has completed unless a second command has been buffered. The number of bus cycles required to execute the erase verify operation is equal to the number of addresses in the Flash block plus 12 bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set. The result of the erase verify operation is reflected in the state of the BLANK flag in the FSTAT register. If the BLANK flag is set in the FSTAT register, the Flash memory is erased.

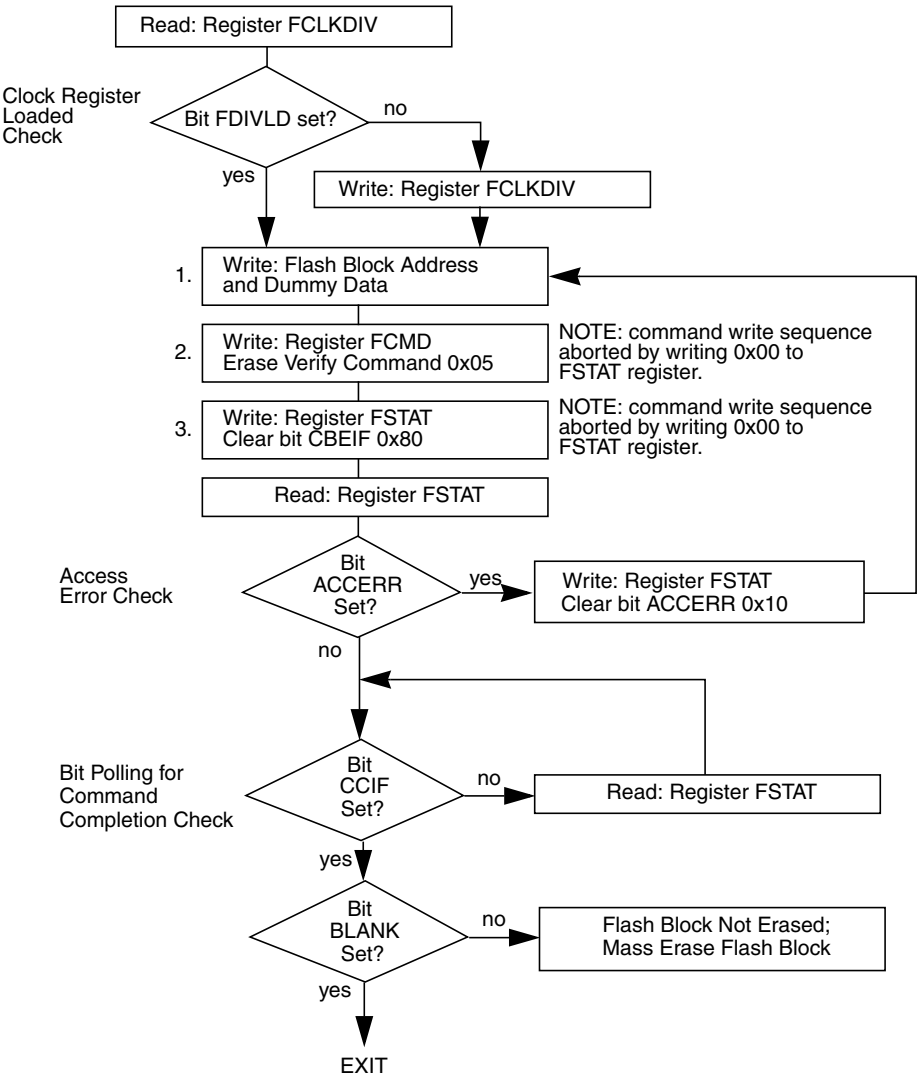


Figure 2-22. Example Erase Verify Command Flow

2.4.1.3.2 Data Compress Command

The data compress command is used to check Flash code integrity by compressing data from a selected portion of the Flash block into a signature analyzer. The starting address for the data compress operation is defined by the address written during the command write sequence. The number of consecutive word addresses compressed is defined by the data written during the command write sequence. The number of words that can be compressed in a single data compress operation ranges from 1 to 16,384. After launching the data compress command, the CCIF flag in the FSTAT register will set after the data compress operation has completed. The number of bus cycles required to execute the data compress operation is equal to two times the number of addresses read plus 20 bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set. After the CCIF flag is set, the signature generated by the data compress operation is available in the FDATA register. The signature in the FDATA register can be compared to the expected signature to determine the integrity of the selected data stored in the Flash block. If the last address of the Flash block is reached during the data compress operation, data compression will continue with the starting address of the Flash block.

NOTE

Since the FDATA register (or data buffer) is written to as part of the data compress operation, a command write sequence is not allowed to be buffered behind a data compress command write sequence. The CBEIF flag will not set after launching the data compress command to indicate that a command must not be buffered behind it. If an attempt is made to start a new command write sequence with a data compress operation active, the ACCERR flag in the FSTAT register will be set. A new command write sequence must only be started after reading the signature stored in the FDATA register.

In order to take corrective action, it is recommended that the data compress command be executed on a Flash sector or subset of a Flash sector. If the data compress operation on a Flash sector returns an invalid signature, the Flash sector must be erased using the sector erase command and then reprogrammed using the program command.

The data compress command can be used to verify that a sector or sequential set of sectors are erased.

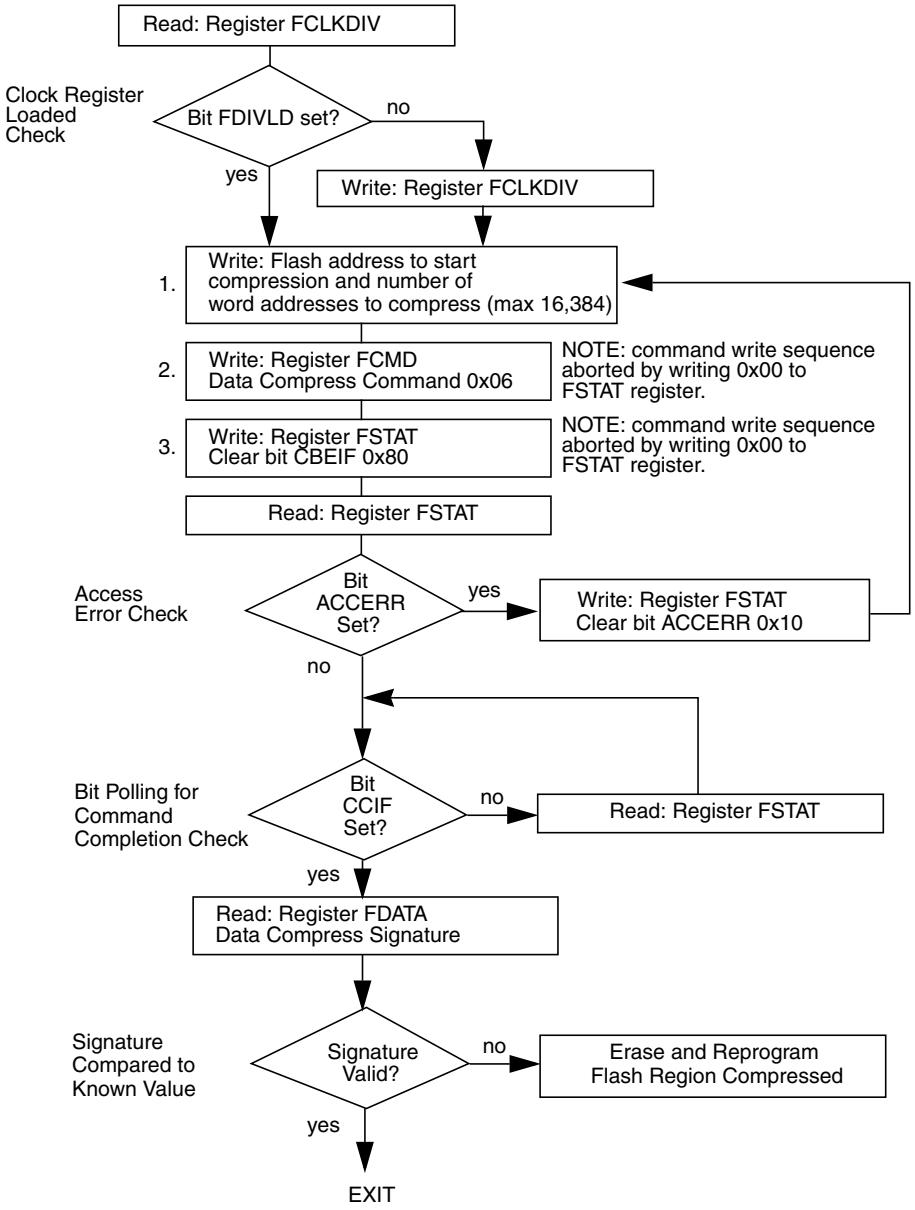


Figure 2-23. Example Data Compress Command Flow

2.4.1.3.3 Program Command

The program command is used to program a previously erased word in the Flash memory using an embedded algorithm. If the word to be programmed is in a protected area of the Flash block, the PVIOL flag in the FSTAT register will set and the program command will not launch. After the program command has successfully launched, the CCIF flag in the FSTAT register will set after the program operation has completed unless a second command has been buffered.

A summary of the launching of a program operation is shown in Figure 2-24.

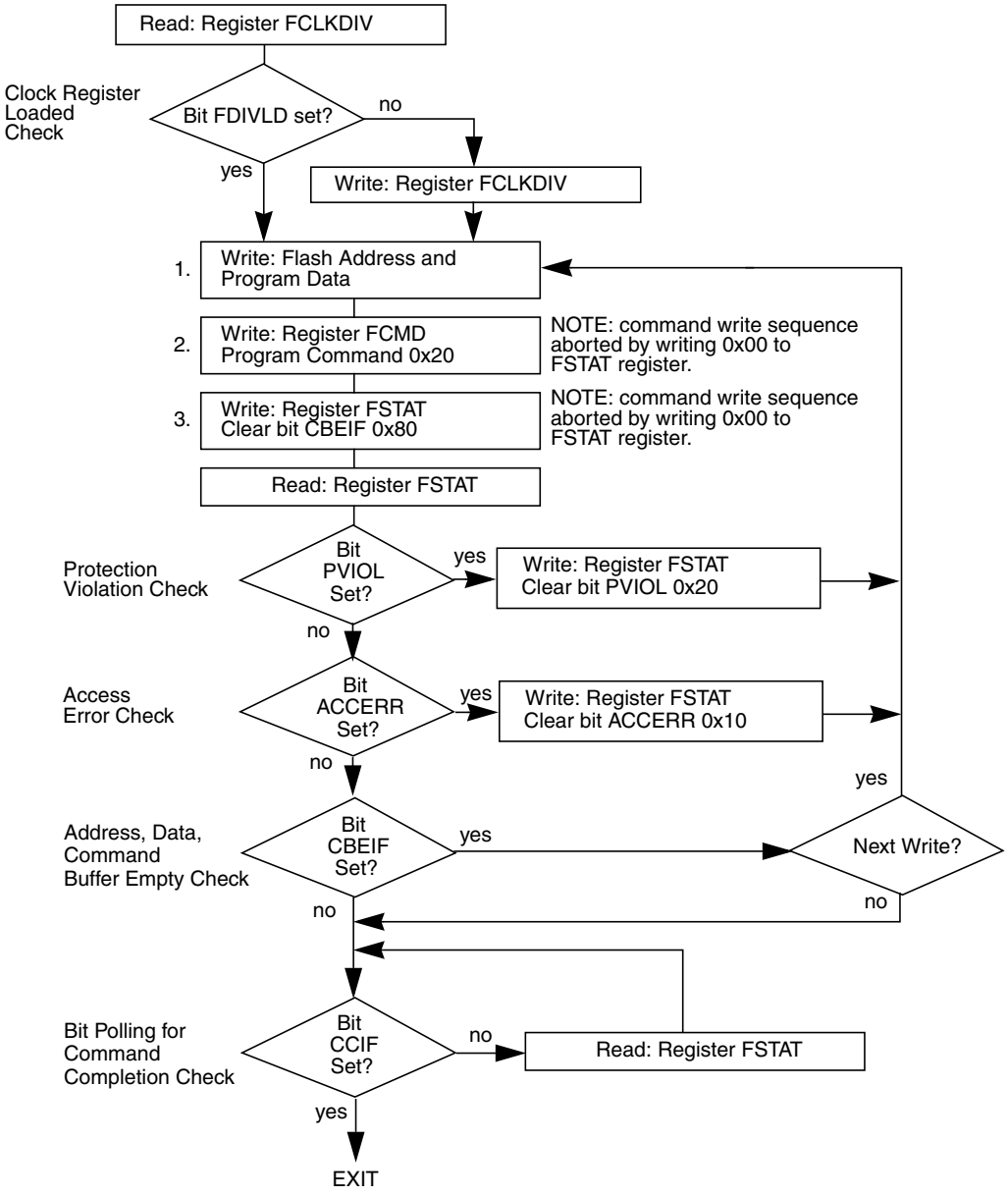


Figure 2-24. Example Program Command Flow

2.4.1.3.4 Sector Erase Command

The sector erase command is used to erase the addressed sector in the Flash memory using an embedded algorithm. If the Flash sector to be erased is in a protected area of the Flash block, the PVIOL flag in the FSTAT register will set and the sector erase command will not launch. After the sector erase command has successfully launched, the CCIF flag in the FSTAT register will set after the sector erase operation has completed unless a second command has been buffered.

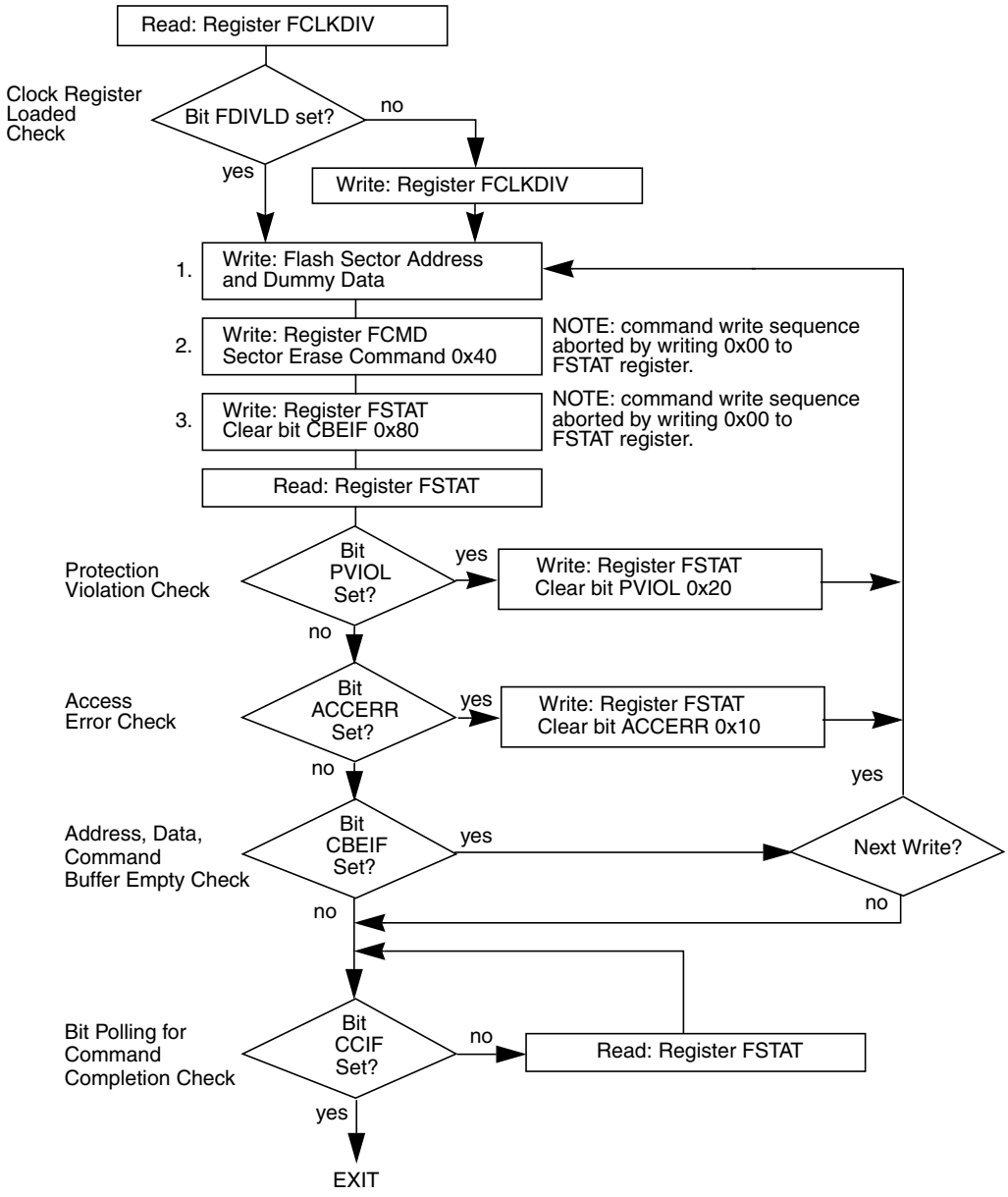


Figure 2-25. Example Sector Erase Command Flow

2.4.1.3.5 Mass Erase Command

The mass erase command is used to erase a Flash memory block using an embedded algorithm. If the Flash block to be erased contains any protected area, the PVIOL flag in the FSTAT register will set and the mass erase command will not launch. After the mass erase command has successfully launched, the CCIF flag in the FSTAT register will set after the mass erase operation has completed unless a second command has been buffered.

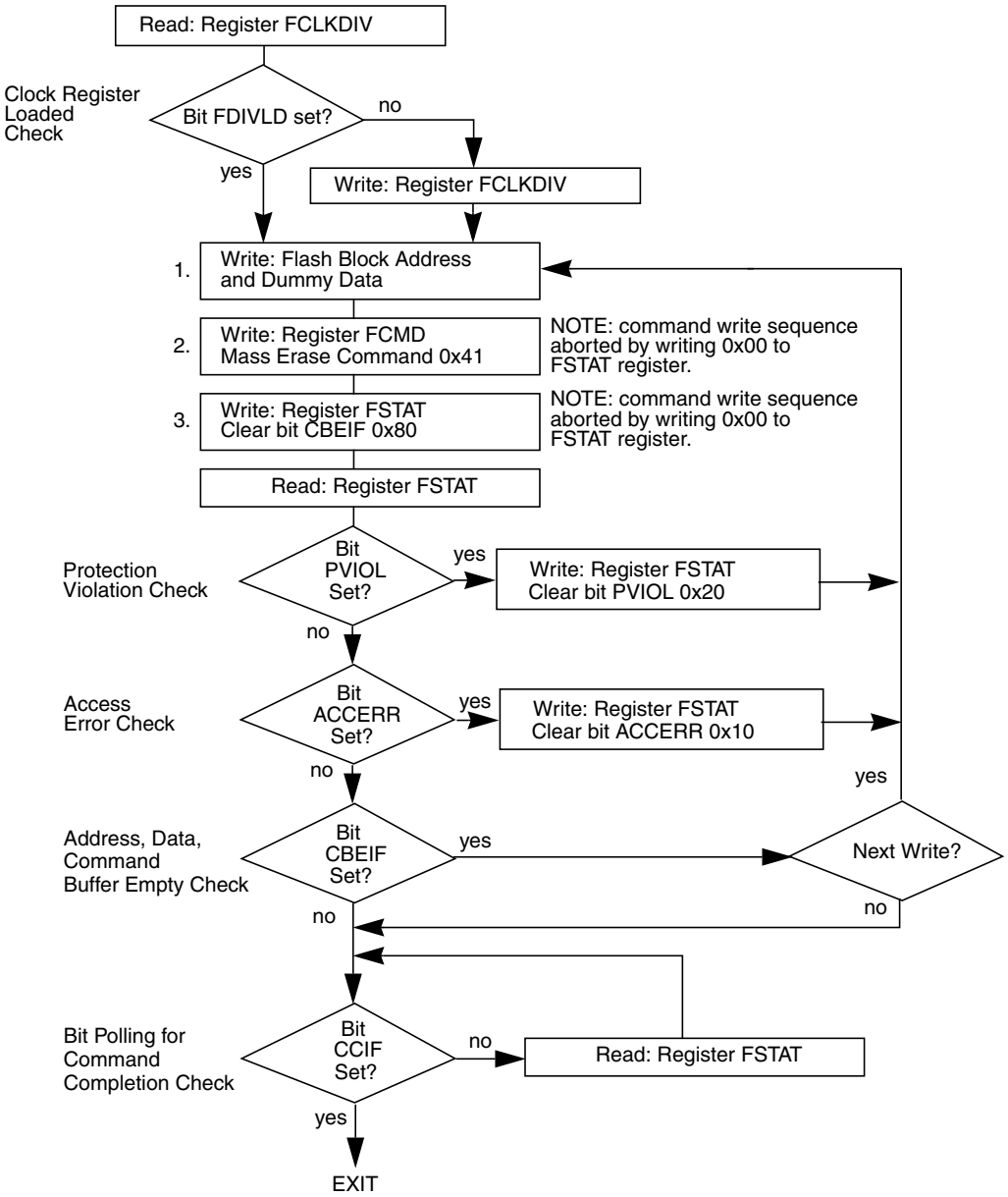


Figure 2-26. Example Mass Erase Command Flow

2.4.1.3.6 Sector Erase Abort Command

The sector erase abort command is used to terminate the sector erase operation so that other sectors in the Flash block are available for read and program operations without waiting for the sector erase operation to complete. If the sector erase abort command is launched resulting in the early termination of an active sector erase operation, the ACCERR flag will set after the operation completes as indicated by the CCIF flag being set. The ACCERR flag sets to inform the user that the sector may not be fully erased and a new sector erase command must be launched before programming any location in that specific sector. If the sector erase abort command is launched but the active sector erase operation completes normally, the ACCERR flag will not set upon completion of the operation as indicated by the CCIF flag being set. Therefore, if the ACCERR flag is not set after the sector erase abort command has completed, the sector being erased when the abort command was launched is fully erased. The maximum number of cycles required to abort a sector erase operation is equal to four FCLK periods (see Section 2.4.1.1, “Writing the FCLKDIV Register”) plus five bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set.

NOTE

Since the ACCERR bit in the FSTAT register may be set at the completion of the sector erase abort operation, a command write sequence is not allowed to be buffered behind a sector erase abort command write sequence. The CBEIF flag will not set after launching the sector erase abort command to indicate that a command must not be buffered behind it. If an attempt is made to start a new command write sequence with a sector erase abort operation active, the ACCERR flag in the FSTAT register will be set. A new command write sequence may be started after clearing the ACCERR flag, if set.

NOTE

The sector erase abort command must be used sparingly because a sector erase operation that is aborted counts as a complete program/erase cycle.

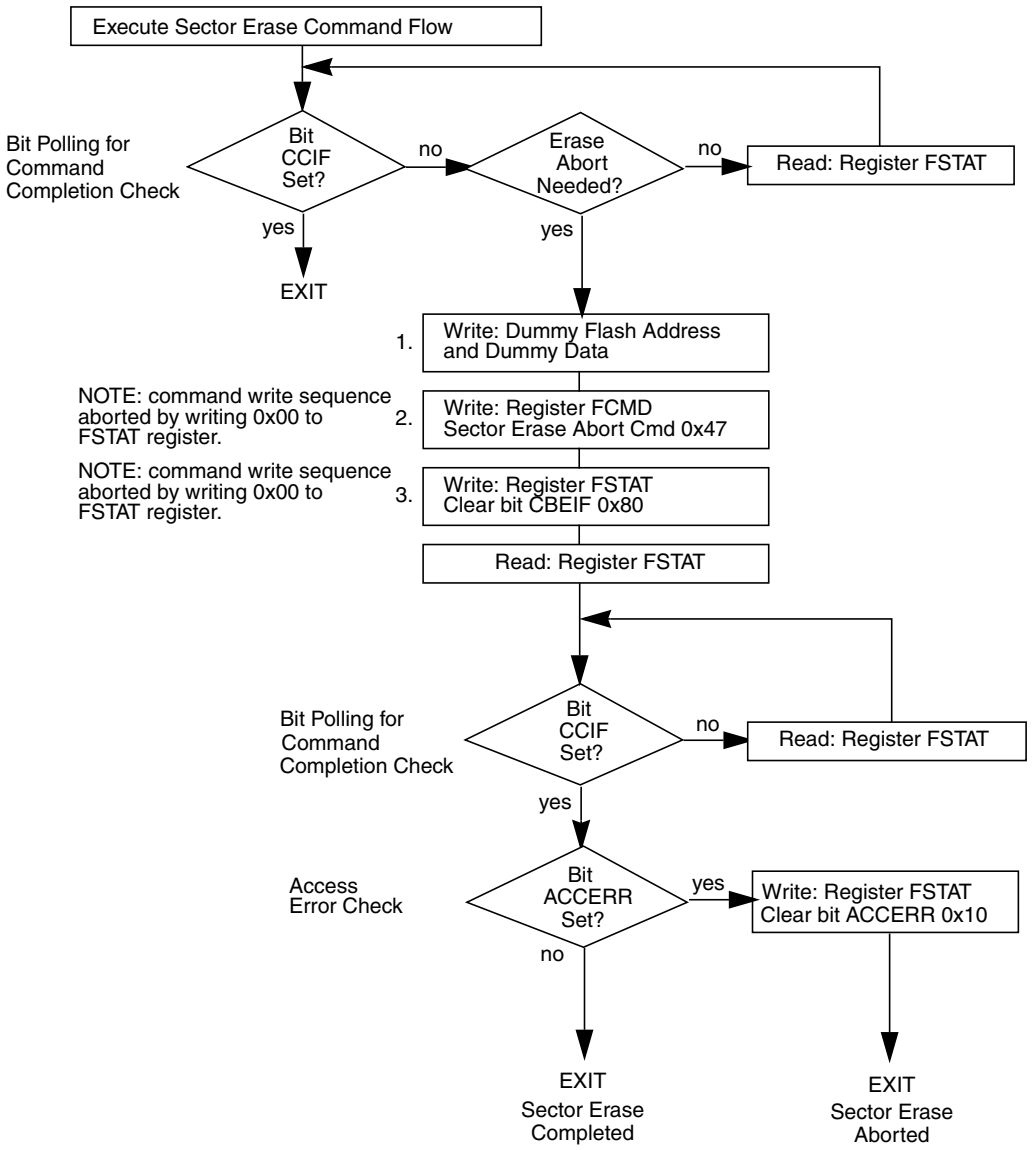


Figure 2-27. Example Sector Erase Abort Command Flow

2.4.1.4 Illegal Flash Operations

The ACCERR flag will be set during the command write sequence if any of the following illegal steps are performed, causing the command write sequence to immediately abort:

1. Writing to a Flash address before initializing the FCLKDIV register.
2. Writing a byte or misaligned word to a valid Flash address.
3. Starting a command write sequence while a data compress operation is active.
4. Starting a command write sequence while a sector erase abort operation is active.
5. Writing a second word to a Flash address in the same command write sequence.
6. Writing to any Flash register other than FCMD after writing a word to a Flash address.
7. Writing a second command to the FCMD register in the same command write sequence.
8. Writing an invalid command to the FCMD register.
9. When security is enabled, writing a command other than mass erase to the FCMD register when the write originates from a non-secure memory location or from the Background Debug Mode.
10. Writing to any Flash register other than FSTAT (to clear CBEIF) after writing to the FCMD register.
11. Writing a 0 to the CBEIF flag in the FSTAT register to abort a command write sequence.

The ACCERR flag will not be set if any Flash register is read during a valid command write sequence.

The ACCERR flag will also be set if any of the following events occur:

1. Launching the sector erase abort command while a sector erase operation is active which results in the early termination of the sector erase operation (see Section 2.4.1.3.6, “Sector Erase Abort Command”)
2. The MCU enters stop mode and a program or erase operation is in progress. The operation is aborted immediately and any pending command is purged (see Section 2.5.2, “Stop Mode”).

If the Flash memory is read during execution of an algorithm (i.e., CCIF flag in the FSTAT register is low), the read operation will return invalid data and the ACCERR flag will not be set.

If the ACCERR flag is set in the FSTAT register, the user must clear the ACCERR flag before starting another command write sequence (see Section 2.3.2.7, “Flash Status Register (FSTAT)”).

The PVIOL flag will be set after the command is written to the FCMD register during a command write sequence if any of the following illegal operations are attempted, causing the command write sequence to immediately abort:

1. Writing the program command if the address written in the command write sequence was in a protected area of the Flash memory.
2. Writing the sector erase command if the address written in the command write sequence was in a protected area of the Flash memory.
3. Writing the mass erase command while any Flash protection is enabled.

If the PVIOL flag is set in the FSTAT register, the user must clear the PVIOL flag before starting another command write sequence (see Section 2.3.2.7, “Flash Status Register (FSTAT)”).

2.5 Operating Modes

2.5.1 Wait Mode

If a command is active (CCIF = 0) when the MCU enters wait mode, the active command and any buffered command will be completed.

The Flash module can recover the MCU from wait mode if the CBEIF and CCIF interrupts are enabled (Section 2.8, “Interrupts”).

2.5.2 Stop Mode

If a command is active (CCIF = 0) when the MCU enters stop mode, the operation will be aborted and, if the operation is program or erase, the Flash array data being programmed or erased may be corrupted and the CCIF and ACCERR flags will be set. If active, the high voltage circuitry to the Flash memory will immediately be switched off when entering stop mode. Upon exit from stop mode, the CBEIF flag is set and any buffered command will not be launched. The ACCERR flag must be cleared before starting a command write sequence (see Section 2.4.1.2, “Command Write Sequence”).

NOTE

As active commands are immediately aborted when the MCU enters stop mode, it is strongly recommended that the user does not use the STOP instruction during program or erase operations.

2.5.3 Background Debug Mode

In background debug mode (BDM), the FPROT register is writable. If the MCU is unsecured, then all Flash commands listed in Table 2-18 can be executed.

2.6 Flash Module Security

The Flash module provides the necessary security information to the MCU. After each reset, the Flash module determines the security state of the MCU as defined in Section 2.3.2.2, “Flash Security Register (FSEC)”.

The contents of the Flash security byte at 0xFF0F in the Flash configuration field must be changed directly by programming 0xFF0F when the MCU is unsecured and the higher address sector is unprotected. If the Flash security byte remains in a secured state, any reset will cause the MCU to initialize to a secure operating mode.

2.6.1 Unsecuring the MCU using Backdoor Key Access

The MCU may be unsecured by using the backdoor key access feature which requires knowledge of the contents of the backdoor keys (four 16-bit words programmed at addresses 0xFF00–0xFF07). If the KEYEN[1:0] bits are in the enabled state (see Section 2.3.2.2, “Flash Security Register (FSEC)”) and the

KEYACC bit is set, a write to a backdoor key address in the Flash memory triggers a comparison between the written data and the backdoor key data stored in the Flash memory. If all four words of data are written to the correct addresses in the correct order and the data matches the backdoor keys stored in the Flash memory, the MCU will be unsecured. The data must be written to the backdoor keys sequentially starting with 0xFF00–0xFF01 and ending with 0xFF06–0xFF07. 0x0000 and 0xFFFF are not permitted as backdoor keys. While the KEYACC bit is set, reads of the Flash memory will return invalid data.

The user code stored in the Flash memory must have a method of receiving the backdoor key from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN[1:0] bits are in the enabled state (see Section 2.3.2.2, “Flash Security Register (FSEC)”), the MCU can be unsecured by the backdoor access sequence described below:

1. Set the KEYACC bit in the Flash configuration register (FCNFG).
2. Write the correct four 16-bit words to Flash addresses 0xFF00–0xFF07 sequentially starting with 0xFF00.
3. Clear the KEYACC bit.
4. If all four 16-bit words match the backdoor keys stored in Flash addresses 0xFF00–0xFF07, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 1:0.

The backdoor key access sequence is monitored by an internal security state machine. An illegal operation during the backdoor key access sequence will cause the security state machine to lock, leaving the MCU in the secured state. A reset of the MCU will cause the security state machine to exit the lock state and allow a new backdoor key access sequence to be attempted. The following operations during the backdoor key access sequence will lock the security state machine:

1. If any of the four 16-bit words does not match the backdoor keys programmed in the Flash array.
2. If the four 16-bit words are written in the wrong sequence.
3. If more than four 16-bit words are written.
4. If any of the four 16-bit words written are 0x0000 or 0xFFFF.
5. If the KEYACC bit does not remain set while the four 16-bit words are written.
6. If any two of the four 16-bit words are written on successive MCU clock cycles.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the Flash security byte can be programmed to the unsecure state, if desired.

In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0xFF00–0xFF07 in the Flash configuration field.

The security as defined in the Flash security byte (0xFF0F) is not changed by using the backdoor key access sequence to unsecure. The backdoor keys stored in addresses 0xFF00–0xFF07 are unaffected by the backdoor key access sequence. After the next reset of the MCU, the security state of the Flash module is determined by the Flash security byte (0xFF0F). The backdoor key access sequence has no effect on the program and erase protections defined in the Flash protection register.

It is not possible to unsecure the MCU in special single-chip mode by using the backdoor key access sequence via the background debug mode (BDM).

2.6.2 Unsecuring the Flash Module in Special Single-Chip Mode using BDM

The MCU can be unsecured in special single-chip mode by erasing the Flash module by the following method :

- Reset the MCU into special single-chip mode, delay while the erase test is performed by the BDM secure ROM, send BDM commands to disable protection in the Flash module, and execute a mass erase command write sequence to erase the Flash memory.

After the CCIF flag sets to indicate that the mass operation has completed, reset the MCU into special single-chip mode. The BDM secure ROM will verify that the Flash memory is erased and will assert the UNSEC bit in the BDM status register. This BDM action will cause the MCU to override the Flash security state and the MCU will be unsecured. All BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state by the following method:

- Send BDM commands to execute a word program sequence to program the Flash security byte to the unsecured state and reset the MCU.

2.7 Resets

2.7.1 Flash Reset Sequence

On each reset, the Flash module executes a reset sequence to hold CPU activity while loading the following registers from the Flash memory according to Table 2-1:

- FPROT — Flash Protection Register (see Section 2.3.2.5).
- FCTL — Flash Control Register (see Section 2.3.2.9).
- FSEC — Flash Security Register (see Section 2.3.2.2).

2.7.2 Reset While Flash Command Active

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector / block being erased is not guaranteed.

2.8 Interrupts

The Flash module can generate an interrupt when all Flash command operations have completed, when the Flash address, data, and command buffers are empty.

Table 2-19. Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash address, data and command buffers empty	CBEIF (FSTAT register)	CBEIE (FCNFG register)	I Bit
All Flash commands completed	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

2.8.1 Description of Flash Interrupt Operation

The logic used for generating interrupts is shown in Figure 2-28.

The Flash module uses the CBEIF and CCIF flags in combination with the CBIE and CCIE enable bits to generate the Flash command interrupt request.

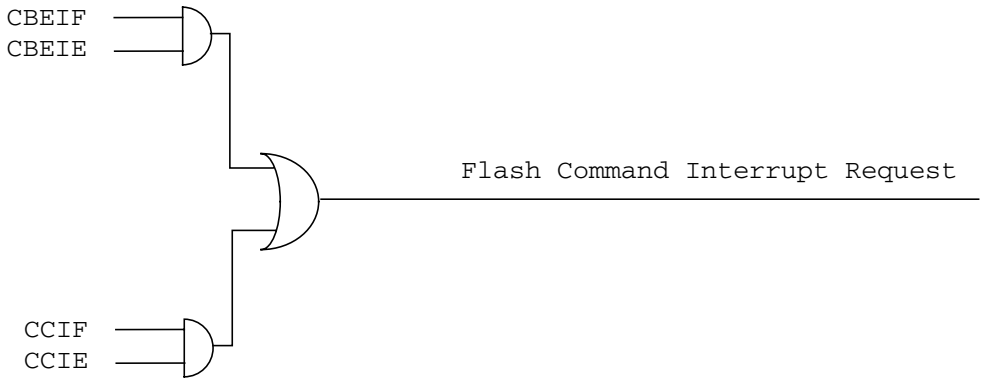


Figure 2-28. Flash Interrupt Implementation

For a detailed description of the register bits, refer to Section 2.3.2.4, “Flash Configuration Register (FCNFG)” and Section 2.3.2.7, “Flash Status Register (FSTAT)”.

Chapter 3
Port Integration Module (PIM9NE64V1)

3.1 Introduction

Figure 3-1 is a block diagram of the PIM_9NE64.

The port integration module establishes the interface between the peripheral modules and the I/O pins for all ports.

- This section covers:
- port A, B, E, and K related to the core logic and multiplexed bus interface
- port T connected to the timer module
- port S associated with 2 SCI and 1 SPI modules
- port G, H, and J connected to EMAC module, each of them also can be used as an external interrupt source.
- port L connected to EPHY module

Each I/O pin can be configured by several registers: Input/output selection, drive strength reduction, enable and select of pull resistors, interrupt enable and status flags.

The implementation of the port integration module is device dependent.

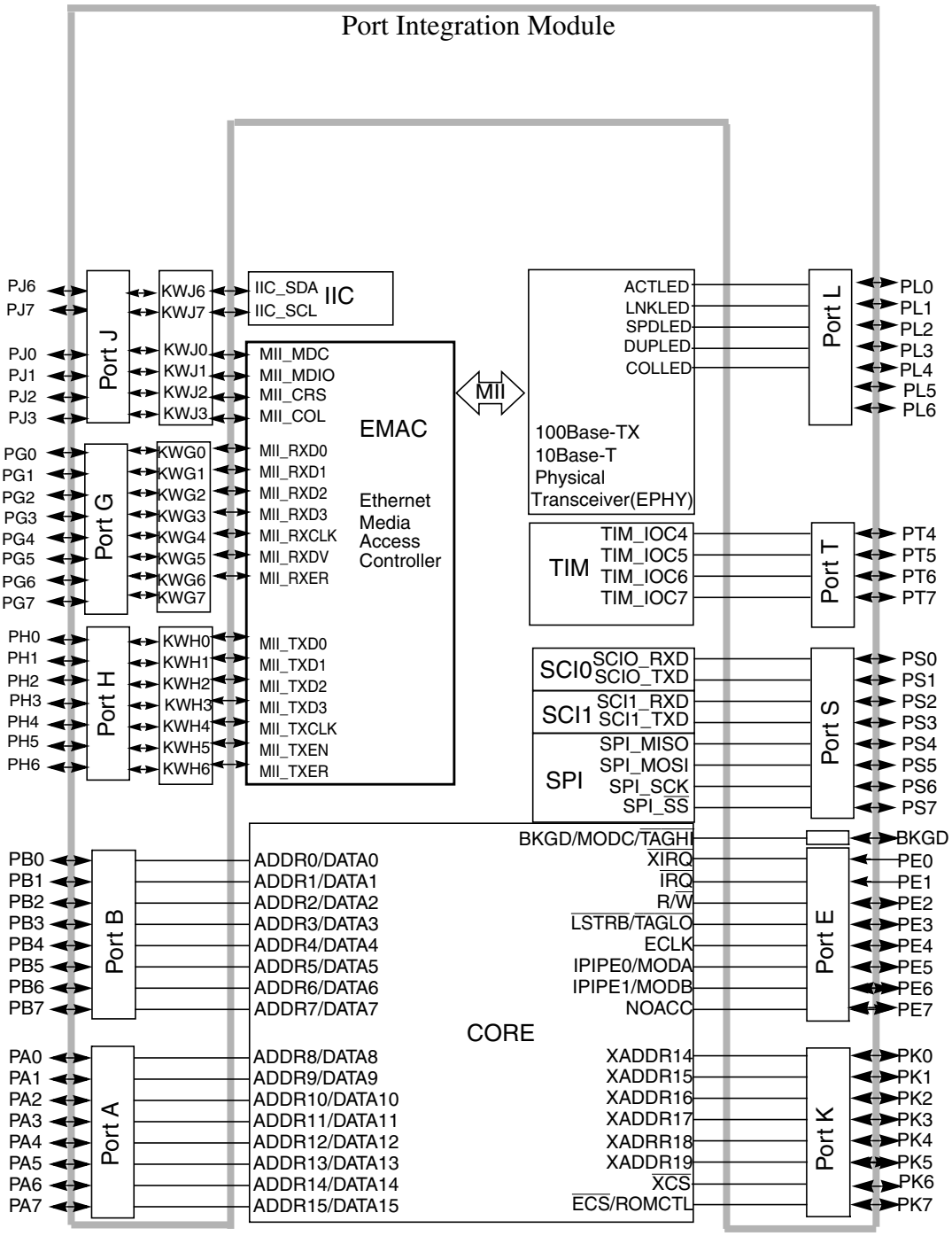


Figure 3-1. PIM_9NE64 Block Diagram

3.1.1 Features

A standard port has the following minimum features:

- Input/output selection

- 3.3 V output drive with two selectable drive strength
- 3.3 V digital and analog input
- Input with selectable pull-up or pull-down device

Optional features:

- Open drain for wired-or connections
- Interrupt inputs with glitch filtering

3.2 External Signal Description

This section lists and describes the signals that connect off-chip.

Table 3-1 shows all pins and their functions that are controlled by the PIM_9NE64 module. If there is more than one function associated to a pin, the priority is indicated by the position in the table from top (highest priority) to bottom (lowest priority).

Table 3-1. Pin Functions and Priorities (Sheet 1 of 4)

Port	Pin Name	Pin Function	Description	Pin Function after Reset
Port A	PA[7:0]	ADDR[15:8]/ DATA[15:8]/ GPIO	Refer the MEBI block description chapter.	
Port B	PB[7:0]	ADDR[7:0]/ DATA[7:0]/ GPIO	Refer the MEBI block description chapter.	
Port E	PE7	NOACC/ GPIO	Refer the MEBI block description chapter.	
	PE6	IPIPE1/ MODB/ GPIO		
	PE5	IPIPE0/ MODA/ GPIO		
	PE4	ECLK/GPIO		
	PE3	$\overline{\text{LSTRB}}$/ $\overline{\text{TAGLO}}$/ GPIO		
	PE2	R/$\overline{\text{W}}$ / GPIO		
	PE1	$\overline{\text{IRQ}}$/GPI		
	PE0	$\overline{\text{XIRQ}}$/GPI		

Table 3-1. Pin Functions and Priorities (Sheet 2 of 4)

Port	Pin Name	Pin Function	Description	Pin Function after Reset
Port K	PK7	$\overline{ECS}$/ ROMCTL/ GPIO	Refer to the MEBI block description chapters.	
	PK6	$\overline{XCS}$		
	PK[5:0]	XADDR[19:14]/ GPIO		
—	BKGD	BKGD/ MODC/ $\overline{TAGHI}$	Refer to the MEBI and BDM block description chapters.	
Port G	PG[7]	KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	GPIO
	PG[6]	MII_RXER	MII Receive Coding Error	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PG[5]	MII_RXDV	MII Receive Data Valid	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PG[4]	MII_RXCLK	MII Receive Clock	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PG[3:0]	MII_RXD[3:0]	MII Receive Data	
		KWU/GPIO	Key board wake up Interrupts or General-purpose I/O	
Port H	PH[6]	MII_TXER	MII Transmit Coding Error	GPIO
		KWU/GPIO	Key board wake up Interrupts or General-purpose I/O	
	PH[5]	MII_TXEN	MII Transmit Enable	
		KWU/GPIO	Key board wake up Interrupts or General-purpose I/O	
	PH[4]	MII_TXCLK	MII Transmit Clock	
		KWU/GPIO	Key board wake up Interrupts or General-purpose I/O	
	PH[3:0]	MII_TXD[3:0]	MII Transmit Data	
		KWU/GPIO	Key board wake up Interrupts or General-purpose I/O	

Table 3-1. Pin Functions and Priorities (Sheet 3 of 4)

Port	Pin Name	Pin Function	Description	Pin Function after Reset
Port J	PJ[7]	IIC_SCL	Serial Clock Line bidirectional pin of IIC module	GPIO
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PJ[6]	IIC_SDA	Serial Data Line bidirectional pin of IIC module	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PJ[3]	MII_COL	MII Collision	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PJ[2]	MII_CRS	MII Carrier Sense	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PJ[1]	MII_MDIO	MII Management Data Input/Output	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
	PJ[0]	MII_MDC	MII Management Data Clock	
		KWU/GPIO	Key board wake up Interrupt or General-purpose I/O	
Port L	PL[6]	GPIO	General-purpose I/O	GPIO
	PL[5]	GPIO	General-purpose I/O	
	PL[4]	COLLED	EPHY Collision LED	
		GPIO	General-purpose I/O	
	PL[3]	DUPLED	EPHY Duplex LED	
		GPIO	General-purpose I/O	
	PL[2]	SPDLED	EPHY Speed LED	
		GPIO	General-purpose I/O	
	PL[1]	LNKLED	EPHY Link LED	
		GPIO	General-purpose I/O	
	PL[0]	ACTLED	EPHY Active LED	
		GPIO	General-purpose I/O	

Table 3-1. Pin Functions and Priorities (Sheet 4 of 4)

Port	Pin Name	Pin Function	Description	Pin Function after Reset
Port S	PS[7]	SPI_$\overline{SS}$	Serial Peripheral Interface slave select output in master mode, input in slave mode or master mode.	GPIO
		GPIO	General-purpose I/O	
	PS[6]	SPI_SCK	Serial Peripheral Interface serial clock pin	
		GPIO	General-purpose I/O	
	PS[5]	SPI_MOSI	Serial Peripheral Interface master out/slave in pin	
		GPIO	General-purpose I/O	
	PS[4]	SPI_MISO	Serial Peripheral Interface master in/slave out pin	
		GPIO	General-purpose I/O	
	PS[3]	SCI1_TXD	Serial Communication Interface 1 transmit pin	
		GPIO	General-purpose I/O	
	PS[2]	SCI1_RXD	Serial Communication Interface 1 receive pin	
		GPIO	General-purpose I/O	
	PS[1]	SCI0_TXD	Serial Communication Interface 0 transmit pin	
		GPIO	General-purpose I/O	
	PS[0]	SCI0_RXD	Serial Communication Interface 0 receive pin	
		GPIO	General-purpose I/O	
Port T	PT[7:4]	IOC[7:4]	Standard Timer1 Channels 7 to 4	GPIO
		GPIO	General-purpose I/O	

3.3 Memory Map and Register Descriptions

This section provides a detailed description of all registers.

3.3.1 Module Memory Map

Table 3-2 shows the memory map of the port integration module.

Table 3-2. PIM Module Memory Map

Address Offset	Use	Access
$00	Port T I/O Register (PTT)	R/W
$01	Port T Input Register (PTIT)	R
$02	Port T Data Direction Register (DDRT)	R/W
$03	Port T Reduced Drive Register (RDRT)	R/W
$04	Port T Pull Device Enable Register (PERT)	R/W
$05	Port T Polarity Select Register (PPST)	R/W

Table 3-2. PIM Module Memory Map (continued)

$06-07	Reserved	—
$08	Port S I/O Register (PTS)	R/W
$09	Port S Input Register (PTIS)	R
$0A	Port S Data Direction Register (DDRS)	R/W
$0B	Port S Reduced Drive Register (RDRS)	R/W
$0C	Port S Pull Device Enable Register (PERS)	R/W
$0D	Port S Polarity Select Register (PPSS)	R/W
$0E	Port S Wired-Or Mode Register (WOMS)	R/W
$0F	Reserved	—
$10	Port G I/O Register (PTG)	R/W
$11	Port G Input Register (PTIG)	R
$12	Port G Data Direction Register (DDRG)	R/W
$13	Port G Reduced Drive Register (RDRG)	R/W
$14	Port G Pull Device Enable Register (PERG)	R/W
$15	Port G Polarity Select Register (PPSG)	R/W
$16	Port G Interrupt Enable Register (PIEG)	R/W
$17	Port G Interrupt Flag Register (PIFG)	R/W
$18	Port H I/O Register (PTH)	R/W
$19	Port H Input Register (PTIH)	R
$1A	Port H Data Direction Register (DDRH)	R/W
$1B	Port H Reduced Drive Register (RDRH)	R/W
$1C	Port H Pull Device Enable Register (PERH)	R/W
$1D	Port H Polarity Select Register (PPSH)	R/W
$1E	Port H Interrupt Enable Register (PIEH)	R/W
$1F	Port H Interrupt Flag Register (PIFH)	R/W
$20	Port J I/O Register (PTJ)	R/W
$21	Port J Input Register (PTIJ)	R
$22	Port J Data Direction Register (DDRJ)	R/W
$23	Port J Reduced Drive Register (RDRJ)	RW
$24	Port J Pull Device Enable Register (PERJ)	R/W
$25	Port J Polarity Select Register (PPSJ)	R/W
$26	Port J Interrupt Enable Register (PIEJ)	R/W
$27	Port J Interrupt Flag Register (PIFJ)	R/W
$28	Port L I/O Register (PTL)	R/W
$29	Port L Input Register (PTIL)	R
$2A	Port L Data Direction Register (DDRL)	R/W
$2B	Port L Reduced Drive Register (RDRL)	R/W
$2C	Port L Pull Device Enable Register (PERL)	R/W
$2D	Port L Polarity Select Register (PPSL)	R/W
$2E	Port L Wired-Or Mode Register (WOML)	RW
$2F-$3F	Reserved	—

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

3.3.2 Register Descriptions

The following table summarizes the effect on the various configuration bits - data direction (DDR), input / output level (I/O), reduced drive (RDR), pull enable (PE), pull select (PS) and interrupt enable (IE) for the ports. The configuration bit PS is used for two purposes:

1. Configure the sensitive interrupt edge (rising or falling), if interrupt is enabled.
2. Select either a pull-up or pull-down device if PE is active.

Table 3-3. Pin Configuration Summary

DDR	IO	RDR	PE	PS	IE[1]	Function	Pull Device	Interrupt
0	X	X	0	X	0	Input	Disabled	Disabled
0	X	X	1	0	0	Input	Pull Up	Disabled
0	X	X	1	1	0	Input	Pull Down	Disabled
0	X	X	0	0	1	Input	Disabled	Falling edge
0	X	X	0	1	1	Input	Disabled	Rising edge
0	X	X	1	0	1	Input	Pull Up	Falling edge
0	X	X	1	1	1	Input	Pull Down	Rising edge
1	0	0	X	X	0	Output, full drive to 0	Disabled	Disabled
1	1	0	X	X	0	Output, full drive to 1	Disabled	Disabled
1	0	1	X	X	0	Output, reduced drive to 0	Disabled	Disabled
1	1	1	X	X	0	Output, reduced drive to 1	Disabled	Disabled
1	0	0	X	0	1	Output, full drive to 0	Disabled	Falling edge
1	1	0	X	1	1	Output, full drive to 1	Disabled	Rising edge
1	0	1	X	0	1	Output, reduced drive to 0	Disabled	Falling edge
1	1	1	X	1	1	Output, reduced drive to 1	Disabled	Rising edge

[1] Applicable only on ports G, H, and J.

NOTE

All bits of all registers in this module are completely synchronous to internal clocks during a register read.

3.3.2.1 Port T Registers

3.3.2.1.1 I/O Register (PTT)

Module Base + $0

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTT7	PTT6	PTT5	PTT4	0	0	0	0
Write								
TIM	IOC7	IOC6	IOC5	IOC4				
Reset:	0	0	0	0	—	—	—	—

= Reserved or unimplemented

Figure 3-2. Port T I/O Register (PTT)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The standard timer module (TIM) can be configured to use the PT[7:4] as timer input capture/output compare pins. If IOC[7:4]-channel is defined as output, the related port T is assigned to IOC function.

3.3.2.1.2 Input Register (PTIT)

Module Base + $1

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIT7	PTIT6	PTIT5	PTIT4	0	0	0	0
Write:								
Reset:	—	—	—	—	—	—	—	—

= Reserved or unimplemented

Figure 3-3. Port T Input Register (PTIT)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

3.3.2.1.3 Data Direction Register (DDRT)

Module Base + $2

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DDRT7	DDRT6	DDRT5	DDRT4	0	0	0	0
Write:								
Reset:	0	0	0	0	—	—	—	—

= Reserved or unimplemented

Figure 3-4. Port T Data Direction Register (DDRT)

Read:Anytime.

Write:Anytime.

This register configures each port T pin as either input or output.
The standard TIM module forces the I/O state to be an output for each port pin associated with an enabled output compare. When the pin is configured as an output compare the corresponding data direction register (DDRT) bits do not have any effect on the I/O direction of the pin, and will maintain their previously latched value. The DDRT bits revert to controlling the I/O direction of a pin when the associated timer output compare is disabled. If a pin is being used as a timer input capture, the DDRT remains in control of the pin's I/O direction and the timer monitors the state of the pin.

DDRT[7:4] — Data Direction Port T

1 = Associated pin is configured as output.
0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTT or PTIT registers, when changing the DDRT register.

3.3.2.1.4 Reduced Drive Register (RDRT)

Module Base + $3

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RDRT7	RDRT6	RDRT5	RDRT4	0	0	0	0
Write:								
Reset:	0	0	0	0	—	—	—	—

= Reserved or unimplemented

Figure 3-5. Port T Reduced Drive Register (RDRT)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port T output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRT[7:4] — Reduced Drive Port T

1 = Associated pin drives at about 1/3 of the full drive strength.
0 = Full drive strength at output.

3.3.2.1.5 Pull Device Enable Register (PERT)

Module Base + $4

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PERT7	PERT6	PERT5	PERT4	0	0	0	0
Write:								
Reset:	0	0	0	0	—	—	—	—

= Reserved or unimplemented

Figure 3-6. Port T Pull Device Enable Register (PERT)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input. This bit has no effect if the port is used as output. Out of reset no pull device is enabled.

PERT[7:4] — Pull Device Enable Port T

1 = Either a pull-up or pull-down device is enabled.
0 = Pull-up or pull-down device is disabled.

3.3.2.1.6 Polarity Select Register (PPST)

Module Base + $5

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PPST7	PPST6	PPST5	PPST4	0	0	0	0
Write:								
Reset:	0	0	0	0	—	—	—	—

= Reserved or unimplemented

Figure 3-7. Port T Polarity Select Register (PPST)

Read:Anytime.

Write:Anytime.

This register selects whether a pull-down or a pull-up device is connected to the pin.

PPST[7:4] — Pull Select Port T

1 = A pull-down device is connected to the associated port T pin, if enabled by the associated bit in register PERT and if the port is used as input.
0 = A pull-up device is connected to the associated port T pin, if enabled by the associated bit in register PERT and if the port is used as input.

3.3.2.2 Port S Registers

3.3.2.2.1 I/O Register (PTS)

Module Base + $8

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
SPI	SS	SCK	MOSI	MISO	—	—	—	—
SCI	—	—	—	—	SCI1_TXD	SCI1_RXD	SCI0_TXD	SCI0_RXD
Reset:	0	0	0	0	0	0	0	0

Figure 3-8. Port S I/O Register (PTS)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The SPI function takes precedence over the general-purpose I/O function if the SPI module is enabled. If the SPI is enabled the PS[7:4] pins become SPI_$\overline{SS}$, SPI_SCK, SPI_MOSI, and SPI_MISO, and their configuration is determined by several status bits in the SPI module. Refer to the SPI block description chapter for details.

The SCI1 and SCI0 function take precedence over the general-purpose I/O function on pins PS[3:0]. If the SCI1 or SCI0 transmitters or receivers are enabled, the SCI1 and SCI0 transmit pins, SCI1_TXD and SCI0_TXD, are configured as outputs if the corresponding transmitter is enabled. The SCI1 and SCI0 receive pins, SCI1_RXD and SCI0_RXD, are configured as inputs if the corresponding receiver is enabled. Refer to the SCI block description chapter for details.

3.3.2.2.2 Input Register (PTIS)

Module Base + $9

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
Write:								
Reset:	—	—	—	—	—	—	—	—

= Reserved or unimplemented

Figure 3-9. Port S Input Register (PTIS)

Read:Anytime.

Write: writes to this register have no effect.

This register always reads back the status of the associated pins. This also can be used to detect overload or short circuit conditions on output pins.

3.3.2.2.3 Data Direction Register (DDRS)

Module Base + $A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
Reset:	0	0	0	0	0	0	0	0

Figure 3-10. Port S Data Direction Register (DDRS)

Read:Anytime.

Write:Anytime.

This register configures each port S pin as either input or output.

If the SPI is enabled, the SPI controls the SPI related pins (SPI_$\overline{SS}$, SPI_SCK, SPI_MOSI, SPI_MISO) I/O direction, and the corresponding DDRS[7:4] bits have no effect on the SPI pins I/O direction. Refer to the SPI block description chapter for details.

When the SCI0 or SCI1 transmitters are enabled, the corresponding transmit pins, SCI0_TxD and SCI0_TxD, I/O direction is controlled by the SCI0 and SCI1 respectively, and the corresponding DDRS3 and DDRS1 bits have no effect on their I/O direction. When the SCI0 or SCI1 receivers are enabled, the corresponding receive pins, SCI0_RXD and SCI1_RXD, I/O direction is controlled by the SCI0 and SCI1 respectively, and the DDRS2 and DDRS0 bits have no effect on their I/O direction. Refer to the SCI block description chapter for further details.

The DDRS[7:0] bits revert to controlling the I/O direction of the pins when the associated SPI or SCI function is disabled.

DDRS[7:0] — Data Direction Port S

1 = Associated pin is configured as output.
0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTS or PTIS registers, when changing the DDRS register.

3.3.2.2.4 Reduced Drive Register (RDRS)

Module Base + $B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
Reset:	0	0	0	0	0	0	0	0

Figure 3-11. Port S Reduced Drive Register (RDRS)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port S output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRS[7:0] — Reduced Drive Port S

1 = Associated pin drives at about 1/3 of the full drive strength.
0 = Full drive strength at output.

3.3.2.2.5 Pull Device Enable Register (PERS)

Module Base + $C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
Reset:	1	1	1	1	1	1	1	1

Figure 3-12. Port S Pull Device Enable Register (PERS)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input or as output in wired-or (open drain) mode. These bits have no effect if the port is used as push-pull output. Out of reset a pull-up device is enabled.

PERS[7:0] — Pull Device Enable Port S

1 = Either a pull-up or pull-down device is enabled.
0 = Pull-up or pull-down device is disabled.

3.3.2.2.6 Polarity Select Register (PPSS)

Module Base + $D

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
Reset:	0	0	0	0	0	0	0	0

Figure 3-13. Port S Polarity Select Register (PPSS)

Read:Anytime.

Write:Anytime.

This register selects whether a pull-down or a pull-up device is connected to the pin.

PPSS[7:0] — Pull Select Port S

1 = A pull-down device is connected to the associated port S pin, if enabled by the associated bit in register PERS and if the port is used as input.
0 = A pull-up device is connected to the associated port S pin, if enabled by the associated bit in PERS register and if the port is used as input or as wired-or output.

3.3.2.2.7 Wired-Or Mode Register (WOMS)

Module Base + $E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
Reset:	0	0	0	0	0	0	0	0

Figure 3-14. Port S Wired-Or Mode Register (WOMS)

Read:Anytime.

Write:Anytime.

This register configures the output pins as wired-or. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven. It applies also to the SPI and SCI outputs and allows a multipoint connection of several serial modules. These bits have no influence on pins used as inputs.

WOMS[7:0] — Wired-Or Mode Port S

1 = Open-drain mode enabled for output buffers.
0 = Open-drain mode disabled for output buffers.

3.3.2.3 Port G Registers

3.3.2.3.1 I/O Register (PTG)

Module Base + $10

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTG7	PTG6	PTG5	PTG4	PTG3	PTG2	PTG1	PTG0
EMAC	—	MII_RXER	MII_RXDV	MII_RXCLK	MII_RXD3	MII_RXD2	MII_RXD1	MII_RXD0
KWU	KWG							
Reset:	—	0	0	0	0	0	0	0

Figure 3-15. Port G I/O Register (PTG)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read. The EMAC MII external interface takes precedence over general-purpose I/O function if the EMAC module is enabled in external PHY mode. If the EMAC is enabled PG[6:0] pins become inputs MII_RXER, MII_RXDV, MII_RXCLK, MII_RXD[3:0]. Please refer to the EMAC block description chapter for details.

3.3.2.3.2 Input Register (PTIG)

Module Base + $11

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIG7	PTIG6	PTIG5	PTIG4	PTIG3	PTIG2	PTIG1	PTIG0
Write:								
Reset:	—	—	—	—	—	—	—	—

= Reserved or unimplemented

Figure 3-16. Port G Input Register (PTIG)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This also can be used to detect overload or short circuit conditions on output pins.

3.3.2.3.3 Data Direction Register (DDRG)

Module Base + $12

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRG7	DDRG6	DDRG5	DDRG4	DDRG3	DDRG2	DDRG1	DDRG0
Reset:	0	0	0	0	0	0	0	0

Figure 3-17. Port G Data Direction Register (DDRG)

Read:Anytime.

Write:Anytime.

This register configures each port G pin as either input or output.

DDRG[7:0] — Data Direction Port G

1 = Associated pin is configured as output.
0 = Associated pin is configured as input.

If the EMAC MII external interface is enabled, the pins G[6:0] are forced to be inputs and DDRG has no effect on the them. Please refer to the EMAC block description chapter for details.

The DDRG bits revert to controlling the I/O direction of a pin when the EMAC MII external interface is disabled.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTG or PTIG registers, when changing the DDRG register.

3.3.2.3.4 Reduced Drive Register (RDRG)

Module Base + $13

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	RDRG7	RDRG6	RDRG5	RDRG4	RDRG3	RDRG2	RDRG1	RDRG0
Reset:	0	0	0	0	0	0	0	0

Figure 3-18. Port G Reduced Drive Register (RDRG)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port G output pin as either full or reduced. If the port is used as input these bits are ignored.

RDRG[7:0] — Reduced Drive Port G

1 = Associated pin drives at about 1/3 of the full drive strength.
0 = Full drive strength at output.

3.3.2.3.5 Pull Device Enable Register (PERG)

Module Base + $14

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PERG7	PERG6	PERG5	PERG4	PERG3	PERG2	PERG1	PERG0
Reset:	0	0	0	0	0	0	0	0

Figure 3-19. Port G Pull Device Enable Register (PERG)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input. These bits have no effect if the port is used as output. Out of reset no pull device is enabled.

PERG[7:0] — Pull Device Enable Port G

1 = Either a pull-up or pull-down device is enabled.
0 = Pull-up or pull-down device is disabled.

3.3.2.3.6 Polarity Select Register (PPSG)

Module Base + $15

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PPSG7	PPSG6	PPSG5	PPSG4	PPSG3	PPSG2	PPSG1	PPSG0
Reset:	0	0	0	0	0	0	0	0

Figure 3-20. Port G Polarity Select Register (PPSG)

Read:Anytime.

Write:Anytime.

This register selects whether a pull-down or a pull-up device is connected to the pin.

PPSG[7:0] — Pull Select Port G

1 = Rising edge on the associated port G pin sets the associated flag bit in the PIFG register. A pull-down device is connected to the associated port G pin, if enabled by the associated bit in register PERG and if the port is used as input.
0 = Falling edge on the associated port G pin sets the associated flag bit in the PIFG register. A pull-up device is connected to the associated port G pin, if enabled by the associated bit in register PERG and if the port is used as input.

3.3.2.3.7 Interrupt Enable Register (PIEG)

Module Base + $16

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PIEG7	PIEG6	PIEG5	PIEG4	PIEG3	PIEG2	PIEG1	PIEG0
Reset:	0	0	0	0	0	0	0	0

Figure 3-21. Port G Interrupt Enable Register (PIEG)

Read:Anytime.

Write:Anytime.

This register disables or enables on a per pin basis the edge sensitive external interrupt associated with port G.

PIEG[7:0] — Interrupt Enable Port G

1 = Interrupt is enabled.
0 = Interrupt is disabled (interrupt flag masked).

3.3.2.3.8 Interrupt Flag Register (PIFG)

Module Base + $17

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PIFG7	PIFG6	PIFG5	PIFG4	PIFG3	PIFG2	PIFG1	PIFG0
Reset:	0	0	0	0	0	0	0	0

Figure 3-22. Port G Interrupt Flag Register (PIFG)

Read:Anytime.

Write:Anytime.

Each flag is set by an active edge on the associated input pin. This could be a rising or a falling edge based on the state of the PPSG register. To clear this flag, write a “1” to the corresponding bit in the PIFG register. Writing a “0” has no effect.

PIFG[7:0] — Interrupt Flags Port G

1 = Active edge on the associated bit has occurred (an interrupt will occur if the associated enable bit is set).
Writing a “1” clears the associated flag.
0 = No active edge pending.
Writing a “0” has no effect.

3.3.2.4 Port H Registers

3.3.2.4.1 I/O Register (PTH)

Module Base + $18

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTH6	PTH5	PTH4	PTH3	PTH2	PTH1	PTH0
Write:								
EMAC		MII_TXER	MII_TXEN	MII_TXCLK	MII_TXD3	MII_TXD2	MII_TXD1	MII_TXD0
KWU		KWH						
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-23. Port H I/O Register (PTH)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The EMAC MII external interface takes precedence over general-purpose I/O function if the EMAC module is enabled in external PHY mode. If the EMAC MII external interface is enabled PH[6:0] pins become MII_TXER, MII_TXEN, MII_TXCLK, MII_TXD[3:0]. Please refer to the EMAC block description chapter for details.

3.3.2.4.2 Input Register (PTIH)

Module Base + $19

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTIH6	PTIH5	PTIH4	PTIH3	PTIH2	PTIH1	PTIH0
Write:								
Reset:	—	—	—	—	—	—	—	—

= Reserved or unimplemented

Figure 3-24. Port H Input Register (PTIH)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This can be also used to detect overload or short circuit conditions on output pins.

3.3.2.4.3 Data Direction Register (DDRH)

Module Base + $1A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	DDRH6	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-25. Port H Data Direction Register (DDRH)

Read:Anytime.

Write:Anytime.

This register configures each port H pin as either input or output.
DDRH[6:0] — Data Direction Port H

1 = Associated pin is configured as output.
0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTH or PTIH registers, when changing the DDRH register.

If the EMAC MII external interface is enabled, pins PH[6:0] become MII_TXER, MII_TXEN, MII_TXCLK, MII_TXD[3:0]. In that case, DDRH[6:0] bits have no effect on their I/O direction.

3.3.2.4.4 Reduced Drive Register (RDRH)

Module Base + $1B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	RDRH6	RDRH5	RDRH4	RDRH3	RDRH2	RDRH1	RDRH0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-26. Port H Reduced Drive Register (RDRH)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port H output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRH[6:0] — Reduced Drive Port H

1 = Associated pin drives at about 1/3 of the full drive strength.
0 = Full drive strength at output.

3.3.2.4.5 Pull Device Enable Register (PERH)

Module Base + $1C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PERH6	PERH5	PERH4	PERH3	PERH2	PERH1	PERH0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-27. Port H Pull Device Enable Register (PERH)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input. These bits have no effect if the port is used as output. Out of reset no pull device is enabled.

PERH[6:0] — Pull Device Enable Port H

1 = Either a pull-up or pull-down device is enabled.
0 = Pull-up or pull-down device is disabled.

3.3.2.4.6 Polarity Select Register (PPSH)

Module Base + $1D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PPSH6	PPSH5	PPSH4	PPSH3	PPSH2	PPSH1	PPSH0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-28. Port H Polarity Select Register (PPSH)

Read:Anytime.

Write:Anytime.

This register serves a dual purpose by selecting the polarity of the active interrupt edge as well as selecting a pull-up or pull-down device if enabled.

PPSH[6:0] — Pull Select Port H

1 = Rising edge on the associated port H pin sets the associated flag bit in the PIFH register.A pull-down device is connected to the associated port H pin, if enabled by the associated bit in register PERH and if the port is used as input.
0 = Falling edge on the associated port H pin sets the associated flag bit in the PIFH register.A pull-up device is connected to the associated port H pin, if enabled by the associated bit in register PERH and if the port is used as input.

3.3.2.4.7 Interrupt Enable Register (PIEH)

Module Base + $1E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PIEH6	PIEH5	PIEH4	PIEH3	PIEH2	PIEH1	PIEH0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-29. Port H Interrupt Enable Register (PIEH)

Read:Anytime.

Write:Anytime.

This register disables or enables on a per pin basis the edge sensitive external interrupt associated with port H.

PIEH[6:0] — Interrupt Enable Port H

1 = Interrupt is enabled.
0 = Interrupt is disabled (interrupt flag masked).

3.3.2.4.8 Interrupt Flag Register (PIFH)

Module Base + $1F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PIFH6	PIFH5	PIFH4	PIFH3	PIFH2	PIFH1	PIFH0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-30. Port H Interrupt Flag Register (PIFH)

Read:Anytime.

Write:Anytime.

Each flag is set by an active edge on the associated input pin. This could be a rising or a falling edge based on the state of the PPSH register. To clear this flag, write a “1” to the corresponding bit in the PIFH register. Writing a “0” has no effect.

PIFH[6:0] — Interrupt Flags Port H

1 = Active edge on the associated bit has occurred (an interrupt will occur if the associated enable bit is set).
Writing a “1” clears the associated flag.
0 = No active edge pending.
Writing a “0” has no effect.

3.3.2.5 Port J Registers

3.3.2.5.1 I/O Register (PTJ)

Module Base + $20

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTJ7	PTJ6	0	0	PTJ3	PTJ2	PTJ1	PTJ0
Write:								
EMAC	—	—			MII_COL	MII_CRS	MII_MDIO	MII_MDC
IIC	IIC_SCL	IICSDA			—	—	—	—
KWU	KWJ				KWJ			
Reset:	0	0	—	—	0	0	0	0

= Reserved or unimplemented

Figure 3-31. Port J I/O Register (PTJ)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The EMAC MII external interface and IIC take precedence over general-purpose I/O function. If the EMAC MII external interface is enabled in external PHY mode, PJ[3:0] pins become MII_MDC, MII_MDIO, MII_CRS, MII_COL. If IIC is enabled, PJ[7:6] pins become IIC_SDA and IIC_SCL. Please refer to the EMAC and IIC block description chapters for details.

3.3.2.5.2 Input Register (PTIJ)

Module Base + $21

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIJ7	PTIJ6	0	0	PTIJ3	PTIJ2	PTIJ1	PTIJ0
Write:								
Reset:	—	—	—	—	—	—	—	—

= Reserved or unimplemented

Figure 3-32. Port J Input Register (PTIJ)

Read:Anytime.

Write: writes to this register have no effect.

This register always reads back the status of the associated pins. This can be used to detect overload or short circuit conditions on output pins.

3.3.2.5.3 Data Direction Register (DDRJ)

Module Base + $22

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DDRJ7	DDRJ6	0	0	DDRJ3	DDRJ2	DDRJ1	DDRJ0
Write:								
Reset:	0	0	—	—	0	0	0	0

= Reserved or unimplemented

Figure 3-33. Port J Data Direction Register (DDRJ)

Read:Anytime.

Write:Anytime.

This register configures port pins J[7:6]and PJ[3:0] as either input or output.

DDRJ[7:6][3:0] — Data Direction Port J

1 = Associated pin is configured as output.
0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTJ or PTIJ registers, when changing the DDRJ register.

If the IIC is enabled, It controls the direction of SCL and SDA and the corresponding DDRJ[7:6] bits have no effect on their I/O direction. Refer to the IIC block description chapter for details.

If the EMAC MII external interface is enabled, It controls the direction of MDC, MDIO, CRS and COL and the corresponding DDRJ[3:0] bits have no effect on their I/O direction. Refer to the EMAC block description chapter for details.

3.3.2.5.4 Reduced Drive Register (RDRJ)

Module Base + $23

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RDRJ7	RDRJ6	0	0	RDRJ3	RDRJ2	RDRJ1	RDRJ0
Write:								
Reset:	0	0	—	—	0	0	0	0

= Reserved or unimplemented

Figure 3-34. Port J Reduced Drive Register (RDRJ)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port J output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRJ[7:6][3:0] — Reduced Drive Port J

1 = Associated pin drives at about 1/3 of the full drive strength.
0 = Full drive strength at output.

3.3.2.5.5 Pull Device Enable Register (PERJ)

Module Base + $24

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PERJ7	PERJ6	0	0	PERJ3	PERJ2	PERJ1	PERJ0
Write:								
Reset:	1	1	—	—	0	0	0	0

= Reserved or unimplemented

Figure 3-35. Port J Pull Device Enable Register (PERJ)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input. This bit has no effect if the port is used as output. Out of reset pull-up device is enabled for bits PJ[7:6] and disabled for bits PJ[3:0].

PERJ[7:6][3:0] — Pull Device Enable Port J

1 = Either a pull-up or pull-down device is enabled.
0 = Pull-up or pull-down device is disabled.

3.3.2.5.6 Polarity Select Register (PPSJ)

Module Base + $25

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PPSJ7	PPSJ6	0	0	PPSJ3	PPSJ2	PPSJ1	PPSJ0
Write:								
Reset:	0	0	—	—	0	0	0	0

= Reserved or unimplemented

Figure 3-36. Port J Polarity Select Register (PPSJ)

Read:Anytime.

Write:Anytime.

This register serves a dual purpose by selecting the polarity of the active interrupt edge as well as selecting a pull-up or pull-down device if enabled.

PPSJ[7:6][3:0] — Polarity Select Port J

1 = Rising edge on the associated port J pin sets the associated flag bit in the PIFJ register. A pull-down device is connected to the associated port J pin, if enabled by the associated bit in register PERJ and if the port is used as input.
0 = Falling edge on the associated port J pin sets the associated flag bit in the PIFJ register. A pull-up device is connected to the associated port J pin, if enabled by the associated bit in register PERJ and if the port is used as input.

3.3.2.5.7 Interrupt Enable Register (PIEJ)

Module Base + $26

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIEJ7	PIEJ6	0	0	PIEJ3	PIEJ2	PIEJ1	PIEJ0
Write:								
Reset:	0	0	—	—	0	0	0	0

= Reserved or unimplemented

Figure 3-37. Port J Interrupt Enable Register (PIEJ)

Read:Anytime.

Write:Anytime.

This register disables or enables on a per pin basis the edge sensitive external interrupt associated with port J.

PIEJ[7:6][3:0]— Interrupt Enable Port J

1 = Interrupt is enabled.
0 = Interrupt is disabled (interrupt flag masked).

3.3.2.5.8 Interrupt Flag Register (PIFJ)

Module Base + $27

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIFJ7	PIFJ6	0	0	PIFJ3	PIFJ2	PIFJ1	PIFJ0
Write:								
Reset:	0	0	—	—	0	0	0	0

= Reserved or unimplemented

Figure 3-38. Port J Interrupt Flag Register (PIFJ)

Read:Anytime.

Write:Anytime.

Each flag is set by an active edge on the associated input pin. This could be a rising or a falling edge based on the state of the PPSJ register. To clear this flag, write “1” to the corresponding bit in the PIFJ register. Writing a “0” has no effect.

PIFJ[7:6][3:0] — Interrupt Flags Port J

1 = Active edge on the associated bit has occurred (an interrupt will occur if the associated enable bit is set).
Writing a “1” clears the associated flag.
0 = No active edge pending.
Writing a “0” has no effect.

3.3.2.6 Port L Registers

3.3.2.6.1 I/O Register (PTL)

Module Base + $28

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTL6	PTL5	PTL4	PTL3	PTL2	PTL1	PTL0
Write:								
PHY				COLLED	DUPLED	SPDLED	LNKLED	ACTLED
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-39. Port L I/O Register (PTL)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The EPHY LED drive takes precedence over general-purpose I/O function if the EPHYCTL0 LEDEN bit is enabled. With the LEDEN bit set, PTL[4:0] become COLLED, DUPLED, SPDLED, LNKLED, and ACTLED. Refer to EPHY block description chapter for more detail.

3.3.2.6.2 Input Register (PTIL)

Module Base + $29

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTIL6	PTIL5	PTIL4	PTIL3	PTIL2	PTIL1	PTIL0
Write:								
Reset:	—	—	—	—	—	—	—	—

= Reserved or unimplemented

Figure 3-40. Port L Input Register (PTIL)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This also can be used to detect overload or short circuit conditions on output pins.

3.3.2.6.3 Data Direction Register (DDRL)

Module Base + $2A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	DDRL6	DDRL5	DDRL4	DDRL3	DDRL2	DDRL1	DDRL0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-41. Port L Data Direction Register (DDRL)

Read:Anytime.

Write:Anytime.

DDRL[6:0] — Data Direction Port L

1 = Associated pin is configured as output.
0 = Associated pin is configured as input.

This register configures each port L pin as either input or output. If EPHY port status LEDs are enabled, pins PL[4:0] are forced to be outputs and this register has no effect on their directions. Refer to the EPHY block description chapter for more information.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTL or PTIL registers, when changing the DDRL register.

3.3.2.6.4 Reduced Drive Register (RDRL)

Module Base + $2B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	RDRL6	RDRL5	RDRL4	RDRL3	RDRL2	RDRL1	RDRL0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-42. Port L Reduced Drive Register (RDRL)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port L output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRL[6:0] — Reduced Drive Port L

1 = Associated pin drives at about 1/3 of the full drive strength.
0 = Full drive strength at output.

3.3.2.6.5 Pull Device Enable Register (PERL)

Module Base + $2C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PERL6	PERL5	PERL4	PERL3	PERL2	PERL1	PERL0
Write:								
Reset:	—	1	1	1	1	1	1	1

= Reserved or unimplemented

Figure 3-43. Port L Pull Device Enable Register (PERL)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input or as output in wired-or (open drain) mode. These bits have no effect if the port is used as push-pull output. Out of reset a pull-up device is enabled.

PERL[6:0] — Pull Device Enable Port L

1 = Either a pull-up or pull-down device is enabled.
0 = Pull-up or pull-down device is disabled.

3.3.2.6.6 Polarity Select Register (PPSL)

Module Base + $2D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PPSL6	PPSL5	PPSL4	PPSL3	PPSL2	PPSL1	PPSL0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-44. Port L Polarity Select Register (PPSL)

Read:Anytime.

Write:Anytime.

This register selects whether a pull-down or a pull-up device is connected to the pin.

PPSL[6:0] — Pull Select Port L

1 = A pull-down device is connected to the associated port L pin, if enabled by the associated bit in register PERL and if the port is used as input.
0 = A pull-up device is connected to the associated port L pin, if enabled by the associated bit in register PERL and if the port is used as input or as wired-or output.

3.3.2.6.7 Wired-Or Mode Register (WOML)

Address Offset: $__2E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	WOML6	WOML5	WOML4	WOML3	WOML2	WOML1	WOML0
Write:								
Reset:	—	0	0	0	0	0	0	0

= Reserved or unimplemented

Figure 3-45. Port L Wired-Or Mode Register (WOML)

Read:Anytime.

Write:Anytime.

This register configures the output pins as wired-or. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven. This bit has no effect on pins used as inputs.

WOML[6:0] — Wired-Or Mode Port L

1 = Open-drain mode enabled for output buffers.
0 = Open-drain mode disabled for output buffers.

3.4 Functional Description

Each pin can act as general-purpose I/O. In addition the pin can act as an output from a peripheral module or an input to a peripheral module.

A set of configuration registers is common to all ports. All registers can be written at any time, however a specific configuration might not become active.

Example:

Selecting a pull-up resistor. This resistor does not become active while the port is used as a push-pull output.

3.4.1 I/O Register

This register holds the value driven out to the pin if the port is used as a general-purpose I/O.

Writing to this register has only an effect on the pin if the port is used as general-purpose output. When reading this address, the value of the pins are returned if the data direction register bits are set to 0.

If the data direction register bits are set to 1, the contents of the I/O register is returned. This is independent of any other configuration (Figure 3-46).

3.4.2 Input Register

This is a read-only register and always returns the value of the pin (Figure 3-46).

Data direction register

This register defines whether the pin is used as an input or an output.

If a peripheral module controls the pin the contents of the data direction register is ignored (Figure 3-46).

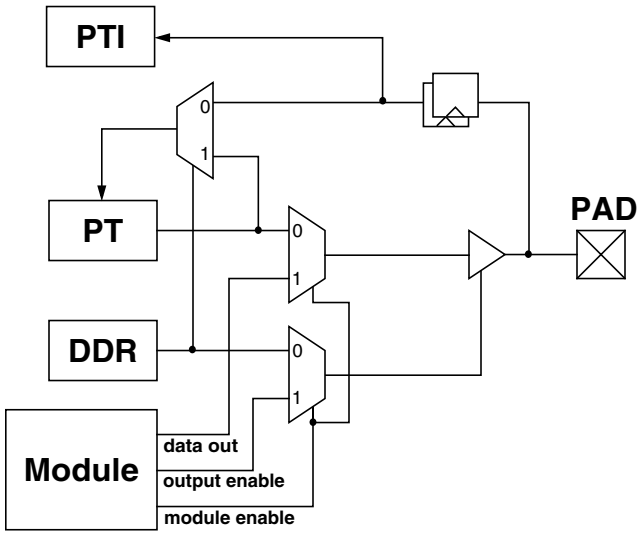


Figure 3-46. Illustration of I/O Pin Functionality

3.4.3 Reduced Drive Register

If the port is used as an output the register allows the configuration of the drive strength.

3.4.4 Pull Device Enable Register

This register turns on a pull-up or pull-down device.

It becomes only active if the pin is used as an input or as a wired-or output.

3.4.5 Polarity Select Register

This register selects either a pull-up or pull-down device if enabled.

It becomes only active if the pin is used as an input or wired-or output. A pull-up device can also be activated if the pin is used as a wired-or output.

3.4.6 Port T

This port is associated with the standard Timer.

In all modes, port T pins PT[7:4] can be used for either general-purpose I/O or standard timer I/O.

During reset, port T pins are configured as high-impedance inputs.

3.4.7 Port S

This port is associated with the serial SCI and SPI modules.

Port S pins PS[7:0] can be used either for general-purpose I/O, or with the SCI0, SCI1, and SPI subsystems.

During reset, port S pins are configured as inputs with pull-up.

3.4.8 Port G

This port is associated with the EMAC module.

Port G pins PG[7:0] can be used either for general-purpose I/O or with the EMAC subsystems. Further the Keypad Wake-Up function is implemented on pins G[7:0].

During reset, port G pins are configured as high-impedance inputs.

3.4.8.1 Interrupts

Port G offers eight general-purpose I/O pins with edge triggered interrupt capability in wired-or fashion. The interrupt enable as well as the sensitivity to rising or falling edges can be individually configured on per pin basis. All eight bits/pins share the same interrupt vector. Interrupts can be used with the pins configured as inputs or outputs.

An interrupt is generated when a bit in the port interrupt flag register and its corresponding port interrupt enable bit are both set. This external interrupt feature is capable to wake up the CPU when it is in STOP or WAIT mode.

A digital filter on each pin prevents pulses (Figure 3-48) shorter than a specified time from generating an interrupt. The minimum time varies over process conditions, temperature and voltage (Figure 3-47 and Table 3-4).

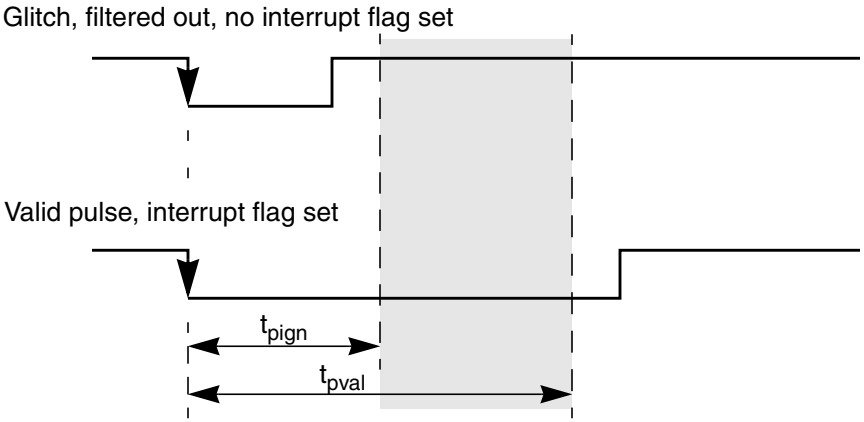


Figure 3-47. Interrupt Glitch Filter on Port G, H, and J (PPS=0)

Table 3-4. Pulse Detection Criteria

Pulse	Mode			
	$\overline{STOP}$		STOP[1]	
		Unit		Unit
Ignored	t_{pign} <= 3	bus clocks	t_{pign} <= 3.2	µs
Uncertain	3 < t_{pulse} < 4	bus clocks	3.2 < t_{pulse} < 10	µs
Valid	t_{pval} >= 4	bus clocks	t_{pval} >= 10	µs

1 These values include the spread of the oscillator frequency over temperature, voltage and process.

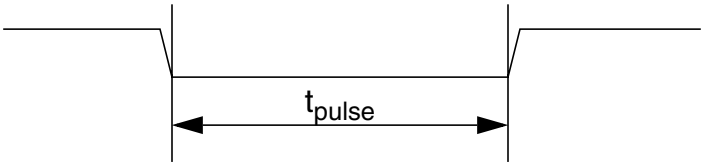


Figure 3-48. Pulse Illustration

A valid edge on input is detected if 4 consecutive samples of a passive level are followed by 4 consecutive samples of an active level directly or indirectly.

The filters are continuously clocked by the bus clock in RUN and WAIT mode. In STOP mode the clock is generated by a single RC oscillator in the Port Integration Module. To maximize current saving the RC oscillator runs only if the following condition is true on any pin:

Sample count <= 4 and port interrupt enabled (PIE=1) and port interrupt flag not set (PIF=0).

3.4.9 Port H

The EMAC module is connected to port H.

Port H pins PH[6:0] can be used either for general-purpose I/O or with the EMAC subsystems. Further the keypad wake-up function is implemented on pins H[6:0].

Port H offers the same interrupt features as on port G.

During reset, port H pins are configured as high-impedance inputs.

3.4.10 Port J

The EMAC and IIC modules are connected to port J.

Port J pins PJ[7:6] can be used either for general-purpose I/O or with the IIC subsystem. Port J pins PJ[3:0] can be used either for general-purpose I/O or with the EMAC subsystems. Further the Keypad Wake-Up function is implemented on pins H[6:0].

Port J offers the same interrupt features as on port G.

If IIC takes precedence thePJ[7:6] pins become IIC open drain output pins.

During reset, pins PJ[7:6] are configured as inputs with pull-ups and pins PJ[3:0] are configured as high-impedance inputs.

3.4.11 Port L

In all modes, port L pins PL[6:0] can be used either for general-purpose I/O or with the EPHY subsystem. During reset, port L pins are configured as inputs with pull-ups.

3.4.12 Port A, B, E and BKGD Pin

All port and pin logic is located in the core module. Please refer to MEBI block description chapter for details.

3.4.13 External Pin Descriptions

All ports start up as general-purpose inputs on reset.

3.4.14 Low Power Options

3.4.14.1 Run Mode

No low power options exist for this module in run mode.

3.4.14.2 Wait Mode

No low power options exist for this module in wait mode.

3.4.14.3 Stop Mode

All clocks are stopped. There are asynchronous paths to generate interrupts from STOP on port G, H, and J.

3.5 Initialization/Application Information

The reset values of all registers are given in Section 3.3, “Memory Map and Register Descriptions.”

3.5.1 Reset Initialization

All registers including the data registers get set/reset asynchronously. Table 3-5 summarizes the port properties after reset initialization.

Table 3-5. Port Reset State Summary

Port	Reset States				
	Data Direction	Pull Mode	Red. Drive	Wired-Or Mode	Interrupt
T	input	hiz	disabled	n/a	n/a
S	input	pull-up	disabled	disabled	n/a
G	input	hiz	disabled	n/a	disabled
H	input	hiz	disabled	n/a	disabled
J[7:6]	input	pull-up	disabled	n/a	disabled
J[3:0]	input	hiz	disabled	n/a	disabled
L	input	pull-up	disabled	disabled	n/a
A	Refer to the MEBI block description chapter for details.				
B					
E					
K					
BKGD pin	Refer to the BDM block description chapter for details.				

3.6 Interrupts

Port G, H, and J generate a separate edge sensitive interrupt if enabled.

3.6.1 Interrupt Sources

Table 3-6. Port Integration Module Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Port G	PIFG[7:0]	PIEG[7:0]	I Bit
Port H	PIFH[6:0]	PIEH[6:0]	I Bit
Port J	PIFJ[7:6],[3:0]	PIEJ[7:6],[3:0]	I Bit

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

3.6.2 Recovery from Stop

The PIM_9NE64 can generate wake-up interrupts from stop on port G, H, and J. For other sources of external interrupts please refer to the respective block description chapter.

Chapter 4
Clocks and Reset Generator (CRGV4)

4.1 Introduction

This specification describes the function of the clocks and reset generator (CRGV4).

4.1.1 Features

The main features of this block are:

- Phase-locked loop (PLL) frequency multiplier
  - Reference divider
  - Automatic bandwidth control mode for low-jitter operation
  - Automatic frequency lock detector
  - CPU interrupt on entry or exit from locked condition
  - Self-clock mode in absence of reference clock
- System clock generator
  - Clock quality check
  - Clock switch for either oscillator- or PLL-based system clocks
  - User selectable disabling of clocks during wait mode for reduced power consumption
- Computer operating properly (COP) watchdog timer with time-out clear window
- System reset generation from the following possible sources:
  - Power-on reset
  - Low voltage reset
    Refer to the device overview section for availability of this feature.
  - COP reset
  - Loss of clock reset
  - External pin reset
- Real-time interrupt (RTI)

4.1.2 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the CRG.

- Run mode

  All functional parts of the CRG are running during normal run mode. If RTI or COP functionality is required the individual bits of the associated rate select registers (COPCTL, RTICTL) have to be set to a nonzero value.

- Wait mode

  This mode allows to disable the system and core clocks depending on the configuration of the individual bits in the CLKSEL register.

- Stop mode

  Depending on the setting of the PSTP bit, stop mode can be differentiated between full stop mode (PSTP = 0) and pseudo-stop mode (PSTP = 1).

  — Full stop mode

  The oscillator is disabled and thus all system and core clocks are stopped. The COP and the RTI remain frozen.

  — Pseudo-stop mode

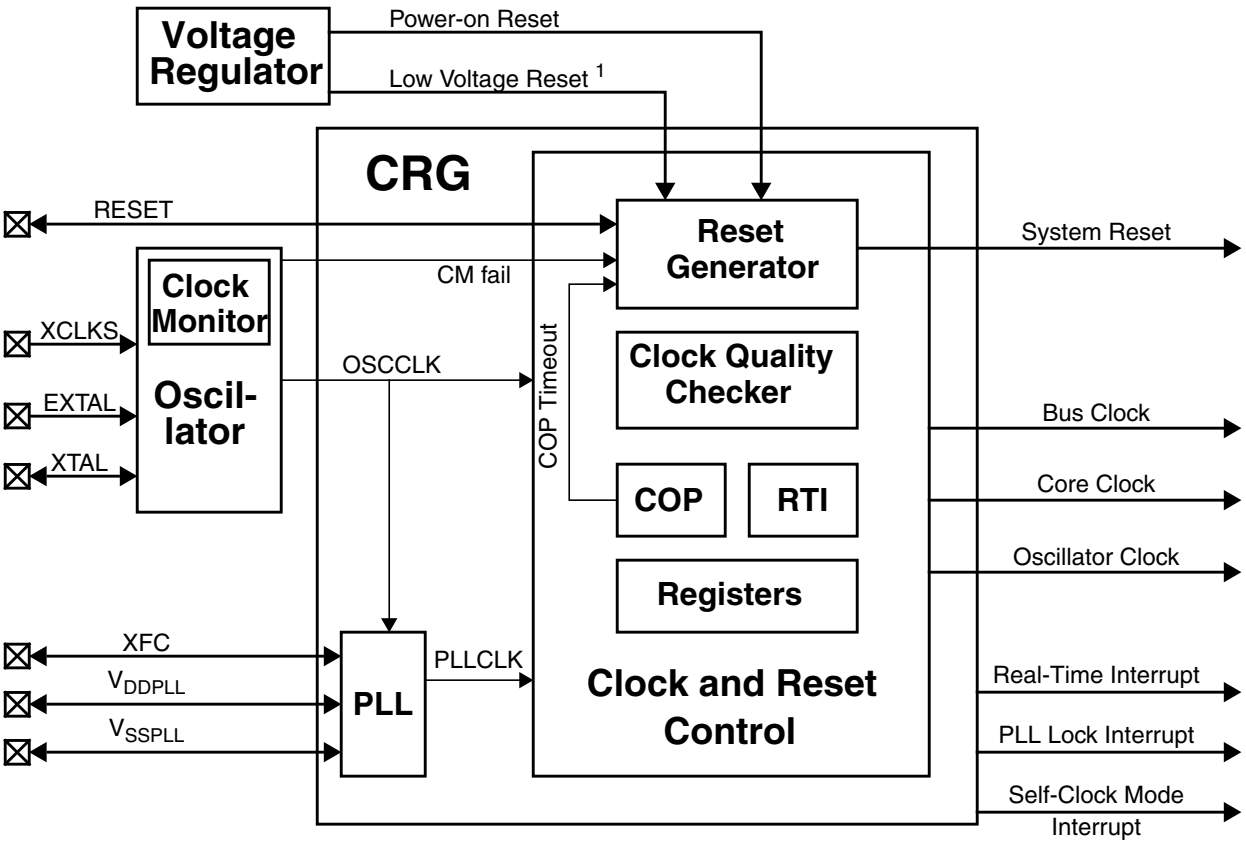
  The oscillator continues to run and most of the system and core clocks are stopped. If the respective enable bits are set the COP and RTI will continue to run, else they remain frozen.

- Self-clock mode

  Self-clock mode will be entered if the clock monitor enable bit (CME) and the self-clock mode enable bit (SCME) are both asserted and the clock monitor in the oscillator block detects a loss of clock. As soon as self-clock mode is entered the CRGV4 starts to perform a clock quality check. Self-clock mode remains active until the clock quality check indicates that the required quality of the incoming clock signal is met (frequency and amplitude). Self-clock mode should be used for safety purposes only. It provides reduced functionality to the MCU in case a loss of clock is causing severe system conditions.

4.1.3 Block Diagram

Figure 4-1 shows a block diagram of the CRGV4.



1 Refer to the device overview section for availability of the low-voltage reset feature.

Figure 4-1. CRG Block Diagram

4.2 External Signal Description

This section lists and describes the signals that connect off chip.

4.2.1 V_{DDPLL}, V_{SSPLL} — PLL Operating Voltage, PLL Ground

These pins provides operating voltage (V_{DDPLL}) and ground (V_{SSPLL}) for the PLL circuitry. This allows the supply voltage to the PLL to be independently bypassed. Even if PLL usage is not required V_{DDPLL} and V_{SSPLL} must be connected properly.

4.2.2 XFC — PLL Loop Filter Pin

A passive external loop filter must be placed on the XFC pin. The filter is a second-order, low-pass filter to eliminate the VCO input ripple. The value of the external filter network and the reference frequency determines the speed of the corrections and the stability of the PLL. Refer to the device overview chapter for calculation of PLL loop filter (XFC) components. If PLL usage is not required the XFC pin must be tied to V_{DDPLL}.

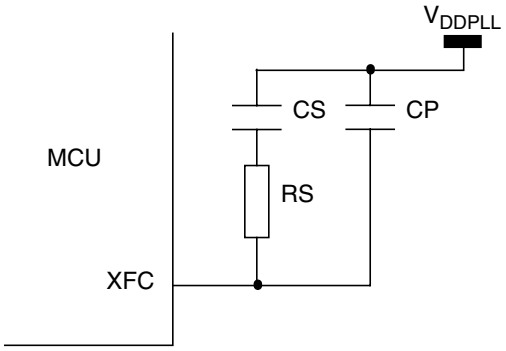


Figure 4-2. PLL Loop Filter Connections

4.2.3 $\overline{\text{RESET}}$ — Reset Pin

$\overline{\text{RESET}}$ is an active low bidirectional reset pin. As an input it initializes the MCU asynchronously to a known start-up state. As an open-drain output it indicates that an system reset (internal to MCU) has been triggered.

4.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the CRGV4.

4.3.1 Module Memory Map

Table 4-1 gives an overview on all CRGV4 registers.

Table 4-1. CRGV4 Memory Map

Address Offset	Use	Access
0x0000	CRG Synthesizer Register (SYNR)	R/W
0x0001	CRG Reference Divider Register (REFDV)	R/W
0x0002	CRG Test Flags Register (CTFLG)[1]	R/W
0x0003	CRG Flags Register (CRGFLG)	R/W
0x0004	CRG Interrupt Enable Register (CRGINT)	R/W
0x0005	CRG Clock Select Register (CLKSEL)	R/W
0x0006	CRG PLL Control Register (PLLCTL)	R/W
0x0007	CRG RTI Control Register (RTICTL)	R/W
0x0008	CRG COP Control Register (COPCTL)	R/W
0x0009	CRG Force and Bypass Test Register (FORBYP)[2]	R/W
0x000A	CRG Test Control Register (CTCTL)[3]	R/W
0x000B	CRG COP Arm/Timer Reset (ARMCOP)	R/W

1 CTFLG is intended for factory test purposes only.
2 FORBYP is intended for factory test purposes only.
3 CTCTL is intended for factory test purposes only.

NOTE

Register address = base address + address offset, where the base address is defined at the MCU level and the address offset is defined at the module level.

4.3.2 Register Descriptions

This section describes in address order all the CRGV4 registers and their individual bits.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
SYNR	R	0	0	SYN5	SYN4	SYN3	SYN2	SYN1	SYN0
	W								
REFDV	R	0	0	0	0	REFDV3	REFDV2	REFDV1	REFDV0
	W								
CTFLG	R	0	0	0	0	0	0	0	0
	W								
CRGFLG	R	RTIF	PORF	LVRF	LOCKIF	LOCK	TRACK	SCMIF	SCM
	W								
CRGINT	R	RTIE	0	0	LOCKIE	0	0	SCMIE	0
	W								
CLKSEL	R	PLLSEL	PSTP	SYSWAI	ROAWAI	PLLWAI	CWAI	RTIWAI	COPWAI
	W								
PLLCTL	R	CME	PLLON	AUTO	ACQ	0	PRE	PCE	SCME
	W								
RTICTL	R	0	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
	W								
COPCTL	R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
	W								
FORBYP	R	0	0	0	0	0	0	0	0
	W								
CTCTL	R	0	0	0	0	0	0	0	0
	W								

= Unimplemented or Reserved

Figure 4-3. CRG Register Summary

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
ARMCOP	R	0	0	0	0	0	0	0	0
	W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

= Unimplemented or Reserved

Figure 4-3. CRG Register Summary (continued)

4.3.2.1 CRG Synthesizer Register (SYNR)

The SYNR register controls the multiplication factor of the PLL. If the PLL is on, the count in the loop divider (SYNR) register effectively multiplies up the PLL clock (PLLCLK) from the reference frequency by 2 x (SYNR+1). PLLCLK will not be below the minimum VCO frequency (f_{SCM}).

$$PLLCLK = 2 \times OSCCLK \times \frac{(SYNR + 1)}{(REFDV + 1)}$$

NOTE

If PLL is selected (PLLSEL=1), Bus Clock = PLLCLK / 2
Bus Clock must not exceed the maximum operating system frequency.

	7	6	5	4	3	2	1	0
R	0	0	SYN5	SYNR	SYN3	SYN2	SYN1	SYN0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-4. CRG Synthesizer Register (SYNR)

Read: anytime

Write: anytime except if PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit and the track detector bit.

4.3.2.2 CRG Reference Divider Register (REFDV)

The REFDV register provides a finer granularity for the PLL multiplier steps. The count in the reference divider divides OSCCLK frequency by REFDV + 1.

	7	6	5	4	3	2	1	0
R	0	0	0	0	REFDV3	REFDV2	REFDV1	REFDV0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-5. CRG Reference Divider Register (REFDV)

Read: anytime

Write: anytime except when PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit and the track detector bit.

4.3.2.3 Reserved Register (CTFLG)

This register is reserved for factory testing of the CRGV4 module and is not available in normal modes.

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

= Unimplemented or Reserved

Figure 4-6. CRG Reserved Register (CTFLG)

Read: always reads 0x0000 in normal modes

Write: unimplemented in normal modes

NOTE

Writing to this register when in special mode can alter the CRGV4 functionality.

4.3.2.4 CRG Flags Register (CRGFLG)

This register provides CRG status bits and flags.

	7	6	5	4	3	2	1	0
R	RTIF	PORF	LVRF	LOCKIF	LOCK	TRACK	SCMIF	SCM
W								
Reset	0	Note 1	Note 2	0	0	0	0	0

1. PORF is set to 1 when a power-on reset occurs. Unaffected by system reset.
2. LVRF is set to 1 when a low-voltage reset occurs. Unaffected by system reset.

= Unimplemented or Reserved

Figure 4-7. CRG Flag Register (CRGFLG)

Read: anytime

Write: refer to each bit for individual write conditions

Table 4-2. CRGFLG Field Descriptions

Field	Description
7 RTIF	**Real-Time Interrupt Flag** — RTIF is set to 1 at the end of the RTI period. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (RTIE = 1), RTIF causes an interrupt request. 0 RTI time-out has not yet occurred. 1 RTI time-out has occurred.
6 PORF	**Power-on Reset Flag** — PORF is set to 1 when a power-on reset occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Power-on reset has not occurred. 1 Power-on reset has occurred.
5 LVRF	**Low-Voltage Reset Flag** — If low voltage reset feature is not available (see the device overview chapter), LVRF always reads 0. LVRF is set to 1 when a low voltage reset occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Low voltage reset has not occurred. 1 Low voltage reset has occurred.
4 LOCKIF	**PLL Lock Interrupt Flag** — LOCKIF is set to 1 when LOCK status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect.If enabled (LOCKIE = 1), LOCKIF causes an interrupt request. 0 No change in LOCK bit. 1 LOCK bit has changed.
3 LOCK	**Lock Status Bit** — LOCK reflects the current state of PLL lock condition. This bit is cleared in self-clock mode. Writes have no effect. 0 PLL VCO is not within the desired tolerance of the target frequency. 1 PLL VCO is within the desired tolerance of the target frequency.
2 TRACK	**Track Status Bit** — TRACK reflects the current state of PLL track condition. This bit is cleared in self-clock mode. Writes have no effect. 0 Acquisition mode status. 1 Tracking mode status.

Table 4-2. CRGFLG Field Descriptions (continued)

Field	Description
1 SCMIF	**Self-Clock Mode Interrupt Flag** — SCMIF is set to 1 when SCM status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (SCMIE=1), SCMIF causes an interrupt request. 0 No change in SCM bit. 1 SCM bit has changed.
0 SCM	**Self-Clock Mode Status Bit** — SCM reflects the current clocking mode. Writes have no effect. 0 MCU is operating normally with OSCCLK available. 1 MCU is operating in self-clock mode with OSCCLK in an unknown state. All clocks are derived from PLLCLK running at its minimum frequency f_{SCM}.

4.3.2.5 CRG Interrupt Enable Register (CRGINT)

This register enables CRG interrupt requests.

	7	6	5	4	3	2	1	0
R	RTIE	0	0	LOCKIE	0	0	SCMIE	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-8. CRG Interrupt Enable Register (CRGINT)

Read: anytime

Write: anytime

Table 4-3. CRGINT Field Descriptions

Field	Description
7 RTIE	**Real-Time Interrupt Enable Bit** 0 Interrupt requests from RTI are disabled. 1 Interrupt will be requested whenever RTIF is set.
4 LOCKIE	**Lock Interrupt Enable Bit** 0 LOCK interrupt requests are disabled. 1 Interrupt will be requested whenever LOCKIF is set.
1 SCMIE	**Self-Clock Mode Interrupt Enable Bit** 0 SCM interrupt requests are disabled. 1 Interrupt will be requested whenever SCMIF is set.

4.3.2.6 CRG Clock Select Register (CLKSEL)

This register controls CRG clock selection. Refer to Figure 4-17 for details on the effect of each bit.

	7	6	5	4	3	2	1	0
R W	PLLSEL	PSTP	SYSWAI	ROAWAI	PLLWAI	CWAI	RTIWAI	COPWAI
Reset	0	0	0	0	0	0	0	0

Figure 4-9. CRG Clock Select Register (CLKSEL)

Read: anytime

Write: refer to each bit for individual write conditions

Table 4-4. CLKSEL Field Descriptions

Field	Description
7 PLLSEL	**PLL Select Bit** — Write anytime. Writing a 1 when LOCK = 0 and AUTO = 1, or TRACK = 0 and AUTO = 0 has no effect. This prevents the selection of an unstable PLLCLK as SYSCLK. PLLSEL bit is cleared when the MCU enters self-clock mode, stop mode or wait mode with PLLWAI bit set. 0 System clocks are derived from OSCCLK (Bus Clock = OSCCLK / 2). 1 System clocks are derived from PLLCLK (Bus Clock = PLLCLK / 2).
6 PSTP	**Pseudo-Stop Bit** — Write: anytime — This bit controls the functionality of the oscillator during stop mode. 0 Oscillator is disabled in stop mode. 1 Oscillator continues to run in stop mode (pseudo-stop). The oscillator amplitude is reduced. Refer to oscillator block description for availability of a reduced oscillator amplitude. **Note:** Pseudo-stop allows for faster stop recovery and reduces the mechanical stress and aging of the resonator in case of frequent stop conditions at the expense of a slightly increased power consumption. **Note:** Lower oscillator amplitude exhibits lower power consumption but could have adverse effects during any electro-magnetic susceptibility (EMS) tests.
5 SYSWAI	**System Clocks Stop in Wait Mode Bit** — Write: anytime 0 In wait mode, the system clocks continue to run. 1 In wait mode, the system clocks stop. **Note:** RTI and COP are not affected by SYSWAI bit.
4 ROAWAI	**Reduced Oscillator Amplitude in Wait Mode Bit** — Write: anytime — Refer to oscillator block description chapter for availability of a reduced oscillator amplitude. If no such feature exists in the oscillator block then setting this bit to 1 will not have any effect on power consumption. 0 Normal oscillator amplitude in wait mode. 1 Reduced oscillator amplitude in wait mode. **Note:** Lower oscillator amplitude exhibits lower power consumption but could have adverse effects during any electro-magnetic susceptibility (EMS) tests.
3 PLLWAI	**PLL Stops in Wait Mode Bit** — Write: anytime — If PLLWAI is set, the CRGV4 will clear the PLLSEL bit before entering wait mode. The PLLON bit remains set during wait mode but the PLL is powered down. Upon exiting wait mode, the PLLSEL bit has to be set manually if PLL clock is required. While the PLLWAI bit is set the AUTO bit is set to 1 in order to allow the PLL to automatically lock on the selected target frequency after exiting wait mode. 0 PLL keeps running in wait mode. 1 PLL stops in wait mode.
2 CWAI	**Core Stops in Wait Mode Bit** — Write: anytime 0 Core clock keeps running in wait mode. 1 Core clock stops in wait mode.

Table 4-4. CLKSEL Field Descriptions (continued)

Field	Description
1 RTIWAI	**RTI Stops in Wait Mode Bit** — Write: anytime 0 RTI keeps running in wait mode. 1 RTI stops and initializes the RTI dividers whenever the part goes into wait mode.
0 COPWAI	**COP Stops in Wait Mode Bit** — Normal modes: Write once —Special modes: Write anytime 0 COP keeps running in wait mode. 1 COP stops and initializes the COP dividers whenever the part goes into wait mode.

4.3.2.7 CRG PLL Control Register (PLLCTL)

This register controls the PLL functionality.

	7	6	5	4	3	2	1	0
R	CME	PLLON	AUTO	ACQ	0	PRE	PCE	SCME
W								
Reset	1	1	1	1	0	0	0	1

= Unimplemented or Reserved

Figure 4-10. CRG PLL Control Register (PLLCTL)

Read: anytime

Write: refer to each bit for individual write conditions

Table 4-5. PLLCTL Field Descriptions

Field	Description
7 CME	**Clock Monitor Enable Bit** — CME enables the clock monitor. Write anytime except when SCM = 1. 0 Clock monitor is disabled. 1 Clock monitor is enabled. Slow or stopped clocks will cause a clock monitor reset sequence or self-clock mode. **Note:** Operating with CME = 0 will not detect any loss of clock. In case of poor clock quality this could cause unpredictable operation of the MCU. **Note:** In Stop Mode (PSTP = 0) the clock monitor is disabled independently of the CME bit setting and any loss of clock will not be detected.
6 PLLON	**Phase Lock Loop On Bit** — PLLON turns on the PLL circuitry. In self-clock mode, the PLL is turned on, but the PLLON bit reads the last latched value. Write anytime except when PLLSEL = 1. 0 PLL is turned off. 1 PLL is turned on. If AUTO bit is set, the PLL will lock automatically.
5 AUTO	**Automatic Bandwidth Control Bit** — AUTO selects either the high bandwidth (acquisition) mode or the low bandwidth (tracking) mode depending on how close to the desired frequency the VCO is running. Write anytime except when PLLWAI=1, because PLLWAI sets the AUTO bit to 1. 0 Automatic mode control is disabled and the PLL is under software control, using ACQ bit. 1 Automatic mode control is enabled and ACQ bit has no effect.
4 ACQ	**Acquisition Bit** — Write anytime. If AUTO=1 this bit has no effect. 0 Low bandwidth filter is selected. 1 High bandwidth filter is selected.

Table 4-5. PLLCTL Field Descriptions (continued)

Field	Description
2 PRE	**RTI Enable during Pseudo-Stop Bit** — PRE enables the RTI during pseudo-stop mode. Write anytime. 0 RTI stops running during pseudo-stop mode. 1 RTI continues running during pseudo-stop mode. **Note:** If the PRE bit is cleared the RTI dividers will go static while pseudo-stop mode is active. The RTI dividers will not initialize like in wait mode with RTIWAI bit set.
1 PCE	**COP Enable during Pseudo-Stop Bit** — PCE enables the COP during pseudo-stop mode. Write anytime. 0 COP stops running during pseudo-stop mode 1 COP continues running during pseudo-stop mode **Note:** If the PCE bit is cleared the COP dividers will go static while pseudo-stop mode is active. The COP dividers will *not* initialize like in wait mode with COPWAI bit set.
0 SCME	**Self-Clock Mode Enable Bit** — Normal modes: Write once —Special modes: Write anytime — SCME can not be cleared while operating in self-clock mode (SCM=1). 0 Detection of crystal clock failure causes clock monitor reset (see Section 4.5.1, “Clock Monitor Reset”). 1 Detection of crystal clock failure forces the MCU in self-clock mode (see Section 4.4.7.2, “Self-Clock Mode”).

4.3.2.8 CRG RTI Control Register (RTICTL)

This register selects the timeout period for the real-time interrupt.

	7	6	5	4	3	2	1	0
R	0	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-11. CRG RTI Control Register (RTICTL)

Read: anytime

Write: anytime

NOTE

A write to this register initializes the RTI counter.

Table 4-6. RTICTL Field Descriptions

Field	Description
6:4 RTR[6:4]	**Real-Time Interrupt Prescale Rate Select Bits** — These bits select the prescale rate for the RTI. See Table 4-7.
3:0 RTR[3:0]	**Real-Time Interrupt Modulus Counter Select Bits** — These bits select the modulus counter target value to provide additional granularity. Table 4-7 shows all possible divide values selectable by the RTICTL register. The source clock for the RTI is OSCCLK.

Table 4-7. RTI Frequency Divide Rates

RTR[3:0]	RTR[6:4] =							
	000 (OFF)	**001 (2^{10})**	**010 (2^{11})**	**011 (2^{12})**	**100 (2^{13})**	**101 (2^{14})**	**110 (2^{15})**	**111 (2^{16})**
0000 (÷1)	OFF*	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}
0001 (÷2)	OFF*	$2x2^{10}$	$2x2^{11}$	$2x2^{12}$	$2x2^{13}$	$2x2^{14}$	$2x2^{15}$	$2x2^{16}$
0010 (÷3)	OFF*	$3x2^{10}$	$3x2^{11}$	$3x2^{12}$	$3x2^{13}$	$3x2^{14}$	$3x2^{15}$	$3x2^{16}$
0011 (÷4)	OFF*	$4x2^{10}$	$4x2^{11}$	$4x2^{12}$	$4x2^{13}$	$4x2^{14}$	$4x2^{15}$	$4x2^{16}$
0100 (÷5)	OFF*	$5x2^{10}$	$5x2^{11}$	$5x2^{12}$	$5x2^{13}$	$5x2^{14}$	$5x2^{15}$	$5x2^{16}$
0101 (÷6)	OFF*	$6x2^{10}$	$6x2^{11}$	$6x2^{12}$	$6x2^{13}$	$6x2^{14}$	$6x2^{15}$	$6x2^{16}$
0110 (÷7)	OFF*	$7x2^{10}$	$7x2^{11}$	$7x2^{12}$	$7x2^{13}$	$7x2^{14}$	$7x2^{15}$	$7x2^{16}$
0111 (÷8)	OFF*	$8x2^{10}$	$8x2^{11}$	$8x2^{12}$	$8x2^{13}$	$8x2^{14}$	$8x2^{15}$	$8x2^{16}$
1000 (÷9)	OFF*	$9x2^{10}$	$9x2^{11}$	$9x2^{12}$	$9x2^{13}$	$9x2^{14}$	$9x2^{15}$	$9x2^{16}$
1001 (÷10)	OFF*	$10x2^{10}$	$10x2^{11}$	$10x2^{12}$	$10x2^{13}$	$10x2^{14}$	$10x2^{15}$	$10x2^{16}$
1010 (÷11)	OFF*	$11x2^{10}$	$11x2^{11}$	$11x2^{12}$	$11x2^{13}$	$11x2^{14}$	$11x2^{15}$	$11x2^{16}$
1011 (÷12)	OFF*	$12x2^{10}$	$12x2^{11}$	$12x2^{12}$	$12x2^{13}$	$12x2^{14}$	$12x2^{15}$	$12x2^{16}$
1100 (÷ 13)	OFF*	$13x2^{10}$	$13x2^{11}$	$13x2^{12}$	$13x2^{13}$	$13x2^{14}$	$13x2^{15}$	$13x2^{16}$
1101 (÷14)	OFF*	$14x2^{10}$	$14x2^{11}$	$14x2^{12}$	$14x2^{13}$	$14x2^{14}$	$14x2^{15}$	$14x2^{16}$
1110 (÷15)	OFF*	$15x2^{10}$	$15x2^{11}$	$15x2^{12}$	$15x2^{13}$	$15x2^{14}$	$15x2^{15}$	$15x2^{16}$
1111 (÷ 16)	OFF*	$16x2^{10}$	$16x2^{11}$	$16x2^{12}$	$16x2^{13}$	$16x2^{14}$	$16x2^{15}$	$16x2^{16}$

* Denotes the default value out of reset.This value should be used to disable the RTI to ensure future backwards compatibility.

4.3.2.9 CRG COP Control Register (COPCTL)

This register controls the COP (computer operating properly) watchdog.

	7	6	5	4	3	2	1	0
R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-12. CRG COP Control Register (COPCTL)

Read: anytime

Write: WCOP, CR2, CR1, CR0: once in user mode, anytime in special mode

Write: RSBCK: once

Table 4-8. COPCTL Field Descriptions

Field	Description
7 WCOP	**Window COP Mode Bit** — When set, a write to the ARMCOP register must occur in the last 25% of the selected period. A write during the first 75% of the selected period will reset the part. As long as all writes occur during this window, 0x0055 can be written as often as desired. As soon as 0x00AA is written after the 0x0055, the time-out logic restarts and the user must wait until the next window before writing to ARMCOP. Table 4-9 shows the exact duration of this window for the seven available COP rates. 0 Normal COP operation 1 Window COP operation
6 RSBCK	COP and RTI Stop in Active BDM Mode Bit 0 Allows the COP and RTI to keep running in active BDM mode. 1 Stops the COP and RTI counters whenever the part is in active BDM mode.
2:0 CR[2:0]	**COP Watchdog Timer Rate Select** — These bits select the COP time-out rate (see Table 4-9). The COP time-out period is OSCCLK period divided by CR[2:0] value. Writing a nonzero value to CR[2:0] enables the COP counter and starts the time-out period. A COP counter time-out causes a system reset. This can be avoided by periodically (before time-out) reinitializing the COP counter via the ARMCOP register.

Table 4-9. COP Watchdog Rates[1]

CR2	CR1	CR0	OSCCLK Cycles to Time Out
0	0	0	COP disabled
0	0	1	2^{14}
0	1	0	2^{16}
0	1	1	2^{18}
1	0	0	2^{20}
1	0	1	2^{22}
1	1	0	2^{23}
1	1	1	2^{24}

[1] OSCCLK cycles are referenced from the previous COP time-out reset (writing 0x0055/0x00AA to the ARMCOP register)

4.3.2.10 Reserved Register (FORBYP)

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special modes can alter the CRG's functionality.

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

= Unimplemented or Reserved

Figure 4-13. Reserved Register (FORBYP)

Read: always read 0x0000 except in special modes

Write: only in special modes

4.3.2.11 Reserved Register (CTCTL)

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special test modes can alter the CRG's functionality.

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

= Unimplemented or Reserved

Figure 4-14. Reserved Register (CTCTL)

Read: always read 0x0080 except in special modes

Write: only in special modes

4.3.2.12 CRG COP Timer Arm/Reset Register (ARMCOP)

This register is used to restart the COP time-out period.

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

Figure 4-15. ARMCOP Register Diagram

Read: always reads 0x0000

Write: anytime

When the COP is disabled (CR[2:0] = “000”) writing to this register has no effect.

When the COP is enabled by setting CR[2:0] nonzero, the following applies:

Writing any value other than 0x0055 or 0x00AA causes a COP reset. To restart the COP time-out period you must write 0x0055 followed by a write of 0x00AA. Other instructions may be executed between these writes but the sequence (0x0055, 0x00AA) must be completed prior to COP end of time-out period to avoid a COP reset. Sequences of 0x0055 writes or sequences of 0x00AA writes are allowed. When the WCOP bit is set, 0x0055 and 0x00AA writes must be done in the last 25% of the selected time-out period; writing any value in the first 75% of the selected period will cause a COP reset.

4.4 Functional Description

This section gives detailed informations on the internal operation of the design.

4.4.1 Phase Locked Loop (PLL)

The PLL is used to run the MCU from a different time base than the incoming OSCCLK. For increased flexibility, OSCCLK can be divided in a range of 1 to 16 to generate the reference frequency. This offers a finer multiplication granularity. The PLL can multiply this reference clock by a multiple of 2, 4, 6,... 126,128 based on the SYNR register.

$$PLLCLK = 2 \times OSCCLK \times \frac{[SYNR + 1]}{[REFDV + 1]}$$

CAUTION

Although it is possible to set the two dividers to command a very high clock frequency, do not exceed the specified bus frequency limit for the MCU.
If (PLLSEL = 1), Bus Clock = PLLCLK / 2

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

The VCO has a minimum operating frequency, which corresponds to the self-clock mode frequency f_{SCM}.

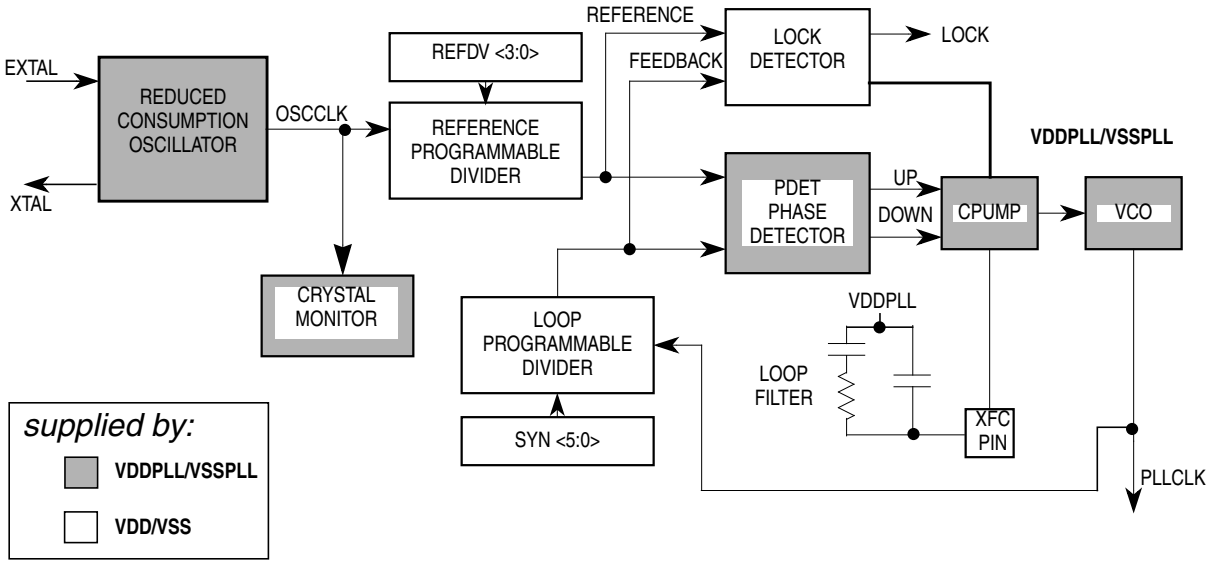


Figure 4-16. PLL Functional Diagram

4.4.1.1 PLL Operation

The oscillator output clock signal (OSCCLK) is fed through the reference programmable divider and is divided in a range of 1 to 16 (REFDV+1) to output the reference clock. The VCO output clock, (PLLCLK) is fed back through the programmable loop divider and is divided in a range of 2 to 128 in increments of [2 x (SYNR +1)] to output the feedback clock. See Figure 4-16.

The phase detector then compares the feedback clock, with the reference clock. Correction pulses are generated based on the phase difference between the two signals. The loop filter then slightly alters the DC voltage on the external filter capacitor connected to XFC pin, based on the width and direction of the correction pulse. The filter can make fast or slow corrections depending on its mode, as described in the next subsection. The values of the external filter network and the reference frequency determine the speed of the corrections and the stability of the PLL.

4.4.1.2 Acquisition and Tracking Modes

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

The PLL filter can be manually or automatically configured into one of two possible 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 TRACK status bit is cleared in the CRGFLG 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 and the TRACK bit is set in the CRGFLG register.

The PLL can change the bandwidth or operational mode of the loop filter manually or automatically.

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 PLL clock (PLLCLK) is safe to use as the source for the system and core clocks. If PLL LOCK interrupt requests are enabled, the software can wait for an interrupt request and then check the LOCK bit. If CPU interrupts are disabled, software can poll the LOCK bit continuously (during PLL start-up, usually) or at periodic intervals. In either case, only when the LOCK bit is set, is the PLLCLK clock safe to use as the source for the system and core clocks. If the PLL is selected as the source for the system and core clocks 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.

The following conditions apply when the PLL is in automatic bandwidth control mode (AUTO = 1):

- The TRACK bit is a read-only indicator of the mode of the filter.
- The TRACK bit is set when the VCO frequency is within a certain tolerance, Δ_{trk}, and is clear when the VCO frequency is out of a certain tolerance, Δ_{unt}.
- 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, Δ_{Lock}, and is cleared when the VCO frequency is out of a certain tolerance, Δ_{unl}.
- CPU interrupts can occur if enabled (LOCKIE = 1) when the lock condition changes, toggling the LOCK bit.

The PLL can also 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 the maximum system frequency (f_{sys}) and require fast start-up. The following conditions apply when in manual mode:

- ACQ is a writable control bit that controls the mode of the filter. Before turning on the PLL in manual mode, the ACQ bit should be asserted to configure the filter in acquisition mode.
- After turning on the PLL by setting the PLLON bit software must wait a given time (t_{acq}) before entering tracking mode (ACQ = 0).
- After entering tracking mode software must wait a given time (t_{al}) before selecting the PLLCLK as the source for system and core clocks (PLLSEL = 1).

4.4.2 System Clocks Generator

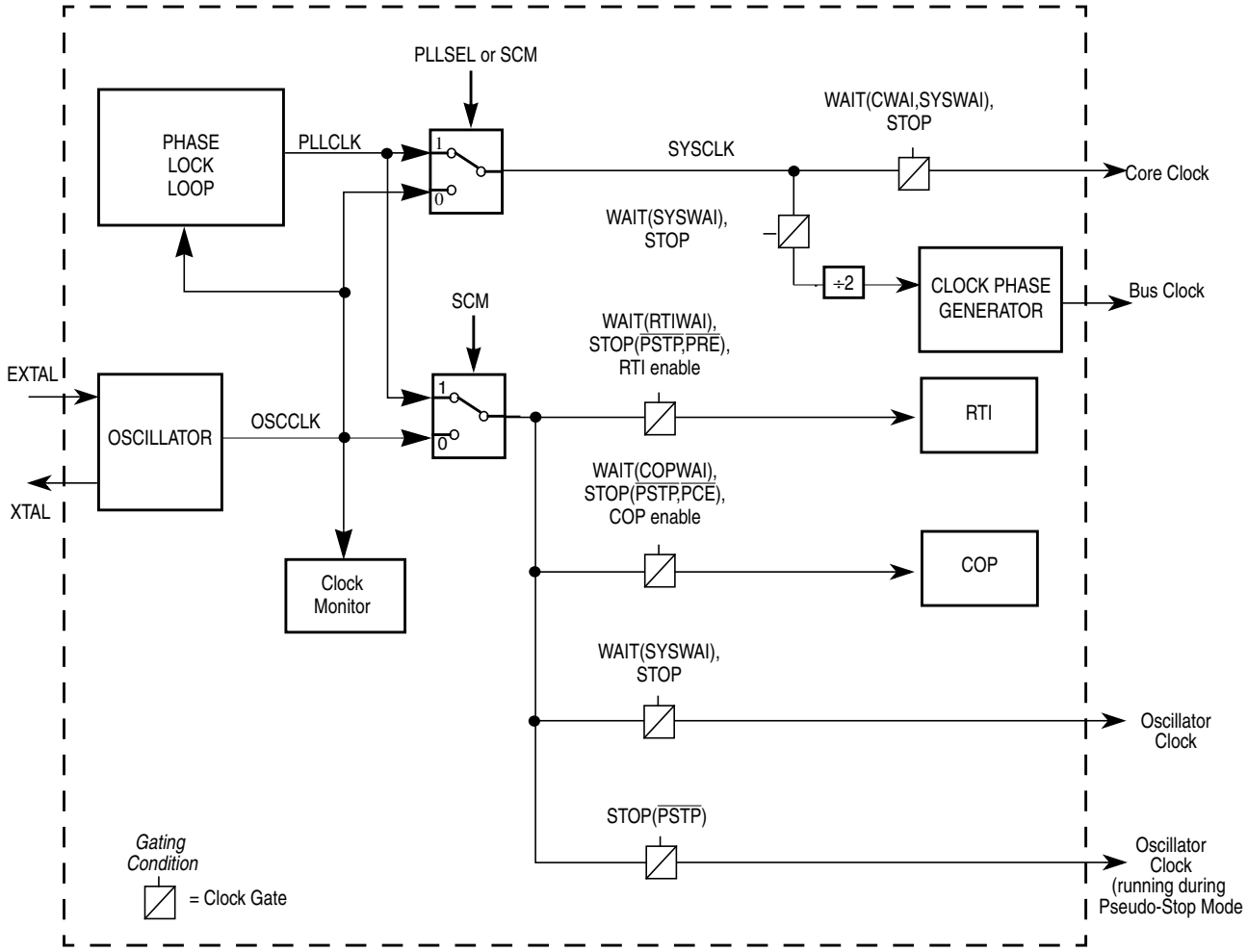


Figure 4-17. System Clocks Generator

The clock generator creates the clocks used in the MCU (see Figure 4-17). The gating condition placed on top of the individual clock gates indicates the dependencies of different modes (stop, wait) and the setting of the respective configuration bits.

The peripheral modules use the bus clock. Some peripheral modules also use the oscillator clock. The memory blocks use the bus clock. If the MCU enters self-clock mode (see Section 4.4.7.2, “Self-Clock Mode”), oscillator clock source is switched to PLLCLK running at its minimum frequency f_{SCM}. The bus clock is used to generate the clock visible at the ECLK pin. The core clock signal is the clock for the CPU. The core clock is twice the bus clock as shown in Figure 4-18. But note that a CPU cycle corresponds to one bus clock.

PLL clock mode is selected with PLLSEL bit in the CLKSEL register. When selected, the PLL output clock drives SYSCLK for the main system including the CPU and peripherals. The PLL cannot be turned off by clearing the PLLON bit, if the PLL clock is selected. When PLLSEL is changed, it takes a maximum

of 4 OSCCLK plus 4 PLLCLK cycles to make the transition. During the transition, all clocks freeze and CPU activity ceases.

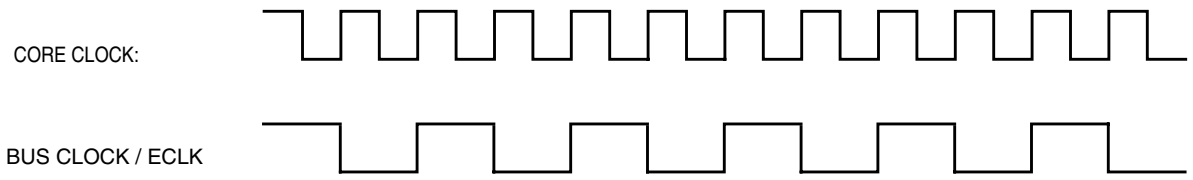


Figure 4-18. Core Clock and Bus Clock Relationship

4.4.3 Clock Monitor (CM)

If no OSCCLK edges are detected within a certain time, the clock monitor within the oscillator block generates a clock monitor fail event. The CRGV4 then asserts self-clock mode or generates a system reset depending on the state of SCME bit. If the clock monitor is disabled or the presence of clocks is detected no failure is indicated by the oscillator block.The clock monitor function is enabled/disabled by the CME control bit.

4.4.4 Clock Quality Checker

The clock monitor performs a coarse check on the incoming clock signal. The clock quality checker provides a more accurate check in addition to the clock monitor.

A clock quality check is triggered by any of the following events:

- Power-on reset (POR)
- Low voltage reset (LVR)
- Wake-up from full stop mode (exit full stop)
- Clock monitor fail indication (CM fail)

A time window of 50000 VCO clock cycles[1] is called *check window*.

A number greater equal than 4096 rising OSCCLK edges within a *check window* is called *osc ok*. Note that *osc ok* immediately terminates the current *check window*. See Figure 4-19 as an example.

1. VCO clock cycles are generated by the PLL when running at minimum frequency f_{SCM}.

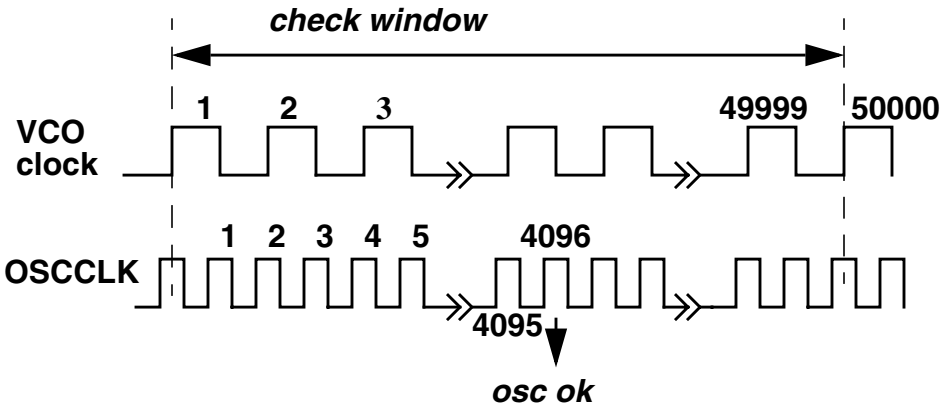


Figure 4-19. Check Window Example

The sequence for clock quality check is shown in Figure 4-20.

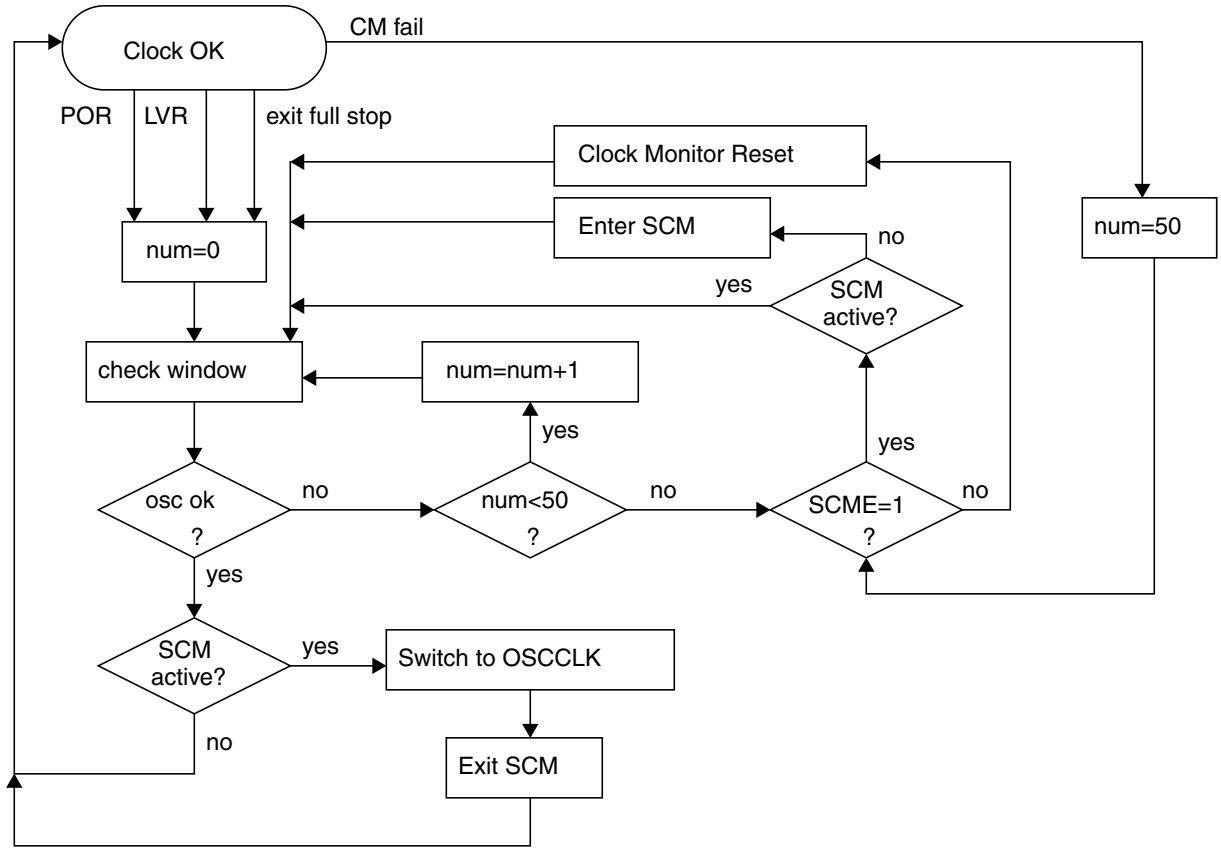


Figure 4-20. Sequence for Clock Quality Check

NOTE

Remember that in parallel to additional actions caused by self-clock mode or clock monitor reset[1] handling the clock quality checker **continues** to check the OSCCLK signal.

1. A Clock Monitor Reset will always set the SCME bit to logical'1'

NOTE

The clock quality checker enables the PLL and the voltage regulator (VREG) anytime a clock check has to be performed. An ongoing clock quality check could also cause a running PLL (f_{SCM}) and an active VREG during pseudo-stop mode or wait mode

4.4.5 Computer Operating Properly Watchdog (COP)

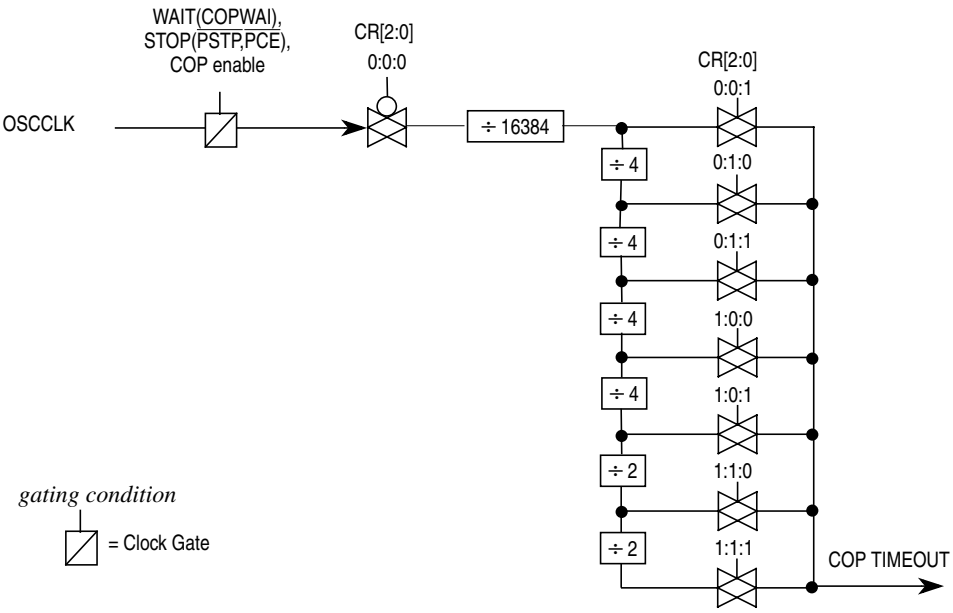


Figure 4-21. Clock Chain for COP

The COP (free running watchdog timer) enables the user to check that a program is running and sequencing properly. The COP is disabled out of reset. When the COP is being used, software is responsible for keeping the COP from timing out. If the COP times out it is an indication that the software is no longer being executed in the intended sequence; thus a system reset is initiated (see Section 4.5.2, "Computer Operating Properly Watchdog (COP) Reset)." The COP runs with a gated OSCCLK (see Section Figure 4-21., "Clock Chain for COP"). Three control bits in the COPCTL register allow selection of seven COP time-out periods.

When COP is enabled, the program must write 0x0055 and 0x00AA (in this order) to the ARMCOP register during the selected time-out period. As soon as this is done, the COP time-out period is restarted. If the program fails to do this and the COP times out, the part will reset. Also, if any value other than 0x0055 or 0x00AA is written, the part is immediately reset.

Windowed COP operation is enabled by setting WCOP in the COPCTL register. In this mode, writes to the ARMCOP register to clear the COP timer must occur in the last 25% of the selected time-out period. A premature write will immediately reset the part.

If PCE bit is set, the COP will continue to run in pseudo-stop mode.

4.4.6 Real-Time Interrupt (RTI)

The RTI can be used to generate a hardware interrupt at a fixed periodic rate. If enabled (by setting RTIE=1), this interrupt will occur at the rate selected by the RTICTL register. The RTI runs with a gated OSCCLK (see Section Figure 4-22., “Clock Chain for RTI”). At the end of the RTI time-out period the RTIF flag is set to 1 and a new RTI time-out period starts immediately.

A write to the RTICTL register restarts the RTI time-out period.

If the PRE bit is set, the RTI will continue to run in pseudo-stop mode.

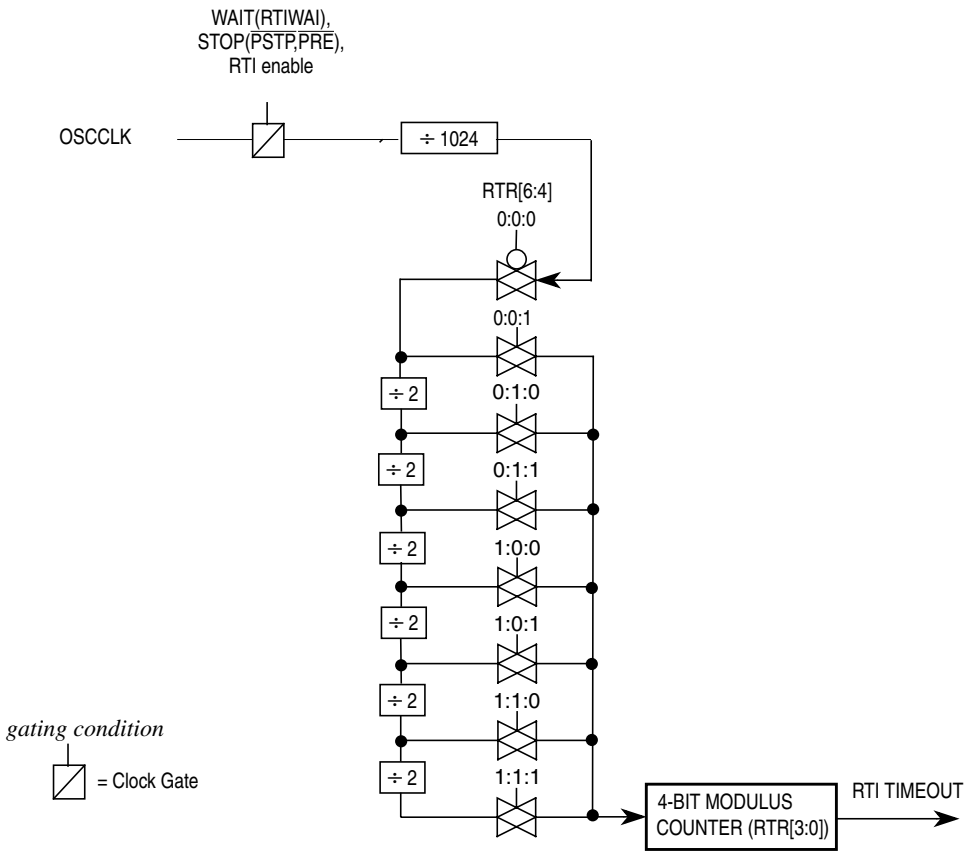


Figure 4-22. Clock Chain for RTI

4.4.7 Modes of Operation

4.4.7.1 Normal Mode

The CRGV4 block behaves as described within this specification in all normal modes.

4.4.7.2 Self-Clock Mode

The VCO has a minimum operating frequency, f_{SCM}. If the external clock frequency is not available due to a failure or due to long crystal start-up time, the bus clock and the core clock are derived from the VCO

running at minimum operating frequency; this mode of operation is called self-clock mode. This requires CME = 1 and SCME = 1. If the MCU was clocked by the PLL clock prior to entering self-clock mode, the PLLSEL bit will be cleared. If the external clock signal has stabilized again, the CRG will automatically select OSCCLK to be the system clock and return to normal mode. See Section 4.4.4, “Clock Quality Checker” for more information on entering and leaving self-clock mode.

NOTE

In order to detect a potential clock loss, the CME bit should be always enabled (CME=1).

If CME bit is disabled and the MCU is configured to run on PLL clock (PLLCLK), a loss of external clock (OSCCLK) will not be detected and will cause the system clock to drift towards the VCO’s minimum frequency f_{SCM}. As soon as the external clock is available again the system clock ramps up to its PLL target frequency. If the MCU is running on external clock any loss of clock will cause the system to go static.

4.4.8 Low-Power Operation in Run Mode

The RTI can be stopped by setting the associated rate select bits to 0.

The COP can be stopped by setting the associated rate select bits to 0.

4.4.9 Low-Power Operation in Wait Mode

The WAI instruction puts the MCU in a low power consumption stand-by mode depending on setting of the individual bits in the CLKSEL register. All individual wait mode configuration bits can be superposed. This provides enhanced granularity in reducing the level of power consumption during wait mode. Table 4-10 lists the individual configuration bits and the parts of the MCU that are affected in wait mode.

Table 4-10. MCU Configuration During Wait Mode

	PLLWAI	CWAI	SYSWAI	RTIWAI	COPWAI	ROAWAI
PLL	stopped	—	—	—	—	—
Core	—	stopped	stopped	—	—	—
System	—	—	stopped	—	—	—
RTI	—	—	—	stopped	—	—
COP	—	—	—	—	stopped	—
Oscillator	—	—	—	—	—	reduced[1]

[1] Refer to oscillator block description for availability of a reduced oscillator amplitude.

After executing the WAI instruction the core requests the CRG to switch MCU into wait mode. The CRG then checks whether the PLLWAI, CWAI and SYSWAI bits are asserted (see Figure 4-23). Depending on the configuration the CRG switches the system and core clocks to OSCCLK by clearing the PLLSEL bit, disables the PLL, disables the core clocks and finally disables the remaining system clocks. As soon as all clocks are switched off wait mode is active.

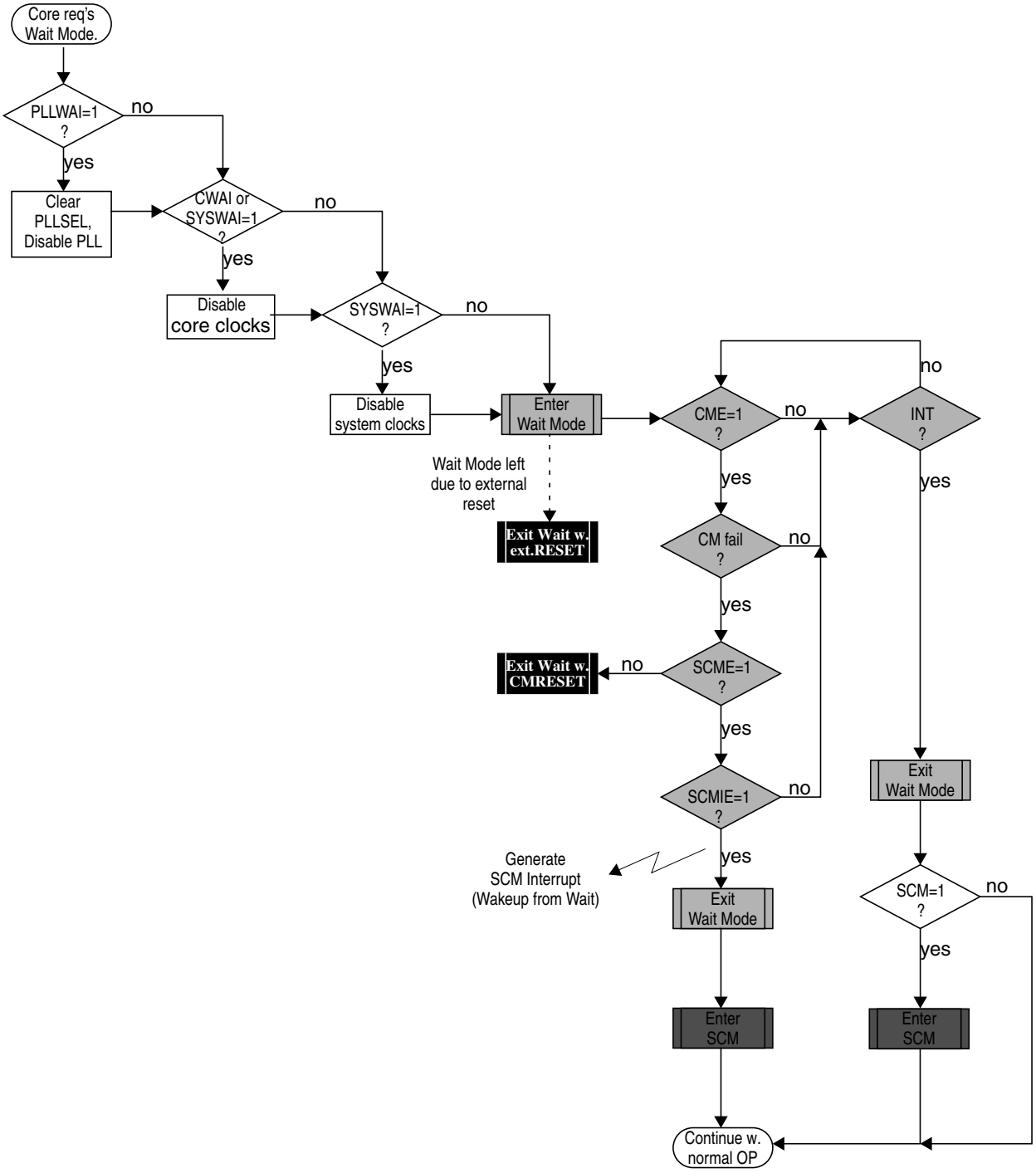


Figure 4-23. Wait Mode Entry/Exit Sequence

There are five different scenarios for the CRG to restart the MCU from wait mode:

- External reset
- Clock monitor reset
- COP reset
- Self-clock mode interrupt
- Real-time interrupt (RTI)

If the MCU gets an external reset during wait mode active, the CRG asynchronously restores all configuration bits in the register space to its default settings and starts the reset generator. After completing the reset sequence processing begins by fetching the normal reset vector. Wait mode is exited and the MCU is in run mode again.

If the clock monitor is enabled (CME=1) the MCU is able to leave wait mode when loss of oscillator/external clock is detected by a clock monitor fail. If the SCME bit is not asserted the CRG generates a clock monitor fail reset (CMRESET). The CRG's behavior for CMRESET is the same compared to external reset, but another reset vector is fetched after completion of the reset sequence. If the SCME bit is asserted the CRG generates a SCM interrupt if enabled (SCMIE=1). After generating the interrupt the CRG enters self-clock mode and starts the clock quality checker (see Section 4.4.4, "Clock Quality Checker"). Then the MCU continues with normal operation.If the SCM interrupt is blocked by SCMIE = 0, the SCMIF flag will be asserted and clock quality checks will be performed but the MCU will not wake-up from wait mode.

If any other interrupt source (e.g. RTI) triggers exit from wait mode the MCU immediately continues with normal operation. If the PLL has been powered-down during wait mode the PLLSEL bit is cleared and the MCU runs on OSCCLK after leaving wait mode. The software must manually set the PLLSEL bit again, in order to switch system and core clocks to the PLLCLK.

If wait mode is entered from self-clock mode, the CRG will continue to check the clock quality until clock check is successful. The PLL and voltage regulator (VREG) will remain enabled.

Table 4-11 summarizes the outcome of a clock loss while in wait mode.

Table 4-11. Outcome of Clock Loss in Wait Mode

CME	SCME	SCMIE	CRG Actions
0	X	X	Clock failure --> No action, clock loss not detected.
1	0	X	Clock failure --> CRG performs Clock Monitor Reset immediately
1	1	0	Clock failure --> Scenario 1: OSCCLK **recovers** prior to exiting Wait Mode. – MCU remains in Wait Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag. *Some time later OSCCLK recovers.* – CM no longer indicates a failure, – 4096 OSCCLK cycles later Clock Quality Check indicates clock o.k., – SCM deactivated, – PLL disabled depending on PLLWAI, – VREG remains enabled *(never gets disabled in Wait Mode)*. – MCU remains in Wait Mode. *Some time later either a wakeup interrupt occurs (no SCM interrupt)* – Exit Wait Mode using OSCCLK as system clock (SYSCLK), – Continue normal operation. *or an External Reset is applied.* – Exit Wait Mode using OSCCLK as system clock, – Start reset sequence. Scenario 2: OSCCLK **does not recover** prior to exiting Wait Mode. – MCU remains in Wait Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag, – Keep performing Clock Quality Checks (could continue infinitely) while in Wait Mode. *Some time later either a wakeup interrupt occurs (no SCM interrupt)* – Exit Wait Mode in SCM using PLL clock (f_{SCM}) as system clock, – Continue to perform additional Clock Quality Checks until OSCCLK is o.k. again. or an External RESET is applied. – Exit Wait Mode in SCM using PLL clock (f_{SCM}) as system clock, – Start reset sequence, – Continue to perform additional Clock Quality Checks until OSCCLK is o.k.again.

Table 4-11. Outcome of Clock Loss in Wait Mode (continued)

CME	SCME	SCMIE	CRG Actions
1	1	1	Clock failure --> – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – SCMIF set. SCMIF generates Self-Clock Mode wakeup interrupt. – Exit Wait Mode in SCM using PLL clock (f_{SCM}) as system clock, – Continue to perform a additional Clock Quality Checks until OSCCLK is o.k. again.

4.4.10 Low-Power Operation in Stop Mode

All clocks are stopped in STOP mode, dependent of the setting of the PCE, PRE and PSTP bit. The oscillator is disabled in STOP mode unless the PSTP bit is set. All counters and dividers remain frozen but do not initialize. If the PRE or PCE bits are set, the RTI or COP continues to run in pseudo-stop mode. In addition to disabling system and core clocks the CRG requests other functional units of the MCU (e.g. voltage-regulator) to enter their individual power-saving modes (if available). This is the main difference between pseudo-stop mode and wait mode.

After executing the STOP instruction the core requests the CRG to switch the MCU into stop mode. If the PLLSEL bit remains set when entering stop mode, the CRG will switch the system and core clocks to OSCCLK by clearing the PLLSEL bit. Then the CRG disables the PLL, disables the core clock and finally disables the remaining system clocks. As soon as all clocks are switched off, stop mode is active.

If pseudo-stop mode (PSTP = 1) is entered from self-clock mode the CRG will continue to check the clock quality until clock check is successful. The PLL and the voltage regulator (VREG) will remain enabled. If full stop mode (PSTP = 0) is entered from self-clock mode an ongoing clock quality check will be stopped. A complete timeout window check will be started when stop mode is exited again.

Wake-up from stop mode also depends on the setting of the PSTP bit.

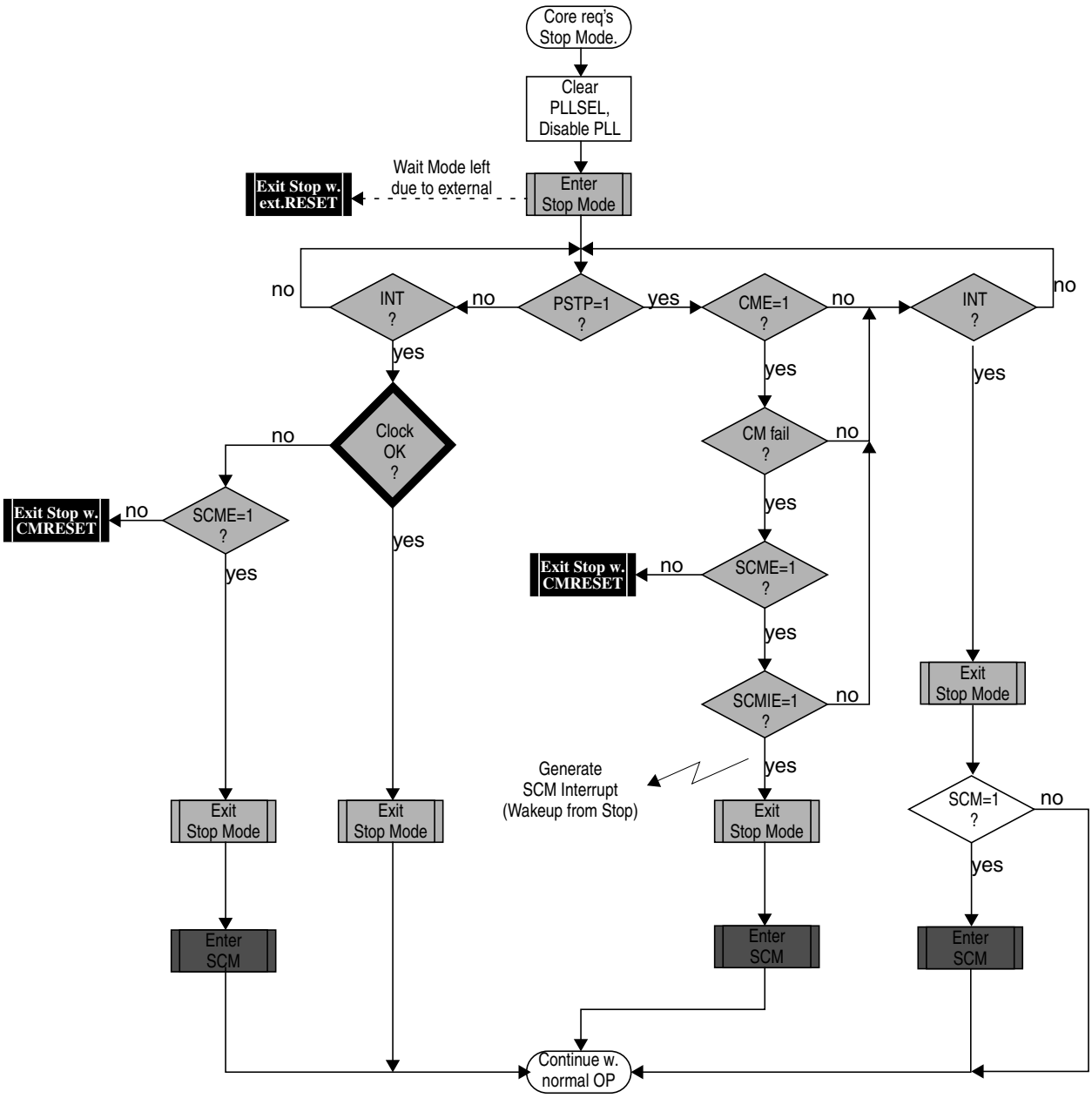


Figure 4-24. Stop Mode Entry/Exit Sequence

4.4.10.1 Wake-Up from Pseudo-Stop (PSTP=1)

Wake-up from pseudo-stop is the same as wake-up from wait mode. There are also three different scenarios for the CRG to restart the MCU from pseudo-stop mode:

- External reset
- Clock monitor fail
- Wake-up interrupt

If the MCU gets an external reset during pseudo-stop mode active, the CRG asynchronously restores all configuration bits in the register space to its default settings and starts the reset generator. After completing the reset sequence processing begins by fetching the normal reset vector. Pseudo-stop mode is exited and the MCU is in run mode again.

If the clock monitor is enabled (CME = 1) the MCU is able to leave pseudo-stop mode when loss of oscillator/external clock is detected by a clock monitor fail. If the SCME bit is not asserted the CRG generates a clock monitor fail reset (CMRESET). The CRG's behavior for CMRESET is the same compared to external reset, but another reset vector is fetched after completion of the reset sequence. If the SCME bit is asserted the CRG generates a SCM interrupt if enabled (SCMIE=1). After generating the interrupt the CRG enters self-clock mode and starts the clock quality checker (see Section 4.4.4, "Clock Quality Checker"). Then the MCU continues with normal operation. If the SCM interrupt is blocked by SCMIE = 0, the SCMIF flag will be asserted but the CRG will not wake-up from pseudo-stop mode.

If any other interrupt source (e.g. RTI) triggers exit from pseudo-stop mode the MCU immediately continues with normal operation. Because the PLL has been powered-down during stop mode the PLLSEL bit is cleared and the MCU runs on OSCCLK after leaving stop mode. The software must set the PLLSEL bit again, in order to switch system and core clocks to the PLLCLK.

Table 4-12 summarizes the outcome of a clock loss while in pseudo-stop mode.

Table 4-12. Outcome of Clock Loss in Pseudo-Stop Mode

CME	SCME	SCMIE	CRG Actions
0	X	X	Clock failure --> No action, clock loss not detected.
1	0	X	Clock failure --> CRG performs Clock Monitor Reset immediately
1	1	0	Clock Monitor failure --> Scenario 1: OSCCLK **recovers** prior to exiting Pseudo-Stop Mode. – MCU remains in Pseudo-Stop Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag. *Some time later OSCCLK recovers.* – CM no longer indicates a failure, – 4096 OSCCLK cycles later Clock Quality Check indicates clock o.k., – SCM deactivated, – PLL disabled, – VREG disabled. – MCU remains in Pseudo-Stop Mode. S*ome time later either a wakeup interrupt occurs (no SCM interrupt)* – Exit Pseudo-Stop Mode using OSCCLK as system clock (SYSCLK), – Continue normal operation. *or an External Reset is applied.* – Exit Pseudo-Stop Mode using OSCCLK as system clock, – Start reset sequence. Scenario 2: OSCCLK **does not recover** prior to exiting Pseudo-Stop Mode. – MCU remains in Pseudo-Stop Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag, – Keep performing Clock Quality Checks (could continue infinitely) while in Pseudo-Stop Mode. S*ome time later either a wakeup interrupt occurs (no SCM interrupt)* – Exit Pseudo-Stop Mode in SCM using PLL clock (f_{SCM}) as system clock – Continue to perform additional Clock Quality Checks until OSCCLK is o.k. again. or an External RESET is applied. – Exit Pseudo-Stop Mode in SCM using PLL clock (f_{SCM}) as system clock – Start reset sequence, – Continue to perform additional Clock Quality Checks until OSCCLK is o.k.again.

Table 4-12. Outcome of Clock Loss in Pseudo-Stop Mode (continued)

CME	SCME	SCMIE	CRG Actions
1	1	1	Clock failure --> – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – SCMIF set. SCMIF generates Self-Clock Mode wakeup interrupt. – Exit Pseudo-Stop Mode in SCM using PLL clock (f_{SCM}) as system clock, – Continue to perform a additional Clock Quality Checks until OSCCLK is o.k. again.

4.4.10.2 Wake-up from Full Stop (PSTP=0)

The MCU requires an external interrupt or an external reset in order to wake-up from stop mode.

If the MCU gets an external reset during full stop mode active, the CRG asynchronously restores all configuration bits in the register space to its default settings and will perform a maximum of 50 clock *check_windows* (see Section 4.4.4, “Clock Quality Checker”). After completing the clock quality check the CRG starts the reset generator. After completing the reset sequence processing begins by fetching the normal reset vector. Full stop mode is exited and the MCU is in run mode again.

If the MCU is woken-up by an interrupt, the CRG will also perform a maximum of 50 clock *check_windows* (see Section 4.4.4, “Clock Quality Checker”). If the clock quality check is successful, the CRG will release all system and core clocks and will continue with normal operation. If all clock checks within the timeout-window are failing, the CRG will switch to self-clock mode or generate a clock monitor reset (CMRESET) depending on the setting of the SCME bit.

Because the PLL has been powered-down during stop mode the PLLSEL bit is cleared and the MCU runs on OSCCLK after leaving stop mode. The software must manually set the PLLSEL bit again, in order to switch system and core clocks to the PLLCLK.

NOTE

In full stop mode, the clock monitor is disabled and any loss of clock will not be detected.

4.5 Resets

This section describes how to reset the CRGV4 and how the CRGV4 itself controls the reset of the MCU. It explains all special reset requirements. Because the reset generator for the MCU is part of the CRG, this section also describes all automatic actions that occur during or as a result of individual reset conditions. The reset values of registers and signals are provided in Section 4.3, “Memory Map and Register

Definition." All reset sources are listed in Table 4-13. Refer to the device overview chapter for related vector addresses and priorities.

Table 4-13. Reset Summary

Reset Source	Local Enable
Power-on Reset	None
Low Voltage Reset	None
External Reset	None
Clock Monitor Reset	PLLCTL (CME=1, SCME=0)
COP Watchdog Reset	COPCTL (CR[2:0] nonzero)

The reset sequence is initiated by any of the following events:

- Low level is detected at the $\overline{\text{RESET}}$ pin (external reset).
- Power on is detected.
- Low voltage is detected.
- COP watchdog times out.
- Clock monitor failure is detected and self-clock mode was disabled (SCME = 0).

Upon detection of any reset event, an internal circuit drives the $\overline{\text{RESET}}$ pin low for 128 SYSCLK cycles (see Figure 4-25). Because entry into reset is asynchronous it does not require a running SYSCLK. However, the internal reset circuit of the CRGV4 cannot sequence out of current reset condition without a running SYSCLK. The number of 128 SYSCLK cycles might be increased by n = 3 to 6 additional SYSCLK cycles depending on the internal synchronization latency. After 128+n SYSCLK cycles the $\overline{\text{RESET}}$ pin is released. The reset generator of the CRGV4 waits for additional 64 SYSCLK cycles and then samples the RESET pin to determine the originating source. Table 4-14 shows which vector will be fetched.

Table 4-14. Reset Vector Selection

Sampled $\overline{\text{RESET}}$ Pin (64 Cycles After Release)	Clock Monitor Reset Pending	COP Reset Pending	Vector Fetch
1	0	0	POR / LVR / External Reset
1	1	X	Clock Monitor Reset
1	0	1	COP Reset
0	X	X	POR / LVR / External Reset with rise of $\overline{\text{RESET}}$ pin

NOTE

External circuitry connected to the $\overline{\text{RESET}}$ pin should not include a large capacitance that would interfere with the ability of this signal to rise to a valid logic 1 within 64 SYSCLK cycles after the low drive is released.

The internal reset of the MCU remains asserted while the reset generator completes the 192 SYSCLK long reset sequence. The reset generator circuitry always makes sure the internal reset is deasserted synchronously after completion of the 192 SYSCLK cycles. In case the RESET pin is externally driven low for more than these 192 SYSCLK cycles (external reset), the internal reset remains asserted too.

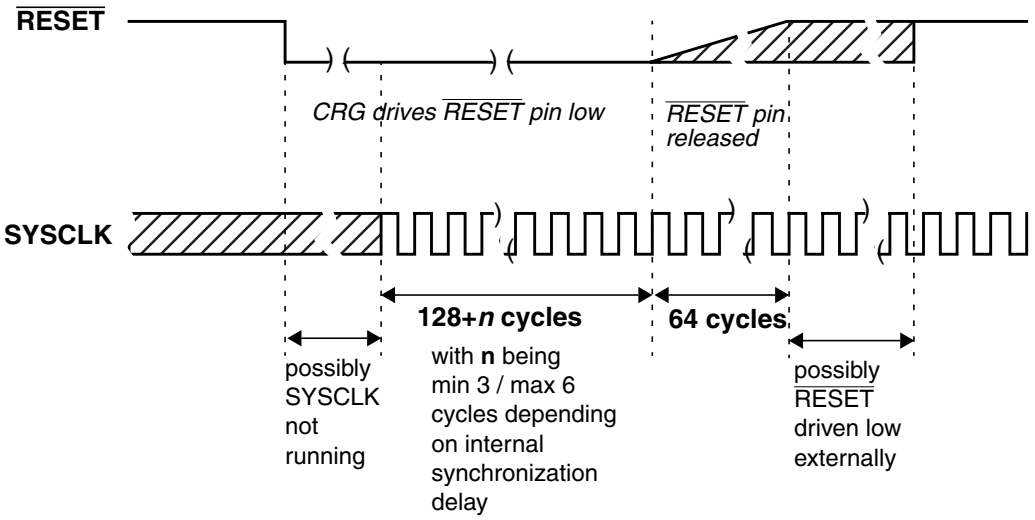


Figure 4-25. RESET Timing

4.5.1 Clock Monitor Reset

The CRGV4 generates a clock monitor reset in case all of the following conditions are true:

- Clock monitor is enabled (CME=1)
- Loss of clock is detected
- Self-clock mode is disabled (SCME=0)

The reset event asynchronously forces the configuration registers to their default settings (see Section 4.3, “Memory Map and Register Definition”). In detail the CME and the SCME are reset to logical ‘1’ (which doesn’t change the state of the CME bit, because it has already been set). As a consequence, the CRG immediately enters self-clock mode and starts its internal reset sequence. In parallel the clock quality check starts. As soon as clock quality check indicates a valid oscillator clock the CRG switches to OSCCLK and leaves self-clock mode. Because the clock quality checker is running in parallel to the reset generator, the CRG may leave self-clock mode while completing the internal reset sequence. When the reset sequence is finished the CRG checks the internally latched state of the clock monitor fail circuit. If a clock monitor fail is indicated processing begins by fetching the clock monitor reset vector.

4.5.2 Computer Operating Properly Watchdog (COP) Reset

When COP is enabled, the CRG expects sequential write of 0x0055 and 0x00AA (in this order) to the ARMCOP register during the selected time-out period. As soon as this is done, the COP time-out period restarts. If the program fails to do this the CRG will generate a reset. Also, if any value other than 0x0055 or 0x00AA is written, the CRG immediately generates a reset. In case windowed COP operation is enabled

writes (0x0055 or 0x00AA) to the ARMCOP register must occur in the last 25% of the selected time-out period. A premature write the CRG will immediately generate a reset.

As soon as the reset sequence is completed the reset generator checks the reset condition. If no clock monitor failure is indicated and the latched state of the COP timeout is true, processing begins by fetching the COP vector.

4.5.3 Power-On Reset, Low Voltage Reset

The on-chip voltage regulator detects when V_{DD} to the MCU has reached a certain level and asserts power-on reset or low voltage reset or both. As soon as a power-on reset or low voltage reset is triggered the CRG performs a quality check on the incoming clock signal. As soon as clock quality check indicates a valid oscillator clock signal the reset sequence starts using the oscillator clock. If after 50 check windows the clock quality check indicated a non-valid oscillator clock the reset sequence starts using self-clock mode.

Figure 4-26 and Figure 4-27 show the power-up sequence for cases when the $\overline{\text{RESET}}$ pin is tied to V_{DD} and when the $\overline{\text{RESET}}$ pin is held low.

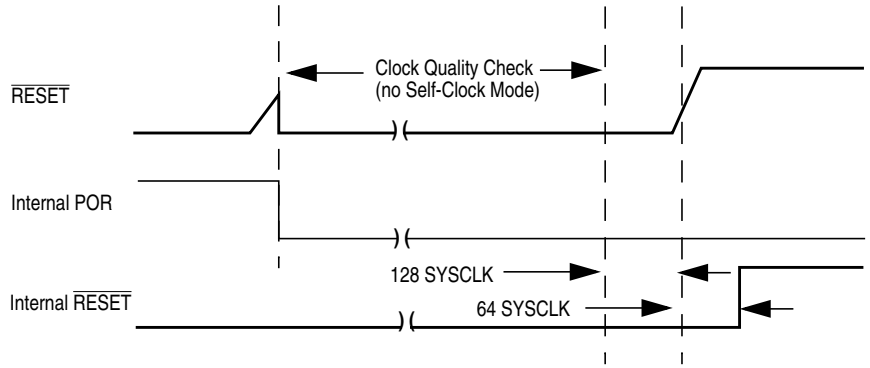


Figure 4-26. $\overline{\text{RESET}}$ Pin Tied to V_{DD} (by a Pull-Up Resistor)

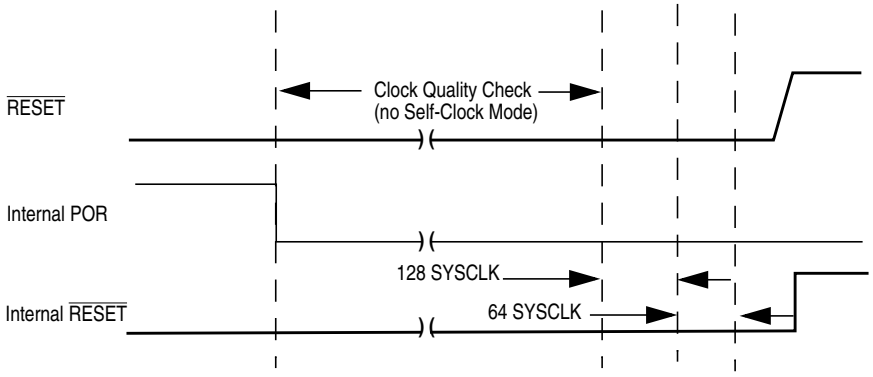


Figure 4-27. $\overline{\text{RESET}}$ Pin Held Low Externally

4.6 Interrupts

The interrupts/reset vectors requested by the CRG are listed in Table 4-15. Refer to the device overview chapter for related vector addresses and priorities.

Table 4-15. CRG Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Real-time interrupt	I bit	CRGINT (RTIE)
LOCK interrupt	I bit	CRGINT (LOCKIE)
SCM interrupt	I bit	CRGINT (SCMIE)

4.6.1 Real-Time Interrupt

The CRGV4 generates a real-time interrupt when the selected interrupt time period elapses. RTI interrupts are locally disabled by setting the RTIE bit to 0. The real-time interrupt flag (RTIF) is set to 1 when a timeout occurs, and is cleared to 0 by writing a 1 to the RTIF bit.

The RTI continues to run during pseudo-stop mode if the PRE bit is set to 1. This feature can be used for periodic wakeup from pseudo-stop if the RTI interrupt is enabled.

4.6.2 PLL Lock Interrupt

The CRGV4 generates a PLL lock interrupt when the LOCK condition of the PLL has changed, either from a locked state to an unlocked state or vice versa. Lock interrupts are locally disabled by setting the LOCKIE bit to 0. The PLL Lock interrupt flag (LOCKIF) is set to1 when the LOCK condition has changed, and is cleared to 0 by writing a 1 to the LOCKIF bit.

4.6.3 Self-Clock Mode Interrupt

The CRGV4 generates a self-clock mode interrupt when the SCM condition of the system has changed, either entered or exited self-clock mode. SCM conditions can only change if the self-clock mode enable bit (SCME) is set to 1. SCM conditions are caused by a failing clock quality check after power-on reset (POR) or low voltage reset (LVR) or recovery from full stop mode (PSTP = 0) or clock monitor failure. For details on the clock quality check refer to Section 4.4.4, “Clock Quality Checker.” If the clock monitor is enabled (CME = 1) a loss of external clock will also cause a SCM condition (SCME = 1).

SCM interrupts are locally disabled by setting the SCMIE bit to 0. The SCM interrupt flag (SCMIF) is set to 1 when the SCM condition has changed, and is cleared to 0 by writing a 1 to the SCMIF bit.

Chapter 5
Oscillator (OSCV2)

5.1 Introduction

The OSCV2 module provides two alternative oscillator concepts:

- A low noise and low power Colpitts oscillator with amplitude limitation control (ALC)
- A robust full swing Pierce oscillator with the possibility to feed in an external square wave

5.1.1 Features

The Colpitts OSCV2 option provides the following features:

- Amplitude limitation control (ALC) loop:
  - Low power consumption and low current induced RF emission
  - Sinusoidal waveform with low RF emission
  - Low crystal stress (an external damping resistor is not required)
  - Normal and low amplitude mode for further reduction of power and emission
- An external biasing resistor is not required

The Pierce OSC option provides the following features:

- Wider high frequency operation range
- No DC voltage applied across the crystal
- Full rail-to-rail (2.5 V nominal) swing oscillation with low EM susceptibility
- Fast start up

Common features:

- Clock monitor (CM)
- Operation from the V_{DDPLL} 2.5 V (nominal) supply rail

5.1.2 Modes of Operation

Two modes of operation exist:

- Amplitude limitation controlled Colpitts oscillator mode suitable for power and emission critical applications
- Full swing Pierce oscillator mode that can also be used to feed in an externally generated square wave suitable for high frequency operation and harsh environments

5.2 External Signal Description

This section lists and describes the signals that connect off chip.

5.2.1 V_{DDPLL} and V_{SSPLL} — PLL Operating Voltage, PLL Ground

These pins provide the operating voltage (V_{DDPLL}) and ground (V_{SSPLL}) for the OSCV2 circuitry. This allows the supply voltage to the OSCV2 to be independently bypassed.

5.2.2 EXTAL and XTAL — Clock/Crystal Source Pins

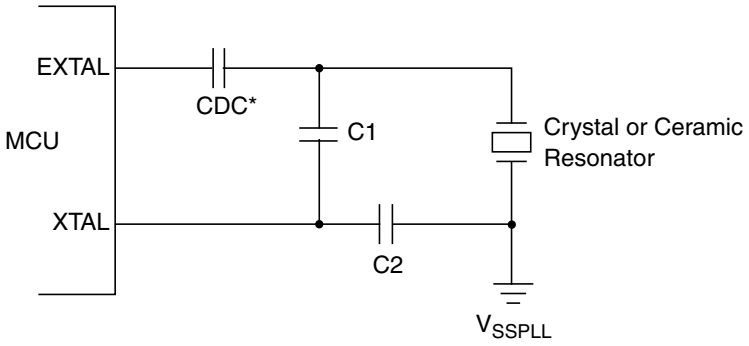
These pins provide the interface for either a crystal or a CMOS compatible clock to control the internal clock generator circuitry. EXTAL is the external clock input or the input to the crystal oscillator amplifier. XTAL is the output of the crystal oscillator amplifier. All the MCU internal system clocks are derived from the EXTAL input frequency. In full stop mode (PSTP = 0) the EXTAL pin is pulled down by an internal resistor of typical 200 kΩ.

NOTE

Freescale Semiconductor recommends an evaluation of the application board and chosen resonator or crystal by the resonator or crystal supplier.

The Crystal circuit is changed from standard.

The Colpitts circuit is not suited for overtone resonators and crystals.



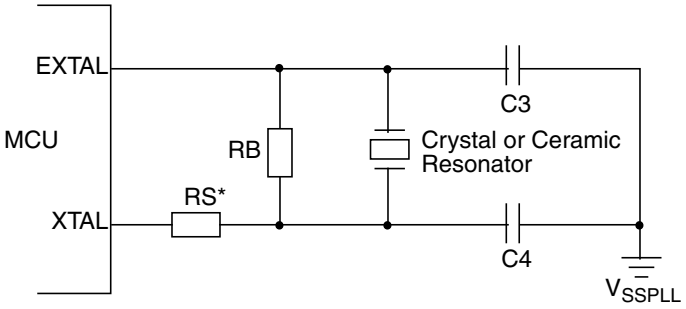
* Due to the nature of a translated ground Colpitts oscillator a DC voltage bias is applied to the crystal.

Please contact the crystal manufacturer for crystal DC bias conditions and recommended capacitor value CDC.

Figure 5-1. Colpitts Oscillator Connections (XCLKS = 0)

NOTE

The Pierce circuit is not suited for overtone resonators and crystals without a careful component selection.



* Rs can be zero (shorted) when used with higher frequency crystals.
Refer to manufacturer's data.

Figure 5-2. Pierce Oscillator Connections (XCLKS = 1)

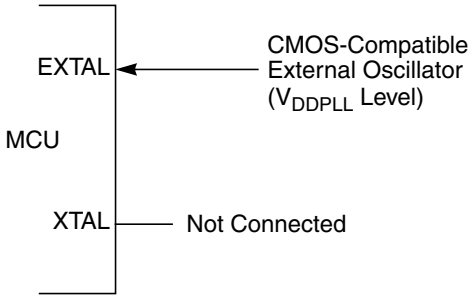


Figure 5-3. External Clock Connections (XCLKS = 1)

5.2.3 XCLKS — Colpitts/Pierce Oscillator Selection Signal

The XCLKS is an input signal which controls whether a crystal in combination with the internal Colpitts (low power) oscillator is used or whether the Pierce oscillator/external clock circuitry is used. The XCLKS signal is sampled during reset with the rising edge of $\overline{\text{RESET}}$. Table 5-1 lists the state coding of the sampled XCLKS signal. Refer to the device overview chapter for polarity of the XCLKS pin.

Table 5-1. Clock Selection Based on XCLKS

XCLKS	Description
0	Colpitts oscillator selected
1	Pierce oscillator/external clock selected

5.3 Memory Map and Register Definition

The CRG contains the registers and associated bits for controlling and monitoring the OSCV2 module.

5.4 Functional Description

The OSCV2 block has two external pins, EXTAL and XTAL. The oscillator input pin, EXTAL, is intended to be connected to either a crystal or an external clock source. The selection of Colpitts oscillator or Pierce oscillator/external clock depends on the XCLKS signal which is sampled during reset. The XTAL pin is an output signal that provides crystal circuit feedback.

A buffered EXTAL signal, OSCCLK, becomes the internal reference clock. To improve noise immunity, the oscillator is powered by the V_{DDPLL} and V_{SSPLL} power supply pins.

The Pierce oscillator can be used for higher frequencies compared to the low power Colpitts oscillator.

5.4.1 Amplitude Limitation Control (ALC)

The Colpitts oscillator is equipped with a feedback system which does not waste current by generating harmonics. Its configuration is “Colpitts oscillator with translated ground.” The transconductor used is driven by a current source under the control of a peak detector which will measure the amplitude of the AC signal appearing on EXTAL node in order to implement an amplitude limitation control (ALC) loop. The ALC loop is in charge of reducing the quiescent current in the transconductor as a result of an increase in the oscillation amplitude. The oscillation amplitude can be limited to two values. The normal amplitude which is intended for non power saving modes and a small amplitude which is intended for low power operation modes. Please refer to the CRG block description chapter for the control and assignment of the amplitude value to operation modes.

5.4.2 Clock Monitor (CM)

The clock monitor circuit is based on an internal resistor-capacitor (RC) time delay so that it can operate without any MCU clocks. If no OSCCLK edges are detected within this RC time delay, the clock monitor indicates a failure which asserts self clock mode or generates a system reset depending on the state of SCME bit. If the clock monitor is disabled or the presence of clocks is detected no failure is indicated.The clock monitor function is enabled/disabled by the CME control bit, described in the CRG block description chapter.

5.5 Interrupts

OSCV2 contains a clock monitor, which can trigger an interrupt or reset. The control bits and status bits for the clock monitor are described in the CRG block description chapter.

Chapter 6
Timer Module (TIM16B4CV1)

6.1 Introduction

The basic timer consists of a 16-bit, software-programmable counter driven by a seven-stage programmable prescaler.

This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from microseconds to many seconds.

This timer contains 4 complete input capture/output compare channels IOC[7:4] and one pulse accumulator. The input capture function is used to detect a selected transition edge and record the time. The output compare function is used for generating output signals or for timer software delays. The 16-bit pulse accumulator is used to operate as a simple event counter or a gated time accumulator. The pulse accumulator shares timer channel 7 when in event mode.

A full access for the counter registers or the input capture/output compare registers should take place in one clock cycle. Accessing high byte and low byte separately for all of these registers may not yield the same result as accessing them in one word.

6.1.1 Features

The TIM16B4CV1 includes these distinctive features:

- Four input capture/output compare channels.
- Clock prescaling.
- 16-bit counter.
- 16-bit pulse accumulator.

6.1.2 Modes of Operation

Stop: Timer is off because clocks are stopped.

Freeze: Timer counter keep on running, unless TSFRZ in TSCR (0x0006) is set to 1.

Wait: Counters keep on running, unless TSWAI in TSCR (0x0006) is set to 1.

Normal: Timer counter keep on running, unless TEN in TSCR (0x0006) is cleared to 0.

6.1.3 Block Diagrams

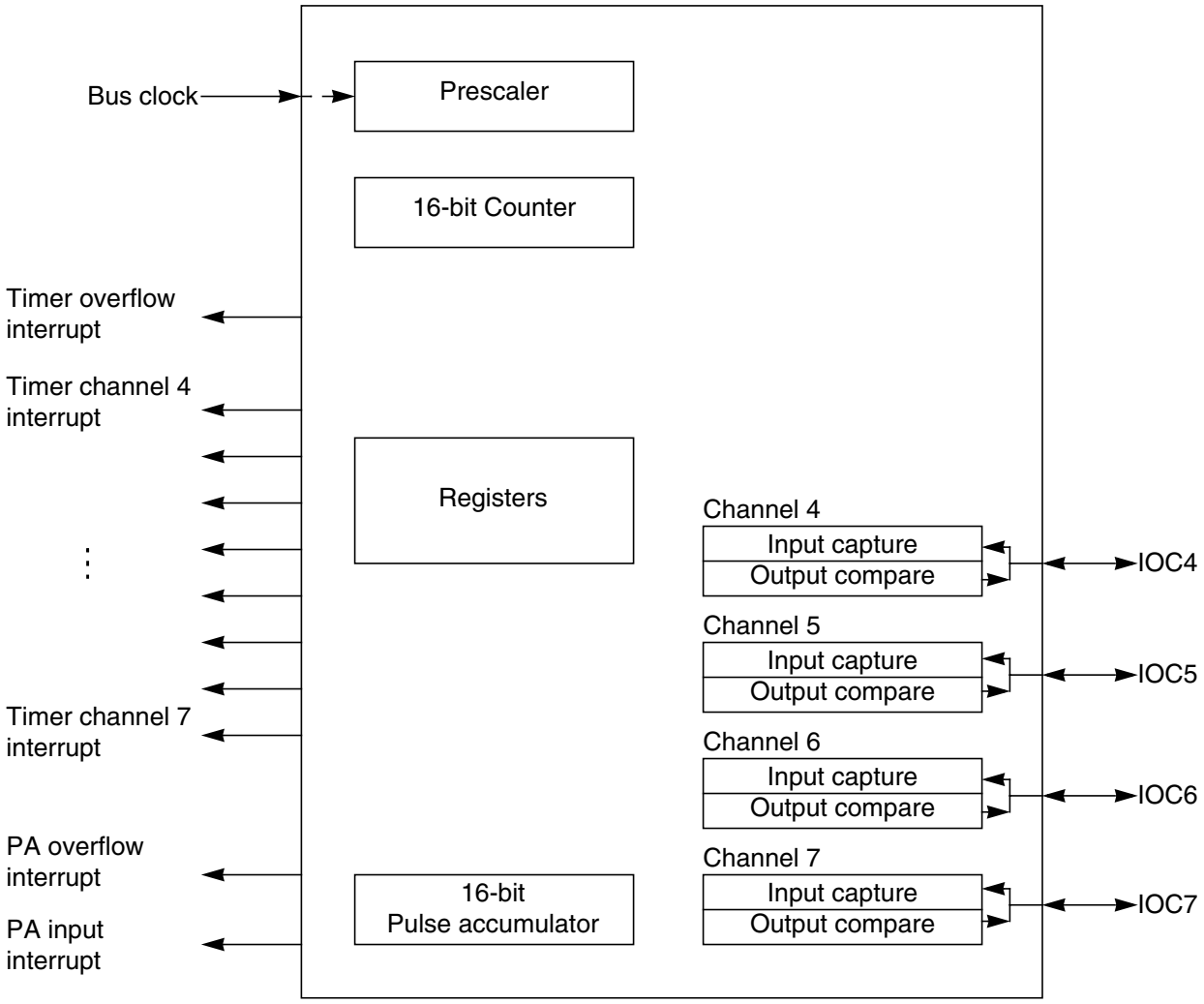


Figure 6-1. TIM16B4CV1 Block Diagram

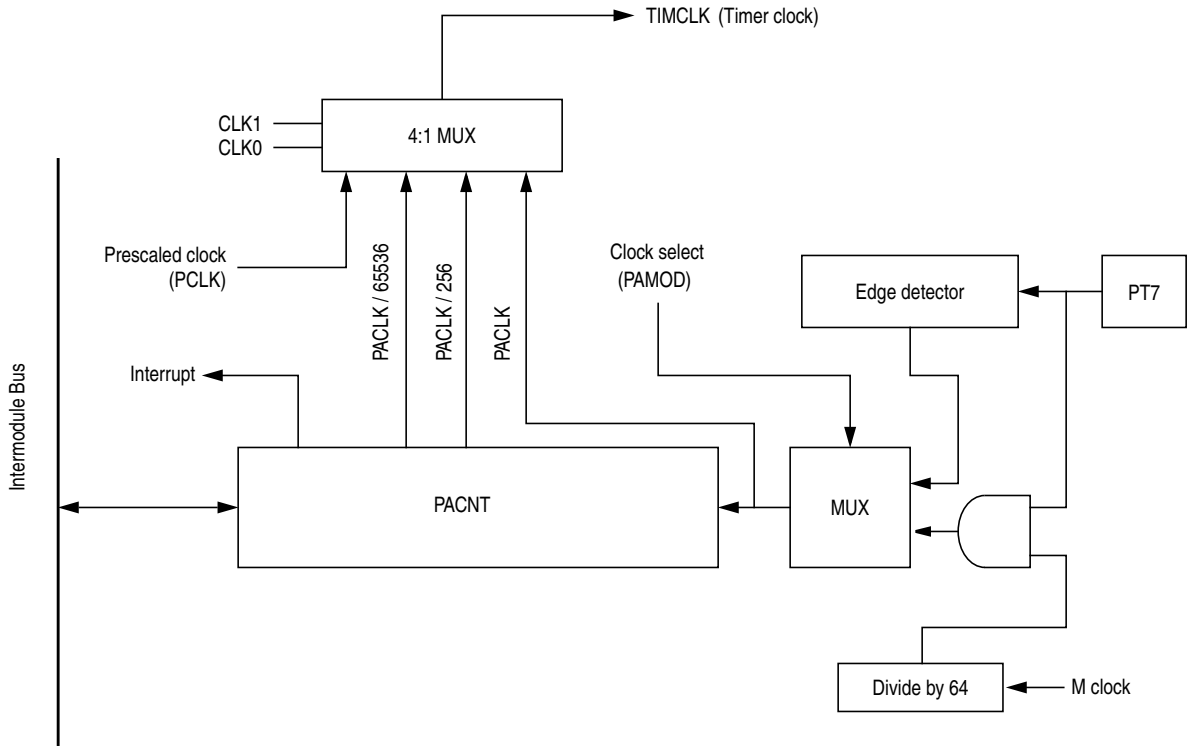


Figure 6-2. 16-Bit Pulse Accumulator Block Diagram

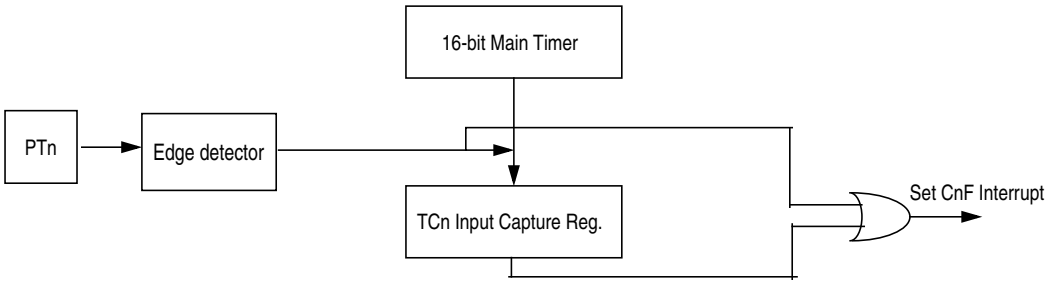


Figure 6-3. Interrupt Flag Setting

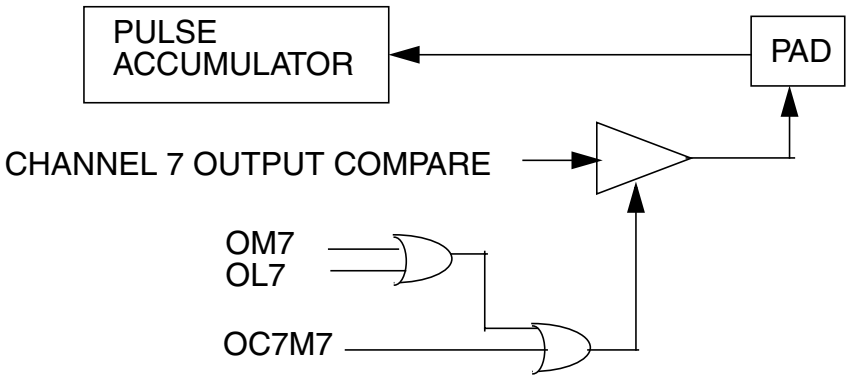


Figure 6-4. Channel 7 Output Compare/Pulse Accumulator Logic

NOTE

For more information see the respective functional descriptions in Section 6.4, “Functional Description,” of this document.

6.2 External Signal Description

The TIM16B4CV1 module has a total of four external pins.

6.2.1 IOC7 — Input Capture and Output Compare Channel 7 Pin

This pin serves as input capture or output compare for channel 7. This can also be configured as pulse accumulator input.

6.2.2 IOC6 — Input Capture and Output Compare Channel 6 Pin

This pin serves as input capture or output compare for channel 6.

6.2.3 IOC5 — Input Capture and Output Compare Channel 5 Pin

This pin serves as input capture or output compare for channel 5.

6.2.4 IOC4 — Input Capture and Output Compare Channel 4 Pin

This pin serves as input capture or output compare for channel 4.

NOTE

For the description of interrupts see Section 6.6, “Interrupts”.

6.3 Memory Map and Register Definition

This section provides a detailed description of all memory and registers.

6.3.1 Module Memory Map

The memory map for the TIM16B4CV1 module is given below in Table 6-1. The address listed for each register is the address offset. The total address for each register is the sum of the base address for the TIM16B4CV1 module and the address offset for each register.

Table 6-1. TIM16B4CV1 Memory Map

Address Offset	Use	Access
0x0000	Timer Input Capture/Output Compare Select (TIOS)	R/W
0x0001	Timer Compare Force Register (CFORC)	R/W[1]
0x0002	Output Compare 7 Mask Register (OC7M)	R/W
0x0003	Output Compare 7 Data Register (OC7D)	R/W
0x0004	Timer Count Register (TCNT(hi))	R/W[2]
0x0005	Timer Count Register (TCNT(lo))	R/W[2]
0x0006	Timer System Control Register1 (TSCR1)	R/W
0x0007	Timer Toggle Overflow Register (TTOV)	R/W
0x0008	Timer Control Register1 (TCTL1)	R/W
0x0009	Reserved	—[3]
0x000A	Timer Control Register3 (TCTL3)	R/W
0x000B	Reserved	—[3]
0x000C	Timer Interrupt Enable Register (TIE)	R/W
0x000D	Timer System Control Register2 (TSCR2)	R/W
0x000E	Main Timer Interrupt Flag1 (TFLG1)	R/W
0x000F	Main Timer Interrupt Flag2 (TFLG2)	R/W
0x0010 - 0x0017	Reserved	—[3]
0x0018	Timer Input Capture/Output Compare Register4 (TC4(hi))	R/W[4]
0x0019	Timer Input Capture/Output Compare Register 4 (TC4(lo))	R/W[4]
0x001A	Timer Input Capture/Output Compare Register 5 (TC5(hi))	R/W[4]
0x001B	Timer Input Capture/Output Compare Register 5 (TC5(lo))	R/W[4]
0x001C	Timer Input Capture/Output Compare Register 6 (TC6(hi))	R/W[4]
0x001D	Timer Input Capture/Output Compare Register 6 (TC6(lo))	R/W[4]
0x001E	Timer Input Capture/Output Compare Register 7 (TC7(hi))	R/W[4]
0x001F	Timer Input Capture/Output Compare Register 7 (TC7(lo))	R/W[4]
0x0020	16-Bit Pulse Accumulator Control Register (PACTL)	R/W
0x0021	Pulse Accumulator Flag Register (PAFLG)	R/W
0x0022	Pulse Accumulator Count Register (PACNT(hi))	R/W
0x0023	Pulse Accumulator Count Register (PACNT(lo))	R/W
0x0024 – 0x002C	Reserved	—[3]
0x002D	Timer Test Register (TIMTST)	R/W[2]
0x002E – 0x002F	Reserved	—[3]

[1] Always read 0x0000.

[2] Only writable in special modes (test_mode = 1).
[3] Write has no effect; return 0 on read
[4] Write to these registers have no meaning or effect during input capture.

6.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 TIOS	R	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
	W								
0x0001 CFORC	R	0	0	0	0	0	0	0	0
	W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
0x0002 OC7M	R	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
	W								
0x0003 OC7D	R	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
	W								
0x0004 TCNTH	R	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
	W								
0x0005 TCNTL	R	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
	W								
0x0006 TSCR2	R	TEN	TSWAI	TSFRZ	TFFCA	0	0	0	0
	W								
0x0007 TTOV	R	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
	W								
0x0008 TCTL1	R	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
	W								
0x0009 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x000A TCTL3	R	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
	W								

= Unimplemented or Reserved

Figure 6-5. TIM16B4CV1 Register Summary

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000B Reserved	R	0	0	0	0	0	0	0	0
	W								
0x000C TIE	R	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
	W								
0x000D TSCR2	R	TOI	0	0	0	TCRE	PR2	PR1	PR0
	W								
0x000E TFLG1	R	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
	W								
0x000F TFLG2	R	TOF	0	0	0	0	0	0	0
	W								
0x0010–0x0017 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0018–0x001F TCxH–TCxL	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0020 PACTL	R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
	W								
0x0021 PAFLG	R	0	0	0	0	0	0	PAOVF	PAIF
	W								
0x0022 PACNTH	R	PACNT15	PACNT14	PACNT13	PACNT12	PACNT11	PACNT10	PACNT9	PACNT8
	W								
0x0023 PACNTL	R	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
	W								
0x0024–0x002F Reserved	R								
	W								

= Unimplemented or Reserved

Figure 6-5. TIM16B4CV1 Register Summary (continued)

6.3.2.1 Timer Input Capture/Output Compare Select (TIOS)

	7	6	5	4	3	2	1	0
R	IOS7	IOS6	IOS5	IOS4	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-6. Timer Input Capture/Output Compare Select (TIOS)

Read: Anytime

Write: Anytime

Table 6-2. TIOS Field Descriptions

Field	Description
7:4 IOS[7:4]	**Input Capture or Output Compare Channel Configuration** 0 The corresponding channel acts as an input capture. 1 The corresponding channel acts as an output compare.

6.3.2.2 Timer Compare Force Register (CFORC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	FOC7	FOC6	FOC5	FOC4				
Reset	0	0	0	0	0	0	0	0

Figure 6-7. Timer Compare Force Register (CFORC)

Read: Anytime but will always return 0x0000 (1 state is transient)

Write: Anytime

Table 6-3. CFORC Field Descriptions

Field	Description
7:4 FOC[7:4]	**Force Output Compare Action for Channel 7:4** — A write to this register with the corresponding data bit(s) set causes the action which is programmed for output compare “x” to occur immediately. The action taken is the same as if a successful comparison had just taken place with the TCx register except the interrupt flag does not get set. **Note:** A successful channel 7 output compare overrides any channel 6:4 compares. If forced output compare on any channel occurs at the same time as the successful output compare then forced output compare action will take precedence and interrupt flag won’t get set.

6.3.2.3 Output Compare 7 Mask Register (OC7M)

	7	6	5	4	3	2	1	0
R	OC7M7	OC7M6	OC7M5	OC7M4	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-8. Output Compare 7 Mask Register (OC7M)

Read: Anytime

Write: Anytime

Table 6-4. OC7M Field Descriptions

Field	Description
7:4 OC7M[7:4]	**Output Compare 7 Mask** — Setting the OC7Mx (x ranges from 4 to 6) will set the corresponding port to be an output port when the corresponding TIOSx (x ranges from 4 to 6) bit is set to be an output compare. **Note:** A successful channel 7 output compare overrides any channel 6:4 compares. For each OC7M bit that is set, the output compare action reflects the corresponding OC7D bit.

6.3.2.4 Output Compare 7 Data Register (OC7D)

	7	6	5	4	3	2	1	0
R	OC7D7	OC7D6	OC7D5	OC7D4	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-9. Output Compare 7 Data Register (OC7D)

Read: Anytime

Write: Anytime

Table 6-5. OC7D Field Descriptions

Field	Description
7:4 OC7D[7:4]	**Output Compare 7 Data** — A channel 7 output compare can cause bits in the output compare 7 data register to transfer to the timer port data register depending on the output compare 7 mask register.

6.3.2.5 Timer Count Register (TCNT)

	15	14	13	12	11	10	9	9
R	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-10. Timer Count Register High (TCNTH)

	7	6	5	4	3	2	1	0
R W	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
Reset	0	0	0	0	0	0	0	0

Figure 6-11. Timer Count Register Low (TCNTL)

The 16-bit main timer is an up counter.

A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

Read: Anytime

Write: Has no meaning or effect in the normal mode; only writable in special modes (test_mode = 1).

The period of the first count after a write to the TCNT registers may be a different size because the write is not synchronized with the prescaler clock.

6.3.2.6 Timer System Control Register 1 (TSCR1)

	7	6	5	4	3	2	1	0
R	TEN	TSWAI	TSFRZ	TFFCA	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 6-12. Timer System Control Register 1 (TSCR2)

Read: Anytime

Write: Anytime

Table 6-6. TSCR1 Field Descriptions

Field	Description
7 TEN	**Timer Enable** 0 Disables the main timer, including the counter. Can be used for reducing power consumption. 1 Allows the timer to function normally. If for any reason the timer is not active, there is no ÷64 clock for the pulse accumulator because the ÷64 is generated by the timer prescaler.
6 TSWAI	**Timer Module Stops While in Wait** 0 Allows the timer module to continue running during wait. 1 Disables the timer module when the MCU is in the wait mode. Timer interrupts cannot be used to get the MCU out of wait. TSWAI also affects pulse accumulator.

Table 6-6. TSCR1 Field Descriptions (continued)

Field	Description
5 TSFRZ	**Timer Stops While in Freeze Mode** 0 Allows the timer counter to continue running while in freeze mode. 1 Disables the timer counter whenever the MCU is in freeze mode. This is useful for emulation. TSFRZ does not stop the pulse accumulator.
4 TFFCA	**Timer Fast Flag Clear All** 0 Allows the timer flag clearing to function normally. 1 For TFLG1(0x000E), a read from an input capture or a write to the output compare channel (0x0010–0x001F) causes the corresponding channel flag, CnF, to be cleared. For TFLG2 (0x000F), any access to the TCNT register (0x0004, 0x0005) clears the TOF flag. Any access to the PACNT registers (0x0022, 0x0023) clears the PAOVF and PAIF flags in the PAFLG register (0x0021). This has the advantage of eliminating software overhead in a separate clear sequence. Extra care is required to avoid accidental flag clearing due to unintended accesses.

6.3.2.7 Timer Toggle On Overflow Register 1 (TTOV)

	7	6	5	4	3	2	1	0
R	TOV7	TOV6	TOV5	TOV4	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-13. Timer Toggle On Overflow Register 1 (TTOV)

Read: Anytime

Write: Anytime

Table 6-7. TTOV Field Descriptions

Field	Description
7:4 TOV[7:4]	**Toggle On Overflow Bits** — TOVx toggles output compare pin on overflow. This feature only takes effect when in output compare mode. When set, it takes precedence over forced output compare but not channel 7 override events. 0 Toggle output compare pin on overflow feature disabled. 1 Toggle output compare pin on overflow feature enabled.

6.3.2.8 Timer Control Register 1 (TCTL1)

	7	6	5	4	3	2	1	0
R	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-14. Timer Control Register 1 (TCTL1)

Read: Anytime

Write: Anytime

Table 6-8. TCTL1/TCTL2 Field Descriptions

Field	Description
7:4 OMx	**Output Mode** — These four pairs of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. When either OMx or OLx is 1, the pin associated with OCx becomes an output tied to OCx. **Note:** To enable output action by OMx bits on timer port, the corresponding bit in OC7M should be cleared.
7:4 OLx	**Output Level** — These four pairs of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. When either OMx or OLx is 1, the pin associated with OCx becomes an output tied to OCx. **Note:** To enable output action by OLx bits on timer port, the corresponding bit in OC7M should be cleared.

Table 6-9. Compare Result Output Action

OMx	OLx	Action
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

To operate the 16-bit pulse accumulator independently of input capture or output compare 7 and 4 respectively the user must set the corresponding bits IOSx = 1, OMx = 0 and OLx = 0. OC7M7 in the OC7M register must also be cleared.

6.3.2.9 Timer Control Register 3 (TCTL3)

	7	6	5	4	3	2	1	0
R W	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
Reset	0	0	0	0	0	0	0	0

Figure 6-15. Timer Control Register 3 (TCTL3)

Read: Anytime

Write: Anytime.

Table 6-10. TCTL3/TCTL4 Field Descriptions

Field	Description
7:0 EDGnB EDGnA	**Input Capture Edge Control** — These eight pairs of control bits configure the input capture edge detector circuits.

Table 6-11. Edge Detector Circuit Configuration

EDGnB	EDGnA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge (rising or falling)

6.3.2.10 Timer Interrupt Enable Register (TIE)

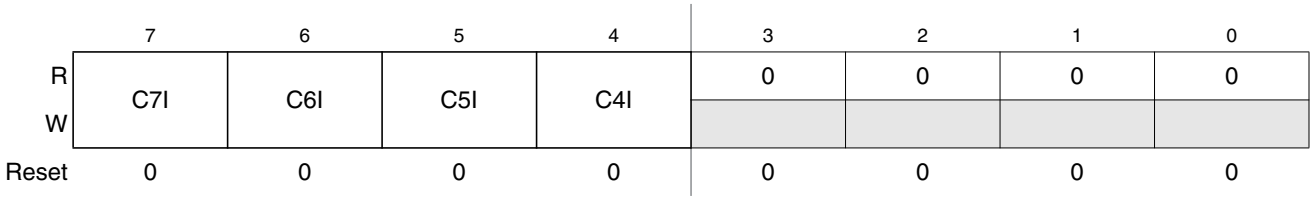


Figure 6-16. Timer Interrupt Enable Register (TIE)

Read: Anytime

Write: Anytime.

Table 6-12. TIE Field Descriptions

Field	Description
7:4 C7I:C0I	**Input Capture/Output Compare "x" Interrupt Enable —** The bits in TIE correspond bit-for-bit with the bits in the TFLG1 status register. If cleared, the corresponding flag is disabled from causing a hardware interrupt. If set, the corresponding flag is enabled to cause a interrupt.

6.3.2.11 Timer System Control Register 2 (TSCR2)

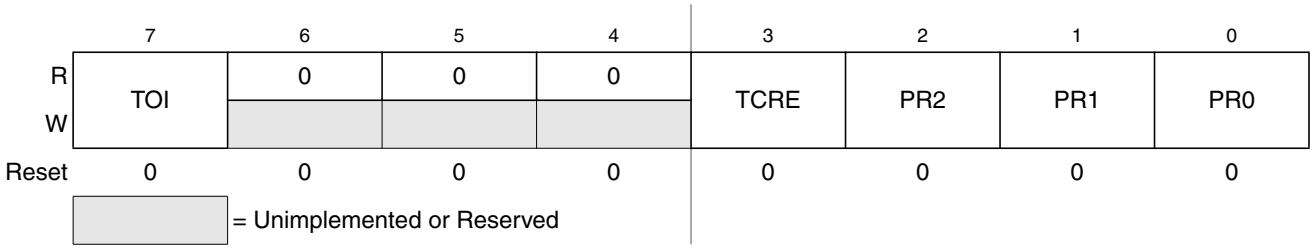


Figure 6-17. Timer System Control Register 2 (TSCR2)

Read: Anytime

Write: Anytime.

Table 6-13. TSCR2 Field Descriptions

Field	Description
7 TOI	**Timer Overflow Interrupt Enable** 0 Interrupt inhibited. 1 Hardware interrupt requested when TOF flag set.
3 TCRE	**Timer Counter Reset Enable** — This bit allows the timer counter to be reset by a successful output compare 7 event. This mode of operation is similar to an up-counting modulus counter. 0 Counter reset inhibited and counter free runs. 1 Counter reset by a successful output compare 7. If TC7 = 0x0000 and TCRE = 1, TCNT will stay at 0x0000 continuously. If TC7 = 0xFFFF and TCRE = 1, TOF will never be set when TCNT is reset from 0xFFFF to 0x0000.
2 PR[2:0]	**Timer Prescaler Select** — These three bits select the frequency of the timer prescaler clock derived from the Bus Clock as shown in Table 6-14.

Table 6-14. Timer Clock Selection

PR2	PR1	PR0	Timer Clock
0	0	0	Bus Clock / 1
0	0	1	Bus Clock / 2
0	1	0	Bus Clock / 4
0	1	1	Bus Clock / 8
1	0	0	Bus Clock / 16
1	0	1	Bus Clock / 32
1	1	0	Bus Clock / 64
1	1	1	Bus Clock / 128

NOTE

The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero.

6.3.2.12 Main Timer Interrupt Flag 1 (TFLG1)

	7	6	5	4	3	2	1	0
R	C7F	C6F	C5F	C4F	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-18. Main Timer Interrupt Flag 1 (TFLG1)

Read: Anytime

Write: Used in the clearing mechanism (set bits cause corresponding bits to be cleared). Writing a zero will not affect current status of the bit.

Table 6-15. TRLG1 Field Descriptions

Field	Description
7:4 C[7:4]F	**Input Capture/Output Compare Channel "x" Flag** — These flags are set when an input capture or output compare event occurs. Clear a channel flag by writing one to it. When TFFCA bit in TSCR register is set, a read from an input capture or a write into an output compare channel (0x0010–0x001F) will cause the corresponding channel flag CxF to be cleared.

6.3.2.13 Main Timer Interrupt Flag 2 (TFLG2)

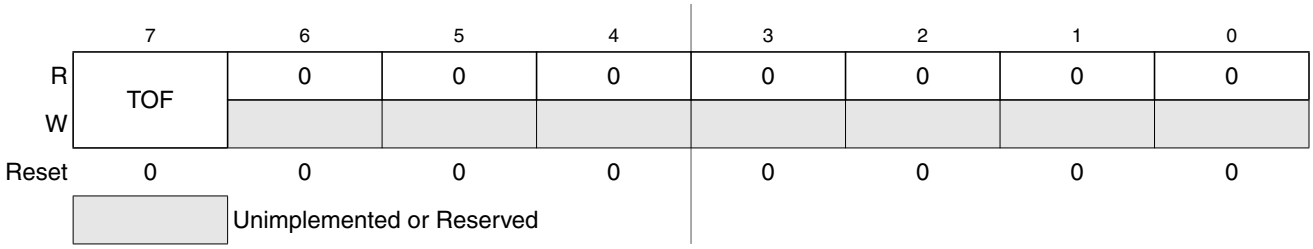


Figure 6-19. Main Timer Interrupt Flag 2 (TFLG2)

TFLG2 indicates when interrupt conditions have occurred. To clear a bit in the flag register, write the bit to one.

Read: Anytime

Write: Used in clearing mechanism (set bits cause corresponding bits to be cleared).

Any access to TCNT will clear TFLG2 register if the TFFCA bit in TSCR register is set.

Table 6-16. TRLG2 Field Descriptions

Field	Description
7 TOF	**Timer Overflow Flag** — Set when 16-bit free-running timer overflows from 0xFFFF to 0x0000. This bit is cleared automatically by a write to the TFLG2 register with bit 7 set. (See also TCRE control bit explanation.)

6.3.2.14 Timer Input Capture/Output Compare Registers High and Low 4–7 (TCxH and TCxL)

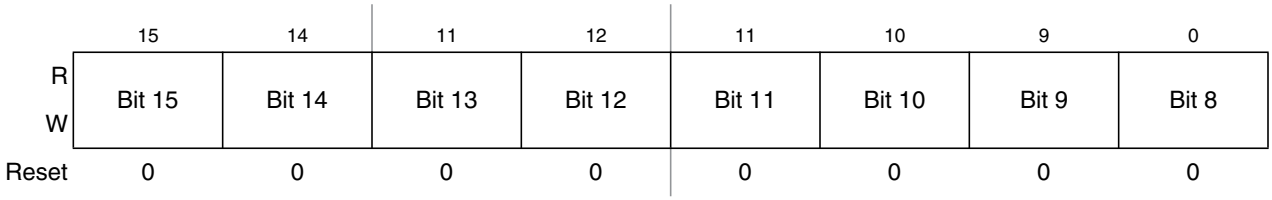


Figure 6-20. Timer Input Capture/Output Compare Register x High (TCxH)

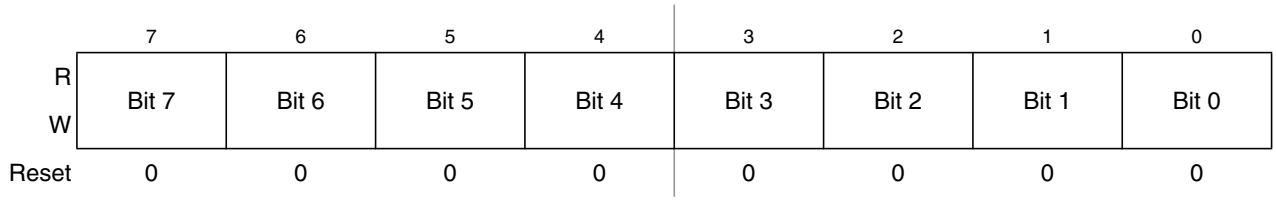


Figure 6-21. Timer Input Capture/Output Compare Register x Low (TCxL)

Depending on the TIOS bit for the corresponding channel, these registers are used to latch the value of the free-running counter when a defined transition is sensed by the corresponding input capture edge detector or to trigger an output action for output compare.

Read: Anytime

Write: Anytime for output compare function.Writes to these registers have no meaning or effect during input capture. All timer input capture/output compare registers are reset to 0x0000.

NOTE

Read/Write access in byte mode for high byte should takes place before low byte otherwise it will give a different result.

6.3.2.15 16-Bit Pulse Accumulator Control Register (PACTL)

	7	6	5	4	3	2	1	0
R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
W								
Reset	0	0	0	0	0	0	0	0

Unimplemented or Reserved

Figure 6-22. 16-Bit Pulse Accumulator Control Register (PACTL)

When PAEN is set, the PACT is enabled.The PACT shares the input pin with IOC7.

Read: Any time

Write: Any time

Table 6-17. PACTL Field Descriptions

Field	Description
6 PAEN	**Pulse Accumulator System Enable** — PAEN is independent from TEN. With timer disabled, the pulse accumulator can function unless pulse accumulator is disabled. 0 16-Bit Pulse Accumulator system disabled. 1 Pulse Accumulator system enabled.
5 PAMOD	**Pulse Accumulator Mode** — This bit is active only when the Pulse Accumulator is enabled (PAEN = 1). See Table 6-18. 0 Event counter mode. 1 Gated time accumulation mode.
4 PEDGE	**Pulse Accumulator Edge Control** — This bit is active only when the Pulse Accumulator is enabled (PAEN = 1). For PAMOD bit = 0 (event counter mode). See Table 6-18. 0 Falling edges on IOC7 pin cause the count to be incremented. 1 Rising edges on IOC7 pin cause the count to be incremented. For PAMOD bit = 1 (gated time accumulation mode). 0 IOC7 input pin high enables M (bus clock) divided by 64 clock to Pulse Accumulator and the trailing falling edge on IOC7 sets the PAIF flag. 1 IOC7 input pin low enables M (bus clock) divided by 64 clock to Pulse Accumulator and the trailing rising edge on IOC7 sets the PAIF flag.
3:2 CLK[1:0]	**Clock Select Bits —** Refer to Table 6-19.

Table 6-17. PACTL Field Descriptions (continued)

Field	Description
1 PAOVI	**Pulse Accumulator Overflow Interrupt Enable** 0 Interrupt inhibited. 1 Interrupt requested if PAOVF is set.
0 PAI	**Pulse Accumulator Input Interrupt Enable** 0 Interrupt inhibited. 1 Interrupt requested if PAIF is set.

Table 6-18. Pin Action

PAMOD	PEDGE	Pin Action
0	0	Falling edge
0	1	Rising edge
1	0	Div. by 64 clock enabled with pin high level
1	1	Div. by 64 clock enabled with pin low level

NOTE

If the timer is not active (TEN = 0 in TSCR), there is no divide-by-64 because the ÷64 clock is generated by the timer prescaler.

Table 6-19. Timer Clock Selection

CLK1	CLK0	Timer Clock
0	0	Use timer prescaler clock as timer counter clock
0	1	Use PACLK as input to timer counter clock
1	0	Use PACLK/256 as timer counter clock frequency
1	1	Use PACLK/65536 as timer counter clock frequency

For the description of PACLK please refer Figure 6-22**.**

If the pulse accumulator is disabled (PAEN = 0), the prescaler clock from the timer is always used as an input clock to the timer counter. The change from one selected clock to the other happens immediately after these bits are written.

6.3.2.16 Pulse Accumulator Flag Register (PAFLG)

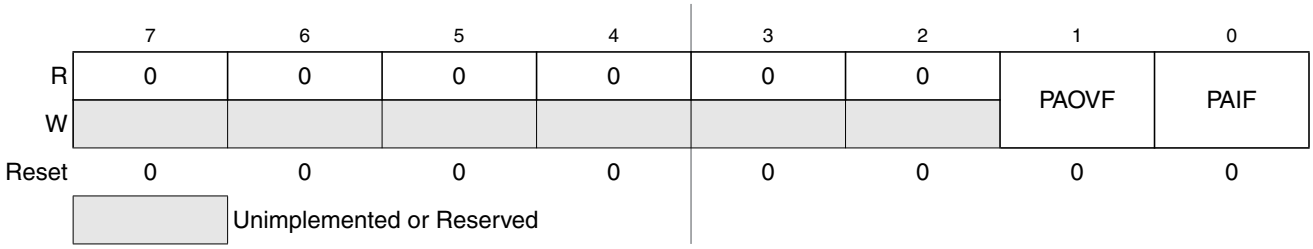


Figure 6-23. Pulse Accumulator Flag Register (PAFLG)

Read: Anytime

Write: Anytime

When the TFFCA bit in the TSCR register is set, any access to the PACNT register will clear all the flags in the PAFLG register.

Table 6-20. PAFLG Field Descriptions

Field	Description
1 PAOVF	**Pulse Accumulator Overflow Flag** — Set when the 16-bit pulse accumulator overflows from 0xFFFF to 0x0000. This bit is cleared automatically by a write to the PAFLG register with bit 1 set.
0 PAIF	**Pulse Accumulator Input edge Flag** — Set when the selected edge is detected at the IOC7 input pin.In event mode the event edge triggers PAIF and in gated time accumulation mode the trailing edge of the gate signal at the IOC7 input pin triggers PAIF. This bit is cleared by a write to the PAFLG register with bit 0 set. Any access to the PACNT register will clear all the flags in this register when TFFCA bit in register TSCR(0x0006) is set.

6.3.2.17 Pulse Accumulators Count Registers (PACNT)

	15	14	13	12	11	10	9	0
R W	PACNT15	PACNT14	PACNT13	PACNT12	PACNT11	PACNT10	PACNT9	PACNT8
Reset	0	0	0	0	0	0	0	0

Figure 6-24. Pulse Accumulator Count Register High (PACNTH)

	7	6	5	4	3	2	1	0
R W	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
Reset	0	0	0	0	0	0	0	0

Figure 6-25. Pulse Accumulator Count Register Low (PACNTL)

Read: Anytime

Write: Anytime

These registers contain the number of active input edges on its input pin since the last reset.

When PACNT overflows from 0xFFFF to 0x0000, the Interrupt flag PAOVF in PAFLG (0x0021) is set.

Full count register access should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

NOTE

Reading the pulse accumulator counter registers immediately after an active edge on the pulse accumulator input pin may miss the last count because the input has to be synchronized with the bus clock first.

6.4 Functional Description

This section provides a complete functional description of the timer TIM16B4CV1 block. Please refer to the detailed timer block diagram in Figure 6-26 as necessary.

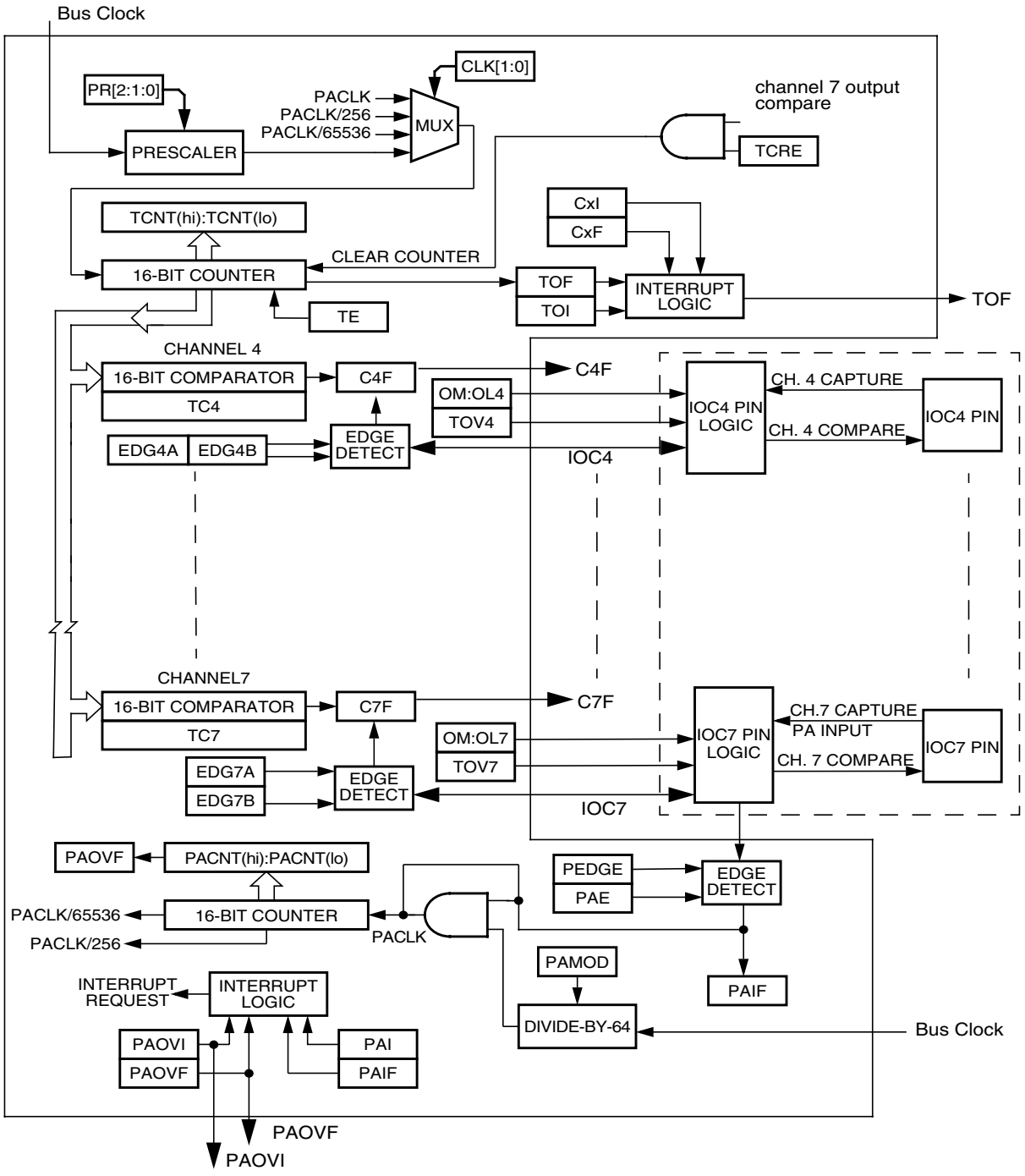


Figure 6-26. Detailed Timer Block Diagram

6.4.1 Prescaler

The prescaler divides the bus clock by 1,2,4,8,16,32,64 or 128. The prescaler select bits, PR[2:0], select the prescaler divisor. PR[2:0] are in timer system control register 2 (TSCR2).

6.4.2 Input Capture

Clearing the I/O (input/output) select bit, IOSx, configures channel x as an input capture channel. The input capture function captures the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the timer transfers the value in the timer counter into the timer channel registers, TCx.

The minimum pulse width for the input capture input is greater than two bus clocks.

An input capture on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests.

6.4.3 Output Compare

Setting the I/O select bit, IOSx, configures channel x as an output compare channel. The output compare function can generate a periodic pulse with a programmable polarity, duration, and frequency. When the timer counter reaches the value in the channel registers of an output compare channel, the timer can set, clear, or toggle the channel pin. An output compare on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests.

The output mode and level bits, OMx and OLx, select set, clear, toggle on output compare. Clearing both OMx and OLx disconnects the pin from the output logic.

Setting a force output compare bit, FOCx, causes an output compare on channel x. A forced output compare does not set the channel flag.

A successful output compare on channel 7 overrides output compares on all other output compare channels. The output compare 7 mask register masks the bits in the output compare 7 data register. The timer counter reset enable bit, TCRE, enables channel 7 output compares to reset the timer counter. A channel 7 output compare can reset the timer counter even if the IOC7 pin is being used as the pulse accumulator input.

Writing to the timer port bit of an output compare pin does not affect the pin state. The value written is stored in an internal latch. When the pin becomes available for general-purpose output, the last value written to the bit appears at the pin.

6.4.4 Pulse Accumulator

The pulse accumulator (PACNT) is a 16-bit counter that can operate in two modes:

Event counter mode — Counting edges of selected polarity on the pulse accumulator input pin, PAI.

Gated time accumulation mode — Counting pulses from a divide-by-64 clock. The PAMOD bit selects the mode of operation.

The minimum pulse width for the PAI input is greater than two bus clocks.

6.4.5 Event Counter Mode

Clearing the PAMOD bit configures the PACNT for event counter operation. An active edge on the IOC7 pin increments the pulse accumulator counter. The PEDGE bit selects falling edges or rising edges to increment the count.

NOTE

The PACNT input and timer channel 7 use the same pin IOC7. To use the IOC7, disconnect it from the output logic by clearing the channel 7 output mode and output level bits, OM7 and OL7. Also clear the channel 7 output compare 7 mask bit, OC7M7.

The Pulse Accumulator counter register reflect the number of active input edges on the PACNT input pin since the last reset.

The PAOVF bit is set when the accumulator rolls over from 0xFFFF to 0x0000. The pulse accumulator overflow interrupt enable bit, PAOVI, enables the PAOVF flag to generate interrupt requests.

NOTE

The pulse accumulator counter can operate in event counter mode even when the timer enable bit, TEN, is clear.

6.4.6 Gated Time Accumulation Mode

Setting the PAMOD bit configures the pulse accumulator for gated time accumulation operation. An active level on the PACNT input pin enables a divided-by-64 clock to drive the pulse accumulator. The PEDGE bit selects low levels or high levels to enable the divided-by-64 clock.

The trailing edge of the active level at the IOC7 pin sets the PAIF. The PAI bit enables the PAIF flag to generate interrupt requests.

The pulse accumulator counter register reflect the number of pulses from the divided-by-64 clock since the last reset.

NOTE

The timer prescaler generates the divided-by-64 clock. If the timer is not active, there is no divided-by-64 clock.

6.5 Resets

The reset state of each individual bit is listed within Section 6.3, “Memory Map and Register Definition” which details the registers and their bit fields.

6.6 Interrupts

This section describes interrupts originated by the TIM16B4CV1 block. Table 6-21 lists the interrupts generated by the TIM16B4CV1 to communicate with the MCU.

Table 6-21. TIM16B8CV1 Interrupts

Interrupt	Offset[1]	Vector[1]	Priority[1]	Source	Description
C[7:4]F	—	—	—	Timer Channel 7–4	Active high timer channel interrupts 7–4
PAOVI	—	—	—	Pulse Accumulator Input	Active high pulse accumulator input interrupt
PAOVF	—	—	—	Pulse Accumulator Overflow	Pulse accumulator overflow interrupt
TOF	—	—	—	Timer Overflow	Timer Overflow interrupt

1 Chip Dependent.

The TIM16B4CV1 uses a total of 7 interrupt vectors. The interrupt vector offsets and interrupt numbers are chip dependent.

6.6.1 Channel [7:4] Interrupt (C[7:4]F)

This active high outputs will be asserted by the module to request a timer channel 7 – 4 interrupt to be serviced by the system controller.

6.6.2 Pulse Accumulator Input Interrupt (PAOVI)

This active high output will be asserted by the module to request a timer pulse accumulator input interrupt to be serviced by the system controller.

6.6.3 Pulse Accumulator Overflow Interrupt (PAOVF)

This active high output will be asserted by the module to request a timer pulse accumulator overflow interrupt to be serviced by the system controller.

6.6.4 Timer Overflow Interrupt (TOF)

This active high output will be asserted by the module to request a timer overflow interrupt to be serviced by the system controller.

Chapter 7
Analog-to-Digital Converter (ATD10B8CV3)

7.1 Introduction

The ATD10B8C is an 8-channel, 10-bit, multiplexed input successive approximation analog-to-digital converter. Refer to device electrical specifications for ATD accuracy.

7.1.1 Features

- 8/10-bit resolution
- 7 µsec, 10-bit single conversion time
- Sample buffer amplifier
- Programmable sample time
- Left/right justified, signed/unsigned result data
- External trigger control
- Conversion completion interrupt generation
- Analog input multiplexer for 8 analog input channels
- Analog/digital input pin multiplexing
- 1-to-8 conversion sequence lengths
- Continuous conversion mode
- Multiple channel scans
- Configurable external trigger functionality on any AD channel or any of four additional external trigger inputs. The four additional trigger inputs can be chip external or internal. Refer to the device overview chapter for availability and connectivity.
- Configurable location for channel wrap around (when converting multiple channels in a sequence).

7.1.2 Modes of Operation

7.1.2.1 Conversion Modes

There is software programmable selection between performing single or continuous conversion on a single channel or multiple channels.

7.1.2.2 MCU Operating Modes

- Stop mode

  Entering stop mode causes all clocks to halt and thus the system is placed in a minimum power standby mode. This aborts any conversion sequence in progress. During recovery from stop mode, there must be a minimum delay for the stop recovery time t_{SR} before initiating a new ATD conversion sequence.
- Wait mode

  Entering wait mode the ATD conversion either continues or aborts for low power depending on the logical value of the AWAIT bit.
- Freeze mode

  In freeze mode the ATD will behave according to the logical values of the FRZ1 and FRZ0 bits. This is useful for debugging and emulation.

7.1.3 Block Diagram

Figure 7-1 shows a block diagram of the ATD.

7.2 External Signal Description

This section lists all inputs to the ATD block.

7.2.1 AN*x* (*x* = 7, 6, 5, 4, 3, 2, 1, 0) — Analog Input Pin

This pin serves as the analog input channel *x*. It can also be configured as general purpose digital port pin and/or external trigger for the ATD conversion.

7.2.2 ETRIG3, ETRIG2, ETRIG1, and ETRIG0 — External Trigger Pins

These inputs can be configured to serve as an external trigger for the ATD conversion.

Refer to the device overview chapter for availability and connectivity of these inputs.

7.2.3 V_{RH} and V_{RL} — High and Low Reference Voltage Pins

V_{RH} is the high reference voltage and V_{RL} is the low reference voltage for ATD conversion.

7.2.4 V_{DDA} and V_{SSA} — Power Supply Pins

These pins are the power supplies for the analog circuitry of the ATD block.

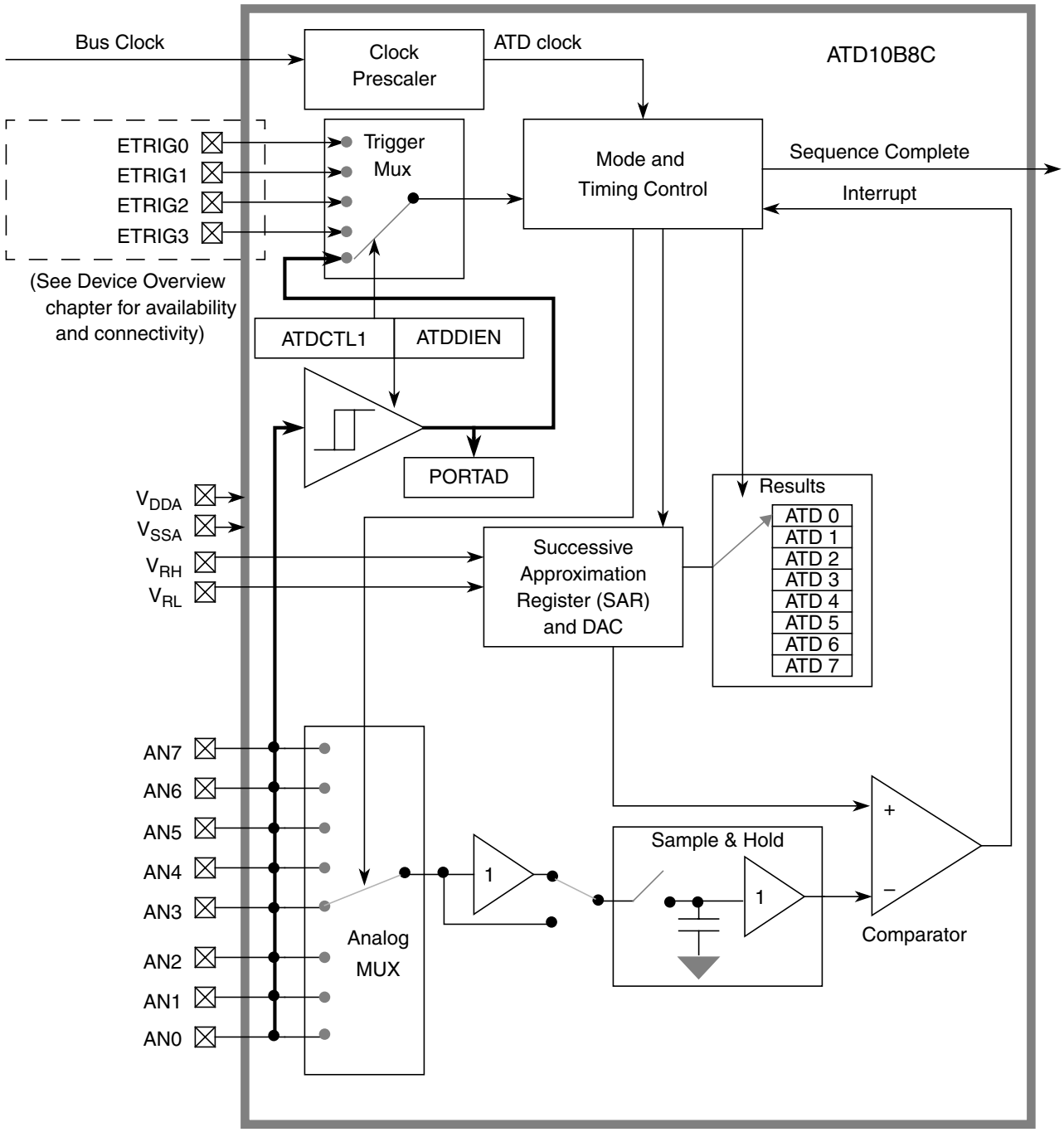


Figure 7-1. ATD Block Diagram

7.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the ATD.

7.3.1 Module Memory Map

Figure 7-2 gives an overview of all ATD registers.

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

7.3.2 Register Descriptions

This section describes in address order all the ATD registers and their individual bits.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
ATDCTL0	R	0	0	0	0	0	WRAP2	WRAP1	WRAP0
	W								
ATDCTL1	R	ETRIGSEL	0	0	0	0	ETRIGCH2	ETRIGCH1	ETRIGCH0
	W								
ATDCTL2	R	ADPU	AFFC	AWAI	ETRIGLE	ETRIGP	ETRIGE	ASCIE	ASCIF
	W								
ATDCTL3	R	0	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
	W								
ATDCTL4	R	SRES8	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
	W								
ATDCTL5	R	DJM	DSGN	SCAN	MULT	0	CC	CB	CA
	W								
ATDSTAT0	R	SCF	0	ETORF	FIFOR	0	CC2	CC1	CC0
	W								
Unimplemented	R								
	W								
ATDTEST0	R	U	U	U	U	U	U	U	U
	W								
ATDTEST1	R	U	U	0	0	0	0	0	SC
	W								

= Unimplemented or Reserved

Figure 7-2. ATD Register Summary (Sheet 1 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
Unimplemented	R								
	W								
ATDSTAT1	R	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
	W								
Unimplemented	R								
	W								
ATDDIEN	R	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0
	W								
Unimplemented	R								
	W								
PORTAD	R	PTAD7	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
	W								

Left Justified Result Data

Note: The read portion of the left justified result data registers has been divided to show the bit position when reading 10-bit and 8-bit conversion data. For more detailed information refer to Section 7.3.2.13, “ATD Conversion Result Registers (ATDDRx)”.

ATDDR0H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR0L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								
ATDDR1H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR1L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								
ATDDR2H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR2L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								
ATDDR3H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								

= Unimplemented or Reserved

Figure 7-2. ATD Register Summary (Sheet 2 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
ATDDR3L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								
ATDDR4H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR4L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								
ATDD45H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDD45L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								
ATDD46H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR6L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								
ATDD47H	10-BIT	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDD47L	10-BIT	BIT 1	BIT 0	0	0	0	0	0	0
	8-BIT	U	U	0	0	0	0	0	0
	W								

Right Justified Result Data

Note: The read portion of the right justified result data registers has been divided to show the bit position when reading 10-bit and 8-bit conversion data. For more detailed information refer to Section 7.3.2.13, “ATD Conversion Result Registers (ATDDRx)”.

ATDDR0H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDDR0L	10-BIT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								

= Unimplemented or Reserved

Figure 7-2. ATD Register Summary (Sheet 3 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
ATDDR1H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDDR1L	10-BIT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR2H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDDR2L	10-BIT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR3H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDDR3L	10-BIT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDDR4H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDDR4L	10-BIT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDD45H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDD45L	10-BIT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								
ATDD46H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDDR6L	10-BIT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	8-BIT	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	W								

= Unimplemented or Reserved

Figure 7-2. ATD Register Summary (Sheet 4 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
ATDD47H	10-BIT	0	0	0	0	0	0	BIT 9 MSB	BIT 8
	8-BIT	0	0	0	0	0	0	0	0
	W								
ATDD47L	10-BIT	BIT 7 BIT 7 MSB	BIT 6 BIT 6	BIT 5 BIT 5	BIT 4 BIT 4	BIT 3 BIT 3	BIT 2 BIT 2	BIT 1 BIT 1	BIT 0 BIT 0
	8-BIT								

= Unimplemented or Reserved

Figure 7-2. ATD Register Summary (Sheet 5 of 5)

7.3.2.1 ATD Control Register 0 (ATDCTL0)

Writes to this register will abort current conversion sequence but will not start a new sequence.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	WRAP2	WRAP1	WRAP0
W								
Reset	0	0	0	0	0	1	1	1

= Unimplemented or Reserved

Figure 7-3. ATD Control Register 0 (ATDCTL0)

Read: Anytime

Write: Anytime

Table 7-1. ATDCTL0 Field Descriptions

Field	Description
2–0 WRAP[2:0]	**Wrap Around Channel Select Bits** — These bits determine the channel for wrap around when doing multi-channel conversions. The coding is summarized in Table 7-2.

Table 7-2. Multi-Channel Wrap Around Coding

WRAP2	WRAP1	WRAP0	Multiple Channel Conversions (MULT = 1) Wrap Around to AN0 after Converting
0	0	0	Reserved
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3
1	0	0	AN4
1	0	1	AN5
1	1	0	AN6
1	1	1	AN7

7.3.2.2 ATD Control Register 1 (ATDCTL1)

Writes to this register will abort current conversion sequence but will not start a new sequence.

	7	6	5	4	3	2	1	0
R	ETRIGSEL	0	0	0	0	ETRIGCH2	ETRIGCH1	ETRIGCH0
W								
Reset	0	0	0	0	0	1	1	1

= Unimplemented or Reserved

Figure 7-4. ATD Control Register 1 (ATDCTL1)

Read: Anytime

Write: Anytime

Table 7-3. ATDCTL1 Field Descriptions

Field	Description
7 ETRIGSEL	**External Trigger Source Select** — This bit selects the external trigger source to be either one of the AD channels or one of the ETRIG3–0 inputs. See the device overview chapter for availability and connectivity of ETRIG3–0 inputs. If ETRIG3–0 input option is not available, writing a 1 to ETRISEL only sets the bit but has not effect, that means still one of the AD channels (selected by ETRIGCH2–0) is the source for external trigger. The coding is summarized in Table 7-4.
2–0 ETRIGCH[2:0]	**External Trigger Channel Select** — These bits select one of the AD channels or one of the ETRIG3–0 inputs as source for the external trigger. The coding is summarized in Table 7-4.

Table 7-4. External Trigger Channel Select Coding

ETRIGSEL	ETRIGCH2	ETRIGCH1	ETRIGCH0	External trigger source is
0	0	0	0	AN0
0	0	0	1	AN1
0	0	1	0	AN2
0	0	1	1	AN3
0	1	0	0	AN4
0	1	0	1	AN5
0	1	1	0	AN6
0	1	1	1	AN7
1	0	0	0	ETRIG0[1]
1	0	0	1	ETRIG1[1]
1	0	1	0	ETRIG2[1]
1	0	1	1	ETRIG3[1]
1	1	X	X	Reserved

1 Only if ETRIG3–0 input option is available (see device overview chapter), else ETRISEL is ignored, that means external trigger source is still on one of the AD channels selected by ETRIGCH2–0

7.3.2.3 ATD Control Register 2 (ATDCTL2)

This register controls power down, interrupt and external trigger. Writes to this register will abort current conversion sequence but will not start a new sequence.

	7	6	5	4	3	2	1	0
R	ADPU	AFFC	AWAI	ETRIGLE	ETRIGP	ETRIGE	ASCIE	ASCIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-5. ATD Control Register 2 (ATDCTL2)

Read: Anytime

Write: Anytime

Table 7-5. ATDCTL2 Field Descriptions

Field	Description
7 ADPU	**ATD Power Up** — This bit provides on/off control over the ATD block allowing reduced MCU power consumption. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after ADPU bit is enabled. 0 Power down ATD 1 Normal ATD functionality
6 AFFC	**ATD Fast Flag Clear All** 0 ATD flag clearing operates normally (read the status register ATDSTAT1 before reading the result register to clear the associate CCF flag). 1 Changes all ATD conversion complete flags to a fast clear sequence. Any access to a result register will cause the associate CCF flag to clear automatically.
5 AWAI	**ATD Power Down in Wait Mode** — When entering wait mode this bit provides on/off control over the ATD block allowing reduced MCU power. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after exit from Wait mode. 0 ATD continues to run in Wait mode 1 Halt conversion and power down ATD during wait mode After exiting wait mode with an interrupt conversion will resume. But due to the recovery time the result of this conversion should be ignored.
4 ETRIGLE	**External Trigger Level/Edge Control** — This bit controls the sensitivity of the external trigger signal. See Table 7-6 for details.
3 ETRIGP	**External Trigger Polarity** — This bit controls the polarity of the external trigger signal. See Table 7-6 for details.
2 ETRIGE	**External Trigger Mode Enable** — This bit enables the external trigger on one of the AD channels or one of the ETRIG3–0 inputs as described in Table 7-4. If external trigger source is one of the AD channels, the digital input buffer of this channel is enabled. The external trigger allows to synchronize sample and ATD conversions processes with external events. 0 Disable external trigger 1 Enable external trigger **Note:** If using one of the AD channel as external trigger (ETRIGSEL = 0) the conversion results for this channel have no meaning while external trigger mode is enabled.

Table 7-5. ATDCTL2 Field Descriptions (continued)

Field	Description
1 ASCIE	**ATD Sequence Complete Interrupt Enable** 0 ATD Sequence Complete interrupt requests are disabled. 1 ATD Interrupt will be requested whenever ASCIF = 1 is set.
0 ASCIF	**ATD Sequence Complete Interrupt Flag** — If ASCIE = 1 the ASCIF flag equals the SCF flag (see Section 7.3.2.7, "ATD Status Register 0 (ATDSTAT0)"), else ASCIF reads zero. Writes have no effect. 0 No ATD interrupt occurred 1 ATD sequence complete interrupt pending

Table 7-6. External Trigger Configurations

ETRIGLE	ETRIGP	External Trigger Sensitivity
0	0	Falling edge
0	1	Rising edge
1	0	Low level
1	1	High level

7.3.2.4 ATD Control Register 3 (ATDCTL3)

This register controls the conversion sequence length, FIFO for results registers and behavior in freeze mode. Writes to this register will abort current conversion sequence but will not start a new sequence.

	7	6	5	4	3	2	1	0
R	0	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-6. ATD Control Register 3 (ATDCTL3)

Read: Anytime

Write: Anytime

Table 7-7. ATDCTL3 Field Descriptions

Field	Description
6–3 S8C, S4C, S2C, S1C	**Conversion Sequence Length** — These bits control the number of conversions per sequence. Table 7-8 shows all combinations. At reset, S4C is set to 1 (sequence length is 4). This is to maintain software continuity to HC12 Family.

Table 7-7. ATDCTL3 Field Descriptions (continued)

Field	Description
2 FIFO	**Result Register FIFO Mode** — If this bit is zero (non-FIFO mode), the A/D conversion results map into the result registers based on the conversion sequence; the result of the first conversion appears in the first result register, the second result in the second result register, and so on. If this bit is one (FIFO mode) the conversion counter is not reset at the beginning or ending of a conversion sequence; sequential conversion results are placed in consecutive result registers. In a continuously scanning conversion sequence, the result register counter will wrap around when it reaches the end of the result register file. The conversion counter value (CC2-0 in ATDSTAT0) can be used to determine where in the result register file, the current conversion result will be placed. Aborting a conversion or starting a new conversion by write to an ATDCTL register (ATDCTL5-0) clears the conversion counter even if FIFO=1. So the first result of a new conversion sequence, started by writing to ATDCTL5, will always be place in the first result register (ATDDDR0). Intended usage of FIFO mode is continuos conversion (SCAN=1) or triggered conversion (ETRIG=1). Finally, which result registers hold valid data can be tracked using the conversion complete flags. Fast flag clear mode may or may not be useful in a particular application to track valid data. 0 Conversion results are placed in the corresponding result register up to the selected sequence length. 1 Conversion results are placed in consecutive result registers (wrap around at end).
1–0 FRZ[1:0]	**Background Debug Freeze Enable** — When debugging an application, it is useful in many cases to have the ATD pause when a breakpoint (Freeze Mode) is encountered. These 2 bits determine how the ATD will respond to a breakpoint as shown in Table 7-9. Leakage onto the storage node and comparator reference capacitors may compromise the accuracy of an immediately frozen conversion depending on the length of the freeze period.

Table 7-8. Conversion Sequence Length Coding

S8C	S4C	S2C	S1C	Number of Conversions per Sequence
0	0	0	0	8
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	X	X	X	8

Table 7-9. ATD Behavior in Freeze Mode (Breakpoint)

FRZ1	FRZ0	Behavior in Freeze Mode
0	0	Continue conversion
0	1	Reserved
1	0	Finish current conversion, then freeze
1	1	Freeze Immediately

7.3.2.5 ATD Control Register 4 (ATDCTL4)

This register selects the conversion clock frequency, the length of the second phase of the sample time and the resolution of the A/D conversion (i.e.: 8-bits or 10-bits). Writes to this register will abort current conversion sequence but will not start a new sequence.

	7	6	5	4	3	2	1	0
R W	SRES8	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
Reset	0	0	0	0	0	1	0	1

Figure 7-7. ATD Control Register 4 (ATDCTL4)

Read: Anytime

Write: Anytime

Table 7-10. ATDCTL4 Field Descriptions

Field	Description
7 SRES8	**A/D Resolution Select** — This bit selects the resolution of A/D conversion results as either 8 or 10 bits. The A/D converter has an accuracy of 10 bits; however, if low resolution is required, the conversion can be speeded up by selecting 8-bit resolution. 0 10-bit resolution 8-bit resolution
6–5 SMP[1:0]	**Sample Time Select** — These two bits select the length of the second phase of the sample time in units of ATD conversion clock cycles. Note that the ATD conversion clock period is itself a function of the prescaler value (bits PRS4–0). The sample time consists of two phases. The first phase is two ATD conversion clock cycles long and transfers the sample quickly (via the buffer amplifier) onto the A/D machine's storage node. The second phase attaches the external analog signal directly to the storage node for final charging and high accuracy. Table 7-11 lists the lengths available for the second sample phase.
4–0 PRS[4:0]	**ATD Clock Prescaler —** These 5 bits are the binary value prescaler value PRS. The ATD conversion clock frequency is calculated as follows: $ATDclock = \frac{[BusClock]}{[PRS+1]} \times 0.5$ **Note:** The maximum ATD conversion clock frequency is half the bus clock. The default (after reset) prescaler value is 5 which results in a default ATD conversion clock frequency that is bus clock divided by 12. Table 7-12 illustrates the divide-by operation and the appropriate range of the bus clock.

Table 7-11. Sample Time Select

SMP1	SMP0	Length of 2nd Phase of Sample Time
0	0	2 A/D conversion clock periods
0	1	4 A/D conversion clock periods
1	0	8 A/D conversion clock periods
1	1	16 A/D conversion clock periods

Table 7-12. Clock Prescaler Values

Prescale Value	Total Divisor Value	Max. Bus Clock[1]	Min. Bus Clock[2]
00000	Divide by 2	4 MHz	1 MHz
00001	Divide by 4	8 MHz	2 MHz
00010	Divide by 6	12 MHz	3 MHz
00011	Divide by 8	16 MHz	4 MHz
00100	Divide by 10	20 MHz	5 MHz
00101	Divide by 12	24 MHz	6 MHz
00110	Divide by 14	28 MHz	7 MHz
00111	Divide by 16	32 MHz	8 MHz
01000	Divide by 18	36 MHz	9 MHz
01001	Divide by 20	40 MHz	10 MHz
01010	Divide by 22	44 MHz	11 MHz
01011	Divide by 24	48 MHz	12 MHz
01100	Divide by 26	52 MHz	13 MHz
01101	Divide by 28	56 MHz	14 MHz
01110	Divide by 30	60 MHz	15 MHz
01111	Divide by 32	64 MHz	16 MHz
10000	Divide by 34	68 MHz	17 MHz
10001	Divide by 36	72 MHz	18 MHz
10010	Divide by 38	76 MHz	19 MHz
10011	Divide by 40	80 MHz	20 MHz
10100	Divide by 42	84 MHz	21 MHz
10101	Divide by 44	88 MHz	22 MHz
10110	Divide by 46	92 MHz	23 MHz
10111	Divide by 48	96 MHz	24 MHz
11000	Divide by 50	100 MHz	25 MHz
11001	Divide by 52	104 MHz	26 MHz
11010	Divide by 54	108 MHz	27 MHz
11011	Divide by 56	112 MHz	28 MHz
11100	Divide by 58	116 MHz	29 MHz
11101	Divide by 60	120 MHz	30 MHz
11110	Divide by 62	124 MHz	31 MHz
11111	Divide by 64	128 MHz	32 MHz

1 Maximum ATD conversion clock frequency is 2 MHz. The maximum allowed bus clock frequency is shown in this column.

2 Minimum ATD conversion clock frequency is 500 kHz. The minimum allowed bus clock frequency is shown in this column.

7.3.2.6 ATD Control Register 5 (ATDCTL5)

This register selects the type of conversion sequence and the analog input channels sampled. Writes to this register will abort current conversion sequence and start a new conversion sequence.

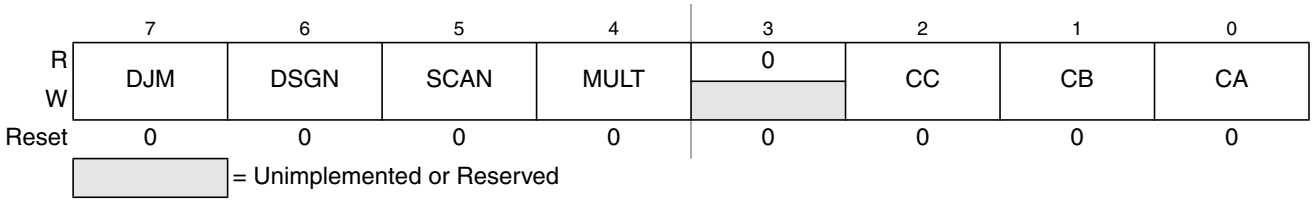


Figure 7-8. ATD Control Register 5 (ATDCTL5)

Read: Anytime

Write: Anytime

Table 7-13. ATDCTL5 Field Descriptions

Field	Description
7 DJM	**Result Register Data Justification** — This bit controls justification of conversion data in the result registers. See Section 7.3.2.13, “ATD Conversion Result Registers (ATDDRx),” for details. 0 Left justified data in the result registers 1 Right justified data in the result registers
6 DSGN	**Result Register Data Signed or Unsigned Representation** — This bit selects between signed and unsigned conversion data representation in the result registers. Signed data is represented as 2’s complement. Signed data is not available in right justification. See Section 7.3.2.13, “ATD Conversion Result Registers (ATDDRx),” for details. 0 Unsigned data representation in the result registers 1 Signed data representation in the result registers Table 7-14 summarizes the result data formats available and how they are set up using the control bits. Table 7-15 illustrates the difference between the signed and unsigned, left justified output codes for an input signal range between 0 and 5.12 Volts.
5 SCAN	**Continuous Conversion Sequence Mode** — This bit selects whether conversion sequences are performed continuously or only once. 0 Single conversion sequence 1 Continuous conversion sequences (scan mode)
4 MULT	**Multi-Channel Sample Mode** — When MULT is 0, the ATD sequence controller samples only from the specified analog input channel for an entire conversion sequence. The analog channel is selected by channel selection code (control bits CC/CB/CA located in ATDCTL5). When MULT is 1, the ATD sequence controller samples across channels. The number of channels sampled is determined by the sequence length value (S8C, S4C, S2C, S1C). The first analog channel examined is determined by channel selection code (CC, CB, CA control bits); subsequent channels sampled in the sequence are determined by incrementing the channel selection code. 0 Sample only one channel 1 Sample across several channels
2–0 CC, CB, CA	**Analog Input Channel Select Code** — These bits select the analog input channel(s) whose signals are sampled and converted to digital codes. Table 7-16 lists the coding used to select the various analog input channels. In the case of single channel scans (MULT = 0), this selection code specified the channel examined. In the case of multi-channel scans (MULT = 1), this selection code represents the first channel to be examined in the conversion sequence. Subsequent channels are determined by incrementing channel selection code; selection codes that reach the maximum value wrap around to the minimum value.

Table 7-14. Available Result Data Formats

SRES8	DJM	DSGN	Result Data Formats Description and Bus Bit Mapping
1	0	0	8-bit / left justified / unsigned — bits 8–15
1	0	1	8-bit / left justified / signed — bits 8–15
1	1	X	8-bit / right justified / unsigned — bits 0–7
0	0	0	10-bit / left justified / unsigned — bits 6–15
0	0	1	10-bit / left justified / signed — bits 6–15
0	1	X	10-bit / right justified / unsigned — bits 0–9

Table 7-15. Left Justified, Signed, and Unsigned ATD Output Codes

Input Signal V_{RL} = 0 Volts V_{RH} = 5.12 Volts	Signed 8-Bit Codes	Unsigned 8-Bit Codes	Signed 10-Bit Codes	Unsigned 10-Bit Codes
5.120 Volts	7F	FF	7FC0	FFC0
5.100	7F	FF	7F00	FF00
5.080	7E	FE	7E00	FE00
2.580	01	81	0100	8100
2.560	00	80	0000	8000
2.540	FF	7F	FF00	7F00
0.020	81	01	8100	0100
0.000	80	00	8000	0000

Table 7-16. Analog Input Channel Select Coding

CC	CB	CA	Analog Input Channel
0	0	0	AN0
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3
1	0	0	AN4
1	0	1	AN5
1	1	0	AN6
1	1	1	AN7

7.3.2.7 ATD Status Register 0 (ATDSTAT0)

This read-only register contains the sequence complete flag, overrun flags for external trigger and FIFO mode, and the conversion counter.

	7	6	5	4	3	2	1	0
R	SCF	0	ETORF	FIFOR	0	CC2	CC1	CC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-9. ATD Status Register 0 (ATDSTAT0)

Read: Anytime

Write: Anytime (No effect on (CC2, CC1, CC0))

Table 7-17. ATDSTAT0 Field Descriptions

Field	Description
7 SCF	**Sequence Complete Flag** — This flag is set upon completion of a conversion sequence. If conversion sequences are continuously performed (SCAN = 1), the flag is set after each one is completed. This flag is cleared when one of the following occurs: A) Write “1” to SCF B) Write to ATDCTL5 (a new conversion sequence is started) C) If AFFC=1 and read of a result register 0 Conversion sequence not completed 1 Conversion sequence has completed
5 ETORF	**External Trigger Overrun Flag** — While in edge trigger mode (ETRIGLE = 0), if additional active edges are detected while a conversion sequence is in process the overrun flag is set. This flag is cleared when one of the following occurs: A) Write “1” to ETORF B) Write to ATDCTL2, ATDCTL3 or ATDCTL4 (a conversion sequence is aborted) C) Write to ATDCTL5 (a new conversion sequence is started) 0 No External trigger over run error has occurred 1 External trigger over run error has occurred
4 FIFOR	**FIFO Over Run Flag** — This bit indicates that a result register has been written to before its associated conversion complete flag (CCF) has been cleared. This flag is most useful when using the FIFO mode because the flag potentially indicates that result registers are out of sync with the input channels. However, it is also practical for non-FIFO modes, and indicates that a result register has been over written before it has been read (i.e., the old data has been lost). This flag is cleared when one of the following occurs: A) Write “1” to FIFOR B) Start a new conversion sequence (write to ATDCTL5 or external trigger) 0 No over run has occurred 1 An over run condition exists
2–0 CC[2:0]	**Conversion Counter** — These 3 read-only bits are the binary value of the conversion counter. The conversion counter points to the result register that will receive the result of the current conversion. E.g. CC2 = 1, CC1 = 1, CC0 = 0 indicates that the result of the current conversion will be in ATD result register 6. If in non-FIFO mode (FIFO = 0) the conversion counter is initialized to zero at the begin and end of the conversion sequence. If in FIFO mode (FIFO = 1) the register counter is not initialized. The conversion counters wraps around when its maximum value is reached. Aborting a conversion or starting a new conversion by write to an ATDCTL register (ATDCTL5-0) clears the conversion counter even if FIFO=1.

7.3.2.8 Reserved Register (ATDTEST0)

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

= Unimplemented or Reserved

Figure 7-10. Reserved Register (ATDTEST0)

Read: Anytime, returns unpredictable values

Write: Anytime in special modes, unimplemented in normal modes

NOTE

Writing to this register when in special modes can alter functionality.

7.3.2.9 ATD Test Register 1 (ATDTEST1)

This register contains the SC bit used to enable special channel conversions.

	7	6	5	4	3	2	1	0
R	U	U	0	0	0	0	0	SC
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-11. ATD Test Register 1 (ATDTEST1)

Read: Anytime, returns unpredictable values for Bit7 and Bit6

Write: Anytime

Table 7-18. ATDTEST1 Field Descriptions

Field	Description
0 SC	**Special Channel Conversion Bit** — If this bit is set, then special channel conversion can be selected using CC, CB and CA of ATDCTL5. Table 7-19 lists the coding. 0 Special channel conversions disabled 1 Special channel conversions enabled **Note:** Always write remaining bits of ATDTEST1 (Bit7 to Bit1) zero when writing SC bit. Not doing so might result in unpredictable ATD behavior.

Table 7-19. Special Channel Select Coding

SC	CC	CB	CA	Analog Input Channel
1	0	X	X	Reserved
1	1	0	0	V_{RH}
1	1	0	1	V_{RL}
1	1	1	0	$(V_{RH}+V_{RL}) / 2$
1	1	1	1	Reserved

7.3.2.10 ATD Status Register 1 (ATDSTAT1)

This read-only register contains the conversion complete flags.

	7	6	5	4	3	2	1	0
R	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-12. ATD Status Register 1 (ATDSTAT1)

Read: Anytime

Write: Anytime, no effect

Table 7-20. ATDSTAT1 Field Descriptions

Field	Description
7–0 CCF[7:0]	**Conversion Complete Flag x (x = 7, 6, 5, 4, 3, 2, 1, 0)** — A conversion complete flag is set at the end of each conversion in a conversion sequence. The flags are associated with the conversion position in a sequence (and also the result register number). Therefore, CCF0 is set when the first conversion in a sequence is complete and the result is available in result register ATDDR0; CCF1 is set when the second conversion in a sequence is complete and the result is available in ATDDR1, and so forth. A flag CCFx (x = 7, 6, 5, 4, 3, 2,1, 70) is cleared when one of the following occurs: A) Write to ATDCTL5 (a new conversion sequence is started) B) If AFFC=0 and read of ATDSTAT1 followed by read of result register ATDDRx C) If AFFC=1 and read of result register ATDDRx In case of a concurrent set and clear on CCFx: The clearing by method A) will overwrite the set. The clearing by methods B) or C) will be overwritten by the set. 0 Conversion number x not completed 1 Conversion number x has completed, result ready in ATDDRx

7.3.2.11 ATD Input Enable Register (ATDDIEN)

	7	6	5	4	3	2	1	0
R W	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0
Reset	0	0	0	0	0	0	0	0

Figure 7-13. ATD Input Enable Register (ATDDIEN)

Read: Anytime

Write: Anytime

Table 7-21. ATDDIEN Field Descriptions

Field	Description
7–0 IEN[7:0]	**ATD Digital Input Enable on channel x (x = 7, 6, 5, 4, 3, 2, 1, 0)** — This bit controls the digital input buffer from the analog input pin (ANx) to PTADx data register. 0 Disable digital input buffer to PTADx 1 Enable digital input buffer to PTADx. **Note:** Setting this bit will enable the corresponding digital input buffer continuously. If this bit is set while simultaneously using it as an analog port, there is potentially increased power consumption because the digital input buffer maybe in the linear region.

7.3.2.12 Port Data Register (PORTAD)

The data port associated with the ATD can be configured as general-purpose I/O or input only, as specified in the device overview. The port pins are shared with the analog A/D inputs AN7–0.

	7	6	5	4	3	2	1	0
R	PTAD7	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
W								
Reset	1	1	1	1	1	1	1	1
Pin Function	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0

= Unimplemented or Reserved

Figure 7-14. Port Data Register (PORTAD)

Read: Anytime

Write: Anytime, no effect

The A/D input channels may be used for general purpose digital input.

Table 7-22. PORTAD Field Descriptions

Field	Description
7–0 PTAD[7:0]	**A/D Channel x (ANx) Digital Input (x = 7, 6, 5, 4, 3, 2, 1, 0)** — If the digital input buffer on the ANx pin is enabled (IENx = 1) or channel x is enabled as external trigger (ETRIGE = 1,ETRIGCH[2–0] = x,ETRIGSEL = 0) read returns the logic level on ANx pin (signal potentials not meeting V_{IL} or V_{IH} specifications will have an indeterminate value). If the digital input buffers are disabled (IENx = 0) and channel x is not enabled as external trigger, read returns a “1”. Reset sets all PORTAD0 bits to “1”.

7.3.2.13 ATD Conversion Result Registers (ATDDRx)

The A/D conversion results are stored in 8 read-only result registers. The result data is formatted in the result registers based on two criteria. First there is left and right justification; this selection is made using the DJM control bit in ATDCTL5. Second there is signed and unsigned data; this selection is made using the DSGN control bit in ATDCTL5. Signed data is stored in 2's complement format and only exists in left justified format. Signed data selected for right justified format is ignored.

Read: Anytime

Write: Anytime in special mode, unimplemented in normal modes

7.3.2.13.1 Left Justified Result Data

	7	6	5	4	3	2	1	0	
R	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	10-bit data
R	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	8-bit data
W									
Reset	0	0	0	0	0	0	0	0	

= Unimplemented or Reserved

Figure 7-15. Left Justified, ATD Conversion Result Register, High Byte (ATDDRxH)

	7	6	5	4	3	2	1	0
R	BIT 1	BIT 0	0	0	0	0	0	0
R	U	U	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 7-16. Left Justified, ATD Conversion Result Register, Low Byte (ATDDRxL)

7.3.2.13.2 Right Justified Result Data

	7	6	5	4	3	2	1	0	
R	0	0	0	0	0	0	BIT 9 MSB	BIT 8	10-bit data
R	0	0	0	0	0	0	0	0	8-bit data
W									
Reset	0	0	0	0	0	0	0	0	

= Unimplemented or Reserved

Figure 7-17. Right Justified, ATD Conversion Result Register, High Byte (ATDDRxH)

	7	6	5	4	3	2	1	0	
R	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	10-bit data
R	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	8-bit data
W									
Reset	0	0	0	0	0	0	0	0	

= Unimplemented or Reserved

Figure 7-18. Right Justified, ATD Conversion Result Register, Low Byte (ATDDRxL)

7.4 Functional Description

The ATD is structured in an analog and a digital sub-block.

7.4.1 Analog Sub-Block

The analog sub-block contains all analog electronics required to perform a single conversion. Separate power supplies V_{DDA} and V_{SSA} allow to isolate noise of other MCU circuitry from the analog sub-block.

7.4.1.1 Sample and Hold Machine

The sample and hold (S/H) machine accepts analog signals from the external surroundings and stores them as capacitor charge on a storage node.

The sample process uses a two stage approach. During the first stage, the sample amplifier is used to quickly charge the storage node.The second stage connects the input directly to the storage node to complete the sample for high accuracy.

When not sampling, the sample and hold machine disables its own clocks. The analog electronics still draw their quiescent current. The power down (ADPU) bit must be set to disable both the digital clocks and the analog power consumption.

The input analog signals are unipolar and must fall within the potential range of V_{SSA} to V_{DDA}.

7.4.1.2 Analog Input Multiplexer

The analog input multiplexer connects one of the 8 external analog input channels to the sample and hold machine.

7.4.1.3 Sample Buffer Amplifier

The sample amplifier is used to buffer the input analog signal so that the storage node can be quickly charged to the sample potential.

7.4.1.4 Analog-to-Digital (A/D) Machine

The A/D Machine performs analog to digital conversions. The resolution is program selectable at either 8 or 10 bits. The A/D machine uses a successive approximation architecture. It functions by comparing the stored analog sample potential with a series of digitally generated analog potentials. By following a binary search algorithm, the A/D machine locates the approximating potential that is nearest to the sampled potential.

When not converting the A/D machine disables its own clocks. The analog electronics still draws quiescent current. The power down (ADPU) bit must be set to disable both the digital clocks and the analog power consumption.

Only analog input signals within the potential range of V_{RL} to V_{RH} (A/D reference potentials) will result in a non-railed digital output codes.

7.4.2 Digital Sub-Block

This subsection explains some of the digital features in more detail. See register descriptions for all details.

7.4.2.1 External Trigger Input

The external trigger feature allows the user to synchronize ATD conversions to the external environment events rather than relying on software to signal the ATD module when ATD conversions are to take place. The external trigger signal (out of reset ATD channel 7, configurable in ATDCTL1) is programmable to be edge or level sensitive with polarity control. Table 7-23 gives a brief description of the different combinations of control bits and their effect on the external trigger function.

Table 7-23. External Trigger Control Bits

ETRIGLE	ETRIGP	ETRIGE	SCAN	Description
X	X	0	0	Ignores external trigger. Performs one conversion sequence and stops.
X	X	0	1	Ignores external trigger. Performs continuous conversion sequences.
0	0	1	X	Falling edge triggered. Performs one conversion sequence per trigger.
0	1	1	X	Rising edge triggered. Performs one conversion sequence per trigger.
1	0	1	X	Trigger active low. Performs continuous conversions while trigger is active.
1	1	1	X	Trigger active high. Performs continuous conversions while trigger is active.

During a conversion, if additional active edges are detected the overrun error flag ETORF is set.

In either level or edge triggered modes, the first conversion begins when the trigger is received. In both cases, the maximum latency time is one bus clock cycle plus any skew or delay introduced by the trigger circuitry.

NOTE

The conversion results for the external trigger ATD channel 7 have no meaning while external trigger mode is enabled.

Once ETRIGE is enabled, conversions cannot be started by a write to ATDCTL5, but rather must be triggered externally.

If the level mode is active and the external trigger both de-asserts and re-asserts itself during a conversion sequence, this does not constitute an overrun; therefore, the flag is not set. If the trigger is left asserted in level mode while a sequence is completing, another sequence will be triggered immediately.

7.4.2.2 General Purpose Digital Input Port Operation

The input channel pins can be multiplexed between analog and digital data. As analog inputs, they are multiplexed and sampled to supply signals to the A/D converter. As digital inputs, they supply external input data that can be accessed through the digital port register PORTAD (input-only).

The analog/digital multiplex operation is performed in the input pads. The input pad is always connected to the analog inputs of the ATD. The input pad signal is buffered to the digital port registers. This buffer can be turned on or off with the ATDDIEN register. This is important so that the buffer does not draw excess current when analog potentials are presented at its input.

7.4.2.3 Low Power Modes

The ATD can be configured for lower MCU power consumption in 3 different ways:

1. Stop mode: This halts A/D conversion. Exit from stop mode will resume A/D conversion, but due to the recovery time the result of this conversion should be ignored.
2. Wait mode with AWAI = 1: This halts A/D conversion. Exit from wait mode will resume A/D conversion, but due to the recovery time the result of this conversion should be ignored.
3. Writing ADPU = 0 (Note that all ATD registers remain accessible.): This aborts any A/D conversion in progress.

Note that the reset value for the ADPU bit is zero. Therefore, when this module is reset, it is reset into the power down state.

7.5 Resets

At reset the ATD is in a power down state. The reset state of each individual bit is listed within the Register Description section (see Section 7.3, “Memory Map and Register Definition”), which details the registers and their bit-field.

7.6 Interrupts

The interrupt requested by the ATD is listed in Table 7-24. Refer to the device overview chapter for related vector address and priority.

Table 7-24. ATD Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Sequence complete interrupt	I bit	ASCIE in ATDCTL2

See register descriptions for further details.

Chapter 8
Serial Communication Interface (SCIV3)

8.1 Introduction

This block description chapter provides an overview of serial communication interface (SCI) module.

The SCI allows full duplex, asynchronous, serial communication between the CPU and remote devices, including other CPUs. The SCI transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

8.1.1 Glossary

IR: infrared

IrDA: Infrared Design Association

IRQ: interrupt request

LSB: least significant bit

MSB: most significant bit

NRZ: non-return-to-Zero

RZI: return-to-zero-inverted

RXD: receive pin

SCI: serial communication interface

TXD: transmit pin

8.1.2 Features

The SCI includes these distinctive features:

- Full-duplex or single-wire operation
- Standard mark/space non-return-to-zero (NRZ) format
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse widths
- 13-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Programmable transmitter output parity
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup

- Interrupt-driven operation with eight 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

8.1.3 Modes of Operation

The SCI functions the same in normal, special, and emulation modes. It has two low-power modes, wait and stop modes.

8.1.3.1 Run Mode

Normal mode of operation.

8.1.3.2 Wait Mode

SCI operation in wait mode depends on the state of the SCISWAI bit in the SCI control register 1 (SCICR1).

- If SCISWAI is clear, the SCI operates normally when the CPU is in wait mode.
- If SCISWAI is set, SCI clock generation ceases and the SCI module enters a power-conservation state when the CPU is in wait mode. Setting SCISWAI does not affect the state of the receiver enable bit, RE, or the transmitter enable bit, TE.

  If SCISWAI is set, any transmission or reception in progress stops at wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the CPU out of wait mode. Exiting wait mode by reset aborts any transmission or reception in progress and resets the SCI.

8.1.3.3 Stop Mode

The SCI is inactive during stop mode for reduced power consumption. The STOP instruction does not affect the SCI register states, but the SCI bus clock will be disabled. The SCI operation resumes after an external interrupt brings the CPU out of stop mode. Exiting stop mode by reset aborts any transmission or reception in progress and resets the SCI.

8.1.4 Block Diagram

Figure 8-1 is a high level block diagram of the SCI module, showing the interaction of various function blocks.

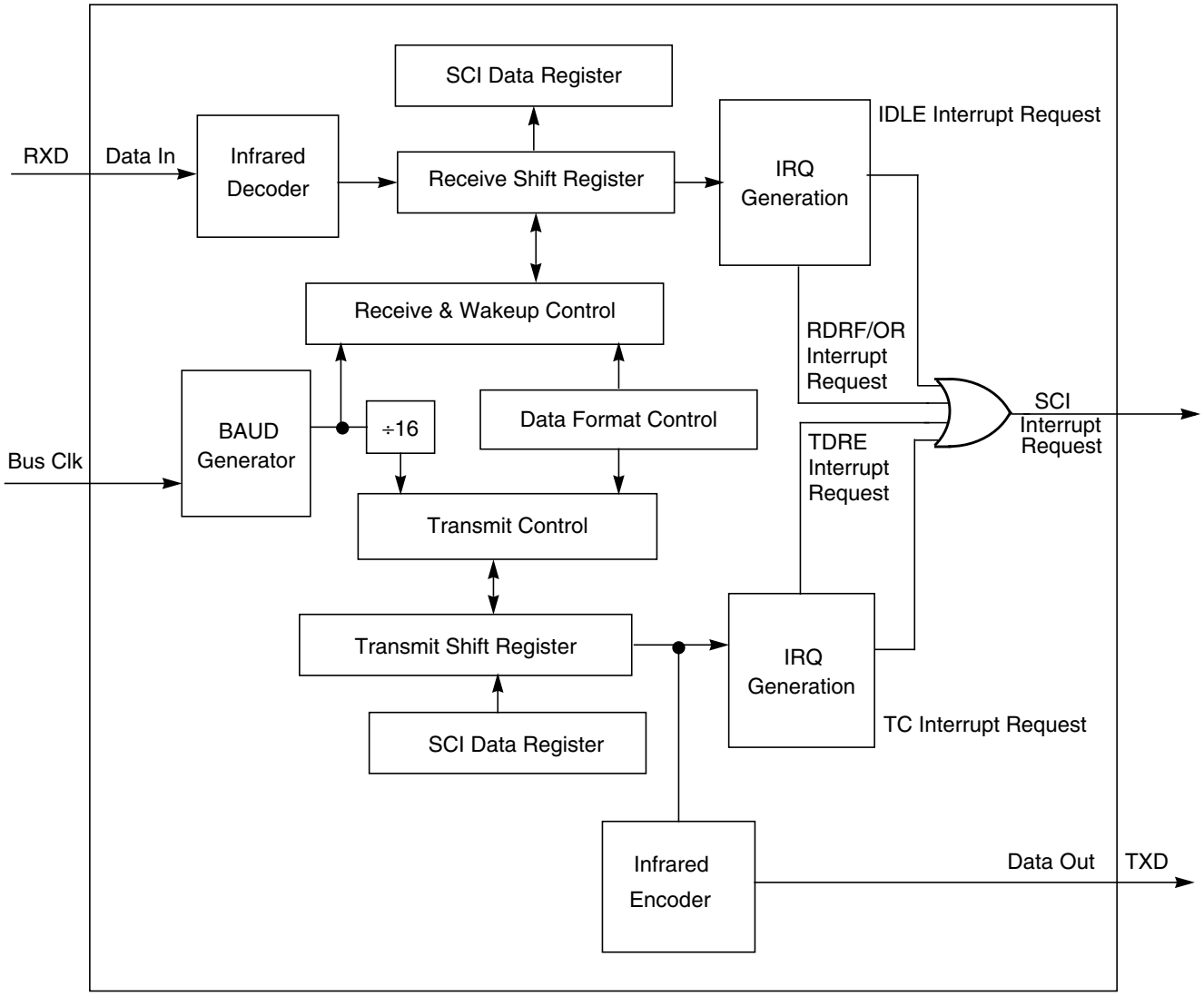


Figure 8-1. SCI Block Diagram

8.2 External Signal Descriptions

The SCI module has a total of two external pins.

8.2.1 TXD — SCI Transmit Pin

The TXD pin transmits SCI (standard or infrared) data. It will idle high in either mode and is high impedance anytime the transmitter is disabled.

8.2.2 RXD — SCI Receive Pin

The RXD pin receives SCI (standard or infrared) data. An idle line is detected as a line high. This input is ignored when the receiver is disabled and should be terminated to a known voltage.

8.3 Memory Map and Register Definition

This subsection provides a detailed description of all the SCI registers.

8.3.1 Module Memory Map

The memory map for the SCI module is given in Figure 8-2. The address listed for each register is the address offset. The total address for each register is the sum of the base address for the SCI module and the address offset for each register.

8.3.2 Register Descriptions

This subsection consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Writes to reserved register locations do not have any effect and reads of these locations return a 0. Details of register bit and field function follow the register diagrams, in bit order.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
SCIBDH	R	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
	W								
SCIBDL	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
	W								
SCICR1	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
	W								

= Unimplemented or Reserved

Figure 8-2. SCI Registers Summary

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
SCICR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	W								
SCISR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
	W								
SCISR2	R	0	0	0	0	0	BRK13	TXDIR	RAF
	W								
SCIDRH	R	R8	T8	0	0	0	0	0	0
	W								
SCIDRL	R	R7	R6	R5	R4	R3	R2	R1	R0
	W	T7	T6	T5	T4	T3	T2	T1	T0

= Unimplemented or Reserved

Figure 8-2. SCI Registers Summary

8.3.2.1 SCI Baud Rate Registers (SCIBDH and SCIBDL)

	7	6	5	4	3	2	1	0
R W	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
Reset	0	0	0	0	0	0	0	0

Figure 8-3. SCI Baud Rate Register High (SCIBDH)

Table 8-1. SCIBDH Field Descriptions

Field	Description
7 IREN	**Infrared Enable Bit** — This bit enables/disables the infrared modulation/demodulation submodule. 0 IR disabled 1 IR enabled
6:5 TNP[1:0]	**Transmitter Narrow Pulse Bits** — These bits determine if the SCI will transmit a 1/16, 3/16 or 1/32 narrow pulse. Refer to Table 8-3.
4:0 SBR[11:8]	**SCI Baud Rate Bits** — The baud rate for the SCI is determined by the bits in this register. The baud rate is calculated two different ways depending on the state of the IREN bit. The formulas for calculating the baud rate are: When IREN = 0 then, SCI baud rate = SCI module clock / (16 x SBR[12:0]) When IREN = 1 then, SCI baud rate = SCI module clock / (32 x SBR[12:1])

	7	6	5	4	3	2	1	0
R W	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
Reset	0	0	0	0	0	1	0	0

Figure 8-4. SCI Baud Rate Register Low (SCIBDL)

Table 8-2. SCIBDL Field Descriptions

Field	Description
7:0 SBR[7:0]	**SCI Baud Rate Bits** — The baud rate for the SCI is determined by the bits in this register. The baud rate is calculated two different ways depending on the state of the IREN bit. The formulas for calculating the baud rate are: When IREN = 0 then, SCI baud rate = SCI module clock / (16 x SBR[12:0]) When IREN = 1 then, SCI baud rate = SCI module clock / (32 x SBR[12:1])

Read: anytime

NOTE

If only SCIBDH is written to, a read will not return the correct data until SCIBDL is written to as well, following a write to SCIBDH.

Write: anytime

The SCI baud rate register is used to determine the baud rate of the SCI and to control the infrared modulation/demodulation submodule.

Table 8-3. IRSCI Transmit Pulse Width

TNP[1:0]	Narrow Pulse Width
11	Reserved
10	1/32
01	1/16
00	3/16

NOTE

The baud rate generator is disabled after reset and not started until the TE bit or the RE bit is set for the first time. The baud rate generator is disabled when (SBR[12:0] = 0 and IREN = 0) or (SBR[12:1] = 0 and IREN = 1).

Writing to SCIBDH has no effect without writing to SCIBDL, because writing to SCIBDH puts the data in a temporary location until SCIBDL is written to.

8.3.2.2 SCI Control Register 1 (SCICR1)

	7	6	5	4	3	2	1	0
R W	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

Figure 8-5. SCI Control Register 1 (SCICR1)

Read: anytime

Write: anytime

Table 8-4. SCICR1 Field Descriptions

Field	Description
7 LOOPS	**Loop Select Bit** — LOOPS enables loop operation. In loop operation, the RXD pin is disconnected from the SCI and the transmitter output is internally connected to the receiver input. Both the transmitter and the receiver must be enabled to use the loop function. 0 Normal operation enabled 1 Loop operation enabled The receiver input is determined by the RSRC bit.
6 SCISWAI	**SCI Stop in Wait Mode Bit** — SCISWAI disables the SCI in wait mode. 0 SCI enabled in wait mode 1 SCI disabled in wait mode
5 RSRC	**Receiver Source Bit** — When LOOPS = 1, the RSRC bit determines the source for the receiver shift register input. 0 Receiver input internally connected to transmitter output 1 Receiver input connected externally to transmitter Refer to Table 8-5.
4 M	**Data Format Mode Bit** — MODE determines whether data characters are eight or nine bits long. 0 One start bit, eight data bits, one stop bit 1 One start bit, nine data bits, one stop bit
3 WAKE	**Wakeup Condition Bit** — WAKE determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit position of a received data character or an idle condition on the RXD pin. 0 Idle line wakeup 1 Address mark wakeup
2 ILT	**Idle Line Type Bit** — ILT determines when the receiver 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. 0 Idle character bit count begins after start bit 1 Idle character bit count begins after stop bit
1 PE	**Parity Enable Bit** — PE enables the parity function. When enabled, the parity function inserts a parity bit in the most significant bit position. 0 Parity function disabled 1 Parity function enabled
0 PT	**Parity Type Bit** — PT determines whether the SCI generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit. 0 Even parity 1 Odd parity

Table 8-5. Loop Functions

LOOPS	RSRC	Function
0	x	Normal operation
1	0	Loop mode with transmitter output internally connected to receiver input
1	1	Single-wire mode with TXD pin connected to receiver input

8.3.2.3 SCI Control Register 2 (SCICR2)

	7	6	5	4	3	2	1	0
R W	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

Figure 8-6. SCI Control Register 2 (SCICR2)

Read: anytime

Write: anytime

Table 8-6. SCICR2 Field Descriptions

Field	Description
7 TIE	**Transmitter Interrupt Enable Bit** —TIE enables the transmit data register empty flag, TDRE, to generate interrupt requests. 0 TDRE interrupt requests disabled 1 TDRE interrupt requests enabled
6 TCIE	**Transmission Complete Interrupt Enable Bit** — TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0 TC interrupt requests disabled 1 TC interrupt requests enabled
5 RIE	**Receiver Full Interrupt Enable Bit** — RIE enables the receive data register full flag, RDRF, or the overrun flag, OR, to generate interrupt requests. 0 RDRF and OR interrupt requests disabled 1 RDRF and OR interrupt requests enabled
4 ILIE	**Idle Line Interrupt Enable Bit** — ILIE enables the idle line flag, IDLE, to generate interrupt requests. 0 IDLE interrupt requests disabled 1 IDLE interrupt requests enabled
3 TE	**Transmitter Enable Bit** — TE enables the SCI transmitter and configures the TXD pin as being controlled by the SCI. The TE bit can be used to queue an idle preamble. 0 Transmitter disabled 1 Transmitter enabled
2 RE	**Receiver Enable Bit** — RE enables the SCI receiver. 0 Receiver disabled 1 Receiver enabled
1 RWU	**Receiver Wakeup Bit** — Standby state 0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	**Send Break Bit** — Toggling SBK sends one break character (10 or 11 logic 0s, respectively 13 or 14 logics 0s if BRK13 is set). Toggling implies clearing the SBK bit before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10 or 11 bits, respectively 13 or 14 bits). 0 No break characters 1 Transmit break characters

8.3.2.4 SCI Status Register 1 (SCISR1)

The SCISR1 and SCISR2 registers provide inputs to the MCU for generation of SCI interrupts. Also, these registers can be polled by the MCU to check the status of these bits. The flag-clearing procedures require that the status register be read followed by a read or write to the SCI data register. It is permissible to execute other instructions between the two steps as long as it does not compromise the handling of I/O. Note that the order of operations is important for flag clearing.

	7	6	5	4	3	2	1	0
R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W								
Reset	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-7. SCI Status Register 1 (SCISR1)

Read: anytime

Write: has no meaning or effect

Table 8-7. SCISR1 Field Descriptions

Field	Description
7 TDRE	**Transmit Data Register Empty Flag** — TDRE is set when the transmit shift register receives a byte from the SCI data register. When TDRE is 1, the transmit data register (SCIDRH/L) is empty and can receive a new value to transmit.Clear TDRE by reading SCI status register 1 (SCISR1), with TDRE set and then writing to SCI data register low (SCIDRL). 0 No byte transferred to transmit shift register 1 Byte transferred to transmit shift register; transmit data register empty
6 TC	**Transmit Complete Flag** — TC is set low when there is a transmission in progress or when a preamble or break character is loaded. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted.When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL). TC is cleared automatically when data, preamble, or break is queued and ready to be sent. TC is cleared in the event of a simultaneous set and clear of the TC flag (transmission not complete). 0 Transmission in progress 1 No transmission in progress
5 RDRF	**Receive Data Register Full Flag** — RDRF is set when the data in the receive shift register transfers to the SCI data register. Clear RDRF by reading SCI status register 1 (SCISR1) with RDRF set and then reading SCI data register low (SCIDRL). 0 Data not available in SCI data register 1 Received data available in SCI data register
4 IDLE	**Idle Line Flag**[1] — IDLE is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M = 1) appear on the receiver input. After the IDLE flag is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag.Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL). 0 Receiver input is either active now or has never become active since the IDLE flag was last cleared 1 Receiver input has become idle

Table 8-7. SCISR1 Field Descriptions (continued)

Field	Description
3 OR	**Overrun Flag**[2] — OR is set when software fails to read the SCI data register before the receive shift register receives the next frame. The OR bit is set immediately after the stop bit has been completely received for the second frame. The data in the shift register is lost, but the data already in the SCI data registers is not affected. Clear OR by reading SCI status register 1 (SCISR1) with OR set and then reading SCI data register low (SCIDRL). 0 No overrun 1 Overrun
2 NF	**Noise Flag** — NF is set when the SCI detects noise on the receiver input. NF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear NF by reading SCI status register 1(SCISR1), and then reading SCI data register low (SCIDRL). 0 No noise 1 Noise
1 FE	**Framing Error Flag** — FE is set when a logic 0 is accepted as the stop bit. FE bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. FE inhibits further data reception until it is cleared. Clear FE by reading SCI status register 1 (SCISR1) with FE set and then reading the SCI data register low (SCIDRL). 0 No framing error 1 Framing error
0 PF	**Parity Error Flag** — PF is set when the parity enable bit (PE) is set and the parity of the received data does not match the parity type bit (PT). PF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear PF by reading SCI status register 1 (SCISR1), and then reading SCI data register low (SCIDRL). 0 No parity error 1 Parity error

[1] When the receiver wakeup bit (RWU) is set, an idle line condition does not set the IDLE flag.

[2] The OR flag may read back as set when RDRF flag is clear. This may happen if the following sequence of events occurs:

1. After the first frame is received, read status register SCISR1 (returns RDRF set and OR flag clear);
2. Receive second frame without reading the first frame in the data register (the second frame is not received and OR flag is set);
3. Read data register SCIDRL (returns first frame and clears RDRF flag in the status register);
4. Read status register SCISR1 (returns RDRF clear and OR set).

Event 3 may be at exactly the same time as event 2 or any time after. When this happens, a dummy SCIDRL read following event 4 will be required to clear the OR flag if further frames are to be received.

8.3.2.5 SCI Status Register 2 (SCISR2)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	BRK13	TXDIR	RAF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-8. SCI Status Register 2 (SCISR2)

Read: anytime

Write: anytime

Table 8-8. SCISR2 Field Descriptions

Field	Description
2 BRK13	**Break Transmit Character Length** — This bit determines whether the transmit break character is 10 or 11 bit respectively 13 or 14 bits long. The detection of a framing error is not affected by this bit. 0 Break character is 10 or 11 bit long 1 Break character is 13 or 14 bit long
1 TXDIR	**Transmitter Pin Data Direction in Single-Wire Mode** — This bit determines whether the TXD pin is going to be used as an input or output, in the single-wire mode of operation. This bit is only relevant in the single-wire mode of operation. 0 TXD pin to be used as an input in single-wire mode 1 TXD pin to be used as an output in single-wire mode
0 RAF	**Receiver Active Flag** — RAF is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character. 0 No reception in progress 1 Reception in progress

8.3.2.6 SCI Data Registers (SCIDRH and SCIDRL)

	7	6	5	4	3	2	1	0
R	R8	T8	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-9. SCI Data Register High (SCIDRH)

Table 8-9. SCIDRH Field Descriptions

Field	Description
7 R8	**Received Bit 8** — R8 is the ninth data bit received when the SCI is configured for 9-bit data format (M = 1).
6 T8	**Transmit Bit 8** — T8 is the ninth data bit transmitted when the SCI is configured for 9-bit data format (M = 1).

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

Figure 8-10. SCI Data Register Low (SCIDRL)

Read: anytime; reading accesses SCI receive data register

Write: anytime; writing accesses SCI transmit data register; writing to R8 has no effect

Table 8-10. SCIDRL Field Descriptions

Field	Description
7:0 R[7:0] T[7:0}	**Received bits 7 through 0** — For 9-bit or 8-bit data formats **Transmit bits 7 through 0** — For 9-bit or 8-bit formats

NOTE

If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten.The same value is transmitted until T8 is rewritten

In 8-bit data format, only SCI data register low (SCIDRL) needs to be accessed.

When transmitting in 9-bit data format and using 8-bit write instructions, write first to SCI data register high (SCIDRH) then to SCIDRL.

8.4 Functional Description

This subsection provides a complete functional description of the SCI block, detailing the operation of the design from the end user's perspective in a number of descriptions.

Figure 8-11 shows the structure of the SCI module. The SCI allows full duplex, asynchronous, serial communication between the CPU and remote devices, including other CPUs. The SCI transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

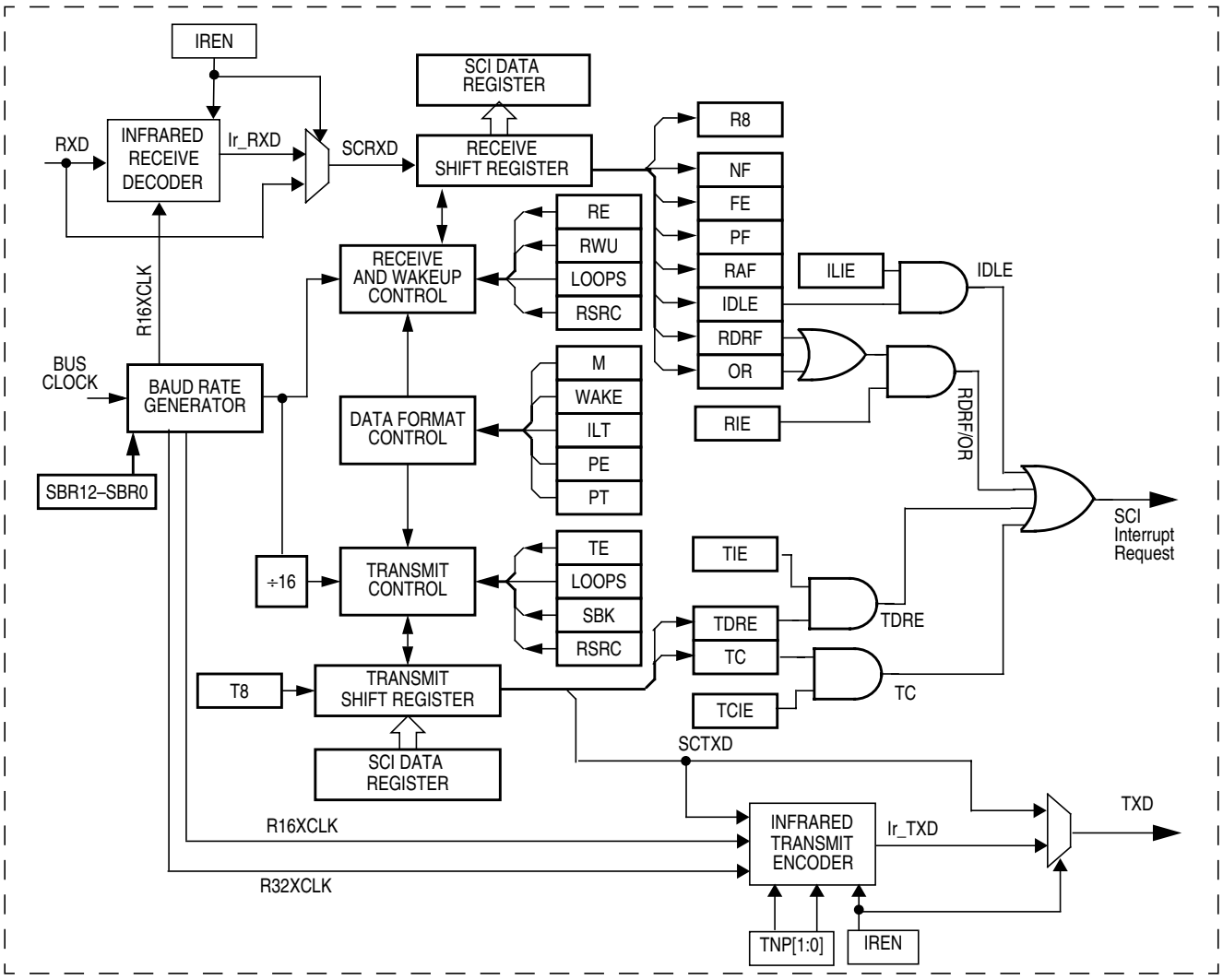


Figure 8-11. Detailed SCI Block Diagram

8.4.1 Infrared Interface Submodule

This module provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the SCI. The IrDA physical layer specification defines a half-duplex infrared communication link for exchange data. The full standard includes data rates up to 16 Mbits/s. This design covers only data rates between 2.4 kbits/s and 115.2 kbits/s.

The infrared submodule consists of two major blocks: the transmit encoder and the receive decoder. The SCI transmits serial bits of data which are encoded by the infrared submodule to transmit a narrow pulse for every 0 bit. No pulse is transmitted for every 1 bit. When receiving data, the IR pulses should be detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder (external from the MCU). The narrow pulses are then stretched by the infrared submodule to get back to a serial bit stream to be received by the SCI. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that uses active low pulses.

The infrared submodule receives its clock sources from the SCI. One of these two clocks are selected in the infrared submodule in order to generate either 3/16, 1/16, or 1/32 narrow pulses during transmission. The infrared block receives two clock sources from the SCI, R16XCLK, and R32XCLK, which are configured to generate the narrow pulse width during transmission. The R16XCLK and R32XCLK are internal clocks with frequencies 16 and 32 times the baud rate respectively. Both R16XCLK and R32XCLK clocks are used for transmitting data. The receive decoder uses only the R16XCLK clock.

8.4.1.1 Infrared Transmit Encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the TXD pin. A narrow pulse is transmitted for a 0 bit and no pulse for a 1 bit. The narrow pulse is sent in the middle of the bit with a duration of 1/32, 1/16, or 3/16 of a bit time.

8.4.1.2 Infrared Receive Decoder

The infrared receive block converts data from the RXD pin to the receive shift register. A narrow pulse is expected for each 0 received and no pulse is expected for each 1 received. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

8.4.2 Data Format

The SCI uses the standard NRZ mark/space data format. When Infrared is enabled, the SCI uses RZI data format where 0s are represented by light pulses and 1s remain low. See Figure 8-12.

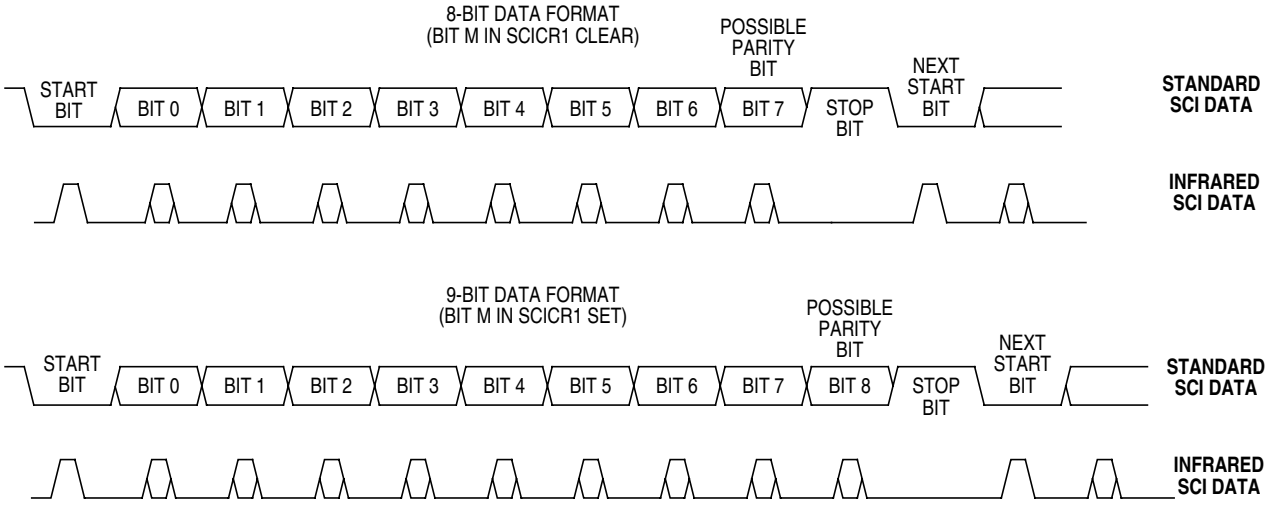


Figure 8-12. SCI Data Formats

Each data character is contained in a frame that includes a start bit, eight or nine data bits, and a stop bit. Clearing the M bit in SCI control register 1 configures the SCI for 8-bit data characters. A frame with eight data bits has a total of 10 bits. Setting the M bit configures the SCI for nine-bit data characters. A frame with nine data bits has a total of 11 bits

Table 8-11. Example of 8-bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	8	0	0	1
1	7	0	1	1
1	7	1[1]	0	1

[1] The address bit identifies the frame as an address character. See Section 8.4.5.6, “Receiver Wakeup”.

When the SCI is configured for 9-bit data characters, the ninth data bit is the T8 bit in SCI data register high (SCIDRH). It remains unchanged after transmission and can be used repeatedly without rewriting it. A frame with nine data bits has a total of 11 bits.

Table 8-12. Example of 9-Bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	9	0	0	1
1	8	0	1	1
1	8	1[1]	0	1

[1] The address bit identifies the frame as an address character. See Section 8.4.5.6, “Receiver Wakeup”.

8.4.3 Baud Rate Generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 0 to 8191 written to the SBR[12:0] bits determines the module clock divisor. The SBR bits are in the SCI baud rate registers (SCIBDH and SCIBDL). The baud rate clock is synchronized with the bus clock and drives the receiver. The baud rate clock divided by 16 drives the transmitter. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to one source of error:

- Integer division of the module clock may not give the exact target frequency.

Table 8-13 lists some examples of achieving target baud rates with a module clock frequency of 10.2 MHz.

When IREN = 0 then,

SCI baud rate = SCI module clock / (16 * SCIBR[12:0])

Table 8-13. Baud Rates (Example: Module Clock = 10.2 MHz)

Bits SBR[12–0]	Receiver Clock (Hz)	Transmitter Clock (Hz)	Target Baud Rate	Error (%)
17	600,000.0	37,500.0	38,400	2.3
33	309,090.9	19,318.2	19,200	.62
66	154,545.5	9659.1	9600	.62
133	76,691.7	4793.2	4800	.14
266	38,345.9	2396.6	2400	.14
531	19,209.0	1200.6	1200	.11
1062	9604.5	600.3	600	.05
2125	4800.0	300.0	300	.00
4250	2400.0	150.0	150	.00
5795	1760.1	110.0	110	.00

8.4.4 Transmitter

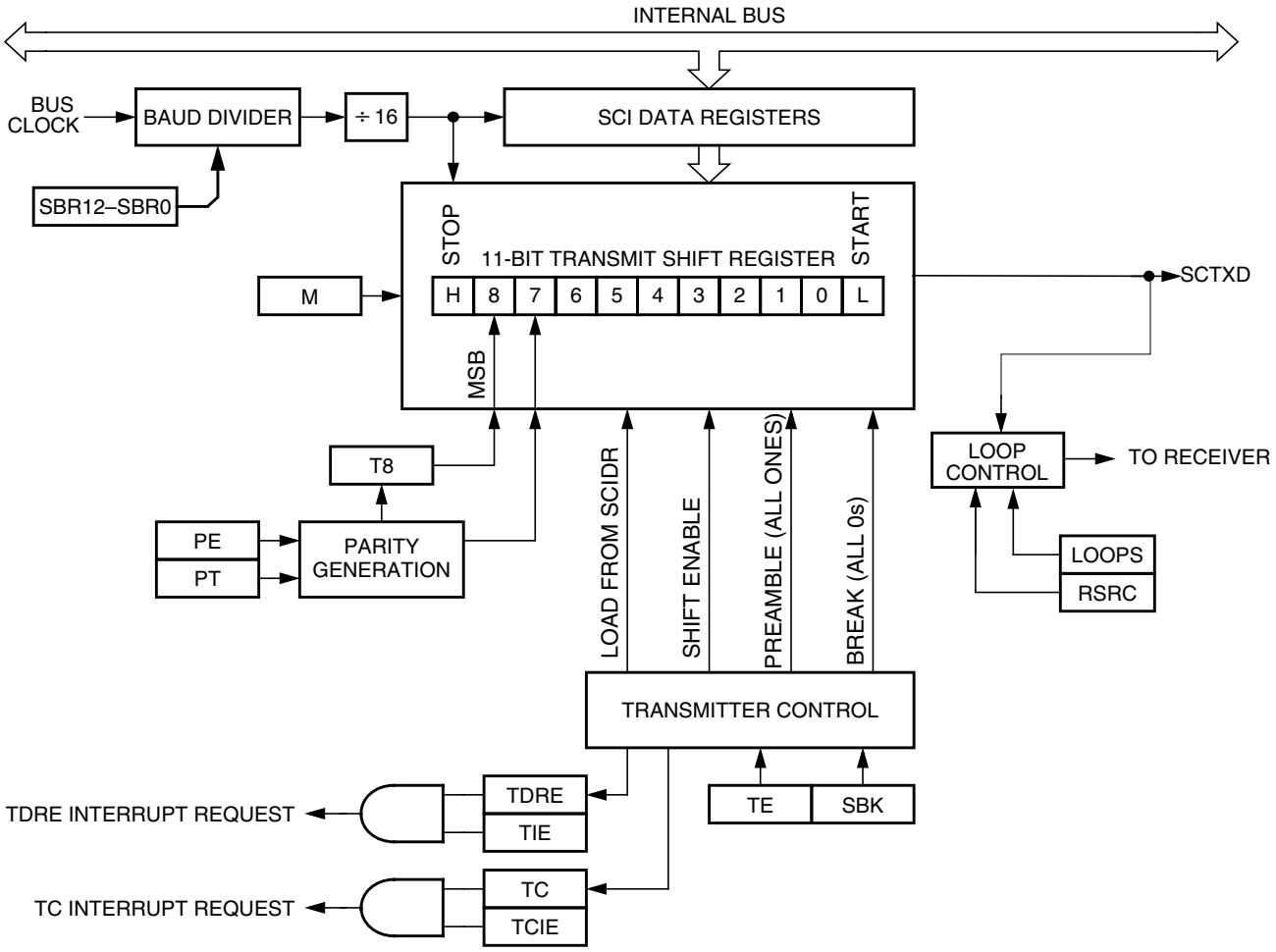


Figure 8-13. Transmitter Block Diagram

8.4.4.1 Transmitter Character Length

The SCI transmitter can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When transmitting 9-bit data, bit T8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

8.4.4.2 Character Transmission

To transmit data, the MCU writes the data bits to the SCI data registers (SCIDRH/SCIDRL), which in turn are transferred to the transmitter shift register. The transmit shift register then shifts a frame out through the TXD pin, after it has prefaced them with a start bit and appended them with a stop bit. The SCI data registers (SCIDRH and SCIDRL) are the write-only buffers between the internal data bus and the transmit shift register.

The SCI also sets a flag, the transmit data register empty flag (TDRE), every time it transfers data from the buffer (SCIDRH/L) to the transmitter shift register. The transmit driver routine may respond to this

flag by writing another byte to the transmitter buffer (SCIDRH/SCIDRL), while the shift register is shifting out the first byte.

To initiate an SCI transmission:

1. Configure the SCI:
   a) Select a baud rate. Write this value to the SCI baud registers (SCIBDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is 0. Writing to the SCIBDH has no effect without also writing to SCIBDL.
   b) Write to SCICR1 to configure word length, parity, and other configuration bits (LOOPS, RSRC, M, WAKE, ILT, PE, and PT).
   c) Enable the transmitter, interrupts, receive, and wake up as required, by writing to the SCICR2 register bits (TIE, TCIE, RIE, ILIE, TE, RE, RWU, and SBK). A preamble or idle character will now be shifted out of the transmitter shift register.
2. Transmit procedure for each byte:
   a) Poll the TDRE flag by reading the SCISR1 or responding to the TDRE interrupt. Keep in mind that the TDRE bit resets to 1.
   b) If the TDRE flag is set, write the data to be transmitted to SCIDRH/L, where the ninth bit is written to the T8 bit in SCIDRH if the SCI is in 9-bit data format. A new transmission will not result until the TDRE flag has been cleared.
3. Repeat step 2 for each subsequent transmission.

NOTE

The TDRE flag is set when the shift register is loaded with the next data to be transmitted from SCIDRH/L, which happens, generally speaking, a little over half-way through the stop bit of the previous frame. Specifically, this transfer occurs 9/16ths of a bit time AFTER the start of the stop bit of the previous frame.

Writing the TE bit from 0 to a 1 automatically loads the transmit shift register with a preamble of 10 logic 1s (if M = 0) or 11 logic 1s (if M = 1). After the preamble shifts out, control logic transfers the data from the SCI data register 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.

Hardware supports odd or even parity. When parity is enabled, the most significant bit (MSB) of the data character is the parity bit.

The transmit data register empty flag, TDRE, in SCI status register 1 (SCISR1) becomes set when the SCI data register transfers a byte to the transmit shift register. The TDRE flag indicates that the SCI data register can accept new data from the internal data bus. If the transmit interrupt enable bit, TIE, in SCI control register 2 (SCICR2) is also set, the TDRE flag generates a transmitter interrupt request.

When the transmit shift register is not transmitting a frame, the TXD pin goes to the idle condition, logic 1. If at any time software clears the TE bit in SCI control register 2 (SCICR2), the transmitter enable signal goes low and the transmit signal goes idle.

If software clears TE while a transmission is in progress (TC = 0), the frame in the transmit shift register continues to shift out. To avoid accidentally cutting off the last frame in a message, always wait for TDRE to go high after the last frame before clearing TE.

To separate messages with preambles with minimum idle line time, use this sequence between messages:

1. Write the last byte of the first message to SCIDRH/L.
2. Wait for the TDRE flag to go high, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting the TE bit.
4. Write the first byte of the second message to SCIDRH/L.

8.4.4.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCI control register 2 (SCICR2) 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 SCI control register 1 (SCICR1). 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 frame.

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 these effects on SCI registers:

- Sets the framing error flag, FE
- Sets the receive data register full flag, RDRF
- Clears the SCI data registers (SCIDRH/L)
- May set the overrun flag, OR, noise flag, NF, parity error flag, PE, or the receiver active flag, RAF (see Section 8.3.2.4, “SCI Status Register 1 (SCISR1)” and Section 8.3.2.5, “SCI Status Register 2 (SCISR2)”).

8.4.4.4 Idle Characters

An idle character (or preamble) contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in SCI control register 1 (SCICR1). The preamble is a synchronizing idle character that begins the first transmission initiated after writing the TE bit from 0 to 1.

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 frame currently being transmitted.

NOTE

When queueing an idle character, return the TE bit to logic 1 before the stop bit of the current frame shifts out through the TXD pin. Setting TE after the stop bit appears on TXD causes data previously written to the SCI data register to be lost. Toggle the TE bit for a queued idle character while the

TDRE flag is set and immediately before writing the next byte to the SCI data register.

If the TE bit is clear and the transmission is complete, the SCI is not the master of the TXD pin

8.4.5 Receiver

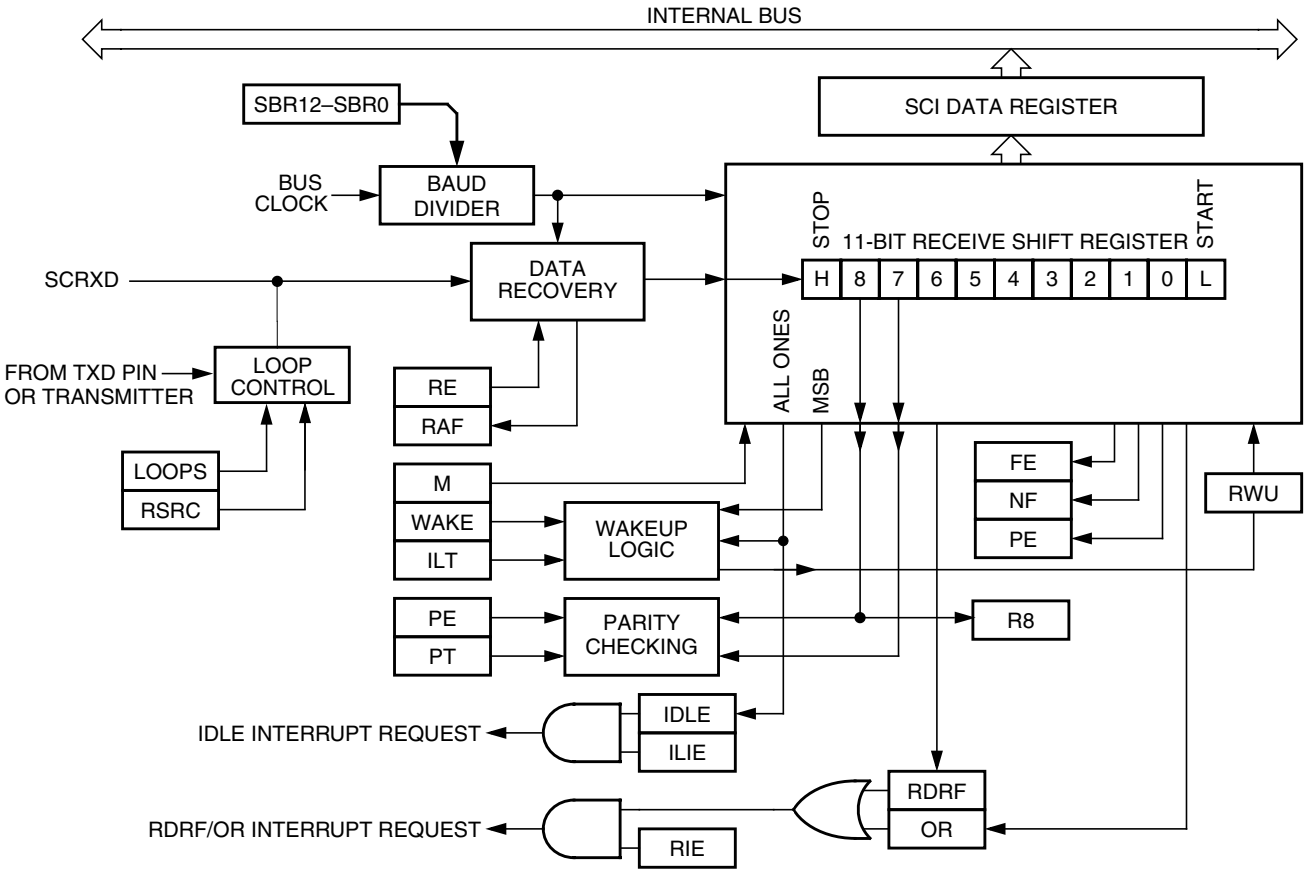


Figure 8-14. SCI Receiver Block Diagram

8.4.5.1 Receiver Character Length

The SCI receiver can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When receiving 9-bit data, bit R8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

8.4.5.2 Character Reception

During an SCI reception, the receive shift register shifts a frame in from the RXD pin. The SCI data register is the read-only buffer between the internal data bus and the receive shift register.

After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the SCI data register. The receive data register full flag, RDRF, in SCI status register 1 (SCISR1) becomes set,

indicating that the received byte can be read. If the receive interrupt enable bit, RIE, in SCI control register 2 (SCICR2) is also set, the RDRF flag generates an RDRF interrupt request.

8.4.5.3 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 (see Figure 8-15) is re-synchronized:

- 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)

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.

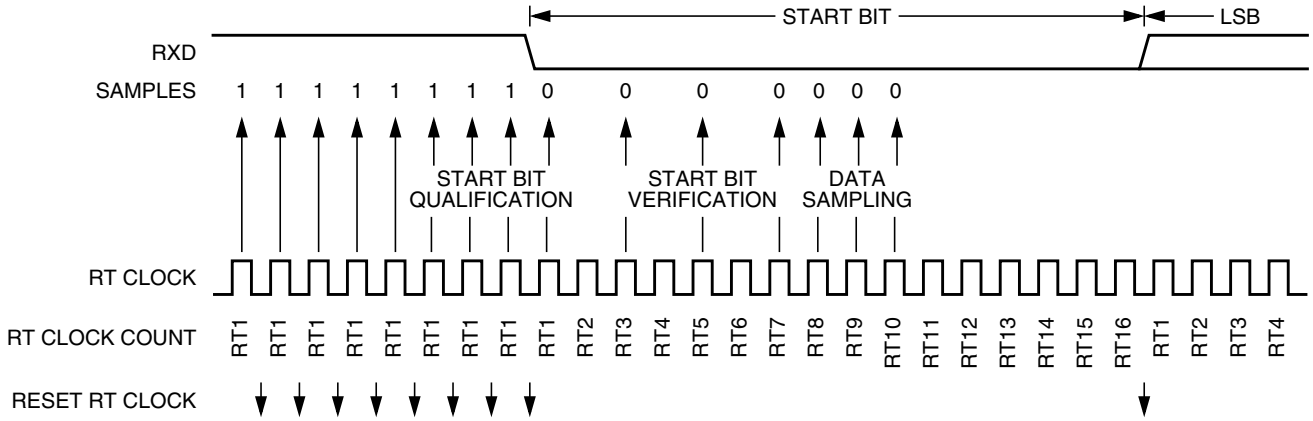


Figure 8-15. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Table 8-14 summarizes the results of the start bit verification samples.

Table 8-14. 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 8-15 summarizes the results of the data bit samples.

Table 8-15. 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 (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 8-16 summarizes the results of the stop bit samples.

Table 8-16. 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

In Figure 8-16 the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

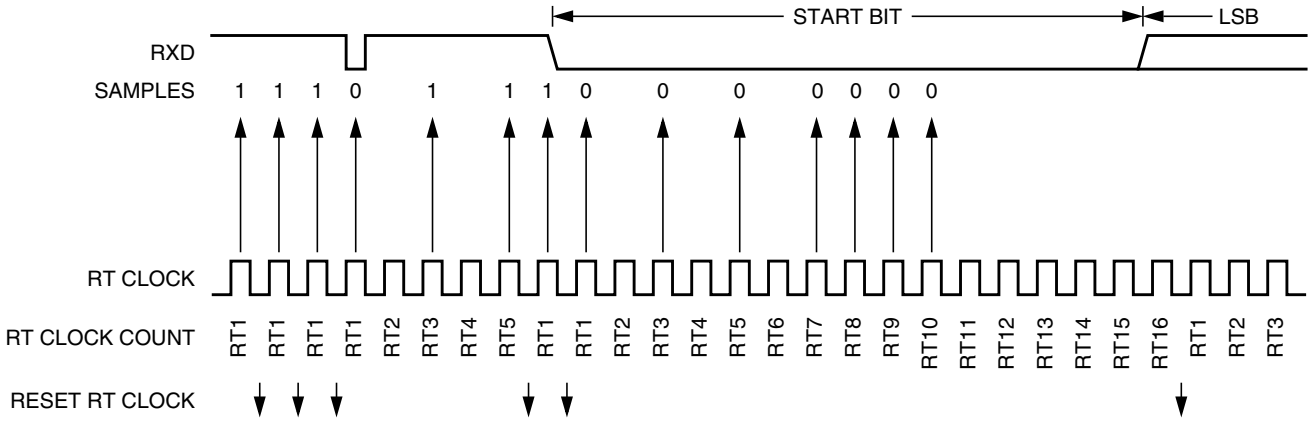


Figure 8-16. Start Bit Search Example 1

In Figure 8-17, verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

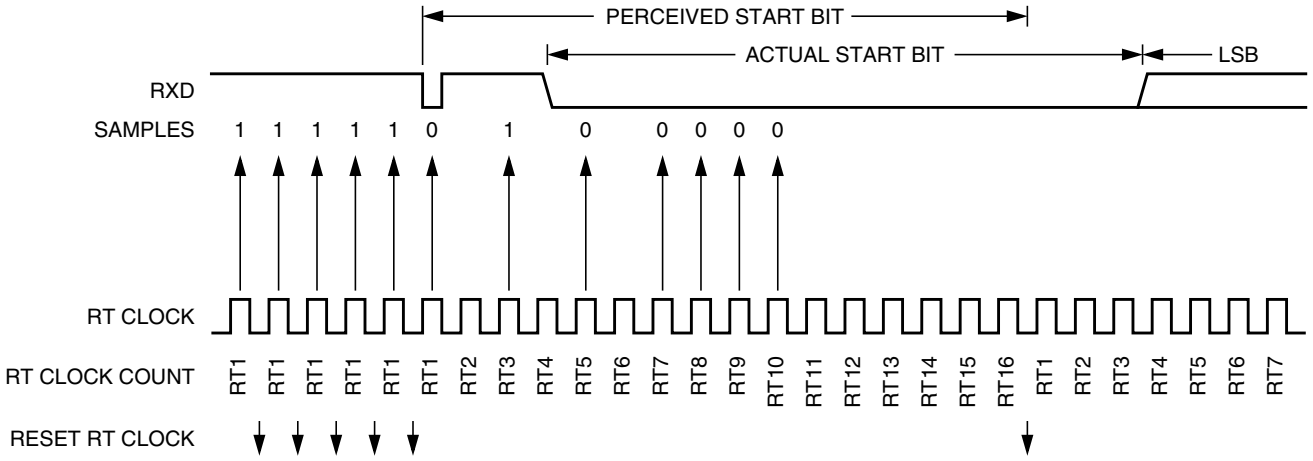


Figure 8-17. Start Bit Search Example 2

In Figure 8-18, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

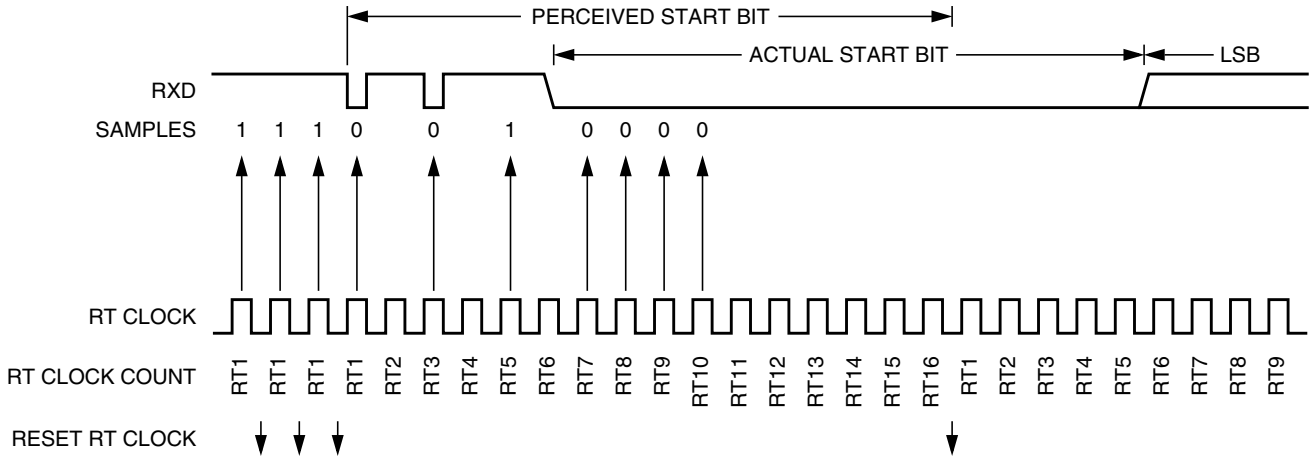


Figure 8-18. Start Bit Search Example 3

Figure 8-19 shows the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

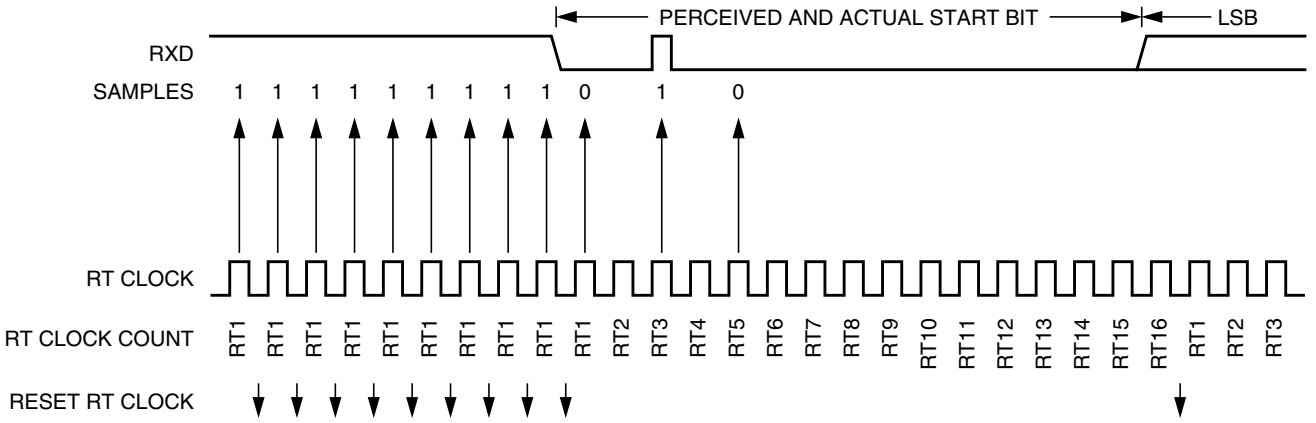


Figure 8-19. Start Bit Search Example 4

Figure 8-20 shows a burst of noise near the beginning of the start bit that resets the RT clock. The sample after the reset is low but is not preceded by three high samples that would qualify as a falling edge. Depending on the timing of the start bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

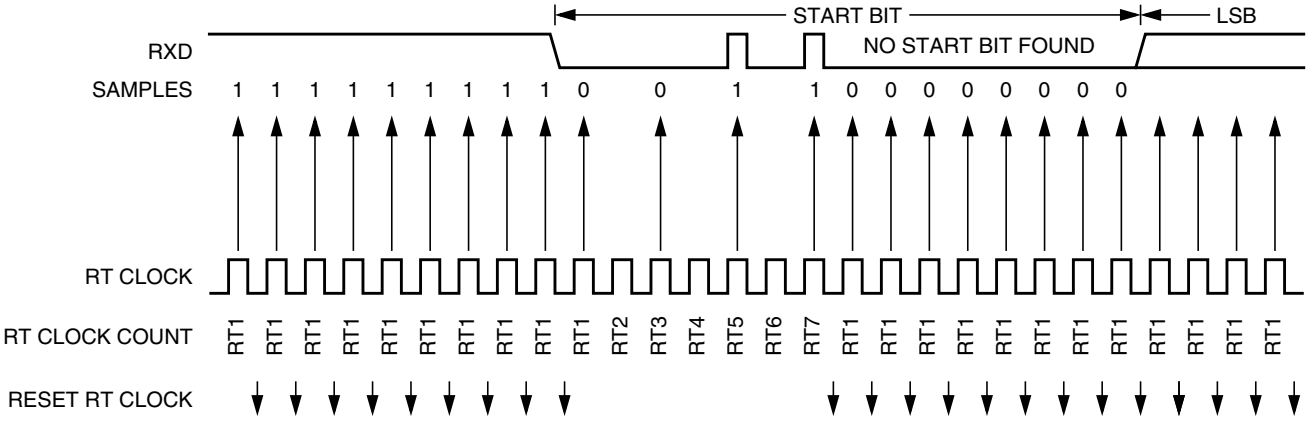


Figure 8-20. Start Bit Search Example 5

In Figure 8-21, a noise burst makes the majority of data samples RT8, RT9, and RT10 high. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored.

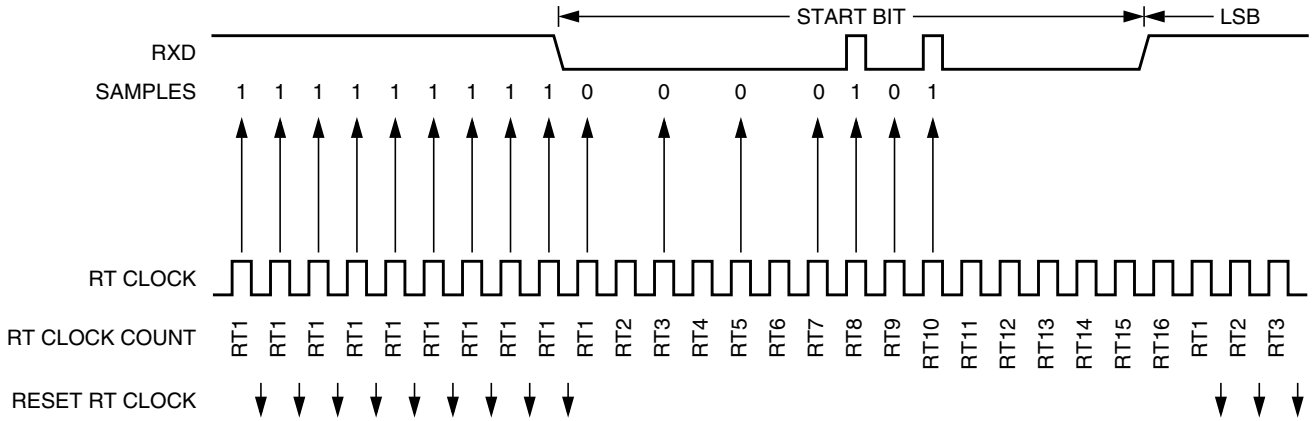


Figure 8-21. Start Bit Search Example 6

8.4.5.4 Framing Errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, FE, in SCI status register 1 (SCISR1). A break character also sets the FE flag because a break character has no stop bit. The FE flag is set at the same time that the RDRF flag is set.

8.4.5.5 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 (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic 0.

As the receiver samples an incoming frame, it re-synchronizes the RT clock on any valid falling edge within the frame. Re synchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

8.4.5.5.1 Slow Data Tolerance

Figure 8-22 shows how much a slow received frame 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.

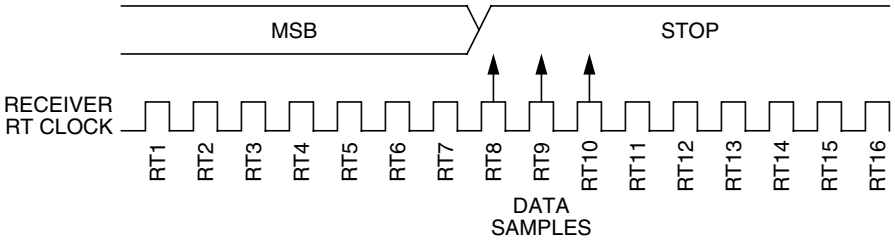


Figure 8-22. Slow Data

Let’s take RTr as receiver RT clock and RTt as transmitter RT clock.

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles +7 RTr cycles =151 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in Figure 8-22, the receiver counts 151 RTr cycles at the point when the count of the transmitting device is 9 bit times x 16 RTt cycles = 144 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

((151 – 144) / 151) x 100 = 4.63%

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 7 RTr cycles = 167 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in Figure 8-22, the receiver counts 167 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

((167 – 160) / 167) X 100 = 4.19%

8.4.5.5.2 Fast Data Tolerance

Figure 8-23 shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but continues to be sampled at RT8, RT9, and RT10.

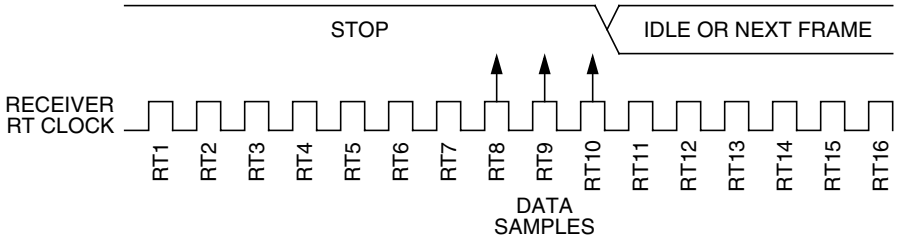


Figure 8-23. Fast Data

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles + 10 RTr cycles = 154 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 8-23, the receiver counts 154 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

((160 – 154) / 160) x 100 = 3.75%

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 10 RTr cycles = 170 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 8-23, the receiver counts 170 RTr cycles at the point when the count of the transmitting device is 11 bit times x 16 RTt cycles = 176 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

((176 – 170) / 176) x 100 = 3.40%

8.4.5.6 Receiver Wakeup

To enable the SCI to 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 SCI control register 2 (SCICR2) puts the receiver into standby state during which receiver interrupts are disabled.The SCI will continue to load the receive data into the SCIDRH/L registers, but it will not set the RDRF flag.

The transmitting device can address messages to selected receivers by including addressing information in the initial frame or frames of each message.

The WAKE bit in SCI control register 1 (SCICR1) determines how the SCI is brought out of the standby state to process an incoming message. The WAKE bit enables either idle line wakeup or address mark wakeup.

8.4.5.6.1 Idle Input Line Wakeup (WAKE = 0)

In this wakeup method, an idle condition on the RXD pin clears the RWU bit and wakes up the SCI. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another idle character appears on the RXD pin.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

The idle character that wakes a receiver does not set the receiver idle bit, IDLE, or the receive data register full flag, RDRF.

The idle line type bit, ILT, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit. ILT is in SCI control register 1 (SCICR1).

8.4.5.6.2 Address Mark Wakeup (WAKE = 1)

In this wakeup method, a logic 1 in the most significant bit (MSB) position of a frame clears the RWU bit and wakes up the SCI. The logic 1 in the MSB position marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another address frame appears on the RXD pin.

The logic 1 MSB of an address frame clears the receiver's RWU bit before the stop bit is received and sets the RDRF flag.

Address mark wakeup allows messages to contain idle characters but requires that the MSB be reserved for use in address frames.

NOTE

With the WAKE bit clear, setting the RWU bit after the RXD pin has been idle can cause the receiver to wake up immediately.

8.4.6 Single-Wire Operation

Normally, the SCI uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the SCI. The SCI uses the TXD pin for both receiving and transmitting.

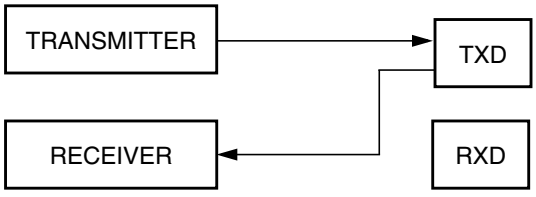


Figure 8-24. Single-Wire Operation (LOOPS = 1, RSRC = 1)

Enable single-wire operation by setting the LOOPS bit and the receiver source bit, RSRC, in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Setting the RSRC bit connects the TXD pin to the receiver. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).The TXDIR bit (SCISR2[1]) determines whether the TXD pin is going to be used as an input (TXDIR = 0) or an output (TXDIR = 1) in this mode of operation.

8.4.7 Loop Operation

In loop operation the transmitter output goes to the receiver input. The RXD pin is disconnected from the SCI

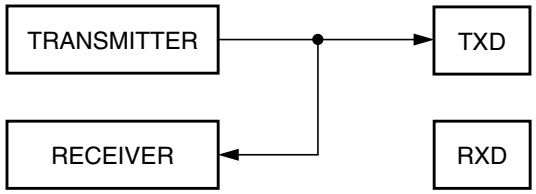


Figure 8-25. Loop Operation (LOOPS = 1, RSRC = 0)

Enable loop operation by setting the LOOPS bit and clearing the RSRC bit in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Clearing the RSRC bit connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).

8.5 Interrupts

This section describes the interrupt originated by the SCI block.The MCU must service the interrupt requests. Table 8-17 lists the five interrupt sources of the SCI.

Table 8-17. SCI Interrupt Sources

Interrupt	Source	Local Enable	Description
TDRE	SCISR1[7]	TIE	Active high level. Indicates that a byte was transferred from SCIDRH/L to the transmit shift register.
TC	SCISR1[6]	TCIE	Active high level. Indicates that a transmit is complete.
RDRF	SCISR1[5]	RIE	Active high level. The RDRF interrupt indicates that received data is available in the SCI data register.
OR	SCISR1[3]		Active high level. This interrupt indicates that an overrun condition has occurred.
IDLE	SCISR1[4]	ILIE	Active high level. Indicates that receiver input has become idle.

8.5.1 Description of Interrupt Operation

The SCI only originates interrupt requests. The following is a description of how the SCI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt number are

chip dependent. The SCI only has a single interrupt line (SCI interrupt signal, active high operation) and all the following interrupts, when generated, are ORed together and issued through that port.

8.5.1.1 TDRE Description

The TDRE interrupt is set high by the SCI when the transmit shift register receives a byte from the SCI data register. A TDRE interrupt indicates that the transmit data register (SCIDRH/L) is empty and that a new byte can be written to the SCIDRH/L for transmission.Clear TDRE by reading SCI status register 1 with TDRE set and then writing to SCI data register low (SCIDRL).

8.5.1.2 TC Description

The TC interrupt is set by the SCI when a transmission has been completed.A TC interrupt indicates that there is no transmission in progress. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL).TC is cleared automatically when data, preamble, or break is queued and ready to be sent.

8.5.1.3 RDRF Description

The RDRF interrupt is set when the data in the receive shift register transfers to the SCI data register. A RDRF interrupt indicates that the received data has been transferred to the SCI data register and that the byte can now be read by the MCU. The RDRF interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

8.5.1.4 OR Description

The OR interrupt is set when software fails to read the SCI data register before the receive shift register receives the next frame. The newly acquired data in the shift register will be lost in this case, but the data already in the SCI data registers is not affected. The OR interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

8.5.1.5 IDLE Description

The IDLE interrupt is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M = 1) appear on the receiver input. After the IDLE is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag. Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL).

8.5.2 Recovery from Wait Mode

The SCI interrupt request can be used to bring the CPU out of wait mode.

Chapter 9
Serial Peripheral Interface (SPIV3)

9.1 Introduction

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or the SPI operation can be interrupt driven.

9.1.1 Features

The SPIV3 includes these distinctive features:

- Master mode and slave mode
- Bidirectional mode
- Slave select output
- Mode fault error flag with CPU interrupt capability
- Double-buffered data register
- Serial clock with programmable polarity and phase
- Control of SPI operation during wait mode

9.1.2 Modes of Operation

The SPI functions in three modes, run, wait, and stop.

- Run Mode
  This is the basic mode of operation.
- Wait Mode
  SPI operation in wait mode is a configurable low power mode, controlled by the SPISWAI bit located in the SPICR2 register. In wait mode, if the SPISWAI bit is clear, the SPI operates like in Run Mode. If the SPISWAI bit is set, the SPI goes into a power conservative state, with the SPI clock generation turned off. If the SPI is configured as a master, any transmission in progress stops, but is resumed after CPU goes into Run Mode. If the SPI is configured as a slave, reception and transmission of a byte continues, so that the slave stays synchronized to the master.
- Stop Mode
  The SPI is inactive in stop mode for reduced power consumption. If the SPI is configured as a master, any transmission in progress stops, but is resumed after CPU goes into run mode. If the SPI is configured as a slave, reception and transmission of a byte continues, so that the slave stays synchronized to the master.

This is a high level description only, detailed descriptions of operating modes are contained in Section 9.4, “Functional Description.”

9.1.3 Block Diagram

Figure 9-1 gives an overview on the SPI architecture. The main parts of the SPI are status, control, and data registers, shifter logic, baud rate generator, master/slave control logic, and port control logic.

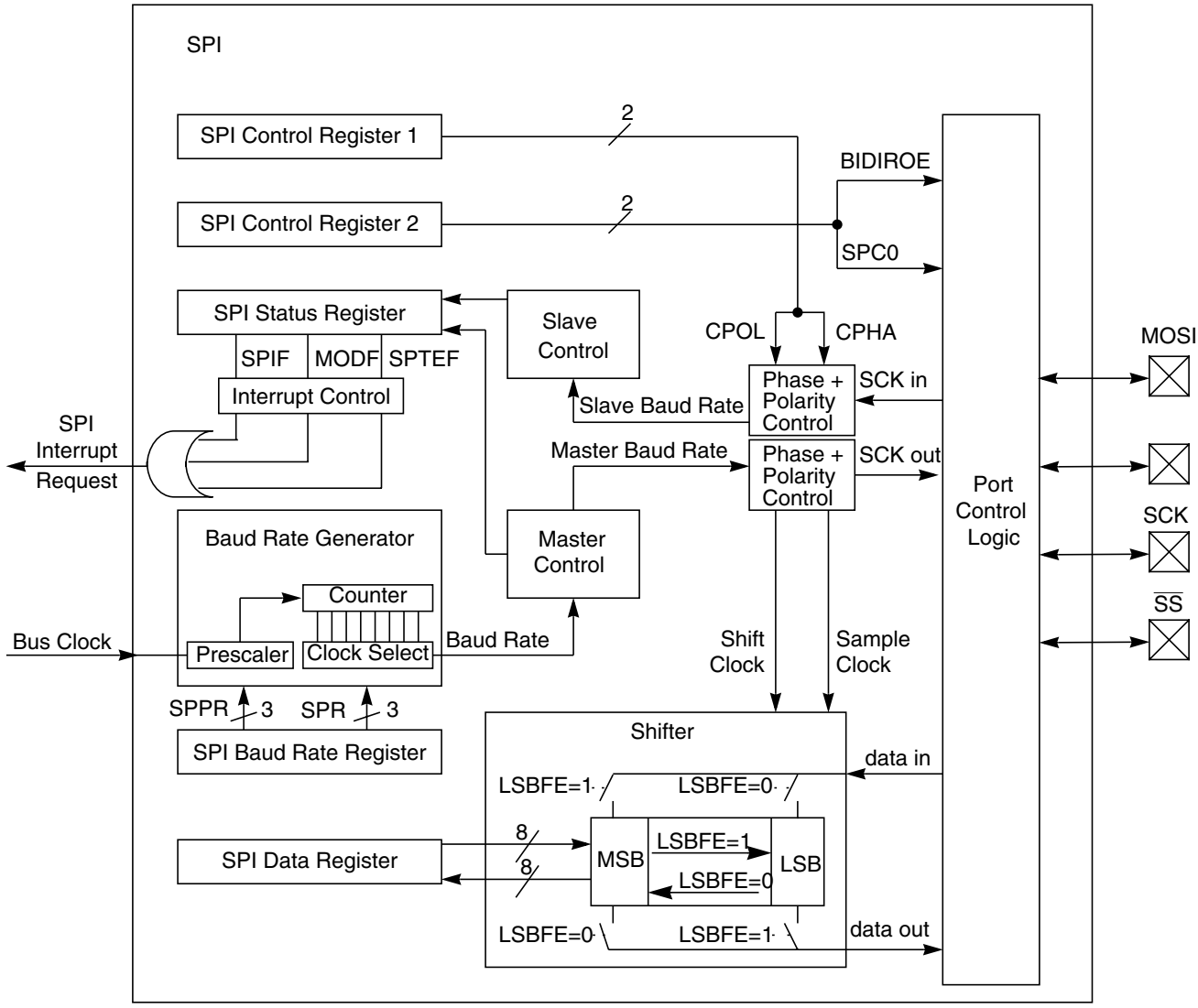


Figure 9-1. SPI Block Diagram

9.2 External Signal Description

This section lists the name and description of all ports including inputs and outputs that do, or may, connect off chip. The SPIV3 module has a total of four external pins.

9.2.1 MOSI — Master Out/Slave In Pin

This pin is used to transmit data out of the SPI module when it is configured as a master and receive data when it is configured as slave.

9.2.2 MISO — Master In/Slave Out Pin

This pin is used to transmit data out of the SPI module when it is configured as a slave and receive data when it is configured as master.

9.2.3 $\overline{SS}$ — Slave Select Pin

This pin is used to output the select signal from the SPI module to another peripheral with which a data transfer is to take place when its configured as a master and its used as an input to receive the slave select signal when the SPI is configured as slave.

9.2.4 SCK — Serial Clock Pin

This pin is used to output the clock with respect to which the SPI transfers data or receive clock in case of slave.

9.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the SPI.

The memory map for the SPIV3 is given below in Table 9-1. The address listed for each register is the sum of a base address and an address offset. The base address is defined at the SoC level and the address offset is defined at the module level. Reads from the reserved bits return zeros and writes to the reserved bits have no effect.

9.3.1 Module Memory Map

Table 9-1. SPIV3 Memory Map

Address	Use	Access
0x0000	SPI Control Register 1 (SPICR1)	R/W
0x0001	SPI Control Register 2 (SPICR2)	R/W[1]
0x0002	SPI Baud Rate Register (SPIBR)	R/W[1]
0x0003	SPI Status Register (SPISR)	R[2]
0x0004	Reserved	—[2,3]
0x0005	SPI Data Register (SPIDR)	R/W
0x0006	Reserved	—[2,3]
0x0007	Reserved	—[2,3]

1 Certain bits are non-writable.
2 Writes to this register are ignored.
3 Reading from this register returns all zeros.

9.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Name		7	6	5	4	3	2	1	0
SPICR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
	W								
SPICR2	R	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
	W								
SPIBR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
	W								
SPISR	R	SPIF	0	SPTEF	MODF	0	0	0	0
	W								
Reserved	R								
	W								
SPIDR	R	Bit 7	6	5	4	3	2	2	Bit 0
	W								
Reserved	R								
	W								
Reserved	R								
	W								

= Unimplemented or Reserved

Figure 9-2. SPI Register Summary

9.3.2.1 SPI Control Register 1 (SPICR1)

	7	6	5	4	3	2	1	0
R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
W								
Reset	0	0	0	0	0	1	0	0

Figure 9-3. SPI Control Register 1 (SPICR1)

Read: anytime

Write: anytime

Table 9-2. SPICR1 Field Descriptions

Field	Description
7 SPIE	**SPI Interrupt Enable Bit** — This bit enables SPI interrupt requests, if SPIF or MODF status flag is set. 0 SPI interrupts disabled. 1 SPI interrupts enabled.
6 SPE	**SPI System Enable Bit** — This bit enables the SPI system and dedicates the SPI port pins to SPI system functions. If SPE is cleared, SPI is disabled and forced into idle state, status bits in SPISR register are reset. 0 SPI disabled (lower power consumption). 1 SPI enabled, port pins are dedicated to SPI functions.
5 SPTIE	**SPI Transmit Interrupt Enable** — This bit enables SPI interrupt requests, if SPTEF flag is set. 0 SPTEF interrupt disabled. 1 SPTEF interrupt enabled.
4 MSTR	**SPI Master/Slave Mode Select Bit** — This bit selects, if the SPI operates in master or slave mode. Switching the SPI from master to slave or vice versa forces the SPI system into idle state. 0 SPI is in slave mode 1 SPI is in master mode
3 CPOL	**SPI Clock Polarity Bit** — This bit selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Active-high clocks selected. In idle state SCK is low. 1 Active-low clocks selected. In idle state SCK is high.
2 CPHA	**SPI Clock Phase Bit** — This bit is used to select the SPI clock format. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Sampling of data occurs at odd edges (1,3,5,...,15) of the SCK clock 1 Sampling of data occurs at even edges (2,4,6,...,16) of the SCK clock
1 SSOE	**Slave Select Output Enable** — The $\overline{SS}$ output feature is enabled only in master mode, if MODFEN is set, by asserting the SSOE as shown in Table 9-3. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.
0 LSBFE	**LSB-First Enable** — This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in bit 7. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Data is transferred most significant bit first. 1 Data is transferred least significant bit first.

Table 9-3. $\overline{SS}$ Input / Output Selection

MODFEN	SSOE	Master Mode	Slave Mode
0	0	$\overline{SS}$ not used by SPI	$\overline{SS}$ input
0	1	$\overline{SS}$ not used by SPI	$\overline{SS}$ input
1	0	$\overline{SS}$ input with MODF feature	$\overline{SS}$ input
1	1	$\overline{SS}$ is slave select output	$\overline{SS}$ input

9.3.2.2 SPI Control Register 2 (SPICR2)

	7	6	5	4	3	2	1	0
R	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-4. SPI Control Register 2 (SPICR2)

Read: anytime

Write: anytime; writes to the reserved bits have no effect

Table 9-4. SPICR2 Field Descriptions

Field	Description
4 MODFEN	**Mode Fault Enable Bit** — This bit allows the MODF failure being detected. If the SPI is in master mode and MODFEN is cleared, then the $\overline{SS}$ port pin is not used by the SPI. In slave mode, the $\overline{SS}$ is available only as an input regardless of the value of MODFEN. For an overview on the impact of the MODFEN bit on the $\overline{SS}$ port pin configuration refer to Table 9-3. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 $\overline{SS}$ port pin is not used by the SPI 1 $\overline{SS}$ port pin with MODF feature
3 BIDIROE	**Output Enable in the Bidirectional Mode of Operation** — This bit controls the MOSI and MISO output buffer of the SPI, when in bidirectional mode of operation (SPC0 is set). In master mode this bit controls the output buffer of the MOSI port, in slave mode it controls the output buffer of the MISO port. In master mode, with SPC0 set, a change of this bit will abort a transmission in progress and force the SPI into idle state. 0 Output buffer disabled 1 Output buffer enabled
1 SPISWAI	**SPI Stop in Wait Mode Bit** — This bit is used for power conservation while in wait mode. 0 SPI clock operates normally in wait mode 1 Stop SPI clock generation when in wait mode
0 SPC0	**Serial Pin Control Bit 0** — This bit enables bidirectional pin configurations as shown in Table 9-5. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state

Table 9-5. Bidirectional Pin Configurations

Pin Mode	SPC0	BIDIROE	MISO	MOSI
Master Mode of Operation				
Normal	0	X	Master In	Master Out
Bidirectional	1	0	MISO not used by SPI	Master In
		1		Master I/O
Slave Mode of Operation				
Normal	0	X	Slave Out	Slave In
Bidirectional	1	0	Slave In	MOSI not used by SPI
		1	Slave I/O	

9.3.2.3 SPI Baud Rate Register (SPIBR)

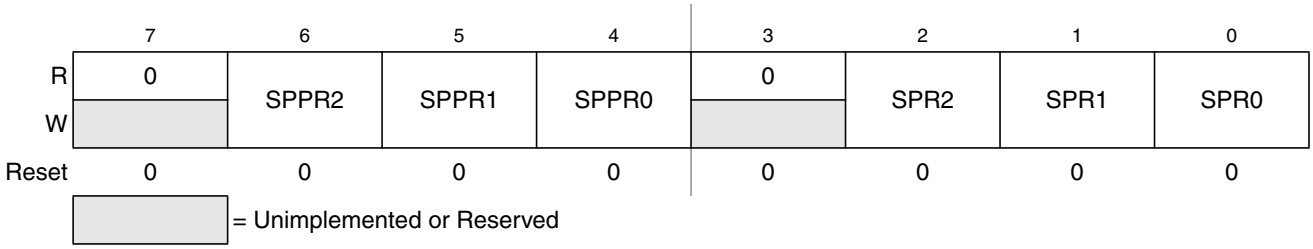


Figure 9-5. SPI Baud Rate Register (SPIBR)

Read: anytime

Write: anytime; writes to the reserved bits have no effect

Table 9-6. SPIBR Field Descriptions

Field	Description
6:4 SPPR[2:0]	**SPI Baud Rate Preselection Bits** — These bits specify the SPI baud rates as shown in Table 9-7. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.
2:0 SPR[2:0}	**SPI Baud Rate Selection Bits** — These bits specify the SPI baud rates as shown in Table 9-7. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.

The baud rate divisor equation is as follows:

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)}$$

The baud rate can be calculated with the following equation:

$$\text{Baud Rate} = \text{BusClock} / \text{BaudRateDivisor}$$

Table 9-7. Example SPI Baud Rate Selection (25 MHz Bus Clock)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
0	0	0	0	0	0	2	12.5 MHz
0	0	0	0	0	1	4	6.25 MHz
0	0	0	0	1	0	8	3.125 MHz
0	0	0	0	1	1	16	1.5625 MHz
0	0	0	1	0	0	32	781.25 kHz
0	0	0	1	0	1	64	390.63 kHz
0	0	0	1	1	0	128	195.31 kHz
0	0	0	1	1	1	256	97.66 kHz
0	0	1	0	0	0	4	6.25 MHz
0	0	1	0	0	1	8	3.125 MHz
0	0	1	0	1	0	16	1.5625 MHz
0	0	1	0	1	1	32	781.25 kHz
0	0	1	1	0	0	64	390.63 kHz
0	0	1	1	0	1	128	195.31 kHz
0	0	1	1	1	0	256	97.66 kHz
0	0	1	1	1	1	512	48.83 kHz
0	1	0	0	0	0	6	4.16667 MHz
0	1	0	0	0	1	12	2.08333 MHz
0	1	0	0	1	0	24	1.04167 MHz
0	1	0	0	1	1	48	520.83 kHz
0	1	0	1	0	0	96	260.42 kHz
0	1	0	1	0	1	192	130.21 kHz
0	1	0	1	1	0	384	65.10 kHz
0	1	0	1	1	1	768	32.55 kHz
0	1	1	0	0	0	8	3.125 MHz
0	1	1	0	0	1	16	1.5625 MHz
0	1	1	0	1	0	32	781.25 kHz
0	1	1	0	1	1	64	390.63 kHz
0	1	1	1	0	0	128	195.31 kHz
0	1	1	1	0	1	256	97.66 kHz
0	1	1	1	1	0	512	48.83 kHz
0	1	1	1	1	1	1024	24.41 kHz
1	0	0	0	0	0	10	2.5 MHz
1	0	0	0	0	1	20	1.25 MHz
1	0	0	0	1	0	40	625 kHz
1	0	0	0	1	1	80	312.5 kHz
1	0	0	1	0	0	160	156.25 kHz
1	0	0	1	0	1	320	78.13 kHz
1	0	0	1	1	0	640	39.06 kHz

Table 9-7. Example SPI Baud Rate Selection (25 MHz Bus Clock) (continued)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
1	0	0	1	1	1	1280	19.53 kHz
1	0	1	0	0	0	12	2.08333 MHz
1	0	1	0	0	1	24	1.04167 MHz
1	0	1	0	1	0	48	520.83 kHz
1	0	1	0	1	1	96	260.42 kHz
1	0	1	1	0	0	192	130.21 kHz
1	0	1	1	0	1	384	65.10 kHz
1	0	1	1	1	0	768	32.55 kHz
1	0	1	1	1	1	1536	16.28 kHz
1	1	0	0	0	0	14	1.78571 MHz
1	1	0	0	0	1	28	892.86 kHz
1	1	0	0	1	0	56	446.43 kHz
1	1	0	0	1	1	112	223.21 kHz
1	1	0	1	0	0	224	111.61 kHz
1	1	0	1	0	1	448	55.80 kHz
1	1	0	1	1	0	896	27.90 kHz
1	1	0	1	1	1	1792	13.95 kHz
1	1	1	0	0	0	16	1.5625 MHz
1	1	1	0	0	1	32	781.25 kHz
1	1	1	0	1	0	64	390.63 kHz
1	1	1	0	1	1	128	195.31 kHz
1	1	1	1	0	0	256	97.66 kHz
1	1	1	1	0	1	512	48.83 kHz
1	1	1	1	1	0	1024	24.41 kHz
1	1	1	1	1	1	2048	12.21 kHz

NOTE

In slave mode of SPI S-clock speed DIV2 is not supported.

9.3.2.4 SPI Status Register (SPISR)

	7	6	5	4	3	2	1	0
R	SPIF	0	SPTEF	MODF	0	0	0	0
W								
Reset	0	0	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-6. SPI Status Register (SPISR)

Read: anytime

Write: has no effect

Table 9-8. SPISR Field Descriptions

Field	Description
7 SPIF	**SPIF Interrupt Flag** — This bit is set after a received data byte has been transferred into the SPI Data Register. This bit is cleared by reading the SPISR register (with SPIF set) followed by a read access to the SPI Data Register. 0 Transfer not yet complete 1 New data copied to SPIDR
5 SPTEF	**SPI Transmit Empty Interrupt Flag** — If set, this bit indicates that the transmit data register is empty. To clear this bit and place data into the transmit data register, SPISR has to be read with SPTEF = 1, followed by a write to SPIDR. Any write to the SPI Data Register without reading SPTEF = 1, is effectively ignored. 0 SPI Data register not empty 1 SPI Data register empty
4 MODF	**Mode Fault Flag** — This bit is set if the $\overline{SS}$ input becomes low while the SPI is configured as a master and mode fault detection is enabled, MODFEN bit of SPICR2 register is set. Refer to MODFEN bit description in Section 9.3.2.2, “SPI Control Register 2 (SPICR2).” The flag is cleared automatically by a read of the SPI Status Register (with MODF set) followed by a write to the SPI Control Register 1. 0 Mode fault has not occurred. 1 Mode fault has occurred.

9.3.2.5 SPI Data Register (SPIDR)

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

= Unimplemented or Reserved

Figure 9-7. SPI Data Register (SPIDR)

Read: anytime; normally read only after SPIF is set

Write: anytime

The SPI Data Register is both the input and output register for SPI data. A write to this register allows a data byte to be queued and transmitted. For a SPI configured as a master, a queued data byte is transmitted immediately after the previous transmission has completed. The SPI Transmitter Empty Flag SPTEF in the SPISR register indicates when the SPI Data Register is ready to accept new data.

Reading the data can occur anytime from after the SPIF is set to before the end of the next transfer. If the SPIF is not serviced by the end of the successive transfers, those data bytes are lost and the data within the SPIDR retains the first byte until SPIF is serviced.

9.4 Functional Description

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or SPI operation can be interrupt driven.

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI Control Register 1. While SPE bit is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select ($\overline{SS}$)
- Serial clock (SCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

The main element of the SPI system is the SPI Data Register. The 8-bit data register in the master and the 8-bit data register in the slave are linked by the MOSI and MISO pins to form a distributed 16-bit register. When a data transfer operation is performed, this 16-bit register is serially shifted eight bit positions by the S-clock from the master, so data is exchanged between the master and the slave. Data written to the master SPI Data Register becomes the output data for the slave, and data read from the master SPI Data Register after a transfer operation is the input data from the slave.

A read of SPISR with SPTEF = 1 followed by a write to SPIDR puts data into the transmit data register. When a transfer is complete, received data is moved into the receive data register. Data may be read from this double-buffered system any time before the next transfer has completed. This 8-bit data register acts as the SPI receive data register for reads and as the SPI transmit data register for writes. A single SPI register address is used for reading data from the read data buffer and for writing data to the transmit data register.

The clock phase control bit (CPHA) and a clock polarity control bit (CPOL) in the SPI Control Register 1 (SPICR1) select one of four possible clock formats to be used by the SPI system. The CPOL bit simply selects a non-inverted or inverted clock. The CPHA bit is used to accommodate two fundamentally different protocols by sampling data on odd numbered SCK edges or on even numbered SCK edges (see Section 9.4.3, “Transmission Formats”).

The SPI can be configured to operate as a master or as a slave. When the MSTR bit in SPI Control Register1 is set, master mode is selected, when the MSTR bit is clear, slave mode is selected.

9.4.1 Master Mode

The SPI operates in master mode when the MSTR bit is set. Only a master SPI module can initiate transmissions. A transmission begins by writing to the master SPI Data Register. If the shift register is empty, the byte immediately transfers to the shift register. The byte begins shifting out on the MOSI pin under the control of the serial clock.

- S-clock

  The SPR2, SPR1, and SPR0 baud rate selection bits in conjunction with the SPPR2, SPPR1, and SPPR0 baud rate preselection bits in the SPI Baud Rate register control the baud rate generator and determine the speed of the transmission. The SCK pin is the SPI clock output. Through the SCK pin, the baud rate generator of the master controls the shift register of the slave peripheral.
- MOSI and MISO Pins

  In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and BIDIROE control bits.
- $\overline{SS}$ Pin

  If MODFEN and SSOE bit are set, the $\overline{SS}$ pin is configured as slave select output. The $\overline{SS}$ output becomes low during each transmission and is high when the SPI is in idle state.

  If MODFEN is set and SSOE is cleared, the $\overline{SS}$ pin is configured as input for detecting mode fault error. If the $\overline{SS}$ input becomes low this indicates a mode fault error where another master tries to drive the MOSI and SCK lines. In this case, the SPI immediately switches to slave mode, by clearing the MSTR bit and also disables the slave output buffer MISO (or SISO in bidirectional mode). So the result is that all outputs are disabled and SCK, MOSI and MISO are inputs. If a transmission is in progress when the mode fault occurs, the transmission is aborted and the SPI is forced into idle state.

This mode fault error also sets the mode fault (MODF) flag in the SPI Status Register (SPISR). If the SPI interrupt enable bit (SPIE) is set when the MODF flag gets set, then an SPI interrupt sequence is also requested.

When a write to the SPI Data Register in the master occurs, there is a half SCK-cycle delay. After the delay, SCK is started within the master. The rest of the transfer operation differs slightly, depending on the clock format specified by the SPI clock phase bit, CPHA, in SPI Control Register 1 (see Section 9.4.3, “Transmission Formats”).

NOTE

A change of the bits CPOL, CPHA, SSOE, LSBFE, MODFEN, SPC0, BIDIROE with SPC0 set, SPPR2–SPPR0 and SPR2–SPR0 in master mode will abort a transmission in progress and force the SPI into idle state. The remote slave cannot detect this, therefore the master has to ensure that the remote slave is set back to idle state.

9.4.2 Slave Mode

The SPI operates in slave mode when the MSTR bit in SPI Control Register1 is clear.

- SCK Clock

  In slave mode, SCK is the SPI clock input from the master.

- MISO and MOSI Pins

  In slave mode, the function of the serial data output pin (MISO) and serial data input pin (MOSI) is determined by the SPC0 bit and BIDIROE bit in SPI Control Register 2.

- $\overline{SS}$ Pin

  The $\overline{SS}$ pin is the slave select input. Before a data transmission occurs, the $\overline{SS}$ pin of the slave SPI must be low. $\overline{SS}$ must remain low until the transmission is complete. If $\overline{SS}$ goes high, the SPI is forced into idle state.

  The $\overline{SS}$ input also controls the serial data output pin, if $\overline{SS}$ is high (not selected), the serial data output pin is high impedance, and, if $\overline{SS}$ is low the first bit in the SPI Data Register is driven out of the serial data output pin. Also, if the slave is not selected ($\overline{SS}$ is high), then the SCK input is ignored and no internal shifting of the SPI shift register takes place.

Although the SPI is capable of duplex operation, some SPI peripherals are capable of only receiving SPI data in a slave mode. For these simpler devices, there is no serial data out pin.

NOTE

When peripherals with duplex capability are used, take care not to simultaneously enable two receivers whose serial outputs drive the same system slave's serial data output line.

As long as no more than one slave device drives the system slave's serial data output line, it is possible for several slaves to receive the same transmission from a master, although the master would not receive return information from all of the receiving slaves.

If the CPHA bit in SPI Control Register 1 is clear, odd numbered edges on the SCK input cause the data at the serial data input pin to be latched. Even numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

If the CPHA bit is set, even numbered edges on the SCK input cause the data at the serial data input pin to be latched. Odd numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

When CPHA is set, the first edge is used to get the first data bit onto the serial data output pin. When CPHA is clear and the $\overline{SS}$ input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the eighth shift, the transfer is considered complete and the received data is transferred into the SPI Data Register. To indicate transfer is complete, the SPIF flag in the SPI Status Register is set.

NOTE

A change of the bits CPOL, CPHA, SSOE, LSBFE, MODFEN, SPC0 and BIDIROE with SPC0 set in slave mode will corrupt a transmission in progress and has to be avoided.

9.4.3 Transmission Formats

During an SPI transmission, data is transmitted (shifted out serially) and received (shifted in serially) simultaneously. The serial clock (SCK) synchronizes shifting and sampling of the information 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. Optionally, on a master SPI device, the slave select line can be used to indicate multiple-master bus contention.

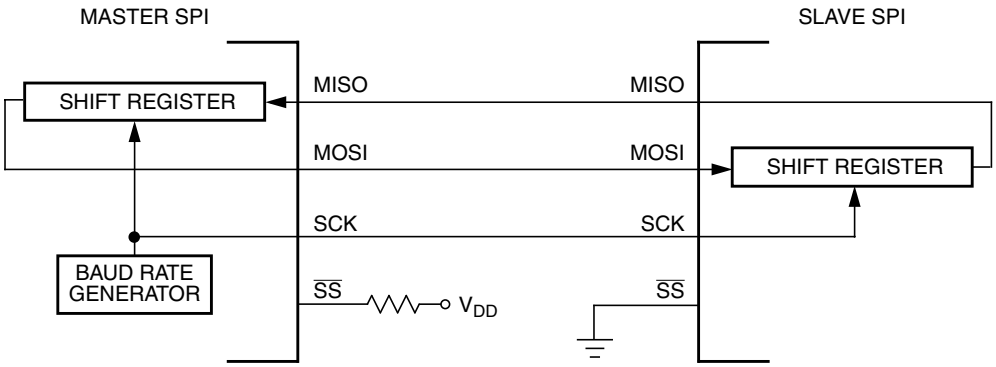


Figure 9-8. Master/Slave Transfer Block Diagram

9.4.3.1 Clock Phase and Polarity Controls

Using two bits in the SPI Control Register1, software selects one of four combinations of serial clock phase and polarity.

The CPOL clock polarity control bit specifies an active high or low clock and has no significant effect on the transmission format.

The CPHA clock phase control bit selects one of two fundamentally different transmission formats.

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.

9.4.3.2 CPHA = 0 Transfer Format

The first edge on the SCK line is used to clock the first data bit of the slave into the master and the first data bit of the master into the slave. In some peripherals, the first bit of the slave's data is available at the slave's data out pin as soon as the slave is selected. In this format, the first SCK edge is issued a half cycle after $\overline{SS}$ has become low.

A half SCK cycle later, the second edge appears on the SCK line. When this second edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the shift register, depending on LSBFE bit.

After this second edge, the next bit of the SPI master data is transmitted out of the serial data output pin of the master to the serial input pin on the slave. This process continues for a total of 16 edges on the SCK line, with data being latched on odd numbered edges and shifted on even numbered edges.

Data reception is double buffered. Data is shifted serially into the SPI shift register during the transfer and is transferred to the parallel SPI Data Register after the last bit is shifted in.

After the 16th (last) SCK edge:

- Data that was previously in the master SPI Data Register should now be in the slave data register and the data that was in the slave data register should be in the master.
- The SPIF flag in the SPI Status Register is set indicating that the transfer is complete.

Figure 9-9 is a timing diagram of an SPI transfer where CPHA = 0. SCK waveforms are shown for CPOL = 0 and CPOL = 1. The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly 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 $\overline{SS}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

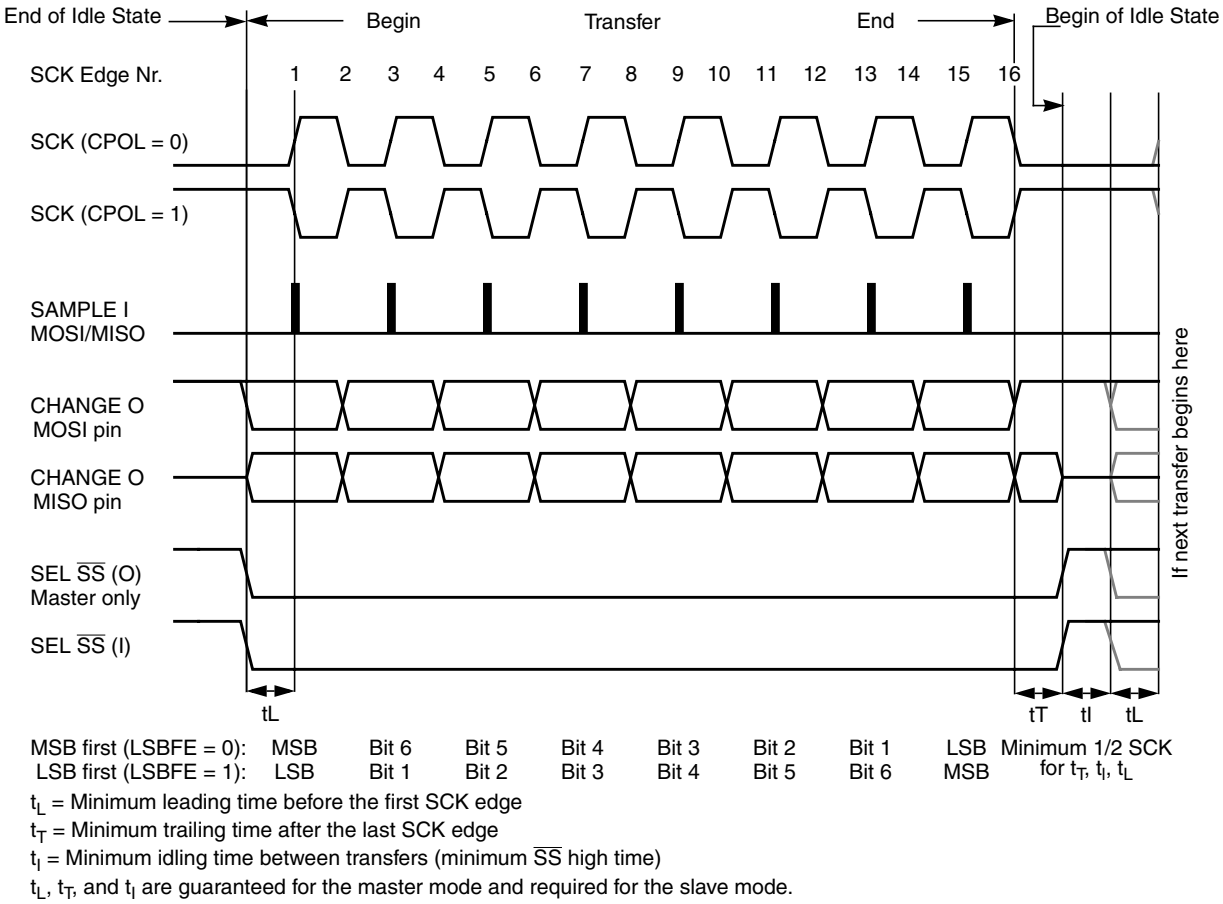


Figure 9-9. SPI Clock Format 0 (CPHA = 0)

In slave mode, if the $\overline{SS}$ line is not deasserted between the successive transmissions then the content of the SPI Data Register is not transmitted, instead the last received byte is transmitted. If the $\overline{SS}$ line is deasserted for at least minimum idle time (half SCK cycle) between successive transmissions then the content of the SPI Data Register is transmitted.

In master mode, with slave select output enabled the $\overline{SS}$ line is always deasserted and reasserted between successive transfers for at least minimum idle time.

9.4.3.3 CPHA = 1 Transfer Format

Some peripherals require the first SCK edge before the first data bit becomes available at the data out pin, the second edge clocks data into the system. In this format, the first SCK edge is issued by setting the CPHA bit at the beginning of the 8-cycle transfer operation.

The first edge of SCK occurs immediately after the half SCK clock cycle synchronization delay. This first edge commands the slave to transfer its first data bit to the serial data input pin of the master.

A half SCK cycle later, the second edge appears on the SCK pin. This is the latching edge for both the master and slave.

When the third edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the SPI shift register, depending on LSBFE bit. After this edge, the next bit of the master data is coupled out of the serial data output pin of the master to the serial input pin on the slave.

This process continues for a total of 16 edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

Data reception is double buffered, data is serially shifted into the SPI shift register during the transfer and is transferred to the parallel SPI Data Register after the last bit is shifted in.

After the 16th SCK edge:

- Data that was previously in the SPI Data Register of the master is now in the data register of the slave, and data that was in the data register of the slave is in the master.
- The SPIF flag bit in SPISR is set indicating that the transfer is complete.

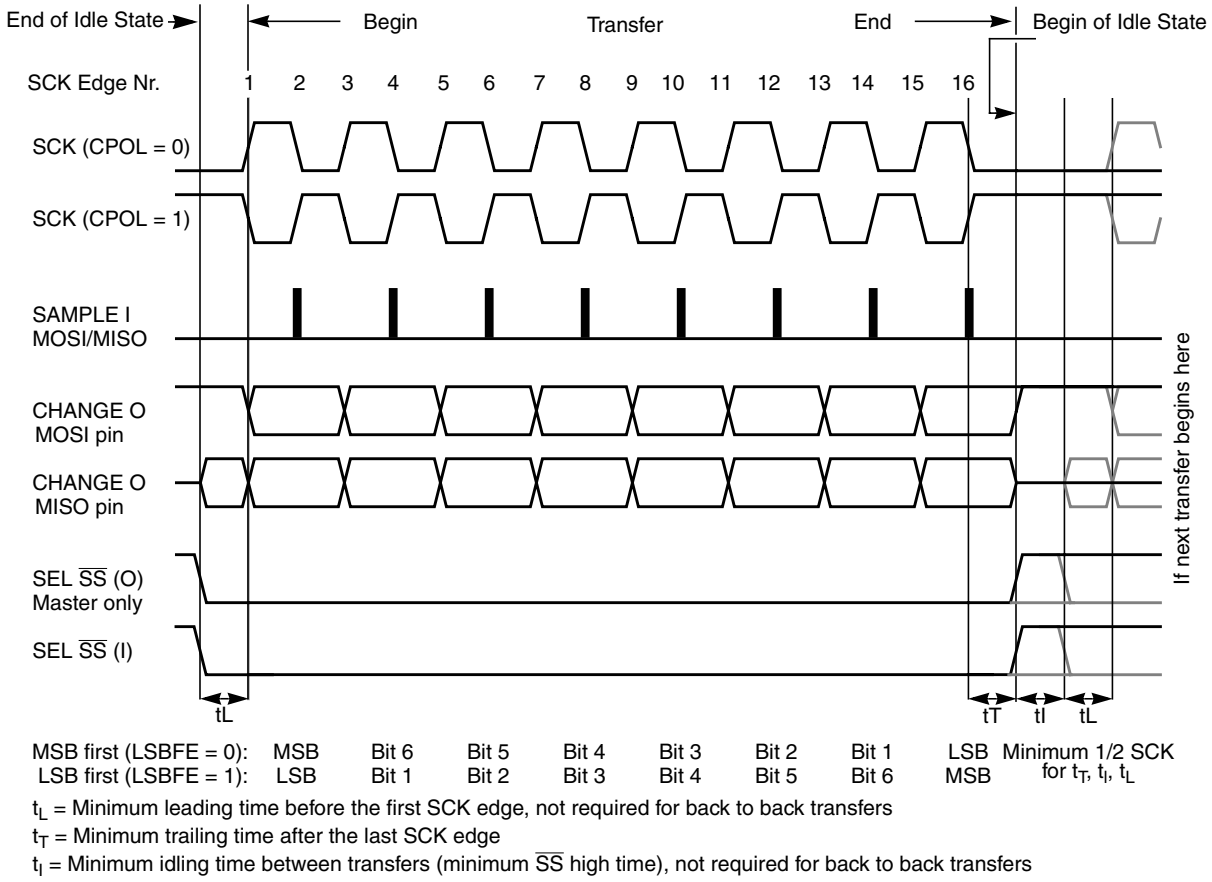
Figure 9-10 shows two clocking variations for CPHA = 1. The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{SS}$ line is the slave select input to the slave. The $\overline{SS}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

The $\overline{SS}$ line can remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave that drive the MISO data line.

- Back-to-back transfers in master mode

In master mode, if a transmission has completed and a new data byte is available in the SPI Data Register, this byte is send out immediately without a trailing and minimum idle time.

The SPI interrupt request flag (SPIF) is common to both the master and slave modes. SPIF gets set one half SCK cycle after the last SCK edge.



t_L = Minimum leading time before the first SCK edge, not required for back to back transfers
t_T = Minimum trailing time after the last SCK edge
t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time), not required for back to back transfers

Figure 9-10. SPI Clock Format 1 (CPHA = 1)

9.4.4 SPI Baud Rate Generation

Baud rate generation consists of a series of divider stages. Six bits in the SPI Baud Rate register (SPPR2, SPPR1, SPPR0, SPR2, SPR1, and SPR0) determine the divisor to the SPI module clock which results in the SPI baud rate.

The SPI clock rate is determined by the product of the value in the baud rate preselection bits (SPPR2–SPPR0) and the value in the baud rate selection bits (SPR2–SPR0). The module clock divisor equation is shown in Figure 9-11

When all bits are clear (the default condition), the SPI module clock is divided by 2. When the selection bits (SPR2–SPR0) are 001 and the preselection bits (SPPR2–SPPR0) are 000, the module clock divisor becomes 4. When the selection bits are 010, the module clock divisor becomes 8 etc.

When the preselection bits are 001, the divisor determined by the selection bits is multiplied by 2. When the preselection bits are 010, the divisor is multiplied by 3, etc. See Table 9-7 for baud rate calculations for all bit conditions, based on a 25-MHz bus clock. The two sets of selects allows the clock to be divided by a non-power of two to achieve other baud rates such as divide by 6, divide by 10, etc.

The baud rate generator is activated only when the SPI is in the master mode and a serial transfer is taking place. In the other cases, the divider is disabled to decrease I_{DD} current.

$$BaudRateDivisor = (SPPR + 1) \bullet 2^{(SPR + 1)}$$

Figure 9-11. Baud Rate Divisor Equation

9.4.5 Special Features

9.4.5.1 $\overline{SS}$ Output

The $\overline{SS}$ output feature automatically drives the $\overline{SS}$ pin low during transmission to select external devices and drives it high during idle to deselect external devices. When $\overline{SS}$ output is selected, the $\overline{SS}$ output pin is connected to the $\overline{SS}$ input pin of the external device.

The $\overline{SS}$ output is available only in master mode during normal SPI operation by asserting SSOE and MODFEN bit as shown in Table 9-3.

The mode fault feature is disabled while $\overline{SS}$ output is enabled.

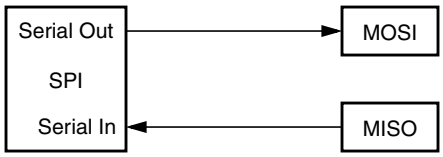
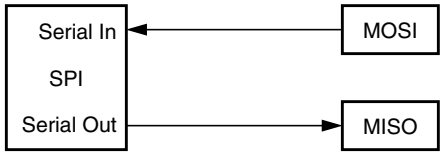
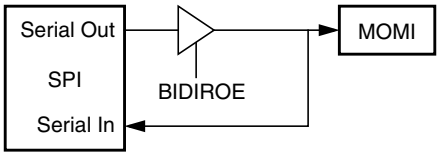
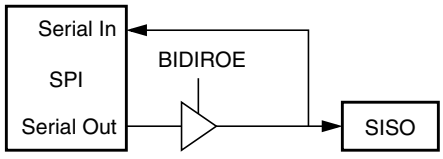
NOTE

Care must be taken when using the $\overline{SS}$ output feature in a multimaster system because the mode fault feature is not available for detecting system errors between masters.

9.4.5.2 Bidirectional Mode (MOSI or MISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI Control Register 2 (see Table 9-9). In this mode, the SPI uses only one serial data pin for the interface with external device(s). The MSTR bit decides which pin to use. The MOSI pin becomes the serial data I/O (MOMI) pin for the master mode, and the MISO pin becomes serial data I/O (SISO) pin for the slave mode. The MISO pin in master mode and MOSI pin in slave mode are not used by the SPI.

Table 9-9. Normal Mode and Bidirectional Mode

When SPE = 1	Master Mode MSTR = 1	Slave Mode MSTR = 0
Normal Mode SPC0 = 0		
Bidirectional Mode SPC0 = 1		

The direction of each serial I/O pin depends on the BIDIROE bit. If the pin is configured as an output, serial data from the shift register is driven out on the pin. The same pin is also the serial input to the shift register.

The SCK is output for the master mode and input for the slave mode.

The $\overline{SS}$ is the input or output for the master mode, and it is always the input for the slave mode.

The bidirectional mode does not affect SCK and $\overline{SS}$ functions.

NOTE

In bidirectional master mode, with mode fault enabled, both data pins MISO and MOSI can be occupied by the SPI, though MOSI is normally used for transmissions in bidirectional mode and MISO is not used by the SPI. If a mode fault occurs, the SPI is automatically switched to slave mode, in this case MISO becomes occupied by the SPI and MOSI is not used. This has to be considered, if the MISO pin is used for other purpose.

9.4.6 Error Conditions

The SPI has one error condition:

- Mode fault error

9.4.6.1 Mode Fault Error

If the $\overline{SS}$ input becomes low while the SPI is configured as a master, it indicates a system error where more than one master may be trying to drive the MOSI and SCK lines simultaneously. This condition is not permitted in normal operation, the MODF bit in the SPI Status Register is set automatically provided the MODFEN bit is set.

In the special case where the SPI is in master mode and MODFEN bit is cleared, the $\overline{SS}$ pin is not used by the SPI. In this special case, the mode fault error function is inhibited and MODF remains cleared. In case the SPI system is configured as a slave, the $\overline{SS}$ pin is a dedicated input pin. Mode fault error doesn’t occur in slave mode.

If a mode fault error occurs the SPI is switched to slave mode, with the exception that the slave output buffer is disabled. So SCK, MISO and MOSI pins are forced to be high impedance inputs to avoid any possibility of conflict with another output driver. A transmission in progress is aborted and the SPI is forced into idle state.

If the mode fault error occurs in the bidirectional mode for a SPI system configured in master mode, output enable of the MOMI (MOSI in bidirectional mode) is cleared if it was set. No mode fault error occurs in the bidirectional mode for SPI system configured in slave mode.

The mode fault flag is cleared automatically by a read of the SPI Status Register (with MODF set) followed by a write to SPI Control Register 1. If the mode fault flag is cleared, the SPI becomes a normal master or slave again.

9.4.7 Operation in Run Mode

In run mode with the SPI system enable (SPE) bit in the SPI control register clear, the SPI system is in a low-power, disabled state. SPI registers remain accessible, but clocks to the core of this module are disabled.

9.4.8 Operation in Wait Mode

SPI operation in wait mode depends upon the state of the SPISWAI bit in SPI Control Register 2.

- If SPISWAI is clear, the SPI operates normally when the CPU is in wait mode
- If SPISWAI is set, SPI clock generation ceases and the SPI module enters a power conservation state when the CPU is in wait mode.
  - If SPISWAI is set and the SPI is configured for master, any transmission and reception in progress stops at wait mode entry. The transmission and reception resumes when the SPI exits wait mode.
  - If SPISWAI is set and the SPI is configured as a slave, any transmission and reception in progress continues if the SCK continues to be driven from the master. This keeps the slave synchronized to the master and the SCK.

    If the master transmits several bytes while the slave is in wait mode, the slave will continue to send out bytes consistent with the operation mode at the start of wait mode (i.e. If the slave is currently sending its SPIDR to the master, it will continue to send the same byte. Else if the slave is currently sending the last received byte from the master, it will continue to send each previous master byte).

NOTE

Care must be taken when expecting data from a master while the slave is in wait or stop mode. Even though the shift register will continue to operate, the rest of the SPI is shut down (i.e. a SPIF interrupt will **not** be generated until exiting stop or wait mode). Also, the byte from the shift register will not be copied into the SPIDR register until after the slave SPI has exited wait or stop mode. A SPIF flag and SPIDR copy is only generated if wait mode is entered or exited during a tranmission. If the slave enters wait mode in idle mode and exits wait mode in idle mode, neither a SPIF nor a SPIDR copy will occur.

9.4.9 Operation in Stop Mode

Stop mode is dependent on the system. The SPI enters stop mode when the module clock is disabled (held high or low). If the SPI is in master mode and exchanging data when the CPU enters stop mode, the transmission is frozen until the CPU exits stop mode. After stop, data to and from the external SPI is exchanged correctly. In slave mode, the SPI will stay synchronized with the master.

The stop mode is not dependent on the SPISWAI bit.

9.5 Reset

The reset values of registers and signals are described in the Memory Map and Registers section (see Section 9.3, “Memory Map and Register Definition”) which details the registers and their bit-fields.

- If a data transmission occurs in slave mode after reset without a write to SPIDR, it will transmit garbage, or the byte last received from the master before the reset.
- Reading from the SPIDR after reset will always read a byte of zeros.

9.6 Interrupts

The SPIV3 only originates interrupt requests when SPI is enabled (SPE bit in SPICR1 set). The following is a description of how the SPIV3 makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt priority are chip dependent.

The interrupt flags MODF, SPIF and SPTEF are logically ORed to generate an interrupt request.

9.6.1 MODF

MODF occurs when the master detects an error on the $\overline{SS}$ pin. The master SPI must be configured for the MODF feature (see Table 9-3). After MODF is set, the current transfer is aborted and the following bit is changed:

- MSTR = 0, The master bit in SPICR1 resets.

The MODF interrupt is reflected in the status register MODF flag. Clearing the flag will also clear the interrupt. This interrupt will stay active while the MODF flag is set. MODF has an automatic clearing process which is described in Section 9.3.2.4, “SPI Status Register (SPISR).”

9.6.2 SPIF

SPIF occurs when new data has been received and copied to the SPI Data Register. After SPIF is set, it does not clear until it is serviced. SPIF has an automatic clearing process which is described in Section 9.3.2.4, “SPI Status Register (SPISR).” In the event that the SPIF is not serviced before the end of the next transfer (i.e. SPIF remains active throughout another transfer), the latter transfers will be ignored and no new data will be copied into the SPIDR.

9.6.3 SPTEF

SPTEF occurs when the SPI Data Register is ready to accept new data. After SPTEF is set, it does not clear until it is serviced. SPTEF has an automatic clearing process which is described in Section 9.3.2.4, “SPI Status Register (SPISR).”

Chapter 10
Inter-Integrated Circuit (IICV2)

10.1 Introduction

The inter-IC bus (IIC) is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange between devices. Being a two-wire device, the IIC bus minimizes the need for large numbers of connections between devices, and eliminates the need for an address decoder.

This bus is suitable for applications requiring occasional communications over a short distance between a number of devices. It also provides flexibility, allowing additional devices to be connected to the bus for further expansion and system development.

The interface is designed to operate up to 100 kbps with maximum bus loading and timing. The device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF.

10.1.1 Features

The IIC module has the following key features:

- Compatible with I2C bus standard
- Multi-master operation
- Software programmable for one of 256 different serial clock frequencies
- Software selectable acknowledge bit
- Interrupt driven byte-by-byte data transfer
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation
- Acknowledge bit generation/detection
- Bus busy detection

10.1.2 Modes of Operation

The IIC functions the same in normal, special, and emulation modes. It has two low power modes: wait and stop modes.

10.1.3 Block Diagram

The block diagram of the IIC module is shown in Figure 10-1.

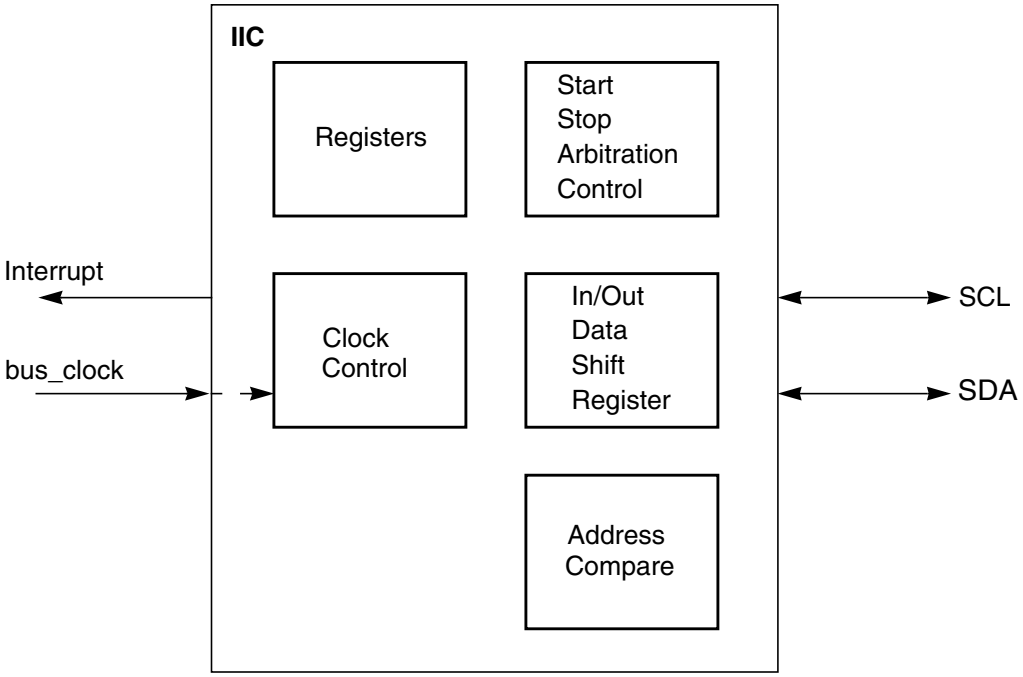


Figure 10-1. IIC Block Diagram

10.2 External Signal Description

The IICV2 module has two external pins.

10.2.1 IIC_SCL — Serial Clock Line Pin

This is the bidirectional serial clock line (SCL) of the module, compatible to the IIC bus specification.

10.2.2 IIC_SDA — Serial Data Line Pin

This is the bidirectional serial data line (SDA) of the module, compatible to the IIC bus specification.

10.3 Memory Map and Register Definition

This section provides a detailed description of all memory and registers for the IIC module.

10.3.1 Module Memory Map

The memory map for the IIC module is given below in Figure 10-2. The address listed for each register is the address offset.The total address for each register is the sum of the base address for the IIC module and the address offset for each register.

10.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
IBAD	R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
	W								
IBFD	R	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
	W								
IBCR	R	IBEN	IBIE	MS/$\overline{SL}$	Tx/$\overline{Rx}$	TXAK	0	0	IBSWAI
	W						RSTA		
IBSR	R	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
	W								
IBDR	R	D7	D6	D5	D4	D3	D2	D1	D0
	W								

= Unimplemented or Reserved

Figure 10-2. IIC Register Summary

10.3.2.1 IIC Address Register (IBAD)

	7	6	5	4	3	2	1	0
R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-3. IIC Bus Address Register (IBAD)

Read and write anytime

This register contains the address the IIC bus will respond to when addressed as a slave; note that it is not the address sent on the bus during the address transfer.

Table 10-1. IBAD Field Descriptions

Field	Description
7:1 ADR[7:1]	**Slave Address** — Bit 1 to bit 7 contain the specific slave address to be used by the IIC bus module.The default mode of IIC bus is slave mode for an address match on the bus.
0 Reserved	Reserved — Bit 0 of the IBAD is reserved for future compatibility. This bit will always read 0.

10.3.2.2 IIC Frequency Divider Register (IBFD)

	7	6	5	4	3	2	1	0
R W	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-4. IIC Bus Frequency Divider Register (IBFD)

Read and write anytime

Table 10-2. IBFD Field Descriptions

Field	Description
7:0 IBC[7:0]	**I Bus Clock Rate 7:0** — This field is used to prescale the clock for bit rate selection. The bit clock generator is implemented as a prescale divider — IBC7:6, prescaled shift register — IBC5:3 select the prescaler divider and IBC2-0 select the shift register tap point. The IBC bits are decoded to give the tap and prescale values as shown in Table 10-3.

Table 10-3. I-Bus Tap and Prescale Values

IBC2-0 (bin)	SCL Tap (clocks)	SDA Tap (clocks)
000	5	1
001	6	1
010	7	2
011	8	2
100	9	3
101	10	3
110	12	4
111	15	4

IBC5-3 (bin)	scl2start (clocks)	scl2stop (clocks)	scl2tap (clocks)	tap2tap (clocks)
000	2	7	4	1
001	2	7	4	2
010	2	9	6	4
011	6	9	6	8
100	14	17	14	16
101	30	33	30	32
110	62	65	62	64
111	126	129	126	128

Table 10-4. Multiplier Factor

IBC7-6	MUL
00	01
01	02
10	04
11	RESERVED

The number of clocks from the falling edge of SCL to the first tap (Tap[1]) is defined by the values shown in the scl2tap column of Table 10-3, all subsequent tap points are separated by 2^{IBC5-3} as shown in the tap2tap column in Table 10-3. The SCL Tap is used to generated the SCL period and the SDA Tap is used to determine the delay from the falling edge of SCL to SDA changing, the SDA hold time.

IBC7–6 defines the multiplier factor MUL. The values of MUL are shown in the Table 10-4.

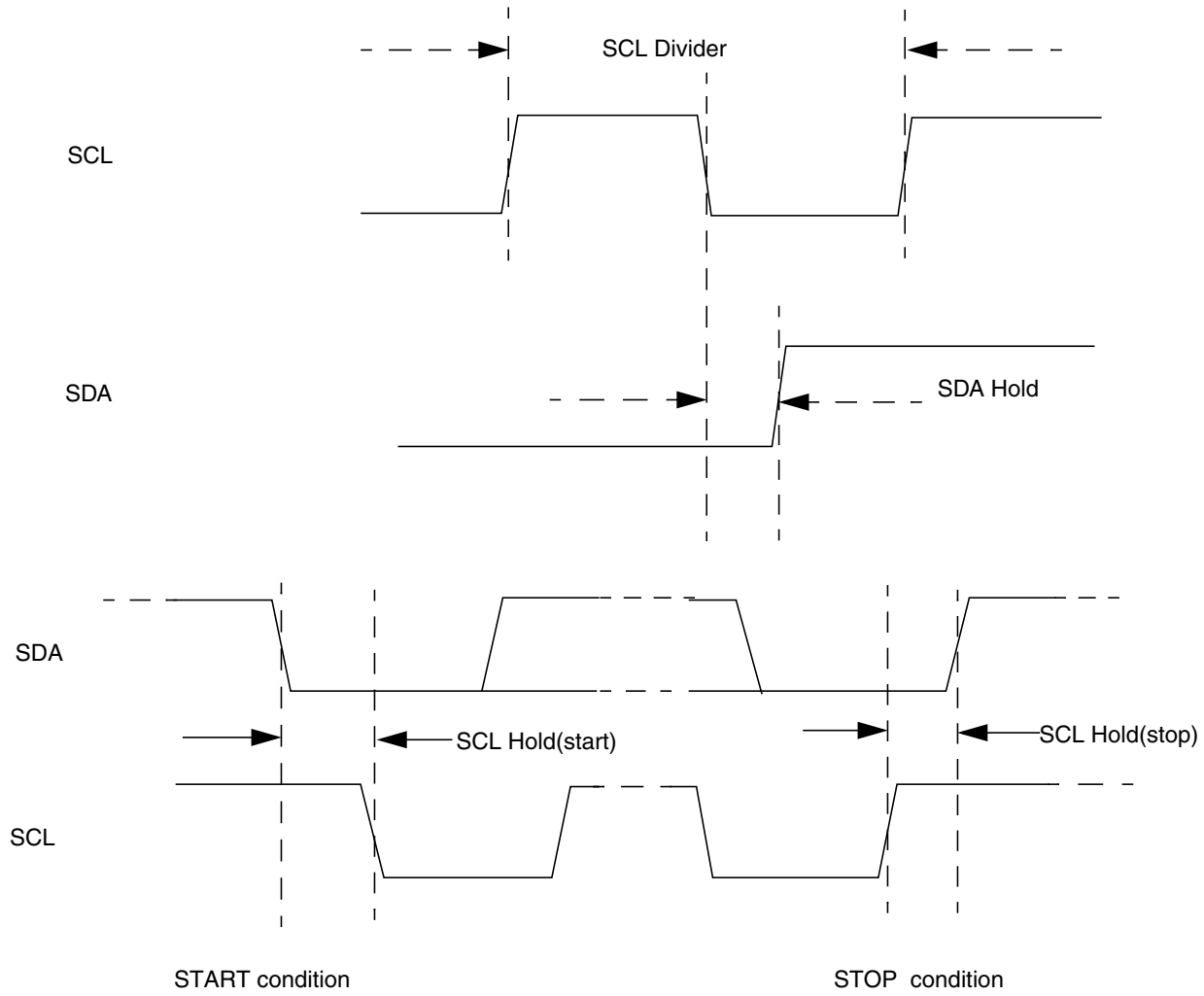


Figure 10-5. SCL Divider and SDA Hold

The equation used to generate the divider values from the IBFD bits is:

SCL Divider = MUL x {2 x (scl2tap + [(SCL_Tap -1) x tap2tap] + 2)}

The SDA hold delay is equal to the CPU clock period multiplied by the SDA Hold value shown in Table 10-5. The equation used to generate the SDA Hold value from the IBFD bits is:

SDA Hold = MUL x {scl2tap + [(SDA_Tap - 1) x tap2tap] + 3}

The equation for SCL Hold values to generate the start and stop conditions from the IBFD bits is:

SCL Hold(start) = MUL x [scl2start + (SCL_Tap - 1) x tap2tap]

SCL Hold(stop) = MUL x [scl2stop + (SCL_Tap - 1) x tap2tap]

Table 10-5. IIC Divider and Hold Values (Sheet 1 of 5)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
MUL=1				
00	20	7	6	11
01	22	7	7	12
02	24	8	8	13
03	26	8	9	14
04	28	9	10	15
05	30	9	11	16
06	34	10	13	18
07	40	10	16	21
08	28	7	10	15
09	32	7	12	17
0A	36	9	14	19
0B	40	9	16	21
0C	44	11	18	23
0D	48	11	20	25
0E	56	13	24	29
0F	68	13	30	35
10	48	9	18	25
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97
1F	240	33	118	121
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113

Table 10-5. IIC Divider and Hold Values (Sheet 2 of 5)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921
MUL=2				
40	40	14	12	22
41	44	14	14	24
42	48	16	16	26
43	52	16	18	28
44	56	18	20	30
45	60	18	22	32
46	68	20	26	36
47	80	20	32	42
48	56	14	20	30
49	64	14	24	34
4A	72	18	28	38
4B	80	18	32	42
4C	88	22	36	46
4D	96	22	40	50
4E	112	26	48	58

Table 10-5. IIC Divider and Hold Values (Sheet 3 of 5)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
4F	136	26	60	70
50	96	18	36	50
51	112	18	44	58
52	128	26	52	66
53	144	26	60	74
54	160	34	68	82
55	176	34	76	90
56	208	42	92	106
57	256	42	116	130
58	160	18	76	82
59	192	18	92	98
5A	224	34	108	114
5B	256	34	124	130
5C	288	50	140	146
5D	320	50	156	162
5E	384	66	188	194
5F	480	66	236	242
60	320	34	156	162
61	384	34	188	194
62	448	66	220	226
63	512	66	252	258
64	576	98	284	290
65	640	98	316	322
66	768	130	380	386
67	960	130	476	482
68	640	66	316	322
69	768	66	380	386
6A	896	130	444	450
6B	1024	130	508	514
6C	1152	194	572	578
6D	1280	194	636	642
6E	1536	258	764	770
6F	1920	258	956	962
70	1280	130	636	642
71	1536	130	764	770
72	1792	258	892	898
73	2048	258	1020	1026
74	2304	386	1148	1154
75	2560	386	1276	1282
76	3072	514	1532	1538
77	3840	514	1916	1922
78	2560	258	1276	1282
79	3072	258	1532	1538
7A	3584	514	1788	1794
7B	4096	514	2044	2050

Table 10-5. IIC Divider and Hold Values (Sheet 4 of 5)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
7C	4608	770	2300	2306
7D	5120	770	2556	2562
7E	6144	1026	3068	3074
7F	7680	1026	3836	3842
MUL=4				
80	80	28	24	44
81	88	28	28	48
82	96	32	32	52
83	104	32	36	56
84	112	36	40	60
85	120	36	44	64
86	136	40	52	72
87	160	40	64	84
88	112	28	40	60
89	128	28	48	68
8A	144	36	56	76
8B	160	36	64	84
8C	176	44	72	92
8D	192	44	80	100
8E	224	52	96	116
8F	272	52	120	140
90	192	36	72	100
91	224	36	88	116
92	256	52	104	132
93	288	52	120	148
94	320	68	136	164
95	352	68	152	180
96	416	84	184	212
97	512	84	232	260
98	320	36	152	164
99	384	36	184	196
9A	448	68	216	228
9B	512	68	248	260
9C	576	100	280	292
9D	640	100	312	324
9E	768	132	376	388
9F	960	132	472	484
A0	640	68	312	324
A1	768	68	376	388
A2	896	132	440	452
A3	1024	132	504	516
A4	1152	196	568	580
A5	1280	196	632	644
A6	1536	260	760	772
A7	1920	260	952	964

Table 10-5. IIC Divider and Hold Values (Sheet 5 of 5)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
A8	1280	132	632	644
A9	1536	132	760	772
AA	1792	260	888	900
AB	2048	260	1016	1028
AC	2304	388	1144	1156
AD	2560	388	1272	1284
AE	3072	516	1528	1540
AF	3840	516	1912	1924
B0	2560	260	1272	1284
B1	3072	260	1528	1540
B2	3584	516	1784	1796
B3	4096	516	2040	2052
B4	4608	772	2296	2308
B5	5120	772	2552	2564
B6	6144	1028	3064	3076
B7	7680	1028	3832	3844
B8	5120	516	2552	2564
B9	6144	516	3064	3076
BA	7168	1028	3576	3588
BB	8192	1028	4088	4100
BC	9216	1540	4600	4612
BD	10240	1540	5112	5124
BE	12288	2052	6136	6148
BF	15360	2052	7672	7684

10.3.2.3 IIC Control Register (IBCR)

	7	6	5	4	3	2	1	0
R	IBEN	IBIE	MS/SL	Tx/Rx	TXAK	0	0	IBSWAI
W						RSTA		
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-6. IIC Bus Control Register (IBCR)

Read and write anytime

Table 10-6. IBCR Field Descriptions

Field	Description
7 IBEN	**I-Bus Enable** — This bit controls the software reset of the entire IIC bus module. 0 The module is reset and disabled. This is the power-on reset situation. When low the interface is held in reset but registers can be accessed 1 The IIC bus module is enabled.This bit must be set before any other IBCR bits have any effect If the IIC bus module is enabled in the middle of a byte transfer the interface behaves as follows: slave mode ignores the current transfer on the bus and starts operating whenever a subsequent start condition is detected. Master mode will not be aware that the bus is busy, hence if a start cycle is initiated then the current bus cycle may become corrupt. This would ultimately result in either the current bus master or the IIC bus module losing arbitration, after which bus operation would return to normal.
6 IBIE	**I-Bus Interrupt Enable** 0 Interrupts from the IIC bus module are disabled. Note that this does not clear any currently pending interrupt condition 1 Interrupts from the IIC bus module are enabled. An IIC bus interrupt occurs provided the IBIF bit in the status register is also set.
5 MS/SL	**Master/Slave Mode Select Bit** — Upon reset, this bit is cleared. When this bit is changed from 0 to 1, a START signal is generated on the bus, and the master mode is selected. When this bit is changed from 1 to 0, a STOP signal is generated and the operation mode changes from master to slave.A STOP signal should only be generated if the IBIF flag is set. MS/$\overline{SL}$ is cleared without generating a STOP signal when the master loses arbitration. 0 Slave Mode 1 Master Mode
4 Tx/Rx	**Transmit/Receive Mode Select Bit** — This bit selects the direction of master and slave transfers. When addressed as a slave this bit should be set by software according to the SRW bit in the status register. In master mode this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit will always be high. 0 Receive 1 Transmit
3 TXAK	**Transmit Acknowledge Enable** — This bit specifies the value driven onto SDA during data acknowledge cycles for both master and slave receivers. The IIC module will always acknowledge address matches, provided it is enabled, regardless of the value of TXAK. Note that values written to this bit are only used when the IIC bus is a receiver, not a transmitter. 0 An acknowledge signal will be sent out to the bus at the 9th clock bit after receiving one byte data 1 No acknowledge signal response is sent (i.e., acknowledge bit = 1)
2 RSTA	**Repeat Start** — Writing a 1 to this bit will generate a repeated START condition on the bus, provided it is the current bus master. This bit will always be read as a low. Attempting a repeated start at the wrong time, if the bus is owned by another master, will result in loss of arbitration. 1 Generate repeat start cycle
1 RESERVED	**Reserved** — Bit 1 of the IBCR is reserved for future compatibility. This bit will always read 0.
0 IBSWAI	**I Bus Interface Stop in Wait Mode** 0 IIC bus module clock operates normally 1 Halt IIC bus module clock generation in wait mode

Wait mode is entered via execution of a CPU WAI instruction. In the event that the IBSWAI bit is set, all clocks internal to the IIC will be stopped and any transmission currently in progress will halt.If the CPU were woken up by a source other than the IIC module, then clocks would restart and the IIC would resume

from where was during the previous transmission. It is not possible for the IIC to wake up the CPU when its internal clocks are stopped.

If it were the case that the IBSWAI bit was cleared when the WAI instruction was executed, the IIC internal clocks and interface would remain alive, continuing the operation which was currently underway. It is also possible to configure the IIC such that it will wake up the CPU via an interrupt at the conclusion of the current operation. See the discussion on the IBIF and IBIE bits in the IBSR and IBCR, respectively.

10.3.2.4 IIC Status Register (IBSR)

	7	6	5	4	3	2	1	0
R	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
W								
Reset	1	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-7. IIC Bus Status Register (IBSR)

This status register is read-only with exception of bit 1 (IBIF) and bit 4 (IBAL), which are software clearable.

Table 10-7. IBSR Field Descriptions

Field	Description
7 TCF	**Data Transferring Bit** — While one byte of data is being transferred, this bit is cleared. It is set by the falling edge of the 9th clock of a byte transfer. Note that this bit is only valid during or immediately following a transfer to the IIC module or from the IIC module. 0 Transfer in progress 1 Transfer complete
6 IAAS	**Addressed as a Slave Bit** — When its own specific address (I-bus address register) is matched with the calling address, this bit is set.The CPU is interrupted provided the IBIE is set.Then the CPU needs to check the SRW bit and set its Tx/$\overline{\text{Rx}}$ mode accordingly.Writing to the I-bus control register clears this bit. 0 Not addressed 1 Addressed as a slave
5 IBB	Bus Busy Bit 0 This bit indicates the status of the bus. When a START signal is detected, the IBB is set. If a STOP signal is detected, IBB is cleared and the bus enters idle state. 1 Bus is busy
4 IBAL	**Arbitration Lost** — The arbitration lost bit (IBAL) is set by hardware when the arbitration procedure is lost. Arbitration is lost in the following circumstances: 1. SDA sampled low when the master drives a high during an address or data transmit cycle. 2. SDA sampled low when the master drives a high during the acknowledge bit of a data receive cycle. 3. A start cycle is attempted when the bus is busy. 4. A repeated start cycle is requested in slave mode. 5. A stop condition is detected when the master did not request it. This bit must be cleared by software, by writing a one to it. A write of 0 has no effect on this bit.
3 RESERVED	**Reserved** — Bit 3 of IBSR is reserved for future use. A read operation on this bit will return 0.

Table 10-7. IBSR Field Descriptions (continued)

Field	Description
2 SRW	**Slave Read/Write** — When IAAS is set this bit indicates the value of the R/W command bit of the calling address sent from the master This bit is only valid when the I-bus is in slave mode, a complete address transfer has occurred with an address match and no other transfers have been initiated. Checking this bit, the CPU can select slave transmit/receive mode according to the command of the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IBIF	**I-Bus Interrupt** — The IBIF bit is set when one of the following conditions occurs: — Arbitration lost (IBAL bit set) — Byte transfer complete (TCF bit set) — Addressed as slave (IAAS bit set) It will cause a processor interrupt request if the IBIE bit is set. This bit must be cleared by software, writing a one to it. A write of 0 has no effect on this bit.
0 RXAK	**Received Acknowledge** — The value of SDA during the acknowledge bit of a bus cycle. If the received acknowledge bit (RXAK) is low, it indicates an acknowledge signal has been received after the completion of 8 bits data transmission on the bus. If RXAK is high, it means no acknowledge signal is detected at the 9th clock. 0 Acknowledge received 1 No acknowledge received

10.3.2.5 IIC Data I/O Register (IBDR)

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

Figure 10-8. IIC Bus Data I/O Register (IBDR)

In master transmit mode, when data is written to the IBDR a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates next byte data receiving. In slave mode, the same functions are available after an address match has occurred.Note that the Tx/Rx bit in the IBCR must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For instance, if the IIC is configured for master transmit but a master receive is desired, then reading the IBDR will not initiate the receive.

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

In master transmit mode, the first byte of data written to IBDR following assertion of MS/$\overline{SL}$ is used for the address transfer and should com.prise of the calling address (in position D7:D1) concatenated with the required R/$\overline{W}$ bit (in position D0).

10.4 Functional Description

This section provides a complete functional description of the IICV2.

10.4.1 I-Bus Protocol

The IIC bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfer. All devices connected to it must have open drain or open collector outputs. Logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors is system dependent.

Normally, a standard communication is composed of four parts: START signal, slave address transmission, data transfer and STOP signal. They are described briefly in the following sections and illustrated in Figure 10-9.

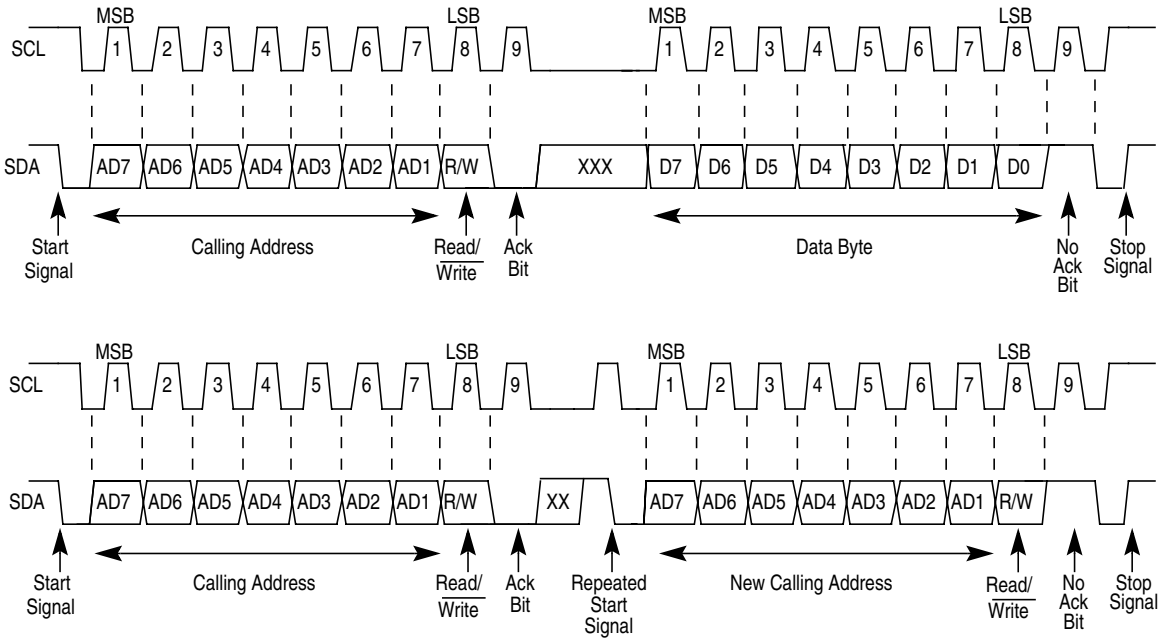


Figure 10-9. IIC-Bus Transmission Signals

10.4.1.1 START Signal

When the bus is free, i.e. no master device is engaging the bus (both SCL and SDA lines are at logical high), a master may initiate communication by sending a START signal.As shown in Figure 10-9, a START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer may contain several bytes of data) and brings all slaves out of their idle states.

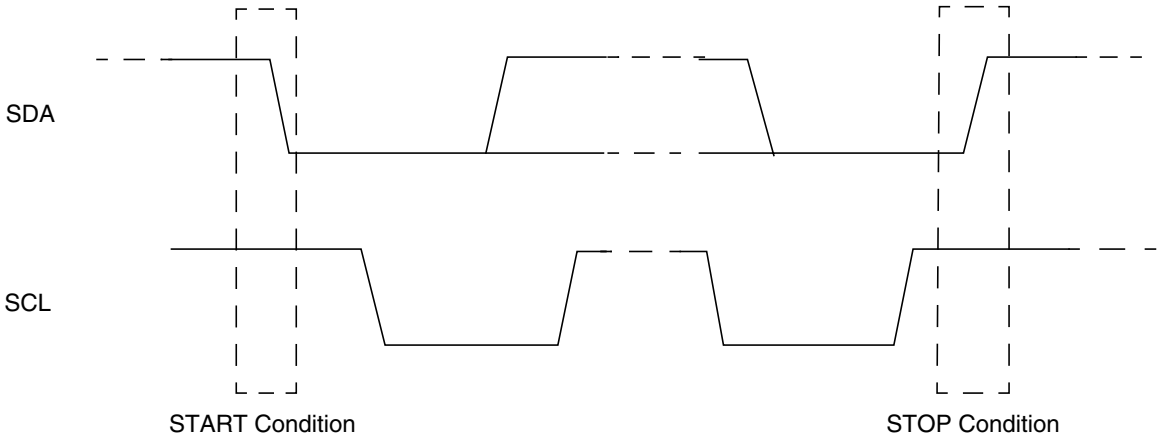


Figure 10-10. Start and Stop Conditions

10.4.1.2 Slave Address Transmission

The first byte of data transfer immediately after the START signal is the slave address transmitted by the master. This is a seven-bit calling address followed by a R/W bit. The R/W bit tells the slave the desired direction of data transfer.

1 = Read transfer, the slave transmits data to the master.
0 = Write transfer, the master transmits data to the slave.

Only the slave with a calling address that matches the one transmitted by the master will respond by sending back an acknowledge bit. This is done by pulling the SDA low at the 9th clock (see Figure 10-9).

No two slaves in the system may have the same address. If the IIC bus is master, it must not transmit an address that is equal to its own slave address. The IIC bus cannot be master and slave at the same time.However, if arbitration is lost during an address cycle the IIC bus will revert to slave mode and operate correctly even if it is being addressed by another master.

10.4.1.3 Data Transfer

As soon as successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the R/W bit sent by the calling master

All transfers that come after an address cycle are referred to as data transfers, even if they carry sub-address information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low and must be held stable while SCL is high as shown in Figure 10-9. There is one clock pulse on SCL for each data bit, the MSB being transferred first. Each data byte has to be followed by an acknowledge bit, which is signalled from the receiving device by pulling the SDA low at the ninth clock. So one complete data byte transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master, the SDA line must be left high by the slave. The master can then generate a stop signal to abort the data transfer or a start signal (repeated start) to commence a new calling.

If the master receiver does not acknowledge the slave transmitter after a byte transmission, it means 'end of data' to the slave, so the slave releases the SDA line for the master to generate STOP or START signal.

10.4.1.4 STOP Signal

The master can terminate the communication by generating a STOP signal to free the bus. However, the master may generate a START signal followed by a calling command without generating a STOP signal first. This is called repeated START. A STOP signal is defined as a low-to-high transition of SDA while SCL at logical 1 (see Figure 10-9).

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

10.4.1.5 Repeated START Signal

As shown in Figure 10-9, a repeated START signal is a START signal generated without first generating a STOP signal to terminate the communication. This is used by the master to communicate with another slave or with the same slave in different mode (transmit/receive mode) without releasing the bus.

10.4.1.6 Arbitration Procedure

The Inter-IC bus is a true multi-master bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock, for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. The relative priority of the contending masters is determined by a data arbitration procedure, a bus master loses arbitration if it transmits logic 1 while another master transmits logic 0. The losing masters immediately switch over to slave receive mode and stop driving SDA output. In this case the transition from master to slave mode does not generate a STOP condition. Meanwhile, a status bit is set by hardware to indicate loss of arbitration.

10.4.1.7 Clock Synchronization

Because wire-AND logic is performed on SCL line, a high-to-low transition on SCL line affects all the devices connected on the bus. The devices start counting their low period and as soon as a device's clock has gone low, it holds the SCL line low until the clock high state is reached.However, the change of low to high in this device clock may not change the state of the SCL line if another device clock is within its low period. Therefore, synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see Figure 10-10). When all devices concerned have counted off their low period, the synchronized clock SCL line is released and pulled high. There is then no difference between the device clocks and the state of the SCL line and all the devices start counting their high periods.The first device to complete its high period pulls the SCL line low again.

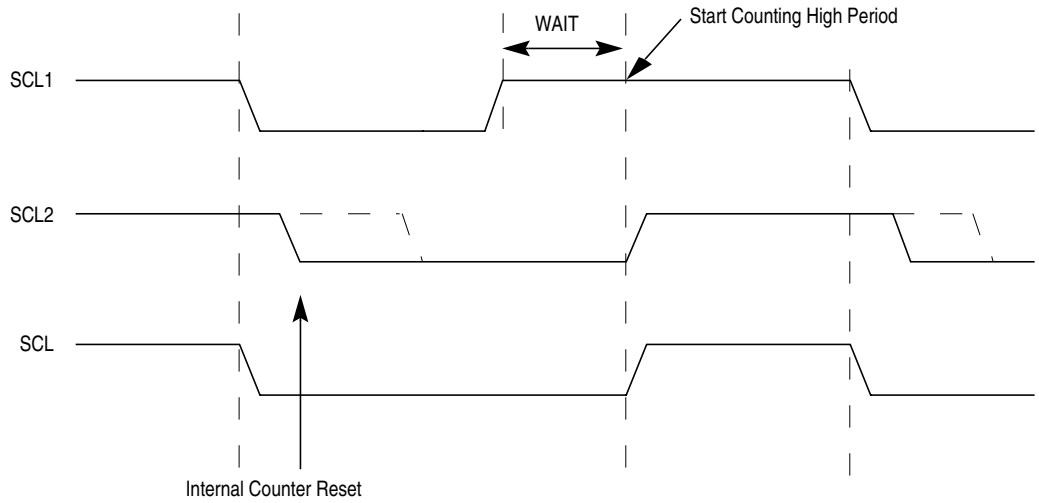


Figure 10-11. IIC-Bus Clock Synchronization

10.4.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte transfer (9 bits). In such case, it halts the bus clock and forces the master clock into wait states until the slave releases the SCL line.

10.4.1.9 Clock Stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master has driven SCL low the slave can drive SCL low for the required period and then release it.If the slave SCL low period is greater than the master SCL low period then the resulting SCL bus signal low period is stretched.

10.4.2 Operation in Run Mode

This is the basic mode of operation.

10.4.3 Operation in Wait Mode

IIC operation in wait mode can be configured. Depending on the state of internal bits, the IIC can operate normally when the CPU is in wait mode or the IIC clock generation can be turned off and the IIC module enters a power conservation state during wait mode. In the later case, any transmission or reception in progress stops at wait mode entry.

10.4.4 Operation in Stop Mode

The IIC is inactive in stop mode for reduced power consumption. The STOP instruction does not affect IIC register states.

10.5 Resets

The reset state of each individual bit is listed in Section 10.3, “Memory Map and Register Definition,” which details the registers and their bit-fields.

10.6 Interrupts

IICV2 uses only one interrupt vector.

Table 10-8. Interrupt Summary

Interrupt	Offset	Vector	Priority	Source	Description
IIC Interrupt	—	—	—	IBAL, TCF, IAAS bits in IBSR register	When either of IBAL, TCF or IAAS bits is set may cause an interrupt based on arbitration lost, transfer complete or address detect conditions

Internally there are three types of interrupts in IIC. The interrupt service routine can determine the interrupt type by reading the status register.

IIC Interrupt can be generated on

1. Arbitration lost condition (IBAL bit set)
2. Byte transfer condition (TCF bit set)
3. Address detect condition (IAAS bit set)

The IIC interrupt is enabled by the IBIE bit in the IIC control register. It must be cleared by writing 0 to the IBF bit in the interrupt service routine.

10.7 Initialization/Application Information

10.7.1 IIC Programming Examples

10.7.1.1 Initialization Sequence

Reset will put the IIC bus control register to its default status. Before the interface can be used to transfer serial data, an initialization procedure must be carried out, as follows:

1. Update the frequency divider register (IBFD) and select the required division ratio to obtain SCL frequency from system clock.
2. Update the IIC bus address register (IBAD) to define its slave address.
3. Set the IBEN bit of the IIC bus control register (IBCR) to enable the IIC interface system.
4. Modify the bits of the IIC bus control register (IBCR) to select master/slave mode, transmit/receive mode and interrupt enable or not.

10.7.1.2 Generation of START

After completion of the initialization procedure, serial data can be transmitted by selecting the 'master transmitter' mode. If the device is connected to a multi-master bus system, the state of the IIC bus busy bit (IBB) must be tested to check whether the serial bus is free.

If the bus is free (IBB=0), the start condition and the first byte (the slave address) can be sent. The data written to the data register comprises the slave calling address and the LSB set to indicate the direction of transfer required from the slave.

The bus free time (i.e., the time between a STOP condition and the following START condition) is built into the hardware that generates the START cycle. Depending on the relative frequencies of the system clock and the SCL period it may be necessary to wait until the IIC is busy after writing the calling address to the IBDR before proceeding with the following instructions. This is illustrated in the following example.

An example of a program which generates the START signal and transmits the first byte of data (slave address) is shown below:

```
CHFLAG      BRSET   IBSR,#$20,*     ;WAIT FOR IBB FLAG TO CLEAR
TXSTART     BSET    IBCR,#$30       ;SET TRANSMIT AND MASTER MODE;i.e. GENERATE START CONDITION
            MOVB    CALLING,IBDR    ;TRANSMIT THE CALLING ADDRESS, D0=R/W
IBFREE      BRCLR   IBSR,#$20,*     ;WAIT FOR IBB FLAG TO SET
```

10.7.1.3 Post-Transfer Software Response

Transmission or reception of a byte will set the data transferring bit (TCF) to 1, which indicates one byte communication is finished. The IIC bus interrupt bit (IBIF) is set also; an interrupt will be generated if the interrupt function is enabled during initialization by setting the IBIE bit. Software must clear the IBIF bit in the interrupt routine first. The TCF bit will be cleared by reading from the IIC bus data I/O register (IBDR) in receive mode or writing to IBDR in transmit mode.

Software may service the IIC I/O in the main program by monitoring the IBIF bit if the interrupt function is disabled. Note that polling should monitor the IBIF bit rather than the TCF bit because their operation is different when arbitration is lost.

Note that when an interrupt occurs at the end of the address cycle the master will always be in transmit mode, i.e. the address is transmitted. If master receive mode is required, indicated by R/W bit in IBDR, then the Tx/Rx bit should be toggled at this stage.

During slave mode address cycles (IAAS=1), the SRW bit in the status register is read to determine the direction of the subsequent transfer and the Tx/Rx bit is programmed accordingly. For slave mode data cycles (IAAS=0) the SRW bit is not valid, the Tx/Rx bit in the control register should be read to determine the direction of the current transfer.

The following is an example of a software response by a 'master transmitter' in the interrupt routine.

```
ISR         BCLR    IBSR,#$02           ;CLEAR THE IBIF FLAG
            BRCLR   IBCR,#$20,SLAVE     ;BRANCH IF IN SLAVE MODE
            BRCLR   IBCR,#$10,RECEIVE   ;BRANCH IF IN RECEIVE MODE
            BRSET   IBSR,#$01,END       ;IF NO ACK, END OF TRANSMISSION
TRANSMIT    MOVB    DATABUF,IBDR        ;TRANSMIT NEXT BYTE OF DATA
```

10.7.1.4 Generation of STOP

A data transfer ends with a STOP signal generated by the 'master' device. A master transmitter can simply generate a STOP signal after all the data has been transmitted. The following is an example showing how a stop condition is generated by a master transmitter.

```
MASTX   TST    TXCNT       ;GET VALUE FROM THE TRANSMITING COUNTER
        BEQ    END         ;END IF NO MORE DATA
        BRSET  IBSR,#$01,END ;END IF NO ACK
        MOVB   DATABUF,IBDR ;TRANSMIT NEXT BYTE OF DATA
        DEC    TXCNT       ;DECREASE THE TXCNT
        BRA    EMASTX      ;EXIT
END     BCLR   IBCR,#$20   ;GENERATE A STOP CONDITION
EMASTX  RTI                ;RETURN FROM INTERRUPT
```

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data which can be done by setting the transmit acknowledge bit (TXAK) before reading the 2nd last byte of data. Before reading the last byte of data, a STOP signal must be generated first. The following is an example showing how a STOP signal is generated by a master receiver.

```
MASR    DEC    RXCNT       ;DECREASE THE RXCNT
        BEQ    ENMASR      ;LAST BYTE TO BE READ
        MOVB   RXCNT,D1    ;CHECK SECOND LAST BYTE
        DEC    D1          ;TO BE READ
        BNE    NXMAR       ;NOT LAST OR SECOND LAST
LAMAR   BSET   IBCR,#$08   ;SECOND LAST, DISABLE ACK
                           ;TRANSMITTING
        BRA    NXMAR
ENMASR  BCLR   IBCR,#$20   ;LAST ONE, GENERATE 'STOP' SIGNAL
NXMAR   MOVB   IBDR,RXBUF  ;READ DATA AND STORE
        RTI
```

10.7.1.5 Generation of Repeated START

At the end of data transfer, if the master continues to want to communicate on the bus, it can generate another START signal followed by another slave address without first generating a STOP signal. A program example is as shown.

```
RESTART BSET   IBCR,#$04   ;ANOTHER START (RESTART)
        MOVB   CALLING,IBDR ;TRANSMIT THE CALLING ADDRESS;D0=R/W
```

10.7.1.6 Slave Mode

In the slave interrupt service routine, the module addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the transmit/receive mode select bit (Tx/Rx bit of IBCR) according to the R/W command bit (SRW). Writing to the IBCR clears the IAAS automatically. Note that the only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred, interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer may now be initiated by writing information to IBDR, for slave transmits, or dummy reading from IBDR, in slave receive mode. The slave will drive SCL low in-between byte transfers, SCL is released when the IBDR is accessed in the required mode.

In slave transmitter routine, the received acknowledge bit (RXAK) must be tested before transmitting the next byte of data. Setting RXAK means an 'end of data' signal from the master receiver, after which it must be switched from transmitter mode to receiver mode by software. A dummy read then releases the SCL line so that the master can generate a STOP signal.

10.7.1.7 Arbitration Lost

If several masters try to engage the bus simultaneously, only one master wins and the others lose arbitration. The devices which lost arbitration are immediately switched to slave receive mode by the hardware. Their data output to the SDA line is stopped, but SCL continues to be generated until the end of the byte during which arbitration was lost. An interrupt occurs at the falling edge of the ninth clock of this transfer with IBAL=1 and MS/SL=0. If one master attempts to start transmission while the bus is being engaged by another master, the hardware will inhibit the transmission; switch the MS/SL bit from 1 to 0 without generating STOP condition; generate an interrupt to CPU and set the IBAL to indicate that the attempt to engage the bus is failed. When considering these cases, the slave service routine should test the IBAL first and the software should clear the IBAL bit if it is set.

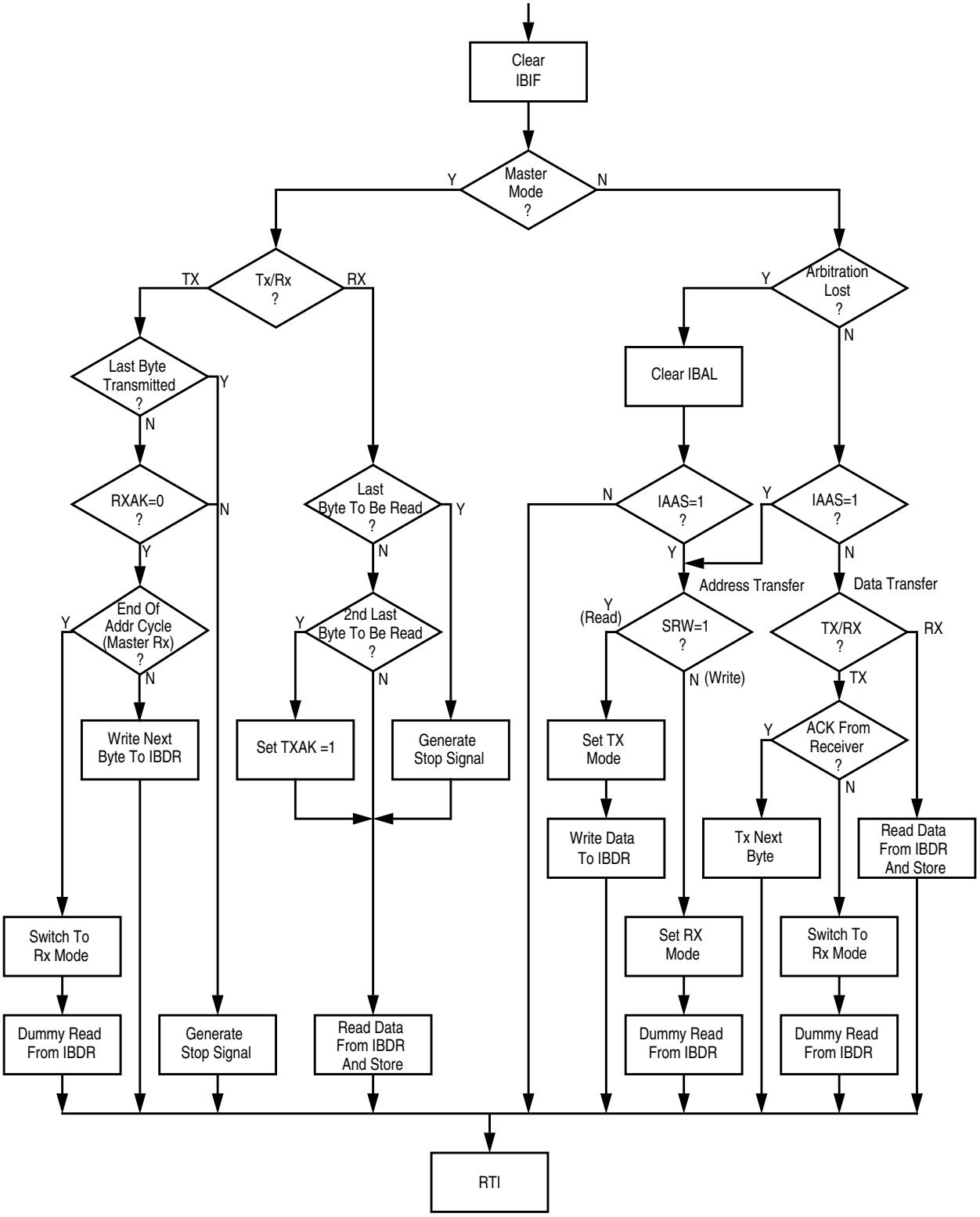


Figure 10-12. Flow-Chart of Typical IIC Interrupt Routine

Chapter 11
Ethernet Media Access Controller (EMACV1)

11.1 Introduction

The Ethernet media access controller (EMAC) is IEEE 802.3 compliant supporting 10/100 Ethernet operation. The EMAC module supports the medium-independent interface (MII) and the MII management interface (MI). By connecting a physical layer device (PHY) supporting MII, a 10/100 Mbps Ethernet network is implemented.

11.1.1 Features

- IEEE 802.3 compliant
- Medium-independent interface (MII)
- Full-duplex and half-duplex modes
- Flow control using pause frames
- MII management function
- Address recognition
  - Frames with broadcast address are always accepted or always rejected
  - Exact match for single 48-bit individual (unicast) address
  - Hash (64-bit hash) check of group (multicast) addresses
  - Promiscuous mode
- Ethertype filter
- Loopback mode
- Two receive and one transmit Ethernet buffer interfaces

11.1.2 Block Diagram

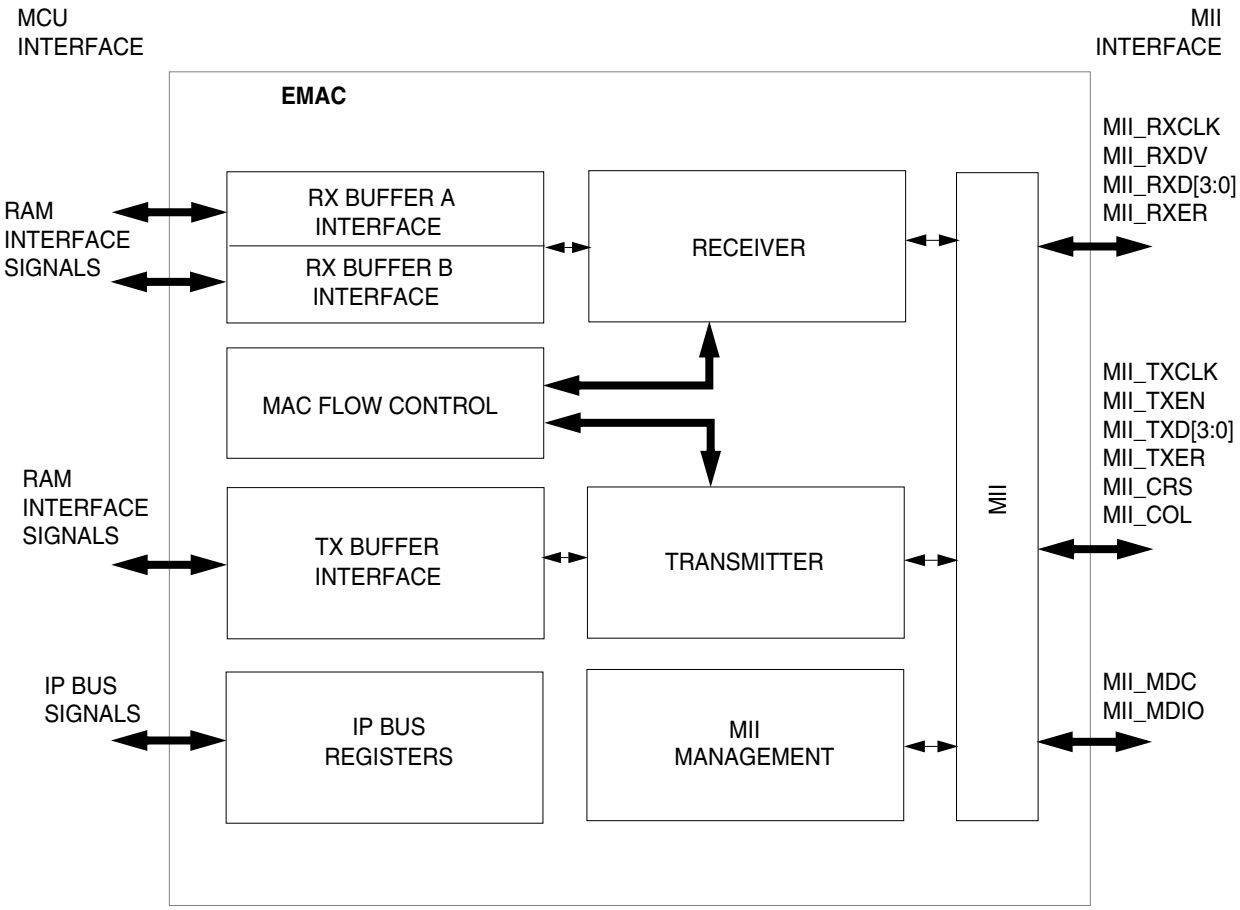


Figure 11-1. EMAC Block Diagram

11.2 External Signal Description

The EMAC module supports the medium-independent interface (MII) which requires 18 input/output (I/O) pins. The transmit and receive functions require seven signals each (four data signals, a delimiter, error, and clock). In addition, there are two signals which indicate the status of the media (one indicates the presence of a carrier and the other indicates that a collision has occurred). The MII management function requires the remaining two signals, MII_MDC and MII_MDIO. Each MII signal is described below. These signals are available externally only when the EMAC is enabled in external PHY mode. MII signals are available only in certain MCU modes.

11.2.1 MII_TXCLK — MII Transmit Clock

The PHY provides this input clock, which is used as a timing reference for MII_TXD, MII_TXEN, and MII_TXER. It operates at 25% of the transmit data rate (25 MHz for 100 Mbps or 2.5 MHz for 10 Mbps). The EMAC bus clock frequency must be greater-than or equal-to MII_TXCLK.

11.2.2 MII_TXD[3:0] — MII Transmit Data

MII_TXD[3:0] is a transmit nibble of data to be transferred from the EMAC to the PHY. The nibble is synchronized to the rising edge of MII_TXCLK. When MII_TXEN is asserted, the PHY accepts MII_TXD[3:0], and at all other times, MII_TXD[3:0] is ignored. MII_TXD[0] is the least significant bit.

Table 11-1 summarizes the permissible encoding of MII_TXD[3:0], MII_TXEN, and MII_TXER.

Table 11-1. Permissible Encoding of MII_TXD, MII_TXEN, and MII_TXER

MII_TXEN	MII_TXER	MII_TXD[3:0]	Indication
0	0	0000 through 1111	Normal interframe
0	1	0000 through 1111	Reserved
1	0	0000 through 1111	Normal data transmission
1	1	0000 through 1111	Transmit error propagation

11.2.3 MII_TXEN — MII Transmit Enable

Assertion of this output signal indicates that there are valid nibbles being presented on the MII and the transmission can start. This signal is asserted with the first nibble of the preamble, remains asserted until all nibbles to be transmitted have been presented to the PHY, and is negated following the final nibble of the frame.

11.2.4 MII_TXER — MII Transmit Coding Error

Assertion of this output signal for one or more clock cycles while MII_TXEN is asserted causes the PHY to transmit one or more illegal symbols. MII_TXER is asserted if the ABORT command is issued during a transmit. This signal transitions synchronously with respect to MII_TXCLK.

11.2.5 MII_RXCLK — MII Receive Clock

The PHY provides this input clock, which is used as a timing reference for MII_RXD, MII_RXDV, and MII_RXER. It operates at 25% of the receive data rate (25 MHz for 100 Mbps or 2.5 MHz for 10 Mbps). The EMAC bus clock frequency must be greater-than or equal-to MII_RXCLK.

11.2.6 MII_RXD[3:0] — MII Receive Data

MII_RXD[3:0] is a receive nibble of data to be transferred from the PHY to the EMAC. The nibble is synchronized to the rising edge of MII_RXCLK. When MII_RXDV is asserted, the EMAC accepts the MII_RXD[3:0], and at all other times, MII_RXD[3:0] is ignored. MII_RXD[0] is the least significant bit.

Table 11-2 summarizes the permissible encoding of MII_RXD, MII_RXDV, and MII_RXER, as well as the specific indication provided by each code. A false carrier indication is ignored by the EMAC.

Table 11-2. Permissible Encoding of MII_RXD, MII_RXDV, and MII_RXER

MII_RXDV	MII_RXER	MII_RXD[3:0]	Indication
0	0	0000 through 1111	Normal interframe
0	1	0000	Normal interframe
0	1	0001 through 1101	Reserved
0	1	1110	False carrier
0	1	1111	Reserved
1	0	0000 through 1111	Normal data reception
1	1	0000 through 1111	Data reception with errors

11.2.7 MII_RXDV — MII Receive Data Valid

When this input signal is asserted, the PHY is indicating that a valid nibble is present on the MII. This signal remains asserted from the first recovered nibble of the frame through the last nibble. Assertion of MII_RXDV must start no later than the start frame delimiter (SFD).

11.2.8 MII_RXER — MII Receive Error

When this input signal and MII_RXDV are asserted, the PHY is indicating that a media error has been detected during the transmission of the current frame. At all other times, MII_RXER is ignored. This signal transitions synchronously with MII_RXCLK.

11.2.9 MII_CRS — MII Carrier Sense

This input signal is asserted when the transmit or receive medium is in a non-idle state. When de-asserted, this signal indicates that the medium is in an idle state and a transmission can start. In the event of a collision, MII_CRS remains asserted through the duration of the collision. In full-duplex mode, this signal is undefined. This signal is not required to transition synchronously with MII_TXCLK or MII_RXCLK.

11.2.10 MII_COL — MII Collision

This input signal is asserted upon detection of a collision, and remains asserted through the duration of the collision. In full-duplex mode, this signal is undefined. This signal is not required to transition synchronously with MII_TXCLK or MII_RXCLK.

11.2.11 MII_MDC — MII Management Data Clock

This output signal provides a timing reference to the PHY for data transfers on the MII_MDIO signal. MII_MDC is aperiodic and has no maximum high or low times. The maximum clock frequency is 2.5 MHz, regardless of the nominal period of MII_TXCLK and MII_RXCLK.

11.2.12 MII_MDIO — MII Management Data Input/Output

This bidirectional signal transfers control/status information between the PHY and EMAC. Control information is driven by the EMAC synchronously with respect to MII_MDC and is sampled synchronously by the PHY. Status information is driven by the PHY synchronously with respect to MII_MDC and is sampled synchronously by the EMAC.

11.3 Memory Map and Register Descriptions

This section provides a detailed description of all registers accessible in the EMAC.

11.3.1 Module Memory Map

Table 11-3 gives an overview of all registers in the EMAC memory map. The EMAC occupies 48 bytes in the memory space. The register address results from the addition of *base address* and *address offset*. The *base address* is determined at the MCU level and is given in the device user guide. The *address offset* is defined at the module level and is provided in Table 11-3.

Table 11-3. EMAC Module Memory Map

Address Offset	Use	Access
$__00	Network Control (NETCTL)	R/W
$__01	Reserved	
$__02		
$__03	Receive Control and Status (RXCTS)	R/W
$__04	Transmit Control and Status (TXCTS)	R/W
$__05	Ethertype Control (ETCTL)	R/W
$__06	Programmable Ethertype (ETYPE)	R/W
$__07		
$__08	PAUSE Timer Value and Counter (PTIME)	R/W
$__09		
$__0A	Interrupt Event (IEVENT)	R/W
$__0B		
$__0C	Interrupt Mask (IMASK)	R/W
$__0D		
$__0E	Software Reset (SWRST)	R/W
$__0F	Reserved	
$__10	MII Management PHY Address (MPADR)	R/W
$__11	MII Management Register Address (MRADR)	R/W

Table 11-3. EMAC Module Memory Map (continued)

Address Offset	Use	Access
$__12	MII Management Write Data (MWDATA)	R/W
$__13		
$__14	MII Management Read Data (MRDATA)	R
$__15		
$__16	MII Management Command and Status (MCMST)	R/W
$__17	Reserved	
$__18	Ethernet Buffer Configuration (BUFCFG)	R/W
$__19		
$__1A	Receive A End-of-Frame Pointer (RXAEFP)	R
$__1B		
$__1C	Receive B End-of-Frame Pointer (RXBEFP)	R
$__1D		
$__1E	Transmit End-of-Frame Pointer (TXEFP)	R/W
$__1F		
$__20	Multicast Hash Table (MCHASH)	R/W
$__21		
$__22		
$__23		
$__24		
$__25		
$__26		
$__27		
$__28	MAC Address (MACAD)	R/W
$__29		
$__2A		
$__2B		
$__2C		
$__2D		
$__2E	Miscellaneous (EMISC)	R/W
$__2F		

11.3.2 Register Descriptions

This section describes in detail all the registers and register bits in the EMAC module. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

11.3.2.1 Network Control (NETCTL)

Module Base + $0

	7	6	5	4	3	2	1	0
R	EMACE	0	0	ESWAI	EXTPHY	MLB	FDX	0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-2. Network Control (NETCTL)

Read: Anytime.

Write: See each bit description.

NOTE

When configuring for loopback mode or for an external PHY, the user must set the MLB or EXTPHY bit before enabling the EMAC by setting EMACE. That is, when setting MLB or EXTPHY, the initial write to this register should not also set the EMACE bit; separate writes must be performed.

NOTE

When configuring MLB and EXTPHY bits, any internal or external PHY connected should be disabled to protect against possible glitches generated on MII signals as port configuration logic settles.

EMACE — EMAC Enable

This bit can be written anytime, but the user must not modify this bit while TXACT is set.

While this bit is set, the EMAC is enabled, and reception and transmission are possible. When this bit is cleared, the EMAC receiver and transmitter are immediately disabled, any receive in progress is dropped, and any PAUSE timeout is cleared. EMACE has no effect on the MII management functions.

1 = Enables EMAC.
0 = Disables EMAC.

ESWAI — EMAC Disabled during Wait Mode

This bit can be written anytime.

When this bit is set, the EMAC receiver, transmitter, and MII management logic are disabled during wait mode, any receive in progress is dropped, and any PAUSE timeout is cleared. The user must not enter wait mode with the ESWAI bit set if TXACT or BUSY are asserted. While the ESWAI bit is clear, the EMAC continues to operate during wait mode.

1 = EMAC is disabled during wait mode.
0 = EMAC continues to operate normally during wait mode.

EXTPHY — External PHY

This bit can be written once after a hardware or software reset, but the user must not modify this bit while EMACE or BUSY is set. While this bit is set, the EMAC is configured for an external PHY, all the EMAC MII I/O pins are available externally, and the MII to the internal PHY is not available.

While this bit is clear, the EMAC is configured for the internal PHY, all the EMAC MII I/O pins are not available externally, and the MII interface to the internal PHY is available.

1 = External PHY.

0 = Internal PHY.

NOTE

If MLB is set, EXTPHY is ignored. If EXTPHY is set, it is recommended that any internal PHY be disabled.

MLB — MAC Loopback

This bit can be written once after a hardware or software reset, but the user must not change this bit while EMACE or BUSY is set.

While this bit is set, the EMAC is in the loopback mode which routes all transmit traffic to the receiver and disables the MII.

1 = Loopback mode.

0 = Normal operation.

NOTE

While configured for loopback mode, receiver frame recognition algorithms remain active and transmitted frames failing to meet acceptance criteria will be dropped by the receiver.

FDX — Full Duplex

This bit can be written anytime, but the user must not modify this bit while EMACE is set.

While this bit is set, the EMAC is set for full-duplex mode, which bypasses the carrier sense multiple access with collision detect (CSMA/CD) protocol. Frame reception occurs independently of frame transmission.

While this bit is clear, the EMAC is set for half-duplex mode. Frame reception is disabled during frame transmission. The mode used is the traditional mode of operation that relies on the CSMA/CD protocol to manage collisions and network access.

1 = Full-duplex mode.

0 = Half-duplex mode.

11.3.2.2 Receive Control and Status (RXCTS)

Module Base + $3

	7	6	5	4	3	2	1	0
R	RXACT	0	0	RFCE	0	PROM	CONMC	BCREJ
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-3. Receive Control and Status (RXCTS)

Read: Anytime.

Write: See each bit description.

RXACT — Receiver Active Status

This is a read-only status bit that indicates activity in the EMAC receiver. RXACT is asserted when MII_RXDV is asserted and clears when the EMAC has finished processing the receive frame after MII_RXDV is negated.

1 = Receiver is active.
0 = Receiver is idle.

RFCE — Reception Flow Control Enable

This bit can be written anytime, but the user must not change this bit while EMACE is set.

While this bit is set, the receiver detects PAUSE frames (full-duplex mode only). Upon PAUSE frame detection, the transmitter stops transmitting data frames for a given duration (PAUSE time in received frame). The value of the PAUSE timer counter is updated when a valid PAUSE control frame is received.

While this bit is clear, the receiver ignores any PAUSE frames.

1 = Upon PAUSE frame detection, transmitter stops for a given duration.
0 = Received PAUSE control frames are ignored.

PROM — Promiscuous Mode

This bit can be written anytime, but the user must not change this bit while EMACE is set. Changing values while the receiver is active may affect the outcome of the receive filters.

While set, the address recognition filter is ignored and all frames are received regardless of destination address.

While clear, the destination address is checked for incoming frames.

1 = All frames are received regardless of address.
0 = Destination address is checked for incoming frames.

CONMC — Conditional Multicast

This bit can be written anytime, but the user must not change this bit while EMACE is set. Changing values while the receiver is active may affect the outcome of the receive filters.

While set, the multicast hash table is used to check all multicast addresses received unless the PROM bit is set.

While clear, all multicast address frames are accepted.

1 = Multicast hash table is used for checking multicast addresses.
0 = Multicast address frames are accepted.

BCREJ — Broadcast Reject

This bit can be written anytime, but the user must not change this bit while EMACE is set.

While set, all broadcast addresses are rejected unless the PROM bit is set.

While clear, all broadcast address frames are accepted.

1 = All broadcast address frames are rejected.
0 = All broadcast address frames are accepted.

11.3.2.3 Transmit Control and Status (TXCTS)

Module Base + $4

	7	6	5	4	3	2	1	0
R	TXACT	0	CSLF	PTRC	SSB	0	0	0
W							TCMD	
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-4. Transmit Control and Status (TXCTS)

Read: Anytime.

Write: See each bit description.

TXACT — Transmitter Active Status

This is a read-only status bit that indicates activity in the EMAC transmitter. TXACT is set after a valid TCMD write and is cleared when the EMAC has finished sending the transmit frame.

1 = Transmitter is active.
0 = Transmitter is idle.

CSLF — Carrier Sense Lost Flag

This status bit is set when the carrier sense (half-duplex mode) drops out or is never sensed during transmission (excluding the preamble) without collision. The frame is transmitted normally and no retries are performed as a result of this flag. This flag bit is cleared by writing a 1 to it. A write of 0 has no effect.

1 = Carrier sense lost has been detected without collision during transmission.
0 = No carrier sense lost has been detected.

PTRC — PAUSE Timer Register Control

This bit can be written anytime.

While set, writes to the PTIME register update the PAUSE duration used in the transmission of a PAUSE control frame. Reads of the PTIME register return the PAUSE duration used in the transmission of a hardware-generated PAUSE control frame.

While clear, PTIME register read accesses return the current number of slot times (512 bit times) remaining in a PAUSE period after the receiver accepts a PAUSE frame. Writes to PTIME are ignored.

1 = PTIME controls the transmit PAUSE duration parameter for PAUSE control frames.
0 = PTIME read accesses return the PAUSE timer counter value.

SSB — Single Slot Backoff

This bit can be written anytime, but the user must not change this bit while TXACT is set.

Setting this bit forces the transmitter to backoff for only a single Ethernet slot time instead of following the random backoff algorithm. For more information about the backoff algorithm, refer to Section 11.4.3.3.3, “Backoff Generator.”

1 = Single slot backoff.
0 = Random backoff.

TCMD — Transmit Command

This is a 2-bit write-only field that can launch three different transmission commands: START, PAUSE, or ABORT. The START command starts transmission of the frame in the transmit buffer. The PAUSE command starts transmission of a hardware-generated PAUSE frame. The ABORT command terminates any current transmission after a bad CRC is appended to the frame currently being transmitted and MII_TXER is asserted. The ABORT command does not affect any received PAUSE time out. See Table 11-4, Section 11.4.3, “Transmitter,” and section Section 11.4.5.2, “Hardware Generated PAUSE Control Frame Transmission,” for more detail.

NOTE

The START and PAUSE commands are ignored if there is a transmission in progress (TXACT is set). After the reception of a PAUSE frame, a launched START command is suspended until the pause time has expired. During the pause time, the EMAC may transmit a control PAUSE frame if no START transmission is pending.

Table 11-4. Transmit Commands

TCMD	Command	Description
0	Reserved	Ignore
1	START	Transmit buffer frame
2	PAUSE	Transmit PAUSE frame (full-duplex mode only)
3	ABORT	Abort transmission

11.3.2.4 Ethertype Control (ETCTL)

Module Base + $5

	7	6	5	4	3	2	1	0
R	FPET	0	0	FEMW	FIPV6	FARP	FIPV4	FIEEE
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-5. Ethertype Control (ETCTL)

Read: Anytime.

Write: Anytime, but the user must not change this field while EMACE is set. Changing values while the receiver is active will affect the outcome of the Ethertype filter.

If every bit in ETCTL is clear, there is no mask for Ethertype messages so *all* are received. Conversely, if any bit in ETCTL is set, Ethertype filtering will occur and will be defined by the configuration bits.

FPET — Programmable Ethertype

If this bit is set, all messages with the Ethertype in ETYPE are accepted. If this bit is clear, messages of this type are ignored.

1 = Accept Ethertype messages selected in ETYPE.
0 = Ignore Ethertype messages selected in ETYPE.

FEMW — Emware Ethertype

If this bit is set, all messages with 0x8876 Ethertype are accepted. If this bit is clear, messages of this type are ignored.

1 = Accept Emware messages.
0 = Ignore Emware messages.

FIPV6 — Internet Protocol Version 6 (IPv6) Ethertype

If this bit is set, all messages with 0x86DD Ethertype are accepted. If this bit is clear, messages of this type are ignored.

1 = Accept IPv6 messages.
0 = Ignore IPv6 messages.

FARP — Address Resolution Protocol (ARP) Ethertype

If this bit is set, all messages with 0x0806 Ethertype are accepted. If this bit is clear, messages of this type are ignored.

1 = Accept ARP messages.
0 = Ignore ARP messages.

FIPV4 — Internet Protocol Version 4 (IPv4) Ethertype

If this bit is set, all messages with 0x0800 Ethertype are accepted. If this bit is clear, messages of this type are ignored.

1 = Accept IPv4 messages.
0 = Ignore IPv4 messages.

FIEEE — IEEE802.3 Length Field Ethertype

If this bit is set, all messages with 0x0000 to 0x05DC Ethertype are accepted. If this bit is clear, messages of this type are ignored.

1 = Accept length field messages.
0 = Ignore length field messages.

11.3.2.5 Programmable Ethertype (ETYPE)

Module Base + $6

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ETYPE															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-6. Programmable Ethertype (ETYPE)

Read: Anytime.

Write: Anytime, but the user must not change this field while EMACE is set. Changing values while the receiver is active may affect the outcome of the Ethertype filter.

ETYPE — Programmable Ethertype

This 16-bit field is used to program an Ethertype value to be used for the Ethertype filter.

11.3.2.6 PAUSE Timer Value and Counter (PTIME)

Module Base + $8

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PTIME															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-7. PAUSE Timer Value and Counter (PTIME)

Read: Anytime.

Write: Anytime except while PTRC is clear, but the user must not change this field while TXACT is set.

PTIME — PAUSE Timer Value and Counter

While the PTRC bit is set, the PTIME register controls the PAUSE duration parameter in units of slot times (512 bit times) used in a transmission of a PAUSE control frame.

While PTRC bit is clear, the PTIME register indicates the current number of slot times (512 bit times) remaining in a PAUSE period after the receiver accepts a PAUSE frame.

The value of the PAUSE timer counter is updated when a valid PAUSE control frame is accepted, regardless of PTRC.

11.3.2.7 Interrupt Event (IEVENT)

When an event occurs that sets a bit in the interrupt event register, an interrupt is generated if the corresponding bit in the interrupt mask registers is also set. Each bit in the interrupt event register is cleared by writing a 1 to that bit position. A write of 0 has no effect.

Module Base + $A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RFCIF	0	BREIF	RXEIF	RXAOIF	RXBOIF	RXACIF	RXBCIF	MMCIF	0	LCIF	ECIF	0	0	TXCIF	0
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-8. Interrupt Event (IEVENT)

Read: Anytime.

Write: Anytime (0s have no effect).

RFCIF — Receive Flow Control Interrupt Flag

This flag is set when a full-duplex flow control PAUSE frame has been received. If not masked (RFCIE is set), a receive flow control interrupt is pending while this flag is set.

1 = Transmitter stopped due to reception of a PAUSE frame.
0 = Normal transmit operation.

BREIF — Babbling Receive Error Interrupt Flag

This flag is set when the receive frame length exceeds the value of MAXFL. If not masked (BREIE is set), a babbling receive error interrupt is pending while this flag is set.

1 = A babbling receive error has been detected.
0 = No babbling receive errors have been detected.

RXEIF — Receive Error Interrupt Flag

This flag is set when MII_RXER signal is asserted during reception, when there is a receive frame length mismatch, an alignment error, or when a CRC error has occurred. If not masked (RXEIE is set), a receive error interrupt is pending while this flag is set.

1 = Receive errors have been detected.
0 = No receive errors have been detected.

RXAOIF — Receive Buffer A Overrun Interrupt Flag

This flag is set when an overrun occurs in receive buffer A. If not masked (RXAOIE is set), a receive buffer A overrun interrupt is pending while this flag is set.

1 = Receive buffer A overrun has occurred.
0 = No receive buffer A overrun has been detected.

RXBOIF — Receive Buffer B Overrun Interrupt Flag

This flag is set when an overrun occurs in receive buffer B. If not masked (RXBOIE is set), a receive buffer B overrun interrupt is pending while this flag is set.

1 = Receive buffer B overrun has occurred.
0 = No receive buffer B overrun has been detected.

RXACIF — Valid Frame Reception to Receive Buffer A Complete Interrupt Flag

This flag is set when a complete valid frame has been received in receive buffer A. If not masked (RXACIE is set), a valid frame reception to receive buffer A complete interrupt is pending while this flag is set.

1 = Frame to receive buffer A has been validated.
0 = Frame to receive buffer A has not been validated.

RXBCIF — Valid Frame Reception to Receive Buffer B Complete Interrupt Flag

This flag is set when a complete valid frame has been received in receive buffer B. If not masked (RXBCIE is set), a valid frame reception to receive buffer B complete interrupt is pending while this flag is set.

1 = Frame to receive buffer B has been validated.
0 = Frame to receive buffer B has not been validated.

MMCIF — MII Management Transfer Complete Interrupt Flag

This flag is set when the MII has completed a requested MII management transfer. If not masked (MMCIE is set), an MII management transfer complete interrupt is pending while this flag is set.

1 = MII management transfer completion.
0 = MII management transfer in progress or none requested.

LCIF — Late Collision Interrupt Flag

This flag is set if a collision has occurred after the collision window of 512 bit times while in half-duplex mode. If not masked (LCIE is set), a late collision interrupt is pending while this flag is set.

1 = Late collision during transmission.
0 = No collisions after collision window.

ECIF — Excessive Collision Interrupt Flag

This flag is set if the total number of collisions has exceeded the maximum retransmission count of 15 while in half-duplex mode. The frame is discarded and another START command must be invoked to commence a new transmission. If not masked (ECIE is set), an excessive collision interrupt is pending while this flag is set.

1 = Number of collisions exceeds 15.
0 = Number of collisions is 15 or less.

TXCIF — Frame Transmission Complete Interrupt Flag

This flag is set when a transmit frame has been completed. If not masked (TXCIE is set), a frame transmission complete interrupt is pending while this flag is set.

1 = Frame transmission has been completed.
0 = Frame transmission has not been confirmed.

11.3.2.8 Interrupt Mask (IMASK)

The interrupt mask register provides control over which possible interrupt events are allowed to generate an interrupt. If the corresponding bits in both IEVENT and IMASK registers are set, an interrupt is generated and remains active until a 1 is written to the IEVENT bit or a 0 is written to the IMASK bit.

Module Base + $C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RFCIE	0	BREIE	RXEIE	RXAOIE	RXBOIE	RXACIE	RXBCIE	MMCIE	0	LCIE	ECIE	0	0	TXCIE	0
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-9. Interrupt Mask (IMASK)

Read: Anytime.

Write: Anytime.

RFCIE — Receive Flow Control Interrupt Enable

1 = A receive flow control event causes a receive flow control interrupt request.
0 = No interrupt request is generated by this event.

BREIE — Babbling Receive Error Interrupt Enable

1 = A babbling receive error event causes a babbling receive error interrupt request.
0 = No interrupt request is generated by this event.

RXEIE — Receive Error Interrupt Enable

1 = A receive error event causes a receive error interrupt request.
0 = No interrupt request is generated by this event.

RXAOIE — Receive Buffer A Overrun Interrupt Enable

1 = A receive buffer A overrun event causes a receive buffer A overrun interrupt request.
0 = No interrupt request is generated by this event.

RXBOIE — Receive Buffer B Overrun Interrupt Enable

1 = A receive buffer B overrun event causes a receive buffer B overrun interrupt request.

0 = No interrupt request is generated by this event.

RXACIE — Valid Frame Reception to Receive Buffer A Complete Interrupt Enable
1 = A valid frame reception to receive buffer A complete event causes a valid frame reception to receive buffer A complete interrupt request.
0 = No interrupt request is generated by this event.

RXBCIE — Valid Frame Reception to Receive Buffer B Complete Interrupt Enable
1 = A valid frame reception to receive buffer B complete event causes a valid frame reception to receive buffer B complete interrupt request.
0 = No interrupt request is generated by this event.

MMCIE — MII Management Transfer Complete Interrupt Enable
1 = An MII management transfer complete event causes an MII management transfer complete interrupt request.
0 = No interrupt request is generated by this event.

LCIE — Late Collision Interrupt Enable
1 = A late collision event causes a late collision interrupt request.
0 = No interrupt request is generated by this event.

ECIE — Excessive Collision Interrupt Enable
1 = An excessive collision event causes an excessive collision interrupt request.
0 = No interrupt request is generated by this event.

TXCIE — Frame Transmission Complete Interrupt Enable
1 = A frame transmission complete event causes a frame transmission complete interrupt request.
0 = No interrupt request is generated by this event.

11.3.2.9 Software Reset (SWRST)

Module Base + $E

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	MACRST							
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-10. Software Reset (SWRST)

Read: Anytime.

Write: Anytime, but the user must not change this bit while BUSY is set.

MACRST — MAC Software Reset

Writing a 0 to this bit has no effect. This bit always reads 0.

When this bit is set, the equivalent of a hardware reset is performed but it is local to the EMAC. The EMAC logic is initialized and all EMAC registers take their reset values. Any transmission/reception currently in progress is abruptly aborted.

1 = EMAC is reset.
0 = Normal operation.

11.3.2.10 MII Management PHY Address (MPADR)

Module Base + $10

	7	6	5	4	3	2	1	0
R	0	0	0	PADDR				
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-11. MII Management PHY Address (MPADR)

Read: Anytime.

Write: Anytime, but the user must not change this field while BUSY is set.

PADDR — MII Management PHY Address

This field specifies 1 of up to 32 attached PHY devices. The default address for the internal PHY after reset is 0, but can be changed by writing the PHY address register.

11.3.2.11 MII Management Register Address (MRADR)

Module Base + $11

	7	6	5	4	3	2	1	0
R	0	0	0	RADDR				
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-12. MII Management Register Address (MRADR)

Read: Anytime.

Write: Anytime, but the user must not change this field while BUSY is set.

RADDR — MII Management Register Address

This field selects 1 of the 32 MII registers of a PHY device to be accessed. The default address for the internal PHY after reset is 0, but can be changed by writing the PHY address register.

11.3.2.12 MII Management Write Data (MWDATA)

Module Base + $12

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WDATA															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-13. MII Management Write Data (MWDATA)

Read: Anytime.

Write: Anytime, but the user must not change this field while BUSY is set.

WDATA — MII Management Write Data

This data field contains the write data to be used when sourcing a write MII management frame.

11.3.2.13 MII Management Read Data (MRDATA)

Module Base + $14

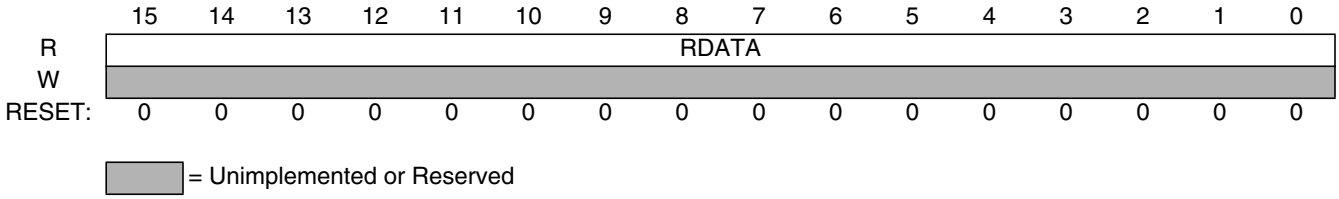


Figure 11-14. MII Management Read Data (MRDATA)

Read: Anytime.

Write: Never.

RDATA — MII Management Read Data

This data field contains the read data resulting from a read MII management frame. RDATA is valid only when MMCIF is set after a valid read frame operation.

11.3.2.14 MII Management Command and Status (MCMST)

Module Base + $16

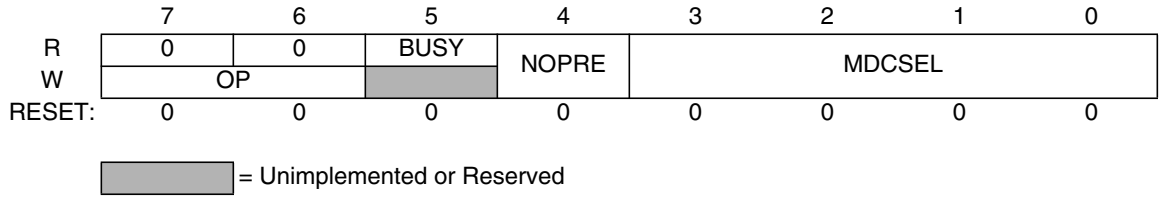


Figure 11-15. MII Management Command and Status (MCMST)

Read: Anytime.

Write: See each bit description.

OP — Operation Code

This field must be programmed to 10 to generate a valid read frame operation. See Section 11.4.6.2, “Read Operation.” This field must be programmed to 01 to generate a valid write frame operation. See Section 11.4.6.3, “Write Operation.” A programmed value of 00, 11, or any value programmed while BUSY is set is ignored. While programming MCMST, the OP write is ignored if MDCSEL is a 0 value. This field always reads 00.

Table 11-5. MII Management Frame Operation

BUSY	OP	Operation
1	xx	Ignore
0	00	Ignore
0	01	Write
0	10	Read
0	11	Ignore

BUSY — Operation in Progress

This read-only status bit indicates MII management activity. BUSY is asserted after a valid OP write and is cleared when the MMCIF flag is set.

1 = MII is busy (operation in progress).
0 = MII is idle (ready for operation).

NOPRE — No Preamble

Any value written while BUSY is set is ignored. The IEEE 802.3 standard allows the preamble to be dropped if the attached PHY does not require it. While this bit is set, a preamble is not prepended to the MII management frame.

1 = No preamble is sent.
0 = 32-bit preamble is sent.

MDCSEL — Management Clock Rate Select

Any value programmed while BUSY bit is set is ignored. This field controls the frequency of the MII management data clock (MDC) relative to the IP bus clock. MDC toggles only during a valid MII management transaction. While MDC is not active, it remains low. Any nonzero value results in an MDC frequency given by the following formula:

MDC frequency = Bus clock frequency / (2 * MDCSEL)

The MDCSEL field must be programmed with a value to provide an MDC frequency of less-than or equal-to 2.5 MHz to be compliant with the IEEE MII specification. The MDCSEL must be set to a nonzero value in order to source a read or write MII management frame.

Table 11-6. Programming Examples for MDCSEL

IP Bus Clock Frequency	MDCSEL	MDC Frequency
20 MHz	0x4	2.5 MHz
25 MHz	0x5	2.5 MHz
33 MHz	0x7	2.36 MHz
40 MHz	0x8	2.5 MHz
50 MHz	0xA	2.5 MHz

11.3.2.15 Ethernet Buffer Configuration (BUFCFG)

Module Base + $18

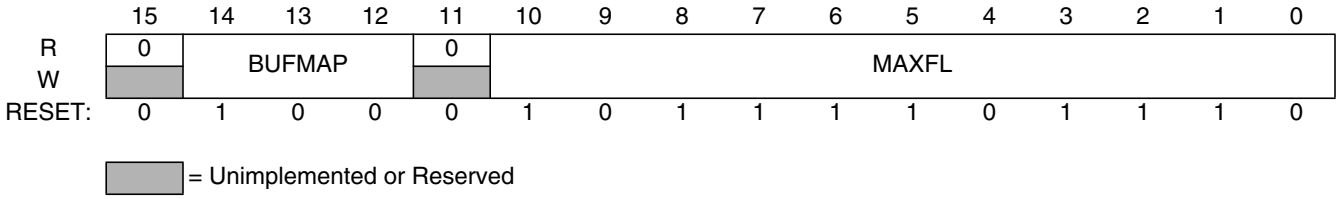


Figure 11-16. Ethernet Buffer Configuration (BUFCFG)

Read: Anytime.

Write: See each field description.

BUFMAP — Buffer Size and Starting Address Mapping

This 3-bit field can be written once after a hardware or software reset and only while EMACE is clear. Any write to this field while EMACE is set is ignored.

This field specifies the buffer size and the base address within system RAM for the receive and transmit Ethernet buffers. Table 11-7 shows the mapping configuration for the system RAM. The starting address of the system RAM depends on its position within the on-chip system memory map.

Table 11-7. Buffer Mapping Configuration on System RAM

BUFMAP	System RAM Starting Address	RX Buffer A Size (Bytes)	RX Buffer A Address Space	RX Buffer B size (Bytes)	RX Buffer B Address Space	TX Buffer Start Address
0	0x0000	128	0x0000 - 0x007F	128	0x0080 - 0x00FF	0x0100
1	0x0000	256	0x0000 - 0x00FF	256	0x0100 - 0x01FF	0x0200
2	0x0000	512	0x0000 - 0x01FF	512	0x0200 - 0x03FF	0x0400
3	0x0000	1K	0x0000 - 0x03FF	1K	0x0400 - 0x07FF	0x0800
4	0x0000	1.5K	0x0000 - 0x05FF	1.5K	0x0600 - 0x0BFF	0x0C00
5 – 7	Reserved					

MAXFL — Receive Maximum Frame Length

This 11-bit field can be written anytime, but the user must not change this field while EMACE is set.

The 11-bit field specifies the maximum receive frame length in bytes. Receive frames exceeding MAXFL causes the BREIF event bit to set and an interrupt occurs if the BREIE is also set. Written values equal-to or less-than 0x040 (64 decimal) use the minimum of 0x040. Written values equal-to or greater-than 0x5EE (1518 decimal) use the maximum of 0x5EE.

11.3.2.16 Receive A End-of-Frame Pointer (RXAEFP)

Module Base + $1A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	RXAEFP										
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-17. Receive A End-of-Frame Pointer (RXAEFP)

Read: Anytime.

Write: Never.

The receive A end-of-frame pointer (RXAEFP) 11-bit field specifies the address offset of the last byte was that written to the receive buffer A. The base address of receive buffer A is determined by BUFMAP. RXAEFP is valid only while RXACIF is set.

11.3.2.17 Receive B End-of-Frame Pointer (RXBEFP)

Module Base + $1C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	RXBEFP										
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-18. Receive B End-of-Frame Pointer (RXBEFP)

Read: Anytime.

Write: Never.

The receive B end-of-frame pointer (RXBEFP) 11-bit field specifies the address offset of the last byte that was written to the receive buffer B. The base address of receive buffer B is determined by BUFMAP. RXBEFP is valid only while RXBCIF is set.

11.3.2.18 Transmit End-of-Frame Pointer (TXEFP)

Module Base + $1E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TXEFP										
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-19. Transmit End-of-Frame Pointer (TXEFP)

Read: Anytime.

Write: Anytime, but the user must not change this field while TXACT is set.

The transmit end-of-frame pointer (TXEFP) 11-bit field specifies the address offset of the last frame byte that was stored in the transmit buffer. The base address of the transmit buffer is determined by BUFMAP.

11.3.2.19 Multicast Hash Table (MCHASH)

The multicast hash table (MCHASH) contains the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Section 11.4.2.1.4, “Multicast Filter,” explains how to configure this register.

Read: Anytime.

Write: Anytime, but the user must not change this field while EMACE is set.

Module Base + $20

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCHASH[63:48]															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Module Base + $22

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCHASH[47:32]															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Module Base + $24

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCHASH[31:16]															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Module Base + $26

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCHASH[15:0]															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-20. Multicast Hash Table (MCHASH)

MCHASH — Multicast Hash Table Index

11.3.2.20 MAC Unicast Address (MACAD)

The MAC unicast address (MACAD) registers contain the 48-bit address used for identifying an exact match in the address recognition process by comparing the 48-bit address with the destination address field of unicast receive frames. In addition, the 48-bit address is used in the 6-byte source address field while transmitting PAUSE frames.

These registers are write-once after reset. The Ethernet MAC address must be a unique number for each device. Ethernet MAC addresses are assigned by the IEEE Standards Association (IEEE-SA). This address

is normally stored in nonvolatile memory and copied to the MAC address register during initialization by user software.

Read: Anytime.

Write: Once after a hardware or software reset, but the user must not change this field while EMACE is set.

Module Base + $28

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACAD[47:32]															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Module Base + $2A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACAD[31:16]															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Module Base + $2C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACAD[15:0]															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-21. MAC Address (MACAD)

MACAD — MAC Unicast Address

11.3.2.21 Miscellaneous (EMISC)

The miscellaneous (EMISC) register provides visibility of internal counters used by the EMAC.

Read: Anytime.

Write: Anytime for the INDEX field and never for the MISC field.

Module Base + $2E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INDEX			0	0	MISC										
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-22. Miscellaneous (EMISC)

INDEX — Miscellaneous Index

This 3-bit field selects different counters to be read in the MISC field.

Table 11-8. Miscellaneous Fields

Index	Field	Unit
0-2	None	Read 0s
3	TXBYT	Bytes
4	BSLOT	Slot time
5	RETX	Retransmissions
6	RANDOM	N/A
7	Reserved	Reserved

TXBYT — Transmit Frame Byte Counter

This11-bit read-only field indicates the number of bytes of the current frame that have been read from the transmit buffer by the EMAC transmitter. This register does not include transmitted pad data that is added to frames if less than the minimum amount of data is transmitted nor the FCS data that is appended to the end of transmit frames. While sending pause frames with the PAUSE command, this register is ignored.

BSLOT — Backoff Slot Time Counter

This 10-bit read-only field indicates the number of slot times (512 bit times) in progress during the backoff delay. This counter clears at the end of backoff delay, which is set by the random algorithm. The MISC[10] bit reads 0.

RETX — Retransmission Counter

This 4-bit read-only field indicates the current retransmission count if retransmission takes place due to collision. The MISC[10:4] bits read 0.

RANDOM — Backoff Random Number

This10-bit read-only random number is generated for use by the backoff logic. The value returned when reading this field is random if the transmitter is enabled. The MISC[10] bit reads 0.

11.4 Functional Description

The EMAC provides a 10/100 Mbps Ethernet media access control (MAC) function and is designed to connect to a PHY device supporting MII. The EMAC is an 802.3 compliant Ethernet controller specifically optimized for 8-/16-bit embedded processors. The main components of the EMAC are the receiver, transmitter, MAC flow control, MII management, and receive and transmit Ethernet buffer interfaces.

11.4.1 Ethernet Frame

In an Ethernet network, information is received or transmitted in the form of a frame. The frame format used for Ethernet consists of preamble (PA), start frame delimiter (SFD), destination address (DA), source address (SA), type/length field, data field, and frame check sequence (FCS). See Table 11-9.

Table 11-9. Ethernet Frame Structure

Preamble	Start Frame Delimiter	Destination Address	Source Address	Type/ Length	Data	Frame Check Sequence
7 bytes	1 bytes	6 bytes	6 bytes	2 bytes	46 to 1500 bytes	4 bytes

The frame length is defined to be 64 bytes at minimum and 1518 bytes at maximum, excluding the preamble and SFD. Transmission and reception of each byte of data is performed one nibble at a time across the MII interface with the order of nibble as shown in Figure 11-23

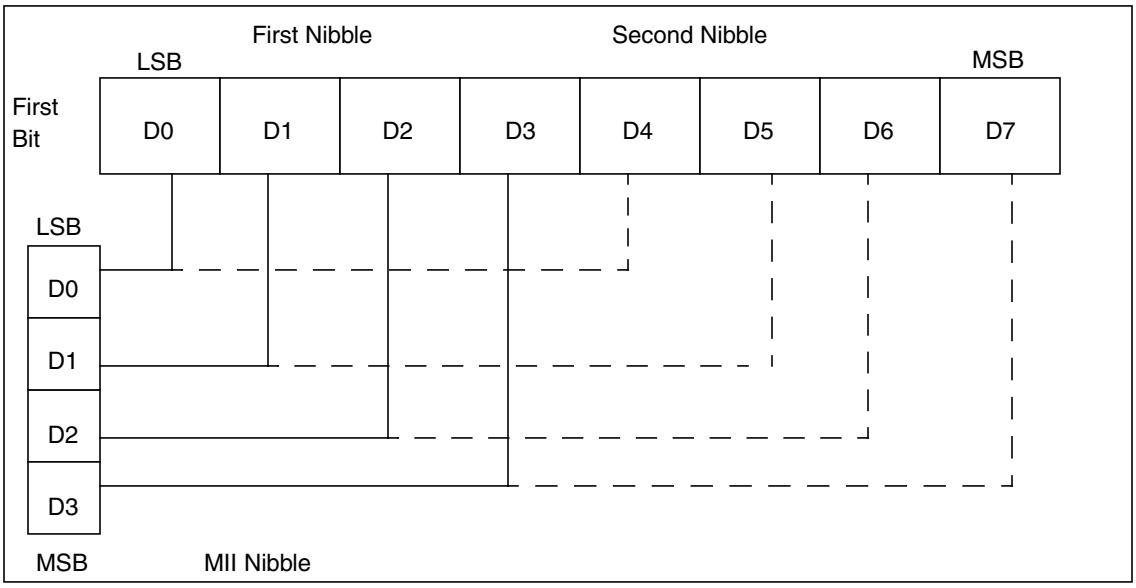


Figure 11-23. MII Nibble/Byte-to-Byte/Nibble Mapping

11.4.1.1 Preamble and SFD

The preamble is a 56-bit field that consists of a fixed pattern of alternating 1s and 0s.

1010 1010 1010 1010 1010 1010 1010 1010 1010 1010 1010 1010 1010 1010

The left-most 1 value represents the byte LSB and the right-most 0 value represents the byte MSB.

The SFD field is the sequence 10101011 and immediately follows the preamble pattern. The preamble and SFD are used to allow the Ethernet interfaces on the network to synchronize themselves with the incoming data stream before the data fields arrive.

The EMAC does not require any preamble before the SFD byte. If a preamble is detected, the preamble must be a valid preamble pattern until the SFD or else the frame is dropped.

11.4.1.2 Address Fields

Each frame contains two address fields: the destination address field and the source address field, in that order. The destination address field specifies the network node(s) for which the frame is intended. The source address field specifies the network node that sent the frame.

A 48-bit address is written as 12 hexadecimal digits with the digits paired in groups of two, representing a byte of information. The byte order of transmission on the network is from the most- to least-significant byte. The transmission order within the byte, however, is starting from the least-significant bit (LSB) of the byte through the most-significant bit (MSB). For example, an Ethernet address that is written as the

hexadecimal string F0-4E-77-8A-35-1D is equivalent to the following sequence of bits, sent over the network from left to right:

0000 1111 0111 0010 1110 1110 0101 0001 1010 1100 1011 1000

If the LSB of the most-significant byte of the destination address field is a 0, the address field contains an individual (unicast) address. If the LSB is a 1, the address field contains a group (multicast) address that identifies none, one or more, or all network nodes connected. There is a special case of the multicast address known as the broadcast address, which is the 48-bit address of all 1s.

11.4.1.3 Type/Length Field

This 16-bit field takes one of two meanings depending on its numeric value and is transmitted and received with the high order byte first.

If the value in this field is numerically equal-to or less-than the maximum data size in bytes of 1500 decimal (0x05DC hex), the field is being used as a length field. In this case, the value in the field indicates the number of bytes contained in the subsequent data field of the frame. When receiving this type frame, a compare of the value in the type/length field is made to the actual number of bytes received in the data field of the frame and an error is reported if there is not an exact match.

If the value in this field is numerically greater-than or equal-to 1536 decimal (0x0600 hex), the field is being used as a type field. In this case, the hexadecimal identifier in the field is used to indicate the type of protocol data being carried in the data field of the frame. For example, the hexadecimal value of 0x0800 has been assigned as the identifier for the Internet protocol (IP). When receiving this type frame, no comparison of the value in the type/length field is made to the actual number of bytes received in the data field of the frame.

If the value in this field is between 1501 and 1535, this frame is invalid but is not automatically rejected. When receiving this type frame, no comparison of the value in the type/length field is made to the actual number of bytes received in the data field of the frame.

When transmitting, if the length of the data field is less than the minimum required for the data field of the frame, bytes of pad data are automatically added at the end of the data field but before the FCS field to make the data field meet the minimum length requirement. The content of pad data is all 0s. Upon reception of a frame, the length field stored in the receive buffer is used to determine the length of valid data in the data field, and any pad data is discarded by software.

11.4.1.4 Data Field

This field must contain a minimum of 46 bytes of data, and may range up to a maximum of 1500 bytes of data.

11.4.1.5 Frame Check Sequence

This 32-bit field contains the value that is used to check the integrity of the various bits in the frame fields excluding the preamble and SFD. This value is computed using the cyclic redundancy check (CRC), which is a polynomial calculated using the contents of the destination address, source address, type/length, and data fields.

While the frame is being generated by the transmitting network node, the CRC value is simultaneously being calculated. The 32 bits of the CRC value are placed in the FCS field while the frame is sent. The X^{31} coefficient of the CRC polynomial is sent as the first bit of the field and the X^0 coefficient as the last bit.

The CRC is calculated again by the receiving network node while the frame is read in. The result of this second calculation is compared with the value sent in the FCS field by the originating network node. If the two values are identical, the receiving network node is provided with a high level of assurance that no errors have occurred during transmission over the network.

11.4.1.6 End-of-Frame Delimiter

The end-of-frame (EOF) delimiter is indicated by the de-assertion of the MII_TXEN signal for data on MII_TXD. This informs the PHY to send a special EOF symbol on the Ethernet. For data on the MII_RXD signal, the de-assertion of MII_RXDV constitutes an end-of-frame delimiter.

11.4.1.7 Interframe

The interframe period provides an observation window for a specified amount of time during which no data activity occurs on the MII. The de-assertion of MII_RXDV on the receive path and the de-assertion of MII_TXEN in the transmit path indicate the absence of data activity.

11.4.2 Receiver

The EMAC receiver is designed to work with very little intervention from the CPU.

When the EMAC is enabled, it immediately starts processing receive frames as long as one of the receive buffer complete interrupt flags is clear. If both RXACIF and RXBCIF are clear, receive buffer A is used first. If one flag is set, reception occurs on the buffer with the cleared flag. If both flags are set, no data is stored to the received buffers.

When MII_RXDV asserts, the receiver first checks for a valid PA/SFD sequence. If the PA/SFD is valid, it is stripped and the frame is processed by the receiver. If a valid PA/SFD is not found, the frame is ignored.

The receiver checks for at least one byte matching the SFD (10101011). Zero or more PA bytes sent before the SFD byte are acceptable, but if an invalid PA is detected prior to the SFD byte, the frame is ignored.

Following the SFD, the EMAC converts the nibble stream to a byte data stream. See Figure 11-23.

After the first six bytes of the frame have been received, the EMAC performs address recognition on the frame. See Section 11.4.2.1, “Address Recognition.” If address recognition rejects the frame, the receiver goes idle, the receive buffer stops receiving data, and the receive end-of-frame pointer is invalid. If address recognition accepts the frame, the receive buffer continues to receive data.

After the first 14 bytes of the frame have been received, the EMAC performs type/length recognition on the frame. See Section 11.4.2.2, “Type/Length Recognition.” If type/length recognition rejects the frame, the receiver goes idle, the receive buffer stops receiving data, and the receive end-of-frame pointer is invalid. If type/length recognition accepts the frame, the receive buffer continues to receive data.

If a receive frame length is less than 64 bytes, the receive frame is considered a fragment and is dropped. Most fragments are the result of a collision, and as such are a completely normal and expected event on an Ethernet.

If a receive frame length exceeds 1518, the receive frame is considered too long and is an error. The RXEIF bit becomes set and if not masked (RXEIE set to 1), the EMAC generates the receive error interrupt.

If MII_RXER is asserted during reception, indicating a media error, the RXEIF bit becomes set and if not masked (RXEIE set to 1), the EMAC generates the receive error interrupt.

If the type/length field is less-than or equal-to 1500 (but greater than 46), a length mismatch error occurs if the receive frame data field length does not match the length specified in the type/length field. If the type/length field is less than or equal to 46, a length mismatch error occurs if the receive frame data field length is not 46. If a length mismatch error occurs, the RXEIF bit becomes set and if not masked (RXEIE set to 1), the EMAC generates the receive error interrupt.

The EMAC receiver automatically calculates a 4-byte frame check sequence from the receive frame and compares it with the CRC data suffixed to the receive frame. If a CRC error occurs, the RXEIF bit becomes set and if not masked (RXEIE set to 1), the EMAC generates the receive error interrupt.

After the end of frame delimiter, the received frame is truncated to the nearest byte boundary. If there is an extra nibble, this dribble nibble is discarded. If the CRC value in the received frame is correct, the frame is accepted as valid. If the CRC value is incorrect and there is a dribble nibble, an alignment error has occurred and the RXEIF bit becomes set and if not masked (RXEIE set to 1), the EMAC generates the receive error interrupt.

Frames that exceed the MAXFL field in byte length are not truncated. However, the BREIF bit becomes set and if not masked (BREIE set to 1), the EMAC generates the babbling receive error interrupt.

If a receive frame exceeds the receive buffer size, the corresponding receive overrun error flag is set. In the overrun error event, the frame is not accepted and neither the corresponding complete flag nor the receive error flag is set. A babbling receive error condition is ignored if it occurs after a buffer overrun event and thus BREIF does not become set.

Upon MAC flow control PAUSE frame reception, the RFCIF bit in the IEVENT register is asserted. If not masked (RFCIE is set), a receive flow control interrupt is pending while this flag is set. PAUSE frames may be accepted even if both receive buffers are full. The frame is accepted and the RFCIF flag is set only if there no receive error.

When frame reception to either receive buffer A or receive buffer B is complete, the corresponding receive buffer complete flag is set, the value in the corresponding receive end-of-frame pointer is valid. If not masked (corresponding receive buffer complete interrupt enable is set to 1), the EMAC generates the corresponding receive buffer complete interrupt. The receiver buffer complete flag is set only if there are no receive errors and the frame has not been accepted as a MAC flow control PAUSE frame. If both receiver buffer complete flags are set, new receive frames are dropped until one of the complete flags is cleared.

The receiver receives back-to-back frames with a minimum spacing of at least 96 bit times. If an interframe gap between receive frames is less than 96 bit times, the latter frame is not guaranteed to be accepted by the receiver.

11.4.2.1 Address Recognition

The EMAC executes filtering by using the destination address of a receive frame and eliminates a frame that does not satisfy a given condition. See Figure 11-24 for the address recognition algorithm.

11.4.2.1.1 Promiscuous Mode

If the PROM bit is set, promiscuous mode is enabled and all frames are accepted regardless of address. The PROM bit does not affect any other filtering in the EMAC.

11.4.2.1.2 Unicast Filter

Unless the PROM bit is set, the 48-bit MAC address (MACAD) is compared for an exact match with the destination address of a receive frame with an individual address (group bit is 0). If the unicast address of the receive frame matches MACAD, the frame is accepted; otherwise, it is rejected.

11.4.2.1.3 Broadcast Filter

A broadcast frame (48-bit address of all 1s) is accepted if the BCREJ bit is 0 and rejected if the BCREJ bit is 1 unless the PROM bit is set.

11.4.2.1.4 Multicast Filter

If the CONMC bit is set to 0, all multicast frames are accepted. If the CONMC bit is 1 and the PROM bit is 0, only multicast frames with the hash table match are accepted.

The hash table algorithm operates as follows. The 48-bit destination address is mapped into one of 64 bits, which are represented by the 64 bits stored in MCHASH. This mapping is performed by passing the 48-bit address through the 32-bit CRC generator and selecting the 6 most significant bits of the CRC-encoded result to generate a number between 0 and 63. If the CRC generator selects a bit that is set in the hash table, the frame is accepted; otherwise, it is rejected. To set the hash table, the CRC of a multicast address must be calculated and the corresponding bit must be set in advance.

11.4.2.1.5 PAUSE Destination Address

If the EMAC is in full-duplex mode and the RFCE bit is set, the receiver detects incoming PAUSE frames. A PAUSE frame has a 48-bit destination multicast address of 01-80-C2-00-00-01 or unique DA. Upon detection of a PAUSE frame, the frame is temporarily accepted for further type/length recognition.

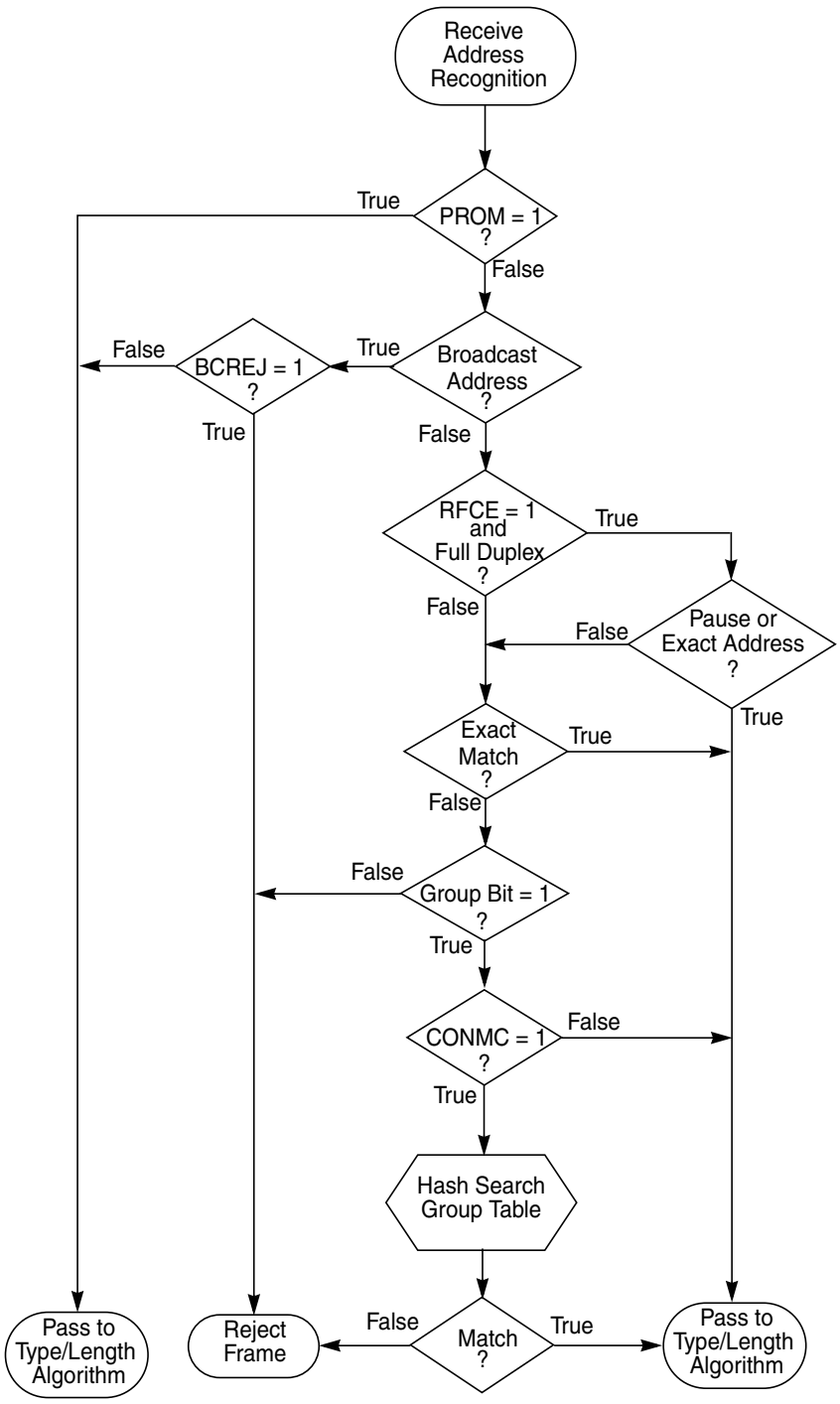


Figure 11-24. Receive Address Recognition Algorithm

11.4.2.2 Type/Length Recognition

The EMAC executes filtering by using the type/length field of a receive frame and rejects a frame that does not meet acceptance criteria. See Figure 11-25 for the type/length recognition algorithm.

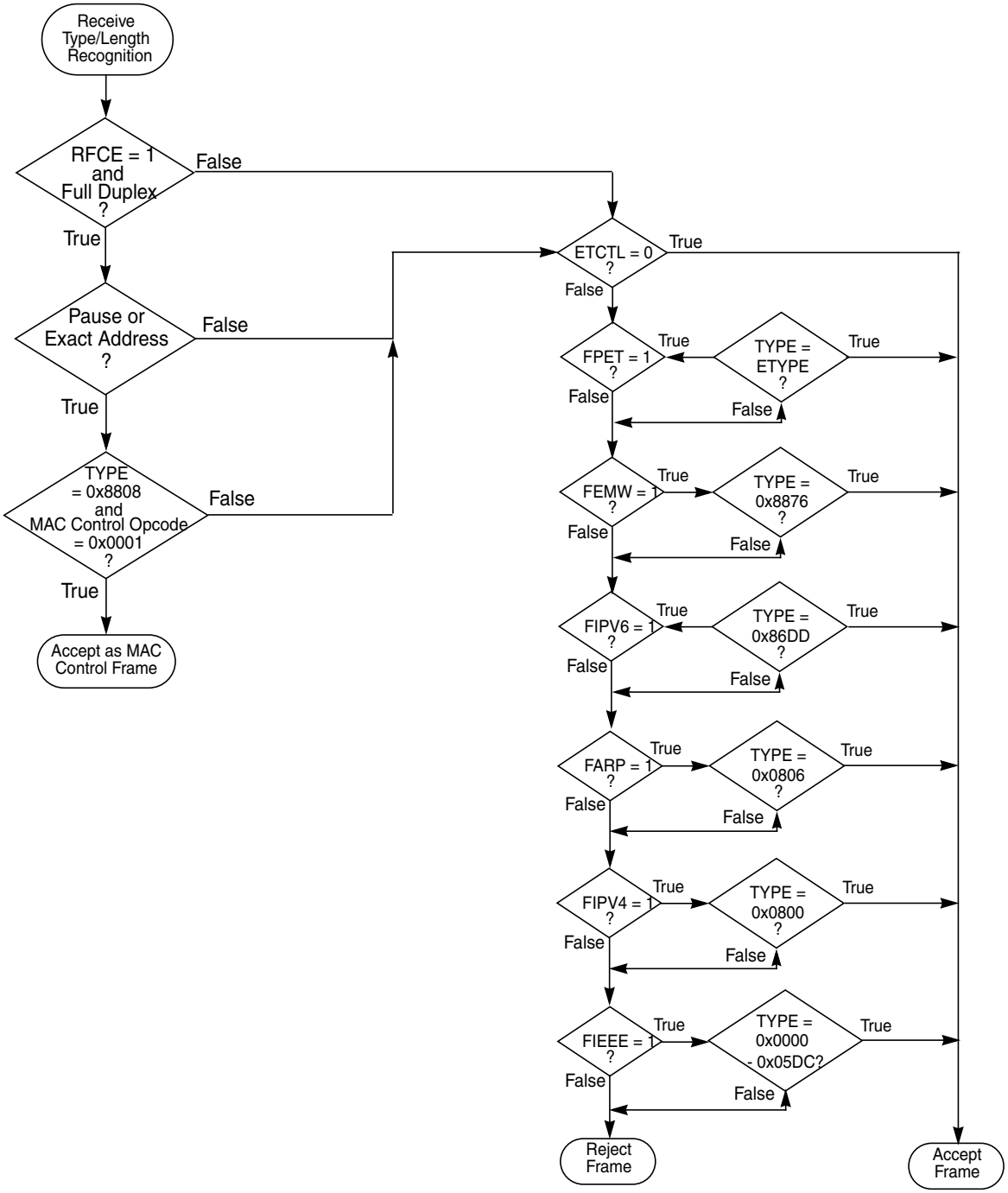


Figure 11-25. Receive Type/Length Recognition Algorithm

11.4.2.2.1 Ethertype Filter

While any of the ETCTL register bits are set, the Ethertype filter is enabled to reject frames that are not standard Ethernet protocols. In this case, the collection of set bits determines which Ethertypes are accepted; all other Ethertypes are rejected. If all bits of the ETCTL register are clear, Ethertype filtering is

not performed. If the FPET bit is set, frames with Ethertype matching the value in the ETYPE register are accepted. If the FEMW bit is set, frames with Emware Ethertype are accepted. If the FIPV6 bit is set, frames with Internet protocol version 6 Ethertype are accepted. If the FARP bit is set, frames with address resolution protocol Ethertype are accepted. If the FIPV4 bit is set, frames with Internet protocol Ethertype are accepted. If the FIEEE bit is set, frames with valid IEEE 802.3 length Ethertype are accepted.

11.4.2.2.2 PAUSE MAC Control Type

If the EMAC is in full-duplex mode and the RFCE bit is set, the receiver detects incoming PAUSE frames. After a PAUSE destination address has been detected, the type/length field is checked looking for a type value of 0x8808. If the type/length field does not contain this value, the frame is rejected; otherwise, the MAC control function reads the frame looking for MAC control operation codes carried in the data field. For more information on the function of MAC control, see Section 11.4.5.1, “MAC Flow Control.”

11.4.3 Transmitter

The transmit data, which the user must write to the transmit buffer, consists of the destination address followed by the source address, type/length field, and the data field. The EMAC transmitter automatically appends the preamble, SFD, and FCS necessary for a transmit frame. It also automatically appends pad data to extend the data length to the 46-byte minimum frame length.

After a frame has been written to the transmit buffer and the corresponding transmit end-of-frame pointer has been initialized, the EMAC transmitter is ready to transmit on the network. When a START command is executed by writing to the TCMD field, the EMAC transmit logic asserts MII_TXEN and starts transmitting the preamble sequence, the start frame delimiter, and then the frame information from the transmit buffer. The EMAC transmits bytes least significant nibble first.

In half-duplex operation, the EMAC transmitter defers transmission if the network is busy and data transmission is started after the interframe gap interval. In full-duplex mode, the carrier sense is ignored, and data transmission is started after the interframe gap interval. See Section 11.4.3.1, “Interframe Gap,” and Section 11.4.3.2, “Deferring.”

If a collision occurs within the collision window of 64 bytes during transmission of the frame (half-duplex mode), the EMAC transmitter follows the specified backoff procedures and attempts to retransmit the frame until the retry limit threshold is reached. See Section 11.4.3.3, “Collision Detection and Backoff.”

If the carrier sense is lost during transmission and no collision is detected in the frame, the EMAC sets the CSLF status bit. The frame is transmitted normally and no retries are performed as a result of a CSLF error.

After the transmit frame is complete, the TXCIF bits are set. If not masked (TXCIE set to 1), the EMAC generates the frame transmission complete interrupt.

11.4.3.1 Interframe Gap

When the network becomes idle, a network node waits for a brief period called the interframe gap (IFG), and then transmits its frame. This is provided to allow a brief recovery time between frame reception for the Ethernet interfaces. The minimum interframe gap time for back-to-back transmission is 96 bit times.

11.4.3.2 Deferring

In half-duplex mode, if there is a carrier (the network is busy), the network node continues to listen until the carrier ceases (network is idle). This is known as deferring to the passing traffic. As soon as the network becomes idle (which includes waiting for the interframe gap interval), the network node may begin transmitting a frame. The transmitter waits for the carrier sense to be negated for 60 bit times and then begins transmit after another 36 bit times.

11.4.3.3 Collision Detection and Backoff

The collision detection and backoff feature is a normal part of the operation of Ethernet 802.3 MAC protocol, and results in fast and automatic rescheduling of transmissions. This feature enables independent network nodes to compete for network access in a fair manner. It provides a way for network nodes to automatically adjust their behavior in response to the load of the network.

11.4.3.3.1 Collision Window

The collision window period is set to 64 byte times (512 bit times) starting after the SFD. If a collision occurs within the collision window period, the retry process is initiated. If a late collision occurs (that is, a collision after the collision window period), no retransmission is performed, the LCIF bit sets to 1, the transmit retry counter is cleared, and transmission is aborted. If not masked (LCIE is set), the EMAC generates a late collision interrupt. Due to latency associated with synchronizing the MII_COL signal, assertions in the last three MII_TXCLK cycles of a normally completed transmission (during the FCS) are ignored and a collision event is not recognized.

11.4.3.3.2 Jam Period

If a collision is detected anytime during transmission, the EMAC transmitter continues to transmit 32 bits of data (called the collision enforcement jam signal) so that other devices on the Ethernet network, including the offending transmitter, can detect the collision. If the collision is detected very early in the frame transmission, the EMAC transmitter continues sending until it has completed the preamble of the frame, after which it sends the 32 bits of jam data. If the collision is detected during the FCS, up to and including the transfer of the last nibble of FCS data, the 32 bit jam is still sent.

11.4.3.3.3 Backoff Generator

After a collision occurs within the collision window period, the delay time that the EMAC transmitter waits before attempting to retransmit the frame data is set at a multiple of the 512-bit Ethernet slot time. The amount of total backoff delay is calculated by multiplying the slot time by a pseudo-randomly chosen integer.

The backoff algorithm uses the following formula to determine the integer ***r***, which is used to multiply the slot time and generate a backoff delay.

$$0 \leq r < 2^k$$

The exponent *k* is assigned a value that is equal to either the number of transmission attempts or the number 10, whichever is less. The coefficient *r* of the slot time is an integer randomly selected from a range of integers from 0 to one less than the value of two to the exponent *k*. Table 11-10 shows the range of backoff times that may occur on a channel.

Table 11-10. Backoff Times

Collision on Attempt Number	Range of Random Numbers	Range of Backoff Times (10 Mbps)	Range of Backoff Times (100 Mbps)
1	0...1	0...51.2 µs	0...5.1 µs
2	0...3	0...153.6 µs	0...15.4 µs
3	0...7	0...358.4 µs	0...35.8 µs
4	0...15	0...768.0 µs	0...76.8 µs
5	0...31	0...1.59 ms	0...158.7 µs
6	0...63	0...3.23 ms	0...322.6 µs
7	0...127	0...6.50 ms	0...650.2 µs
8	0...255	0...13.06 ms	0...1.31 ms
9	0...511	0...26.16 ms	0...2.62 ms
10-15	0...1023	0...52.38 ms	0...5.24 ms
16	N/A	Discard frame	Discard frame

The RANDOM field in the EMISC register contains the 10-bit random number generated by the random generator in the backoff logic. If the SSB bit is set, the transmitter backoff logic forces a single slot backoff time of 512 bit times instead of following the random backoff algorithm.

11.4.3.3.4 Retry Counter

The EMAC transmitter has a retry counter, RETX, that counts the number of collisions within the collision window period that occur while attempting to send a single frame. The retry counter increments by 1 after each collision and resets to 0 when each frame is successfully transmitted. RETX is held at 0 when TXACT is clear. The EMAC transmitter attempts to retransmit up to 15 times. If a collision occurs when RETX is 15, the excessive collision interrupt flag (ECIF) is set to 1, the entire transmit frame is discarded, and the retry counters resets to 0. If not masked (ECIE set to 1), the EMAC generates the excessive collision interrupt. The TXACT bit in TXCTS will be asserted for the entire duration of the retry process. The next transmission can start as soon as TXACT is clear.

11.4.4 Ethernet Buffers

There are two receive Ethernet buffers and one transmit Ethernet buffer allocated within the system RAM. The size and starting address for each buffer is configured by the BUFMAP field in the BUFCFG register. See Section 11.3.2.15, “Ethernet Buffer Configuration (BUFCFG).”

11.4.4.1 Receive Ethernet Buffer

Upon reception, the receive Ethernet buffers store the destination address (DA), the source address (SA), the type/length field, the data field, and the frame check sequence (FCS). If the receiver has data to put into a receive buffer and the receive buffers are full, the receive frame is dropped. If the length of the receive frame is larger than the receive buffer, the corresponding receive buffer overrun flag bit is set to 1, and if

not masked (corresponding receive buffer overrun interrupt enable bit is set to 1), the EMAC generates an overrun interrupt. In the receive buffer overrun event, buffer storage is halted and adjacent storage buffers are not corrupted.

11.4.4.2 Transmit Ethernet Buffer

Only the destination address (DA), the source address (SA), the type/length field, and the data field must be stored in the transmit Ethernet buffer. The transmitter automatically appends the frame check sequence. It also automatically appends pad data to extend the data length to 46 bytes if the data length of the frame written to the transmit buffer is less than the minimum data length. The value of the transmit end-of-frame pointer must correspond to the last byte in the data field byte, not including pad data.

11.4.5 Full-Duplex Operation

The IEEE 802.3x standard defines a second mode of operation, called *full duplex*, that bypasses the CSMA/CD (carrier sense multiple access/collision detect) protocol. The CSMA/CD protocol is *half duplex,* meaning two or more network nodes share a common transmission medium implying that a network node may either transmit data, or receive data, but never both at the same time. Full-duplex mode allows exactly two network nodes to simultaneously exchange data over a point-to-point link that provides independent transmit and receive paths. Because each network node can simultaneously transmit and receive data, the aggregate throughput of the link is effectively doubled. Because there is no contention for a shared medium, collisions cannot occur and the CSMA/CD protocol is unnecessary.

11.4.5.1 MAC Flow Control

Full-duplex mode includes an optional flow control mechanism for real-time control and manipulation of the frame transmission and reception process. This mechanism allows a receiving node that is becoming congested to request the sending node to stop sending frames for a selected short period of time. This is performed through the use of a PAUSE frame. If the congestion is relieved before the requested wait has expired, a second PAUSE frame with a zero time-to-wait value can be sent to request resumption of transmission.

MAC control frames are identified by the exclusive assigned type value of 0x8808 (hex). They contain operational codes (opcodes) in the first two bytes of the data field. The MAC control opcode field for a PAUSE command is 0x0001 (hex). The next two bytes of the data field are the MAC control parameters field, which is a 16-bit value that specifies the duration of the PAUSE event in units of 512 bit times. Valid values are 0x0000 to 0xFFFF (hex). If an additional PAUSE frame arrives before the current PAUSE time has expired, its parameter replaces the current PAUSE time, so a PAUSE frame with parameter 0 allows traffic to resume immediately. A 42-byte reserved field (transmitted as all 0s) is required to pad the length of the PAUSE frame to the minimum Ethernet frame size. The destination address of the PAUSE frame must be set to the globally assigned multicast address 01-80-C2-00-00-01 (hex) or to the unique DA. This multicast address has been reserved by the IEEE 802.3 standard for use in MAC control PAUSE frames.

Table 11-11. Ethernet PAUSE Frame Structure

Preamble	SFD	DA	SA	Type/Length	MAC Control Opcode	MAC Control Parameters	Reserved	FCS
7 bytes	1 byte	6 bytes = (01-80-C2-00-00-01) or unique DA	6 bytes	2 bytes = MAC Control (88-08)	2 bytes = (00-01)	2 bytes = (00-00 to FF-FF)	42 bytes = all 0s	4 bytes

11.4.5.2 Hardware Generated PAUSE Control Frame Transmission

As long as there is no transmission in progress and EMAC is in full-duplex mode, a PAUSE command can be launched by writing to the TCMD field. The EMAC builds a PAUSE frame according to Table 11-11 using the parameter value in the PTIME field and then transmits this frame. The DA field is set to 01-80-C2-00-00-01 (hex).

When the transmitted PAUSE frame is complete, the TXCIF bit is set. If not masked (TXCIE set to 1) the EMAC generates the frame transmission complete interrupt.

NOTE

To transmit a MAC flow control pause frame using a unique DA, the user must construct a valid pause frame in the transmit buffer, configure the TXEFP register, and issue a START command. However, whenever issuing pause frames in this manner, the command is suspended while the received pause time counter value (PTIME) is nonzero and is sent after the time has expired.

11.4.5.3 PAUSE Control Frame Reception

While RFCE bit is set, the receiver detects PAUSE frames in full-duplex mode. Upon PAUSE frame detection, the RFCIF bit in the IEVENT register is asserted and the EMAC transmitter stops transmitting data frames for a duration after the current transmission is complete. The duration is given by the PAUSE time parameter in the received frame. If not masked (RFCIE is set), a receive flow control interrupt is pending while this flag is set. Although the reception of a PAUSE frame stops transmission of frames initiated with a START command, it does not prevent transmission of PAUSE control frames. PAUSE frames may be accepted even if both receive buffers are full.

11.4.6 MII Management

MII management access to a PHY is via the MII_MDC and MII_MDIO signals. MII_MDC has a maximum clock rate of 2.5 MHz. MII_MDIO is bidirectional and can be connected to 32 external devices or the internal PHY. When using the internal PHY, the MII_MDC and MII_MDIO signals are not visible to the user.

11.4.6.1 Frame Structure

A transmitted MII management frame uses the MII_MDIO and MII_MDC pins. This frame has the following format:

<pre><st><op><phyad><regad><ta><data><idle>

11.4.6.1.1 PRE (Preamble)

The *preamble* (pre) consists of 32 contiguous logic 1 bits on MII_MDIO with 32 corresponding cycles on MII_MDC to provide the PHY with a pattern that it can use to establish synchronization. The preamble is optional as determined by NOPRE.

11.4.6.1.2 ST (Start of Frame)

The *start of frame* (st) is indicated by a <01> pattern. This pattern ensures transitions from the default logic 1 line state to 0 and then returns to 1.

11.4.6.1.3 OP (Operation Code)

The o*peration code* (op) for a read instruction is <10>. For a write operation, the operation code is <01>.

11.4.6.1.4 PHYAD (PHY Address)

The *PHY address* (phyad) is a 5-bit field, allowing up to 32 unique PHY addresses. The first address bit transmitted is the MSB of the address.

11.4.6.1.5 REGAD (Register Address)

The *register address* (regad) is a 5-bit field, allowing 32 individual registers to be addresses within each PHY. The first register bit transmitted is the MSB of the address.

11.4.6.1.6 TA (Turnaround)

The *turnaround* (ta) field is a two bit time spacing between the register address field and the data field of an MII management frame to avoid contention on the MII_MDIO signal during a read operation. For a read transaction, both the MAC and the PHY remain in a high impedance state for the first bit time of the *turnaround.* The PHY drives a 0 bit during the second bit time of the *turnaround* of a read transaction. During a write transaction, the MAC drives a 1 bit for the first bit time of the *turnaround* and a 0 bit for the second bit time of the *turnaround*.

11.4.6.1.7 DATA (Data)

The *data* (data) field is 16 bits wide. The first data bit transmitted and received is the MSB of the data.

11.4.6.1.8 IDLE (IDLE Condition)

During *idle condition* (idle), MII_MDIO is in the high impedance state.

11.4.6.2 Read Operation

To perform a read operation through MII management, the OP field in MCMST must be written to 10 while the BUSY bit is clear. The PADDR field in MPADR indicates which PHY device is addressed and the RADDR in MRADR indicates which 16-bit register is read from the PHY device. The MII management creates an MII management frame and serially shifts it out to the PHY through the MII_MDIO pin. After the turnaround field, the PHY serially shifts the register data from the PHY to the EMAC through the MII_MDIO pin. After the read MII management frame operation has completed, the BUSY bit clears, the MRDATA register is updated, and the MMCIF bit in IEVENT is set. If not masked (MMCIE in IMASK is set), an MII management transfer complete interrupt is pending while this flag is set.

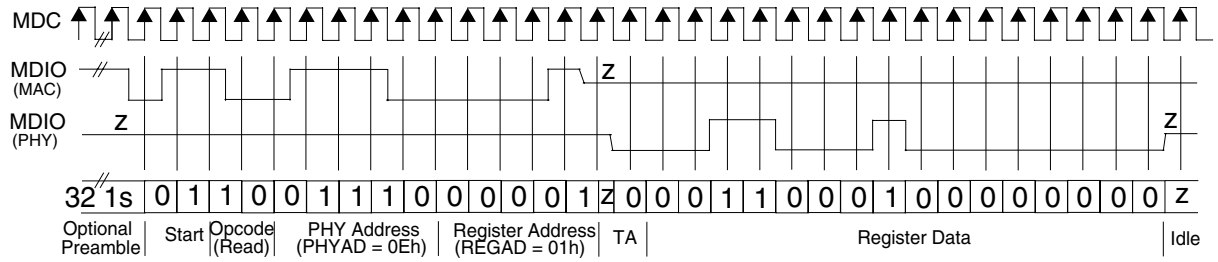


Figure 11-26. Typical MDC/MDIO Read Operation

11.4.6.3 Write Operation

To perform a write operation through MII management, the OP field in MCMST must be written to 01 while the BUSY bit is clear. The PADDR field in MPADR indicates which PHY device is addressed and the RADDR bit in MRADR indicates which 16-bit register is read from the PHY device. The MII management creates an MII management frame and serially shifts it out to the PHY through the MII_MDIO pin. After the turnaround field, the MWDATA register is serially shifted to the PHY through the MII_MDIO pin. After the write MII management frame operation has completed, the BUSY bit is cleared and the MMCIF bit in IEVENT is set. If not masked (MMCIE in IMASK is set), an MII management transfer complete interrupt is pending while this flag is set.

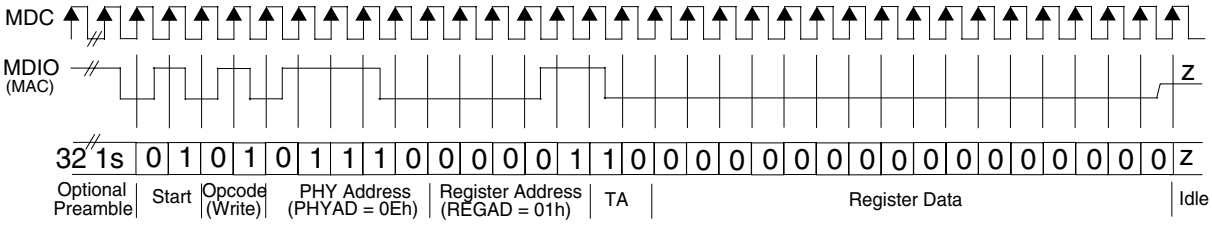


Figure 11-27. Typical MDC/MDIO Write Operation

11.4.7 Loopback

The MII transmit data stream is internally looped back as an MII receive data stream if the MLB bit is set. The MII_TXCLK and MII_RXCLK are internally driven from the system clock. MII_RXD is driven from MII_TXD. MII_RXDV is driven from MII_TXEN. MII_RXER is driven from MII_TXER. The

MII_COL, MII_CRS, MII_MDC, and MII_MDIO signals are disabled. During loopback, the MII to external and internal PHYs are disabled. Loopback mode requires that the user set the FDX bit to configure for full-duplex mode. Loopback connects the outs to the ins and relies on the unidirectional nature of full duplex to transfer data in parallel. The bidirectional nature of half duplex does not allow the RX to accept transmit data without using some kind of intermediate storage buffer.

11.4.8 Software Reset

The EMAC provides a software reset capability. When the MACRST bit is set, all registers are reset to their default values. The EMACE bit is cleared. The receiver and transmitter are initialized. Any reception and/or transmission currently in progress is abruptly aborted.

11.4.9 Interrupts

When an interrupt event occurs, a bit is set in the IEVENT register. Note that bits in the IEVENT register are set by the event and remain set until cleared by software. If a bit in the IEVENT register is set and the corresponding bit is set in the IMASK register, the corresponding interrupt signal asserts. Individual interrupts are cleared by software by writing a 1 to the corresponding bit in the IEVENT register.

The interrupt sources are listed in Table 11-12.

Table 11-12. Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Receive Flow Control (RFCIF)	I bit	IMASK (RFCIE)
Babbling Receive Error (BREIF)	I bit	IMASK (BREIE)
Receive Error (RXEIF)	I bit	IMASK (RXEIE)
Receive Buffer A Overrun (RXAOIF)	I bit	IMASK (RXAOIE)
Receive Buffer B Overrun (RXBOIF)	I bit	IMASK (RXBOIE)
Receive Buffer A Complete (RXACIF)	I bit	IMASK (RXACIE)
Receive Buffer B Complete (RXBCIF)	I bit	IMASK (RXBCIE)
MII Management Transfer Complete (MMCIF)	I bit	IMASK (MMCIE)
Late Collision (LCIF)	I bit	IMASK (LCIE)
Excessive Collision (ECIF)	I bit	IMASK (ECIE)
Frame Transmission Complete (TXCIF)	I bit	IMASK (TXCIE)

11.4.10 Debug and Stop

During system debug (freeze) mode, the EMAC functions normally.When the system enters low-power stop mode, the EMAC is immediately disabled. Any receive in progress is dropped and any PAUSE timeout is cleared. The user must not enter low-power stop mode while TXACT or BUSY is set.

Chapter 12
Ethernet Physical Transceiver (EPHYV2)

12.1 Introduction

The Ethernet physical transceiver (EPHY) is an IEEE 802.3 compliant 10BASE-T/100BASE-TX Ethernet PHY transceiver. The EPHY module supports both the medium-independent interface (MII) and the MII management interface. The EPHY requires a 25-MHz crystal for its basic operation.

12.1.1 Features

- IEEE 802.3 compliant
- Full-/half-duplex support in all modes
- Medium-independent interface (MII), which has these characteristics:
  - Capable of supporting both 10 Mbps and 100 Mbps data rates
  - Data and delimiters are synchronous to clock references
  - Provides independent four-bit wide transmit and receive data paths
  - Provides a simple management interface
- Supports auto-negotiation
- Auto-negotiation next page ability
- Single RJ45 connection
- 1:1 common transformer
- Baseline wander correction
- Digital adaptive equalization
- Integrated wave-shaping circuitry
- Far-end fault detect
- MDC rates up to 25 MHz
- Supports MDIO preamble suppression
- Jumbo packet
- 2.5 V CMOS
- 2.5 V MII interface
- 125 MHz clock generator and timing recovery
- Loopback modes

12.1.2 Block Diagram

.

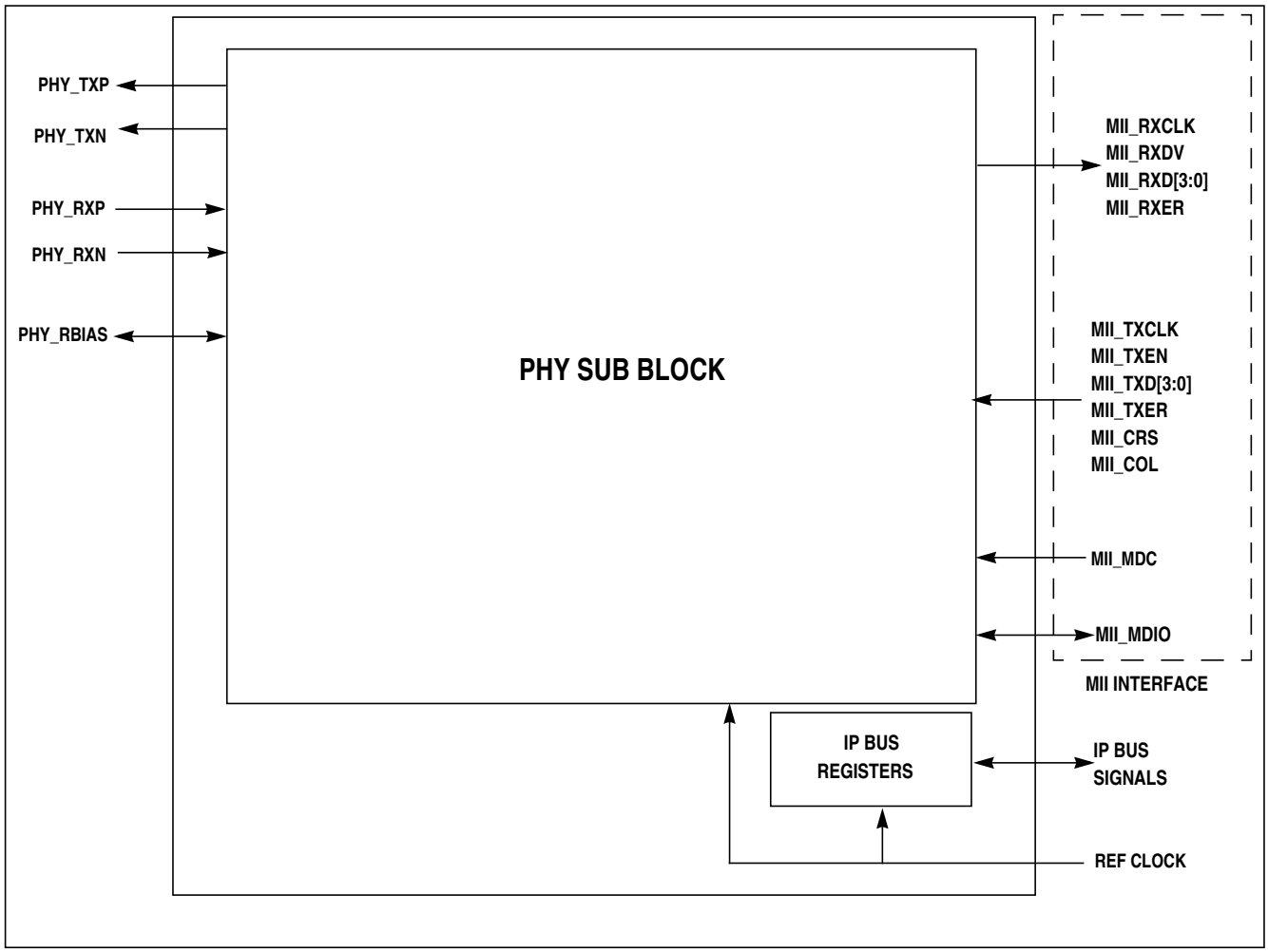


Figure 12-1. Ethernet Physical Transceiver (EPHY) Block Diagram

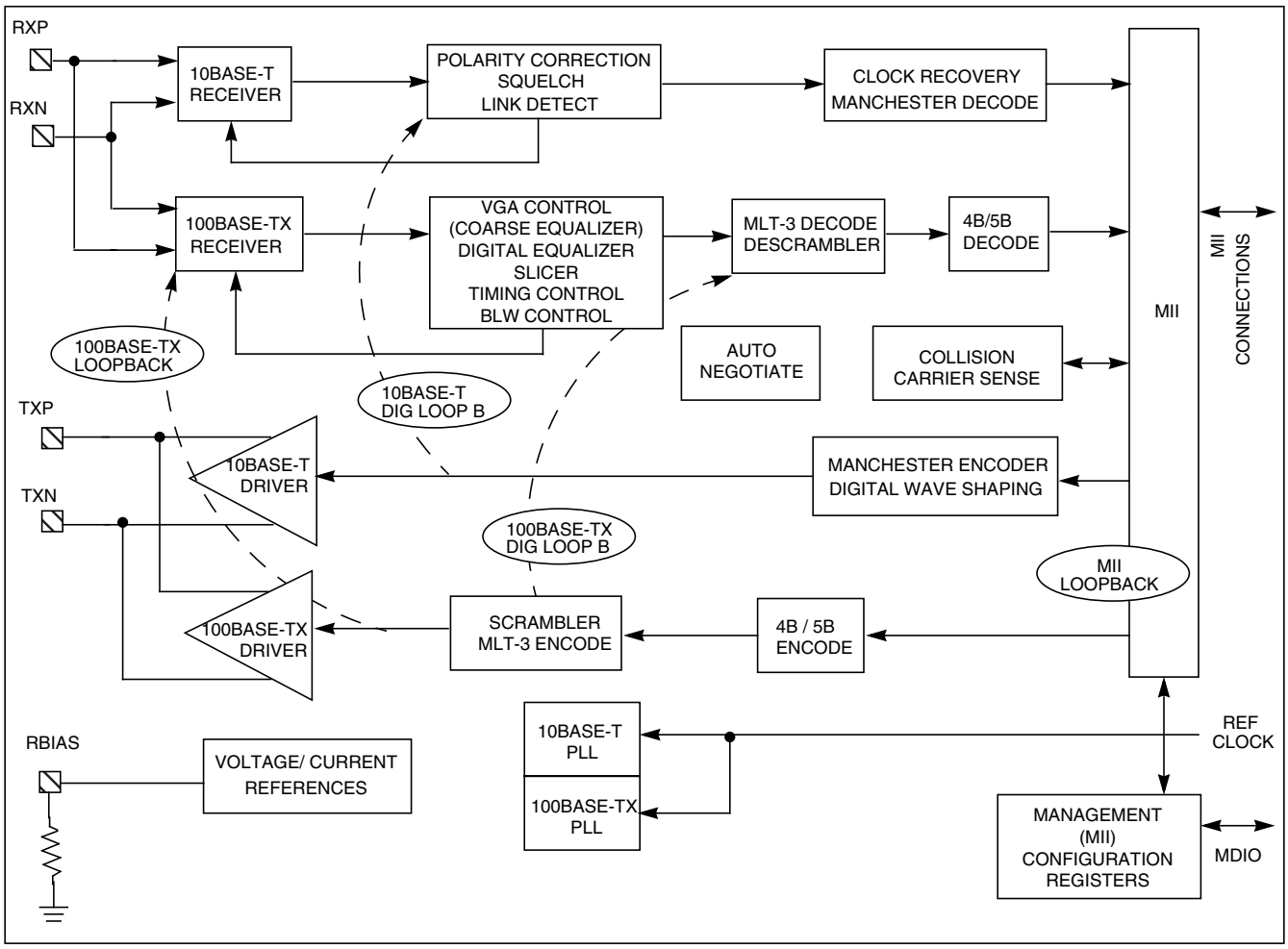


Figure 12-2. PHY Sub Block Diagram

12.2 External Signal Descriptions

This section contains the EPHY external pin descriptions.

12.2.1 PHY_TXP — EPHY Twisted Pair Output +

Ethernet twisted-pair output pin. This pin is high-impedance out of reset.

12.2.2 PHY_TXN — EPHY Twisted Pair Output –

Ethernet twisted-pair output pin. This pin is high-impedance out of reset.

12.2.3 PHY_RXP — EPHY Twisted Pair Input +

Ethernet twisted-pair input pin. This pin is high-impedance out of reset.

12.2.4 PHY_RXN — EPHY Twisted Pair Input –

Ethernet twisted-pair input pin. This pin is high-impedance out of reset.

12.2.5 PHY_RBIAS — EPHY Bias Control Resistor

Connect a 1.0% external resistor, RBIAS (see Electrical Characteristics chapter), between the PHY_RBIAS pin and analog ground. Place this resistor as near to the chip pin as possible. Stray capacitance must be kept to less than 10 pF (>50 pF will cause instability). No high-speed signals are permitted in the region of RBIAS.

12.2.6 PHY_VDDRX, PHY_VSSRX — Power Supply Pins for EPHY Receiver

Power is supplied to the EPHY receiver through PHY_VDDRX and PHY_VSSRX. This 2.5 V supply is derived from the internal voltage regulator. There is no static load on those pins allowed. The internal voltage regulator is turned off, if V_{DDR} is tied to ground.

12.2.7 PHY_VDDTX, PHY_VSSTX — Power Supply Pins for EPHY Transmitter

External power is supplied to the EPHY transmitter through PHY_VDDTX and PHY_VSSTX. This 2.5 V supply is derived from the internal voltage regulator. There is no static load on those pins allowed. The internal voltage regulator is turned off, if V_{DDR} is tied to ground.

12.2.8 PHY_VDDA, PHY_VSSA — Power Supply Pins for EPHY Analog

Power is supplied to the EPHY PLLs through PHY_VDDA and PHY_VSSA. This 2.5 V supply is derived from the internal voltage regulator. There is no static load on those pins allowed. The internal voltage regulator is turned off, if V_{DDR} is tied to ground.

12.2.9 COLLED — Collision LED

Flashes in half-duplex mode when a collision occurs on the network if EPHYCTL0 LEDEN bit is set.

12.2.10 DUPLED — Duplex LED

Indicates the duplex of the link, which can be full-duplex or half-duplex if EPHYCTL0 LEDEN bit is set.

12.2.11 SPDLED — Speed LED

Indicates the speed of a link, which can be 10 Mbps or 100 Mbps if EPHYCTL0 LEDEN bit is set.

12.2.12 LNKLED — Link LED

Indicates whether a link is established with another network device if EPHYCTL0 LEDEN bit is set.

12.2.13 ACTLEC — Activity LED

Flashes when data is received by the device if EPHYCTL0 LEDEN bit is set.

12.3 Memory Map and Register Descriptions

This section provides a detailed description of all registers accessible in the EPHY.

12.3.1 Module Memory Map

Table 12-1 gives an overview of all registers in the EPHY memory map. The EPHY occupies 48 bytes in the memory space. The register address results from the addition of *base address* and *address offset*. The *base address* is determined at the MCU level. The *address offset* is defined at the module level.

Table 12-1. EPHY Module Memory Map

Address Offset	Use	Access
$__00	Ethernet Physical Transceiver Control Register 0 (EPHYCTL0)	R/W
$__01	Ethernet Physical Transceiver Control Register 1 (EPHYCTL1)	R/W
$__02	Ethernet Physical Transceiver Status Register (EPHYSR)	R/W
$__03	RESERVED	R

12.3.2 Register Descriptions

12.3.2.1 Ethernet Physical Transceiver Control Register 0 (EPHYCTL0)

Module Base + $0

	7	6	5	4	3	2	1	0
R W	EPHYEN	ANDIS	DIS100	DIS10	LEDEN	EPHYWAI	0	EPHYIEN
RESET:	0	1	1	1	0	0	0	0

= Unimplemented or Reserved

Figure 12-3. Ethernet Physical Transceiver Control Register 0 (EPHYCTL0)

Read: Anytime

Write: See each bit description

EPHYEN — EPHY Enable

This bit can be written anytime.

1 = Enables EPHY

0 = Disables EPHY

ANDIS — Auto Negotiation Disable

This bit can be written anytime, but the value is latched in the ANE bit of the MII PHY control register (MII address 0.12) only when the EPHYEN bit transitions from 0 to 1.

1 = Auto negotiation is disabled after start-up. A 0 is latched in the ANE bit of the MII PHY control register (MII address 0.12), and upon completion of the start-up delay ($t_{Start-up}$), the EPHY will bypass auto-negotiation. The mode of operation will be determined by the manual setting of MII registers.

0 = Auto negotiation is enabled after start-up. A 1 is latched in the ANE bit of the MII PHY control register (MII address 0.12), and upon completion of the start-up delay ($t_{Start-up}$), the EPHY will enter auto-negotiation. The mode of operation will be automatically determined.

DIS100 — Disable 100 BASE-TX PLL

This bit can be written anytime. Allows user to power down the clock generation PLL for 100BASE-TX clocks.

1 = Disables 100BASE-TX PLL

0 = 100BASE-TX PLL state determined by EPHY operation mode

DIS10 — Disable 10BASE-T PLL

This bit can be written anytime. Allows user to power down the clock generation PLL for 10BASE-T clocks.

1 = Disables 10BASE-T PLL

0 = 10 BASE-T PLL state determined by EPHY operation mode

LEDEN — LED Drive Enable

This bit can be written anytime.

1 = Enables the EPHY to drive LED signals.

0 = Disables the EPHY to drive LED signals.

EPHYWAI — EPHY Module Stops While in Wait

This bit can be written anytime.

1 = Disables the EPHY module while the MCU is in wait mode. EPHY interrupts cannot be used to bring the MCU out of wait.

0 = Allows the EPHY module to continue running during wait.

EPHYIEN — EPHY Interrupt Enable

This bit can be written anytime.

1 = Enables EPHY module interrupts

0 = Disables EPHY module interrupts

12.3.2.2 Ethernet Physical Transceiver Control Register 1 (EPHYCTL1)

Module Base + $1

	7	6	5	4	3	2	1	0
R	0	0	0	PHYADD4	PHYADD3	PHYADD2	PHYADD1	PHYADD0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-4. Ethernet Physical Transceiver Control Register 1 (EPHYCTL1)

Read: Anytime

Write: See each bit description

PHYADD[4:0] — EPHY Address for MII Requests

These bits can be written anytime, but the EPHY address is latched to the MII PHY address register (MII address 21.4:0) only when the EPHYEN bit transitions from 0 to 1. PHYADD4 is the MSB of the of the EPHY address.

12.3.2.3 Ethernet Physical Transceiver Status Register (EPHYSR)

Module Base + $2

	7	6	5	4	3	2	1	0
R	0	0	100DIS	10DIS	0	0	0	EPHYIF
W								
RESET:	0	0	1	1	0	0	0	0

= Unimplemented or Reserved

Figure 12-5. Ethernet Physical Transceiver Status Register (EPHYSR)

Read: Anytime

Write: See bit descriptions

100DIS — EPHY Port 100BASE-TX mode status

This bit is not writable — read only. Output to indicate EPHY port Base100-TX mode status.
1 = EPHY port 100BASE-TX disabled
0 = EPHY port 100BASE-TX enabled

10DIS — EPHY Port 10BASE-T mode status

This bit is not writable. Output to indicate EPHY port 10BASE-T mode status.
1 = EPHY port 10BASE-T disabled
0 = EPHY port 10BASE-T enabled

EPHYIF — EPHY Interrupt Flag

EPHYIF indicates that interrupt conditions have occurred. To clear the interrupt flag, write a 1 to this bit after reading the interrupt control register via the MII management interface.
1 = EPHY interrupt has occurred
0 = EPHY interrupt has not occurred

12.3.3 MII Registers

Table 12-2 gives an overview of all registers in the EPHY that are accessible via the MII management interface. These registers are not part of the MCU memory map.

Table 12-2. MII Registers

Address		Use	Access
0	%00000	Control Register	Read/Write
1	%00001	Status Register	Read/Write[4]
2	%00010	PHY Identification Register 1	Read/Write[4]
3	%00011	PHY Identification Register 2	Read/Write[4]
4	%00100	Auto-Negotiation Advertisement Register	Read/Write
5	%00101	Auto-Negotiation Link Partner Ability Register	Read/Write[4]
6	%00110	Auto-Negotiation Expansion Register	Read/Write[4]
7	%00111	Auto-Negotiation Next Page Transmit	Read/Write
8	%01000	RESERVED	Read/Write[1]
9	%01001	RESERVED	Read/Write[1]
10	%01010	RESERVED	Read/Write[1]
11	%01011	RESERVED	Read/Write[1]
12	%01100	RESERVED	Read/Write[1]
13	%01101	RESERVED	Read/Write[1]
14	%01110	RESERVED	Read/Write[1]
15	%01111	RESERVED	Read/Write[1]
16	%10000	Interrupt Control Register	Read/Write
17	%10001	Proprietary Status Register	Read/Write[4]
18	%10010	Proprietary Control Register	Read/Write

1. Always read $00
2. Writable only in special modes (test_mode = 1)
4. Write has no effect.

NOTE

Bit notation for MII registers is: Bit 20.15 refers to MII register address 20 and bit number 15.

12.3.3.1 EPHY Control Register

MII Register Address 0 (%00000)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RESET	LOOP BACK	DATA RATE	ANE	PDWN	ISOL	RAN	DPLX	COL TEST	0	0	0	0	0	0	0
W																
RESET:	0	0	1	X	0	0	0	1	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-6. Control Register

Read: Anytime

Write: Anytime

RESET — EPHY Reset

Resetting a port is accomplished by setting this bit to 1.

1 = The PHY will reset the port's status and registers to the default values. The PHY will also reset the PHY to its initial state. After the reset is complete, the PHY clears this bit automatically. The reset process will be completed within 1.3 ms of this bit being set. While the preamble is suppressed, the management interface must not receive an ST within three MDC clock cycles following a software reset.

0 = No effect

LOOPBACK — Digital Loopback Mode

Determines Digital Loopback Mode

1 = Enables digital loopback mode. Port will be placed in loopback mode. Loopback mode will allow the TXD data to be sent to the RXD data circuitry within 512 bit times. The PHY will be isolated from the medium (no transmit or receive to the medium allowed) and the MII_COL signal will remain de-asserted, unless this bit is set.

0 = Disables digital loopback mode

DATARATE — Speed Selection

The link speed will be selected either through the auto-negotiation process or by manual speed selection. ANE allows manual speed selection while it is set to 0. While auto-negotiation is enabled, DATARATE can be read or written but its value is not required to reflect speed of the link.

1 = While auto-negotiation is disabled, selects 100 Mbps operation

0 = While auto-negotiation is disabled, selects 10 Mbps operation

ANE — Auto-Negotiation Enable

The ANE bit determines whether the A/N process is enabled. When auto-negotiation is disabled, DATARATE and DPLX determine the link configuration. While auto-negotiation is enabled, bits DATARATE and DPLX do not affect the link.

1 = Enables auto-negotiation

0 = Disables auto-negotiation

PDWN — Power Down

When this bit is set, the port is placed in a low power consumption mode.

1 = Port is placed in a low power consumption mode. Normal operation will be allowed within 0.5 s after PDWN and ISOL are changed to 0. During a transition to power-down mode (or if already in power down mode), the port will respond only to management function requests through the MI interface. All other port operations will be disabled. When power-down mode is exited, all register values are maintained. The port will start its operation based on the register values.

0 = Normal operation

ISOL — Isolate

1 = Isolates the port's data path signals from the MII. The port will not respond to changes on MII_TXDx, MII_TXEN, and MII_TXER inputs, and it will present high impedance on MII_TXCLK, MII_RXCLK, MII_RXDV, MII_RXER, MII_RXDx, MII_COL, and MII_CRS outputs. The port will respond to management transactions while in isolate mode.

0 = Normal operation

RAN — Restart Auto-Negotiation

The RAN bit determines when the A/N process can start processing.

1 = When auto-negotiation is enabled (ANE=1), the auto-negotiation process will be restarted. After auto-negotiation indicates that it has been initialized, this bit is cleared. When bit ANE is cleared to indicate auto-negotiation is disabled, RAN must also be 0.

0 = Normal operation.

DPLX — Duplex Mode

This mode can be selected by either the auto-negotiation process or manual duplex selection. Manual duplex selection is allowed only while the auto-negotiation process is disabled (ANE=0). While the auto-negotiation process is enabled (ANE = 1), the state of DPLX has no effect on the link configuration. While loopback mode is asserted (LOOPBACK =1), the value of DPLX will have no effect on the PHY.

1 = Indicates full-duplex mode

0 = Indicates half-duplex mode

COLTEST — Collision Test

The collision test function will be enabled only if the loopback mode of operation is also selected (LOOPBACK = 1).

1 = Forces the PHY to assert the MII_COL signal within 512 bit times from the assertion of MII_TXEN and de-assert MII_COL within 4 bit times of MII_TXEN being de-asserted.

0 = Normal operation

12.3.3.2 Status Register

This register advertises the capabilities of the port to the MII.

MII Register Address 1 (%00001)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	100 T4	100X FD	100X HD	10T FD	10T HD	0	0	0	0	SUP PRE	AN COMP	REM FLT	AN ABL	LNK STST	JAB DT	EX CAP
W																
RESET:	0	1	1	1	1	0	0	0	0	1	0	0	1	0	0	1

= Unimplemented or Reserved

Figure 12-7. Status Register

Read: Anytime

Write: Writes have no effect

100T4 —100BASE-T4

1 = Indicates PHY supports 100BASE-T4 transmission

0 = Indicates the PHY does not support 100BASE-T4 transmission

This function is not implemented in the EPHY module.

100XFD —100BASE-TX Full-Duplex

1 = Indicates PHY supports 100BASE-TX full-duplex mode

0 = Indicates PHY does not support 100BASE-TX full-duplex mode

100XHD —100BASE-TX Half-Duplex
1 = Indicates the PHY supports 100BASE-TX half-duplex mode
0 = Indicates the PHY does not support 100BASE-TX half-duplex mode

10TFD —10BASE-T Full-Duplex
1 = Indicates the PHY supports 10BASE-T full-duplex mode
0 = Indicates the PHY does not support 10BASE-T full-duplex mode

10THD —10BASE-T Half-Duplex
1 = Indicates the PHY supports 10BASE-T half-duplex mode
0 = Indicates the PHY does not support 10BASE-T half-duplex mode

SUPPRE —MF Preamble Suppression
1 = Indicates that management frames are not required to contain the preamble stream
0 = Indicates that management frames are required to contain the preamble stream

ANCOMP —Auto-Negotiation Complete

To inform the management interface (MI) that it has completed processing, ANCOMP is set by the A/N process. After it has been started, the auto-negotiation process uses link code words to exchange capability information and establish the highest common denominator (HCD) for link transactions.
1 = Indicates that the auto-negotiation process has completed and that the contents of registers 4 through 7 are valid.
0 = Indicates that the auto-negotiation process has not completed and that the contents of registers 4 through 7 are not valid

REMFLT — Remote Fault

Possible remote faults (RF)
a) The link partner transmits the RF bit (5.13=1)
b) Link partner protocol is not 00001 (5.4:0)
c) Link partner advertises only T4 capability (5.9:5)
d) No common operation mode found between PHY and the link partner.

After it is set, REMFLT is cleared each time register 1 is read via the management interface. REMFLT is also cleared by a PHY reset.
1 = Indicates that a remote fault condition has been detected.
0 = No fault detected

ANABL — Auto-Negotiation Ability
1 = Indicates that PHY has auto-negotiation ability
0 = Indicates that PHY does not have auto-negotiation ability

LNKSTST — Link Status

The PHY sets this bit when it determines that a valid link has been established. The occurrence of a link failure will cause LNKSTST to be cleared. After it has been cleared, it remains cleared until it is read via the management interface.
1 = Indicates a valid link has been established
0 = Indicates a valid link has NOT been established

JABDT —Jabber Detect

After it is set, JABDT is cleared each time register 1 is read via the management interface. JABDT is also cleared by a PHY reset. For 100BASE-TX operation, this signal will always be cleared.

1 = Indicates that a jabber condition has been detected
0 = Indicates that no jabber condition has been detected

EXCAP — Extended capability

1 = Indicates that the extended register set (registers 2–31) has been implemented in the PHY.
0 = Indicates that the extended register set (registers 2–31) has NOT been implemented in the PHY

12.3.3.3 EPHY Identifier Register 1

Registers $_02 and $_03 provide the PHY identification code.

MII Register Address 2 (%00010)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PHYID															
W																
RESET:	0	0	0	0	01	0	0	0	0	0	1	0	1	1	0	0

= Unimplemented or Reserved

Figure 12-8. EPHY Identifier Register 1

Read: Anytime

Write: Writes have no effect — Read only

PHYID — PHY ID Number

Composed of bits 3:18 of the organization unique identifier (OUI).

12.3.3.4 EPHY Identifier Register 2

Registers $_02 and $_03 provide the PHY identification code.

MII Register Address 3 (%00011)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PHYID						MODELNUMBER						REVISIONNUMBER			
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

= Unimplemented or Reserved

Figure 12-9. EPHY Identifier Register 2

Read: Anytime

Write: Writes have no effect — Read only

PHYID — PHY ID number organization unique identifier. Composed of bits 15:10.

MODELNUMBER — Manufacturers model number. Composed of bits 9:4.

REVISIONNUMBER — Manufacturers revision number. Composed of bits 3:0.

12.3.3.5 Auto-Negotiate (A/N) Advertisement Register

The auto-negotiation (A/N) process requires four registers to communicate link information with its link partner: A/N advertisement register (MII register 4), A/N link partner ability register (MII register 5), A/N expansion register (MII register 6), and the A/N next page transmit register (MII register 7).

Figure 12-10 shows the contents of the A/N advertisement register. On power-up, before A/N starts, the register sets the selector field, bits 4.4:0, to 00001 to indicate that it is IEEE Standard 802.3 compliant. The technology ability fields (4.9:5) are set according to the values in the MII status register (1.15:11). The MI can set the technology ability field bits before renegotiations to allow management to auto-negotiate to an alternate common mode.

MII Register Address 4 (%00100)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NXTP	0	RFLT	0	0	FLCTL	0	TAF 100FD	TAF 100HD	TAF 10FD	TAF 10HD	SELECTORFIELD[4:0]				
W																
RESET:	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1

= Unimplemented or Reserved

Figure 12-10. Auto Negotiate Advertisement Register

Read: Anytime

Write: Never

NXTP — Next Page
1 = Capable of sending next pages
0 = Not capable of sending next pages

RFLT — Remote Fault
1 = Remote fault
0 = No remote fault

FLCTL — Flow Control
1 = Advertise implementation of the optional MAC control sublayer and pause function as specified in IEEE standard clause 31 and anex 31B of 802.3. Setting FLCTL has no effect except to set the corresponding bit in the FLP stream
0 = No MAC-based flow control

TAF100FD — 100BASE-TX Full-Duplex
1 = 100BASE-TX full -duplex capable
0 = Not 100BASE-TX full-duplex capable

TAF100HD — 100BASE-TX Half-Duplex
1 = 100BASE-TX half-duplex capable
0 = Not 100BASE-TX half-duplex capable

TAF10FD — 10BASE-T Full-Duplex
1 = 10BASE-T full-duplex capable
0 = Not 10BASE-T full-duplex capable

TAF10HD — 10BASE-T Half-Duplex
1 = 10BASE-T half-duplex capable
0 = Not 10BASE-T half-duplex capable

12.3.3.6 Auto Negotiation Link Partner Ability (Base Page)

Figure 12-11 shows the contents of the A/N link partner ability register. The register can only be read by the MI and will be written by the auto-negotiation process when it receives a link code word advertising the capabilities of the link partner. This register has a dual purpose: exchange of base page information as shown in Figure 12-11, and exchange of next page information as shown in Figure 12-12.

MII Register Address 5 (%00101) (Base Page)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NXTP	ACK	RFLT	TAF[1:0]		FCTL	TAF 100T4	TAF 100FD	TAF 100HD	TAF 10FD	TAF 10HD	SELECTORFIELD[4:0]				
W																
RESET:	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

= Unimplemented or Reserved

Figure 12-11. Auto Negotiation Link Partner Ability Register (Base Page)

Read:

Write:

NXTP — Next Page
1 = Link partner capable of sending next pages
0 = Link partner not capable of sending next pages

ACK — Acknowledge
1 = Link Partner has received link code word
0 = Link Partner has not received link code word

RFLT — Remote Fault
1 = Remote fault
0 = No remote fault

FLCTL — Flow Control
1 = Advertises implementation of the optional MAC control sublayer and pause function as specified in IEEE standard clause 31 and anex 31B of 802.3. Setting FLCTL has no effect on the PHY.
0 = No MAC-based flow control

TAF100T4 — 100BASE-T4 Full-Duplex
1 = Link partner is 100BASE-T4 capable
0 = Link partner is not 100BASE-T4 capable
This function is not implemented in the EPHY.

TAF100FD — 100BASE-TX Full-Duplex
1 = Link partner is 100BASE-TX full-duplex capable
0 = Link partner is not 100BASE-TX full-duplex capable

TAF100HD — 100BASE-TX Half-Duplex
1 = Link partner is 100BASE-TX half-duplex capable
0 = Link partner is not 100BASE-TX half-duplex capable

TAF10FD — 10BASE-T Full-Duplex
1 = Link partner is10BASE-T full-duplex capable
0 = Link partner is not 10BASE-T full-duplex capable

TAF10HD — 10BASE-T Half-Duplex
1 = Link partner is 10BASE-T half-duplex capable
0 = Link partner is not 10BASE-T half-duplex capable

12.3.3.7 Auto Negotiation Link Partner Ability (Next Page)

MII Register Address 5 (%00101) (Next Page)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NXTP	ACK	MSGP	ACK2	TGL	Message/Unformatted Code Field [10:0]										
W																
RESET:	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

= Unimplemented or Reserved

Figure 12-12. Auto Negotiation Link Partner Ability Register (Next Page)

Read: Anytime

Write: See each field description

NXTP — Next Page
1 = Additional next pages will follow
0 = Last page transmitted

ACK — Acknowledge
ACK is used to acknowledge receipt of information.
1 = Link partner has received link code word
0 = Link partner has not received link code word

MSGP — Message Page
1 = Message page
0 = Unformatted page

ACK2 — Acknowledge 2
ACK2 is used to indicate that the receiver is able to act on the information (or perform the task) defined in the message.
1 = Receiver is able to perform the task defined in the message
0 = Receiver is unable to perform the task defined in the message

TGL — Toggle
1 = Previous value of the transmitted link code word equalled 0
0 = Previous value of the transmitted link code word equalled 1

Message/Unformatted Code Field

Message code field — Predefined code fields defined in IEEE 802.3u-1995 Annex 28C
Unformatted code filed — 11-bit field containing an arbitrary value

12.3.3.8 Auto-Negotiation Expansion Register

Figure 12-13 shows the contents of the A/N expansion register. The MI process can only read this register. This register contains information about the A/N capabilities of the port's link partner and information on the status of the parallel detection mechanism.

MII Register Address 6 (%00110)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	PDFLT	LPNPA	NXTPA	PRCVD	LPANA
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

= Unimplemented or Reserved

Figure 12-13. Auto-Negotiation Expansion Register

Read: Anytime

Write: Never

PDFLT — Parallel Detection Fault

This bit is used to indicate that zero or more than one of the NLP receive link integrity test function for 100BASE-TX have indicated that the link is ready (link_status=READY) when the A/N wait timer has expired. PDFLT will be reset to 0 after a read of register 6.
1 = Parallel detection fault has occurred
0 = Parallel detection fault has not occurred

LPNPA — Link Partner Next Page Able

Bit to indicate whether the link partner has the capability of using NP.
1 = Link partner is next page able
0 = Link partner is not next page able

NXTPA — Next Page Able

This bit is used to inform the MI and the link partner whether the port has next page capabilities.
1 = The port has next page capabilities
0 = The port does not have next page capabilities

PRCVD — Page Received

Bit is used to indicate whether a new link code word has been received and stored in the A/N link partner ability register (MII register 5). PRCVD is reset to 0 after register 6 is read.
1 = Three identical and consecutive link code words have been received from link partner
0 = Three identical and consecutive link code words have not been received from link partner

LPANA — Link Partner A/N Able

Indicates whether the link partner has A/N capabilities.

1 = Link partner is A/N able

0 = Link partner is not A/N able

12.3.3.9 Auto Negotiation Next Page Transmit

Figure 12-14 shows the contents of the A/N next page transmit register. The MI writes to this register if it needs to exchange more information with the link partner. The PHY defaults to sending only a NULL message page to the link partner unless the STA overrides the values in the register. Next pages will be transmitted until the link partner has no more pages to transmit and bit 7.15 has been cleared by the STA.

MII Register Address 7 (%00111)

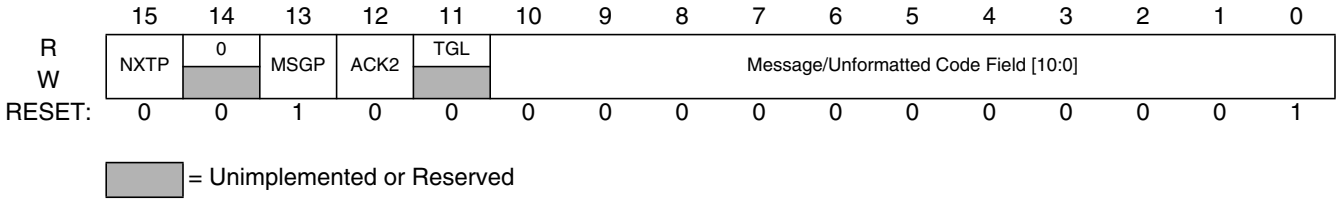


Figure 12-14. Auto Negotiation Next Page Transmit Register

Read: Anytime

Write: Never

NXTP — Next Page

1 = Additional next pages will follow

0 = Last page to transmit

MSGP — Message Page

1 = Message page

0 = Unformatted page

ACK2 — Acknowledge 2

ACK2 is used to indicate that the receiver is able to act on the information (or perform the task) defined in the message.

1 = Receiver is able to perform the task defined in the message

0 = Receiver is unable to perform the task defined in the message

TGL — Toggle

1 = Previous value of the transmitted link code word equalled 0

0 = Previous value of the transmitted link code word equalled 1

Message/Unformatted Code Field

Message code field — Predefined code fields defined in IEEE 802.3u-1995 Annex 28C

Unformatted code field — Eleven bit field containing an arbitrary value

12.3.4 PHY-Specific Registers

PHY also contains a number of registers to set its internal mode of operation. These registers can be set through the external management interface to determine capabilities such as speed, test-mode, circuit bypass mode, interrupt setting, etc. The PHY register set includes registers 16 through 29. These registers are not part of the MCU memory map.

12.3.4.1 Interrupt Control Register

MII Register Address 16 (%10000)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	ACKIE	PRIE	LCIE	ANIE	PDFIE	RFIE	JABIE	0	ACKR	PGR	LKC	ANC	PDF	RMTF	JABI
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-15. Interrupt Control Register

Read: Anytime

Write: Anytime

ACKIE — Acknowledge Bit Received Interrupt Enable
1 = Enable interrupt when the acknowledge bit is received from the link partner
0 = Disable interrupt when acknowledge bit is received

PRIE — Page Received INT Enable
1 = Enable interrupt when a new page is received
0 = Disable interrupt when a page is received

LCIE — Link Changed Enable
1 = Enable interrupt when the link status changes
0 = Disable interrupt when the link status changes

ANIE — Auto-Negotiation Changed Enable
1 = Enable interrupt when the state of the auto-negotiation state machine has changed since the last access of this register
0 = Disable interrupt when the state of the auto-negotiation state machine has changed since the last access of this register

PDFIE — Parallel Detect Fault Enable
1 = Enable interrupt on a parallel detect fault
0 = Disable interrupt on a parallel detect fault

RFIE — Remote Fault Interrupt Enable
1 = Enable interrupt on a parallel detect fault
0 = Disable interrupt on a parallel detect fault

JABIE — Jabber Interrupt Enable
1 = Enable setting interrupt on detection of a jabber condition
0 = Disable setting interrupt on detection of a jabber condition

ACKR — Acknowledge Bit Received
1 = Acknowledge bit has been received from the link partner
0 = Acknowledge bit has not been received since the last access of this register. (ACK bit 14 of the auto-negotiation link partner ability register was set by receipt of link code word)

PGR — Page Received
1 = A new page has been received from the link partner
0 = A new page has not been received from the link partner since the last access of this register (Bit 1 was set by a page received event)

LKC — Link Changed
1 = The link status has changed since the last access of this register
0 = The link status has not changed since the last access of this register. (LNK bit 14 of the proprietary status register was changed)

ANC — Auto-Negotiation Changed
1 = The auto-negotiation status has changed since the last access of this register
0 = The auto-negotiation status has not changed since the last access of this register

PDF — Parallel Detect Fault
1 = A parallel-detect fault has occurred since the last access of this register
0 = A parallel-detect fault has not been detected since the last access of this register. (Bit 4 was set by rising edge of parallel detection fault)

RMTF — Remote Fault
1 = A remote fault condition has been detected since the last access of this register
0 = A remote fault condition has not been detected since the last access of this register. (RMTF bit 4 of the status register was set by rising edge of a remote fault)

JABI — Jabber Interrupt
1 = A jabber condition has been detected since the last access of this register
0 = A jabber condition has not been detected since the last access of this register (JABD bit 1 of the status register was set by rising edge of jabber condition)

12.3.4.2 Proprietary Status Register

MII Register Address 17 (%10001)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	LNK	DPMD	SPD	0	ANNC	PRCVD	ANC MODE	0	0	PLR	0	0	0	0	0
W																
RESET:	0	1	1	1	0	0	0	(1)	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-16. Proprietary Status Register

Read: Anytime

Write:

LNK — Link Status

This is a duplicate of LNKSTAT bit 2 of the status register (1.2).

1 = Link is down
0 = Link is up

DPMD — Duplex Mode

1 = Full-duplex
0 = Half-duplex

SPD — Speed

1 = 100 Mbps
0 = 10 Mbps

ANNC — Auto-Negotiation Complete

This is a duplicate of ANCOMP bit 5 of the status register (1.5)

1 = A-N complete
0 = A-N not complete

PRCVD — Page Received

1 = Three identical and consecutive link code words have been received
0 = Three identical and consecutive link code words have not been received

ANCMODE — Auto-Negotiation (A-N) Common Operating Mode

This bit is only valid while the ANNC bit 10 is 1

1 = A common operation mode was not found
0 = A-N is complete and a common operation mode has been found

PLR — Polarity Reversed (10BASE-T)

1 = 10BASE-T receive polarity is reversed
0 = 10BASE-T receive polarity is normal

12.3.4.3 Proprietary Control Register

MII Register Address 18 (%10010)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	FE FLTD	MIILBD	0	1	JBDE	LNK TSTD	POL CORD	ALGD	ENC BYP	SCR BYP	TRD ANALB	TR TST	0	0	0
W																
RESET:	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-17. Proprietary Control Register

The miscellaneous (EMISC) register provides visibility of internal counters used by the EMAC.

Read: Anytime

Write: Anytime

FEFLTD — Far End Fault Disable
1 = Far end fault detect is disabled
0 = Far end fault detect on receive and transmit is enabled. This applies only while auto-negotiation is disabled

MIILBO — MII Loopback Disable
1 = Disable MII loopback
0 = MII transmit data is looped back to the MII receive pins

JBDE — Jabber Detect Enable (10BASE-T)
1 = Enable jabber detection
0 = Disable jabber detection

LNKTSTD — Link Test Disable (10BASE-T)
1 = Disable 10BASE-T link integrity test
0 = 10BASE-T link integrity test enabled

POLCORD — Disable Polarity Correction (10BASE-T)
1 = 10BASE-T receive polarity correction is disabled
0 = 10BASE-T receive polarity is automatically corrected

ALGD — Disable Alignment
1 = Un-aligned mode. Available only in symbol mode
0 = Aligned mode

ENCBYP — Encoder Bypass
1 = Symbol mode and bypass 4B/5B encoder and decoder
0 = Normal mode

SCRBYP — Scrambler Bypass Mode (100BASE-TX)
1 = Bypass the scrambler and de-scrambler
0 = Normal

TRDANALB — Transmit and Receive Disconnect and Analog Loopback
1 = High-impedance twisted pair transmitter. Analog loopback mode overrides and forces this bit
0 = Normal operation

TRTST — Transmit and Receive Test (100BASE-TX)
1 = Transmit and receive data regardless of link status
0 = Normal operation

12.4 Functional Description

The EPHY is an IEEE 802.3 compliant 10/100 Ethernet physical transceiver. The EPHY can be configured to support 10BASE-T or 100BASE-TX applications. The EPHY is configurable via internal registers which are accessible through the MII management interface as well as limited configurability using the EPHY register map.

There are five basic modes of operation for the EPHY:

- Power down/initialization
- Auto-negotiate

- 10BASE-T
- 100BASE-TX
- Low-power

12.4.1 Power Down/Initialization

Upon reset, the EPHYEN bit, in the Ethernet physical transceiver control register 0 (EPHYCTL0), is cleared and EPHY is in its lowest power consumption state. All analog circuits are powered down. The twisted-pair transmitter and receiver pins (PHY_TXP, PHY_TXN, PHY_RXP, and PHY_RXN) are high-impedance. The MII management interface is not accessible. All MII registers are initialized to their reset state. The ANDIS, DIS100, and DIS10 bits, in the EPHYCTL0 register, have no effect until the EPHYEN bit is set.

The EPHYEN bit can be set or cleared by a register write at any time. Prior to enabling the EPHY, setting EPHYEN to 1, the MII PHY address PHYADD[4:0] must be set in the Ethernet physical transceiver control register 1 (EPHYCTL1), and the ANDIS, DIS100, DIS10 bits, in the EPHYCTL0 register, must be configured for the desired start-up operation. Whenever the EPHYEN bit transitions from 0 to 1, MDIO communications must be delayed until the completion of a start-up delay period ($t_{Start\text{-}up}$, see Figure 12-19).

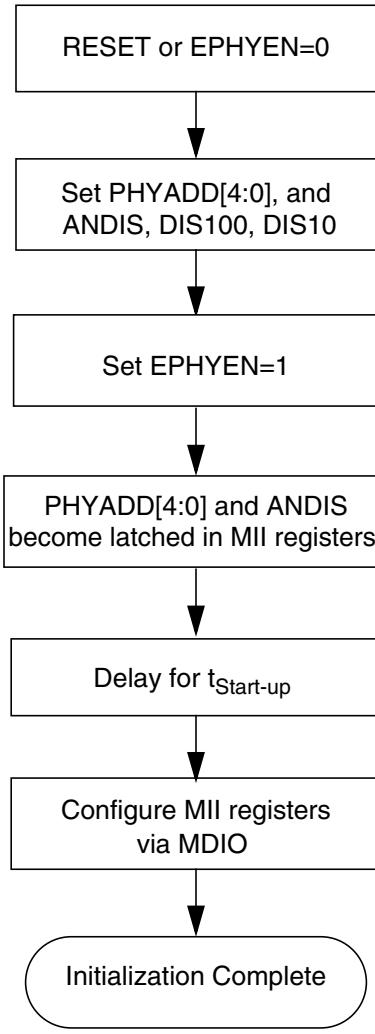


Figure 12-18. EPHY Start-Up / Initialization Sequence

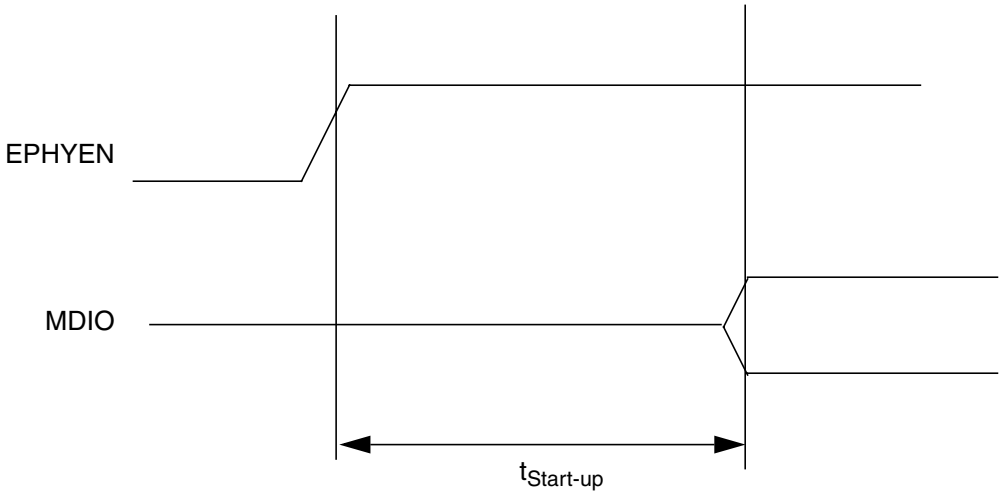


Figure 12-19. EPHY Start-Up Delay

If the auto-negotiation mode of operation is desired, the ANDIS bit in the EPHYCTL0 must be set to 0 and the DIS100 and DIS10 bits must be cleared prior to setting EPHYEN to 1. Refer to Section 12.4.2, "Auto-Negotiation," for more information on auto-negotiation operation.

If the mode of operation will be set manually, the ANDIS bit must be set to 1 in the EPHYCTL0 register and the DIS100 and DIS10 bits must be cleared prior to setting EPHYEN to 1. After the EPHYEN bit has been set and the start-up delay period is completed, the mode of operation can be configured through the MII registers. Table 12-3 summarizes the MII register configuration and operational modes.

Table 12-3. Operational Configuration While Auto-Negotiation is Disabled[1]

Bit 0.12 Auto Neg.	Bit 0.13 Data Rate	Bit 0.8 Duplex	Bit 18.6 Encoder Bypass	Bit 18.5 Scrambler Bypass	Bit 18.7 Symbol Unalign	Operation
0	0	1	X	X	X	10BASE-T full-duplex
0	0	0	X	X	X	10BASE-T half-duplex
0	1	1	0	0	0	100BASE-TX full-duplex
0	1	1	1	0	0	100BASE-TX full-duplex with encoder bypass (symbol mode) — aligned
0	1	1	1	0	1	100BASE-TX full-duplex with encoder bypass (symbol mode) — unaligned
0	1	1	1	1	0	100BASE-TX full-duplex with scrambler and encoder bypassed (symbol mode), aligned
0	1	1	1	1	1	100BASE-TX full-duplex with scrambler and encoder bypassed (symbol mode), unaligned
0	1	0	0	0	0	100BASE-TX half-duplex

[1] Symbol mode is not supported.

12.4.2 Auto-Negotiation

Auto-negotiation is used to determine the capabilities of the link partner. Auto-negotiation is compliant with IEEE 802.3 clause 28. In this case, the PHY will transmit fast link pulse (FLP) bursts to share its capabilities with the link partner.

If the link partner is also capable of performing auto-negotiation, it will also send FLP bursts. The information shared through the FLP bursts will allow both link partners to find the highest common mode (if it exists).

If no common mode is found, the remote fault bit (1.4) will be set. A remote fault is defined as a condition in which the PHY and the link partner cannot establish a common operating mode. Configuring auto-negotiation advertisement register sets the different auto-negotiation advertisement modes.

If the link partner does not support auto-negotiation, it will transmit either normal link pulses (NLP) for 10 Mbps operation, or 100 Mbps idle symbols. Based on the received signal, the PHY determines whether the link partner is 10 Mbps capable or 100 Mbps capable. The ability to do this is called parallel detection. If using parallel detection, the link will be configured as a half-duplex link. After parallel detection has established the link configuration, the remote fault bit will be set if the operating mode does not match the pre-set operating modes.

Figure 12-20 shows the main blocks used in the auto-negotiation function. The transmit block allows transmission of fast link pulses to establish communications with partners that are auto-negotiation able. The receive block determines the capabilities of the link partner and writes to the link partner ability register (register 5). The arbitration block determines the highest common mode of operation to establish the link.

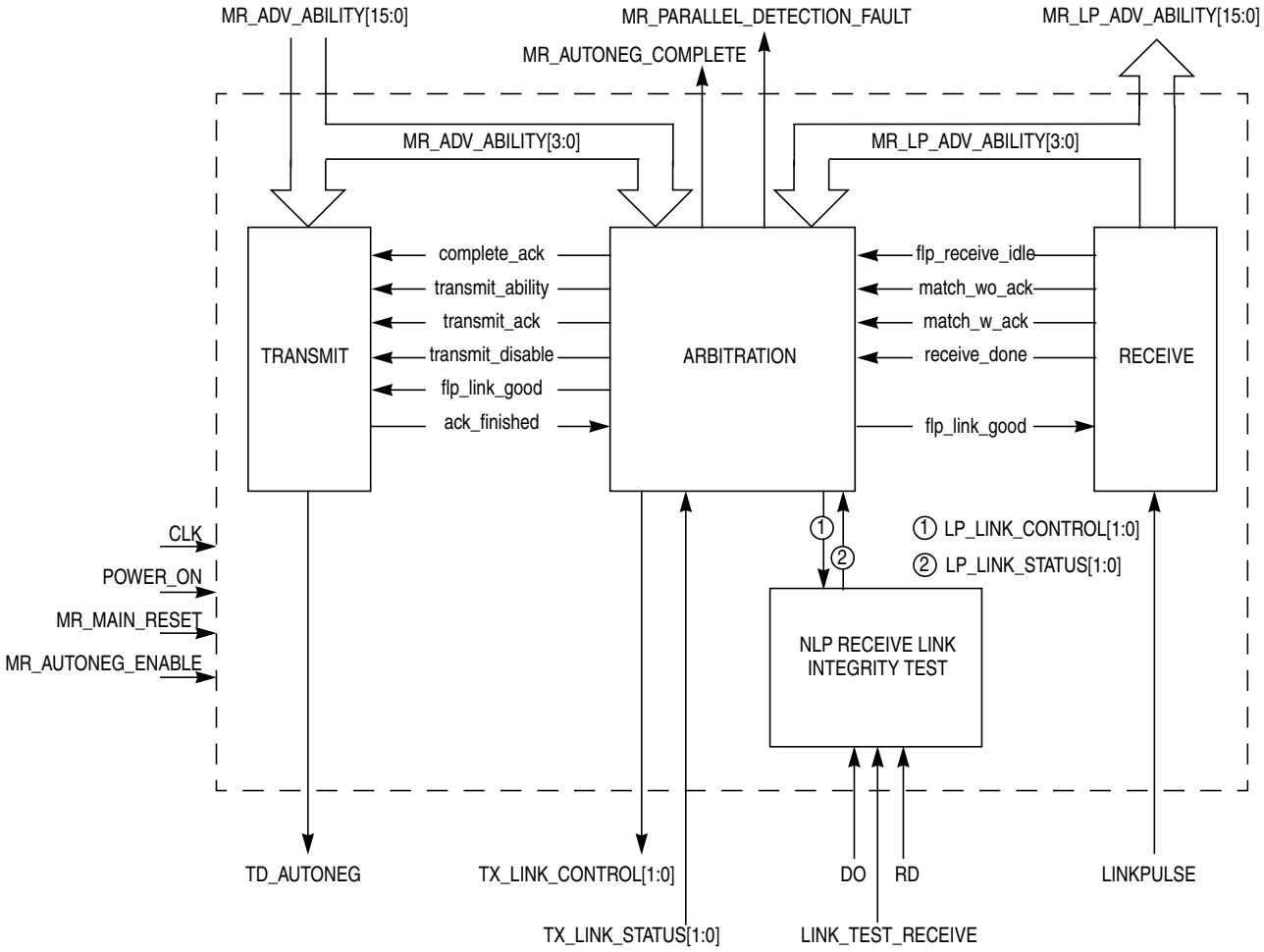


Figure 12-20. Auto-Negotiation

12.4.3 10BASE-T

The 10BASE-T interface implements the physical layer specification for a 10 Mbps over two pairs of twisted-pair cables. The specifications are given in clause 14 of the IEEE 802.3 standard.

In 10BASE-T mode, Manchester encoding is used. When transmitting, nibbles from the MII are converted to a serial bit stream and then Manchester encoded. When receiving, the Manchester encoded bit stream is decoded and converted to nibbles for presentation to the MII.

A 2.5 MHz internal clock is used for nibble wide transactions. A 10 MHz internal clock is used for serial transactions.

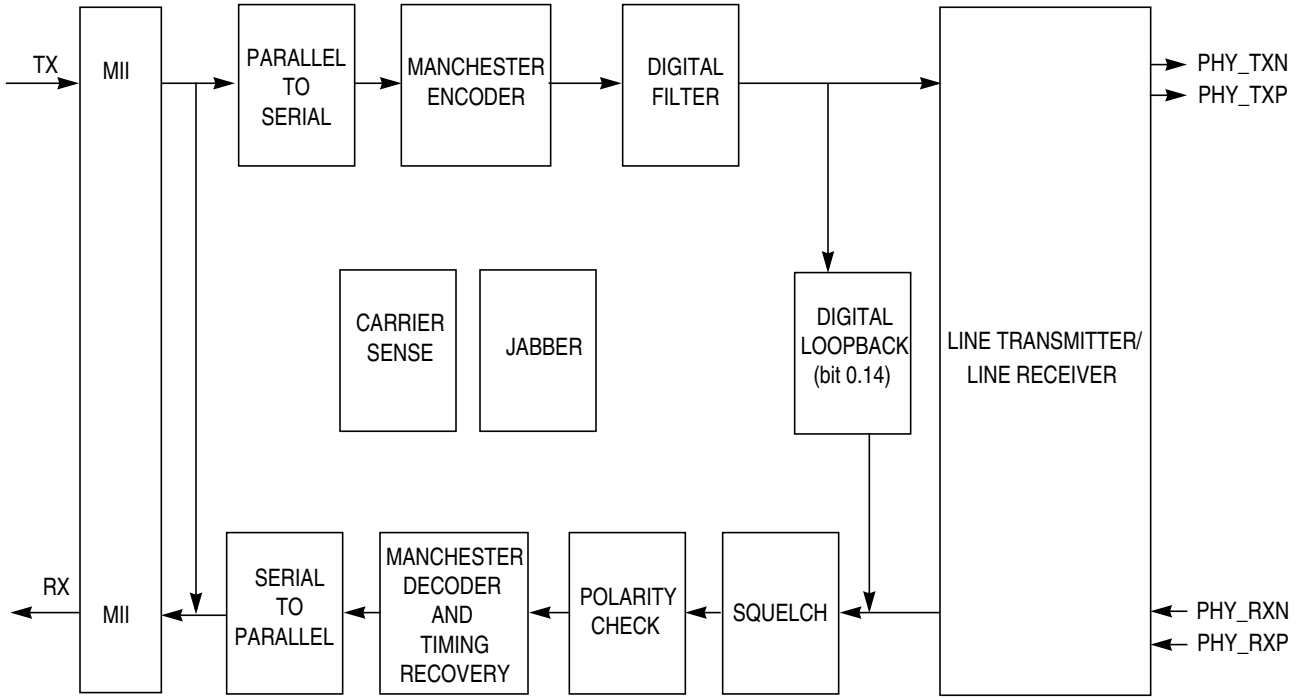


Figure 12-21. 10BASE-T Block Diagram

Parallel to Serial: Converts the 4-bit wide nibbles from the MII to serial format before the information is processed by subsequent blocks.

Manchester Encoder: Allows encoding of both the clock and data in one bit stream. A logical one is encoded as a zero when the clock is high and a one when the clock is low. A logical zero is encoded as a one when the clock is high and a zero when the clock is low.

Digital Filter: Performs pre-emphasis and low pass filtering of the input Manchester data.

DAC: Converts the digital data to an analog format before transmission on the media.

Carrier Sense: In half-duplex operation, carrier is asserted when either the transmit or receive medium is active. In full-duplex operation, carrier asserted only on reception of data. During receive, carrier sense is asserted during reception of a valid preamble, and de-asserted after reception of an EOF.

Loopback: Enabled when bit 0.14 is asserted. This loopback mode allows for the Manchester encoded and filtered data to be looped back to the squelch block in the receive path. All the 10BASE-T digital functions are exercised during this mode. The transmit and receive channels are disconnected from the media.

MII loopback (18.13) must be disabled to allow for correct operation of the digital loopback (0.14).

Link Generator: Generates a 100 ns duration pulse at the end of every 12 ms period of the transmission path being idle (TXEN de-asserted). This pulse is used to keep the 10BASE-T link operational in the absence of data transmission.

Link Integrity Test: Used to determine whether the 10BASE-T link is operational. If neither data nor a link pulse is received for 64 ms, then the link is considered down. While the link is down, the transmit, loopback, collision detect, and SQE functions are disabled. The link down state is exited after receiving data or four link pulses.

Jabber: Prevents the transmitter from erroneously transmitting for too long a period. The maximum time the device can transmit is 50,000 bit times. When the jabber timer is exceeded, the transmit output goes idle for 0.525 s.

This function can be disabled with the jabber inhibit register bit (18.10).

Squelch: Used to determine whether active data, a link pulse, or an idle condition exists on the 10BASE-T receive channel. While an idle or link pulse condition exists, a higher squelch level is used for greater noise immunity. The squelch output is used to determine when the Manchester decoder should operate. The output is also used to determine when an end of packet is received.

Polarity Check: By examining the polarity of the received link pulses, EPHY can determine whether the received signal is inverted. If the pulses are inverted, this function changes the polarity of the signal.This feature is activated if eight inverted link pulses are received or four frames with inverted EOF are encountered.

Manchester Decoder and Timing Recovery: Decodes the Manchester encoded data. The receive data and clock are recovered during this process.

Serial to Parallel: Converts the serial bit stream from the Manchester decoder to the required MII parallel format.

PMD Sublayer: Transmits and receives signals compliant with IEEE 802.3, Section 14.

Line Transmitter and Line Receiver: These analog blocks allow the EPHY to drive and receive data from the 10BASE-T media.

12.4.4 100BASE-TX

100BASE-TX specifies operation over two pairs of category 5 unshielded twisted-pair cable (UTP).

The EPHY implementation includes the physical coding sublayer (PCS), the physical medium attachment (PMA), and the physical medium dependent (PMD) sublayer.

The block diagram for 100BASE-TX operation is shown in Figure 12-22.

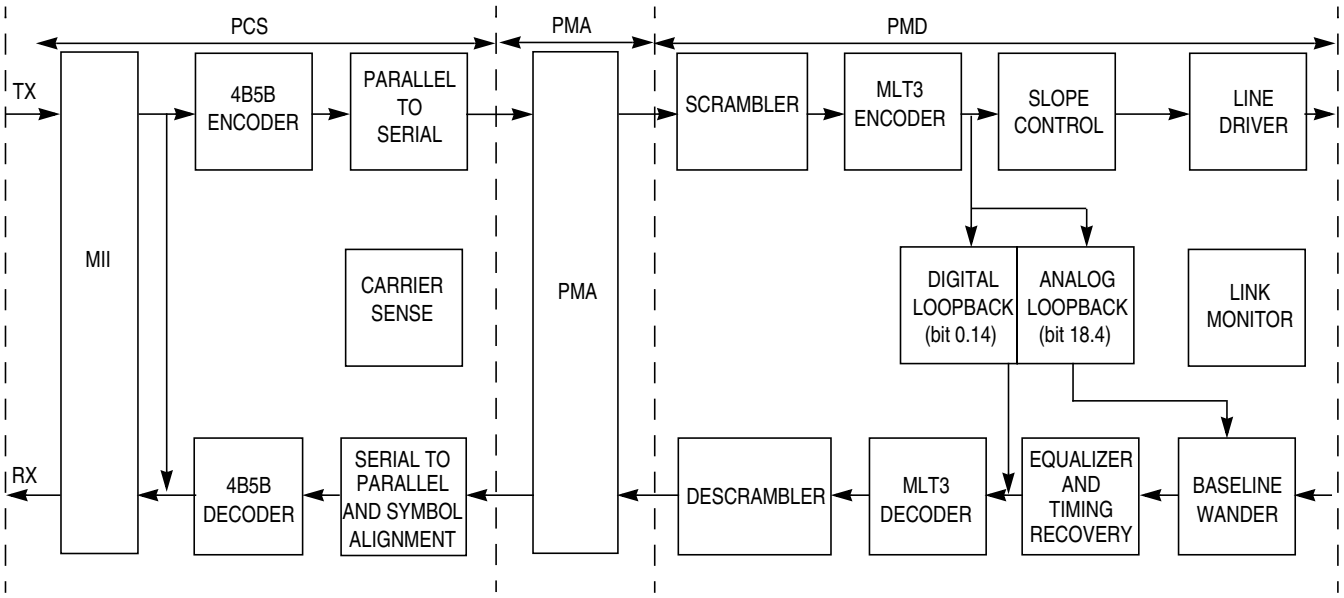


Figure 12-22. 100BASE-TX Block Diagram

12.4.4.1 Sublayers

12.4.4.1.1 PCS Sublayer

The PCS sublayer is the MII interface that provides a uniform interface to the reconciliation sublayer.

The services provided by the PCS include:

- Encoding/decoding of MII data nibbles to/from 5-bit code-groups (4B/5B)
- Carrier sense and collision indications
- Serialization/deserialization of code-groups for transmission/reception on the PMA
- Mapping of transmit, receive, carrier sense, and collision detection between the MII and the underlying PMA

Serial to Parallel and Symbol Alignment: This block looks for the occurrence of the JK symbol to align the serial bit stream and convert it to a parallel format.

Carrier Sense: In full-duplex mode, carrier sense is only asserted while the receive channel is active**.** The carrier sense examines the received data bit stream looking for the SSD, the JK symbol pair. In the idle state, IDLE symbols (all logic ones) will be received. If the first 5-bit symbols received after an idle stream forms the J symbol (11000) it asserts the CRS signal. At this point the second symbol is checked to confirm the K symbol (10001). If successful, the following aligned data (symbols) are presented to the 4B/5B decoder. If the JK pair is not confirmed, the false carrier detect is asserted and the idle state is re-entered.

Carrier sense is de-asserted when the ESD (end-of-stream) delimiter, the TR symbol pair, is found, or when an idle state is detected.

In half-duplex, CRS is also asserted on transmit.

Parallel to Serial: This block takes parallel data and converts it to serial format.

4B/5B Encoder/Decoder: The 4B/5B encoder converts the 4-bit nibbles from the reconciliation sublayer to a 5-bit code group.

12.4.4.1.2 PMA Sublayer

The PMA provides medium-independent means for the PCS and other bit-oriented clients (e.g., repeaters) to support the use of a range of physical media. For 100BASE-TX the PMA performs these functions:

- Mapping of transmit and receive code-bits between the PMA's client and the underlying PMD
- Generating a control signal indicating the availability of the PMD to a PCS or other client
- Synchronization with the auto-negotiation function
- Generating indications of carrier activity and carrier errors from the PMD
- Recovery of clock from the NRZI data supplied by the PMD

12.4.4.1.3 PMD Sublayer

For 100BASE-TX, the ANSI X3.263: 199X (TP-PMD) standard is used. These signalling standards, called PMD sublayers, define 125 Mbps, full-duplex signalling systems that use STP and UTP wiring.

Scrambler/De-scrambler: The scrambler and de-scrambler used in EPHY meet the ANSI Standard X3.263-1995 FDDI TP-PMD. The purpose of the scrambler is to randomize the 125 Mbps data on transmission resulting in a reduction of the peak amplitudes in the frequency spectrum. The de-scrambler restores the received 5-bit code groups to their unscrambled values.

The scrambler input data (plaintext) is encoded by modulo 2 addition of a key stream to produce a ciphertext bit stream. The key stream is a periodic sequence of 2047 bits generated by the recursive linear function X[n] = X[n-11] + X[n-9] (modulo 2).

If not transmitting data, the scrambler encodes and transmits idles. This allows a pattern to use by the de-scrambler to synchronize and decode the scrambled data.

The implementation of the scrambler and de-scrambler is as shown in Appendix G of the ANSI Standard X3.263-1995.

For test, the scrambler can be bypassed by setting bit 18.5. Scrambler bypass mode is a special type of interface for 100BASE-TX operation that bypasses the scrambler and de-scrambler operation. This mode is typically used for test so that input and output test vectors match. In this mode, idles are not sent and the MAC must provide idles.

MLT-3 Encoder/Decoder: An MLT-3 encoder is used in the transmit path to convert NRZ bit stream data from the PMA sublayer into a three-level code. This encoding results in a reduction in the energy over the critical frequency range. The MLT-3 decoder converts the received three-level code back to an NRZ bit stream.

Baseline Wander: The use of the scrambler and MLT-3 encoding can cause long run lengths of 0s and 1s that can produce a DC component. The DC component cannot be transmitted through the isolation transformers and results in baseline wander. Baseline wander reduces noise immunity because the base line moves nearer to either the positive or negative signal comparators. To correct for this EPHY uses DC

restoration to restore the lost DC component of the recovered digital data to correct the baseline wander problem.

Timing Recovery: The timing recovery block locks onto the incoming data stream, extracts the embedded clock, and presents the data synchronized to the recovered clock.

In the event that the receive path is unable to converge to the receive signal, it resets the MSE-good (bit 25.15) signal. The clock synthesizer provides a center frequency reference for operation of the clock recovery circuit in the absence of data.

Adaptive Equalizer: At a data rate of 125 Mbps, the cable introduces significant distortion due to high frequency roll off and phase shift. The high frequency loss is mainly due to skin effect, which causes the conductor resistance to rise as the square of the frequency.

The adaptive equalizer will compensate for signal amplitude and phase distortion incurred from transmitting with different cable lengths.

Loopback: If asserted by bit 0.14, data encoded by the MLT3 encoder block is looped back to the MLT3 decoder block while the transmit and receive paths are disconnected from the media.

A second loopback mode for 100BASE-TX is available by setting bit 18.13 (MII loopback) to a logical 1. This loopback mode takes the MII transmit data and loops it directly back to the MII receive pins. Again, the transmit and receive paths are disconnected from the media.

MII loopback has precedence over the digital loopback if both are enabled at the same time.

A third loopback mode is available by setting bit 18.4 high. This analog loopback mode takes the MLT3 encoded data and loops it back through the base line wander and analog receive circuits.

Line Transmitter and Line Receiver: These analog blocks allow EPHY to drive and receive data to/from the 100BASE-TX media. The transmitter is designed to drive a 100-Ω UTP cable.

Link Monitor: The link monitor process is responsible for determining whether the underlying receive channel is providing reliable data. If a failure is found, normal operation will be disabled. As specified in the IEEE 802.3 standard, the link is operating reliably if a signal is detected for a period of 330 µs.

Far End Fault: While the auto-negotiation function is disabled, this function is used to exchange fault information between the PHY and the link partner.

12.4.5 Low Power Modes

There are several reduced power configurations available for the EPHY.

12.4.5.1 Stop Mode

If the MCU executes a STOP instruction, the EPHY will be powered down and all internal MII registers reset to their default state. Upon exiting stop mode, the EPHY will exit the power-down state and latch the values previously written to the EPHYCTL0 and EPHYCTL1 registers. The MII registers will have to be re-initialized after the start-up delay ($t_{Start\text{-}up}$) has expired.

12.4.5.2 Wait Mode

If the MCU executes a WAIT instruction with the EPHYWAI bit set, the EPHY will be powered down and all internal MII registers reset to their default state. Upon exiting STOP mode the EPHY will exit the power-down state and latch the values previously written to the EPHYCTL0 and EPHYCTL1 registers. The MII registers must be re-initialized after the start-up delay ($t_{Start\text{-}up}$) has expired.

12.4.5.3 MII Power Down

This mode disconnects the PHY from the network interface (three-state receiver and driver pins).

Setting bit 0.11 of the port enters this mode. In this mode, the management interface is accessible but all internal chip functions are in a zero power state.

In this mode all analog blocks except the PLL clock generator and band gap reference are in low power mode. All digital blocks except the MDIO interface and management registers are inactive.

Chapter 13
Penta Output Voltage Regulator (VREGPHYV1)

13.1 Introduction

13.1.1 Overview

Block VREG_PHY is a penta output voltage regulator providing five separate 2.5V (typ) supplies differing in the amount of current that can be sourced. The regulator input voltage is 3.3V+/-10% .

13.1.2 Features

The block VREG_PHY includes these distinctive features:

- Five parallel, linear voltage regulators
  - — Bandgap reference
- Power On Reset (POR)
- Low Voltage Reset (LVR)

13.1.3 Modes of Operation

There are three modes VREG_PHY can operate in:

- Full Performance Mode (FPM) (CPU is not in Stop Mode)

  The regulator is active, providing the nominal supply voltage of 2.5V with full current sourcing capability at all outputs. Features LVR (Low Voltage Reset) and POR (Power-On Reset) are available.
- Reduced Power Mode (RPM) (CPU is in Stop Mode)

  The purpose is to reduce power consumption of the device. The output voltage may degrade to a lower value than in Full Performance Mode, additionally the current sourcing capability is substantially reduced. Only the POR is available in this mode, LVR are disabled.
- Shutdown Mode

  Controlled by VREGEN (see device level specification for connectivity of VREGEN).

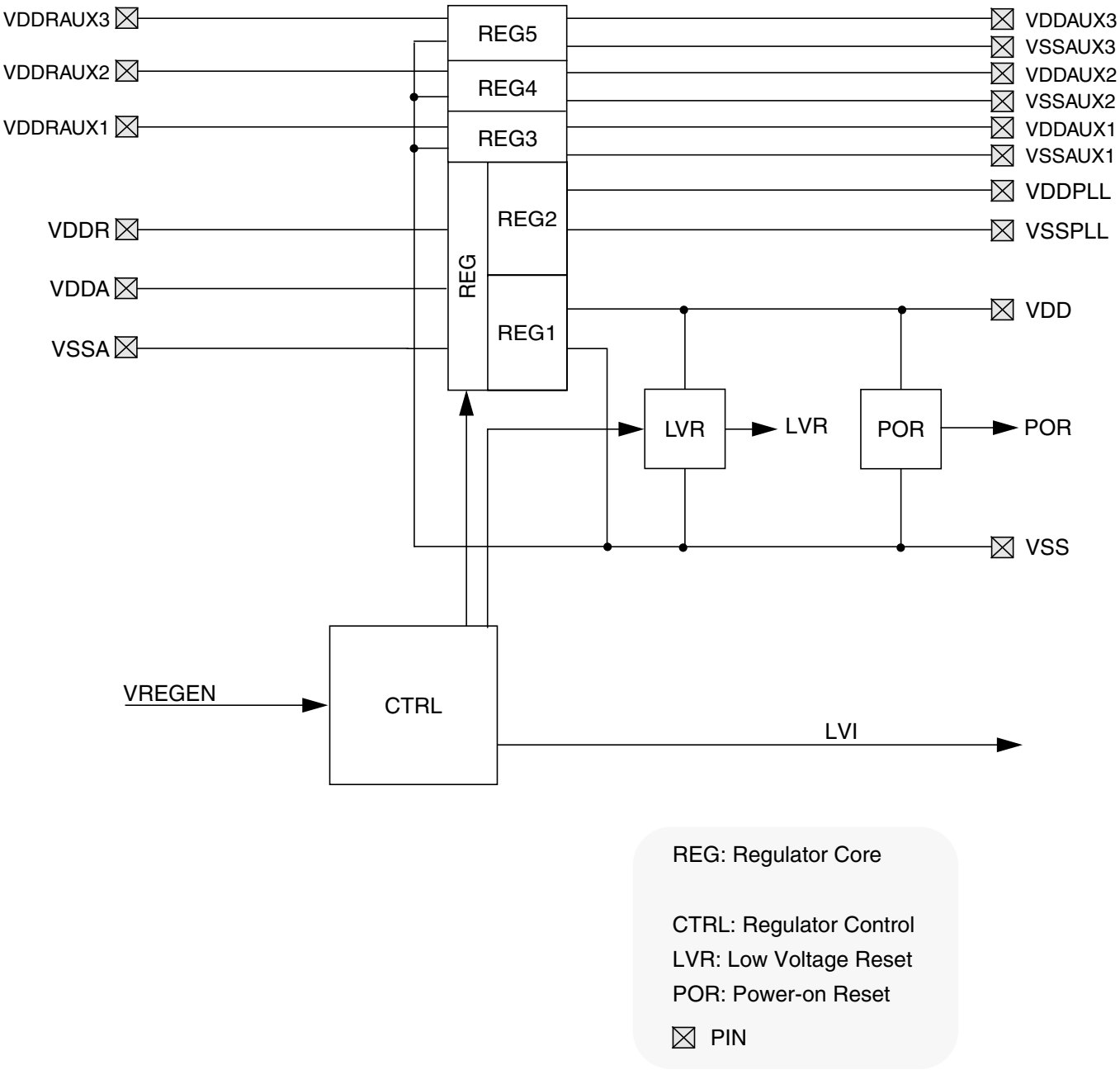
  This mode is characterized by minimum power consumption. The regulator outputs are in a high impedance state, only the POR feature is available, LVR is disabled.

  This mode must be used to disable the chip internal regulator VREG_PHY, i.e. to bypass the VREG_PHY to use external supplies.

13.1.4 Block Diagram

Figure 13-1 shows the function principle of VREG_PHY by means of a block diagram. The regulator core REG consists of file parallel subblocks, providing five independent output voltages.

Figure 13-1. VREG_PHY - Block Diagram



13.2 Signal Description

13.2.1 Overview

Due to the nature of VREG_PHY being a voltage regulator providing the chip internal power supply voltages most signals are power supply signals connected to pads.

Table 13-1 shows all signals of VREG_PHY associated with pins.

Table 13-1. VREG_PHY - Signal Properties

Name	Port	Function	Reset State	Pull up
VDDR	—	VREG_PHY power input (positive supply)	—	—
VDDRAUX1	—	VREG_PHY power input (positive supply)	—	—
VDDRAUX2	—	VREG_PHY power input (positive supply)	—	—
VDDRAUX3	—	VREG_PHY power input (positive supply)	—	—
VDDA	—	VREG_PHY quiet input (positive supply)	—	—
VSSA	—	VREG_PHY quiet input (ground)	—	—
VDD	—	VREG_PHY primary output (positive supply)	—	—
VSS	—	VREG_PHY primary output (ground)	—	—
VDDPLL	—	VREG_PHY secondary output (positive supply)	—	—
VSSPLL	—	VREG_PHY secondary output (ground)	—	—
VDDAUX1	—	VREG_PHY third output (positive supply)	—	—
VSSAUX1	—	VREG_PHY third output (ground)	—	—
VDDAUX2	—	VREG_PHY fourth output (positive supply)	—	—
VSSAUX2	—	VREG_PHY fourth output (ground)	—	—
VDDAUX3	—	VREG_PHY fifth output (positive supply)	—	—
VSSAUX3	—	VREG_PHY fifth output (ground)	—	—
VREGEN (optional)	—	VREG_PHY (Optional) Regulator Enable	—	—

13.2.2 Detailed Signal Descriptions

Check device level specification for connectivity of the signals.

13.2.2.1 VDDR,VDDRAUX1,2,3, VSS - Regulator Power Inputs

Signal VDDR/VDDRAUX1,2,3 are the power inputs of VREG_PHY. All currents sourced into the regulator loads flow through these pins. A chip external decoupling capacitor (100nF...220nF, X7R ceramic) between VDDR/VDDRAUX1,2,3 and VSS can smoothen ripple on VDDR/VDDRAUX1,2,3.

If the regulator shall be bypassed, VDDR should be tied to ground. In Shutdown Mode pin VDDR should also be tied to ground on devices without VREGEN pin.

13.2.2.2 VDDA, VSSA - Regulator Reference Supply

Signals VDDA/VSSA which are supposed to be relatively quiet are used to supply the analog parts of the regulator. Internal precision reference circuits are supplied from these signals. A chip external decoupling capacitor (100nF...220nF, X7R ceramic) between VDDA and VSSA can further improve the quality of this supply.

13.2.2.3 VDD, VSS - Regulator Output1 (Core Logic)

Signals VDD/VSS are the primary outputs of VREG_PHY that provide the power supply for the core logic. These signals are connected to device pins to allow external decoupling capacitors (100nF...220nF, X7R ceramic).

In Shutdown Mode an external supply at VDD/VSS can replace the voltage regulator.

13.2.2.4 VDDPLL, VSSPLL - Regulator Output2 (PLL)

Signals VDDPLL/VSSPLL are the secondary outputs of VREG_PHY that provide the power supply for the PLL and Oscillator. These signals are connected to device pins to allow external decoupling capacitors (100nF...220nF, X7R ceramic).

In Shutdown Mode an external supply at VDDPLL/VSSPLL can replace the voltage regulator.

13.2.2.5 VDDAUX1,2,3, VSSAUX1,2,3 - Regulator Output3,4,5

Signals VDDAUX1,2,3/VSSAUX1,2,3 are the auxilliary outputs of VREG_PHY. These signals are connected to device pins to allow external decoupling capacitors (100nF...220nF, X7R ceramic).

In Shutdown Mode an external supply at VDDAUX1,2,3/VSSAUX1,2,3 can replace the voltage regulator.

13.2.2.6 VREGEN - Optional Regulator Enable

This optional signal is used to shutdown VREG_PHY. In that case VDD/VSS and VDDPLL/VSSPLL must be provided externally. Shutdown Mode is entered with VREGEN being low. If VREGEN is high, the VREG_PHY is either in Full Performance Mode or in Reduced Power Mode.

For the connectivity of VREGEN see device specification.

NOTE

Switching from FPM or RPM to shutdown of VREG_PHY and vice versa is not supported while MCU is powered.

13.3 Memory Map and Registers

13.3.1 Overview

VREG_PHY does not contain any CPU accessible registers.

13.4 Functional Description

13.4.1 General

Block VREG_PHY is a voltage regulator as depicted in Figure 13-1. The regulator functional elements are the regulator core (REG), a power-on reset module (POR) and a low voltage reset module (LVR). There is also the regulator control block (CTRL) which represents the interface to the digital core logic but also handles the operating modes of VREG_PHY.

13.4.2 REG - Regulator Core

VREG_PHY, respectively its regulator core has five parallel, independent regulation loops (REG1 to REG5) that differ only in the amount of current that can be sourced to the connected loads. Therefore only REG1, providing the supply at VDD/VSS, is explained. The principle is also valid for REG2 to REG5.

The regulator is a linear series regulator with a bandgap reference in its Full Performance Mode and a voltage clamp in Reduced Power Mode. All load currents flow from input VDDR or VDDRAUX1,2,3 to VSS or VSSPLL or VSSAUX1,2,3, the reference circuits are connected to VDDA and VSSA.

13.4.2.1 Full Performance Mode

In Full Performance Mode a fraction of the output voltage (VDD) and the bandgap reference voltage are fed to an operational amplifier. The amplified input voltage difference controls the gate of an output driver which basically is a large NMOS transistor connected to the output.

13.4.2.2 Reduced Power Mode

In Reduced Power Mode the driver gate is connected to a buffered fraction of the input voltage(VDDR). The operational amplifier and the bandgap are disabled to reduce power consumption.

13.4.3 POR - Power-On Reset

This functional block monitors output VDD. If V_{DD} is below V_{PORD}, signal POR is high, if it exceeds V_{PORD}, the signal goes low. The transition to low forces the CPU in the power-on sequence.

Due to its role during chip power-up this module must be active in all operating modes of VREG_PHY.

13.4.4 LVR - Low Voltage Reset

Block LVR monitors the primary output voltage V_{DD}. If it drops below the assertion level (V_{LVRA}) signal LVR asserts and when rising above the deassertion level (V_{LVRD}) signal LVR negates again. The LVR function is available only in Full Performance Mode.

13.4.5 CTRL - Regulator Control

This part contains the register block of VREG_PHY and further digital functionality needed to control the operating modes. CTRL also represents the interface to the digital core logic.

13.5 Resets

13.5.1 General

This section describes how VREG_PHY controls the reset of the MCU.The reset values of registers and signals are provided in Section 13.3, “Memory Map and Registers.” Possible reset sources are listed in Table 13-2.

Table 13-2. VREG_PHY - Reset Sources

Reset Source	Local Enable
Power-on Reset	always active
Low Voltage Reset	available only in Full Performance Mode

13.5.2 Description of Reset Operation

13.5.2.1 Power-On Reset

During chip power-up the digital core may not work if its supply voltage V_{DD} is below the POR deassertion level (V_{PORD}). Therefore signal POR which forces the other blocks of the device into reset is kept high until V_{DD} exceeds V_{PORD}. Then POR becomes low and the reset generator of the device continues the start-up sequence. The power-on reset is active in all operation modes of VREG_PHY.

13.5.2.2 Low Voltage Reset

For details on low voltage reset see section Section 13.4.4, “LVR - Low Voltage Reset.”

13.6 Interrupts

13.6.1 General

VREG_PHY does not generate any interrupts.

Chapter 14
Interrupt (INTV1)

14.1 Introduction

This section describes the functionality of the interrupt (INT) sub-block of the S12 core platform.

A block diagram of the interrupt sub-block is shown in Figure 14-1.

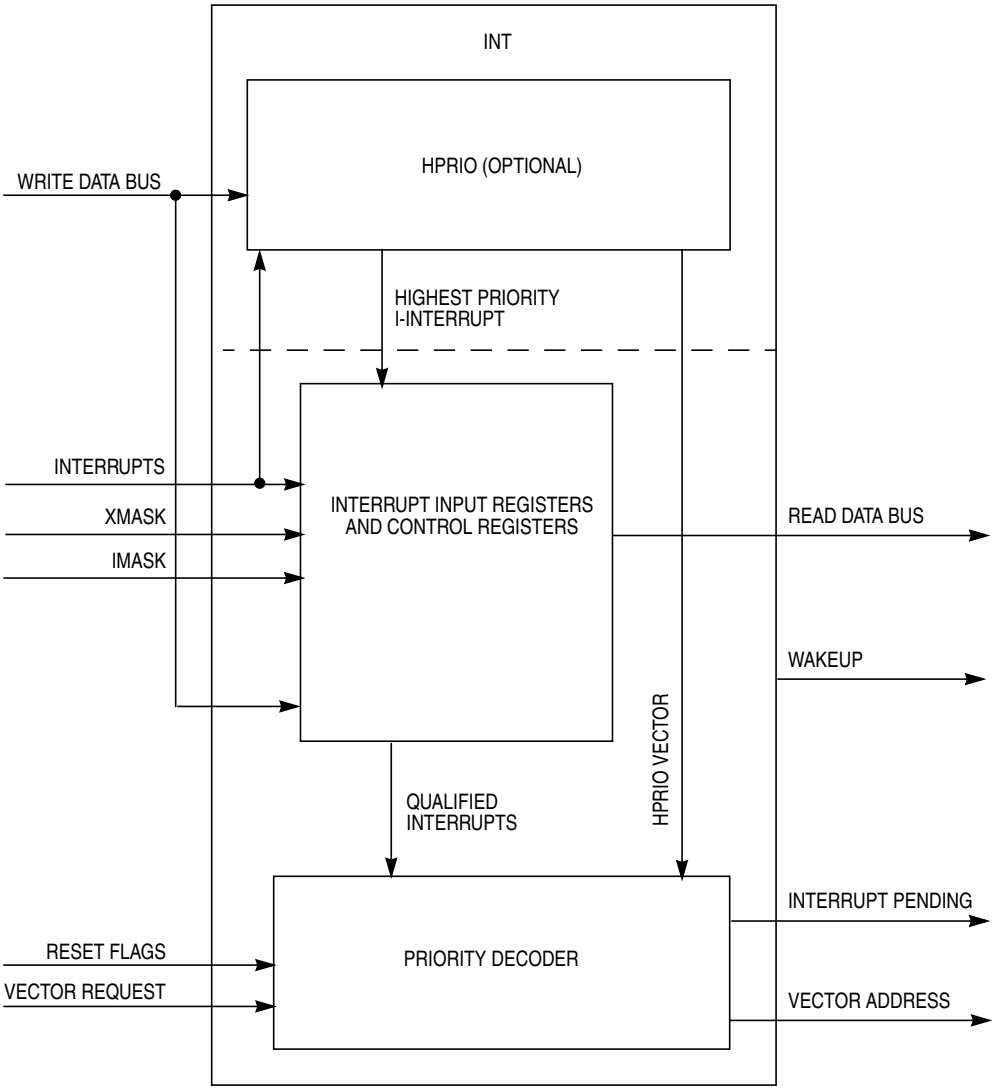


Figure 14-1. INTV1 Block Diagram

The interrupt sub-block decodes the priority of all system exception requests and provides the applicable vector for processing the exception. The INT supports I-bit maskable and X-bit maskable interrupts, a non-maskable unimplemented opcode trap, a non-maskable software interrupt (SWI) or background debug

mode request, and three system reset vector requests. All interrupt related exception requests are managed by the interrupt sub-block (INT).

14.1.1 Features

The INT includes these features:

- Provides two to 122 I-bit maskable interrupt vectors (0xFF00–0xFFF2)
- Provides one X-bit maskable interrupt vector (0xFFF4)
- Provides a non-maskable software interrupt (SWI) or background debug mode request vector (0xFFF6)
- Provides a non-maskable unimplemented opcode trap (TRAP) vector (0xFFF8)
- Provides three system reset vectors (0xFFFA–0xFFFE) (reset, CMR, and COP)
- Determines the appropriate vector and drives it onto the address bus at the appropriate time
- Signals the CPU that interrupts are pending
- Provides control registers which allow testing of interrupts
- Provides additional input signals which prevents requests for servicing I and X interrupts
- Wakes the system from stop or wait mode when an appropriate interrupt occurs or whenever $\overline{\text{XIRQ}}$ is active, even if $\overline{\text{XIRQ}}$ is masked
- Provides asynchronous path for all I and X interrupts, (0xFF00–0xFFF4)
- (Optional) selects and stores the highest priority I interrupt based on the value written into the HPRIO register

14.1.2 Modes of Operation

The functionality of the INT sub-block in various modes of operation is discussed in the subsections that follow.

- **Normal operation**

  The INT operates the same in all normal modes of operation.
- **Special operation**

  Interrupts may be tested in special modes through the use of the interrupt test registers.
- **Emulation modes**

  The INT operates the same in emulation modes as in normal modes.
- **Low power modes**

  See Section 14.4.1, “Low-Power Modes,” for details

14.2 External Signal Description

Most interfacing with the interrupt sub-block is done within the core. However, the interrupt does receive direct input from the multiplexed external bus interface (MEBI) sub-block of the core for the $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$ pin data.

14.3 Memory Map and Register Definition

Detailed descriptions of the registers and associated bits are given in the subsections that follow.

14.3.1 Module Memory Map

Table 14-1. INT Memory Map

Address Offset	Use	Access
0x0015	Interrupt Test Control Register (ITCR)	R/W
0x0016	Interrupt Test Registers (ITEST)	R/W
0x001F	Highest Priority Interrupt (Optional) (HPRIO)	R/W

14.3.2 Register Descriptions

14.3.2.1 Interrupt Test Control Register

	7	6	5	4	3	2	1	0
R	0	0	0	WRTINT	ADR3	ADR2	ADR1	ADR0
W								
Reset	0	0	0	0	1	1	1	1

= Unimplemented or Reserved

Figure 14-2. Interrupt Test Control Register (ITCR)

Read: See individual bit descriptions

Write: See individual bit descriptions

Table 14-2. ITCR Field Descriptions

Field	Description
4 WRTINT	**Write to the Interrupt Test Registers** Read: anytime Write: only in special modes and with I-bit mask and X-bit mask set. 0 Disables writes to the test registers; reads of the test registers will return the state of the interrupt inputs. 1 Disconnect the interrupt inputs from the priority decoder and use the values written into the ITEST registers instead. **Note:** Any interrupts which are pending at the time that WRTINT is set will remain until they are overwritten.
3:0 ADR[3:0]	**Test Register Select Bits** Read: anytime Write: anytime These bits determine which test register is selected on a read or write. The hexadecimal value written here will be the same as the upper nibble of the lower byte of the vector selects. That is, an “F” written into ADR[3:0] will select vectors 0xFFFE–0xFFF0 while a “7” written to ADR[3:0] will select vectors 0xFF7E–0xFF70.

14.3.2.2 Interrupt Test Registers

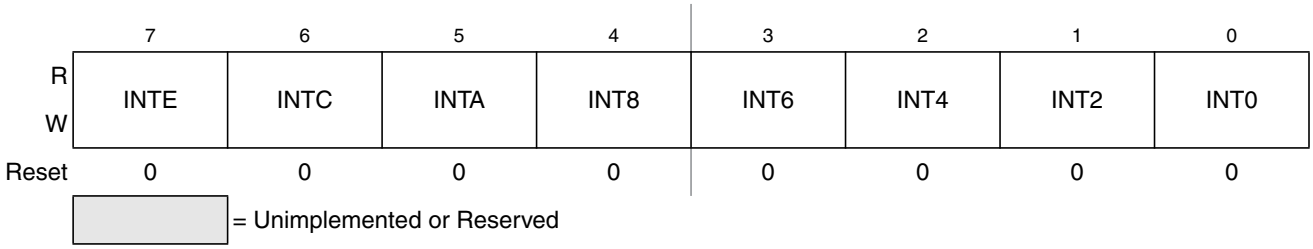


Figure 14-3. Interrupt TEST Registers (ITEST)

Read: Only in special modes. Reads will return either the state of the interrupt inputs of the interrupt sub-block (WRTINT = 0) or the values written into the TEST registers (WRTINT = 1). Reads will always return 0s in normal modes.

Write: Only in special modes and with WRTINT = 1 and CCR I mask = 1.

Table 14-3. ITEST Field Descriptions

Field	Description
7:0 INT[E:0]	**Interrupt TEST Bits** — These registers are used in special modes for testing the interrupt logic and priority independent of the system configuration. Each bit is used to force a specific interrupt vector by writing it to a logic 1 state. Bits are named INTE through INT0 to indicate vectors 0xFFxE through 0xFFx0. These bits can be written only in special modes and only with the WRTINT bit set (logic 1) in the interrupt test control register (ITCR). In addition, I interrupts must be masked using the I bit in the CCR. In this state, the interrupt input lines to the interrupt sub-block will be disconnected and interrupt requests will be generated only by this register. These bits can also be read in special modes to view that an interrupt requested by a system block (such as a peripheral block) has reached the INT module. There is a test register implemented for every eight interrupts in the overall system. All of the test registers share the same address and are individually selected using the value stored in the ADR[3:0] bits of the interrupt test control register (ITCR). **Note:** When ADR[3:0] have the value of 0x000F, only bits 2:0 in the ITEST register will be accessible. That is, vectors higher than 0xFFF4 cannot be tested using the test registers and bits 7:3 will always read as a logic 0. If ADR[3:0] point to an unimplemented test register, writes will have no effect and reads will always return a logic 0 value.

14.3.2.3 Highest Priority I Interrupt (Optional)

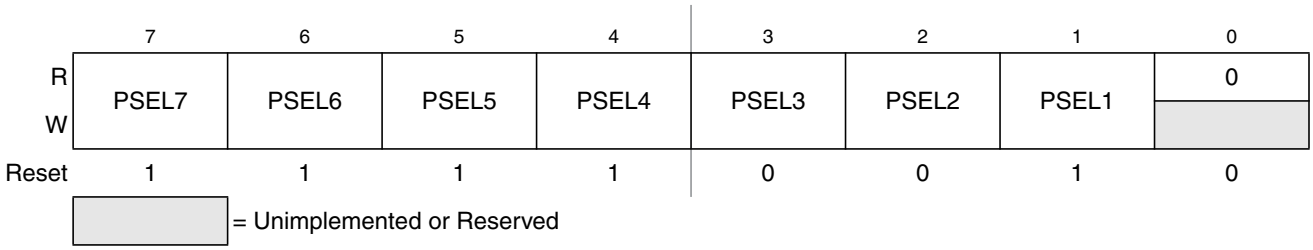


Figure 14-4. Highest Priority I Interrupt Register (HPRIO)

Read: Anytime

Write: Only if I mask in CCR = 1

Table 14-4. HPRIO Field Descriptions

Field	Description
7:1 PSEL[7:1]	**Highest Priority I Interrupt Select Bits** — The state of these bits determines which I-bit maskable interrupt will be promoted to highest priority (of the I-bit maskable interrupts). To promote an interrupt, the user writes the least significant byte of the associated interrupt vector address to this register. If an unimplemented vector address or a non I-bit masked vector address (value higher than 0x00F2) is written, IRQ (0xFFF2) will be the default highest priority interrupt.

14.4 Functional Description

The interrupt sub-block processes all exception requests made by the CPU. These exceptions include interrupt vector requests and reset vector requests. Each of these exception types and their overall priority level is discussed in the subsections below.

14.4.1 Low-Power Modes

The INT does not contain any user-controlled options for reducing power consumption. The operation of the INT in low-power modes is discussed in the following subsections.

14.4.1.1 Operation in Run Mode

The INT does not contain any options for reducing power in run mode.

14.4.1.2 Operation in Wait Mode

Clocks to the INT can be shut off during system wait mode and the asynchronous interrupt path will be used to generate the wake-up signal upon recognition of a valid interrupt or any $\overline{\text{XIRQ}}$ request.

14.4.1.3 Operation in Stop Mode

Clocks to the INT can be shut off during system stop mode and the asynchronous interrupt path will be used to generate the wake-up signal upon recognition of a valid interrupt or any $\overline{\text{XIRQ}}$ request.

14.5 Resets

The INT supports three system reset exception request types: normal system reset or power-on-reset request, crystal monitor reset request, and COP watchdog reset request. The type of reset exception request must be decoded by the system and the proper request made to the core. The INT will then provide the service routine address for the type of reset requested.

14.6 Interrupts

As shown in the block diagram in Figure 14-1, the INT contains a register block to provide interrupt status and control, an optional highest priority I interrupt (HPRIO) block, and a priority decoder to evaluate whether pending interrupts are valid and assess their priority.

14.6.1 Interrupt Registers

The INT registers are accessible only in special modes of operation and function as described in Section 14.3.2.1, “Interrupt Test Control Register,” and Section 14.3.2.2, “Interrupt Test Registers,” previously.

14.6.2 Highest Priority I-Bit Maskable Interrupt

When the optional HPRIO block is implemented, the user is allowed to promote a single I-bit maskable interrupt to be the highest priority I interrupt. The HPRIO evaluates all interrupt exception requests and passes the HPRIO vector to the priority decoder if the highest priority I interrupt is active. RTI replaces the promoted interrupt source.

14.6.3 Interrupt Priority Decoder

The priority decoder evaluates all interrupts pending and determines their validity and priority. When the CPU requests an interrupt vector, the decoder will provide the vector for the highest priority interrupt request. Because the vector is not supplied until the CPU requests it, it is possible that a higher priority interrupt request could override the original exception that caused the CPU to request the vector. In this case, the CPU will receive the highest priority vector and the system will process this exception instead of the original request.

NOTE

Care must be taken to ensure that all exception requests remain active until the system begins execution of the applicable service routine; otherwise, the exception request may not be processed.

If for any reason the interrupt source is unknown (e.g., an interrupt request becomes inactive after the interrupt has been recognized but prior to the vector request), the vector address will default to that of the last valid interrupt that existed during the particular interrupt sequence. If the CPU requests an interrupt vector when there has never been a pending interrupt request, the INT will provide the software interrupt (SWI) vector address.

14.7 Exception Priority

The priority (from highest to lowest) and address of all exception vectors issued by the INT upon request by the CPU is shown in Table 14-5.

Table 14-5. Exception Vector Map and Priority

Vector Address	Source
0xFFFE–0xFFFF	System reset
0xFFFC–0xFFFD	Crystal monitor reset
0xFFFA–0xFFFB	COP reset
0xFFF8–0xFFF9	Unimplemented opcode trap
0xFFF6–0xFFF7	Software interrupt instruction (SWI) or BDM vector request
0xFFF4–0xFFF5	$\overline{\text{XIRQ}}$ signal

Table 14-5. Exception Vector Map and Priority

Vector Address	Source
0xFFF2–0xFFF3	$\overline{\text{IRQ}}$ signal
0xFFF0–0xFF00	Device-specific I-bit maskable interrupt sources (priority in descending order)

Chapter 15
Multiplexed External Bus Interface (MEBIV3)

15.1 Introduction

This section describes the functionality of the multiplexed external bus interface (MEBI) sub-block of the S12 core platform. The functionality of the module is closely coupled with the S12 CPU and the memory map controller (MMC) sub-blocks.

Figure 15-1 is a block diagram of the MEBI. In Figure 15-1, the signals on the right hand side represent pins that are accessible externally. On some chips, these may not all be bonded out.

The MEBI sub-block of the core serves to provide access and/or visibility to internal core data manipulation operations including timing reference information at the external boundary of the core and/or system. Depending upon the system operating mode and the state of bits within the control registers of the MEBI, the internal 16-bit read and write data operations will be represented in 8-bit or 16-bit accesses externally. Using control information from other blocks within the system, the MEBI will determine the appropriate type of data access to be generated.

15.1.1 Features

The block name includes these distinctive features:

- External bus controller with four 8-bit ports A,B, E, and K
- Data and data direction registers for ports A, B, E, and K when used as general-purpose I/O
- Control register to enable/disable alternate functions on ports E and K
- Mode control register
- Control register to enable/disable pull resistors on ports A, B, E, and K
- Control register to enable/disable reduced output drive on ports A, B, E, and K
- Control register to configure external clock behavior
- Control register to configure $\overline{\text{IRQ}}$ pin operation
- Logic to capture and synchronize external interrupt pin inputs

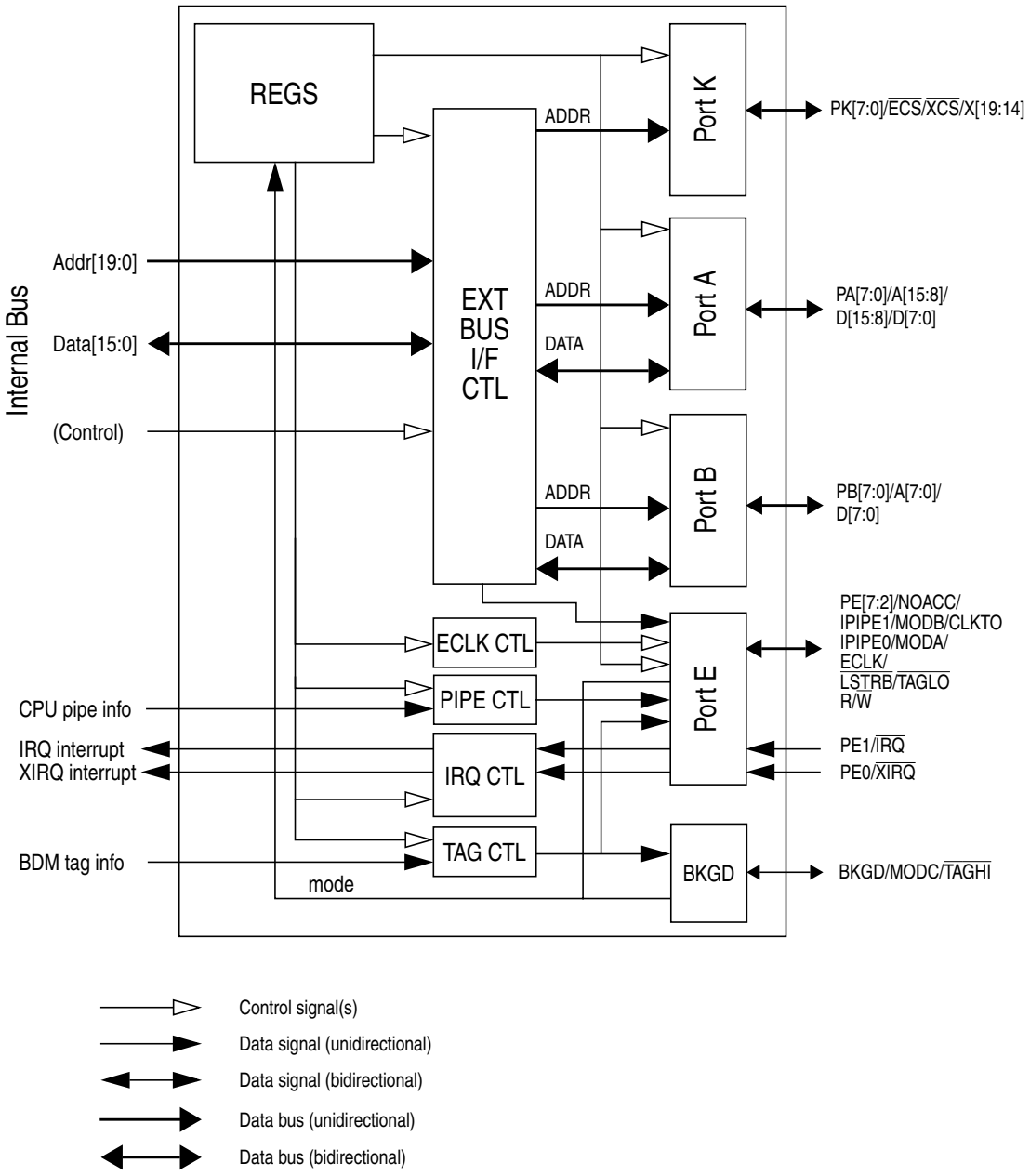


Figure 15-1. MEBI Block Diagram

15.1.2 Modes of Operation

- Normal expanded wide mode

  Ports A and B are configured as a 16-bit multiplexed address and data bus and port E provides bus control and status signals. This mode allows 16-bit external memory and peripheral devices to be interfaced to the system.
- Normal expanded narrow mode

  Ports A and B are configured as a 16-bit address bus and port A is multiplexed with 8-bit data. Port E provides bus control and status signals. This mode allows 8-bit external memory and peripheral devices to be interfaced to the system.
- Normal single-chip mode

  There is no external expansion bus in this mode. The processor program is executed from internal memory. Ports A, B, K, and most of E are available as general-purpose I/O.
- Special single-chip mode

  This mode is generally used for debugging single-chip operation, boot-strapping, or security related operations. The active background mode is in control of CPU execution and BDM firmware is waiting for additional serial commands through the BKGD pin. There is no external expansion bus after reset in this mode.
- Emulation expanded wide mode

  Developers use this mode for emulation systems in which the users target application is normal expanded wide mode.
- Emulation expanded narrow mode

  Developers use this mode for emulation systems in which the users target application is normal expanded narrow mode.
- Special test mode

  Ports A and B are configured as a 16-bit multiplexed address and data bus and port E provides bus control and status signals. In special test mode, the write protection of many control bits is lifted so that they can be thoroughly tested without needing to go through reset.
- Special peripheral mode

  This mode is intended for Freescale Semiconductor factory testing of the system. The CPU is inactive and an external (tester) bus master drives address, data, and bus control signals.

15.2 External Signal Description

In typical implementations, the MEBI sub-block of the core interfaces directly with external system pins. Some pins may not be bonded out in all implementations.

Table 15-1 outlines the pin names and functions and gives a brief description of their operation reset state of these pins and associated pull-ups or pull-downs is dependent on the mode of operation and on the integration of this block at the chip level (chip dependent).

Table 15-1. External System Pins Associated With MEBI

Pin Name	Pin Functions	Description
BKGD/MODC/ TAGHI	MODC	At the rising edge on RESET, the state of this pin is registered into the MODC bit to set the mode. (This pin always has an internal pullup.)
	BKGD	Pseudo open-drain communication pin for the single-wire background debug mode. There is an internal pull-up resistor on this pin.
	TAGHI	When instruction tagging is on, a 0 at the falling edge of E tags the high half of the instruction word being read into the instruction queue.
PA7/A15/D15/D7 thru PA0/A8/D8/D0	PA7–PA0	General-purpose I/O pins, see PORTA and DDRA registers.
	A15–A8	High-order address lines multiplexed during ECLK low. Outputs except in special peripheral mode where they are inputs from an external tester system.
	D15–D8	High-order bidirectional data lines multiplexed during ECLK high in expanded wide modes, special peripheral mode, and visible internal accesses (IVIS = 1) in emulation expanded narrow mode. Direction of data transfer is generally indicated by R/W.
	D15/D7 thru D8/D0	Alternate high-order and low-order bytes of the bidirectional data lines multiplexed during ECLK high in expanded narrow modes and narrow accesses in wide modes. Direction of data transfer is generally indicated by R/W.
PB7/A7/D7 thru PB0/A0/D0	PB7–PB0	General-purpose I/O pins, see PORTB and DDRB registers.
	A7–A0	Low-order address lines multiplexed during ECLK low. Outputs except in special peripheral mode where they are inputs from an external tester system.
	D7–D0	Low-order bidirectional data lines multiplexed during ECLK high in expanded wide modes, special peripheral mode, and visible internal accesses (with IVIS = 1) in emulation expanded narrow mode. Direction of data transfer is generally indicated by R/W.
PE7/NOACC	PE7	General-purpose I/O pin, see PORTE and DDRE registers.
	NOACC	CPU No Access output. Indicates whether the current cycle is a free cycle. Only available in expanded modes.
PE6/IPIPE1/ MODB/CLKTO	MODB	At the rising edge of RESET, the state of this pin is registered into the MODB bit to set the mode.
	PE6	General-purpose I/O pin, see PORTE and DDRE registers.
	IPIPE1	Instruction pipe status bit 1, enabled by PIPOE bit in PEAR.
	CLKTO	System clock test output. Only available in special modes. PIPOE = 1 overrides this function. The enable for this function is in the clock module.
PE5/IPIPE0/MODA	MODA	At the rising edge on RESET, the state of this pin is registered into the MODA bit to set the mode.
	PE5	General-purpose I/O pin, see PORTE and DDRE registers.
	IPIPE0	Instruction pipe status bit 0, enabled by PIPOE bit in PEAR.

Table 15-1. External System Pins Associated With MEBI (continued)

Pin Name	Pin Functions	Description
PE4/ECLK	PE4	General-purpose I/O pin, see PORTE and DDRE registers.
	ECLK	Bus timing reference clock, can operate as a free-running clock at the system clock rate or to produce one low-high clock per visible access, with the high period stretched for slow accesses. ECLK is controlled by the NECLK bit in PEAR, the IVIS bit in MODE, and the ESTR bit in EBICTL.
PE3/$\overline{\text{LSTRB}}$/ $\overline{\text{TAGLO}}$	PE3	General-purpose I/O pin, see PORTE and DDRE registers.
	$\overline{\text{LSTRB}}$	Low strobe bar, 0 indicates valid data on D7–D0.
	SZ8	In special peripheral mode, this pin is an input indicating the size of the data transfer (0 = 16-bit; 1 = 8-bit).
	$\overline{\text{TAGLO}}$	In expanded wide mode or emulation narrow modes, when instruction tagging is on and low strobe is enabled, a 0 at the falling edge of E tags the low half of the instruction word being read into the instruction queue.
PE2/R/$\overline{\text{W}}$	PE2	General-purpose I/O pin, see PORTE and DDRE registers.
	R/$\overline{\text{W}}$	Read/write, indicates the direction of internal data transfers. This is an output except in special peripheral mode where it is an input.
PE1/$\overline{\text{IRQ}}$	PE1	General-purpose input-only pin, can be read even if $\overline{\text{IRQ}}$ enabled.
	$\overline{\text{IRQ}}$	Maskable interrupt request, can be level sensitive or edge sensitive.
PE0/$\overline{\text{XIRQ}}$	PE0	General-purpose input-only pin.
	$\overline{\text{XIRQ}}$	Non-maskable interrupt input.
PK7/$\overline{\text{ECS}}$	PK7	General-purpose I/O pin, see PORTK and DDRK registers.
	$\overline{\text{ECS}}$	Emulation chip select
PK6/$\overline{\text{XCS}}$	PK6	General-purpose I/O pin, see PORTK and DDRK registers.
	$\overline{\text{XCS}}$	External data chip select
PK5/X19 thru PK0/X14	PK5–PK0	General-purpose I/O pins, see PORTK and DDRK registers.
	X19–X14	Memory expansion addresses

Detailed descriptions of these pins can be found in the device overview chapter.

15.3 Memory Map and Register Definition

A summary of the registers associated with the MEBI sub-block is shown in Table 15-2. Detailed descriptions of the registers and bits are given in the subsections that follow. On most chips the registers are mappable. Therefore, the upper bits may not be all 0s as shown in the table and descriptions.

15.3.1 Module Memory Map

Table 15-2. MEBI Memory Map

Address Offset	Use	Access
0x0000	Port A Data Register (PORTA)	R/W
0x0001	Port B Data Register (PORTB)	R/W
0x0002	Data Direction Register A (DDRA)	R/W
0x0003	Data Direction Register B (DDRB)	R/W
0x0004	Reserved	R
0x0005	Reserved	R
0x0006	Reserved	R
0x0007	Reserved	R
0x0008	Port E Data Register (PORTE)	R/W
0x0009	Data Direction Register E (DDRE)	R/W
0x000A	Port E Assignment Register (PEAR)	R/W
0x000B	Mode Register (MODE)	R/W
0x000C	Pull Control Register (PUCR)	R/W
0x000D	Reduced Drive Register (RDRIV)	R/W
0x000E	External Bus Interface Control Register (EBICTL)	R/W
0x000F	Reserved	R
0x001E	IRQ Control Register (IRQCR)	R/W
0x00032	Port K Data Register (PORTK)	R/W
0x00033	Data Direction Register K (DDRK)	R/W

15.3.2 Register Descriptions

15.3.2.1 Port A Data Register (PORTA)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0
Single Chip	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Expanded Wide, Emulation Narrow with IVIS, and Peripheral	AB/DB15	AB/DB14	AB/DB13	AB/DB12	AB/DB11	AB/DB10	AB/DB9	AB/DB8
Expanded Narrow	AB15 and DB15/DB7	AB14 and DB14/DB6	AB13 and DB13/DB5	AB12 and DB12/DB4	AB11 and DB11/DB3	AB10 and DB10/DB2	AB9 and DB9/DB1	AB8 and DB8/DB0

Figure 15-2. Port A Data Register (PORTA)

Read: Anytime when register is in the map

Write: Anytime when register is in the map

Port A bits 7 through 0 are associated with address lines A15 through A8 respectively and data lines D15/D7 through D8/D0 respectively. When this port is not used for external addresses such as in single-chip mode, these pins can be used as general-purpose I/O. Data direction register A (DDRA) determines the primary direction of each pin. DDRA also determines the source of data for a read of PORTA.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

NOTE

To ensure that you read the value present on the PORTA pins, always wait at least one cycle after writing to the DDRA register before reading from the PORTA register.

15.3.2.2 Port B Data Register (PORTB)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0
Single Chip	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
Expanded Wide, Emulation Narrow with IVIS, and Peripheral	AB/DB7	AB/DB6	AB/DB5	AB/DB4	AB/DB3	AB/DB2	AB/DB1	AB/DB0
Expanded Narrow	AB7	AB6	AB5	AB4	AB3	AB2	AB1	AB0

Figure 15-3. Port A Data Register (PORTB)

Read: Anytime when register is in the map

Write: Anytime when register is in the map

Port B bits 7 through 0 are associated with address lines A7 through A0 respectively and data lines D7 through D0 respectively. When this port is not used for external addresses, such as in single-chip mode, these pins can be used as general-purpose I/O. Data direction register B (DDRB) determines the primary direction of each pin. DDRB also determines the source of data for a read of PORTB.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

NOTE

To ensure that you read the value present on the PORTB pins, always wait at least one cycle after writing to the DDRB register before reading from the PORTB register.

15.3.2.3 Data Direction Register A (DDRA)

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

Figure 15-4. Data Direction Register A (DDRA)

Read: Anytime when register is in the map

Write: Anytime when register is in the map

This register controls the data direction for port A. When port A is operating as a general-purpose I/O port, DDRA determines the primary direction for each port A pin. A 1 causes the associated port pin to be an output and a 0 causes the associated pin to be a high-impedance input. The value in a DDR bit also affects the source of data for reads of the corresponding PORTA register. If the DDR bit is 0 (input) the buffered pin input state is read. If the DDR bit is 1 (output) the associated port data register bit state is read.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally. It is reset to 0x00 so the DDR does not override the three-state control signals.

Table 15-3. DDRA Field Descriptions

Field	Description
7:0 DDRA	**Data Direction Port A** 0 Configure the corresponding I/O pin as an input 1 Configure the corresponding I/O pin as an output

15.3.2.4 Data Direction Register B (DDRB)

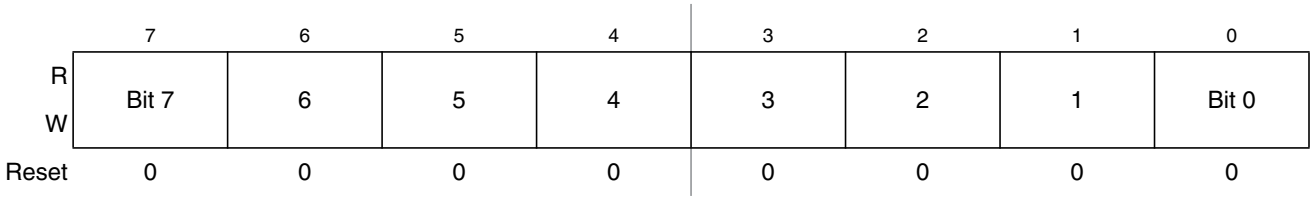


Figure 15-5. Data Direction Register B (DDRB)

Read: Anytime when register is in the map

Write: Anytime when register is in the map

This register controls the data direction for port B. When port B is operating as a general-purpose I/O port, DDRB determines the primary direction for each port B pin. A 1 causes the associated port pin to be an output and a 0 causes the associated pin to be a high-impedance input. The value in a DDR bit also affects the source of data for reads of the corresponding PORTB register. If the DDR bit is 0 (input) the buffered pin input state is read. If the DDR bit is 1 (output) the associated port data register bit state is read.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally. It is reset to 0x00 so the DDR does not override the three-state control signals.

Table 15-4. DDRB Field Descriptions

Field	Description
7:0 DDRB	**Data Direction Port B** 0 Configure the corresponding I/O pin as an input 1 Configure the corresponding I/O pin as an output

15.3.2.5 Reserved Registers

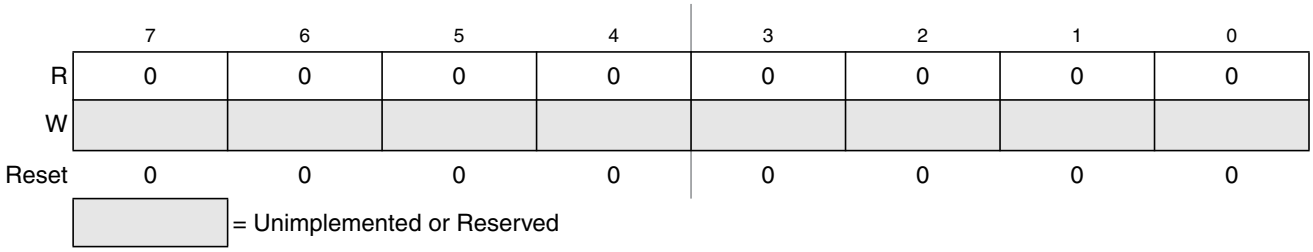


Figure 15-6. Reserved Register

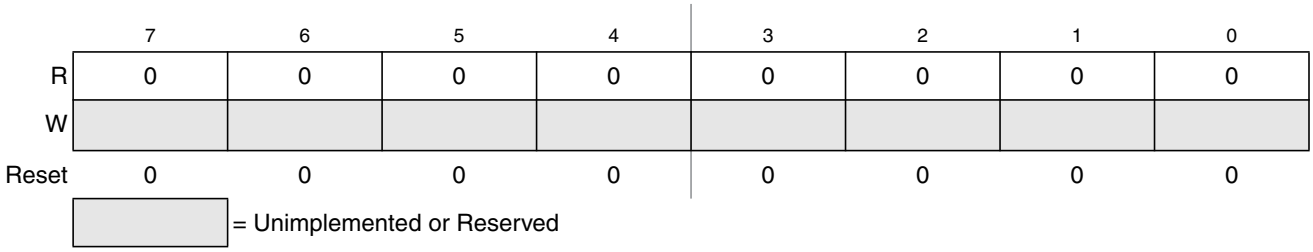


Figure 15-7. Reserved Register

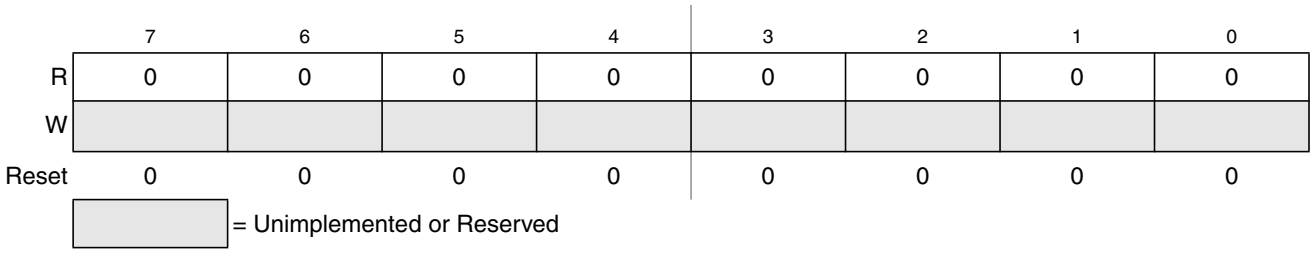


Figure 15-8. Reserved Register

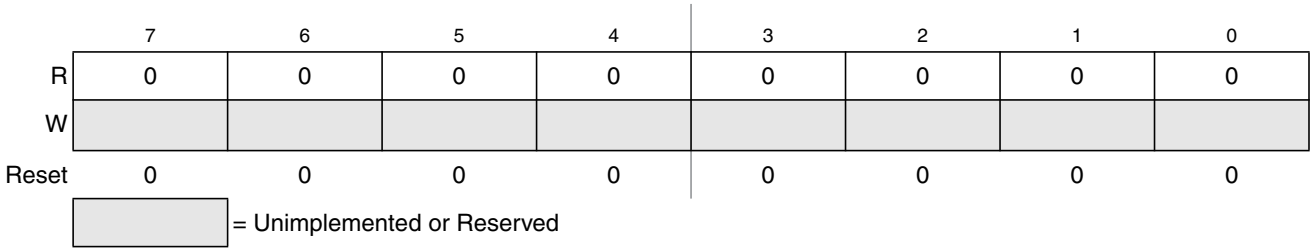


Figure 15-9. Reserved Register

These register locations are not used (reserved). All unused registers and bits in this block return logic 0s when read. Writes to these registers have no effect.

These registers are not in the on-chip map in special peripheral mode.

15.3.2.6 Port E Data Register (PORTE)

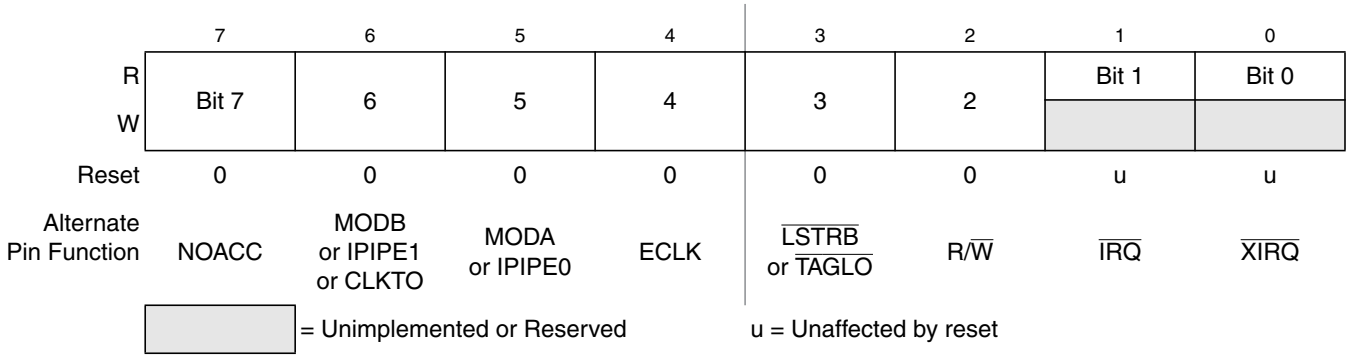


Figure 15-10. Port E Data Register (PORTE)

Read: Anytime when register is in the map

Write: Anytime when register is in the map

Port E is associated with external bus control signals and interrupt inputs. These include mode select (MODB/IPIPE1, MODA/IPIPE0), E clock, size ($\overline{\text{LSTRB}}$/$\overline{\text{TAGLO}}$), read/write (R/$\overline{\text{W}}$), $\overline{\text{IRQ}}$, and $\overline{\text{XIRQ}}$. When not used for one of these specific functions, port E pins 7:2 can be used as general-purpose I/O and pins 1:0 can be used as general-purpose input. The port E assignment register (PEAR) selects the function of each pin and DDRE determines whether each pin is an input or output when it is configured to be general-purpose I/O. DDRE also determines the source of data for a read of PORTE.

Some of these pins have software selectable pull resistors. $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$ can only be pulled up whereas the polarity of the PE7, PE4, PE3, and PE2 pull resistors are determined by chip integration. Please refer to the device overview chapter (Signal Property Summary) to determine the polarity of these resistors. A single control bit enables the pull devices for all of these pins when they are configured as inputs.

This register is not in the on-chip map in special peripheral mode or in expanded modes when the EME bit is set. Therefore, these accesses will be echoed externally.

NOTE

It is unwise to write PORTE and DDRE as a word access. If you are changing port E pins from being inputs to outputs, the data may have extra transitions during the write. It is best to initialize PORTE before enabling as outputs.

NOTE

To ensure that you read the value present on the PORTE pins, always wait at least one cycle after writing to the DDRE register before reading from the PORTE register.

15.3.2.7 Data Direction Register E (DDRE)

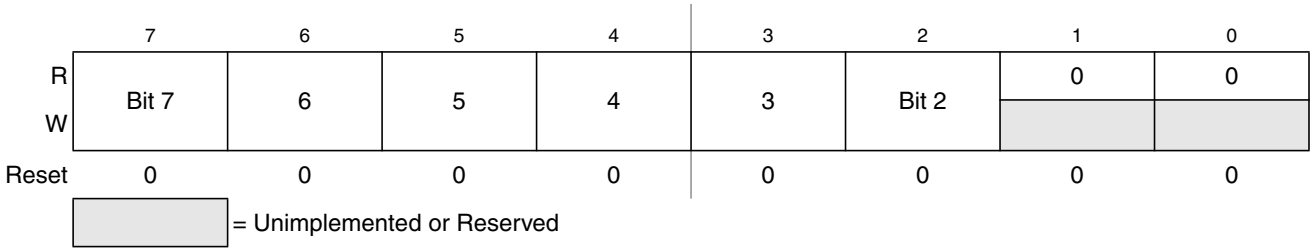


Figure 15-11. Data Direction Register E (DDRE)

Read: Anytime when register is in the map

Write: Anytime when register is in the map

Data direction register E is associated with port E. For bits in port E that are configured as general-purpose I/O lines, DDRE determines the primary direction of each of these pins. A 1 causes the associated bit to be an output and a 0 causes the associated bit to be an input. Port E bit 1 (associated with $\overline{IRQ}$) and bit 0 (associated with $\overline{XIRQ}$) cannot be configured as outputs. Port E, bits 1 and 0, can be read regardless of whether the alternate interrupt function is enabled. The value in a DDR bit also affects the source of data for reads of the corresponding PORTE register. If the DDR bit is 0 (input) the buffered pin input state is read. If the DDR bit is 1 (output) the associated port data register bit state is read.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally. Also, it is not in the map in expanded modes while the EME control bit is set.

Table 15-5. DDRE Field Descriptions

Field	Description
7:2 DDRE	**Data Direction Port E** 0 Configure the corresponding I/O pin as an input 1 Configure the corresponding I/O pin as an output **Note:** It is unwise to write PORTE and DDRE as a word access. If you are changing port E pins from inputs to outputs, the data may have extra transitions during the write. It is best to initialize PORTE before enabling as outputs.

15.3.2.8 Port E Assignment Register (PEAR)

	7	6	5	4	3	2	1	0
R	NOACCE	0	PIPOE	NECLK	LSTRE	RDWE	0	0
W								
Reset								
Special Single Chip	0	0	0	0	0	0	0	0
Special Test	0	0	1	0	1	1	0	0
Peripheral	0	0	0	0	0	0	0	0
Emulation Expanded Narrow	1	0	1	0	1	1	0	0
Emulation Expanded Wide	1	0	1	0	1	1	0	0
Normal Single Chip	0	0	0	1	0	0	0	0
Normal Expanded Narrow	0	0	0	0	0	0	0	0
Normal Expanded Wide	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-12. Port E Assignment Register (PEAR)

Read: Anytime (provided this register is in the map).

Write: Each bit has specific write conditions. Please refer to the descriptions of each bit on the following pages.

Port E serves as general-purpose I/O or as system and bus control signals. The PEAR register is used to choose between the general-purpose I/O function and the alternate control functions. When an alternate control function is selected, the associated DDRE bits are overridden.

The reset condition of this register depends on the mode of operation because bus control signals are needed immediately after reset in some modes. In normal single-chip mode, no external bus control signals are needed so all of port E is configured for general-purpose I/O. In normal expanded modes, only the E clock is configured for its alternate bus control function and the other bits of port E are configured for general-purpose I/O. As the reset vector is located in external memory, the E clock is required for this access. R/$\overline{W}$ is only needed by the system when there are external writable resources. If the normal expanded system needs any other bus control signals, PEAR would need to be written before any access that needed the additional signals. In special test and emulation modes, IPIPE1, IPIPE0, E, $\overline{LSTRB}$, and R/$\overline{W}$ are configured out of reset as bus control signals.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

Table 15-6. PEAR Field Descriptions

Field	Description
7 NOACCE	**CPU No Access Output Enable** Normal: write once Emulation: write never Special: write anytime 1 The associated pin (port E, bit 7) is general-purpose I/O. 0 The associated pin (port E, bit 7) is output and indicates whether the cycle is a CPU free cycle. This bit has no effect in single-chip or special peripheral modes.
5 PIPOE	**Pipe Status Signal Output Enable** Normal: write once Emulation: write never Special: write anytime. 0 The associated pins (port E, bits 6:5) are general-purpose I/O. 1 The associated pins (port E, bits 6:5) are outputs and indicate the state of the instruction queue This bit has no effect in single-chip or special peripheral modes.
4 NECLK	**No External E Clock** Normal and special: write anytime Emulation: write never 0 The associated pin (port E, bit 4) is the external E clock pin. External E clock is free-running if ESTR = 0 1 The associated pin (port E, bit 4) is a general-purpose I/O pin. External E clock is available as an output in all modes.
3 LSTRE	**Low Strobe ($\overline{\text{LSTRB}}$) Enable** Normal: write once Emulation: write never Special: write anytime. 0 The associated pin (port E, bit 3) is a general-purpose I/O pin. 1 The associated pin (port E, bit 3) is configured as the $\overline{\text{LSTRB}}$ bus control output. If BDM tagging is enabled, $\overline{\text{TAGLO}}$ is multiplexed in on the rising edge of ECLK and $\overline{\text{LSTRB}}$ is driven out on the falling edge of ECLK. This bit has no effect in single-chip, peripheral, or normal expanded narrow modes. **Note:** $\overline{\text{LSTRB}}$ is used during external writes. After reset in normal expanded mode, $\overline{\text{LSTRB}}$ is disabled to provide an extra I/O pin. If $\overline{\text{LSTRB}}$ is needed, it should be enabled before any external writes. External reads do not normally need $\overline{\text{LSTRB}}$ because all 16 data bits can be driven even if the system only needs 8 bits of data.
2 RDWE	**Read/Write Enable** Normal: write once Emulation: write never Special: write anytime 0 The associated pin (port E, bit 2) is a general-purpose I/O pin. 1 The associated pin (port E, bit 2) is configured as the R/$\overline{\text{W}}$ pin This bit has no effect in single-chip or special peripheral modes. **Note:** R/$\overline{\text{W}}$ is used for external writes. After reset in normal expanded mode, R/$\overline{\text{W}}$ is disabled to provide an extra I/O pin. If R/$\overline{\text{W}}$ is needed it should be enabled before any external writes.

15.3.2.9 Mode Register (MODE)

	7	6	5	4	3	2	1	0
R	MODC	MODB	MODA	0	IVIS	0	EMK	EME
W								
Reset								
Special Single Chip	0	0	0	0	0	0	0	0
Emulation Expanded Narrow	0	0	1	0	1	0	1	1
Special Test	0	1	0	0	1	0	0	0
Emulation Expanded Wide	0	1	1	0	1	0	1	1
Normal Single Chip	1	0	0	0	0	0	0	0
Normal Expanded Narrow	1	0	1	0	0	0	0	0
Peripheral	1	1	0	0	0	0	0	0
Normal Expanded Wide	1	1	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-13. Mode Register (MODE)

Read: Anytime (provided this register is in the map).

Write: Each bit has specific write conditions. Please refer to the descriptions of each bit on the following pages.

The MODE register is used to establish the operating mode and other miscellaneous functions (i.e., internal visibility and emulation of port E and K).

In special peripheral mode, this register is not accessible but it is reset as shown to system configuration features. Changes to bits in the MODE register are delayed one cycle after the write.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

Table 15-7. MODE Field Descriptions

Field	Description
7:5 MOD[C:A]	**Mode Select Bits** — These bits indicate the current operating mode. If MODA = 1, then MODC, MODB, and MODA are write never. If MODC = MODA = 0, then MODC, MODB, and MODA are writable with the exception that you cannot change to or from special peripheral mode If MODC = 1, MODB = 0, and MODA = 0, then MODC is write never. MODB and MODA are write once, except that you cannot change to special peripheral mode. From normal single-chip, only normal expanded narrow and normal expanded wide modes are available. See Table 15-8 and Table 15-16.
3 IVIS	**Internal Visibility (for both read and write accesses)** — This bit determines whether internal accesses generate a bus cycle that is visible on the external bus. Normal: write once Emulation: write never Special: write anytime 0 No visibility of internal bus operations on external bus. 1 Internal bus operations are visible on external bus.
1 EMK	**Emulate Port K** Normal: write once Emulation: write never Special: write anytime 0 PORTK and DDRK are in the memory map so port K can be used for general-purpose I/O. 1 If in any expanded mode, PORTK and DDRK are removed from the memory map. In single-chip modes, PORTK and DDRK are always in the map regardless of the state of this bit. In special peripheral mode, PORTK and DDRK are never in the map regardless of the state of this bit.
0 EME	**Emulate Port E** Normal and Emulation: write never Special: write anytime 0 PORTE and DDRE are in the memory map so port E can be used for general-purpose I/O. 1 If in any expanded mode or special peripheral mode, PORTE and DDRE are removed from the memory map. Removing the registers from the map allows the user to emulate the function of these registers externally. In single-chip modes, PORTE and DDRE are always in the map regardless of the state of this bit.

Table 15-8. MODC, MODB, and MODA Write Capability[1]

MODC	MODB	MODA	Mode	MODx Write Capability
0	0	0	Special single chip	MODC, MODB, and MODA write anytime but not to 110[2]
0	0	1	Emulation narrow	No write
0	1	0	Special test	MODC, MODB, and MODA write anytime but not to 110[(2)]
0	1	1	Emulation wide	No write
1	0	0	Normal single chip	MODC write never, MODB and MODA write once but not to 110
1	0	1	Normal expanded narrow	No write
1	1	0	Special peripheral	No write
1	1	1	Normal expanded wide	No write

1 No writes to the MOD bits are allowed while operating in a secure mode. For more details, refer to the device overview chapter.

2 If you are in a special single-chip or special test mode and you write to this register, changing to normal single-chip mode, then one allowed write to this register remains. If you write to normal expanded or emulation mode, then no writes remain.

15.3.2.10 Pull Control Register (PUCR)

	7	6	5	4	3	2	1	0
R	PUPKE	0	0	PUPEE	0	0	PUPBE	PUPAE
W								
Reset[1]	1	0	0	1	0	0	0	0

NOTES:
1. The default value of this parameter is shown. Please refer to the device overview chapter to determine the actual reset state of this register.

= Unimplemented or Reserved

Figure 15-14. Pull Control Register (PUCR)

Read: Anytime (provided this register is in the map).

Write: Anytime (provided this register is in the map).

This register is used to select pull resistors for the pins associated with the core ports. Pull resistors are assigned on a per-port basis and apply to any pin in the corresponding port that is currently configured as an input. The polarity of these pull resistors is determined by chip integration. Please refer to the device overview chapter to determine the polarity of these resistors.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

NOTE

These bits have no effect when the associated pin(s) are outputs. (The pull resistors are inactive.)

Table 15-9. PUCR Field Descriptions

Field	Description
7 PUPKE	**Pull resistors Port K Enable** 0 Port K pull resistors are disabled. 1 Enable pull resistors for port K input pins.
4 PUPEE	**Pull resistors Port E Enable** 0 Port E pull resistors on bits 7, 4:0 are disabled. 1 Enable pull resistors for port E input pins bits 7, 4:0. **Note:** Pins 5 and 6 of port E have pull resistors which are only enabled during reset. This bit has no effect on these pins.
1 PUPBE	**Pull resistors Port B Enable** 0 Port B pull resistors are disabled. 1 Enable pull resistors for all port B input pins.
0 PUPAE	**Pull resistors Port A Enable** 0 Port A pull resistors are disabled. 1 Enable pull resistors for all port A input pins.

15.3.2.11 Reduced Drive Register (RDRIV)

	7	6	5	4	3	2	1	0
R	RDRK	0	0	RDPE	0	0	RDPB	RDPA
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-15. Reduced Drive Register (RDRIV)

Read: Anytime (provided this register is in the map)

Write: Anytime (provided this register is in the map)

This register is used to select reduced drive for the pins associated with the core ports. This gives reduced power consumption and reduced RFI with a slight increase in transition time (depending on loading). This feature would be used on ports which have a light loading. The reduced drive function is independent of which function is being used on a particular port.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

Table 15-10. RDRIV Field Descriptions

Field	Description
7 RDRK	**Reduced Drive of Port K** 0 All port K output pins have full drive enabled. 1 All port K output pins have reduced drive enabled.
4 RDPE	**Reduced Drive of Port E** 0 All port E output pins have full drive enabled. 1 All port E output pins have reduced drive enabled.
1 RDPB	**Reduced Drive of Port B** 0 All port B output pins have full drive enabled. 1 All port B output pins have reduced drive enabled.
0 RDPA	**Reduced Drive of Ports A** 0 All port A output pins have full drive enabled. 1 All port A output pins have reduced drive enabled.

15.3.2.12 External Bus Interface Control Register (EBICTL)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	ESTR
W								
Reset:								
Peripheral	0	0	0	0	0	0	0	0
All other modes	0	0	0	0	0	0	0	1

= Unimplemented or Reserved

Figure 15-16. External Bus Interface Control Register (EBICTL)

Read: Anytime (provided this register is in the map)

Write: Refer to individual bit descriptions below

The EBICTL register is used to control miscellaneous functions (i.e., stretching of external E clock).

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

Table 15-11. EBICTL Field Descriptions

Field	Description
0 ESTR	**E Clock Stretches** — This control bit determines whether the E clock behaves as a simple free-running clock or as a bus control signal that is active only for external bus cycles. Normal and Emulation: write once Special: write anytime 0 E never stretches (always free running). 1 E stretches high during stretched external accesses and remains low during non-visible internal accesses. This bit has no effect in single-chip modes.

15.3.2.13 Reserved Register

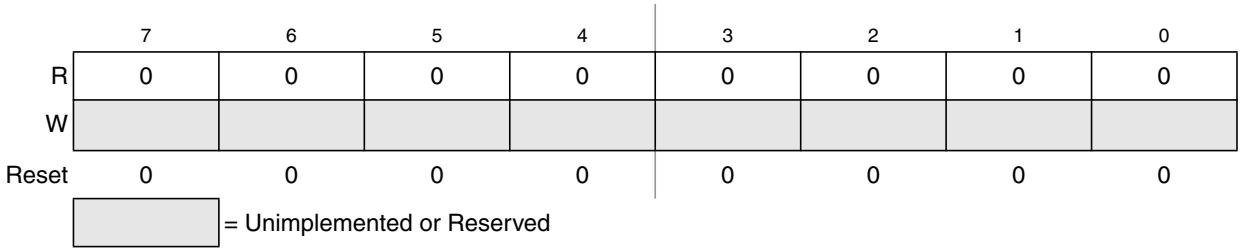


Figure 15-17. Reserved Register

This register location is not used (reserved). All bits in this register return logic 0s when read. Writes to this register have no effect.

This register is not in the on-chip memory map in expanded and special peripheral modes. Therefore, these accesses will be echoed externally.

15.3.2.14 IRQ Control Register (IRQCR)

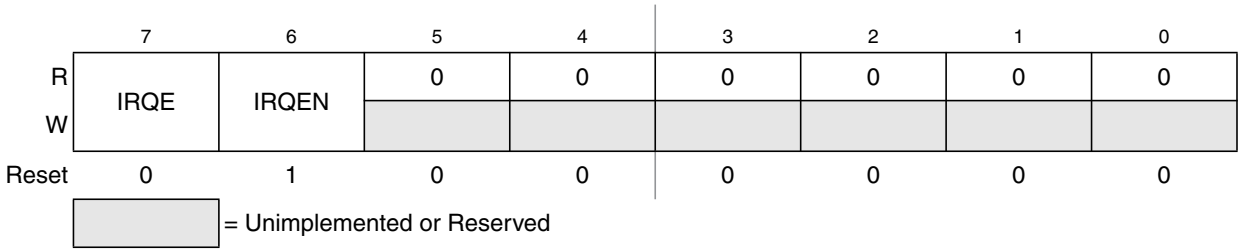


Figure 15-18. IRQ Control Register (IRQCR)

Read: See individual bit descriptions below

Write: See individual bit descriptions below

Table 15-12. IRQCR Field Descriptions

Field	Description
7 IRQE	**IRQ Select Edge Sensitive Only** Special modes: read or write anytime Normal and Emulation modes: read anytime, write once 0 IRQ configured for low level recognition. 1 IRQ configured to respond only to falling edges. Falling edges on the IRQ pin will be detected anytime IRQE = 1 and will be cleared only upon a reset or the servicing of the IRQ interrupt.
6 IRQEN	**External IRQ Enable** Normal, emulation, and special modes: read or write anytime 0 External IRQ pin is disconnected from interrupt logic. 1 External IRQ pin is connected to interrupt logic. **Note:** When IRQEN = 0, the edge detect latch is disabled.

15.3.2.15 Port K Data Register (PORTK)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0
Alternate Pin Function	$\overline{ECS}$	$\overline{XCS}$	XAB19	XAB18	XAB17	XAB16	XAB15	XAB14

Figure 15-19. Port K Data Register (PORTK)

Read: Anytime

Write: Anytime

This port is associated with the internal memory expansion emulation pins. When the port is not enabled to emulate the internal memory expansion, the port pins are used as general-purpose I/O. When port K is operating as a general-purpose I/O port, DDRK determines the primary direction for each port K pin. A 1 causes the associated port pin to be an output and a 0 causes the associated pin to be a high-impedance input. The value in a DDR bit also affects the source of data for reads of the corresponding PORTK register. If the DDR bit is 0 (input) the buffered pin input is read. If the DDR bit is 1 (output) the output of the port data register is read.

This register is not in the map in peripheral or expanded modes while the EMK control bit in MODE register is set. Therefore, these accesses will be echoed externally.

When inputs, these pins can be selected to be high impedance or pulled up, based upon the state of the PUPKE bit in the PUCR register.

Table 15-13. PORTK Field Descriptions

Field	Description
7 Port K, Bit 7	**Port K, Bit 7** — This bit is used as an emulation chip select signal for the emulation of the internal memory expansion, or as general-purpose I/O, depending upon the state of the EMK bit in the MODE register. While this bit is used as a chip select, the external bit will return to its de-asserted state (V_{DD}) for approximately 1/4 cycle just after the negative edge of ECLK, unless the external access is stretched and ECLK is free-running (ESTR bit in EBICTL = 0). See the MMC block description chapter for additional details on when this signal will be active.
6 Port K, Bit 6	**Port K, Bit 6** — This bit is used as an external chip select signal for most external accesses that are not selected by $\overline{ECS}$ (see the MMC block description chapter for more details), depending upon the state the of the EMK bit in the MODE register. While this bit is used as a chip select, the external pin will return to its de-asserted state (V_{DD}) for approximately 1/4 cycle just after the negative edge of ECLK, unless the external access is stretched and ECLK is free-running (ESTR bit in EBICTL = 0).
5:0 Port K, Bits 5:0	**Port K, Bits 5:0** — These six bits are used to determine which FLASH/ROM or external memory array page is being accessed. They can be viewed as expanded addresses XAB19–XAB14 of the 20-bit address used to access up to1M byte internal FLASH/ROM or external memory array. Alternatively, these bits can be used for general-purpose I/O depending upon the state of the EMK bit in the MODE register.

15.3.2.16 Port K Data Direction Register (DDRK)

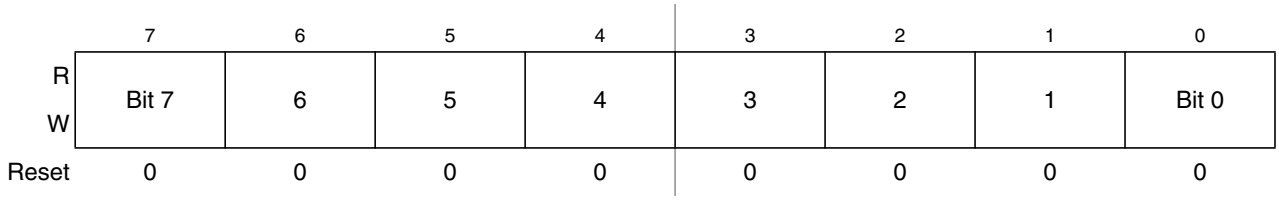


Figure 15-20. Port K Data Direction Register (DDRK)

Read: Anytime

Write: Anytime

This register determines the primary direction for each port K pin configured as general-purpose I/O. This register is not in the map in peripheral or expanded modes while the EMK control bit in MODE register is set. Therefore, these accesses will be echoed externally.

Table 15-14. EBICTL Field Descriptions

Field	Description
7:0 DDRK	**Data Direction Port K Bits** 0 Associated pin is a high-impedance input 1 Associated pin is an output **Note:** It is unwise to write PORTK and DDRK as a word access. If you are changing port K pins from inputs to outputs, the data may have extra transitions during the write. It is best to initialize PORTK before enabling as outputs. **Note:** To ensure that you read the correct value from the PORTK pins, always wait at least one cycle after writing to the DDRK register before reading from the PORTK register.

15.4 Functional Description

15.4.1 Detecting Access Type from External Signals

The external signals $\overline{\text{LSTRB}}$, R/$\overline{\text{W}}$, and AB0 indicate the type of bus access that is taking place. Accesses to the internal RAM module are the only type of access that would produce $\overline{\text{LSTRB}}$ = AB0 = 1, because the internal RAM is specifically designed to allow misaligned 16-bit accesses in a single cycle. In these cases the data for the address that was accessed is on the low half of the data bus and the data for address + 1 is on the high half of the data bus. This is summarized in Table 15-15.

Table 15-15. Access Type vs. Bus Control Pins

$\overline{\text{LSTRB}}$	AB0	R/$\overline{\text{W}}$	Type of Access
1	0	1	8-bit read of an even address
0	1	1	8-bit read of an odd address
1	0	0	8-bit write of an even address
0	1	0	8-bit write of an odd address
0	0	1	16-bit read of an even address

Table 15-15. Access Type vs. Bus Control Pins

$\overline{\text{LSTRB}}$	AB0	R/$\overline{\text{W}}$	Type of Access
1	1	1	16-bit read of an odd address (low/high data swapped)
0	0	0	16-bit write to an even address
1	1	0	16-bit write to an odd address (low/high data swapped)

15.4.2 Stretched Bus Cycles

In order to allow fast internal bus cycles to coexist in a system with slower external memory resources, the HCS12 supports the concept of stretched bus cycles (module timing reference clocks for timers and baud rate generators are not affected by this stretching). Control bits in the MISC register in the MMC sub-block of the core specify the amount of stretch (0, 1, 2, or 3 periods of the internal bus-rate clock). While stretching, the CPU state machines are all held in their current state. At this point in the CPU bus cycle, write data would already be driven onto the data bus so the length of time write data is valid is extended in the case of a stretched bus cycle. Read data would not be captured by the system until the E clock falling edge. In the case of a stretched bus cycle, read data is not required until the specified setup time before the falling edge of the stretched E clock. The chip selects, and R/$\overline{\text{W}}$ signals remain valid during the period of stretching (throughout the stretched E high time).

NOTE

The address portion of the bus cycle is not stretched.

15.4.3 Modes of Operation

The operating mode out of reset is determined by the states of the MODC, MODB, and MODA pins during reset (Table 15-16). The MODC, MODB, and MODA bits in the MODE register show the current operating mode and provide limited mode switching during operation. The states of the MODC, MODB, and MODA pins are latched into these bits on the rising edge of the reset signal.

Table 15-16. Mode Selection

MODC	MODB	MODA	Mode Description
0	0	0	Special Single Chip, BDM allowed and ACTIVE. BDM is allowed in all other modes but a serial command is required to make BDM active.
0	0	1	Emulation Expanded Narrow, BDM allowed
0	1	0	Special Test (Expanded Wide), BDM allowed
0	1	1	Emulation Expanded Wide, BDM allowed
1	0	0	Normal Single Chip, BDM allowed
1	0	1	Normal Expanded Narrow, BDM allowed
1	1	0	Peripheral; BDM allowed but bus operations would cause bus conflicts (must not be used)
1	1	1	Normal Expanded Wide, BDM allowed

There are two basic types of operating modes:

1. Normal modes: Some registers and bits are protected against accidental changes.
2. Special modes: Allow greater access to protected control registers and bits for special purposes such as testing.

A system development and debug feature, background debug mode (BDM), is available in all modes. In special single-chip mode, BDM is active immediately after reset.

Some aspects of Port E are not mode dependent. Bit 1 of Port E is a general purpose input or the $\overline{\text{IRQ}}$ interrupt input. $\overline{\text{IRQ}}$ can be enabled by bits in the CPU's condition codes register but it is inhibited at reset so this pin is initially configured as a simple input with a pull-up. Bit 0 of Port E is a general purpose input or the $\overline{\text{XIRQ}}$ interrupt input. $\overline{\text{XIRQ}}$ can be enabled by bits in the CPU's condition codes register but it is inhibited at reset so this pin is initially configured as a simple input with a pull-up. The ESTR bit in the EBICTL register is set to one by reset in any user mode. This assures that the reset vector can be fetched even if it is located in an external slow memory device. The PE6/MODB/IPIPE1 and PE5/MODA/IPIPE0 pins act as high-impedance mode select inputs during reset.

The following paragraphs discuss the default bus setup and describe which aspects of the bus can be changed after reset on a per mode basis.

15.4.3.1 Normal Operating Modes

These modes provide three operating configurations. Background debug is available in all three modes, but must first be enabled for some operations by means of a BDM background command, then activated.

15.4.3.1.1 Normal Single-Chip Mode

There is no external expansion bus in this mode. All pins of Ports A, B and E are configured as general purpose I/O pins Port E bits 1 and 0 are available as general purpose input only pins with internal pull resistors enabled. All other pins of Port E are bidirectional I/O pins that are initially configured as high-impedance inputs with internal pull resistors enabled. Ports A and B are configured as high-impedance inputs with their internal pull resistors disabled.

The pins associated with Port E bits 6, 5, 3, and 2 cannot be configured for their alternate functions IPIPE1, IPIPE0, $\overline{\text{LSTRB}}$, and R/$\overline{\text{W}}$ while the MCU is in single chip modes. In single chip modes, the associated control bits PIPOE, LSTRE, and RDWE are reset to zero. Writing the opposite state into them in single chip mode does not change the operation of the associated Port E pins.

In normal single chip mode, the MODE register is writable one time. This allows a user program to change the bus mode to narrow or wide expanded mode and/or turn on visibility of internal accesses.

Port E, bit 4 can be configured for a free-running E clock output by clearing NECLK=0. Typically the only use for an E clock output while the MCU is in single chip modes would be to get a constant speed clock for use in the external application system.

15.4.3.1.2 Normal Expanded Wide Mode

In expanded wide modes, Ports A and B are configured as a 16-bit multiplexed address and data bus and Port E bit 4 is configured as the E clock output signal. These signals allow external memory and peripheral devices to be interfaced to the MCU.

Port E pins other than PE4/ECLK are configured as general purpose I/O pins (initially high-impedance inputs with internal pull resistors enabled). Control bits PIPOE, NECLK, LSTRE, and RDWE in the PEAR register can be used to configure Port E pins to act as bus control outputs instead of general purpose I/O pins.

It is possible to enable the pipe status signals on Port E bits 6 and 5 by setting the PIPOE bit in PEAR, but it would be unusual to do so in this mode. Development systems where pipe status signals are monitored would typically use the special variation of this mode.

The Port E bit 2 pin can be reconfigured as the R/$\overline{W}$ bus control signal by writing “1” to the RDWE bit in PEAR. If the expanded system includes external devices that can be written, such as RAM, the RDWE bit would need to be set before any attempt to write to an external location. If there are no writable resources in the external system, PE2 can be left as a general purpose I/O pin.

The Port E bit 3 pin can be reconfigured as the $\overline{\text{LSTRB}}$ bus control signal by writing “1” to the LSTRE bit in PEAR. The default condition of this pin is a general purpose input because the $\overline{\text{LSTRB}}$ function is not needed in all expanded wide applications.

The Port E bit 4 pin is initially configured as ECLK output with stretch. The E clock output function depends upon the settings of the NECLK bit in the PEAR register, the IVIS bit in the MODE register and the ESTR bit in the EBICTL register. The E clock is available for use in external select decode logic or as a constant speed clock for use in the external application system.

15.4.3.1.3 Normal Expanded Narrow Mode

This mode is used for lower cost production systems that use 8-bit wide external EPROMs or RAMs. Such systems take extra bus cycles to access 16-bit locations but this may be preferred over the extra cost of additional external memory devices.

Ports A and B are configured as a 16-bit address bus and Port A is multiplexed with data. Internal visibility is not available in this mode because the internal cycles would need to be split into two 8-bit cycles.

Since the PEAR register can only be written one time in this mode, use care to set all bits to the desired states during the single allowed write.

The PE3/$\overline{\text{LSTRB}}$ pin is always a general purpose I/O pin in normal expanded narrow mode. Although it is possible to write the LSTRE bit in PEAR to “1” in this mode, the state of LSTRE is overridden and Port E bit 3 cannot be reconfigured as the $\overline{\text{LSTRB}}$ output.

It is possible to enable the pipe status signals on Port E bits 6 and 5 by setting the PIPOE bit in PEAR, but it would be unusual to do so in this mode. LSTRB would also be needed to fully understand system activity. Development systems where pipe status signals are monitored would typically use special expanded wide mode or occasionally special expanded narrow mode.

The PE4/ECLK pin is initially configured as ECLK output with stretch. The E clock output function depends upon the settings of the NECLK bit in the PEAR register, the IVIS bit in the MODE register and the ESTR bit in the EBICTL register. In normal expanded narrow mode, the E clock is available for use in external select decode logic or as a constant speed clock for use in the external application system.

The PE2/R/W pin is initially configured as a general purpose input with an internal pull resistor enabled but this pin can be reconfigured as the R/$\overline{W}$ bus control signal by writing “1” to the RDWE bit in PEAR. If the expanded narrow system includes external devices that can be written such as RAM, the RDWE bit would need to be set before any attempt to write to an external location. If there are no writable resources in the external system, PE2 can be left as a general purpose I/O pin.

15.4.3.1.4 Emulation Expanded Wide Mode

In expanded wide modes, Ports A and B are configured as a 16-bit multiplexed address and data bus and Port E provides bus control and status signals. These signals allow external memory and peripheral devices to be interfaced to the MCU. These signals can also be used by a logic analyzer to monitor the progress of application programs.

The bus control related pins in Port E (PE7/NOACC, PE6/MODB/IPIPE1, PE5/MODA/IPIPE0, PE4/ECLK, PE3/$\overline{LSTRB}$/$\overline{TAGLO}$, and PE2/R/$\overline{W}$) are all configured to serve their bus control output functions rather than general purpose I/O. Notice that writes to the bus control enable bits in the PEAR register in emulation mode are restricted.

15.4.3.1.5 Emulation Expanded Narrow Mode

Expanded narrow modes are intended to allow connection of single 8-bit external memory devices for lower cost systems that do not need the performance of a full 16-bit external data bus. Accesses to internal resources that have been mapped external (i.e. PORTA, PORTB, DDRA, DDRB, PORTE, DDRE, PEAR, PUCR, RDRIV) will be accessed with a 16-bit data bus on Ports A and B. Accesses of 16-bit external words to addresses which are normally mapped external will be broken into two separate 8-bit accesses using Port A as an 8-bit data bus. Internal operations continue to use full 16-bit data paths. They are only visible externally as 16-bit information if IVIS=1.

Ports A and B are configured as multiplexed address and data output ports. During external accesses, address A15, data D15 and D7 are associated with PA7, address A0 is associated with PB0 and data D8 and D0 are associated with PA0. During internal visible accesses and accesses to internal resources that have been mapped external, address A15 and data D15 is associated with PA7 and address A0 and data D0 is associated with PB0.

The bus control related pins in Port E (PE7/NOACC, PE6/MODB/IPIPE1, PE5/MODA/IPIPE0, PE4/ECLK, PE3/$\overline{LSTRB}$/$\overline{TAGLO}$, and PE2/R/$\overline{W}$) are all configured to serve their bus control output functions rather than general purpose I/O. Notice that writes to the bus control enable bits in the PEAR register in emulation mode are restricted.

The main difference between special modes and normal modes is that some of the bus control and system control signals cannot be written in emulation modes.

15.4.3.2 Special Operating Modes

There are two special operating modes that correspond to normal operating modes. These operating modes are commonly used in factory testing and system development.

15.4.3.2.1 Special Single-Chip Mode

When the MCU is reset in this mode, the background debug mode is enabled and active. The MCU does not fetch the reset vector and execute application code as it would in other modes. Instead the active background mode is in control of CPU execution and BDM firmware is waiting for additional serial commands through the BKGD pin. When a serial command instructs the MCU to return to normal execution, the system will be configured as described below unless the reset states of internal control registers have been changed through background commands after the MCU was reset.

There is no external expansion bus after reset in this mode. Ports A and B are initially simple bidirectional I/O pins that are configured as high-impedance inputs with internal pull resistors disabled; however, writing to the mode select bits in the MODE register (which is allowed in special modes) can change this after reset. All of the Port E pins (except PE4/ECLK) are initially configured as general purpose high-impedance inputs with internal pull resistors enabled. PE4/ECLK is configured as the E clock output in this mode.

The pins associated with Port E bits 6, 5, 3, and 2 cannot be configured for their alternate functions IPIPE1, IPIPE0, $\overline{\text{LSTRB}}$, and R/$\overline{\text{W}}$ while the MCU is in single chip modes. In single chip modes, the associated control bits PIPOE, LSTRE and RDWE are reset to zero. Writing the opposite value into these bits in single chip mode does not change the operation of the associated Port E pins.

Port E, bit 4 can be configured for a free-running E clock output by clearing NECLK=0. Typically the only use for an E clock output while the MCU is in single chip modes would be to get a constant speed clock for use in the external application system.

15.4.3.2.2 Special Test Mode

In expanded wide modes, Ports A and B are configured as a 16-bit multiplexed address and data bus and Port E provides bus control and status signals. In special test mode, the write protection of many control bits is lifted so that they can be thoroughly tested without needing to go through reset.

15.4.3.3 Test Operating Mode

There is a test operating mode in which an external master, such as an I.C. tester, can control the on-chip peripherals.

15.4.3.3.1 Peripheral Mode

This mode is intended for factory testing of the MCU. In this mode, the CPU is inactive and an external (tester) bus master drives address, data and bus control signals in through Ports A, B and E. In effect, the whole MCU acts as if it was a peripheral under control of an external CPU. This allows faster testing of on-chip memory and peripherals than previous testing methods. Since the mode control register is not accessible in peripheral mode, the only way to change to another mode is to reset the MCU into a different

mode. Background debugging should not be used while the MCU is in special peripheral mode as internal bus conflicts between BDM and the external master can cause improper operation of both functions.

15.4.4 Internal Visibility

Internal visibility is available when the MCU is operating in expanded wide modes or emulation narrow mode. It is not available in single-chip, peripheral or normal expanded narrow modes. Internal visibility is enabled by setting the IVIS bit in the MODE register.

If an internal access is made while E, R/$\overline{W}$, and $\overline{LSTRB}$ are configured as bus control outputs and internal visibility is off (IVIS=0), E will remain low for the cycle, R/$\overline{W}$ will remain high, and address, data and the $\overline{LSTRB}$ pins will remain at their previous state.

When internal visibility is enabled (IVIS=1), certain internal cycles will be blocked from going external. During cycles when the BDM is selected, R/$\overline{W}$ will remain high, data will maintain its previous state, and address and $\overline{LSTRB}$ pins will be updated with the internal value. During CPU no access cycles when the BDM is not driving, R/$\overline{W}$ will remain high, and address, data and the $\overline{LSTRB}$ pins will remain at their previous state.

NOTE

When the system is operating in a secure mode, internal visibility is not available (i.e., IVIS = 1 has no effect). Also, the IPIPE signals will not be visible, regardless of operating mode. IPIPE1–IPIPE0 will display 0es if they are enabled. In addition, the MOD bits in the MODE control register cannot be written.

15.4.5 Low-Power Options

The MEBI does not contain any user-controlled options for reducing power consumption. The operation of the MEBI in low-power modes is discussed in the following subsections.

15.4.5.1 Operation in Run Mode

The MEBI does not contain any options for reducing power in run mode; however, the external addresses are conditioned to reduce power in single-chip modes. Expanded bus modes will increase power consumption.

15.4.5.2 Operation in Wait Mode

The MEBI does not contain any options for reducing power in wait mode.

15.4.5.3 Operation in Stop Mode

The MEBI will cease to function after execution of a CPU STOP instruction.

Chapter 16
Module Mapping Control (MMCV4)

16.1 Introduction

This section describes the functionality of the module mapping control (MMC) sub-block of the S12 core platform.

The block diagram of the MMC is shown in Figure 16-1.

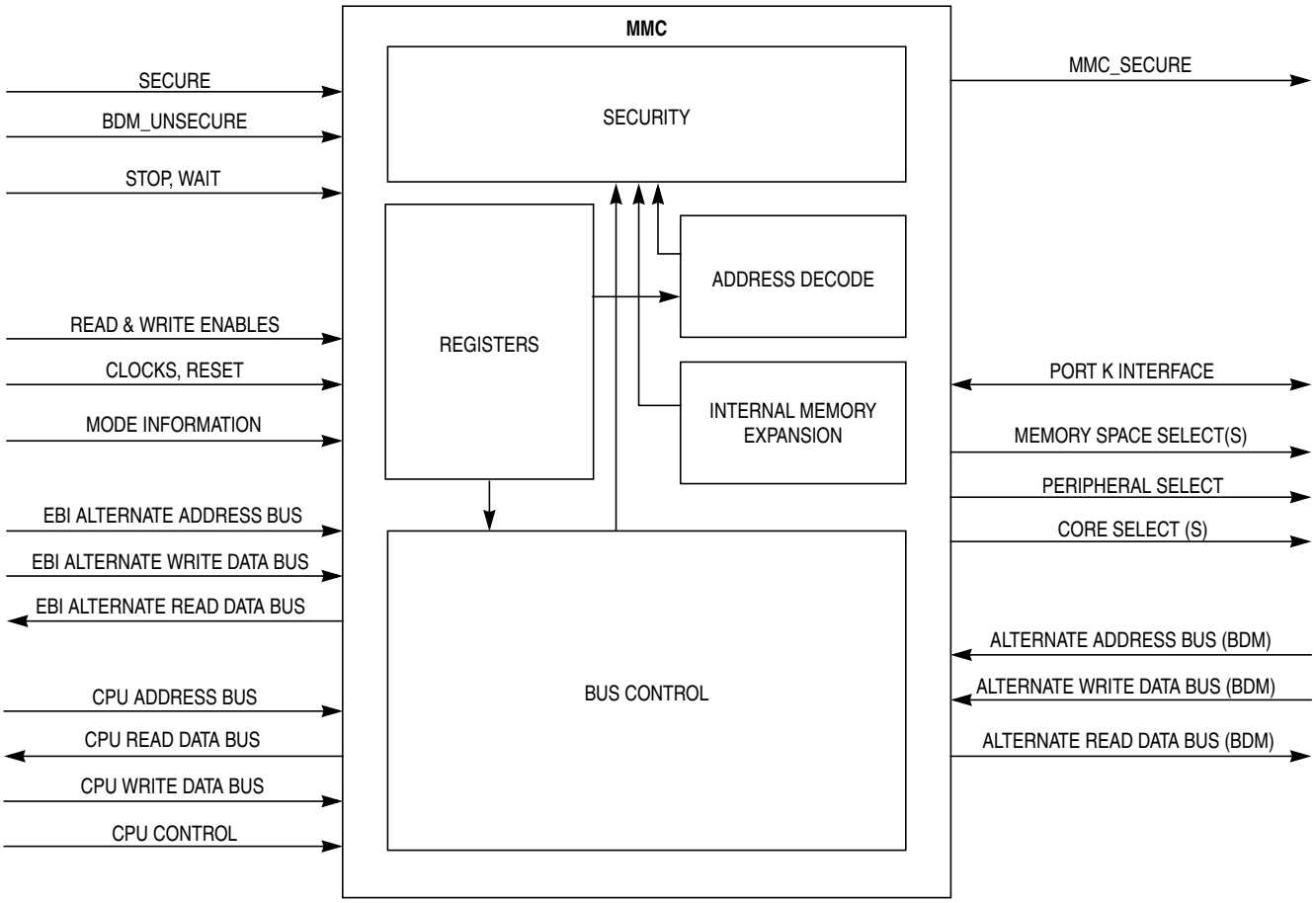


Figure 16-1. MMC Block Diagram

The MMC is the sub-module which controls memory map assignment and selection of internal resources and external space. Internal buses between the core and memories and between the core and peripherals is controlled in this module. The memory expansion is generated in this module.

16.1.1 Features

- Registers for mapping of address space for on-chip RAM, EEPROM, and FLASH (or ROM) memory blocks and associated registers

- Memory mapping control and selection based upon address decode and system operating mode
- Core address bus control
- Core data bus control and multiplexing
- Core security state decoding
- Emulation chip select signal generation ($\overline{ECS}$)
- External chip select signal generation ($\overline{XCS}$)
- Internal memory expansion
- External stretch and ROM mapping control functions via the MISC register
- Reserved registers for test purposes
- Configurable system memory options defined at integration of core into the system-on-a-chip (SoC).

16.1.2 Modes of Operation

Some of the registers operate differently depending on the mode of operation (i.e., normal expanded wide, special single chip, etc.). This is best understood from the register descriptions.

16.2 External Signal Description

All interfacing with the MMC sub-block is done within the core, it has no external signals.

16.3 Memory Map and Register Definition

A summary of the registers associated with the MMC sub-block is shown in Figure 16-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

16.3.1 Module Memory Map

Table 16-1. MMC Memory Map

Address Offset	Register	Access
	Initialization of Internal RAM Position Register (INITRM)	R/W
	Initialization of Internal Registers Position Register (INITRG)	R/W
	Initialization of Internal EEPROM Position Register (INITEE)	R/W
	Miscellaneous System Control Register (MISC)	R/W
	Reserved	—
. .	. .	—
	Reserved	—
. .	. .	—

Table 16-1. MMC Memory Map (continued)

Address Offset	Register	Access
	Memory Size Register 0 (MEMSIZ0)	R
	Memory Size Register 1 (MEMSIZ1)	R
. .	. .	
	Program Page Index Register (PPAGE)	R/W
	Reserved	—

16.3.2 Register Descriptions

Name		Bit 7	6	5	4	3	2	1	Bit 0
INITRM	R	RAM15	RAM14	RAM13	RAM12	RAM11	0	0	RAMHAL
	W								
INITRG	R	0	REG14	REG13	REG12	REG11	0	0	0
	W								
INITEE	R	EE15	EE14	EE13	EE12	EE11	0	0	EEON
	W								
MISC	R	0	0	0	0	EXSTR1	EXSTR0	ROMHM	ROMON
	W								
MTSTO	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
MTST1	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
MEMSIZ0	R	REG_SW0	0	EEP_SW1	EEP_SW0	0	RAM_SW2	RAM_SW1	RAM_SW0
	W								
MEMSIZ1	R	ROM_SW1	ROM_SW0	0	0	0	0	PAG_SW1	PAG_SW0
	W								
PPAGE	R	0	0	PIX5	PIX4	PIX3	PIX2	PIX1	PIX0
	W								
Reserved	R	0	0	0	0	0	0	0	0
	W								

= Unimplemented

Figure 16-2. MMC Register Summary

16.3.2.1 Initialization of Internal RAM Position Register (INITRM)

	7	6	5	4	3	2	1	0
R	RAM15	RAM14	RAM13	RAM12	RAM11	0	0	RAMHAL
W								
Reset	0	0	0	0	1	0	0	1

= Unimplemented or Reserved

Figure 16-3. Initialization of Internal RAM Position Register (INITRM)

Read: Anytime

Write: Once in normal and emulation modes, anytime in special modes

NOTE

Writes to this register take one cycle to go into effect.

This register initializes the position of the internal RAM within the on-chip system memory map.

Table 16-2. INITRM Field Descriptions

Field	Description
7:3 RAM[15:11]	**Internal RAM Map Position** — These bits determine the upper five bits of the base address for the system's internal RAM array.
0 RAMHAL	**RAM High-Align** — RAMHAL specifies the alignment of the internal RAM array. 0 Aligns the RAM to the lowest address (0x0000) of the mappable space 1 Aligns the RAM to the higher address (0xFFFF) of the mappable space

16.3.2.2 Initialization of Internal Registers Position Register (INITRG)

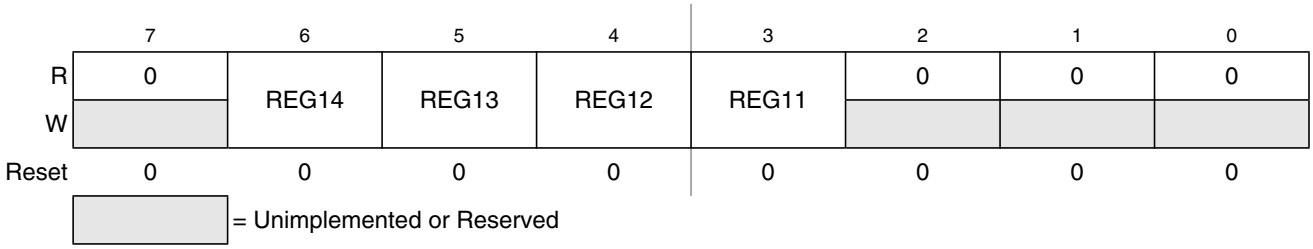


Figure 16-4. Initialization of Internal Registers Position Register (INITRG)

Read: Anytime

Write: Once in normal and emulation modes and anytime in special modes

This register initializes the position of the internal registers within the on-chip system memory map. The registers occupy either a 1K byte or 2K byte space and can be mapped to any 2K byte space within the first 32K bytes of the system's address space.

Table 16-3. INITRG Field Descriptions

Field	Description
6:3 REG[14:11]	**Internal Register Map Position** — These four bits in combination with the leading zero supplied by bit 7 of INITRG determine the upper five bits of the base address for the system's internal registers (i.e., the minimum base address is 0x0000 and the maximum is 0x7FFF).

16.3.2.3 Initialization of Internal EEPROM Position Register (INITEE)

	7	6	5	4	3	2	1	0
R	EE15	EE14	EE13	EE12	EE11	0	0	EEON
W								
Reset[1]	—	—	—	—	—	—	—	—

1. The reset state of this register is controlled at chip integration. Please refer to the device overview section to determine the actual reset state of this register.

= Unimplemented or Reserved

Figure 16-5. Initialization of Internal EEPROM Position Register (INITEE)

Read: Anytime

Write: The EEON bit can be written to any time on all devices. Bits E[11:15] are “write anytime in all modes” on most devices. On some devices, bits E[11:15] are “write once in normal and emulation modes and write anytime in special modes”. See device overview chapter to determine the actual write access rights.

NOTE

Writes to this register take one cycle to go into effect.

This register initializes the position of the internal EEPROM within the on-chip system memory map.

Table 16-4. INITEE Field Descriptions

Field	Description
7:3 EE[15:11]	**Internal EEPROM Map Position** — These bits determine the upper five bits of the base address for the system’s internal EEPROM array.
0 EEON	**Enable EEPROM** — This bit is used to enable the EEPROM memory in the memory map. 0 Disables the EEPROM from the memory map. 1 Enables the EEPROM in the memory map at the address selected by EE[15:11].

16.3.2.4 Miscellaneous System Control Register (MISC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	EXSTR1	EXSTR0	ROMHM	ROMON
W								
Reset: Expanded or Emulation	0	0	0	0	1	1	0	—[1]
Reset: Peripheral or Single Chip	0	0	0	0	1	1	0	1
Reset: Special Test	0	0	0	0	1	1	0	0

1. The reset state of this bit is determined at the chip integration level.

= Unimplemented or Reserved

Figure 16-6. Miscellaneous System Control Register (MISC)

Read: Anytime

Write: As stated in each bit description

NOTE

Writes to this register take one cycle to go into effect.

This register initializes miscellaneous control functions.

Table 16-5. INITEE Field Descriptions

Field	Description
3:2 EXSTR[1:0]	**External Access Stretch Bits 1 and 0** Write: once in normal and emulation modes and anytime in special modes This two-bit field determines the amount of clock stretch on accesses to the external address space as shown in Table 16-6. In single chip and peripheral modes these bits have no meaning or effect.
1 ROMHM	**FLASH EEPROM or ROM Only in Second Half of Memory Map** Write: once in normal and emulation modes and anytime in special modes 0 The fixed page(s) of FLASH EEPROM or ROM in the lower half of the memory map can be accessed. 1 Disables direct access to the FLASH EEPROM or ROM in the lower half of the memory map. These physical locations of the FLASH EEPROM or ROM remain accessible through the program page window.
0 ROMON	**ROMON — Enable FLASH EEPROM or ROM** Write: once in normal and emulation modes and anytime in special modes This bit is used to enable the FLASH EEPROM or ROM memory in the memory map. 0 Disables the FLASH EEPROM or ROM from the memory map. 1 Enables the FLASH EEPROM or ROM in the memory map.

Table 16-6. External Stretch Bit Definition

Stretch Bit EXSTR1	Stretch Bit EXSTR0	Number of E Clocks Stretched
0	0	0
0	1	1
1	0	2
1	1	3

16.3.2.5 Reserved Test Register 0 (MTST0)

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

= Unimplemented or Reserved

Figure 16-7. Reserved Test Register 0 (MTST0)

Read: Anytime

Write: No effect — this register location is used for internal test purposes.

16.3.2.6 Reserved Test Register 1 (MTST1)

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

= Unimplemented or Reserved

Figure 16-8. Reserved Test Register 1 (MTST1)

Read: Anytime

Write: No effect — this register location is used for internal test purposes.

16.3.2.7 Memory Size Register 0 (MEMSIZ0)

	7	6	5	4	3	2	1	0
R	REG_SW0	0	EEP_SW1	EEP_SW0	0	RAM_SW2	RAM_SW1	RAM_SW0
W								
Reset	—	—	—	—	—	—	—	—

= Unimplemented or Reserved

Figure 16-9. Memory Size Register 0 (MEMSIZ0)

Read: Anytime

Write: Writes have no effect

Reset: Defined at chip integration, see device overview section.

The MEMSIZ0 register reflects the state of the register, EEPROM and RAM memory space configuration switches at the core boundary which are configured at system integration. This register allows read visibility to the state of these switches.

Table 16-7. MEMSIZ0 Field Descriptions

Field	Description
7 REG_SW0	**Allocated System Register Space** 0 Allocated system register space size is 1K byte 1 Allocated system register space size is 2K byte
5:4 EEP_SW[1:0]	**Allocated System EEPROM Memory Space** — The allocated system EEPROM memory space size is as given in Table 16-8.
2 RAM_SW[2:0]	**Allocated System RAM Memory Space** — The allocated system RAM memory space size is as given in Table 16-9.

Table 16-8. Allocated EEPROM Memory Space

eep_sw1:eep_sw0	Allocated EEPROM Space
00	0K byte
01	2K bytes
10	4K bytes
11	8K bytes

Table 16-9. Allocated RAM Memory Space

ram_sw2:ram_sw0	Allocated RAM Space	RAM Mappable Region	INITRM Bits Used	RAM Reset Base Address[1]
000	2K bytes	2K bytes	RAM[15:11]	0x0800
001	4K bytes	4K bytes	RAM[15:12]	0x0000
010	6K bytes	8K bytes[2]	RAM[15:13]	0x0800
011	8K bytes	8K bytes	RAM[15:13]	0x0000
100	10K bytes	16K bytes [2]	RAM[15:14]	0x1800

Table 16-9. Allocated RAM Memory Space (continued)

ram_sw2:ram_sw0	Allocated RAM Space	RAM Mappable Region	INITRM Bits Used	RAM Reset Base Address[1]
101	12K bytes	16K bytes [2]	RAM[15:14]	0x1000
110	14K bytes	16K bytes [2]	RAM[15:14]	0x0800
111	16K bytes	16K bytes	RAM[15:14]	0x0000

1 The RAM Reset BASE Address is based on the reset value of the INITRM register, 0x0009.

2 Alignment of the Allocated RAM space within the RAM mappable region is dependent on the value of RAMHAL.

NOTE

As stated, the bits in this register provide read visibility to the system physical memory space allocations defined at system integration. The actual array size for any given type of memory block may differ from the allocated size. Please refer to the device overview chapter for actual sizes.

16.3.2.8 Memory Size Register 1 (MEMSIZ1)

	7	6	5	4	3	2	1	0
R	ROM_SW1	ROM_SW0	0	0	0	0	PAG_SW1	PAG_SW0
W								
Reset	—	—	—	—	—	—	—	—

= Unimplemented or Reserved

Figure 16-10. Memory Size Register 1 (MEMSIZ1)

Read: Anytime

Write: Writes have no effect

Reset: Defined at chip integration, see device overview section.

The MEMSIZ1 register reflects the state of the FLASH or ROM physical memory space and paging switches at the core boundary which are configured at system integration. This register allows read visibility to the state of these switches.

Table 16-10. MEMSIZ0 Field Descriptions

Field	Description
7:6 ROM_SW[1:0]	**Allocated System FLASH or ROM Physical Memory Space** — The allocated system FLASH or ROM physical memory space is as given in Table 16-11.
1:0 PAG_SW[1:0]	**Allocated Off-Chip FLASH or ROM Memory Space** — The allocated off-chip FLASH or ROM memory space size is as given in Table 16-12.

Table 16-11. Allocated FLASH/ROM Physical Memory Space

rom_sw1:rom_sw0	Allocated FLASH or ROM Space
00	0K byte
01	16K bytes
10	48K bytes(1)
11	64K bytes(1)

NOTES:

1. The ROMHM software bit in the MISC register determines the accessibility of the FLASH/ROM memory space. Please refer to Section 16.3.2.8, “Memory Size Register 1 (MEMSIZ1),” for a detailed functional description of the ROMHM bit.

Table 16-12. Allocated Off-Chip Memory Options

pag_sw1:pag_sw0	Off-Chip Space	On-Chip Space
00	876K bytes	128K bytes
01	768K bytes	256K bytes
10	512K bytes	512K bytes
11	0K byte	1M byte

NOTE

As stated, the bits in this register provide read visibility to the system memory space and on-chip/off-chip partitioning allocations defined at system integration. The actual array size for any given type of memory block may differ from the allocated size. Please refer to the device overview chapter for actual sizes.

16.3.2.9 Program Page Index Register (PPAGE)

	7	6	5	4	3	2	1	0
R	0	0	PIX5	PIX4	PIX3	PIX2	PIX1	PIX0
W								
Reset[1]	—	—	—	—	—	—	—	—

1. The reset state of this register is controlled at chip integration. Please refer to the device overview section to determine the actual reset state of this register.

= Unimplemented or Reserved

Figure 16-11. Program Page Index Register (PPAGE)

Read: Anytime

Write: Determined at chip integration. Generally it’s: “write anytime in all modes;” on some devices it will be: “write only in special modes.” Check specific device documentation to determine which applies.

Reset: Defined at chip integration as either 0x00 (paired with write in any mode) or 0x3C (paired with write only in special modes), see device overview chapter.

The HCS12 core architecture limits the physical address space available to 64K bytes. The program page index register allows for integrating up to 1M byte of FLASH or ROM into the system by using the six page index bits to page 16K byte blocks into the program page window located from 0x8000 to 0xBFFF as defined in Table 16-14. CALL and RTC instructions have special access to read and write this register without using the address bus.

NOTE

Normal writes to this register take one cycle to go into effect. Writes to this register using the special access of the CALL and RTC instructions will be complete before the end of the associated instruction.

Table 16-13. MEMSIZ0 Field Descriptions

Field	Description
5:0 PIX[5:0]	**Program Page Index Bits 5:0** — These page index bits are used to select which of the 64 FLASH or ROM array pages is to be accessed in the program page window as shown in Table 16-14.

Table 16-14. Program Page Index Register Bits

PIX5	PIX4	PIX3	PIX2	PIX1	PIX0	Program Space Selected
0	0	0	0	0	0	16K page 0
0	0	0	0	0	1	16K page 1
0	0	0	0	1	0	16K page 2
0	0	0	0	1	1	16K page 3
. . . . .	. . . . .	. . . .	. . . . .	. . . . .	. . . . .	. . . . .
1	1	1	1	0	0	16K page 60
1	1	1	1	0	1	16K page 61
1	1	1	1	1	0	16K page 62
1	1	1	1	1	1	16K page 63

16.4 Functional Description

The MMC sub-block performs four basic functions of the core operation: bus control, address decoding and select signal generation, memory expansion, and security decoding for the system. Each aspect is described in the following subsections.

16.4.1 Bus Control

The MMC controls the address bus and data buses that interface the core with the rest of the system. This includes the multiplexing of the input data buses to the core onto the main CPU read data bus and control

of data flow from the CPU to the output address and data buses of the core. In addition, the MMC manages all CPU read data bus swapping operations.

16.4.2 Address Decoding

As data flows on the core address bus, the MMC decodes the address information, determines whether the internal core register or firmware space, the peripheral space or a memory register or array space is being addressed and generates the correct select signal. This decoding operation also interprets the mode of operation of the system and the state of the mapping control registers in order to generate the proper select. The MMC also generates two external chip select signals, emulation chip select ($\overline{ECS}$) and external chip select ($\overline{XCS}$).

16.4.2.1 Select Priority and Mode Considerations

Although internal resources such as control registers and on-chip memory have default addresses, each can be relocated by changing the default values in control registers. Normally, I/O addresses, control registers, vector spaces, expansion windows, and on-chip memory are mapped so that their address ranges do not overlap. The MMC will make only one select signal active at any given time. This activation is based upon the priority outlined in Table 16-15. If two or more blocks share the same address space, only the select signal for the block with the highest priority will become active. An example of this is if the registers and the RAM are mapped to the same space, the registers will have priority over the RAM and the portion of RAM mapped in this shared space will not be accessible. The expansion windows have the lowest priority. This means that registers, vectors, and on-chip memory are always visible to a program regardless of the values in the page select registers.

Table 16-15. Select Signal Priority

Priority	Address Space
Highest	BDM (internal to core) firmware or register space
...	Internal register space
...	RAM memory block
...	EEPROM memory block
...	On-chip FLASH or ROM
Lowest	Remaining external space

In expanded modes, all address space not used by internal resources is by default external memory space. The data registers and data direction registers for ports A and B are removed from the on-chip memory map and become external accesses. If the EME bit in the MODE register (see MEBI block description chapter) is set, the data and data direction registers for port E are also removed from the on-chip memory map and become external accesses.

In special peripheral mode, the first 16 registers associated with bus expansion are removed from the on-chip memory map (PORTA, PORTB, DDRA, DDRB, PORTE, DDRE, PEAR, MODE, PUCR, RDRIV, and the EBI reserved registers).

In emulation modes, if the EMK bit in the MODE register (see MEBI block description chapter) is set, the data and data direction registers for port K are removed from the on-chip memory map and become external accesses.

16.4.2.2 Emulation Chip Select Signal

When the EMK bit in the MODE register (see MEBI block description chapter) is set, port K bit 7 is used as an active-low emulation chip select signal, $\overline{\text{ECS}}$. This signal is active when the system is in emulation mode, the EMK bit is set and the FLASH or ROM space is being addressed subject to the conditions outlined in Section 16.4.3.2, “Extended Address (XAB19:14) and ECS Signal Functionality.” When the EMK bit is clear, this pin is used for general purpose I/O.

16.4.2.3 External Chip Select Signal

When the EMK bit in the MODE register (see MEBI block description chapter) is set, port K bit 6 is used as an active-low external chip select signal, $\overline{\text{XCS}}$. This signal is active only when the $\overline{\text{ECS}}$ signal described above is not active and when the system is addressing the external address space. Accesses to unimplemented locations within the register space or to locations that are removed from the map (i.e., ports A and B in expanded modes) will not cause this signal to become active. When the EMK bit is clear, this pin is used for general purpose I/O.

16.4.3 Memory Expansion

The HCS12 core architecture limits the physical address space available to 64K bytes. The program page index register allows for integrating up to 1M byte of FLASH or ROM into the system by using the six page index bits to page 16K byte blocks into the program page window located from 0x8000 to 0xBFFF in the physical memory space. The paged memory space can consist of solely on-chip memory or a combination of on-chip and off-chip memory. This partitioning is configured at system integration through the use of the paging configuration switches (*pag_sw1:pag_sw0*) at the core boundary. The options available to the integrator are as given in Table 16-16 (this table matches Table 16-12 but is repeated here for easy reference).

Table 16-16. Allocated Off-Chip Memory Options

pag_sw1:pag_sw0	Off-Chip Space	On-Chip Space
00	876K bytes	128K bytes
01	768K bytes	256K bytes
10	512K bytes	512K bytes
11	0K byte	1M byte

Based upon the system configuration, the program page window will consider its access to be either internal or external as defined in Table 16-17.

Table 16-17. External/Internal Page Window Access

pag_sw1:pag_sw0	Partitioning	PIX5:0 Value	Page Window Access
00	876K off-Chip, 128K on-Chip	0x0000–0x0037	External
		0x0038–0x003F	Internal
01	768K off-chip, 256K on-chip	0x0000–0x002F	External
		0x0030–0x003F	Internal
10	512K off-chip, 512K on-chip	0x0000–0x001F	External
		0x0020–0x003F	Internal
11	0K off-chip, 1M on-chip	N/A	External
		0x0000–0x003F	Internal

NOTE

The partitioning as defined in Table 16-17 applies only to the allocated memory space and the actual on-chip memory sizes implemented in the system may differ. Please refer to the device overview chapter for actual sizes.

The PPAGE register holds the page select value for the program page window. The value of the PPAGE register can be manipulated by normal read and write (some devices don't allow writes in some modes) instructions as well as the CALL and RTC instructions.

Control registers, vector spaces, and a portion of on-chip memory are located in unpaged portions of the 64K byte physical address space. The stack and I/O addresses should also be in unpaged memory to make them accessible from any page.

The starting address of a service routine must be located in unpaged memory because the 16-bit exception vectors cannot point to addresses in paged memory. However, a service routine can call other routines that are in paged memory. The upper 16K byte block of memory space (0xC000–0xFFFF) is unpaged. It is recommended that all reset and interrupt vectors point to locations in this area.

16.4.3.1 CALL and Return from Call Instructions

CALL and RTC are uninterruptable instructions that automate page switching in the program expansion window. CALL is similar to a JSR instruction, but the subroutine that is called can be located anywhere in the normal 64K byte address space or on any page of program expansion memory. CALL calculates and stacks a return address, stacks the current PPAGE value, and writes a new instruction-supplied value to PPAGE. The PPAGE value controls which of the 64 possible pages is visible through the 16K byte expansion window in the 64K byte memory map. Execution then begins at the address of the called subroutine.

During the execution of a CALL instruction, the CPU:

- Writes the old PPAGE value into an internal temporary register and writes the new instruction-supplied PPAGE value into the PPAGE register.

- Calculates the address of the next instruction after the CALL instruction (the return address), and pushes this 16-bit value onto the stack.
- Pushes the old PPAGE value onto the stack.
- Calculates the effective address of the subroutine, refills the queue, and begins execution at the new address on the selected page of the expansion window.

This sequence is uninterruptable; there is no need to inhibit interrupts during CALL execution. A CALL can be performed from any address in memory to any other address.

The PPAGE value supplied by the instruction is part of the effective address. For all addressing mode variations except indexed-indirect modes, the new page value is provided by an immediate operand in the instruction. In indexed-indirect variations of CALL, a pointer specifies memory locations where the new page value and the address of the called subroutine are stored. Using indirect addressing for both the new page value and the address within the page allows values calculated at run time rather than immediate values that must be known at the time of assembly.

The RTC instruction terminates subroutines invoked by a CALL instruction. RTC unstacks the PPAGE value and the return address and refills the queue. Execution resumes with the next instruction after the CALL.

During the execution of an RTC instruction, the CPU:

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

This sequence is uninterruptable; an RTC can be executed from anywhere in memory, even from a different page of extended memory in the expansion window.

The CALL and RTC instructions behave like JSR and RTS, except they use more execution cycles. Therefore, routinely substituting CALL/RTC for JSR/RTS is not recommended. JSR and RTS can be used to access subroutines that are on the same page in expanded memory. However, a subroutine in expanded memory that can be called from other pages must be terminated with an RTC. And the RTC unstacks a PPAGE value. So any access to the subroutine, even from the same page, must use a CALL instruction so that the correct PPAGE value is in the stack.

16.4.3.2 Extended Address (XAB19:14) and $\overline{\text{ECS}}$ Signal Functionality

If the EMK bit in the MODE register is set (see MEBI block description chapter) the PIX5:0 values will be output on XAB19:14 respectively (port K bits 5:0) when the system is addressing within the physical program page window address space (0x8000–0xBFFF) and is in an expanded mode. When addressing anywhere else within the physical address space (outside of the paging space), the XAB19:14 signals will be assigned a constant value based upon the physical address space selected. In addition, the active-low emulation chip select signal, $\overline{\text{ECS}}$, will likewise function based upon the assigned memory allocation. In the cases of 48K byte and 64K byte allocated physical FLASH/ROM space, the operation of the $\overline{\text{ECS}}$ signal will additionally depend upon the state of the ROMHM bit (see Section 16.3.2.4, “Miscellaneous System Control Register (MISC)”) in the MISC register. Table 16-18, Table 16-19, Table 16-20, and

Table 16-21 summarize the functionality of these signals based upon the allocated memory configuration. Again, this signal information is only available externally when the EMK bit is set and the system is in an expanded mode.

Table 16-18. 0K Byte Physical FLASH/ROM Allocated

Address Space	Page Window Access	ROMHM	ECS	XAB19:14
0x0000–0x3FFF	N/A	N/A	1	0x3D
0x4000–0x7FFF	N/A	N/A	1	0x3E
0x8000–0xBFFF	N/A	N/A	0	PIX[5:0]
0xC000–0xFFFF	N/A	N/A	0	0x3F

Table 16-19. 16K Byte Physical FLASH/ROM Allocated

Address Space	Page Window Access	ROMHM	ECS	XAB19:14
0x0000–0x3FFF	N/A	N/A	1	0x3D
0x4000–0x7FFF	N/A	N/A	1	0x3E
0x8000–0xBFFF	N/A	N/A	1	PIX[5:0]
0xC000–0xFFFF	N/A	N/A	0	0x3F

Table 16-20. 48K Byte Physical FLASH/ROM Allocated

Address Space	Page Window Access	ROMHM	ECS	XAB19:14
0x0000–0x3FFF	N/A	N/A	1	0x3D
0x4000–0x7FFF	N/A	0	0	0x3E
	N/A	1	1	
0x8000–0xBFFF	External	N/A	1	PIX[5:0]
	Internal	N/A	0	
0xC000–0xFFFF	N/A	N/A	0	0x3F

Table 16-21. 64K Byte Physical FLASH/ROM Allocated

Address Space	Page Window Access	ROMHM	ECS	XAB19:14
0x0000–0x3FFF	N/A	0	0	0x3D
	N/A	1	1	
0x4000–0x7FFF	N/A	0	0	0x3E
	N/A	1	1	
0x8000–0xBFFF	External	N/A	1	PIX[5:0]
	Internal	N/A	0	
0xC000–0xFFFF	N/A	N/A	0	0x3F

A graphical example of a memory paging for a system configured as 1M byte on-chip FLASH/ROM with 64K allocated physical space is given in Figure 16-12.

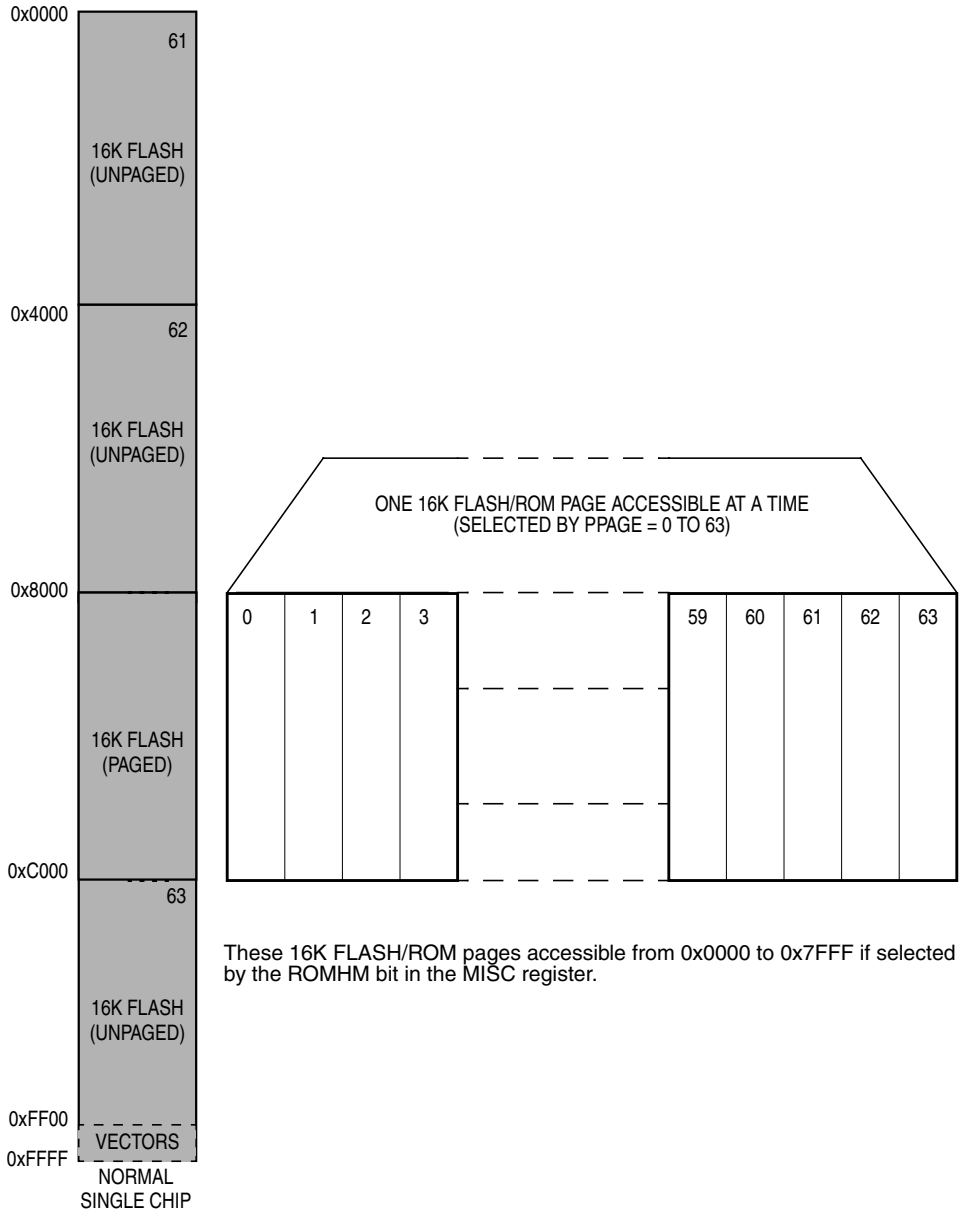


Figure 16-12. Memory Paging Example: 1M Byte On-Chip FLASH/ROM, 64K Allocation

Chapter 17
Background Debug Module (BDMV4)

17.1 Introduction

This section describes the functionality of the background debug module (BDM) sub-block of the HCS12 core platform.

A block diagram of the BDM is shown in Figure 17-1.

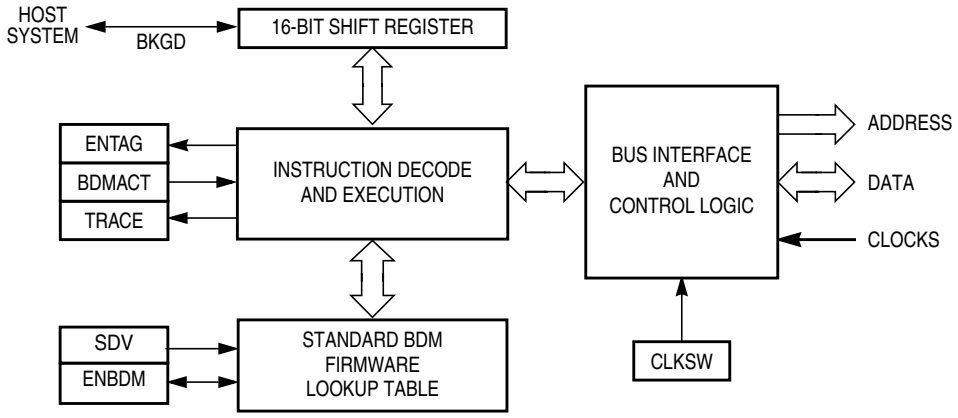


Figure 17-1. BDM Block Diagram

The background debug module (BDM) sub-block is a single-wire, background debug system implemented in on-chip hardware for minimal CPU intervention. All interfacing with the BDM is done via the BKGD pin.

BDMV4 has enhanced capability for maintaining synchronization between the target and host while allowing more flexibility in clock rates. This includes a sync signal to show the clock rate and a handshake signal to indicate when an operation is complete. The system is backwards compatible with older external interfaces.

17.1.1 Features

- Single-wire communication with host development system
- BDMV4 (and BDM2): Enhanced capability for allowing more flexibility in clock rates
- BDMV4: SYNC command to determine communication rate
- BDMV4: GO_UNTIL command
- BDMV4: Hardware handshake protocol to increase the performance of the serial communication
- Active out of reset in special single-chip mode
- Nine hardware commands using free cycles, if available, for minimal CPU intervention
- Hardware commands not requiring active BDM
- 15 firmware commands execute from the standard BDM firmware lookup table

- Instruction tagging capability
- Software control of BDM operation during wait mode
- Software selectable clocks
- When secured, hardware commands are allowed to access the register space in special single-chip mode, if the FLASH and EEPROM erase tests fail.

17.1.2 Modes of Operation

BDM is available in all operating modes but must be enabled before firmware commands are executed. Some system peripherals may have a control bit which allows suspending the peripheral function during background debug mode.

17.1.2.1 Regular Run Modes

All of these operations refer to the part in run mode. The BDM does not provide controls to conserve power during run mode.

- Normal operation

  General operation of the BDM is available and operates the same in all normal modes.
- Special single-chip mode

  In special single-chip mode, background operation is enabled and active out of reset. This allows programming a system with blank memory.
- Special peripheral mode

  BDM is enabled and active immediately out of reset. BDM can be disabled by clearing the BDMACT bit in the BDM status (BDMSTS) register. The BDM serial system should not be used in special peripheral mode.
- Emulation modes

  General operation of the BDM is available and operates the same as in normal modes.

17.1.2.2 Secure Mode Operation

If the part is in secure mode, the operation of the BDM is reduced to a small subset of its regular run mode operation. Secure operation prevents access to FLASH or EEPROM other than allowing erasure.

17.2 External Signal Description

A single-wire interface pin is used to communicate with the BDM system. Two additional pins are used for instruction tagging. These pins are part of the multiplexed external bus interface (MEBI) sub-block and all interfacing between the MEBI and BDM is done within the core interface boundary. Functional descriptions of the pins are provided below for completeness.

- BKGD — Background interface pin
- $\overline{\text{TAGHI}}$ — High byte instruction tagging pin
- $\overline{\text{TAGLO}}$ — Low byte instruction tagging pin

- BKGD and $\overline{\text{TAGHI}}$ share the same pin.
- $\overline{\text{TAGLO}}$ and $\overline{\text{LSTRB}}$ share the same pin.

NOTE

Generally these pins are shared as described, but it is best to check the device overview chapter to make certain. All MCUs at the time of this writing have followed this pin sharing scheme.

17.2.1 BKGD — Background Interface Pin

Debugging control logic communicates with external devices serially via the single-wire background interface pin (BKGD). During reset, this pin is a mode select input which selects between normal and special modes of operation. After reset, this pin becomes the dedicated serial interface pin for the background debug mode.

17.2.2 $\overline{\text{TAGHI}}$ — High Byte Instruction Tagging Pin

This pin is used to tag the high byte of an instruction. When instruction tagging is on, a logic 0 at the falling edge of the external clock (ECLK) tags the high half of the instruction word being read into the instruction queue.

17.2.3 $\overline{\text{TAGLO}}$ — Low Byte Instruction Tagging Pin

This pin is used to tag the low byte of an instruction. When instruction tagging is on and low strobe is enabled, a logic 0 at the falling edge of the external clock (ECLK) tags the low half of the instruction word being read into the instruction queue.

17.3 Memory Map and Register Definition

A summary of the registers associated with the BDM is shown in Figure 17-2. Registers are accessed by host-driven communications to the BDM hardware using READ_BD and WRITE_BD commands. Detailed descriptions of the registers and associated bits are given in the subsections that follow.

17.3.1 Module Memory Map

Table 17-1. INT Memory Map

Register Address	Use	Access
	Reserved	—
	BDM Status Register (BDMSTS)	R/W
	Reserved	—
	BDM CCR Holding Register (BDMCCR)	R/W
7	BDM Internal Register Position (BDMINR)	R
8–	Reserved	—

17.3.2 Register Descriptions

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
Reserved	R	X	X	X	X	X	X	0	0
	W								
BDMSTS	R	ENBDM	BDMACT	ENTAG	SDV	TRACE	CLKSW	UNSEC	0
	W								
Reserved	R	X	X	X	X	X	X	X	X
	W								
Reserved	R	X	X	X	X	X	X	X	X
	W								
Reserved	R	X	X	X	X	X	X	X	X
	W								
Reserved	R	X	X	X	X	X	X	X	X
	W								
BDMCCR	R	CCR7	CCR6	CCR5	CCR4	CCR3	CCR2	CCR1	CCR0
	W								
BDMINR	R	0	REG14	REG13	REG12	REG11	0	0	0
	W								
Reserved	R	0	0	0	0	0	0	0	0
	W								
Reserved	R	0	0	0	0	0	0	0	0
	W								
Reserved	R	X	X	X	X	X	X	X	X
	W								
Reserved	R	X	X	X	X	X	X	X	X
	W								

	= Unimplemented, Reserved		= Implemented (do not alter)
X	= Indeterminate	0	= Always read zero

Figure 17-2. BDM Register Summary

17.3.2.1 BDM Status Register (BDMSTS)

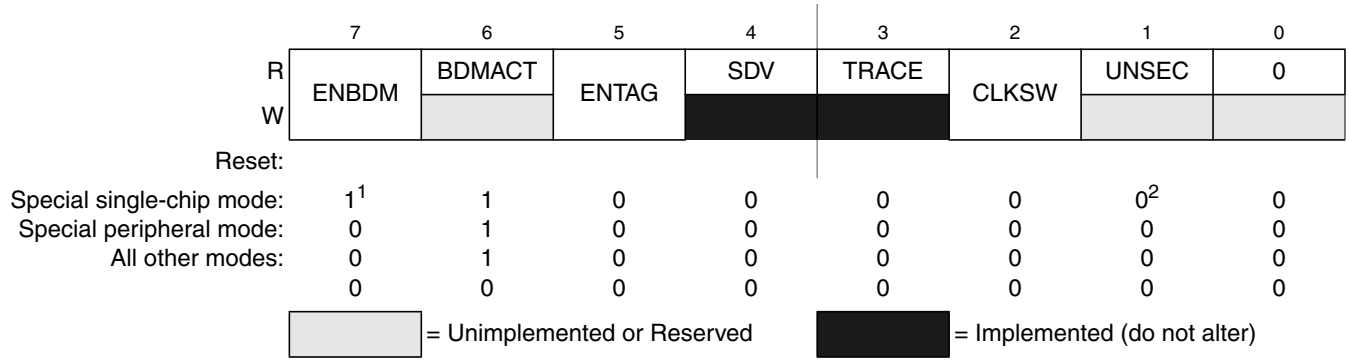


Figure 17-3. BDM Status Register (BDMSTS)

Note:

1 ENBDM is read as "1" by a debugging environment in Special single-chip mode when the device is not secured or secured but fully erased (Flash and EEPROM).This is because the ENBDM bit is set by the standard firmware before a BDM command can be fully transmitted and executed.

2 UNSEC is read as "1" by a debugging environment in Special single-chip mode when the device is secured and fully erased, else it is "0" and can only be read if not secure (see also bit description).

Read: All modes through BDM operation

Write: All modes but subject to the following:

- BDMACT can only be set by BDM hardware upon entry into BDM. It can only be cleared by the standard BDM firmware lookup table upon exit from BDM active mode.
- CLKSW can only be written via BDM hardware or standard BDM firmware write commands.
- All other bits, while writable via BDM hardware or standard BDM firmware write commands, should only be altered by the BDM hardware or standard firmware lookup table as part of BDM command execution.
- ENBDM should only be set via a BDM hardware command if the BDM firmware commands are needed. (This does not apply in special single-chip mode).

Table 17-2. BDMSTS Field Descriptions

Field	Description
7 ENBDM	**Enable BDM** — This bit controls whether the BDM is enabled or disabled. When enabled, BDM can be made active to allow firmware commands to be executed. When disabled, BDM cannot be made active but BDM hardware commands are allowed. 0 BDM disabled 1 BDM enabled **Note:** ENBDM is set by the firmware immediately out of reset in special single-chip mode. In secure mode, this bit will not be set by the firmware until after the EEPROM and FLASH erase verify tests are complete.
6 BDMACT	**BDM Active Status** — This bit becomes set upon entering BDM. The standard BDM firmware lookup table is then enabled and put into the memory map. BDMACT is cleared by a carefully timed store instruction in the standard BDM firmware as part of the exit sequence to return to user code and remove the BDM memory from the map. 0 BDM not active 1 BDM active

Table 17-2. BDMSTS Field Descriptions (continued)

Field	Description
5 ENTAG	**Tagging Enable** — This bit indicates whether instruction tagging in enabled or disabled. It is set when the TAGGO command is executed and cleared when BDM is entered. The serial system is disabled and the tag function enabled 16 cycles after this bit is written. BDM cannot process serial commands while tagging is active. 0 Tagging not enabled or BDM active 1 Tagging enabled
4 SDV	**Shift Data Valid** — This bit is set and cleared by the BDM hardware. It is set after data has been transmitted as part of a firmware read command or after data has been received as part of a firmware write command. It is cleared when the next BDM command has been received or BDM is exited. SDV is used by the standard BDM firmware to control program flow execution. 0 Data phase of command not complete 1 Data phase of command is complete
3 TRACE	**TRACE1 BDM Firmware Command is Being Executed** — This bit gets set when a BDM TRACE1 firmware command is first recognized. It will stay set as long as continuous back-to-back TRACE1 commands are executed. This bit will get cleared when the next command that is not a TRACE1 command is recognized. 0 TRACE1 command is not being executed 1 TRACE1 command is being executed
2 CLKSW	**Clock Switch** — The CLKSW bit controls which clock the BDM operates with. It is only writable from a hardware BDM command. A 150 cycle delay at the clock speed that is active during the data portion of the command will occur before the new clock source is guaranteed to be active. The start of the next BDM command uses the new clock for timing subsequent BDM communications. Table 17-3 shows the resulting BDM clock source based on the CLKSW and the PLLSEL (PII select from the clock and reset generator) bits. **Note:** The BDM alternate clock source can only be selected when CLKSW = 0 and PLLSEL = 1. The BDM serial interface is now fully synchronized to the alternate clock source, when enabled. This eliminates frequency restriction on the alternate clock which was required on previous versions. Refer to the device overview section to determine which clock connects to the alternate clock source input. **Note:** If the acknowledge function is turned on, changing the CLKSW bit will cause the ACK to be at the new rate for the write command which changes it.
1 UNSEC	**Unsecure** — This bit is only writable in special single-chip mode from the BDM secure firmware and always gets reset to zero. It is in a zero state as secure mode is entered so that the secure BDM firmware lookup table is enabled and put into the memory map along with the standard BDM firmware lookup table. The secure BDM firmware lookup table verifies that the on-chip EEPROM and FLASH EEPROM are erased. This being the case, the UNSEC bit is set and the BDM program jumps to the start of the standard BDM firmware lookup table and the secure BDM firmware lookup table is turned off. If the erase test fails, the UNSEC bit will not be asserted. 0 System is in a secured mode 1 System is in a unsecured mode **Note:** When UNSEC is set, security is off and the user can change the state of the secure bits in the on-chip FLASH EEPROM. Note that if the user does not change the state of the bits to “unsecured” mode, the system will be secured again when it is next taken out of reset.

Table 17-3. BDM Clock Sources

PLLSEL	CLKSW	BDMCLK
0	0	Bus clock
0	1	Bus clock

Table 17-3. BDM Clock Sources

PLLSEL	CLKSW	BDMCLK
1	0	Alternate clock (refer to the device overview chapter to determine the alternate clock source)
1	1	Bus clock dependent on the PLL

17.3.2.2 BDM CCR Holding Register (BDMCCR)

	7	6	5	4	3	2	1	0
R W	CCR7	CCR6	CCR5	CCR4	CCR3	CCR2	CCR1	CCR0
Reset	0	0	0	0	0	0	0	0

Figure 17-4. BDM CCR Holding Register (BDMCCR)

Read: All modes

Write: All modes

NOTE

When BDM is made active, the CPU stores the value of the CCR register in the BDMCCR register. However, out of special single-chip reset, the BDMCCR is set to 0xD8 and not 0xD0 which is the reset value of the CCR register.

When entering background debug mode, the BDM CCR holding register is used to save the contents of the condition code register of the user's program. It is also used for temporary storage in the standard BDM firmware mode. The BDM CCR holding register can be written to modify the CCR value.

17.3.2.3 BDM Internal Register Position Register (BDMINR)

	7	6	5	4	3	2	1	0
R	0	REG14	REG13	REG12	REG11	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 17-5. BDM Internal Register Position (BDMINR)

Read: All modes

Write: Never

Table 17-4. BDMINR Field Descriptions

Field	Description
6:3 REG[14:11]	**Internal Register Map Position** — These four bits show the state of the upper five bits of the base address for the system's relocatable register block. BDMINR is a shadow of the INITRG register which maps the register block to any 2K byte space within the first 32K bytes of the 64K byte address space.

17.4 Functional Description

The BDM receives and executes commands from a host via a single wire serial interface. There are two types of BDM commands, namely, hardware commands and firmware commands.

Hardware commands are used to read and write target system memory locations and to enter active background debug mode, see Section 17.4.3, “BDM Hardware Commands.” Target system memory includes all memory that is accessible by the CPU.

Firmware commands are used to read and write CPU resources and to exit from active background debug mode, see Section 17.4.4, “Standard BDM Firmware Commands.” The CPU resources referred to are the accumulator (D), X index register (X), Y index register (Y), stack pointer (SP), and program counter (PC).

Hardware commands can be executed at any time and in any mode excluding a few exceptions as highlighted, see Section 17.4.3, “BDM Hardware Commands.” Firmware commands can only be executed when the system is in active background debug mode (BDM).

17.4.1 Security

If the user resets into special single-chip mode with the system secured, a secured mode BDM firmware lookup table is brought into the map overlapping a portion of the standard BDM firmware lookup table. The secure BDM firmware verifies that the on-chip EEPROM and FLASH EEPROM are erased. This being the case, the UNSEC bit will get set. The BDM program jumps to the start of the standard BDM firmware and the secured mode BDM firmware is turned off and all BDM commands are allowed. If the EEPROM or FLASH do not verify as erased, the BDM firmware sets the ENBDM bit, without asserting UNSEC, and the firmware enters a loop. This causes the BDM hardware commands to become enabled, but does not enable the firmware commands. This allows the BDM hardware to be used to erase the EEPROM and FLASH. After execution of the secure firmware, regardless of the results of the erase tests, the CPU registers, INITEE and PPAGE, will no longer be in their reset state.

17.4.2 Enabling and Activating BDM

The system must be in active BDM to execute standard BDM firmware commands. BDM can be activated only after being enabled. BDM is enabled by setting the ENBDM bit in the BDM status (BDMSTS) register. The ENBDM bit is set by writing to the BDM status (BDMSTS) register, via the single-wire interface, using a hardware command such as WRITE_BD_BYTE.

After being enabled, BDM is activated by one of the following[1]:

- Hardware BACKGROUND command
- BDM external instruction tagging mechanism
- CPU BGND instruction
- Breakpoint sub-block’s force or tag mechanism[2]

When BDM is activated, the CPU finishes executing the current instruction and then begins executing the firmware in the standard BDM firmware lookup table. When BDM is activated by the breakpoint

1. BDM is enabled and active immediately out of special single-chip reset.
2. This method is only available on systems that have a a breakpoint or a debug sub-block.

sub-block, the type of breakpoint used determines if BDM becomes active before or after execution of the next instruction.

NOTE

If an attempt is made to activate BDM before being enabled, the CPU resumes normal instruction execution after a brief delay. If BDM is not enabled, any hardware BACKGROUND commands issued are ignored by the BDM and the CPU is not delayed.

In active BDM, the BDM registers and standard BDM firmware lookup table are mapped to addresses 0xFF00 to 0xFFFF. BDM registers are mapped to addresses 0xFF00 to 0xFF07. The BDM uses these registers which are readable anytime by the BDM. However, these registers are not readable by user programs.

17.4.3 BDM Hardware Commands

Hardware commands are used to read and write target system memory locations and to enter active background debug mode. Target system memory includes all memory that is accessible by the CPU such as on-chip RAM, EEPROM, FLASH EEPROM, I/O and control registers, and all external memory.

Hardware commands are executed with minimal or no CPU intervention and do not require the system to be in active BDM for execution, although they can continue to be executed in this mode. When executing a hardware command, the BDM sub-block waits for a free CPU bus cycle so that the background access does not disturb the running application program. If a free cycle is not found within 128 clock cycles, the CPU is momentarily frozen so that the BDM can steal a cycle. When the BDM finds a free cycle, the operation does not intrude on normal CPU operation provided that it can be completed in a single cycle. However, if an operation requires multiple cycles the CPU is frozen until the operation is complete, even though the BDM found a free cycle.

The BDM hardware commands are listed in Table 17-5.

Table 17-5. Hardware Commands

Command	Opcode (hex)	Data	Description
BACKGROUND	90	None	Enter background mode if firmware is enabled. If enabled, an ACK will be issued when the part enters active background mode.
ACK_ENABLE	D5	None	Enable handshake. Issues an ACK pulse after the command is executed.
ACK_DISABLE	D6	None	Disable handshake. This command does not issue an ACK pulse.
READ_BD_BYTE	E4	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table in map. Odd address data on low byte; even address data on high byte.
READ_BD_WORD	EC	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table in map. Must be aligned access.
READ_BYTE	E0	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table out of map. Odd address data on low byte; even address data on high byte.
READ_WORD	E8	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table out of map. Must be aligned access.
WRITE_BD_BYTE	C4	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table in map. Odd address data on low byte; even address data on high byte.
WRITE_BD_WORD	CC	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table in map. Must be aligned access.
WRITE_BYTE	C0	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table out of map. Odd address data on low byte; even address data on high byte.
WRITE_WORD	C8	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table out of map. Must be aligned access.

NOTE:
If enabled, ACK will occur when data is ready for transmission for all BDM READ commands and will occur after the write is complete for all BDM WRITE commands.

The READ_BD and WRITE_BD commands allow access to the BDM register locations. These locations are not normally in the system memory map but share addresses with the application in memory. To distinguish between physical memory locations that share the same address, BDM memory resources are enabled just for the READ_BD and WRITE_BD access cycle. This allows the BDM to access BDM locations unobtrusively, even if the addresses conflict with the application memory map.

17.4.4 Standard BDM Firmware Commands

Firmware commands are used to access and manipulate CPU resources. The system must be in active BDM to execute standard BDM firmware commands, see Section 17.4.2, “Enabling and Activating BDM.” Normal instruction execution is suspended while the CPU executes the firmware located in the standard BDM firmware lookup table. The hardware command BACKGROUND is the usual way to activate BDM.

As the system enters active BDM, the standard BDM firmware lookup table and BDM registers become visible in the on-chip memory map at 0xFF00–0xFFFF, and the CPU begins executing the standard BDM

firmware. The standard BDM firmware watches for serial commands and executes them as they are received.

The firmware commands are shown in Table 17-6.

Table 17-6. Firmware Commands

Command[1]	Opcode (hex)	Data	Description
READ_NEXT	62	16-bit data out	Increment X by 2 (X = X + 2), then read word X points to.
READ_PC	63	16-bit data out	Read program counter.
READ_D	64	16-bit data out	Read D accumulator.
READ_X	65	16-bit data out	Read X index register.
READ_Y	66	16-bit data out	Read Y index register.
READ_SP	67	16-bit data out	Read stack pointer.
WRITE_NEXT	42	16-bit data in	Increment X by 2 (X = X + 2), then write word to location pointed to by X.
WRITE_PC	43	16-bit data in	Write program counter.
WRITE_D	44	16-bit data in	Write D accumulator.
WRITE_X	45	16-bit data in	Write X index register.
WRITE_Y	46	16-bit data in	Write Y index register.
WRITE_SP	47	16-bit data in	Write stack pointer.
GO	08	None	Go to user program. If enabled, ACK will occur when leaving active background mode.
GO_UNTIL[2]	0C	None	Go to user program. If enabled, ACK will occur upon returning to active background mode.
TRACE1	10	None	Execute one user instruction then return to active BDM. If enabled, ACK will occur upon returning to active background mode.
TAGGO	18	None	Enable tagging and go to user program. There is no ACK pulse related to this command.

1 If enabled, ACK will occur when data is ready for transmission for all BDM READ commands and will occur after the write is complete for all BDM WRITE commands.

2 Both WAIT (with clocks to the S12 CPU core disabled) and STOP disable the ACK function. The GO_UNTIL command will not get an Acknowledge if one of these two CPU instructions occurs before the “UNTIL” instruction. This can be a problem for any instruction that uses ACK, but GO_UNTIL is a lot more difficult for the development tool to time-out.

17.4.5 BDM Command Structure

Hardware and firmware BDM commands start with an 8-bit opcode followed by a 16-bit address and/or a 16-bit data word depending on the command. All the read commands return 16 bits of data despite the byte or word implication in the command name.

NOTE

8-bit reads return 16-bits of data, of which, only one byte will contain valid data. If reading an even address, the valid data will appear in the MSB. If reading an odd address, the valid data will appear in the LSB.

NOTE

16-bit misaligned reads and writes are not allowed. If attempted, the BDM will ignore the least significant bit of the address and will assume an even address from the remaining bits.

For hardware data read commands, the external host must wait 150 bus clock cycles after sending the address before attempting to obtain the read data. This is to be certain that valid data is available in the BDM shift register, ready to be shifted out. For hardware write commands, the external host must wait 150 bus clock cycles after sending the data to be written before attempting to send a new command. This is to avoid disturbing the BDM shift register before the write has been completed. The 150 bus clock cycle delay in both cases includes the maximum 128 cycle delay that can be incurred as the BDM waits for a free cycle before stealing a cycle.

For firmware read commands, the external host should wait 44 bus clock cycles after sending the command opcode and before attempting to obtain the read data. This includes the potential of an extra 7 cycles when the access is external with a narrow bus access (+1 cycle) and / or a stretch (+1, 2, or 3 cycles), (7 cycles could be needed if both occur). The 44 cycle wait allows enough time for the requested data to be made available in the BDM shift register, ready to be shifted out.

NOTE

This timing has increased from previous BDM modules due to the new capability in which the BDM serial interface can potentially run faster than the bus. On previous BDM modules this extra time could be hidden within the serial time.

For firmware write commands, the external host must wait 32 bus clock cycles after sending the data to be written before attempting to send a new command. This is to avoid disturbing the BDM shift register before the write has been completed.

The external host should wait 64 bus clock cycles after a TRACE1 or GO command before starting any new serial command. This is to allow the CPU to exit gracefully from the standard BDM firmware lookup table and resume execution of the user code. Disturbing the BDM shift register prematurely may adversely affect the exit from the standard BDM firmware lookup table.

NOTE

If the bus rate of the target processor is unknown or could be changing, it is recommended that the ACK (acknowledge function) be used to indicate when an operation is complete. When using ACK, the delay times are automated.

Figure 17-6 represents the BDM command structure. The command blocks illustrate a series of eight bit times starting with a falling edge. The bar across the top of the blocks indicates that the BKGD line idles in the high state. The time for an 8-bit command is 8×16 target clock cycles.[1]

1. Target clock cycles are cycles measured using the target MCU's serial clock rate. See Section 17.4.6, "BDM Serial Interface," and Section 17.3.2.1, "BDM Status Register (BDMSTS)," for information on how serial clock rate is selected.

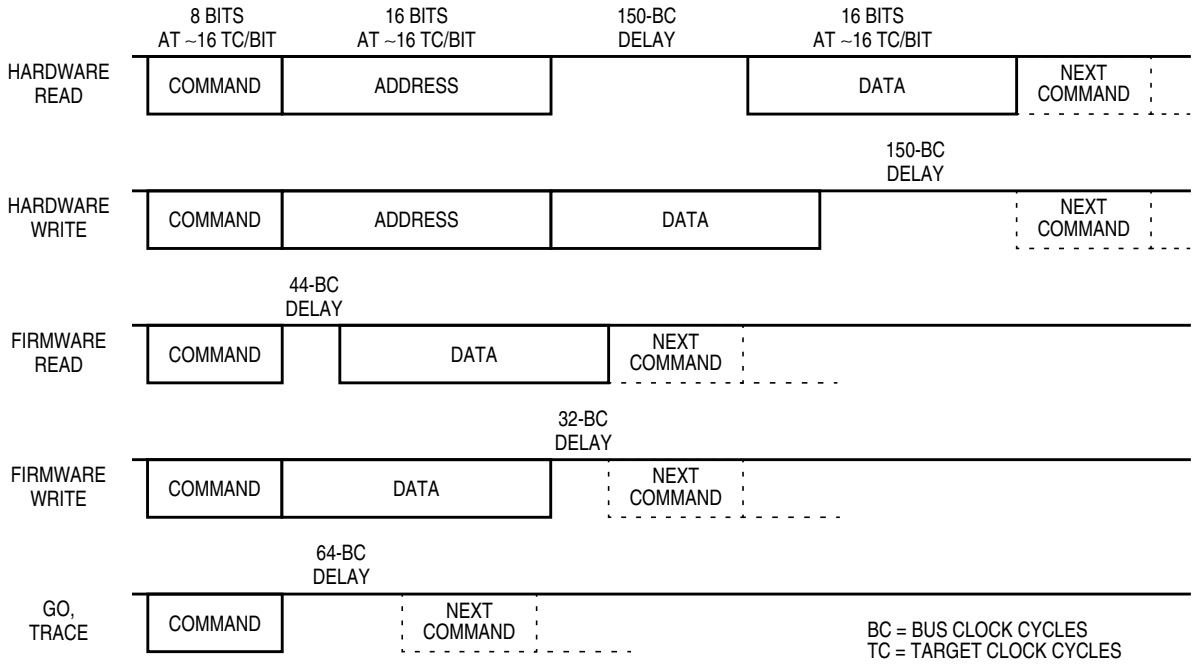


Figure 17-6. BDM Command Structure

17.4.6 BDM Serial Interface

The BDM communicates with external devices serially via the BKGD pin. During reset, this pin is a mode select input which selects between normal and special modes of operation. After reset, this pin becomes the dedicated serial interface pin for the BDM.

The BDM serial interface is timed using the clock selected by the CLKSW bit in the status register see Section 17.3.2.1, “BDM Status Register (BDMSTS).” This clock will be referred to as the target clock in the following explanation.

The BDM serial interface uses a clocking scheme in which the external host generates a falling edge on the BKGD pin to indicate the start of each bit time. This falling edge is sent for every bit whether data is transmitted or received. Data is transferred most significant bit (MSB) first at 16 target clock cycles per bit. The interface times out if 512 clock cycles occur between falling edges from the host.

The BKGD pin is a pseudo open-drain pin and has an weak on-chip active pull-up that is enabled at all times. It is assumed that there is an external pull-up and that drivers connected to BKGD do not typically drive the high level. Because R-C rise time could be unacceptably long, the target system and host provide brief driven-high (speedup) pulses to drive BKGD to a logic 1. The source of this speedup pulse is the host for transmit cases and the target for receive cases.

The timing for host-to-target is shown in Figure 17-7 and that of target-to-host in Figure 17-8 and Figure 17-9. All four cases begin when the host drives the BKGD pin low to generate a falling edge. Because the host and target are operating from separate clocks, it can take the target system up to one full clock cycle to recognize this edge. The target measures delays from this perceived start of the bit time while the host measures delays from the point it actually drove BKGD low to start the bit up to one target

clock cycle earlier. Synchronization between the host and target is established in this manner at the start of every bit time.

Figure 17-7 shows an external host transmitting a logic 1 and transmitting a logic 0 to the BKGD pin of a target system. The host is asynchronous to the target, so there is up to a one clock-cycle delay from the host-generated falling edge to where the target recognizes this edge as the beginning of the bit time. Ten target clock cycles later, the target senses the bit level on the BKGD pin. Internal glitch detect logic requires the pin be driven high no later that eight target clock cycles after the falling edge for a logic 1 transmission.

Because the host drives the high speedup pulses in these two cases, the rising edges look like digitally driven signals.

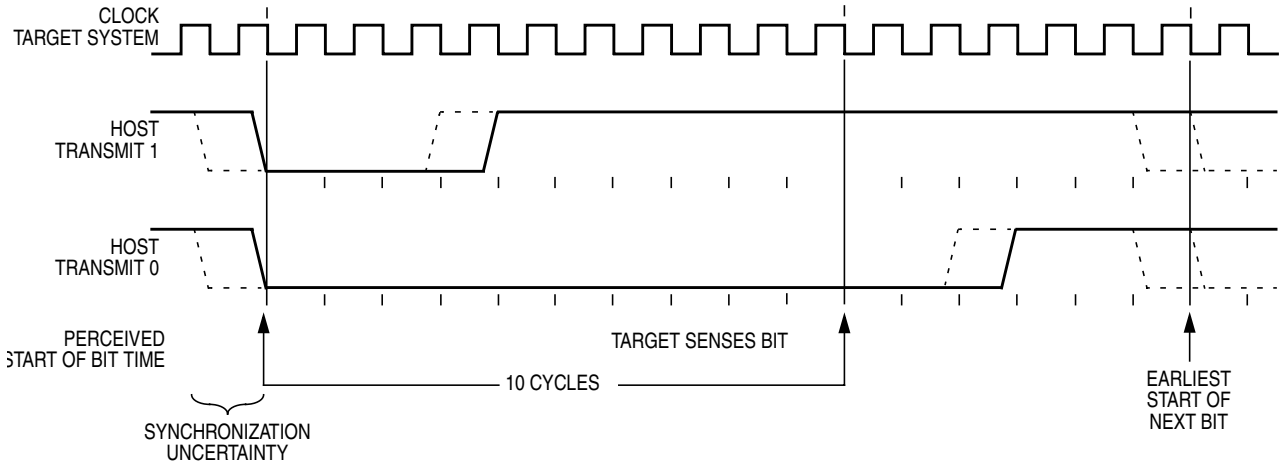


Figure 17-7. BDM Host-to-Target Serial Bit Timing

The receive cases are more complicated. Figure 17-8 shows the host receiving a logic 1 from the target system. Because the host is asynchronous to the target, there is up to one clock-cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target. The host holds the BKGD pin low long enough for the target to recognize it (at least two target clock cycles). The host must release the low drive before the target drives a brief high speedup pulse seven target clock cycles after the perceived start of the bit time. The host should sample the bit level about 10 target clock cycles after it started the bit time.

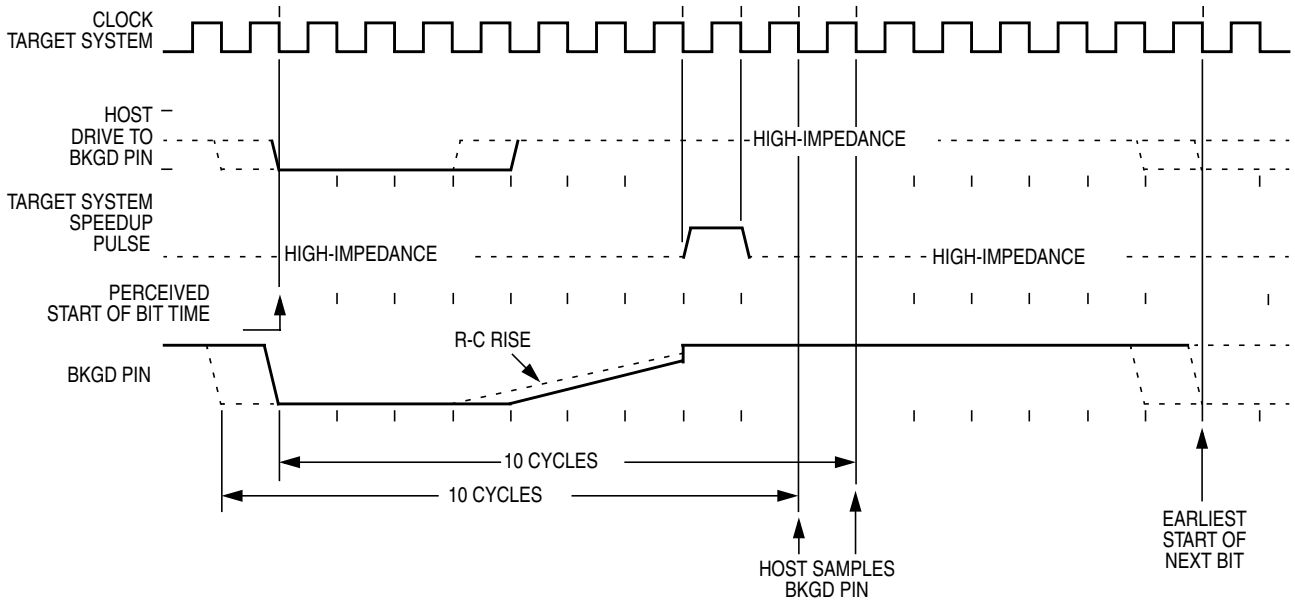


Figure 17-8. BDM Target-to-Host Serial Bit Timing (Logic 1)

Figure 17-9 shows the host receiving a logic 0 from the target. Because the host is asynchronous to the target, there is up to a one clock-cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target. The host initiates the bit time but the target finishes it. Because the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 target clock cycles then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 target clock cycles after starting the bit time.

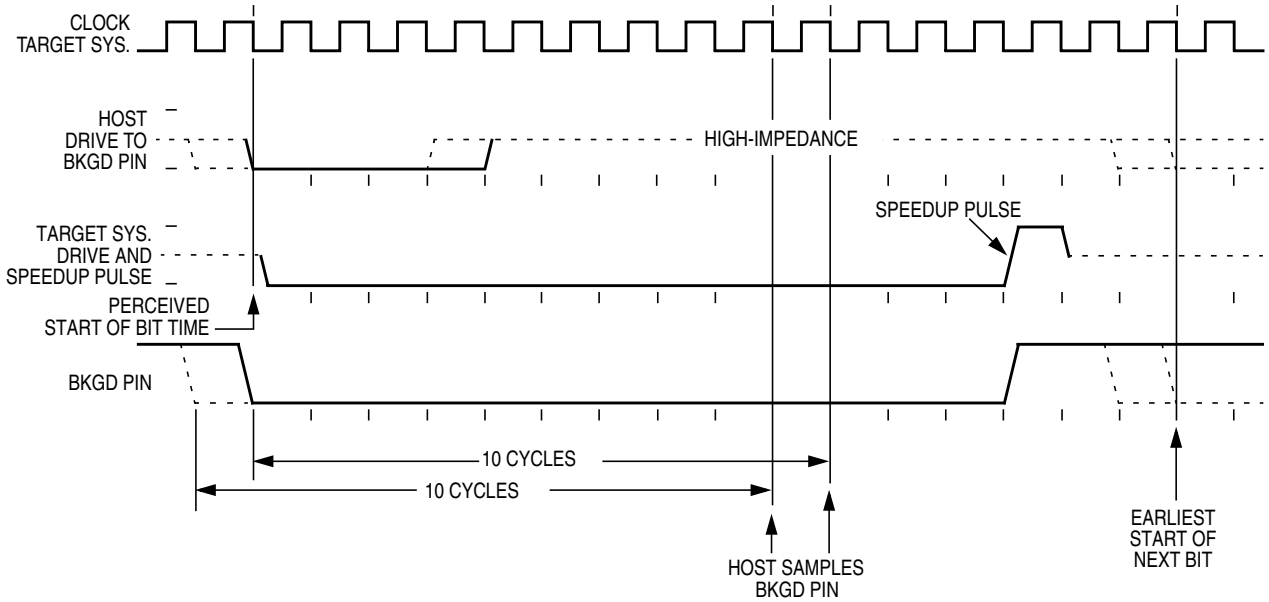


Figure 17-9. BDM Target-to-Host Serial Bit Timing (Logic 0)

17.4.7 Serial Interface Hardware Handshake Protocol

BDM commands that require CPU execution are ultimately treated at the MCU bus rate. Because the BDM clock source can be asynchronously related to the bus frequency, when CLKSW = 0, it is very helpful to provide a handshake protocol in which the host could determine when an issued command is executed by the CPU. The alternative is to always wait the amount of time equal to the appropriate number of cycles at the slowest possible rate the clock could be running. This sub-section will describe the hardware handshake protocol.

The hardware handshake protocol signals to the host controller when an issued command was successfully executed by the target. This protocol is implemented by a 16 serial clock cycle low pulse followed by a brief speedup pulse in the BKGD pin. This pulse is generated by the target MCU when a command, issued by the host, has been successfully executed (see Figure 17-10). This pulse is referred to as the ACK pulse. After the ACK pulse has finished: the host can start the bit retrieval if the last issued command was a read command, or start a new command if the last command was a write command or a control command (BACKGROUND, GO, GO_UNTIL, or TRACE1). The ACK pulse is not issued earlier than 32 serial clock cycles after the BDM command was issued. The end of the BDM command is assumed to be the 16th tick of the last bit. This minimum delay assures enough time for the host to perceive the ACK pulse. Note also that, there is no upper limit for the delay between the command and the related ACK pulse, because the command execution depends upon the CPU bus frequency, which in some cases could be very slow compared to the serial communication rate. This protocol allows a great flexibility for the POD designers, because it does not rely on any accurate time measurement or short response time to any event in the serial communication.

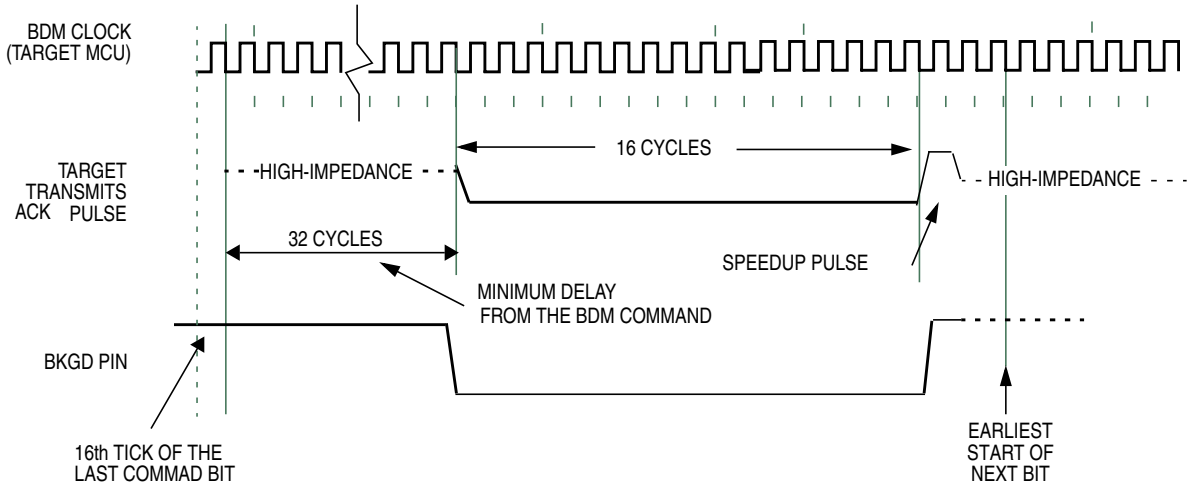


Figure 17-10. Target Acknowledge Pulse (ACK)

NOTE

If the ACK pulse was issued by the target, the host assumes the previous command was executed. If the CPU enters WAIT or STOP prior to executing a hardware command, the ACK pulse will not be issued meaning that the BDM command was not executed. After entering wait or stop mode, the BDM command is no longer pending.

Figure 17-11 shows the ACK handshake protocol in a command level timing diagram. The READ_BYTE instruction is used as an example. First, the 8-bit instruction opcode is sent by the host, followed by the address of the memory location to be read. The target BDM decodes the instruction. A bus cycle is grabbed (free or stolen) by the BDM and it executes the READ_BYTE operation. Having retrieved the data, the BDM issues an ACK pulse to the host controller, indicating that the addressed byte is ready to be retrieved. After detecting the ACK pulse, the host initiates the byte retrieval process. Note that data is sent in the form of a word and the host needs to determine which is the appropriate byte based on whether the address was odd or even.

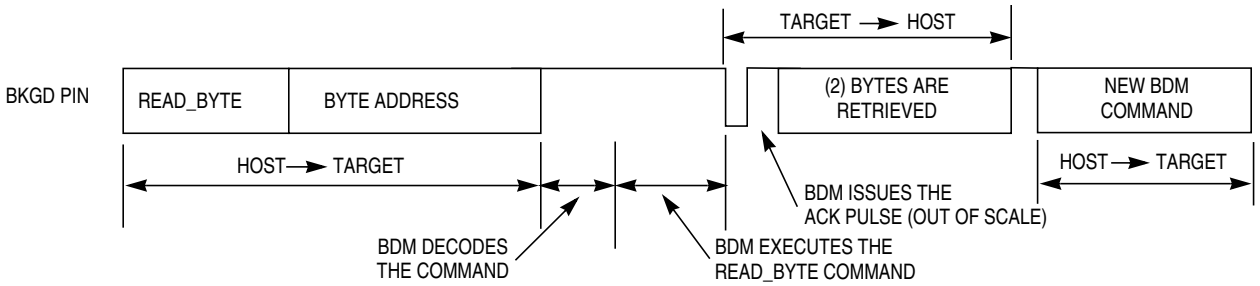


Figure 17-11. Handshake Protocol at Command Level

Differently from the normal bit transfer (where the host initiates the transmission), the serial interface ACK handshake pulse is initiated by the target MCU by issuing a falling edge in the BKGD pin. The hardware handshake protocol in Figure 17-10 specifies the timing when the BKGD pin is being driven, so the host should follow this timing constraint in order to avoid the risk of an electrical conflict in the BKGD pin.

NOTE

The only place the BKGD pin can have an electrical conflict is when one side is driving low and the other side is issuing a speedup pulse (high). Other “highs” are pulled rather than driven. However, at low rates the time of the speedup pulse can become lengthy and so the potential conflict time becomes longer as well.

The ACK handshake protocol does not support nested ACK pulses. If a BDM command is not acknowledge by an ACK pulse, the host needs to abort the pending command first in order to be able to issue a new BDM command. When the CPU enters WAIT or STOP while the host issues a command that requires CPU execution (e.g., WRITE_BYTE), the target discards the incoming command due to the WAIT or STOP being detected. Therefore, the command is not acknowledged by the target, which means that the ACK pulse will not be issued in this case. After a certain time the host should decide to abort the ACK sequence in order to be free to issue a new command. Therefore, the protocol should provide a mechanism in which a command, and therefore a pending ACK, could be aborted.

NOTE

Differently from a regular BDM command, the ACK pulse does not provide a time out. This means that in the case of a WAIT or STOP instruction being executed, the ACK would be prevented from being issued. If not aborted, the ACK would remain pending indefinitely. See the handshake abort procedure described in Section 17.4.8, “Hardware Handshake Abort Procedure.”

17.4.8 Hardware Handshake Abort Procedure

The abort procedure is based on the SYNC command. In order to abort a command, which had not issued the corresponding ACK pulse, the host controller should generate a low pulse in the BKGD pin by driving it low for at least 128 serial clock cycles and then driving it high for one serial clock cycle, providing a speedup pulse. By detecting this long low pulse in the BKGD pin, the target executes the SYNC protocol, see Section 17.4.9, “SYNC — Request Timed Reference Pulse,” and assumes that the pending command and therefore the related ACK pulse, are being aborted. Therefore, after the SYNC protocol has been completed the host is free to issue new BDM commands.

Although it is not recommended, the host could abort a pending BDM command by issuing a low pulse in the BKGD pin shorter than 128 serial clock cycles, which will not be interpreted as the SYNC command. The ACK is actually aborted when a falling edge is perceived by the target in the BKGD pin. The short abort pulse should have at least 4 clock cycles keeping the BKGD pin low, in order to allow the falling edge to be detected by the target. In this case, the target will not execute the SYNC protocol but the pending command will be aborted along with the ACK pulse. The potential problem with this abort procedure is when there is a conflict between the ACK pulse and the short abort pulse. In this case, the target may not perceive the abort pulse. The worst case is when the pending command is a read command (i.e., READ_BYTE). If the abort pulse is not perceived by the target the host will attempt to send a new command after the abort pulse was issued, while the target expects the host to retrieve the accessed memory byte. In this case, host and target will run out of synchronism. However, if the command to be aborted is not a read command the short abort pulse could be used. After a command is aborted the target assumes the next falling edge, after the abort pulse, is the first bit of a new BDM command.

NOTE

The details about the short abort pulse are being provided only as a reference for the reader to better understand the BDM internal behavior. It is not recommended that this procedure be used in a real application.

Because the host knows the target serial clock frequency, the SYNC command (used to abort a command) does not need to consider the lower possible target frequency. In this case, the host could issue a SYNC very close to the 128 serial clock cycles length. Providing a small overhead on the pulse length in order to assure the SYNC pulse will not be misinterpreted by the target. See Section 17.4.9, “SYNC — Request Timed Reference Pulse.”

Figure 17-12 shows a SYNC command being issued after a READ_BYTE, which aborts the READ_BYTE command. Note that, after the command is aborted a new command could be issued by the host computer.

NOTE

Figure 17-12 does not represent the signals in a true timing scale

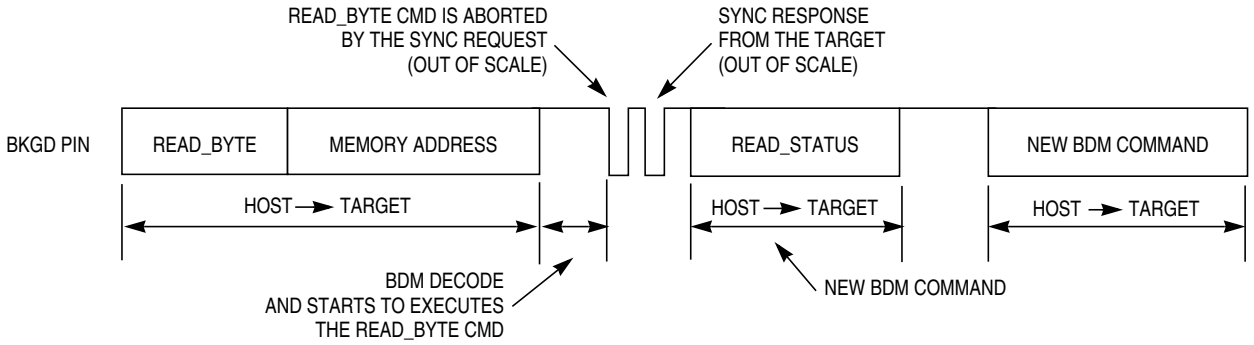


Figure 17-12. ACK Abort Procedure at the Command Level

Figure 17-13 shows a conflict between the ACK pulse and the SYNC request pulse. This conflict could occur if a POD device is connected to the target BKGD pin and the target is already in debug active mode. Consider that the target CPU is executing a pending BDM command at the exact moment the POD is being connected to the BKGD pin. In this case, an ACK pulse is issued along with the SYNC command. In this case, there is an electrical conflict between the ACK speedup pulse and the SYNC pulse. Because this is not a probable situation, the protocol does not prevent this conflict from happening.

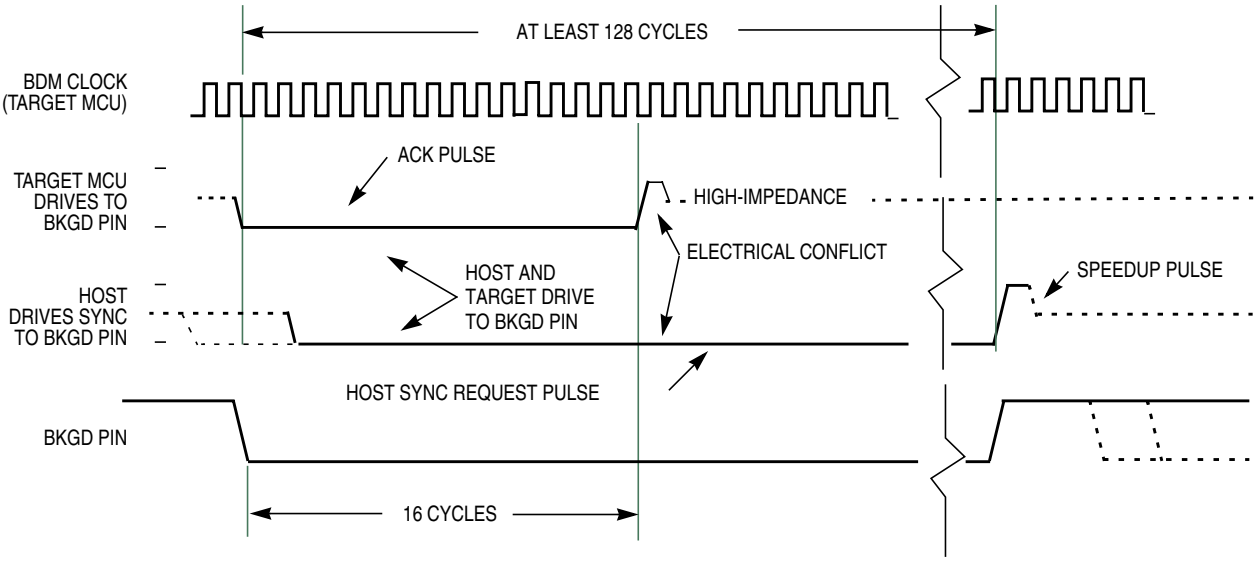


Figure 17-13. ACK Pulse and SYNC Request Conflict

NOTE

This information is being provided so that the MCU integrator will be aware that such a conflict could eventually occur.

The hardware handshake protocol is enabled by the ACK_ENABLE and disabled by the ACK_DISABLE BDM commands. This provides backwards compatibility with the existing POD devices which are not able to execute the hardware handshake protocol. It also allows for new POD devices, that support the hardware handshake protocol, to freely communicate with the target device. If desired, without the need for waiting for the ACK pulse.

The commands are described as follows:

- ACK_ENABLE — enables the hardware handshake protocol. The target will issue the ACK pulse when a CPU command is executed by the CPU. The ACK_ENABLE command itself also has the ACK pulse as a response.
- ACK_DISABLE — disables the ACK pulse protocol. In this case, the host needs to use the worst case delay time at the appropriate places in the protocol.

The default state of the BDM after reset is hardware handshake protocol disabled.

All the read commands will ACK (if enabled) when the data bus cycle has completed and the data is then ready for reading out by the BKGD serial pin. All the write commands will ACK (if enabled) after the data has been received by the BDM through the BKGD serial pin and when the data bus cycle is complete. See Section 17.4.3, “BDM Hardware Commands,” and Section 17.4.4, “Standard BDM Firmware Commands,” for more information on the BDM commands.

The ACK_ENABLE sends an ACK pulse when the command has been completed. This feature could be used by the host to evaluate if the target supports the hardware handshake protocol. If an ACK pulse is issued in response to this command, the host knows that the target supports the hardware handshake protocol. If the target does not support the hardware handshake protocol the ACK pulse is not issued. In this case, the ACK_ENABLE command is ignored by the target because it is not recognized as a valid command.

The BACKGROUND command will issue an ACK pulse when the CPU changes from normal to background mode. The ACK pulse related to this command could be aborted using the SYNC command.

The GO command will issue an ACK pulse when the CPU exits from background mode. The ACK pulse related to this command could be aborted using the SYNC command.

The GO_UNTIL command is equivalent to a GO command with exception that the ACK pulse, in this case, is issued when the CPU enters into background mode. This command is an alternative to the GO command and should be used when the host wants to trace if a breakpoint match occurs and causes the CPU to enter active background mode. Note that the ACK is issued whenever the CPU enters BDM, which could be caused by a breakpoint match or by a BGND instruction being executed. The ACK pulse related to this command could be aborted using the SYNC command.

The TRACE1 command has the related ACK pulse issued when the CPU enters background active mode after one instruction of the application program is executed. The ACK pulse related to this command could be aborted using the SYNC command.

The TAGGO command will not issue an ACK pulse because this would interfere with the tagging function shared on the same pin.

17.4.9 SYNC — Request Timed Reference Pulse

The SYNC command is unlike other BDM commands because the host does not necessarily know the correct communication speed to use for BDM communications until after it has analyzed the response to the SYNC command. To issue a SYNC command, the host should perform the following steps:

1. Drive the BKGD pin low for at least 128 cycles at the lowest possible BDM serial communication frequency (the lowest serial communication frequency is determined by the crystal oscillator or the clock chosen by CLKSW.)
2. Drive BKGD high for a brief speedup pulse to get a fast rise time (this speedup pulse is typically one cycle of the host clock.)
3. Remove all drive to the BKGD pin so it reverts to high impedance.
4. Listen to the BKGD pin for the sync response pulse.

Upon detecting the SYNC request from the host, the target performs the following steps:

1. Discards any incomplete command received or bit retrieved.
2. Waits for BKGD to return to a logic 1.
3. Delays 16 cycles to allow the host to stop driving the high speedup pulse.
4. Drives BKGD low for 128 cycles at the current BDM serial communication frequency.
5. Drives a one-cycle high speedup pulse to force a fast rise time on BKGD.
6. Removes all drive to the BKGD pin so it reverts to high impedance.

The host measures the low time of this 128 cycle SYNC response pulse and determines the correct speed for subsequent BDM communications. Typically, the host can determine the correct communication speed within a few percent of the actual target speed and the communication protocol can easily tolerate speed errors of several percent.

As soon as the SYNC request is detected by the target, any partially received command or bit retrieved is discarded. This is referred to as a soft-reset, equivalent to a time-out in the serial communication. After the SYNC response, the target will consider the next falling edge (issued by the host) as the start of a new BDM command or the start of new SYNC request.

Another use of the SYNC command pulse is to abort a pending ACK pulse. The behavior is exactly the same as in a regular SYNC command. Note that one of the possible causes for a command to not be acknowledged by the target is a host-target synchronization problem. In this case, the command may not have been understood by the target and so an ACK response pulse will not be issued.

17.4.10 Instruction Tracing

When a TRACE1 command is issued to the BDM in active BDM, the CPU exits the standard BDM firmware and executes a single instruction in the user code. As soon as this has occurred, the CPU is forced to return to the standard BDM firmware and the BDM is active and ready to receive a new command. If the TRACE1 command is issued again, the next user instruction will be executed. This facilitates stepping or tracing through the user code one instruction at a time.

If an interrupt is pending when a TRACE1 command is issued, the interrupt stacking operation occurs but no user instruction is executed. Upon return to standard BDM firmware execution, the program counter points to the first instruction in the interrupt service routine.

17.4.11 Instruction Tagging

The instruction queue and cycle-by-cycle CPU activity are reconstructible in real time or from trace history that is captured by a logic analyzer. However, the reconstructed queue cannot be used to stop the CPU at a specific instruction. This is because execution already has begun by the time an operation is visible outside the system. A separate instruction tagging mechanism is provided for this purpose.

The tag follows program information as it advances through the instruction queue. When a tagged instruction reaches the head of the queue, the CPU enters active BDM rather than executing the instruction.

NOTE

Tagging is disabled when BDM becomes active and BDM serial commands are not processed while tagging is active.

Executing the BDM TAGGO command configures two system pins for tagging. The $\overline{\text{TAGLO}}$ signal shares a pin with the $\overline{\text{LSTRB}}$ signal, and the $\overline{\text{TAGHI}}$ signal shares a pin with the BKGD signal.

Table 17-7 shows the functions of the two tagging pins. The pins operate independently, that is the state of one pin does not affect the function of the other. The presence of logic level 0 on either pin at the fall of the external clock (ECLK) performs the indicated function. High tagging is allowed in all modes. Low tagging is allowed only when low strobe is enabled (LSTRB is allowed only in wide expanded modes and emulation expanded narrow mode).

Table 17-7. Tag Pin Function

$\overline{\text{TAGHI}}$	$\overline{\text{TAGLO}}$	Tag
1	1	No tag
1	0	Low byte
0	1	High byte
0	0	Both bytes

17.4.12 Serial Communication Time-Out

The host initiates a host-to-target serial transmission by generating a falling edge on the BKGD pin. If BKGD is kept low for more than 128 target clock cycles, the target understands that a SYNC command was issued. In this case, the target will keep waiting for a rising edge on BKGD in order to answer the SYNC request pulse. If the rising edge is not detected, the target will keep waiting forever without any time-out limit.

Consider now the case where the host returns BKGD to logic one before 128 cycles. This is interpreted as a valid bit transmission, and not as a SYNC request. The target will keep waiting for another falling edge marking the start of a new bit. If, however, a new falling edge is not detected by the target within 512 clock cycles since the last falling edge, a time-out occurs and the current command is discarded without affecting memory or the operating mode of the MCU. This is referred to as a soft-reset.

If a read command is issued but the data is not retrieved within 512 serial clock cycles, a soft-reset will occur causing the command to be disregarded. The data is not available for retrieval after the time-out has occurred. This is the expected behavior if the handshake protocol is not enabled. However, consider the behavior where the BDC is running in a frequency much greater than the CPU frequency. In this case, the command could time out before the data is ready to be retrieved. In order to allow the data to be retrieved even with a large clock frequency mismatch (between BDC and CPU) when the hardware handshake protocol is enabled, the time out between a read command and the data retrieval is disabled. Therefore, the host could wait for more then 512 serial clock cycles and continue to be able to retrieve the data from an issued read command. However, as soon as the handshake pulse (ACK pulse) is issued, the time-out feature is re-activated, meaning that the target will time out after 512 clock cycles. Therefore, the host needs to retrieve the data within a 512 serial clock cycles time frame after the ACK pulse had been issued. After that period, the read command is discarded and the data is no longer available for retrieval. Any falling edge of the BKGD pin after the time-out period is considered to be a new command or a SYNC request.

Note that whenever a partially issued command, or partially retrieved data, has occurred the time out in the serial communication is active. This means that if a time frame higher than 512 serial clock cycles is observed between two consecutive negative edges and the command being issued or data being retrieved is not complete, a soft-reset will occur causing the partially received command or data retrieved to be disregarded. The next falling edge of the BKGD pin, after a soft-reset has occurred, is considered by the target as the start of a new BDM command, or the start of a SYNC request pulse.

17.4.13 Operation in Wait Mode

The BDM cannot be used in wait mode if the system disables the clocks to the BDM.

There is a clearing mechanism associated with the WAIT instruction when the clocks to the BDM (CPU core platform) are disabled. As the clocks restart from wait mode, the BDM receives a soft reset (clearing any command in progress) and the ACK function will be disabled. This is a change from previous BDM modules.

17.4.14 Operation in Stop Mode

The BDM is completely shutdown in stop mode.

There is a clearing mechanism associated with the STOP instruction. STOP must be enabled and the part must go into stop mode for this to occur. As the clocks restart from stop mode, the BDM receives a soft reset (clearing any command in progress) and the ACK function will be disabled. This is a change from previous BDM modules.

Chapter 18
Debug Module (DBGV1)

18.1 Introduction

This section describes the functionality of the debug (DBG) sub-block of the HCS12 core platform.

The DBG module is designed to be fully compatible with the existing BKP_HCS12_A module (BKP mode) and furthermore provides an on-chip trace buffer with flexible triggering capability (DBG mode). The DBG module provides for non-intrusive debug of application software. The DBG module is optimized for the HCS12 16-bit architecture.

18.1.1 Features

The DBG module in BKP mode includes these distinctive features:

- Full or dual breakpoint mode
  - Compare on address and data (full)
  - Compare on either of two addresses (dual)
- BDM or SWI breakpoint
  - Enter BDM on breakpoint (BDM)
  - Execute SWI on breakpoint (SWI)
- Tagged or forced breakpoint
  - Break just before a specific instruction will begin execution (TAG)
  - Break on the first instruction boundary after a match occurs (Force)
- Single, range, or page address compares
  - Compare on address (single)
  - Compare on address 256 byte (range)
  - Compare on any 16K page (page)
- At forced breakpoints compare address on read or write
- High and/or low byte data compares
- Comparator C can provide an additional tag or force breakpoint (enhancement for BKP mode)

The DBG in DBG mode includes these distinctive features:

- Three comparators (A, B, and C)
  - Dual mode, comparators A and B used to compare addresses
  - Full mode, comparator A compares address and comparator B compares data
  - Can be used as trigger and/or breakpoint
  - Comparator C used in LOOP1 capture mode or as additional breakpoint
- Four capture modes
  - Normal mode, change-of-flow information is captured based on trigger specification
  - Loop1 mode, comparator C is dynamically updated to prevent redundant change-of-flow storage.
  - Detail mode, address and data for all cycles except program fetch (P) and free (f) cycles are stored in trace buffer
  - Profile mode, last instruction address executed by CPU is returned when trace buffer address is read
- Two types of breakpoint or debug triggers
  - Break just before a specific instruction will begin execution (tag)
  - Break on the first instruction boundary after a match occurs (force)
- BDM or SWI breakpoint
  - Enter BDM on breakpoint (BDM)
  - Execute SWI on breakpoint (SWI)
- Nine trigger modes for comparators A and B
  - A
  - A or B
  - A then B
  - A and B, where B is data (full mode)
  - A and not B, where B is data (full mode)
  - Event only B, store data
  - A then event only B, store data
  - Inside range, $A \leq address \leq B$
  - Outside range, $address < A$ or $address > B$
- Comparator C provides an additional tag or force breakpoint when capture mode is not configured in LOOP1 mode.
- Sixty-four word (16 bits wide) trace buffer for storing change-of-flow information, event only data and other bus information.
  - Source address of taken conditional branches (long, short, bit-conditional, and loop constructs)
  - Destination address of indexed JMP, JSR, and CALL instruction.
  - Destination address of RTI, RTS, and RTC instructions
  - Vector address of interrupts, except for SWI and BDM vectors

— Data associated with event B trigger modes
— Detail report mode stores address and data for all cycles except program (P) and free (f) cycles
— Current instruction address when in profiling mode
— BGND is not considered a change-of-flow (cof) by the debugger

18.1.2 Modes of Operation

There are two main modes of operation: breakpoint mode and debug mode. Each one is mutually exclusive of the other and selected via a software programmable control bit.

In the breakpoint mode there are two sub-modes of operation:

- Dual address mode, where a match on either of two addresses will cause the system to enter background debug mode (BDM) or initiate a software interrupt (SWI).
- Full breakpoint mode, where a match on address and data will cause the system to enter background debug mode (BDM) or initiate a software interrupt (SWI).

In debug mode, there are several sub-modes of operation.

- Trigger modes

  There are many ways to create a logical trigger. The trigger can be used to capture bus information either starting from the trigger or ending at the trigger. Types of triggers (A and B are registers):

  — A only
  — A or B
  — A then B
  — Event only B (data capture)
  — A then event only B (data capture)
  — A and B, full mode
  — A and not B, full mode
  — Inside range
  — Outside range

- Capture modes

  There are several capture modes. These determine which bus information is saved and which is ignored.

  — Normal: save change-of-flow program fetches
  — Loop1: save change-of-flow program fetches, ignoring duplicates
  — Detail: save all bus operations except program and free cycles
  — Profile: poll target from external device

18.1.3 Block Diagram

Figure 18-1 is a block diagram of this module in breakpoint mode. Figure 18-2 is a block diagram of this module in debug mode.

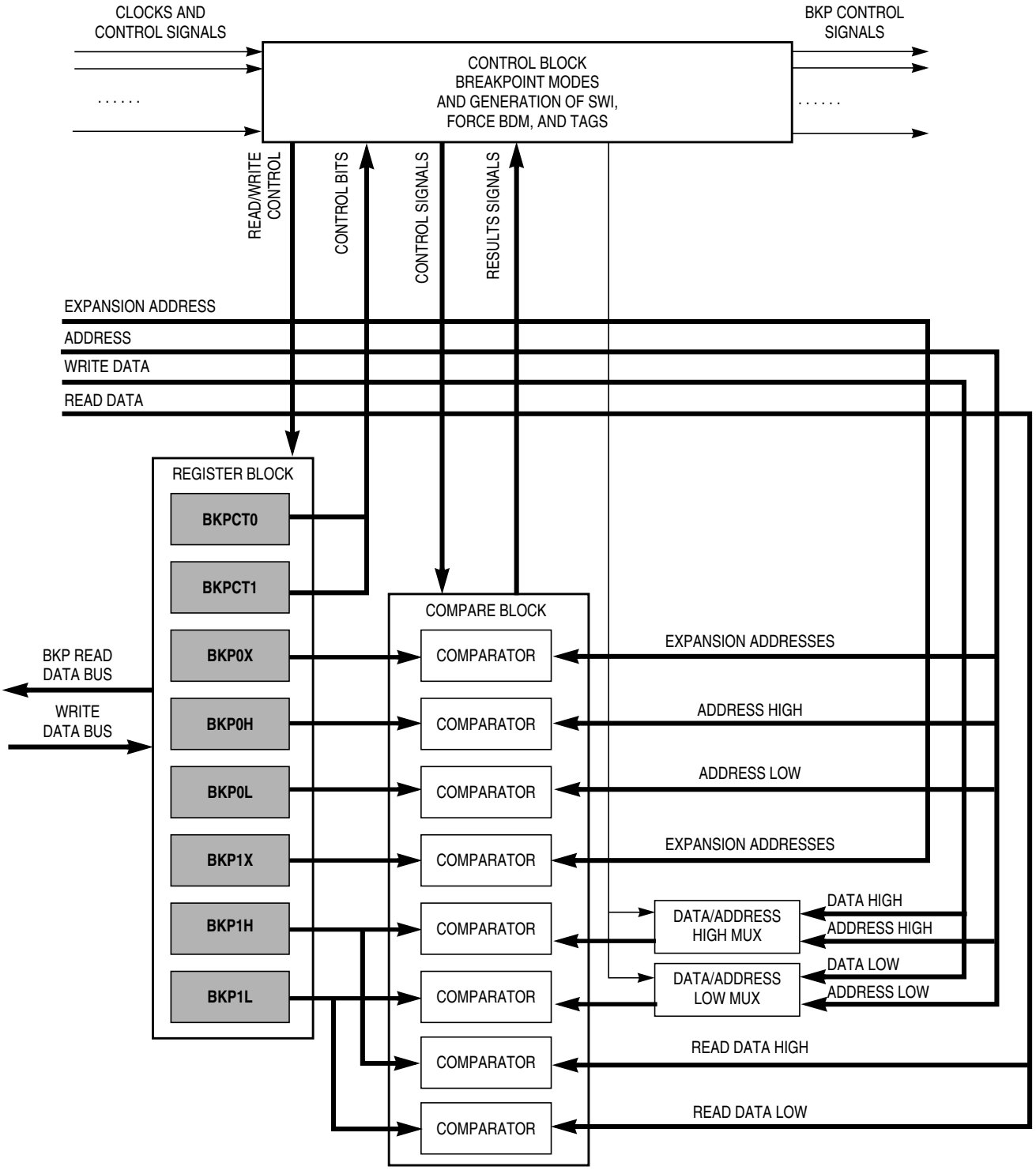


Figure 18-1. DBG Block Diagram in BKP Mode

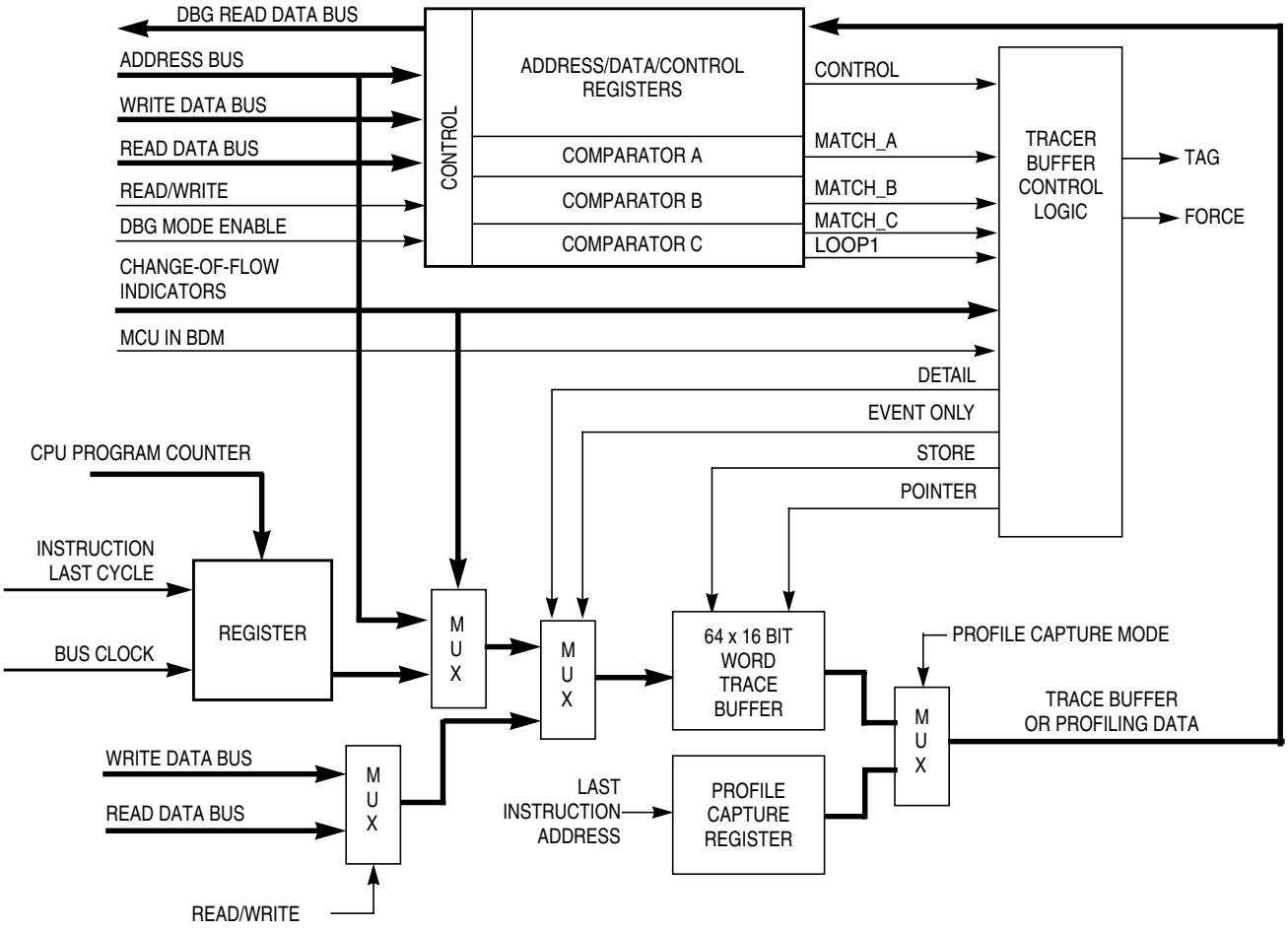


Figure 18-2. DBG Block Diagram in DBG Mode

18.2 External Signal Description

The DBG sub-module relies on the external bus interface (generally the MEBI) when the DBG is matching on the external bus.

The tag pins in Table 18-1 (part of the MEBI) may also be a part of the breakpoint operation.

Table 18-1. External System Pins Associated with DBG and MEBI

Pin Name	Pin Functions	Description
BKGD/MODC/ $\overline{\text{TAGHI}}$	$\overline{\text{TAGHI}}$	When instruction tagging is on, a 0 at the falling edge of E tags the high half of the instruction word being read into the instruction queue.
PE3/$\overline{\text{LSTRB}}$/ $\overline{\text{TAGLO}}$	$\overline{\text{TAGLO}}$	In expanded wide mode or emulation narrow modes, when instruction tagging is on and low strobe is enabled, a 0 at the falling edge of E tags the low half of the instruction word being read into the instruction queue.

18.3 Memory Map and Register Definition

A summary of the registers associated with the DBG sub-block is shown in Figure 18-3. Detailed descriptions of the registers and bits are given in the subsections that follow.

18.3.1 Module Memory Map

Table 18-2. DBGV1 Memory Map

Address Offset	Use	Access
	Debug Control Register (DBGC1)	R/W
	Debug Status and Control Register (DBGSC)	R/W
	Debug Trace Buffer Register High (DBGTBH)	R
	Debug Trace Buffer Register Low (DBGTBL)	R
4	Debug Count Register (DBGCNT)	R
5	Debug Comparator C Extended Register (DBGCCX)	R/W
6	Debug Comparator C Register High (DBGCCH)	R/W
	Debug Comparator C Register Low (DBGCCL)	R/W
8	Debug Control Register 2 (DBGC2) / (BKPCT0)	R/W
9	Debug Control Register 3 (DBGC3) / (BKPCT1)	R/W
A	Debug Comparator A Extended Register (DBGCAX) / (/BKP0X)	R/W
B	Debug Comparator A Register High (DBGCAH) / (BKP0H)	R/W
	Debug Comparator A Register Low (DBGCAL) / (BKP0L)	R/W
	Debug Comparator B Extended Register (DBGCBX) / (BKP1X)	R/W
E	Debug Comparator B Register High (DBGCBH) / (BKP1H)	R/W
F	Debug Comparator B Register Low (DBGCBL) / (BKP1L)	R/W

18.3.2 Register Descriptions

This section consists of the DBG register descriptions in address order. Most of the register bits can be written to in either BKP or DBG mode, although they may not have any effect in one of the modes. However, the only bits in the DBG module that can be written while the debugger is armed (ARM = 1) are DBGEN and ARM

Name[1]		Bit 7	6	5	4	3	2	1	Bit 0
DBGC1	R	DBGEN	ARM	TRGSEL	BEGIN	DBGBRK	0	CAPMOD	
	W								
DBGSC	R	AF	BF	CF	0	TRG			
	W								

= Unimplemented or Reserved

Figure 18-3. DBG Register Summary

Name[1]		Bit 7	6	5	4	3	2	1	Bit 0
DBGTBH	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
DBGTBL	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
DBGCNT	R	TBF	0	CNT					
	W								
DBGCCX(2)	R	PAGSEL		EXTCMP					
	W								
DBGCCH(2)	R	Bit 15	14	13	12	11	10	9	Bit 8
	W								
DBGCCL(2)	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
DBGC2 BKPCT0	R	BKABEN	FULL	BDM	TAGAB	BKCEN	TAGC	RWCEN	RWC
	W								
DBGC3 BKPCT1	R	BKAMBH	BKAMBL	BKBMBH	BKBMBL	RWAEN	RWA	RWBEN	RWB
	W								
DBGCAX BKP0X	R	PAGSEL		EXTCMP					
	W								
DBGCAH BKP0H	R	Bit 15	14	13	12	11	10	9	Bit 8
	W								
DBGCAL BKP0L	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
DBGCBX BKP1X	R	PAGSEL		EXTCMP					
	W								
DBGCBH BKP1H	R	Bit 15	14	13	12	11	10	9	Bit 8
	W								
DBGCBL BKP1L	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								

= Unimplemented or Reserved

Figure 18-3. DBG Register Summary (continued)

1 The DBG module is designed for backwards compatibility to existing BKP modules. Register and bit names have changed from the BKP module. This column shows the DBG register name, as well as the BKP register name for reference.

2 Comparator C can be used to enhance the BKP mode by providing a third breakpoint.

18.3.2.1 Debug Control Register 1 (DBGC1)

NOTE

All bits are used in DBG mode only.

	7	6	5	4	3	2	1	0
R	DBGEN	ARM	TRGSEL	BEGIN	DBGBRK	0	CAPMOD	
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 18-4. Debug Control Register (DBGC1)

NOTE

This register cannot be written if BKP mode is enabled (BKABEN in DBGC2 is set).

Table 18-3. DBGC1 Field Descriptions

Field	Description
7 DBGEN	**DBG Mode Enable Bit** — The DBGEN bit enables the DBG module for use in DBG mode. This bit cannot be set if the MCU is in secure mode. 0 DBG mode disabled 1 DBG mode enabled
6 ARM	**Arm Bit** — The ARM bit controls whether the debugger is comparing and storing data in the trace buffer. See Section 18.4.2.4, “Arming the DBG Module,” for more information. 0 Debugger unarmed 1 Debugger armed **Note:** This bit cannot be set if the DBGEN bit is not also being set at the same time. For example, a write of 01 to DBGEN[7:6] will be interpreted as a write of 00.
5 TRGSEL	**Trigger Selection Bit** — The TRGSEL bit controls the triggering condition for comparators A and B in DBG mode. It serves essentially the same function as the TAGAB bit in the DBGC2 register does in BKP mode. See Section 18.4.2.1.2, “Trigger Selection,” for more information. TRGSEL may also determine the type of breakpoint based on comparator A and B if enabled in DBG mode (DBGBRK = 1). Please refer to Section 18.4.3.1, “Breakpoint Based on Comparator A and B.” 0 Trigger on any compare address match 1 Trigger before opcode at compare address gets executed (tagged-type)
4 BEGIN	**Begin/End Trigger Bit** — The BEGIN bit controls whether the trigger begins or ends storing of data in the trace buffer. See Section 18.4.2.8.1, “Storing with Begin-Trigger,” and Section 18.4.2.8.2, “Storing with End-Trigger,” for more details. 0 Trigger at end of stored data 1 Trigger before storing data

Table 18-3. DBGC1 Field Descriptions (continued)

Field	Description
3 DBGBRK	**DBG Breakpoint Enable Bit** — The DBGBRK bit controls whether the debugger will request a breakpoint based on comparator A and B to the CPU upon completion of a tracing session. Please refer to Section 18.4.3, “Breakpoints,” for further details. 0 CPU break request not enabled 1 CPU break request enabled
1:0 CAPMOD	**Capture Mode Field** — See Table 18-4 for capture mode field definitions. In LOOP1 mode, the debugger will automatically inhibit redundant entries into capture memory. In detail mode, the debugger is storing address and data for all cycles except program fetch (P) and free (f) cycles. In profile mode, the debugger is returning the address of the last instruction executed by the CPU on each access of trace buffer address. Refer to Section 18.4.2.6, “Capture Modes,” for more information.

Table 18-4. CAPMOD Encoding

CAPMOD	Description
00	Normal
01	LOOP1
10	DETAIL
11	PROFILE

18.3.2.2 Debug Status and Control Register (DBGSC)

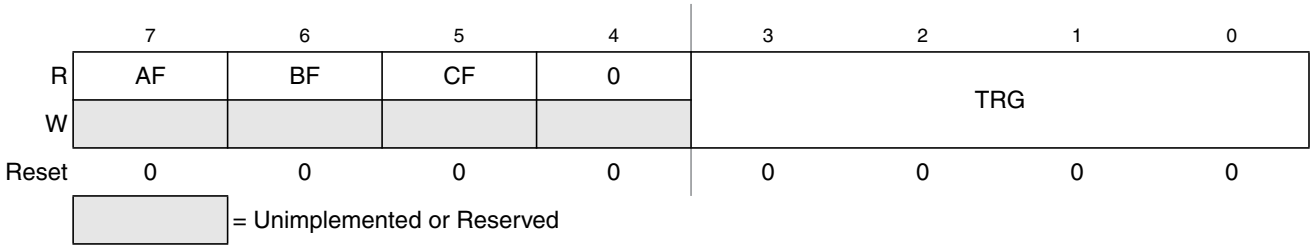


Figure 18-5. Debug Status and Control Register (DBGSC)

Table 18-5. DBGSC Field Descriptions

Field	Description
7 AF	**Trigger A Match Flag** — The AF bit indicates if trigger A match condition was met since arming. This bit is cleared when ARM in DBGC1 is written to a 1 or on any write to this register. 0 Trigger A did not match 1 Trigger A match
6 BF	**Trigger B Match Flag** — The BF bit indicates if trigger B match condition was met since arming.This bit is cleared when ARM in DBGC1 is written to a 1 or on any write to this register. 0 Trigger B did not match 1 Trigger B match
5 CF	**Comparator C Match Flag** — The CF bit indicates if comparator C match condition was met since arming.This bit is cleared when ARM in DBGC1 is written to a 1 or on any write to this register. 0 Comparator C did not match 1 Comparator C match
3:0 TRG	**Trigger Mode Bits** — The TRG bits select the trigger mode of the DBG module as shown Table 18-6. See Section 18.4.2.5, “Trigger Modes,” for more detail.

Table 18-6. Trigger Mode Encoding

TRG Value	Meaning
0000	A only
0001	A or B
0010	A then B
0011	Event only B
0100	A then event only B
0101	A and B (full mode)
0110	A and Not B (full mode)
0111	Inside range
1000	Outside range
1001 ↓ 1111	Reserved (Defaults to A only)

18.3.2.3 Debug Trace Buffer Register (DBGTB)

	15	14	13	12	11	10	9	8
R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved

Figure 18-6. Debug Trace Buffer Register High (DBGTBH)

	7	6	5	4	3	2	1	0
R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved

Figure 18-7. Debug Trace Buffer Register Low (DBGTBL)

Table 18-7. DBGTB Field Descriptions

Field	Description
15:0	**Trace Buffer Data Bits** — The trace buffer data bits contain the data of the trace buffer. This register can be read only as a word read. Any byte reads or misaligned access of these registers will return 0 and will not cause the trace buffer pointer to increment to the next trace buffer address. The same is true for word reads while the debugger is armed. In addition, this register may appear to contain incorrect data if it is not read with the same capture mode bit settings as when the trace buffer data was recorded (See Section 18.4.2.9, “Reading Data from Trace Buffer”). Because reads will reflect the contents of the trace buffer RAM, the reset state is undefined.

18.3.2.4 Debug Count Register (DBGCNT)

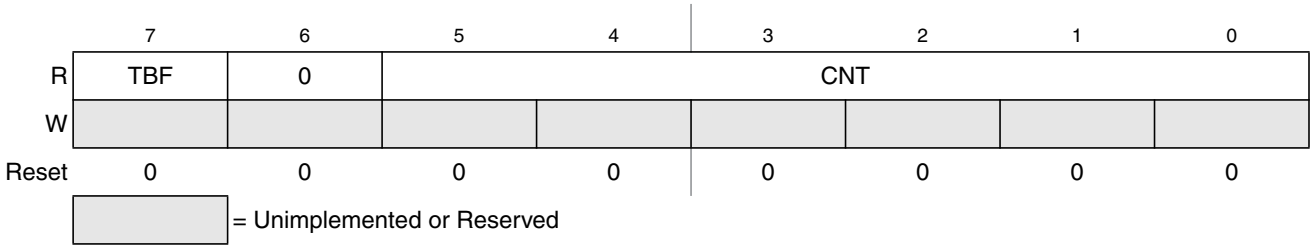


Figure 18-8. Debug Count Register (DBGCNT)

Table 18-8. DBGCNT Field Descriptions

Field	Description
7 TBF	**Trace Buffer Full** — The TBF bit indicates that the trace buffer has stored 64 or more words of data since it was last armed. If this bit is set, then all 64 words will be valid data, regardless of the value in CNT[5:0]. The TBF bit is cleared when ARM in DBGC1 is written to a 1.
5:0 CNT	**Count Value** — The CNT bits indicate the number of valid data words stored in the trace buffer. Table 18-9 shows the correlation between the CNT bits and the number of valid data words in the trace buffer. When the CNT rolls over to 0, the TBF bit will be set and incrementing of CNT will continue if DBG is in end-trigger mode. The DBGCNT register is cleared when ARM in DBGC1 is written to a 1.

Table 18-9. CNT Decoding Table

TBF	CNT	Description
0	000000	No data valid
0	000001	1 word valid
0	000010 111110	2 words valid .. .. 62 words valid
0	111111	63 words valid
1	000000	64 words valid; if BEGIN = 1, the ARM bit will be cleared. A breakpoint will be generated if DBGBRK = 1
1	000001 111111	64 words valid, oldest data has been overwritten by most recent data

18.3.2.5 Debug Comparator C Extended Register (DBGCCX)

Figure 18-9. Debug Comparator C Extended Register (DBGCCX)

Table 18-10. DBGCCX Field Descriptions

Field	Description
7:6 PAGSEL	**Page Selector Field** — In both BKP and DBG mode, PAGSEL selects the type of paging as shown in Table 18-11. DPAGE and EPAGE are not yet implemented so the value in bit 7 will be ignored (i.e., PAGSEL values of 10 and 11 will be interpreted as values of 00 and 01, respectively).
5:0 EXTCMP	**Comparator C Extended Compare Bits** — The EXTCMP bits are used as comparison address bits as shown in Table 18-11 along with the appropriate PPAGE, DPAGE, or EPAGE signal from the core. **Note:** Comparator C can be used when the DBG module is configured for BKP mode. Extended addressing comparisons for comparator C use PAGSEL and will operate differently to the way that comparator A and B operate in BKP mode.

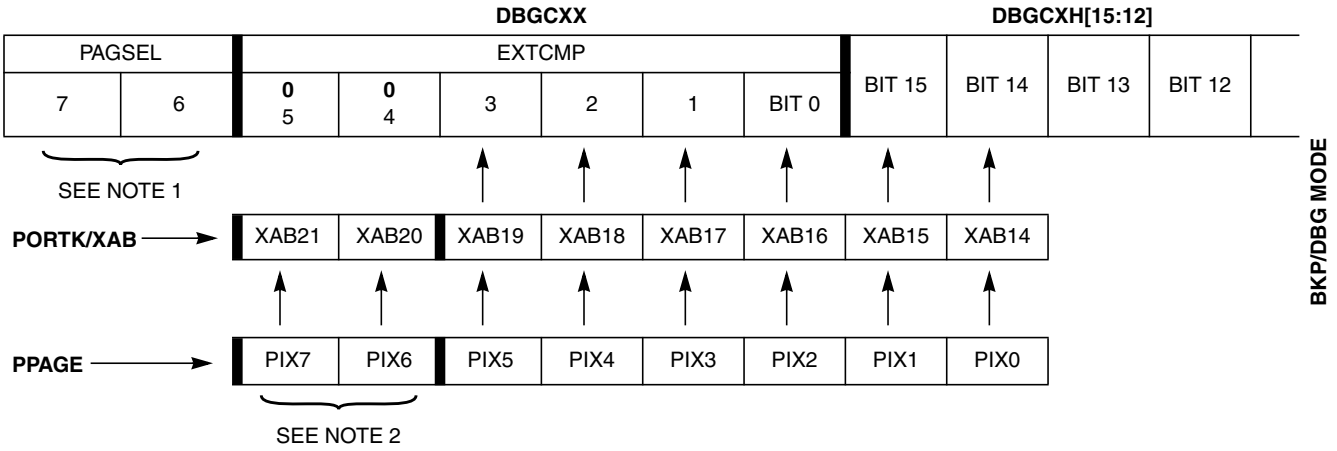
Table 18-11. PAGSEL Decoding[1]

PAGSEL	Description	EXTCMP	Comment
00	Normal (64k)	Not used	No paged memory
01	PPAGE (256 — 16K pages)	EXTCMP[5:0] is compared to address bits [21:16][2]	PPAGE[7:0] / XAB[21:14] becomes address bits [21:14][1]
10[3]	DPAGE (reserved) (256 — 4K pages)	EXTCMP[3:0] is compared to address bits [19:16]	DPAGE / XAB[21:14] becomes address bits [19:12]
11[2]	EPAGE (reserved) (256 — 1K pages)	EXTCMP[1:0] is compared to address bits [17:16]	EPAGE / XAB[21:14] becomes address bits [17:10]

1 See Figure 18-10.

2 Current HCS12 implementations have PPAGE limited to 6 bits. Therefore, EXTCMP[5:4] should be set to 00.

3 Data page (DPAGE) and Extra page (EPAGE) are reserved for implementation on devices that support paged data and extra space.



NOTES:
1. In BKP and DBG mode, PAGSEL selects the type of paging as shown in Table 18-11.
2. Current HCS12 implementations are limited to six PPAGE bits, PIX[5:0]. Therefore, EXTCMP[5:4] = 00.

Figure 18-10. Comparator C Extended Comparison in BKP/DBG Mode

18.3.2.6 Debug Comparator C Register (DBGCC)

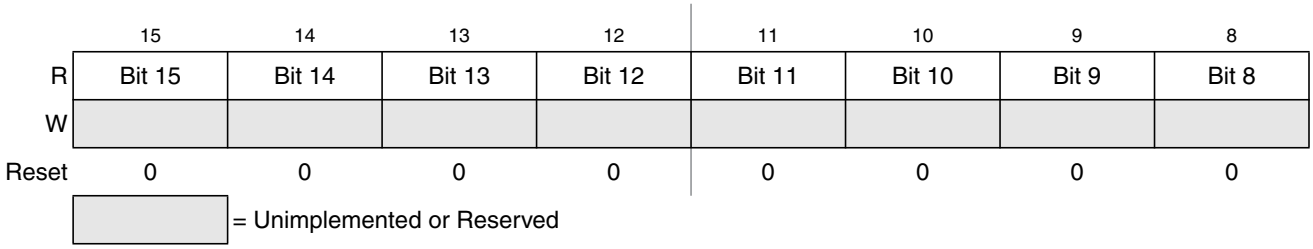


Figure 18-11. Debug Comparator C Register High (DBGCCH)

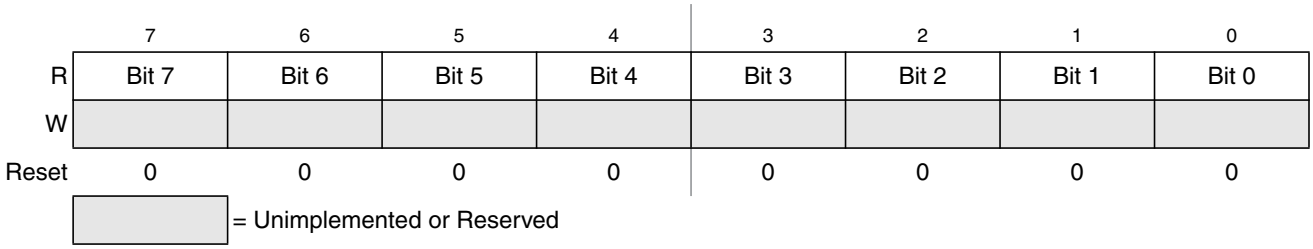


Figure 18-12. Debug Comparator C Register Low (DBGCCL)

Table 18-12. DBGCC Field Descriptions

Field	Description
15:0	**Comparator C Compare Bits** — The comparator C compare bits control whether comparator C will compare the address bus bits [15:0] to a logic 1 or logic 0. See Table 18-13. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1 **Note:** This register will be cleared automatically when the DBG module is armed in LOOP1 mode.

Table 18-13. Comparator C Compares

PAGSEL	EXTCMP Compare	High-Byte Compare
x0	No compare	DBGCCH[7:0] = AB[15:8]
x1	EXTCMP[5:0] = XAB[21:16]	DBGCCH[7:0] = XAB[15:14],AB[13:8]

18.3.2.7 Debug Control Register 2 (DBGC2)

	7	6	5	4	3	2	1	0
R W	BKABEN[1]	FULL	BDM	TAGAB	BKCEN[2]	TAGC[2]	RWCEN[2]	RWC[2]
Reset	0	0	0	0	0	0	0	0

1 When BKABEN is set (BKP mode), all bits in DBGC2 are available. When BKABEN is cleared and DBG is used in DBG mode, bits FULL and TAGAB have no meaning.

2 These bits can be used in BKP mode and DBG mode (when capture mode is not set in LOOP1) to provide a third breakpoint.

Figure 18-13. Debug Control Register 2 (DBGC2)

Table 18-14. DBGC2 Field Descriptions

Field	Description
7 BKABEN	**Breakpoint Using Comparator A and B Enable** — This bit enables the breakpoint capability using comparator A and B, when set (BKP mode) the DBGEN bit in DBGC1 cannot be set. 0 Breakpoint module off 1 Breakpoint module on
6 FULL	**Full Breakpoint Mode Enable** — This bit controls whether the breakpoint module is in dual mode or full mode. In full mode, comparator A is used to match address and comparator B is used to match data. See Section 18.4.1.2, “Full Breakpoint Mode,” for more details. 0 Dual address mode enabled 1 Full breakpoint mode enabled
5 BDM	**Background Debug Mode Enable** — This bit determines if the breakpoint causes the system to enter background debug mode (BDM) or initiate a software interrupt (SWI). 0 Go to software interrupt on a break request 1 Go to BDM on a break request
4 TAGAB	**Comparator A/B Tag Select** — This bit controls whether the breakpoint will cause a break on the next instruction boundary (force) or on a match that will be an executable opcode (tagged). Non-executed opcodes cannot cause a tagged breakpoint. 0 On match, break at the next instruction boundary (force) 1 On match, break if/when the instruction is about to be executed (tagged)
3 BKCEN	**Breakpoint Comparator C Enable Bit** — This bit enables the breakpoint capability using comparator C. 0 Comparator C disabled for breakpoint 1 Comparator C enabled for breakpoint **Note:** This bit will be cleared automatically when the DBG module is armed in loop1 mode.
2 TAGC	**Comparator C Tag Select** — This bit controls whether the breakpoint will cause a break on the next instruction boundary (force) or on a match that will be an executable opcode (tagged). Non-executed opcodes cannot cause a tagged breakpoint. 0 On match, break at the next instruction boundary (force) 1 On match, break if/when the instruction is about to be executed (tagged)

Table 18-14. DBGC2 Field Descriptions (continued)

Field	Description
1 RWCEN	**Read/Write Comparator C Enable Bit** — The RWCEN bit controls whether read or write comparison is enabled for comparator C. RWCEN is not useful for tagged breakpoints. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
0 RWC	**Read/Write Comparator C Value Bit** — The RWC bit controls whether read or write is used in compare for comparator C. The RWC bit is not used if RWCEN = 0. 0 Write cycle will be matched 1 Read cycle will be matched

18.3.2.8 Debug Control Register 3 (DBGC3)

	7	6	5	4	3	2	1	0
R W	BKAMBH[1]	BKAMBL[1]	BKBMBH[2]	BKBMBL[2]	RWAEN	RWA	RWBEN	RWB
Reset	0	0	0	0	0	0	0	0

1 In DBG mode, BKAMBH:BKAMBL has no meaning and are forced to 0's.

2 In DBG mode, BKBMBH:BKBMBL are used in full mode to qualify data.

Figure 18-14. Debug Control Register 3 (DBGC3)

Table 18-15. DBGC3 Field Descriptions

Field	Description
7:6 BKAMB[H:L]	**Breakpoint Mask High Byte for First Address** — In dual or full mode, these bits may be used to mask (disable) the comparison of the high and/or low bytes of the first address breakpoint. The functionality is as given in Table 18-16. The x:0 case is for a full address compare. When a program page is selected, the full address compare will be based on bits for a 20-bit compare. The registers used for the compare are {DBGCAX[5:0], DBGCAH[5:0], DBGCAL[7:0]}, where DBGAX[5:0] corresponds to PPAGE[5:0] or extended address bits [19:14] and CPU address [13:0]. When a program page is not selected, the full address compare will be based on bits for a 16-bit compare. The registers used for the compare are {DBGCAH[7:0], DBGCAL[7:0]} which corresponds to CPU address [15:0]. **Note:** This extended address compare scheme causes an aliasing problem in BKP mode in which several physical addresses may match with a single logical address. This problem may be avoided by using DBG mode to generate breakpoints. The 1:0 case is not sensible because it would ignore the high order address and compare the low order and expansion addresses. Logic forces this case to compare all address lines (effectively ignoring the BKAMBH control bit). The 1:1 case is useful for triggering a breakpoint on any access to a particular expansion page. This only makes sense if a program page is being accessed so that the breakpoint trigger will occur only if DBGCAX compares.

Table 18-15. DBGC3 Field Descriptions (continued)

Field	Description
5:4 BKBMB[H:L]	**Breakpoint Mask High Byte and Low Byte of Data (Second Address)** — In dual mode, these bits may be used to mask (disable) the comparison of the high and/or low bytes of the second address breakpoint. The functionality is as given in Table 18-17. The x:0 case is for a full address compare. When a program page is selected, the full address compare will be based on bits for a 20-bit compare. The registers used for the compare are {DBGCBX[5:0], DBGCBH[5:0], DBGCBL[7:0]} where DBGCBX[5:0] corresponds to PPAGE[5:0] or extended address bits [19:14] and CPU address [13:0]. When a program page is not selected, the full address compare will be based on bits for a 16-bit compare. The registers used for the compare are {DBGCBH[7:0], DBGCBL[7:0]} which corresponds to CPU address [15:0]. **Note:** This extended address compare scheme causes an aliasing problem in BKP mode in which several physical addresses may match with a single logical address. This problem may be avoided by using DBG mode to generate breakpoints. The 1:0 case is not sensible because it would ignore the high order address and compare the low order and expansion addresses. Logic forces this case to compare all address lines (effectively ignoring the BKBMBH control bit). The 1:1 case is useful for triggering a breakpoint on any access to a particular expansion page. This only makes sense if a program page is being accessed so that the breakpoint trigger will occur only if DBGCBX compares. In full mode, these bits may be used to mask (disable) the comparison of the high and/or low bytes of the data breakpoint. The functionality is as given in Table 18-18.
3 RWAEN	**Read/Write Comparator A Enable Bit** — The RWAEN bit controls whether read or write comparison is enabled for comparator A. See Section 18.4.2.1.1, “Read or Write Comparison,” for more information. This bit is not useful for tagged operations. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
2 RWA	**Read/Write Comparator A Value Bit** — The RWA bit controls whether read or write is used in compare for comparator A. The RWA bit is not used if RWAEN = 0. 0 Write cycle will be matched 1 Read cycle will be matched
1 RWBEN	**Read/Write Comparator B Enable Bit** — The RWBEN bit controls whether read or write comparison is enabled for comparator B. See Section 18.4.2.1.1, “Read or Write Comparison,” for more information. This bit is not useful for tagged operations. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
0 RWB	**Read/Write Comparator B Value Bit** — The RWB bit controls whether read or write is used in compare for comparator B. The RWB bit is not used if RWBEN = 0. 0 Write cycle will be matched 1 Read cycle will be matched **Note:** RWB and RWBEN are not used in full mode.

Table 18-16. Breakpoint Mask Bits for First Address

BKAMBH:BKAMBL	Address Compare	DBGCAX	DBGCAH	DBGCAL
x:0	Full address compare	Yes[1]	Yes	Yes
0:1	256 byte address range	Yes[1]	Yes	No
1:1	16K byte address range	Yes[1]	No	No

[1] If PPAGE is selected.

Table 18-17. Breakpoint Mask Bits for Second Address (Dual Mode)

BKBMBH:BKBMBL	Address Compare	DBGCBX	DBGCBH	DBGCBL
x:0	Full address compare	Yes[1]	Yes	Yes
0:1	256 byte address range	Yes[1]	Yes	No
1:1	16K byte address range	Yes[1]	No	No

1 If PPAGE is selected.

Table 18-18. Breakpoint Mask Bits for Data Breakpoints (Full Mode)

BKBMBH:BKBMBL	Data Compare	DBGCBX	DBGCBH	DBGCBL
0:0	High and low byte compare	No[1]	Yes	Yes
0:1	High byte	No[1]	Yes	No
1:0	Low byte	No[1]	No	Yes
1:1	No compare	No[1]	No	No

1 Expansion addresses for breakpoint B are not applicable in this mode.

18.3.2.9 Debug Comparator A Extended Register (DBGCAX)

	7	6	5	4	3	2	1	0
R W	PAGSEL		EXTCMP					
Reset	0	0	0	0	0	0	0	0

Figure 18-15. Debug Comparator A Extended Register (DBGCAX)

Table 18-19. DBGCAX Field Descriptions

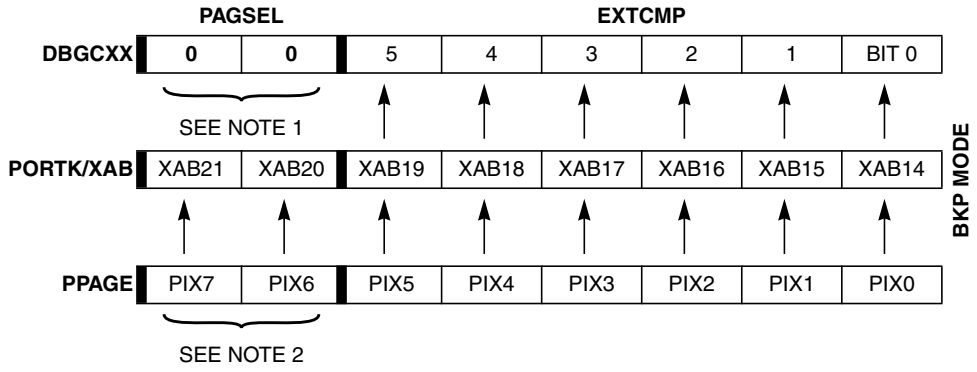
Field	Description
7:6 PAGSEL	**Page Selector Field** — If DBGEN is set in DBGC1, then PAGSEL selects the type of paging as shown in Table 18-20. DPAGE and EPAGE are not yet implemented so the value in bit 7 will be ignored (i.e., PAGSEL values of 10 and 11 will be interpreted as values of 00 and 01, respectively). In BKP mode, PAGSEL has no meaning and EXTCMP[5:0] are compared to address bits [19:14] if the address is in the FLASH/ROM memory space.
5:0 EXTCMP	**Comparator A Extended Compare Bits** — The EXTCMP bits are used as comparison address bits as shown in Table 18-20 along with the appropriate PPAGE, DPAGE, or EPAGE signal from the core.

Table 18-20. Comparator A or B Compares

Mode		EXTCMP Compare	High-Byte Compare
BKP[1]	Not FLASH/ROM access	No compare	DBGCxH[7:0] = AB[15:8]
	FLASH/ROM access	EXTCMP[5:0] = XAB[19:14]	DBGCxH[5:0] = AB[13:8]
DBG[2]	PAGSEL = 00	No compare	DBGCxH[7:0] = AB[15:8]
	PAGSEL = 01	EXTCMP[5:0] = XAB[21:16]	DBGCxH[7:0] = XAB[15:14], AB[13:8]

1 See Figure 18-16.

2 See Figure 18-10 (note that while this figure provides extended comparisons for comparator C, the figure also pertains to comparators A and B in DBG mode only).



NOTES:
1. In BKP mode, PAGSEL has no functionality. Therefore, set PAGSEL to 00 (reset state).
2. Current HCS12 implementations are limited to six PPAGE bits, PIX[5:0].

Figure 18-16. Comparators A and B Extended Comparison in BKP Mode

18.3.2.10 Debug Comparator A Register (DBGCA)

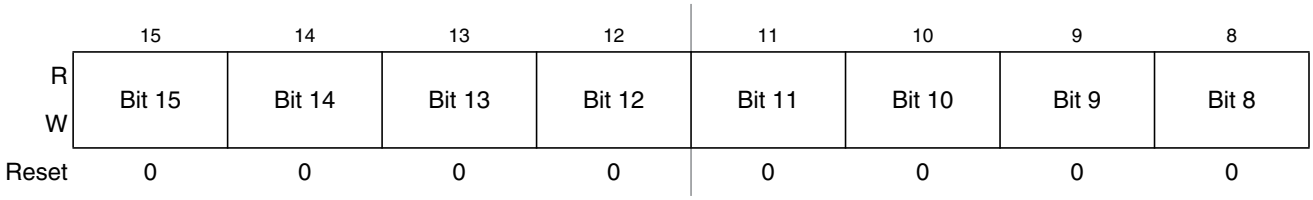


Figure 18-17. Debug Comparator A Register High (DBGCAH)

Figure 18-18. Debug Comparator A Register Low (DBGCAL)

Table 18-21. DBGCA Field Descriptions

Field	Description
15:0 15:0	**Comparator A Compare Bits** — The comparator A compare bits control whether comparator A compares the address bus bits [15:0] to a logic 1 or logic 0. See Table 18-20. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

18.3.2.11 Debug Comparator B Extended Register (DBGCBX)

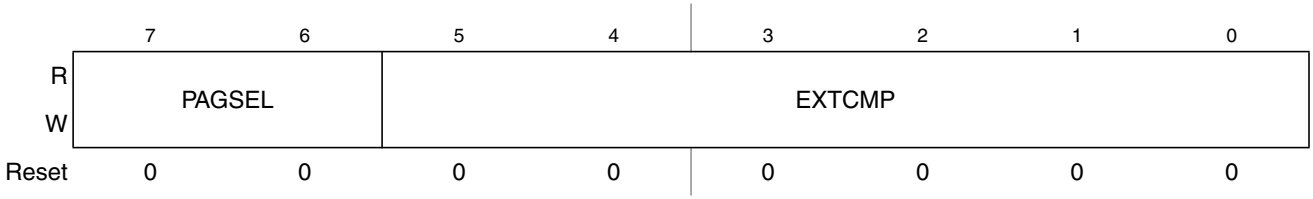


Figure 18-19. Debug Comparator B Extended Register (DBGCBX)

Table 18-22. DBGCBX Field Descriptions

Field	Description
7:6 PAGSEL	**Page Selector Field** — If DBGEN is set in DBGC1, then PAGSEL selects the type of paging as shown in Table 18-11. DPAGE and EPAGE are not yet implemented so the value in bit 7 will be ignored (i.e., PAGSEL values of 10 and 11 will be interpreted as values of 00 and 01, respectively.) In BKP mode, PAGSEL has no meaning and EXTCMP[5:0] are compared to address bits [19:14] if the address is in the FLASH/ROM memory space.
5:0 EXTCMP	**Comparator B Extended Compare Bits** — The EXTCMP bits are used as comparison address bits as shown in Table 18-11 along with the appropriate PPAGE, DPAGE, or EPAGE signal from the core. Also see Table 18-20.

18.3.2.12 Debug Comparator B Register (DBGCB)

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 18-20. Debug Comparator B Register High (DBGCBH)

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

Figure 18-21. Debug Comparator B Register Low (DBGCBL)

Table 18-23. DBGCB Field Descriptions

Field	Description
15:0 15:0	**Comparator B Compare Bits** — The comparator B compare bits control whether comparator B compares the address bus bits [15:0] or data bus bits [15:0] to a logic 1 or logic 0. See Table 18-20. 0 Compare corresponding address bit to a logic 0, compares to data if in Full mode 1 Compare corresponding address bit to a logic 1, compares to data if in Full mode

18.4 Functional Description

This section provides a complete functional description of the DBG module. The DBG module can be configured to run in either of two modes, BKP or DBG. BKP mode is enabled by setting BKABEN in DBGC2. DBG mode is enabled by setting DBGEN in DBGC1. Setting BKABEN in DBGC2 overrides the DBGEN in DBGC1 and prevents DBG mode. If the part is in secure mode, DBG mode cannot be enabled.

18.4.1 DBG Operating in BKP Mode

In BKP mode, the DBG will be fully backwards compatible with the existing BKP_ST12_A module. The DBGC2 register has four additional bits that were not available on existing BKP_ST12_A modules. As long as these bits are written to either all 1s or all 0s, they should be transparent to the user. All 1s would enable comparator C to be used as a breakpoint, but tagging would be enabled. The match address register would be all 0s if not modified by the user. Therefore, code executing at address 0x0000 would have to occur before a breakpoint based on comparator C would happen.

The DBG module in BKP mode supports two modes of operation: dual address mode and full breakpoint mode. Within each of these modes, forced or tagged breakpoint types can be used. Forced breakpoints occur at the next instruction boundary if a match occurs and tagged breakpoints allow for breaking just before the tagged instruction executes. The action taken upon a successful match can be to either place the CPU in background debug mode or to initiate a software interrupt.

The breakpoint can operate in dual address mode or full breakpoint mode. Each of these modes is discussed in the subsections below.

18.4.1.1 Dual Address Mode

When dual address mode is enabled, two address breakpoints can be set. Each breakpoint can cause the system to enter background debug mode or to initiate a software interrupt based upon the state of BDM in DBGC2 being logic 1 or logic 0, respectively. BDM requests have a higher priority than SWI requests. No data breakpoints are allowed in this mode.

TAGAB in DBGC2 selects whether the breakpoint mode is forced or tagged. The BKxMBH:L bits in DBGC3 select whether or not the breakpoint is matched exactly or is a range breakpoint. They also select whether the address is matched on the high byte, low byte, both bytes, and/or memory expansion. The RWx and RWxEN bits in DBGC3 select whether the type of bus cycle to match is a read, write, or read/write when performing forced breakpoints.

18.4.1.2 Full Breakpoint Mode

Full breakpoint mode requires a match on address and data for a breakpoint to occur. Upon a successful match, the system will enter background debug mode or initiate a software interrupt based upon the state of BDM in DBGC2 being logic 1 or logic 0, respectively. BDM requests have a higher priority than SWI requests. R/W matches are also allowed in this mode.

TAGAB in DBGC2 selects whether the breakpoint mode is forced or tagged. When TAGAB is set in DBGC2, only addresses are compared and data is ignored. The BKAMBH:L bits in DBGC3 select whether or not the breakpoint is matched exactly, is a range breakpoint, or is in page space. The BKBMBH:L bits in DBGC3 select whether the data is matched on the high byte, low byte, or both bytes. RWA and RWAEN bits in DBGC2 select whether the type of bus cycle to match is a read or a write when performing forced breakpoints. RWB and RWBEN bits in DBGC2 are not used in full breakpoint mode.

NOTE

The full trigger mode is designed to be used for either a word access or a byte access, but not both at the same time. Confusing trigger operation (seemingly false triggers or no trigger) can occur if the trigger address occurs in the user program as both byte and word accesses.

18.4.1.3 Breakpoint Priority

Breakpoint operation is first determined by the state of the BDM module. If the BDM module is already active, meaning the CPU is executing out of BDM firmware, breakpoints are not allowed. In addition, while executing a BDM TRACE command, tagging into BDM is not allowed. If BDM is not active, the breakpoint will give priority to BDM requests over SWI requests. This condition applies to both forced and tagged breakpoints.

In all cases, BDM related breakpoints will have priority over those generated by the Breakpoint sub-block. This priority includes breakpoints enabled by the $\overline{\text{TAGLO}}$ and $\overline{\text{TAGHI}}$ external pins of the system that interface with the BDM directly and whose signal information passes through and is used by the breakpoint sub-block.

NOTE

BDM should not be entered from a breakpoint unless the ENABLE bit is set in the BDM. Even if the ENABLE bit in the BDM is cleared, the CPU actually executes the BDM firmware code. It checks the ENABLE and returns if ENABLE is not set. If the BDM is not serviced by the monitor then the breakpoint would be re-asserted when the BDM returns to normal CPU flow.

There is no hardware to enforce restriction of breakpoint operation if the BDM is not enabled.

When program control returns from a tagged breakpoint through an RTI or a BDM GO command, it will return to the instruction whose tag generated the breakpoint. Unless breakpoints are disabled or modified in the service routine or active BDM session, the instruction will be tagged again and the breakpoint will be repeated. In the case of BDM breakpoints, this situation can also be avoided by executing a TRACE1 command before the GO to increment the program flow past the tagged instruction.

18.4.1.4 Using Comparator C in BKP Mode

The original BKP_ST12_A module supports two breakpoints. The DBG_ST12_A module can be used in BKP mode and allow a third breakpoint using comparator C. Four additional bits, BKCEN, TAGC, RWCEN, and RWC in DBGC2 in conjunction with additional comparator C address registers, DBGCCX, DBGCCH, and DBGCCL allow the user to set up a third breakpoint. Using PAGSEL in DBGCCX for expanded memory will work differently than the way paged memory is done using comparator A and B in BKP mode. See Section 18.3.2.5, “Debug Comparator C Extended Register (DBGCCX),” for more information on using comparator C.

18.4.2 DBG Operating in DBG Mode

Enabling the DBG module in DBG mode, allows the arming, triggering, and storing of data in the trace buffer and can be used to cause CPU breakpoints. The DBG module is made up of three main blocks, the comparators, trace buffer control logic, and the trace buffer.

NOTE

In general, there is a latency between the triggering event appearing on the bus and being detected by the DBG circuitry. In general, tagged triggers will be more predictable than forced triggers.

18.4.2.1 Comparators

The DBG contains three comparators, A, B, and C. Comparator A compares the core address bus with the address stored in DBGCAH and DBGCAL. Comparator B compares the core address bus with the address stored in DBGCBH and DBGCBL except in full mode, where it compares the data buses to the data stored in DBGCBH and DBGCBL. Comparator C can be used as a breakpoint generator or as the address comparison unit in the loop1 mode. Matches on comparator A, B, and C are signaled to the trace buffer

control (TBC) block. When PAGSEL = 01, registers DBGCAX, DBGCBX, and DBGCCX are used to match the upper addresses as shown in Table 18-11.

NOTE

If a tagged-type C breakpoint is set at the same address as an A/B tagged-type trigger (including the initial entry in an inside or outside range trigger), the C breakpoint will have priority and the trigger will not be recognized.

18.4.2.1.1 Read or Write Comparison

Read or write comparisons are useful only with TRGSEL = 0, because only opcodes should be tagged as they are “read” from memory. RWAEN and RWBEN are ignored when TRGSEL = 1.

In full modes (“A and B” and “A and not B”) RWAEN and RWA are used to select read or write comparisons for both comparators A and B. Table 18-24 shows the effect for RWAEN, RWA, and RW on the DBGCB comparison conditions. The RWBEN and RWB bits are not used and are ignored in full modes.

Table 18-24. Read or Write Comparison Logic Table

RWAEN bit	RWA bit	RW signal	Comment
0	x	0	Write data bus
0	x	1	Read data bus
1	0	0	Write data bus
1	0	1	No data bus compare since RW=1
1	1	0	No data bus compare since RW=0
1	1	1	Read data bus

18.4.2.1.2 Trigger Selection

The TRGSEL bit in DBGC1 is used to determine the triggering condition in DBG mode. TRGSEL applies to both trigger A and B except in the event only trigger modes. By setting TRGSEL, the comparators A and B will qualify a match with the output of opcode tracking logic and a trigger occurs before the tagged instruction executes (tagged-type trigger). With the TRGSEL bit cleared, a comparator match forces a trigger when the matching condition occurs (force-type trigger).

NOTE

If the TRGSEL is set, the address stored in the comparator match address registers must be an opcode address for the trigger to occur.

18.4.2.2 Trace Buffer Control (TBC)

The TBC is the main controller for the DBG module. Its function is to decide whether data should be stored in the trace buffer based on the trigger mode and the match signals from the comparator. The TBC also determines whether a request to break the CPU should occur.

18.4.2.3 Begin- and End-Trigger

The definitions of begin- and end-trigger as used in the DBG module are as follows:

- Begin-trigger: Storage in trace buffer occurs after the trigger and continues until 64 locations are filled.
- End-trigger: Storage in trace buffer occurs until the trigger, with the least recent data falling out of the trace buffer if more than 64 words are collected.

18.4.2.4 Arming the DBG Module

In DBG mode, arming occurs by setting DBGEN and ARM in DBGC1. The ARM bit in DBGC1 is cleared when the trigger condition is met in end-trigger mode or when the Trace Buffer is filled in begin-trigger mode. The TBC logic determines whether a trigger condition has been met based on the trigger mode and the trigger selection.

18.4.2.5 Trigger Modes

The DBG module supports nine trigger modes. The trigger modes are encoded as shown in Table 18-6. The trigger mode is used as a qualifier for either starting or ending the storing of data in the trace buffer. When the match condition is met, the appropriate flag A or B is set in DBGSC. Arming the DBG module clears the A, B, and C flags in DBGSC. In all trigger modes except for the event-only modes and DETAIL capture mode, change-of-flow addresses are stored in the trace buffer. In the event-only modes only the value on the data bus at the trigger event B will be stored. In DETAIL capture mode address and data for all cycles except program fetch (P) and free (f) cycles are stored in trace buffer.

18.4.2.5.1 A Only

In the A only trigger mode, if the match condition for A is met, the A flag in DBGSC is set and a trigger occurs.

18.4.2.5.2 A or B

In the A or B trigger mode, if the match condition for A or B is met, the corresponding flag in DBGSC is set and a trigger occurs.

18.4.2.5.3 A then B

In the A then B trigger mode, the match condition for A must be met before the match condition for B is compared. When the match condition for A or B is met, the corresponding flag in DBGSC is set. The trigger occurs only after A then B have matched.

NOTE

When tagging and using A then B, if addresses A and B are close together, then B may not complete the trigger sequence. This occurs when A and B are in the instruction queue at the same time. Basically the A trigger has not yet occurred, so the B instruction is not tagged. Generally, if address B is at

least six addresses higher than address A (or B is lower than A) and there are not changes of flow to put these in the queue at the same time, then this operation should trigger properly.

18.4.2.5.4 Event-Only B (Store Data)

In the event-only B trigger mode, if the match condition for B is met, the B flag in DBGSC is set and a trigger occurs. The event-only B trigger mode is considered a begin-trigger type and the BEGIN bit in DBGC1 is ignored. Event-only B is incompatible with instruction tagging (TRGSEL = 1), and thus the value of TRGSEL is ignored. Please refer to Section 18.4.2.7, “Storage Memory,” for more information.

This trigger mode is incompatible with the detail capture mode so the detail capture mode will have priority. TRGSEL and BEGIN will not be ignored and this trigger mode will behave as if it were “B only”.

18.4.2.5.5 A then Event-Only B (Store Data)

In the A then event-only B trigger mode, the match condition for A must be met before the match condition for B is compared, after the A match has occurred, a trigger occurs each time B matches. When the match condition for A or B is met, the corresponding flag in DBGSC is set. The A then event-only B trigger mode is considered a begin-trigger type and BEGIN in DBGC1 is ignored. TRGSEL in DBGC1 applies only to the match condition for A. Please refer to Section 18.4.2.7, “Storage Memory,” for more information.

This trigger mode is incompatible with the detail capture mode so the detail capture mode will have priority. TRGSEL and BEGIN will not be ignored and this trigger mode will be the same as A then B.

18.4.2.5.6 A and B (Full Mode)

In the A and B trigger mode, comparator A compares to the address bus and comparator B compares to the data bus. In the A and B trigger mode, if the match condition for A and B happen on the same bus cycle, both the A and B flags in the DBGSC register are set and a trigger occurs.

If TRGSEL = 1, only matches from comparator A are used to determine if the trigger condition is met and comparator B matches are ignored. If TRGSEL = 0, full-word data matches on an odd address boundary (misaligned access) do not work unless the access is to a RAM that manages misaligned accesses in a single clock cycle (which is typical of RAM modules used in HCS12 MCUs).

18.4.2.5.7 A and Not B (Full Mode)

In the A and not B trigger mode, comparator A compares to the address bus and comparator B compares to the data bus. In the A and not B trigger mode, if the match condition for A and not B happen on the same bus cycle, both the A and B flags in DBGSC are set and a trigger occurs.

If TRGSEL = 1, only matches from comparator A are used to determine if the trigger condition is met and comparator B matches are ignored. As described in Section 18.4.2.5.6, “A and B (Full Mode),” full-word data compares on misaligned accesses will not match expected data (and thus will cause a trigger in this mode) unless the access is to a RAM that manages misaligned accesses in a single clock cycle.

18.4.2.5.8 Inside Range (A ≤ address ≤ B)

In the inside range trigger mode, if the match condition for A and B happen on the same bus cycle, both the A and B flags in DBGSC are set and a trigger occurs. If a match condition on only A or only B occurs no flags are set. If TRGSEL = 1, the inside range is accurate only to word boundaries. If TRGSEL = 0, an aligned word access which straddles the range boundary will cause a trigger only if the aligned address is within the range.

18.4.2.5.9 Outside Range (address < A or address > B)

In the outside range trigger mode, if the match condition for A or B is met, the corresponding flag in DBGSC is set and a trigger occurs. If TRGSEL = 1, the outside range is accurate only to word boundaries. If TRGSEL = 0, an aligned word access which straddles the range boundary will cause a trigger only if the aligned address is outside the range.

18.4.2.5.10 Control Bit Priorities

The definitions of some of the control bits are incompatible with each other. Table 18-25 and the notes associated with it summarize how these incompatibilities are managed:

- Read/write comparisons are not compatible with TRGSEL = 1. Therefore, RWAEN and RWBEN are ignored.
- Event-only trigger modes are always considered a begin-type trigger. See Section 18.4.2.8.1, “Storing with Begin-Trigger,” and Section 18.4.2.8.2, “Storing with End-Trigger.”
- Detail capture mode has priority over the event-only trigger/capture modes. Therefore, event-only modes have no meaning in detail mode and their functions default to similar trigger modes.

Table 18-25. Resolution of Mode Conflicts

Mode	Normal / Loop1		Detail	
	Tag	Force	Tag	Force
A only				
A or B				
A then B				
Event-only B	1		1, 3	3
A then event-only B	2		4	4
A and B (full mode)	5		5	
A and not B (full mode)	5		5	
Inside range	6		6	
Outside range	6		6	

1 — Ignored — same as force
2 — Ignored for comparator B
3 — Reduces to effectively “B only”
4 — Works same as A then B
5 — Reduces to effectively “A only” — B not compared
6 — Only accurate to word boundaries

18.4.2.6 Capture Modes

The DBG in DBG mode can operate in four capture modes. These modes are described in the following subsections.

18.4.2.6.1 Normal Mode

In normal mode, the DBG module uses comparator A and B as triggering devices. Change-of-flow information or data will be stored depending on TRG in DBGSC.

18.4.2.6.2 Loop1 Mode

The intent of loop1 mode is to prevent the trace buffer from being filled entirely with duplicate information from a looping construct such as delays using the DBNE instruction or polling loops using BRSET/BRCLR instructions. Immediately after address information is placed in the trace buffer, the DBG module writes this value into the C comparator and the C comparator is placed in ignore address mode. This will prevent duplicate address entries in the trace buffer resulting from repeated bit-conditional branches. Comparator C will be cleared when the ARM bit is set in loop1 mode to prevent the previous contents of the register from interfering with loop1 mode operation. Breakpoints based on comparator C are disabled.

Loop1 mode only inhibits duplicate source address entries that would typically be stored in most tight looping constructs. It will not inhibit repeated entries of destination addresses or vector addresses, because repeated entries of these would most likely indicate a bug in the user's code that the DBG module is designed to help find.

NOTE

In certain very tight loops, the source address will have already been fetched again before the C comparator is updated. This results in the source address being stored twice before further duplicate entries are suppressed. This condition occurs with branch-on-bit instructions when the branch is fetched by the first P-cycle of the branch or with loop-construct instructions in which the branch is fetched with the first or second P cycle. See examples below:

```
LOOP   INCX                    ; 1-byte instruction fetched by 1st P-cycle of BRCLR
       BRCLR CMPTMP,#$0c,LOOP  ; the BRCLR instruction also will be fetched by 1st P-cycle of BRCLR

LOOP2  BRN   *                 ; 2-byte instruction fetched by 1st P-cycle of DBNE
       NOP                     ; 1-byte instruction fetched by 2nd P-cycle of DBNE
       DBNE  A,LOOP2           ; this instruction also fetched by 2nd P-cycle of DBNE
```

NOTE

Loop1 mode does not support paged memory, and inhibits duplicate entries in the trace buffer based solely on the CPU address. There is a remote possibility of an erroneous address match if program flow alternates between paged and unpaged memory space.

18.4.2.6.3 Detail Mode

In the detail mode, address and data for all cycles except program fetch (P) and free (f) cycles are stored in trace buffer. This mode is intended to supply additional information on indexed, indirect addressing modes where storing only the destination address would not provide all information required for a user to determine where his code was in error.

18.4.2.6.4 Profile Mode

This mode is intended to allow a host computer to poll a running target and provide a histogram of program execution. Each read of the trace buffer address will return the address of the last instruction executed. The DBGCNT register is not incremented and the trace buffer does not get filled. The ARM bit is not used and all breakpoints and all other debug functions will be disabled.

18.4.2.7 Storage Memory

The storage memory is a 64 words deep by 16-bits wide dual port RAM array. The CPU accesses the RAM array through a single memory location window (DBGTBH:DBGTBL). The DBG module stores trace information in the RAM array in a circular buffer format. As data is read via the CPU, a pointer into the RAM will increment so that the next CPU read will receive fresh information. In all trigger modes except for event-only and detail capture mode, the data stored in the trace buffer will be change-of-flow addresses. change-of-flow addresses are defined as follows:

- Source address of conditional branches (long, short, BRSET, and loop constructs) taken
- Destination address of indexed JMP, JSR, and CALL instruction
- Destination address of RTI, RTS, and RTC instructions
- Vector address of interrupts except for SWI and BDM vectors

In the event-only trigger modes only the 16-bit data bus value corresponding to the event is stored. In the detail capture mode, address and then data are stored for all cycles except program fetch (P) and free (f) cycles.

18.4.2.8 Storing Data in Memory Storage Buffer

18.4.2.8.1 Storing with Begin-Trigger

Storing with begin-trigger can be used in all trigger modes. When DBG mode is enabled and armed in the begin-trigger mode, data is not stored in the trace buffer until the trigger condition is met. As soon as the trigger condition is met, the DBG module will remain armed until 64 words are stored in the trace buffer. If the trigger is at the address of the change-of-flow instruction the change-of-flow associated with the trigger event will be stored in the trace buffer.

18.4.2.8.2 Storing with End-Trigger

Storing with end-trigger cannot be used in event-only trigger modes. When DBG mode is enabled and armed in the end-trigger mode, data is stored in the trace buffer until the trigger condition is met. When the trigger condition is met, the DBG module will become de-armed and no more data will be stored. If

the trigger is at the address of a change-of-flow address the trigger event will not be stored in the trace buffer.

18.4.2.9 Reading Data from Trace Buffer

The data stored in the trace buffer can be read using either the background debug module (BDM) module or the CPU provided the DBG module is enabled and not armed. The trace buffer data is read out first-in first-out. By reading CNT in DBGCNT the number of valid words can be determined. CNT will not decrement as data is read from DBGTBH:DBGTBL. The trace buffer data is read by reading DBGTBH:DBGTBL with a 16-bit read. Each time DBGTBH:DBGTBL is read, a pointer in the DBG will be incremented to allow reading of the next word.

Reading the trace buffer while the DBG module is armed will return invalid data and no shifting of the RAM pointer will occur.

NOTE

The trace buffer should be read with the DBG module enabled and in the same capture mode that the data was recorded. The contents of the trace buffer counter register (DBGCNT) are resolved differently in detail mode verses the other modes and may lead to incorrect interpretation of the trace buffer data.

18.4.3 Breakpoints

There are two ways of getting a breakpoint in DBG mode. One is based on the trigger condition of the trigger mode using comparator A and/or B, and the other is using comparator C. External breakpoints generated using the $\overline{\text{TAGHI}}$ and $\overline{\text{TAGLO}}$ external pins are disabled in DBG mode.

18.4.3.1 Breakpoint Based on Comparator A and B

A breakpoint request to the CPU can be enabled by setting DBGBRK in DBGC1. The value of BEGIN in DBGC1 determines when the breakpoint request to the CPU will occur. When BEGIN in DBGC1 is set, begin-trigger is selected and the breakpoint request will not occur until the trace buffer is filled with 64 words. When BEGIN in DBGC1 is cleared, end-trigger is selected and the breakpoint request will occur immediately at the trigger cycle.

There are two types of breakpoint requests supported by the DBG module, tagged and forced. Tagged breakpoints are associated with opcode addresses and allow breaking just before a specific instruction executes. Forced breakpoints are not associated with opcode addresses and allow breaking at the next instruction boundary. The type of breakpoint based on comparators A and B is determined by TRGSEL in the DBGC1 register (TRGSEL = 1 for tagged breakpoint, TRGSEL = 0 for forced breakpoint). Table 18-26 illustrates the type of breakpoint that will occur based on the debug run.

Table 18-26. Breakpoint Setup

BEGIN	TRGSEL	DBGBRK	Type of Debug Run
0	0	0	Fill trace buffer until trigger address (no CPU breakpoint — keep running)
0	0	1	Fill trace buffer until trigger address, then a forced breakpoint request occurs
0	1	0	Fill trace buffer until trigger opcode is about to execute (no CPU breakpoint — keep running)
0	1	1	Fill trace buffer until trigger opcode about to execute, then a tagged breakpoint request occurs
1	0	0	Start trace buffer at trigger address (no CPU breakpoint — keep running)
1	0	1	Start trace buffer at trigger address, a forced breakpoint request occurs when trace buffer is full
1	1	0	Start trace buffer at trigger opcode (no CPU breakpoint — keep running)
1	1	1	Start trace buffer at trigger opcode, a forced breakpoint request occurs when trace buffer is full

18.4.3.2 Breakpoint Based on Comparator C

A breakpoint request to the CPU can be created if BKCEN in DBGC2 is set. Breakpoints based on a successful comparator C match can be accomplished regardless of the mode of operation for comparator A or B, and do not affect the status of the ARM bit. TAGC in DBGC2 is used to select either tagged or forced breakpoint requests for comparator C. Breakpoints based on comparator C are disabled in LOOP1 mode.

NOTE

Because breakpoints cannot be disabled when the DBG is armed, one must be careful to avoid an “infinite breakpoint loop” when using tagged-type C breakpoints while the DBG is armed. If BDM breakpoints are selected, executing a TRACE1 instruction before the GO instruction is the recommended way to avoid re-triggering a breakpoint if one does not wish to de-arm the DBG. If SWI breakpoints are selected, disarming the DBG in the SWI interrupt service routine is the recommended way to avoid re-triggering a breakpoint.

18.5 Resets

The DBG module is disabled after reset.

The DBG module cannot cause a MCU reset.

18.6 Interrupts

The DBG contains one interrupt source. If a breakpoint is requested and BDM in DBGC2 is cleared, an SWI interrupt will be generated.

Appendix A
Electrical Characteristics

NOTE

The MC9S12NE64 is specified and tested at the 3.3-V range.

This section contains the most accurate electrical information for the MC9S12NE64 microcontroller available at the time of publication. The information is subject to change.

This introduction is intended to give an overview on several common topics like power supply, current injection etc.

A.1 Parameter Classification

The electrical parameters shown in this supplement are guaranteed by various methods. To give the customer a better understanding the following classification is used and the parameters are tagged accordingly in the tables where appropriate, under the “C” column heading.

P: Those parameters are guaranteed during production testing on each individual device.

C: Those parameters are achieved by the design characterization by measuring a statistically relevant sample size across process variations. They are regularly verified by production monitors.

T: Those parameters are achieved by design characterization on a small sample size from typical devices. All values shown in the typical column are within this category.

D: Those parameters are derived mainly from simulations.

A.2 Power Supply

The MC9S12NE64 uses several pins to supply power to the I/O ports, A/D converter, oscillator and PLL, the Ethernet Physical Transceiver (EPHY), as well as the digital core.

- The V_{DDA}, V_{SSA} pair supplies the A/D converter and portions of the EPHY
- The V_{DDX1}, V_{DDX2}, V_{SSX1}, V_{SSX2} pairs supply the I/O pins, and internal voltage regulator
- The V_{DDR} supplies the internal voltage regulator, and is the V_{REGEN} signal
- V_{DD1}, V_{SS1}, V_{DD2}, and V_{SS2} are the supply pins for the digital logic
- V_{DDPLL}, V_{SSPLL} supply the oscillator and the PLL
- V_{SS1} and V_{SS2} are internally connected by metal
- V_{DD1} and V_{DD2} are internally connected by metal
- PHY_VDDA, PHY_VSSA are power supply pins for EPHY analog
- PHY_VDDRX, PHY_VSSRX are power supply pins for EPHY receiver

- PHY_VDDTX, PHY_VSSTX are power supply pins for EPHY transmitter
- V_{DDA}, V_{DDX1}, V_{DDX2} as well as V_{SSA}, V_{SSX1}, V_{SSX2} are connected by anti-parallel diodes for ESD protection.

NOTE

In the following context:

- V_{DD3} is used for either V_{DDA}, V_{DDR}, and V_{DDX1}/V_{DDX2}
- V_{SS3} is used for either V_{SSA}, V_{SSR}, and V_{SSX1}/V_{SSX2} unless otherwise noted
- I_{DD3} denotes the sum of the currents flowing into the V_{DDA}, V_{DDR}, and V_{DDX1}/V_{DDX2} pins
- V_{DD} is used for V_{DD1}, V_{DD2}, V_{DDPLL}, PHY_VDDTX, PHY_VDDRX, and PHY_VDDA
- V_{SS} is used for V_{SS1}, V_{SS2}, V_{SSPL}, PHY_VSSTX, PHY_VSSRX, and PHY_VSSA
- I_{DD} is used for the sum of the currents flowing into V_{DD1}, V_{DD2}
- I_{DDPHY} is used for the sum of currents flowing into PHY_VDDTX, PHY_VDDRX, and PHY_VDDA
- V_{DDPHY} is used for PHY_VDDTX, PHY_VDDRX, and PHY_VDDA
- V_{DDTX} is used for twisted pair differential voltage present on the PHY_TXP and PHY_TXN pins
- I_{DDTX} is used for twisted pair differential current flowing into the PHY_TXP or PHY_TXN pins

A.3 Pins

There are four groups of functional pins.

A.3.1 3.3 V I/O Pins

These I/O pins have a nominal level of 3.3 V. This group of pins is comprised of all port I/O pins, the analog inputs, BKGD pin and the $\overline{\text{RESET}}$ inputs. The internal structure of these pins are identical, however some of the functionality may be disabled.

A.3.2 Analog Reference, Special Function Analog

This group of pins is comprised of the V_{RH}, V_{RL}, PHY_TXN, PHY_TXP, PHY_RXN, PHY_RXP, and R_{BIAS} pins.

A.3.3 Oscillator

The pins XFC, EXTAL, XTAL dedicated to the oscillator have a nominal 2.5 V level. They are supplied by V_{DDPLL}.

A.3.4 TEST

This pin is used for production testing only, and should be tied to ground during normal operation.

A.4 Current Injection

Power supply must maintain regulation within operating V_{DD3} or V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{in} > V_{DD3}$) is greater than I_{DD3}, the injection current may flow out of V_{DD3} and could result in external power supply going out of regulation. Ensure external V_{DD3} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power; e.g., if no system clock is present, or if clock rate is very low, which would reduce overall power consumption.

A.5 Absolute Maximum Ratings

Absolute maximum ratings are stress ratings only. A functional operation under or outside those maxima is not guaranteed. Stress beyond those limits may affect the reliability or cause permanent damage of the device.

This device contains circuitry protecting against damage due to high static voltage or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS3} or V_{DD3}).

Table A-1. Absolute Maximum Ratings

Num	Rating	Symbol	Min	Max	Unit
1	I/O, Regulator and Analog Supply Voltage	V_{DD3}	–0.3	4.5	V
2	Digital Logic Supply Voltage [1]	V_{DD}	–0.3	3.0	V
3	PLL Supply Voltage [1]	V_{DDPLL}	–0.3	3.0	V
4	Voltage difference V_{DDX} to V_{DDR} and V_{DDA}	Δ_{VDDX}	–0.3	0.3	V
5	Voltage difference V_{SSX} to V_{SSR} and V_{SSA}	Δ_{VSSX}	–0.3	0.3	V
6	Digital I/O Input Voltage	V_{IN}	–0.3	6.5	V
7	Analog Reference	V_{RH}, V_{RL}	–0.3	6.5	V
8	XFC, EXTAL, XTAL inputs	V_{ILV}	–0.3	3.0	V
9	TEST input	V_{TEST}	–0.3	10.0	V
10	Instantaneous Maximum Current Single pin limit for all digital I/O pins [2]	I_D	–25	+25	mA
11	Instantaneous Maximum Current Single pin limit for XFC, EXTAL, XTAL [3]	I_{DL}	–25	+25	mA
12	Instantaneous Maximum Current Single pin limit for TEST [4]	I_{DT}	–0.25	0	mA
13	Operating Temperature Range (ambient)	T_A	–40	105 [5]	°C
14	Operating Temperature Range (junction)	T_J	–40	140	°C
15	Storage Temperature Range	T_{stg}	–65	155	°C

1 The device contains an internal voltage regulator to generate the logic and PLL supply out of the I/O supply. The absolute maximum ratings apply when the device is powered from an external source.

2 All digital I/O pins are internally clamped to V_{SSX} and V_{DDX}, V_{DDR} or V_{SSA} and V_{DDA}.

3 These pins are internally clamped to V_{SSPLL} and V_{DDPLL}.

4 This pin is clamped low to V_{SSPLL}, but not clamped high. This pin must be tied low in applications.

5 Maximum ambient temperature is package dependent.

A.6 ESD Protection and Latch-Up Immunity

All ESD testing is in conformity with CDF-AEC-Q100 stress test qualification for automotive grade integrated circuits. During the device qualification ESD stresses were performed for the human body model (HBM), the machine model (MM), and the charge device model.

A device will be defined as a failure if after exposure to ESD pulses the device no longer meets the device specification. Complete DC parametric and functional testing is performed per the applicable device specification at room temperature followed by hot temperature, unless specified otherwise in the device specification.

Table A-2. ESD and Latch-Up Test Conditions

Model	Description	Symbol	Value	Unit
Human Body	Series Resistance	R1	1500	Ω
	Storage Capacitance	C	100	pF
	Number of Pulse per pin positive negative	—	— 3 3	
Machine	Series Resistance	R1	0	Ω
	Storage Capacitance	C	200	pF
	Number of Pulse per pin positive negative	—	— 3 3	
Latch-up	Minimum input voltage limit		–2.5	V
	Maximum input voltage limit		7.5	V

Table A-3. ESD and Latch-Up Protection Characteristics

Num	C	Rating	Symbol	Min	Max	Unit
1	C	Human Body Model (HBM)	V_{HBM}	2000	—	V
		Only PHY_TXP, PHY_TXN, PHY_RXP, PHY_RXN pins		1000	—	V
2	C	Machine Model (MM)	V_{MM}	200	—	V
		Only PHY_TXP, PHY_TXN, PHY_RXP, PHY_RXN pins		100	—	V
3	C	Charge Device Model (CDM)	V_{CDM}	500	—	V
		Only PHY_TXP, PHY_TXN, PHY_RXP, PHY_RXN pins		250	—	V
4	C	Latch-up Current at 125°C positive negative	I_{LAT}	+100 –100	—	mA
5	C	Latch-up Current at 27°C positive negative	I_{LAT}	+200 –200	—	mA

A.7 Operating Conditions

This section describes the operating conditions of the device. Unless otherwise noted those conditions apply to all the following data.

NOTE

Instead of specifying ambient temperature, all parameters are specified for the more meaningful silicon junction temperature. For power dissipation calculations refer to Section A.8, “Power Dissipation and Thermal Characteristics.”

Table A-4. Operating Conditions

Rating	Symbol	Min	Typ	Max	Unit
I/O and Regulator Supply Voltage	V_{DDX}	3.135	3.3	3.465	V
Analog Supply Voltage	V_{DDA}	3.135	3.3	3.465	V
Regulator Supply Voltage	V_{DDR}	3.135	3.3	3.465	V
Digital Logic Supply Voltage [1]	V_{DD}	2.375	2.5	2.625	V
PLL Supply Voltage [1]	V_{DDPLL}	2.375	2.5	2.625	V
Voltage Difference V_{DDX1}/V_{SSX2} to V_{DDA}	Δ_{VDDX}	–0.1	0	0.1	V
Voltage Difference V_{SSX}/V_{SSX2} to V_{SSA}	Δ_{VSSX}	–0.1	0	0.1	V
Oscillator [2]	f_{osc}	0.5	—	25	MHz
Bus Frequency	f_{bus}	0.5	—	25	MHz
Operating Junction Temperature Range	T_J	–40	—	125	°C

1 The device contains an internal voltage regulator to generate V_{DD1}, V_{DD2}, V_{DDPLL}, PHY_VDDRX, PHY_VDDTX and PHY_VDDA supplies out of the V_{DDX} and V_{DDR} supply. The absolute maximum ratings apply when this regulator is disabled and the device is powered from an external source.

2 For the internal Ethernet physical transceiver (EPHY) to operate properly a 25 MHz oscillator is required.

A.8 Power Dissipation and Thermal Characteristics

Power dissipation and thermal characteristics are closely related. The user must assure that the maximum operating junction temperature is not exceeded. The average chip-junction temperature (T_J) in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \Theta_{JA})$$

T_J = Junction Temperature, [°C]

T_A = Ambient Temperature, [°C]

P_D = Total Chip Power Dissipation, [W]

Θ_{JA} = Package Thermal Resistance, [°C/W]

The total power dissipation can be calculated from:

$$P_D = P_{INT} + P_{IO}$$

P_{INT} = Chip Internal Power Dissipation, [W]

Two cases with internal voltage regulator enabled and disabled must be considered:

1. Internal Voltage Regulator disabled

$$P_{INT} = I_{DD} \cdot V_{DD} + I_{DDPLL} \cdot V_{DDPLL} + I_{DDA} \cdot V_{DDA} + I_{DDPHY} \cdot V_{DDPHY} + I_{DDTX} \cdot V_{DDTX}$$

$$P_{IO} = \sum_i R_{DSON} \cdot I_{IO_i}^2$$

Which is the sum of all output currents on I/O ports associated with V_{DDX}.

For R_{DSON} is valid:

$$R_{DSON} = \frac{V_{OL}}{I_{OL}}; \text{for outputs driven low}$$

respectively

$$R_{DSON} = \frac{V_{DD3(5)} - V_{OH}}{I_{OH}}; \text{for outputs driven high}$$

2. Internal voltage regulator enabled

$$P_{INT} = I_{DDR} \cdot V_{DDR} + I_{DDA} \cdot V_{DDA} + I_{DDX} \cdot V_{DDX} + I_{DDTX} \cdot V_{DDTX}$$

I_{DDX} is the current shown in Table A-7 and not the overall current flowing into V_{DDX}, which additionally contains the current flowing into the external loads with output high.

$$P_{IO} = \sum_i R_{DSON} \cdot I_{IO_i}^2$$

Which is the sum of all output currents on I/O ports associated with V_{DDX}.

Table A-5. Thermal Package Characteristics [1]

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	T	Thermal Resistance LQFP112, single sided PCB [2]	θ_{JA}	—	—	54	°C/W
2	T	Thermal Resistance LQFP112, double sided PCB with two internal planes [3]	θ_{JA}	—	—	41	°C/W
3	T	Junction to Board LQFP112	θ_{JB}	—	—	31	°C/W
4	T	Junction to Case LQFP112	θ_{JC}	—	—	11	°C/W
5	T	Junction to Package Top LQFP112	Ψ_{JT}	—	—	2	°C/W
6	T	Thermal Resistance TQFP-EP80, single sided PCB	θ_{JA}	—	—	51	°C/W
7	T	Thermal Resistance TQFP-EP80, double sided PCB with two internal planes	θ_{JA}	—	—	41	°C/W
8	T	Junction to Board TQFP-EP80	θ_{JB}	—	—	27	°C/W
9	T	Junction to Case TQFP-EP80	θ_{JC}	—	—	14	°C/W
10	T	Junction to Package Top TQFP-EP80	Ψ_{JT}	—	—	3	°C/W
6	T	Thermal Resistance Epad TQFP-EP80, single sided PCB	θ_{JA}	—	—	48	°C/W
7	T	Thermal Resistance Epad TQFP-EP80, double sided PCB with two internal planes	θ_{JA}	—	—	24	°C/W
8	T	Junction to Board TQFP-EP80	θ_{JB}	—	—	10	°C/W
9	T	Junction to Case TQFP-EP80 [4]	θ_{JC}	—	—	0.7	°C/W
10	T	Junction to Package Top TQFP-EP80	Ψ_{JT}	—	—	2	°C/W

1 The values for thermal resistance are achieved by package simulations

2 PC Board according to EIA/JEDEC Standard 51-3

3 PC Board according to EIA/JEDEC Standard 51-7

4 Thermal resistance between the die and the exposed die pad.

A.9 I/O Characteristics

This section describes the characteristics of all 3.3 V I/O pins. All parameters are not always applicable; e.g., not all pins feature pullup/pulldown resistances.

Table A-6. Preliminary 3.3 V I/O Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	Input High Voltage	V_{IH}	$0.65*V_{DD3}$	—	—	V
	T	Input High Voltage	V_{IH}	—	—	$V_{DD3} + 0.3$	V
2	P	Input Low Voltage	V_{IL}	—	—	$0.35*V_{DD3}$	V
	T	Input Low Voltage	V_{IL}	$V_{SS3} - 0.3$	—	—	V
3	C	Input Hysteresis	V_{HYS}		250		mV
4	P	Input Leakage Current (pins in high ohmic input mode) [1] $V_{in} = V_{DD5}$ or V_{SS5}	I_{in}	–2.5	—	2.5	µA
5	C	Output High Voltage (pins in output mode) Partial Drive I_{OH} = –0.75 mA	V_{OH}	$V_{DD3} - 0.4$	—	—	V
6	P	Output High Voltage (pins in output mode) Full Drive I_{OH} = –4.5 mA	V_{OH}	$V_{DD3} - 0.4$	—	—	V
7	C	Output Low Voltage (pins in output mode) Partial Drive I_{OL} = +0.9 mA	V_{OL}	—	—	0.4	V
8	P	Output Low Voltage (pins in output mode) Full Drive I_{OL} = +5.5 mA	V_{OL}	—	—	0.4	V
9	P	Internal Pull Up Device Current, tested at V_{IL} Max.	I_{PUL}	—	—	–60	µA
10	C	Internal Pull Up Device Current, tested at V_{IH} Min.	I_{PUH}	–6	—	—	µA
11	P	Internal Pull Down Device Current, tested at V_{IH} Min.	I_{PDH}	—	—	60	µA
12	C	Internal Pull Down Device Current, tested at V_{IL} Max.	I_{PDL}	6	—	—	µA
13	D	Input Capacitance	C_{in}		7	—	pF
14	T	Injection current [2] Single Pin limit Total Device Limit. Sum of all injected currents	I_{ICS} I_{ICP}	–2.5 –25	—	2.5 25	µA
15	P	Port G, H, and J Interrupt Input Pulse filtered [3]	t_{PIGN}			3	µs
16	P	Port G, H, and J Interrupt Input Pulse passed[3]	t_{PVAL}	10			µs

[1] Maximum leakage current occurs at maximum operating temperature. Current decreases by approximately one-half for each 8°C to 12°C in the temperature range from 50°C to 125°C.

[2] Refer to Section A.4, “Current Injection,” for more details.

[3] Parameter only applies in stop or pseudo stop mode.

A.10 Supply Currents

This section describes the current consumption characteristics of the device as well as the conditions for the measurements.

A.10.1 Measurement Conditions

All measurements are without output loads. Unless otherwise noted the currents are measured in single chip mode, internal voltage regulator enabled and at 25 MHz bus frequency using a 25 MHz oscillator.

A.10.2 Additional Remarks

In expanded modes the currents flowing in the system are highly dependent on the load at the address, data and control signals as well as on the duty cycle of those signals. No generally applicable numbers can be given. A very good estimate is to take the single chip currents and add the currents due to the external loads.

Table A-7. Supply Current Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P C P C	Run supply currents Single chip, internal regulator enabled, EPHY disabled Single chip, internal regulator enabled, EPHY auto negotiate [1] Single chip, internal regulator enabled, EPHY 100BASE-TX[1] Single chip, internal regulator enabled, EPHY 10BASE-T[1]	I_{DD3}			 65 285 265 185	mA
2	C C P	Wait supply current All modules enabled All modules but EPHY enabled only RTI enabled	I_{DDW}			270 50 5	mA
3	C P C C	Pseudo stop current (RTI and COP enabled) –40°C 27°C 85°C 105°C	I_{DDPS}		600 600 1000 1000	 750 5000	µA
4	C C C C	Pseudo stop current (RTI and COP disabled) –40°C 27°C 85°C 105°C	I_{DDPS}		160 160 700 700	 400 5000	µA
5	C P C C	Stop current –40°C 27°C 85°C 105°C	I_{DDS}		60 60 400 500	 200 5000	µA

[1] When calculating power consumption, the additional current sunk by the PHY_TXN and PHY_TXP pins must be taken into account. See Table A-8 for currents and voltages to use in the power calculations.

Table A-8. EPHY Twisted Pair Transmit Pin Characteristics

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	C	Auto-negotiate transmitter current	I_{DDTX}		130		mA
2	C	Auto-negotiate transmitter voltage	V_{DDTX}			V_{DD3} – 1.1 V	V
3	C	10BASE-T mode transmitter current	I_{DDTX}		130		mA
4	C	10BASE-T mode transmitter voltage	V_{DDTX}			V_{DD3} – 1.1 V	V
5	C	100BASE-TX mode transmitter current	I_{DDTX}		45		mA
6	C	100BASE-TX mode transmitter voltage	V_{DDTX}			V_{DD3} – 0.95 V	V

A.11 ATD Electrical Characteristics

This section describes the characteristics of the analog-to-digital converter.

A.11.1 ATD Operating Characteristics — 3.3 V Range

Table 18-27 shows conditions under which the ATD operates.

The following constraints exist to obtain full-scale, full range results:
$V_{SSA} \le V_{RL} \le V_{IN} \le V_{RH} \le V_{DDA}$. This constraint exists since the sample buffer amplifier can not drive beyond the power supply levels that it ties to. If the input level goes outside of this range it will effectively be clipped.

Table 18-27. 3.3V ATD Operating Characteristics

Conditions are shown in Table A-4 unless otherwise noted; Supply Voltage 3.3 V-10% <= V_{DDA} <= 3.3 V +10%							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	Reference Potential Low High	 V_{RL} V_{RH}	 V_{SSA} $V_{DDA}/2$		 $V_{DDA}/2$ V_{DDA}	 V V
2	C	Differential Reference Voltage	V_{RH}-V_{RL}	3.0	3.3	3.6	V
3	D	ATD Clock Frequency	f_{ATDCLK}	0.5		2.0	MHz
4	D	ATD 10-Bit Conversion Period Clock Cycles [1] Conv, Time at 2.0 MHz ATD Clock f_{ATDCLK}	 N_{CONV10} T_{CONV10}	 14 7		 28 14	 Cycles µs
5	D	ATD 8-Bit Conversion Period Clock Cycles[1] Conv, Time at 2.0 MHz ATD Clock f_{ATDCLK}	 N_{CONV8} T_{CONV8}	 12 6		 26 13	 Cycles µs
6	D	Recovery Time (V_{DDA} = 3.3 V)	t_{REC}			20	µs
7	P	Reference Supply current	I_{REF}			0.250	mA

[1] The minimum time assumes a final sample period of 2 ATD clocks cycles while the maximum time assumes a final sample period of 16 ATD clocks.

A.11.2 Factors Influencing Accuracy

Three factors — source resistance, source capacitance, and current injection — have an influence on the accuracy of the ATD.

A.11.2.1 Source Resistance

Due to the input pin leakage current as specified in Table A-6 in conjunction with the source resistance there will be a voltage drop from the signal source to the ATD input. The maximum source resistance R_S specifies results in an error of less than 1/2 LSB (2.5 mV) at the maximum leakage current. If device or operating conditions are less than worst case or leakage-induced error is acceptable, larger values of source resistance are allowed.

A.11.2.2 Source Capacitance

When sampling an additional internal capacitor is switched to the input. This can cause a voltage drop due to charge sharing with the external and the pin capacitance. For a maximum sampling error of the input voltage ≤ 1LSB, then the external filter capacitor, $C_f \geq 1024 * (C_{INS} - C_{INN})$.

A.11.2.3 Current Injection

There are two cases to consider.

- A current is injected into the channel being converted. The channel being stressed has conversion values of $3FF ($FF in 8-bit mode) for analog inputs greater than VRH and $000 for values less than VRL unless the current is higher than specified as disruptive conditions.
- Current is injected into pins in the neighborhood of the channel being converted. A portion of this current is picked up by the channel (coupling ratio K), This additional current impacts the accuracy of the conversion depending on the source resistance.
  The additional input voltage error on the converted channel can be calculated as $V_{ERR} = K * R_S * I_{INJ}$, with I_{INJ} being the sum of the currents injected into the two pins adjacent to the converted channel.

Table A-9. ATD Electrical Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	C	Max input Source Resistance	R_S	—	—	1	kΩ
2	T	Total Input Capacitance Non Sampling Sampling	 C_{INN} C_{INS}			 10 15	pF
3	C	Disruptive Analog Input Current	I_{NA}	–2.5		2.5	mA
4	C	Coupling Ratio positive current injection	K_p			10^{-4}	A/A
5	C	Coupling Ratio negative current injection	K_n			10^{-2}	A/A

A.11.3 ATD Accuracy — 3.3 V Range

Table A-10 specifies the ATD conversion performance excluding any errors due to current injection, input capacitance and source resistance.

Table A-10. 3.3-V A/D Conversion Performance

Conditions are shown in Table A-4 unless otherwise noted $V_{REF} = V_{RH} - V_{RL}$ = 3.328 V. Resulting to one 8 bit count = 13 mV and one 10 bit count = 3.25 mV f_{ATDCLK} = 2.0 MHz							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	10-Bit Resolution	LSB		3.25		mV
2	P	10-Bit Differential Nonlinearity	DNL	–1.5		1.5	Counts
3	P	10-Bit Integral Nonlinearity	INL	–3.5	±1.5	3.5	Counts
4	P	10-Bit Absolute Error [1]	AE	–5	±2.5	5	Counts
5	C	10-Bit Absolute Error at f_{ATDCLK}= 4MHz	AE		±7.0		Counts
6	P	8-Bit Resolution	LSB		13		mV
7	P	8-Bit Differential Nonlinearity	DNL	–0.5		0.5	Counts
8	P	8-Bit Integral Nonlinearity	INL	–1.5	±1.0	1.5	Counts
9	P	8-Bit Absolute Error[1]	AE	–2.0	±1.5	2.0	Counts

[1] These values include the quantization error which is inherently 1/2 count for any A/D converter.

For the following definitions see also Figure A-1.

Differential non-linearity (DNL) is defined as the difference between two adjacent switching steps.

$$DNL(i) = \frac{V_i - V_{i-1}}{1LSB} - 1$$

The integral non-linearity (INL) is defined as the sum of all DNLs:

$$INL(n) = \sum_{i=1}^{n} DNL(i) = \frac{V_n - V_0}{1LSB} - n$$

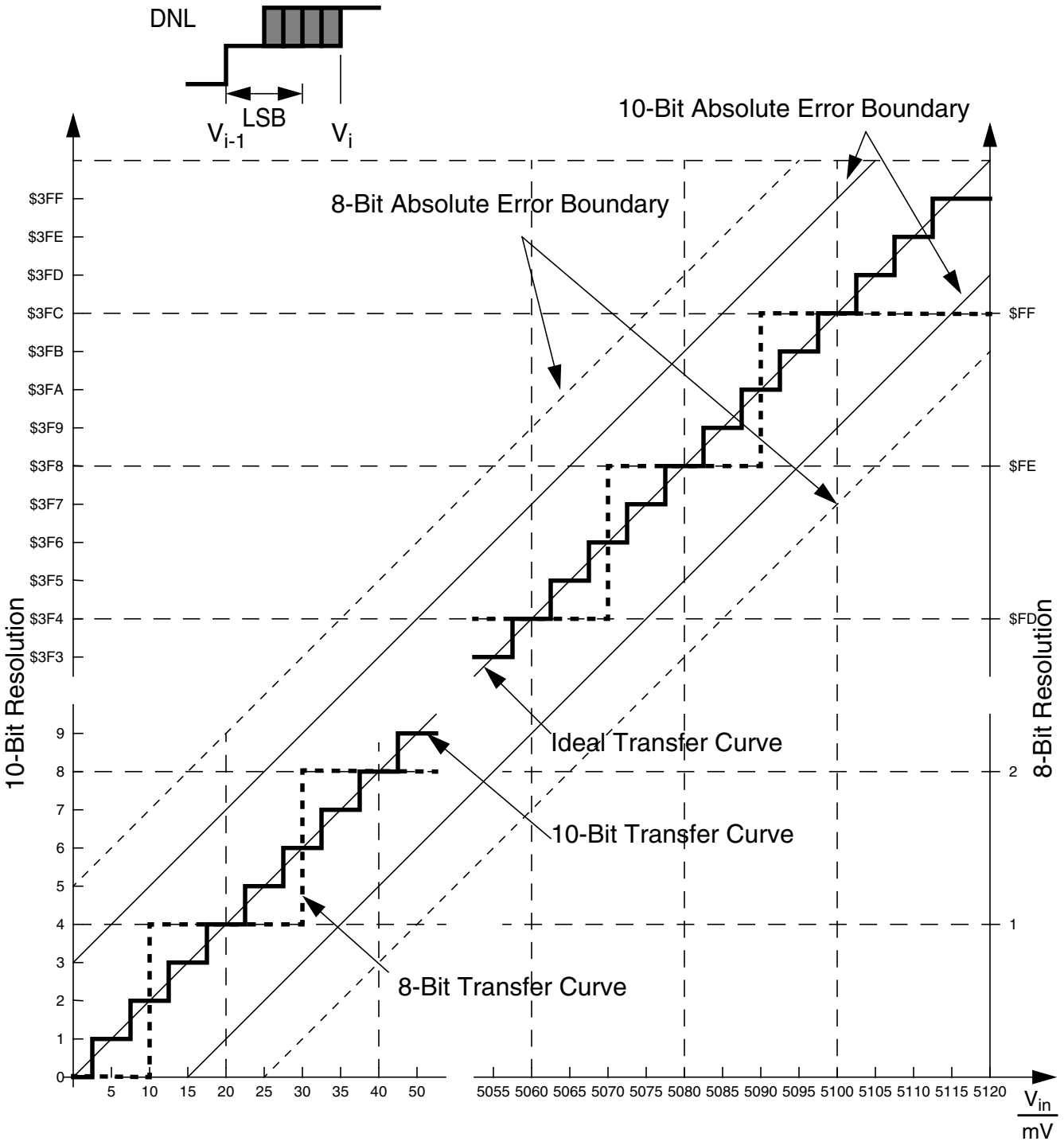


Figure A-1. ATD Accuracy Definitions

NOTE

Figure A-1 shows only definitions, for specification values refer to Table A-10.

A.12 Reset, Oscillator, and PLL Electrical Characteristics

This section summarizes the electrical characteristics of the various startup scenarios for oscillator and phase-locked loop (PLL).

A.12.1 Startup

Table A-11 summarizes several startup characteristics explained in this section. Detailed description of the startup behavior can be found in the clock and reset generator (CRG) block description chapter.

Table A-11. Startup Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	T	POR release level	V_{PORR}			2.07	V
2	T	POR assert level	V_{PORA}	0.97			V
3	D	Reset input pulse width, minimum input time	PW_{RSTL}	2			t_{osc}
4	D	Startup from Reset	n_{RST}	192		196	n_{osc}
5	D	Interrupt pulse width, $\overline{IRQ}$ edge-sensitive mode	PW_{IRQ}	20			ns
6	D	Wait recovery startup time	t_{WRS}			14	t_{cyc}
7	P	LVR release level	V_{LVRR}	2.25			V
8	P	LVR assert level	V_{LVRA}			2.55	V

A.12.1.1 POR

The release level V_{PORR} and the assert level V_{PORA} are derived from the V_{DD} supply. They are also valid if the device is powered externally. After releasing the POR reset the oscillator and the clock quality check are started. If after a time t_{CQOUT} no valid oscillation is detected, the MCU will start using the internal self clock. The fastest startup time possible is given by n_{uposc}.

A.12.1.2 LVR

The release level V_{LVRR} and the assert level V_{LVRA} are derived from the V_{DD} supply. They are also valid if the device is powered externally. After releasing the LVR reset the oscillator and the clock quality check are started. If after a time t_{CQOUT} no valid oscillation is detected, the MCU will start using the internal self clock. The fastest startup time possible is given by n_{uposc}.

A.12.1.3 SRAM Data Retention

Provided an appropriate external reset signal is applied to the MCU, preventing the CPU from executing code when V_{DD5} is out of specification limits, the SRAM contents integrity is guaranteed if after the reset the PORF bit in the CRG flags register has not been set.

A.12.1.4 External Reset

When external reset is asserted for a time greater than PW_{RSTL} the CRG module generates an internal reset, and the CPU starts fetching the reset vector without doing a clock quality check, if there was an oscillation before reset.

A.12.1.5 Stop Recovery

Out of stop, the controller can be woken up by an external interrupt. A clock quality check as after POR is performed before releasing the clocks to the system.

A.12.1.6 Pseudo Stop and Wait Recovery

The recovery from pseudo stop and wait are essentially the same because the oscillator was not stopped in either mode. The controller can be woken up by internal or external interrupts. After t_{wrs} the CPU starts fetching the interrupt vector.

A.12.2 Oscillator

The device features an internal Pierce oscillator. Before asserting the oscillator to the internal system clocks the quality of the oscillation is checked for each start from either power-on, STOP or oscillator fail. t_{CQOUT} specifies the maximum time before switching to the internal self clock mode after POR or STOP if a proper oscillation is not detected. The quality check also determines the minimum oscillator start-up time t_{UPOSC}. The device also features a clock monitor. A clock monitor failure is asserted if the frequency of the incoming clock signal is below the assert frequency f_{CMFA}.

Table A-12. Oscillator Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	C	Crystal oscillator range (Pierce) [1, 2]	f_{OSC}	0.5		40	MHz
2	P	Startup current	i_{OSC}	100			µA
3	C	Oscillator start-up time (Pierce)	t_{UPOSC}		8 [3]	100 [4]	ms
4	D	Clock quality check time-out	t_{CQOUT}	0.45		2.5	s
5	P	Clock monitor failure assert frequency	f_{CMFA}	50	100	200	kHz
6	P	External square wave input frequency [2]	f_{EXT}	0.5		50	MHz
7	D	External square wave pulse width low	t_{EXTL}	9.5			ns
8	D	External square wave pulse width high	t_{EXTH}	9.5			ns
9	D	External square wave rise time	t_{EXTR}			1	ns
10	D	External square wave fall time	t_{EXTF}			1	ns
11	D	Input capacitance (EXTAL, XTAL pins)	C_{IN}		7		pF
12	C	DC operating bias in Colpitts configuration on EXTAL pin	V_{DCBIAS}		1.1		V
		EXTAL pin input high voltage[4]	$V_{IH,EXTAL}$	$0.7*V_{DDPLL}$			V
		EXTAL pin input high voltage[4]	$V_{IH,EXTAL}$			V_{DDPLL}+ 0.3	V
		EXTAL pin input low voltage[4]	$V_{IL,EXTAL}$			$0.3*V_{DDPLL}$	V
		EXTAL pin input low voltage[4]	$V_{IL,EXTAL}$	V_{SSPLL}– 0.3			V
		EXTAL pin input hysteresis[4]	$V_{HYS,EXTAL}$		250		mV

1 Depending on the crystal a damping series resistor might be necessary

2 $\overline{\text{XCLKS}}$ =0 during reset

3 f_{osc} = 25 MHz, C = 22 pF.

4 Maximum value is for extreme cases using high Q, low frequency crystals

A.12.3 Phase-Locked Loop

The oscillator provides the reference clock for the PLL. The PLL´s voltage controlled oscillator (VCO) is also the system clock source in self clock mode.

A.12.3.1 XFC Component Selection

This section describes the selection of the XFC components to achieve a good filter characteristics.

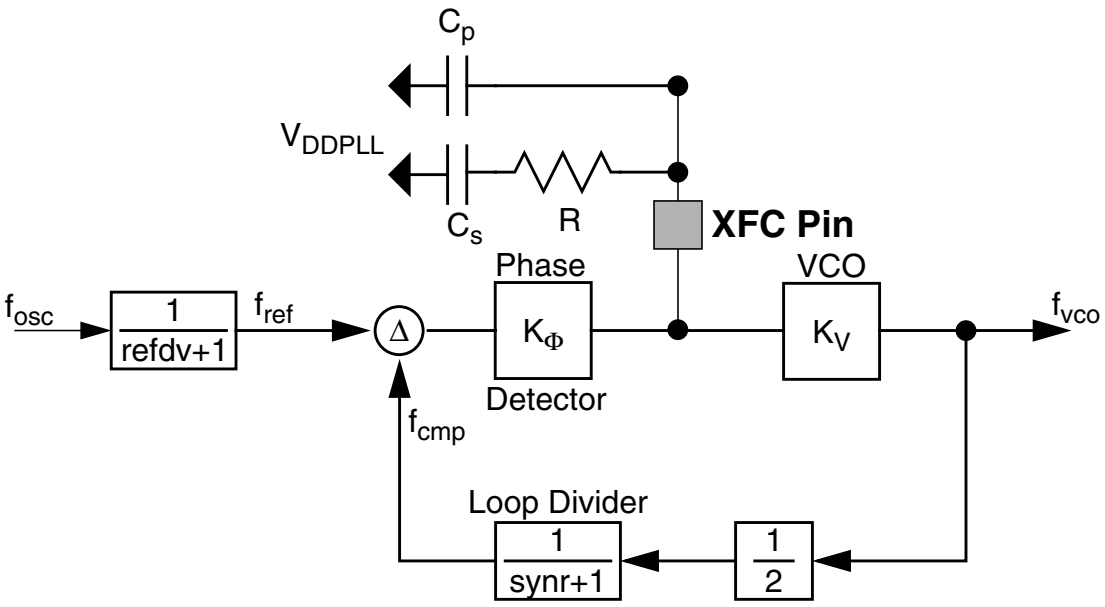


Figure 18-22. Basic PLL Functional Diagram

The following procedure can be used to calculate the resistance and capacitance values using typical values for K_1, f_1 and i_{ch} from Table A-13.

The grey boxes show the calculation for $f_{VCO} = 50$ MHz and $f_{ref} = 1$ MHz. E.g., these frequencies are used for $f_{OSC} = 4$ MHz and a 25 MHz bus clock.

The VCO Gain at the desired VCO frequency is approximated by:

$$K_V = K_1 \cdot e^{\frac{(f_1 - f_{vco})}{K_1 \cdot 1V}} = -100 \cdot e^{\frac{(60-50)}{-100}} = -90.48\text{MHz/V}$$

The phase detector relationship is given by:

$$K_\Phi = -|i_{ch}| \cdot K_V \qquad = 316.7\text{Hz}/\Omega$$

i_{ch} is the current in tracking mode.

The loop bandwidth f_C should be chosen to fulfill the Gardner's stability criteria by at least a factor of 10, typical values are 50. $\zeta = 0.9$ ensures a good transient response.

$$f_C < \frac{2 \cdot \zeta \cdot f_{ref}}{\pi \cdot \left(\zeta + \sqrt{1 + \zeta^2}\right)} \quad \frac{1}{10} \rightarrow f_C < \frac{f_{ref}}{4 \cdot 10}; (\zeta = 0.9)$$

$$f_C < 25\text{kHz}$$

And finally the frequency relationship is defined as:

$$n = \frac{f_{VCO}}{f_{ref}} = 2 \cdot (synr + 1) \quad = 50$$

With the above values the resistance can be calculated. The example is shown for a loop bandwidth f_C = 10 kHz:

$$R = \frac{2 \cdot \pi \cdot n \cdot f_C}{K_\Phi} \quad = 2*\pi*50*10kHz/(316.7Hz/\Omega)=9.9k\Omega=\sim 10k\Omega$$

The capacitance C_s can now be calculated as:

$$C_s = \frac{2 \cdot \zeta^2}{\pi \cdot f_C \cdot R} \approx \frac{0.516}{f_C \cdot R};(\zeta = 0.9) \quad = 5.19nF =\sim 4.7nF$$

The capacitance C_p should be chosen in the range of:

$$C_s/20 \le C_p \le C_s/10 \quad C_p = 470pF$$

A.12.3.1.1 Jitter Information

The basic functionality of the PLL is shown in Figure 18-22. With each transition of the clock f_{cmp}, the deviation from the reference clock f_{ref} is measured and input voltage to the VCO is adjusted accordingly. The adjustment is done continuously with no abrupt changes in the clock output frequency. Noise, voltage, temperature, and other factors cause slight variations in the control loop resulting in a clock jitter. This jitter affects the real minimum and maximum clock periods as illustrated in Figure 18-23.

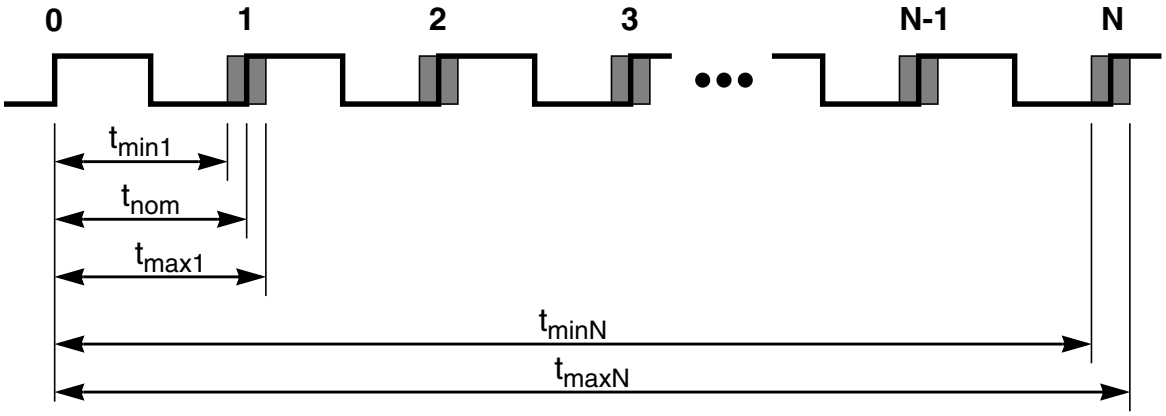


Figure 18-23. Jitter Definitions

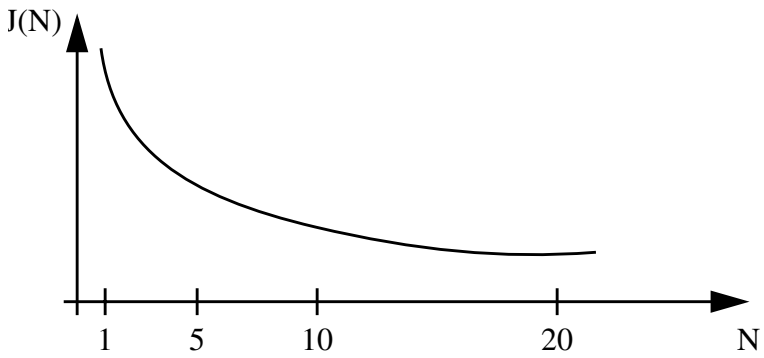
The relative deviation of t_{nom} is at its maximum for one clock period, and decreases towards zero for larger number of clock periods (N).

Defining the jitter as:

$$J(N) = \max\left(\left|1 - \frac{t_{max}(N)}{N \cdot t_{nom}}\right|, \left|1 - \frac{t_{min}(N)}{N \cdot t_{nom}}\right|\right)$$

For N < 100, the following equation is a good fit for the maximum jitter:

$$J(N) = \frac{j_1}{\sqrt{N}} + j_2$$



This is very important to notice with respect to timers, serial modules where a prescaler will eliminate the effect of the jitter to a large extent.

Table A-13. PLL Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	Self clock mode frequency	f_{SCM}	1		5.5	MHz
2	D	VCO locking range	f_{VCO}	8		50	MHz
3	D	Lock detector transition from acquisition to tracking mode	$\|\Delta_{trk}\|$	3		4	% [1]
4	D	Lock detection	$\|\Delta_{Lock}\|$	0		1.5	%[1]
5	D	Unlock detection	$\|\Delta_{unl}\|$	0.5		2.5	%[1]
6	D	Lock detector transition from tracking to acquisition mode	$\|\Delta_{unt}\|$	6		8	%[1]
7	C	PLLON total stabilization delay (auto mode) [2]	t_{stab}		0.5		ms
8	D	PLLON acquisition mode stabilization delay [2]	t_{acq}		0.3		ms
9	D	PLLON tracking mode stabilization delay [2]	t_{al}		0.2		ms
10	D	Fitting parameter VCO loop gain	K_1		–100		MHz/V
11	D	Fitting parameter VCO loop frequency	f_1		60		MHz
12	D	Charge pump current acquisition mode	$\| i_{ch} \|$		38.5		µA
13	D	Charge pump current tracking mode	$\| i_{ch} \|$		3.5		µA
14	C	Jitter fit parameter 1[2]	j_1			1.1	%
15	C	Jitter fit parameter 2[2]	j_2			0.13	%

1 % deviation from target frequency

2 f_{REF} = 25 MHz, f_{BUS} = 25 MHz equivalent f_{VCO} = 50 MHz: REFDV = #$00, SYNR = #$00, Cs = 4700 pF, Cp = 470 pF, Rs = 2.2 kΩ.

A.13 EMAC Electrical Characteristics

NOTE

The electrical characteristics given in the EMAC section are preliminary and should be used as a guide only. Values cannot be guaranteed by Freescale Semiconductor and are subject to change without notice.

A.13.1 MII Timing

The following MII timing is based on IEEE Std 802.3.

A.13.1.1 MII Receive Signal Timing (MII_RXD[3:0], MII_RXDV, MII_RXER, MII_RXCLK)

Table A-14. MII Receive Signal Timing

Num	Characteristic	Min	Max	Unit
M1	RXD[3:0], RXDV, RXER setup to RXCLK rise	10	—	ns
M2	RXCLK rise to RXD[3:0], RXDV, RXER hold	10	—	ns
M3	RXCLK pulse width high	35%	65%	RXCLK period
M4	RXCLK pulse width low	35%	65%	RXCLK period

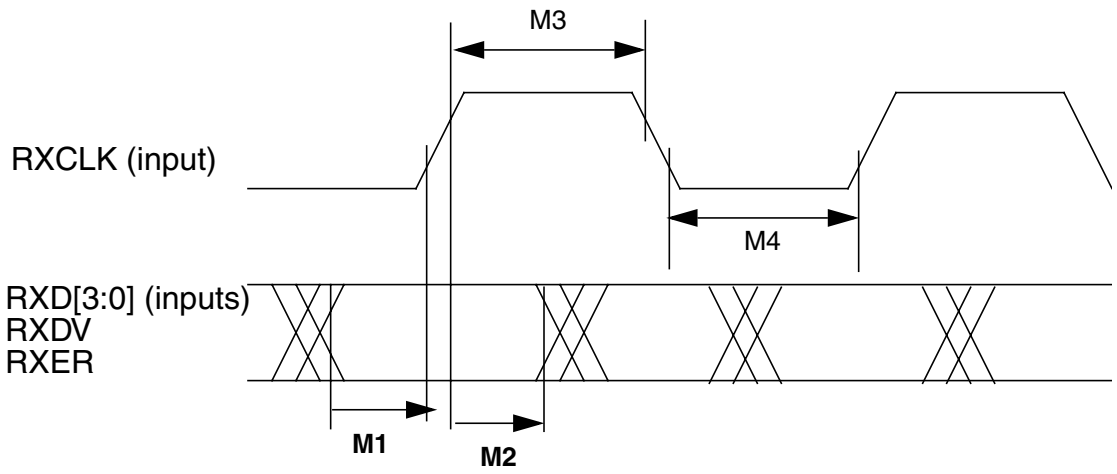


Figure 18-24. MII Receive Signal Timing Diagram

A.13.1.2 MII Transmit Signal Timing (TXD[3:0], TXEN, TXER, TXCLK)

Table A-15. MII Transmit Signal Timing

Num	Characteristic	Min	Max	Unit
M5	TXCLK rise to TXD[3:0], TXEN, TXER invalid	0	—	ns
M6	TXCLK rise to TXD[3:0], TXEN, TXER valid	—	25	ns
M7	TXCLK pulse width high	35%	65%	TXCLK period
M8	TXCLK pulse width low	35%	65%	TXCLK period

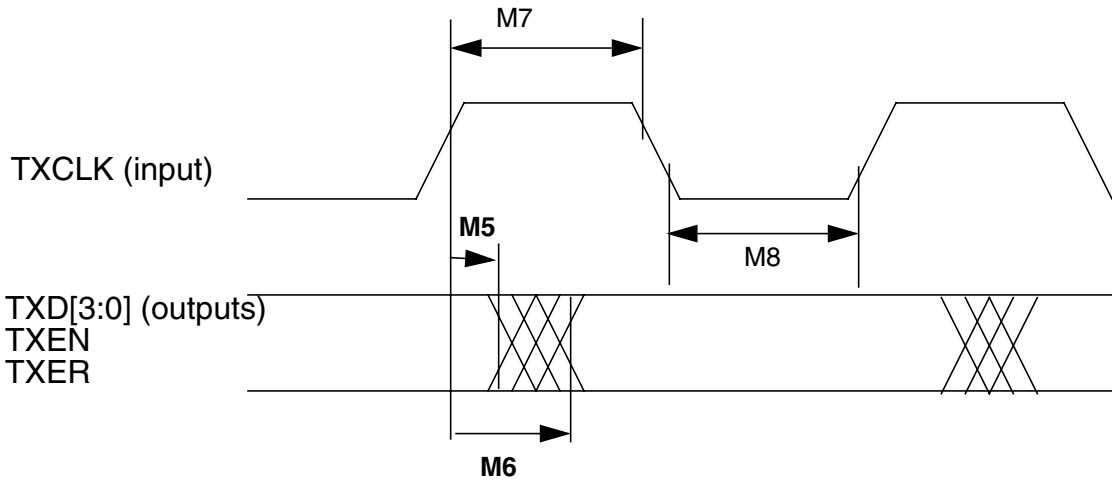


Figure A-2. MII Transmit Signal Timing Diagram

A.13.1.3 MII Asynchronous Inputs Signal Timing (CRS, COL)

Table A-16. MII Transmit Signal Timing

Num	Characteristic	Min	Max	Unit
M9	CRS, COL minimum pulse width	1.5	—	TXCLK period

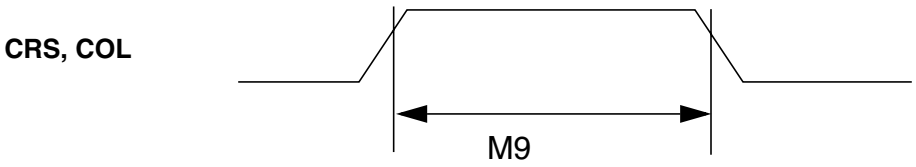


Figure A-3. MII Asynchronous Inputs Timing Diagram

A.13.1.4 MII Management Timing (MDIO, MDC)

Table A-17. MII Management Signal Timing

Num	Characteristic	Min	Max	Unit
M10	MDC rise to MDIO (output) invalid	10	—	ns
M11	MDC rise to MDIO (output) valid	—	390	ns
M12	MDIO (input) setup to MDC rise	100	—	ns
M13	MDC rise to MDIO (input) hold	0	—	ns
M14	MDC pulse width high	40%	60%	MDC period
M15	MDC pulse width low	40%	60%	MDC period

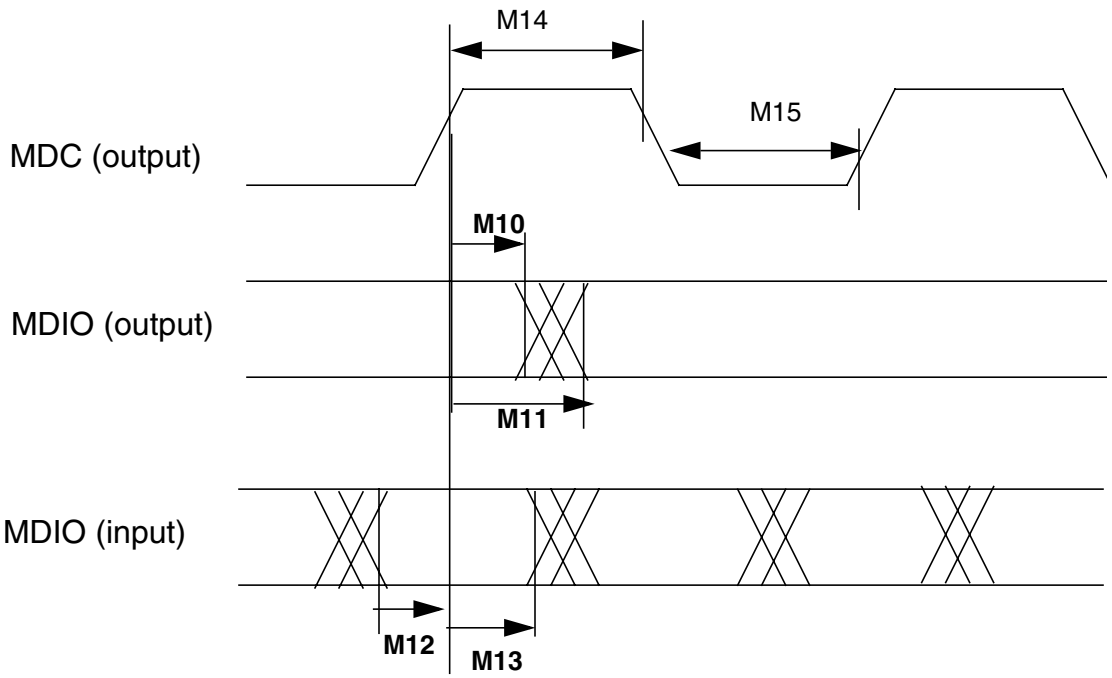


Figure A-4. MII Serial Management Channel Timing Diagram

A.14 EPHY Electrical Characteristics

NOTE

The electrical characteristics given in the EPHY section are preliminary and should be used as a guide only. Values cannot be guaranteed by Freescale Semiconductor and are subject to change without notice.

Table A-18. 10BASE-T SQE (Heartbeat) Timing Parameters

Num	C	Parameter	Sym	Min	Typ	Max	Units
1	D	COL (SQE) delay after TXEN off	t1		1.0		µs
2	D	COL (SQE) pulse duration	t2		1.0		µs

Typical values are at 25°C.
1 BT = Bit Time = 100 ns

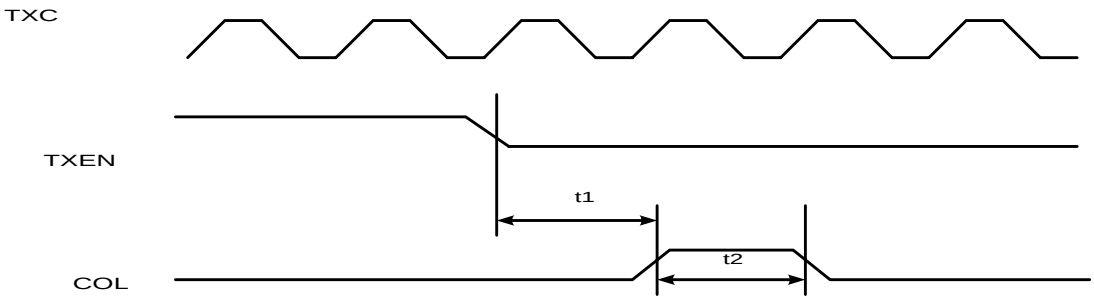


Figure A-5. 10BASE-T SQE (Heartbeat) Timing

A.14.1 10BASE-T Jab and Unjab Timing

Table A-19. 10BASE-T Jab and Unjab Timing Parameters

Num	C	Parameter	Sym	Min	Typ	Max	Units
1	D	Maximum Transmit time	t1		98		ms
2	D	Unjab time	t2		525		ms

Typical values are at 25°C.
1 BT = Bit Time = 100 ns

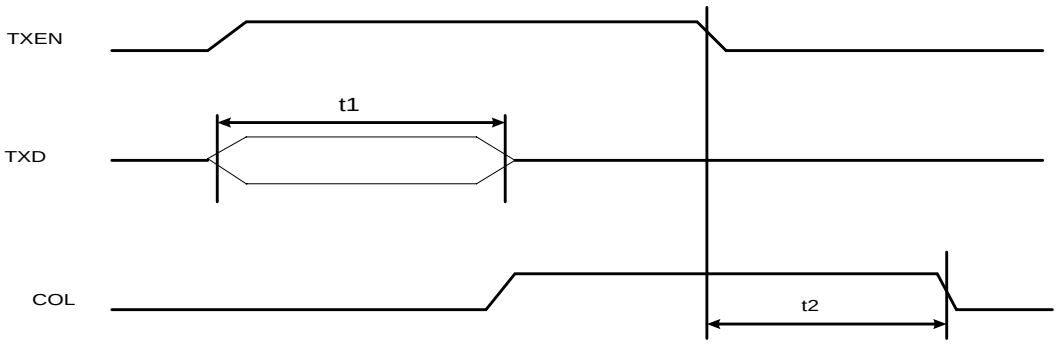


Figure A-6. 10BASE-T SQE (Heartbeat) Timing

A.14.2 Auto Negotiation

A.14.2.1 MII – 100BASE-TX Transmit Timing Parameters

Table A-20. MII – Auto Negotiation and Fast Link Pulse Timing Parameters

Num	C	Parameter	Sym	Min	Typ	Max	Units
1	D	Clock/data pulse width	t1		100		ns
2	D	Clock pulse to clock pulse	t2	111	125	139	µs
3	D	Clock pulse to data pulse (data = 1)	t3	55.5	62.5	69.5	µs
4	D	Pulses in a burst	t4	17		33	#
5	D	FLP burst width	t5		2		ms
6	D	FLP burst to FLP burst	t6	8	16	24	ms

Typical values are at 25°C.
1 BT = Bit Time = 100 ns
These parameters are the minimum and maximum times as specified in section 24.6 of the IEEE 802.3u Standard

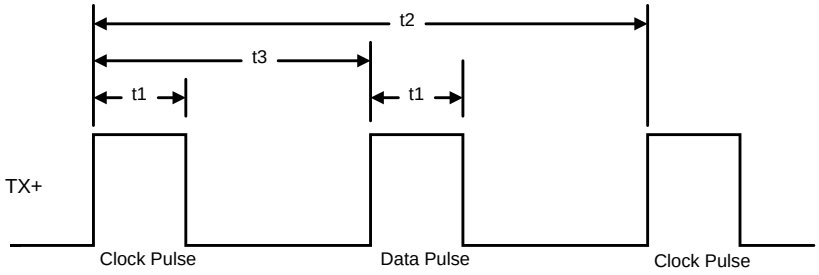


Figure A-7. Auto-Negotiation and Fast Link Pulse Timing

A.14.2.2 MII — 10BASE-T Receive Timing

Table A-21. Auto-Negotiation and Fast Link Pulse Timing

Num	C	Parameter	Sym	Min	Typ	Max	Units
1	D	Transmit FLNP width		1.25	1.5	1.75	µs
2	D	Receive FLNP width		1	1.5	2	µs
3	D	Clock/data pulse width	t1		100		ns
4	D	Clock FLNP to clock FLNP	t2	111	125	139	µs
5	D	Clock FLNP to data FLNP (data = 1)	t3	55.5	62.5	69.5	µs
6	D	Pulses in a burst	t4	17		33	#
7	D	FLNP burst width	t5		2		ms
8	D	FLNP burst to FLNP burst	t6	8	16	24	ms

Typical values are at 25°C.
1 BT = Bit Time = 100 ns

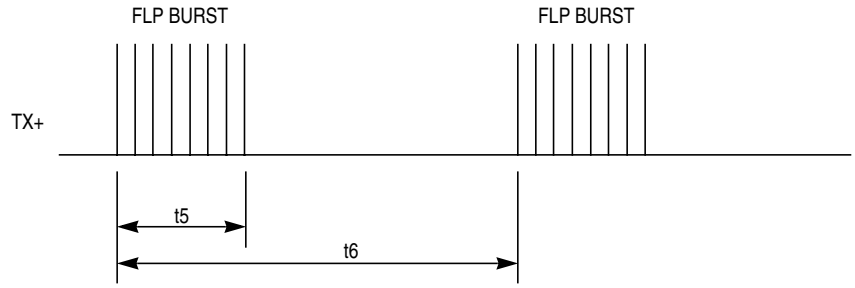


Figure A-8. Fast Link Pulse Timing

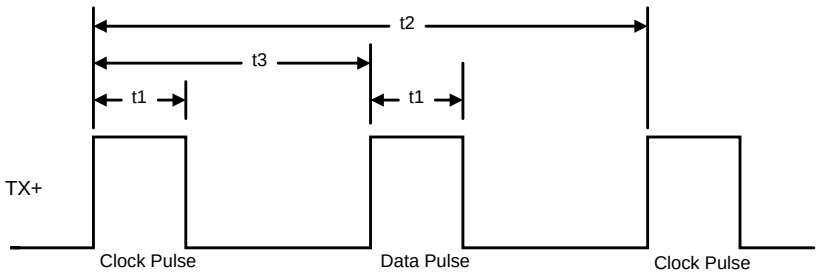


Figure A-9. Auto-Negotiation Pulse Timing

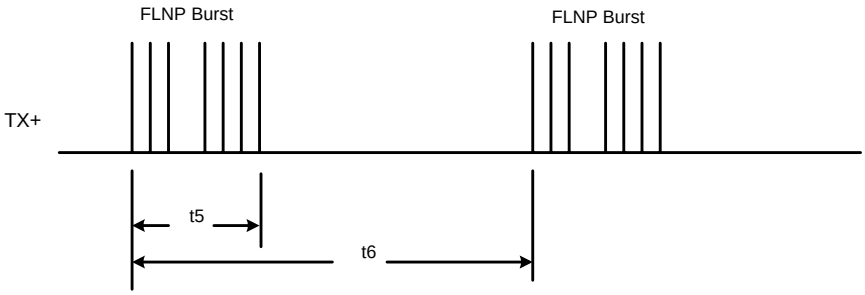


Figure A-10. Fast Link Pulse Timing

Table A-22. 10BASE-T Transceiver Characteristics

Num	C	Parameter	Sym	Min	Typ	Max	Units	Test Conditions
1	D	Peak differential output voltage	V_{OP}	2.2	2.5	2.8	V	With specified transformer and line replaced by 100 Ω(1%) load
2	D	Transmit timing jitter	—	0	2	11	ns	Using line model specified in the IEEE 802.3
3	D	Receive dc input impedance	Z_{in}	—	10	—	kΩ	0.0 < V_{in} < 3.3 V
4	D	Receive differential squelch level	$V_{squelch}$	300	400	585	mV	3.3 MHz sine wave input

Table A-23. 100BASE-TX Transceiver Characteristics

Num	C	Parameter	Sym	Min	Typ	Max	Units	Test Conditions
1	D	Transmit Peak Differential Output Voltage	V_{OP}	0.95	1.00	1.05	V	With specified transformer and line replaced by 100 Ω(1%) load
2	D	Transmit Signal Amplitude Symmetry	V_{sym}	98	100	102	%	With specified transformer and line replaced by 100 Ω (1%) load
3	D	Transmit Rise/Fall Time	t_{rf}	3	4	5	ns	With specified transformer and line replaced by 100 Ω (1%) load
4	D	Transmit Rise/Fall Time Symmetry	t_{rfs}	–0.5	0	+0.5	ns	See IEEE 802.3 for details
5	D	Transmit Overshoot/UnderShoot	V_{osh}	—	2.5	5	%	
6	D	Transmit Jitter	—	0	.6	1.4	ns	
7	D	Receive Common Mode Voltage	V_{cm}	—	1.6	—	V	V_{DDRX} = 2.5 V
8	D	Receiver Maximum Input Voltage	V_{max}	—	—	4.7	V	V_{DDRX} = 2.5 V. Internal circuits protected by divider in shutdown

Table A-24. EPHY Operating Conditions

Num	Parameter	Min	Typ	Max	Units
1	Crystal [1]	25	—	25	MHz
2	Bus clock in single chip mode — 10 Mbps operation	2.5	—	25	MHz
3	Bus clock in external mode — 10 Mbps operation	2.5	—	f_o	MHz
4	Bus clock in single chip mode — 100 Mbps operation	25	—	25	MHz
5	Bus clock in external mode — 100 Mbps	Not available			
6	Bias resistor	12.4, 1%			kΩ

[1] Crystal tolerance must conform to IEEE requirements.

A.15 FLASH NVM Electrical Characteristics

A.15.1 NVM timing

The time base for all NVM program or erase operations is derived from the oscillator. A minimum oscillator frequency f_{NVMOSC} is required for performing program or erase operations. The NVM modules do not have any means to monitor the frequency and will not prevent program or erase operation at frequencies above or below the specified minimum. Programming or erasing the NVM modules at a lower frequency will not result in a full program or erase transition.

The FLASH program and erase operations are timed using a clock derived from the oscillator using the FCLKDIV register. The frequency of this clock must be set within the limits specified as f_{NVMOP}.

The minimum program and erase times shown in Table A-25 are calculated for maximum f_{NVMOP} and maximum f_{bus}. The maximum times are calculated for minimum f_{NVMOP} and a f_{bus} of 2 MHz.

A.15.1.1 Single Word Programming

The programming time for single word programming is dependent on the bus frequency as a well as on the frequency f_{NVMOP} and can be calculated according to the following formula.

$$t_{swpgm} = 9 \cdot \frac{1}{f_{NVMOP}} + 25 \cdot \frac{1}{f_{bus}}$$

A.15.1.2 Burst Programming

FLASH programming where up to 32 words in a row can be programmed consecutively using burst programming by keeping the command pipeline filled. The time to program a consecutive word can be calculated as:

$$t_{bwpgm} = 4 \cdot \frac{1}{f_{NVMOP}} + 9 \cdot \frac{1}{f_{bus}}$$

The time to program a whole row is:

$$t_{brpgm} = t_{swpgm} + 31 \cdot t_{bwpgm}$$

Burst programming is more than two times faster than single word programming.

A.15.1.3 Sector Erase

Erasing a 512 byte FLASH sector takes:

$$t_{era} \approx 4000 \cdot \frac{1}{f_{NVMOP}}$$

The setup times can be ignored for this operation.

A.15.1.4 Mass Erase

Erasing a NVM block takes:

$$t_{mass} \approx 20000 \cdot \frac{1}{f_{NVMOP}}$$

The setup times can be ignored for this operation.

Table A-25. NVM Timing Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	External oscillator clock	f_{NVMOSC}	0.5		50 [1]	MHz
2	D	Bus frequency for programming or erase operations	f_{NVMBUS}	1			MHz
3	D	Operating frequency	f_{NVMOP}	150		200	kHz
4	P	Single Word programming time	t_{swpgm}	46 [2]		74.5 [3]	µs
5	D	FLASH burst programming consecutive word	t_{bwpgm}	20.4[2]		31[3]	µs
6	D	FLASH burst programming time for 32 words	t_{brpgm}	678.4[2]		1035.5[3]	µs
7	P	Sector erase time	t_{era}	20 [4]		26.7[3]	ms
8	P	Mass erase time	t_{mass}	100[4]		133[3]	ms
9	D	Blank check time FLASH per block	t_{check}	11 [5]		32778 [6]	t_{cyc}

1 Restrictions for oscillator in crystal mode apply!

2 Minimum programming times are achieved under maximum NVM operating frequency f_{NVMOP} and maximum bus frequency f_{bus}.

3 Maximum erase and programming times are achieved under particular combinations of f_{NVMOP} and bus frequency f bus.

4 Minimum Erase times are achieved under maximum NVM operating frequency f_{NVMOP}

5 Minimum time, if first word in the array is not blank

6 Maximum time to complete check on an erased block.

A.15.2 NVM Reliability

The reliability of the NVM blocks is guaranteed by stress test during qualification, constant process monitors and burn-in to screen early life failures.

The failure rates for data retention and program/erase cycling are specified at < 2 ppm defects over lifetime at the operating conditions noted.

A program/erase cycle is specified as two transitions of the cell value from erased → programmed → erased, 1 → 0 → 1.

NOTE

All values shown in Table A-26 are target values and subject to further extensive characterization.

Table A-26. NVM Reliability Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	C	Data retention at an average junction temperature of T_{Javg} = 85°C	t_{NVMRET}	15			Years
2	C	Data retention at a junction temperature of T_J = 140°C	t_{NVMRET}	10			Years
3	C	FLASH number of program/erase cycles	n_{FLPE}	10,000			Cycles

A.16 SPI Electrical Characteristics

This section provides electrical parametrics and ratings for the SPI.

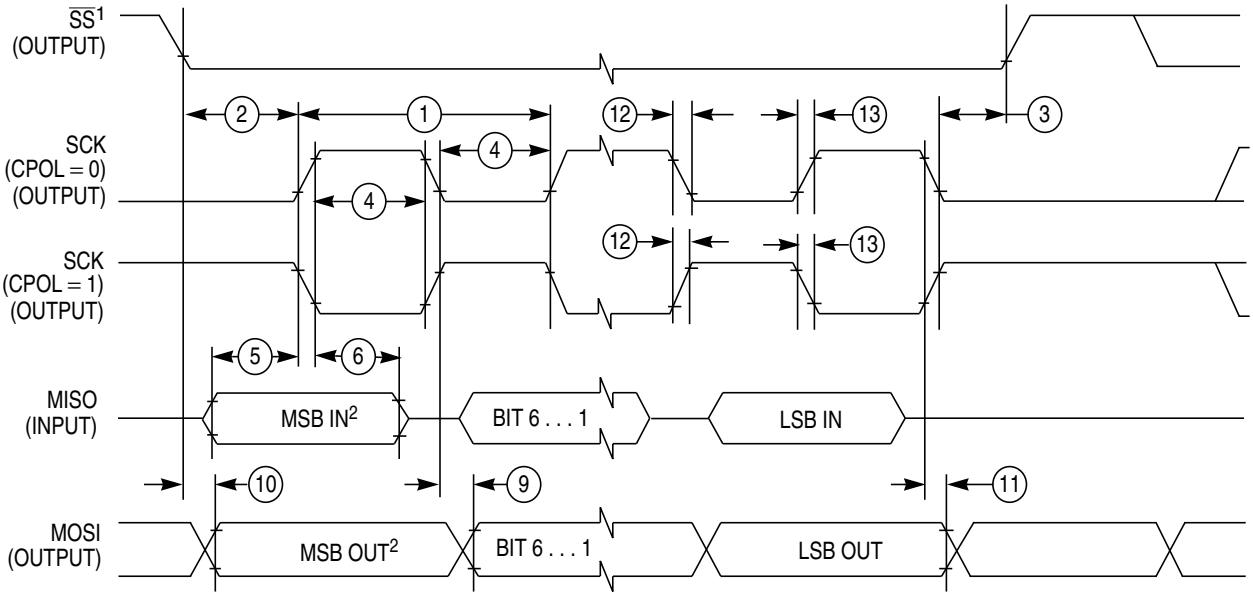
In Table A-27 the measurement conditions are listed.

Table A-27. Measurement Conditions

Description	Value	Unit
Drive mode	full drive mode	—
Load capacitance C_{LOAD}, on all outputs	50	pF
Thresholds for delay measurement points	(20% / 80%) VDDX	V

A.16.1 Master Mode

In Figure A-11 the timing diagram for master mode with transmission format CPHA=0 is depicted.

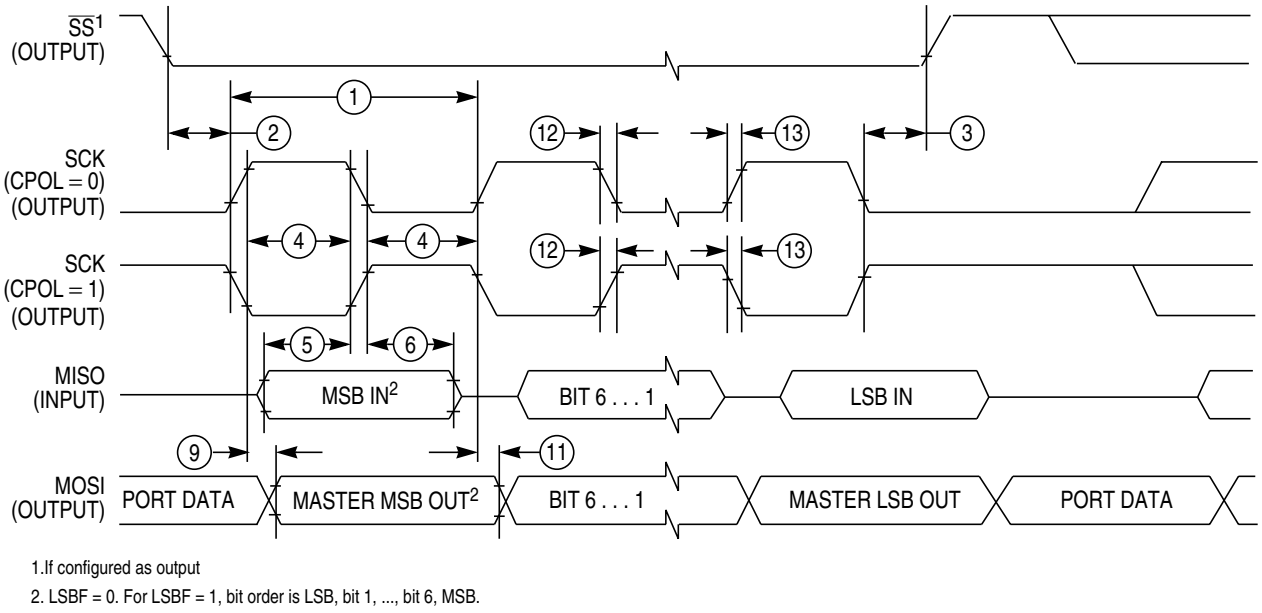


1.if configured as an output.

2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure A-11. SPI Master Timing (CPHA=0)

In Figure A-12 the timing diagram for master mode with transmission format CPHA=1 is depicted.



1.If configured as output

2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure A-12. SPI Master Timing (CPHA=1)

In Table A-28 the timing characteristics for master mode are listed.

Table A-28. SPI Master Mode Timing Characteristics

Num	C	Characteristic	Symbol	Min	Typ	Max	Unit
1	P	SCK Frequency	f_{sck}	1/2048	—	1/2	f_{bus}
1	P	SCK Period	t_{sck}	2	—	2048	t_{bus}
2	D	Enable Lead Time	t_{lead}	—	1/2	—	t_{sck}
3	D	Enable Lag Time	t_{lag}	—	1/2	—	t_{sck}
4	D	Clock (SCK) High or Low Time	t_{wsck}	—	1/2	—	t_{sck}
5	D	Data Setup Time (Inputs)	t_{su}	8	—	—	ns
6	D	Data Hold Time (Inputs)	t_{hi}	8	—	—	ns
9	D	Data Valid after SCK Edge	t_{vsck}	—	—	30	ns
10	D	Data Valid after $\overline{SS}$ fall (CPHA=0)	t_{vss}	—	—	15	ns
11	D	Data Hold Time (Outputs)	t_{ho}	20	—	—	ns
12	D	Rise and Fall Time Inputs	t_{rfi}	—	—	8	ns
13	D	Rise and Fall Time Outputs	t_{rfo}	—	—	8	ns

A.16.2 Slave Mode

In Figure A-13 the timing diagram for slave mode with transmission format CPHA = 0 is depicted.

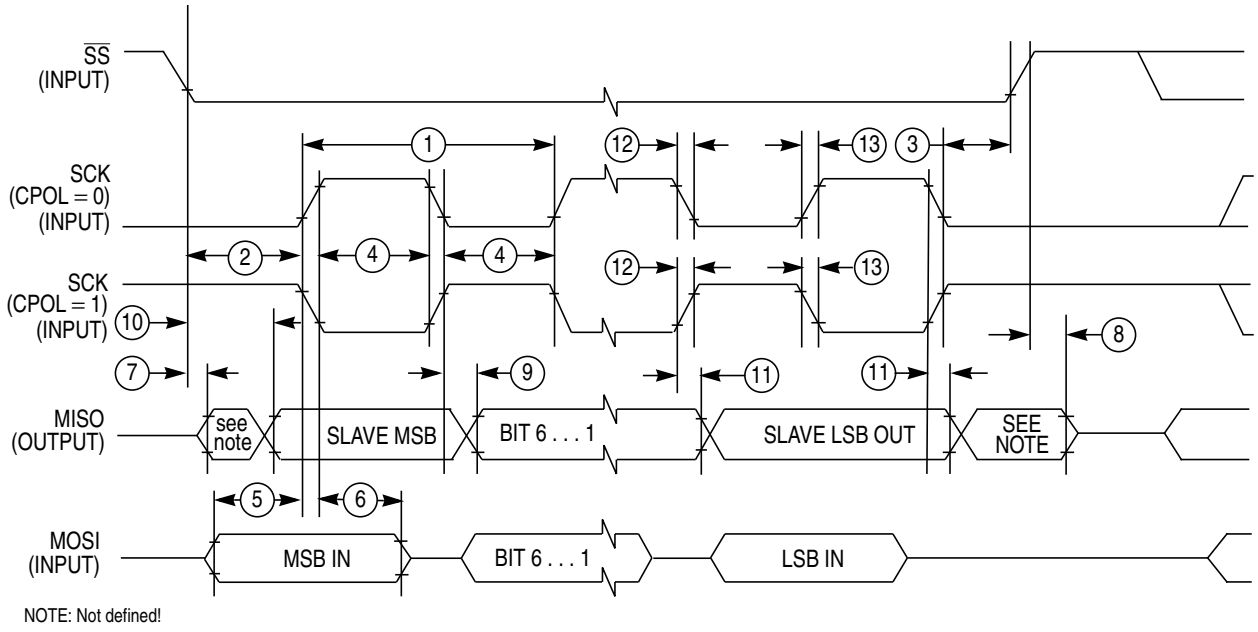


Figure A-13. SPI Slave Timing (CPHA = 0)

In Figure A-14 the timing diagram for slave mode with transmission format CPHA = 1 is depicted.

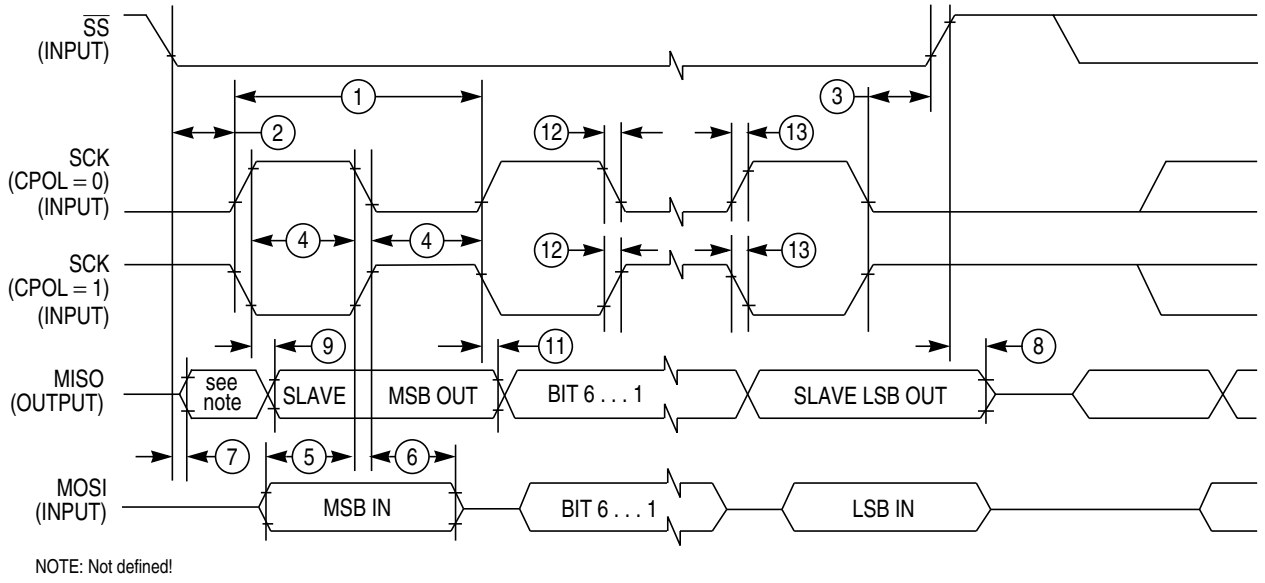


Figure A-14. SPI Slave Timing (CPHA = 1)

In Table A-29 the timing characteristics for slave mode are listed.

Table A-29. SPI Slave Mode Timing Characteristics

Num	C	Characteristic	Symbol	Min	Typ	Max	Unit
1	P	SCK Frequency	f_{sck}	DC	—	1/4	f_{bus}
1	P	SCK Period	t_{sck}	4	—	∞	t_{bus}
2	D	Enable Lead Time	t_{lead}	4	—	—	t_{bus}
3	D	Enable Lag Time	t_{lag}	4	—	—	t_{bus}
4	D	Clock (SCK) High or Low Time	t_{wsck}	4	—	—	t_{bus}
5	D	Data Setup Time (Inputs)	t_{su}	8	—	—	ns
6	D	Data Hold Time (Inputs)	t_{hi}	8	—	—	ns
7	D	Slave Access Time (time to data active)	t_a	—	—	20	ns
8	D	Slave MISO Disable Time	t_{dis}	—	—	22	ns
9	D	Data Valid after SCK Edge	t_{vsck}	—	—	30 + t_{bus} [1]	ns
10	D	Data Valid after $\overline{SS}$ fall	t_{vss}	—	—	30 + t_{bus} [1]	ns
11	D	Data Hold Time (Outputs)	t_{ho}	20	—	—	ns
12	D	Rise and Fall Time Inputs	t_{rfi}	—	—	8	ns
13	D	Rise and Fall Time Outputs	t_{rfo}	—	—	8	ns

[1] t_{bus} added due to internal synchronization delay

A.17 Voltage Regulator Operating Characteristics

This section describes the characteristics of the on-chip voltage regulator (VREG_PHY).

Table A-30. VREG_PHY - Operating Conditions

Num	C	Characteristic	Symbol	Min	Typical	Max	Unit
1	P	Input Voltages	$V_{VDDR,A,X1,X2}$	3.135	—	3.465	V
2	P	Regulator Current Reduced Power Mode Shutdown Mode	I_{REG}	 — —	 20 12	 50 40	 µA µA
3	P	Output Voltage Core Full Performance Mode Reduced Power Mode Shutdown Mode	V_{DD}	 2.375 1.6 —	 2.5 2.5 [1] —	 2.625 2.75 —	 V V V
4	P	Output Voltage PLL Full Performance Mode Reduced Power Mode [2] Shutdown Mode	V_{DDPLL}	 2.375 1.6 —	 2.5 2.5 [3] —	 2.625 2.75 —	 V V V
5	P	Low Voltage Reset [4] Assert Level Deassert Level	 V_{LVRA} V_{LVRD}	 2.25 —	 — —	 — 2.55	 V V
7	C	Power-on Reset [5] Assert Level Deassert Level	 V_{PORA} V_{PORD}	 0.97 —	 --- ---	 — 2.05	 V V

1 High Impedance Output

2 Current I_{DDPLL} = 3 mA (Pierce Oscillator)

3 High Impedance Output

4 Monitors V_{DD}, active only in Full Performance Mode. V_{LVRA} and V_{PORD} must overlap

5 Monitors V_{DD}. Active in all modes.

The electrical characteristics given in this section are preliminary and should be used as a guide only. Values in this section cannot be guaranteed by Freescale Semiconductor and are subject to change without notice.

A.17.1 MCU Power-Up and LVR Graphical Explanation

Voltage regulator sub modules POR (power-on reset) and LVR (low-voltage reset) handle chip power-up or drops of the supply voltage. Their function is described in Figure A-15.

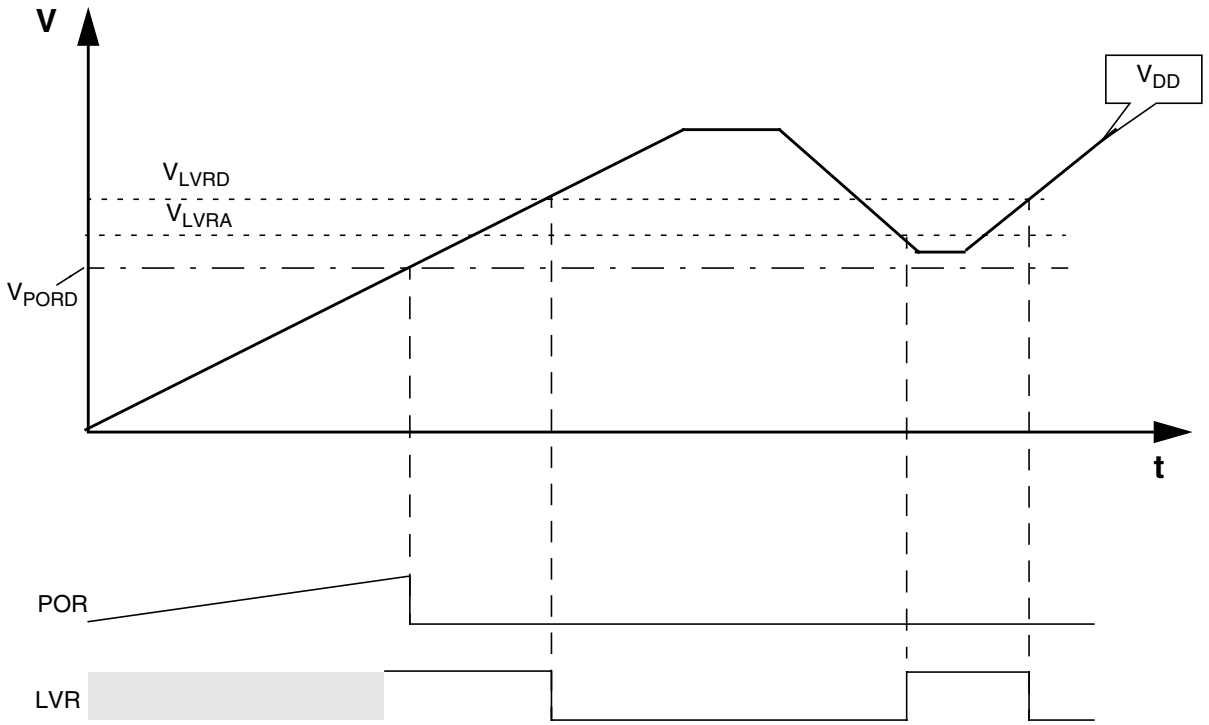


Figure A-15. Voltage Regulator — MCU Power-Up And Voltage Drops (Not Scaled)

A.17.2 Output Loads

A.17.2.1 Resistive Loads

The on-chip voltage regulator is intended to supply the internal logic and oscillator circuits allows no external DC loads.

A.17.2.2 Capacitive Loads

The capacitive loads are specified in Table A-31. Ceramic capacitors with X7R dielectricum are required.

Table A-31. Voltage Regulator — Capacitive Loads

Num	Characteristic	Symbol	Min	Typical	Max	Unit
1	V_{DD} external capacitive load	C_{DDext}	200	440	12000	nF
2	PHY_VDDTX external capacitive load	$C_{DDPLLext}$	90	220	5000	nF
2	PHY_VDDRX external capacitive load	$C_{DDPLLext}$	90	220	5000	nF
2	PHY_VDDA external capacitive load	$C_{DDPLLext}$	90	220	5000	nF
2	V_{DDPLL} external capacitive load	$C_{DDPLLext}$	90	220	5000	nF

A.18 External Bus Timing

A timing diagram of the external multiplexed-bus is illustrated in Figure 18-25 with the actual timing values shown on table Table A-32. All major bus signals are included in the diagram. Although both a data write and data read cycle are shown, only one or the other would occur on a particular bus cycle.

The expanded bus timings are highly dependent on the load conditions. The timing parameters shown assume a balanced load across all outputs.

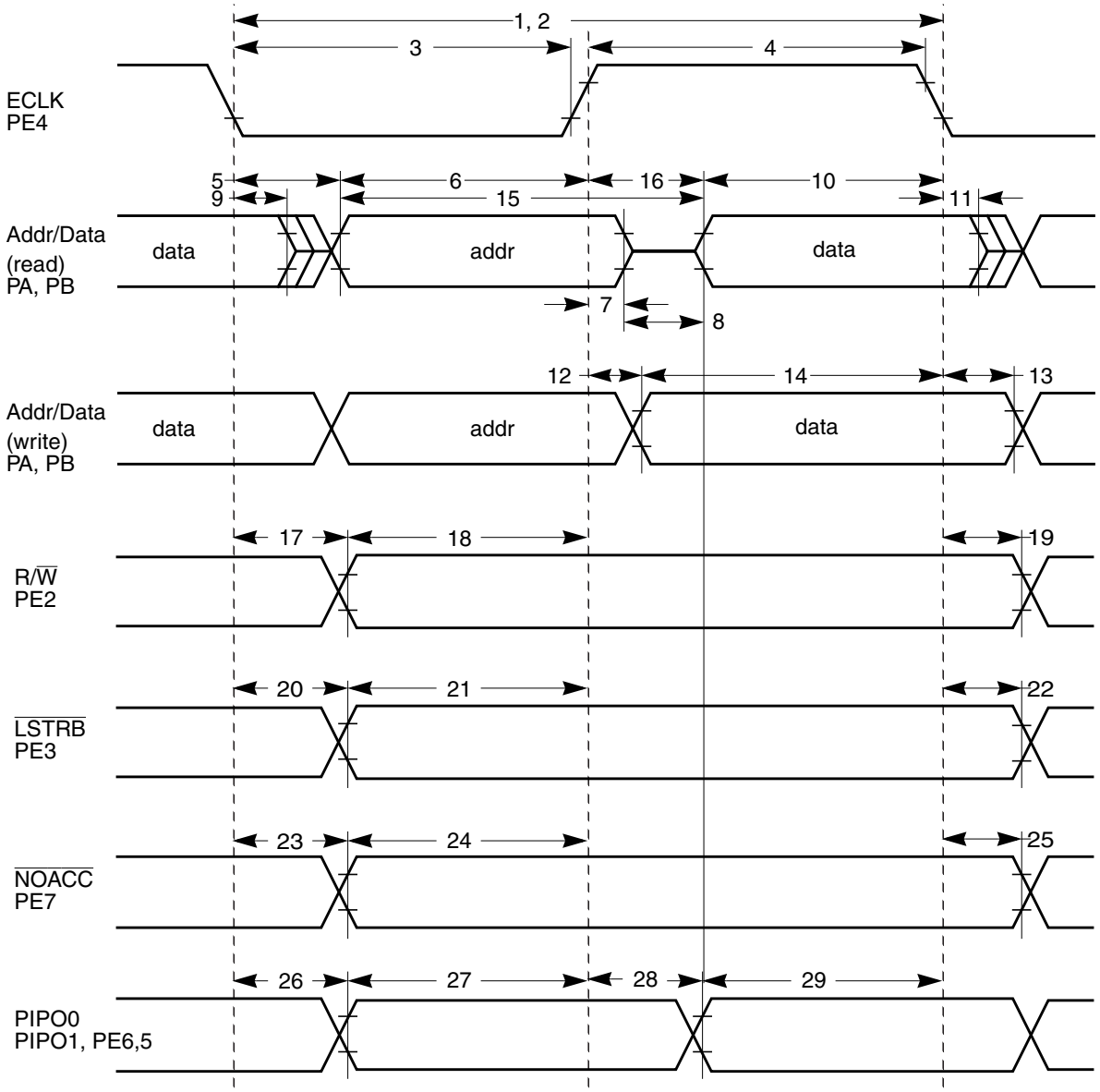


Figure 18-25. General External Bus Timing

Table A-32. Expanded Bus Timing Characteristics (3.3 V Range)

Conditions are VDDX = 3.3 V 5%, Junction Temperature -40˚C to +125˚C, C_{LOAD} = 50 pF

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Frequency of operation (E-clock)	f_o	0		16.0	MHz
2	P	Cycle time	t_{cyc}	62.5			ns
3	D	Pulse width, E low	PW_{EL}	30			ns
4	D	Pulse width, E high [1]	PW_{EH}	30			ns
5	D	Address delay time	t_{AD}			16	ns
6	D	Address valid time to E rise (PW_{EL}–t_{AD})	t_{AV}	16			ns
7	D	Muxed address hold time	t_{MAH}	2			ns
8	D	Address hold to data valid	t_{AHDS}	7			ns
9	D	Data hold to address	t_{DHA}	2			ns
10	D	Read data setup time	t_{DSR}	15			ns
11	D	Read data hold time	t_{DHR}	0			ns
12	D	Write data delay time	t_{DDW}			15	ns
13	D	Write data hold time	t_{DHW}	2			ns
14	D	Write data setup time[1] (PW_{EH}–t_{DDW})	t_{DSW}	15			ns
15	D	Address access time[1]	t_{ACCA}	29			ns
16	D	E high access time[1] (PW_{EH}–t_{DSR})	t_{ACCE}	15			ns
17	D	Read/write delay time	t_{RWD}			14	ns
18	D	Read/write valid time to E rise (PW_{EL}–t_{RWD})	t_{RWV}	16			ns
19	D	Read/write hold time	t_{RWH}	2			ns
20	D	Low strobe delay time	t_{LSD}			14	ns
21	D	Low strobe valid time to E rise (PW_{EL}–t_{LSD})	t_{LSV}	16			ns
22	D	Low strobe hold time	t_{LSH}	2			ns
23	D	NOACC strobe delay time	t_{NOD}			14	ns
24	D	NOACC valid time to E rise (PW_{EL}–t_{LSD})	t_{NOV}	16			ns
25	D	NOACC hold time	t_{NOH}	2			ns
26	D	IPIPO[1:0] delay time	t_{P0D}	2		14	ns
27	D	IPIPO[1:0] valid time to E rise (PW_{EL}–t_{P0D})	t_{P0V}	16			ns
28	D	IPIPO[1:0] delay time[1]	t_{P1D}	2		25	ns
29	D	IPIPO[1:0] valid time to E fall	t_{P1V}	11			ns

[1]Affected by clock stretch: add N x t_{cyc} where N=0,1,2 or 3, depending on the number of clock stretches.

Appendix B
Schematic and PCB Layout Design Recommendations

B.1 Introduction

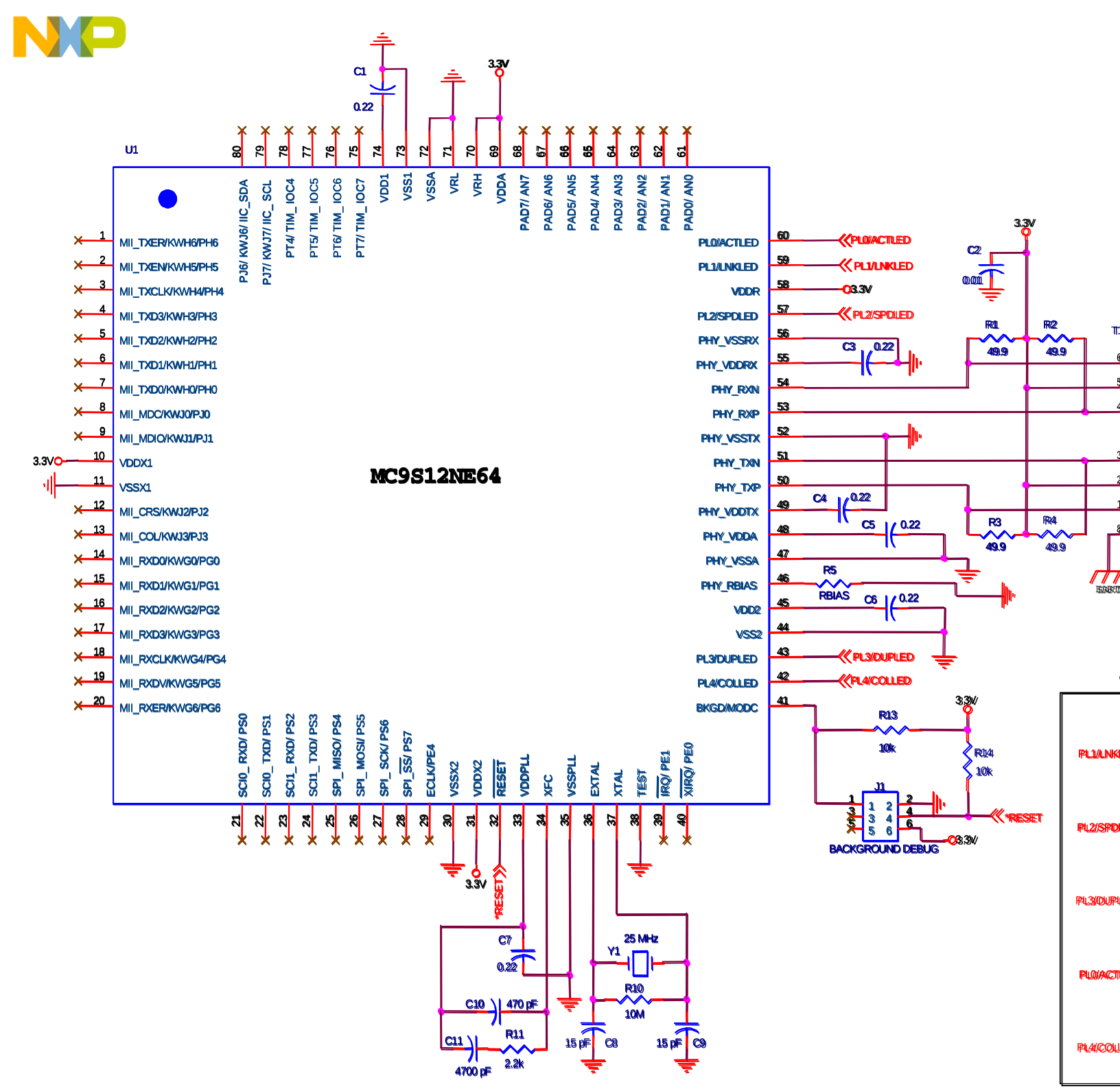
This sections provides recommendations for schematic and PCB layout design for implementing an Ethernet interface with the MC9S12NE64 microcontroller unit (MCU).

B.1.1 Schematic Designing with the MC9S12NE64 and Adding an Ethernet Interface

Figure B-1 is a schematic of a MC9S12NE64 80-pin package minimum system implementation configured in normal-chip mode and utilizing the internal voltage regulator. This configuration is the recommended implementation for the MC9S12NE64. The schematic provides a reference for the following MC9S12NE64 design items.

- Operation mode
- Clocks
- Power
- Ethernet, high-speed LAN magnetics isolation module, and RJ45 Ethernet connector
- EPHY status indicators
- Background debug connector (J1)

To configure the MC9S12NE64 in normal single-chip mode, the MODC, MODB, and MODA pins should be configured as documented in the device overview chapter of this book.


U1
MC9S12NE64
MII_TXER/KWH6/PH6
MII_TXEN/KWH5/PH5
MII_TXCLK/KWH4/PH4
MII_TXD3/KWH3/PH3
MII_TXD2/KWH2/PH2
MII_TXD1/KWH1/PH1
MII_TXD0/KWH0/PH0
MII_MDC/KWJ0/PJ0
MII_MDIO/KWJ1/PJ1
VDDX1
VSSX1
MII_CRS/KWJ2/PJ2
MII_COL/KWJ3/PJ3
MII_RXD0/KWG0/PG0
MII_RXD1/KWG1/PG1
MII_RXD2/KWG2/PG2
MII_RXD3/KWG3/PG3
MII_RXCLK/KWG4/PG4
MII_RXDV/KWG5/PG5
MII_RXER/KWG6/PG6
PJ6/ KWJ6/ IIC_SDA
PJ7/ KWJ7/ IIC_SCL
PT4/ TIM_IOC4
PT5/ TIM_IOC5
PT6/ TIM_IOC6
PT7/ TIM_IOC7
VDD1
VSS1
VSSA
VRL
VRH
VDDA
PAD7/ AN7
PAD6/ AN6
PAD5/ AN5
PAD4/ AN4
PAD3/ AN3
PAD2/ AN2
PAD1/ AN1
PAD0/ AN0
PL0/ACTLED
PL1/LNKLED
VDDR
PL2/SPDLED
PHY_VSSRX
PHY_VDDRX
PHY_RXN
PHY_RXP
PHY_VSSTX
PHY_TXN
PHY_TXP
PHY_VDDTX
PHY_VDDA
PHY_VSSA
PHY_RBIAS
VDD2
VSS2
PL3/DUPLED
PL4/COLLED
BKGD/MODC
SCI0_ RXD/ PS0
SCI0_ TXD/ PS1
SCI1_ RXD/ PS2
SCI1_ TXD/ PS3
SPI_ MISO/ PS4
SPI_ MOSI/ PS5
SPI_ SCK/ PS6
SPI_ SS/ PS7
ECLK/PE4
VSSX2
VDDX2
RESET
VDDPLL
XFC
VSSPLL
EXTAL
XTAL
TEST
IRQ/ PE1
XIRQ/ PE0
3.3V
C1
0.22
C2
0.001
C3
0.22
C4
0.22
C5
0.22
C6
0.22
C7
0.22
C8
15 pF
C9
15 pF
C10
470 pF
C11
4700 pF
R1
49.9
R2
49.9
R3
49.9
R4
49.9
R5
RBIAS
R10
10M
R11
2.2k
R13
10k
R14
10k
Y1
25 MHz
J1
BACKGROUND DEBUG
*RESET

B.1.2 Power Supply Notes

A 3.3-V power supply is required. This power supply shall be compatible with **Table A-7. Supply Current Characteristics**.

B.1.3 Clocking Notes

For basic operation of the MC9S12NE64, a 25-MHz crystal is required to provide the clock input to the integrated PHY. The crystal must connect to the MC9S12NE64 in a Pierce configuration by the XTAL and EXTAL pins as shown in Figure B-1.

In addition to providing a 25-MHz crystal input, to operate at 100 Mbps, the internal bus clock must be configured as shown in the EPHY electrical characteristics.

B.1.4 EPHY Notes

Figure B-2 provides a close-up view of the EPHY pin connections to a high-speed LAN magnetics isolation module and RJ45 Ethernet connector.

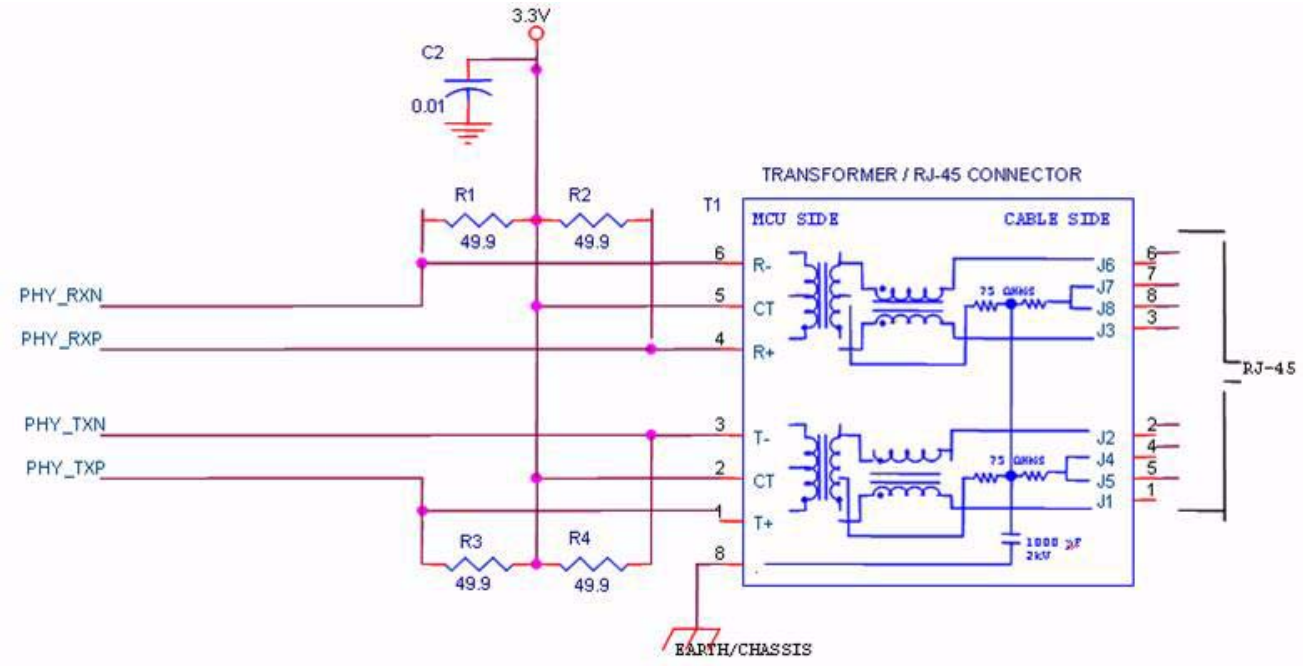


Figure B-2. Ethernet Interface Circuitry

B.1.5 EPHY LED Indicator Notes

The EPHY can be configuring by software to drive indicator pins (PTL[5:0]) automatically by setting the LEDEN bit of the EPHY EPHYCTL0 register. When LEDEN = 1, PTL[5:0] pins are dedicated to the EPHY.

B.2 PCB Design Recommendation

The section provides recommendations for general HCS12 PCB design and recommendations for PCB design with Ethernet.

B.2.1 General PCB Design Recommendations

The PCB layout must be designed to ensure proper operation of the voltage regulator and the MCU. The following recommendations are provided to ensure a robust PCB design:

- Every supply pair must be decoupled by a ceramic capacitor connected as near as possible to the corresponding pins (C1 - C6).
- Central point of the ground star should be the VSSX pin.
- Use low ohmic low inductance connections between VSS1, VSS2 and VSSX.
- VSSPLL must be directly connected to VSSX.
- Keep traces of VSSPLL, EXTAL and XTAL as short as possible and occupied board area for C7,C8, C11 and Q1 as small as possible.
- Do not place other signals or supplies underneath area occupied by C7, C8, C10 and Q1 and the connection area to the MCU.
- Central power input should be fed in at the VDDA/VSSA pins.

B.2.2 Ethernet PCB Design Recommendations

When designing a PCB that uses the MC9S12NE64 Ethernet module, several design considerations must be made to ensure that Ethernet operation conforms to the IEEE 802.3 physical interface specification. These Ethernet PCB design recommendations include:

- The distance between the magnetic module and the RJ-45 jack is the most critical and must always be as short as possible (less than one inch).
- Never use 90° traces. Use 45° angles or radius curves in traces.
- Trace widths of 0.010” are recommended. Wider is better. Trace widths should not vary.
- Route differential Tx and Rx pairs near together (max 0.010” separation with 0.010” traces).
- Trace lengths must always be as short as possible (must be less than one inch).
- Make trace lengths as equal as possible.
- Keep TX and RX differential pairs routes separated (at least 0.020” separation). Better to separate with a ground plane.
- Avoid routing Tx and Rx traces over or under a plane. Areas under the Tx and Rx traces should be open.
- Use precision components in the line termination circuitry with 1% tolerance.
- Ensure that the power supply is rated for a load of 300 mA minimum.
- Avoid vias and layer changes.

- All termination resistors should be near to the driving source. The MCU is the driving source for PHY_TXP and PHY_TXN pins. The high-speed LAN magnetics isolation module is the driving source for PHY_RXP and PHY_RXN pins.
- 4-layer PCBs recommended to provide better heat dissipation

B.2.2.1 High-Speed LAN Magnetics Isolation Module Requirements

The MC9S12NE64 requires a 1:1 ratio for the high-speed LAN magnetics isolation module for both the receive and the transmit signals. Because the MC9S12NE64 does not implement Auto-MDIX, an Auto-MDIX capable high-speed LAN magnetics isolation module is not required. A high-speed LAN magnetics isolation module with improved return loss characteristics is recommended to avoid Ethernet return loss issues.

B.2.2.2 80-Pin Package Exposed Flag

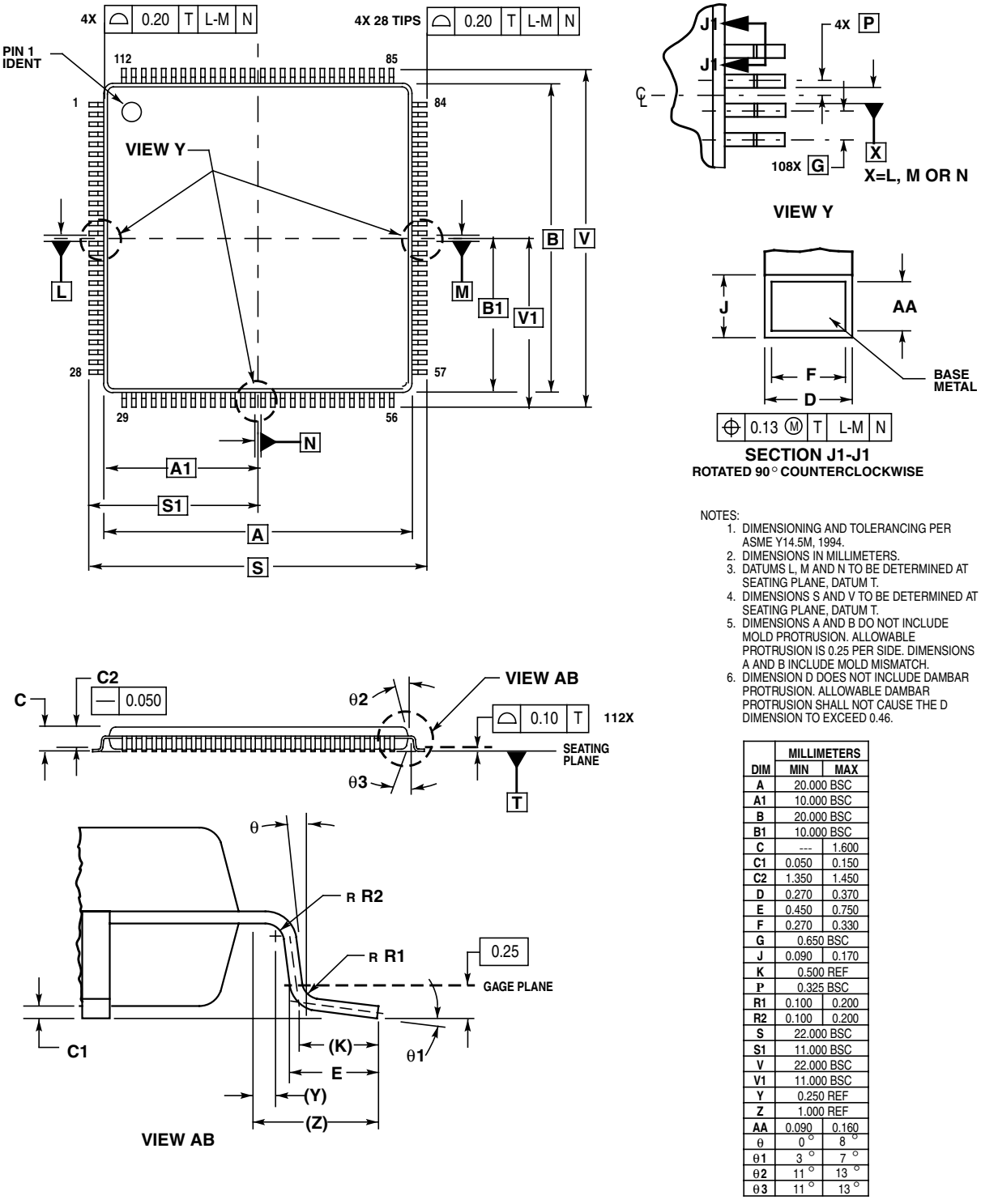
The 80-pin TQFP-EP package has an exposed flag for heat dissipation and requires special PCB layout to accommodate the flag. There are two ways to accommodate the flag:

- Have a hatched pattern in the solder mask
- Use small copper areas under the flag

The requirement is to have about 50% of the flag soldered to the PC board.

Appendix C
Package Information

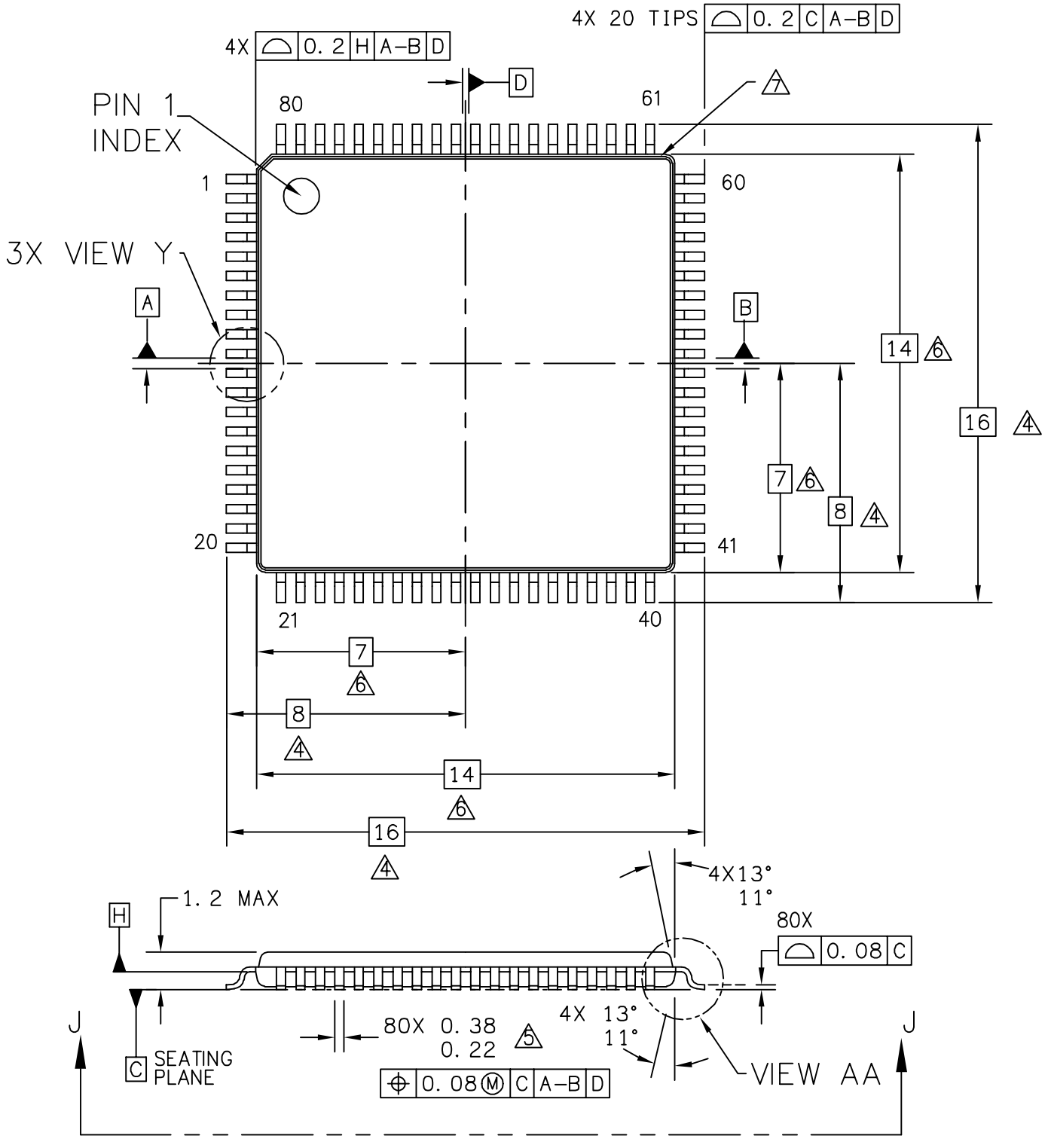
C.1 112-Pin LQFP Package



NOTES:
1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M, 1994.
2. DIMENSIONS IN MILLIMETERS.
3. DATUMS L, M AND N TO BE DETERMINED AT SEATING PLANE, DATUM T.
4. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE, DATUM T.
5. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 PER SIDE. DIMENSIONS A AND B INCLUDE MOLD MISMATCH.
6. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE D DIMENSION TO EXCEED 0.46.

	MILLIMETERS	
DIM	MIN	MAX
A	20.000 BSC	
A1	10.000 BSC	
B	20.000 BSC	
B1	10.000 BSC	
C	---	1.600
C1	0.050	0.150
C2	1.350	1.450
D	0.270	0.370
E	0.450	0.750
F	0.270	0.330
G	0.650 BSC	
J	0.090	0.170
K	0.500 REF	
P	0.325 BSC	
R1	0.100	0.200
R2	0.100	0.200
S	22.000 BSC	
S1	11.000 BSC	
V	22.000 BSC	
V1	11.000 BSC	
Y	0.250 REF	
Z	1.000 REF	
AA	0.090	0.160
θ	0°	8°
θ1	3°	7°
θ2	11°	13°
θ3	11°	13°

Figure C-1. 112-Pin LQFP Mechanical Drawing (Case No. 987-01)

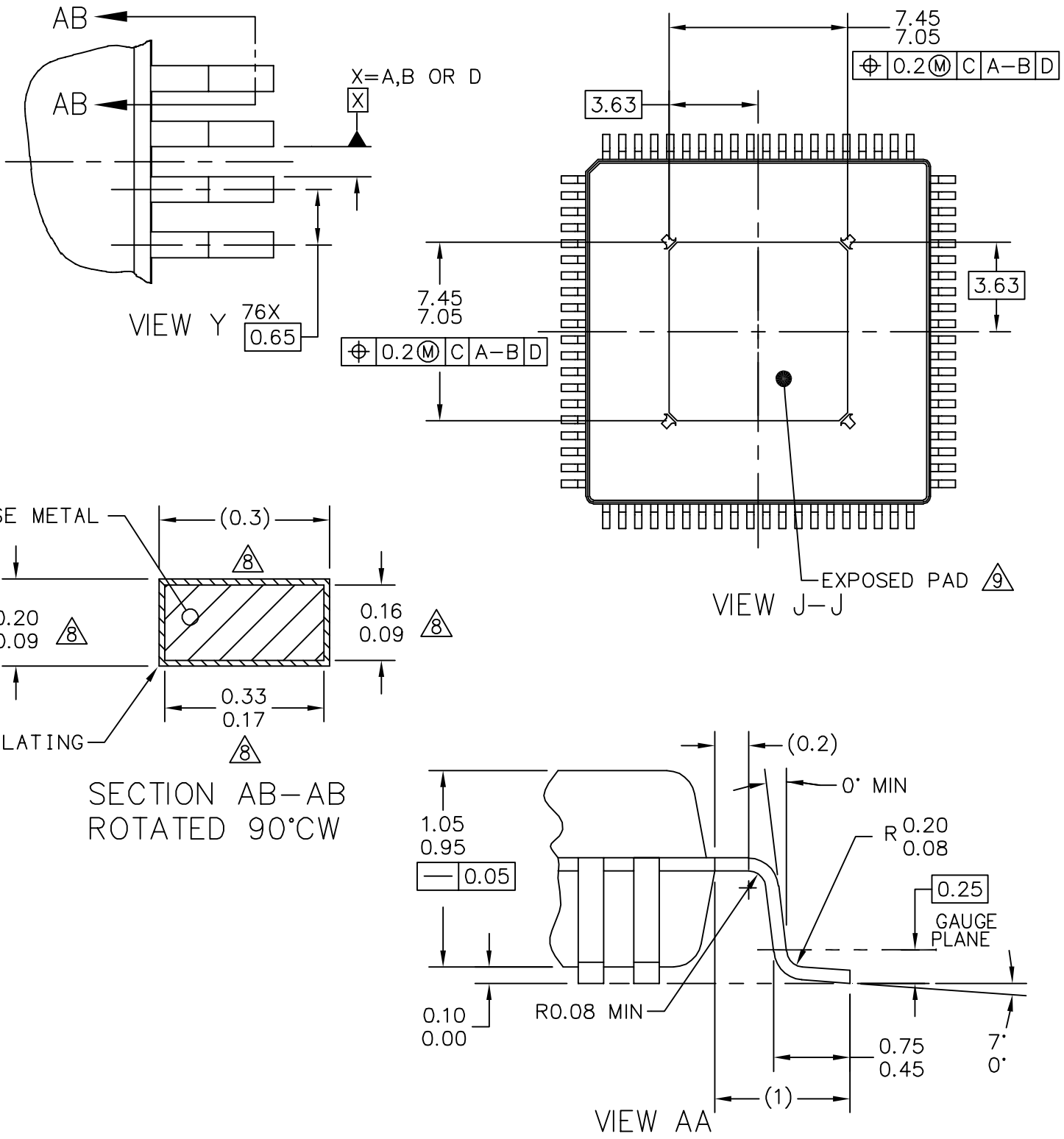


80LD TQFP,
14 X 14 X 1.0 PKG,
0.65 PITCH, EXPO

= NUMBER: 1355-01

STANDARD: JEDEC MS-026 AEC

PACKAGE CO



80LD TQFP,
14 X 14 X 1.0 PKG,
0.65 PITCH, EXPOS

BER: 1355-01
STANDARD: JEDEC MS-026 AEC-HD
PACKAGE CODE 6061

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.
2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
3. DATUMS A, B AND D TO BE DETERMINED AT DATUM PLANE H.
4. DIMENSIONS TO BE DETERMINED AT SEATING PLANE C
5. THIS DIMENSION DOES NOT INCLUDE DAMBER PROTRUSION. ALLOWABLE DAMBER PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE UPPER LIMIT BY MORE THAN 0.08 mm. DAMBER CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07 mm.
6. THIS DIMENSION DOES NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 mm PER SIDE. THIS DIMENSION IS MAXIMUM PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH.
7. EXACT SHAPE OF EACH CORNER IS OPTIONAL.
8. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.10 mm AND 0.25 mm FROM THE LEAD TIP.
9. AT LEAST 80% OF THE EXPOSED PAD AREA SHOULD BE FREE OF RESIN.

How to Reach Us:

Home Page:
www.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 480-768-2130

Europe, Middle East, and Africa:
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)

Japan:
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-0047, Japan
0120-191014 or +81-3-3440-3569

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
852-26668334

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

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.

MC9S12NE64V1
Rev. 1.1
06/2006

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View MC9S12NE64CPVE 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