# THE DATASHEET OF

# S9S12VR48AF0MLCR

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

# MC9S12VR-Family Reference Manual

***S12 Microcontrollers***

MC9S12VRRMV3

Rev. 3.11

October 10, 2013

freescale.com

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/

A full list of family members and options is included in the device overview section.

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

This document contains information for all constituent modules, with the exception of the CPU. For CPU information please refer to **CPU12-1** in the **CPU12 & CPU12X Reference Manual**.

## Revision History

| Date | Revision Level | Description |
|---|---|---|
| 14-August-2013 | Rev 3.9 | • Updated Equation for Jitter J(N) Appendix E |
| 16-August-2013 | Rev 3.10 | • Changed HBM Test Condition Number of Pulse per Pin from 3 to 1 Table A-3 |
| 10-October-2013 | Rev 3.11 | • Added Num 4 VSENSE input voltage to Table A-2<br>• Added $R_{EXT_HVI}$ to HVI input voltage Table A-2<br>• Moved Num 2 & 3 voltage differences from Table C-1 to Table A-5<br>• Removed Num 1 & 4 from Table C-1 VRH/VRL connected to VDDA/VSSA<br>• Added $T_J$ to Table B-1 Num 12<br>• Removed Table C-4 3.3V conversion performence<br>• Added HSDRV Static and Dynamic Characteristics for 5V operation<br>• Updated S12 Clock, Reset and Power Management Unit (S12CPMU_UHV)<br>• Updated LIN Physical Layer (S12LINPHYV2)<br>• Updated 64 KByte Flash Module (S12FTMRG64K512V1)<br>• Updated Device Overview MC9S12VR-Family |

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.

Chapter 1 Device Overview MC9S12VR-Family . . . . . .21
Chapter 2 Port Integration Module (S12VRPIMV2) . . . . . .51
Chapter 3 S12G Memory Map Controller (S12GMMCV1) . . . . . .103
Chapter 4 S12 Clock, Reset and Power Unit (S12CPMU_UHV) . . . . . .115
Chapter 5 Background Debug Module (S12SBDMV1) . . . . . .173
Chapter 6 S12S Debug Module (S12DBGV2). . . . . .197
Chapter 7 Interrupt Module (S12SINTV1). . . . . .241
Chapter 8 Analog-to-Digital Converter (ADC12B6CV2) . . . . . .249
Chapter 9 Pulse-Width Modulator (S12PWM8B8CV2) . . . . . .273
Chapter 10 Serial Communication Interface (S12SCIV6) . . . . . .303
Chapter 11 Serial Peripheral Interface (S12SPIV5) . . . . . .343
Chapter 12 Timer Module (TIM16B4CV3). . . . . .369
Chapter 13 High-Side Drivers - HSDRV (S12HSDRVV2). . . . . .387
Chapter 14 Low-Side Drivers - LSDRV (S12LSDRV1). . . . . .397
Chapter 15 LIN Physical Layer (S12LINPHYV2) . . . . . .409
Chapter 16 Supply Voltage Sensor - (BATSV2). . . . . .431
Chapter 17 64 KByte Flash Module (S12FTMRG64K512V1). . . . . .443
Appendix A MCU Electrical Specifications. . . . . .495
Appendix B VREG Electrical Specifications. . . . . .509
Appendix C ATD Electrical Specifications . . . . . .511
Appendix D HSDRV Electrical Specifications. . . . . .517
Appendix E PLL Electrical Specifications . . . . . .519
Appendix F IRC Electrical Specifications. . . . . .523
Appendix G LINPHY Electrical Specifications . . . . . .525
Appendix H LSDRV Electrical Specifications. . . . . .529
Appendix I BATS Electrical Specifications . . . . . .531
Appendix J PIM Electrical Specifications. . . . . .535
Appendix K SPI Electrical Specifications . . . . . .537

Appendix L XOSCLCP Electrical Specifications . . . . . .543
Appendix M FTMRG Electrical Specifications . . . . . .545
Appendix N Package Information . . . . . .549
Appendix O Ordering Information . . . . . .555
Appendix P Detailed Register Address Map. . . . . .557

# Chapter 1
# Device Overview MC9S12VR-Family


1.1 Introduction . . . 21
1.2 Features . . . 22
1.2.1 MC9S12VR-Family Member Comparison . . . 22
1.3 Chip-Level Features . . . 23
1.4 Module Features . . . 24
1.4.1 HCS12 16-Bit Central Processor Unit (CPU) . . . 24
1.4.2 On-Chip Flash with ECC . . . 24
1.4.3 On-Chip SRAM . . . 24
1.4.4 Main External Oscillator (XOSCLCP) . . . 25
1.4.5 Internal RC Oscillator (IRC) . . . 25
1.4.6 Internal Phase-Locked Loop (IPLL) . . . 25
1.4.7 Clock and Power Management Unit (CPMU) . . . 25
1.4.8 System Integrity Support . . . 25
1.4.9 Timer (TIM) . . . 26
1.4.10 Pulse Width Modulation Module (PWM) . . . 26
1.4.11 LIN physical layer transceiver (LINPHY) . . . 26
1.4.12 Serial Peripheral Interface Module (SPI) . . . 26
1.4.13 Serial Communication Interface Module (SCI) . . . 27
1.4.14 Analog-to-Digital Converter Module (ATD) . . . 27
1.4.15 Supply Voltage Sense (BATS) . . . 27
1.4.16 On-Chip Voltage Regulator system (VREG) . . . 27
1.4.17 Low-side drivers (LSDRV) . . . 28
1.4.18 High-side drivers (HSDRV) . . . 28
1.4.19 Background Debug (BDM) . . . 28
1.4.20 Debugger (DBG) . . . 28
1.5 Block Diagram . . . 29
1.6 Family Memory Map . . . 30
1.6.1 Part ID Assignments . . . 33
1.7 Signal Description and Device Pinouts . . . 33
1.7.1 Pin Assignment Overview . . . 33
1.7.2 Detailed Signal Descriptions . . . 34
1.7.3 Power Supply Pins . . . 37
1.8 Device Pinouts . . . 38
1.8.1 Pinout 48-pin LQFP . . . 39
1.8.2 Pinout 32-pin LQFP . . . 40
1.9 Modes of Operation . . . 43
1.9.1 Chip Configuration Summary . . . 43
1.9.2 Low Power Operation . . . 43
1.10 Security . . . 44
1.11 Resets and Interrupts . . . 44
1.11.1 Resets . . . 44
1.11.2 Interrupt Vectors . . . 45
1.11.3 Effects of Reset . . . 47

1.12 API external clock output (API_EXTCLK) . . . 47
1.13 COP Configuration . . . 48
1.14 Flash IFR Mapping . . . 48
1.15 ADC External Trigger Input Connection . . . 49
1.16 ADC Special Conversion Channels . . . 49
1.17 ADC Result Reference . . . 49

## Chapter 2
## Port Integration Module (S12VRPIMV2)

2.1 Introduction . . . 51
2.1.1 Overview . . . 51
2.1.2 Features . . . 52
2.2 External Signal Description . . . 52
2.3 Memory Map and Register Definition . . . 55
2.3.1 Register Map . . . 56
2.3.2 Register Descriptions . . . 59
2.3.3 Port E Data Register (PORTE) . . . 61
2.3.4 Port E Data Direction Register (DDRE) . . . 61
2.3.5 Port E, BKGD pin Pull Control Register (PUCR) . . . 62
2.3.6 ECLK Control Register (ECLKCTL) . . . 62
2.3.7 PIM Miscellaneous Register (PIMMISC) . . . 63
2.3.8 IRQ Control Register (IRQCR) . . . 63
2.3.9 Reserved Register . . . 64
2.3.10 Port T Data Register (PTT) . . . 64
2.3.11 Port T Input Register (PTIT) . . . 65
2.3.12 Port T Data Direction Register (DDRT) . . . 66
2.3.13 Port T Pull Device Enable Register (PERT) . . . 67
2.3.14 Port T Polarity Select Register (PPST) . . . 67
2.3.15 Module Routing Register 0 (MODRR0) . . . 68
2.3.16 Module Routing Register 1 (MODRR1) . . . 69
2.3.17 Port S Data Register (PTS) . . . 69
2.3.18 Port S Input Register (PTIS) . . . 71
2.3.19 Port S Data Direction Register (DDRS) . . . 71
2.3.20 Port S Pull Device Enable Register (PERS) . . . 72
2.3.21 Port S Polarity Select Register (PPSS) . . . 73
2.3.22 Port S Wired-Or Mode Register (WOMS) . . . 73
2.3.23 Module Routing Register 2 (MODRR2) . . . 74
2.3.24 Port P Data Register (PTP) . . . 76
2.3.25 Port P Input Register (PTIP) . . . 78
2.3.26 Port P Data Direction Register (DDRP) . . . 79
2.3.27 Port P Reduced Drive Register (RDRP) . . . 80
2.3.28 Port P Pull Device Enable Register (PERP) . . . 80
2.3.29 Port P Polarity Select Register (PPSP) . . . 81
2.3.30 Port P Interrupt Enable Register (PIEP) . . . 81
2.3.31 Port P Interrupt Flag Register (PIFP) . . . 82

2.3.32 Port L Input Register (PTIL) . . . 83
2.3.33 Port L Digital Input Enable Register (DIENL) . . . 83
2.3.34 Port L Analog Access Register (PTAL) . . . 84
2.3.35 Port L Input Divider Ratio Selection Register (PIRL) . . . 86
2.3.36 Port L Polarity Select Register (PPSL) . . . 86
2.3.37 Port L Interrupt Enable Register (PIEL) . . . 87
2.3.38 Port L Interrupt Flag Register (PIFL) . . . 87
2.3.39 Port AD Data Register (PT1AD) . . . 87
2.3.40 Port AD Input Register (PTI1AD) . . . 88
2.3.41 Port AD Data Direction Register (DDR1AD) . . . 89
2.3.42 Port AD Pull Enable Register (PER1AD) . . . 89
2.3.43 Port AD Polarity Select Register (PPS1AD) . . . 90
2.3.44 Port AD Interrupt Enable Register (PIE1AD) . . . 90
2.3.45 Port AD Interrupt Flag Register (PIF1AD) . . . 91
2.4 Functional Description . . . 92
2.4.1 General . . . 92
2.4.2 Registers . . . 92
2.4.3 Pins and Ports . . . 94
2.4.4 Interrupts . . . 97
2.5 Initialization and Application Information . . . 99
2.5.1 Port Data and Data Direction Register writes . . . 99
2.5.2 ADC External Triggers ETRIG1-0 . . . 99
2.5.3 Over-Current Protection on EVDD . . . 100
2.5.4 Open Input Detection on HVI Pins . . . 100

## Chapter 3
## S12G Memory Map Controller (S12GMMCV1)

3.1 Introduction . . . 103
3.1.1 Glossary . . . 103
3.1.2 Overview . . . 103
3.1.3 Features . . . 104
3.1.4 Modes of Operation . . . 104
3.1.5 Block Diagram . . . 104
3.2 External Signal Description . . . 105
3.3 Memory Map and Registers . . . 105
3.3.1 Module Memory Map . . . 105
3.3.2 Register Descriptions . . . 106
3.4 Functional Description . . . 110
3.4.1 MCU Operating Modes . . . 110
3.4.2 Memory Map Scheme . . . 110
3.4.3 Unimplemented and Reserved Address Ranges . . . 114
3.4.4 Prioritization of Memory Accesses . . . 114
3.4.5 Interrupts . . . 114

# Chapter 4
# S12 Clock, Reset and Power Management Unit (S12CPMU_UHV)


4.1 Introduction . . . 115
4.1.1 Features . . . 116
4.1.2 Modes of Operation . . . 118
4.1.3 S12CPMU_UHV Block Diagram . . . 121
4.2 Signal Description . . . 123
4.2.1 $\overline{\text{RESET}}$ . . . 123
4.2.2 EXTAL and XTAL . . . 123
4.2.3 VSUP — Regulator Power Input Pin . . . 123
4.2.4 VDDA, VSSA — Regulator Reference Supply Pins . . . 123
4.2.5 VDDX, VSSX— Pad Supply Pins . . . 123
4.2.6 VSS— Ground Pin . . . 124
4.2.7 $_{\text{API_EXTCLK}}$ — API external clock output pin . . . 124
4.2.8 $_{\text{VDD}}$— Internal Regulator Output Supply (Core Logic) . . . 124
4.2.9 $_{\text{VDDF}}$— Internal Regulator Output Supply (NVM Logic) . . . 124
4.2.10 TEMPSENSE — Internal Temperature Sensor Output Voltage . . . 124
4.3 Memory Map and Registers . . . 125
4.3.1 Module Memory Map . . . 125
4.3.2 Register Descriptions . . . 127
4.4 Functional Description . . . 160
4.4.1 Phase Locked Loop with Internal Filter (PLL) . . . 160
4.4.2 Startup from Reset . . . 162
4.4.3 Stop Mode using PLLCLK as source of the Bus Clock . . . 162
4.4.4 Full Stop Mode using Oscillator Clock as source of the Bus Clock . . . 163
4.4.5 External Oscillator . . . 164
4.4.6 System Clock Configurations . . . 165
4.5 Resets . . . 166
4.5.1 General . . . 166
4.5.2 Description of Reset Operation . . . 166
4.5.3 Power-On Reset (POR) . . . 169
4.5.4 Low-Voltage Reset (LVR) . . . 169
4.6 Interrupts . . . 170
4.6.1 Description of Interrupt Operation . . . 170
4.7 Initialization/Application Information . . . 172
4.7.1 General Initialization information . . . 172
4.7.2 Application information for COP and API usage . . . 172


# Chapter 5
# Background Debug Module (S12SBDMV1)


5.1 Introduction . . . 173
5.1.1 Features . . . 173
5.1.2 Modes of Operation . . . 174
5.1.3 Block Diagram . . . 175

5.2 External Signal Description . . . . . 175
5.3 Memory Map and Register Definition . . . . . 175
5.3.1 Module Memory Map . . . . . 175
5.3.2 Register Descriptions . . . . . 176
5.3.3 Family ID Assignment . . . . . 179
5.4 Functional Description . . . . . 179
5.4.1 Security . . . . . 180
5.4.2 Enabling and Activating BDM . . . . . 180
5.4.3 BDM Hardware Commands . . . . . 181
5.4.4 Standard BDM Firmware Commands . . . . . 182
5.4.5 BDM Command Structure . . . . . 183
5.4.6 BDM Serial Interface . . . . . 185
5.4.7 Serial Interface Hardware Handshake Protocol . . . . . 188
5.4.8 Hardware Handshake Abort Procedure . . . . . 190
5.4.9 SYNC — Request Timed Reference Pulse . . . . . 193
5.4.10 Instruction Tracing . . . . . 193
5.4.11 Serial Communication Time Out . . . . . 194

## Chapter 6
## S12S Debug Module (S12DBGV2)

6.1 Introduction . . . . . 197
6.1.1 Glossary Of Terms . . . . . 197
6.1.2 Overview . . . . . 198
6.1.3 Features . . . . . 198
6.1.4 Modes of Operation . . . . . 199
6.1.5 Block Diagram . . . . . 199
6.2 External Signal Description . . . . . 200
6.3 Memory Map and Registers . . . . . 200
6.3.1 Module Memory Map . . . . . 200
6.3.2 Register Descriptions . . . . . 201
6.4 Functional Description . . . . . 218
6.4.1 S12DBGV2 Operation . . . . . 218
6.4.2 Comparator Modes . . . . . 219
6.4.3 Match Modes (Forced or Tagged) . . . . . 223
6.4.4 State Sequence Control . . . . . 224
6.4.5 Trace Buffer Operation . . . . . 225
6.4.6 Tagging . . . . . 231
6.4.7 Breakpoints . . . . . 232
6.5 Application Information . . . . . 234
6.5.1 State Machine scenarios . . . . . 234
6.5.2 Scenario 1 . . . . . 234
6.5.3 Scenario 2 . . . . . 234
6.5.4 Scenario 3 . . . . . 235
6.5.5 Scenario 4 . . . . . 235
6.5.6 Scenario 5 . . . . . 237

6.5.7 Scenario 6 . . . 237
6.5.8 Scenario 7 . . . 237
6.5.9 Scenario 8 . . . 238
6.5.10 Scenario 9 . . . 238
6.5.11 Scenario 10 . . . 238

## Chapter 7
## Interrupt Module (S12SINTV1)

7.1 Introduction . . . 241
7.1.1 Glossary . . . 241
7.1.2 Features . . . 241
7.1.3 Modes of Operation . . . 242
7.1.4 Block Diagram . . . 242
7.2 External Signal Description . . . 243
7.3 Memory Map and Register Definition . . . 243
7.3.1 Register Descriptions . . . 243
7.4 Functional Description . . . 244
7.4.1 S12S Exception Requests . . . 244
7.4.2 Interrupt Prioritization . . . 244
7.4.3 Reset Exception Requests . . . 245
7.4.4 Exception Priority . . . 245
7.5 Initialization/Application Information . . . 246
7.5.1 Initialization . . . 246
7.5.2 Interrupt Nesting . . . 246
7.5.3 Wake Up from Stop or Wait Mode . . . 246

## Chapter 8
## Analog-to-Digital Converter (ADC12B6CV2)

8.1 Introduction . . . 249
8.1.1 Features . . . 249
8.1.2 Modes of Operation . . . 250
8.1.3 Block Diagram . . . 251
8.2 Signal Description . . . 252
8.2.1 Detailed Signal Descriptions . . . 252
8.3 Memory Map and Register Definition . . . 252
8.3.1 Module Memory Map . . . 252
8.3.2 Register Descriptions . . . 254
8.4 Functional Description . . . 269
8.4.1 Analog Sub-Block . . . 269
8.4.2 Digital Sub-Block . . . 269
8.5 Resets . . . 271
8.6 Interrupts . . . 271

## Chapter 9
## Pulse-Width Modulator (S12PWM8B8CV2)


9.1 Introduction . . . 273
9.1.1 Features . . . 273
9.1.2 Modes of Operation . . . 273
9.1.3 Block Diagram . . . 274
9.2 External Signal Description . . . 274
9.2.1 PWM7 - PWM0 — PWM Channel 7 - 0 . . . 275
9.3 Memory Map and Register Definition . . . 275
9.3.1 Module Memory Map . . . 275
9.3.2 Register Descriptions . . . 275
9.4 Functional Description . . . 290
9.4.1 PWM Clock Select . . . 290
9.4.2 PWM Channel Timers . . . 293
9.5 Resets . . . 301
9.6 Interrupts . . . 302


## Chapter 10
## Serial Communication Interface (S12SCIV6)


10.1 Introduction . . . 303
10.1.1 Glossary . . . 303
10.1.2 Features . . . 304
10.1.3 Modes of Operation . . . 304
10.1.4 Block Diagram . . . 305
10.2 External Signal Description . . . 306
10.2.1 TXD — Transmit Pin . . . 306
10.2.2 RXD — Receive Pin . . . 306
10.3 Memory Map and Register Definition . . . 306
10.3.1 Module Memory Map and Register Definition . . . 306
10.3.2 Register Descriptions . . . 307
10.4 Functional Description . . . 319
10.4.1 Infrared Interface Submodule . . . 320
10.4.2 LIN Support . . . 320
10.4.3 Data Format . . . 321
10.4.4 Baud Rate Generation . . . 322
10.4.5 Transmitter . . . 323
10.4.6 Receiver . . . 328
10.4.7 Single-Wire Operation . . . 336
10.4.8 Loop Operation . . . 337
10.5 Initialization/Application Information . . . 337
10.5.1 Reset Initialization . . . 337
10.5.2 Modes of Operation . . . 338
10.5.3 Interrupt Operation . . . 338
10.5.4 Recovery from Wait Mode . . . 341

10.5.5 Recovery from Stop Mode . . . 341

## Chapter 11
## Serial Peripheral Interface (S12SPIV5)

11.1 Introduction . . . 343
11.1.1 Glossary of Terms . . . 343
11.1.2 Features . . . 343
11.1.3 Modes of Operation . . . 343
11.1.4 Block Diagram . . . 344
11.2 External Signal Description . . . 345
11.2.1 MOSI — Master Out/Slave In Pin . . . 345
11.2.2 MISO — Master In/Slave Out Pin . . . 345
11.2.3 $\overline{SS}$ — Slave Select Pin . . . 346
11.2.4 SCK — Serial Clock Pin . . . 346
11.3 Memory Map and Register Definition . . . 346
11.3.1 Module Memory Map . . . 346
11.3.2 Register Descriptions . . . 347
11.4 Functional Description . . . 355
11.4.1 Master Mode . . . 356
11.4.2 Slave Mode . . . 357
11.4.3 Transmission Formats . . . 358
11.4.4 SPI Baud Rate Generation . . . 363
11.4.5 Special Features . . . 364
11.4.6 Error Conditions . . . 365
11.4.7 Low Power Mode Options . . . 366

## Chapter 12
## Timer Module (TIM16B4CV3)

12.1 Introduction . . . 369
12.1.1 Features . . . 369
12.1.2 Modes of Operation . . . 370
12.1.3 Block Diagrams . . . 370
12.2 External Signal Description . . . 371
12.2.1 IOC3 - IOC0 — Input Capture and Output Compare Channel 3-0 . . . 371
12.3 Memory Map and Register Definition . . . 371
12.3.1 Module Memory Map . . . 371
12.3.2 Register Descriptions . . . 371
12.4 Functional Description . . . 383
12.4.1 Prescaler . . . 384
12.4.2 Input Capture . . . 385
12.4.3 Output Compare . . . 385
12.5 Resets . . . 386
12.6 Interrupts . . . 386
12.6.1 Channel [3:0] Interrupt (C[3:0]F) . . . 386

12.6.2 Timer Overflow Interrupt (TOF) . . . . . 386

## Chapter 13
## High-Side Drivers - HSDRV (S12HSDRVV2)

13.1 Introduction . . . . . 387
13.1.1 Features . . . . . 387
13.1.2 Modes of Operation . . . . . 387
13.1.3 Block Diagram . . . . . 388
13.2 External Signal Description . . . . . 389
13.2.1 HS0, HS1— High Side Driver Pins . . . . . 389
13.2.2 VSUPHS — High Side Driver Power Pin . . . . . 389
13.3 Memory Map and Register Definition . . . . . 389
13.3.1 Module Memory Map . . . . . 389
13.3.2 Register Definition . . . . . 391
13.3.3 Port HS Data Register (HSDR) . . . . . 391
13.3.4 HSDRV Configuration Register (HSCR) . . . . . 392
13.3.5 Reserved Register . . . . . 393
13.3.6 HSDRV Interrupt Enable Register (HSIE) . . . . . 394
13.3.7 HSDRV Interrupt Flag Register (HSIF) . . . . . 395
13.4 Functional Description . . . . . 396
13.4.1 General . . . . . 396
13.4.2 Over-Current Shutdown . . . . . 396
13.4.3 Interrupts . . . . . 396

## Chapter 14
## Low-Side Drivers - LSDRV (S12LSDRV1)

14.1 Introduction . . . . . 397
14.1.1 Features . . . . . 397
14.1.2 Modes of Operation . . . . . 398
14.1.3 Block Diagram . . . . . 398
14.2 External Signal Description . . . . . 399
14.2.1 LS0, LS1— Low Side Driver Pins . . . . . 399
14.2.2 LSGND — Low Side Driver Ground Pin . . . . . 399
14.3 Memory Map and Register Definition . . . . . 399
14.3.1 Module Memory Map . . . . . 399
14.3.2 Register Definition . . . . . 401
14.3.3 Port LS Data Register (LSDR) . . . . . 401
14.3.4 LSDRV Configuration Register (LSCR) . . . . . 402
14.3.5 LSDRV Status Register (LSSR) . . . . . 403
14.3.6 LSDRV Interrupt Enable Register (LSIE) . . . . . 404
14.3.7 LSDRV Interrupt Flag Register (LSIF) . . . . . 405
14.4 Functional Description . . . . . 406
14.4.1 General . . . . . 406
14.4.2 Open-Load Detection . . . . . 406

14.4.3 Over-Current Detection . . . . . 406
14.4.4 Interrupts . . . . . 406

## Chapter 15
## LIN Physical Layer (S12LINPHYV2)

15.1 Introduction . . . . . 409
15.1.1 Features . . . . . 409
15.1.2 Modes of Operation . . . . . 410
15.1.3 Block Diagram . . . . . 411
15.2 External Signal Description . . . . . 412
15.2.1 LIN — LIN Bus Pin . . . . . 412
15.2.2 LGND — LIN Ground Pin . . . . . 412
15.2.3 VLINSUP — Positive Power Supply . . . . . 412
15.2.4 LPTxD — LIN Transmit Pin . . . . . 412
15.2.5 LPRxD — LIN Receive Pin . . . . . 412
15.3 Memory Map and Register Definition . . . . . 413
15.3.1 Module Memory Map . . . . . 413
15.3.2 Register Descriptions . . . . . 414
15.4 Functional Description . . . . . 421
15.4.1 General . . . . . 421
15.4.2 Slew Rate and LIN Mode Selection . . . . . 421
15.4.3 Modes . . . . . 422
15.4.4 Interrupts . . . . . 425
15.5 Application Information . . . . . 428
15.5.1 Module Initialization . . . . . 428
15.5.2 Interrupt handling in Interrupt Service Routine (ISR) . . . . . 428

## Chapter 16
## Supply Voltage Sensor - (BATSV2)

16.1 Introduction . . . . . 431
16.1.1 Features . . . . . 431
16.1.2 Modes of Operation . . . . . 431
16.1.3 Block Diagram . . . . . 432
16.2 External Signal Description . . . . . 432
16.2.1 VSENSE — Supply (Battery) Voltage Sense Pin . . . . . 432
16.2.2 VSUP — Voltage Supply Pin . . . . . 433
16.3 Memory Map and Register Definition . . . . . 433
16.3.1 Register Summary . . . . . 433
16.3.2 Register Descriptions . . . . . 434
16.4 Functional Description . . . . . 439
16.4.1 General . . . . . 439
16.4.2 Interrupts . . . . . 440

## Chapter 17
## 64 KByte Flash Module (S12FTMRG64K512V1)


17.1 Introduction . . . 443
17.1.1 Glossary . . . 443
17.1.2 Features . . . 444
17.1.3 Block Diagram . . . 445
17.2 External Signal Description . . . 446
17.3 Memory Map and Registers . . . 446
17.3.1 Module Memory Map . . . 446
17.3.2 Register Descriptions . . . 450
17.4 Functional Description . . . 470
17.4.1 Modes of Operation . . . 470
17.4.2 IFR Version ID Word . . . 470
17.4.3 Internal NVM resource (NVMRES) . . . 471
17.4.4 Flash Command Operations . . . 471
17.4.5 Allowed Simultaneous P-Flash and EEPROM Operations . . . 476
17.4.6 Flash Command Description . . . 477
17.4.7 Interrupts . . . 491
17.4.8 Wait Mode . . . 491
17.4.9 Stop Mode . . . 492
17.5 Security . . . 492
17.5.1 Unsecuring the MCU using Backdoor Key Access . . . 492
17.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM . . . 493
17.5.3 Mode and Security Effects on Flash Command Availability . . . 493
17.6 Initialization . . . 494


## Appendix A
## MCU Electrical Specifications


A.1 General . . . 495
A.1.1 Parameter Classification . . . 495
A.1.2 Pins . . . 496
A.1.3 Current Injection . . . 497
A.1.4 Absolute Maximum Ratings . . . 497
A.1.5 ESD Protection and Latch-up Immunity . . . 499
A.1.6 Operating Conditions . . . 501
A.1.7 Power Dissipation and Thermal Characteristics . . . 501
A.1.8 I/O Characteristics . . . 505
A.1.9 Supply Currents . . . 506

## Appendix B
## VREG Electrical Specifications

## Appendix C
## ATD Electrical Specifications


C.1 ATD Operating Characteristics . . . 511
C.2 Factors Influencing Accuracy . . . 511
C.2.1 Port AD Output Drivers Switching . . . 511
C.2.2 Source Resistance . . . 512
C.2.3 Source Capacitance . . . 512
C.2.4 Current Injection . . . 512
C.3 ATD Accuracy . . . 513
C.3.1 ATD Accuracy Definitions . . . 513
C.3.2 ATD Analog Input Parasitics . . . 516


## Appendix D
## HSDRV Electrical Specifications


D.1 Operating Characteristics . . . 517
D.2 Static Characteristics . . . 517
D.3 Dynamic Characteristics . . . 518


## Appendix E
## PLL Electrical Specifications


E.1 Reset, Oscillator and PLL . . . 519
E.1.1 Phase Locked Loop . . . 519


## Appendix F
## IRC Electrical Specifications

## Appendix G
## LINPHY Electrical Specifications


G.1 Maximum Ratings . . . 525
G.2 Static Electrical Characteristics . . . 525
G.3 Dynamic Electrical Characteristics . . . 527


## Appendix H
## LSDRV Electrical Specifications


H.1 Static Characteristics . . . 529
H.2 Dynamic Characteristics . . . 530


## Appendix I
## BATS Electrical Specifications


I.1 Maximum Ratings . . . 531

I.2 Static Electrical Characteristics . . . 532
I.3 Dynamic Electrical Characteristics . . . 533

## Appendix J
## PIM Electrical Specifications

J.1 High-Voltage Inputs (HVI) Electrical Characteristics . . . 535
J.2 Pin Interrupt Characteristics . . . 536

## Appendix K
## SPI Electrical Specifications

K.1 Timing . . . 537
K.1.1 Master Mode . . . 537
K.1.2 Slave Mode . . . 539

## Appendix L
## XOSCLCP Electrical Specifications

## Appendix M
## FTMRG Electrical Specifications

M.1 Timing Parameters . . . 545
M.1.1 NVM Reliability Parameters . . . 547
M.1.2 NVM Factory Shipping Condition . . . 547

## Appendix N
## Package Information

## Appendix O
## Ordering Information

## Appendix P
## Detailed Register Address Map

P.1 Detailed Register Map . . . 557

# Chapter 1
# Device Overview MC9S12VR-Family

## Revision History

| Version Number | Revision Date | Description of Changes |
|---|---|---|
| Rev 3.0 | 13-October-2012 | • Maskset and Part ID changed |
| Rev 3.1 | 3-July-2013 | • Added Section 1.14 Flash IFR Mapping |
| Rev 3.2 | 4-August-2013 | • Updated Section 1.17 ADC Result Reference |
| Rev 3.3 | 23-September-2013 | • Section 1.3 & 1.4.4 4-20 MHz XOSCLCP<br>• Section 1.7.2.16 Added VLINSUP description<br>• Section 1.4.5 Removed temperature range from feature list, this is specified in Appendix F<br>• Section 1.8.1 Added API_EXTCLK to pinout pin PS2<br>• Corrected Table 1-10 High Temperature Interrupt cannot wake up MCU from STOP<br>• Corrected Table 1-10 Local enable and wake up for SCI0 and SCI1 |

## 1.1 Introduction

The MC9S12VR-Family is an optimized automotive 16-bit microcontroller product line focused on low-cost, high-performance, and low pin-count. This family integrates an S12 microcontroller with a LIN Physical interface, a 5V regulator system to supply the microcontroller, and analog blocks to control other elements of the system which operate at vehicle battery level (e.g. relay drivers, high-side driver outputs, wake up inputs). The MC9S12VR-Family is targeted at generic automotive applications requiring single node LIN communications. Typical examples of these applications include window lift modules, seat modules and sun-roof modules to name a few.

The MC9S12VR-Family uses many of the same features found on the MC9S12G family, including error correction code (ECC) on flash memory, EEPROM for diagnostic or data storage, a fast analog-to-digital converter (ADC) and a frequency modulated phase locked loop (IPLL) that improves the EMC performance. The MC9S12VR-Family delivers an optimized solution with the integration of several key system components into a single device, optimizing system architecture and achieving significant space savings. The MC9S12VR-Family delivers all the advantages and efficiencies of a 16-bit MCU while retaining the low cost, power consumption, EMC, and code-size efficiency advantages currently enjoyed by users of Freescale's existing 8-bit and 16-bit MCU families. Like the MC9S12XS family, the MC9S12VR-Family will run 16-bit wide accesses without wait states for all peripherals and memories. Misaligned single cycle 16 bit RAM access is not supported. The MC9S12VR-Family will be available in 32-pin and 48-pin LQFP. In addition to the I/O ports available in each module, further I/O ports are available with interrupt capability allowing wake-up from stop or wait modes.

The MC9S12VR-Family is a general-purpose family of devices created with relay based motor control in mind and is suitable for a range of applications, including:

- Window lift modules
- Door modules
- Seat controllers
- Smart actuators
- Sun roof modules

## 1.2 Features

This section describes the key features of the MC9S12VR-Family.

### 1.2.1 MC9S12VR-Family Member Comparison

Table 1-1 provides a summary of different members of the MC9S12VR-Family and their features. This information is intended to provide an understanding of the range of functionality offered by this microcontroller family.

**Table 1-1. MC9S12VR - Family**

| Feature | MC9S12VR48 | MC9S12VR64 |
|---|---|---|
| CPU | HCS12 | |
| Flash memory (ECC) | 48 Kbytes | 64 Kbytes |
| EEPROM (ECC) | 512 Bytes | |
| RAM | 2 Kbytes | |
| LIN physical layer | 1 | |
| SPI | 1 | |
| SCI | Up to 2 | |
| Timer | 4ch x 16-bit | |
| PWM | 8ch x 8-bit or<br>4ch x 16-bit | |
| ADC | 6 ch x 10-bit available on external pins and four internal channels. see Table 1-13. | |
| Frequency modulated PLL | Yes | |
| Internal 1 MHz RC oscillator | Yes | |
| Autonomous window watchdog | 1 | |
| Low-side drivers<br>(protected for inductive loads) | 2 | |
| High-side drivers | Up to 2 | |
| High voltage Inputs | 4 | |
| General purpose I/Os (5V) | Up to 28 | |
| Direct battery sense pin | Yes | |

| Feature | MC9S12VR48 | MC9S12VR64 |
|---|---|---|
| Supply voltage sense | Yes | |
| Chip temperature sensor | 1 general sensor | |
| Supply voltage | $V_{SUP}$ = 6V – 18 V (normal operation)<br>up to 40V (protected operation) | |
| EVDD output current | 20mA @ 5V | |
| Maximum execution speed | 25 MHz | |
| Package | 32 LQFP<br>48 LQFP | |

## 1.3 Chip-Level Features

On-chip modules available within the family include the following features:

- HCS12 CPU core
- 64 or 48 Kbyte on-chip flash with ECC
- 512 byte EEPROM with ECC
- 2 Kbyte on-chip SRAM
- Phase locked loop (IPLL) frequency multiplier with internal filter
- 1 MHz internal RC oscillator with +/-1.3% accuracy over rated temperature range
- 4-20 MHz amplitude controlled pierce oscillator
- Internal COP (watchdog) module (with separate clock source)
- Timer module (TIM) supporting input/output channels that provide a range of 16-bit input capture, output compare and counter (up to 4 channels)
- Pulse width modulation (PWM) module (up to 8 x 8-bit channels)
- 10-bit resolution successive approximation analog-to-digital converter (ADC) with up to 6 channels available on external pins
- One serial peripheral interface (SPI) module
- One serial communication interface (SCI) module supporting LIN communications (with RX connected to a timer channel for internal oscillator calibration purposes, if desired)
- Up to one additional SCI (not connected to LIN physical layer)
- One on-chip LIN physical layer transceiver fully compliant with the LIN 2.2 standard & SAE J2602-2 LIN standard
- On-chip voltage regulator (VREG) for regulation of input supply and all internal voltages
- Autonomous periodic interrupt (API) (combination with cyclic, watchdog)
- Two protected low-side outputs to drive inductive loads
- Up to two protected high-side outputs
- 4 high-voltage inputs with wake-up capability and readable internally on ADC
- Up to two 10mA high-current outputs

- 20mA high-current output for use as Hall sensor supply
- Battery voltage sense with low battery warning, internally reverse battery protected
- Chip temperature sensor

## 1.4 Module Features

The following sections provide more details of the modules implemented on the MC9S12VR-Family.

### 1.4.1 HCS12 16-Bit Central Processor Unit (CPU)

The HCS12 CPU is a high-speed, 16-bit processing unit that has a programming model identical to that of the industry standard M68HC11 central processor unit (CPU).

- Full 16-bit data paths supports efficient arithmetic operation and high-speed math execution
- Supports instructions with odd byte counts, including many single-byte instructions. This allows much more efficient use of ROM space.
- Extensive set of indexed addressing capabilities, including:
  - Using the stack pointer as an indexing register in all indexed operations
  - Using the program counter as an indexing register in all but auto increment/decrement mode
  - Accumulator offsets using A, B, or D accumulators
  - Automatic index predecrement, preincrement, postdecrement, and postincrement (by –8 to +8)

### 1.4.2 On-Chip Flash with ECC

On-chip flash memory on the MC9S12VR features the following:

- 64 or 48 Kbyte of program flash memory
  - Automated program and erase algorithm
  - Protection scheme to prevent accidental program or erase
- 512 Byte EEPROM
  - 16 data bits plus 6 syndrome ECC (error correction code) bits allow single bit error correction and double fault detection
  - Erase sector size 4 bytes
  - Automated program and erase algorithm
  - User margin level setting for reads

### 1.4.3 On-Chip SRAM

- 2 Kbytes of general-purpose RAM

### 1.4.4 Main External Oscillator (XOSCLCP)

- Loop control Pierce oscillator using 4 MHz to 20 MHz crystal
  - Current gain control on amplitude output
  - Signal with low harmonic distortion
  - Low power
  - Good noise immunity
  - Eliminates need for external current limiting resistor
  - Transconductance sized for optimum start-up margin for typical crystals
  - Oscillator pins shared with GPIO functionality

### 1.4.5 Internal RC Oscillator (IRC)

- Factory trimmed internal reference clock
  - 1 MHz internal RC oscillator with ±1.3% accuracy over rated temperature range

### 1.4.6 Internal Phase-Locked Loop (IPLL)

- Phase-locked-loop clock frequency multiplier
  - No external components required
  - Reference divider and multiplier allow large variety of clock rates
  - Automatic bandwidth control mode for low-jitter operation
  - Automatic frequency lock detector
  - Configurable option to spread spectrum for reduced EMC radiation (frequency modulation)
  - Reference clock sources:
    - Internal 1 MHz RC oscillator (IRC)

### 1.4.7 Clock and Power Management Unit (CPMU)

- Real time interrupt (RTI)
- Clock monitor (CM)
- System reset generation

### 1.4.8 System Integrity Support

- Power-on reset (POR)
- Illegal address detection with reset
- Low-voltage detection with interrupt or reset
- Computer operating properly (COP) watchdog with option to run on internal RC oscillator
  - Configurable as window COP for enhanced failure detection
  - Can be initialized out of reset using option bits located in flash memory

- Clock monitor supervising the correct function of the oscillator

### 1.4.9 Timer (TIM)

- Up to 4 x 16-bit channels for input capture or output compare
- 16-bit free-running counter with 8-bit precision prescaler

### 1.4.10 Pulse Width Modulation Module (PWM)

- Up to eight 8-bit channels or reconfigurable four 16-bit channel PWM resolution
  - — Programmable period and duty cycle per channel
  - — Center-aligned or left-aligned outputs
  - — Programmable clock select logic with a wide range of frequencies

### 1.4.11 LIN physical layer transceiver (LINPHY)

- Compliant with LIN Physical Layer 2.2 specification.
- Compliant with the SAE J2602-2 LIN standard.
- Standby mode with glitch-filtered wake-up.
- Slew rate selection optimized for the baud rates: 10.4kBit/s, 20kBit/s and Fast Mode (up to 250kBit/s).
- Switchable 34kΩ/330kΩ pull-ups (in shutdown mode, 330kΩ only)
- Current limitation for LIN Bus pin falling edge.
- Over-current protection.
- LIN TxD-dominant timeout feature monitoring the LPTxD signal.
- Automatic transmitter shutdown in case of an over-current or TxD-dominant timeout.
- Fulfills the OEM “Hardware Requirements for LIN (CAN and FlexRay) Interfaces in Automotive Applications” v1.3.

### 1.4.12 Serial Peripheral Interface Module (SPI)

- Configurable 8- or 16-bit data size
- Full-duplex or single-wire bidirectional
- Double-buffered transmit and receive
- Master or slave
- MSB-first or LSB-first shifting
- Serial clock phase and polarity options

### 1.4.13 Serial Communication Interface Module (SCI)

- 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
- 16-bit baud rate selection
- Programmable character length
- Programmable polarity for transmitter and receiver
- Active edge receive wake-up
- Break detect and transmit collision detect supporting LIN
- Internal connection to one SCI routable to external pins

### 1.4.14 Analog-to-Digital Converter Module (ATD)

- Up to 6-channel, 10-bit analog-to-digital converter
  - 8-/10-bit resolution
  - 3 us, 10-bit single conversion time
  - Left or right justified result data
  - Internal oscillator for conversion in stop modes
  - Continuous conversion mode
  - Multiple channel scans
- Pins can also be used as digital I/O
- Up to 6 pins can be used as keyboard wake-up interrupt (KWI)
- Internal voltages monitored with the ATD module
  - $V_{SUP}$, $V_{SENSE}$, chip temperature sensor, high voltage inputs,$V_{RH}$, $V_{RL}$, $V_{DDF}$

### 1.4.15 Supply Voltage Sense (BATS)

- VSENSE & VSUP pin low or a high voltage interrupt
- VSENSE & VSUP pin can be routed via an internal divider to the internal ADC

### 1.4.16 On-Chip Voltage Regulator system (VREG)

- Voltage regulator
  - Linear voltage regulator directly supplied by VSUP (protected VBAT)
  - Low-voltage detect with low-voltage interrupt on $V_{SUP}$
  - Capable of supplying both the MCU internally and providing additional external current (approximately 20mA) to supply other components within the electronic control unit.
  - Over-temperature protection and interrupt
- Internal Voltage regulator
  - Linear voltage regulator with bandgap reference

— Low-voltage detect with low-voltage interrupt on VDDA
— Power-on reset (POR) circuit
— Low-voltage reset (LVR)

## 1.4.17 Low-side drivers (LSDRV)

- 2x low-side drivers targeted for up to approximately 150mA current capability.
- Internal timer or PWM channels can be routed to control the low-side drivers
- Open-load detection
- Over-current protection with shutdown and interrupt
- Active clamp (for driving relays)
- Recirculation detection

## 1.4.18 High-side drivers (HSDRV)

- 2 High-side drivers targeted for up to approximately 50mA current capability
- Internal timer or PWM channels can be routed to control the high-side drivers
- Over-current protection with shutdown and interrupt

## 1.4.19 Background Debug (BDM)

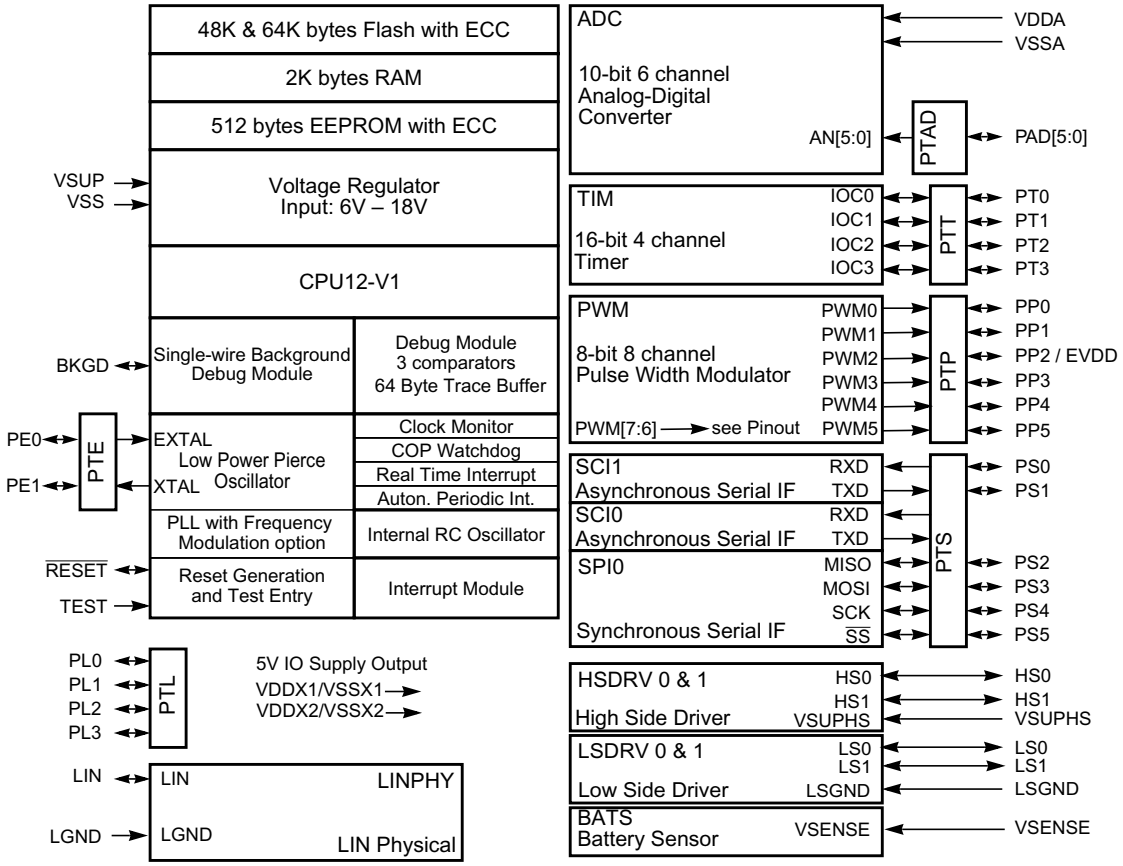
- Background debug module (BDM) with single-wire interface
  — Non-intrusive memory access commands
  — Supports in-circuit programming of on-chip nonvolatile memory

## 1.4.20 Debugger (DBG)

- Trace buffer with depth of 64 entries
- Three comparators (A, B and C)
  — Access address comparisons with optional data comparisons
  — Program counter comparisons
  — Exact address or address range comparisons
- Two types of comparator matches
  — Tagged This matches just before a specific instruction begins execution
  — Force This is valid on the first instruction boundary after a match occurs
- Four trace modes
- Four stage state sequencer

# 1.5 Block Diagram

**Figure 1-1. MC9S12VR Block Diagram**




Block Diagram shows the maximum configuration!
Not all pins or all peripherals are available on all devices and packages.
Rerouting options are not shown.

## 1.6 Family Memory Map

Table 1-2 shows the MC9S12VR-Family register memory map.

**Table 1-2. Device Register Memory Map**

| Address | Module | Size (Bytes) |
|---|---|---|
| 0x0000–0x0009 | PIM (port integration module) | 10 |
| 0x000A–0x000B | MMC (memory map control) | 2 |
| 0x000C–0x000D | PIM (port integration module) | 2 |
| 0x000E–0x000F | Reserved | 2 |
| 0x0010–0x0017 | MMC (memory map control) | 8 |
| 0x0018–0x0019 | Reserved | 2 |
| 0x001A–0x001B | Device ID register | 2 |
| 0x001C–0x001F | PIM (port integration module) | 4 |
| 0x0020–0x002F | DBG (debug module) | 16 |
| 0x0030–0x0033 | Reserved | 4 |
| 0x0034–0x003F | CPMU (clock and power management) | 12 |
| 0x0040–0x006F | TIM (timer module <= 4channels) | 48 |
| 0x0070–0x009F | ADC (analog to digital converter <= 6 channels) | 48 |
| 0x00A0–0x00C7 | PWM (pulse-width modulator <= 2channels) | 40 |
| 0x00C8–0x00CF | SCI0 (serial communication interface) | 8 |
| 0x00D0–0x00D7 | SCI1 (serial communication interface) | 8 |
| 0x00D8–0x00DF | SPI (serial peripheral interface) | 8 |
| 0x00E0–0x00FF | Reserved | 32 |
| 0x0100–0x0113 | FTMRG control registers | 20 |
| 0x0114–0x011F | Reserved | 12 |
| 0x0120 | INT (interrupt module) | 1 |
| 0x0121–0x013F | Reserved | 31 |
| 0x0140-0x0147 | HSDRV (high-side driver) | 8 |
| 0x0148-0x014F | Reserved | 8 |
| 0x0150-0x0157 | LSDRV (low-side driver) | 8 |
| 0x0158-0x015F | Reserved | 8 |
| 0x0160-0x0167 | LINPHY (LIN physical layer) | 8 |
| 0x0168-0x016F | Reserved | 8 |
| 0x0170-0x0177 | BATS (Supply Voltage Sense) | 8 |
| 0x0178–0x023F | Reserved | 200 |
| 0x0240–0x027F | PIM (port integration module) | 64 |

| Address | Module | Size (Bytes) |
|---|---|---|
| 0x0280–0x02EF | Reserved | 112 |
| 0x02F0–0x02FF | CPMU (clock and power management) | 16 |
| 0x0300–0x03FF | Reserved | 256 |

**NOTE**

Reserved register space shown in Table 1-2 is not allocated to any module. This register space is reserved for future use. Writing to these locations has no effect. Read access to these locations returns zero.

Figure 1-2 shows MC9S12VR-Family CPU and BDM local address translation to the global memory map as a graphical representation. The whole 256K global memory space is visible through the P-Flash window located in the 64k local memory map located at 0x8000 - 0xBFFF using the PPAGE register.

**NOTE**

Flash space on page 0xC in Figure 1-2 is not available on S12VR48. This is only available on S12VR64.

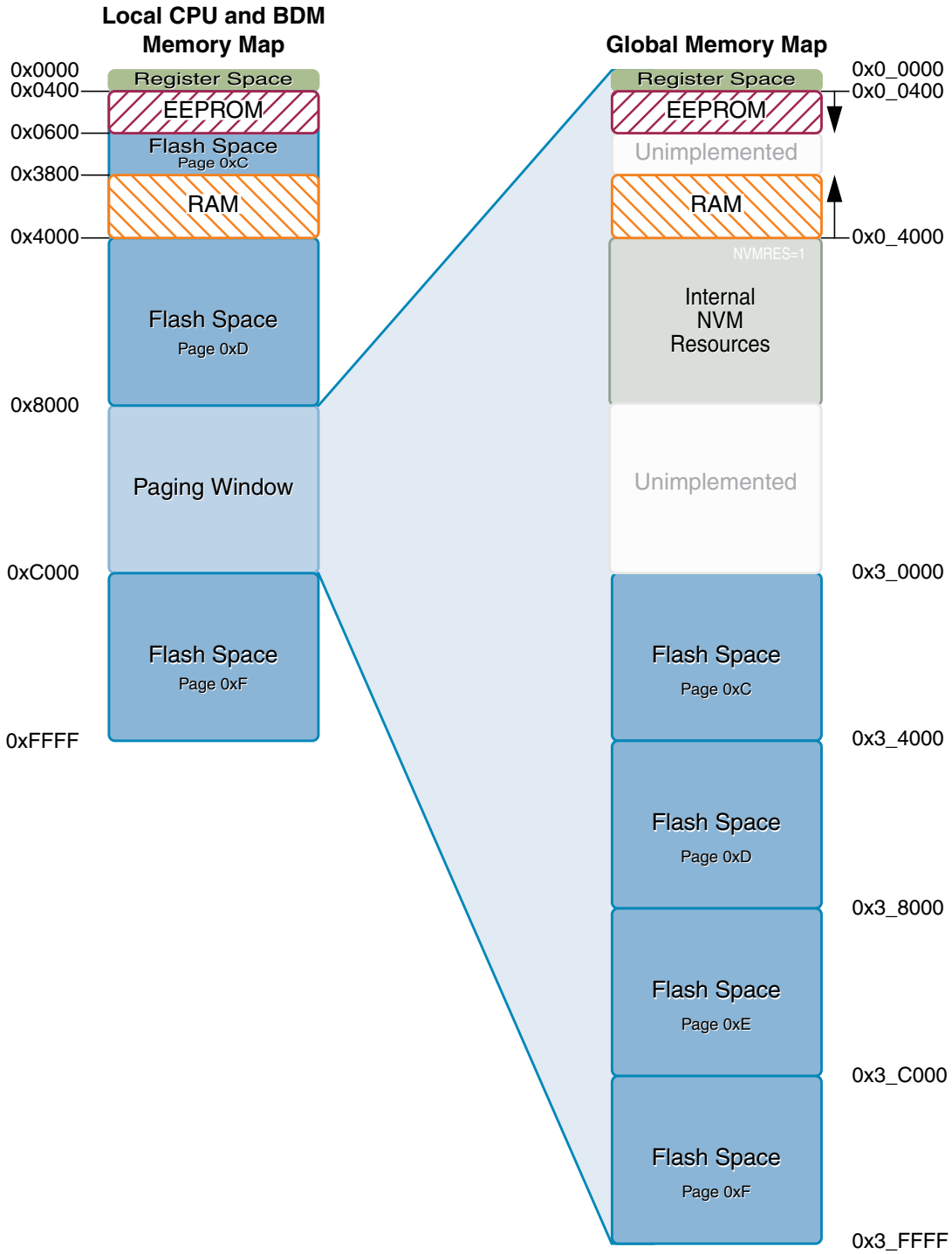



**Figure 1-2. MC9S12VR-Family Global Memory Map.**

### 1.6.1 Part ID Assignments

The part ID is located in two 8-bit registers PARTIDH and PARTIDL (addresses 0x001A and 0x001B). The read-only value is a unique part ID for each revision of the chip. Table 1-3 shows the assigned part ID number and mask set number.

**Table 1-3. Assigned Part ID Numbers**

| Device | Mask Set Number | Part ID |
|---|---|---|
| MC9S12VR48 | N59H | $3290 |
| MC9S12VR64 | N59H | $3290 |

## 1.7 Signal Description and Device Pinouts

This section describes signals that connect off-chip. It includes a pinout diagram, a table of signal properties, and detailed discussion of signals. It is built from the signal description sections of the individual IP blocks on the device.

### 1.7.1 Pin Assignment Overview

Table 1-4 provides a summary of which ports are available for 32-pin and 48-pin package option.

**Table 1-4. Port Availability by Package Option**

| Port | 32 LQFP | 48 LQFP |
|---|---|---|
| Port AD | PAD[1:0] | PAD[5:0] |
| Port E | PE[1:0] | PE[1:0] |
| Port P | PP[2:1] | PP[5:0] |
| Port S | PS[3:2] | PS[5:0] |
| Port T | PT[3:0] | PT[3:0] |
| Port L | PL[3:0] | PL[3:0] |
| sum of ports | 16 | 28 |
| I/O power pairs VDDX/VSSX | 1/1 | 2/2 |

**NOTE**

To avoid current drawn from floating inputs, all non-bonded pins should be configured as output or configured as input with a pull up or pull down device enabled

## 1.7.2 Detailed Signal Descriptions

This section describes the signal properties.

### 1.7.2.1 $\overline{\text{RESET}}$ — External Reset Signal

The $\overline{\text{RESET}}$ signal is an active low bidirectional control signal. It acts as an input to initialize the MCU to a known start-up state, and an output when an internal MCU function causes a reset. The $\overline{\text{RESET}}$ pin has an internal pull-up device.

### 1.7.2.2 TEST — Test Pin

This input only pin is reserved for factory test. This pin has an internal pull-down device.

**NOTE**

The TEST pin must be tied to ground in all applications.

### 1.7.2.3 BKGD / MODC — Background Debug Signal

The BKGD/MODC pin is used as a pseudo-open-drain pin for the background debug communication. It is used as a MCU operating mode select pin during reset. The state of this pin is latched to the MODC bit at the rising edge of $\overline{\text{RESET}}$. The BKGD pin has an internal pull-up device.

### 1.7.2.4 PAD[5:0] / KWAD[5:0] — Port AD Input Signals of ADC

PAD[5:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWAD[5:0]).These signals can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull devices are disabled.

### 1.7.2.5 PE[1:0] — Port E I/O Signals

PE[1:0] are general-purpose input or output signals. The signals can have pull-down device, enabled by a single control bit for this signal group. Out of reset the pull-down devices are enabled.

### 1.7.2.6 PP[5:0] / KWP[5:0] — Port P I/O Signals

PP[5:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWP[5:0]). PP[2] has a high current drive strength and an over-current interrupt feature. They can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull devices are disabled.

### 1.7.2.7 PS[5:0] — Port S I/O Signals

PS[5:0] are general-purpose input or output signals. They can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull-up devices are enabled.

#### 1.7.2.8 PT[3:0] — Port T I/O Signals

PT[3:0] are general-purpose input or output signals. They can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull devices are disabled.

#### 1.7.2.9 PL[3:0] / KWL[3:0] — Port L Input Signals

PL[3:0] are high voltage input ports. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWL[3:0]).

#### 1.7.2.10 LIN — LIN Physical Layer Signal

This pad is connected to the single-wire LIN data bus.

#### 1.7.2.11 HS[1:0] — High-Side Drivers Output Signals

Outputs of the two high-side drivers intended to drive incandescent bulbs or LEDs.

#### 1.7.2.12 LS[1:0] — Low-Side Drivers Output Signals

Outputs of the two low-side drivers intended to drive inductive loads (relays).

#### 1.7.2.13 VSENSE — Voltage Sensor Input

This pin can be connected to the supply (Battery) line for voltage measurements. The voltage present at this input is scaled down by an internal voltage divider, and can be routed to the internal ADC via an analog multiplexer. The pin itself is protected against reverse battery connections. To protect the pin from external fast transients an external resistor is needed.

#### 1.7.2.14 AN[5:0] — ADC Input Signals

AN[5:0] are the analog inputs of the Analog-to-Digital Converter.

#### 1.7.2.15 SPI Signals

##### 1.7.2.15.1 $\overline{SS}$ Signal

This signal is associated with the slave select SS functionality of the serial peripheral interface SPI.

##### 1.7.2.15.2 SCK Signal

This signal is associated with the serial clock SCK functionality of the serial peripheral interface SPI.

##### 1.7.2.15.3 MISO Signal

This signal is associated with the MISO functionality of the serial peripheral interface SPI. This signal acts as master input during master mode or as slave output during slave mode.

#### 1.7.2.15.4 MOSI Signal

This signal is associated with the MOSI functionality of the serial peripheral interface SPI. This signal acts as master output during master mode or as slave input during slave mode

### 1.7.2.16 LINPHY Signals

#### 1.7.2.16.1 VLINSUP — Positive Power Supply

This is the power supply to the LINPHY. The VLINSUP is connected to VSUP.

#### 1.7.2.16.2 LPTXD Signal

This signal is the LINPHY transmit input. See Figure 2-22

#### 1.7.2.16.3 LPRXD Signal

This signal is the LINPHY receive output. See Figure 2-22

### 1.7.2.17 SCI Signals

#### 1.7.2.17.1 RXD[1:0] Signals

Those signals are associated with the receive functionality of the serial communication interfaces SCI1-0.

#### 1.7.2.17.2 TXD[1:0] Signals

Those signals are associated with the transmit functionality of the serial communication interfaces SCI1-0.

### 1.7.2.18 PWM[7:0] Signals

The signals PWM[7:0] are associated with the PWM module outputs.

### 1.7.2.19 Internal Clock outputs

#### 1.7.2.19.1 ECLK

This signal is associated with the output of the divided bus clock (ECLK).

**NOTE**

This feature is only intended for debug purposes at room temperature.
It must not be used for clocking external devices in an application.

### 1.7.2.20 ETRIG[1:0]

These signals are inputs to the Analog-to-Digital Converter. Their purpose is to trigger ADC conversions.

#### 1.7.2.21 IOC[3:0] Signals

The signals IOC[3:0] are associated with the input capture or output compare functionality of the timer (TIM) module.

### 1.7.3 Power Supply Pins

MC9S12VR-Family power and ground pins are described below. 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 close to the MCU as possible.

**NOTE**

All ground pins must be connected together in the application.

#### 1.7.3.1 VDDX1, VDDX2, VSSX1, VSSX2 — Power Pins and Ground Pins

VDDX1 and VDDX2 are the 5V power supply output for the I/O drivers. This voltage is generated by the on chip voltage regulator. Bypass requirements on VDDX1 and VDDX2 pins depend on how heavily the MCU pins are loaded. All VDDX pins are connected together internally. All VSSX pins are connected together internally.

**NOTE**

Not all power and ground pins are available on all packages. Refer to pinout section for further details.

#### 1.7.3.2 VDDA, VSSA — Power Supply Pins for ADC

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

**NOTE**

The reference voltages VRH and VRL mentioned in Appendix C, “ATD Electrical Specifications are internally connected to VDDA and VSSA.

#### 1.7.3.3 VSS — Core Ground Pin

The voltage supply of nominally 1.8V is generated by the internal voltage regulator. The return current path is through the VSS pin.

#### 1.7.3.4 LGND — LINPHY Ground Pin

LGND is the the ground pin for the LIN physical layer LINPHY.

#### 1.7.3.5 LSGND — Ground Pin for Low-Side Drivers

LSGND is the shared ground pin for the low-side drivers.

### 1.7.3.6 VSUP — Voltage Supply Pin for Voltage Regulator

VSUP is the 12V/18V shared supply voltage pin for the on chip voltage regulator.

### 1.7.3.7 VSUPHS — Voltage Supply Pin for High-Side Drivers

VSUPHS is the 12V/18V shared supply voltage pin for the high-side drivers.

### 1.7.3.8 Power and Ground Connection Summary

**Table 1-5. Power and Ground Connection Summary**

| Mnemonic | Nominal Voltage | Description |
|---|---|---|
| VSS | 0V | Ground pin for 1.8V core supply voltage generated by on chip voltage regulator |
| VDDX1 | 5.0 V | 5V power supply output for I/O drivers generated by on chip voltage regulator |
| VSSX1 | 0V | Ground pin for I/O drivers |
| VDDX2 | 5.0 V | 5V power supply output for I/O drivers generated by on chip voltage regulator |
| VSSX2 | 0V | Ground pin for I/O drivers |
| VDDA | 5.0 V | External power supply for the analog-to-digital converter and for the reference circuit of the internal voltage regulator |
| VSSA | 0V | Ground pin for VDDA analog supply |
| LGND | 0V | Ground pin for LIN physical |
| LSGND | 0V | Ground pin for low-side driver |
| VSUP | 12V/18V | External power supply for voltage regulator |
| VSUPHS | 12V/18V | External power supply for high-side driver |

## 1.8 Device Pinouts

MC9S12VR-Familyis available in 48-pin package and 32-pin package. Signals in parentheses in **Figure 1-3.** and **Figure 1-4.** denote alternative module routing options.

## 1.8.1 Pinout 48-pin LQFP

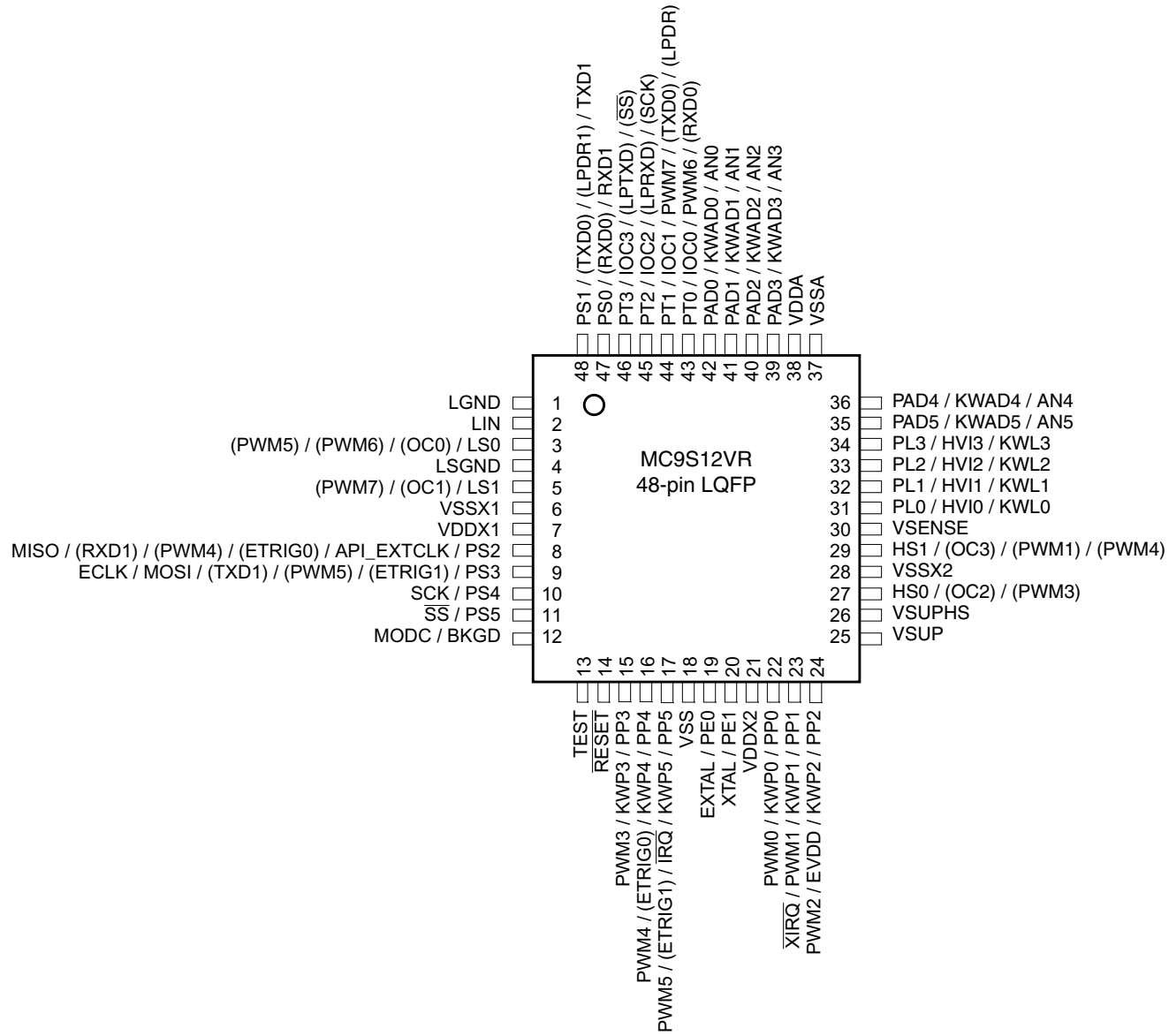



**Figure 1-3. MC9S12VR 48-pin LQFP pinout**

## 1.8.2 Pinout 32-pin LQFP

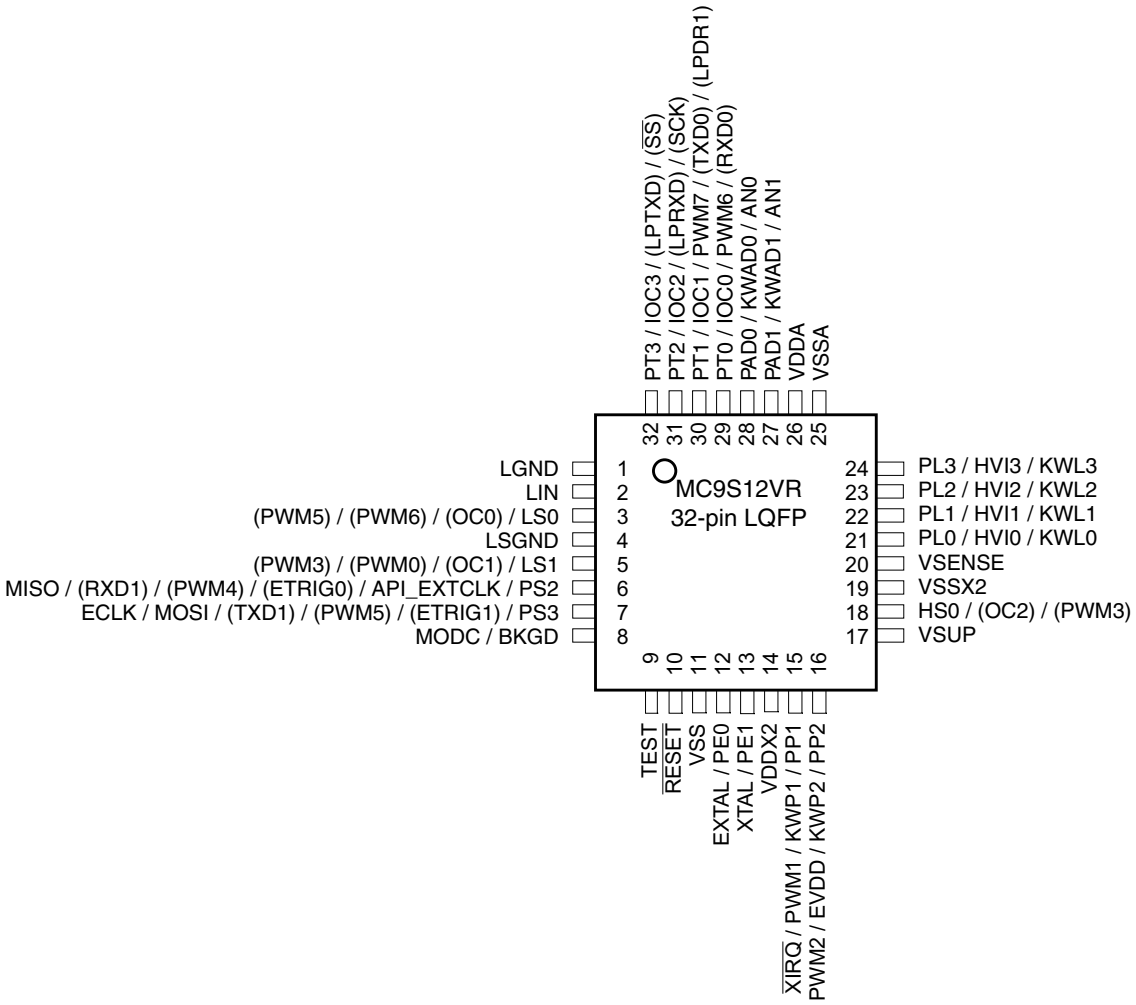



**Figure 1-4. MC9S12VR 32-pin LQFP pinout**

**Table 1-6. Pin Summary**

| Package | | | Function | | | | | Power Supply | Internal Pull Resistor | |
|---|---|---|---|---|---|---|---|---|---|---|
| 48 LQ FP | 32 LQ FP | Pin | 1th Func. | 2nd Func. | 3rd Func. | 4th Func. | 5th Func. | | CTRL | Reset State |
| 1 | 1 | LGND | — | — | — | — | — | — | — | — |
| 2 | 2 | LIN | — | — | — | — | — | — | — | — |
| 3 | 3 | LS0 | OC0[1] | PWM5 | PWM6 | — | — | — | — | — |
| 4 | 4 | LSGND | — | — | — | — | — | — | — | — |
| 5 | 5 | LS1 | OC1[2] | PWM7 | — | — | — | — | — | — |
| 6 | — | VSSX1 | — | — | — | — | — | — | — | — |
| 7 | — | VDDX1 | — | — | — | — | — | $V_{DDX}$ | — | — |
| 8 | 6 | PS2 | ETRIG0 | PWM4 | RXD1 | MISO | — | $V_{DDX}$ | PERS/PPSS | Up |
| 9 | 7 | PS3 | ETRIG1 | PWM5 | TXD1 | MOSI | ECLK | $V_{DDX}$ | PERS/PPSS | Up |
| 10 | — | PS4 | SCK | — | — | — | — | $V_{DDX}$ | PERS/PPSS | Up |
| 11 | — | PS5 | $\overline{SS}$ | — | — | — | — | $V_{DDX}$ | PERS/PPSS | Up |
| 12 | 8 | BKGD | MODC | — | — | — | — | $V_{DDX}$ | PUCR/BKPUE | Up |
| 13 | 9 | TEST | — | — | — | — | — | N.A | $\overline{RESET}$ pin | Down |
| 14 | 10 | $\overline{RESET}$ | — | — | — | — | — | $V_{DDX}$ | TEST pin | Up |
| 15 | — | PP3 | KWP3 | PWM3 | — | — | — | $V_{DDX}$ | PERP/PPSP | Disabled |
| 16 | — | PP4 | KWP4 | ETRIG0 | PWM4 | — | — | $V_{DDX}$ | PERP/PPSP | Disabled |
| 17 | — | PP5 | KWP5 | ETRIG1 | PWM5 | $\overline{IRQ}$ | — | $V_{DDX}$ | PERP/PPSP | Disabled |
| 18 | 11 | VSS | — | — | — | — | — | — | — | — |
| 19 | 12 | PE0 | EXTAL | — | — | — | — | $V_{DDX}$ | PUCR/PUPEE | Down |
| 20 | 13 | PE1 | XTAL | — | — | — | — | $V_{DDX}$ | PUCR/PUPEE | Down |
| 21 | 14 | VDDX2 | — | — | — | — | — | — | — | — |
| 22 | — | PP0 | KWP0 | PWM0 | — | — | — | $V_{DDX}$ | PERP/PPSP | Disabled |
| 23 | 15 | PP1 | KWP1 | PWM1 | $\overline{XIRQ}$ | — | — | $V_{DDX}$ | PERP/PPSP | Disabled |
| 24 | 16 | PP2 | KWP2 | EVDD | PWM2 | — | — | $V_{DDX}$ | PERP/PPSP | Disabled |
| 25 | 17 | VSUP | — | — | — | — | — | — | — | — |
| 26 | — | VSUPHS | — | — | — | — | — | — | — | — |
| 27 | 18 | HS0 | OC2[3] | PWM3 | — | — | — | $V_{SUPHS}$ | — | — |

| Package | | | Function | | | | | Power Supply | Internal Pull Resistor | |
|---|---|---|---|---|---|---|---|---|---|---|
| 48 LQ FP | 32 LQ FP | Pin | 1th Func. | 2nd Func. | 3rd Func. | 4th Func. | 5th Func. | | CTRL | Reset State |
| 28 | 19 | VSSX2 | — | — | — | — | — | — | — | — |
| 29 | — | HS1 | OC3[4] | PWM1 | PWM4 | — | — | $V_{SUPHS}$ | — | — |
| 30 | 20 | VSENSE | — | — | — | — | — | — | — | — |
| 31 | 21 | PL0 | HVI0 | KWL0 | — | — | — | $V_{DDX}$ | — | — |
| 32 | 22 | PL1 | HVI1 | KWL1 | — | — | — | $V_{DDX}$ | — | — |
| 33 | 23 | PL2 | HVI2 | KWL2 | — | — | — | $V_{DDX}$ | — | — |
| 34 | 24 | PL3 | HVI3 | KWL3 | — | — | — | $V_{DDX}$ | — | — |
| 35 | — | PAD5 | KWAD5 | AN5 | — | — | — | $V_{DDA}$ | PER1AD/ PPS1AD | Disabled |
| 36 | — | PAD4 | KWAD4 | AN4 | — | — | — | $V_{DDA}$ | PER1AD/ PPS1AD | Disabled |
| 37 | 25 | VSSA | — | — | — | — | — | — | — | — |
| 38 | 26 | VDDA | — | — | — | — | — | — | — | — |
| 39 | — | PAD3 | KWAD3 | AN3 | — | — | — | $V_{DDA}$ | PER1AD/ PPS1AD | Disabled |
| 40 | — | PAD2 | KWAD2 | AN2 | — | — | — | $V_{DDA}$ | PER1AD/ PPS1AD | Disabled |
| 41 | 27 | PAD1 | KWAD1 | AN1 | — | — | — | $V_{DDA}$ | PER1AD/ PPS1AD | Disabled |
| 42 | 28 | PAD0 | KWAD0 | AN0 | — | — | — | $V_{DDA}$ | PER1AD/ PPS1AD | Disabled |
| 43 | 29 | PT0 | IOC0 | PWM6 | RXD0 | — | — | $V_{DDX}$ | PERT/PPST | Disabled |
| 44 | 30 | PT1 | IOC1 | PWM7 | TXD0 | LPDR1 | — | $V_{DDX}$ | PERT/PPST | Disabled |
| 45 | 31 | PT2 | IOC2 | LPRXD | SCK | — | — | $V_{DDX}$ | PERT/PPST | Disabled |
| 46 | 32 | PT3 | IOC3 | LPTXD | $\overline{SS}$ | — | — | $V_{DDX}$ | PERT/PPST | Disabled |
| 47 | — | PS0 | RXD0 | RXD1 | — | — | — | $V_{DDX}$ | PERS/PPSS | Up |
| 48 | — | PS1 | TXD0 | LPDR1 | TXD1 | — | — | $V_{DDX}$ | PERS/PPSS | Up |

1 Timer Output Compare Channel 0
2 Timer Output Compare Channel 1
3 Timer Output Compare Channel 2
4 Timer Output Compare Channel 3

## 1.9 Modes of Operation

The MCU can operate in different modes. These are described in 1.9.1 Chip Configuration Summary.

The MCU can operate in different power modes to facilitate power saving when full system performance is not required. These are described in 1.9.2 Low Power Operation.

Some modules feature a software programmable option to freeze the module status whilst the background debug module is active to facilitate debugging.

### 1.9.1 Chip Configuration Summary

The different modes and the security state of the MCU affect the debug features (enabled or disabled).

The operating mode out of reset is determined by the state of the MODC signal during reset (see Table 1-7). The MODC bit in the MODE register shows the current operating mode and provides limited mode switching during operation. The state of the MODC signal is latched into this bit on the rising edge of $\overline{\text{RESET}}$.

**Table 1-7. Chip Modes**

| Chip Modes | MODC |
|---|---|
| Normal single chip | 1 |
| Special single chip | 0 |

#### 1.9.1.1 Normal Single-Chip Mode

This mode is intended for normal device operation. The opcode from the on-chip memory is being executed after reset (requires the reset vector to be programmed correctly). The processor program is executed from internal memory.

#### 1.9.1.2 Special Single-Chip Mode

This mode is used for debugging single-chip operation, boot-strapping, or security related operations. The background debug module BDM is active in this mode. The CPU executes a monitor program located in an on-chip ROM. BDM firmware waits for additional serial commands through the BKGD pin.

### 1.9.2 Low Power Operation

The MC9S12VR-Family has two dynamic-power modes (run and wait) and two static low-power modes stop and pseudo stop). For a detailed description refer to **Section Chapter 4 S12 Clock, Reset and Power Management Unit (S12CPMU_UHV)**.

- Dynamic power mode: Run
  - Run mode is the main full performance operating mode with the entire device clocked. The user can configure the device operating speed through selection of the clock source and the phase locked loop (PLL) frequency. To save power, unused peripherals must not be enabled.

- Dynamic power mode: Wait
  - This mode is entered when the CPU executes the WAI instruction. In this mode the CPU will not execute instructions. The internal CPU clock is switched off. All peripherals can be active in system wait mode. For further power consumption the peripherals can individually turn off their local clocks. Asserting $\overline{\text{RESET}}$, $\overline{\text{XIRQ}}$, $\overline{\text{IRQ}}$, or any other interrupt that is not masked ends system wait mode.
- Static power mode Pseudo-stop:
  - In this mode the system clocks are stopped but the oscillator is still running and the real time interrupt (RTI) and watchdog (COP), Autonomous Periodic Interrupt (API) and ATD modules may be enabled. Other peripherals are turned off. This mode consumes more current than system STOP mode but, as the oscillator continues to run, the full speed wake up time from this mode is significantly shorter.
- Static power mode: Stop
  - The oscillator is stopped in this mode. By default, all clocks are switched off and all counters and dividers remain frozen. The autonomous periodic interrupt (API), key wake-up and the SCI modules may be enabled to wake the device.

## 1.10 Security

The MCU security mechanism prevents unauthorized access to the Flash memory. Refer to **Section 5.4.1 Security** and **Section 17.5 Security**.

## 1.11 Resets and Interrupts

Consult the S12 CPU manual and the S12SINT section for information on exception processing.

### 1.11.1 Resets

Table 1-8. lists all Reset sources and the vector locations. Resets are explained in detail in the Chapter 4, “S12 Clock, Reset and Power Management Unit (S12CPMU_UHV)”.

**Table 1-8. Reset Sources and Vector Locations**

| Vector Address | Reset Source | CCR Mask | Local Enable |
|---|---|---|---|
| $FFFE | Power-On Reset (POR) | None | None |
| $FFFE | Low Voltage Reset (LVR) | None | None |
| $FFFE | External pin $\overline{\text{RESET}}$ | None | None |
| $FFFE | Illegal Address Reset | None | None |
| $FFFC | Clock monitor reset | None | OSCE Bit in CPMUOSC register |

| Vector Address | Reset Source | CCR Mask | Local Enable |
|---|---|---|---|
| $FFFA | COP watchdog reset | None | CR[2:0] in CPMUCOP register |

## 1.11.2 Interrupt Vectors

Table 1-9 lists all interrupt sources and vectors in the default order of priority. The interrupt module (see Chapter 7, "Interrupt Module (S12SINTV1)") provides an interrupt vector base register (IVBR) to relocate the vectors.

**Table 1-9. Interrupt Vector Locations (Sheet 1 of 3)**

| Vector Address | Interrupt Source | CCR Mask | Local Enable | Wake up from STOP | Wake up from WAIT |
|---|---|---|---|---|---|
| Vector base + $F8 | Unimplemented instruction trap | None | None | - | - |
| Vector base+ $F6 | SWI | None | None | - | - |
| Vector base+ $F4 | $\overline{\text{XIRQ}}$ | X Bit | None | Yes | Yes |
| Vector base+ $F2 | $\overline{\text{IRQ}}$ | I bit | IRQCR (IRQEN) | Yes | Yes |
| Vector base+ $F0 | RTI time-out interrupt | I bit | CPMUINT (RTIE) | **see Section 4.1.2.3 Stop Mode** | YES |
| Vector base+ $EE | TIM timer channel 0 | I bit | TIE (C0I) | No | Yes |
| Vector base + $EC | TIM timer channel 1 | I bit | TIE (C1I) | No | Yes |
| Vector base+ $EA | TIM timer channel 2 | I bit | TIE (C2I) | No | Yes |
| Vector base+ $E8 | TIM timer channel 3 | I bit | TIE (C3I) | No | Yes |
| Vector base+ $E6 to Vector base + $E0 | Reserved | | | | |
| Vector base+ $DE | TIM timer overflow | I bit | TSCR2(TOF) | No | Yes |
| Vector base+ $DC to Vector base + $DA | Reserved | | | | |
| Vector base + $D8 | SPI | I bit | SPICR1 (SPIE, SPTIE) | No | Yes |
| Vector base+ $D6 | SCI0 | I bit | SCI0CR2 (TIE, TCIE, RIE, ILIE) SCI0ACR1 (RXEDGIE, BERRIE, BKDIE) | RXEDGIF only | Yes |

**Table 1-9. Interrupt Vector Locations (Sheet 2 of 3)**

| Vector Address | Interrupt Source | CCR Mask | Local Enable | Wake up from STOP | Wake up from WAIT |
|---|---|---|---|---|---|
| Vector base + $D4 | SCI1 | I bit | SCI1CR2 (TIE, TCIE, RIE, ILIE) SCI1ACR1 (RXEDGIE, BERRIE, BKDIE) | RXEDGIF only | Yes |
| Vector base + $D2 | ADC | I bit | ATDCTL2 (ASCIE) | No | Yes |
| Vector base + $D0 | Reserved | | | | |
| Vector base + $CE | Port L | I bit | PIEL (PIEL3-PIEL0) | Yes | Yes |
| Vector base + $CC to Vector base + $CA | Reserved | | | | |
| Vector base + $C8 | Oscillator status interrupt | I bit | CPMUINT (OSCIE) | No | Yes |
| Vector base + $C6 | PLL lock interrupt | I bit | CPMUINT (LOCKIE) | No | Yes |
| Vector base + $C4 to Vector base + $BC | Reserved | | | | |
| Vector base + $BA | FLASH error | I bit | FERCNFG (SFDIE, DFDIE) | No | No |
| Vector base + $B8 | FLASH command | I bit | FCNFG (CCIE) | No | Yes |
| Vector base + $B6 to Vector base + $B0 | Reserved | | | | |
| Vector base + $AE | HSDRV over-current interrupt | I bit | HSIE (HSERR) | No | Yes |
| Vector base + $AC | LSDRV over-current interrupt | I bit | LSIE (LSERR) | No | Yes |
| Vector base + $AA | LINPHY over-current interrupt or Txd-dominant timeout interrupt | I bit | LPIE (LPDTIE & LPOCIE) | No | Yes |
| Vector base + $A8 | BATS low & high battery voltage interrupt | I bit | BATIE (BVHIE,BVLIE) | No | Yes |
| Vector base + $A6 to Vector base + $90 | Reserved | | | | |
| Vector base + $8E | Port P interrupt | I bit | PIEP (PIEP5-PIEP3, PIEP1-PIEP0) | Yes | Yes |
| Vector base+ $8C | Port P2 (EVDD Hall Sensor Supply) over-current interrupt | I bit | PIEP (OCIE) | No | Yes |
| Vector base + $8A | Low-voltage interrupt (LVI) | I bit | CPMUCTRL (LVIE) | No | Yes |
| Vector base + $88 | Autonomous periodical interrupt (API) | I bit | CPMUAPICTRL (APIE) | Yes | Yes |
| Vector base + $86 | High temperature interrupt | I bit | CPMUHTCTL(HTIE) | No | Yes |
| Vector base + $84 | ADC compare interrupt | I bit | ATDCTL2 (ACMPIE) | No | Yes |
| Vector base + $82 | Port AD interrupt | I bit | PIE1AD(PIE1AD5-PIE1AD0) | Yes | Yes |

**Table 1-9. Interrupt Vector Locations (Sheet 3 of 3)**

| Vector Address | Interrupt Source | CCR Mask | Local Enable | Wake up from STOP | Wake up from WAIT |
|---|---|---|---|---|---|
| Vector base + $80 | Spurious interrupt | — | None | - | - |

### 1.11.3 Effects of Reset

When a reset occurs, MCU registers and control bits are initialized. Refer to the respective block sections for register reset states.

On each reset, the Flash module executes a reset sequence to load Flash configuration registers.

#### 1.11.3.1 Flash Configuration Reset Sequence Phase

On each reset, the Flash module will hold CPU activity while loading Flash module registers from the Flash memory. If double faults are detected in the reset phase, Flash module protection and security may be active on leaving reset. This is explained in more detail in the Flash module Section 17.1, “Introduction”.

#### 1.11.3.2 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.

#### 1.11.3.3 I/O Pins

Refer to the PIM section for reset configurations of all peripheral module ports.

#### 1.11.3.4 RAM

The RAM arrays are not initialized out of reset.

## 1.12 API external clock output (API_EXTCLK)

The API_EXTCLK option which is described 4.3.2.15 Autonomous Periodical Interrupt Control Register (CPMUAPICTL) is available on PS2 on S12VR-Family.

## 1.13 COP Configuration

The COP time-out rate bits CR[2:0] and the WCOP bit in the CPMUCOP register at address 0x003C are loaded from the Flash configuration field byte at global address 0x3_FF0E during the reset sequence. See Table 1-10 and Table 1-11 for coding

**Table 1-10. Initial COP Rate Configuration**

| NV[2:0] in FOPT Register | CR[2:0] in COPCTL Register |
|---|---|
| 000 | 111 |
| 001 | 110 |
| 010 | 101 |
| 011 | 100 |
| 100 | 011 |
| 101 | 010 |
| 110 | 001 |
| 111 | 000 |

**Table 1-11. Initial WCOP Configuration**

| NV[3] in FOPT Register | WCOP in COPCTL Register |
|---|---|
| 1 | 0 |
| 0 | 1 |

## 1.14 Flash IFR Mapping

**Table 1-12. Flash IFR Mapping**

| IFR Byte Address | Target | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | F | E | D | C | B | A | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x405A - 0x405B | | | | | | | ADC Bandgap Reference[1] | | | | | | | | | |
| 0x40B8 - 0x40B9 | | | ACLKTR[5:0][2] | | | | | | | | | | HTTR[3:0][3] | | | |
| 0x40BA - 0x40BB | | TCTRIM[4:0][4] | | | | | IRCTRIM[9:0][4] | | | | | | | | | |

1 see Section 1.17 ADC Result Reference

2 see Section 4.3.2.16 Autonomous Clock Trimming Register (CPMUACLKTR)

3 see Section 4.3.2.19 High Temperature Trimming Register (CPMUHTTR)

4 see Section 4.3.2.20 S12CPMU_UHV IRC1M Trim Registers (CPMUIRCTRIMH / CPMUIRCTRIML)

## 1.15 ADC External Trigger Input Connection

The ADC module includes external trigger inputs ETRIG0, ETRIG1, ETRIG2, and ETRIG3. The external trigger allows the user to synchronize ADC conversion to external trigger events. ETRIG0 is connected to PP0 / PWM0 and ETRIG1 is connected to PP1 / PWM1. ETRIG2 and ETRIG3 are not used. ETRIG0 can be routed to PS2 and ETRIG1 can be routed to PS3.

## 1.16 ADC Special Conversion Channels

Whenever the ADC's Special Channel Conversion Bit (SC) in 8.3.2.6 ATD Control Register 5 (ATDCTL5) is set, it is capable of running conversion on a number of internal channels. Table 1-13 lists the internal sources which are connected to these special conversion channels.

**Table 1-13. Usage of ADC Special Conversion Channels**

| ATDCTL5 Register Bits | | | | | | Usage |
|---|---|---|---|---|---|---|
| SC | CD | CC | CB | CA | ADC Channel | |
| 1 | 0 | 0 | 0 | 1 | Internal_7 | Bandgap Voltage $V_{BG}$ or Chip temperature sensor $V_{HT}$ see Section 4.3.2.13 High Temperature Control Register (CPMUHTCTL) |
| 1 | 0 | 0 | 1 | 0 | Internal_0 | Flash Supply Voltage VDDF |
| 1 | 1 | 0 | 1 | 0 | Internal_4 | $V_{SENSE}$ or $V_{SUP}$ selectable in BATS module see Section 16.1.1 Features |
| 1 | 1 | 0 | 1 | 1 | Internal_5 | High voltage inputs Port L see Section 2.3.34 Port L Analog Access Register (PTAL) |

## 1.17 ADC Result Reference

MCUs of the MC9S12VR-Family are able to measure the internal bandgap reference voltage $V_{BG}$ with the analog digital converter. (see Table 1-13.) $V_{BG}$ is a constant voltage with a narrow distribution over temperature and external voltage supply. The ADC conversion result of $V_{BG}$ is provided at address 0x0_405A/0x0_405B in the NVM IFR for reference. The measurement conditions of the reference conversion are listed in Table 1-14. By measuring the voltage $V_{BG}$ and comparing the result to the reference value in the IFR it is possible to determine the reference voltage of the ADC $V_{RH}$ in the application environment:

$$V_{RH} = \frac{StoredReference}{ConvertedReference} \bullet 5V$$

The exact absolute value of an analog conversion can be determined as follows:

$$Result = ConvertedADInput \bullet \frac{StoredReference \bullet 5V}{ConvertedReference \bullet 2^n}$$

With:

| | |
|---|---|
| ConvertedADInput: | Result of the analog to digital conversion of the desired pin |
| ConvertedReference: | Result of channel “Internal_7” conversion |
| StoredReference: | Value in IFR locatio 0x0_405A/0x0_405B |
| n: | ADC resolution (10 bit) |

**CAUTION**

The ADC’s reference voltage $V_{RH}$ must must remain at a constant level throughout the conversion process.

The reference voltage $V_{BG}$ is measured under the conditions shown in Table 1-14.. The value stored in the IFR is the average of eight consecutive conversions at $T_J$=150°C and eight consecutive conversions at $T_J$=-40°C.

**Table 1-14. Measurement Conditions for $V_{BG}$ ADC Conversion[1]**

| Description | Symbol | Value | Unit |
|---|---|---|---|
| Regulator supply voltage | $V_{SUP}$ | 5 | V |
| I/O supply voltage | $V_{DDX}$ | 5 | V |
| Analog supply voltage | $V_{DDA}$ | 5 | V |
| ADC clock | $f_{adcclk}$ | 6.25 | MHz |
| ADC sample time | $t_{smp}$ | 4 | ADC clock cycles |
| Bus frequency | $f_{bus}$ | 25 | MHz |
| Junction temperature | $T_j$ | -40 .. 150 | °C |
| Code execution | from RAM | | |

1 Voltage Regulator bypassed , VDDX and VDDA supplied from tester

# Chapter 2
# Port Integration Module (S12VRPIMV2)

## Revision History

| Rev. No. (Item No.) | Date (Submitted By) | Sections Affected | Substantial Change(s) |
|---|---|---|---|
| V02.04 | 24 May 2012 | | • Corrected PTAENL bit value in PTIL bit description |
| V02.05 | 22 May 2013 | Table 2-48 | • Added note to HVI stop mode |
| V02.06 | 05 Sep 2013 | Table 2-48 | • Undid last change. Removed reference to RPM mode.<br>• |

## 2.1 Introduction

### 2.1.1 Overview

The S12VR port integration module (PIM) establishes the interface between the peripheral modules and the I/O pins for all ports. It controls the electrical pin properties as well as the signal prioritization and multiplexing on shared pins.

This section covers:

- 2-pin port E associated with the external oscillator
- 4-pin port T associated with 4 TIM channels and 2 PWM channels
- 6-pin port S associated with 2 SCI and 1 SPI
- 6-pin port P with pin interrupts and wakeup function; associated with
  - — $\overline{\text{IRQ}}$, $\overline{\text{XIRQ}}$ interrupt inputs
  - — Six PWM channels with two of those capable of driving up to 10 mA
  - — One output with over-current protection and interrupt capable of supplying up to 20 mA to external devices such as Hall sensors
- 6-pin port AD with pin interrupts and wakeup function; associated with 6 ADC channels
- 4-pin port L with pin interrupts and wakeup function; associated with 4 high-voltage inputs for digital or analog use with optional voltage divider bypass and open input detection

Most I/O pins can be configured by register bits to select data direction and to enable and select pullup or pulldown devices.

### 2.1.2 Features

The PIM includes these distinctive registers:

- Data registers and data direction registers for Ports E, T, S, P and AD when used as general-purpose I/O
- Control registers to enable/disable pull devices and select pullups/pulldowns on Ports T, S, P, AD on per-pin basis
- Single control register to enable/disable pullups on Port E on per-port basis and on BKGD pin
- Control registers to enable/disable open-drain (wired-or) mode on Port S
- Control register to enable/disable reduced output drive on Port P high-current pins
- Interrupt flag register for pin interrupts on Port P, L and AD
- Control register to configure $\overline{\text{IRQ}}$ pin operation
- Control register to enable ECLK clock output
- Routing registers to support module port relocation and control internal module routings:
  - PWM and ETRIG to alternative pins
  - SPI $\overline{\text{SS}}$ and SCK to alternative pins
  - SCI1 to alternative pins
  - HSDRV and LSDRV control selection from PWM, TIM or related register bit
  - Various SCI0-LINPHY routing options supporting standalone use and conformance testing
  - Optional LINPHY to TIM link
  - Optional HVI to ADC link

A standard port pin has the following minimum features:

- Input/output selection
- 5 V output drive
- 5 V digital and analog input
- Input with selectable pullup or pulldown device

Optional features supported on dedicated pins:

- Two selectable output drive strengths
- Open drain for wired-or connections
- Interrupt input with glitch filtering
- High-voltage input
- 10 mA high-current output
- 20 mA high-current output with over-current protection for use as Hall sensor supply

## 2.2 External Signal Description

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

Table 2-1 shows all the pins and their functions that are controlled by the PIM. Routing options are denoted in parenthesis.

**NOTE**

If there is more than one function associated with a pin, the **output** priority is indicated by the position in the table from top (highest priority) to bottom (lowest priority).

**Table 2-1. Pin Functions and Priorities**

| Port | Pin Name | Pin Function & Priority[1] | I/O | Description | Pin Function after Reset |
|---|---|---|---|---|---|
| - | BKGD | MODC[2] | I | MODC input during $\overline{\text{RESET}}$ | BKGD |
| | | BKGD | I/O | BDM communication pin | |
| E | PE1 | XTAL | - | CPMU OSC signal | GPIO |
| | | PTE[1] | I/O | General-purpose | |
| | PE0 | EXTAL | - | CPMU OSC signal | |
| | | PTE[0] | I/O | General-purpose | |
| T | PT3 | ($\overline{\text{SS}}$) | I/O | SPI slave select | GPIO |
| | | (LPTXD) | I | LINPHY transmit pin | |
| | | IOC3 | I/O | TIM channel 3 | |
| | | PTT[3] | I/O | General-purpose | |
| | PT2 | (SCK) | I/O | SPI serial clock | |
| | | (LPRXD) | O | LINPHY receive pin | |
| | | IOC2 | I/O | TIM channel 2 | |
| | | PTT[2] | I/O | General-purpose | |
| | PT1 | (LPDR1) | O | LINPHY register LPDR[LPDR1] | |
| | | (TXD0) | I/O | Serial Communication Interface 0 transmit pin | |
| | | PWM7 | O | Pulse Width Modulator channel 7 | |
| | | IOC1 | I/O | TIM channel 1 | |
| | | PTT[1] | I/O | General-purpose | |
| | PT0 | (RXD0) | I | Serial Communication Interface 0 receive pin | |
| | | PWM6 | O | Pulse Width Modulator channel 6 | |
| | | IOC0 | I/O | TIM channel 0 | |
| | | PTT[0] | I/O | General-purpose | |

| Port | Pin Name | Pin Function & Priority[1] | I/O | Description | Pin Function after Reset |
|---|---|---|---|---|---|
| S | PS5 | $\overline{SS}$ | I/O | SPI slave select | GPIO |
| | | PTS[5] | I/O | General-purpose | |
| | PS4 | SCK | I/O | SPI serial clock | |
| | | PTS[4] | I/O | General-purpose | |
| | PS3 | ECLK | O | Free running clock | |
| | | MOSI | I/O | SPI master out / slave in | |
| | | (TXD1) | I/O | Serial Communication Interface 1 transmit pin | |
| | | (PWM5) | O | Pulse Width Modulator channel 5 | |
| | | (ETRIG1) | I | ADC external trigger input | |
| | | PTS[3] | I/O | General-purpose | |
| | PS2 | MISO | I/O | SPI master in / slave out | |
| | | (RXD1) | I | Serial Communication Interface 1 receive pin | |
| | | (PWM4) | O | Pulse Width Modulator channel 4 | |
| | | (ETRIG0) | I | ADC external trigger input | |
| | | PTS[2] | I/O | General-purpose | |
| | PS1 | TXD1 | I/O | Serial Communication Interface 1 transmit pin | |
| | | (LPDR1) | O | LINPHY register LPDR[LPDR1] | |
| | | (TXD0) | I/O | Serial Communication Interface 0 transmit pin | |
| | | PTS[1] | I/O | General-purpose | |
| | PS0 | RXD1 | I | Serial Communication Interface 1 receive pin | |
| | | (RXD0) | I | Serial Communication Interface 0 receive pin | |
| | | PTS[0] | I/O | General-purpose | |

| Port | Pin Name | Pin Function & Priority[1] | I/O | Description | Pin Function after Reset |
|---|---|---|---|---|---|
| P | PP5 | $\overline{IRQ}$ | I | Maskable level- or falling edge-sensitive interrupt | GPIO |
| | | PWM5 | O | Pulse Width Modulator channel 5 | |
| | | ETRIG1 | I | ADC external trigger input | |
| | | PTP[5]/ KWP[5] | I/O | General-purpose; with pin interrupt and wakeup | |
| | PP4 | PWM4 | O | Pulse Width Modulator channel 4 | |
| | | ETRIG0 | I | ADC external trigger input | |
| | | PTP[4]/ KWP[4] | I/O | General-purpose; with pin interrupt and wakeup | |
| | PP3 | PWM3 | O | Pulse Width Modulator channel 3 | |
| | | PTP[3]/ KWP[3] | I/O | General-purpose; with pin interrupt and wakeup | |
| | PP2 | PWM2 | O | Pulse Width Modulator channel 2 | |
| | | PTP[2]/ KWP[2]/ EVDD | I/O | General-purpose; with pin interrupt and wakeup; switchable external power supply output with over-current interrupt; high-current capable (20 mA) | |
| | PP1 | $\overline{XIRQ}$ | I | Non-maskable level-sensitive interrupt | |
| | | PWM1 | O | Pulse Width Modulator channel 1; high-current capable (10 mA) | |
| | | PTP[1]/ KWP[1] | I/O | General-purpose; with interrupt and wakeup; high-current capable (10 mA) | |
| | PP0 | PWM0 | O | Pulse Width Modulator channel 0; high-current capable (10 mA) | |
| | | PTP[0]/ KWP[0] | I/O | General-purpose; with interrupt and wakeup; high-current capable (10 mA) | |
| L | PL3-0 | PTL[3:0]/ KWL[3:0] | I | General-purpose high-voltage input (HVI); with interrupt and wakeup; optional ADC link | GPI (HVI) |
| AD | PAD5-0 | AN[5:0] | I | ADC analog | GPIO |
| | | PTAD[5:0]/ KWAD[5:0] | I/O | General-purpose; with interrupt and wakeup | |

1 Signals in parentheses denote alternative module routing pins

2 Function active when $\overline{RESET}$ asserted

## 2.3 Memory Map and Register Definition

This section provides a detailed description of all PIM registers.

## 2.3.1 Register Map

| Global Address | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0000–0x0007 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0008 | PORTE | R | 0 | 0 | 0 | 0 | 0 | 0 | PE1 | PE0 |
| | | W | | | | | | | | |
| 0x0009 | DDRE | R | 0 | 0 | 0 | 0 | 0 | 0 | DDRE1 | DDRE0 |
| | | W | | | | | | | | |
| 0x000A–0x000B | Non-PIM Address Range | R | Non-PIM Address Range | | | | | | | |
| | | W | | | | | | | | |
| 0x000C | PUCR | R | 0 | BKPUE | 0 | PDPEE | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000D | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000E–0x001B | Non-PIM Address Range | R | Non-PIM Address Range | | | | | | | |
| | | W | | | | | | | | |
| 0x001C | ECLKCTL | R | NECLK | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x001D | PIMMISC | R | OCPE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x001E | IRQCR | R | IRQE | IRQEN | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x001F | Reserved | R | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| | | W | | | | | | | | |
| 0x0020–0x023F | Non-PIM Address Range | R | Non-PIM Address Range | | | | | | | |
| | | W | | | | | | | | |
| 0x0240 | PTT | R | 0 | 0 | 0 | 0 | PTT3 | PTT2 | PTT1 | PTT0 |
| | | W | | | | | | | | |
| 0x0241 | PTIT | R | 0 | 0 | 0 | 0 | PTIT3 | PTIT2 | PTIT1 | PTIT0 |
| | | W | | | | | | | | |
| 0x0242 | DDRT | R | 0 | 0 | 0 | 0 | DDRT3 | DDRT2 | DDRT1 | DDRT0 |
| | | W | | | | | | | | |

| Global Address | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0243 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0244 | PERT | R | 0 | 0 | 0 | 0 | PERT3 | PERT2 | PERT1 | PERT0 |
| | | W | | | | | | | | |
| 0x0245 | PPST | R | 0 | 0 | 0 | 0 | PPST3 | PPST2 | PPST1 | PPST0 |
| | | W | | | | | | | | |
| 0x0246 | MODRR0 | R | MODRR07 | MODRR06 | MODRR05 | MODRR04 | MODRR03 | MODRR02 | MODRR01 | MODRR00 |
| | | W | | | | | | | | |
| 0x0247 | MODRR1 | R | 0 | 0 | MODRR15 | MODRR14 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0248 | PTS | R | 0 | 0 | PTS5 | PTS4 | PTS3 | PTS2 | PTS1 | PTS0 |
| | | W | | | | | | | | |
| 0x0249 | PTIS | R | 0 | 0 | PTIS5 | PTIS4 | PTIS3 | PTIS2 | PTIS1 | PTIS0 |
| | | W | | | | | | | | |
| 0x024A | DDRS | R | 0 | 0 | DDRS5 | DDRS4 | DDRS3 | DDRS2 | DDRS1 | DDRS0 |
| | | W | | | | | | | | |
| 0x024B | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x024C | PERS | R | 0 | 0 | PERS5 | PERS4 | PERS3 | PERS2 | PERS1 | PERS0 |
| | | W | | | | | | | | |
| 0x024D | PPSS | R | 0 | 0 | PPSS5 | PPSS4 | PPSS3 | PPSS2 | PPSS1 | PPSS0 |
| | | W | | | | | | | | |
| 0x024E | WOMS | R | 0 | 0 | WOMS5 | WOMS4 | WOMS3 | WOMS2 | WOMS1 | WOMS0 |
| | | W | | | | | | | | |
| 0x024F | MODRR2 | R | MODRR27 | 0 | MODRR25 | MODRR24 | MODRR23 | MODRR22 | MODRR21 | MODRR20 |
| | | W | | | | | | | | |
| 0x0250–0x0257 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0258 | PTP | R | 0 | 0 | PTP5 | PTP4 | PTP3 | PTP2 | PTP1 | PTP0 |
| | | W | | | | | | | | |
| 0x0259 | PTIP | R | 0 | 0 | PTIP5 | PTIP4 | PTIP3 | PTIP2 | PTIP1 | PTIP0 |
| | | W | | | | | | | | |

| Global Address | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x025A | DDRP | R | 0 | 0 | DDRP5 | DDRP4 | DDRP3 | DDRP2 | DDRP1 | DDRP0 |
| | | W | | | | | | | | |
| 0x025B | RDRP | R | 0 | 0 | 0 | 0 | 0 | RDRP2 | RDRP1 | RDRP0 |
| | | W | | | | | | | | |
| 0x025C | PERP | R | 0 | 0 | PERP5 | PERP4 | PERP3 | PERP2 | PERP1 | PERP0 |
| | | W | | | | | | | | |
| 0x025D | PPSP | R | 0 | 0 | PPSP5 | PPSP4 | PPSP3 | PPSP2 | PPSP1 | PPSP0 |
| | | W | | | | | | | | |
| 0x025E | PIEP | R | OCIE | 0 | PIEP5 | PIEP4 | PIEP3 | PIEP2 | PIEP1 | PIEP0 |
| | | W | | | | | | | | |
| 0x025F | PIFP | R | OCIF | 0 | PIFP5 | PIFP4 | PIFP3 | PIFP2 | PIFP1 | PIFP0 |
| | | W | | | | | | | | |
| 0x0260–0x0268 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0269 | PTIL | R | 0 | 0 | 0 | 0 | PTIL3 | PTIL2 | PTIL1 | PTIL0 |
| | | W | | | | | | | | |
| 0x026A | DIENL | R | 0 | 0 | 0 | 0 | DIENL3 | DIENL2 | DIENL1 | DIENL0 |
| | | W | | | | | | | | |
| 0x026B | PTAL | R | PTTEL | PTPSL | PTABYPL | PTADIRL | PTAENL | 0 | PTAL1 | PTAL0 |
| | | W | | | | | | | | |
| 0x026C | PIRL | R | 0 | 0 | 0 | 0 | PIRL3 | PIRL2 | PIRL1 | PIRL0 |
| | | W | | | | | | | | |
| 0x026D | PPSL | R | 0 | 0 | 0 | 0 | PPSL3 | PPSL2 | PPSL1 | PPSL0 |
| | | W | | | | | | | | |
| 0x026E | PIEL | R | 0 | 0 | 0 | 0 | PIEL3 | PIEL2 | PIEL1 | PIEL0 |
| | | W | | | | | | | | |
| 0x026F | PIFL | R | 0 | 0 | 0 | 0 | PIFL3 | PIFL2 | PIFL1 | PIFL0 |
| | | W | | | | | | | | |
| 0x0270 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0271 | PT1AD | R | 0 | 0 | PT1AD5 | PT1AD4 | PT1AD3 | PT1AD2 | PT1AD1 | PT1AD0 |
| | | W | | | | | | | | |

| Global Address | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0272 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0273 | PTI1AD | R | 0 | 0 | PTI1AD5 | PTI1AD4 | PTI1AD3 | PTI1AD2 | PTI1AD1 | PTI1AD0 |
| | | W | | | | | | | | |
| 0x0274 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0275 | DDR1AD | R | 0 | 0 | DDR1AD5 | DDR1AD4 | DDR1AD3 | DDR1AD2 | DDR1AD1 | DDR1AD0 |
| | | W | | | | | | | | |
| 0x0276–0x0278 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0279 | PER1AD | R | 0 | 0 | PER1AD5 | PER1AD4 | PER1AD3 | PER1AD2 | PER1AD1 | PER1AD0 |
| | | W | | | | | | | | |
| 0x027A | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x027B | PPS1AD | R | 0 | 0 | PPS1AD5 | PPS1AD4 | PPS1AD3 | PPS1AD2 | PPS1AD1 | PPS1AD0 |
| | | W | | | | | | | | |
| 0x027C | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x027D | PIE1AD | R | 0 | 0 | PIE1AD5 | PIE1AD4 | PIE1AD3 | PIE1AD2 | PIE1AD1 | PIE1AD0 |
| | | W | | | | | | | | |
| 0x027E | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x027F | PIF1AD | R | 0 | 0 | PIF1AD5 | PIF1AD4 | PIF1AD3 | PIF1AD2 | PIF1AD1 | PIF1AD0 |
| | | W | | | | | | | | |

= Unimplemented

## 2.3.2 Register Descriptions

The following table summarizes the effect of the various configuration bits, that is data direction (DDR), output level (PORT/PT), pull enable (PER), pull select (PPS), interrupt enable (PIE) on the pin function, pull device and interrupt activity.

The configuration bit PPS is used for two purposes:

1. Configure the sensitive interrupt edge (rising or falling), if interrupt is enabled.

2. Select either a pullup or pulldown device if PER is active.

**Table 2-2. Pin Configuration Summary[1]**

| DDR | PORT PT | PER | PPS[1] | PIE[2] | Function | Pull Device | Interrupt |
|---|---|---|---|---|---|---|---|
| 0 | x | 0 | x | 0 | Input | Disabled | Disabled |
| 0 | x | 1 | 0 | 0 | Input | Pullup | Disabled |
| 0 | x | 1 | 1 | 0 | Input | Pulldown | Disabled |
| 0 | x | 0 | 0 | 1 | Input | Disabled | Falling edge |
| 0 | x | 0 | 1 | 1 | Input | Disabled | Rising edge |
| 0 | x | 1 | 0 | 1 | Input | Pullup | Falling edge |
| 0 | x | 1 | 1 | 1 | Input | Pulldown | Rising edge |
| 1 | 0 | x | x | 0 | Output, drive to 0 | Disabled | Disabled |
| 1 | 1 | x | x | 0 | Output, drive to 1 | Disabled | Disabled |
| 1 | 0 | x | 0 | 1 | Output, drive to 0 | Disabled | Falling edge |
| 1 | 1 | x | 1 | 1 | Output, drive to 1 | Disabled | Rising edge |

1 Always “0” on Port E

2 Applicable only on Port P and AD

**NOTE**

- All register bits in this module are completely synchronous to internal clocks during a register read.
- Figure of port data registers also display the alternative functions if applicable on the related pin as defined in Table 2-1. Names in parentheses denote the availability of the function when using a specific routing option.
- Figures of module routing registers also display the module instance or module channel associated with the related routing bit.

1. Not applicable for Port L. Refer to register descriptions.

## 2.3.3 Port E Data Register (PORTE)

Address 0x0008 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | PE1 | PE0 |
| W | | | | | | | | |
| Altern. Function | — | — | — | — | — | — | XTAL | EXTAL |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-1. Port E Data Register (PORTE)**

[1] Read: Anytime. The data source is depending on the data direction value.
Write: Anytime

**Table 2-3. PORTE Register Field Descriptions**

| Field | Description |
|---|---|
| 1<br>PE | **PorT data register port E** — General-purpose input/output data, CPMU OSC XTAL signal<br>If the CPMU OSC function is active this pin is used as XTAL signal and the pulldown device is disabled. When not used with the alternative function, this pin can be used as general-purpose I/O. In general-purpose output mode the register bit is driven to the pin.<br>If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the synchronized pin input state is read.<br>• The CPMU OSC function takes precedence over the general purpose I/O function if enabled. |
| 0<br>PE | **PorT data register port E** — General-purpose input/output data, CPMU OSC EXTAL signal<br>If the CPMU OSC function is active this pin is used as EXTAL signal and the pulldown device is disabled. When not used with the alternative function, this pin can be used as general-purpose I/O. In general-purpose output mode the register bit is driven to the pin.<br>If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the synchronized pin input state is read.<br>• The CPMU OSC function takes precedence over the general purpose I/O function if enabled. |

## 2.3.4 Port E Data Direction Register (DDRE)

Address 0x0009 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | DDRE1 | DDRE0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-2. Port E Data Direction Register (DDRE)**

[1] Read: Anytime
Write: Anytime

**Table 2-4. DDRE Register Field Descriptions**

| Field | Description |
|---|---|
| 1-0<br>DDRE | **Data Direction Register port E** —<br>This bit determines whether the associated pin is an input or output.<br><br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |

## 2.3.5 Port E, BKGD pin Pull Control Register (PUCR)

Address 0x000C — Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | BKPUE | 0 | PDPEE | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |

**Figure 2-3. Port E, BKGD pin Pull Control Register (PUCR)**

[1] Read:Anytime
Write:Anytime, except BKPUE, which is writable in special mode only

**Table 2-5. PUCR Register Field Descriptions**

| Field | Description |
|---|---|
| 6<br>BKPUE | **BKGD pin Pullup Enable** — Activate pullup device on pin<br>This bit configures whether a pullup device is activated, if the pin is used as input. If a pin is used as output this bit has no effect.<br>1 Pullup device enabled<br>0 Pullup device disabled |
| 4<br>PDPEE | Pull-Down Port E Enable — Activate pulldown devices on all port input pins<br>This bit configures whether a pulldown device is activated on all associated port input pins. If a pin is used as output or used with the CPMU OSC function this bit has no effect. Out of reset the pulldown devices are enabled.<br>1 Pulldown devices enabled<br>0 Pulldown devices disabled |

## 2.3.6 ECLK Control Register (ECLKCTL)

Address 0x001C — Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | NECLK | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-4. ECLK Control Register (ECLKCTL)**

[1] Read: Anytime
Write: Anytime

**Table 2-6. ECLKCTL Register Field Descriptions**

| Field | Description |
|---|---|
| 7<br>NECLK | **No ECLK** — Disable ECLK output<br>This bit controls the availability of a free-running clock on the ECLK pin. This clock has a fixed rate equivalent to the internal bus clock.<br>1 ECLK disabled<br>0 ECLK enabled |

## 2.3.7 PIM Miscellaneous Register (PIMMISC)

Address 0x001D — Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | OCPE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-5. PIM Miscellaneous Register (PIMMISC)**

[1] Read: Anytime
Write: Anytime

**Table 2-7. PIMMISC Register Field Descriptions**

| Field | Description |
|---|---|
| 7<br>OCPE | **Over-Current Protection Enable**— Activate over-current detector on PP2<br>Refer to Section 2.5.3, “Over-Current Protection on EVDD”<br>1 PP2 over-current detector enabled<br>0 PP2 over-current detector disabled |

## 2.3.8 IRQ Control Register (IRQCR)

Address 0x001E — Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | IRQE | IRQEN | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-6. IRQ Control Register (IRQCR)**

[1] Read: Anytime
Write:
IRQE: Once in normal mode, anytime in special mode
IRQEN: Anytime

**Table 2-8. IRQCR Register Field Descriptions**

| Field | Description |
|---|---|
| 7<br>IRQE | **IRQ select Edge sensitive only** —<br>1 $\overline{\text{IRQ}}$ pin configured to respond only to falling edges. Falling edges on the $\overline{\text{IRQ}}$ pin will be detected anytime IRQE=1 and will be cleared only upon a reset or the servicing of the $\overline{\text{IRQ}}$ interrupt.<br>0 $\overline{\text{IRQ}}$ pin configured for low level recognition |
| 6<br>IRQEN | **IRQ ENable** —<br>1 $\overline{\text{IRQ}}$ pin is connected to interrupt logic<br>0 $\overline{\text{IRQ}}$ pin is disconnected from interrupt logic |

## 2.3.9 Reserved Register

Address 0x001F Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| Reset | x | x | x | x | x | x | x | x |

**Figure 2-7. Reserved Register**

[1] Read: Anytime
Write: Only in special mode

**NOTE**

These reserved registers are designed for factory test purposes only and are not intended for general user access. Writing to these registers when in special modes can alter the module's functionality.

## 2.3.10 Port T Data Register (PTT)

Address 0x0240 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PTT3 | PTT2 | PTT1 | PTT0 |
| W | | | | | | | | |
| Altern. Function | — | — | — | — | ($\overline{\text{SS}}$) | (SCK) | PWM7[2] | PWM6[2] |
| | — | — | — | — | (LPTXD) | (LPRXD) | (TXD0) | (RXD0) |
| | — | — | — | — | — | — | (LPDR1) | — |
| | — | — | — | — | IOC3[3] | IOC2[4] | IOC1[5] | IOC0[5] |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-8. Port T Data Register (PTT)**

1 Read: Anytime. The data source is depending on the data direction value.
Write: Anytime

2 PWM function available on this pin only if not used with a routed HSDRV or LSDRV function. Refer to Section 2.3.15, “Module Routing Register 0 (MODRR0)”

3 TIM output compare function available on this pin only if not used with routed HSDRV. Refer to Section 2.3.15, “Module Routing Register 0 (MODRR0)”. TIM input capture function available on this pin only if not used with LPRXD. Refer to Section 2.3.23, “Module Routing Register 2 (MODRR2)”.

4 TIM output compare function available on this pin only if not used with routed HSDRV. Refer to Section 2.3.15, “Module Routing Register 0 (MODRR0)”

5 TIM output compare function available on this pin only if not used with routed LSDRV. Refer to Section 2.3.15, “Module Routing Register 0 (MODRR0)”

**Table 2-9. PTT Register Field Descriptions**

| Field | Description |
|---|---|
| 3-2<br>PTT | **PorT data register port T** — General-purpose input/output data, SPI $\overline{SS}$ and SCK, TIM input/output, routed LINPHY<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The routed SPI takes precedence over the routed LINPHY function, TIM output function and the general-purpose I/O function if enabled.<br>• The routed LINPHY function takes precedence over the TIM output function and the general-purpose I/O function if the related channel is enabled.<br>• The TIM function takes precedence over the general-purpose I/O function. |
| 1<br>PTT | **PorT data register port T** — General-purpose input/output data, TIM input/output, routed SCI0, LPDR[LPDR1]<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The routed SCI0 or LPDR[LPDR1] takes precedence over the TIM output function and the general-purpose I/O function if enabled.<br>• The TIM function takes precedence over the general-purpose I/O function if enabled. |
| 0<br>PTT | **PorT data register port T** — General-purpose input/output data, TIM input/output, routed SCI0<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The routed SCI0 takes precedence over the TIM output function and the general-purpose I/O function if enabled.<br>• The TIM function takes precedence over the general-purpose I/O function if enabled. |

## 2.3.11 Port T Input Register (PTIT)

Address 0x0241 Access: User read only[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PTIT3 | PTIT2 | PTIT1 | PTIT0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-9. Port T Input Register (PTIT)**

[1] Read: Anytime
Write:Never

**Table 2-10. PTIT Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PTIT | **PorT Input data register port T** —<br>A read always returns the synchronized input state of the associated pin. It can be used to detect overload or short circuit conditions on output pins. |

## 2.3.12 Port T Data Direction Register (DDRT)

Address 0x0242 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | DDRT3 | DDRT2 | DDRT1 | DDRT0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-10. Port T Data Direction Register (DDRT)**

[1] Read: Anytime
Write: Anytime

**Table 2-11. DDRT Register Field Descriptions**

| Field | Description |
|---|---|
| 3<br>DDRT | **Data Direction Register port T** —<br>This bit determines whether the pin is an input or output<br>Depending on the configuration of the enabled SPI the I/O state will be forced to be input or output. The enabled routed LINPHY forces the I/O state to be an input (LPTXD). Else the TIM forces the I/O state to be an output for a TIM port associated with an enabled TIM output compare. In these cases the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 2<br>DDRT | **Data Direction Register port T** —<br>This bit determines whether the pin is an input or output.<br>Depending on the configuration of the enabled SPI the I/O state will be forced to be input or output. The enabled routed LINPHY forces the I/O state to be an output (LPRXD). Else the TIM forces the I/O state to be an output for a TIM port associated with an enabled TIM output compare. In these cases the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 1-0<br>DDRT | **Data Direction Register port T** —<br>This bit determines whether the pin is an input or output.<br>Depending on the configuration of the enabled routed SCI0 the I/O state will be forced to be input or output. The enabled routed LINPHY forces the I/O state to be an output (LPDR[LPDR1]). Else the TIM forces the I/O state to be an output for a TIM port associated with an enabled TIM output compare. In these cases the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |

## 2.3.13 Port T Pull Device Enable Register (PERT)

Address 0x0244 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PERT3 | PERT2 | PERT1 | PERT0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-11. Port T Pull Device Enable Register (PERT)**

[1] Read: Anytime
Write: Anytime

**Table 2-12. PERT Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PERT | **Pull device Enable Register port T** — Enable pull device on input pin<br>This bit controls whether a pull device on the associated port input pin is active. If a pin is used as output this bit has no effect. The polarity is selected by the related polarity select register bit.<br>1 Pull device enabled<br>0 Pull device disabled |

## 2.3.14 Port T Polarity Select Register (PPST)

Address 0x0245 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PPST3 | PPST2 | PPST1 | PPST0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-12. Port T Polarity Select Register (PPST)**

[1] Read: Anytime
Write: Anytime

**Table 2-13. PPST Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PPST | **Pull device Polarity Select register port T** — Configure pull device polarity on input pin<br>This bit selects a pullup or a pulldown device if enabled on the associated port input pin.<br>1 A pulldown device is selected<br>0 A pullup device is selected |

## 2.3.15 Module Routing Register 0 (MODRR0)

Address 0x0246 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | MODRR07 | MODRR06 | MODRR05 | MODRR04 | MODRR03 | MODRR02 | MODRR01 | MODRR00 |
| Routing Option | HS1 | | HS0 | | LS1 | | LS0 | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-13. Module Routing Register 0 (MODRR0)**

[1] Read: Anytime
Write: Once in normal, anytime in special mode

**Table 2-14. Module Routing Register 0 Field Descriptions**

| Field | Description |
|---|---|
| 7-6<br>MODRR0 | **MODule Routing Register 0** — HS1<br>This register controls the routing of PWM and TIM channels to pin HS1 of HSDRV module. By default the pin is controlled by the related HSDRV port register bit.<br>11 PWM channel 1 routed to HS1 if enabled<br>10 PWM channel 4 routed to HS1 if enabled<br>01 TIM output compare channel 3 routed to HS1 if enabled<br>00 HS1 controlled by register bit HSDR[HSDR1]. Refer to HSDRV section. |
| 5-4<br>MODRR0 | **MODule Routing Register 0** — HS0<br>This register controls the routing of PWM and TIM channels to pin HS0 of HSDRV module. By default the pin is controlled by the related HSDRV port register bit.<br>11 PWM channel 3 routed to HS0 if enabled<br>10 PWM channel 3 routed to HS0 if enabled<br>01 TIM output compare channel 2 routed to HS0 if enabled<br>00 HS0 controlled by register bit HSDR[HSDR0]. Refer to HSDRV section. |
| 3-2<br>MODRR0 | **MODule Routing Register 0** — LS1<br>This register controls the routing of PWM and TIM channels to pin LS1 of LSDRV module. By default the pin is controlled by the related LSDRV port register bit.<br>11 PWM channel 7 routed to LS1 if enabled<br>10 PWM channel 7 routed to LS1 if enabled<br>01 TIM output compare channel 1 routed to LS1 if enabled<br>00 LS1 controlled by register bit LSDR[LSDR1]. Refer to LSDRV section. |
| 1-0<br>MODRR0 | **MODule Routing Register 0** — LS0<br>This register controls the routing of PWM and TIM channels to pin LS0 of LSDRV module. By default the pin is controlled by the related LSDRV port register bit.<br>11 PWM channel 5 routed to LS0 if enabled<br>10 PWM channel 6 routed to LS0 if enabled<br>01 TIM output compare channel 0 routed to LS0 if enabled<br>00 LS0 controlled by register bit LSDR[LSDR0]. Refer to LSDRV section. |

## 2.3.16 Module Routing Register 1 (MODRR1)

Address 0x0247 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | MODRR15 | MODRR14 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Routing Option | — | — | PWM5 ETRIG1 | PWM4 ETRIG0 | — | — | — | — |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-14. Module Routing Register 1 (MODRR1)**

[1] Read: Anytime
Write: Once in normal, anytime in special mode

**Table 2-15. Module Routing Register 1 Field Descriptions**

| Field | Description |
|---|---|
| 5 MODRR1 | **MODule Routing Register 1 — PWM5, ETRIG1**<br>1 PWM channel 5 on PS3; ETRIG1 on PS3<br>0 PWM channel 5 on PP5; ETRIG1 on PP5 |
| 4 MODRR1 | **MODule Routing Register 1 — PWM4, ETRIG0**<br>1 PWM channel 4 on PS2; ETRIG0 on PS2<br>0 PWM channel 4 on PP4; ETRIG0 on PP4 |

## 2.3.17 Port S Data Register (PTS)

Address 0x0248 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PTS5 | PTS4 | PTS3 | PTS2 | PTS1 | PTS0 |
| W | | | | | | | | |
| Altern. Function | — | — | — | — | ECLK | — | — | — |
| | — | — | $\overline{SS}$ | SCK | MOSI | MISO | — | — |
| | — | — | — | — | (TXD1) | (RXD1) | TXD1 | RXD1 |
| | — | — | — | — | (PWM5[2]) | (PWM4[2]) | (LPDR1) | — |
| | — | — | — | — | (ETRIG1) | (ETRIG0) | (TXD0) | (RXD0) |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-15. Port S Data Register (PTS)**

[1] Read: Anytime. The data source is depending on the data direction value.
Write: Anytime

[2] PWM function available on this pin only if not used with a routed HSDRV or LSDRV function. Refer to Section 2.3.15, “Module Routing Register 0 (MODRR0)”

**Table 2-16. PTS Register Field Descriptions**

| Field | Description |
|---|---|
| 5<br>PTS | **PorT data register port S** — General-purpose input/output data, SPI $\overline{SS}$<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The SPI function takes precedence over the general-purpose I/O function if enabled. |
| 4<br>PTS | **PorT data register port S** — General-purpose input/output data, SPI SCK<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The SPI function takes precedence over the general-purpose I/O function if enabled. |
| 3<br>PTS | **PorT data register port S** — General-purpose input/output data, ECLK, SPI MOSI, routed SCI1, routed PWM, routed ETRIG<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The ECLK output function takes precedence over the SPI, routed SCI1 and PWM and the general purpose I/O function if enabled.<br>• The SPI function takes precedence over the routed SCI1, routed PWM and the general purpose I/O function if enabled.<br>• The routed SCI1 function takes precedence over the PWM and general-purpose I/O function if enabled.<br>• The routed PWM function takes precedence over the general-purpose I/O function if enabled. |
| 2<br>PTS | **PorT data register port S** — General-purpose input/output data, SPI MISO, routed SCI1, routed PWM, routed ETRIG<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The SPI function takes precedence over the routed SCI1, routed PWM and the general purpose I/O function if enabled.<br>• The routed SCI1 function takes precedence over the routed PWM and the general-purpose I/O function if enabled.<br>• The routed PWM function takes precedence over the general-purpose I/O function if enabled. |
| 1<br>PTS | **PorT data register port S** — General-purpose input/output data, SCI1, routed SCI0 or LPDR[LPDR1]<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The SCI1 function takes precedence over the routed SCI0 or LPDR[LPDR1] function and the general-purpose I/O function if enabled.<br>• The routed SCI0 or LPDR[LPDR1] function takes precedence over the general-purpose I/O function if enabled. |
| 0<br>PTS | **PorT data register port S** — General-purpose input/output data, SCI1, routed SCI0<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The SCI1 function takes precedence over the routed SCI0 function and the general-purpose I/O function if enabled.<br>• The routed SCI0 function takes precedence over the general-purpose I/O function if enabled. |

## 2.3.18 Port S Input Register (PTIS)

Address 0x0249 Access: User read only[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PTIS5 | PTIS4 | PTIS3 | PTIS2 | PTIS1 | PTIS0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-16. Port S Input Register (PTIS)**

[1] Read: Anytime
Write:Never

**Table 2-17. PTIS Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0<br>PTIS | **PorT Input data register port S** —<br>A read always returns the synchronized input state of the associated pin. It can be used to detect overload or short circuit conditions on output pins. |

## 2.3.19 Port S Data Direction Register (DDRS)

Address 0x024A Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | DDRS5 | DDRS4 | DDRS3 | DDRS2 | DDRS1 | DDRS0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-17. Port S Data Direction Register (DDRS)**

[1] Read: Anytime
Write: Anytime

**Table 2-18. DDRS Register Field Descriptions**

| Field | Description |
|---|---|
| 5<br>DDRS | **Data Direction Register port S** —<br>This bit determines whether the associated pin is an input or output. Depending on the configuration of the enabled SPI the I/O state will be forced to be input or output. In this case the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 4<br>DDRS | **Data Direction Register port S** —<br>This bit determines whether the associated pin is an input or output. Depending on the configuration of the enabled SPI the I/O state will be forced to be input or output. In this case the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |

**Table 2-18. DDRS Register Field Descriptions (continued)**

| Field | Description |
|---|---|
| 3<br>DDRS | **Data Direction Register port S** —<br>This bit determines whether the associated pin is an input or output. The ECLK output function, routed SCI1 and routed PWM function forces the I/O state to output if enabled. Depending on the configuration of the enabled SPI the I/O state will be forced to be input or output. In these cases the data direction bit will not change. The routed ETRIG function has no effect on the I/O state.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 2<br>DDRS | **Data Direction Register port S** —<br>This bit determines whether the associated pin is an input or output. Depending on the configuration of the enabled SPI the I/O state will be forced to be input or output. The routed SCI1 function forces the I/O state to input if enabled. The routed PWM function forces the I/O state to output if enabled. In these cases the data direction bit will not change. The routed ETRIG function has no effect on the I/O state.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 1<br>DDRS | **Data Direction Register port S** —<br>This bit determines whether the associated pin is an input or output.<br>Depending on the configuration of the enabled SCI the I/O state will be forced to be input or output. The enabled routed LINPHY forces the I/O state to be an output (LPDR[LPDR1]). In these cases the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 0<br>DDRS | **Data Direction Register port S** —<br>This bit determines whether the associated pin is an input or output.<br>Depending on the configuration of the enabled SCI the I/O state will be forced to be input or output. In this case the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |

## 2.3.20 Port S Pull Device Enable Register (PERS)

Address 0x024C — Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PERS5 | PERS4 | PERS3 | PERS2 | PERS1 | PERS0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figure 2-18. Port S Pull Device Enable Register (PERS)**

[1] Read: Anytime
Write: Anytime

**Table 2-19. PERS Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0<br>PERS | **Pull device Enable Register port S** — Enable pull device on input pin or wired-or output pin<br>This bit controls whether a pull device on the associated port input pin is active. If a pin is used as output this bit has only effect if used in wired-or mode. The polarity is selected by the related polarity select register bit.<br>1 Pull device enabled<br>0 Pull device disabled |

## 2.3.21 Port S Polarity Select Register (PPSS)

Address 0x024D Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PPSS5 | PPSS4 | PPSS3 | PPSS2 | PPSS1 | PPSS0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-19. Port S Polarity Select Register (PPSS)**

[1] Read: Anytime
Write: Anytime

**Table 2-20. PPSS Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0<br>PPSS | **Pull device Polarity Select register port S** — Configure pull device polarity on input pin<br>This bit selects a pullup or a pulldown device if enabled on the associated port input pin.<br>1 A pulldown device is selected<br>0 A pullup device is selected |

## 2.3.22 Port S Wired-Or Mode Register (WOMS)

Address 0x024E Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | WOMS5 | WOMS4 | WOMS3 | WOMS2 | WOMS1 | WOMS0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-20. Port S Wired-Or Mode Register (WOMS)**

[1] Read: Anytime
Write: Anytime

**Table 2-21. WOMS Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0<br>WOMS | **Wired-Or Mode register port S** — Enable open-drain functionality on output pin<br>This bit configures an output pin as wired-or (open-drain) or push-pull. In wired-or mode a logic “0” is driven active-low while a logic “1” remains undriven. This allows a multipoint connection of several serial modules. The bit has no influence on pins used as input.<br>1 Output buffer operates as open-drain output<br>0 Output buffer operates as push-pull output |

## 2.3.23 Module Routing Register 2 (MODRR2)

Address 0x024F Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | MODRR27 | 0 | MODRR25 | MODRR24 | MODRR23 | MODRR22 | MODRR21 | MODRR20 |
| W | | | | | | | | |
| Routing Option | LPRXD to TIM | — | SPI $\overline{SS}$ and SCK | SCI1 | SCI0-to-LINPHY interface | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-21. Module Routing Register 2 (MODRR2)**

[1] Read: Anytime
Write: Once in normal, anytime in special mode

**Table 2-22. Module Routing Register 2 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>MODRR2 | **MODule Routing Register 2** — TIM routing<br>1 TIM input capture channel 3 is connected to LPRXD<br>0 TIM input capture channel 3 is connected to PT3 |
| 5<br>MODRR2 | **MODule Routing Register 2** — SPI $\overline{SS}$ and SCK routing<br>1 $\overline{SS}$ on PT3; SCK on PT2<br>0 $\overline{SS}$ on PS5; SCK on PS4 |
| 4<br>MODRR2 | **MODule Routing Register 2** — SCI1 routing<br>1 TXD1 on PS3; RXD1 on PS2<br>0 TXD1 on PS1; RXD1 on PS0 |
| 3-0<br>MODRR2 | **MODule Routing Register 2** — SCI0-to-LINPHY routing<br>Selection of SCI0-to-LINPHY interface routing options to support probing and conformance testing. Refer to Figure 2-22 for an illustration and Table 2-23 for preferred settings. SCI0 must be enabled for TXD0 routing to take effect on pins. LINPHY must be enabled for LPRXD and LPDR[LPDR1] routings to take effect on pins. |

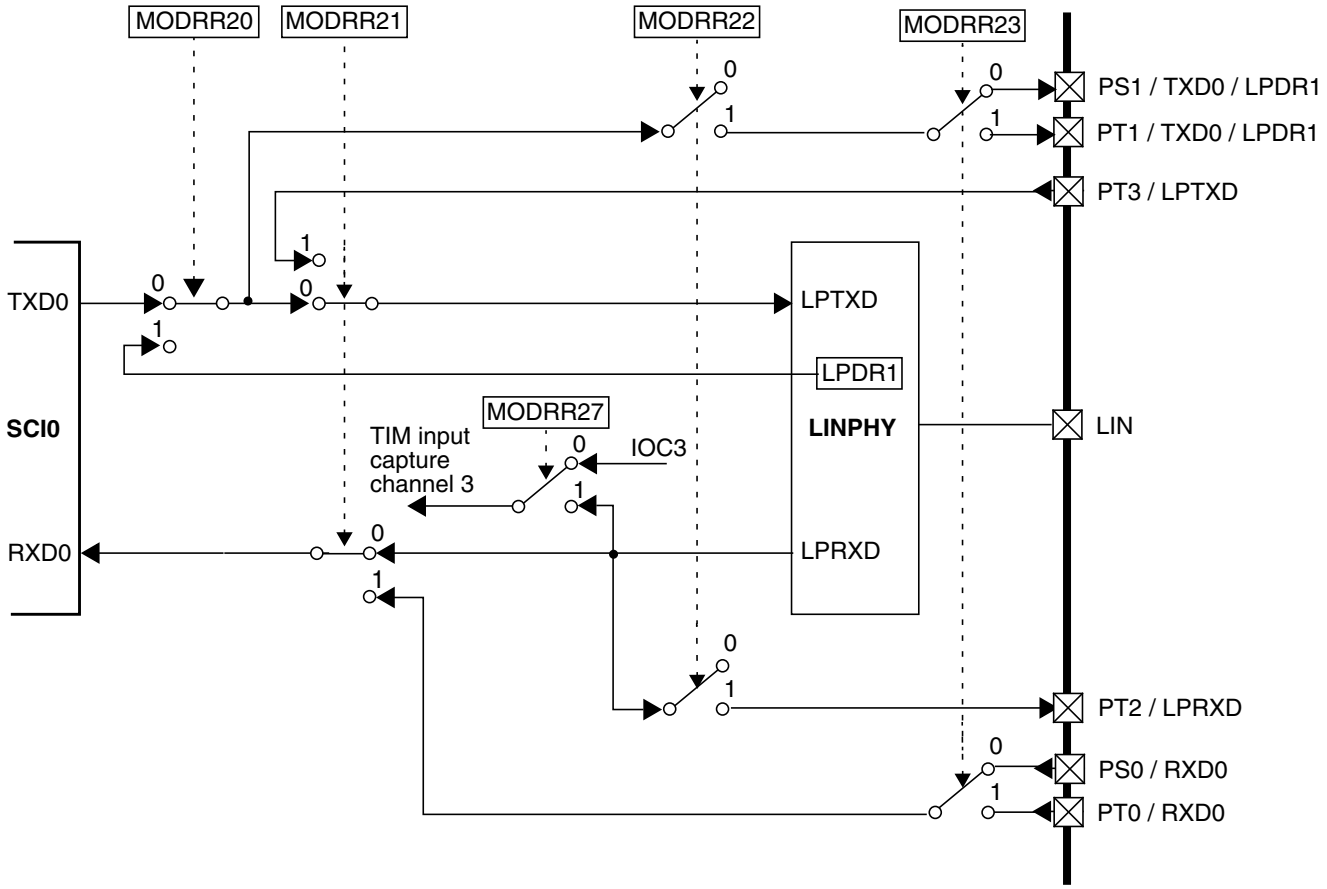
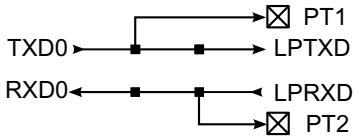
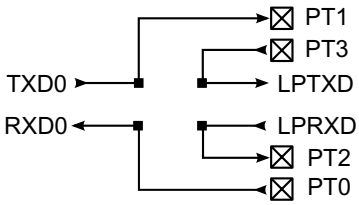



**Figure 2-22. SCI0-to-LINPHY Routing Options Illustration**

**Table 2-23. Preferred Interface Configurations**

| MODRR2[3:0] | Signal Routing | Description |
|---|---|---|
| 0000 | TXD0 → LPTXD<br>RXD0 ← LPRXD | Default setting:<br>SCI0 connects to LINPHY, interface internal only |
| 0001 | LPDR1 → LPTXD<br>RXD0 ← LPRXD | Direct control setting:<br>LPDR[LPDR1] register bit controls LPTXD, interface internal only |

| MODRR2[3:0] | Signal Routing | Description |
|---|---|---|
| 1100 |  | Probe setting:<br><br>SCI0 connects to LINPHY, interface accessible on 2 external pins |
| 1110 |  | Conformance test setting:<br><br>Interface opened and all 4 signals routed externally |

**NOTE**

For standalone usage of SCI0 on external pins set MODRR2[3:0]=0b1110 and disable the LINPHY (LPCR[LPE]=0). This releases PT2 and PT3 to other associated functions and maintains TXD0 and RXD0 signals on PT1 and PT0, respectively, if no other function with higher priority takes precedence.

## 2.3.24 Port P Data Register (PTP)

Address 0x0258 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PTP5 | PTP4 | PTP3 | PTP2 | PTP1 | PTP0 |
| W | | | | | | | | |
| Altern. Function | — | — | PWM5[2,3] | PWM4[2,3] | PWM3[2] | PWM2 | PWM1[2] | PWM0 |
| | — | — | $\overline{\text{IRQ}}$ | — | — | EVDD | $\overline{\text{XIRQ}}$ | — |
| | — | — | ETRIG1 | ETRIG0 | — | — | — | — |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-23. Port P Data Register (PTP)**

1 Read: Anytime. The data source is depending on the data direction value.
Write: Anytime

2 PWM function available on this pin only if not used with a routed HSDRV or LSDRV function. Refer to Section 2.3.15, “Module Routing Register 0 (MODRR0)”

3 PWM function available on this pin only if not routed to port S. Refer to Section 2.3.16, “Module Routing Register 1 (MODRR1)”

**Table 2-24. PTP Register Field Descriptions**

| Field | Description |
|---|---|
| 5<br>PTP | **PorT data register port P** — General-purpose input/output data, PWM output, ETRIG input, pin interrupt input/output, $\overline{IRQ}$ input<br>The $\overline{IRQ}$ signal is mapped to this pin when used with the IRQ interrupt function. If enabled (IRQCR[IRQEN]=1) the I/O state of the pin is forced to be an input.<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The IRQ function takes precedence over the PWM and the general-purpose I/O function if enabled.<br>• The PWM function takes precedence over the general-purpose I/O function if the related channel is enabled.<br>• Pin interrupts can be generated if enabled in input or output mode.<br>• The ETRIG function has no effect on the I/O state. |
| 4<br>PTP | **PorT data register port P** — General-purpose input/output data, PWM output, ETRIG input, pin interrupt input/output<br>The associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The PWM function takes precedence over the general-purpose I/O function if the related channel is enabled.<br>• Pin interrupts can be generated if enabled in input or output mode.<br>• The ETRIG function has no effect on the I/O state. |
| 3<br>PTP | **PorT data register port P** — General-purpose input/output data, PWM output, pin interrupt input/output<br>The associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The PWM function takes precedence over the general-purpose I/O function if the related channel is enabled.<br>• Pin interrupts can be generated if enabled in input or output mode. |
| 2<br>PTP | **PorT data register port P** — General-purpose input/output data, PWM output, switchable high-current capable external supply with over-current protection (EVDD)<br>The associated pin can be used as general-purpose I/O or as a supply for external devices such as Hall sensors (see Section 2.5.3, “Over-Current Protection on EVDD”. In output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The PWM function takes precedence over the general-purpose I/O function if the related channel is enabled.<br>• Pin interrupts can be generated if enabled in input or output mode.<br>• An over-current interrupt can be generated if enabled. Refer to Section 2.4.4.3, “Over-Current Interrupt” |

**Table 2-24. PTP Register Field Descriptions (continued)**

| Field | Description |
|---|---|
| 1<br>PTP | **PorT data register port P** — General-purpose input/output data, PWM output, pin interrupt input/output, $\overline{\text{XIRQ}}$ input<br>The $\overline{\text{XIRQ}}$ signal is mapped to this pin when used with the XIRQ interrupt function. The interrupt is enabled by clearing the X mask bit in the CPU Condition Code register. The I/O state of the pin is forced to input level upon the first clearing of the X bit and held in this state even if the bit is set again. A stop or wait recovery with the X bit set (refer to CPU12/CPU12X Reference Manual) is not available.<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The XIRQ function takes precedence over the PWM and the general-purpose I/O function if enabled.<br>• The PWM function takes precedence over the general-purpose I/O function if the related channel is enabled.<br>• Pin interrupts can be generated if enabled in input or output mode. |
| 0<br>PTP | **PorT data register port P** — General-purpose input/output data, PWM output, pin interrupt input/output<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit is set to 1, a read returns the value of the port register bit, otherwise the synchronized pin input state is read.<br>• The PWM function takes precedence over the general-purpose I/O function if the related channel is enabled.<br>• Pin interrupts can be generated if enabled in input or output mode. |

## 2.3.25 Port P Input Register (PTIP)

Address 0x0259 Access: User read only[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PTIP5 | PTIP4 | PTIP3 | PTIP2 | PTIP1 | PTIP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-24. Port P Input Register (PTIP)**

[1] Read: Anytime
Write:Never

**Table 2-25. PTIP Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0<br>PTIP | **PorT Input data register port P** —<br>A read always returns the synchronized input state of the associated pin. It can be used to detect overload or short circuit conditions on output pins. |

## 2.3.26 Port P Data Direction Register (DDRP)

Address 0x025A Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | DDRP5 | DDRP4 | DDRP3 | DDRP2 | DDRP1 | DDRP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-25. Port P Data Direction Register (DDRP)**

[1] Read: Anytime
Write: Anytime

**Table 2-26. DDRP Register Field Descriptions**

| Field | Description |
|---|---|
| 5<br>DDRP | **Data Direction Register port P** —<br>This bit determines whether the associated pin is an input or output.<br>The enabled IRQ function forces the I/O state to be an input if enabled. In this case the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 4-2<br>DDRP | **Data Direction Register port P** —<br>This bit determines whether the associated pin is an input or output.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 1<br>DDRP | **Data Direction Register port P** —<br>This bit determines whether the associated pin is an input or output.<br>The I/O state of the pin is forced to input level upon the first clearing of the X bit and held in this state even if the bit is set again. The PWM forces the I/O state to be an output for an enabled channel. In this case the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |
| 0<br>DDRP | **Data Direction Register port P** —<br>This bit determines whether the associated pin is an input or output.<br>The PWM forces the I/O state to be an output for an enabled channel. In this case the data direction bit will not change.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |

## 2.3.27 Port P Reduced Drive Register (RDRP)

Address 0x025B Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | RDRP2 | RDRP1 | RDRP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-26. Port P Reduced Drive Register (RDRP)**

[1] Read: Anytime
Write: Anytime

**Table 2-27. RDRP Register Field Descriptions**

| Field | Description |
|---|---|
| 2<br>RDRP | **Reduced Drive Register port P** — Select reduced drive for output pin<br>This bit configures the drive strength of the associated output pin as either full or reduced. If a pin is used as input this bit has no effect. The reduced drive function is independent of which function is being used on a particular pin.<br>1 Reduced drive selected (approx. 1/10 of the full drive strength)<br>0 Full drive strength enabled |
| 1-0<br>RDRP | **Reduced Drive Register port P** — Select reduced drive for output pin<br>This bit configures the drive strength of the associated output pin as either full or reduced. If a pin is used as input this bit has no effect. The reduced drive function is independent of which function is being used on a particular pin.<br>1 Reduced drive selected (approx. 1/5 of the full drive strength)<br>0 Full drive strength enabled |

## 2.3.28 Port P Pull Device Enable Register (PERP)

Address 0x025C Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PERP5 | PERP4 | PERP3 | PERP2 | PERP1 | PERP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-27. Port P Pull Device Enable Register (PERP)**

[1] Read: Anytime
Write: Anytime

**Table 2-28. PERP Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0<br>PERP | **Pull device Enable Register port P** — Enable pull device on input pin<br>This bit controls whether a pull device on the associated port input pin is active. If a pin is used as output this bit has no effect. The polarity is selected by the related polarity select register bit.<br>1 Pull device enabled<br>0 Pull device disabled |

## 2.3.29 Port P Polarity Select Register (PPSP)

Address 0x025D Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PPSP5 | PPSP4 | PPSP3 | PPSP2 | PPSP1 | PPSP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-28. Port P Polarity Select Register (PPSP)**

[1] Read: Anytime
Write: Anytime

**Table 2-29. PPSP Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0 PPSP | **Pull device Polarity Select register port P** — Configure pull device polarity and pin interrupt edge polarity on input pin<br>This bit selects a pullup or a pulldown device if enabled on the associated port input pin.<br>This bit also selects the polarity of the active pin interrupt edge.<br>1 A pulldown device is selected; rising edge selected<br>0 A pullup device is selected; falling edge selected |

## 2.3.30 Port P Interrupt Enable Register (PIEP)

Address 0x025E Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | OCIE | 0 | PIEP5 | PIEP4 | PIEP3 | PIEP2 | PIEP1 | PIEP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-29. Port P Interrupt Enable Register (PIEP)**

[1] Read: Anytime
Write: Anytime

**Table 2-30. PIEP Register Field Descriptions**

| Field | Description |
|---|---|
| 7 OCIE | **Over-Current Interrupt Enable register port P** —<br>This bit enables or disables the over-current interrupt on PP2.<br>1 PP2 over-current interrupt enabled<br>0 PP2 over-current interrupt disabled (interrupt flag masked) |
| 5-0 PIEP | **Pin Interrupt Enable register port P** —<br>This bit enables or disables the edge sensitive pin interrupt on the associated pin. An interrupt can be generated if the pin is operating in input or output mode when in use with the general-purpose or related peripheral function.<br>1 Interrupt is enabled<br>0 Interrupt is disabled (interrupt flag masked) |

## 2.3.31 Port P Interrupt Flag Register (PIFP)

Address 0x025F Access: User read/write[1]

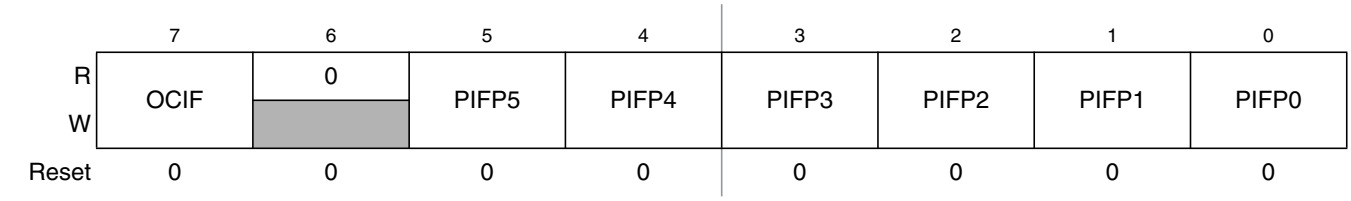



**Figure 2-30. Port P Interrupt Flag Register (PIFP)**

[1] Read: Anytime
Write: Anytime, write 1 to clear

**Table 2-31. PIFP Register Field Descriptions**

| Field | Description |
|---|---|
| 7<br>OCIF | **Over-Current Interrupt Flag** register port P —<br>This flag asserts if an over-current condition is detected on PP2 (Section 2.4.4.3, “Over-Current Interrupt”).<br>1 PP2 Over-current event occurred<br>0 No PP2 over-current event occurred |
| 5-0<br>PIFP | **Pin Interrupt Flag** register port P —<br>This flag asserts after a valid active edge was detected on the related pin (Section 2.4.4, “Interrupts”). This can be a rising or a falling edge based on the state of the polarity select register. An interrupt will occur if the associated interrupt enable bit is set.<br>1 Active edge on the associated bit has occurred<br>0 No active edge occurred |

## 2.3.32 Port L Input Register (PTIL)

Address 0x0269 Access: User read only[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PTIL3 | PTIL2 | PTIL1 | PTIL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-31. Port L Input Register (PTIL)**

[1] Read: Anytime
Write: No Write

**Table 2-32. PTIL - Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PTIL | **PorT Input data register port L** —<br>A read returns the synchronized input state if the associated pin is used in digital mode, that is the related DIENL bit is set to 1 and the pin is not used in analog mode (PTAL[PTAENL]=0). See Section 2.3.34, “Port L Analog Access Register (PTAL)”. A one is read in any other case[1]. |

[1] Refer to PTTEL bit description in Section 2.3.34, “Port L Analog Access Register (PTAL) for an override condition.

## 2.3.33 Port L Digital Input Enable Register (DIENL)

Address 0x26A Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | DIENL3 | DIENL2 | DIENL1 | DIENL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-32. Port L Digital Input Enable Register (DIENL)**

[1] Read: Anytime
Write: Anytime

**Table 2-33. DIENL Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>DIENL | **Digital Input ENable port L** — Input buffer control<br>This bit controls the HVI digital input function. If set to 1 the input buffers are enabled and the pin can be used with the digital function. If the analog input function is enabled (PTAL[PTAENL]=1) the input buffer of the selected HVI pin is forced off[1] in run mode and is released to be active in stop mode only if DIENL=1.<br>1 Associated pin digital input is enabled if not used as analog input in run mode[1]<br>0 Associated pin digital input is disabled[1] |

[1] Refer to PTTEL bit description in Section 2.3.34, “Port L Analog Access Register (PTAL) for an override condition.

## 2.3.34 Port L Analog Access Register (PTAL)

Address 0x026B Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | PTTEL | PTPSL | PTABYPL | PTADIRL | PTAENL | 0 | PTAL1 | PTAL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-33. Port L Analog Access Register (PTAL)**

[1] Read: Anytime
Write: Anytime

**Table 2-34. PTAL Register Field Descriptions**

| Field | Description |
|---|---|
| 7<br>PTTEL | **PorT Test Enable port L** —<br>This bit forces the input buffer of the selected HVI pin (PTAL[1:0]) to be active while using the analog function to support open input detection in run mode. Refer to Section 2.5.4, “Open Input Detection on HVI Pins”). In stop mode this bit has no effect.<br>**Note:** In direct input connection (PTAL[PTADIRL]=1) the digital input buffer is not enabled.<br>1 Input buffer enabled when used with analog function and not in direct mode (PTAL[PTADIRL]=0)<br>0 Input buffer disabled when used with analog function |
| 6<br>PTPSL | **PorT Pull Select port L** —<br>This bit selects a pull device on the selected HVI pin (PTAL[1:0]) in analog mode for open input detection. By default a pulldown device is active as part of the input voltage divider. If set to 1 and PTTEL=1 and not in stop mode a pullup to a level close to $V_{DDX}$ takes effect and overrides the weak pulldown device. Refer to Section 2.5.4, “Open Input Detection on HVI Pins”).<br>1 Pullup enabled<br>0 Pulldown enabled |
| 5<br>PTABYPL | **PorT ADC connection BYPass port L** —<br>This bit bypasses and powers down the impedance converter stage in the signal path from the analog input pin to the ADC channel input. This bit takes effect only if using direct input connection to the ADC channel (PTADIRL=1).<br>1 Bypass impedance converter in ADC channel signal path<br>0 Use impedance converter in ADC channel signal path |
| 4<br>PTADIRL | **PorT ADC DIRect connection port L** —<br>This bit connects the selected analog input signal (PTAL[1:0]) directly to the ADC channel bypassing the voltage divider. This bit takes effect only in analog mode (PTAENL=1).<br>1 Input pin directly connected to ADC channel<br>0 Input voltage divider active on analog input to ADC channel |
| 3<br>PTAENL | **PorT ADC connection ENable port L** —<br>This bit enables the analog signal link of an HVI pin selected by PTAL[1:0] to an ADC channel. If set to 1 the analog input function takes precedence over the digital input in run mode by forcing off the input buffers if not overridden by PTTEL=1.<br>1 Selected pin by PTAL[1:0] is connected to ADC channel<br>0 No Port L pin is connected to ADC |
| 1-0<br>PTAL | **PorT ADC connection selector port L** —<br>These selector bits choose the HVI pin connecting to an ADC channel if enabled (PTAENL=1). Refer to Table 2-35 for details. |

**NOTE**

When enabling the resistor paths to ground by setting PTAL[PTAENL]=1 or by changing PTAL[PTAL1:PTAL0], a settling time of $t_{UNC_HVI}$ + two bus cycles must be considered to let internal nodes be loaded with correct values.

**Table 2-35. HVI pin connected to ADC channel**

| PTAL[PTAL1] | PTAL[PTAL0] | HVI pin connected to ADC[1] |
|---|---|---|
| 0 | 0 | HVI0 |
| 0 | 1 | HVI1 |
| 1 | 0 | HVI2 |
| 1 | 1 | HVI3 |

[1] Refer to device overview section for channel assignment

## 2.3.35 Port L Input Divider Ratio Selection Register (PIRL)

Address 0x026C Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PIRL3 | PIRL2 | PIRL1 | PIRL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-34. Port L Input Divider Ratio Selection Register (PIRL)**

[1] Read: Anytime
Write: Anytime

**Table 2-36. PIRL Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PIRL | **Port L Input Divider Ratio Select** —<br>This bit selects one of two voltage divider ratios for the associated high-voltage input pin in analog mode.<br>1 $Ratio_{L_HVI}$ selected<br>0 $Ratio_{H_HVI}$ selected |

## 2.3.36 Port L Polarity Select Register (PPSL)

Address 0x026D Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PPSL3 | PPSL2 | PPSL1 | PPSL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-35. Port L Polarity Select Register (PPSL)**

[1] Read: Anytime
Write: Anytime

**Table 2-37. PPSL Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PPSL | **Pin interrupt Polarity Select register port L** —<br>This bit selects the polarity of the active pin interrupt edge.<br>1 Rising edge selected<br>0 Falling edge selected |

## 2.3.37 Port L Interrupt Enable Register (PIEL)

Address 0x026E Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PIEL3 | PIEL2 | PIEL1 | PIEL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-36. Port L Interrupt Enable Register (PIEL)**

[1] Read: Anytime
Write: Anytime

**Table 2-38. PIEL Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PIEL | **Pin Interrupt Enable register port L —**<br>This bit enables or disables the edge sensitive pin interrupt on the associated pin. For wakeup from stop mode this bit must be set.<br>1 Interrupt is enabled<br>0 Interrupt is disabled (interrupt flag masked) |

## 2.3.38 Port L Interrupt Flag Register (PIFL)

Address 0x026F Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PIFL3 | PIFL2 | PIFL1 | PIFL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-37. Port L Interrupt Flag Register (PIFL)**

[1] Read: Anytime
Write: Anytime, write 1 to clear

**Table 2-39. PIFL Register Field Descriptions**

| Field | Description |
|---|---|
| 3-0<br>PIFL | **Pin Interrupt Flag register port L —**<br>This flag asserts after a valid active edge was detected on the related pin (Section 2.4.4, “Interrupts”). This can be a rising or a falling edge based on the state of the polarity select register. An interrupt will occur if the associated interrupt enable bit is set.<br>1 Active edge on the associated bit has occurred<br>0 No active edge occurred |

## 2.3.39 Port AD Data Register (PT1AD)

Address 0x0271 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PT1AD5 | PT1AD4 | PT1AD3 | PT1AD2 | PT1AD1 | PT1AD0 |
| W | | | | | | | | |
| Altern. Function | — | — | AN5 | AN4 | AN3 | AN2 | AN1 | AN0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-38. Port AD Data Register (PT1AD)**

[1] Read: Anytime. The data source is depending on the data direction value.
Write: Anytime

**Table 2-40. PT1AD Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0 PT1AD | **PorT data register 1 port AD** — General-purpose input/output data, ADC AN analog input<br>When not used with the alternative function, the associated pin can be used as general-purpose I/O. In general-purpose output mode the register bit value is driven to the pin.<br>If the associated data direction bit set to 1, a read returns the value of the port register bit. If the data direction bit is set to 0 and the ADC Digital Input Enable Register (ATDDIEN) is set to 1 the synchronized pin input state is read. |

## 2.3.40 Port AD Input Register (PTI1AD)

Address 0x0273 Access: User read only[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PTI1AD5 | PTI1AD4 | PTI1AD3 | PTI1AD2 | PTI1AD1 | PTI1AD0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

u = Unaffected by reset

**Figure 2-39. Port P Input Register (PTI1AD)**

[1] Read: Anytime
Write:Never

**Table 2-41. PTI1AD Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0 PTI1AD | **PorT Input data register 1 port AD** —<br>A read always returns the synchronized input state of the associated pin if the ADC Digital Input Enable Register (ATDDIEN) is set to 1. Else a logic 1 is read. It can be used to detect overload or short circuit conditions on output pins. |

## 2.3.41 Port AD Data Direction Register (DDR1AD)

Address 0x0275 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | DDR1AD5 | DDR1AD4 | DDR1AD3 | DDR1AD2 | DDR1AD1 | DDR1AD0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-40. Port AD Data Direction Register (DDR1AD)**

[1] Read: Anytime
Write: Anytime

**Table 2-42. DDR1AD Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0 DDR1AD | **Data Direction Register 1 port AD** —<br>This bit determines whether the associated pin is an input or output.<br>To use the digital input function the ADC Digital Input Enable Register (ATDDIEN) has to be set to logic level “1”.<br>1 Associated pin is configured as output<br>0 Associated pin is configured as input |

## 2.3.42 Port AD Pull Enable Register (PER1AD)

Address 0x0279 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PER1AD5 | PER1AD4 | PER1AD3 | PER1AD2 | PER1AD1 | PER1AD0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-41. Port AD Pullup Enable Register (PER1AD)**

[1] Read: Anytime
Write: Anytime

**Table 2-43. PER1AD Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0 PER1AD | **Pull device Enable Register 1 port AD** — Enable pull device on input pin<br>This bit controls whether a pull device on the associated port input pin is active. If a pin is used as output this bit has no effect. The polarity is selected by the related polarity select register bit.<br>1 Pull device enabled<br>0 Pull device disabled |

## 2.3.43 Port AD Polarity Select Register (PPS1AD)

Address 0x027B Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PPS1AD5 | PPS1AD4 | PPS1AD3 | PPS1AD2 | PPS1AD1 | PPS1AD0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-42. Port AD Polarity Select Register (PPS1AD)**

[1] Read: Anytime
Write: Anytime

**Table 2-44. PPS1AD Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0 PPS1AD | **Pull device Polarity Select register 1 port AD** — Configure pull device polarity and pin interrupt edge polarity on input pin<br>This bit selects a pullup or a pulldown device if enabled on the associated port input pin.<br>This bit also selects the polarity of the active pin interrupt edge.<br>1 A pulldown device is selected; rising edge selected<br>0 A pullup device is selected; falling edge selected |

## 2.3.44 Port AD Interrupt Enable Register (PIE1AD)

Address 0x027D Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PIE1AD5 | PIE1AD4 | PIE1AD3 | PIE1AD2 | PIE1AD1 | PIE1AD0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-43. Port AD Interrupt Enable Register (PIE1AD)**

[1] Read: Anytime
Write: Anytime

**Table 2-45. PIE1AD Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0 PIE1AD | **Pin Interrupt Enable register 1 port AD** —<br>This bit enables or disables the edge sensitive pin interrupt on the associated pin. An interrupt can be generated if the pin is operating in input or output mode when in use with the general-purpose or related peripheral function.<br>For wakeup from stop mode this bit must be set to allow activating the RC oscillator.<br>1 Interrupt is enabled<br>0 Interrupt is disabled (interrupt flag masked) |

## 2.3.45 Port AD Interrupt Flag Register (PIF1AD)

Address 0x027F Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | PIF1AD5 | PIF1AD4 | PIF1AD3 | PIF1AD2 | PIF1AD1 | PIF1AD0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2-44. Port AD Interrupt Flag Register (PIF1AD)**

[1] Read: Anytime
Write: Anytime, write 1 to clear

**Table 2-46. PIF1AD Register Field Descriptions**

| Field | Description |
|---|---|
| 5-0<br>PIF1AD | **Pin Interrupt Flag register 1 port AD** —<br>This flag asserts after a valid active edge was detected on the related pin (Section 2.4.4, “Interrupts”). This can be a rising or a falling edge based on the state of the polarity select register. An interrupt will occur if the associated interrupt enable bit is set.<br>1 Active edge on the associated bit has occurred<br>0 No active edge occurred |

## 2.4 Functional Description

### 2.4.1 General

Each pin except BKGD and port L pins can act as general-purpose I/O. In addition each pin can act as an output or input of a peripheral module.

### 2.4.2 Registers

Table 2-47 lists the configuration registers which are available on each port. These registers except the pin input and routing registers can be written at any time, however a specific configuration might not become active.

For example selecting a pullup device: This device does not become active while the port is used as a push-pull output.

**Table 2-47. Register availability per port[1]**

| Port | Data | Input | Data Direction | Reduced Drive | Pull Enable | Polarity Select | Wired-Or Mode | Interrupt Enable | Interrupt Flag | Routing |
|---|---|---|---|---|---|---|---|---|---|---|
| E | yes | - | yes | - | yes | - | - | - | - | - |
| T | yes | yes | yes | - | yes | yes | - | - | - | yes |
| S | yes | yes | yes | - | yes | yes | yes | - | - | yes |
| P | yes | yes | yes | yes | yes | yes | - | yes | yes | - |
| L | - | yes | yes[2] | - | - | yes | - | yes | yes | - |
| AD | yes | yes | yes | - | yes | yes | - | yes | yes | - |

1 Each cell represents one register with individual configuration bits

2 Input buffer control only

#### 2.4.2.1 Data register (PTx)

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

Writing to this register has only an effect on the pin if the pin is used as general-purpose output. When reading this address, the synchronized state of the pin is returned if the associated data direction register bit is set to “0”.

If the data direction register bits are set to logic level “1”, the contents of the data register is returned. This is independent of any other configuration (Figure 2-45).

#### 2.4.2.2 Input register (PTIx)

This register is read-only and always returns the synchronized state of the pin (Figure 2-45).

#### 2.4.2.3 Data direction register (DDRx)

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

If a peripheral module controls the pin the contents of the data direction register is ignored (Figure 2-45).

Independent of the pin usage with a peripheral module this register determines the source of data when reading the associated data register address (2.4.2.1/2-90).

**NOTE**

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on port data or port input registers, when changing the data direction register.

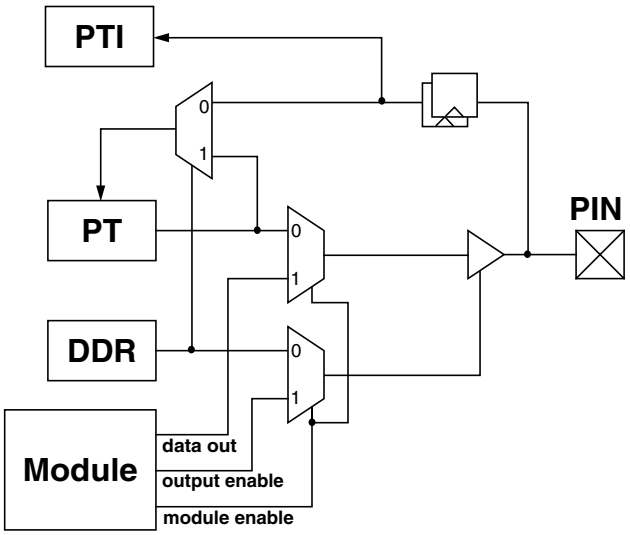



**Figure 2-45. Illustration of I/O pin functionality**

### 2.4.2.4 Reduced drive register (RDRx)

If the pin is used as an output this register allows the configuration of the drive strength independent of the use with a peripheral module.

### 2.4.2.5 Pull device enable register (PERx)

This register turns on a pullup or pulldown device on the related pins determined by the associated polarity select register (2.4.2.6/2-91).

The pull device becomes active only if the pin is used as an input or as a wired-or output. Some peripheral module only allow certain configurations of pull devices to become active. Refer to the respective bit descriptions.

### 2.4.2.6 Polarity select register (PPSx)

This register selects either a pullup or pulldown device if enabled.

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

#### 2.4.2.7 Wired-or mode register (WOMx)

If the pin is used as an output this register turns off the active-high drive. This allows wired-or type connections of outputs.

#### 2.4.2.8 Interrupt enable register (PIEx)

If the pin is used as an interrupt input this register serves as a mask to the interrupt flag to enable/disable the interrupt.

#### 2.4.2.9 Interrupt flag register (PIFx)

If the pin is used as an interrupt input this register holds the interrupt flag after a valid pin event.

#### 2.4.2.10 Module routing register (MODRRx)

Routing registers allow software re-configuration of specific peripheral inputs and outputs:

- MODRR0 selects the driving source of the HSDRV and LSDRV pins
- MODRR1 selects optional pins for PWM channels and ETRIG inputs
- MODRR2 supports options to test the internal SCI-LINPHY interface signals

### 2.4.3 Pins and Ports

**NOTE**

Please refer to the device pinout section to determine the pin availability in the different package options.

#### 2.4.3.1 BKGD pin

The BKGD pin is associated with the BDM module.

During reset, the BKGD pin is used as MODC input.

#### 2.4.3.2 Port E

This port is associated with the CPMU OSC.

Port E pins PE1-0 can be used for general-purpose or with the CPMU OSC module.

#### 2.4.3.3 Port T

This port is associated with TIM, routed SCI-LINPHY interface and routed SPI.

Port T pins can be used for either general-purpose I/O or with the channels of the standard TIM, SPI, or SCI and LINPHY subsystems.

### 2.4.3.4 Port S

This port is associated with the ECLK, SPI, SCI1, routed SCI0, routed PWM channels and ETRIG inputs.

Port S pins can be used either for general-purpose I/O, or with the ECLK, SPI, SCI, and PWM subsystems.

### 2.4.3.5 Port P

Port P pins can be used for either general-purpose I/O, $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$ or with the PWM subsystem. All pins feature pin interrupt functionality.

PP2 has an increased current capability to drive up to 20 mA to supply external devices for external Hall sensors. An over-current protection is available.

PP1 and PP0 have an increased current capability to drive up to 10 mA.

PP4 and PP5 support ETRIG functionality.

PP5 can be used for either general-purpose input or as the level- or falling edge-sensitive $\overline{\text{IRQ}}$ interrupt input. $\overline{\text{IRQ}}$ will be enabled by setting the IRQCR[IRQEN] configuration bit (2.3.8/2-61) and clearing the I-bit in the CPU condition code register. It is inhibited at reset so this pin is initially configured as a simple input with a pullup.

PP0 can be used for either general-purpose input or as the level-sensitive $\overline{\text{XIRQ}}$ interrupt input. $\overline{\text{XIRQ}}$ can be enabled by clearing the X-bit in the CPU condition code register. It is inhibited at reset so this pin is initially configured as a high-impedance input with a pullup.

### 2.4.3.6 Port L

Port L provides four high-voltage inputs (HVI) with the following features:

- Input voltage proof up to $V_{HVIx}$
- Digital input function with pin interrupt and wakeup from stop capability
- Analog input function with selectable divider ratio routable to ADC channel. Optional direct input bypassing voltage divider and impedance converter. Capable to wakeup from stop (pin interrupts in run mode not available). Open input detection.

Figure 2-46 shows a block diagram of the HVI.

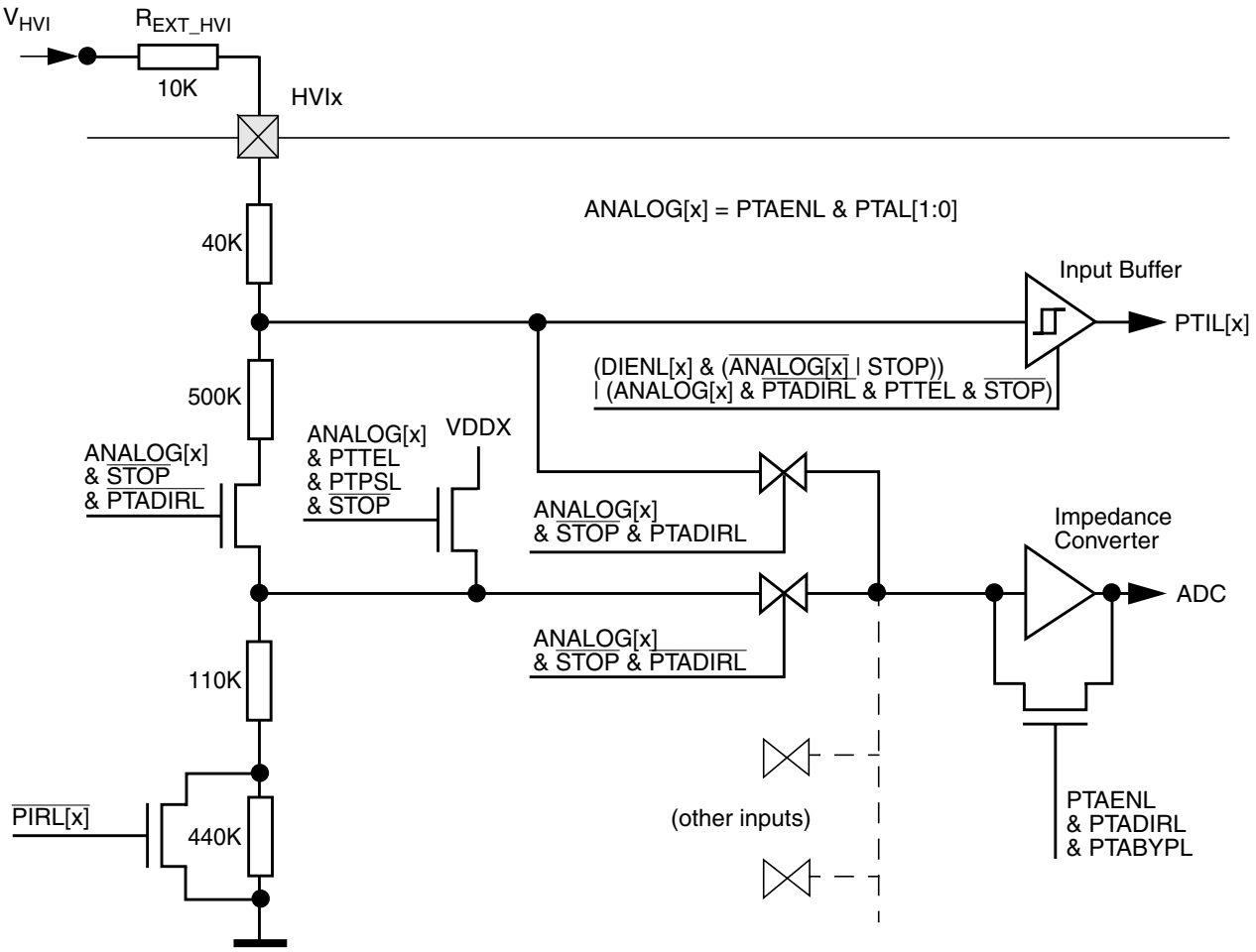



**Figure 2-46. HVI Block Diagram**

Voltages up to $V_{HVIx}$ can be applied to all HVI pins. Internal voltage dividers scale the input signals down to logic level. There are two modes, digital and analog, where these signals can be processed.

#### 2.4.3.6.1 Digital Mode Operation

In digital mode the input buffers are enabled (DIENL[x]=1 & PTAL[PTAENL]=0). The synchronized pin input state determined at threshold level $V_{TH_HVI}$ can be read in register PTIL. Interrupt flags (PIFL) are set on input transitions if enabled (PIEL[x]=1) and configured for the related edge polarity (PPSL). Wakeup from stop mode is supported.

#### 2.4.3.6.2 Analog Mode Operation

In analog mode (PTAL[PTAENL]=1) the voltage applied to a selectable pin (PTAL[PTAL1:PTAL0]) can be measured on an internal ADC channel (refer to device overview section for channel assignment). One of two input divider ratios ($Ratio_{H_HVI}$, $Ratio_{L_HVI}$) can be chosen on each analog input (PIRL[x]) or the

voltage divider can be bypassed (PTAL[PTADIRL]=1). Additionally in latter case the impedance converter in the ADC signal path can be configured to be used or bypassed in direct input mode (PTAL[PTABYPL]).

In run mode the digital input buffer of the selected pin is disabled to avoid shoot-through current. Thus pin interrupts cannot be generated.

In stop mode the digital input buffer is enabled only if DIENL[x]=1 to support wakeup functionality.

Table 2-48 shows the HVI input configuration depending on register bits and operation mode.

**Table 2-48. HVI Input Configurations**

| Mode | DIENL | PTAENL | Digital Input | Analog Input | Resulting Function |
|---|---|---|---|---|---|
| Run | 0 | 0 | off | off | Input disabled (Reset) |
| | 0 | 1 | off[1] | enabled | Analog input, interrupt not supported |
| | 1 | 0 | enabled | off | Digital input, interrupt supported |
| | 1 | 1 | off[1] | enabled | Analog input, interrupt not supported |
| Stop | 0 | 0 | off | off | Input disabled, wakeup from stop not supported |
| | 0 | 1 | off | off | |
| | 1 | 0 | enabled | off | Digital input, wakeup from stop supported |
| | 1 | 1 | enabled | off | |

1 Enabled if (PTAL[PTTEL]=1 & PTAL[PTADIRL]=0)

**NOTE**

An external resistor $R_{EXT_HVI}$ must always be connected to the high-voltage inputs to protect the device pins from fast transients and to achieve the specified pin input divider ratios when using the HVI in analog mode.

### 2.4.3.7 Port AD

This port is associated with the ADC.

Port AD pins can be used for either general-purpose I/O, or with the ADC subsystem.

## 2.4.4 Interrupts

This section describes the interrupts generated by the PIM and their individual sources. Vector addresses and interrupt priorities are defined at MCU level.

**Table 2-49. PIM Interrupt Sources**

| Module Interrupt Sources | Local Enable |
|---|---|
| $\overline{\text{XIRQ}}$ | None |
| $\overline{\text{IRQ}}$ | IRQCR[IRQEN] |
| Port P pin interrupt | PIEP[PIEP5-PIEP0] |

**Table 2-49. PIM Interrupt Sources**

| Module Interrupt Sources | Local Enable |
|---|---|
| Port L pin interrupt | PIEL[PIEL3-PIEL0] |
| Port AD pin interrupt | PIE1AD[PIE1AD5-PIE1AD0] |
| Port P over-current | PIEP[OCIE] |

### 2.4.4.1 XIRQ, IRQ Interrupts

The $\overline{\text{XIRQ}}$ pin allows requesting non-maskable interrupts after reset initialization. During reset, the X bit in the condition code register is set and any interrupts are masked until software enables them.

The $\overline{\text{IRQ}}$ pin allows requesting asynchronous interrupts. The interrupt input is disabled out of reset. To enable the interrupt the IRQCR[IRQEN] bit must be set and the I bit cleared in the condition code register. The interrupt can be configured for level-sensitive or falling-edge-sensitive triggering. If IRQCR[IRQEN] is cleared while an interrupt is pending, the request will deassert.

Both interrupts are capable to wake-up the device from stop mode. Means for glitch filtering are not provided on these pins.

### 2.4.4.2 Pin Interrupts and Wakeup

Ports P, L and AD offer pin interrupt capability. The related interrupt enable (PIE) as well as the sensitivity to rising or falling edges (PPS) can be individually configured on per-pin basis. All bits/pins in a port 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 (PIF) and its corresponding port interrupt enable (PIE) are both set. The pin interrupt feature is also capable to wake up the CPU when it is in stop or wait mode.

A digital filter on each pin prevents short pulses from generating an interrupt. A valid edge on an input is detected if 4 consecutive samples of a passive level are followed by 4 consecutive samples of an active level. Else the sampling logic is restarted.

In run and wait mode the filters are continuously clocked by the bus clock. Pulses with a duration of $t_{PULSE} < n_{P_MASK}/f_{bus}$ are assuredly filtered out while pulses with a duration of $t_{PULSE} > n_{P_PASS}/f_{bus}$ guarantee a pin interrupt.

In stop mode the clock is generated by an RC-oscillator. The minimum pulse length varies over process conditions, temperature and voltage (Figure 2-47). Pulses with a duration of $t_{PULSE} < t_{P_MASK}$ are assuredly filtered out while pulses with a duration of $t_{PULSE} > t_{P_PASS}$ guarantee a wakeup event.

Please refer to the appendix table “Pin Interrupt Characteristics” for pulse length limits.

To maximize current saving the RC oscillator is active only if the following condition is true on any individual pin:

Sample count <= 4 (at active or passive level) and interrupt enabled (PIE[x]=1) and interrupt flag not set (PIF[x]=0).

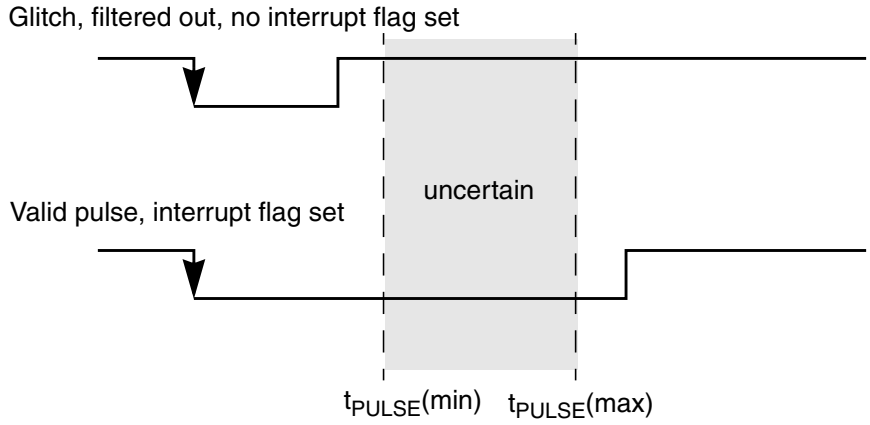



**Figure 2-47. Interrupt Glitch Filter (here: active low level selected)**

#### 2.4.4.3 Over-Current Interrupt

In case of an over-current condition on PP2 (see Section 2.5.3, "Over-Current Protection on EVDD") the over-current interrupt flag PIFP[OCIF] asserts. This flag generates an interrupt if the enable bit PIEP[OCIE] is set.

An asserted flag immediately forces the output pin low to protect the device. The flag must be cleared to re-enable the driver.

## 2.5 Initialization and Application Information

### 2.5.1 Port Data and Data Direction Register writes

It is not recommended to write PORT[x]/PT[x] and DDR[x] in a word access. When changing the register pins from inputs to outputs, the data may have extra transitions during the write access. Initialize the port data register before enabling the outputs.

### 2.5.2 ADC External Triggers ETRIG1-0

The ADC external trigger inputs ETRIG1-0 allow the synchronization of conversions to external trigger events if selected as trigger source (for details refer to ATDCTL1[ETRIGSEL] and ATDCTL1[ETRIGCH] configuration bits in ADC section). These signals are related to PWM channels 5-4 to support periodic trigger applications with the ADC. Other pin functions can also be used as triggers.

If a PWM channel is routed to an alternative pin, the ETRIG input function will follow the relocation accordingly.

If the related PWM channel is enabled and not routed for internal use, the PWM signal as seen on the pin will drive the ETRIG input. Else the ETRIG function will be triggered by other functions on the pin including general-purpose input.

### 2.5.3 Over-Current Protection on EVDD

Pin PP2 can be used as general-purpose I/O or due to its increased current capability in output mode as a switchable external power supply pin (EVDD) for external devices like Hall sensors. An over-current monitor is implemented to protect the controller from short circuits or excess currents on the output which can only arise if the pin is configured for full drive. Although the full drive current is available on the high and low side, the protection is only available if the pin is driven high (PTP[PTP2]=1). This is also true if using the pin with the PWM.

To power up the over-current monitor set PIMMISC[OCPE]=1.

In stop mode the over-current monitor is disabled for power saving. The increased current capability cannot be maintained to supply the external device. Therefore when using the pin as power supply the external load must be powered down prior to entering stop mode by setting PTP[PTP2]=0.

An over-current condition is detected if the output current level exceeds the threshold $I_{OCD}$ in run mode. The output driver is immediately forced low and the over-current interrupt flag PIFP[OCIF] asserts. Refer to Section 2.4.4.3, “Over-Current Interrupt”.

### 2.5.4 Open Input Detection on HVI Pins

The connection of an external pull device on any port L high-voltage input can be validated by using the built-in pull functionality of the HVI pins. Depending on the application type an external pulldown circuit can be detected with the internal pullup device whereas an external pullup circuit can be detected with the internal pulldown device which is part of the input voltage divider.

Note that the following procedures make use of a function that overrides the automatic disable mechanism of the digital input buffers when using the inputs in analog mode. Make sure to switch off the override function when using an input in analog mode after the check has been completed.

External pulldown device (Figure 2-48):

1. Enable analog function on HVIx in non-direct mode (PTAL[PTAENL]=1, PTAL[PTADIRL]=0, PTAL[PTAL1:PTAL0]=x, where x is 0, 1, 2, or 3)
2. Select internal pullup device on selected HVI (PTAL[PTPSL]=1)
3. Enable function to force input buffer active on selected HVI in analog mode (PTAL[PTTEL]=1)
4. Verify PTILx=0 for a connected external pulldown device; read PTILx=1 for an open input

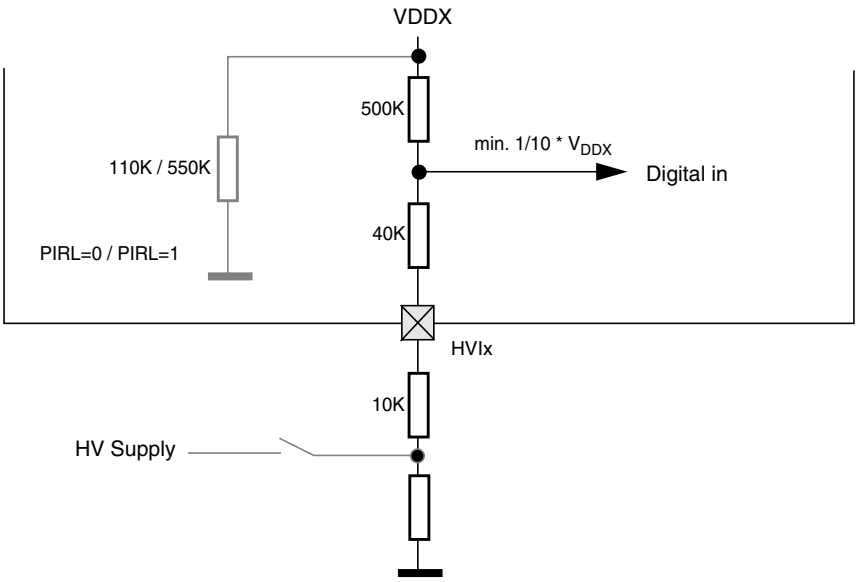



**Figure 2-48. Digital Input Read with Pullup Enabled**

External pullup device (Figure 2-49):

1. Enable analog function on HVIx in non-direct mode (PTAL[PTAENL]=1, PTAL[PTADIRL]=0, PTAL[PTAL1:PTAL0]=x, where x is 0, 1, 2, or 3)
2. Select internal pulldown device on selected HVI (PTAL[PTPSL]=0)
3. Enable function to force input buffer active on selected HVI in analog mode (PTAL[PTTEL]=1)
4. Verify PTILx=1 for a connected external pullup device; read PTILx=0 for an open input

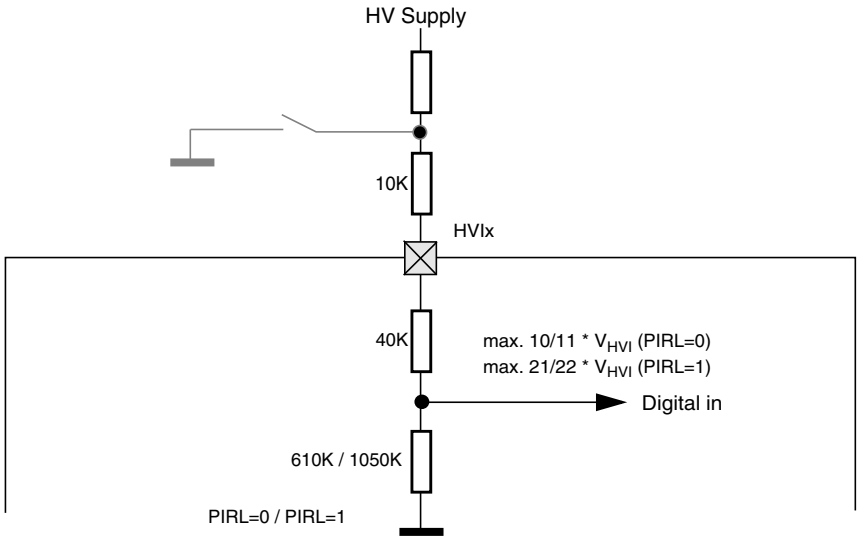



**Figure 2-49. Digital Input Read with Pulldown Enabled**

# Chapter 3
# S12G Memory Map Controller (S12GMMCV1)

**Table 3-1. Revision History Table**

| Rev. No. (Item No.) | Date (Submitted By) | Sections Affected | Substantial Change(s) |
|---|---|---|---|
| 01.02 | 20-May 2010 | | Updates for S12VR48 and S12VR64 |

## 3.1 Introduction

The S12GMMC module controls the access to all internal memories and peripherals for the CPU12 and S12SBDM module. It regulates access priorities and determines the address mapping of the on-chip resources. Figure 3-1 shows a block diagram of the S12GMMC module.

### 3.1.1 Glossary

**Table 3-2. Glossary Of Terms**

| Term | Definition |
|---|---|
| Local Addresses | Address within the CPU12's Local Address Map (Figure 3-11) |
| Global Address | Address within the Global Address Map (Figure 3-11) |
| Aligned Bus Access | Bus access to an even address. |
| Misaligned Bus Access | Bus access to an odd address. |
| NS | Normal Single-Chip Mode |
| SS | Special Single-Chip Mode |
| Unimplemented Address Ranges | Address ranges which are not mapped to any on-chip resource. |
| NVM | Non-volatile Memory; Flash or EEPROM |
| IFR | NVM Information Row. Refer to FTMRG Block Guide |

### 3.1.2 Overview

The S12GMMC connects the CPU12's and the S12SBDM's bus interfaces to the MCU's on-chip resources (memories and peripherals). It arbitrates the bus accesses and determines all of the MCU's memory maps. Furthermore, the S12GMMC is responsible for constraining memory accesses on secured devices and for selecting the MCU's functional mode.

### 3.1.3 Features

The main features of this block are:

- Paging capability to support a global 256 KByte memory address space
- Bus arbitration between the masters CPU12, S12SBDM to different resources.
- MCU operation mode control
- MCU security control
- Generation of system reset when CPU12 accesses an unimplemented address (i.e., an address which does not belong to any of the on-chip modules) in single-chip modes

### 3.1.4 Modes of Operation

The S12GMMC selects the MCU's functional mode. It also determines the devices behavior in secured and unsecured state.

#### 3.1.4.1 Functional Modes

Two functional modes are implemented on devices of the  product family:

- Normal Single Chip (NS)
  The mode used for running applications.
- Special Single Chip Mode (SS)
  A debug mode which causes the device to enter BDM Active Mode after each reset. Peripherals may also provide special debug features in this mode.

#### 3.1.4.2 Security

Devices can be secured to prohibit external access to the on-chip flash. The S12GMMC module determines the access permissions to the on-chip memories in secured and unsecured state.

### 3.1.5 Block Diagram

Figure 3-1 shows a block diagram of the S12GMMC.

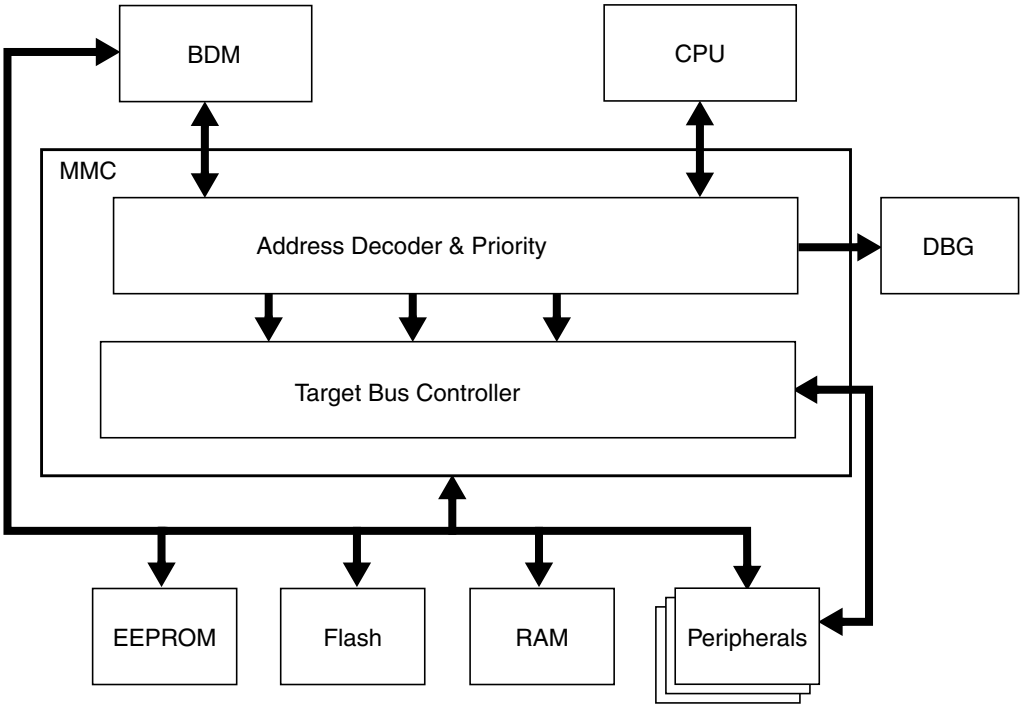



**Figure 3-1. S12GMMC Block Diagram**

## 3.2 External Signal Description

The S12GMMC uses two external pins to determine the devices operating mode: RESET and MODC (Figure 3-3) See Device User Guide (DUG) for the mapping of these signals to device pins.

**Table 3-3. External System Pins Associated With S12GMMC**

| Pin Name | Pin Functions | Description |
|---|---|---|
| $\overline{\text{RESET}}$ (See Section Device Overview) | $\overline{\text{RESET}}$ | The $\overline{\text{RESET}}$ pin is used the select the MCU's operating mode. |
| $\overline{\text{MODC}}$ (See Section Device Overview) | MODC | The MODC pin is captured at the rising edge of the $\overline{\text{RESET}}$ pin. The captured value determines the MCU's operating mode. |

## 3.3 Memory Map and Registers

### 3.3.1 Module Memory Map

A summary of the registers associated with the S12GMMC block is shown in Figure 3-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

| Address | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x000A | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000B | MODE | R | MODC | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0010 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0011 | DIRECT | R | DP15 | DP14 | DP13 | DP12 | DP11 | DP10 | DP9 | DP8 |
| | | W | | | | | | | | |
| 0x0012 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0013 | MMCCTL1 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | NVMRES |
| | | W | | | | | | | | |
| 0x0014 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0015 | PPAGE | R | 0 | 0 | 0 | 0 | PIX3 | PIX2 | PIX1 | PIX0 |
| | | W | | | | | | | | |
| 0x0016-0x0017 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

= Unimplemented or Reserved

**Figure 3-2. MMC Register Summary**

## 3.3.2 Register Descriptions

This section consists of the S12GMMC control register descriptions in address order.

### 3.3.2.1 Mode Register (MODE)

Address: 0x000B

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | MODC | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | MODC[1] | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

1. External signal (see Table 3-3).

= Unimplemented or Reserved

**Figure 3-3. Mode Register (MODE)**

Read: Anytime.

Write: Only if a transition is allowed (see Figure 3-4).

The MODC bit of the MODE register is used to select the MCU's operating mode.

**Table 3-4. MODE Field Descriptions**

| Field | Description |
|---|---|
| 7<br>MODC | **Mode Select Bit** — This bit controls the current operating mode during $\overline{\text{RESET}}$ high (inactive). The external mode pin MODC determines the operating mode during $\overline{\text{RESET}}$ low (active). The state of the pin is registered into the respective register bit after the $\overline{\text{RESET}}$ signal goes inactive (see Figure 3-4).<br>Write restrictions exist to disallow transitions between certain modes. Figure 3-4 illustrates all allowed mode changes. Attempting non authorized transitions will not change the MODE bit, but it will block further writes to the register bit except in special modes.<br>Write accesses to the MODE register are blocked when the device is secured. |

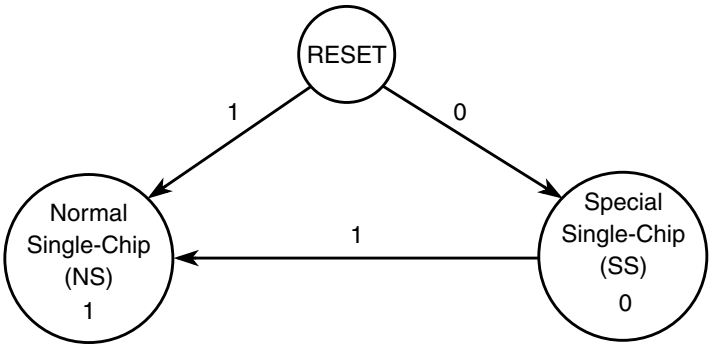



**Figure 3-4. Mode Transition Diagram when MCU is Unsecured**

### 3.3.2.2 Direct Page Register (DIRECT)

Address: 0x0011

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | DP15 | DP14 | DP13 | DP12 | DP11 | DP10 | DP9 | DP8 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 3-5. Direct Register (DIRECT)**

Read: Anytime

Write: anytime in special SS, write-once in NS.

This register determines the position of the 256 Byte direct page within the memory map.It is valid for both global and local mapping scheme.

**Table 3-5. DIRECT Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>DP[15:8] | **Direct Page Index Bits 15–8** — These bits are used by the CPU when performing accesses using the direct addressing mode. These register bits form bits [15:8] of the local address (see Figure 3-6). |

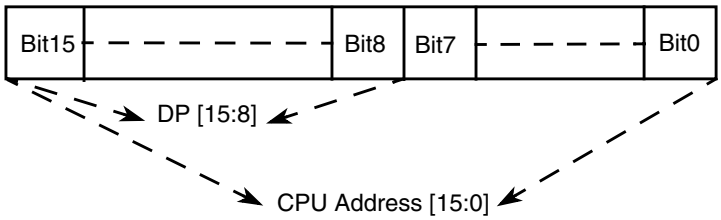



**Figure 3-6. DIRECT Address Mapping**

**Example 3-1. This example demonstrates usage of the Direct Addressing Mode**

```
MOVB     #$04,DIRECT       ;Set DIRECT register to 0x04. From this point on, all memory
                           ;accesses using direct addressing mode will be in the local
                           ;address range from 0x0400 to 0x04FF.
LDY      <$12              ;Load the Y index register from 0x0412 (direct access).
```

### 3.3.2.3 MMC Control Register (MMCCTL1)

Address: 0x0013

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | NVMRES |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 3-7. MMC Control Register (MMCCTL1)**

Read: Anytime.

Write: Anytime.

The NVMRES bit maps 16k of internal NVM resources (see Section FTMRG) to the global address space 0x04000 to 0x07FFF.

**Table 3-6. MODE Field Descriptions**

| Field | Description |
|---|---|
| 0<br>NVMRES | Map internal NVM resources into the global memory map<br>Write: Anytime<br>This bit maps internal NVM resources into the global address space.<br>0 Program flash is mapped to the global address range from 0x04000 to 0x07FFF.<br>1 NVM resources are mapped to the global address range from 0x04000 to 0x07FFF. |

### 3.3.2.4 Program Page Index Register (PPAGE)

Address: 0x0015

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | PIX3 | PIX2 | PIX1 | PIX0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

**Figure 3-8. Program Page Index Register (PPAGE)**

Read: Anytime

Write: Anytime

The four index bits of the PPAGE register select a 16K page in the global memory map (Figure 3-11). The selected 16K page is mapped into the paging window ranging from local address 0x8000 to 0xBFFF. Figure 3-9 illustrates the translation from local to global addresses for accesses to the paging window. The CPU has special access to read and write this register directly during execution of CALL and RTC instructions.

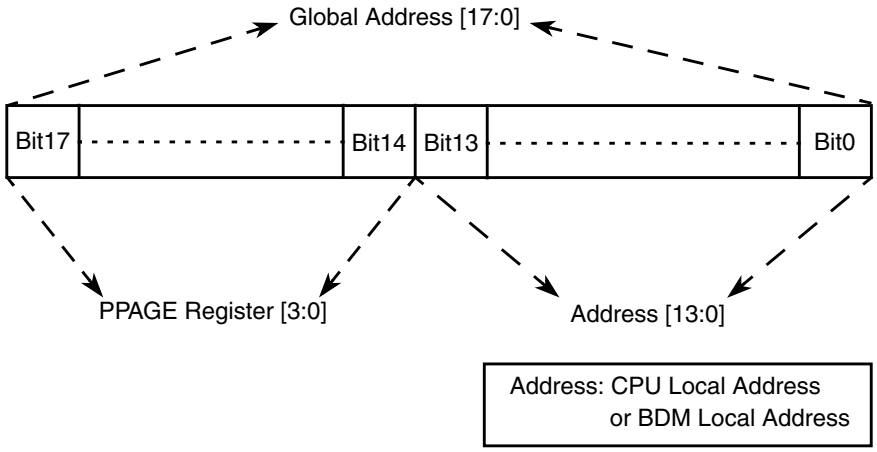



**Figure 3-9. PPAGE Address Mapping**

**NOTE**

Writes to this register using the special access of the CALL and RTC instructions will be complete before the end of the instruction execution.

**Table 3-7. PPAGE Field Descriptions**

| Field | Description |
|---|---|
| 3–0<br>PIX[3:0] | **Program Page Index Bits 3–0** — These page index bits are used to select which of the 256 flash array pages is to be accessed in the Program Page Window. |

The fixed 16KB page from 0x0000 to 0x3FFF is the page number 0xC. Parts of this page are covered by Registers, EEPROM and RAM space. See SoC Guide for details.

The fixed 16KB page from 0x4000–0x7FFF is the page number 0xD.

The reset value of 0xE ensures that there is linear Flash space available between addresses 0x0000 and 0xFFFF out of reset.

The fixed 16KB page from 0xC000-0xFFFF is the page number 0xF.

## 3.4 Functional Description

The S12GMMC block performs several basic functions of the sub-system operation: MCU operation modes, priority control, address mapping, select signal generation and access limitations for the system. Each aspect is described in the following subsections.

### 3.4.1 MCU Operating Modes

- Normal single chip mode

  This is the operation mode for running application code. There is no external bus in this mode.
- Special single chip mode

  This mode is generally used for debugging operation, boot-strapping or security related operations. The active background debug mode is in control of the CPU code execution and the BDM firmware is waiting for serial commands sent through the BKGD pin.

### 3.4.2 Memory Map Scheme

#### 3.4.2.1 CPU and BDM Memory Map Scheme

The BDM firmware lookup tables and BDM register memory locations share addresses with other modules; however they are not visible in the memory map during user's code execution. The BDM memory resources are enabled only during the READ_BD and WRITE_BD access cycles to distinguish between accesses to the BDM memory area and accesses to the other modules. (Refer to BDM Block Guide for further details).

When the MCU enters active BDM mode, the BDM firmware lookup tables and the BDM registers become visible in the local memory map in the range 0xFF00-0xFFFF (global address 0x3_FF00 - 0x3_FFFF) and the CPU begins execution of firmware commands or the BDM begins execution of hardware commands. The resources which share memory space with the BDM module will not be visible in the memory map during active BDM mode.

Please note that after the MCU enters active BDM mode the BDM firmware lookup tables and the BDM registers will also be visible between addresses 0xBF00 and 0xBFFF if the PPAGE register contains value of 0x0F.

##### 3.4.2.1.1 Expansion of the Local Address Map

**Expansion of the CPU Local Address Map**

The program page index register in S12GMMC allows accessing up to 256KB of address space in the global memory map by using the four index bits (PPAGE[3:0]) to page 16x16 KB blocks into the program page window located from address 0x8000 to address 0xBFFF in the local CPU memory map.

The page value for the program page window is stored in the PPAGE register. The value of the PPAGE register can be read or written by normal memory accesses as well as by the CALL and RTC instructions.

Control registers, vector space and parts of the on-chip memories are located in unpaged portions of the 64KB local CPU address space.

The starting address of an interrupt service routine must be located in unpaged memory unless the user is certain that the PPAGE register will be set to the appropriate value when the service routine is called. However an interrupt service routine can call other routines that are in paged memory. The upper 16KB block of the local CPU memory space (0xC000–0xFFFF) is unpaged. It is recommended that all reset and interrupt vectors point to locations in this area or to the other unmapped pages sections of the local CPU memory map.

### Expansion of the BDM Local Address Map

PPAGE and BDMPPR register is also used for the expansion of the BDM local address to the global address. These registers can be read and written by the BDM.

The BDM expansion scheme is the same as the CPU expansion scheme.

The four BDMPPR Program Page index bits allow access to the full 256KB address map that can be accessed with 18 address bits.

The BDM program page index register (BDMPPR) is used only when the feature is enabled in BDM and, in the case the CPU is executing a firmware command which uses CPU instructions, or by a BDM hardware commands. See the BDM Block Guide for further details. (see Figure 3-10).

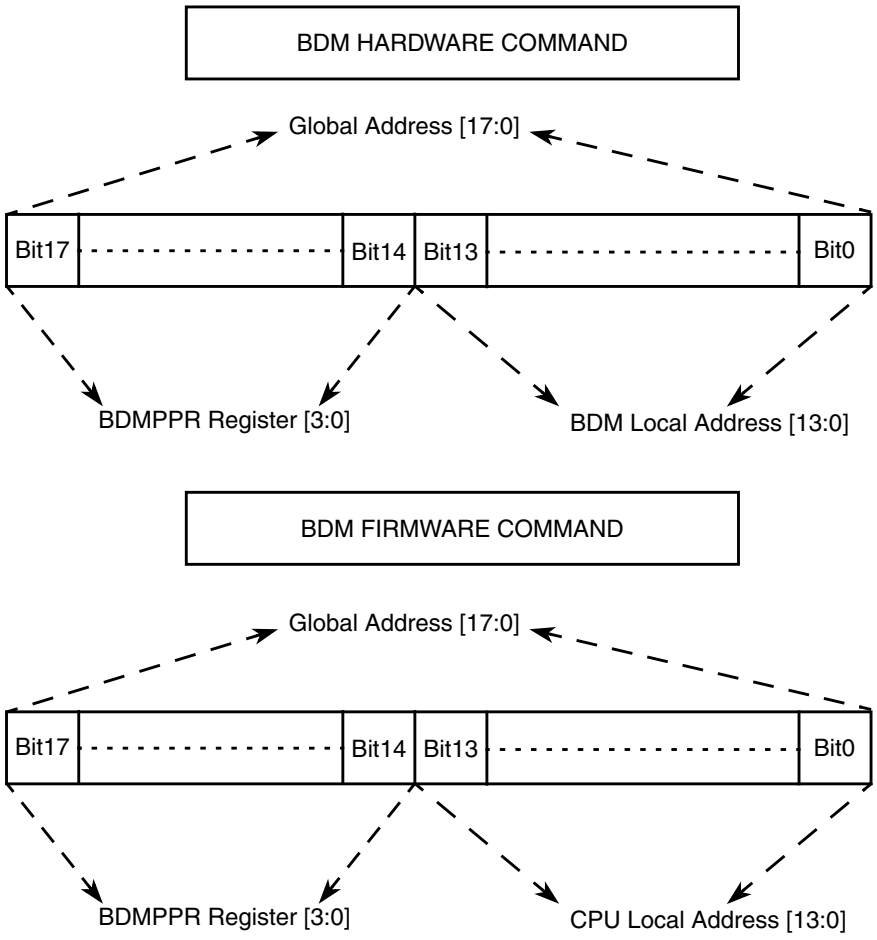



**Figure 3-10. Expansion of BDM local address map**

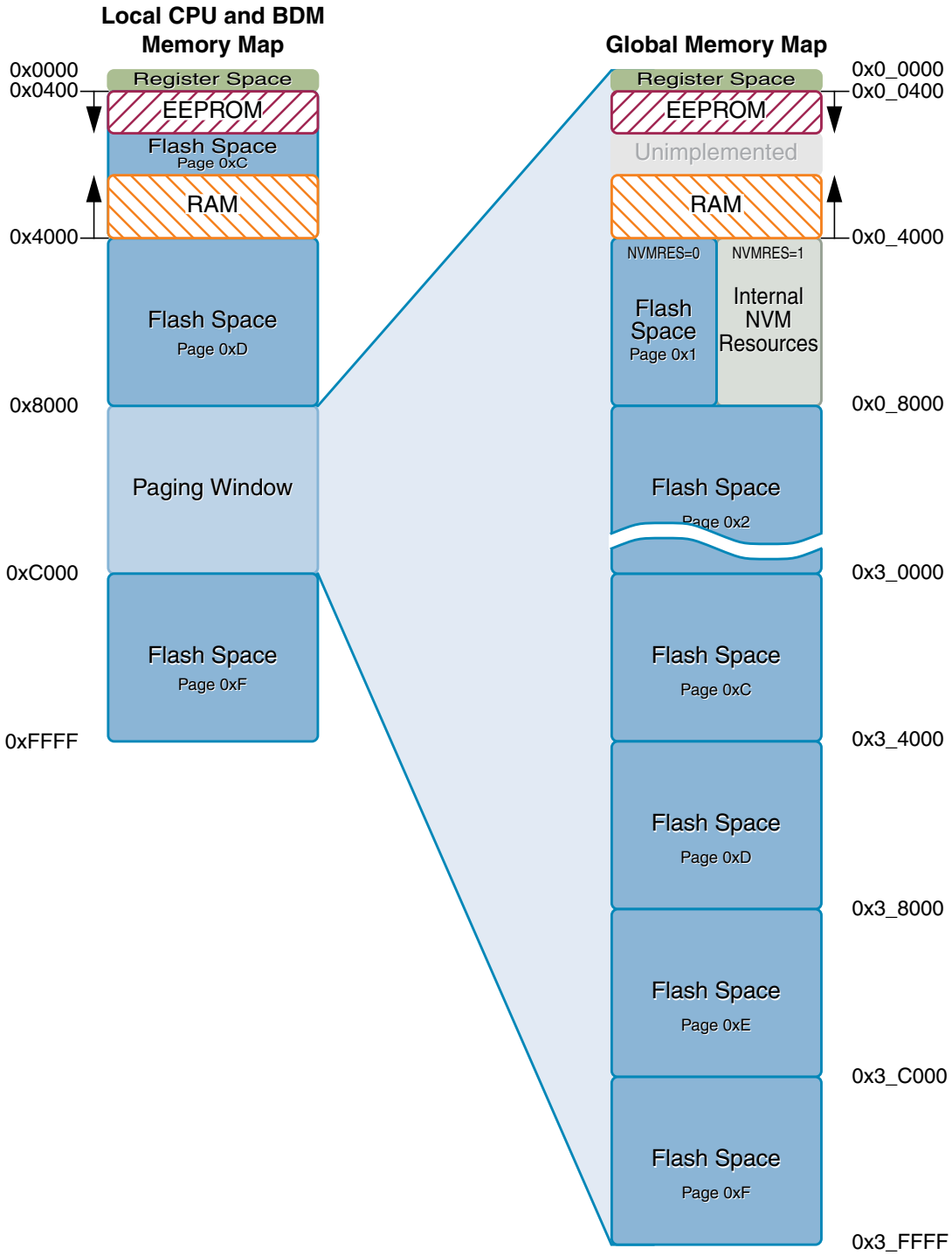



**Figure 3-11. Local to Global Address Mapping**

### 3.4.3 Unimplemented and Reserved Address Ranges

Smaller devices of the-family do not utilize all of the available address space. Address ranges which are not associated with one of the on-chip memories fall into two categories: Unimplemented addresses and reserved addresses.

Unimplemented addresses are not mapped to any of the on-chip memories. The S12GMMC is aware that accesses to these address location have no destination and triggers a system reset (illegal address reset) whenever they are attempted by the CPU. The BDM is not able to trigger illegal address resets.

Reserved addresses are associated with a memory block on the device, even though the memory block does not contain the resources to fill the address space. The S12GMMC is not aware that the associated memory does not physically exist. It does not trigger an illegal address reset when accesses to reserved locations are attempted.

### 3.4.4 Prioritization of Memory Accesses

On  devices, the CPU and the BDM are not able to access the memory in parallel. An arbitration occurs whenever both modules attempt a memory access at the same time. CPU accesses are handled with higher priority than BDM accesses unless the BDM module has been stalled for more then 128 bus cycles. In this case the pending BDM access will be processed immediately.

### 3.4.5 Interrupts

The S12GMMC does not generate any interrupts.

# Chapter 4
# S12 Clock, Reset and Power Management Unit (S12CPMU_UHV)

## Revision History

| Rev. No. (Item No) | Date (Submitted By) | Sections Affected | Substantial Change(s) |
|---|---|---|---|
| V02.02 | 30.July 12 | | Figure “Startup of clock system after Reset”: Corrected PLL clock frequencies to 12.5Mhz and 25MHz (instead of 16MHz and 32Mhz before) |
| V02.03 | 23.Sept 13 | | CPMUCLKS register description: corrected CSAD bit position to Bit number 5<br>Corrected several typos and formats (from shared reference manual review) |
| V02.04 | 23.Sept 13 | | Feature list and Block diagram: External Oscillator now specified from 4MHz to 20MHz (instead 4MHz to 16MHz before) |

## 4.1 Introduction

This specification describes the function of the Clock, Reset and Power Management Unit (S12CPMU_UHV).

- The Pierce oscillator (XOSCLCP) provides a robust, low-noise and low-power external clock source. It is designed for optimal start-up margin with typical crystal oscillators.
- The Voltage regulator (VREGAUTO) operates from the range 6V to 18V. It provides all the required chip internal voltages and voltage monitors.
- The Phase Locked Loop (PLL) provides a highly accurate frequency multiplier with internal filter.
- The Internal Reference Clock (IRC1M) provides a 1MHz internal clock.

### 4.1.1 Features

The Pierce Oscillator (XOSCLCP) contains circuitry to dynamically control current gain in the output amplitude. This ensures a signal with low harmonic distortion, low power and good noise immunity.

- Supports crystals or resonators from 4MHz to 20MHz.
- High noise immunity due to input hysteresis and spike filtering.
- Low RF emissions with peak-to-peak swing limited dynamically
- Transconductance (gm) sized for optimum start-up margin for typical crystals
- Dynamic gain control eliminates the need for external current limiting resistor
- Integrated resistor eliminates the need for external bias resistor
- Low power consumption: Operates from internal 1.8V (nominal) supply, Amplitude control limits power

The Voltage Regulator (VREGAUTO) has the following features:

- Input voltage range from 6 to 18V (nominal operating range)
- Low-voltage detect (LVD) with low-voltage interrupt (LVI)
- Power-on reset (POR)
- Low-voltage reset (LVR)
- On Chip Temperature Sensor and Bandgap Voltage measurement via internal ATD channel.
- High temperature interrupt
- Voltage Regulator providing Full Performance Mode (FPM) and Reduced Performance Mode (RPM)

The Phase Locked Loop (PLL) has the following features:

- highly accurate and phase locked frequency multiplier
- Configurable internal filter for best stability and lock time
- Frequency modulation for defined jitter and reduced emission
- Automatic frequency lock detector
- Interrupt request on entry or exit from locked condition
- Reference clock either external (crystal) or internal square wave (1MHz IRC1M) based.
- PLL stability is sufficient for LIN communication in slave mode, even if using IRC1M as reference clock

The Internal Reference Clock (IRC1M) has the following features:

- Frequency trimming
  (A factory trim value for 1MHz is loaded from Flash Memory into the IRCTRIM register after reset, which can be overwritten by application if required)
- Temperature Coefficient (TC) trimming.
  (A factory trim value is loaded from Flash Memory into the IRCTRIM register to turn off TC trimming after reset. Application can trim the TC if required by overwriting the IRCTRIM register).

Other features of the S12CPMU_UHV include

- Clock monitor to detect loss of crystal
- Autonomous periodical interrupt (API)
- Bus Clock Generator
  - Clock switch to select either PLLCLK or external crystal/resonator based Bus Clock
  - PLLCLK divider to adjust system speed
- System Reset generation from the following possible sources:
  - Power-on reset (POR)
  - Low-voltage reset (LVR)
  - Illegal address access
  - COP time-out
  - Loss of oscillation (clock monitor fail)
  - External pin $\overline{\text{RESET}}$

## 4.1.2 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the S12CPMU_UHV.

### 4.1.2.1 Run Mode

The voltage regulator is in Full Performance Mode (FPM).

**NOTE**

The voltage regulator is active, providing the nominal supply voltages with full current sourcing capability (see also Appendix for VREG electrical parameters). The features ACLK clock source, Low Voltage Interrupt (LVI), Low Voltage Reset (LVR) and Power-On Reset (POR) are available.

The Phase Locked Loop (PLL) is on.

The Internal Reference Clock (IRC1M) is on.

The API is available.

- **PLL Engaged Internal (PEI)**
  - This is the default mode after System Reset and Power-On Reset.
  - The Bus Clock is based on the PLLCLK.
  - After reset the PLL is configured for 50MHz VCOCLK operation
    Post divider is 0x03, so PLLCLK is VCOCLK divided by 4, that is 12.5MHz and Bus Clock is 6.25MHz.
    The PLL can be re-configured for other bus frequencies.
  - The reference clock for the PLL (REFCLK) is based on internal reference clock IRC1M
- **PLL Engaged External (PEE)**
  - The Bus Clock is based on the PLLCLK.
  - This mode can be entered from default mode PEI by performing the following steps:
    - Configure the PLL for desired bus frequency.
    - Program the reference divider (REFDIV[3:0] bits) to divide down oscillator frequency if necessary.
    - Enable the external oscillator (OSCE bit)
    - Wait for oscillator to start up (UPOSC=1) and PLL to lock (LOCK=1)
- **PLL Bypassed External (PBE)**
  - The Bus Clock is based on the Oscillator Clock (OSCCLK).
  - The PLLCLK is always on to qualify the external oscillator clock. Therefore it is necessary to make sure a valid PLL configuration is used for the selected oscillator frequency.
  - This mode can be entered from default mode PEI by performing the following steps:
    - Make sure the PLL configuration is valid for the selected oscillator frequency.

– Enable the external oscillator (OSCE bit)
– Wait for oscillator to start up (UPOSC=1)
– Select the Oscillator Clock (OSCCLK) as source of the Bus Clock (PLLSEL=0).

— The PLLCLK is on and used to qualify the external oscillator clock.

#### 4.1.2.2 Wait Mode

For S12CPMU_UHV Wait Mode is the same as Run Mode.

#### 4.1.2.3 Stop Mode

This mode is entered by executing the CPU STOP instruction.

The voltage regulator is in Reduced Performance Mode (RPM).

**NOTE**

The voltage regulator output voltage may degrade to a lower value than in Full Performance Mode (FPM), additionally the current sourcing capability is substantially reduced (see also Appendix for VREG electrical parameters). Only clock source ACLK is available and the Power On Reset (POR) circuitry is functional. The Low Voltage Interrupt (LVI) and Low Voltage Reset (LVR) are disabled.

The API is available.

The Phase Locked Loop (PLL) is off.

The Internal Reference Clock (IRC1M) is off.

Core Clock, Bus Clock and BDM Clock are stopped.

Depending on the setting of the PSTP and the OSCE bit, Stop Mode can be differentiated between Full Stop Mode (PSTP = 0 or OSCE=0) and Pseudo Stop Mode (PSTP = 1 and OSCE=1). In addition, the behavior of the COP in each mode will change based on the clocking method selected by COPOSCSEL[1:0].

- **Full Stop Mode (PSTP = 0 or OSCE=0)**
  External oscillator (XOSCLCP) is disabled.
  — If COPOSCSEL1=0:
  The COP and RTI counters halt during Full Stop Mode.
  After wake-up from Full Stop Mode the Core Clock and Bus Clock are running on PLLCLK (PLLSEL=1). COP and RTI are running on IRCCLK (COPOSCSEL0=0, RTIOSCSEL=0).
  — If COPOSCSEL1=1:
  The clock for the COP is derived from ACLK (trimmable internal RC-Oscillator clock). During Full Stop Mode the ACLK for the COP can be stopped (COP static) or running (COP active) depending on the setting of bit CSAD. When bit CSAD is set the ACLK clock source for the COP is stopped during Full Stop Mode and COP continues to operate after exit from Full Stop

Mode. For this COP configuration (ACLK clock source, CSAD set) a latency time occurs when entering or exiting (Full, Pseudo) Stop Mode. When bit CSAD is clear the ACLK clock source is on for the COP during Full Stop Mode and COP is operating.
During Full Stop Mode the RTI counter halts.
After wake-up from Full Stop Mode the Core Clock and Bus Clock are running on PLLCLK (PLLSEL=1). The COP runs on ACLK and RTI is running on IRCCLK (COPOSCSEL0=0, RTIOSCSEL=0).

- **Pseudo Stop Mode (PSTP = 1 and OSCE=1)**
  External oscillator (XOSCLCP) continues to run.
  - If COPOSCSEL1=0:
    If the respective enable bits are set (PCE=1 and PRE=1) the COP and RTI will continue to run with a clock derived from the oscillator clock.
    The clock configuration bits PLLSEL, COPOSCSEL0, RTIOSCSEL are unchanged.
  - If COPOSCSEL1=1:
    If the respective enable bit for the RTI is set (PRE=1) the RTI will continue to run with a clock derived from the oscillator clock.
    The clock for the COP is derived from ACLK (trimmable internal RC-Oscillator clock). During Pseudo Stop Mode the ACLK for the COP can be stopped (COP static) or running (COP active) depending on the setting of bit CSAD. When bit CSAD is set the ACLK for the COP is stopped during Pseudo Stop Mode and COP continues to operate after exit from Pseudo Stop Mode. For this COP configuration (ACLK clock source, CSAD set) a latency time occurs when entering or exiting (Pseudo, Full) Stop Mode. When bit CSAD is clear the ACLK clock source is on for the COP during Pseudo Stop Mode and COP is operating.
    The clock configuration bits PLLSEL, COPOSCSEL0, RTIOSCSEL are unchanged.

**NOTE**

When starting up the external oscillator (either by programming OSCE bit to 1 or on exit from Full Stop Mode with OSCE bit already 1) the software must wait for a minimum time equivalent to the startup-time of the external oscillator $t_{UPOSC}$ before entering Pseudo Stop Mode.

#### 4.1.2.4 Freeze Mode (BDM active)

For S12CPMU_UHV Freeze Mode is the same as Run Mode except for RTI and COP which can be stopped in Active BDM Mode with the RSBCK bit in the CPMUCOP register. Additionally the COP can be forced to the maximum time-out period in Active BDM Mode. For details please see also the RSBCK and CR[2:0] bit description field of Table 4-12 in Section 4.3.2.9, “S12CPMU_UHV COP Control Register (CPMUCOP)

### 4.1.3 S12CPMU_UHV Block Diagram

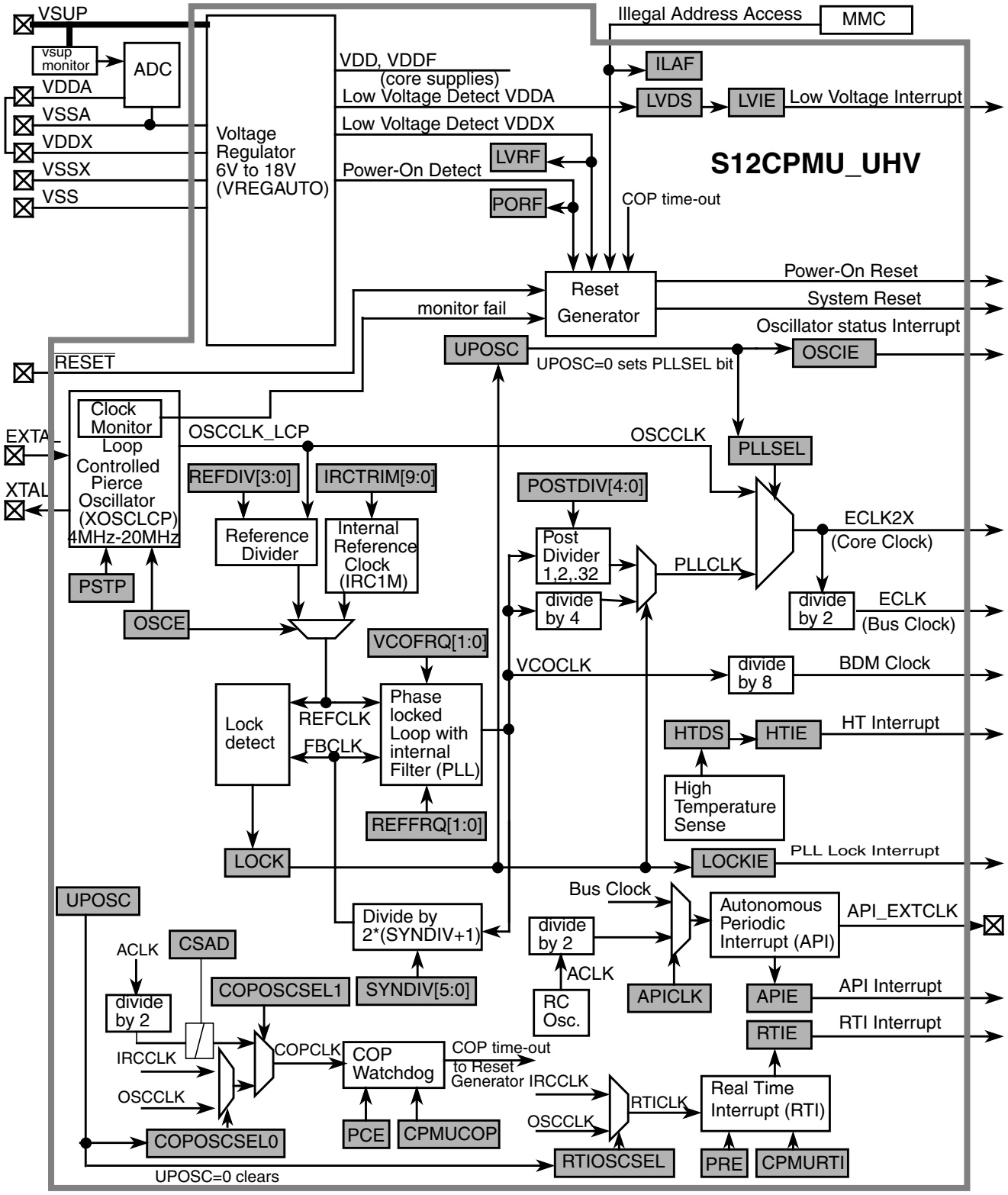



**Figure 4-1. Block diagram of S12CPMU_UHV**

Figure 4-2 shows a block diagram of the XOSCLCP.

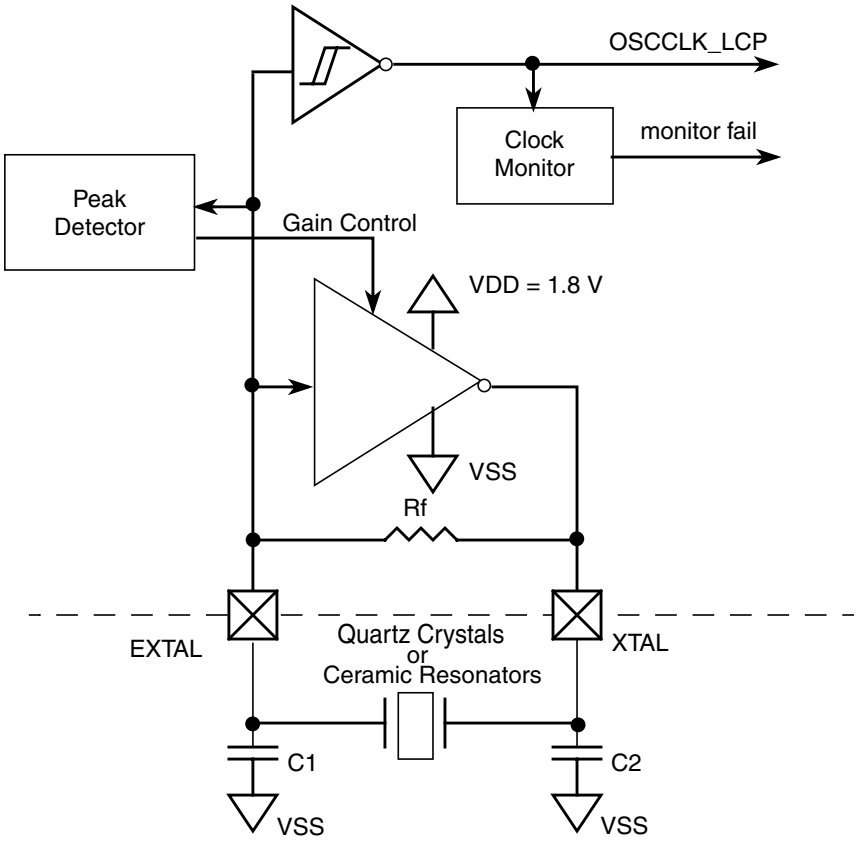



**Figure 4-2. XOSCLCP Block Diagram**

## 4.2 Signal Description

This section lists and describes the signals that connect off chip as well as internal supply nodes and special signals.

### 4.2.1 $\overline{\text{RESET}}$

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

### 4.2.2 EXTAL and XTAL

These pins provide the interface for a crystal to control the internal clock generator circuitry. EXTAL is the input to the crystal oscillator amplifier. XTAL is the output of the crystal oscillator amplifier. If XOSCLCP is enabled, the MCU internal OSCCLK_LCP is derived from the EXTAL input frequency. If OSCE=0, the EXTAL pin is pulled down by an internal resistor of approximately 200 kΩ and the XTAL pin is pulled down by an internal resistor of approximately 700 kΩ.

**NOTE**

Freescale recommends an evaluation of the application board and chosen resonator or crystal by the resonator or crystal supplier.
The loop controlled circuit (XOSCLCP) is not suited for overtone resonators and crystals.

### 4.2.3 VSUP — Regulator Power Input Pin

Pin VSUP is the power input of VREGAUTO. All currents sourced into the regulator loads flow through this pin.

An appropriate reverse battery protection network consisting of a diode and capacitors is recommended.

### 4.2.4 VDDA, VSSA — Regulator Reference Supply Pins

Pins VDDA and VSSA, are used to supply the analog parts of the regulator.
Internal precision reference circuits are supplied from these signals.

A local decoupling capacitor between VDDA and VSSA according to the electrical specification is required. Additionally a bigger tank capacitor is required on the 5 Volt supply network as well to ensure Voltage regulator stability.

VDDA has to be connected externally to VDDX.

### 4.2.5 VDDX, VSSX— Pad Supply Pins

This supply domain is monitored by the Low Voltage Reset circuit.

A local decoupling capacitor between VDDX and VSSX according to the electrical specification is required.

VDDX has to be connected externally to VDDA.

### 4.2.6 VSS— Ground Pin

VSS is the ground pin for the core logic. On the board VSSX, VSSA and VSS need to be connected together to the application ground.

### 4.2.7 API_EXTCLK — API external clock output pin

This pin provides the signal selected via APIES and is enabled with APIEA bit. See the device specification if this clock output is available on this device and to which pin it might be connects.

### 4.2.8 VDD— Internal Regulator Output Supply (Core Logic)

Node VDD is a device internal supply output of the voltage regulator that provides the power supply for the core logic.

This supply domain is monitored by the Low Voltage Reset circuit.

### 4.2.9 VDDF— Internal Regulator Output Supply (NVM Logic)

Node VDDF is a device internal supply output of the voltage regulator that provides the power supply for the NVM logic.

This supply domain is monitored by the Low Voltage Reset circuit.

### 4.2.10 TEMPSENSE — Internal Temperature Sensor Output Voltage

Depending on the VSEL setting either the voltage level generated by the temperature sensor or the VREG bandgap voltage is driven to a special channel input of the ATD Converter. See device level specification for connectivity of ATD special channels.

## 4.3 Memory Map and Registers

This section provides a detailed description of all registers accessible in the S12CPMU_UHV.

### 4.3.1 Module Memory Map

The S12CPMU_UHV registers are shown in Figure 4-3.

| Address | Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0034 | CPMU SYNR | R | VCOFRQ[1:0] | | SYNDIV[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x0035 | CPMU REFDIV | R | REFFRQ[1:0] | | 0 | 0 | REFDIV[3:0] | | | |
| | | W | | | | | | | | |
| 0x0036 | CPMU POSTDIV | R | 0 | 0 | 0 | POSTDIV[4:0] | | | | |
| | | W | | | | | | | | |
| 0x0037 | CPMUFLG | R | RTIF | PORF | LVRF | LOCKIF | LOCK | ILAF | OSCIF | UPOSC |
| | | W | | | | | | | | |
| 0x0038 | CPMUINT | R | RTIE | 0 | 0 | LOCKIE | 0 | 0 | OSCIE | 0 |
| | | W | | | | | | | | |
| 0x0039 | CPMUCLKS | R | PLLSEL | PSTP | CSAD | COP OSCSEL1 | PRE | PCE | RTI OSCSEL | COP OSCSEL0 |
| | | W | | | | | | | | |
| 0x003A | CPMUPLL | R | 0 | 0 | FM1 | FM0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x003B | CPMURTI | R | RTDEC | RTR6 | RTR5 | RTR4 | RTR3 | RTR2 | RTR1 | RTR0 |
| | | W | | | | | | | | |
| 0x003C | CPMUCOP | R | WCOP | RSBCK | 0 | 0 | 0 | CR2 | CR1 | CR0 |
| | | W | | | WRTMASK | | | | | |
| 0x003D | RESERVED CPMUTEST0 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x003E | RESERVED CPMUTEST1 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x003F | CPMU 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 |
| 0x02F0 | CPMU HTCTL | R | 0 | 0 | VSEL | 0 | HTE | HTDS | HTIE | HTIF |
| | | W | | | | | | | | |
| 0x02F1 | CPMU LVCTL | R | 0 | 0 | 0 | 0 | 0 | LVDS | LVIE | LVIF |
| | | W | | | | | | | | |
| 0x02F2 | CPMU APICTL | R | APICLK | 0 | 0 | APIES | APIEA | APIFE | APIE | APIF |
| | | W | | | | | | | | |
| | | | = Unimplemented or Reserved | | | | | | | |

**Figure 4-3. CPMU Register Summary**

| Address | Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02F3 | CPMUACLKTR | R | ACLKTR5 | ACLKTR4 | ACLKTR3 | ACLKTR2 | ACLKTR1 | ACLKTR0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x02F4 | CPMUAPIRH | R | APIR15 | APIR14 | APIR13 | APIR12 | APIR11 | APIR10 | APIR9 | APIR8 |
| | | W | | | | | | | | |
| 0x02F5 | CPMUAPIRL | R | APIR7 | APIR6 | APIR5 | APIR4 | APIR3 | APIR2 | APIR1 | APIR0 |
| | | W | | | | | | | | |
| 0x02F6 | RESERVED CPMUTEST3 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x02F7 | CPMUHTTR | R | HTOE | 0 | 0 | 0 | HTTR3 | HTTR2 | HTTR1 | HTTR0 |
| | | W | | | | | | | | |
| 0x02F8 | CPMU IRCTRIMH | R | TCTRIM[4:0] | | | | | 0 | IRCTRIM[9:8] | |
| | | W | | | | | | | | |
| 0x02F9 | CPMU IRCTRIML | R | IRCTRIM[7:0] | | | | | | | |
| | | W | | | | | | | | |
| 0x02FA | CPMUOSC | R | OSCE | Reserved | 0 | Reserved | | | | |
| | | W | | | | | | | | |
| 0x02FB | CPMUPROT | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PROT |
| | | W | | | | | | | | |
| 0x02FC | RESERVED CPMUTEST2 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

= Unimplemented or Reserved

**Figure 4-3. CPMU Register Summary**

## 4.3.2 Register Descriptions

This section describes all the S12CPMU_UHV registers and their individual bits.

Address order is as listed in Figure 4-3

### 4.3.2.1 S12CPMU_UHV Synthesizer Register (CPMUSYNR)

The CPMUSYNR register controls the multiplication factor of the PLL and selects the VCO frequency range.

0x0034

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | VCOFRQ[1:0] | | | | SYNDIV[5:0] | | | |
| Reset | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |

**Figure 4-4. S12CPMU_UHV Synthesizer Register (CPMUSYNR)**

Read: Anytime

Write: If PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register), then write anytime. Else write has no effect.

**NOTE**

Writing to this register clears the LOCK and UPOSC status bits.

If PLL has locked (LOCK=1) $f_{VCO} = 2 \times f_{REF} \times (SYNDIV + 1)$

**NOTE**

$f_{VCO}$ must be within the specified VCO frequency lock range. Bus frequency $f_{bus}$ must not exceed the specified maximum.

The VCOFRQ[1:0] bits are used to configure the VCO gain for optimal stability and lock time. For correct PLL operation the VCOFRQ[1:0] bits have to be selected according to the actual target VCOCLK frequency as shown in Table 4-1. Setting the VCOFRQ[1:0] bits incorrectly can result in a non functional PLL (no locking and/or insufficient stability).

**Table 4-1. VCO Clock Frequency Selection**

| VCOCLK Frequency Ranges | VCOFRQ[1:0] |
|---|---|
| 32MHz <= $f_{VCO}$<= 48MHz | 00 |
| 48MHz < $f_{VCO}$<= 50MHz | 01 |
| Reserved | 10 |
| Reserved | 11 |

### 4.3.2.2 S12CPMU_UHV Reference Divider Register (CPMUREFDIV)

The CPMUREFDIV register provides a finer granularity for the PLL multiplier steps when using the external oscillator as reference.

0x0035

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | REFFRQ[1:0] | | 0 | 0 | REFDIV[3:0] | | | |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

**Figure 4-5. S12CPMU_UHV Reference Divider Register (CPMUREFDIV)**

Read: Anytime

Write: If PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register), then write anytime. Else write has no effect.

**NOTE**

Write to this register clears the LOCK and UPOSC status bits.

If XOSCLCP is enabled (OSCE=1) $$f_{REF} = \frac{f_{OSC}}{(REFDIV + 1)}$$

If XOSCLCP is disabled (OSCE=0) $$f_{REF} = f_{IRC1M}$$

The REFFRQ[1:0] bits are used to configure the internal PLL filter for optimal stability and lock time. For correct PLL operation the REFFRQ[1:0] bits have to be selected according to the actual REFCLK frequency as shown in Table 4-2.

If IRC1M is selected as REFCLK (OSCE=0) the PLL filter is fixed configured for the 1MHz <= $f_{REF}$ <= 2MHz range. The bits can still be written but will have no effect on the PLL filter configuration.

For OSCE=1, setting the REFFRQ[1:0] bits incorrectly can result in a non functional PLL (no locking and/or insufficient stability).

**Table 4-2. Reference Clock Frequency Selection if OSC_LCP is enabled**

| REFCLK Frequency Ranges (OSCE=1) | REFFRQ[1:0] |
|---|---|
| 1MHz <= $f_{REF}$ <= 2MHz | 00 |
| 2MHz < $f_{REF}$ <= 6MHz | 01 |
| 6MHz < $f_{REF}$ <= 12MHz | 10 |
| $f_{REF}$ >12MHz | 11 |

### 4.3.2.3 S12CPMU_UHV Post Divider Register (CPMUPOSTDIV)

The POSTDIV register controls the frequency ratio between the VCOCLK and the PLLCLK.

0x0036

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | POSTDIV[4:0] | | | | |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

= Unimplemented or Reserved

**Figure 4-6. S12CPMU_UHV Post Divider Register (CPMUPOSTDIV)**

Read: Anytime

Write: If PLLSEL=1 write anytime, else write has no effect

If PLL is locked (LOCK=1) $f_{PLL} = \frac{f_{VCO}}{(POSTDIV + 1)}$

If PLL is not locked (LOCK=0) $f_{PLL} = \frac{f_{VCO}}{4}$

If PLL is selected (PLLSEL=1) $f_{bus} = \frac{f_{PLL}}{2}$

### 4.3.2.4 S12CPMU_UHV Flags Register (CPMUFLG)

This register provides S12CPMU_UHV status bits and flags.

0x0037

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | RTIF | PORF | LVRF | LOCKIF | LOCK | ILAF | OSCIF | UPOSC |
| W | | | | | | | | |
| Reset | 0 | Note 1 | Note 2 | 0 | 0 | Note 3 | 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. Set by power on reset.
3. ILAF is set to 1 when an illegal address reset occurs. Unaffected by System Reset. Cleared by power on reset.

= Unimplemented or Reserved

**Figure 4-7. S12CPMU_UHV Flags Register (CPMUFLG)**

Read: Anytime

Write: Refer to each bit for individual write conditions

**Table 4-3. CPMUFLG Field Descriptions**

| Field | Description |
|---|---|
| 7<br>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.<br>0 RTI time-out has not yet occurred.<br>1 RTI time-out has occurred. |
| 6<br>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.<br>0 Power on reset has not occurred.<br>1 Power on reset has occurred. |
| 5<br>LVRF | **Low Voltage Reset Flag** — 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.<br>0 Low voltage reset has not occurred.<br>1 Low voltage reset has occurred. |
| 4<br>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.<br>0 No change in LOCK bit.<br>1 LOCK bit has changed. |
| 3<br>LOCK | **Lock Status Bit** — LOCK reflects the current state of PLL lock condition. Writes have no effect.While PLL is unlocked (LOCK=0) $f_{PLL}$ is $f_{VCO}$ / 4 to protect the system from high core clock frequencies during the PLL stabilization time tlock.<br>0 VCOCLK is not within the desired tolerance of the target frequency.<br>$f_{PLL} = f_{VCO}/4$.<br>1 VCOCLK is within the desired tolerance of the target frequency.<br>$f_{PLL} = f_{VCO}/(POSTDIV+1)$. |
| 2<br>ILAF | **Illegal Address Reset Flag** — ILAF is set to 1 when an illegal address reset occurs.Refer to MMC chapter for details.This flag can only be cleared by writing a 1. Writing a 0 has no effect.<br>0 Illegal address reset has not occurred.<br>1 Illegal address reset has occurred. |
| 1<br>OSCIF | **Oscillator Interrupt Flag** — OSCIF is set to 1 when UPOSC status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect.If enabled (OSCIE=1), OSCIF causes an interrupt request.<br>0 No change in UPOSC bit.<br>1 UPOSC bit has changed. |
| 0<br>UPOSC | **Oscillator Status Bit —** UPOSC reflects the status of the oscillator. Writes have no effect. Entering Full Stop Mode UPOSC is cleared.<br>0 The oscillator is off or oscillation is not qualified by the PLL.<br>1 The oscillator is qualified by the PLL. |

### 4.3.2.5 S12CPMU_UHV Interrupt Enable Register (CPMUINT)

This register enables S12CPMU_UHV interrupt requests.

0x0038

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | RTIE | 0 | 0 | LOCKIE | 0 | 0 | OSCIE | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 4-8. S12CPMU_UHV Interrupt Enable Register (CPMUINT)**

Read: Anytime

Write: Anytime

**Table 4-4. CPMUINT Field Descriptions**

| Field | Description |
|---|---|
| 7<br>RTIE | **Real Time Interrupt Enable Bit**<br>0 Interrupt requests from RTI are disabled.<br>1 Interrupt will be requested whenever RTIF is set. |
| 4<br>LOCKIE | **PLL Lock Interrupt Enable Bit**<br>0 PLL LOCK interrupt requests are disabled.<br>1 Interrupt will be requested whenever LOCKIF is set. |
| 1<br>OSCIE | **Oscillator Corrupt Interrupt Enable Bit**<br>0 Oscillator Corrupt interrupt requests are disabled.<br>1 Interrupt will be requested whenever OSCIF is set. |

### 4.3.2.6 S12CPMU_UHV Clock Select Register (CPMUCLKS)

This register controls S12CPMU_UHV clock selection.

0x0039

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | PLLSEL | PSTP | CSAD | COP OSCSEL1 | PRE | PCE | RTI OSCSEL | COP OSCSEL0 |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 4-9. S12CPMU_UHV Clock Select Register (CPMUCLKS)**

Read: Anytime

Write:

5. Only possible if PROT=0 (CPMUPROT register) in all MCU Modes (Normal and Special Mode).
6. All bits in Special Mode (if PROT=0).
7. PLLSEL, PSTP, PRE, PCE, RTIOSCSEL: In Normal Mode (if PROT=0).
8. CSAD: In Normal Mode (if PROT=0) until CPMUCOP write once has taken place.
9. COPOSCSEL0: In Normal Mode (if PROT=0) until CPMUCOP write once has taken place. If COPOSCSEL0 was cleared by UPOSC=0 (entering Full Stop Mode with COPOSCSEL0=1 or insufficient OSCCLK quality), then COPOSCSEL0 can be set once again.
10. COPOSCSEL1: In Normal Mode (if PROT=0) until CPMUCOP write once has taken place. COPOSCSEL1 will not be cleared by UPOSC=0 (entering Full Stop Mode with COPOSCSEL1=1 or insufficient OSCCLK quality if OSCCLK is used as clock source for other clock domains: for instance core clock etc.).

**NOTE**

After writing CPMUCLKS register, it is strongly recommended to read back CPMUCLKS register to make sure that write of PLLSEL, RTIOSCSEL and COPOSCSEL was successful.

**Table 4-5. CPMUCLKS Descriptions**

| Field | Description |
|---|---|
| 7<br>PLLSEL | **PLL Select Bit**<br>This bit selects the PLLCLK as source of the System Clocks (Core Clock and Bus Clock).<br>PLLSEL can only be set to 0, if UPOSC=1.<br>UPOSC= 0 sets the PLLSEL bit.<br>Entering Full Stop Mode sets the PLLSEL bit.<br>0 System clocks are derived from OSCCLK if oscillator is up (UPOSC=1, $f_{bus} = f_{osc} / 2$).<br>1 System clocks are derived from PLLCLK, $f_{bus} = f_{PLL} / 2$. |
| 6<br>PSTP | **Pseudo Stop Bit**<br>This bit controls the functionality of the oscillator during Stop Mode.<br>0 Oscillator is disabled in Stop Mode (Full Stop Mode).<br>1 Oscillator continues to run in Stop Mode (Pseudo Stop Mode), option to run RTI and COP.<br>**Note:** Pseudo Stop Mode 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.<br>**Note:** When starting up the external oscillator (either by programming OSCE bit to 1 or on exit from Full Stop Mode with OSCE bit already 1) the software must wait for a minimum time equivalent to the startup-time of the external oscillator $t_{UPOSC}$ before entering Pseudo Stop Mode. |
| 5<br>CSAD | **COP in Stop Mode ACLK Disable** — This bit disables the ACLK for the COP in Stop Mode. Hence the COP is static while in Stop Mode and continues to operate after exit from Stop Mode.<br>Due to clock domain crossing synchronization there is a latency time to enter and exit Stop Mode if COP clock source is ACLK and this clock is stopped in Stop Mode. This maximum latency time is 4 ACLK cycles which must be added to the Stop Mode recovery time $t_{STP_REC}$ from exit of current Stop Mode to entry of next Stop Mode. This latency time occurs no matter which Stop Mode (Full, Pseudo) is currently exited or entered next. After exit from Stop Mode (Pseudo, Full) for 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter increments at each Stop Mode exit.<br>This bit does not influence the ACLK for the API.<br>0 COP running in Stop Mode (ACLK for COP enabled in Stop Mode).<br>1 COP stopped in Stop Mode (ACLK for COP disabled in Stop Mode) |
| 4<br>COP<br>OSCSEL1 | **COP Clock Select 1** — COPOSCSEL0 and COPOSCSEL1 combined determine the clock source to the COP (see also Table 4-6).<br>If COPOSCSEL1 = 1, COPOSCSEL0 has no effect regarding clock select and changing the COPOSCSEL0 bit does not re-start the COP time-out period.<br>COPOSCSEL1 selects the clock source to the COP to be either ACLK (derived from trimmable internal RC-Oscillator) or clock selected via COPOSCSEL0 (IRCCLK or OSCCLK).<br>Changing the COPOSCSEL1 bit re-starts the COP time-out period.<br>COPOSCSEL1 can be set independent from value of UPOSC.<br>UPOSC= 0 does not clear the COPOSCSEL1 bit.<br>0 COP clock source defined by COPOSCSEL0<br>1 COP clock source is ACLK derived from a trimmable internal RC-Oscillator |
| 3<br>PRE | **RTI Enable During Pseudo Stop Bit** — PRE enables the RTI during Pseudo Stop Mode.<br>0 RTI stops running during Pseudo Stop Mode.<br>1 RTI continues running during Pseudo Stop Mode if RTIOSCSEL=1.<br>**Note:** If PRE=0 or RTIOSCSEL=0 then the RTI will go static while Stop Mode is active. The RTI counter will not be reset. |
| 2<br>PCE | **COP Enable During Pseudo Stop Bit** — PCE enables the COP during Pseudo Stop Mode.<br>0 COP stops running during Pseudo Stop Mode<br>1 COP continues running during Pseudo Stop Mode if COPOSCSEL=1<br>**Note:** If PCE=0 or COPOSCSEL=0 then the COP will go static while Stop Mode is active. The COP counter will not be reset. |

**Table 4-5. CPMUCLKS Descriptions (continued)**

| Field | Description |
|---|---|
| 1<br>RTIOSCSEL | **RTI Clock Select**— RTIOSCSEL selects the clock source to the RTI. Either IRCCLK or OSCCLK. Changing the RTIOSCSEL bit re-starts the RTI time-out period.<br>RTIOSCSEL can only be set to 1, if UPOSC=1.<br>UPOSC= 0 clears the RTIOSCSEL bit.<br>0 RTI clock source is IRCCLK.<br>1 RTI clock source is OSCCLK. |
| 0<br>COP<br>OSCSEL0 | **COP Clock Select 0** — COPOSCSEL0 and COPOSCSEL1 combined determine the clock source to the COP (see also Table 4-6)<br>If COPOSCSEL1 = 1, COPOSCSEL0 has no effect regarding clock select and changing the COPOSCSEL0 bit does not re-start the COP time-out period.<br>When COPOSCSEL1=0,COPOSCSEL0 selects the clock source to the COP to be either IRCCLK or OSCCLK. Changing the COPOSCSEL0 bit re-starts the COP time-out period.<br>COPOSCSEL0 can only be set to 1, if UPOSC=1.<br>UPOSC= 0 clears the COPOSCSEL0 bit.<br>0 COP clock source is IRCCLK.<br>1 COP clock source is OSCCLK |

**Table 4-6. COPOSCSEL1, COPOSCSEL0 clock source select description**

| COPOSCSEL1 | COPOSCSEL0 | COP clock source |
|---|---|---|
| 0 | 0 | IRCCLK |
| 0 | 1 | OSCCLK |
| 1 | x | ACLK |

### 4.3.2.7 S12CPMU_UHV PLL Control Register (CPMUPLL)

This register controls the PLL functionality.

0x003A

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | FM1 | FM0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 4-10. S12CPMU_UHV PLL Control Register (CPMUPLL)**

Read: Anytime

Write: Anytime if PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register). Else write has no effect.

**NOTE**

Write to this register clears the LOCK and UPOSC status bits.

**NOTE**

Care should be taken to ensure that the bus frequency does not exceed the specified maximum when frequency modulation is enabled.

**Table 4-7. CPMUPLL Field Descriptions**

| Field | Description |
|---|---|
| 5, 4<br>FM1, FM0 | PLL **Frequency Modulation Enable Bits** — FM1 and FM0 enable frequency modulation on the VCOCLK. This is to reduce noise emission. The modulation frequency is $f_{ref}$ divided by 16. See Table 4-8 for coding. |

**Table 4-8. FM Amplitude selection**

| FM1 | FM0 | FM Amplitude / $f_{VCO}$ Variation |
|---|---|---|
| 0 | 0 | FM off |
| 0 | 1 | ±1% |
| 1 | 0 | ±2% |
| 1 | 1 | ±4% |

### 4.3.2.8 S12CPMU_UHV RTI Control Register (CPMURTI)

This register selects the time-out period for the Real Time Interrupt.

The clock source for the RTI is either IRCCLK or OSCCLK depending on the setting of the RTIOSCSEL bit. In Stop Mode with PSTP=1 (Pseudo Stop Mode) and RTIOSCSEL=1 the RTI continues to run, else the RTI counter halts in Stop Mode.

0x003B

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | RTDEC | RTR6 | RTR5 | RTR4 | RTR3 | RTR2 | RTR1 | RTR0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 4-11. S12CPMU_UHV RTI Control Register (CPMURTI)**

Read: Anytime

Write: Anytime

**NOTE**

A write to this register starts the RTI time-out period. A change of the RTIOSCSEL bit (writing a different value or loosing UPOSC status) re-starts the RTI time-out period.

**Table 4-9. CPMURTI Field Descriptions**

| Field | Description |
|---|---|
| 7<br>RTDEC | **Decimal or Binary Divider Select Bit** — RTDEC selects decimal or binary based prescaler values.<br>0 Binary based divider value. See Table 4-10<br>1 Decimal based divider value. See Table 4-11 |
| 6–4<br>RTR[6:4] | **Real Time Interrupt Prescale Rate Select Bits** — These bits select the prescale rate for the RTI.See Table 4-10 and Table 4-11. |
| 3–0<br>RTR[3:0] | **Real Time Interrupt Modulus Counter Select Bits** — These bits select the modulus counter target value to provide additional granularity.Table 4-10 and Table 4-11 show all possible divide values selectable by the CPMURTI register. |

**Table 4-10. RTI Frequency Divide Rates for RTDEC = 0**

| 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[1] | $2^{10}$ | $2^{11}$ | $2^{12}$ | $2^{13}$ | $2^{14}$ | $2^{15}$ | $2^{16}$ |
| **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 | $16x2^{10}$ | $16x2^{11}$ | $16x2^{12}$ | $16x2^{13}$ | $16x2^{14}$ | $16x2^{15}$ | $16x2^{16}$ |

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

**Table 4-11. RTI Frequency Divide Rates for RTDEC=1**

| RTR[3:0] | RTR[6:4] = | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 000 ($1x10^3$) | 001 ($2x10^3$) | 010 ($5x10^3$) | 011 ($10x10^3$) | 100 ($20x10^3$) | 101 ($50x10^3$) | 110 ($100x10^3$) | 111 ($200x10^3$) |
| 0000 (÷1) | $1x10^3$ | $2x10^3$ | $5x10^3$ | $10x10^3$ | $20x10^3$ | $50x10^3$ | $100x10^3$ | $200x10^3$ |
| 0001 (÷2) | $2x10^3$ | $4x10^3$ | $10x10^3$ | $20x10^3$ | $40x10^3$ | $100x10^3$ | $200x10^3$ | $400x10^3$ |
| 0010 (÷3) | $3x10^3$ | $6x10^3$ | $15x10^3$ | $30x10^3$ | $60x10^3$ | $150x10^3$ | $300x10^3$ | $600x10^3$ |
| 0011 (÷4) | $4x10^3$ | $8x10^3$ | $20x10^3$ | $40x10^3$ | $80x10^3$ | $200x10^3$ | $400x10^3$ | $800x10^3$ |
| 0100 (÷5) | $5x10^3$ | $10x10^3$ | $25x10^3$ | $50x10^3$ | $100x10^3$ | $250x10^3$ | $500x10^3$ | $1x10^6$ |
| 0101 (÷6) | $6x10^3$ | $12x10^3$ | $30x10^3$ | $60x10^3$ | $120x10^3$ | $300x10^3$ | $600x10^3$ | $1.2x10^6$ |
| 0110 (÷7) | $7x10^3$ | $14x10^3$ | $35x10^3$ | $70x10^3$ | $140x10^3$ | $350x10^3$ | $700x10^3$ | $1.4x10^6$ |
| 0111 (÷8) | $8x10^3$ | $16x10^3$ | $40x10^3$ | $80x10^3$ | $160x10^3$ | $400x10^3$ | $800x10^3$ | $1.6x10^6$ |
| 1000 (÷9) | $9x10^3$ | $18x10^3$ | $45x10^3$ | $90x10^3$ | $180x10^3$ | $450x10^3$ | $900x10^3$ | $1.8x10^6$ |
| 1001 (÷10) | $10\ x10^3$ | $20x10^3$ | $50x10^3$ | $100x10^3$ | $200x10^3$ | $500x10^3$ | $1x10^6$ | $2x10^6$ |
| 1010 (÷11) | $11\ x10^3$ | $22x10^3$ | $55x10^3$ | $110x10^3$ | $220x10^3$ | $550x10^3$ | $1.1x10^6$ | $2.2x10^6$ |
| 1011 (÷12) | $12x10^3$ | $24x10^3$ | $60x10^3$ | $120x10^3$ | $240x10^3$ | $600x10^3$ | $1.2x10^6$ | $2.4x10^6$ |
| 1100 (÷13) | $13x10^3$ | $26x10^3$ | $65x10^3$ | $130x10^3$ | $260x10^3$ | $650x10^3$ | $1.3x10^6$ | $2.6x10^6$ |
| 1101 (÷14) | $14x10^3$ | $28x10^3$ | $70x10^3$ | $140x10^3$ | $280x10^3$ | $700x10^3$ | $1.4x10^6$ | $2.8x10^6$ |
| 1110 (÷15) | $15x10^3$ | $30x10^3$ | $75x10^3$ | $150x10^3$ | $300x10^3$ | $750x10^3$ | $1.5x10^6$ | $3x10^6$ |
| 1111 (÷16) | $16x10^3$ | $32x10^3$ | $80x10^3$ | $160x10^3$ | $320x10^3$ | $800x10^3$ | $1.6x10^6$ | $3.2x10^6$ |

### 4.3.2.9 S12CPMU_UHV COP Control Register (CPMUCOP)

This register controls the COP (Computer Operating Properly) watchdog.

The clock source for the COP is either ACLK, IRCCLK or OSCCLK depending on the setting of the COPOSCSEL0 and COPOSCSEL1 bit (see also Table 4-6).
In Stop Mode with PSTP=1 (Pseudo Stop Mode), COPOSCSEL0=1 and COPOSCEL1=0 and PCE=1 the COP continues to run, else the COP counter halts in Stop Mode with COPOSCSEL1 =0.
In Full Stop Mode and Pseudo Stop Mode with COPOSCSEL1=1 the COP continues to run.

0x003C

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

After de-assert of System Reset the values are automatically loaded from the Flash memory. See Device specification for details.

= Unimplemented or Reserved

**Figure 4-12. S12CPMU_UHV COP Control Register (CPMUCOP)**

Read: Anytime

Write:

1. RSBCK: Anytime in Special Mode; write to “1” but not to “0” in Normal Mode
2. WCOP, CR2, CR1, CR0:
   — Anytime in Special Mode, when WRTMASK is 0, otherwise it has no effect
   — Write once in Normal Mode, when WRTMASK is 0, otherwise it has no effect.
     – Writing CR[2:0] to “000” has no effect, but counts for the “write once” condition.
     – Writing WCOP to “0” has no effect, but counts for the “write once” condition.

When a non-zero value is loaded from Flash to CR[2:0] the COP time-out period is started.

A change of the COPOSCSEL0 or COPOSCSEL1 bit (writing a different value) or loosing UPOSC status while COPOSCSEL1 is clear and COPOSCSEL0 is set, re-starts the COP time-out period.

In Normal Mode the COP time-out period is restarted if either of these conditions is true:

1. Writing a non-zero value to CR[2:0] (anytime in special mode, once in normal mode) with WRTMASK = 0.
2. Writing WCOP bit (anytime in Special Mode, once in Normal Mode) with WRTMASK = 0.
3. Changing RSBCK bit from “0” to “1”.

In Special Mode, any write access to CPMUCOP register restarts the COP time-out period.

**Table 4-12. CPMUCOP Field Descriptions**

| Field | Description |
|---|---|
| 7<br>WCOP | **Window COP Mode Bit** — When set, a write to the CPMUARMCOP register must occur in the last 25% of the selected period.A write during the first 75% of the selected period generates a COP reset.As long as all writes occur during this window, $55 can be written as often as desired.Once $AA is written after the $55, the time-out logic restarts and the user must wait until the next window before writing to CPMUARMCOP. Table 4-13 shows the duration of this window for the seven available COP rates.<br>0 Normal COP operation<br>1 Window COP operation |
| 6<br>RSBCK | **COP and RTI Stop in Active BDM Mode Bit**<br>0 Allows the COP and RTI to keep running in Active BDM mode.<br>1 Stops the COP and RTI counters whenever the part is in Active BDM mode. |
| 5<br>WRTMASK | **Write Mask for WCOP and CR[2:0] Bit** — This write-only bit serves as a mask for the WCOP and CR[2:0] bits while writing the CPMUCOP register. It is intended for BDM writing the RSBCK without changing the content of WCOP and CR[2:0].<br>0 Write of WCOP and CR[2:0] has an effect with this write of CPMUCOP<br>1 Write of WCOP and CR[2:0] has no effect with this write of CPMUCOP.<br>(Does not count for “write once”.) |
| 2–0<br>CR[2:0] | **COP Watchdog Timer Rate Select** — These bits select the COP time-out rate (see Table 4-13 and Table 4-14). 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) initializing the COP counter via the CPMUARMCOP register.<br>While all of the following four conditions are true the CR[2:0], WCOP bits are ignored and the COP operates at highest time-out period ($2^{24}$ cycles) in normal COP mode (Window COP mode disabled):<br>1) COP is enabled (CR[2:0] is not 000)<br>2) BDM mode active<br>3) RSBCK = 0<br>4) Operation in Special Mode |

**Table 4-13. COP Watchdog Rates if COPOSCSEL1=0. (default out of reset)**

| CR2 | CR1 | CR0 | COPCLK Cycles to time-out (COPCLK is either IRCCLK or OSCCLK depending on the COPOSCSEL0 bit) |
|---|---|---|---|
| 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 | $2^{24}$ |

**Table 4-14. COP Watchdog Rates if COPOSCSEL1=1.**

| CR2 | CR1 | CR0 | COPCLK Cycles to time-out (COPCLK is ACLK divided by 2) |
|---|---|---|---|
| 0 | 0 | 0 | COP disabled |
| 0 | 0 | 1 | $2^{7}$ |
| 0 | 1 | 0 | $2^{9}$ |
| 0 | 1 | 1 | $2^{11}$ |
| 1 | 0 | 0 | $2^{13}$ |
| 1 | 0 | 1 | $2^{15}$ |
| 1 | 1 | 0 | $2^{16}$ |
| 1 | 1 | 1 | $2^{17}$ |

### 4.3.2.10 Reserved Register CPMUTEST0

**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 Mode can alter the S12CPMU_UHV's functionality.

0x003D

| | 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 (CPMUTEST0)**

Read: Anytime

Write: Only in Special Mode

### 4.3.2.11 Reserved Register CPMUTEST1

**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 Mode can alter the S12CPMU_UHV's functionality.

0x003E

| | 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 (CPMUTEST1)**

Read: Anytime

Write: Only in Special Mode

### 4.3.2.12 S12CPMU_UHV COP Timer Arm/Reset Register (CPMUARMCOP)

This register is used to restart the COP time-out period.

0x003F

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | ARMCOP-Bit 7 | ARMCOP-Bit 6 | ARMCOP-Bit 5 | ARMCOP-Bit 4 | ARMCOP-Bit 3 | ARMCOP-Bit 2 | ARMCOP-Bit 1 | ARMCOP-Bit 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 4-15. S12CPMU_UHV CPMUARMCOP Register**

Read: Always reads $00

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 $55 or $AA causes a COP reset. To restart the COP time-out period write $55 followed by a write of $AA. These writes do not need to occur back-to-back, but the sequence ($55, $AA) must be completed prior to COP end of time-out period to avoid a COP reset. Sequences of $55 writes are allowed. When the WCOP bit is set, $55 and $AA 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.

### 4.3.2.13 High Temperature Control Register (CPMUHTCTL)

The CPMUHTCTL register configures the temperature sense features.

0x02F0

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | VSEL | 0 | HTE | HTDS | HTIE | HTIF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 4-16. High Temperature Control Register (CPMUHTCTL)**

Read: Anytime

Write: VSEL, HTE, HTIE and HTIF are write anytime, HTDS is read only

**Figure 4-17. Voltage Access Select**

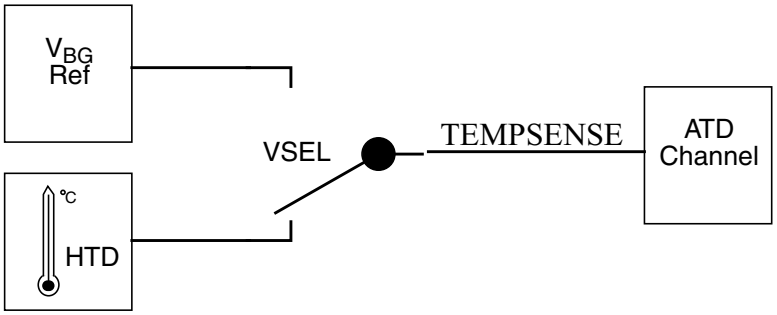



**Table 4-15. CPMUHTCTL Field Descriptions**

| Field | Description |
|---|---|
| 5<br>VSEL | **Voltage Access Select Bit** — If set, the bandgap reference voltage $V_{BG}$ can be accessed internally (i.e. multiplexed to an internal Analog to Digital Converter channel). If not set, the die temperature proportional voltage $V_{HT}$ of the temperature sensor can be accessed internally.See device level specification for connectivity. For any of these access the HTE bit must be set.<br>0 An internal temperature proportional voltage $V_{HT}$ can be accessed internally.<br>1 Bandgap reference voltage $V_{BG}$ can be accessed internally. |
| 3<br>HTE | **High Temperature Sensor/Bandgap Voltage Enable Bit** — This bit enables the high temperature sensor and bandgap voltage amplifier.<br>0 The temperature sensor and bandgap voltage amplifier is disabled.<br>1 The temperature sensor and bandgap voltage amplifier is enabled. |
| 2<br>HTDS | **High Temperature Detect Status Bit** — This read-only status bit reflects the temperature status.Writes have no effect.<br>0 Junction Temperature is below level $T_{HTID}$ or RPM.<br>1 Junction Temperature is above level $T_{HTIA}$ and FPM. |
| 1<br>HTIE | **High Temperature Interrupt Enable Bit**<br>0 Interrupt request is disabled.<br>1 Interrupt will be requested whenever HTIF is set. |
| 0<br>HTIF | **High Temperature Interrupt Flag** — HTIF — High Temperature Interrupt Flag<br>HTIF is set to 1 when HTDS status bit changes.This flag can only be cleared by writing a 1.<br>Writing a 0 has no effect. If enabled (HTIE=1), HTIF causes an interrupt request.<br>0 No change in HTDS bit.<br>1 HTDS bit has changed. |

### 4.3.2.14 Low Voltage Control Register (CPMULVCTL)

The CPMULVCTL register allows the configuration of the low-voltage detect features.

0x02F1

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | LVDS | LVIE | LVIF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | U | 0 | U |

The Reset state of LVDS and LVIF depends on the external supplied VDDA level

= Unimplemented or Reserved

**Figure 4-18. Low Voltage Control Register (CPMULVCTL)**

Read: Anytime

Write: LVIE and LVIF are write anytime, LVDS is read only

**Table 4-16. CPMULVCTL Field Descriptions**

| Field | Description |
|---|---|
| 2<br>LVDS | **Low-Voltage Detect Status Bit** — This read-only status bit reflects the voltage level on VDDA.Writes have no effect.<br>0 Input voltage VDDA is above level $V_{LVID}$ or RPM.<br>1 Input voltage VDDA is below level $V_{LVIA}$ and FPM. |
| 1<br>LVIE | **Low-Voltage Interrupt Enable Bit**<br>0 Interrupt request is disabled.<br>1 Interrupt will be requested whenever LVIF is set. |
| 0<br>LVIF | **Low-Voltage Interrupt Flag** — LVIF is set to 1 when LVDS status bit changes. This flag can only be cleared by writing a 1.Writing a 0 has no effect. If enabled (LVIE = 1), LVIF causes an interrupt request.<br>0 No change in LVDS bit.<br>1 LVDS bit has changed. |

### 4.3.2.15 Autonomous Periodical Interrupt Control Register (CPMUAPICTL)

The CPMUAPICTL register allows the configuration of the autonomous periodical interrupt features.

0x02F2

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | APICLK | 0 | 0 | APIES | APIEA | APIFE | APIE | APIF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 4-19. Autonomous Periodical Interrupt Control Register (CPMUAPICTL)**

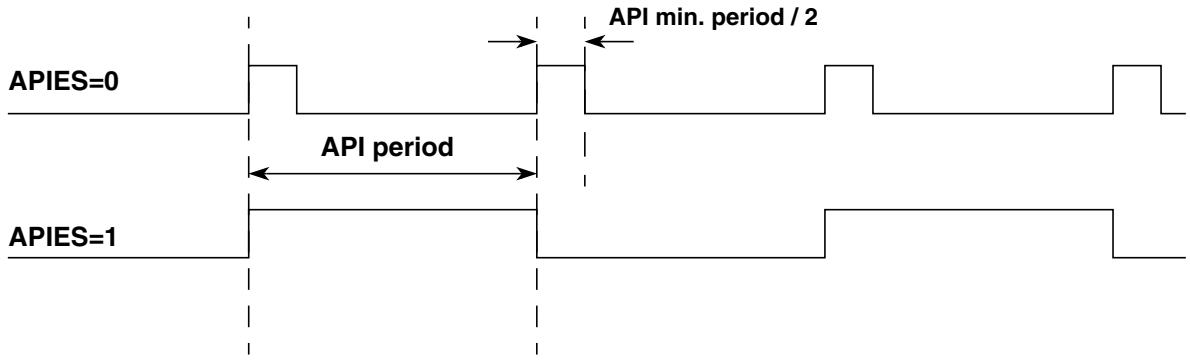
Read: Anytime

Write: Anytime

**Table 4-17. CPMUAPICTL Field Descriptions**

| Field | Description |
|---|---|
| 7<br>APICLK | **Autonomous Periodical Interrupt Clock Select Bit** — Selects the clock source for the API. Writable only if APIFE = 0. APICLK cannot be changed if APIFE is set by the same write operation.<br>0 Autonomous Clock (ACLK) used as source.<br>1 Bus Clock used as source. |
| 4<br>APIES | **Autonomous Periodical Interrupt External Select Bit** — Selects the waveform at the external pin API_EXTCLK as shown in Figure 4-20. See device level specification for connectivity of API_EXTCLK pin.<br>0 If APIEA and APIFE are set, at the external pin API_EXTCLK periodic high pulses are visible at the end of every selected period with the size of half of the minimum period (APIR=0x0000 in Table 4-21).<br>1 If APIEA and APIFE are set, at the external pin API_EXTCLK a clock is visible with 2 times the selected API Period. |
| 3<br>APIEA | **Autonomous Periodical Interrupt External Access Enable Bit** — If set, the waveform selected by bit APIES can be accessed externally. See device level specification for connectivity.<br>0 Waveform selected by APIES can not be accessed externally.<br>1 Waveform selected by APIES can be accessed externally, if APIFE is set. |
| 2<br>APIFE | **Autonomous Periodical Interrupt Feature Enable Bit** — Enables the API feature and starts the API timer when set.<br>0 Autonomous periodical interrupt is disabled.<br>1 Autonomous periodical interrupt is enabled and timer starts running. |
| 1<br>APIE | **Autonomous Periodical Interrupt Enable Bit**<br>0 API interrupt request is disabled.<br>1 API interrupt will be requested whenever APIF is set. |
| 0<br>APIF | **Autonomous Periodical Interrupt Flag** — APIF is set to 1 when the in the API configured time has elapsed. This flag can only be cleared by writing a 1.Writing a 0 has no effect. If enabled (APIE = 1), APIF causes an interrupt request.<br>0 API time-out has not yet occurred.<br>1 API time-out has occurred. |

**Figure 4-20. Waveform selected on API_EXTCLK pin (APIEA=1, APIFE=1)**



### 4.3.2.16 Autonomous Clock Trimming Register (CPMUACLKTR)

The CPMUACLKTR register configures the trimming of the Autonomous Clock (ACLK - trimmable internal RC-Oscillator) which can be selected as clock source for some CPMU features.

0x02F3

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | ACLKTR5 | ACLKTR4 | ACLKTR3 | ACLKTR2 | ACLKTR1 | ACLKTR0 | 0 | 0 |
| W | | | | | | | | |
| Reset | F | F | F | F | F | F | 0 | 0 |

After de-assert of System Reset a value is automatically loaded from the Flash memory.

**Figure 4-21. Autonomous Clock Trimming Register (CPMUACLKTR)**

Read: Anytime

Write: Anytime

**Table 4-18. CPMUACLKTR Field Descriptions**

| Field | Description |
|---|---|
| 7–2<br>ACLKTR[5:0] | **Autonomous Clock Period Trimming Bits** — See Table 4-19 for trimming effects. The ACLKTR[5:0] value represents a signed number influencing the ACLK period time. |

**Table 4-19. Trimming Effect of ACLKTR**

| Bit | Trimming Effect |
|---|---|
| ACLKTR[5] | Increases period |
| ACLKTR[4] | Decreases period less than ACLKTR[5] increased it |
| ACLKTR[3] | Decreases period less than ACLKTR[4] |
| ACLKTR[2] | Decreases period less than ACLKTR[3] |
| ACLKTR[1] | Decreases period less than ACLKTR[2] |
| ACLKTR[0] | Decreases period less than ACLKTR[1] |

### 4.3.2.17 Autonomous Periodical Interrupt Rate High and Low Register (CPMUAPIRH / CPMUAPIRL)

The CPMUAPIRH and CPMUAPIRL registers allow the configuration of the autonomous periodical interrupt rate.

0x02F4

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | APIR15 | APIR14 | APIR13 | APIR12 | APIR11 | APIR10 | APIR9 | APIR8 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 4-22. Autonomous Periodical Interrupt Rate High Register (CPMUAPIRH)**

0x02F5

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | APIR7 | APIR6 | APIR5 | APIR4 | APIR3 | APIR2 | APIR1 | APIR0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 4-23. Autonomous Periodical Interrupt Rate Low Register (CPMUAPIRL)**

Read: Anytime

Write: Anytime if APIFE=0, Else writes have no effect.

**Table 4-20. CPMUAPIRH / CPMUAPIRL Field Descriptions**

| Field | Description |
|---|---|
| 15-0<br>APIR[15:0] | **Autonomous Periodical Interrupt Rate Bits** — These bits define the time-out period of the API. See Table 4-21 for details of the effect of the autonomous periodical interrupt rate bits. |

The period can be calculated as follows depending on logical value of the APICLK bit:

APICLK=0: Period = 2*(APIR[15:0] + 1) * (ACLK Clock Period * 2)
APICLK=1: Period = 2*(APIR[15:0] + 1) * Bus Clock Period

**NOTE**

For APICLK bit clear the first time-out period of the API will show a latency time between two to three $f_{ACLK}$ cycles due to synchronous clock gate release when the API feature gets enabled (APIFE bit set).

**Table 4-21. Selectable Autonomous Periodical Interrupt Periods**

| APICLK | APIR[15:0] | Selected Period |
|---|---|---|
| 0 | 0000 | 0.2 ms[1] |
| 0 | 0001 | 0.4 ms[1] |
| 0 | 0002 | 0.6 ms[1] |
| 0 | 0003 | 0.8 ms[1] |
| 0 | 0004 | 1.0 ms[1] |
| 0 | 0005 | 1.2 ms[1] |
| 0 | ..... | ..... |
| 0 | FFFD | 13106.8 ms[1] |
| 0 | FFFE | 13107.0 ms[1] |
| 0 | FFFF | 13107.2 ms[1] |
| 1 | 0000 | 2 * Bus Clock period |
| 1 | 0001 | 4 * Bus Clock period |
| 1 | 0002 | 6 * Bus Clock period |
| 1 | 0003 | 8 * Bus Clock period |
| 1 | 0004 | 10 * Bus Clock period |
| 1 | 0005 | 12 * Bus Clock period |
| 1 | ..... | ..... |
| 1 | FFFD | 131068 * Bus Clock period |
| 1 | FFFE | 131070 * Bus Clock period |
| 1 | FFFF | 131072 * Bus Clock period |

[1] When $f_{ACLK}$ is trimmed to 20kHz.

### 4.3.2.18 Reserved Register CPMUTEST3

**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 Mode can alter the S12CPMU_UHV's functionality.

0x02F6

| | 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 4-24. Reserved Register (CPMUTEST3)**

Read: Anytime

Write: Only in Special Mode

### 4.3.2.19 High Temperature Trimming Register (CPMUHTTR)

The CPMUHTTR register configures the trimming of the S12CPMU_UHV temperature sense.

0x02F7

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | HTOE | 0 | 0 | 0 | HTTR3 | HTTR2 | HTTR1 | HTTR0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | F | F | F | F |

After de-assert of System Reset a trim value is automatically loaded from the Flash memory. See Device specification for details.

= Unimplemented or Reserved

**Figure 4-25. High Temperature Trimming Register (CPMUHTTR)**

Read: Anytime

Write: Anytime

**Table 4-23. CPMUHTTR Field Descriptions**

| Field | Description |
|---|---|
| 7<br>HTOE | **High Temperature Offset Enable Bit** — If set the temperature sense offset is enabled.<br>0 The temperature sense offset is disabled. HTTR[3:0] bits don't care.<br>1 The temperature sense offset is enabled. HTTR[3:0] select the temperature offset. |
| 3–0<br>HTTR[3:0] | **High Temperature Trimming Bits** — See Table 4-24 for trimming effects. |

**Table 4-24. Trimming Effect of HTTR**

| Bit | Trimming Effect |
|---|---|
| HTTR[3] | Increases $V_{HT}$ twice of HTTR[2] |
| HTTR[2] | Increases $V_{HT}$ twice of HTTR[1] |
| HTTR[1] | Increases $V_{HT}$ twice of HTTR[0] |
| HTTR[0] | Increases $V_{HT}$ (to compensate Temperature Offset) |

### 4.3.2.20 S12CPMU_UHV IRC1M Trim Registers (CPMUIRCTRIMH / CPMUIRCTRIML)

0x02F8

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| R | TCTRIM[4:0] | | | | | 0 | IRCTRIM[9:8] | |
| W | | | | | | | | |
| Reset | F | F | F | F | F | 0 | F | F |

After de-assert of System Reset a factory programmed trim value is automatically loaded from the Flash memory to provide trimmed Internal Reference Frequency $f_{IRC1M_TRIM}$.

**Figure 4-26. S12CPMU_UHV IRC1M Trim High Register (CPMUIRCTRIMH)**

0x02F9

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | IRCTRIM[7:0] | | | | | | | |
| W | | | | | | | | |
| Reset | F | F | F | F | F | F | F | F |

After de-assert of System Reset a factory programmed trim value is automatically loaded from the Flash memory to provide trimmed Internal Reference Frequency $f_{IRC1M_TRIM}$.

**Figure 4-27. S12CPMU_UHV IRC1M Trim Low Register (CPMUIRCTRIML)**

Read: Anytime

Write: Anytime if PROT=0 (CPMUPROT register). Else write has no effect

**NOTE**

Writes to these registers while PLLSEL=1 clears the LOCK and UPOSC status bits.

**Table 4-25. CPMUIRCTRIMH/L Field Descriptions**

| Field | Description |
|---|---|
| 15-11<br>TCTRIM[4:0] | **IRC1M temperature coefficient Trim Bits**<br>Trim bits for the Temperature Coefficient (TC) of the IRC1M frequency.<br>Table 4-26 shows the influence of the bits TCTRIM[4:0] on the relationship between frequency and temperature.<br>Figure 4-29 shows an approximate TC variation, relative to the nominal TC of the IRC1M (i.e. for TCTRIM[4:0]=0x00000 or 0x10000). |
| 9-0<br>IRCTRIM[9:0] | **IRC1M Frequency Trim Bits** — Trim bits for Internal Reference Clock<br>After System Reset the factory programmed trim value is automatically loaded into these registers, resulting in a Internal Reference Frequency $f_{IRC1M_TRIM}$.See device electrical characteristics for value of $f_{IRC1M_TRIM}$.<br>The frequency trimming consists of two different trimming methods:<br>A rough trimming controlled by bits IRCTRIM[9:6] can be done with frequency leaps of about 6% in average.<br>A fine trimming controlled by bits IRCTRIM[5:0] can be done with frequency leaps of about 0.3% (this trimming determines the precision of the frequency setting of 0.15%, i.e. 0.3% is the distance between two trimming values).<br>Figure 4-28 shows the relationship between the trim bits and the resulting IRC1M frequency. |

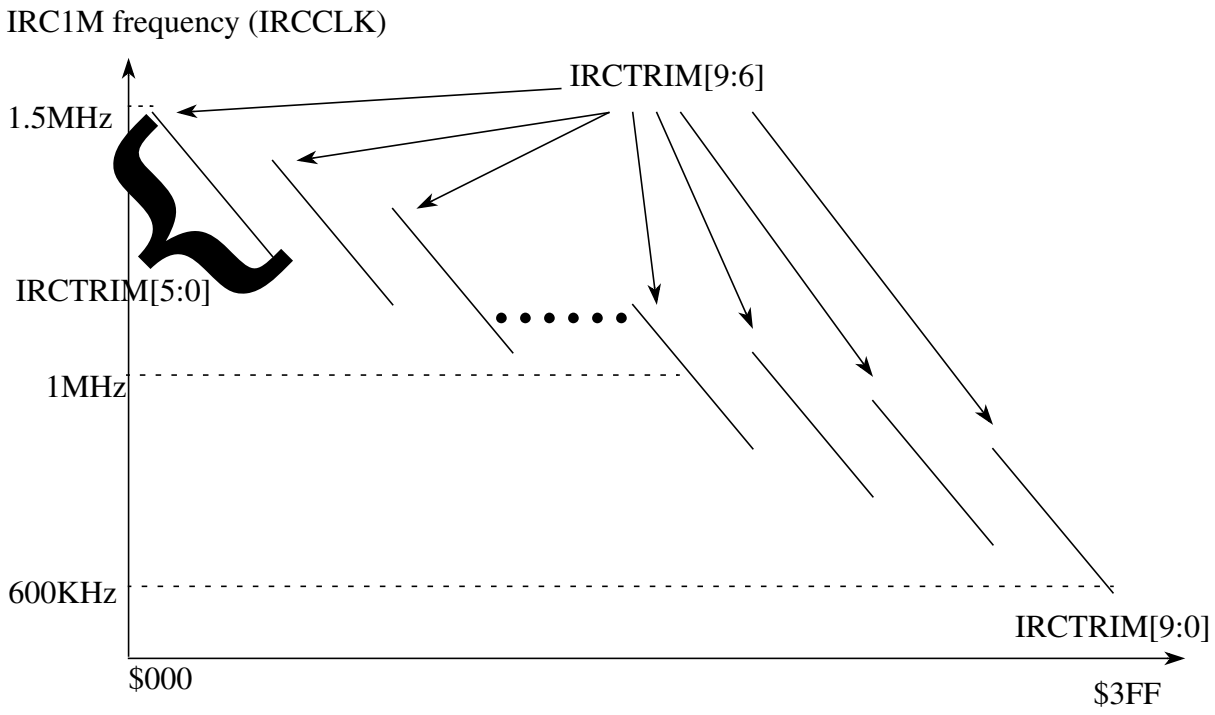



**Figure 4-28. IRC1M Frequency Trimming Diagram**

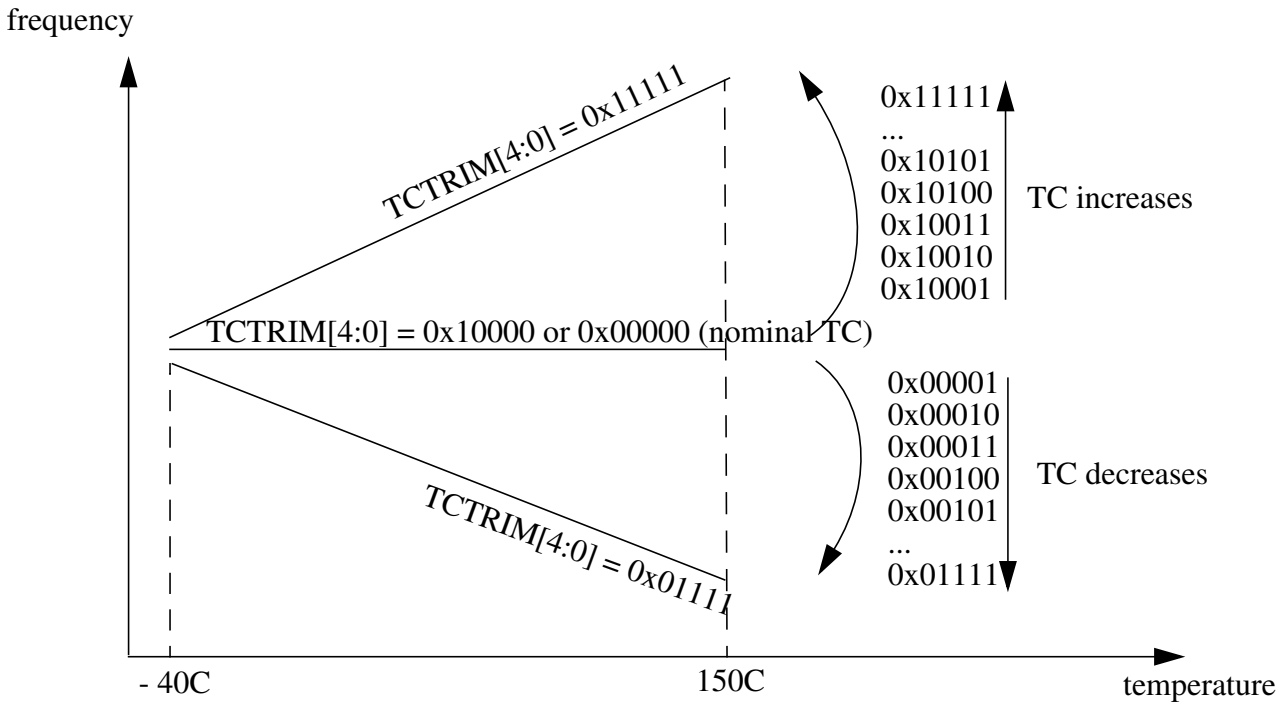



**Figure 4-29. Influence of TCTRIM[4:0] on the Temperature Coefficient**

**NOTE**

The frequency is not necessarily linear with the temperature (in most cases it will not be). The above diagram is meant only to give the direction (positive or negative) of the variation of the TC, relative to the nominal TC.

Setting TCTRIM[4:0] at 0x00000 or 0x10000 does not mean that the temperature coefficient will be zero. These two combinations basically switch off the TC compensation module, which results in the nominal TC of the IRC1M.

**Table 4-26. TC trimming of the frequency of the IRC1M at ambient temperature**

| TCTRIM[4:0] | IRC1M Indicative relative TC variation | IRC1M indicative frequency drift for relative TC variation |
|---|---|---|
| 00000 | 0 (nominal TC of the IRC) | 0% |
| 00001 | -0.27% | -0.5% |
| 00010 | -0.54% | -0.9% |
| 00011 | -0.81% | -1.3% |
| 00100 | -1.08% | -1.7% |
| 00101 | -1.35% | -2.0% |
| 00110 | -1.63% | -2.2% |
| 00111 | -1.9% | -2.5% |
| 01000 | -2.20% | -3.0% |
| 01001 | -2.47% | -3.4% |
| 01010 | -2.77% | -3.9% |
| 01011 | -3.04 | -4.3% |
| 01100 | -3.33% | -4.7% |
| 01101 | -3.6% | -5.1% |
| 01110 | -3.91% | -5.6% |
| 01111 | -4.18% | -5.9% |
| 10000 | 0 (nominal TC of the IRC) | 0% |
| 10001 | +0.27% | +0.5% |
| 10010 | +0.54% | +0.9% |
| 10011 | +0.81% | +1.3% |
| 10100 | +1.07% | +1.7% |
| 10101 | +1.34% | +2.0% |
| 10110 | +1.59% | +2.2% |
| 10111 | +1.86% | +2.5% |
| 11000 | +2.11% | +3.0% |
| 11001 | +2.38% | +3.4% |
| 11010 | +2.62% | +3.9% |
| 11011 | +2.89% | +4.3% |
| 11100 | +3.12% | +4.7% |
| 11101 | +3.39% | +5.1% |
| 11110 | +3.62% | +5.6% |
| 11111 | +3.89% | +5.9% |

**NOTE**

Since the IRC1M frequency is not a linear function of the temperature, but more like a parabola, the above relative variation is only an indication and should be considered with care.

Be aware that the output frequency varies with the TC trimming. A frequency trimming correction is therefore necessary. The values provided in Table 4-26 are typical values at ambient temperature which can vary from device to device.

### 4.3.2.21 S12CPMU_UHV Oscillator Register (CPMUOSC)

This register configures the external oscillator (XOSCLCP).

0x02FA

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | OSCE | Reserved | 0 | Reserved | | | | |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 4-30. S12CPMU_UHV Oscillator Register (CPMUOSC)**

Read: Anytime

Write: Anytime if PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register). Else write has no effect.

NOTE.

Write to this register clears the LOCK and UPOSC status bits.

**Table 4-27. CPMUOSC Field Descriptions**

| Field | Description |
|---|---|
| 7<br>OSCE | **Oscillator Enable Bit** — This bit enables the external oscillator (XOSCLCP). The UPOSC status bit in the CPMUFLG register indicates when the oscillation is stable and OSCCLK can be selected as source of the Bus Clock or source of the COP or RTI.A loss of oscillation will lead to a clock monitor reset.This<br>0 External oscillator is disabled.<br>REFCLK for PLL is IRCCLK.<br>1 External oscillator is enabled. Clock monitor is enabled. External oscillator is qualified by PLLCLK<br>REFCLK for PLL is the external oscillator clock divided by REFDIV.<br>**Note:** When starting up the external oscillator (either by programming OSCE bit to 1 or on exit from Full Stop Mode with OSCE bit already 1) the software must wait for a minimum time equivalent to the startup-time of the external oscillator $t_{UPOSC}$ before entering Pseudo Stop Mode. |
| 6<br>Reserved | Do not alter this bit from its reset value.It is for Manufacturer use only and can change the PLL behavior. |
| 4-0<br>Reserved | Do not alter these bits from their reset value. These are for Manufacturer use only and can change the PLL behavior. |

### 4.3.2.22 S12CPMU_UHV Protection Register (CPMUPROT)

This register protects the clock configuration registers from accidental overwrite:

CPMUSYNR, CPMUREFDIV, CPMUCLKS, CPMUPLL, CPMUIRCTRIMH/L and CPMUOSC

0x02FB

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PROT |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 4-31. S12CPMU_UHV Protection Register (CPMUPROT)**

Read: Anytime

Write: Anytime

| Field | Description |
|---|---|
| PROT | **Clock Configuration Registers Protection Bit** — This bit protects the clock configuration registers from accidental overwrite (see list of protected registers above): Writing 0x26 to the CPMUPROT register clears the PROT bit, other write accesses set the PROT bit.<br>0 Protection of clock configuration registers is disabled.<br>1 Protection of clock configuration registers is enabled. (see list of protected registers above). |

### 4.3.2.23 Reserved Register CPMUTEST2

**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 Mode can alter the S12CPMU_UHV's functionality.

0x02FC

| | 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 4-32. Reserved Register CPMUTEST2**

Read: Anytime

Write: Only in Special Mode

# 4.4 Functional Description

## 4.4.1 Phase Locked Loop with Internal Filter (PLL)

The PLL is used to generate a high speed PLLCLK based on a low frequency REFCLK.

The REFCLK is by default the IRCCLK which is trimmed to $f_{IRC1M_TRIM}$=1MHz.

If using the oscillator (OSCE=1) REFCLK will be based on OSCCLK. For increased flexibility, OSCCLK can be divided in a range of 1 to 16 to generate the reference frequency REFCLK using the REFDIV[3:0] bits. Based on the SYNDIV[5:0] bits the PLL generates the VCOCLK by multiplying the reference clock by a 2, 4, 6,... 126, 128. Based on the POSTDIV[4:0] bits the VCOCLK can be divided in a range of 1,2, 3, 4, 5, 6,... to 32 to generate the PLLCLK.

If oscillator is enabled (OSCE=1) $$f_{REF} = \frac{f_{OSC}}{(REFDIV + 1)}$$

If oscillator is disabled (OSCE=0) $$f_{REF} = f_{IRC1M}$$

$$f_{VCO} = 2 \times f_{REF} \times (SYNDIV + 1)$$

If PLL is locked (LOCK=1) $$f_{PLL} = \frac{f_{VCO}}{(POSTDIV + 1)}$$

If PLL is not locked (LOCK=0) $$f_{PLL} = \frac{f_{VCO}}{4}$$

If PLL is selected (PLLSEL=1) $$f_{bus} = \frac{f_{PLL}}{2}$$

**NOTE**

Although it is possible to set the dividers to command a very high clock frequency, do not exceed the specified bus frequency limit for the MCU.

Several examples of PLL divider settings are shown in Table 4-28. The following rules help to achieve optimum stability and shortest lock time:

- Use lowest possible $f_{VCO}$ / $f_{REF}$ ratio (SYNDIV value).
- Use highest possible REFCLK frequency $f_{REF}$.

**Table 4-28. Examples of PLL Divider Settings**

| $f_{osc}$ | REFDIV[3:0] | $f_{REF}$ | REFFRQ[1:0] | SYNDIV[5:0] | $f_{VCO}$ | VCOFRQ[1:0] | POSTDIV[4:0] | $f_{PLL}$ | $f_{bus}$ |
|---|---|---|---|---|---|---|---|---|---|
| off | $00 | 1MHz | 00 | $18 | 50MHz | 01 | $03 | 12.5MHz | 6.25MHz |
| off | $00 | 1MHz | 00 | $18 | 50MHz | 01 | $00 | 50MHz | 25MHz |
| 4MHz | $00 | 4MHz | 01 | $05 | 48MHz | 00 | $00 | 48MHz | 24MHz |

The phase detector inside the PLL compares the feedback clock (FBCLK = VCOCLK/(SYNDIV+1)) with the reference clock (REFCLK = (IRC1M or OSCCLK)/(REFDIV+1)). Correction pulses are generated based on the phase difference between the two signals. The loop filter alters the DC voltage on the internal filter capacitor, based on the width and direction of the correction pulse which leads to a higher or lower VCO frequency.

The user must select the range of the REFCLK frequency (REFFRQ[1:0] bits) and the range of the VCOCLK frequency (VCOFRQ[1:0] bits) to ensure that the correct PLL loop bandwidth is set.

The lock detector compares the frequencies of the FBCLK and the REFCLK. Therefore the speed of the lock detector is directly proportional to the reference clock frequency. The circuit determines the lock condition based on this comparison.

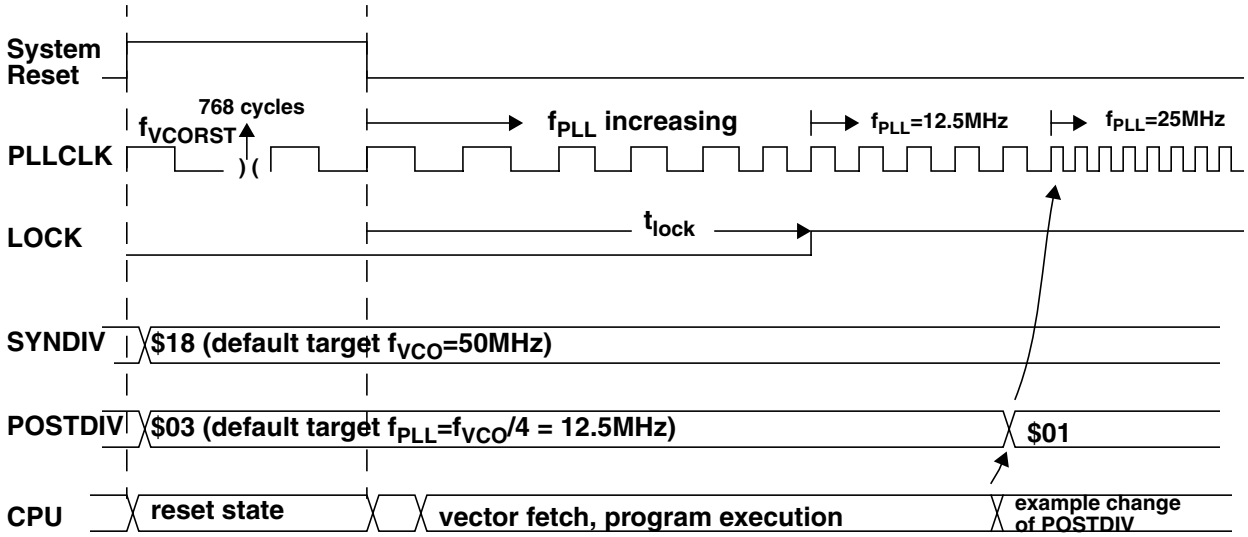
If PLL LOCK interrupt requests are enabled, the software can wait for an interrupt request and for instance check the LOCK bit. If interrupt requests are disabled, software can poll the LOCK bit continuously (during PLL start-up) or at periodic intervals. In either case, only when the LOCK bit is set, the VCOCLK will have stabilized to the programmed frequency.

- 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 the tolerance, $\Delta_{Lock}$, and is cleared when the VCO frequency is out of the tolerance, $\Delta_{unl}$.
- Interrupt requests can occur if enabled (LOCKIE = 1) when the lock condition changes, toggling the LOCK bit.

### 4.4.2 Startup from Reset

An example for startup of the clock system from Reset is given in Figure 4-33.

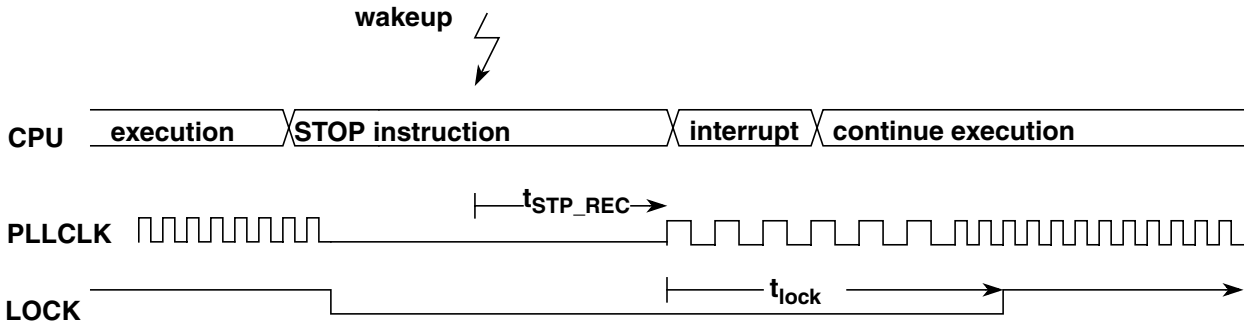
**Figure 4-33. Startup of clock system after Reset**




### 4.4.3 Stop Mode using PLLCLK as source of the Bus Clock

An example of what happens going into Stop Mode and exiting Stop Mode after an interrupt is shown in Figure 4-34. Disable PLL Lock interrupt (LOCKIE=0) before going into Stop Mode.

**Figure 4-34. Stop Mode using PLLCLK as source of the Bus Clock**




Depending on the COP configuration there might be an additional significant latency time until COP is active again after exit from Stop Mode due to clock domain crossing synchronization. This latency time of 2 ACLK cycles occurs if COP clock source is ACLK and the CSAD bit is set and must be added to the device Stop Mode recovery time $t_{STP_REC}$. After exit from Stop Mode (Pseudo, Full) for this latency time

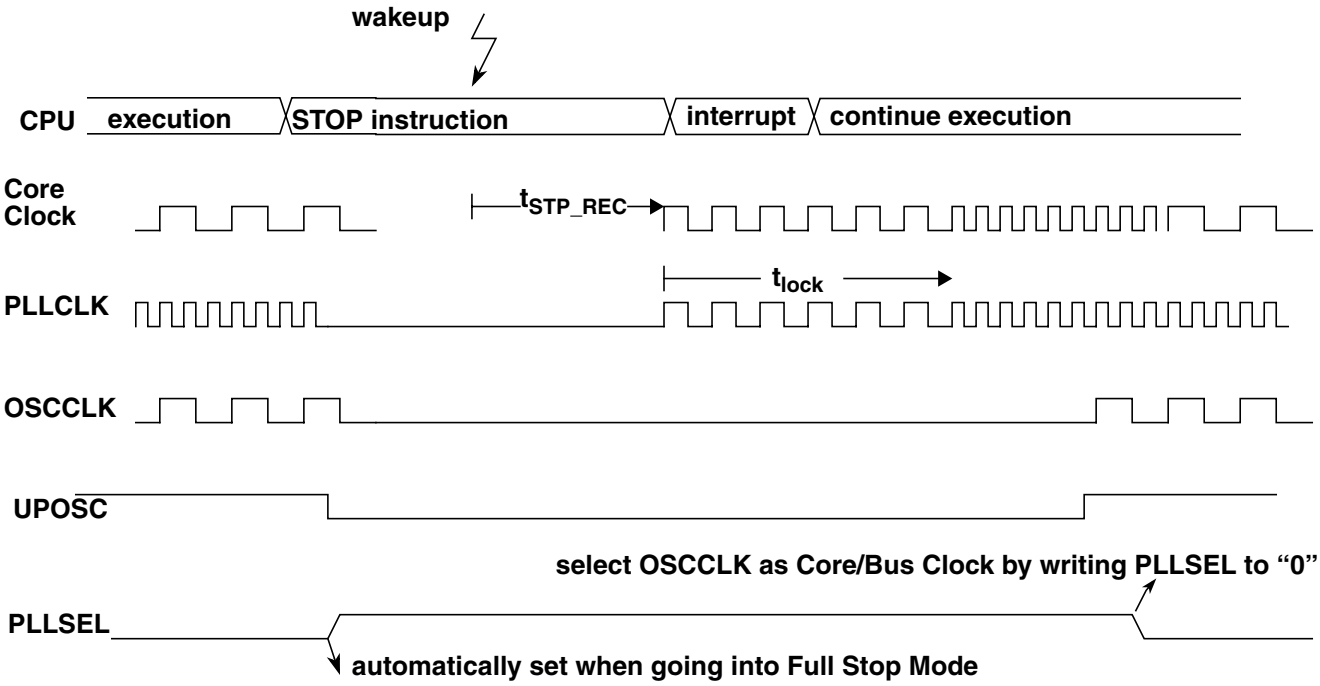
of 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter can increment at each Stop Mode exit.

### 4.4.4 Full Stop Mode using Oscillator Clock as source of the Bus Clock

An example of what happens going into Full Stop Mode and exiting Full Stop Mode after an interrupt is shown in Figure 4-35.

Disable PLL Lock interrupt (LOCKIE=0) and oscillator status change interrupt (OSCIE=0) before going into Full Stop Mode.

**Figure 4-35. Full Stop Mode using Oscillator Clock as source of the Bus Clock**



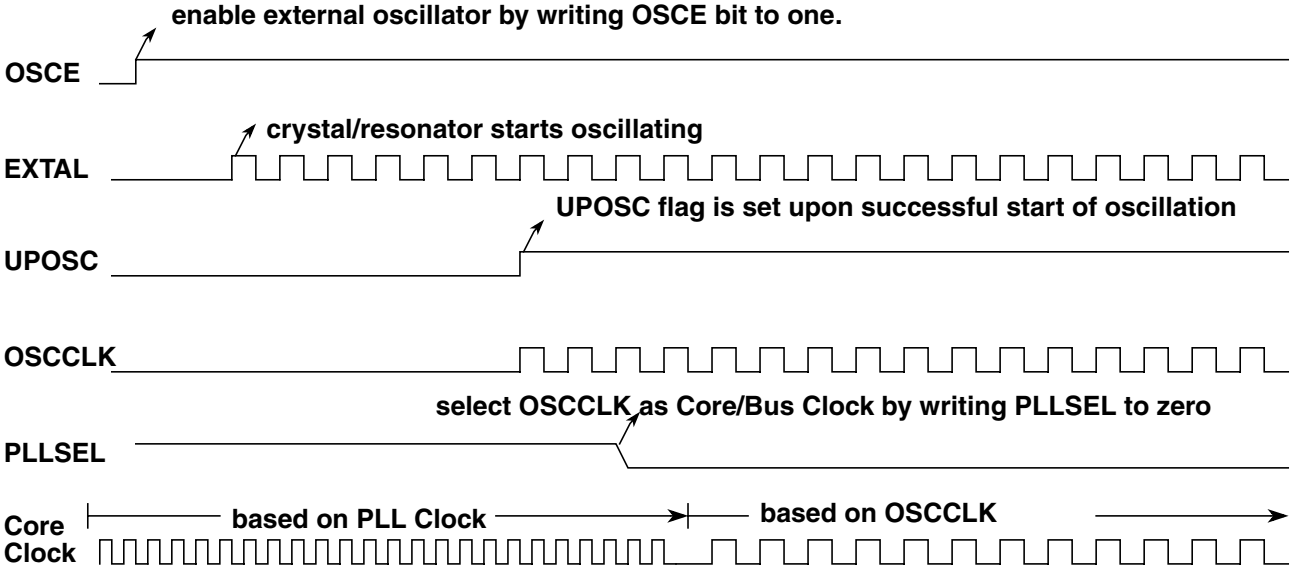

Depending on the COP configuration there might be a significant latency time until COP is active again after exit from Stop Mode due to clock domain crossing synchronization. This latency time of 2 ACLK cycles occurs if COP clock source is ACLK and the CSAD bit is set and must be added to the device Stop Mode recovery time $t_{STP_REC}$. After exit from Stop Mode (Pseudo, Full) for this latency time of 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter can increment at each Stop Mode exit.

## 4.4.5 External Oscillator

### 4.4.5.1 Enabling the External Oscillator

An example of how to use the oscillator as source of the Bus Clock is shown in Figure 4-36.

**Figure 4-36. Enabling the external oscillator**



## 4.4.6 System Clock Configurations

### 4.4.6.1 PLL Engaged Internal Mode (PEI)

This mode is the default mode after System Reset or Power-On Reset.

The Bus Clock is based on the PLLCLK, the reference clock for the PLL is internally generated (IRC1M). The PLL is configured to 50 MHz VCOCLK with POSTDIV set to 0x03. If locked (LOCK=1) this results in a PLLCLK of 12.5 MHz and a Bus Clock of 6.25 MHz. The PLL can be re-configured to other bus frequencies.

The clock sources for COP and RTI can be based on the internal reference clock generator (IRC1M) or the RC-Oscillator (ACLK).

### 4.4.6.2 PLL Engaged External Mode (PEE)

In this mode, the Bus Clock is based on the PLLCLK as well (like PEI). The reference clock for the PLL is based on the external oscillator.

The clock sources for COP and RTI can be based on the internal reference clock generator or on the external oscillator clock or the RC-Oscillator (ACLK).

This mode can be entered from default mode PEI by performing the following steps:

1. Configure the PLL for desired bus frequency.
2. Enable the external Oscillator (OSCE bit).
3. Wait for oscillator to start-up and the PLL being locked (LOCK = 1) and (UPOSC =1).
4. Clear all flags in the CPMUFLG register to be able to detect any future status bit change.
5. Optionally status interrupts can be enabled (CPMUINT register).

Loosing PLL lock status (LOCK=0) means loosing the oscillator status information as well (UPOSC=0).

The impact of loosing the oscillator status (UPOSC=0) in PEE mode is as follows:

- The PLLCLK is derived from the VCO clock (with its actual frequency) divided by four until the PLL locks again.

Application software needs to be prepared to deal with the impact of loosing the oscillator status at any time.

#### 4.4.6.3 PLL Bypassed External Mode (PBE)

In this mode, the Bus Clock frequency is half the external oscillator clock. The reference clock for the PLL is based on the external oscillator.

The clock sources for COP and RTI can be based on the internal reference clock generator or on the external oscillator clock or the RC-Oscillator (ACLK).

This mode can be entered from default mode PEI by performing the following steps:

1. Make sure the PLL configuration is valid.
2. Enable the external Oscillator (OSCE bit)
3. Wait for the oscillator to start-up and the PLL being locked (LOCK = 1) and (UPOSC =1)
4. Clear all flags in the CPMUFLG register to be able to detect any status bit change.
5. Optionally status interrupts can be enabled (CPMUINT register).
6. Select the Oscillator clock as source of the Bus Clock (PLLSEL=0)

Loosing PLL lock status (LOCK=0) means loosing the oscillator status information as well (UPOSC=0).

The impact of loosing the oscillator status (UPOSC=0) in PBE mode is as follows:

- PLLSEL is set automatically and the Bus clock is switched back to the PLL clock.
- The PLLCLK is derived from the VCO clock (with its actual frequency) divided by four until the PLL locks again.

Application software needs to be prepared to deal with the impact of loosing the oscillator status at any time.

## 4.5 Resets

### 4.5.1 General

All reset sources are listed in Table 4-29. Refer to MCU specification for related vector addresses and priorities.

**Table 4-29. Reset Summary**

| Reset Source | Local Enable |
|---|---|
| Power-On Reset (POR) | None |
| Low Voltage Reset (LVR) | None |
| External pin $\overline{\text{RESET}}$ | None |
| Illegal Address Reset | None |
| Clock Monitor Reset | OSCE Bit in CPMUOSC register |
| COP Reset | CR[2:0] in CPMUCOP register |

### 4.5.2 Description of Reset Operation

Upon detection of any reset of Table 4-29, an internal circuit drives the $\overline{\text{RESET}}$ pin low for 512 PLLCLK cycles. After 512 PLLCLK cycles the $\overline{\text{RESET}}$ pin is released. The reset generator of the S12CPMU_UHV

waits for additional 256PLLCLK cycles and then samples the $\overline{\text{RESET}}$ pin to determine the originating source. Table 4-30 shows which vector will be fetched.

**Table 4-30. Reset Vector Selection**

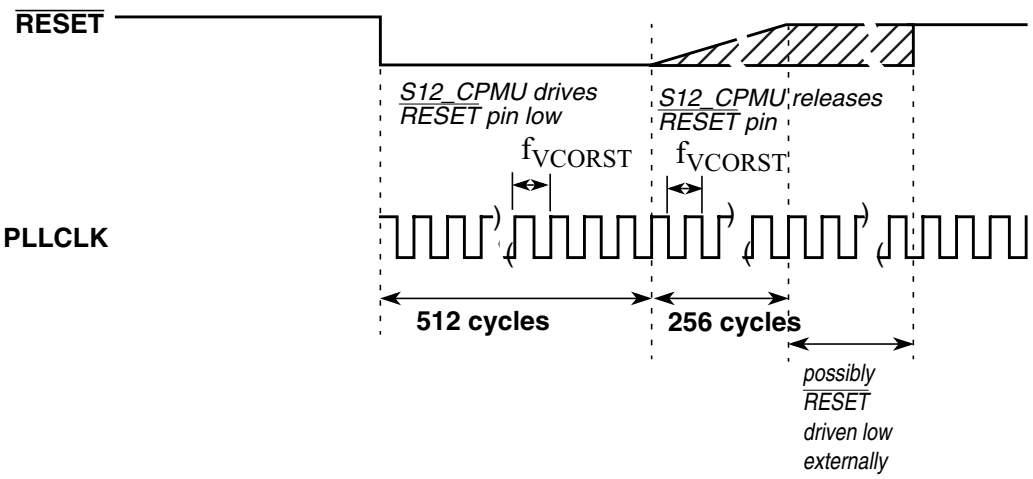
| Sampled $\overline{\text{RESET}}$ Pin (256 cycles after release) | Oscillator monitor fail pending | COP time-out pending | Vector Fetch |
|---|---|---|---|
| 1 | 0 | 0 | POR<br>LVR<br>Illegal Address Reset<br>External pin $\overline{\text{RESET}}$ |
| 1 | 1 | X | Clock Monitor Reset |
| 1 | 0 | 1 | COP Reset |
| 0 | X | X | POR<br>LVR<br>Illegal Address Reset<br>External pin $\overline{\text{RESET}}$ |

**NOTE**

While System Reset is asserted the PLLCLK runs with the frequency $f_{VCORST}$.

The internal reset of the MCU remains asserted while the reset generator completes the 768 PLLCLK cycles long reset sequence. In case the $\overline{\text{RESET}}$ pin is externally driven low for more than these 768 PLLCLK cycles (External Reset), the internal reset remains asserted longer.

**Figure 4-37. RESET Timing**




### 4.5.2.1 Clock Monitor Reset

If the external oscillator is enabled (OSCE=1) in case of loss of oscillation or the oscillator frequency is below the failure assert frequency $f_{CMFA}$ (see device electrical characteristics for values), the

S12CPMU_UHV generates a Clock Monitor Reset. In Full Stop Mode the external oscillator and the clock monitor are disabled.

#### 4.5.2.2 Computer Operating Properly Watchdog (COP) Reset

The COP (free running watchdog timer) enables the user to check that a program is running and sequencing properly. 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 COP reset is generated.

The clock source for the COP is either ACLK, IRCCLK or OSCCLK depending on the setting of the COPOSCSEL0 and COPOSCSEL1 bit.
Due to clock domain crossing synchronization there is a latency time to enter and exit Stop Mode if the COP clock source is ACLK and this clock is stopped in Stop Mode. This maximum total latency time is 4 ACLK cycles (2 ACLK cycles for Stop Mode entry and exit each) which must be added to the Stop Mode recovery time $t_{STP_REC}$ from exit of current Stop Mode to entry of next Stop Mode. This latency time occurs no matter which Stop Mode (Full, Pseudo) is currently exited or entered next.
After exit from Stop Mode (Pseudo, Full) for this latency time of 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter can increment at each Stop Mode exit.
Table 4-31 gives an overview of the COP condition (run, static) in Stop Mode depending on legal configuration and status bit settings:

**Table 4-31. COP condition (run, static) in Stop Mode**

| COPOSCSEL1 | CSAD | PSTP | PCE | COPOSCSEL0 | OSCE | UPOSC | COP counter behavior in Stop Mode (clock source) |
|---|---|---|---|---|---|---|---|
| 1 | 0 | x | x | x | x | x | Run (ACLK) |
| 1 | 1 | x | x | x | x | x | Static (ACLK) |
| 0 | x | 1 | 1 | 1 | 1 | 1 | Run (OSCCLK) |
| 0 | x | 1 | 1 | 0 | 0 | x | Static (IRCCLK) |
| 0 | x | 1 | 1 | 0 | 1 | x | Static (IRCCLK) |
| 0 | x | 1 | 0 | 0 | x | x | Static (IRCCLK) |
| 0 | x | 1 | 0 | 1 | 1 | 1 | Static (OSCCLK) |
| 0 | x | 0 | 1 | 1 | 1 | 1 | Static (OSCCLK) |
| 0 | x | 0 | 1 | 0 | 1 | x | Static (IRCCLK) |
| 0 | x | 0 | 1 | 0 | 0 | 0 | Static (IRCCLK) |
| 0 | x | 0 | 0 | 1 | 1 | 1 | Satic (OSCCLK) |
| 0 | x | 0 | 0 | 0 | 1 | 1 | Static (IRCCLK) |
| 0 | x | 0 | 0 | 0 | 1 | 0 | Static (IRCCLK) |
| 0 | x | 0 | 0 | 0 | 0 | 0 | Static (IRCCLK) |

Three control bits in the CPMUCOP register allow selection of seven COP time-out periods.

When COP is enabled, the program must write $55 and $AA (in this order) to the CPMUARMCOP register during the selected time-out period. Once this is done, the COP time-out period is restarted. If the program fails to do this and the COP times out, a COP reset is generated. Also, if any value other than $55 or $AA is written, a COP reset is generated.

Windowed COP operation is enabled by setting WCOP in the CPMUCOP register. In this mode, writes to the CPMUARMCOP 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.

In MCU Normal Mode the COP time-out period (CR[2:0]) and COP window (WCOP) setting can be automatically pre-loaded at reset release from NVM memory (if values are defined in the NVM by the application). By default the COP is off and no window COP feature is enabled after reset release via NVM memory. The COP control register CPMUCOP can be written once in an application in MCU Normal Mode to update the COP time-out period (CR[2:0]) and COP window (WCOP) setting loaded from NVM memory at reset release. Any value for the new COP time-out period and COP window setting is allowed except COP off value if the COP was enabled during pre-load via NVM memory.
The COP clock source select bits can not be pre-loaded via NVM memory at reset release. The IRC clock is the default COP clock source out of reset.
The COP clock source select bits (COPOSCSEL0/1) and ACLK clock control bit in Stop Mode (CSAD) can be modified until the CPMUCOP register write once has taken place. Therefore these control bits should be modified before the final COP time-out period and window COP setting is written.
The CPMUCOP register access to modify the COP time-out period and window COP setting in MCU Normal Mode after reset release must be done with the WRTMASK bit cleared otherwise the update is ignored and this access does not count as the write once.

### 4.5.3 Power-On Reset (POR)

The on-chip POR circuitry detects when the internal supply VDD drops below an appropriate voltage level. The POR is deasserted, if the internal supply VDD exceeds an appropriate voltage level (voltage levels not specified in this document because this internal supply is not visible on device pins).

### 4.5.4 Low-Voltage Reset (LVR)

The on-chip LVR circuitry detects when one of the supply voltages VDD, VDDX and VDDF drops below an appropriate voltage level. If LVR is deasserted the MCU is fully operational at the specified maximum speed. The LVR assert and deassert levels for the supply voltage VDDX are $V_{LVRXA}$ and $V_{LVRXD}$ and are specified in the device Reference Manual.

## 4.6 Interrupts

The interrupt/reset vectors requested by the S12CPMU_UHV are listed in Table 4-32. Refer to MCU specification for related vector addresses and priorities.

**Table 4-32. S12CPMU_UHV Interrupt Vectors**

| Interrupt Source | CCR Mask | Local Enable |
|---|---|---|
| RTI time-out interrupt | I bit | CPMUINT (RTIE) |
| PLL lock interrupt | I bit | CPMUINT (LOCKIE) |
| Oscillator status interrupt | I bit | CPMUINT (OSCIE) |
| Low voltage interrupt | I bit | CPMULVCTL (LVIE) |
| High temperature interrupt | I bit | CPMUHTCTL (HTIE) |
| Autonomous Periodical Interrupt | I bit | CPMUAPICTL (APIE) |

### 4.6.1 Description of Interrupt Operation

#### 4.6.1.1 Real Time Interrupt (RTI)

The clock source for the RTI is either IRCCLK or OSCCLK depending on the setting of the RTIOSCSEL bit. In Stop Mode with PSTP=1 (Pseudo Stop Mode), RTIOSCSEL=1 and PRE=1 the RTI continues to run, else the RTI counter halts in Stop Mode.

The RTI can be used to generate hardware interrupts at a fixed periodic rate. If enabled (by setting RTIE=1), this interrupt will occur at the rate selected by the CPMURTI register. At the end of the RTI time-out period the RTIF flag is set to one and a new RTI time-out period starts immediately.

A write to the CPMURTI register restarts the RTI time-out period.

#### 4.6.1.2 PLL Lock Interrupt

The S12CPMU_UHV generates a PLL Lock interrupt when the lock condition (LOCK status bit) of the PLL changes, either from a locked state to an unlocked state or vice versa. Lock interrupts are locally disabled by setting the LOCKIE bit to zero. 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.

#### 4.6.1.3 Oscillator Status Interrupt

When the OSCE bit is 0, then UPOSC stays 0. When OSCE=1 the UPOSC bit is set after the LOCK bit is set.

Upon detection of a status change (UPOSC) the OSCIF flag is set. Going into Full Stop Mode or disabling the oscillator can also cause a status change of UPOSC.

Any change in PLL configuration or any other event which causes the PLL lock status to be cleared leads to a loss of the oscillator status information as well (UPOSC=0).

Oscillator status change interrupts are locally enabled with the OSCIE bit.

**NOTE**

Loosing the oscillator status (UPOSC=0) affects the clock configuration of the system[1]. This needs to be dealt with in application software.

#### 4.6.1.4 Low-Voltage Interrupt (LVI)

In FPM the input voltage VDDA is monitored. Whenever VDDA drops below level $V_{LVIA,}$ the status bit LVDS is set to 1. When VDDA rises above level $V_{LVID}$ the status bit LVDS is cleared to 0. An interrupt, indicated by flag LVIF = 1, is triggered by any change of the status bit LVDS if interrupt enable bit LVIE = 1.

#### 4.6.1.5 HTI - High Temperature Interrupt

In FPM the junction temperature $T_J$ is monitored. Whenever $T_J$ exceeds level $T_{HTIA}$ the status bit HTDS is set to 1. Vice versa, HTDS is reset to 0 when $T_J$ get below level $T_{HTID}$. An interrupt, indicated by flag HTIF = 1, is triggered by any change of the status bit HTDS, if interrupt enable bit HTIE = 1.

#### 4.6.1.6 Autonomous Periodical Interrupt (API)

The API sub-block can generate periodical interrupts independent of the clock source of the MCU. To enable the timer, the bit APIFE needs to be set.

The API timer is either clocked by the Autonomous Clock (ACLK - trimmable internal RC oscillator) or the Bus Clock. Timer operation will freeze when MCU clock source is selected and Bus Clock is turned off. The clock source can be selected with bit APICLK. APICLK can only be written when APIFE is not set.

The APIR[15:0] bits determine the interrupt period. APIR[15:0] can only be written when APIFE is cleared. As soon as APIFE is set, the timer starts running for the period selected by APIR[15:0] bits. When the configured time has elapsed, the flag APIF is set. An interrupt, indicated by flag APIF = 1, is triggered if interrupt enable bit APIE = 1. The timer is re-started automatically again after it has set APIF.

The procedure to change APICLK or APIR[15:0] is first to clear APIFE, then write to APICLK or APIR[15:0], and afterwards set APIFE.

The API Trimming bits ACLKTR[5:0] must be set so the minimum period equals 0.2 ms if stable frequency is desired.

See Table 4-19 for the trimming effect of ACLKTR.

1. For details please refer to “4.4.6 System Clock Configurations”

**NOTE**

The first period after enabling the counter by APIFE might be reduced by API start up delay $t_{sdel}$.

It is possible to generate with the API a waveform at the external pin API_EXTCLK by setting APIFE and enabling the external access with setting APIEA.

## 4.7 Initialization/Application Information

### 4.7.1 General Initialization information

Usually applications run in MCU Normal Mode.

It is recommended to write the CPMUCOP register in any case from the application program initialization routine after reset no matter if the COP is used in the application or not, even if a configuration is loaded via the flash memory after reset. By doing a “controlled” write access in MCU Normal Mode (with the right value for the application) the write once for the COP configuration bits (WCOP,CR[2:0]) takes place which protects these bits from further accidental change. In case of a program sequencing issue (code runaway) the COP configuration can not be accidentally modified anymore.

### 4.7.2 Application information for COP and API usage

In many applications the COP is used to check that the program is running and sequencing properly. Often the COP is kept running during Stop Mode and periodic wake-up events are needed to service the COP on time and maybe to check the system status.

For such an application it is recommended to use the ACLK as clock source for both COP and API. This guarantees lowest possible IDD current during Stop Mode. Additionally it eases software implementation using the same clock source for both, COP and API.

The Interrupt Service Routine (ISR) of the Autonomous Periodic Interrupt API should contain the write instruction to the CPMUARMCOP register. The value (byte) written is derived from the “main routine” (alternating sequence of $55 and $AA) of the application software.

Using this method, then in the case of a runtime or program sequencing issue the application “main routine” is not executed properly anymore and the alternating values are not provided properly. Hence the COP is written at the correct time (due to independent API interrupt request) but the wrong value is written (alternating sequence of $55 and $AA is no longer maintained) which causes a COP reset.

If the COP is stopped during any Stop Mode it is recommended to service the COP shortly before Stop Mode is entered.

# Chapter 5
# Background Debug Module (S12SBDMV1)

**Table 5-1. Revision History**

| Revision Number | Date | Sections Affected | Summary of Changes |
|---|---|---|---|
| 1.05 | 07.Dec.2010 | Table 5-1 | Standardized format of revision history table header. |
| 1.06 | 02.Mar.2011 | 5.3.2.2/5-177<br>5.2/5-173 | Corrected BPAE bit description.<br>Removed references to fixed VCO frequencies |
| 1.07 | 27.Sep.2012 | General | Changed references to device level. |

## 5.1 Introduction

This section describes the functionality of the background debug module (BDM) sub-block of the HCS12S core platform.

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.

The BDM has enhanced capability for maintaining synchronization between the target and host while allowing more flexibility in clock rates. This includes a sync signal to determine the communication rate and a handshake signal to indicate when an operation is complete. The system is backwards compatible to the BDM of the S12 family with the following exceptions:

- TAGGO command not supported by S12SBDM
- External instruction tagging feature is part of the DBG module
- S12SBDM register map and register content modified
- Family ID readable from BDM ROM at global address 0x3_FF0F in active BDM (value for devices with HCS12S core is 0xC2)
- Clock switch removed from BDM (CLKSW bit removed from BDMSTS register)

### 5.1.1 Features

The BDM includes these distinctive features:

- Single-wire communication with host development system
- Enhanced capability for allowing more flexibility in clock rates
- SYNC command to determine communication rate
- GO_UNTIL command

- 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
- 14 firmware commands execute from the standard BDM firmware lookup table
- Software control of BDM operation during wait mode
- When secured, hardware commands are allowed to access the register space in special single chip mode, if the Flash erase tests fail.
- Family ID readable from BDM ROM at global address 0x3_FF0F in active BDM (value for devices with HCS12S core is 0xC2)
- BDM hardware commands are operational until system stop mode is entered

## 5.1.2 Modes of Operation

BDM is available in all operating modes but must be enabled before firmware commands are executed. Some systems may have a control bit that allows suspending the function during background debug mode.

### 5.1.2.1 Regular Run Modes

All of these operations refer to the part in run mode and not being secured. The BDM does not provide controls to conserve power during run mode.

- Normal modes

  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.

### 5.1.2.2 Secure Mode Operation

If the device 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 other than allowing erasure. For more information please see Section 5.4.1, “Security”.

### 5.1.2.3 Low-Power Modes

The BDM can be used until stop mode is entered. When CPU is in wait mode all BDM firmware commands as well as the hardware BACKGROUND command cannot be used and are ignored. In this case the CPU can not enter BDM active mode, and only hardware read and write commands are available. Also the CPU can not enter a low power mode (stop or wait) during BDM active mode.

In stop mode the BDM clocks are stopped. When BDM clocks are disabled and stop mode is exited, the BDM clocks will restart and BDM will have a soft reset (clearing the instruction register, any command in progress and disable the ACK function). The BDM is now ready to receive a new command.

### 5.1.3 Block Diagram

A block diagram of the BDM is shown in Figure 5-1.

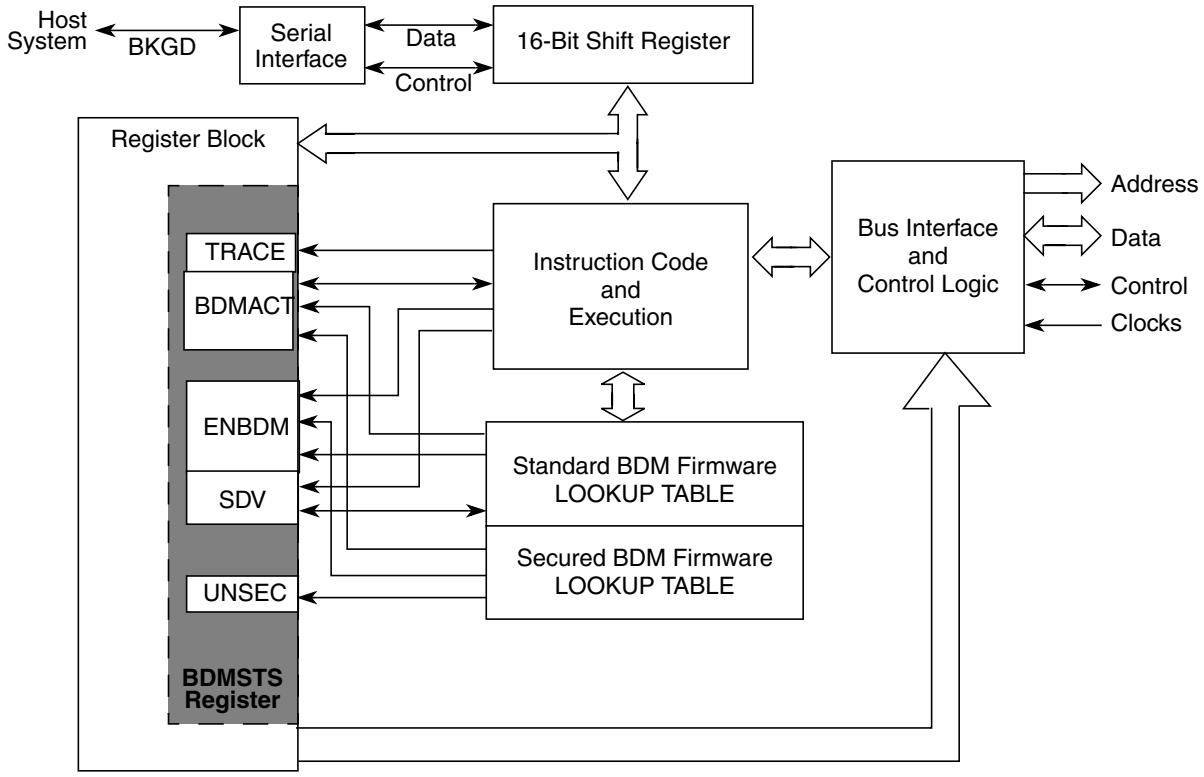



**Figure 5-1. BDM Block Diagram**

## 5.2 External Signal Description

A single-wire interface pin called the background debug interface (BKGD) pin is used to communicate with the BDM system. 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. The communication rate of this pin is always the BDM clock frequency defined at device level (refer to device overview section). When modifying the VCO clock please make sure that the communication rate is adapted accordingly and a communication time-out (BDM soft reset) has occurred.

## 5.3 Memory Map and Register Definition

### 5.3.1 Module Memory Map

Table 5-2 shows the BDM memory map when BDM is active.

**Table 5-2. BDM Memory Map**

| Global Address | Module | Size (Bytes) |
|---|---|---|
| 0x3_FF00–0x3_FF0B | BDM registers | 12 |
| 0x3_FF0C–0x3_FF0E | BDM firmware ROM | 3 |
| 0x3_FF0F | Family ID (part of BDM firmware ROM) | 1 |
| 0x3_FF10–0x3_FFFF | BDM firmware ROM | 240 |

## 5.3.2 Register Descriptions

A summary of the registers associated with the BDM is shown in Figure 5-2. Registers are accessed by host-driven communications to the BDM hardware using READ_BD and WRITE_BD commands.

| Global Address | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x3_FF00 | Reserved | R | X | X | X | X | X | X | 0 | 0 |
| | | W | | | | | | | | |
| 0x3_FF01 | BDMSTS | R | ENBDM | BDMACT | 0 | SDV | TRACE | 0 | UNSEC | 0 |
| | | W | | | | | | | | |
| 0x3_FF02 | Reserved | R | X | X | X | X | X | X | X | X |
| | | W | | | | | | | | |
| 0x3_FF03 | Reserved | R | X | X | X | X | X | X | X | X |
| | | W | | | | | | | | |
| 0x3_FF04 | Reserved | R | X | X | X | X | X | X | X | X |
| | | W | | | | | | | | |
| 0x3_FF05 | Reserved | R | X | X | X | X | X | X | X | X |
| | | W | | | | | | | | |
| 0x3_FF06 | BDMCCR | R | CCR7 | CCR6 | CCR5 | CCR4 | CCR3 | CCR2 | CCR1 | CCR0 |
| | | W | | | | | | | | |
| 0x3_FF07 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x3_FF08 | BDMPPR | R | BPAE | 0 | 0 | 0 | BPP3 | BPP2 | BPP1 | BPP0 |
| | | W | | | | | | | | |

= Unimplemented, Reserved

= Implemented (do not alter)

X = Indeterminate

0 = Always read zero

**Figure 5-2. BDM Register Summary**

| Global Address | Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x3_FF09 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x3_FF0A | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x3_FF0B | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

= Unimplemented, Reserved

= Implemented (do not alter)

X = Indeterminate

0 = Always read zero

**Figure 5-2. BDM Register Summary (continued)**

### 5.3.2.1 BDM Status Register (BDMSTS)

Register Global Address 0x3_FF01

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | ENBDM | BDMACT | 0 | SDV | TRACE | 0 | UNSEC | 0 |
| W | | | | | | | | |
| Reset | | | | | | | | |
| Special Single-Chip Mode | 0[1] | 1 | 0 | 0 | 0 | 0 | 0[2] | 0 |
| All Other Modes | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented, Reserved

= Implemented (do not alter)

0 = Always read zero

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). This is because the ENBDM bit is set by the standard BDM 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).

**Figure 5-3. BDM Status Register (BDMSTS)**

Read: All modes through BDM operation when not secured

Write: All modes through BDM operation when not secured, but subject to the following:

— 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).

— 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.

— 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.

**Table 5-3. BDMSTS Field Descriptions**

| Field | Description |
|---|---|
| 7<br>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 still allowed.<br>0 BDM disabled<br>1 BDM enabled<br>**Note:** ENBDM is set out of reset in special single chip mode. In special single chip mode with the device secured, this bit will not be set until after the Flash erase verify tests are complete. |
| 6<br>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.<br>0 BDM not active<br>1 BDM active |
| 4<br>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 BDM firmware or hardware read command or after data has been received as part of a BDM firmware or hardware 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.<br>0 Data phase of command not complete<br>1 Data phase of command is complete |
| 3<br>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 until BDM firmware is exited by one of the following BDM commands: GO or GO_UNTIL.<br>0 TRACE1 command is not being executed<br>1 TRACE1 command is being executed |
| 1<br>UNSEC | **Unsecure** — If the device is secured this bit is only writable in special single chip mode from the BDM secure firmware. 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 overlapping the standard BDM firmware lookup table.<br>The secure BDM firmware lookup table verifies that the on-chip Flash is 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.<br>0 System is in a secured mode.<br>1 System is in a unsecured mode.<br>**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.After reset this bit has no meaning or effect when the security byte in the Flash EEPROM is configured for unsecure mode. |

Register Global Address 0x3_FF06

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | CCR7 | CCR6 | CCR5 | CCR4 | CCR3 | CCR2 | CCR1 | CCR0 |
| Reset | | | | | | | | |
| Special Single-Chip Mode | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| All Other Modes | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 5-4. BDM CCR Holding Register (BDMCCR)**

Read: All modes through BDM operation when not secured

Write: All modes through BDM operation when not secured

**NOTE**

When BDM is made active, the CPU stores the content of its 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 in this CPU mode. Out of reset in all other modes the BDMCCR register is read zero.

When entering background debug mode, the BDM CCR holding register is used to save 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.

### 5.3.2.2 BDM Program Page Index Register (BDMPPR)

Register Global Address 0x3_FF08

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | BPAE | 0 | 0 | 0 | BPP3 | BPP2 | BPP1 | BPP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented, Reserved

**Figure 5-5. BDM Program Page Register (BDMPPR)**

Read: All modes through BDM operation when not secured

Write: All modes through BDM operation when not secured

**Table 5-4. BDMPPR Field Descriptions**

| Field | Description |
|---|---|
| 7<br>BPAE | **BDM Program Page Access Enable Bit** — BPAE enables program page access for BDM hardware and firmware read/write instructions The BDM hardware commands used to access the BDM registers (READ_BD and WRITE_BD) can not be used for program page accesses even if the BPAE bit is set.<br>0 BDM Program Paging disabled<br>1 BDM Program Paging enabled |
| 3–0<br>BPP[3:0] | **BDM Program Page Index Bits 3–0** — These bits define the selected program page. For more detailed information regarding the program page window scheme, please refer to the device MMC description. |

## 5.3.3 Family ID Assignment

The family ID is an 8-bit value located in the BDM ROM in active BDM (at global address: 0x3_FF0F). The read-only value is a unique family ID which is 0xC2 for devices with an HCS12S core.

# 5.4 Functional Description

The BDM receives and executes commands from a host via a single wire serial interface. There are two types of BDM commands: hardware and firmware commands.

Hardware commands are used to read and write target system memory locations and to enter active background debug mode, see Section 5.4.3, “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 5.4.4, “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 5.4.3, “BDM Hardware Commands”) and in secure mode (see Section 5.4.1, “Security”). BDM firmware commands can only be executed when the system is not secure and is in active background debug mode (BDM).

### 5.4.1 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 Flash EEPROM are erased. This being the case, the UNSEC and ENBDM 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 Flash does 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 Flash.

BDM operation is not possible in any other mode than special single chip mode when the device is secured. The device can only be unsecured via BDM serial interface in special single chip mode. More information regarding security is provided in the security section of the device documentation.

### 5.4.2 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
- CPU BGND instruction
- Breakpoint 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 a breakpoint, the type of breakpoint used determines if BDM becomes active before or after execution of the next instruction.

1. BDM is enabled and active immediately out of special single-chip reset.
2. This method is provided by the S12S_DBG module.

**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 0x3_FF00 to 0x3_FFFF. BDM registers are mapped to addresses 0x3_FF00 to 0x3_FF0B. The BDM uses these registers which are readable anytime by the BDM. However, these registers are not readable by user programs.

When BDM is activated while CPU executes code overlapping with BDM firmware space the saved program counter (PC) will be auto incremented by one from the BDM firmware, no matter what caused the entry into BDM active mode (BGND instruction, BACKGROUND command or breakpoints). In such a case the PC must be set to the next valid address via a WRITE_PC command before executing the GO command.

### 5.4.3 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, Flash, I/O and control registers.

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 still be executed in this mode. When executing a hardware command, the BDM sub-block waits for a free 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 5-5.

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.

**Table 5-5. Hardware Commands**

| Command | Opcode (hex) | Data | Description |
|---|---|---|---|
| BACKGROUND | 90 | None | Enter background mode if BDM 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. |

**Table 5-5. Hardware Commands (continued)**

| Command | Opcode (hex) | Data | Description |
|---|---|---|---|
| READ_BD_BYTE | E4 | 16-bit address<br>16-bit data out | Read from memory with standard BDM firmware lookup table in map.<br>Odd address data on low byte; even address data on high byte. |
| READ_BD_WORD | EC | 16-bit address<br>16-bit data out | Read from memory with standard BDM firmware lookup table in map.<br>Must be aligned access. |
| READ_BYTE | E0 | 16-bit address<br>16-bit data out | Read from memory with standard BDM firmware lookup table out of map.<br>Odd address data on low byte; even address data on high byte. |
| READ_WORD | E8 | 16-bit address<br>16-bit data out | Read from memory with standard BDM firmware lookup table out of map.<br>Must be aligned access. |
| WRITE_BD_BYTE | C4 | 16-bit address<br>16-bit data in | Write to memory with standard BDM firmware lookup table in map.<br>Odd address data on low byte; even address data on high byte. |
| WRITE_BD_WORD | CC | 16-bit address<br>16-bit data in | Write to memory with standard BDM firmware lookup table in map.<br>Must be aligned access. |
| WRITE_BYTE | C0 | 16-bit address<br>16-bit data in | Write to memory with standard BDM firmware lookup table out of map.<br>Odd address data on low byte; even address data on high byte. |
| WRITE_WORD | C8 | 16-bit address<br>16-bit data in | Write to memory with standard BDM firmware lookup table out of map.<br>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.

### 5.4.4 Standard BDM Firmware Commands

BDM 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 5.4.2, “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 0x3_FF00–0x3_FFFF, 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 5-6.

**Table 5-6. Firmware Commands**

| Command[1] | Opcode (hex) | Data | Description |
|---|---|---|---|
| READ_NEXT[2] | 62 | 16-bit data out | Increment X index register 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[2] | 42 | 16-bit data in | Increment X index register 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[3] | 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 -> GO | 18 | none | (Previous enable tagging and go to user program.)<br>This command will be deprecated and should not be used anymore.<br>Opcode will be executed as a GO 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 When the firmware command READ_NEXT or WRITE_NEXT is used to access the BDM address space the BDM resources are accessed rather than user code. Writing BDM firmware is not possible.

3 System stop disables the ACK function and ignored commands will not have an ACK-pulse (e.g., CPU in stop or wait mode). The GO_UNTIL command will not get an Acknowledge if CPU executes the wait or stop instruction before the “UNTIL” condition (BDM active again) is reached (see Section 5.4.7, “Serial Interface Hardware Handshake Protocol” last note).

## 5.4.5 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.

8-bit reads return 16-bits of data, only one byte of which contains 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.

16-bit misaligned reads and writes are generally not allowed. If attempted by BDM hardware command, the BDM ignores the least significant bit of the address and assumes an even address from the remaining bits.

For hardware data read commands, the external host must wait at least 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 BDM firmware read commands, the external host should wait at least 48 bus clock cycles after sending the command opcode and before attempting to obtain the read data. The 48 cycle wait allows enough time for the requested data to be made available in the BDM shift register, ready to be shifted out.

For BDM firmware write commands, the external host must wait 36 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 for at least for 76 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) is used to indicate when an operation is complete. When using ACK, the delay times are automated.

Figure 5-6 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 $8 \times 16$ target clock cycles.[1]

1. Target clock cycles are cycles measured using the target MCU's serial clock rate. See Section 5.4.6, "BDM Serial Interface" and Section 5.3.2.1, "BDM Status Register (BDMSTS)" for information on how serial clock rate is selected.

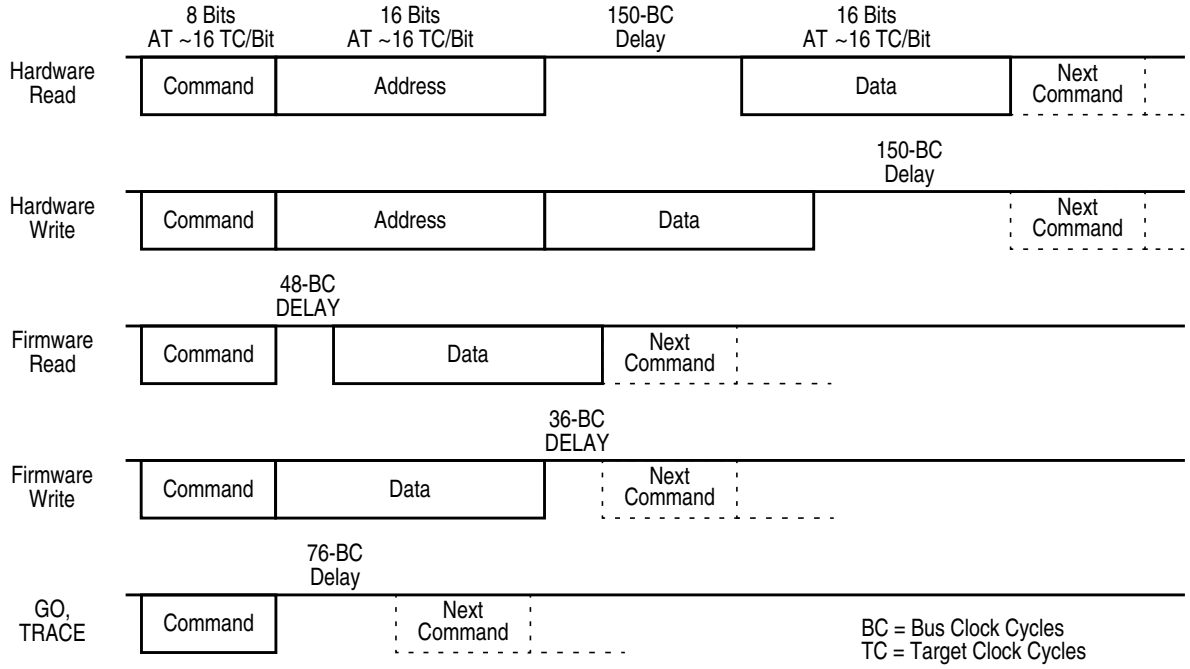



**Figure 5-6. BDM Command Structure**

## 5.4.6 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 based on the VCO clock (please refer to the CPMU Block Guide for more details), which gets divided by 8. 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. Since 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 5-7 and that of target-to-host in Figure 5-8 and Figure 5-9. All four cases begin when the host drives the BKGD pin low to generate a falling edge. Since 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 5-7 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.

Since the host drives the high speedup pulses in these two cases, the rising edges look like digitally driven signals.

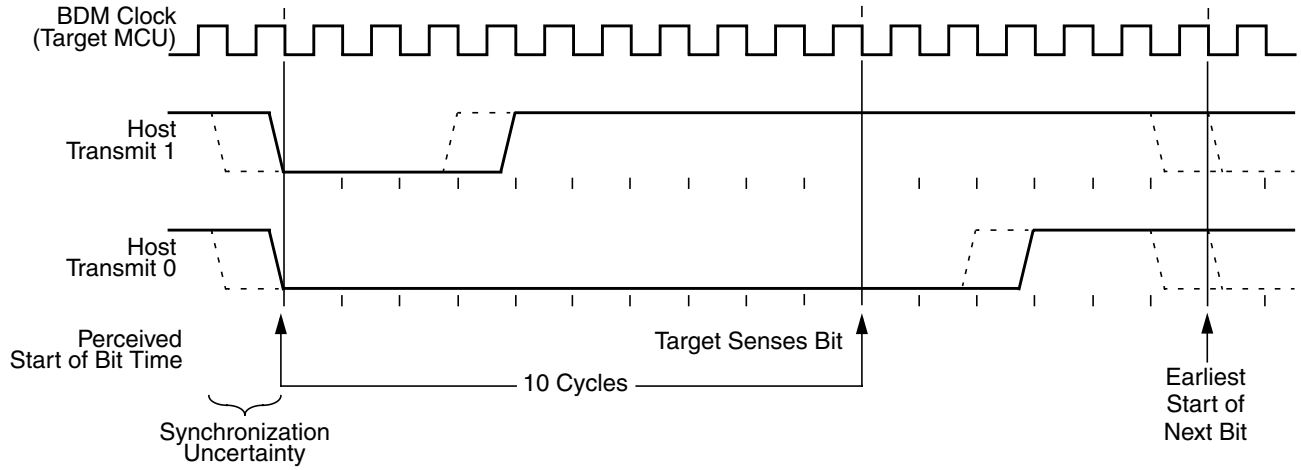



**Figure 5-7. BDM Host-to-Target Serial Bit Timing**

The receive cases are more complicated. Figure 5-8 shows the host receiving a logic 1 from the target system. Since 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.

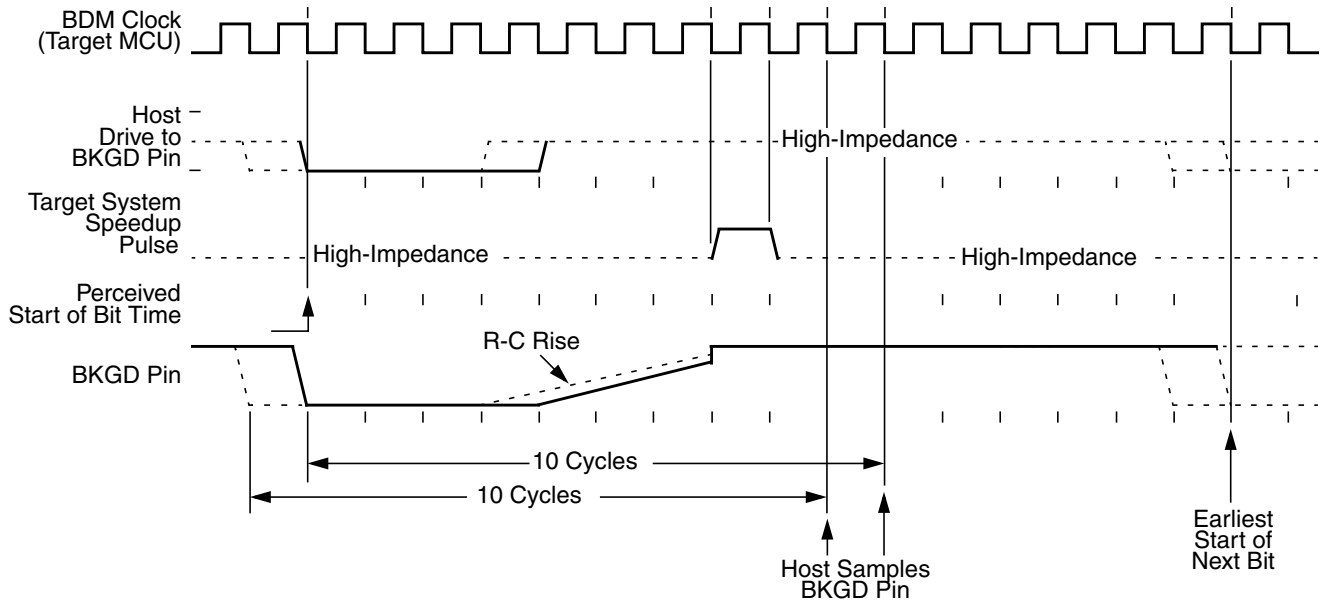



**Figure 5-8. BDM Target-to-Host Serial Bit Timing (Logic 1)**

Figure 5-9 shows the host receiving a logic 0 from the target. Since 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. Since 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.

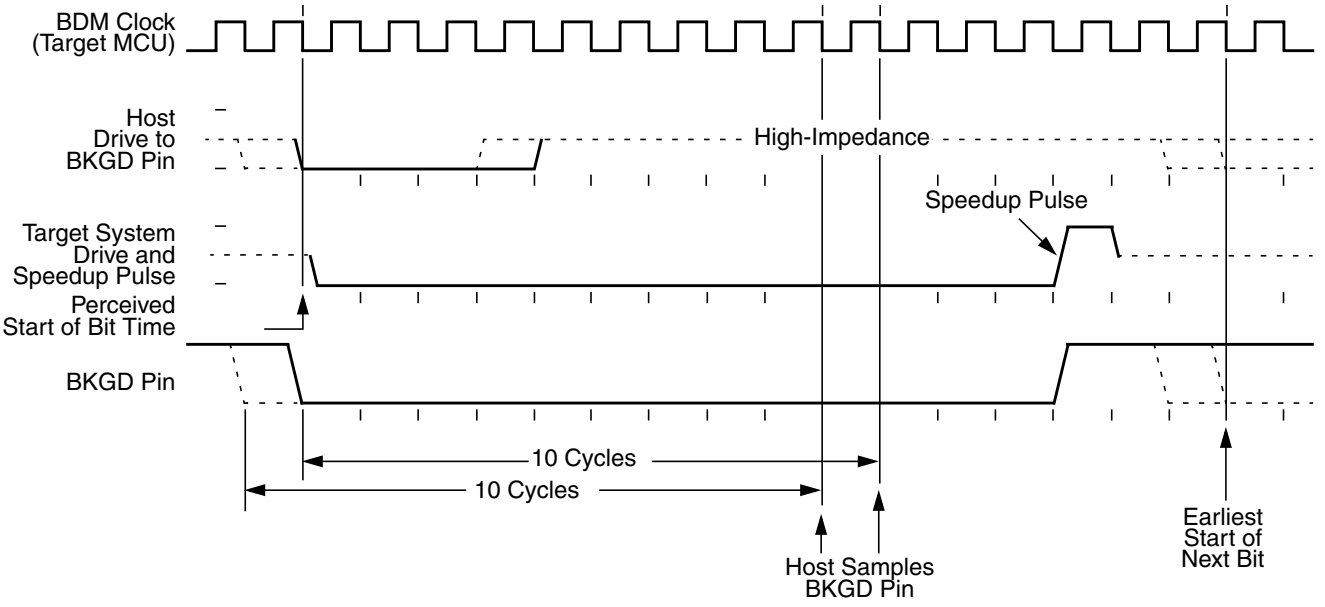



**Figure 5-9. BDM Target-to-Host Serial Bit Timing (Logic 0)**

### 5.4.7 Serial Interface Hardware Handshake Protocol

BDM commands that require CPU execution are ultimately treated at the MCU bus rate. Since the BDM clock source can be modified when changing the settings for the VCO frequency (CPMUSYNR), 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 BDM clock frequency is always VCO frequency divided by 8. 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 5-10). 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, since the command execution depends upon the CPU bus, which in some cases could be very slow due to long accesses taking place.This protocol allows a great flexibility for the POD designers, since it does not rely on any accurate time measurement or short response time to any event in the serial communication.

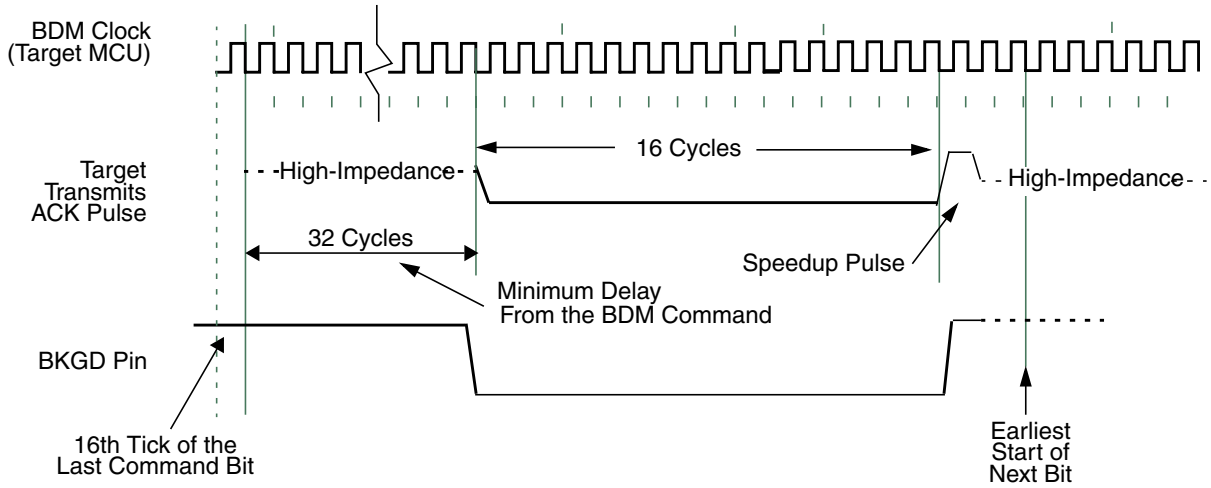



**Figure 5-10. Target Acknowledge Pulse (ACK)**

**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 5-11 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.

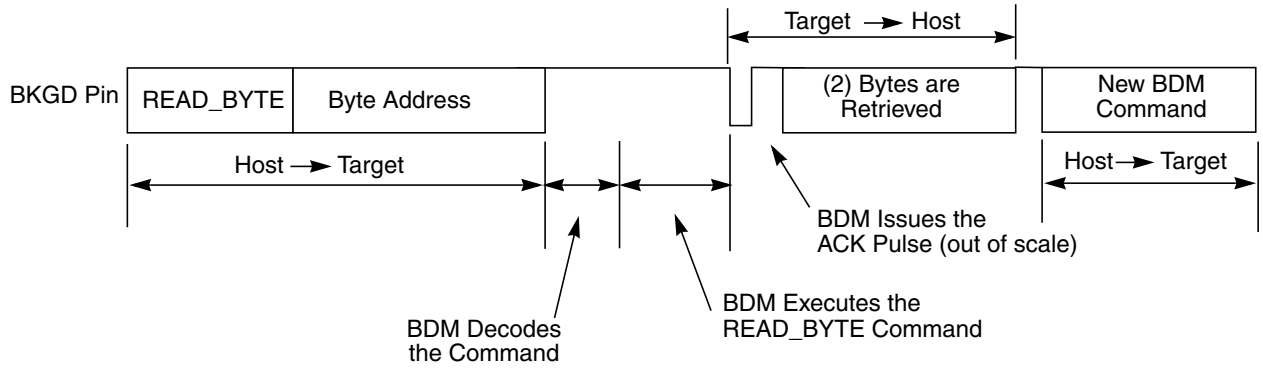



**Figure 5-11. Handshake Protocol at Command Level**

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 negative edge in the BKGD pin. The hardware handshake protocol in Figure 5-10 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 hardware command (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 (not aware of stop or wait) should decide to abort any possible pending ACK pulse in order to be sure a new command can be issued. Therefore, the protocol provides a mechanism in which a command, and its corresponding ACK, can be aborted.

**NOTE**

The ACK pulse does not provide a time out. This means for the GO_UNTIL command that it can not be distinguished if a stop or wait has been executed (command discarded and ACK not issued) or if the “UNTIL” condition (BDM active) is just not reached yet. Hence in any case where the ACK pulse of a command is not issued the possible pending command should be aborted before issuing a new command. See the handshake abort procedure described in Section 5.4.8, “Hardware Handshake Abort Procedure”.

### 5.4.8 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 5.4.9, “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. For BDM firmware READ or WRITE commands it can not be guaranteed that the pending command is aborted when issuing a SYNC before the corresponding ACK pulse. There is a short latency time from the time the READ or WRITE access begins until it is finished and the corresponding ACK pulse is issued. The latency time depends on the firmware READ or WRITE command that is issued and on the selected bus clock rate. When the SYNC command starts during this latency time the READ or WRITE command will not be aborted, but the corresponding ACK pulse will be aborted. A pending GO, TRACE1 or GO_UNTIL command can not be aborted. Only the corresponding ACK pulse can be aborted by the SYNC command.

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 negative 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 negative 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 negative 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.

Since 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 5.4.9, "SYNC — Request Timed Reference Pulse".

Figure 5-12 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.

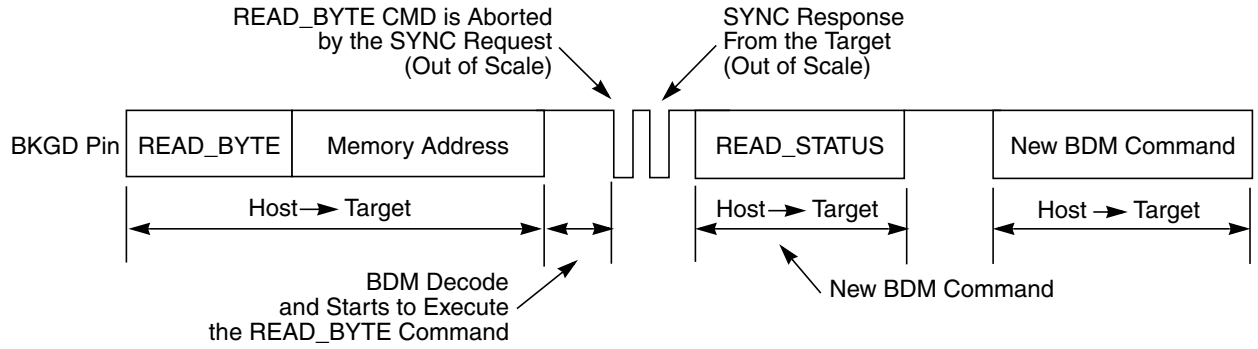



**Figure 5-12. ACK Abort Procedure at the Command Level**

**NOTE**

Figure 5-12 does not represent the signals in a true timing scale

Figure 5-13 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. Since this is not a probable situation, the protocol does not prevent this conflict from happening.

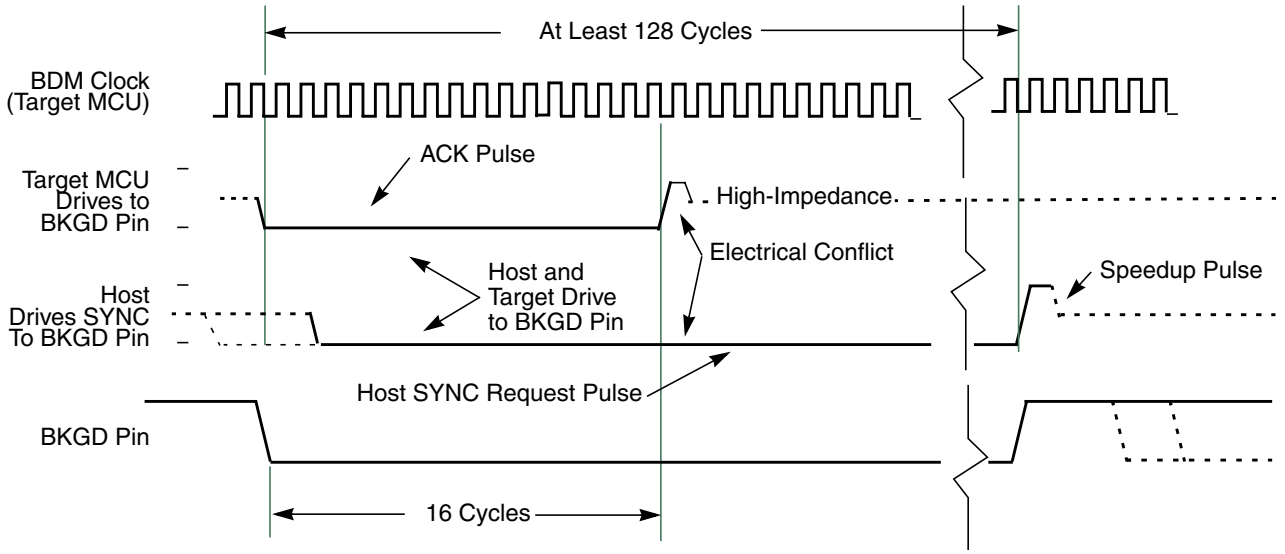



**Figure 5-13. ACK Pulse and SYNC Request Conflict**

**NOTE**

This information is being provided so that the MCU integrator will be aware that such a conflict could 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 5.4.3, “BDM Hardware Commands” and Section 5.4.4, “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 since 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.

### 5.4.9 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 settings for the VCO clock (CPMUSYNR). The BDM clock frequency is always VCO clock frequency divided by 8.)
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 one.
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 negative 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.

### 5.4.10 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. Once 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. Once back in standard BDM firmware execution, the program counter points to the first instruction in the interrupt service routine.

Be aware when tracing through the user code that the execution of the user code is done step by step but all peripherals are free running. Hence possible timing relations between CPU code execution and occurrence of events of other peripherals no longer exist.

Do not trace the CPU instruction BGND used for soft breakpoints. Tracing over the BGND instruction will result in a return address pointing to BDM firmware address space.

When tracing through user code which contains stop or wait instructions the following will happen when the stop or wait instruction is traced:

The CPU enters stop or wait mode and the TRACE1 command can not be finished before leaving the low power mode. This is the case because BDM active mode can not be entered after CPU executed the stop instruction. However all BDM hardware commands except the BACKGROUND command are operational after tracing a stop or wait instruction and still being in stop or wait mode. If system stop mode is entered (all bus masters are in stop mode) no BDM command is operational.

As soon as stop or wait mode is exited the CPU enters BDM active mode and the saved PC value points to the entry of the corresponding interrupt service routine.

In case the handshake feature is enabled the corresponding ACK pulse of the TRACE1 command will be discarded when tracing a stop or wait instruction. Hence there is no ACK pulse when BDM active mode is entered as part of the TRACE1 command after CPU exited from stop or wait mode. All valid commands sent during CPU being in stop or wait mode or after CPU exited from stop or wait mode will have an ACK pulse. The handshake feature becomes disabled only when system stop mode has been reached. Hence after a system stop mode the handshake feature must be enabled again by sending the ACK_ENABLE command.

### 5.4.11 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. In order to allow the data to be retrieved even with a large clock frequency mismatch (between BDM 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 still be able to retrieve the data from an issued read command. However, once 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 negative edge in 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 negative edge in 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.

# Chapter 6
# S12S Debug Module (S12DBGV2)

**Table 6-1. Revision History**

| Revision Number | Revision Date | Sections Affected | Summary of Changes |
|---|---|---|---|
| 02.08 | 09.MAY.2008 | General | Spelling corrections. Revision history format changed. |
| 02.09 | 29.MAY.2008 | 6.4.5.4 | Added note for end aligned, PurePC, rollover case. |
| 02.10 | 27.SEP.2012 | General | Changed cross reference formats |

## 6.1 Introduction

The S12DBGV2 module provides an on-chip trace buffer with flexible triggering capability to allow non-intrusive debug of application software. The S12DBGV2 module is optimized for S12SCPU debugging.

Typically the S12DBGV2 module is used in conjunction with the S12SBDM module, whereby the user configures the S12DBGV2 module for a debugging session over the BDM interface. Once configured the S12DBGV2 module is armed and the device leaves BDM returning control to the user program, which is then monitored by the S12DBGV2 module. Alternatively the S12DBGV2 module can be configured over a serial interface using SWI routines.

### 6.1.1 Glossary Of Terms

COF: Change Of Flow. Change in the program flow due to a conditional branch, indexed jump or interrupt

BDM: Background Debug Mode

S12SBDM: Background Debug Module

DUG: Device User Guide, describing the features of the device into which the DBG is integrated

WORD: 16-bit data entity

Data Line: 20-bit data entity

CPU: S12SCPU module

DBG: S12SDBG module

POR: Power On Reset

Tag: Tags can be attached to CPU opcodes as they enter the instruction pipe. If the tagged opcode reaches the execution stage a tag hit occurs.

### 6.1.2 Overview

The comparators monitor the bus activity of the CPU module. A match can initiate a state sequencer transition. On a transition to the Final State, bus tracing is triggered and/or a breakpoint can be generated.

Independent of comparator matches a transition to Final State with associated tracing and breakpoint can be triggered immediately by writing to the TRIG control bit.

The trace buffer is visible through a 2-byte window in the register address map and can be read out using standard 16-bit word reads. Tracing is disabled when the MCU system is secured.

### 6.1.3 Features

- Three comparators (A, B and C)
  - — Comparators A compares the full address bus and full 16-bit data bus
  - — Comparator A features a data bus mask register
  - — Comparators B and C compare the full address bus only
  - — Each comparator features selection of read or write access cycles
  - — Comparator B allows selection of byte or word access cycles
  - — Comparator matches can initiate state sequencer transitions
- Three comparator modes
  - — Simple address/data comparator match mode
  - — Inside address range mode, Addmin ≤ Address ≤ Addmax
  - — Outside address range match mode, Address < Addmin or Address > Addmax
- Two types of matches
  - — Tagged — This matches just before a specific instruction begins execution
  - — Force — This is valid on the first instruction boundary after a match occurs
- Two types of breakpoints
  - — CPU breakpoint entering BDM on breakpoint (BDM)
  - — CPU breakpoint executing SWI on breakpoint (SWI)
- Trigger mode independent of comparators
  - — TRIG Immediate software trigger
- Four trace modes
  - — Normal: change of flow (COF) PC information is stored (see Section 6.4.5.2.1, “Normal Mode) for change of flow definition.
  - — Loop1: same as Normal but inhibits consecutive duplicate source address entries
  - — Detail: address and data for all cycles except free cycles and opcode fetches are stored
  - — Compressed Pure PC: all program counter addresses are stored

- 4-stage state sequencer for trace buffer control
  - — Tracing session trigger linked to Final State of state sequencer
  - — Begin and End alignment of tracing to trigger

## 6.1.4 Modes of Operation

The DBG module can be used in all MCU functional modes.

During BDM hardware accesses and whilst the BDM module is active, CPU monitoring is disabled. When the CPU enters active BDM Mode through a BACKGROUND command, the DBG module, if already armed, remains armed.

The DBG module tracing is disabled if the MCU is secure, however, breakpoints can still be generated.

**Table 6-2. Mode Dependent Restriction Summary**

| BDM Enable | BDM Active | MCU Secure | Comparator Matches Enabled | Breakpoints Possible | Tagging Possible | Tracing Possible |
|---|---|---|---|---|---|---|
| x | x | 1 | Yes | Yes | Yes | No |
| 0 | 0 | 0 | Yes | Only SWI | Yes | Yes |
| 0 | 1 | 0 | Active BDM not possible when not enabled | | | |
| 1 | 0 | 0 | Yes | Yes | Yes | Yes |
| 1 | 1 | 0 | No | No | No | No |

## 6.1.5 Block Diagram

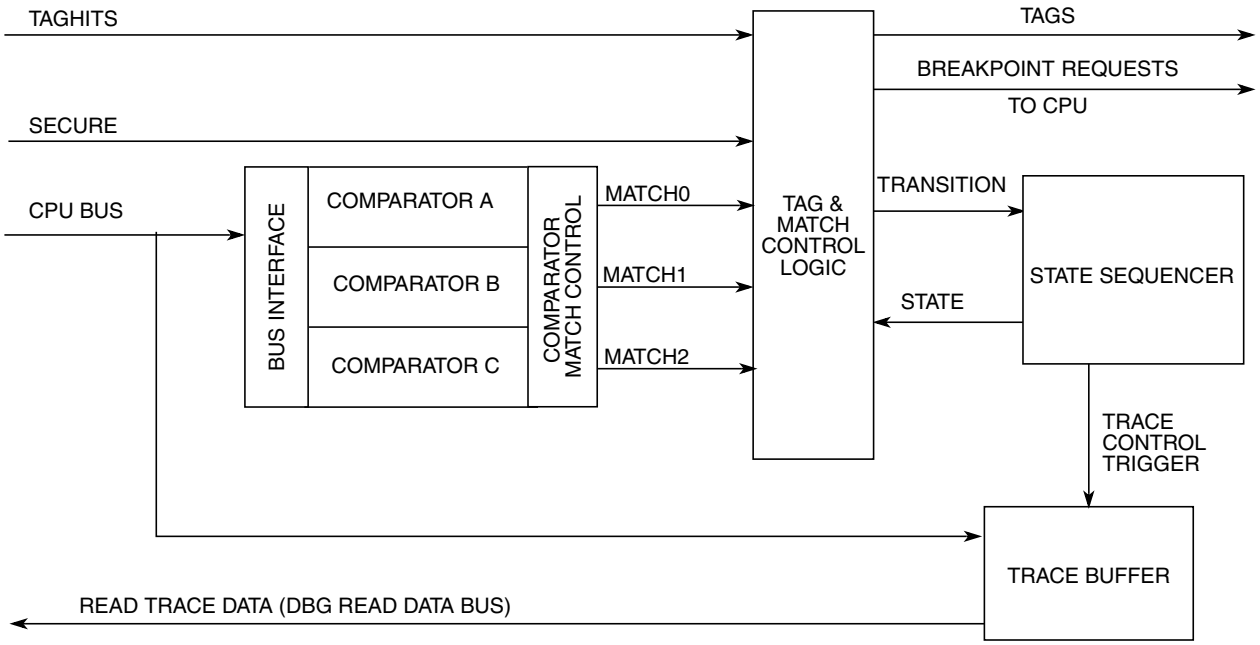



**Figure 6-1. Debug Module Block Diagram**

## 6.2 External Signal Description

There are no external signals associated with this module.

## 6.3 Memory Map and Registers

### 6.3.1 Module Memory Map

A summary of the registers associated with the DBG sub-block is shown in Figure 6-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

| Address | Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0020 | DBGC1 | R | ARM | 0 | 0 | BDM | DBGBRK | 0 | COMRV | |
| | | W | | TRIG | | | | | | |
| 0x0021 | DBGSR | R | [1]TBF | 0 | 0 | 0 | 0 | SSF2 | SSF1 | SSF0 |
| | | W | | | | | | | | |
| 0x0022 | DBGTCR | R | 0 | TSOURCE | 0 | 0 | TRCMOD | | 0 | TALIGN |
| | | W | | | | | | | | |
| 0x0023 | DBGC2 | R | 0 | 0 | 0 | 0 | 0 | 0 | ABCM | |
| | | W | | | | | | | | |
| 0x0024 | DBGTBH | R | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x0025 | DBGTBL | R | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x0026 | DBGCNT | R | [1] TBF | 0 | CNT | | | | | |
| | | W | | | | | | | | |
| 0x0027 | DBGSCRX | R | 0 | 0 | 0 | 0 | SC3 | SC2 | SC1 | SC0 |
| | | W | | | | | | | | |
| 0x0027 | DBGMFR | R | 0 | 0 | 0 | 0 | 0 | MC2 | MC1 | MC0 |
| | | W | | | | | | | | |
| [2] 0x0028 | DBGACTL | R | SZE | SZ | TAG | BRK | RW | RWE | NDB | COMPE |
| | | W | | | | | | | | |
| [3] 0x0028 | DBGBCTL | R | SZE | SZ | TAG | BRK | RW | RWE | 0 | COMPE |
| | | W | | | | | | | | |
| [4] 0x0028 | DBGCCTL | R | 0 | 0 | TAG | BRK | RW | RWE | 0 | COMPE |
| | | W | | | | | | | | |
| 0x0029 | DBGXAH | R | 0 | 0 | 0 | 0 | 0 | 0 | Bit 17 | Bit 16 |
| | | W | | | | | | | | |
| 0x002A | DBGXAM | R | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x002B | DBGXAL | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | | | | | | | | |

**Figure 6-2. Quick Reference to DBG Registers**

| Address | Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x002C | DBGADH | R<br>W | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| 0x002D | DBGADL | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x002E | DBGADHM | R<br>W | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| 0x002F | DBGADLM | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |

1 This bit is visible at DBGCNT[7] and DBGSR[7]

2 This represents the contents if the Comparator A control register is blended into this address.

3 This represents the contents if the Comparator B control register is blended into this address

4 This represents the contents if the Comparator C control register is blended into this address

**Figure 6-2. Quick Reference to DBG Registers**

## 6.3.2 Register Descriptions

This section consists of the DBG control and trace buffer register descriptions in address order. Each comparator has a bank of registers that are visible through an 8-byte window between 0x0028 and 0x002F in the DBG module register address map. When ARM is set in DBGC1, the only bits in the DBG module registers that can be written are ARM, TRIG, and COMRV[1:0].

### 6.3.2.1 Debug Control Register 1 (DBGC1)

Address: 0x0020

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | ARM | 0 | 0 | BDM | DBGBRK | 0 | COMRV | |
| W | | TRIG | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-3. Debug Control Register (DBGC1)**

Read: Anytime

Write: Bits 7, 1, 0 anytime
Bit 6 can be written anytime but always reads back as 0.
Bits 4:3 anytime DBG is not armed.

**NOTE**

When disarming the DBG by clearing ARM with software, the contents of bits[4:3] are not affected by the write, since up until the write operation, ARM = 1 preventing these bits from being written. These bits must be cleared using a second write if required.

**Table 6-3. DBGC1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>ARM | **Arm Bit** — The ARM bit controls whether the DBG module is armed. This bit can be set and cleared by user software and is automatically cleared on completion of a debug session, or if a breakpoint is generated with tracing not enabled. On setting this bit the state sequencer enters State1.<br>0 Debugger disarmed<br>1 Debugger armed |
| 6<br>TRIG | **Immediate Trigger Request Bit** — This bit when written to 1 requests an immediate trigger independent of state sequencer status. When tracing is complete a forced breakpoint may be generated depending upon DBGBRK and BDM bit settings. This bit always reads back a 0. Writing a 0 to this bit has no effect. If the DBGTCR_TSOURCE bit is clear no tracing is carried out. If tracing has already commenced using BEGIN trigger alignment, it continues until the end of the tracing session as defined by the TALIGN bit, thus TRIG has no affect. In secure mode tracing is disabled and writing to this bit cannot initiate a tracing session.<br>The session is ended by setting TRIG and ARM simultaneously.<br>0 Do not trigger until the state sequencer enters the Final State.<br>1 Trigger immediately |
| 4<br>BDM | **Background Debug Mode Enable** — This bit determines if a breakpoint causes the system to enter Background Debug Mode (BDM) or initiate a Software Interrupt (SWI). If this bit is set but the BDM is not enabled by the ENBDM bit in the BDM module, then breakpoints default to SWI.<br>0 Breakpoint to Software Interrupt if BDM inactive. Otherwise no breakpoint.<br>1 Breakpoint to BDM, if BDM enabled. Otherwise breakpoint to SWI |
| 3<br>DBGBRK | **S12DBGV2 Breakpoint Enable Bit** — The DBGBRK bit controls whether the debugger will request a breakpoint on reaching the state sequencer Final State. If tracing is enabled, the breakpoint is generated on completion of the tracing session. If tracing is not enabled, the breakpoint is generated immediately.<br>0 No Breakpoint generated<br>1 Breakpoint generated |
| 1–0<br>COMRV | **Comparator Register Visibility Bits** — These bits determine which bank of comparator register is visible in the 8-byte window of the S12SDBG module address map, located between 0x0028 to 0x002F. Furthermore these bits determine which register is visible at the address 0x0027. See Table 6-4. |

**Table 6-4. COMRV Encoding**

| COMRV | Visible Comparator | Visible Register at 0x0027 |
|---|---|---|
| 00 | Comparator A | DBGSCR1 |
| 01 | Comparator B | DBGSCR2 |
| 10 | Comparator C | DBGSCR3 |
| 11 | None | DBGMFR |

## 6.3.2.2 Debug Status Register (DBGSR)

Address: 0x0021

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | TBF | 0 | 0 | 0 | 0 | SSF2 | SSF1 | SSF0 |
| W | | | | | | | | |
| Reset | — | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-4. Debug Status Register (DBGSR)**

Read: Anytime

Write: Never

**Table 6-5. DBGSR Field Descriptions**

| Field | Description |
|---|---|
| 7<br>TBF | **Trace Buffer Full** — The TBF bit indicates that the trace buffer has stored 64 or more lines of data since it was last armed. If this bit is set, then all 64 lines will be valid data, regardless of the value of DBGCNT bits. The TBF bit is cleared when ARM in DBGC1 is written to a one. The TBF is cleared by the power on reset initialization. Other system generated resets have no affect on this bit<br>This bit is also visible at DBGCNT[7] |
| 2–0<br>SSF[2:0] | **State Sequencer Flag Bits** — The SSF bits indicate in which state the State Sequencer is currently in. During a debug session on each transition to a new state these bits are updated. If the debug session is ended by software clearing the ARM bit, then these bits retain their value to reflect the last state of the state sequencer before disarming. If a debug session is ended by an internal event, then the state sequencer returns to state0 and these bits are cleared to indicate that state0 was entered during the session. On arming the module the state sequencer enters state1 and these bits are forced to SSF[2:0] = 001. See Table 6-6**.** |

**Table 6-6. SSF[2:0] — State Sequence Flag Bit Encoding**

| SSF[2:0] | Current State |
|---|---|
| 000 | State0 (disarmed) |
| 001 | State1 |
| 010 | State2 |
| 011 | State3 |
| 100 | Final State |
| 101,110,111 | Reserved |

### 6.3.2.3 Debug Trace Control Register (DBGTCR)

Address: 0x0022

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | TSOURCE | 0 | 0 | TRCMOD | | 0 | TALIGN |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 6-5. Debug Trace Control Register (DBGTCR)**

Read: Anytime

Write: Bit 6 only when DBG is neither secure nor armed.Bits 3,2,0 anytime the module is disarmed.

**Table 6-7. DBGTCR Field Descriptions**

| Field | Description |
|---|---|
| 6<br>TSOURCE | **Trace Source Control Bit** — The TSOURCE bit enables a tracing session given a trigger condition. If the MCU system is secured, this bit cannot be set and tracing is inhibited.<br>This bit must be set to read the trace buffer.<br>0 Debug session without tracing requested<br>1 Debug session with tracing requested |
| 3–2<br>TRCMOD | **Trace Mode Bits** — See Section 6.4.5.2, “Trace Modes for detailed Trace Mode descriptions. In Normal Mode, change of flow information is stored. In Loop1 Mode, change of flow information is stored but redundant entries into trace memory are inhibited. In Detail Mode, address and data for all memory and register accesses is stored. In Compressed Pure PC mode the program counter value for each instruction executed is stored. See Table 6-8. |
| 0<br>TALIGN | **Trigger Align Bit** — This bit controls whether the trigger is aligned to the beginning or end of a tracing session.<br>0 Trigger at end of stored data<br>1 Trigger before storing data |

**Table 6-8. TRCMOD Trace Mode Bit Encoding**

| TRCMOD | Description |
|---|---|
| 00 | Normal |
| 01 | Loop1 |
| 10 | Detail |
| 11 | Compressed Pure PC |

### 6.3.2.4 Debug Control Register2 (DBGC2)

Address: 0x0023

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | ABCM | |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-6. Debug Control Register2 (DBGC2)**

Read: Anytime

Write: Anytime the module is disarmed.

This register configures the comparators for range matching.

**Table 6-9. DBGC2 Field Descriptions**

| Field | Description |
|---|---|
| 1–0<br>ABCM[1:0] | **A and B Comparator Match Control** — These bits determine the A and B comparator match mapping as described in Table 6-10. |

**Table 6-10. ABCM Encoding**

| ABCM | Description |
|---|---|
| 00 | Match0 mapped to comparator A match: Match1 mapped to comparator B match. |
| 01 | Match 0 mapped to comparator A/B inside range: Match1 disabled. |
| 10 | Match 0 mapped to comparator A/B outside range: Match1 disabled. |
| 11 | Reserved[1] |

[1] Currently defaults to Comparator A, Comparator B disabled

### 6.3.2.5 Debug Trace Buffer Register (DBGTBH:DBGTBL)

Address: 0x0024, 0x0025

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| POR | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
| Other Resets | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |

**Figure 6-7. Debug Trace Buffer Register (DBGTB)**

Read: Only when unlocked AND unsecured AND not armed AND TSOURCE set.

Write: Aligned word writes when disarmed unlock the trace buffer for reading but do not affect trace buffer contents.

**Table 6-11. DBGTB Field Descriptions**

| Field | Description |
|---|---|
| 15–0<br>Bit[15:0] | **Trace Buffer Data Bits** — The Trace Buffer Register is a window through which the 20-bit wide data lines of the Trace Buffer may be read 16 bits at a time. Each valid read of DBGTB increments an internal trace buffer pointer which points to the next address to be read. When the ARM bit is set the trace buffer is locked to prevent reading. The trace buffer can only be unlocked for reading by writing to DBGTB with an aligned word write when the module is disarmed. The DBGTB register can be read only as an aligned word, any byte reads or misaligned access of these registers return 0 and do not cause the trace buffer pointer to increment to the next trace buffer address. Similarly reads while the debugger is armed or with the TSOURCE bit clear, return 0 and do not affect the trace buffer pointer. The POR state is undefined. Other resets do not affect the trace buffer contents. |

### 6.3.2.6 Debug Count Register (DBGCNT)

Address: 0x0026

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | TBF | 0 | CNT | | | | | |
| W | | | | | | | | |
| Reset | — | — | — | — | — | — | — | — |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-8. Debug Count Register (DBGCNT)**

Read: Anytime

Write: Never

**Table 6-12. DBGCNT Field Descriptions**

| Field | Description |
|---|---|
| 7<br>TBF | **Trace Buffer Full** — The TBF bit indicates that the trace buffer has stored 64 or more lines of data since it was last armed. If this bit is set, then all 64 lines will be valid data, regardless of the value of DBGCNT bits. The TBF bit is cleared when ARM in DBGC1 is written to a one. The TBF is cleared by the power on reset initialization. Other system generated resets have no affect on this bit<br>This bit is also visible at DBGSR[7] |
| 5–0<br>CNT[5:0] | **Count Value** — The CNT bits indicate the number of valid data 20-bit data lines stored in the Trace Buffer. Table 6-13 shows the correlation between the CNT bits and the number of valid data lines in the Trace Buffer. When the CNT rolls over to zero, the TBF bit in DBGSR is set and incrementing of CNT will continue in end-trigger mode. The DBGCNT register is cleared when ARM in DBGC1 is written to a one. The DBGCNT register is cleared by power-on-reset initialization but is not cleared by other system resets. Thus should a reset occur during a debug session, the DBGCNT register still indicates after the reset, the number of valid trace buffer entries stored before the reset occurred. The DBGCNT register is not decremented when reading from the trace buffer. |

**Table 6-13. CNT Decoding Table**

| TBF | CNT[5:0] | Description |
|---|---|---|
| 0 | 000000 | No data valid |
| 0 | 000001<br>000010<br>000100<br>000110<br>..<br>111111 | 1 line valid<br>2 lines valid<br>4 lines valid<br>6 lines valid<br>..<br>63 lines valid |
| 1 | 000000 | 64 lines valid; if using Begin trigger alignment,<br>ARM bit will be cleared and the tracing session ends. |
| 1 | 000001<br>..<br>..<br>111110 | 64 lines valid,<br>oldest data has been overwritten by most recent data |

### 6.3.2.7 Debug State Control Registers

There is a dedicated control register for each of the state sequencer states 1 to 3 that determines if transitions from that state are allowed, depending upon comparator matches or tag hits, and defines the next state for the state sequencer following a match. The three debug state control registers are located at the same address in the register address map (0x0027). Each register can be accessed using the COMRV bits in DBGC1 to blend in the required register. The COMRV = 11 value blends in the match flag register (DBGMFR).

**Table 6-14. State Control Register Access Encoding**

| COMRV | Visible State Control Register |
|---|---|
| 00 | DBGSCR1 |
| 01 | DBGSCR2 |
| 10 | DBGSCR3 |
| 11 | DBGMFR |

#### 6.3.2.7.1 Debug State Control Register 1 (DBGSCR1)

Address: 0x0027

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | SC3 | SC2 | SC1 | SC0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-9. Debug State Control Register 1 (DBGSCR1)**

Read: If COMRV[1:0] = 00

Write: If COMRV[1:0] = 00 and DBG is not armed.

This register is visible at 0x0027 only with COMRV[1:0] = 00. The state control register 1 selects the targeted next state whilst in State1. The matches refer to the match channels of the comparator match control logic as depicted in Figure 6-1 and described in Section 6.3.2.8.1, “Debug Comparator Control Register (DBGXCTL). Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

**Table 6-15. DBGSCR1 Field Descriptions**

| Field | Description |
|---|---|
| 3–0<br>SC[3:0] | These bits select the targeted next state whilst in State1, based upon the match event. |

**Table 6-16. State1 Sequencer Next State Selection**

| SC[3:0] | Description (Unspecified matches have no effect) |
|---|---|
| 0000 | Any match to Final State |
| 0001 | Match1 to State3 |
| 0010 | Match2 to State2 |
| 0011 | Match1 to State2 |
| 0100 | Match0 to State2..... Match1 to State3 |
| 0101 | Match1 to State3.....Match0 to Final State |
| 0110 | Match0 to State2..... Match2 to State3 |
| 0111 | Either Match0 or Match1 to State2 |
| 1000 | Reserved |
| 1001 | Match0 to State3 |
| 1010 | Reserved |
| 1011 | Reserved |
| 1100 | Reserved |
| 1101 | Either Match0 or Match2 to Final State.....Match1 to State2 |
| 1110 | Reserved |
| 1111 | Reserved |

The priorities described in Table 6-36 dictate that in the case of simultaneous matches, a match leading to final state has priority followed by the match on the lower channel number (0,1,2). Thus with SC[3:0]=1101 a simultaneous match0/match1 transitions to final state.

### 6.3.2.7.2 Debug State Control Register 2 (DBGSCR2)

Address: 0x0027

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | SC3 | SC2 | SC1 | SC0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-10. Debug State Control Register 2 (DBGSCR2)**

Read: If COMRV[1:0] = 01

Write: If COMRV[1:0] = 01 and DBG is not armed.

This register is visible at 0x0027 only with COMRV[1:0] = 01. The state control register 2 selects the targeted next state whilst in State2. The matches refer to the match channels of the comparator match control logic as depicted in Figure 6-1 and described in Section 6.3.2.8.1, “Debug Comparator Control Register (DBGXCTL). Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

**Table 6-17. DBGSCR2 Field Descriptions**

| Field | Description |
|---|---|
| 3–0<br>SC[3:0] | These bits select the targeted next state whilst in State2, based upon the match event. |

**Table 6-18. State2 —Sequencer Next State Selection**

| SC[3:0] | Description (Unspecified matches have no effect) |
|---|---|
| 0000 | Match0 to State1..... Match2 to State3. |
| 0001 | Match1 to State3 |
| 0010 | Match2 to State3 |
| 0011 | Match1 to State3..... Match0 Final State |
| 0100 | Match1 to State1..... Match2 to State3. |
| 0101 | Match2 to Final State |
| 0110 | Match2 to State1..... Match0 to Final State |
| 0111 | Either Match0 or Match1 to Final State |
| 1000 | Reserved |
| 1001 | Reserved |
| 1010 | Reserved |
| 1011 | Reserved |
| 1100 | Either Match0 or Match1 to Final State.....Match2 to State3 |
| 1101 | Reserved |
| 1110 | Reserved |
| 1111 | Either Match0 or Match1 to Final State.....Match2 to State1 |

The priorities described in Table 6-36 dictate that in the case of simultaneous matches, a match leading to final state has priority followed by the match on the lower channel number (0,1,2).

### 6.3.2.7.3 Debug State Control Register 3 (DBGSCR3)

Address: 0x0027

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | SC3 | SC2 | SC1 | SC0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-11. Debug State Control Register 3 (DBGSCR3)**

Read: If COMRV[1:0] = 10

Write: If COMRV[1:0] = 10 and DBG is not armed.

This register is visible at 0x0027 only with COMRV[1:0] = 10. The state control register three selects the targeted next state whilst in State3. The matches refer to the match channels of the comparator match control logic as depicted in Figure 6-1 and described in Section 6.3.2.8.1, “Debug Comparator Control Register (DBGXCTL). Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

**Table 6-19. DBGSCR3 Field Descriptions**

| Field | Description |
|---|---|
| 3–0<br>SC[3:0] | These bits select the targeted next state whilst in State3, based upon the match event. |

**Table 6-20. State3 — Sequencer Next State Selection**

| SC[3:0] | Description (Unspecified matches have no effect) |
|---|---|
| 0000 | Match0 to State1 |
| 0001 | Match2 to State2..... Match1 to Final State |
| 0010 | Match0 to Final State..... Match1 to State1 |
| 0011 | Match1 to Final State..... Match2 to State1 |
| 0100 | Match1 to State2 |
| 0101 | Match1 to Final State |
| 0110 | Match2 to State2..... Match0 to Final State |
| 0111 | Match0 to Final State |
| 1000 | Reserved |
| 1001 | Reserved |
| 1010 | Either Match1 or Match2 to State1..... Match0 to Final State |
| 1011 | Reserved |
| 1100 | Reserved |
| 1101 | Either Match1 or Match2 to Final State..... Match0 to State1 |
| 1110 | Match0 to State2..... Match2 to Final State |
| 1111 | Reserved |

The priorities described in Table 6-36 dictate that in the case of simultaneous matches, a match leading to final state has priority followed by the match on the lower channel number (0,1,2).

#### 6.3.2.7.4 Debug Match Flag Register (DBGMFR)

Address: 0x0027

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | MC2 | MC1 | MC0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-12. Debug Match Flag Register (DBGMFR)**

Read: If COMRV[1:0] = 11

Write: Never

DBGMFR is visible at 0x0027 only with COMRV[1:0] = 11. It features 3 flag bits each mapped directly to a channel. Should a match occur on the channel during the debug session, then the corresponding flag is set and remains set until the next time the module is armed by writing to the ARM bit. Thus the contents are retained after a debug session for evaluation purposes. These flags cannot be cleared by software, they are cleared only when arming the module. A set flag does not inhibit the setting of other flags. Once a flag is set, further comparator matches on the same channel in the same session have no affect on that flag.

### 6.3.2.8 Comparator Register Descriptions

Each comparator has a bank of registers that are visible through an 8-byte window in the DBG module register address map. Comparator A consists of 8 register bytes (3 address bus compare registers, two data bus compare registers, two data bus mask registers and a control register). Comparator B consists of four register bytes (three address bus compare registers and a control register). Comparator C consists of four register bytes (three address bus compare registers and a control register).

Each set of comparator registers can be accessed using the COMRV bits in the DBGC1 register. Unimplemented registers (e.g. Comparator B data bus and data bus masking) read as zero and cannot be written. The control register for comparator B differs from those of comparators A and C.

**Table 6-21. Comparator Register Layout**

| | | | |
|---|---|---|---|
| 0x0028 | CONTROL | Read/Write | Comparators A,B and C |
| 0x0029 | ADDRESS HIGH | Read/Write | Comparators A,B and C |
| 0x002A | ADDRESS MEDIUM | Read/Write | Comparators A,B and C |
| 0x002B | ADDRESS LOW | Read/Write | Comparators A,B and C |
| 0x002C | DATA HIGH COMPARATOR | Read/Write | Comparator A only |
| 0x002D | DATA LOW COMPARATOR | Read/Write | Comparator A only |
| 0x002E | DATA HIGH MASK | Read/Write | Comparator A only |
| 0x002F | DATA LOW MASK | Read/Write | Comparator A only |

#### 6.3.2.8.1 Debug Comparator Control Register (DBGXCTL)

The contents of this register bits 7 and 6 differ depending upon which comparator registers are visible in the 8-byte window of the DBG module register address map.

Address: 0x0028

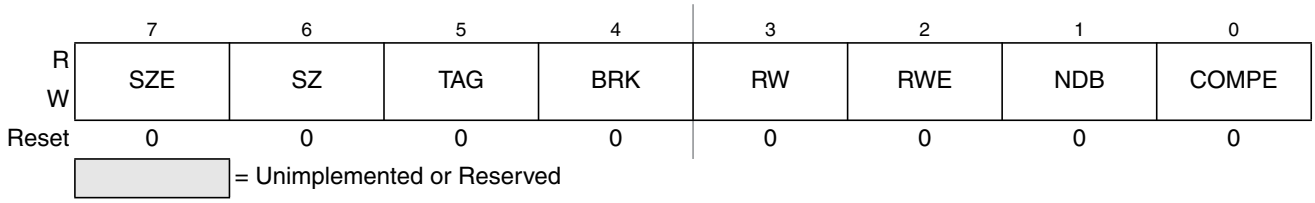



**Figure 6-13. Debug Comparator Control Register DBGACTL (Comparator A)**

Address: 0x0028

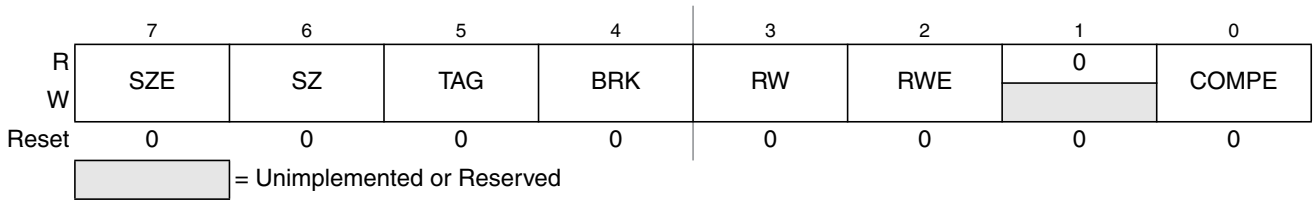



**Figure 6-14. Debug Comparator Control Register DBGBCTL (Comparator B)**

Address: 0x0028

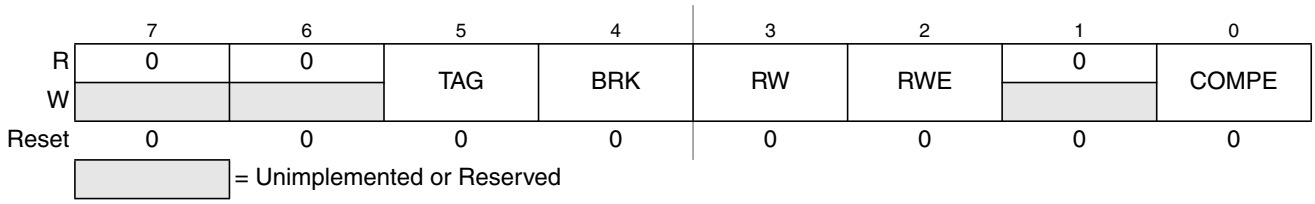



**Figure 6-15. Debug Comparator Control Register DBGCCTL (Comparator C)**

Read: DBGACTL if COMRV[1:0] = 00
DBGBCTL if COMRV[1:0] = 01
DBGCCTL if COMRV[1:0] = 10

Write: DBGACTL if COMRV[1:0] = 00 and DBG not armed
DBGBCTL if COMRV[1:0] = 01 and DBG not armed
DBGCCTL if COMRV[1:0] = 10 and DBG not armed

**Table 6-22. DBGXCTL Field Descriptions**

| Field | Description |
|---|---|
| 7<br>SZE<br>(Comparators A and B) | **Size Comparator Enable Bit** — The SZE bit controls whether access size comparison is enabled for the associated comparator. This bit is ignored if the TAG bit in the same register is set.<br>0 Word/Byte access size is not used in comparison<br>1 Word/Byte access size is used in comparison |
| 6<br>SZ<br>(Comparators A and B) | **Size Comparator Value Bit** — The SZ bit selects either word or byte access size in comparison for the associated comparator. This bit is ignored if the SZE bit is cleared or if the TAG bit in the same register is set.<br>0 Word access size is compared<br>1 Byte access size is compared |

**Table 6-22. DBGXCTL Field Descriptions (continued)**

| Field | Description |
|---|---|
| 5<br>TAG | **Tag Select**— This bit controls whether the comparator match has immediate effect, causing an immediate state sequencer transition or tag the opcode at the matched address. Tagged opcodes trigger only if they reach the execution stage of the instruction queue.<br>0 Allow state sequencer transition immediately on match<br>1 On match, tag the opcode. If the opcode is about to be executed allow a state sequencer transition |
| 4<br>BRK | **Break**— This bit controls whether a comparator match terminates a debug session immediately, independent of state sequencer state. To generate an immediate breakpoint the module breakpoints must be enabled using the DBGC1 bit DBGBRK.<br>0 The debug session termination is dependent upon the state sequencer and trigger conditions.<br>1 A match on this channel terminates the debug session immediately; breakpoints if active are generated, tracing, if active, is terminated and the module disarmed. |
| 3<br>RW | **Read/Write Comparator Value Bit** — The RW bit controls whether read or write is used in compare for the associated comparator. The RW bit is not used if RWE = 0. This bit is ignored if the TAG bit in the same register is set.<br>0 Write cycle is matched1Read cycle is matched |
| 2<br>RWE | **Read/Write Enable Bit** — The RWE bit controls whether read or write comparison is enabled for the associated comparator.This bit is ignored if the TAG bit in the same register is set<br>0 Read/Write is not used in comparison<br>1 Read/Write is used in comparison |
| 1<br>NDB<br>(Comparator A) | **Not Data Bus** — The NDB bit controls whether the match occurs when the data bus matches the comparator register value or when the data bus differs from the register value. This bit is ignored if the TAG bit in the same register is set. This bit is only available for comparator A.<br>0 Match on data bus equivalence to comparator register contents<br>1 Match on data bus difference to comparator register contents |
| 0<br>COMPE | Determines if comparator is enabled<br>0 The comparator is not enabled<br>1 The comparator is enabled |

Table 6-23 shows the effect for RWE and RW on the comparison conditions. These bits are ignored if the corresponding TAG bit is set since the match occurs based on the tagged opcode reaching the execution stage of the instruction queue.

**Table 6-23. Read or Write Comparison Logic Table**

| RWE Bit | RW Bit | RW Signal | Comment |
|---|---|---|---|
| 0 | x | 0 | RW not used in comparison |
| 0 | x | 1 | RW not used in comparison |
| 1 | 0 | 0 | Write data bus |
| 1 | 0 | 1 | No match |
| 1 | 1 | 0 | No match |
| 1 | 1 | 1 | Read data bus |

#### 6.3.2.8.2 Debug Comparator Address High Register (DBGXAH)

Address: 0x0029

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | Bit 17 | Bit 16 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 6-16. Debug Comparator Address High Register (DBGXAH)**

The DBGC1_COMRV bits determine which comparator address registers are visible in the 8-byte window from 0x0028 to 0x002F as shown in Section Table 6-24., “Comparator Address Register Visibility

**Table 6-24. Comparator Address Register Visibility**

| COMRV | Visible Comparator |
|---|---|
| 00 | DBGAAH, DBGAAM, DBGAAL |
| 01 | DBGBAH, DBGBAM, DBGBAL |
| 10 | DBGCAH, DBGCAM, DBGCAL |
| 11 | None |

Read: Anytime. See Table 6-24 for visible register encoding.

Write: If DBG not armed. See Table 6-24 for visible register encoding.

**Table 6-25. DBGXAH Field Descriptions**

| Field | Description |
|---|---|
| 1–0<br>Bit[17:16] | **Comparator Address High Compare Bits** — The Comparator address high compare bits control whether the selected comparator compares the address bus bits [17:16] to a logic one or logic zero.<br>0 Compare corresponding address bit to a logic zero<br>1 Compare corresponding address bit to a logic one |

#### 6.3.2.8.3 Debug Comparator Address Mid Register (DBGXAM)

Address: 0x002A

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 6-17. Debug Comparator Address Mid Register (DBGXAM)**

Read: Anytime. See Table 6-24 for visible register encoding.

Write: If DBG not armed. See Table 6-24 for visible register encoding.

**Table 6-26. DBGXAM Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>Bit[15:8] | **Comparator Address Mid Compare Bits** — The Comparator address mid compare bits control whether the selected comparator compares the address bus bits [15:8] to a logic one or logic zero.<br>0 Compare corresponding address bit to a logic zero<br>1 Compare corresponding address bit to a logic one |

### 6.3.2.8.4 Debug Comparator Address Low Register (DBGXAL)

Address: 0x002B

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 6-18. Debug Comparator Address Low Register (DBGXAL)**

Read: Anytime. See Table 6-24 for visible register encoding.

Write: If DBG not armed. See Table 6-24 for visible register encoding.

**Table 6-27. DBGXAL Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>Bits[7:0] | **Comparator Address Low Compare Bits** — The Comparator address low compare bits control whether the selected comparator compares the address bus bits [7:0] to a logic one or logic zero.<br>0 Compare corresponding address bit to a logic zero<br>1 Compare corresponding address bit to a logic one |

### 6.3.2.8.5 Debug Comparator Data High Register (DBGADH)

Address: 0x002C

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 6-19. Debug Comparator Data High Register (DBGADH)**

Read: If COMRV[1:0] = 00

Write: If COMRV[1:0] = 00 and DBG not armed.

**Table 6-28. DBGADH Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>Bits[15:8] | **Comparator Data High Compare Bits**— The Comparator data high compare bits control whether the selected comparator compares the data bus bits [15:8] to a logic one or logic zero. The comparator data compare bits are only used in comparison if the corresponding data mask bit is logic 1. This register is available only for comparator A. Data bus comparisons are only performed if the TAG bit in DBGACTL is clear.<br>0 Compare corresponding data bit to a logic zero<br>1 Compare corresponding data bit to a logic one |

### 6.3.2.8.6 Debug Comparator Data Low Register (DBGADL)

Address: 0x002D

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 6-20. Debug Comparator Data Low Register (DBGADL)**

Read: If COMRV[1:0] = 00

Write: If COMRV[1:0] = 00 and DBG not armed.

**Table 6-29. DBGADL Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>Bits[7:0] | **Comparator Data Low Compare Bits** — The Comparator data low compare bits control whether the selected comparator compares the data bus bits [7:0] to a logic one or logic zero. The comparator data compare bits are only used in comparison if the corresponding data mask bit is logic 1. This register is available only for comparator A. Data bus comparisons are only performed if the TAG bit in DBGACTL is clear<br>0 Compare corresponding data bit to a logic zero<br>1 Compare corresponding data bit to a logic one |

### 6.3.2.8.7 Debug Comparator Data High Mask Register (DBGADHM)

Address: 0x002E

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 6-21. Debug Comparator Data High Mask Register (DBGADHM)**

Read: If COMRV[1:0] = 00

Write: If COMRV[1:0] = 00 and DBG not armed.

**Table 6-30. DBGADHM Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>Bits[15:8] | **Comparator Data High Mask Bits** — The Comparator data high mask bits control whether the selected comparator compares the data bus bits [15:8] to the corresponding comparator data compare bits. Data bus comparisons are only performed if the TAG bit in DBGACTL is clear<br>0 Do not compare corresponding data bit Any value of corresponding data bit allows match.<br>1 Compare corresponding data bit |

#### 6.3.2.8.8 Debug Comparator Data Low Mask Register (DBGADLM)

Address: 0x002F

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 6-22. Debug Comparator Data Low Mask Register (DBGADLM)**

Read: If COMRV[1:0] = 00

Write: If COMRV[1:0] = 00 and DBG not armed.

**Table 6-31. DBGADLM Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>Bits[7:0] | **Comparator Data Low Mask Bits** — The Comparator data low mask bits control whether the selected comparator compares the data bus bits [7:0] to the corresponding comparator data compare bits. Data bus comparisons are only performed if the TAG bit in DBGACTL is clear<br>0 Do not compare corresponding data bit. Any value of corresponding data bit allows match<br>1 Compare corresponding data bit |

## 6.4 Functional Description

This section provides a complete functional description of the DBG module. If the part is in secure mode, the DBG module can generate breakpoints but tracing is not possible.

### 6.4.1 S12DBGV2 Operation

Arming the DBG module by setting ARM in DBGC1 allows triggering the state sequencer, storing of data in the trace buffer and generation of breakpoints to the CPU. The DBG module is made up of four main blocks, the comparators, control logic, the state sequencer, and the trace buffer.

The comparators monitor the bus activity of the CPU. All comparators can be configured to monitor address bus activity. Comparator A can also be configured to monitor databus activity and mask out individual data bus bits during a compare. Comparators can be configured to use R/W and word/byte access qualification in the comparison. A match with a comparator register value can initiate a state sequencer transition to another state (see Figure 6-24). Either forced or tagged matches are possible. Using

a forced match, a state sequencer transition can occur immediately on a successful match of system busses and comparator registers. Whilst tagging, at a comparator match, the instruction opcode is tagged and only if the instruction reaches the execution stage of the instruction queue can a state sequencer transition occur. In the case of a transition to Final State, bus tracing is triggered and/or a breakpoint can be generated.

A state sequencer transition to final state (with associated breakpoint, if enabled) can be initiated by writing to the TRIG bit in the DBGC1 control register.

The trace buffer is visible through a 2-byte window in the register address map and must be read out using standard 16-bit word reads.

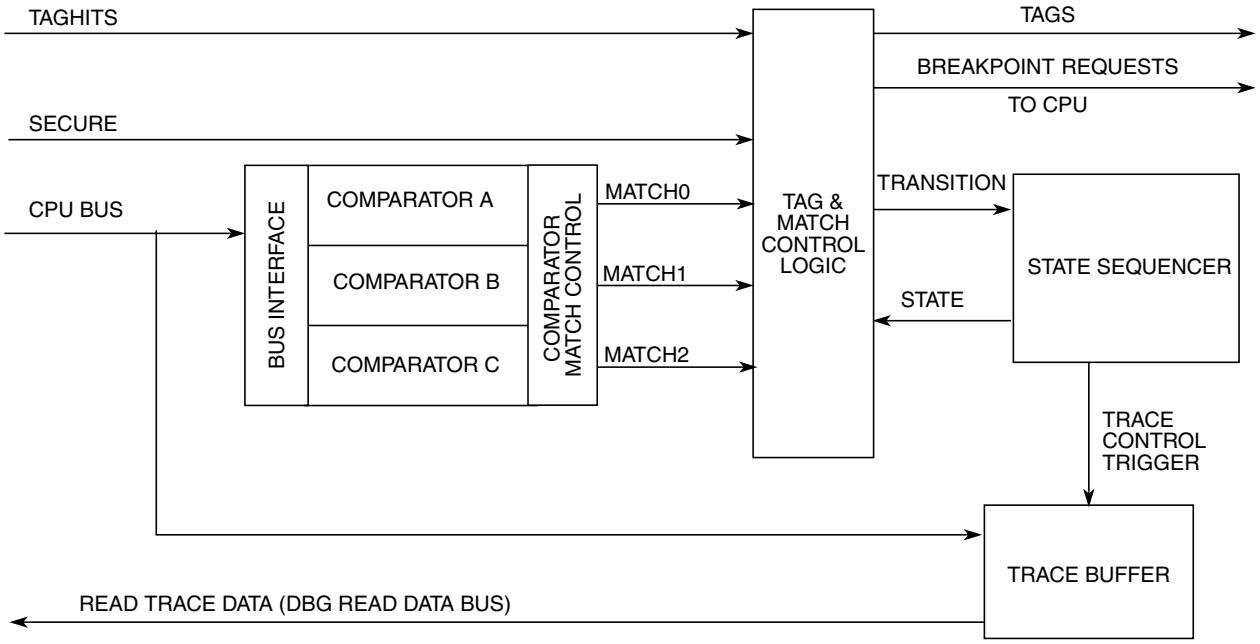



**Figure 6-23. DBG Overview**

## 6.4.2 Comparator Modes

The DBG contains three comparators, A, B and C. Each comparator compares the system address bus with the address stored in DBGXAH, DBGXAM, and DBGXAL. Furthermore, comparator A also compares the data buses to the data stored in DBGADH, DBGADL and allows masking of individual data bus bits.

All comparators are disabled in BDM and during BDM accesses.

The comparator match control logic (see Figure 6-23) configures comparators to monitor the buses for an exact address or an address range, whereby either an access inside or outside the specified range generates a match condition. The comparator configuration is controlled by the control register contents and the range control by the DBGC2 contents.

A match can initiate a transition to another state sequencer state (see Section 6.4.4, “State Sequence Control”). The comparator control register also allows the type of access to be included in the comparison through the use of the RWE, RW, SZE, and SZ bits. The RWE bit controls whether read or write comparison is enabled for the associated comparator and the RW bit selects either a read or write access

for a valid match. Similarly the SZE and SZ bits allow the size of access (word or byte) to be considered in the compare. Only comparators A and B feature SZE and SZ.

The TAG bit in each comparator control register is used to determine the match condition. By setting TAG, the comparator qualifies a match with the output of opcode tracking logic and a state sequencer transition occurs when the tagged instruction reaches the CPU execution stage. Whilst tagging the RW, RWE, SZE, and SZ bits and the comparator data registers are ignored; the comparator address register must be loaded with the exact opcode address.

If the TAG bit is clear (forced type match) a comparator match is generated when the selected address appears on the system address bus. If the selected address is an opcode address, the match is generated when the opcode is fetched from the memory, which precedes the instruction execution by an indefinite number of cycles due to instruction pipelining. For a comparator match of an opcode at an odd address when TAG = 0, the corresponding even address must be contained in the comparator register. Thus for an opcode at odd address (n), the comparator register must contain address (n–1).

Once a successful comparator match has occurred, the condition that caused the original match is not verified again on subsequent matches. Thus if a particular data value is verified at a given address, this address may not still contain that data value when a subsequent match occurs.

Match[0, 1, 2] map directly to Comparators [A, B, C] respectively, except in range modes (see Section 6.3.2.4, “Debug Control Register2 (DBGC2)). Comparator channel priority rules are described in the priority section (Section 6.4.3.4, “Channel Priorities).

### 6.4.2.1 Single Address Comparator Match

With range comparisons disabled, the match condition is an exact equivalence of address bus with the value stored in the comparator address registers. Further qualification of the type of access (R/W, word/byte) and databus contents is possible, depending on comparator channel.

#### 6.4.2.1.1 Comparator C

Comparator C offers only address and direction (R/W) comparison. The exact address is compared, thus with the comparator address register loaded with address (n) a word access of address (n–1) also accesses (n) but does not cause a match.

**Table 6-32. Comparator C Access Considerations**

| Condition For Valid Match | Comp C Address | RWE | RW | Examples |
|---|---|---|---|---|
| Read and write accesses of ADDR[n] | ADDR[n][1] | 0 | X | LDAA ADDR[n]<br>STAA #$BYTE ADDR[n] |
| Write accesses of ADDR[n] | ADDR[n] | 1 | 0 | STAA #$BYTE ADDR[n] |
| Read accesses of ADDR[n] | ADDR[n] | 1 | 1 | LDAA #$BYTE ADDR[n] |

[1] A word access of ADDR[n-1] also accesses ADDR[n] but does not generate a match. The comparator address register must contain the exact address from the code.

#### 6.4.2.1.2 Comparator B

Comparator B offers address, direction (R/W) and access size (word/byte) comparison. If the SZE bit is set the access size (word or byte) is compared with the SZ bit value such that only the specified size of access causes a match. Thus if configured for a byte access of a particular address, a word access covering the same address does not lead to match.

Assuming the access direction is not qualified (RWE=0), for simplicity, the size access considerations are shown in Table 6-33.

**Table 6-33. Comparator B Access Size Considerations**

| Condition For Valid Match | Comp B Address | RWE | SZE | SZ8 | Examples |
|---|---|---|---|---|---|
| Word and byte accesses of ADDR[n] | ADDR[n][1] | 0 | 0 | X | MOVB #$BYTE ADDR[n]<br>MOVW #$WORD ADDR[n] |
| Word accesses of ADDR[n] only | ADDR[n] | 0 | 1 | 0 | MOVW #$WORD ADDR[n]<br>LDD ADDR[n] |
| Byte accesses of ADDR[n] only | ADDR[n] | 0 | 1 | 1 | MOVB #$BYTE ADDR[n]<br>LDAB ADDR[n] |

1 A word access of ADDR[n-1] also accesses ADDR[n] but does not generate a match. The comparator address register must contain the exact address from the code.

Access direction can also be used to qualify a match for Comparator B in the same way as described for Comparator C in Table 6-32.

#### 6.4.2.1.3 Comparator A

Comparator A offers address, direction (R/W), access size (word/byte) and data bus comparison.

Table 6-34 lists access considerations with data bus comparison. On word accesses the data byte of the lower address is mapped to DBGADH. Access direction can also be used to qualify a match for Comparator A in the same way as described for Comparator C in Table 6-32.

**Table 6-34. Comparator A Matches When Accessing ADDR[n]**

| SZE | SZ | DBGADHM, DBGADLM | Access<br>DH=DBGADH, DL=DBGADL | Comment |
|---|---|---|---|---|
| 0 | X | $0000 | Byte<br>Word | No databus comparison |
| 0 | X | $FF00 | Byte, data(ADDR[n])=DH<br>Word, data(ADDR[n])=DH, data(ADDR[n+1])=X | Match data( ADDR[n]) |
| 0 | X | $00FF | Word, data(ADDR[n])=X, data(ADDR[n+1])=DL | Match data( ADDR[n+1]) |
| 0 | X | $00FF | Byte, data(ADDR[n])=X, data(ADDR[n+1])=DL | Possible unintended match |
| 0 | X | $FFFF | Word, data(ADDR[n])=DH, data(ADDR[n+1])=DL | Match data( ADDR[n], ADDR[n+1]) |
| 0 | X | $FFFF | Byte, data(ADDR[n])=DH, data(ADDR[n+1])=DL | Possible unintended match |
| 1 | 0 | $0000 | Word | No databus comparison |
| 1 | 0 | $00FF | Word, data(ADDR[n])=X, data(ADDR[n+1])=DL | Match only data at ADDR[n+1] |
| 1 | 0 | $FF00 | Word, data(ADDR[n])=DH, data(ADDR[n+1])=X | Match only data at ADDR[n] |
| 1 | 0 | $FFFF | Word, data(ADDR[n])=DH, data(ADDR[n+1])=DL | Match data at ADDR[n] & ADDR[n+1] |

| SZE | SZ | DBGADHM, DBGADLM | Access DH=DBGADH, DL=DBGADL | Comment |
|---|---|---|---|---|
| 1 | 1 | $0000 | Byte | No databus comparison |
| 1 | 1 | $FF00 | Byte, data(ADDR[n])=DH | Match data at ADDR[n] |

#### 6.4.2.1.4 Comparator A Data Bus Comparison NDB Dependency

Comparator A features an NDB control bit, which allows data bus comparators to be configured to either trigger on equivalence or trigger on difference. This allows monitoring of a difference in the contents of an address location from an expected value.

When matching on an equivalence (NDB=0), each individual data bus bit position can be masked out by clearing the corresponding mask bit (DBGADHM/DBGADLM) so that it is ignored in the comparison. A match occurs when all data bus bits with corresponding mask bits set are equivalent. If all mask register bits are clear, then a match is based on the address bus only, the data bus is ignored.

When matching on a difference, mask bits can be cleared to ignore bit positions. A match occurs when any data bus bit with corresponding mask bit set is different. Clearing all mask bits, causes all bits to be ignored and prevents a match because no difference can be detected. In this case address bus equivalence does not cause a match.

**Table 6-35. NDB and MASK bit dependency**

| NDB | DBGADHM[n] / DBGADLM[n] | Comment |
|---|---|---|
| 0 | 0 | Do not compare data bus bit. |
| 0 | 1 | Compare data bus bit. Match on equivalence. |
| 1 | 0 | Do not compare data bus bit. |
| 1 | 1 | Compare data bus bit. Match on difference. |

### 6.4.2.2 Range Comparisons

Using the AB comparator pair for a range comparison, the data bus can also be used for qualification by using the comparator A data registers. Furthermore the DBGACTL RW and RWE bits can be used to qualify the range comparison on either a read or a write access. The corresponding DBGBCTL bits are ignored. The SZE and SZ control bits are ignored in range mode. The comparator A TAG bit is used to tag range comparisons. The comparator B TAG bit is ignored in range modes. In order for a range comparison using comparators A and B, both COMPEA and COMPEB must be set; to disable range comparisons both must be cleared. The comparator A BRK bit is used to for the AB range, the comparator B BRK bit is ignored in range mode.

When configured for range comparisons and tagging, the ranges are accurate only to word boundaries.

#### 6.4.2.2.1 Inside Range (CompA_Addr ≤ address ≤ CompB_Addr)

In the Inside Range comparator mode, comparator pair A and B can be configured for range comparisons. This configuration depends upon the control register (DBGC2). The match condition requires that a valid

match for both comparators happens on the same bus cycle. A match condition on only one comparator is not valid. An aligned word access which straddles the range boundary is valid only if the aligned address is inside the range.

#### 6.4.2.2.2 Outside Range (address < CompA_Addr or address > CompB_Addr)

In the Outside Range comparator mode, comparator pair A and B can be configured for range comparisons. A single match condition on either of the comparators is recognized as valid. An aligned word access which straddles the range boundary is valid only if the aligned address is outside the range.

Outside range mode in combination with tagging can be used to detect if the opcode fetches are from an unexpected range. In forced match mode the outside range match would typically be activated at any interrupt vector fetch or register access. This can be avoided by setting the upper range limit to $3FFFF or lower range limit to $00000 respectively.

### 6.4.3 Match Modes (Forced or Tagged)

Match modes are used as qualifiers for a state sequencer change of state. The Comparator control register TAG bits select the match mode. The modes are described in the following sections.

#### 6.4.3.1 Forced Match

When configured for forced matching, a comparator channel match can immediately initiate a transition to the next state sequencer state whereby the corresponding flags in DBGSR are set. The state control register for the current state determines the next state. Forced matches are typically generated 2-3 bus cycles after the final matching address bus cycle, independent of comparator RWE/RW settings. Furthermore since opcode fetches occur several cycles before the opcode execution a forced match of an opcode address typically precedes a tagged match at the same address.

#### 6.4.3.2 Tagged Match

If a CPU taghit occurs a transition to another state sequencer state is initiated and the corresponding DBGSR flags are set. For a comparator related taghit to occur, the DBG must first attach tags to instructions as they are fetched from memory. When the tagged instruction reaches the execution stage of the instruction queue a taghit is generated by the CPU. This can initiate a state sequencer transition.

#### 6.4.3.3 Immediate Trigger

Independent of comparator matches it is possible to initiate a tracing session and/or breakpoint by writing to the TRIG bit in DBGC1. If configured for begin aligned tracing, this triggers the state sequencer into the Final State, if configured for end alignment, setting the TRIG bit disarms the module, ending the session and issues a forced breakpoint request to the CPU.

It is possible to set both TRIG and ARM simultaneously to generate an immediate trigger, independent of the current state of ARM.

#### 6.4.3.4 Channel Priorities

In case of simultaneous matches the priority is resolved according to Table 6-36. The lower priority is suppressed. It is thus possible to miss a lower priority match if it occurs simultaneously with a higher priority. The priorities described in Table 6-36 dictate that in the case of simultaneous matches, the match pointing to final state has highest priority followed by the lower channel number (0,1,2).

**Table 6-36. Channel Priorities**

| Priority | Source | Action |
|---|---|---|
| **Highest** | **TRIG** | Enter Final State |
| | **Channel pointing to Final State** | Transition to next state as defined by state control registers |
| | **Match0 (force or tag hit)** | Transition to next state as defined by state control registers |
| | **Match1 (force or tag hit)** | Transition to next state as defined by state control registers |
| **Lowest** | **Match2 (force or tag hit)** | Transition to next state as defined by state control registers |

### 6.4.4 State Sequence Control

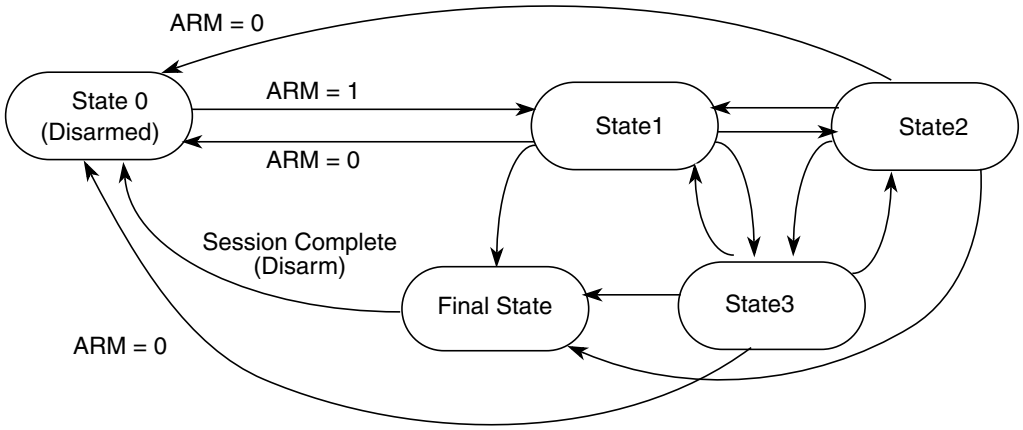



**Figure 6-24. State Sequencer Diagram**

The state sequencer allows a defined sequence of events to provide a trigger point for tracing of data in the trace buffer. Once the DBG module has been armed by setting the ARM bit in the DBGC1 register, then state1 of the state sequencer is entered. Further transitions between the states are then controlled by the state control registers and channel matches. From Final State the only permitted transition is back to the disarmed state0. Transition between any of the states 1 to 3 is not restricted. Each transition updates the SSF[2:0] flags in DBGSR accordingly to indicate the current state.

Alternatively writing to the TRIG bit in DBGSC1, provides an immediate trigger independent of comparator matches.

Independent of the state sequencer, each comparator channel can be individually configured to generate an immediate breakpoint when a match occurs through the use of the BRK bits in the DBGxCTL registers. Thus it is possible to generate an immediate breakpoint on selected channels, whilst a state sequencer transition can be initiated by a match on other channels. If a debug session is ended by a match on a channel the state sequencer transitions through Final State for a clock cycle to state0. This is independent of tracing

and breakpoint activity, thus with tracing and breakpoints disabled, the state sequencer enters state0 and the debug module is disarmed.

#### 6.4.4.1 Final State

On entering Final State a trigger may be issued to the trace buffer according to the trace alignment control as defined by the TALIGN bit (see Section 6.3.2.3, “Debug Trace Control Register (DBGTCR)”). If the TSOURCE bit in DBGTCR is clear then the trace buffer is disabled and the transition to Final State can only generate a breakpoint request. In this case or upon completion of a tracing session when tracing is enabled, the ARM bit in the DBGC1 register is cleared, returning the module to the disarmed state0. If tracing is enabled a breakpoint request can occur at the end of the tracing session. If neither tracing nor breakpoints are enabled then when the final state is reached it returns automatically to state0 and the debug module is disarmed.

### 6.4.5 Trace Buffer Operation

The trace buffer is a 64 lines deep by 20-bits wide RAM array. The DBG module stores trace information in the RAM array in a circular buffer format. The system accesses the RAM array through a register window (DBGTBH:DBGTBL) using 16-bit wide word accesses. After each complete 20-bit trace buffer line is read, an internal pointer into the RAM increments so that the next read receives fresh information. Data is stored in the format shown in Table 6-37 and Table 6-40. After each store the counter register DBGCNT is incremented. Tracing of CPU activity is disabled when the BDM is active. Reading the trace buffer whilst the DBG is armed returns invalid data and the trace buffer pointer is not incremented.

#### 6.4.5.1 Trace Trigger Alignment

Using the TALIGN bit (see Section 6.3.2.3, “Debug Trace Control Register (DBGTCR)) it is possible to align the trigger with the end or the beginning of a tracing session.

If end alignment is selected, tracing begins when the ARM bit in DBGC1 is set and State1 is entered; the transition to Final State signals the end of the tracing session. Tracing with Begin-Trigger starts at the opcode of the trigger. Using end alignment or when the tracing is initiated by writing to the TRIG bit whilst configured for begin alignment, tracing starts in the second cycle after the DBGC1 write cycle.

##### 6.4.5.1.1 Storing with Begin Trigger Alignment

Storing with begin alignment, data is not stored in the Trace Buffer until the Final State is entered. Once the trigger condition is met the DBG module remains armed until 64 lines 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 is stored in the Trace Buffer. Using begin alignment together with tagging, if the tagged instruction is about to be executed then the trace is started. Upon completion of the tracing session the breakpoint is generated, thus the breakpoint does not occur at the tagged instruction boundary.

##### 6.4.5.1.2 Storing with End Trigger Alignment

Storing with end alignment, data is stored in the Trace Buffer until the Final State is entered, at which point the DBG module becomes disarmed and no more data is stored. If the trigger is at the address of a change

of flow instruction, the trigger event is not stored in the Trace Buffer. If all trace buffer lines have been used before a trigger event occurrs then the trace continues at the first line, overwriting the oldest entries.

### 6.4.5.2 Trace Modes

Four trace modes are available. The mode is selected using the TRCMOD bits in the DBGTCR register. Tracing is enabled using the TSOURCE bit in the DBGTCR register. The modes are described in the following subsections.

#### 6.4.5.2.1 Normal Mode

In Normal Mode, change of flow (COF) program counter (PC) addresses are stored.

COF addresses are defined as follows:

- Source address of taken conditional branches (long, short, bit-conditional, and loop primitives)
- Destination address of indexed JMP, JSR, and CALL instruction
- Destination address of RTI, RTS, and RTC instructions
- Vector address of interrupts, except for BDM vectors

LBRA, BRA, BSR, BGND as well as non-indexed JMP, JSR, and CALL instructions are not classified as change of flow and are not stored in the trace buffer.

Stored information includes the full 18-bit address bus and information bits, which contains a source/destination bit to indicate whether the stored address was a source address or destination address.

**NOTE**

When a COF instruction with destination address is executed, the destination address is stored to the trace buffer on instruction completion, indicating the COF has taken place. If an interrupt occurs simultaneously then the next instruction carried out is actually from the interrupt service routine. The instruction at the destination address of the original program flow gets executed after the interrupt service routine.

In the following example an IRQ interrupt occurs during execution of the indexed JMP at address MARK1. The BRN at the destination (SUB_1) is not executed until after the IRQ service routine but the destination address is entered into the trace buffer to indicate that the indexed JMP COF has taken place.

```
        LDX     #SUB_1
MARK1   JMP     0,X                     ; IRQ interrupt occurs during execution of this
MARK2   NOP                             ;

SUB_1   BRN     *                       ; JMP Destination address TRACE BUFFER ENTRY 1
                                        ; RTI Destination address TRACE BUFFER ENTRY 3
        NOP                             ;
ADDR1   DBNE    A,PART5                 ; Source address TRACE BUFFER ENTRY 4

IRQ_ISR LDAB    #$F0                    ; IRQ Vector $FFF2 = TRACE BUFFER ENTRY 2
        STAB    VAR_C1
```

```
        RTI                             ;
```

The execution flow taking into account the IRQ is as follows

```
        LDX     #SUB_1
MARK1   JMP     0,X                     ;
IRQ_ISR LDAB    #$F0                    ;
        STAB    VAR_C1
        RTI                             ;
SUB_1   BRN     *
        NOP                             ;
ADDR1   DBNE    A,PART5                 ;
```

#### 6.4.5.2.2 Loop1 Mode

Loop1 Mode, similarly to Normal Mode also stores only COF address information to the trace buffer, it however allows the filtering out of redundant information.

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 a background register. This prevents consecutive duplicate address entries in the Trace Buffer resulting from repeated branches.

Loop1 Mode only inhibits consecutive duplicate source address entries that would typically be stored in most tight looping constructs. It does not inhibit repeated entries of destination addresses or vector addresses, since repeated entries of these would most likely indicate a bug in the user's code that the DBG module is designed to help find.

#### 6.4.5.2.3 Detail Mode

In Detail Mode, address and data for all memory and register accesses is stored in the 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 the code is in error. This mode also features information bit storage to the trace buffer, for each address byte storage. The information bits indicate the size of access (word or byte) and the type of access (read or write).

When tracing in Detail Mode, all cycles are traced except those when the CPU is either in a free or opcode fetch cycle.

#### 6.4.5.2.4 Compressed Pure PC Mode

In Compressed Pure PC Mode, the PC addresses of all executed opcodes, including illegal opcodes are stored. A compressed storage format is used to increase the effective depth of the trace buffer. This is achieved by storing the lower order bits each time and using 2 information bits to indicate if a 64 byte boundary has been crossed, in which case the full PC is stored.

Each Trace Buffer row consists of 2 information bits and 18 PC address bits

**NOTE:**

When tracing is terminated using forced breakpoints, latency in breakpoint generation means that opcodes following the opcode causing the breakpoint can be stored to the trace buffer. The number of opcodes is dependent on program flow. This can be avoided by using tagged breakpoints.

### 6.4.5.3 Trace Buffer Organization (Normal, Loop1, Detail modes)

ADRH, ADRM, ADRL denote address high, middle and low byte respectively. The numerical suffix refers to the tracing count. The information format for Loop1 and Normal modes is identical. In Detail mode, the address and data for each entry are stored on consecutive lines, thus the maximum number of entries is 32. In this case DBGCNT bits are incremented twice, once for the address line and once for the data line, on each trace buffer entry. In Detail mode CINF comprises of R/W and size access information (CRW and CSZ respectively).

Single byte data accesses in Detail Mode are always stored to the low byte of the trace buffer (DATAL) and the high byte is cleared. When tracing word accesses, the byte at the lower address is always stored to trace buffer byte1 and the byte at the higher address is stored to byte0.

**Table 6-37. Trace Buffer Organization (Normal,Loop1,Detail modes)**

| Mode | Entry Number | 4-bits | 8-bits | 8-bits |
|---|---|---|---|---|
| | | Field 2 | Field 1 | Field 0 |
| Detail Mode | Entry 1 | CINF1,ADRH1 | ADRM1 | ADRL1 |
| | | 0 | DATAH1 | DATAL1 |
| | Entry 2 | CINF2,ADRH2 | ADRM2 | ADRL2 |
| | | 0 | DATAH2 | DATAL2 |
| Normal/Loop1 Modes | Entry 1 | PCH1 | PCM1 | PCL1 |
| | Entry 2 | PCH2 | PCM2 | PCL2 |

#### 6.4.5.3.1 Information Bit Organization

The format of the bits is dependent upon the active trace mode as described below.

**Field2 Bits in Detail Mode**

| Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|
| CSZ | CRW | ADDR[17] | ADDR[16] |

**Figure 6-25. Field2 Bits in Detail Mode**

In Detail Mode the CSZ and CRW bits indicate the type of access being made by the CPU.

**Table 6-38. Field Descriptions**

| Bit | Description |
|---|---|
| 3<br>CSZ | **Access Type Indicator**— This bit indicates if the access was a byte or word size when tracing in Detail Mode<br>0 Word Access<br>1 Byte Access |
| 2<br>CRW | **Read Write Indicator** — This bit indicates if the corresponding stored address corresponds to a read or write access when tracing in Detail Mode.<br>0 Write Access<br>1 Read Access |
| 1<br>ADDR[17] | **Address Bus bit 17**— Corresponds to system address bus bit 17. |
| 0<br>ADDR[16] | **Address Bus bit 16**— Corresponds to system address bus bit 16. |

### Field2 Bits in Normal and Loop1 Modes

| Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|
| CSD | CVA | PC17 | PC16 |

**Figure 6-26. Information Bits PCH**

**Table 6-39. PCH Field Descriptions**

| Bit | Description |
|---|---|
| 3<br>CSD | **Source Destination Indicator** — In Normal and Loop1 mode this bit indicates if the corresponding stored address is a source or destination address. This bit has no meaning in Compressed Pure PC mode.<br>0 Source Address<br>1 Destination Address |
| 2<br>CVA | **Vector Indicator** — In Normal and Loop1 mode this bit indicates if the corresponding stored address is a vector address. Vector addresses are destination addresses, thus if CVA is set, then the corresponding CSD is also set. This bit has no meaning in Compressed Pure PC mode.<br>0 Non-Vector Destination Address<br>1 Vector Destination Address |
| 1<br>PC17 | **Program Counter bit 17**— In Normal and Loop1 mode this bit corresponds to program counter bit 17. |
| 0<br>PC16 | **Program Counter bit 16**— In Normal and Loop1 mode this bit corresponds to program counter bit 16. |

## 6.4.5.4 Trace Buffer Organization (Compressed Pure PC mode)

**Table 6-40. Trace Buffer Organization Example (Compressed PurePC mode)**

| Mode | Line Number | 2-bits | 6-bits | 6-bits | 6-bits |
|---|---|---|---|---|---|
| | | Field 3 | Field 2 | Field 1 | Field 0 |

| | | | | | |
|---|---|---|---|---|---|
| Compressed Pure PC Mode | Line 1 | 00 | PC1 (Initial 18-bit PC Base Address) | | |
| | Line 2 | 11 | PC4 | PC3 | PC2 |
| | Line 3 | 01 | 0 | 0 | PC5 |
| | Line 4 | 00 | PC6 (New 18-bit PC Base Address) | | |
| | Line 5 | 10 | 0 | PC8 | PC7 |
| | Line 6 | 00 | PC9 (New 18-bit PC Base Address) | | |

**NOTE**

Configured for end aligned triggering in compressed PurePC mode, then after rollover it is possible that the oldest base address is overwritten. In this case all entries between the pointer and the next base address have lost their base address following rollover. For example in Table 6-40 if one line of rollover has occurred, Line 1, PC1, is overwritten with a new entry. Thus the entries on Lines 2 and 3 have lost their base address. For reconstruction of program flow the first base address following the pointer must be used, in the example, Line 4. The pointer points to the oldest entry, Line 2.

### Field3 Bits in Compressed Pure PC Modes

**Table 6-41. Compressed Pure PC Mode Field 3 Information Bit Encoding**

| INF1 | INF0 | TRACE BUFFER ROW CONTENT |
|---|---|---|
| 0 | 0 | Base PC address TB[17:0] contains a full PC[17:0] value |
| 0 | 1 | Trace Buffer[5:0] contain incremental PC relative to base address zero value |
| 1 | 0 | Trace Buffer[11:0] contain next 2 incremental PCs relative to base address zero value |
| 1 | 1 | Trace Buffer[17:0] contain next 3 incremental PCs relative to base address zero value |

Each time that PC[17:6] differs from the previous base PC[17:6], then a new base address is stored. The base address zero value is the lowest address in the 64 address range

The first line of the trace buffer always gets a base PC address, this applies also on rollover.

## 6.4.5.5 Reading Data from Trace Buffer

The data stored in the Trace Buffer can be read provided the DBG module is not armed, is configured for tracing (TSOURCE bit is set) and the system not secured. When the ARM bit is written to 1 the trace buffer is locked to prevent reading. The trace buffer can only be unlocked for reading by a single aligned word write to DBGTB when the module is disarmed.

The Trace Buffer can only be read through the DBGTB register using aligned word reads, any byte or misaligned reads return 0 and do not cause the trace buffer pointer to increment to the next trace buffer address. The Trace Buffer data is read out first-in first-out. By reading CNT in DBGCNT the number of valid lines can be determined. DBGCNT does not decrement as data is read.

Whilst reading an internal pointer is used to determine the next line to be read. After a tracing session, the pointer points to the oldest data entry, thus if no rollover has occurred, the pointer points to line0, otherwise it points to the line with the oldest entry. In compressed Pure PC mode on rollover the line with the oldest

data entry may also contain newer data entries in fields 0 and 1. Thus if rollover is indicated by the TBF bit, the line status must be decoded using the INF bits in field3 of that line. If both INF bits are clear then the line contains only entries from before the last rollover.
If INF0=1 then field 0 contains post rollover data but fields 1 and 2 contain pre rollover data.
If INF1=1 then fields 0 and 1 contain post rollover data but field 2 contains pre rollover data.

The pointer is initialized by each aligned write to DBGTBH to point to the oldest data again. This enables an interrupted trace buffer read sequence to be easily restarted from the oldest data entry.

The least significant word of line is read out first. This corresponds to the fields 1 and 0 of Table 6-37. The next word read returns field 2 in the least significant bits [3:0] and “0” for bits [15:4].

Reading the Trace Buffer while the DBG module is armed returns invalid data and no shifting of the RAM pointer occurs.

#### 6.4.5.6 Trace Buffer Reset State

The Trace Buffer contents and DBGCNT bits are not initialized by a system reset. Thus should a system reset occur, the trace session information from immediately before the reset occurred can be read out and the number of valid lines in the trace buffer is indicated by DBGCNT. The internal pointer to the current trace buffer address is initialized by unlocking the trace buffer and points to the oldest valid data even if a reset occurred during the tracing session. To read the trace buffer after a reset, TSOURCE must be set, otherwise the trace buffer reads as all zeroes. Generally debugging occurrences of system resets is best handled using end trigger alignment since the reset may occur before the trace trigger, which in the begin trigger alignment case means no information would be stored in the trace buffer.

The Trace Buffer contents and DBGCNT bits are undefined following a POR.

**NOTE**

An external pin RESET that occurs simultaneous to a trace buffer entry can, in very seldom cases, lead to either that entry being corrupted or the first entry of the session being corrupted. In such cases the other contents of the trace buffer still contain valid tracing information. The case occurs when the reset assertion coincides with the trace buffer entry clock edge.

### 6.4.6 Tagging

A tag follows program information as it advances through the instruction queue. When a tagged instruction reaches the head of the queue a tag hit occurs and can initiate a state sequencer transition.

Each comparator control register features a TAG bit, which controls whether the comparator match causes a state sequencer transition immediately or tags the opcode at the matched address. If a comparator is enabled for tagged comparisons, the address stored in the comparator match address registers must be an opcode address.

Using Begin trigger together with tagging, if the tagged instruction is about to be executed then the transition to the next state sequencer state occurs. If the transition is to the Final State, tracing is started. Only upon completion of the tracing session can a breakpoint be generated. Using End alignment, when

the tagged instruction is about to be executed and the next transition is to Final State then a breakpoint is generated immediately, before the tagged instruction is carried out.

R/W monitoring, access size (SZ) monitoring and data bus monitoring are not useful if tagging is selected, since the tag is attached to the opcode at the matched address and is not dependent on the data bus nor on the type of access. Thus these bits are ignored if tagging is selected.

When configured for range comparisons and tagging, the ranges are accurate only to word boundaries.

Tagging is disabled when the BDM becomes active.

## 6.4.7 Breakpoints

It is possible to generate breakpoints from channel transitions to final state or using software to write to the TRIG bit in the DBGC1 register.

### 6.4.7.1 Breakpoints From Comparator Channels

Breakpoints can be generated when the state sequencer transitions to the Final State. If configured for tagging, then the breakpoint is generated when the tagged opcode reaches the execution stage of the instruction queue.

If a tracing session is selected by the TSOURCE bit, breakpoints are requested when the tracing session has completed, thus if Begin aligned triggering is selected, the breakpoint is requested only on completion of the subsequent trace (see Table 6-42). If no tracing session is selected, breakpoints are requested immediately.

If the BRK bit is set, then the associated breakpoint is generated immediately independent of tracing trigger alignment.

**Table 6-42. Breakpoint Setup For CPU Breakpoints**

| BRK | TALIGN | DBGBRK | Breakpoint Alignment |
|---|---|---|---|
| 0 | 0 | 0 | Fill Trace Buffer until trigger then disarm (no breakpoints) |
| 0 | 0 | 1 | Fill Trace Buffer until trigger, then breakpoint request occurs |
| 0 | 1 | 0 | Start Trace Buffer at trigger (no breakpoints) |
| 0 | 1 | 1 | Start Trace Buffer at trigger<br>A breakpoint request occurs when Trace Buffer is full |
| 1 | x | 1 | Terminate tracing and generate breakpoint immediately on trigger |
| 1 | x | 0 | Terminate tracing immediately on trigger |

### 6.4.7.2 Breakpoints Generated Via The TRIG Bit

If a TRIG triggers occur, the Final State is entered whereby tracing trigger alignment is defined by the TALIGN bit. If a tracing session is selected by the TSOURCE bit, breakpoints are requested when the tracing session has completed, thus if Begin aligned triggering is selected, the breakpoint is requested only on completion of the subsequent trace (see Table 6-42). If no tracing session is selected, breakpoints are

requested immediately. TRIG breakpoints are possible with a single write to DBGC1, setting ARM and TRIG simultaneously.

### 6.4.7.3 Breakpoint Priorities

If a TRIG trigger occurs after Begin aligned tracing has already started, then the TRIG no longer has an effect. When the associated tracing session is complete, the breakpoint occurs. Similarly if a TRIG is followed by a subsequent comparator channel match, it has no effect, since tracing has already started.

If a forced SWI breakpoint coincides with a BGND in user code with BDM enabled, then the BDM is activated by the BGND and the breakpoint to SWI is suppressed.

#### 6.4.7.3.1 DBG Breakpoint Priorities And BDM Interfacing

Breakpoint operation is dependent on the state of the BDM module. If the BDM module is active, the CPU is executing out of BDM firmware, thus comparator matches and associated breakpoints are disabled. In addition, while executing a BDM TRACE command, tagging into BDM is disabled. If BDM is not active, the breakpoint gives priority to BDM requests over SWI requests if the breakpoint happens to coincide with a SWI instruction in user code. On returning from BDM, the SWI from user code gets executed.

**Table 6-43. Breakpoint Mapping Summary**

| DBGBRK | BDM Bit (DBGC1[4]) | BDM Enabled | BDM Active | Breakpoint Mapping |
|---|---|---|---|---|
| 0 | X | X | X | No Breakpoint |
| 1 | 0 | X | 0 | Breakpoint to SWI |
| X | X | 1 | 1 | No Breakpoint |
| 1 | 1 | 0 | X | Breakpoint to SWI |
| 1 | 1 | 1 | 0 | Breakpoint to BDM |

BDM cannot be entered from a breakpoint unless the ENABLE bit is set in the BDM. If entry to BDM via a BGND instruction is attempted and the ENABLE bit in the BDM is cleared, the CPU actually executes the BDM firmware code, checks the ENABLE and returns if ENABLE is not set. If not serviced by the monitor then the breakpoint is re-asserted when the BDM returns to normal CPU flow.

If the comparator register contents coincide with the SWI/BDM vector address then an SWI in user code could coincide with a DBG breakpoint. The CPU ensures that BDM requests have a higher priority than SWI requests. Returning from the BDM/SWI service routine care must be taken to avoid a repeated breakpoint at the same address.

Should a tagged or forced breakpoint coincide with a BGND in user code, then the instruction that follows the BGND instruction is the first instruction executed when normal program execution resumes.

**NOTE**

When program control returns from a tagged breakpoint using an RTI or BDM GO command without program counter modification it returns to the instruction whose tag generated the breakpoint. To avoid a repeated breakpoint at the same location reconfigure the DBG module in the SWI routine, if configured for an SWI breakpoint, or over the BDM interface by executing a TRACE command before the GO to increment the program flow past the tagged instruction.

## 6.5 Application Information

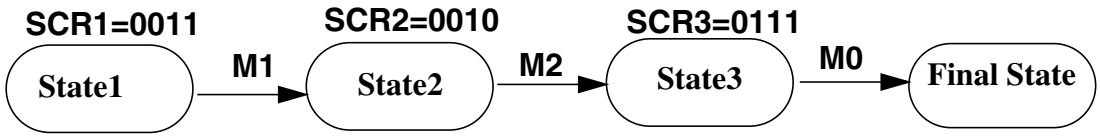
### 6.5.1 State Machine scenarios

Defining the state control registers as SCR1,SCR2, SCR3 and M0,M1,M2 as matches on channels 0,1,2 respectively. SCR encoding supported by S12SDBGV1 are shown in black. SCR encoding supported only in S12SDBGV2 are shown in red. For backwards compatibility the new scenarios use a 4th bit in each SCR register. Thus the existing encoding for SCRx[2:0] is not changed.

### 6.5.2 Scenario 1

A trigger is generated if a given sequence of 3 code events is executed.

**Figure 6-27. Scenario 1**




Scenario 1 is possible with S12SDBGV1 SCR encoding

### 6.5.3 Scenario 2

A trigger is generated if a given sequence of 2 code events is executed.

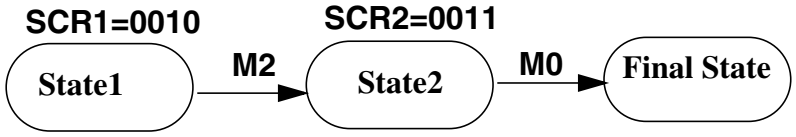
**Figure 6-28. Scenario 2a**

A trigger is generated if a given sequence of 2 code events is executed, whereby the first event is entry into a range (COMPA,COMPB configured for range mode). M1 is disabled in range modes.

**Figure 6-29. Scenario 2b**

A trigger is generated if a given sequence of 2 code events is executed, whereby the second event is entry into a range (COMPA,COMPB configured for range mode)

**Figure 6-30. Scenario 2c**

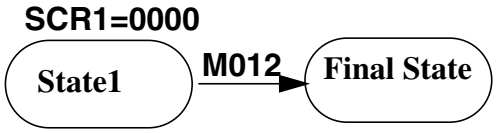



All 3 scenarios 2a,2b,2c are possible with the S12SDBGV1 SCR encoding

### 6.5.4 Scenario 3

A trigger is generated immediately when one of up to 3 given events occurs

**Figure 6-31. Scenario 3**



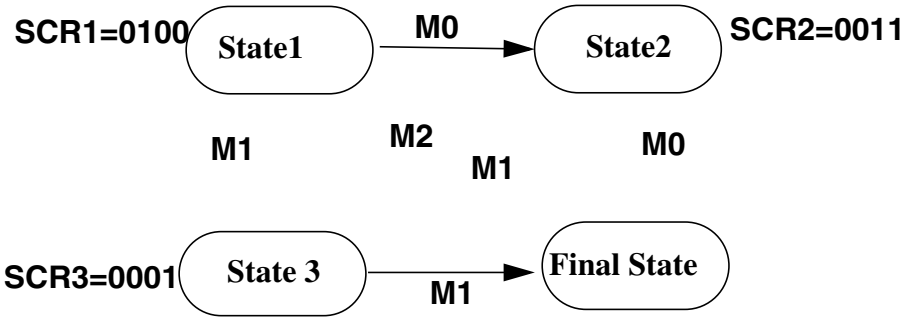

Scenario 3 is possible with S12SDBGV1 SCR encoding

### 6.5.5 Scenario 4

Trigger if a sequence of 2 events is carried out in an incorrect order. Event A must be followed by event B and event B must be followed by event A. 2 consecutive occurrences of event A without an intermediate

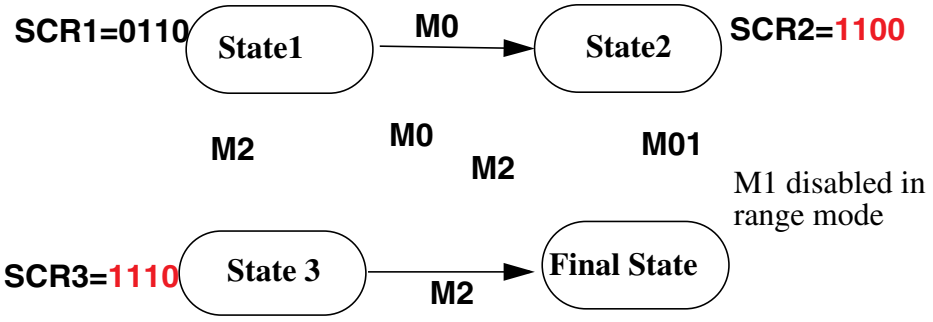
event B cause a trigger. Similarly 2 consecutive occurrences of event B without an intermediate event A cause a trigger. This is possible by using CompA and CompC to match on the same address as shown.

**Figure 6-32. Scenario 4a**




This scenario is currently not possible using 2 comparators only. S12SDBGV2 makes it possible with 2 comparators, State 3 allowing a M0 to return to state 2, whilst a M2 leads to final state as shown.

**Figure 6-33. Scenario 4b (with 2 comparators)**



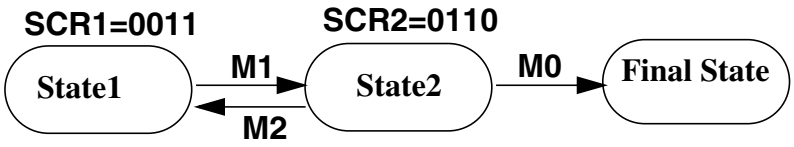

The advantage of using only 2 channels is that now range comparisons can be included (channel0)

This however violates the S12SDBGV1 specification, which states that a match leading to final state always has priority in case of a simultaneous match, whilst priority is also given to the lowest channel number. For S12SDBG the corresponding CPU priority decoder is removed to support this, such that on simultaneous taghits, taghits pointing to final state have highest priority. If no taghit points to final state then the lowest channel number has priority. Thus with the above encoding from State3, the CPU and DBG would break on a simultaneous M0/M2.

### 6.5.6 Scenario 5

Trigger if following event A, event C precedes event B. i.e. the expected execution flow is A->B->C.

**Figure 6-34. Scenario 5**




Scenario 5 is possible with the S12SDBGV1 SCR encoding

### 6.5.7 Scenario 6

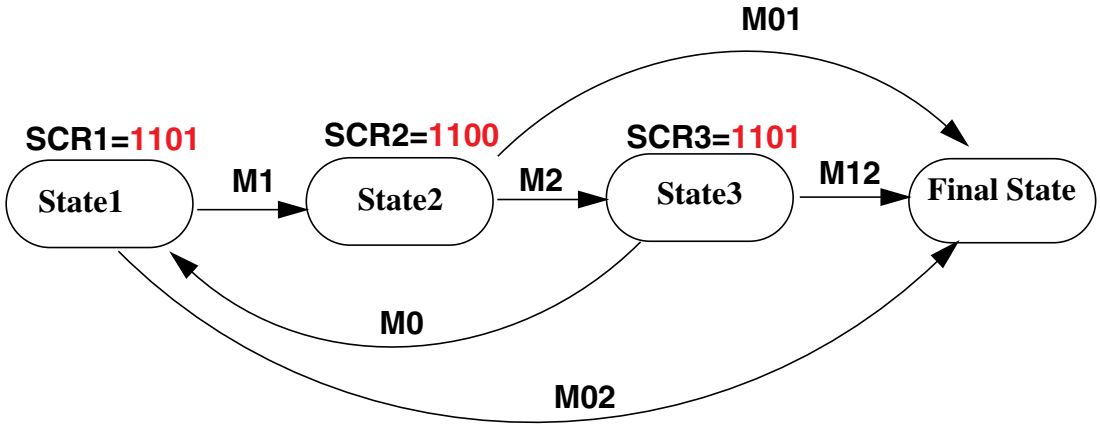
Trigger if event A occurs twice in succession before any of 2 other events (BC) occurs. This scenario is not possible using the S12SDBGV1 SCR encoding. S12SDBGV2 includes additions shown in red. The change in SCR1 encoding also has the advantage that a State1->State3 transition using M0 is now possible. This is advantageous because range and data bus comparisons use channel0 only.

**Figure 6-35. Scenario 6**

### 6.5.8 Scenario 7

Trigger when a series of 3 events is executed out of order. Specifying the event order as M1,M2,M0 to run in loops (120120120). Any deviation from that order should trigger. This scenario is not possible using the S12SDBGV1 SCR encoding because OR possibilities are very limited in the channel encoding. By adding OR forks as shown in red this scenario is possible.

**Figure 6-36. Scenario 7**



On simultaneous matches the lowest channel number has priority so with this configuration the forking from State1 has the peculiar effect that a simultaneous match0/match1 transitions to final state but a simultaneous match2/match1transitions to state2.

### 6.5.9 Scenario 8

Trigger when a routine/event at M2 follows either M1 or M0.

**Figure 6-37. Scenario 8a**

Trigger when an event M2 is followed by either event M0 or event M1

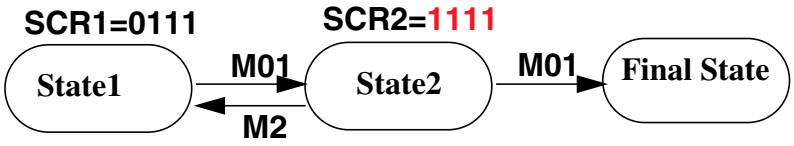
**Figure 6-38. Scenario 8b**

Scenario 8a and 8b are possible with the S12SDBGV1 and S12SDBGV2 SCR encoding

### 6.5.10 Scenario 9

Trigger when a routine/event at A (M2) does not follow either B or C (M1 or M0) before they are executed again. This cannot be realized with theS12SDBGV1 SCR encoding due to OR limitations. By changing the SCR2 encoding as shown in red this scenario becomes possible.

**Figure 6-39. Scenario 9**




### 6.5.11 Scenario 10

Trigger if an event M0 occurs following up to two successive M2 events without the resetting event M1. As shown up to 2 consecutive M2 events are allowed, whereby a reset to State1 is possible after either one or two M2 events. If an event M0 occurs following the second M2, before M1 resets to State1 then a trigger

is generated. Configuring CompA and CompC the same, it is possible to generate a breakpoint on the third consecutive occurrence of event M0 without a reset M1.

**Figure 6-40. Scenario 10a**

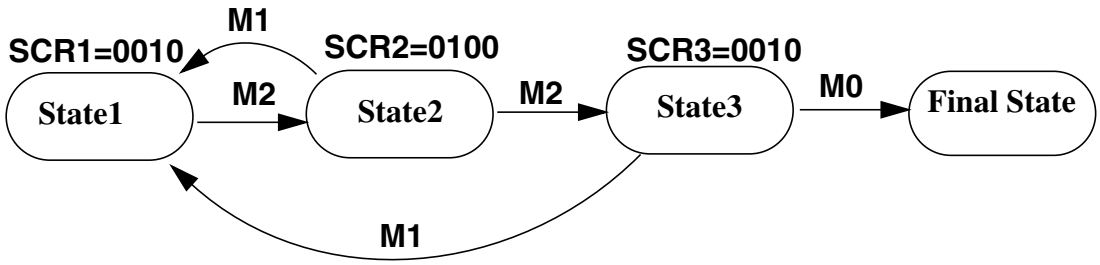
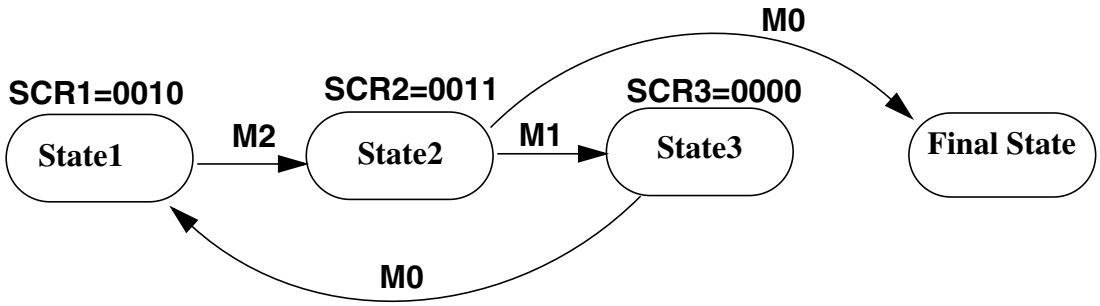



**Figure 6-41. Scenario 10b**




Scenario 10b shows the case that after M2 then M1 must occur before M0. Starting from a particular point in code, event M2 must always be followed by M1 before M0. If after any M2, event M0 occurs before M1 then a trigger is generated.

# Chapter 7
# Interrupt Module (S12SINTV1)

| Version Number | Revision Date | Effective Date | Author | Description of Changes |
|---|---|---|---|---|
| 01.02 | 13 Sep 2007 | | | updates for S12P family devices:<br>- re-added XIRQ and IRQ references since this functionality is used on devices without D2D<br>- added low voltage reset as possible source to the pin reset vector |
| 01.04 | 20 May 2009 | | | added footnote about availability of “Wake-up from STOP or WAIT by XIRQ with X bit set” feature |
| 01.05 | 14 Dec 2011 | | | Re-worded for difference of Wake-up feature between STOP and WAIT modes. |

## 7.1 Introduction

The INT module decodes the priority of all system exception requests and provides the applicable vector for processing the exception to the CPU. The INT module supports:

- I bit and X bit maskable interrupt requests
- A non-maskable unimplemented op-code trap
- A non-maskable software interrupt (SWI) or background debug mode request
- Three system reset vector requests
- A spurious interrupt vector

Each of the I bit maskable interrupt requests is assigned to a fixed priority level.

### 7.1.1 Glossary

Table 7-2 contains terms and abbreviations used in the document.

**Table 7-2. Terminology**

| Term | Meaning |
|---|---|
| CCR | Condition Code Register (in the CPU) |
| ISR | Interrupt Service Routine |
| MCU | Micro-Controller Unit |

### 7.1.2 Features

- Interrupt vector base register (IVBR)
- One spurious interrupt vector (at address vector base[1] + 0x0080).

- 2–58 I bit maskable interrupt vector requests (at addresses vector base + 0x0082–0x00F2).
- I bit maskable interrupts can be nested.
- One X bit maskable interrupt vector request (at address vector base + 0x00F4).
- One non-maskable software interrupt request (SWI) or background debug mode vector request (at address vector base + 0x00F6).
- One non-maskable unimplemented op-code trap (TRAP) vector (at address vector base + 0x00F8).
- Three system reset vectors (at addresses 0xFFFA–0xFFFE).
- Determines the highest priority interrupt vector requests, drives the vector to the bus on CPU request
- Wakes up the system from stop or wait mode when an appropriate interrupt request occurs.

### 7.1.3 Modes of Operation

- Run mode

  This is the basic mode of operation.
- Wait mode

  In wait mode, the clock to the INT module is disabled. The INT module is however capable of waking-up the CPU from wait mode if an interrupt occurs. Please refer to Section 7.5.3, “Wake Up from Stop or Wait Mode” for details.
- Stop Mode

  In stop mode, the clock to the INT module is disabled. The INT module is however capable of waking-up the CPU from stop mode if an interrupt occurs. Please refer to Section 7.5.3, “Wake Up from Stop or Wait Mode” for details.
- Freeze mode (BDM active)

  In freeze mode (BDM active), the interrupt vector base register is overridden internally. Please refer to Section 7.3.1.1, “Interrupt Vector Base Register (IVBR)” for details.

### 7.1.4 Block Diagram

Figure 7-1 shows a block diagram of the INT module.

1. The vector base is a 16-bit address which is accumulated from the contents of the interrupt vector base register (IVBR, used as upper byte) and 0x00 (used as lower byte).

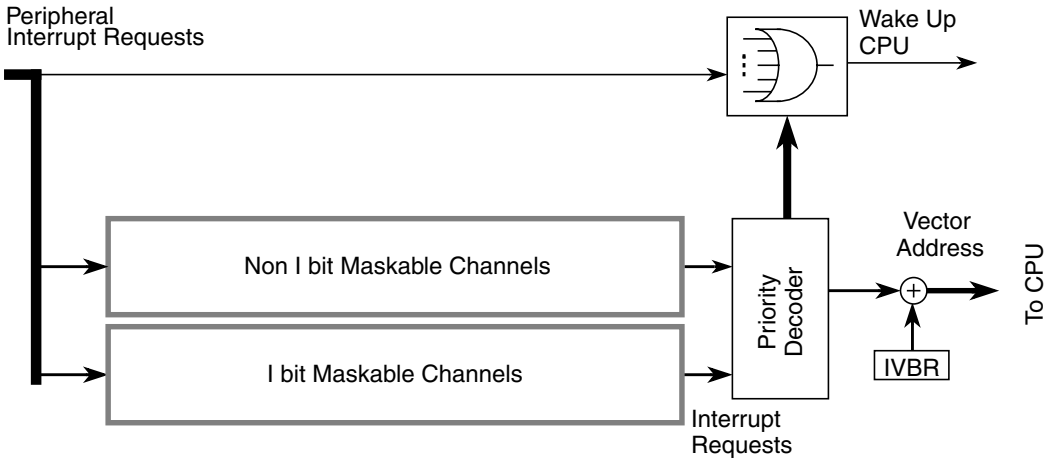



**Figure 7-1. INT Block Diagram**

## 7.2 External Signal Description

The INT module has no external signals.

## 7.3 Memory Map and Register Definition

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

### 7.3.1 Register Descriptions

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

#### 7.3.1.1 Interrupt Vector Base Register (IVBR)

Address: 0x0120

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | IVB_ADDR[7:0] | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figure 7-2. Interrupt Vector Base Register (IVBR)**

Read: Anytime

Write: Anytime

**Table 7-3. IVBR Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>IVB_ADDR[7:0] | **Interrupt Vector Base Address Bits** — These bits represent the upper byte of all vector addresses. Out of reset these bits are set to 0xFF (that means vectors are located at 0xFF80–0xFFFE) to ensure compatibility to HCS12.<br>**Note:** A system reset will initialize the interrupt vector base register with “0xFF” before it is used to determine the reset vector address. Therefore, changing the IVBR has no effect on the location of the three reset vectors (0xFFFA–0xFFFE).<br>**Note:** If the BDM is active (that means the CPU is in the process of executing BDM firmware code), the contents of IVBR are ignored and the upper byte of the vector address is fixed as “0xFF”. This is done to enable handling of all non-maskable interrupts in the BDM firmware. |

## 7.4 Functional Description

The INT module processes all exception requests to be serviced by the CPU module. 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.

### 7.4.1 S12S Exception Requests

The CPU handles both reset requests and interrupt requests. A priority decoder is used to evaluate the priority of pending interrupt requests.

### 7.4.2 Interrupt Prioritization

The INT module contains a priority decoder to determine the priority for all interrupt requests pending for the CPU. If more than one interrupt request is pending, the interrupt request with the higher vector address wins the prioritization.

The following conditions must be met for an I bit maskable interrupt request to be processed.

1. The local interrupt enabled bit in the peripheral module must be set.
2. The I bit in the condition code register (CCR) of the CPU must be cleared.
3. There is no SWI, TRAP, or X bit maskable request pending.

**NOTE**

All non I bit maskable interrupt requests always have higher priority than the I bit maskable interrupt requests. If the X bit in the CCR is cleared, it is possible to interrupt an I bit maskable interrupt by an X bit maskable interrupt. It is possible to nest non maskable interrupt requests, for example by nesting SWI or TRAP calls.

Since an interrupt vector is only supplied at the time when the CPU requests it, it is possible that a higher priority interrupt request could override the original interrupt request 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 interrupt request first, before the original interrupt request is processed.

If the interrupt source is unknown (for example, in the case where an interrupt request becomes inactive after the interrupt has been recognized, but prior to the CPU vector request), the vector address supplied to the CPU will default to that of the spurious interrupt vector.

**NOTE**

Care must be taken to ensure that all interrupt requests remain active until the system begins execution of the applicable service routine; otherwise, the exception request may not get processed at all or the result may be a spurious interrupt request (vector at address (vector base + 0x0080)).

### 7.4.3 Reset Exception Requests

The INT module supports three system reset exception request types (please refer to the Clock and Reset generator module for details):

1. Pin reset, power-on reset or illegal address reset, low voltage reset (if applicable)
2. Clock monitor reset request
3. COP watchdog reset request

### 7.4.4 Exception Priority

The priority (from highest to lowest) and address of all exception vectors issued by the INT module upon request by the CPU is shown in Table 7-4.

**Table 7-4. Exception Vector Map and Priority**

| Vector Address[1] | Source |
|---|---|
| 0xFFFE | Pin reset, power-on reset, illegal address reset, low voltage reset (if applicable) |
| 0xFFFC | Clock monitor reset |
| 0xFFFA | COP watchdog reset |
| (Vector base + 0x00F8) | Unimplemented opcode trap |
| (Vector base + 0x00F6) | Software interrupt instruction (SWI) or BDM vector request |
| (Vector base + 0x00F4) | X bit maskable interrupt request (XIRQ or D2D error interrupt)[2] |
| (Vector base + 0x00F2) | IRQ or D2D interrupt request[3] |
| (Vector base + 0x00F0–0x0082) | Device specific I bit maskable interrupt sources (priority determined by the low byte of the vector address, in descending order) |
| (Vector base + 0x0080) | Spurious interrupt |

1 16 bits vector address based

2 D2D error interrupt on MCUs featuring a D2D initiator module, otherwise $\overline{\text{XIRQ}}$ pin interrupt

3 D2D interrupt on MCUs featuring a D2D initiator module, otherwise $\overline{\text{IRQ}}$ pin interrupt

## 7.5 Initialization/Application Information

### 7.5.1 Initialization

After system reset, software should:

1. Initialize the interrupt vector base register if the interrupt vector table is not located at the default location (0xFF80–0xFFF9).
2. Enable I bit maskable interrupts by clearing the I bit in the CCR.
3. Enable the X bit maskable interrupt by clearing the X bit in the CCR.

### 7.5.2 Interrupt Nesting

The interrupt request scheme makes it possible to nest I bit maskable interrupt requests handled by the CPU.

- I bit maskable interrupt requests can be interrupted by an interrupt request with a higher priority.

I bit maskable interrupt requests cannot be interrupted by other I bit maskable interrupt requests per default. In order to make an interrupt service routine (ISR) interruptible, the ISR must explicitly clear the I bit in the CCR (CLI). After clearing the I bit, other I bit maskable interrupt requests can interrupt the current ISR.

An ISR of an interruptible I bit maskable interrupt request could basically look like this:

1. Service interrupt, that is clear interrupt flags, copy data, etc.
2. Clear I bit in the CCR by executing the instruction CLI (thus allowing other I bit maskable interrupt requests)
3. Process data
4. Return from interrupt by executing the instruction RTI

### 7.5.3 Wake Up from Stop or Wait Mode

#### 7.5.3.1 CPU Wake Up from Stop or Wait Mode

Every I bit maskable interrupt request is capable of waking the MCU from wait mode.

Since bus and core clocks are disabled in stop mode, only interrupt requests that can be generated without these clocks can wake the MCU from stop mode. These are listed in the device overview interrupt vector table.

To determine whether an I bit maskable interrupts is qualified to wake-up the CPU or not, the same conditions as in normal run mode are applied during stop or wait mode:

- If the I bit in the CCR is set, all I bit maskable interrupts are masked from waking-up the MCU.

The X bit maskable interrupt request can wake up the MCU from stop or wait mode at anytime, even if the X bit in CCR is set[1].

If the X bit maskable interrupt request is used to wake-up the MCU with the X bit in the CCR set, the associated ISR is not called. The CPU then resumes program execution with the instruction following the WAI or STOP instruction. This features works following the same rules like any interrupt request, that is care must be taken that the X interrupt request used for wake-up remains active at least until the system begins execution of the instruction following the WAI or STOP instruction; otherwise, wake-up may not occur.

1. The capability of the $\overline{\text{XIRQ}}$ pin to wake-up the MCU with the X bit set may not be available if, for example, the $\overline{\text{XIRQ}}$ pin is shared with other peripheral modules on the device. Please refer to the Device section of the MCU reference manual for details.

# Chapter 8
# Analog-to-Digital Converter (ADC12B6CV2)

**Revision History**

| Version Number | Revision Date | Effective Date | Author | Description of Changes |
|---|---|---|---|---|
| V02.10 | 29. Jun 2012 | 29. Jun 2012 | | Removed IP name in block diagram Figure 8-1 |
| V02.11 | 02 Oct 2012 | 02 Oct 2012 | | Added user information to avoid maybe false external trigger events when enabling the external trigger mode (Section 8.4.2.1, “External Trigger Input). |
| V02.12 | 09 Nov 2012 | 09 Nov 2012 | | Updated Table 8-9 for 8bit and 10bit resolution only version. |

## 8.1 Introduction

The ADC12B6CV2 is a 6-channel, 10-bit, multiplexed input successive approximation analog-to-digital converter. Refer to device electrical specifications for ATD accuracy.

### 8.1.1 Features

- 8-, 10-bit resolution.
- Automatic return to low power after conversion sequence
- Automatic compare with interrupt for higher than or less/equal than programmable value
- Programmable sample time.
- Left/right justified result data.
- External trigger control.
- Sequence complete interrupt.
- Analog input multiplexer for 6 analog input channels.
- Special conversions for VRH, VRL, (VRL+VRH)/2 and ADC temperature sensor.
- 1-to-6 conversion sequence lengths.
- Continuous conversion mode.
- Multiple channel scans.
- Configurable external trigger functionality on any AD channel or any of four additional trigger inputs. The four additional trigger inputs can be chip external or internal. Refer to device specification for availability and connectivity.
- Configurable location for channel wrap around (when converting multiple channels in a sequence).

## 8.1.2 Modes of Operation

### 8.1.2.1 Conversion Modes

There is software programmable selection between performing **single** or **continuous conversion** on a **single channel** or **multiple channels**.

### 8.1.2.2 MCU Operating Modes

- **Stop Mode**
  Entering Stop Mode aborts any conversion sequence in progress and if a sequence was aborted restarts it after exiting stop mode. This has the same effect/consequences as starting a conversion sequence with write to ATDCTL5. So after exiting from stop mode with a previously aborted sequence all flags are cleared etc.
- **Wait Mode**
  ADC12B6CV2 behaves same in Run and Wait Mode. For reduced power consumption continuous conversions should be aborted before entering Wait mode.
- **Freeze Mode**
  In Freeze Mode the ADC12B6CV2 will either continue or finish or stop converting according to the FRZ1 and FRZ0 bits. This is useful for debugging and emulation.

### 8.1.3 Block Diagram

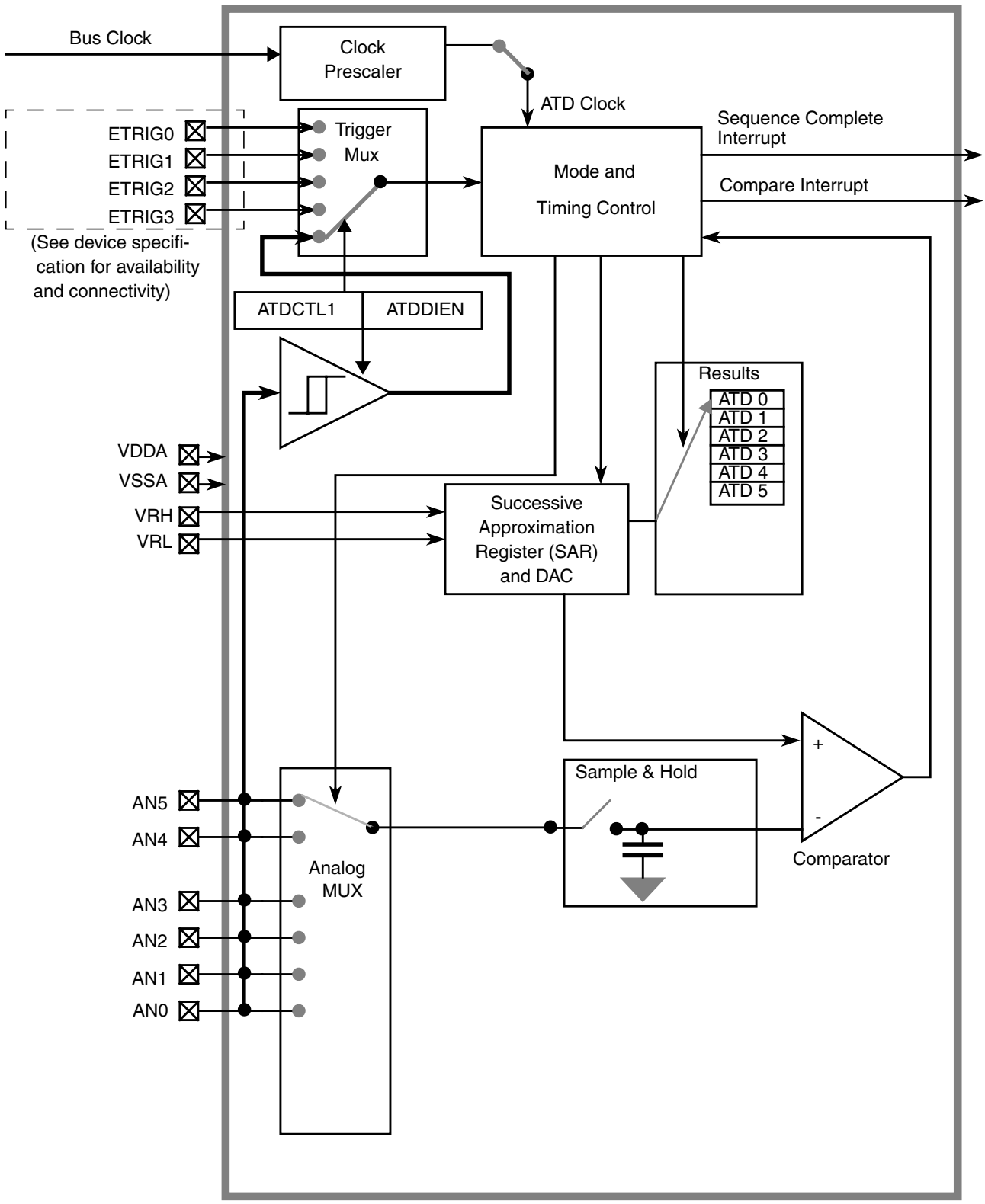



**Figure 8-1. ADC12B6CV2 Block Diagram**

## 8.2 Signal Description

This section lists all inputs to the ADC12B6CV2 block.

### 8.2.1 Detailed Signal Descriptions

#### 8.2.1.1 AN*x* (*x* = 5, 4, 3, 2, 1, 0)

This pin serves as the analog input Channel *x*. It can also be configured as digital port or external trigger for the ATD conversion.

#### 8.2.1.2 ETRIG3, ETRIG2, ETRIG1, ETRIG0

These inputs can be configured to serve as an external trigger for the ATD conversion.

Refer to device specification for availability and connectivity of these inputs!

#### 8.2.1.3 VRH, VRL

VRH is the high reference voltage, VRL is the low reference voltage for ATD conversion.

#### 8.2.1.4 VDDA, VSSA

These pins are the power supplies for the analog circuitry of the ADC12B6CV2 block.

## 8.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the ADC12B6CV2.

### 8.3.1 Module Memory Map

Figure 8-2 gives an overview on all ADC12B6CV2 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.

| Address | Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0000 | ATDCTL0 | R<br>W | Reserved | 0 | 0 | 0 | WRAP3 | WRAP2 | WRAP1 | WRAP0 |
| 0x0001 | ATDCTL1 | R<br>W | ETRIGSEL | SRES1 | SRES0 | SMP_DIS | ETRIGCH3 | ETRIGCH2 | ETRIGCH1 | ETRIGCH0 |
| 0x0002 | ATDCTL2 | R<br>W | 0 | AFFC | Reserved | ETRIGLE | ETRIGP | ETRIGE | ASCIE | ACMPIE |

= Unimplemented or Reserved

**Figure 8-2. ADC12B6CV2 Register Summary (Sheet 1 of 2)**

| Address | Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0003 | ATDCTL3 | R | DJM | S8C | S4C | S2C | S1C | FIFO | FRZ1 | FRZ0 |
| | | W | | | | | | | | |
| 0x0004 | ATDCTL4 | R | SMP2 | SMP1 | SMP0 | PRS[4:0] | | | | |
| | | W | | | | | | | | |
| 0x0005 | ATDCTL5 | R | 0 | SC | SCAN | MULT | CD | CC | CB | CA |
| | | W | | | | | | | | |
| 0x0006 | ATDSTAT0 | R | SCF | 0 | ETORF | FIFOR | CC3 | CC2 | CC1 | CC0 |
| | | W | | | | | | | | |
| 0x0007 | Unimple-mented | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0008 | ATDCMPEH | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0009 | ATDCMPEL | R | 0 | 0 | CMPE[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x000A | ATDSTAT2H | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000B | ATDSTAT2L | R | 0 | 0 | CCF[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x000C | ATDDIENH | R | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | W | | | | | | | | |
| 0x000D | ATDDIENL | R | 1 | 1 | IEN[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x000E | ATDCMPHTH | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000F | ATDCMPHTL | 0 | 0 | 0 | CMPHT[5:0] | | | | | |
| | | | | | | | | | | |
| 0x0010 | ATDDR0 | R | See Section 8.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 8.3.2.12.2, “Right Justified Result Data (DJM=1)” | | | | | | | |
| | | W | | | | | | | | |
| 0x0012 | ATDDR1 | R | See Section 8.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 8.3.2.12.2, “Right Justified Result Data (DJM=1)” | | | | | | | |
| | | W | | | | | | | | |
| 0x0014 | ATDDR2 | R | See Section 8.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 8.3.2.12.2, “Right Justified Result Data (DJM=1)” | | | | | | | |
| | | W | | | | | | | | |
| 0x0016 | ATDDR3 | R | See Section 8.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 8.3.2.12.2, “Right Justified Result Data (DJM=1)” | | | | | | | |
| | | W | | | | | | | | |
| 0x0018 | ATDDR4 | R | See Section 8.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 8.3.2.12.2, “Right Justified Result Data (DJM=1)” | | | | | | | |
| | | W | | | | | | | | |
| 0x001A | ATDDR5 | R | See Section 8.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 8.3.2.12.2, “Right Justified Result Data (DJM=1)” | | | | | | | |
| | | W | | | | | | | | |
| 0x001C-0x002F | Unimple-mented | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

= Unimplemented or Reserved

**Figure 8-2. ADC12B6CV2 Register Summary (Sheet 2 of 2)**

## 8.3.2 Register Descriptions

This section describes in address order all the ADC12B6CV2 registers and their individual bits.

### 8.3.2.1 ATD Control Register 0 (ATDCTL0)

Writes to this register will abort current conversion sequence.

Module Base + 0x0000

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | Reserved | 0 | 0 | 0 | WRAP3 | WRAP2 | WRAP1 | WRAP0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

= Unimplemented or Reserved

**Figure 8-3. ATD Control Register 0 (ATDCTL0)**

Read: Anytime

Write: Anytime, in special modes always write 0 to Reserved Bit 7.

**Table 8-1. ATDCTL0 Field Descriptions**

| Field | Description |
|---|---|
| | |
| 3-0<br>WRAP[3-0] | **Wrap Around Channel Select Bits** — These bits determine the channel for wrap around when doing multi-channel conversions. The coding is summarized in Table 8-2. |

**Table 8-2. Multi-Channel Wrap Around Coding**

| WRAP3 | WRAP2 | WRAP1 | WRAP0 | Multiple Channel Conversions (MULT = 1) Wraparound to AN0 after Converting |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Reserved[1] |
| 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 | AN5 |
| 0 | 1 | 1 | 1 | AN5 |
| 1 | 0 | 0 | 0 | AN5 |
| 1 | 0 | 0 | 1 | AN5 |
| 1 | 0 | 1 | 0 | AN5 |
| 1 | 0 | 1 | 1 | AN5 |
| 1 | 1 | 0 | 0 | AN5 |
| 1 | 1 | 0 | 1 | AN5 |
| 1 | 1 | 1 | 0 | AN5 |
| 1 | 1 | 1 | 1 | AN5 |

[1]If only AN0 should be converted use MULT=0.

### 8.3.2.2 ATD Control Register 1 (ATDCTL1)

Writes to this register will abort current conversion sequence.

Module Base + 0x0001

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | ETRIGSEL | SRES1 | SRES0 | SMP_DIS | ETRIGCH3 | ETRIGCH2 | ETRIGCH1 | ETRIGCH0 |
| Reset | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |

**Figure 8-4. ATD Control Register 1 (ATDCTL1)**

Read: Anytime

Write: Anytime

**Table 8-3. ATDCTL1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>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 device specification for availability and connectivity of ETRIG3-0 inputs. If a particular ETRIG3-0 input option is not available, writing a 1 to ETRISEL only sets the bit but has no effect, this means that one of the AD channels (selected by ETRIGCH3-0) is configured as the source for external trigger. The coding is summarized in Table 8-5. |
| 6–5<br>SRES[1:0] | **A/D Resolution Select** — These bits select the resolution of A/D conversion results. See Table 8-4 for coding. |
| 4<br>SMP_DIS | **Discharge Before Sampling Bit**<br>0 No discharge before sampling.<br>1 The internal sample capacitor is discharged before sampling the channel. This adds 2 ATD clock cycles to the sampling time. This can help to detect an open circuit instead of measuring the previous sampled channel. |
| 3–0<br>ETRIGCH[3:0] | **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 8-5. |

**Table 8-4. A/D Resolution Coding**

| SRES1 | SRES0 | A/D Resolution |
|---|---|---|
| 0 | 0 | 8-bit data |
| 0 | 1 | 10-bit data |
| 1 | 0 | Reserved |
| 1 | 1 | Reserved |

**Table 8-5. External Trigger Channel Select Coding**

| ETRIGSEL | ETRIGCH3 | ETRIGCH2 | ETRIGCH1 | ETRIGCH0 | External trigger source is |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | AN0 |
| 0 | 0 | 0 | 0 | 1 | AN1 |
| 0 | 0 | 0 | 1 | 0 | AN2 |
| 0 | 0 | 0 | 1 | 1 | AN3 |
| 0 | 0 | 1 | 0 | 0 | AN4 |
| 0 | 0 | 1 | 0 | 1 | AN5 |
| 0 | 0 | 1 | 1 | 0 | AN5 |
| 0 | 0 | 1 | 1 | 1 | AN5 |
| 0 | 1 | 0 | 0 | 0 | AN5 |
| 0 | 1 | 0 | 0 | 1 | AN5 |
| 0 | 1 | 0 | 1 | 0 | AN5 |
| 0 | 1 | 0 | 1 | 1 | AN5 |
| 0 | 1 | 1 | 0 | 0 | AN5 |
| 0 | 1 | 1 | 0 | 1 | AN5 |
| 0 | 1 | 1 | 1 | 0 | AN5 |
| 0 | 1 | 1 | 1 | 1 | AN5 |
| 1 | 0 | 0 | 0 | 0 | ETRIG0[1] |
| 1 | 0 | 0 | 0 | 1 | ETRIG1[1] |
| 1 | 0 | 0 | 1 | 0 | ETRIG2[1] |
| 1 | 0 | 0 | 1 | 1 | ETRIG3[1] |
| 1 | 0 | 1 | X | X | Reserved |
| 1 | 1 | X | X | X | Reserved |

[1] Only if ETRIG3-0 input option is available (see device specification), else ETRISEL is ignored, that means external trigger source is still on one of the AD channels selected by ETRIGCH3-0

### 8.3.2.3 ATD Control Register 2 (ATDCTL2)

Writes to this register will abort current conversion sequence.

Module Base + 0x0002

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | AFFC | Reserved | ETRIGLE | ETRIGP | ETRIGE | ASCIE | ACMPIE |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-5. ATD Control Register 2 (ATDCTL2)**

Read: Anytime

Write: Anytime

**Table 8-6. ATDCTL2 Field Descriptions**

| Field | Description |
|---|---|
| 6<br>AFFC | **ATD Fast Flag Clear All**<br>0 ATD flag clearing done by write 1 to respective CCF[*n*] flag.<br>1 Changes all ATD conversion complete flags to a fast clear sequence.<br>For compare disabled (CMPE[*n*]=0) a read access to the result register will cause the associated CCF[*n*] flag to clear automatically.<br>For compare enabled (CMPE[*n*]=1) a write access to the result register will cause the associated CCF[*n*] flag to clear automatically. |
| 5<br>Reserved | Do not alter this bit from its reset value.It is for Manufacturer use only and can change the ATD behavior. |
| 4<br>ETRIGLE | **External Trigger Level/Edge Control** — This bit controls the sensitivity of the external trigger signal. See Table 8-7 for details. |
| 3<br>ETRIGP | **External Trigger Polarity** — This bit controls the polarity of the external trigger signal. See Table 8-7 for details. |
| 2<br>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 8-5. If the external trigger source is one of the AD channels, the digital input buffer of this channel is enabled. The external trigger allows to synchronize the start of conversion with external events.<br>0 Disable external trigger<br>1 Enable external trigger |
| 1<br>ASCIE | **ATD Sequence Complete Interrupt Enable**<br>0 ATD Sequence Complete interrupt requests are disabled.<br>1 ATD Sequence Complete interrupt will be requested whenever SCF=1 is set. |
| 0<br>ACMPIE | **ATD Compare Interrupt Enable** — If automatic compare is enabled for conversion *n* (CMPE[*n*]=1 in ATDCMPE register) this bit enables the compare interrupt. If the CCF[*n*] flag is set (showing a successful compare for conversion *n*), the compare interrupt is triggered.<br>0 ATD Compare interrupt requests are disabled.<br>1 For the conversions in a sequence for which automatic compare is enabled (CMPE[*n*]=1), an ATD Compare Interrupt will be requested whenever any of the respective CCF flags is set. |

**Table 8-7. External Trigger Configurations**

| ETRIGLE | ETRIGP | External Trigger Sensitivity |
|---|---|---|
| 0 | 0 | Falling edge |
| 0 | 1 | Rising edge |
| 1 | 0 | Low level |
| 1 | 1 | High level |

### 8.3.2.4 ATD Control Register 3 (ATDCTL3)

Writes to this register will abort current conversion sequence.

Module Base + 0x0003

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | DJM | S8C | S4C | S2C | S1C | FIFO | FRZ1 | FRZ0 |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-6. ATD Control Register 3 (ATDCTL3)**

Read: Anytime

Write: Anytime

**Table 8-8. ATDCTL3 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>DJM | **Result Register Data Justification** — Result data format is always unsigned. This bit controls justification of conversion data in the result registers.<br>0 Left justified data in the result registers.<br>1 Right justified data in the result registers.<br>Table 8-9 gives example ATD results for an input signal range between 0 and 5.12 Volts. |
| 6–3<br>S8C, S4C,<br>S2C, S1C | **Conversion Sequence Length** — These bits control the number of conversions per sequence. Table 8-10 shows all combinations. At reset, S4C is set to 1 (sequence length is 4). This is to maintain software continuity to HC12 family. |
| 2<br>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 (ATDDR0), the second result in the second result register (ATDDR1), and so on.<br>If this bit is one (FIFO mode) the conversion counter is not reset at the beginning or end 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 (CC3-0 in ATDSTAT0) can be used to determine where in the result register file, the current conversion result will be placed.<br>Aborting a conversion or starting a new conversion 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).<br>Which result registers hold valid data can be tracked using the conversion complete flags. Fast flag clear mode may be useful in a particular application to track valid data.<br>If this bit is one, automatic compare of result registers is always disabled, that is ADC12B6CV2 will behave as if ACMPIE and all CPME[*n*] were zero.<br>0 Conversion results are placed in the corresponding result register up to the selected sequence length.<br>1 Conversion results are placed in consecutive result registers (wrap around at end). |
| 1–0<br>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 8-11. 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 8-9. Examples of ideal decimal ATD Results**

| Input Signal<br>VRL = 0 Volts<br>VRH = 5.12 Volts | 8-Bit<br>Codes<br>(resolution=20mV) | 10-Bit<br>Codes<br>(resolution=5mV) |
|---|---|---|
| 5.120 Volts | 255 | 1023 |
| ... | ... | ... |
| 0.022 | 1 | 4 |
| 0.020 | 1 | 4 |
| 0.018 | 1 | 4 |
| 0.016 | 1 | 3 |
| 0.014 | 1 | 3 |
| 0.012 | 1 | 2 |
| 0.010 | 1 | 2 |
| 0.008 | 0 | 2 |
| 0.006 | 0 | 1 |
| 0.004 | 0 | 1 |
| 0.003 | 0 | 1 |
| 0.002 | 0 | 0 |
| 0.000 | 0 | 0 |

**Table 8-10. Conversion Sequence Length Coding**

| S8C | S4C | S2C | S1C | Number of Conversions per Sequence |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 6 |
| 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 | 6 |
| 1 | 0 | 0 | 0 | 6 |
| 1 | 0 | 0 | 1 | 6 |
| 1 | 0 | 1 | 0 | 6 |
| 1 | 0 | 1 | 1 | 6 |
| 1 | 1 | 0 | 0 | 6 |
| 1 | 1 | 0 | 1 | 6 |
| 1 | 1 | 1 | 0 | 6 |
| 1 | 1 | 1 | 1 | 6 |

**Table 8-11. ATD Behavior in Freeze Mode (Breakpoint)**

| FRZ1 | FRZ0 | Behavior in Freeze Mode |
|---|---|---|
| 0 | 0 | Continue conversion |

**Table 8-11. ATD Behavior in Freeze Mode (Breakpoint)**

| FRZ1 | FRZ0 | Behavior in Freeze Mode |
|---|---|---|
| 0 | 1 | Reserved |
| 1 | 0 | Finish current conversion, then freeze |
| 1 | 1 | Freeze Immediately |

### 8.3.2.5 ATD Control Register 4 (ATDCTL4)

Writes to this register will abort current conversion sequence.

Module Base + 0x0004

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | SMP2 | SMP1 | SMP0 | PRS[4:0] | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

**Figure 8-7. ATD Control Register 4 (ATDCTL4)**

Read: Anytime

Write: Anytime

**Table 8-12. ATDCTL4 Field Descriptions**

| Field | Description |
|---|---|
| 7–5<br>SMP[2:0] | **Sample Time Select** — These three bits select the length 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). Table 8-13 lists the available sample time lengths. |
| 4–0<br>PRS[4:0] | **ATD Clock Prescaler** — These 5 bits are the binary prescaler value PRS. The ATD conversion clock frequency is calculated as follows:<br>$f_{ATDCLK} = \frac{f_{BUS}}{2 \times (PRS + 1)}$<br>Refer to Device Specification for allowed frequency range of $f_{ATDCLK}$. |

**Table 8-13. Sample Time Select**

| SMP2 | SMP1 | SMP0 | Sample Time in Number of ATD Clock Cycles |
|---|---|---|---|
| 0 | 0 | 0 | 4 |
| 0 | 0 | 1 | 6 |
| 0 | 1 | 0 | 8 |
| 0 | 1 | 1 | 10 |
| 1 | 0 | 0 | 12 |
| 1 | 0 | 1 | 16 |
| 1 | 1 | 0 | 20 |
| 1 | 1 | 1 | 24 |

### 8.3.2.6 ATD Control Register 5 (ATDCTL5)

Writes to this register will abort current conversion sequence and start a new conversion sequence. If the external trigger function is enabled (ETRIGE=1) an initial write to ATDCTL5 is required to allow starting of a conversion sequence which will then occur on each trigger event. Start of conversion means the beginning of the sampling phase.

Module Base + 0x0005

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | SC | SCAN | MULT | CD | CC | CB | CA |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-8. ATD Control Register 5 (ATDCTL5)**

Read: Anytime

Write: Anytime

**Table 8-14. ATDCTL5 Field Descriptions**

| Field | Description |
|---|---|
| 6<br>SC | **Special Channel Conversion Bit** — If this bit is set, then special channel conversion can be selected using CD, CC, CB and CA of ATDCTL5. Table 8-15 lists the coding.<br>0 Special channel conversions disabled<br>1 Special channel conversions enabled |
| 5<br>SCAN | **Continuous Conversion Sequence Mode** — This bit selects whether conversion sequences are performed continuously or only once. If external trigger function is enabled (ETRIGE=1) setting this bit has no effect, thus the external trigger always starts a single conversion sequence.<br>0 Single conversion sequence<br>1 Continuous conversion sequences (scan mode) |
| 4<br>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 CD/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 (CD, CC, CB, CA control bits); subsequent channels sampled in the sequence are determined by incrementing the channel selection code or wrapping around to AN0 (channel 0).<br>0 Sample only one channel<br>1 Sample across several channels |
| 3–0<br>CD, CC,<br>CB, CA | **Analog Input Channel Select Code** — These bits select the analog input channel(s). Table 8-15 lists the coding used to select the various analog input channels.<br>In the case of single channel conversions (MULT=0), this selection code specifies the channel to be examined.<br>In the case of multiple channel conversions (MULT=1), this selection code specifies the first channel to be examined in the conversion sequence. Subsequent channels are determined by incrementing the channel selection code or wrapping around to AN0 (after converting the channel defined by the Wrap Around Channel Select Bits WRAP3-0 in ATDCTL0). When starting with a channel number higher than the one defined by WRAP3-0 the first wrap around will be AN5 to AN0. |

**Table 8-15. Analog Input Channel Select Coding**

| SC | CD | CC | CB | CA | Analog Input Channel |
|---|---|---|---|---|---|
| 0 | 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 | AN5 |
| | 0 | 1 | 1 | 1 | AN5 |
| | 1 | 0 | 0 | 0 | AN5 |
| | 1 | 0 | 0 | 1 | AN5 |
| | 1 | 0 | 1 | 0 | AN5 |
| | 1 | 0 | 1 | 1 | AN5 |
| | 1 | 1 | 0 | 0 | AN5 |
| | 1 | 1 | 0 | 1 | AN5 |
| | 1 | 1 | 1 | 0 | AN5 |
| | 1 | 1 | 1 | 1 | AN5 |
| 1 | 0 | 0 | 0 | 0 | Internal_6, Temperature sense of ADC hardmacro |
| | 0 | 0 | 0 | 1 | Internal_7 |
| | 0 | 0 | 1 | 0 | Internal_0 |
| | 0 | 0 | 1 | 1 | Internal_1 |
| | 0 | 1 | 0 | 0 | VRH |
| | 0 | 1 | 0 | 1 | VRL |
| | 0 | 1 | 1 | 0 | (VRH+VRL) / 2 |
| | 0 | 1 | 1 | 1 | Reserved |
| | 1 | 0 | 0 | 0 | Internal_2 |
| | 1 | 0 | 0 | 1 | Internal_3 |
| | 1 | 0 | 1 | 0 | Internal_4 |
| | 1 | 0 | 1 | 1 | Internal_5 |
| | 1 | X | X | X | Reserved |

### 8.3.2.7 ATD Status Register 0 (ATDSTAT0)

This register contains the Sequence Complete Flag, overrun flags for external trigger and FIFO mode, and the conversion counter.

Module Base + 0x0006

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | SCF | 0 | ETORF | FIFOR | CC3 | CC2 | CC1 | CC0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-9. ATD Status Register 0 (ATDSTAT0)**

Read: Anytime

Write: Anytime (No effect on (CC3, CC2, CC1, CC0))

**Table 8-16. ATDSTAT0 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>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:<br>A) Write “1” to SCF<br>B) Write to ATDCTL5 (a new conversion sequence is started)<br>C) If AFFC=1 and a result register is read<br>0 Conversion sequence not completed<br>1 Conversion sequence has completed |
| 5<br>ETORF | **External Trigger Overrun Flag** — While in edge sensitive 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:<br>A) Write “1” to ETORF<br>B) Write to ATDCTL0,1,2,3,4, ATDCMPE or ATDCMPHT (a conversion sequence is aborted)<br>C) Write to ATDCTL5 (a new conversion sequence is started)<br>0 No External trigger overrun error has occurred<br>1 External trigger overrun error has occurred |
| 4<br>FIFOR | **Result Register Overrun 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 overwritten before it has been read (i.e. the old data has been lost). This flag is cleared when one of the following occurs:<br>A) Write “1” to FIFOR<br>B) Write to ATDCTL0,1,2,3,4, ATDCMPE or ATDCMPHT (a conversion sequence is aborted)<br>C) Write to ATDCTL5 (a new conversion sequence is started)<br>0 No overrun has occurred<br>1 Overrun condition exists (result register has been written while associated CCFx flag was still set) |

**Table 8-16. ATDSTAT0 Field Descriptions (continued)**

| Field | Description |
|---|---|
| 3–0<br>CC[3:0] | **Conversion Counter** — These 4 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. CC3=0, 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 beginning and end of the conversion sequence. If in FIFO mode (FIFO=1) the register counter is not initialized. The conversion counter wraps around when its maximum value is reached.<br>Aborting a conversion or starting a new conversion clears the conversion counter even if FIFO=1. |

## 8.3.2.8 ATD Compare Enable Register (ATDCMPE)

Writes to this register will abort current conversion sequence.

Read: Anytime

Write: Anytime

Module Base + 0x0008

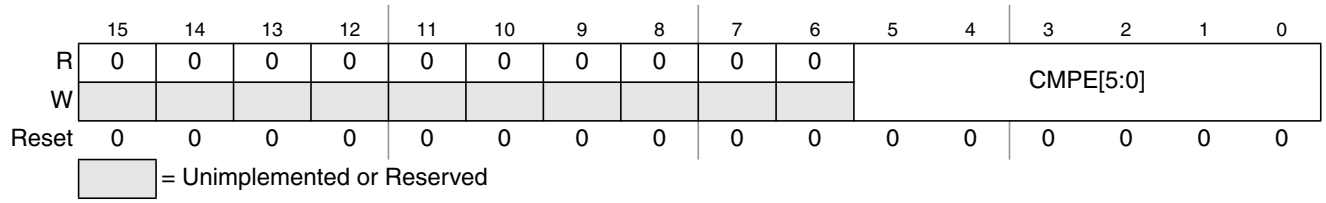



**Figure 8-10. ATD Compare Enable Register (ATDCMPE)**

**Table 8-17. ATDCMPE Field Descriptions**

| Field | Description |
|---|---|
| 5–0<br>CMPE[5:0] | **Compare Enable for Conversion Number *n* (*n*= 5, 4, 3, 2, 1, 0) of a Sequence (*n conversion number, NOT channel number!)*** — These bits enable automatic compare of conversion results individually for conversions of a sequence. The sense of each comparison is determined by the CMPHT[*n*] bit in the ATDCMPHT register.<br>For each conversion number with CMPE[*n*]=1 do the following:<br>1) Write compare value to ATDDR*n* result register<br>2) Write compare operator with CMPHT[*n*] in ATDCPMHT register<br>CCF[*n*] in ATDSTAT2 register will flag individual success of any comparison.<br>0 No automatic compare<br>1 Automatic compare of results for conversion *n* of a sequence is enabled. |

### 8.3.2.9 ATD Status Register 2 (ATDSTAT2)

This read-only register contains the Conversion Complete Flags CCF[5:0].

Module Base + 0x000A

| | 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 | CCF[5:0] | | | | | |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-11. ATD Status Register 2 (ATDSTAT2)**

Read: Anytime

Write: Anytime (for details see Table 8-18 below)

**Table 8-18. ATDSTAT2 Field Descriptions**

| Field | Description |
|---|---|
| 5–0<br>CCF[5:0] | **Conversion Complete Flag *n* (*n*= 5, 4, 3, 2, 1, 0) (*n conversion number, NOT channel number!)***— A conversion complete flag is set at the end of each conversion in a sequence. The flags are associated with the conversion position in a sequence (and also the result register number). Therefore in non-fifo mode, CCF[4] is set when the fifth conversion in a sequence is complete and the result is available in result register ATDDR4; CCF[5] is set when the sixth conversion in a sequence is complete and the result is available in ATDDR5, and so forth.<br>If automatic compare of conversion results is enabled (CMPE[*n*]=1 in ATDCMPE), the conversion complete flag is only set if comparison with ATDDR*n* is true. If ACMPIE=1 a compare interrupt will be requested. In this case, as the ATDDR*n* result register is used to hold the compare value, the result will not be stored there at the end of the conversion but is lost.<br>A flag CCF[*n*] is cleared when one of the following occurs:<br>A) Write to ATDCTL5 (a new conversion sequence is started)<br>B) If AFFC=0, write “1” to CCF[*n*]<br>C) If AFFC=1 and CMPE[*n*]=0, read of result register ATDDR*n*<br>D) If AFFC=1 and CMPE[*n*]=1, write to result register ATDDR*n*<br>In case of a concurrent set and clear on CCF[*n*]: The clearing by method A) will overwrite the set. The clearing by methods B) or C) or D) will be overwritten by the set.<br>0 Conversion number *n* not completed or successfully compared<br>1 If (CMPE[*n*]=0): Conversion number *n* has completed. Result is ready in ATDDR*n*.<br>If (CMPE[*n*]=1): Compare for conversion result number *n* with compare value in ATDDR*n*, using compare operator CMPGT[*n*] is true. (No result available in ATDDR*n*) |

### 8.3.2.10 ATD Input Enable Register (ATDDIEN)

Module Base + 0x000C

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | IEN[5:0] | | | | | |
| W | | | | | | | | | | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-12. ATD Input Enable Register (ATDDIEN)**

Read: Anytime

Write: Anytime

**Table 8-19. ATDDIEN Field Descriptions**

| Field | Description |
|---|---|
| 5–0<br>IEN[5:0] | **ATD Digital Input Enable on channel *x* (*x*= 5, 4, 3, 2, 1, 0)** — This bit controls the digital input buffer from the analog input pin (AN*x*) to the digital data register.<br>0 Disable digital input buffer to AN*x* pin<br>1 Enable digital input buffer on AN*x* pin.<br>**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. |

### 8.3.2.11 ATD Compare Higher Than Register (ATDCMPHT)

Writes to this register will abort current conversion sequence.

Read: Anytime

Write: Anytime

Module Base + 0x000E

| | 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 | CMPHT[5:0] | | | | | |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-13. ATD Compare Higher Than Register (ATDCMPHT)**

**Table 8-20. ATDCMPHT Field Descriptions**

| Field | Description |
|---|---|
| 5–0<br>CMPHT[5:0] | **Compare Operation Higher Than Enable for conversion number *n* (*n*= 5, 4, 3, 2, 1, 0) of a Sequence (*n conversion number, NOT channel number!)*** — This bit selects the operator for comparison of conversion results.<br>0 If result of conversion *n* is **lower or same than** compare value in ATDDR*n*, this is flagged in ATDSTAT2<br>1 If result of conversion *n* is **higher than** compare value in ATDDR*n*, this is flagged in ATDSTAT2 |

### 8.3.2.12 ATD Conversion Result Registers (ATDDR*n*)

The A/D conversion results are stored in 6 result registers. Results are always in unsigned data representation. Left and right justification is selected using the DJM control bit in ATDCTL3.

If automatic compare of conversions results is enabled (CMPE[*n*]=1 in ATDCMPE), these registers must be written with the compare values in left or right justified format depending on the actual value of the DJM bit. In this case, as the ATDDR*n* register is used to hold the compare value, the result will not be stored there at the end of the conversion but is lost.

***Attention, n is the conversion number, NOT the channel number!***

Read: Anytime

Write: Anytime

**NOTE**

For conversions not using automatic compare, results are stored in the result registers after each conversion. In this case avoid writing to ATDDRn except for initial values, because an A/D result might be overwritten.

#### 8.3.2.12.1 Left Justified Result Data (DJM=0)

Module Base +
0x0010 = ATDDR0, 0x0012 = ATDDR1, 0x0014 = ATDDR2, 0x0016 = ATDDR3
0x0018 = ATDDR4, 0x001A = ATDDR5

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | Result-Bit[11:0] | | | | | | | | | | | | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 8-14. Left justified ATD conversion result register (ATDDR*n*)**

Table 8-21 shows how depending on the A/D resolution the conversion result is transferred to the ATD result registers for left justified data. Compare is always done using all 12 bits of both the conversion result and the compare value in ATDDRn.

**Table 8-21. Conversion result mapping to ATDDRn**

| A/D resolution | DJM | conversion result mapping to ATDDR*n* |
|---|---|---|
| 8-bit data | 0 | Result-Bit[11:4] = conversion result,<br>Result-Bit[3:0]=0000 |
| 10-bit data | 0 | Result-Bit[11:2] = conversion result,<br>Result-Bit[1:0]=00 |

### 8.3.2.12.2 Right Justified Result Data (DJM=1)

Module Base +
0x0010 = ATDDR0, 0x0012 = ATDDR1, 0x0014 = ATDDR2, 0x0016 = ATDDR3
0x0018 = ATDDR4, 0x001A = ATDDR5

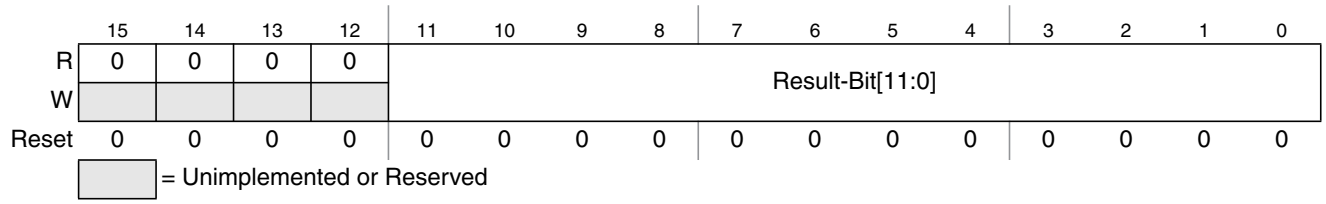



**Figure 8-15. Right justified ATD conversion result register (ATDDR*n*)**

Table 8-22 shows how depending on the A/D resolution the conversion result is transferred to the ATD result registers for right justified data. Compare is always done using all 12 bits of both the conversion result and the compare value in ATDDRn.

**Table 8-22. Conversion result mapping to ATDDRn**

| A/D resolution | DJM | conversion result mapping to ATDDR*n* |
|---|---|---|
| 8-bit data | 1 | Result-Bit[7:0] = result,<br>Result-Bit[11:8]=0000 |
| 10-bit data | 1 | Result-Bit[9:0] = result,<br>Result-Bit[11:10]=00 |

## 8.4 Functional Description

The ADC12B6CV2 consists of an analog sub-block and a digital sub-block.

### 8.4.1 Analog Sub-Block

The analog sub-block contains all analog electronics required to perform a single conversion. Separate power supplies VDDA and VSSA allow to isolate noise of other MCU circuitry from the analog sub-block.

#### 8.4.1.1 Sample and Hold Machine

The Sample and Hold Machine controls the storage and charge of the sample capacitor to the voltage level of the analog signal at the selected ADC input channel.

During the sample process the analog input connects directly to the storage node.

The input analog signals are unipolar and must be within the potential range of VSSA to VDDA.

During the hold process the analog input is disconnected from the storage node.

#### 8.4.1.2 Analog Input Multiplexer

The analog input multiplexer connects one of the 6 external analog input channels to the sample and hold machine.

#### 8.4.1.3 Analog-to-Digital (A/D) Machine

The A/D Machine performs analog to digital conversions. The resolution is program selectable to be either 8 or 10 bits. The A/D machine uses a successive approximation architecture. It functions by comparing the sampled and stored analog voltage with a series of binary coded discrete voltages. By following a binary search algorithm, the A/D machine identifies the discrete voltage that is nearest to the sampled and stored voltage.

When not converting the A/D machine is automatically powered down.

Only analog input signals within the potential range of VRL to VRH (A/D reference potentials) will result in a non-railed digital output code.

### 8.4.2 Digital Sub-Block

This subsection describes some of the digital features in more detail. See Section 8.3.2, “Register Descriptions” for all details.

#### 8.4.2.1 External Trigger Input

The external trigger feature allows the user to synchronize ATD conversions to an external event rather than relying only on software to trigger the ATD module when a conversions is about to take place. The external trigger signal (out of reset ATD channel 5, configurable in ATDCTL1) is programmable to be edge

or level sensitive with polarity control. Table 8-23 gives a brief description of the different combinations of control bits and their effect on the external trigger function.

In order to avoid maybe false trigger events please enable the external digital input via ATDDIEN register first and in the following enable the external trigger mode by bit ETRIGE.

**Table 8-23. 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 | Trigger falling edge sensitive. Performs one conversion sequence per trigger. |
| 0 | 1 | 1 | X | Trigger rising edge sensitive. Performs one conversion sequence per trigger. |
| 1 | 0 | 1 | X | Trigger low level sensitive. Performs continuous conversions while trigger level is active. |
| 1 | 1 | 1 | X | Trigger high level sensitive. Performs continuous conversions while trigger level is active. |

In either level or edge sensitive mode, the first conversion begins when the trigger is received.

Once ETRIGE is enabled a conversion must be triggered externally after writing to ATDCTL5 register.

During a conversion in edge sensitive mode, if additional trigger events are detected the overrun error flag ETORF is set.

If level sensitive mode is active and the external trigger de-asserts and later asserts again during a conversion sequence, this does not constitute an overrun. Therefore, the flag is not set. If the trigger is left active in level sensitive mode when a sequence is about to be complete, another sequence will be triggered immediately.

### 8.4.2.2 General-Purpose Digital Port Operation

Each ATD input pin can be switched between analog or digital input functionality. An analog multiplexer makes each ATD input pin selected as analog input available to the A/D converter.

The pad of the ATD input pin is always connected to the analog input channel of the analog mulitplexer.

Each pad input signal is buffered to the digital port register.

This buffer can be turned on or off with the ATDDIEN register for each ATD input pin.
This is important so that the buffer does not draw excess current when an ATD input pin is selected as analog input to the ADC12B6CV2.

## 8.5 Resets

At reset the ADC12B6CV2 is in a power down state. The reset state of each individual bit is listed within the Register Description section (see Section 8.3.2, “Register Descriptions”) which details the registers and their bit-field.

## 8.6 Interrupts

The interrupts requested by the ADC12B6CV2 are listed in Table 8-24. Refer to MCU specification for related vector address and priority.

**Table 8-24. ATD Interrupt Vectors**

| Interrupt Source | CCR Mask | Local Enable |
|---|---|---|
| Sequence Complete Interrupt | I bit | ASCIE in ATDCTL2 |
| Compare Interrupt | I bit | ACMPIE in ATDCTL2 |

See Section 8.3.2, “Register Descriptions” for further details.

# Chapter 9
# Pulse-Width Modulator (S12PWM8B8CV2)

**Table 9-1. Revision History**

| Revision Number | Revision Date | Sections Affected | Description of Changes |
|---|---|---|---|
| v02.00 | Feb. 20, 2009 | All | Initial revision of scalable PWM. Started from pwm_8b8c (v01.08). |

## 9.1 Introduction

The Version 2 of S12 PWM module is a channel scalable and optimized implementation of S12 PWM8B8C Version 1. The channel is scalable in pairs from PWM0 to PWM7 and the available channel number is 2, 4, 6 and 8. The shutdown feature has been removed and the flexibility to select one of four clock sources per channel has improved. If the corresponding channels exist and shutdown feature is not used, the Version 2 is fully software compatible to Version 1.

### 9.1.1 Features

The scalable PWM block includes these distinctive features:

- Up to eight independent PWM channels, scalable in pairs (PWM0 to PWM7)
- Available channel number could be 2, 4, 6, 8 (refer to device specification for exact number)
- Programmable period and duty cycle for each channel
- Dedicated counter for each PWM channel
- Programmable PWM enable/disable for each channel
- Software selection of PWM duty pulse polarity for each channel
- Period and duty cycle are double buffered. Change takes effect when the end of the effective period is reached (PWM counter reaches zero) or when the channel is disabled.
- Programmable center or left aligned outputs on individual channels
- Up to eight 8-bit channel or four 16-bit channel PWM resolution
- Four clock sources (A, B, SA, and SB) provide for a wide range of frequencies
- Programmable clock select logic

### 9.1.2 Modes of Operation

There is a software programmable option for low power consumption in wait mode that disables the input clock to the prescaler.

In freeze mode there is a software programmable option to disable the input clock to the prescaler. This is useful for emulation.

Wait: The prescaler keeps on running, unless PSWAI in PWMCTL is set to 1.

Freeze: The prescaler keeps on running, unless PFRZ in PWMCTL is set to 1.

### 9.1.3 Block Diagram

Figure 9-1 shows the block diagram for the 8-bit up to 8-channel scalable PWM block.

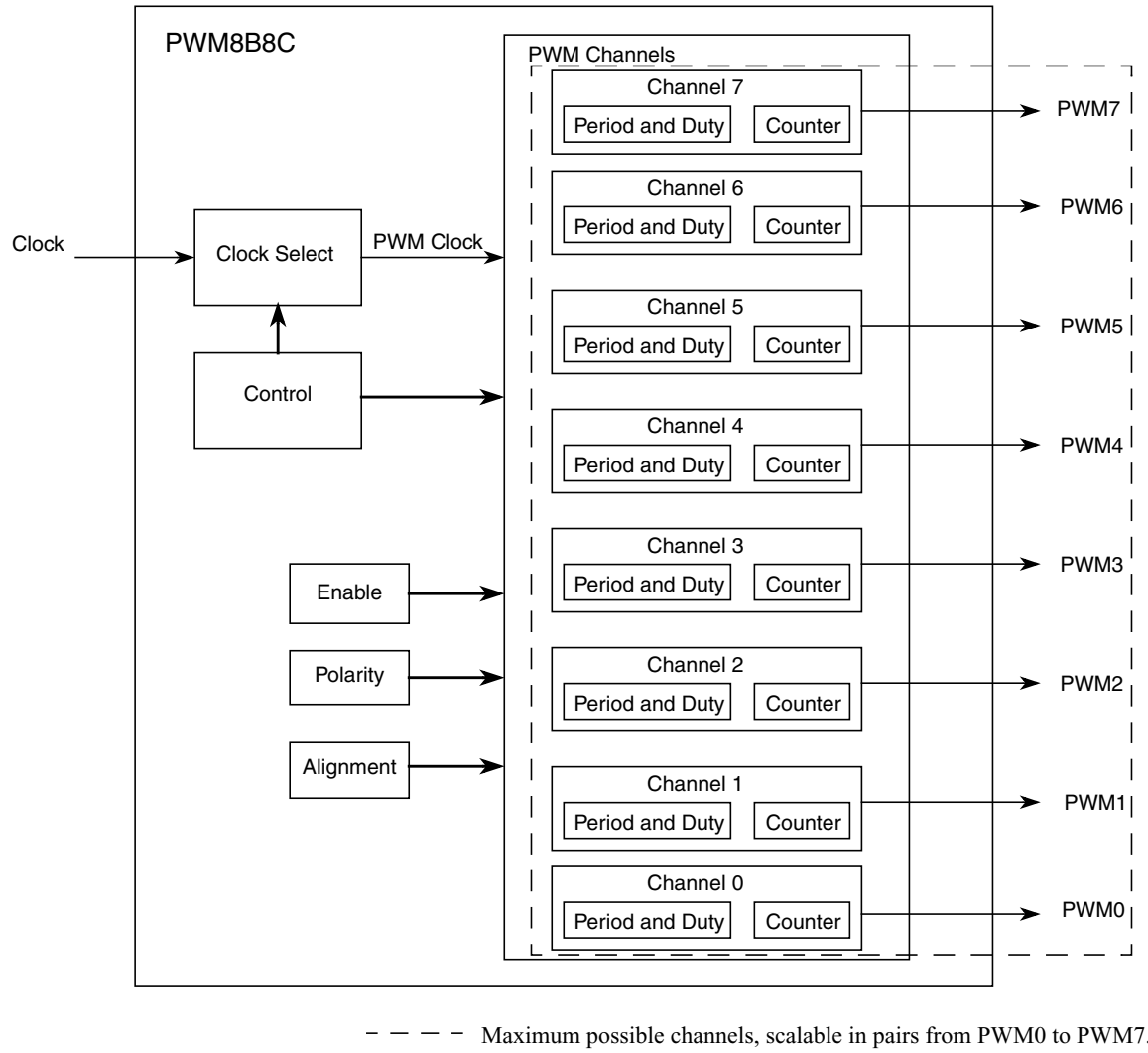



**Figure 9-1. Scalable PWM Block Diagram**

## 9.2 External Signal Description

The scalable PWM module has a selected number of external pins. Refer to device specification for exact number.

### 9.2.1 PWM7 - PWM0 — PWM Channel 7 - 0

Those pins serve as waveform output of PWM channel 7 - 0.

## 9.3 Memory Map and Register Definition

### 9.3.1 Module Memory Map

This section describes the content of the registers in the scalable PWM module. The base address of the scalable PWM module is determined at the MCU level when the MCU is defined. The register decode map is fixed and begins at the first address of the module address offset. The figure below shows the registers associated with the scalable PWM and their relative offset from the base address. The register detail description follows the order they appear in the register map.

Reserved bits within a register will always read as 0 and the write will be unimplemented. Unimplemented functions are indicated by shading the bit.

**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.

### 9.3.2 Register Descriptions

This section describes in detail all the registers and register bits in the scalable PWM module.

| Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0000 PWME[1] | R | PWME7 | PWME6 | PWME5 | PWME4 | PWME3 | PWME2 | PWME1 | PWME0 |
| | W | | | | | | | | |
| 0x0001 PWMPOL[1] | R | PPOL7 | PPOL6 | PPOL5 | PPOL4 | PPOL3 | PPOL2 | PPOL1 | PPOL0 |
| | W | | | | | | | | |
| 0x0002 PWMCLK[1] | R | PCLK7 | PCLKL6 | PCLK5 | PCLK4 | PCLK3 | PCLK2 | PCLK1 | PCLK0 |
| | W | | | | | | | | |
| 0x0003 PWMPRCLK | R | 0 | PCKB2 | PCKB1 | PCKB0 | 0 | PCKA2 | PCKA1 | PCKA0 |
| | W | | | | | | | | |
| 0x0004 PWMCAE[1] | R | CAE7 | CAE6 | CAE5 | CAE4 | CAE3 | CAE2 | CAE1 | CAE0 |
| | W | | | | | | | | |
| 0x0005 PWMCTL[1] | R | CON67 | CON45 | CON23 | CON01 | PSWAI | PFRZ | 0 | 0 |
| | W | | | | | | | | |

= Unimplemented or Reserved

**Figure 9-2. The scalable PWM Register Summary (Sheet 1 of 4)**

| Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0006 PWMCLKAB[1] | R | PCLKAB7 | PCLKAB6 | PCLKAB5 | PCLKAB4 | PCLKAB3 | PCLKAB2 | PCLKAB1 | PCLKAB0 |
| | W | | | | | | | | |
| 0x0007 RESERVED | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0008 PWMSCLA | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | | | | | | | | |
| 0x0009 PWMSCLB | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | | | | | | | | |
| 0x000A RESERVED | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x000B RESERVED | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x000C PWMCNT0[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x000D PWMCNT1[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x000E PWMCNT2[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x000F PWMCNT3[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0010 PWMCNT4[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0011 PWMCNT5[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0012 PWMCNT6[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0013 PWMCNT7[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0014 PWMPER0[2] | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | W | | | | | | | | |
| | | = Unimplemented or Reserved | | | | | | | |

**Figure 9-2. The scalable PWM Register Summary (Sheet 2 of 4)**

| Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0015 PWMPER1[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0016 PWMPER2[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0017 PWMPER3[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0018 PWMPER4[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0019 PWMPER5[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x001A PWMPER6[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x001B PWMPER7[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x001C PWMDTY0[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x001D PWMDTY1[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x001E PWMDTY2[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x001F PWMDTY3[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0010 PWMDTY4[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0021 PWMDTY5[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0022 PWMDTY6[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x0023 PWMDTY7[2] | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |

= Unimplemented or Reserved

**Figure 9-2. The scalable PWM Register Summary (Sheet 3 of 4)**

| Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0024 RESERVED | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0025 RESERVED | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0026 RESERVED | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0027 RESERVED | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |

= Unimplemented or Reserved

**Figure 9-2. The scalable PWM Register Summary (Sheet 4 of 4)**

1 The related bit is available only if corresponding channel exists.

2 The register is available only if corresponding channel exists.

### 9.3.2.1 PWM Enable Register (PWME)

Each PWM channel has an enable bit (PWMEx) to start its waveform output. When any of the PWMEx bits are set (PWMEx = 1), the associated PWM output is enabled immediately. However, the actual PWM waveform is not available on the associated PWM output until its clock source begins its next cycle due to the synchronization of PWMEx and the clock source.

**NOTE**

The first PWM cycle after enabling the channel can be irregular.

An exception to this is when channels are concatenated. Once concatenated mode is enabled (CONxx bits set in PWMCTL register), enabling/disabling the corresponding 16-bit PWM channel is controlled by the low order PWMEx bit. In this case, the high order bytes PWMEx bits have no effect and their corresponding PWM output lines are disabled.

While in run mode, if all existing PWM channels are disabled (PWMEx–0 = 0), the prescaler counter shuts off for power savings.

Module Base + 0x0000

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | PWME7 | PWME6 | PWME5 | PWME4 | PWME3 | PWME2 | PWME1 | PWME0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-3. PWM Enable Register (PWME)**

Read: Anytime

Write: Anytime

**Table 9-2. PWME Field Descriptions**

**Note:** Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

| Field | Description |
|---|---|
| 7<br>PWME7 | **Pulse Width Channel 7 Enable**<br>0 Pulse width channel 7 is disabled.<br>1 Pulse width channel 7 is enabled. The pulse modulated signal becomes available at PWM output bit 7 when its clock source begins its next cycle. |
| 6<br>PWME6 | **Pulse Width Channel 6 Enable**<br>0 Pulse width channel 6 is disabled.<br>1 Pulse width channel 6 is enabled. The pulse modulated signal becomes available at PWM output bit 6 when its clock source begins its next cycle. If CON67=1, then bit has no effect and PWM output line 6 is disabled. |
| 5<br>PWME5 | **Pulse Width Channel 5 Enable**<br>0 Pulse width channel 5 is disabled.<br>1 Pulse width channel 5 is enabled. The pulse modulated signal becomes available at PWM output bit 5 when its clock source begins its next cycle. |
| 4<br>PWME4 | **Pulse Width Channel 4 Enable**<br>0 Pulse width channel 4 is disabled.<br>1 Pulse width channel 4 is enabled. The pulse modulated signal becomes available at PWM, output bit 4 when its clock source begins its next cycle. If CON45 = 1, then bit has no effect and PWM output line 4 is disabled. |
| 3<br>PWME3 | **Pulse Width Channel 3 Enable**<br>0 Pulse width channel 3 is disabled.<br>1 Pulse width channel 3 is enabled. The pulse modulated signal becomes available at PWM, output bit 3 when its clock source begins its next cycle. |
| 2<br>PWME2 | **Pulse Width Channel 2 Enable**<br>0 Pulse width channel 2 is disabled.<br>1 Pulse width channel 2 is enabled. The pulse modulated signal becomes available at PWM, output bit 2 when its clock source begins its next cycle. If CON23 = 1, then bit has no effect and PWM output line 2 is disabled. |
| 1<br>PWME1 | **Pulse Width Channel 1 Enable**<br>0 Pulse width channel 1 is disabled.<br>1 Pulse width channel 1 is enabled. The pulse modulated signal becomes available at PWM, output bit 1 when its clock source begins its next cycle. |
| 0<br>PWME0 | **Pulse Width Channel 0 Enable**<br>0 Pulse width channel 0 is disabled.<br>1 Pulse width channel 0 is enabled. The pulse modulated signal becomes available at PWM, output bit 0 when its clock source begins its next cycle. If CON01 = 1, then bit has no effect and PWM output line 0 is disabled. |

### 9.3.2.2 PWM Polarity Register (PWMPOL)

The starting polarity of each PWM channel waveform is determined by the associated PPOLx bit in the PWMPOL register. If the polarity bit is one, the PWM channel output is high at the beginning of the cycle and then goes low when the duty count is reached. Conversely, if the polarity bit is zero, the output starts low and then goes high when the duty count is reached.

Module Base + 0x0001

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | PPOL7 | PPOL6 | PPOL5 | PPOL4 | PPOL3 | PPOL2 | PPOL1 | PPOL0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-4. PWM Polarity Register (PWMPOL)**

Read: Anytime

Write: Anytime

**NOTE**

PPOLx register bits can be written anytime. If the polarity is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition

**Table 9-3. PWMPOL Field Descriptions**

**Note:** Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

| Field | Description |
|---|---|
| 7–0<br>PPOL[7:0] | **Pulse Width Channel 7–0 Polarity Bits**<br>0 PWM channel 7–0 outputs are low at the beginning of the period, then go high when the duty count is reached.<br>1 PWM channel 7–0 outputs are high at the beginning of the period, then go low when the duty count is reached. |

### 9.3.2.3 PWM Clock Select Register (PWMCLK)

Each PWM channel has a choice of four clocks to use as the clock source for that channel as described below.

Module Base + 0x0002

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | PCLK7 | PCLKL6 | PCLK5 | PCLK4 | PCLK3 | PCLK2 | PCLK1 | PCLK0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-5. PWM Clock Select Register (PWMCLK)**

Read: Anytime

Write: Anytime

**NOTE**

Register bits PCLK0 to PCLK7 can be written anytime. If a clock select is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

**Table 9-4. PWMCLK Field Descriptions**

**Note:** Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

| Field | Description |
|---|---|
| 7-0<br>PCLK[7:0] | **Pulse Width Channel 7-0 Clock Select**<br>0 Clock A or B is the clock source for PWM channel 7-0, as shown in Table 9-5 and Table 9-6.<br>1 Clock SA or SB is the clock source for PWM channel 7-0, as shown in Table 9-5 and Table 9-6. |

The clock source of each PWM channel is determined by PCLKx bits in PWMCLK and PCLKABx bits in PWMCLKAB (see Section 9.3.2.7, “PWM Clock A/B Select Register (PWMCLKAB)). For Channel 0, 1, 4, 5, the selection is shown in Table 9-5; For Channel 2, 3, 6, 7, the selection is shown in Table 9-6.

**Table 9-5. PWM Channel 0, 1, 4, 5 Clock Source Selection**

| PCLKAB[0,1,4,5] | PCLK[0,1,4,5] | Clock Source Selection |
|---|---|---|
| 0 | 0 | Clock A |
| 0 | 1 | Clock SA |
| 1 | 0 | Clock B |
| 1 | 1 | Clock SB |

**Table 9-6. PWM Channel 2, 3, 6, 7 Clock Source Selection**

| PCLKAB[2,3,6,7] | PCLK[2,3,6,7] | Clock Source Selection |
|---|---|---|
| 0 | 0 | Clock B |
| 0 | 1 | Clock SB |
| 1 | 0 | Clock A |
| 1 | 1 | Clock SA |

### 9.3.2.4 PWM Prescale Clock Select Register (PWMPRCLK)

This register selects the prescale clock source for clocks A and B independently.

Module Base + 0x0003

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | PCKB2 | PCKB1 | PCKB0 | 0 | PCKA2 | PCKA1 | PCKA0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 9-6. PWM Prescale Clock Select Register (PWMPRCLK)**

Read: Anytime

Write: Anytime

**NOTE**

PCKB2–0 and PCKA2–0 register bits can be written anytime. If the clock pre-scale is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

**Table 9-7. PWMPRCLK Field Descriptions**

| Field | Description |
|---|---|
| 6–4<br>PCKB[2:0] | **Prescaler Select for Clock B** — Clock B is one of two clock sources which can be used for all channels. These three bits determine the rate of clock B, as shown in Table 9-8. |
| 2–0<br>PCKA[2:0] | **Prescaler Select for Clock A** — Clock A is one of two clock sources which can be used for all channels. These three bits determine the rate of clock A, as shown in Table 9-8. |

**Table 9-8. Clock A or Clock B Prescaler Selects**

| PCKA/B2 | PCKA/B1 | PCKA/B0 | Value of Clock A/B |
|---|---|---|---|
| 0 | 0 | 0 | bus clock |
| 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 |

### 9.3.2.5 PWM Center Align Enable Register (PWMCAE)

The PWMCAE register contains eight control bits for the selection of center aligned outputs or left aligned outputs for each PWM channel. If the CAEx bit is set to a one, the corresponding PWM output will be center aligned. If the CAEx bit is cleared, the corresponding PWM output will be left aligned. See Section 9.4.2.5, “Left Aligned Outputs” and Section 9.4.2.6, “Center Aligned Outputs” for a more detailed description of the PWM output modes.

Module Base + 0x0004

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | CAE7 | CAE6 | CAE5 | CAE4 | CAE3 | CAE2 | CAE1 | CAE0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-7. PWM Center Align Enable Register (PWMCAE)**

Read: Anytime

Write: Anytime

**NOTE**

Write these bits only when the corresponding channel is disabled.

**Table 9-9. PWMCAE Field Descriptions**

**Note:** Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

| Field | Description |
|---|---|
| 7–0<br>CAE[7:0] | **Center Aligned Output Modes on Channels 7–0**<br>0 Channels 7–0 operate in left aligned output mode.<br>1 Channels 7–0 operate in center aligned output mode. |

### 9.3.2.6 PWM Control Register (PWMCTL)

The PWMCTL register provides for various control of the PWM module.

Module Base + 0x0005

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | CON67 | CON45 | CON23 | CON01 | PSWAI | PFRZ | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 9-8. PWM Control Register (PWMCTL)**

Read: Anytime

Write: Anytime

There are up to four control bits for concatenation, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. If the corresponding channels do not exist on a particular derivative, then writes to these bits have no effect and reads will return zeroes. When channels 6 and 7are concatenated, channel 6 registers become the high order bytes of the double byte channel. When channels 4 and 5 are concatenated, channel 4 registers become the high order bytes of the double byte channel. When channels 2 and 3 are concatenated, channel 2 registers become the high order bytes of the double byte channel. When channels 0 and 1 are concatenated, channel 0 registers become the high order bytes of the double byte channel.

See Section 9.4.2.7, “PWM 16-Bit Functions” for a more detailed description of the concatenation PWM Function.

**NOTE**

Change these bits only when both corresponding channels are disabled.

**Table 9-10. PWMCTL Field Descriptions**

**Note:** Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

| Field | Description |
|---|---|
| 7<br>CON67 | **Concatenate Channels 6 and 7**<br>0 Channels 6 and 7 are separate 8-bit PWMs.<br>1 Channels 6 and 7 are concatenated to create one 16-bit PWM channel. Channel 6 becomes the high order byte and channel 7 becomes the low order byte. Channel 7 output pin is used as the output for this 16-bit PWM (bit 7 of port PWMP). Channel 7 clock select control-bit determines the clock source, channel 7 polarity bit determines the polarity, channel 7 enable bit enables the output and channel 7 center aligned enable bit determines the output mode. |
| 6<br>CON45 | **Concatenate Channels 4 and 5**<br>0 Channels 4 and 5 are separate 8-bit PWMs.<br>1 Channels 4 and 5 are concatenated to create one 16-bit PWM channel. Channel 4 becomes the high order byte and channel 5 becomes the low order byte. Channel 5 output pin is used as the output for this 16-bit PWM (bit 5 of port PWMP). Channel 5 clock select control-bit determines the clock source, channel 5 polarity bit determines the polarity, channel 5 enable bit enables the output and channel 5 center aligned enable bit determines the output mode. |
| 5<br>CON23 | **Concatenate Channels 2 and 3**<br>0 Channels 2 and 3 are separate 8-bit PWMs.<br>1 Channels 2 and 3 are concatenated to create one 16-bit PWM channel. Channel 2 becomes the high order byte and channel 3 becomes the low order byte. Channel 3 output pin is used as the output for this 16-bit PWM (bit 3 of port PWMP). Channel 3 clock select control-bit determines the clock source, channel 3 polarity bit determines the polarity, channel 3 enable bit enables the output and channel 3 center aligned enable bit determines the output mode. |
| 4<br>CON01 | **Concatenate Channels 0 and 1**<br>0 Channels 0 and 1 are separate 8-bit PWMs.<br>1 Channels 0 and 1 are concatenated to create one 16-bit PWM channel. Channel 0 becomes the high order byte and channel 1 becomes the low order byte. Channel 1 output pin is used as the output for this 16-bit PWM (bit 1 of port PWMP). Channel 1 clock select control-bit determines the clock source, channel 1 polarity bit determines the polarity, channel 1 enable bit enables the output and channel 1 center aligned enable bit determines the output mode. |
| 3<br>PSWAI | **PWM Stops in Wait Mode —** Enabling this bit allows for lower power consumption in wait mode by disabling the input clock to the prescaler.<br>0 Allow the clock to the prescaler to continue while in wait mode.<br>1 Stop the input clock to the prescaler whenever the MCU is in wait mode. |
| 2<br>PFRZ | **PWM Counters Stop in Freeze Mode** — In freeze mode, there is an option to disable the input clock to the prescaler by setting the PFRZ bit in the PWMCTL register. If this bit is set, whenever the MCU is in freeze mode, the input clock to the prescaler is disabled. This feature is useful during emulation as it allows the PWM function to be suspended. In this way, the counters of the PWM can be stopped while in freeze mode so that once normal program flow is continued, the counters are re-enabled to simulate real-time operations. Since the registers can still be accessed in this mode, to re-enable the prescaler clock, either disable the PFRZ bit or exit freeze mode.<br>0 Allow PWM to continue while in freeze mode.<br>1 Disable PWM input clock to the prescaler whenever the part is in freeze mode. This is useful for emulation. |

### 9.3.2.7 PWM Clock A/B Select Register (PWMCLKAB)

Each PWM channel has a choice of four clocks to use as the clock source for that channel as described below.

Module Base + 0x00006

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | PCLKAB7 | PCLKAB6 | PCLKAB5 | PCLKAB4 | PCLKAB3 | PCLKAB2 | PCLKAB1 | PCLKAB0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-9. PWM Clock Select Register (PWMCLK)**

Read: Anytime

Write: Anytime

**NOTE**

Register bits PCLKAB0 to PCLKAB7 can be written anytime. If a clock select is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

**Table 9-11. PWMCLK Field Descriptions**

**Note:** Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

| Field | Description |
|---|---|
| 7<br>PCLKAB7 | **Pulse Width Channel 7 Clock A/B Select**<br>0 Clock B or SB is the clock source for PWM channel 7, as shown in Table 9-6.<br>1 Clock A or SA is the clock source for PWM channel 7, as shown in Table 9-6. |
| 6<br>PCLKAB6 | **Pulse Width Channel 6 Clock A/B Select**<br>0 Clock B or SB is the clock source for PWM channel 6, as shown in Table 9-6.<br>1 Clock A or SA is the clock source for PWM channel 6, as shown in Table 9-6. |
| 5<br>PCLKAB5 | **Pulse Width Channel 5 Clock A/B Select**<br>0 Clock A or SA is the clock source for PWM channel 5, as shown in Table 9-5.<br>1 Clock B or SB is the clock source for PWM channel 5, as shown in Table 9-5. |
| 4<br>PCLKAB4 | **Pulse Width Channel 4 Clock A/B Select**<br>0 Clock A or SA is the clock source for PWM channel 4, as shown in Table 9-5.<br>1 Clock B or SB is the clock source for PWM channel 4, as shown in Table 9-5. |
| 3<br>PCLKAB3 | **Pulse Width Channel 3 Clock A/B Select**<br>0 Clock B or SB is the clock source for PWM channel 3, as shown in Table 9-6.<br>1 Clock A or SA is the clock source for PWM channel 3, as shown in Table 9-6. |
| 2<br>PCLKAB2 | **Pulse Width Channel 2 Clock A/B Select**<br>0 Clock B or SB is the clock source for PWM channel 2, as shown in Table 9-6.<br>1 Clock A or SA is the clock source for PWM channel 2, as shown in Table 9-6. |
| 1<br>PCLKAB1 | **Pulse Width Channel 1 Clock A/B Select**<br>0 Clock A or SA is the clock source for PWM channel 1, as shown in Table 9-5.<br>1 Clock B or SB is the clock source for PWM channel 1, as shown in Table 9-5. |
| 0<br>PCLKAB0 | **Pulse Width Channel 0 Clock A/B Select**<br>0 Clock A or SA is the clock source for PWM channel 0, as shown in Table 9-5.<br>1 Clock B or SB is the clock source for PWM channel 0, as shown in Table 9-5. |

The clock source of each PWM channel is determined by PCLKx bits in PWMCLK (see Section 9.3.2.3, “PWM Clock Select Register (PWMCLK)) and PCLKABx bits in PWMCLKAB as shown in Table 9-5 and Table 9-6.

### 9.3.2.8 PWM Scale A Register (PWMSCLA)

PWMSCLA is the programmable scale value used in scaling clock A to generate clock SA. Clock SA is generated by taking clock A, dividing it by the value in the PWMSCLA register and dividing that by two.

Clock SA = Clock A / (2 * PWMSCLA)

**NOTE**

When PWMSCLA = $00, PWMSCLA value is considered a full scale value of 256. Clock A is thus divided by 512.

Any value written to this register will cause the scale counter to load the new scale value (PWMSCLA).

Module Base + 0x0008

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-10. PWM Scale A Register (PWMSCLA)**

Read: Anytime

Write: Anytime (causes the scale counter to load the PWMSCLA value)

### 9.3.2.9 PWM Scale B Register (PWMSCLB)

PWMSCLB is the programmable scale value used in scaling clock B to generate clock SB. Clock SB is generated by taking clock B, dividing it by the value in the PWMSCLB register and dividing that by two.

Clock SB = Clock B / (2 * PWMSCLB)

**NOTE**

When PWMSCLB = $00, PWMSCLB value is considered a full scale value of 256. Clock B is thus divided by 512.

Any value written to this register will cause the scale counter to load the new scale value (PWMSCLB).

Module Base + 0x0009

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-11. PWM Scale B Register (PWMSCLB)**

Read: Anytime

Write: Anytime (causes the scale counter to load the PWMSCLB value).

### 9.3.2.10 PWM Channel Counter Registers (PWMCNTx)

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source. The counter can be read at any time without affecting the count or the operation of the PWM channel. In left aligned output mode, the counter counts from 0 to the value in the period register - 1. In center aligned output mode, the counter counts from 0 up to the value in the period register and then back down to 0.

Any value written to the counter causes the counter to reset to $00, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. The counter is also cleared at the end of the effective period (see Section 9.4.2.5, “Left Aligned Outputs” and Section 9.4.2.6, “Center Aligned Outputs” for more details). When the channel is disabled (PWMEx = 0), the PWMCNTx register does not count. When a channel becomes enabled (PWMEx = 1), the associated PWM counter starts at the count in the PWMCNTx register. For more detailed information on the operation of the counters, see Section 9.4.2.4, “PWM Timer Counters”.

In concatenated mode, writes to the 16-bit counter by using a 16-bit access or writes to either the low or high order byte of the counter will reset the 16-bit counter. Reads of the 16-bit counter must be made by 16-bit access to maintain data coherency.

**NOTE**

Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.

Module Base + 0x000C = PWMCNT0, 0x000D = PWMCNT1, 0x000E = PWMCNT2, 0x000F = PWMCNT3
Module Base + 0x0010 = PWMCNT4, 0x0011 = PWMCNT5, 0x0012 = PWMCNT6, 0x0013 = PWMCNT7

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 9-12. PWM Channel Counter Registers (PWMCNTx)**

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Read: Anytime

Write: Anytime (any value written causes PWM counter to be reset to $00).

### 9.3.2.11 PWM Channel Period Registers (PWMPERx)

There is a dedicated period register for each channel. The value in this register determines the period of the associated PWM channel.

The period registers for each channel are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends

- The counter is written (counter resets to $00)
- The channel is disabled

In this way, the output of the PWM will always be either the old waveform or the new waveform, not some variation in between. If the channel is not enabled, then writes to the period register will go directly to the latches as well as the buffer.

**NOTE**

Reads of this register return the most recent value written. Reads do not necessarily return the value of the currently active period due to the double buffering scheme.

See Section 9.4.2.3, "PWM Period and Duty" for more information.

To calculate the output period, take the selected clock source period for the channel of interest (A, B, SA, or SB) and multiply it by the value in the period register for that channel:

- Left aligned output (CAEx = 0)

  PWMx Period = Channel Clock Period * PWMPERx
- Center Aligned Output (CAEx = 1)

  PWMx Period = Channel Clock Period * (2 * PWMPERx)

For boundary case programming values, please refer to Section 9.4.2.8, "PWM Boundary Cases".

Module Base + 0x0014 = PWMPER0, 0x0015 = PWMPER1, 0x0016 = PWMPER2, 0x0017 = PWMPER3
Module Base + 0x0018 = PWMPER4, 0x0019 = PWMPER5, 0x001A = PWMPER6, 0x001B = PWMPER7

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figure 9-13. PWM Channel Period Registers (PWMPERx)**

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Read: Anytime

Write: Anytime

### 9.3.2.12 PWM Channel Duty Registers (PWMDTYx)

There is a dedicated duty register for each channel. The value in this register determines the duty of the associated PWM channel. The duty value is compared to the counter and if it is equal to the counter value a match occurs and the output changes state.

The duty registers for each channel are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to $00)

- The channel is disabled

In this way, the output of the PWM will always be either the old duty waveform or the new duty waveform, not some variation in between. If the channel is not enabled, then writes to the duty register will go directly to the latches as well as the buffer.

**NOTE**

Reads of this register return the most recent value written. Reads do not necessarily return the value of the currently active duty due to the double buffering scheme.

See Section 9.4.2.3, "PWM Period and Duty" for more information.

**NOTE**

Depending on the polarity bit, the duty registers will contain the count of either the high time or the low time. If the polarity bit is one, the output starts high and then goes low when the duty count is reached, so the duty registers contain a count of the high time. If the polarity bit is zero, the output starts low and then goes high when the duty count is reached, so the duty registers contain a count of the low time.

To calculate the output duty cycle (high time as a% of period) for a particular channel:

- Polarity = 0 (PPOL x =0)

  Duty Cycle = [(PWMPERx-PWMDTYx)/PWMPERx] * 100%
- Polarity = 1 (PPOLx = 1)

  Duty Cycle = [PWMDTYx / PWMPERx] * 100%

For boundary case programming values, please refer to Section 9.4.2.8, "PWM Boundary Cases".

Module Base + 0x001C = PWMDTY0, 0x001D = PWMDTY1, 0x001E = PWMDTY2, 0x001F = PWMDTY3
Module Base + 0x0020 = PWMDTY4, 0x0021 = PWMDTY5, 0x0022 = PWMDTY6, 0x0023 = PWMDTY7

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figure 9-14. PWM Channel Duty Registers (PWMDTYx)**

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Read: Anytime

Write: Anytime

# 9.4 Functional Description

## 9.4.1 PWM Clock Select

There are four available clocks: clock A, clock B, clock SA (scaled A), and clock SB (scaled B). These four clocks are based on the bus clock.

Clock A and B can be software selected to be 1, 1/2, 1/4, 1/8,..., 1/64, 1/128 times the bus clock. Clock SA uses clock A as an input and divides it further with a reloadable counter. Similarly, clock SB uses clock B as an input and divides it further with a reloadable counter. The rates available for clock SA are software selectable to be clock A divided by 2, 4, 6, 8,..., or 512 in increments of divide by 2. Similar rates are available for clock SB. Each PWM channel has the capability of selecting one of four clocks, clock A, Clock B, clock SA or clock SB.

The block diagram in Figure 9-15 shows the four different clocks and how the scaled clocks are created.

### 9.4.1.1 Prescale

The input clock to the PWM prescaler is the bus clock. It can be disabled whenever the part is in freeze mode by setting the PFRZ bit in the PWMCTL register. If this bit is set, whenever the MCU is in freeze mode (freeze mode signal active) the input clock to the prescaler is disabled. This is useful for emulation in order to freeze the PWM. The input clock can also be disabled when all available PWM channels are disabled (PWMEx-0 = 0). This is useful for reducing power by disabling the prescale counter.

Clock A and clock B are scaled values of the input clock. The value is software selectable for both clock A and clock B and has options of 1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, or 1/128 times the bus clock. The value selected for clock A is determined by the PCKA2, PCKA1, PCKA0 bits in the PWMPRCLK register. The value selected for clock B is determined by the PCKB2, PCKB1, PCKB0 bits also in the PWMPRCLK register.

### 9.4.1.2 Clock Scale

The scaled A clock uses clock A as an input and divides it further with a user programmable value and then divides this by 2. The scaled B clock uses clock B as an input and divides it further with a user programmable value and then divides this by 2. The rates available for clock SA are software selectable to be clock A divided by 2, 4, 6, 8,..., or 512 in increments of divide by 2. Similar rates are available for clock SB.

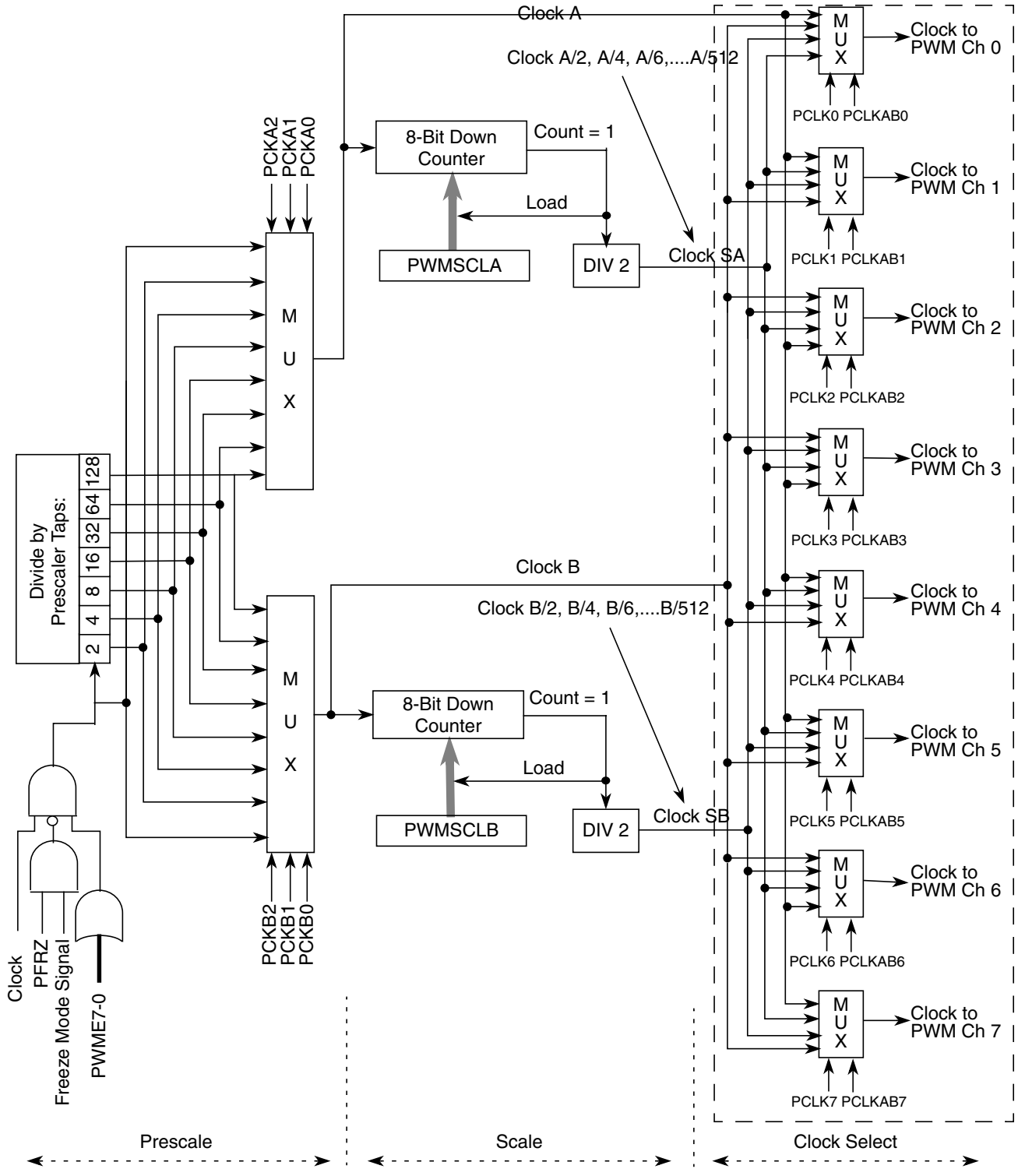



**Figure 9-15. PWM Clock Select Block Diagram**

Clock A is used as an input to an 8-bit down counter. This down counter loads a user programmable scale value from the scale register (PWMSCLA). When the down counter reaches one, a pulse is output and the 8-bit counter is re-loaded. The output signal from this circuit is further divided by two. This gives a greater range with only a slight reduction in granularity. Clock SA equals clock A divided by two times the value in the PWMSCLA register.

**NOTE**

Clock SA = Clock A / (2 * PWMSCLA)

When PWMSCLA = $00, PWMSCLA value is considered a full scale value of 256. Clock A is thus divided by 512.

Similarly, clock B is used as an input to an 8-bit down counter followed by a divide by two producing clock SB. Thus, clock SB equals clock B divided by two times the value in the PWMSCLB register.

**NOTE**

Clock SB = Clock B / (2 * PWMSCLB)

When PWMSCLB = $00, PWMSCLB value is considered a full scale value of 256. Clock B is thus divided by 512.

As an example, consider the case in which the user writes $FF into the PWMSCLA register. Clock A for this case will be bus clock divided by 4. A pulse will occur at a rate of once every 255x4 bus cycles. Passing this through the divide by two circuit produces a clock signal at an bus clock divided by 2040 rate. Similarly, a value of $01 in the PWMSCLA register when clock A is bus clock divided by 4 will produce a clock at an bus clock divided by 8 rate.

Writing to PWMSCLA or PWMSCLB causes the associated 8-bit down counter to be re-loaded. Otherwise, when changing rates the counter would have to count down to $01 before counting at the proper rate. Forcing the associated counter to re-load the scale register value every time PWMSCLA or PWMSCLB is written prevents this.

**NOTE**

Writing to the scale registers while channels are operating can cause irregularities in the PWM outputs.

### 9.4.1.3 Clock Select

Each PWM channel has the capability of selecting one of four clocks, clock A, clock SA, clock B or clock SB. The clock selection is done with the PCLKx control bits in the PWMCLK register and PCLKABx control bits in PWMCLKAB register. For backward compatibility consideration, the reset value of PWMCLK and PWMCLKAB configures following default clock selection.

For channels 0, 1, 4, and 5 the clock choices are clock A.

For channels 2, 3, 6, and 7 the clock choices are clock B.

**NOTE**

Changing clock control bits while channels are operating can cause irregularities in the PWM outputs.

### 9.4.2 PWM Channel Timers

The main part of the PWM module are the actual timers. Each of the timer channels has a counter, a period register and a duty register (each are 8-bit). The waveform output period is controlled by a match between the period register and the value in the counter. The duty is controlled by a match between the duty register and the counter value and causes the state of the output to change during the period. The starting polarity of the output is also selectable on a per channel basis. Shown below in Figure 9-16 is the block diagram for the PWM timer.

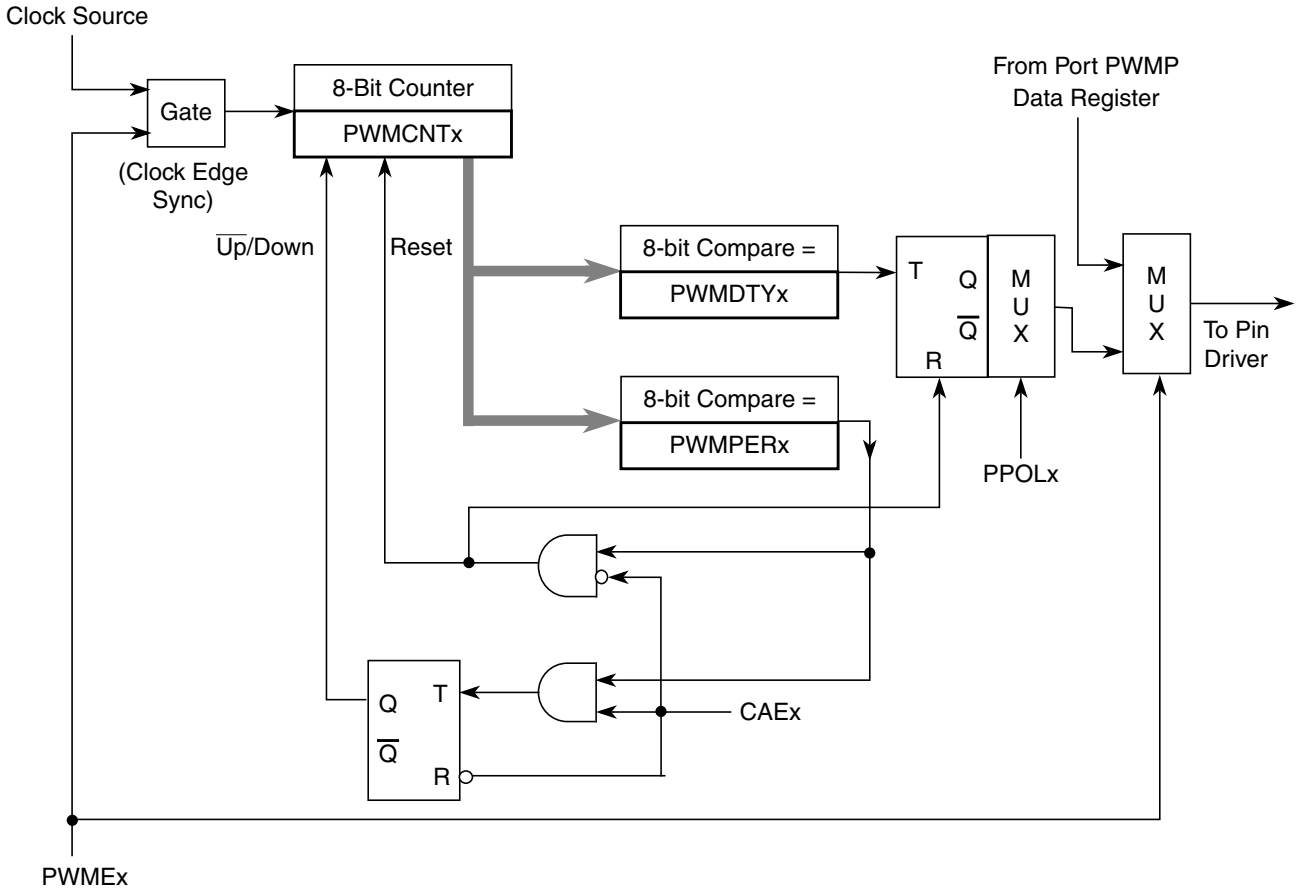



**Figure 9-16. PWM Timer Channel Block Diagram**

#### 9.4.2.1 PWM Enable

Each PWM channel has an enable bit (PWMEx) to start its waveform output. When any of the PWMEx bits are set (PWMEx = 1), the associated PWM output signal is enabled immediately. However, the actual PWM waveform is not available on the associated PWM output until its clock source begins its next cycle due to the synchronization of PWMEx and the clock source. An exception to this is when channels are concatenated. Refer to Section 9.4.2.7, “PWM 16-Bit Functions” for more detail.

**NOTE**

The first PWM cycle after enabling the channel can be irregular.

On the front end of the PWM timer, the clock is enabled to the PWM circuit by the PWMEx bit being high. There is an edge-synchronizing circuit to guarantee that the clock will only be enabled or disabled at an edge. When the channel is disabled (PWMEx = 0), the counter for the channel does not count.

### 9.4.2.2 PWM Polarity

Each channel has a polarity bit to allow starting a waveform cycle with a high or low signal. This is shown on the block diagram Figure 9-16 as a mux select of either the Q output or the $\overline{Q}$ output of the PWM output flip flop. When one of the bits in the PWMPOL register is set, the associated PWM channel output is high at the beginning of the waveform, then goes low when the duty count is reached. Conversely, if the polarity bit is zero, the output starts low and then goes high when the duty count is reached.

### 9.4.2.3 PWM Period and Duty

Dedicated period and duty registers exist for each channel and are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to $00)
- The channel is disabled

In this way, the output of the PWM will always be either the old waveform or the new waveform, not some variation in between. If the channel is not enabled, then writes to the period and duty registers will go directly to the latches as well as the buffer.

A change in duty or period can be forced into effect “immediately” by writing the new value to the duty and/or period registers and then writing to the counter. This forces the counter to reset and the new duty and/or period values to be latched. In addition, since the counter is readable, it is possible to know where the count is with respect to the duty value and software can be used to make adjustments

**NOTE**

When forcing a new period or duty into effect immediately, an irregular PWM cycle can occur.

Depending on the polarity bit, the duty registers will contain the count of either the high time or the low time.

### 9.4.2.4 PWM Timer Counters

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source (see Section 9.4.1, “PWM Clock Select” for the available clock sources and rates). The counter compares to two registers, a duty register and a period register as shown in Figure 9-16. When the PWM counter matches the duty register, the output flip-flop changes state, causing the PWM waveform to also change state. A match between the PWM counter and the period register behaves differently depending on what output mode is selected as shown in Figure 9-16 and described in Section 9.4.2.5, “Left Aligned Outputs” and Section 9.4.2.6, “Center Aligned Outputs”.

Each channel counter can be read at anytime without affecting the count or the operation of the PWM channel.

Any value written to the counter causes the counter to reset to $00, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. When the channel is disabled (PWMEx = 0), the counter stops. When a channel becomes enabled (PWMEx = 1), the associated PWM counter continues from the count in the PWMCNTx register. This allows the waveform to continue where it left off when the channel is re-enabled. When the channel is disabled, writing “0” to the period register will cause the counter to reset on the next selected clock.

**NOTE**

If the user wants to start a new “clean” PWM waveform without any “history” from the old waveform, the user must write to channel counter (PWMCNTx) prior to enabling the PWM channel (PWMEx = 1).

Generally, writes to the counter are done prior to enabling a channel in order to start from a known state. However, writing a counter can also be done while the PWM channel is enabled (counting). The effect is similar to writing the counter when the channel is disabled, except that the new period is started immediately with the output set according to the polarity bit.

**NOTE**

Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.

The counter is cleared at the end of the effective period (see Section 9.4.2.5, “Left Aligned Outputs” and Section 9.4.2.6, “Center Aligned Outputs” for more details).

**Table 9-12. PWM Timer Counter Conditions**

| Counter Clears ($00) | Counter Counts | Counter Stops |
|---|---|---|
| When PWMCNTx register written to any value | When PWM channel is enabled (PWMEx = 1). Counts from last value in PWMCNTx. | When PWM channel is disabled (PWMEx = 0) |
| Effective period ends | | |

### 9.4.2.5 Left Aligned Outputs

The PWM timer provides the choice of two types of outputs, left aligned or center aligned. They are selected with the CAEx bits in the PWMCAE register. If the CAEx bit is cleared (CAEx = 0), the corresponding PWM output will be left aligned.

In left aligned output mode, the 8-bit counter is configured as an up counter only. It compares to two registers, a duty register and a period register as shown in the block diagram in Figure 9-16. When the PWM counter matches the duty register the output flip-flop changes state causing the PWM waveform to also change state. A match between the PWM counter and the period register resets the counter and the output flip-flop, as shown in Figure 9-16, as well as performing a load from the double buffer period and duty register to the associated registers, as described in Section 9.4.2.3, “PWM Period and Duty”. The counter counts from 0 to the value in the period register – 1.

**NOTE**

Changing the PWM output mode from left aligned to center aligned output (or vice versa) while channels are operating can cause irregularities in the PWM output. It is recommended to program the output mode before enabling the PWM channel.

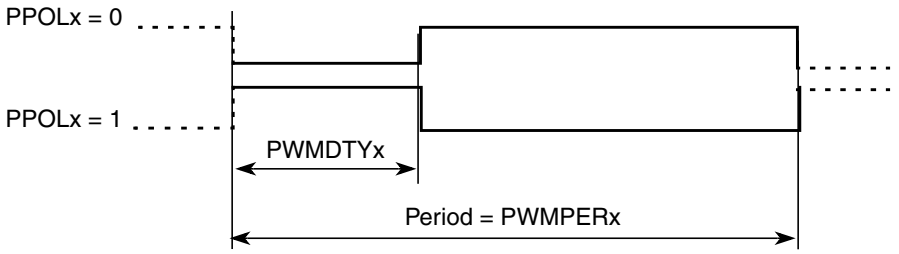



**Figure 9-17. PWM Left Aligned Output Waveform**

To calculate the output frequency in left aligned output mode for a particular channel, take the selected clock source frequency for the channel (A, B, SA, or SB) and divide it by the value in the period register for that channel.

- PWMx Frequency = Clock (A, B, SA, or SB) / PWMPERx
- PWMx Duty Cycle (high time as a% of period):
  - — Polarity = 0 (PPOLx = 0)
    Duty Cycle = [(PWMPERx-PWMDTYx)/PWMPERx] * 100%
  - — Polarity = 1 (PPOLx = 1)
    Duty Cycle = [PWMDTYx / PWMPERx] * 100%

As an example of a left aligned output, consider the following case:

Clock Source = bus clock, where bus clock = 10 MHz (100 ns period)
PPOLx = 0
PWMPERx = 4
PWMDTYx = 1
PWMx Frequency = 10 MHz/4 = 2.5 MHz
PWMx Period = 400 ns
PWMx Duty Cycle = 3/4 *100% = 75%

The output waveform generated is shown in Figure 9-18.

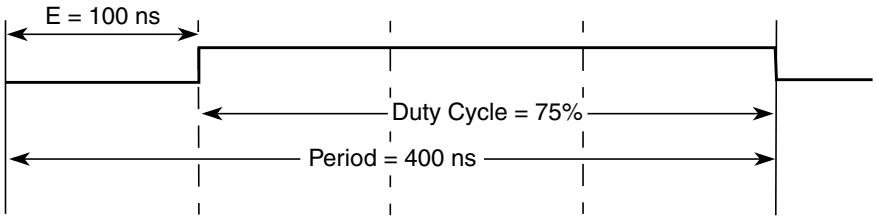



**Figure 9-18. PWM Left Aligned Output Example Waveform**

### 9.4.2.6 Center Aligned Outputs

For center aligned output mode selection, set the CAEx bit (CAEx = 1) in the PWMCAE register and the corresponding PWM output will be center aligned.

The 8-bit counter operates as an up/down counter in this mode and is set to up whenever the counter is equal to $00. The counter compares to two registers, a duty register and a period register as shown in the block diagram in Figure 9-16. When the PWM counter matches the duty register, the output flip-flop changes state, causing the PWM waveform to also change state. A match between the PWM counter and the period register changes the counter direction from an up-count to a down-count. When the PWM counter decrements and matches the duty register again, the output flip-flop changes state causing the PWM output to also change state. When the PWM counter decrements and reaches zero, the counter direction changes from a down-count back to an up-count and a load from the double buffer period and duty registers to the associated registers is performed, as described in Section 9.4.2.3, “PWM Period and Duty”. The counter counts from 0 up to the value in the period register and then back down to 0. Thus the effective period is PWMPERx*2.

**NOTE**

Changing the PWM output mode from left aligned to center aligned output (or vice versa) while channels are operating can cause irregularities in the PWM output. It is recommended to program the output mode before enabling the PWM channel.

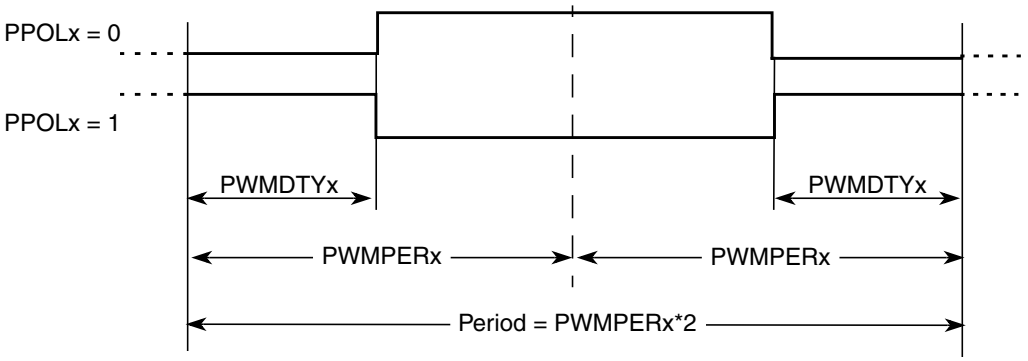



**Figure 9-19. PWM Center Aligned Output Waveform**

To calculate the output frequency in center aligned output mode for a particular channel, take the selected clock source frequency for the channel (A, B, SA, or SB) and divide it by twice the value in the period register for that channel.

- PWMx Frequency = Clock (A, B, SA, or SB) / (2*PWMPERx)
- PWMx Duty Cycle (high time as a% of period):
  - — Polarity = 0 (PPOLx = 0)
    Duty Cycle = [(PWMPERx-PWMDTYx)/PWMPERx] * 100%
  - — Polarity = 1 (PPOLx = 1)
    Duty Cycle = [PWMDTYx / PWMPERx] * 100%

As an example of a center aligned output, consider the following case:

Clock Source = bus clock, where bus clock= 10 MHz (100 ns period)
PPOLx = 0
PWMPERx = 4
PWMDTYx = 1
PWMx Frequency = 10 MHz/8 = 1.25 MHz
PWMx Period = 800 ns
PWMx Duty Cycle = 3/4 *100% = 75%

Shown in Figure 9-20 is the output waveform generated.

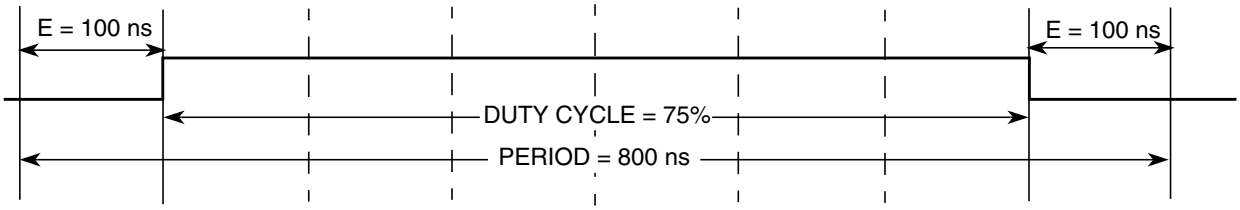



**Figure 9-20. PWM Center Aligned Output Example Waveform**

### 9.4.2.7 PWM 16-Bit Functions

The scalable PWM timer also has the option of generating up to 8-channels of 8-bits or 4-channels of 16-bits for greater PWM resolution. This 16-bit channel option is achieved through the concatenation of two 8-bit channels.

The PWMCTL register contains four control bits, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. Channels 6 and 7 are concatenated with the CON67 bit, channels 4 and 5 are concatenated with the CON45 bit, channels 2 and 3 are concatenated with the CON23 bit, and channels 0 and 1 are concatenated with the CON01 bit.

**NOTE**

Change these bits only when both corresponding channels are disabled.

When channels 6 and 7 are concatenated, channel 6 registers become the high order bytes of the double byte channel, as shown in Figure 9-21. Similarly, when channels 4 and 5 are concatenated, channel 4

registers become the high order bytes of the double byte channel. When channels 2 and 3 are concatenated, channel 2 registers become the high order bytes of the double byte channel. When channels 0 and 1 are concatenated, channel 0 registers become the high order bytes of the double byte channel.

When using the 16-bit concatenated mode, the clock source is determined by the low order 8-bit channel clock select control bits. That is channel 7 when channels 6 and 7 are concatenated, channel 5 when channels 4 and 5 are concatenated, channel 3 when channels 2 and 3 are concatenated, and channel 1 when channels 0 and 1 are concatenated. The resulting PWM is output to the pins of the corresponding low order 8-bit channel as also shown in Figure 9-21. The polarity of the resulting PWM output is controlled by the PPOLx bit of the corresponding low order 8-bit channel as well.

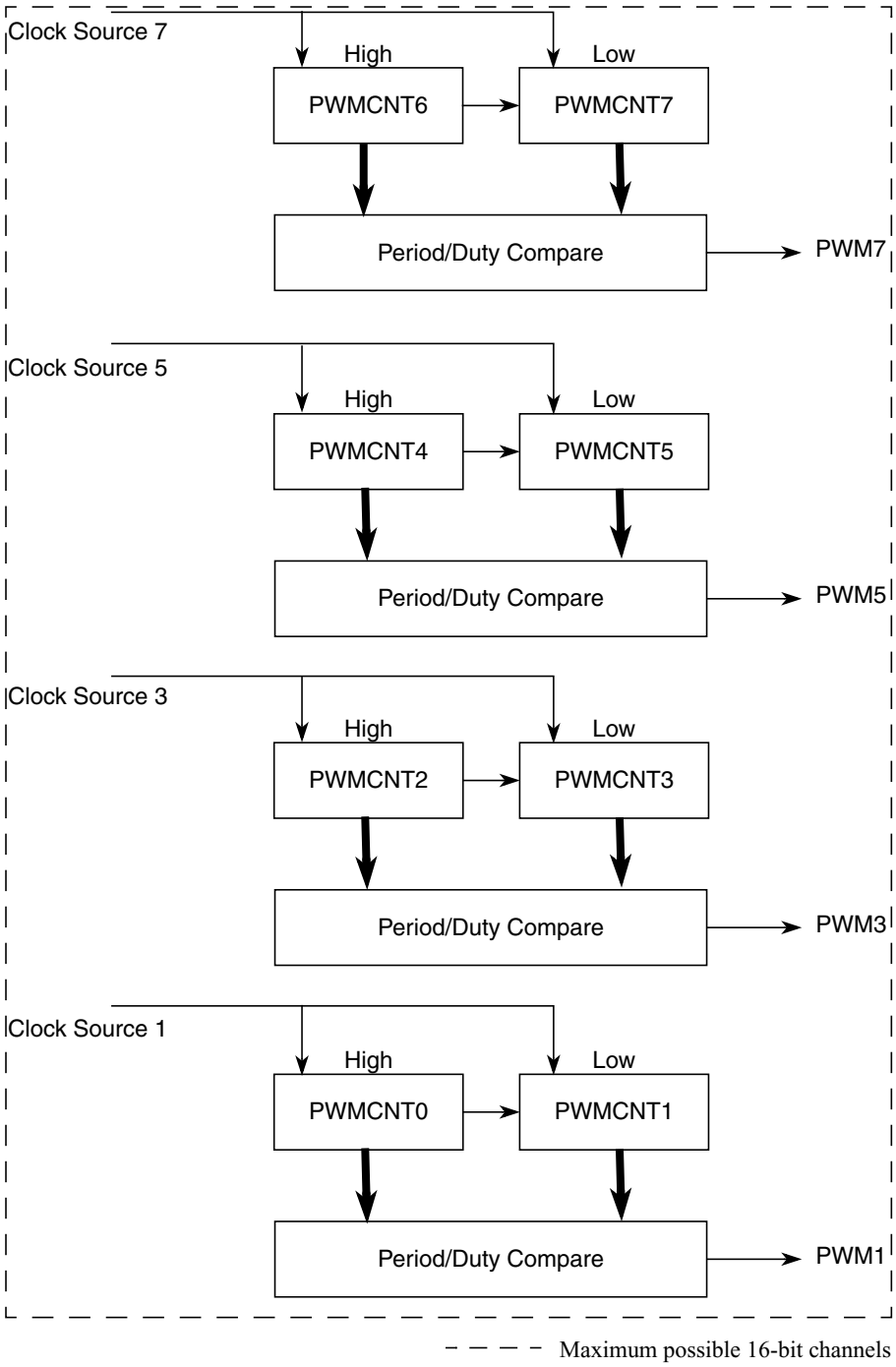



**Figure 9-21. PWM 16-Bit Mode**

Once concatenated mode is enabled (CONxx bits set in PWMCTL register), enabling/disabling the corresponding 16-bit PWM channel is controlled by the low order PWMEx bit. In this case, the high order bytes PWMEx bits have no effect and their corresponding PWM output is disabled.

In concatenated mode, writes to the 16-bit counter by using a 16-bit access or writes to either the low or high order byte of the counter will reset the 16-bit counter. Reads of the 16-bit counter must be made by 16-bit access to maintain data coherency.

Either left aligned or center aligned output mode can be used in concatenated mode and is controlled by the low order CAEx bit. The high order CAEx bit has no effect.

Table 9-13 is used to summarize which channels are used to set the various control bits when in 16-bit mode.

**Table 9-13. 16-bit Concatenation Mode Summary**

**Note:** Bits related to available channels have functional significance.

| CONxx | PWMEx | PPOLx | PCLKx | CAEx | PWMx Output |
|---|---|---|---|---|---|
| CON67 | PWME7 | PPOL7 | PCLK7 | CAE7 | PWM7 |
| CON45 | PWME5 | PPOL5 | PCLK5 | CAE5 | PWM5 |
| CON23 | PWME3 | PPOL3 | PCLK3 | CAE3 | PWM3 |
| CON01 | PWME1 | PPOL1 | PCLK1 | CAE1 | PWM1 |

### 9.4.2.8 PWM Boundary Cases

Table 9-14 summarizes the boundary conditions for the PWM regardless of the output mode (left aligned or center aligned) and 8-bit (normal) or 16-bit (concatenation).

**Table 9-14. PWM Boundary Cases**

| PWMDTYx | PWMPERx | PPOLx | PWMx Output |
|---|---|---|---|
| $00 (indicates no duty) | >$00 | 1 | Always low |
| $00 (indicates no duty) | >$00 | 0 | Always high |
| XX | $00[1] (indicates no period) | 1 | Always high |
| XX | $00[1] (indicates no period) | 0 | Always low |
| >= PWMPERx | XX | 1 | Always high |
| >= PWMPERx | XX | 0 | Always low |

[1] Counter = $00 and does not count.

## 9.5 Resets

The reset state of each individual bit is listed within the Section 9.3.2, “Register Descriptions” which details the registers and their bit-fields. All special functions or modes which are initialized during or just following reset are described within this section.

- The 8-bit up/down counter is configured as an up counter out of reset.
- All the channels are disabled and all the counters do not count.

- For channels 0, 1, 4, and 5 the clock choices are clock A.
- For channels 2, 3, 6, and 7 the clock choices are clock B.

## 9.6 Interrupts

The PWM module has no interrupt.

# Chapter 10
# Serial Communication Interface (S12SCIV6)

**Table 10-1. Revision History**

| Version Number | Revision Date | Effective Date | Author | Description of Changes |
|---|---|---|---|---|
| 06.05 | 02/22/2013 | | | fix typo Figure 10-1./10-303 Figure 10-4./10-306<br>update Table 10-2./10-306 10.4.4/10-320 10.4.6.3/10-327 |
| 06.06 | 03/11/2013 | | | fix typo of BDL reset value,Figure 10-4<br>fix typo of Table 10-2,Table 10-16,reword 10.4.4/10-320 |
| 06.07 | 09/03/2013 | | | update Figure 10-14./10-317 Figure 10-16./10-321 Figure 10-20./10-326<br>update 10.4.4/10-320,more detail for two baud<br>add note for Table 10-16./10-320<br>update Figure 10-2./10-305,Figure 10-12./10-316 |

## 10.1 Introduction

This block guide provides an overview of the serial communication interface (SCI) module.

The SCI allows asynchronous serial communications with peripheral devices and other CPUs.

### 10.1.1 Glossary

IR: InfraRed

IrDA: Infrared Design Associate

IRQ: Interrupt Request

LIN: Local Interconnect Network

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

### 10.1.2 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
- 16-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Programmable polarity for 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
  - Receive wakeup on active edge
  - Transmit collision detect supporting LIN
  - Break Detect supporting LIN
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection

### 10.1.3 Modes of Operation

The SCI functions the same in normal, special, and emulation modes. It has two low power modes, wait and stop modes.

- Run mode
- Wait mode
- Stop mode

## 10.1.4 Block Diagram

Figure 10-1 is a high level block diagram of the SCI module, showing the interaction of various function blocks.

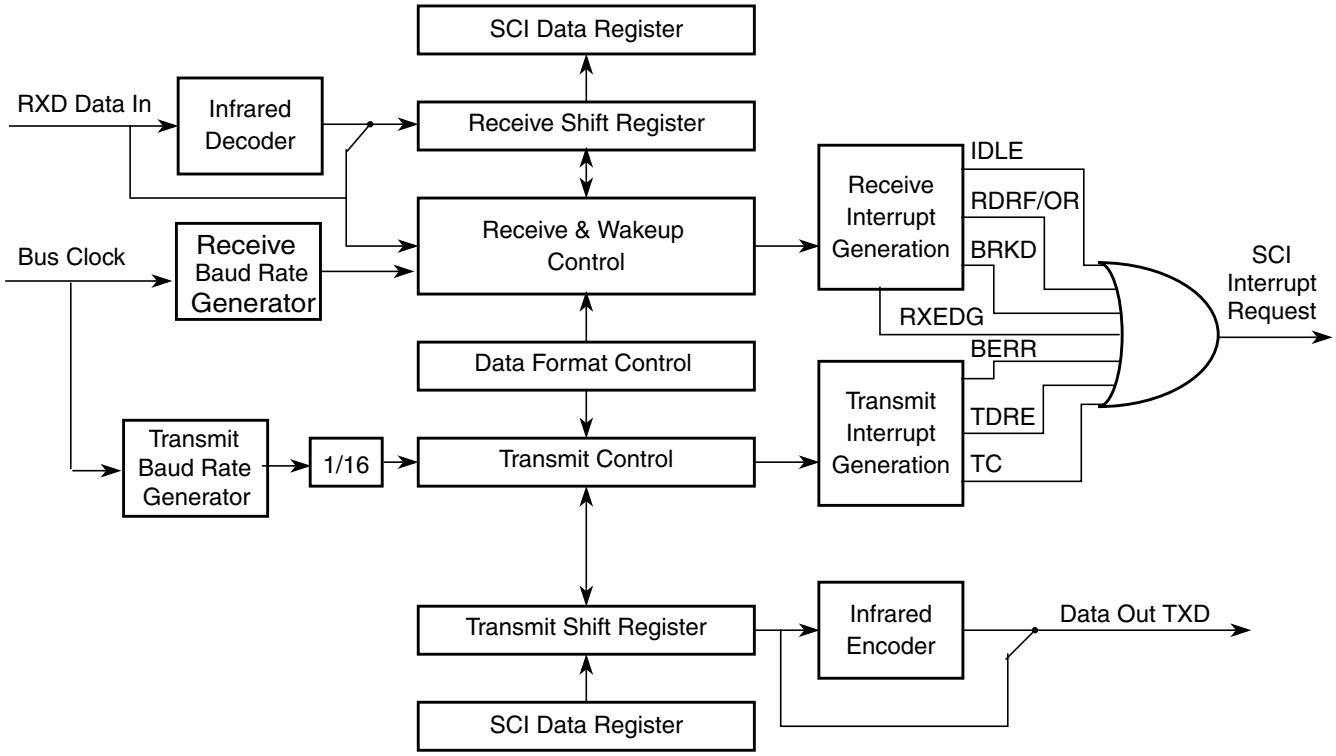



**Figure 10-1. SCI Block Diagram**

## 10.2 External Signal Description

The SCI module has a total of two external pins.

### 10.2.1 TXD — 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.

### 10.2.2 RXD — 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.

## 10.3 Memory Map and Register Definition

This section provides a detailed description of all the SCI registers.

### 10.3.1 Module Memory Map and Register Definition

The memory map for the SCI 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 SCI 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. Writes to a reserved register locations do not have any effect and reads of these locations return a zero. 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 SCIBDH[1] | R | SBR15 | SBR14 | SBR13 | SBR12 | SBR11 | SBR10 | SBR9 | SBR8 |
| | W | | | | | | | | |
| 0x0001 SCIBDL[1] | R | SBR7 | SBR6 | SBR5 | SBR4 | SBR3 | SBR2 | SBR1 | SBR0 |
| | W | | | | | | | | |
| 0x0002 SCICR1[1] | R | LOOPS | SCISWAI | RSRC | M | WAKE | ILT | PE | PT |
| | W | | | | | | | | |
| 0x0000 SCIASR1[2] | R | RXEDGIF | 0 | 0 | 0 | 0 | BERRV | BERRIF | BKDIF |
| | W | | | | | | | | |
| 0x0001 SCIACR1[2] | R | RXEDGIE | 0 | 0 | 0 | 0 | 0 | BERRIE | BKDIE |
| | W | | | | | | | | |
| 0x0002 SCIACR2[2] | R | IREN | TNP1 | TNP0 | 0 | 0 | BERRM1 | BERRM0 | BKDFE |
| | W | | | | | | | | |
| 0x0003 SCICR2 | R | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| | W | | | | | | | | |
| 0x0004 SCISR1 | R | TDRE | TC | RDRF | IDLE | OR | NF | FE | PF |
| | W | | | | | | | | |
| 0x0005 SCISR2 | R | AMAP | 0 | 0 | TXPOL | RXPOL | BRK13 | TXDIR | RAF |
| | W | | | | | | | | |
| 0x0006 SCIDRH | R | R8 | T8 | 0 | 0 | 0 | Reserved | Reserved | Reserved |
| | W | | | | | | | | |
| 0x0007 SCIDRL | R | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| | W | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |

1.These registers are accessible if the AMAP bit in the SCISR2 register is set to zero.

2,These registers are accessible if the AMAP bit in the SCISR2 register is set to one.

= Unimplemented or Reserved

**Figure 10-2. SCI Register Summary**

### 10.3.2.1 SCI Baud Rate Registers (SCIBDH, SCIBDL)

Module Base + 0x0000

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | SBR15 | SBR14 | SBR13 | SBR12 | SBR11 | SBR10 | SBR9 | SBR8 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 10-3. SCI Baud Rate Register (SCIBDH)**

Module Base + 0x0001

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | SBR7 | SBR6 | SBR5 | SBR4 | SBR3 | SBR2 | SBR1 | SBR0 |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Figure 10-4. SCI Baud Rate Register (SCIBDL)**

Read: Anytime, if AMAP = 0.

Write: Anytime, if AMAP = 0.

**NOTE**

Those two registers are only visible in the memory map if AMAP = 0 (reset condition).

The SCI baud rate register is used by to determine the baud rate of the SCI, and to control the infrared modulation/demodulation submodule.

**Table 10-2. SCIBDH and SCIBDL Field Descriptions**

| Field | Description |
|---|---|
| SBR[15:0] | **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.<br>The formulas for calculating the baud rate are:<br>When IREN = 0 then,<br>SCI baud rate = SCI bus clock / (SBR[15:0])<br>When IREN = 1 then,<br>SCI baud rate = SCI bus clock / (2 x SBR[15:1])<br>**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[15:4] = 0 and IREN = 0) or (SBR[15:5] = 0 and IREN = 1).<br>**Note:** . User should write SCIBD by word access. The updated SCIBD may take effect until next RT clock start, write SCIBDH or SCIBDL separately may cause baud generator load wrong data at that time,if second write later then RT clock. |

### 10.3.2.2 SCI Control Register 1 (SCICR1)

Module Base + 0x0002

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | LOOPS | SCISWAI | RSRC | M | WAKE | ILT | PE | PT |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 10-5. SCI Control Register 1 (SCICR1)**

Read: Anytime, if AMAP = 0.

Write: Anytime, if AMAP = 0.

**NOTE**

This register is only visible in the memory map if AMAP = 0 (reset condition).

**Table 10-3. SCICR1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>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.<br>0 Normal operation enabled<br>1 Loop operation enabled<br>The receiver input is determined by the RSRC bit. |
| 6<br>SCISWAI | **SCI Stop in Wait Mode Bit** — SCISWAI disables the SCI in wait mode.<br>0 SCI enabled in wait mode<br>1 SCI disabled in wait mode |
| 5<br>RSRC | **Receiver Source Bit** — When LOOPS = 1, the RSRC bit determines the source for the receiver shift register input. See Table 10-4.<br>0 Receiver input internally connected to transmitter output<br>1 Receiver input connected externally to transmitter |
| 4<br>M | **Data Format Mode Bit** — MODE determines whether data characters are eight or nine bits long.<br>0 One start bit, eight data bits, one stop bit<br>1 One start bit, nine data bits, one stop bit |
| 3<br>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.<br>0 Idle line wakeup<br>1 Address mark wakeup |
| 2<br>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.<br>0 Idle character bit count begins after start bit<br>1 Idle character bit count begins after stop bit |

**Table 10-3. SCICR1 Field Descriptions (continued)**

| Field | Description |
|---|---|
| 1<br>PE | **Parity Enable Bit** — PE enables the parity function. When enabled, the parity function inserts a parity bit in the most significant bit position.<br>0 Parity function disabled<br>1 Parity function enabled |
| 0<br>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.<br>0 Even parity<br>1 Odd parity |

**Table 10-4. 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 |

### 10.3.2.3 SCI Alternative Status Register 1 (SCIASR1)

Module Base + 0x0000

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | RXEDGIF | 0 | 0 | 0 | 0 | BERRV | BERRIF | BKDIF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 10-6. SCI Alternative Status Register 1 (SCIASR1)**

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

**Table 10-5. SCIASR1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>RXEDGIF | **Receive Input Active Edge Interrupt Flag** — RXEDGIF is asserted, if an active edge (falling if RXPOL = 0, rising if RXPOL = 1) on the RXD input occurs. RXEDGIF bit is cleared by writing a “1” to it.<br>0 No active receive on the receive input has occurred<br>1 An active edge on the receive input has occurred |
| 2<br>BERRV | **Bit Error Value** — BERRV reflects the state of the RXD input when the bit error detect circuitry is enabled and a mismatch to the expected value happened. The value is only meaningful, if BERRIF = 1.<br>0 A low input was sampled, when a high was expected<br>1 A high input reassembled, when a low was expected |
| 1<br>BERRIF | **Bit Error Interrupt Flag** — BERRIF is asserted, when the bit error detect circuitry is enabled and if the value sampled at the RXD input does not match the transmitted value. If the BERRIE interrupt enable bit is set an interrupt will be generated. The BERRIF bit is cleared by writing a “1” to it.<br>0 No mismatch detected<br>1 A mismatch has occurred |
| 0<br>BKDIF | **Break Detect Interrupt Flag** — BKDIF is asserted, if the break detect circuitry is enabled and a break signal is received. If the BKDIE interrupt enable bit is set an interrupt will be generated. The BKDIF bit is cleared by writing a “1” to it.<br>0 No break signal was received<br>1 A break signal was received |

### 10.3.2.4 SCI Alternative Control Register 1 (SCIACR1)

Module Base + 0x0001

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | RXEDGIE | 0 | 0 | 0 | 0 | 0 | BERRIE | BKDIE |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 10-7. SCI Alternative Control Register 1 (SCIACR1)**

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

**Table 10-6. SCIACR1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>RXEDGIE | **Receive Input Active Edge Interrupt Enable** — RXEDGIE enables the receive input active edge interrupt flag, RXEDGIF, to generate interrupt requests.<br>0 RXEDGIF interrupt requests disabled<br>1 RXEDGIF interrupt requests enabled |
| 1<br>BERRIE | **Bit Error Interrupt Enable** — BERRIE enables the bit error interrupt flag, BERRIF, to generate interrupt requests.<br>0 BERRIF interrupt requests disabled<br>1 BERRIF interrupt requests enabled |
| 0<br>BKDIE | **Break Detect Interrupt Enable** — BKDIE enables the break detect interrupt flag, BKDIF, to generate interrupt requests.<br>0 BKDIF interrupt requests disabled<br>1 BKDIF interrupt requests enabled |

### 10.3.2.5 SCI Alternative Control Register 2 (SCIACR2)

Module Base + 0x0002

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | IREN | TNP1 | TNP0 | 0 | 0 | BERRM1 | BERRM0 | BKDFE |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 10-8. SCI Alternative Control Register 2 (SCIACR2)**

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

**Table 10-7. SCIACR2 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>IREN | **Infrared Enable Bit** — This bit enables/disables the infrared modulation/demodulation submodule.<br>0 IR disabled<br>1 IR enabled |
| 6:5<br>TNP[1:0] | **Transmitter Narrow Pulse Bits** — These bits enable whether the SCI transmits a 1/16, 3/16, 1/32 or 1/4 narrow pulse. See Table 10-8. |
| 2:1<br>BERRM[1:0] | **Bit Error Mode** — Those two bits determines the functionality of the bit error detect feature. See Table 10-9. |
| 0<br>BKDFE | **Break Detect Feature Enable** — BKDFE enables the break detect circuitry.<br>0 Break detect circuit disabled<br>1 Break detect circuit enabled |

**Table 10-8. IRSCI Transmit Pulse Width**

| TNP[1:0] | Narrow Pulse Width |
|---|---|
| 11 | 1/4 |
| 10 | 1/32 |
| 01 | 1/16 |
| 00 | 3/16 |

**Table 10-9. Bit Error Mode Coding**

| BERRM1 | BERRM0 | Function |
|---|---|---|
| 0 | 0 | Bit error detect circuit is disabled |
| 0 | 1 | Receive input sampling occurs during the 9th time tick of a transmitted bit (refer to Figure 10-19) |
| 1 | 0 | Receive input sampling occurs during the 13th time tick of a transmitted bit (refer to Figure 10-19) |
| 1 | 1 | Reserved |

### 10.3.2.6 SCI Control Register 2 (SCICR2)

Module Base + 0x0003

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 10-9. SCI Control Register 2 (SCICR2)**

Read: Anytime

Write: Anytime

**Table 10-10. SCICR2 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>TIE | **Transmitter Interrupt Enable Bit** — TIE enables the transmit data register empty flag, TDRE, to generate interrupt requests.<br>0 TDRE interrupt requests disabled<br>1 TDRE interrupt requests enabled |
| 6<br>TCIE | **Transmission Complete Interrupt Enable Bit** — TCIE enables the transmission complete flag, TC, to generate interrupt requests.<br>0 TC interrupt requests disabled<br>1 TC interrupt requests enabled |
| 5<br>RIE | **Receiver Full Interrupt Enable Bit** — RIE enables the receive data register full flag, RDRF, or the overrun flag, OR, to generate interrupt requests.<br>0 RDRF and OR interrupt requests disabled<br>1 RDRF and OR interrupt requests enabled |
| 4<br>ILIE | **Idle Line Interrupt Enable Bit** — ILIE enables the idle line flag, IDLE, to generate interrupt requests.<br>0 IDLE interrupt requests disabled<br>1 IDLE interrupt requests enabled |
| 3<br>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.<br>0 Transmitter disabled<br>1 Transmitter enabled |
| 2<br>RE | **Receiver Enable Bit** — RE enables the SCI receiver.<br>0 Receiver disabled<br>1 Receiver enabled |
| 1<br>RWU | **Receiver Wakeup Bit** — Standby state<br>0 Normal operation.<br>1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU. |
| 0<br>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).<br>0 No break characters<br>1 Transmit break characters |

### 10.3.2.7 SCI Status Register 1 (SCISR1)

The SCISR1 and SCISR2 registers provides 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, but the order of operations is important for flag clearing.

Module Base + 0x0004

| | 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 10-10. SCI Status Register 1 (SCISR1)**

Read: Anytime

Write: Has no meaning or effect

**Table 10-11. SCISR1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>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).<br>0 No byte transferred to transmit shift register<br>1 Byte transferred to transmit shift register; transmit data register empty |
| 6<br>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).<br>0 Transmission in progress<br>1 No transmission in progress |
| 5<br>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).<br>0 Data not available in SCI data register<br>1 Received data available in SCI data register |
| 4<br>IDLE | **Idle Line Flag** — IDLE is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M =1) appear on the receiver input. Once 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).<br>0 Receiver input is either active now or has never become active since the IDLE flag was last cleared<br>1 Receiver input has become idle<br>**Note:** When the receiver wakeup bit (RWU) is set, an idle line condition does not set the IDLE flag. |

**Table 10-11. SCISR1 Field Descriptions (continued)**

| Field | Description |
|---|---|
| 3<br>OR | **Overrun Flag** — 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).<br>0 No overrun<br>1 Overrun<br>**Note:** OR flag may read back as set when RDRF flag is clear. This may happen if the following sequence of events occurs:<br>1. After the first frame is received, read status register SCISR1 (returns RDRF set and OR flag clear);<br>2. Receive second frame without reading the first frame in the data register (the second frame is not received and OR flag is set);<br>3. Read data register SCIDRL (returns first frame and clears RDRF flag in the status register);<br>4. Read status register SCISR1 (returns RDRF clear and OR set).<br>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. |
| 2<br>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).<br>0 No noise<br>1 Noise |
| 1<br>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).<br>0 No framing error<br>1 Framing error |
| 0<br>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).<br>0 No parity error<br>1 Parity error |

### 10.3.2.8 SCI Status Register 2 (SCISR2)

Module Base + 0x0005

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | AMAP | 0 | 0 | TXPOL | RXPOL | BRK13 | TXDIR | RAF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 10-11. SCI Status Register 2 (SCISR2)**

Read: Anytime

Write: Anytime

**Table 10-12. SCISR2 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>AMAP | **Alternative Map** — This bit controls which registers sharing the same address space are accessible. In the reset condition the SCI behaves as previous versions. Setting AMAP=1 allows the access to another set of control and status registers and hides the baud rate and SCI control Register 1.<br>0 The registers labelled SCIBDH (0x0000),SCIBDL (0x0001), SCICR1 (0x0002) are accessible<br>1 The registers labelled SCIASR1 (0x0000),SCIACR1 (0x0001), SCIACR2 (0x00002) are accessible |
| 4<br>TXPOL | **Transmit Polarity** — This bit control the polarity of the transmitted data. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity.<br>0 Normal polarity<br>1 Inverted polarity |
| 3<br>RXPOL | **Receive Polarity** — This bit control the polarity of the received data. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity.<br>0 Normal polarity<br>1 Inverted polarity |
| 2<br>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.<br>0 Break character is 10 or 11 bit long<br>1 Break character is 13 or 14 bit long |
| 1<br>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.<br>0 TXD pin to be used as an input in single-wire mode<br>1 TXD pin to be used as an output in single-wire mode |
| 0<br>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.<br>0 No reception in progress<br>1 Reception in progress |

### 10.3.2.9 SCI Data Registers (SCIDRH, SCIDRL)

Module Base + 0x0006

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | R8 | T8 | 0 | 0 | 0 | Reserved | Reserved | Reserved |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 10-12. SCI Data Registers (SCIDRH)**

Module Base + 0x0007

| | 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 10-13. SCI Data Registers (SCIDRL)**

Read: Anytime; reading accesses SCI receive data register

Write: Anytime; writing accesses SCI transmit data register; writing to R8 has no effect

**Table 10-13. SCIDRH and SCIDRL Field Descriptions**

| Field | Description |
|---|---|
| SCIDRH<br>7<br>R8 | **Received Bit 8** — R8 is the ninth data bit received when the SCI is configured for 9-bit data format (M = 1). |
| SCIDRH<br>6<br>T8 | **Transmit Bit 8** — T8 is the ninth data bit transmitted when the SCI is configured for 9-bit data format (M = 1). |
| SCIDRL<br>7:0<br>R[7:0]<br>T[7:0] | **R7:R0** — Received bits seven through zero for 9-bit or 8-bit data formats<br>**T7:T0** — Transmit bits seven through zero 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 SCIDRL.

## 10.4 Functional Description

This section provides a complete functional description of the SCI block, detailing the operation of the design from the end user perspective in a number of subsections.

Figure 10-14 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.

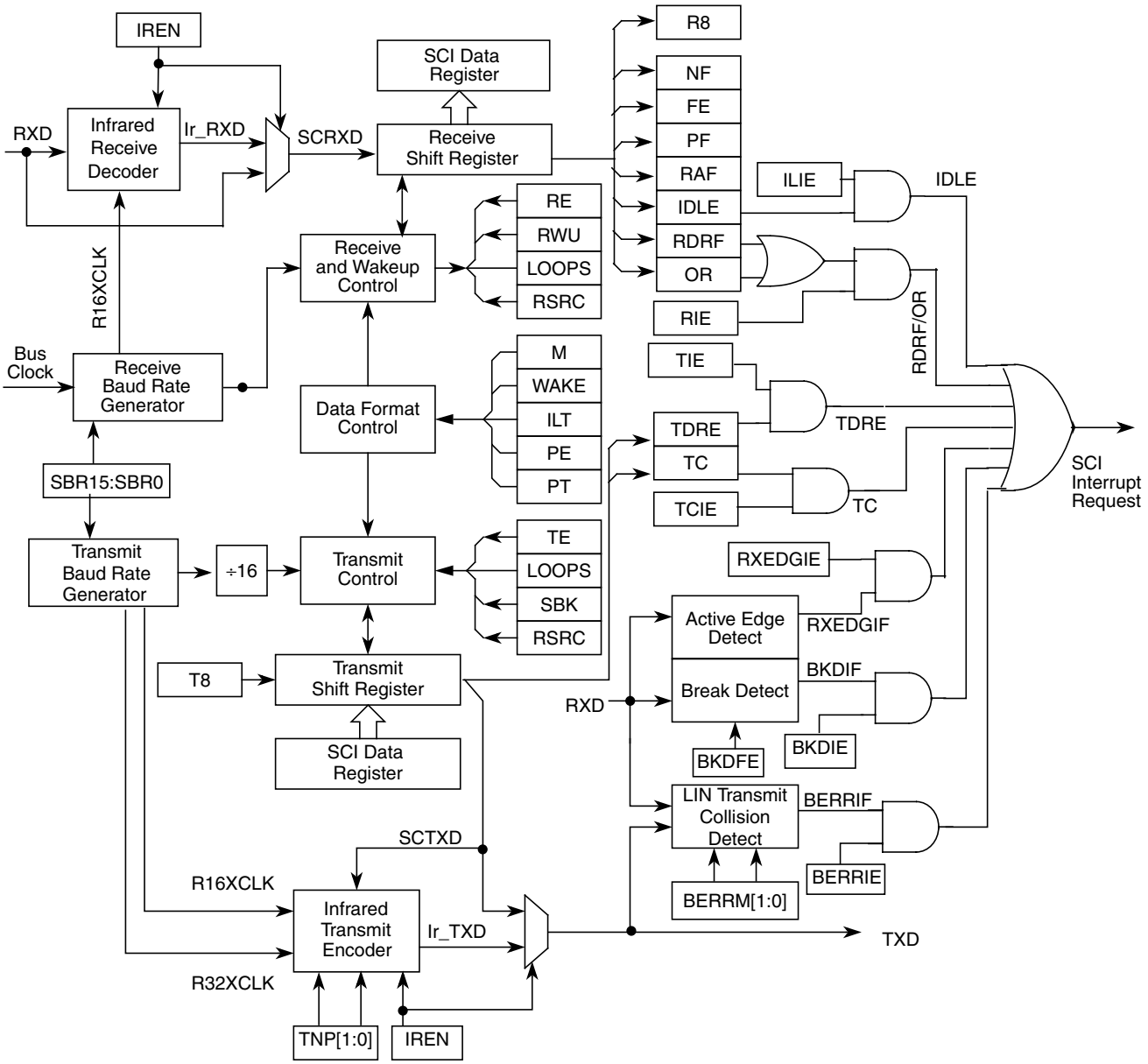



**Figure 10-14. Detailed SCI Block Diagram**

### 10.4.1 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 zero bit. No pulse is transmitted for every one 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 use 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, 1/32 or 1/4 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.

#### 10.4.1.1 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 zero bit and no pulse for a one bit. The narrow pulse is sent in the middle of the bit with a duration of 1/32, 1/16, 3/16 or 1/4 of a bit time. A narrow high pulse is transmitted for a zero bit when TXPOL is cleared, while a narrow low pulse is transmitted for a zero bit when TXPOL is set.

#### 10.4.1.2 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 zero received and no pulse is expected for each one received. A narrow high pulse is expected for a zero bit when RXPOL is cleared, while a narrow low pulse is expected for a zero bit when RXPOL is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

### 10.4.2 LIN Support

This module provides some basic support for the LIN protocol. At first this is a break detect circuitry making it easier for the LIN software to distinguish a break character from an incoming data stream. As a further addition is supports a collision detection at the bit level as well as cancelling pending transmissions.

## 10.4.3 Data Format

The SCI uses the standard NRZ mark/space data format. When Infrared is enabled, the SCI uses RZI data format where zeroes are represented by light pulses and ones remain low. See Figure 10-15 below.

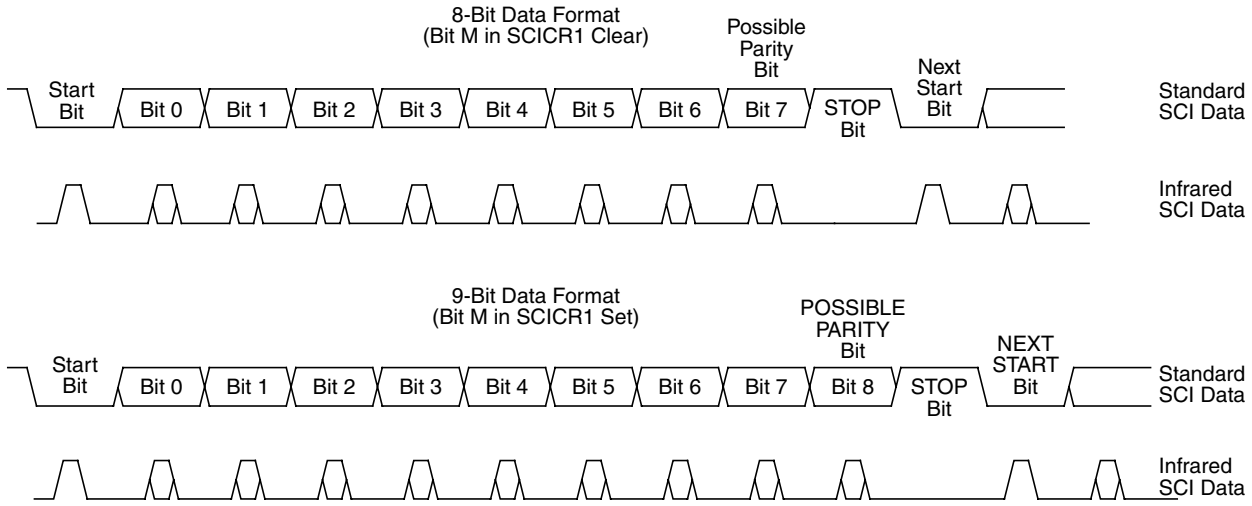



**Figure 10-15. SCI Data Formats**

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 10-14. 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 10.4.6.6, “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 10-15. 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 10.4.6.6, “Receiver Wakeup”.

## 10.4.4 Baud Rate Generation

A 16-bit modulus counter in the two baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 0 to 65535 written to the SBR15:SBR0 bits determines the baud rate. The value from 0 to 4095 written to the SBR15:SBR4 bits determines the baud rate clock with SBR3:SBR0 for fine adjust. The SBR bits are in the SCI baud rate registers (SCIBDH and SCIBDL) for both transmit and receive baud generator. 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 bus clock may not give the exact target frequency.

Table 10-16 lists some examples of achieving target baud rates with a bus clock frequency of 25 MHz.

When IREN = 0 then,

SCI baud rate = SCI bus clock / (SCIBR[15:0])

**Table 10-16. Baud Rates (Example: Bus Clock = 25 MHz)**

| Bits SBR[15:0] | Receiver[1] Clock (Hz) | Transmitter[2] Clock (Hz) | Target Baud Rate | Error (%) |
|---|---|---|---|---|
| 109 | 3669724.8 | 229,357.8 | 230,400 | .452 |
| 217 | 1843318.0 | 115,207.4 | 115,200 | .006 |
| 651 | 614439.3 | 38,402.5 | 38,400 | .006 |
| 1302 | 307219.7 | 19,201.2 | 19,200 | .006 |
| 2604 | 153,609.8 | 9600.6 | 9,600 | .006 |
| 5208 | 76,804.9 | 4800.3 | 4,800 | .006 |
| 10417 | 38,398.8 | 2399.9 | 2,400 | .003 |
| 20833 | 19,200.3 | 1200.02 | 1,200 | .00 |
| 41667 | 9599.9 | 600.0 | 600 | .00 |
| 65535 | 6103.6 | 381.5 | | |

[1] 16x faster then baud rate

[2] divide 1/16 form transmit baud generator

## 10.4.5 Transmitter

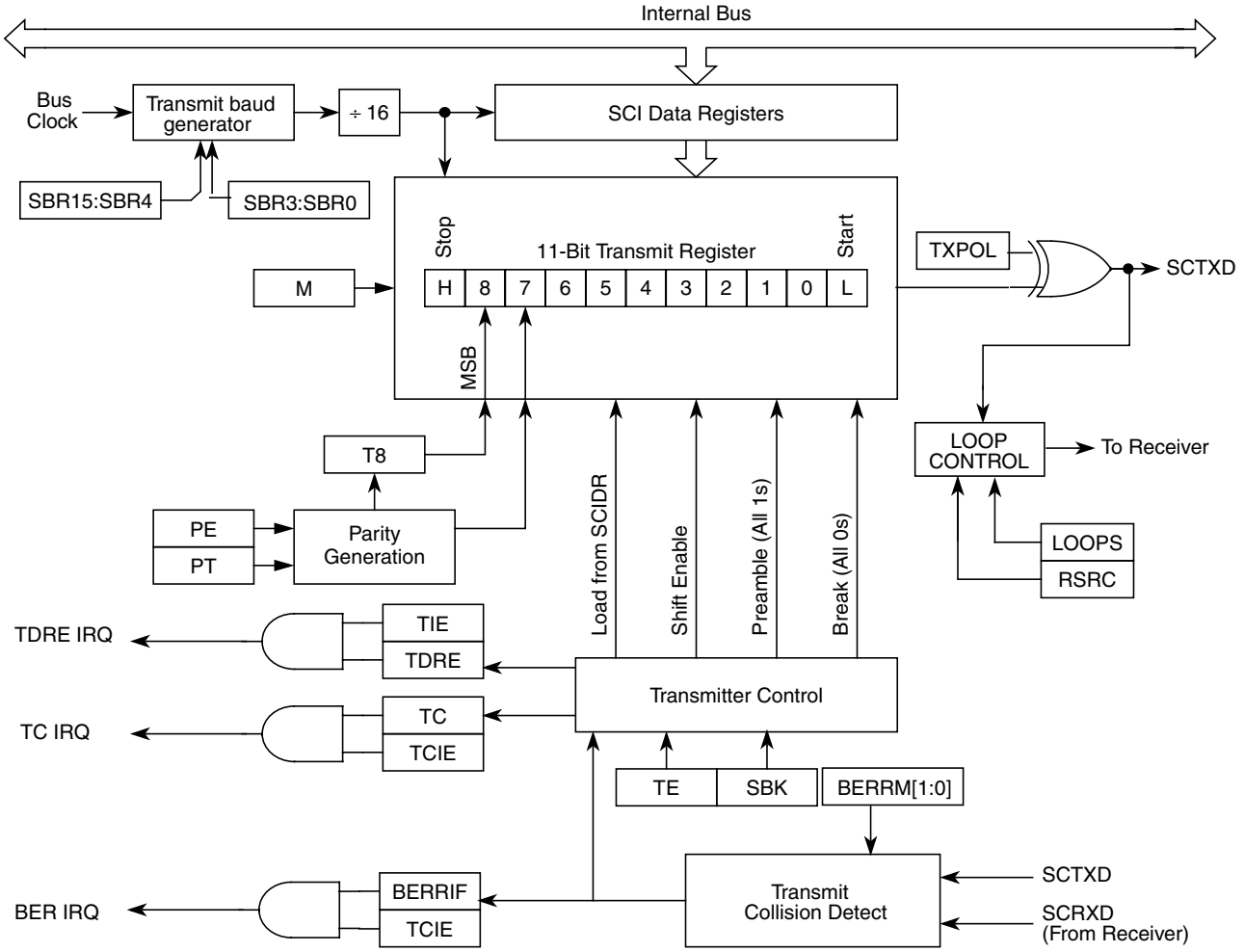



**Figure 10-16. Transmitter Block Diagram**

### 10.4.5.1 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).

### 10.4.5.2 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 still 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 zero. 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,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,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 one.
   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.

### 10.4.5.3 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 there are 10 or 11(M = 0 or M = 1) consecutive zero received. Depending if the break detect feature is enabled or not receiving a break character has these effects on SCI registers.

If the break detect feature is disabled (BKDFE = 0):

- 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 3.4.4 and 3.4.5 SCI Status Register 1 and 2)

If the break detect feature is enabled (BKDFE = 1) there are two scenarios[1]

The break is detected right from a start bit or is detected during a byte reception.

- Sets the break detect interrupt flag, BKDIF
- Does not change the data register full flag, RDRF or overrun flag OR
- Does not change the framing error flag FE, parity error flag PE.
- Does not clear the SCI data registers (SCIDRH/L)
- May set noise flag NF, or receiver active flag RAF.

1. A Break character in this context are either 10 or 11 consecutive zero received bits

Figure 10-17 shows two cases of break detect. In trace RXD_1 the break symbol starts with the start bit, while in RXD_2 the break starts in the middle of a transmission. If BRKDFE = 1, in RXD_1 case there will be no byte transferred to the receive buffer and the RDRF flag will not be modified. Also no framing error or parity error will be flagged from this transfer. In RXD_2 case, however the break signal starts later during the transmission. At the expected stop bit position the byte received so far will be transferred to the receive buffer, the receive data register full flag will be set, a framing error and if enabled and appropriate a parity error will be set. Once the break is detected the BRKDIF flag will be set.

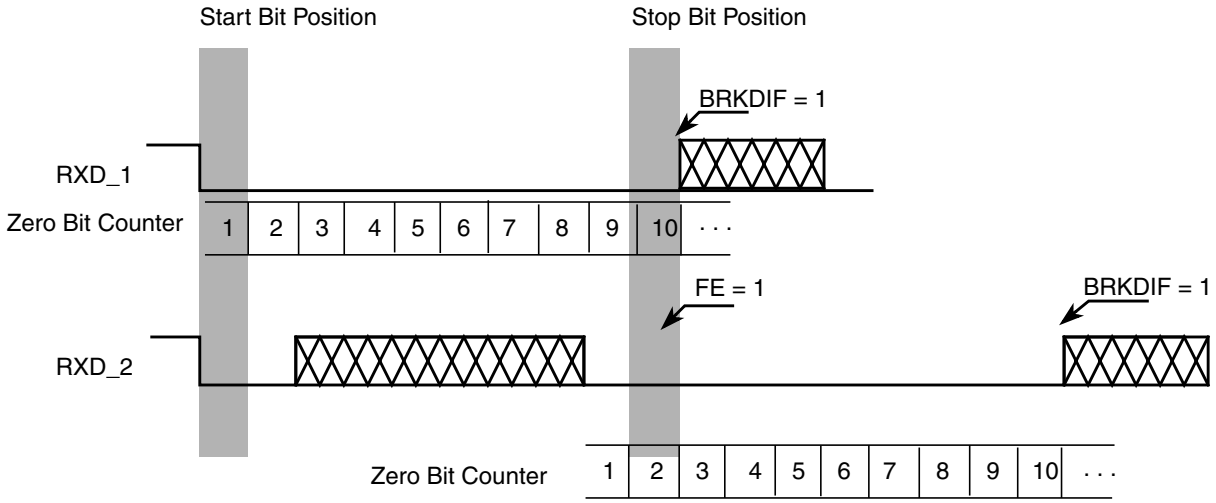



**Figure 10-17. Break Detection if BRKDFE = 1 (M = 0)**

### 10.4.5.4 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

### 10.4.5.5 LIN Transmit Collision Detection

This module allows to check for collisions on the LIN bus.

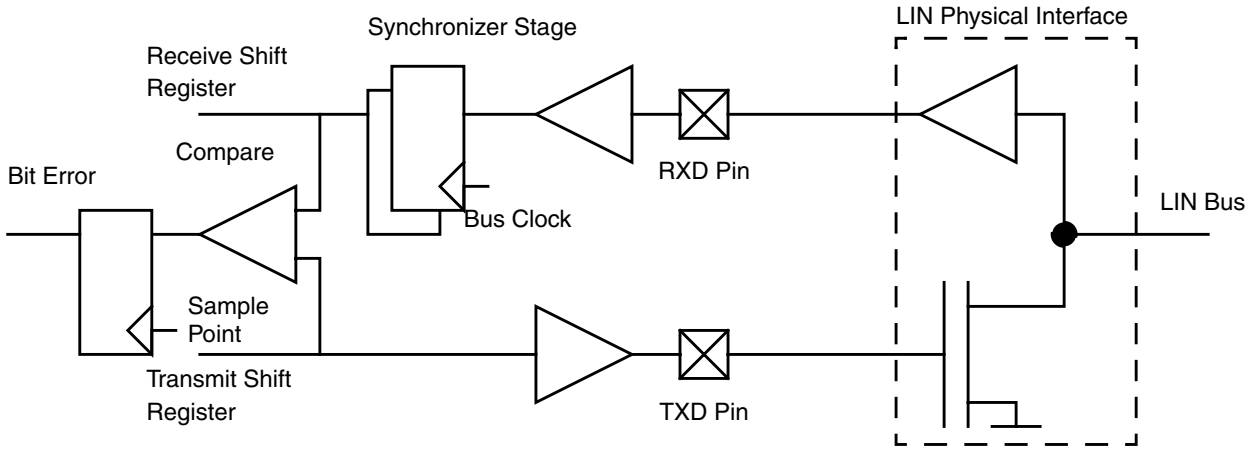



**Figure 10-18. Collision Detect Principle**

If the bit error circuit is enabled (BERRM[1:0] = 0:1 or = 1:0]), the error detect circuit will compare the transmitted and the received data stream at a point in time and flag any mismatch. The timing checks run when transmitter is active (not idle). As soon as a mismatch between the transmitted data and the received data is detected the following happens:

- The next bit transmitted will have a high level (TXPOL = 0) or low level (TXPOL = 1)
- The transmission is aborted and the byte in transmit buffer is discarded.
- the transmit data register empty and the transmission complete flag will be set
- The bit error interrupt flag, BERRIF, will be set.
- No further transmissions will take place until the BERRIF is cleared.

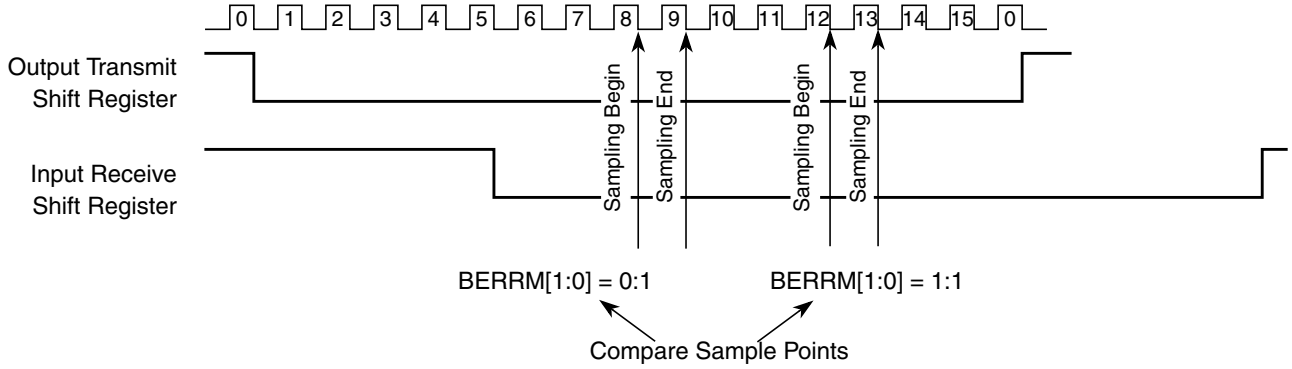



**Figure 10-19. Timing Diagram Bit Error Detection**

If the bit error detect feature is disabled, the bit error interrupt flag is cleared.

**NOTE**

The RXPOL and TXPOL bit should be set the same when transmission collision detect feature is enabled, otherwise the bit error interrupt flag may be set incorrectly.

## 10.4.6 Receiver

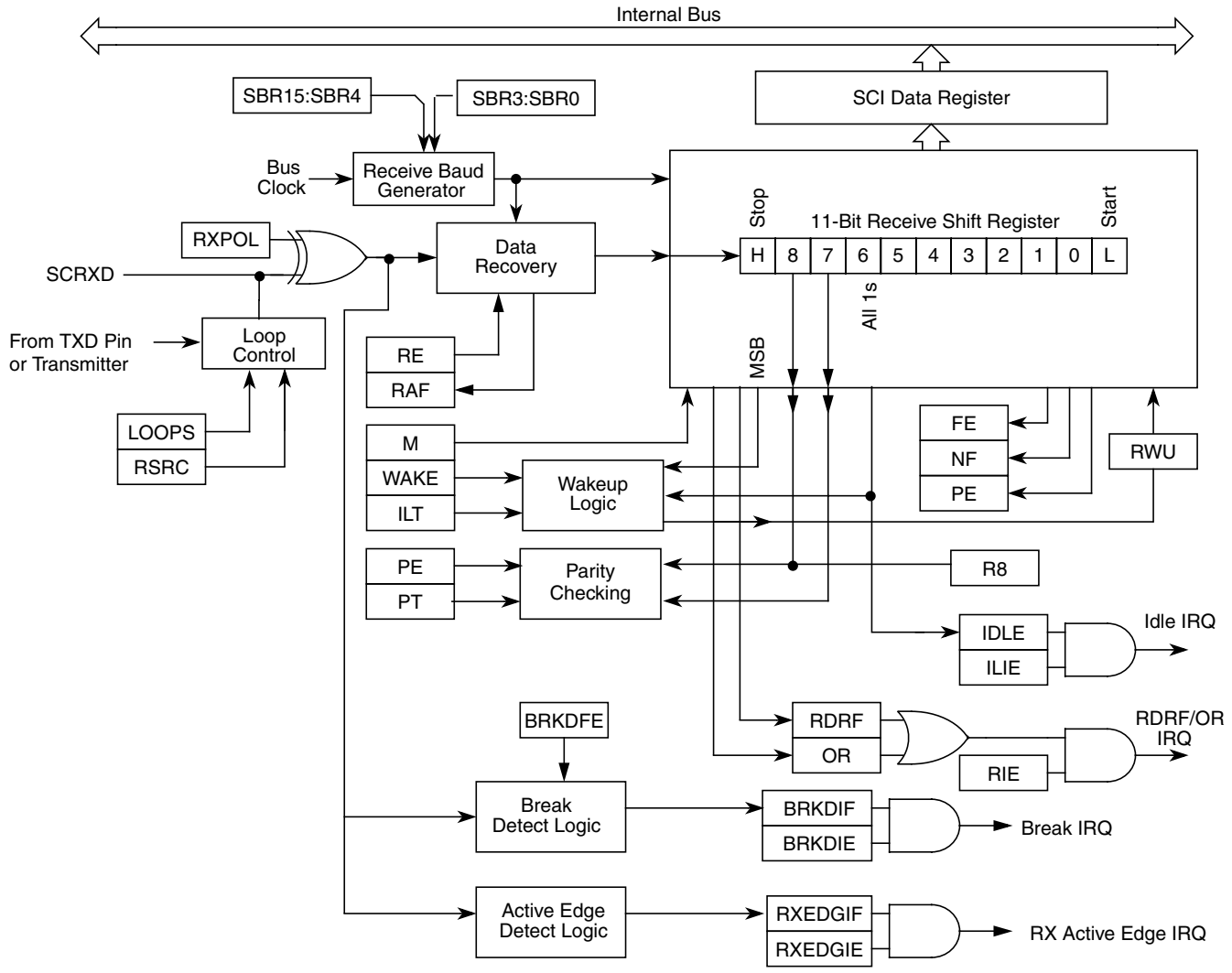



**Figure 10-20. SCI Receiver Block Diagram**

### 10.4.6.1 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).

### 10.4.6.2 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.

### 10.4.6.3 Data Sampling

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 10-21) is re-synchronized immediatelly at bus clock edge:

- 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.

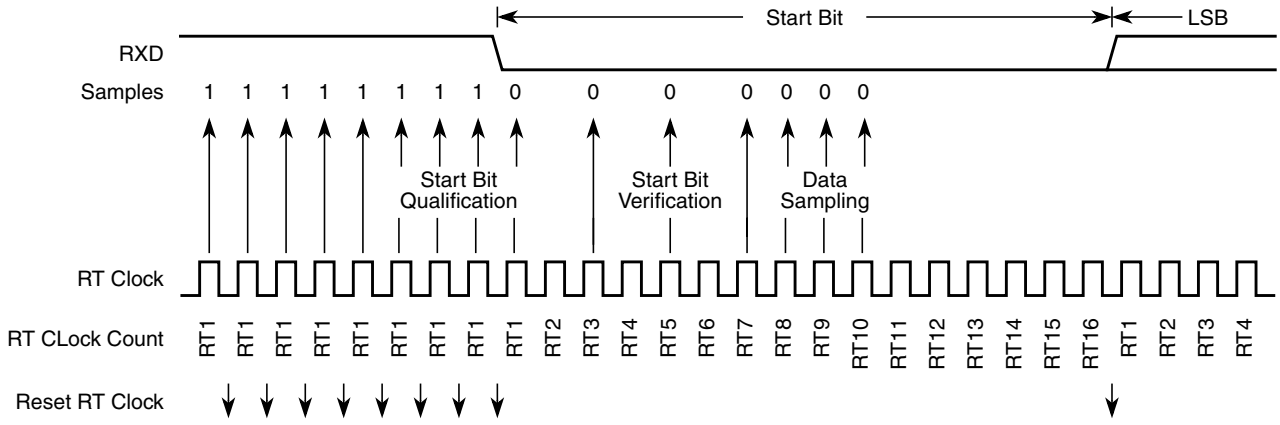



**Figure 10-21. Receiver Data Sampling**

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Figure 10-17 summarizes the results of the start bit verification samples.

**Table 10-17. 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 10-18 summarizes the results of the data bit samples.

**Table 10-18. 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 10-19 summarizes the results of the stop bit samples.

**Table 10-19. 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 10-22 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.

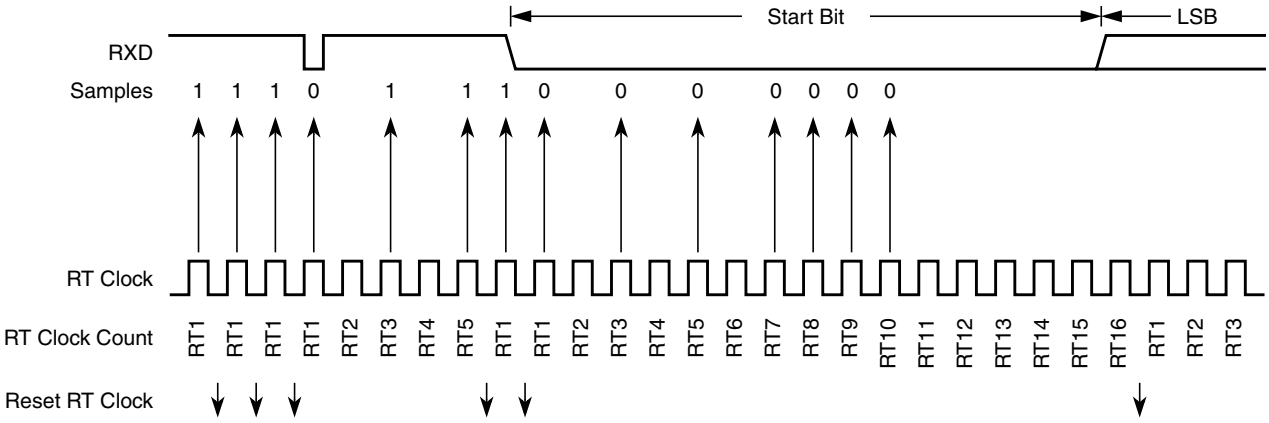



**Figure 10-22. Start Bit Search Example 1**

In Figure 10-23, 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.

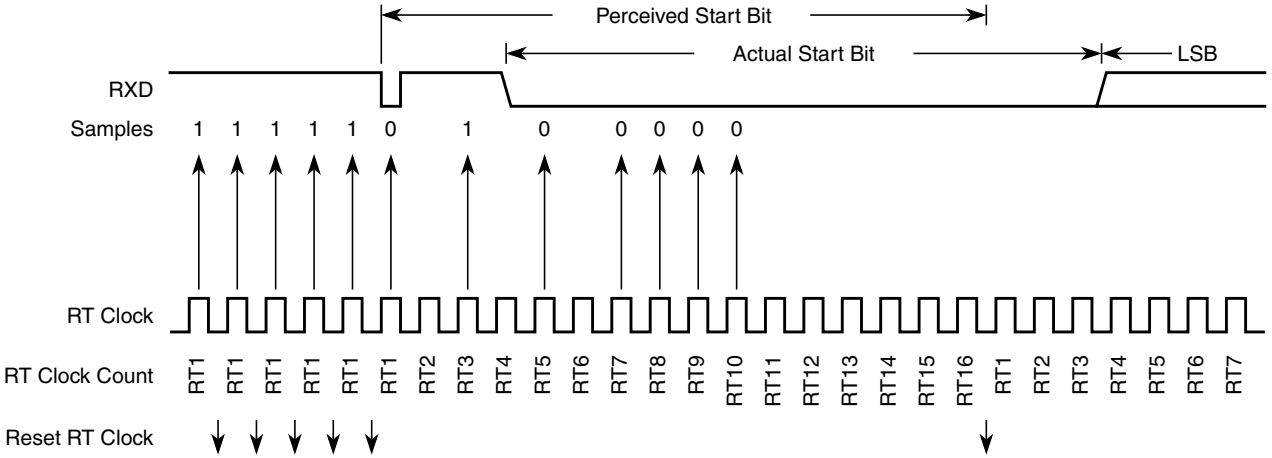



**Figure 10-23. Start Bit Search Example 2**

In Figure 10-24, 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.

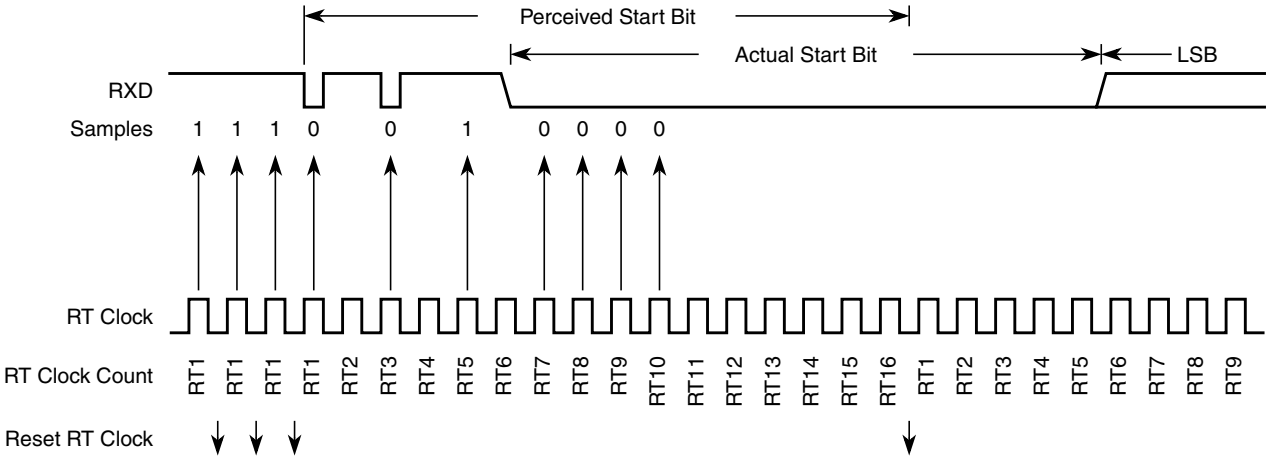



**Figure 10-24. Start Bit Search Example 3**

Figure 10-25 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.

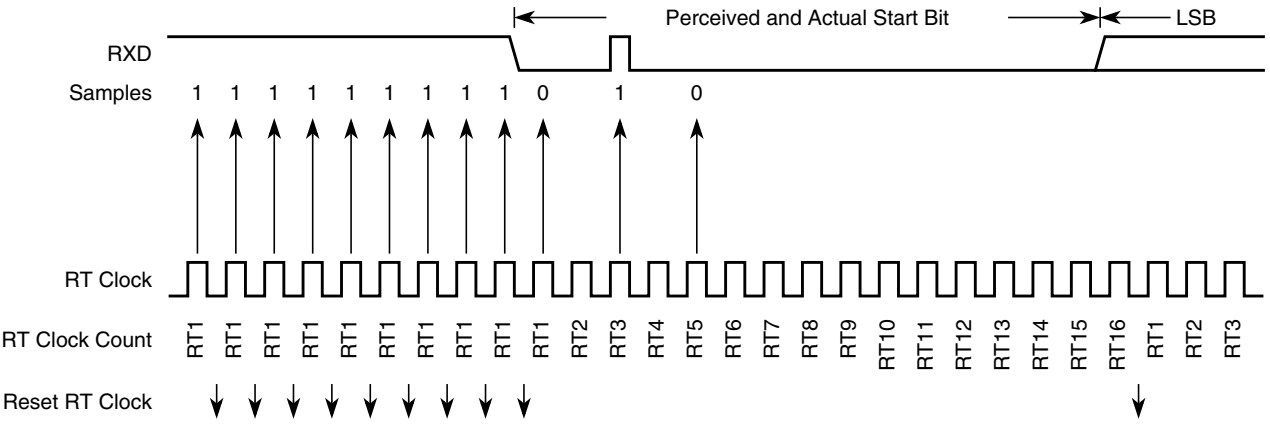



**Figure 10-25. Start Bit Search Example 4**

Figure 10-26 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.

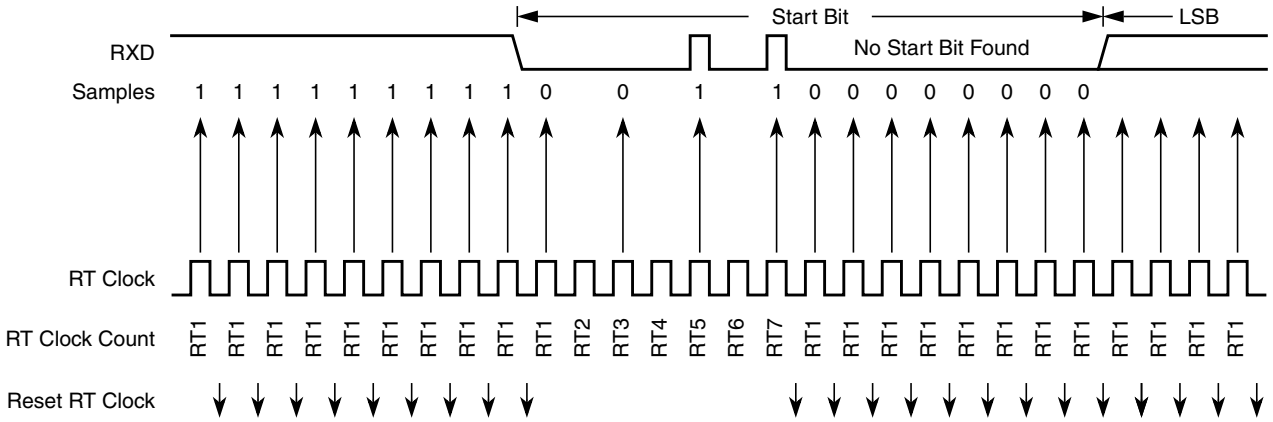



**Figure 10-26. Start Bit Search Example 5**

In Figure 10-27, 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.

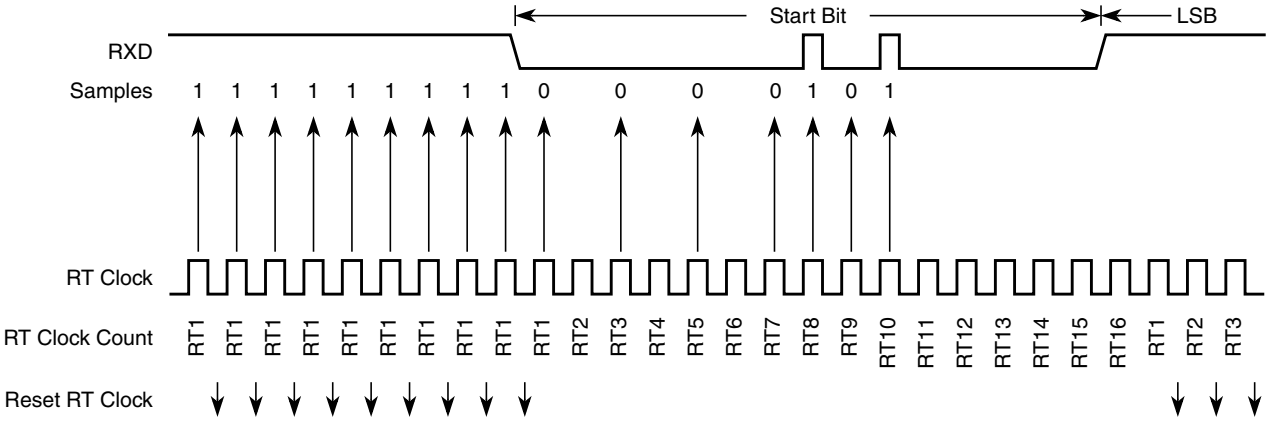



**Figure 10-27. Start Bit Search Example 6**

### 10.4.6.4 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.

### 10.4.6.5 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 zero.

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.

#### 10.4.6.5.1 Slow Data Tolerance

Figure 10-28 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.

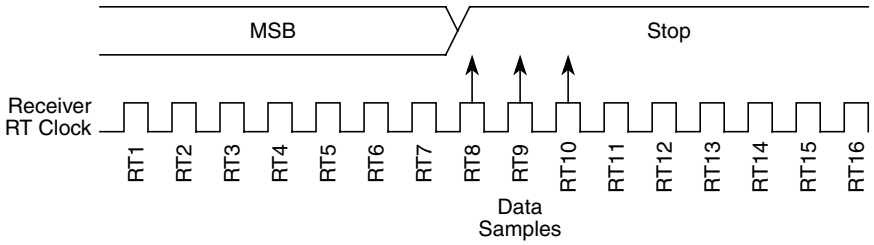



**Figure 10-28. Slow Data**

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 10-28, 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 10-28, 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%

#### 10.4.6.5.2 Fast Data Tolerance

Figure 10-29 shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

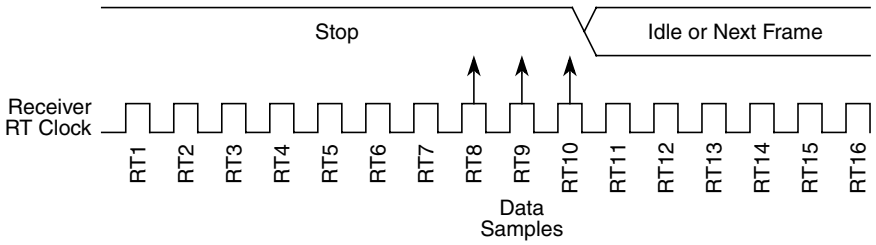



**Figure 10-29. Fast Data**

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles + 9 RTr cycles = 153 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 10-29, the receiver counts 153 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 – 153) / 160) x 100 = 4.375%

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 9 RTr cycles = 169 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 10-29, the receiver counts 169 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 – 169) /176) x 100 = 3.98%

**NOTE**

Due to asynchronous sample and internal logic, there is maximal 2 bus cycles between startbit edge and 1st RT clock, and cause to additional tolerance loss at worst case. The loss should be 2/SBR/10*100%, it is small.For example, for highspeed baud=230400 with 25MHz bus, SBR should be 109, and the tolerance loss is 2/109/10*100=0.18%, and fast data tolerance is 4.375%-0.18%=4.195%.

### 10.4.6.6 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 still 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.

#### 10.4.6.6.1 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).

#### 10.4.6.6.2 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.

### 10.4.7 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.

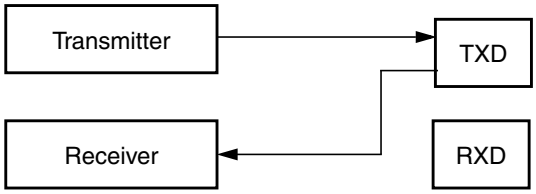



**Figure 10-30. Single-Wire Operation (LOOPS = 1, RSRC = 1)**

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.

**NOTE**

In single-wire operation data from the TXD pin is inverted if RXPOL is set.

### 10.4.8 Loop Operation

In loop operation the transmitter output goes to the receiver input. The RXD pin is disconnected from the SCI.

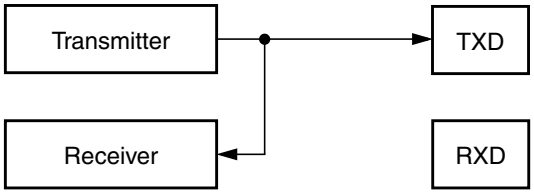



**Figure 10-31. Loop Operation (LOOPS = 1, RSRC = 0)**

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).

**NOTE**

In loop operation data from the transmitter is not recognized by the receiver if RXPOL and TXPOL are not the same.

## 10.5 Initialization/Application Information

### 10.5.1 Reset Initialization

See Section 10.3.2, “Register Descriptions”.

## 10.5.2 Modes of Operation

### 10.5.2.1 Run Mode

Normal mode of operation.

To initialize a SCI transmission, see Section 10.4.5.2, “Character Transmission”.

### 10.5.2.2 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.

### 10.5.2.3 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 from where it left off 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.

The receive input active edge detect circuit is still active in stop mode. An active edge on the receive input can be used to bring the CPU out of stop mode.

## 10.5.3 Interrupt Operation

This section describes the interrupt originated by the SCI block.The MCU must service the interrupt requests. Table 10-20 lists the eight interrupt sources of the SCI.

**Table 10-20. 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. |

**Table 10-20. SCI Interrupt Sources**

| | | | |
|---|---|---|---|
| RXEDGIF | SCIASR1[7] | RXEDGIE | Active high level. Indicates that an active edge (falling for RXPOL = 0, rising for RXPOL = 1) was detected. |
| BERRIF | SCIASR1[1] | BERRIE | Active high level. Indicates that a mismatch between transmitted and received data in a single wire application has happened. |
| BKDIF | SCIASR1[0] | BRKDIE | Active high level. Indicates that a break character has been received. |

### 10.5.3.1 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.

#### 10.5.3.1.1 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).

#### 10.5.3.1.2 TC Description

The TC interrupt is set by the SCI when a transmission has been completed. Transmission is completed when all bits including the stop bit (if transmitted) have been shifted out and no data is queued to be transmitted. No stop bit is transmitted when sending a break character and the TC flag is set (providing there is no more data queued for transmission) when the break character has been shifted out. 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.

#### 10.5.3.1.3 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).

#### 10.5.3.1.4 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).

#### 10.5.3.1.5 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. Once 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).

#### 10.5.3.1.6 RXEDGIF Description

The RXEDGIF interrupt is set when an active edge (falling if RXPOL = 0, rising if RXPOL = 1) on the RXD pin is detected. Clear RXEDGIF by writing a “1” to the SCIASR1 SCI alternative status register 1.

#### 10.5.3.1.7 BERRIF Description

The BERRIF interrupt is set when a mismatch between the transmitted and the received data in a single wire application like LIN was detected. Clear BERRIF by writing a “1” to the SCIASR1 SCI alternative status register 1. This flag is also cleared if the bit error detect feature is disabled.

#### 10.5.3.1.8 BKDIF Description

The BKDIF interrupt is set when a break signal was received. Clear BKDIF by writing a “1” to the SCIASR1 SCI alternative status register 1. This flag is also cleared if break detect feature is disabled.

### 10.5.4 Recovery from Wait Mode

The SCI interrupt request can be used to bring the CPU out of wait mode.

### 10.5.5 Recovery from Stop Mode

An active edge on the receive input can be used to bring the CPU out of stop mode.

# Chapter 11
# Serial Peripheral Interface (S12SPIV5)

**Table 11-1. Revision History**

| Revision Number | Revision Date | Sections Affected | Description of Changes |
|---|---|---|---|
| V05.00 | 24 Mar 2005 | 11.3.2/11-345 | - Added 16-bit transfer width feature. |

## 11.1 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.

### 11.1.1 Glossary of Terms

| | |
|---|---|
| SPI | Serial Peripheral Interface |
| $\overline{SS}$ | Slave Select |
| SCK | Serial Clock |
| MOSI | Master Output, Slave Input |
| MISO | Master Input, Slave Output |
| MOMI | Master Output, Master Input |
| SISO | Slave Input, Slave Output |

### 11.1.2 Features

The SPI includes these distinctive features:

- Master mode and slave mode
- Selectable 8 or 16-bit transfer width
- 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

### 11.1.3 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 data 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 data continues, so that the slave stays synchronized to the master.

For a detailed description of operating modes, please refer to Section 11.4.7, “Low Power Mode Options”.

### 11.1.4 Block Diagram

Figure 11-1 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.

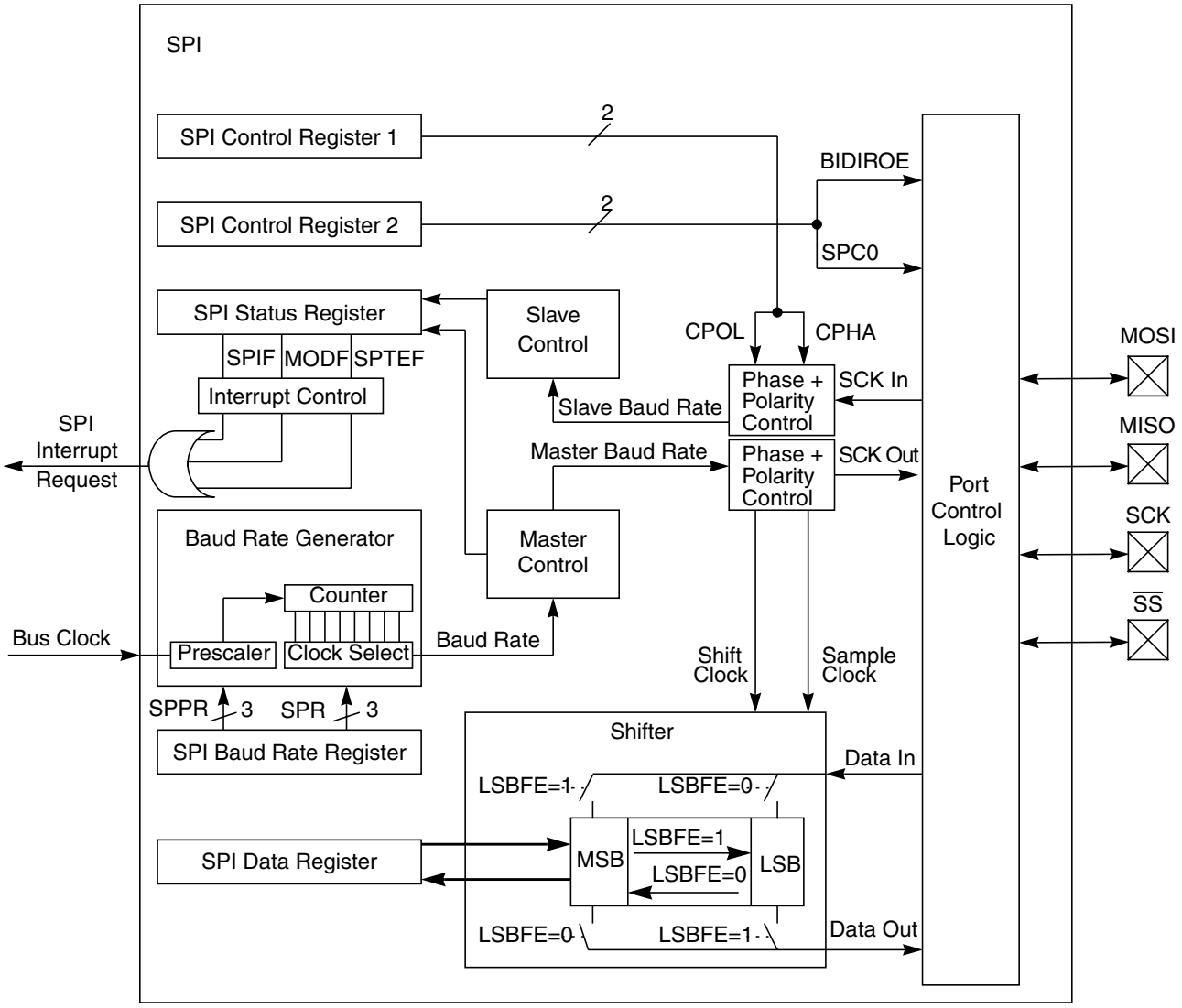



**Figure 11-1. SPI Block Diagram**

## 11.2 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 SPI module has a total of four external pins.

### 11.2.1 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.

### 11.2.2 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.

### 11.2.3 $\overline{SS}$ — 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 it is configured as a master and it is used as an input to receive the slave select signal when the SPI is configured as slave.

### 11.2.4 SCK — Serial Clock Pin

In master mode, this is the synchronous output clock. In slave mode, this is the synchronous input clock.

## 11.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the SPI.

### 11.3.1 Module Memory Map

The memory map for the SPI is given in Figure 11-2. 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.

| Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0000 SPICR1 | R | SPIE | SPE | SPTIE | MSTR | CPOL | CPHA | SSOE | LSBFE |
| | W | | | | | | | | |
| 0x0001 SPICR2 | R | 0 | XFRW | 0 | MODFEN | BIDIROE | 0 | SPISWAI | SPC0 |
| | W | | | | | | | | |
| 0x0002 SPIBR | R | 0 | SPPR2 | SPPR1 | SPPR0 | 0 | SPR2 | SPR1 | SPR0 |
| | W | | | | | | | | |
| 0x0003 SPISR | R | SPIF | 0 | SPTEF | MODF | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0004 SPIDRH | R | R15 | R14 | R13 | R12 | R11 | R10 | R9 | R8 |
| | W | T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 |
| 0x0005 SPIDRL | R | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| | W | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |
| 0x0006 Reserved | R | | | | | | | | |
| | W | | | | | | | | |
| 0x0007 Reserved | R | | | | | | | | |
| | W | | | | | | | | |

= Unimplemented or Reserved

**Figure 11-2. SPI Register Summary**

## 11.3.2 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.

### 11.3.2.1 SPI Control Register 1 (SPICR1)

Module Base +0x0000

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | SPIE | SPE | SPTIE | MSTR | CPOL | CPHA | SSOE | LSBFE |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

**Figure 11-3. SPI Control Register 1 (SPICR1)**

Read: Anytime

Write: Anytime

**Table 11-2. SPICR1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>SPIE | **SPI Interrupt Enable Bit** — This bit enables SPI interrupt requests, if SPIF or MODF status flag is set.<br>0 SPI interrupts disabled.<br>1 SPI interrupts enabled. |
| 6<br>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.<br>0 SPI disabled (lower power consumption).<br>1 SPI enabled, port pins are dedicated to SPI functions. |
| 5<br>SPTIE | **SPI Transmit Interrupt Enable** — This bit enables SPI interrupt requests, if SPTEF flag is set.<br>0 SPTEF interrupt disabled.<br>1 SPTEF interrupt enabled. |
| 4<br>MSTR | **SPI Master/Slave Mode Select Bit** — This bit selects whether 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.<br>0 SPI is in slave mode.<br>1 SPI is in master mode. |
| 3<br>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.<br>0 Active-high clocks selected. In idle state SCK is low.<br>1 Active-low clocks selected. In idle state SCK is high. |
| 2<br>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.<br>0 Sampling of data occurs at odd edges (1,3,5,...) of the SCK clock.<br>1 Sampling of data occurs at even edges (2,4,6,...) of the SCK clock. |

**Table 11-2. SPICR1 Field Descriptions (continued)**

| Field | Description |
|---|---|
| 1<br>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 11-3. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. |
| 0<br>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 the highest bit position. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.<br>0 Data is transferred most significant bit first.<br>1 Data is transferred least significant bit first. |

**Table 11-3. $\overline{SS}$ 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 |

### 11.3.2.2 SPI Control Register 2 (SPICR2)

Module Base +0x0001

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | XFRW | 0 | MODFEN | BIDIROE | 0 | SPISWAI | SPC0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 11-4. SPI Control Register 2 (SPICR2)**

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

**Table 11-4. SPICR2 Field Descriptions**

| Field | Description |
|---|---|
| 6<br>XFRW | **Transfer Width —** This bit is used for selecting the data transfer width. If 8-bit transfer width is selected, SPIDRL becomes the dedicated data register and SPIDRH is unused. If 16-bit transfer width is selected, SPIDRH and SPIDRL form a 16-bit data register. Please refer to Section 11.3.2.4, “SPI Status Register (SPISR) for information about transmit/receive data handling and the interrupt flag clearing mechanism. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.<br>0 8-bit Transfer Width (n = 8)[1]<br>1 16-bit Transfer Width (n = 16)[1] |
| 4<br>MODFEN | **Mode Fault Enable Bit** — This bit allows the MODF failure to be 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 11-3. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.<br>0 $\overline{SS}$ port pin is not used by the SPI.<br>1 $\overline{SS}$ port pin with MODF feature. |
| 3<br>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.<br>0 Output buffer disabled.<br>1 Output buffer enabled. |
| 1<br>SPISWAI | **SPI Stop in Wait Mode Bit** — This bit is used for power conservation while in wait mode.<br>0 SPI clock operates normally in wait mode.<br>1 Stop SPI clock generation when in wait mode. |
| 0<br>SPC0 | **Serial Pin Control Bit 0** — This bit enables bidirectional pin configurations as shown in Table 11-5. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. |

[1] n is used later in this document as a placeholder for the selected transfer width.

**Table 11-5. 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 | |

### 11.3.2.3 SPI Baud Rate Register (SPIBR)

Module Base +0x0002

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | SPPR2 | SPPR1 | SPPR0 | 0 | SPR2 | SPR1 | SPR0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 11-5. SPI Baud Rate Register (SPIBR)**

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

**Table 11-6. SPIBR Field Descriptions**

| Field | Description |
|---|---|
| 6–4<br>SPPR[2:0] | **SPI Baud Rate Preselection Bits** — These bits specify the SPI baud rates as shown in Table 11-7. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state. |
| 2–0<br>SPR[2:0] | **SPI Baud Rate Selection Bits** — These bits specify the SPI baud rates as shown in Table 11-7. 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:

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 11-1}$$

The baud rate can be calculated with the following equation:

$$\text{Baud Rate} = \text{BusClock} / \text{BaudRateDivisor} \qquad \textit{Eqn. 11-2}$$

**NOTE**

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

**Table 11-7. Example SPI Baud Rate Selection (25 MHz Bus Clock) (Sheet 1 of 3)**

| SPPR2 | SPPR1 | SPPR0 | SPR2 | SPR1 | SPR0 | Baud Rate Divisor | Baud Rate |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 2 | 12.5 Mbit/s |
| 0 | 0 | 0 | 0 | 0 | 1 | 4 | 6.25 Mbit/s |
| 0 | 0 | 0 | 0 | 1 | 0 | 8 | 3.125 Mbit/s |
| 0 | 0 | 0 | 0 | 1 | 1 | 16 | 1.5625 Mbit/s |
| 0 | 0 | 0 | 1 | 0 | 0 | 32 | 781.25 kbit/s |
| 0 | 0 | 0 | 1 | 0 | 1 | 64 | 390.63 kbit/s |
| 0 | 0 | 0 | 1 | 1 | 0 | 128 | 195.31 kbit/s |
| 0 | 0 | 0 | 1 | 1 | 1 | 256 | 97.66 kbit/s |
| 0 | 0 | 1 | 0 | 0 | 0 | 4 | 6.25 Mbit/s |

**Table 11-7. Example SPI Baud Rate Selection (25 MHz Bus Clock) (Sheet 2 of 3)**

| SPPR2 | SPPR1 | SPPR0 | SPR2 | SPR1 | SPR0 | Baud Rate Divisor | Baud Rate |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 1 | 8 | 3.125 Mbit/s |
| 0 | 0 | 1 | 0 | 1 | 0 | 16 | 1.5625 Mbit/s |
| 0 | 0 | 1 | 0 | 1 | 1 | 32 | 781.25 kbit/s |
| 0 | 0 | 1 | 1 | 0 | 0 | 64 | 390.63 kbit/s |
| 0 | 0 | 1 | 1 | 0 | 1 | 128 | 195.31 kbit/s |
| 0 | 0 | 1 | 1 | 1 | 0 | 256 | 97.66 kbit/s |
| 0 | 0 | 1 | 1 | 1 | 1 | 512 | 48.83 kbit/s |
| 0 | 1 | 0 | 0 | 0 | 0 | 6 | 4.16667 Mbit/s |
| 0 | 1 | 0 | 0 | 0 | 1 | 12 | 2.08333 Mbit/s |
| 0 | 1 | 0 | 0 | 1 | 0 | 24 | 1.04167 Mbit/s |
| 0 | 1 | 0 | 0 | 1 | 1 | 48 | 520.83 kbit/s |
| 0 | 1 | 0 | 1 | 0 | 0 | 96 | 260.42 kbit/s |
| 0 | 1 | 0 | 1 | 0 | 1 | 192 | 130.21 kbit/s |
| 0 | 1 | 0 | 1 | 1 | 0 | 384 | 65.10 kbit/s |
| 0 | 1 | 0 | 1 | 1 | 1 | 768 | 32.55 kbit/s |
| 0 | 1 | 1 | 0 | 0 | 0 | 8 | 3.125 Mbit/s |
| 0 | 1 | 1 | 0 | 0 | 1 | 16 | 1.5625 Mbit/s |
| 0 | 1 | 1 | 0 | 1 | 0 | 32 | 781.25 kbit/s |
| 0 | 1 | 1 | 0 | 1 | 1 | 64 | 390.63 kbit/s |
| 0 | 1 | 1 | 1 | 0 | 0 | 128 | 195.31 kbit/s |
| 0 | 1 | 1 | 1 | 0 | 1 | 256 | 97.66 kbit/s |
| 0 | 1 | 1 | 1 | 1 | 0 | 512 | 48.83 kbit/s |
| 0 | 1 | 1 | 1 | 1 | 1 | 1024 | 24.41 kbit/s |
| 1 | 0 | 0 | 0 | 0 | 0 | 10 | 2.5 Mbit/s |
| 1 | 0 | 0 | 0 | 0 | 1 | 20 | 1.25 Mbit/s |
| 1 | 0 | 0 | 0 | 1 | 0 | 40 | 625 kbit/s |
| 1 | 0 | 0 | 0 | 1 | 1 | 80 | 312.5 kbit/s |
| 1 | 0 | 0 | 1 | 0 | 0 | 160 | 156.25 kbit/s |
| 1 | 0 | 0 | 1 | 0 | 1 | 320 | 78.13 kbit/s |
| 1 | 0 | 0 | 1 | 1 | 0 | 640 | 39.06 kbit/s |
| 1 | 0 | 0 | 1 | 1 | 1 | 1280 | 19.53 kbit/s |
| 1 | 0 | 1 | 0 | 0 | 0 | 12 | 2.08333 Mbit/s |
| 1 | 0 | 1 | 0 | 0 | 1 | 24 | 1.04167 Mbit/s |
| 1 | 0 | 1 | 0 | 1 | 0 | 48 | 520.83 kbit/s |
| 1 | 0 | 1 | 0 | 1 | 1 | 96 | 260.42 kbit/s |
| 1 | 0 | 1 | 1 | 0 | 0 | 192 | 130.21 kbit/s |
| 1 | 0 | 1 | 1 | 0 | 1 | 384 | 65.10 kbit/s |
| 1 | 0 | 1 | 1 | 1 | 0 | 768 | 32.55 kbit/s |
| 1 | 0 | 1 | 1 | 1 | 1 | 1536 | 16.28 kbit/s |
| 1 | 1 | 0 | 0 | 0 | 0 | 14 | 1.78571 Mbit/s |
| 1 | 1 | 0 | 0 | 0 | 1 | 28 | 892.86 kbit/s |
| 1 | 1 | 0 | 0 | 1 | 0 | 56 | 446.43 kbit/s |

**Table 11-7. Example SPI Baud Rate Selection (25 MHz Bus Clock) (Sheet 3 of 3)**

| SPPR2 | SPPR1 | SPPR0 | SPR2 | SPR1 | SPR0 | Baud Rate Divisor | Baud Rate |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 | 112 | 223.21 kbit/s |
| 1 | 1 | 0 | 1 | 0 | 0 | 224 | 111.61 kbit/s |
| 1 | 1 | 0 | 1 | 0 | 1 | 448 | 55.80 kbit/s |
| 1 | 1 | 0 | 1 | 1 | 0 | 896 | 27.90 kbit/s |
| 1 | 1 | 0 | 1 | 1 | 1 | 1792 | 13.95 kbit/s |
| 1 | 1 | 1 | 0 | 0 | 0 | 16 | 1.5625 Mbit/s |
| 1 | 1 | 1 | 0 | 0 | 1 | 32 | 781.25 kbit/s |
| 1 | 1 | 1 | 0 | 1 | 0 | 64 | 390.63 kbit/s |
| 1 | 1 | 1 | 0 | 1 | 1 | 128 | 195.31 kbit/s |
| 1 | 1 | 1 | 1 | 0 | 0 | 256 | 97.66 kbit/s |
| 1 | 1 | 1 | 1 | 0 | 1 | 512 | 48.83 kbit/s |
| 1 | 1 | 1 | 1 | 1 | 0 | 1024 | 24.41 kbit/s |
| 1 | 1 | 1 | 1 | 1 | 1 | 2048 | 12.21 kbit/s |

## 11.3.2.4 SPI Status Register (SPISR)

Module Base +0x0003

| | 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 11-6. SPI Status Register (SPISR)**

Read: Anytime

Write: Has no effect

**Table 11-8. SPISR Field Descriptions**

| Field | Description |
|---|---|
| 7<br>SPIF | **SPIF Interrupt Flag** — This bit is set after received data has been transferred into the SPI data register. For information about clearing SPIF Flag, please refer to Table 11-9.<br>0 Transfer not yet complete.<br>1 New data copied to SPIDR. |

**Table 11-8. SPISR Field Descriptions (continued)**

| Field | Description |
|---|---|
| 5<br>SPTEF | **SPI Transmit Empty Interrupt Flag** — If set, this bit indicates that the transmit data register is empty. For information about clearing this bit and placing data into the transmit data register, please refer to Table 11-10.<br>0 SPI data register not empty.<br>1 SPI data register empty. |
| 4<br>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 11.3.2.2, “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.<br>0 Mode fault has not occurred.<br>1 Mode fault has occurred. |

**Table 11-9. SPIF Interrupt Flag Clearing Sequence**

<table>
<tr><th>XFRW Bit</th><th colspan="3">SPIF Interrupt Flag Clearing Sequence</th></tr>
<tr><td>0</td><td>Read SPISR with SPIF == 1</td><td>then</td><td>Read SPIDRL</td></tr>
<tr><td rowspan="5">1</td><td rowspan="5">Read SPISR with SPIF == 1</td><td rowspan="5">then</td><td>Byte Read SPIDRL [1]</td></tr>
<tr><td>or</td></tr>
<tr><td>Byte Read SPIDRH [2] | Byte Read SPIDRL</td></tr>
<tr><td>or</td></tr>
<tr><td>Word Read (SPIDRH:SPIDRL)</td></tr>
</table>

1 Data in SPIDRH is lost in this case.

2 SPIDRH can be read repeatedly without any effect on SPIF. SPIF Flag is cleared only by the read of SPIDRL after reading SPISR with SPIF == 1.

**Table 11-10. SPTEF Interrupt Flag Clearing Sequence**

<table>
<tr><th>XFRW Bit</th><th colspan="3">SPTEF Interrupt Flag Clearing Sequence</th></tr>
<tr><td>0</td><td>Read SPISR with SPTEF == 1</td><td>then</td><td>Write to SPIDRL [1]</td></tr>
<tr><td rowspan="5">1</td><td rowspan="5">Read SPISR with SPTEF == 1</td><td rowspan="5">then</td><td>Byte Write to SPIDRL [12]</td></tr>
<tr><td>or</td></tr>
<tr><td>Byte Write to SPIDRH [13] | Byte Write to SPIDRL [1]</td></tr>
<tr><td>or</td></tr>
<tr><td>Word Write to (SPIDRH:SPIDRL) [1]</td></tr>
</table>

1 Any write to SPIDRH or SPIDRL with SPTEF == 0 is effectively ignored.

2 Data in SPIDRH is undefined in this case.

3 SPIDRH can be written repeatedly without any effect on SPTEF. SPTEF Flag is cleared only by writing to SPIDRL after reading SPISR with SPTEF == 1.

### 11.3.2.5 SPI Data Register (SPIDR = SPIDRH:SPIDRL)

Module Base +0x0004

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | R15 | R14 | R13 | R12 | R11 | R10 | R9 | R8 |
| W | T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 11-7. SPI Data Register High (SPIDRH)**

Module Base +0x0005

| | 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 11-8. SPI Data Register Low (SPIDRL)**

Read: Anytime; read data only valid when 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 data to be queued and transmitted. For an SPI configured as a master, queued data 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.

Received data in the SPIDR is valid when SPIF is set.

If SPIF is cleared and data has been received, the received data is transferred from the receive shift register to the SPIDR and SPIF is set.

If SPIF is set and not serviced, and a second data value has been received, the second received data is kept as valid data in the receive shift register until the start of another transmission. The data in the SPIDR does not change.

If SPIF is set and valid data is in the receive shift register, and SPIF is serviced before the start of a third transmission, the data in the receive shift register is transferred into the SPIDR and SPIF remains set (see Figure 11-9).

If SPIF is set and valid data is in the receive shift register, and SPIF is serviced after the start of a third transmission, the data in the receive shift register has become invalid and is not transferred into the SPIDR (see Figure 11-10).

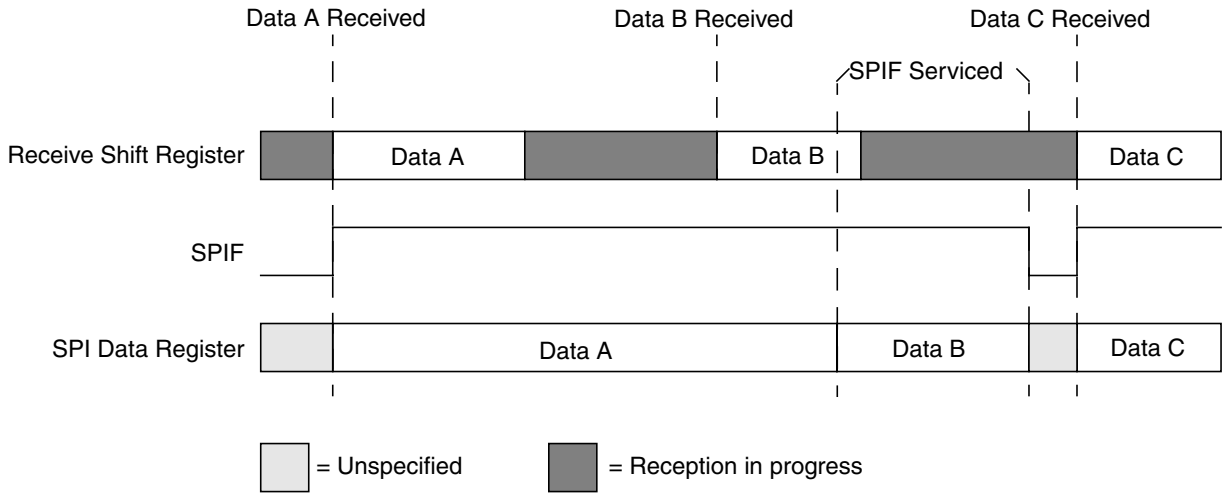



**Figure 11-9. Reception with SPIF serviced in Time**

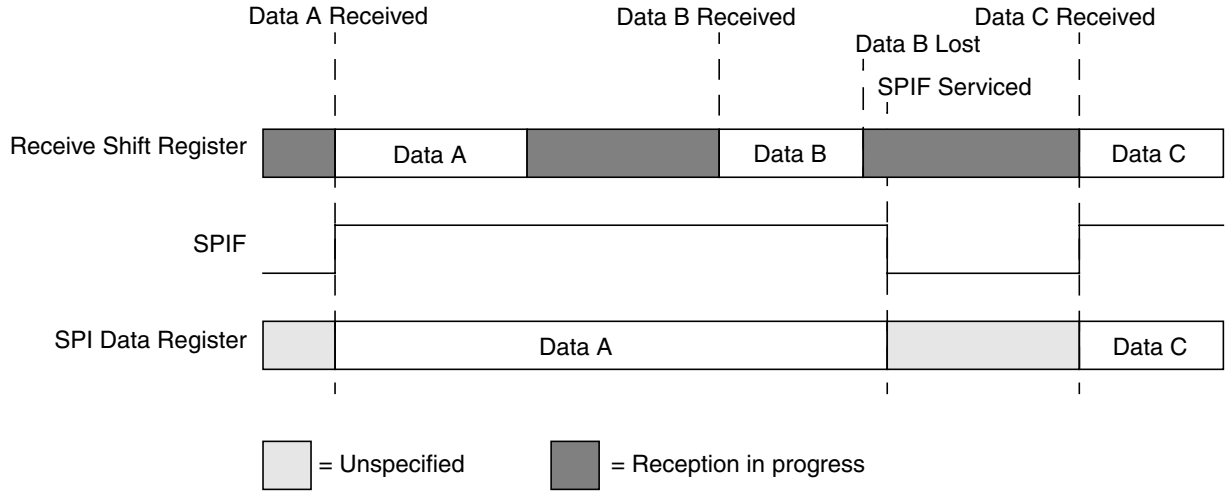



**Figure 11-10. Reception with SPIF serviced too late**

## 11.4 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 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 n-bit[1] data register in the master and the n-bit[1] data register in the slave are linked by the MOSI and MISO pins to form a distributed 2n-bit[1] register. When a data transfer operation is performed, this 2n-bit[1] register is serially shifted n[1] 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 and SPIF is cleared, received data is moved into the receive data register. This data register acts as the SPI receive data register for reads and as the SPI transmit data register for writes. A common SPI data register address is shared 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 11.4.3, “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.

**NOTE**

A change of CPOL or MSTR bit while there is a received byte pending in the receive shift register will destroy the received byte and must be avoided.

### 11.4.1 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, data immediately transfers to the shift register. Data begins shifting out on the MOSI pin under the control of the serial clock.

- Serial 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, MISO pin

  In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and BIDIROE control bits.
- $\overline{SS}$ pin

  If MODFEN and SSOE 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.

1. n depends on the selected transfer width, please refer to Section 11.3.2.2, “SPI Control Register 2 (SPICR2)

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 becomes 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 11.4.3, “Transmission Formats”).

**NOTE**

A change of the bits CPOL, CPHA, SSOE, LSBFE, XFRW, MODFEN, SPC0, or 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 must ensure that the remote slave is returned to idle state.

### 11.4.2 Slave Mode

The SPI operates in slave mode when the MSTR bit in SPI control register 1 is clear.

- Serial clock

  In slave mode, SCK is the SPI clock input from the master.

- MISO, MOSI pin

  In slave mode, the function of the serial data output pin (MISO) and serial data input pin (MOSI) is determined by the SPC0 bit and BIDIROE bit in SPI control register 2.

- $\overline{SS}$ 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 occurs.

  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 nth[1] 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, or BIDIROE with SPC0 set in slave mode will corrupt a transmission in progress and must be avoided.

## 11.4.3 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.

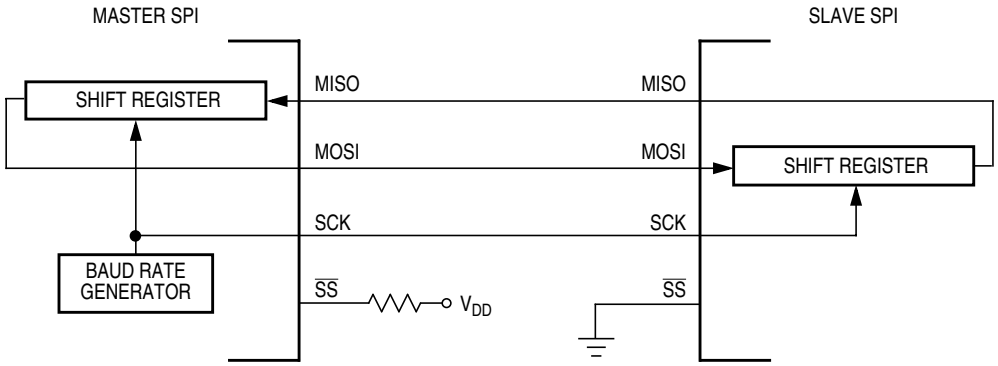



**Figure 11-11. Master/Slave Transfer Block Diagram**

1. n depends on the selected transfer width, please refer to Section 11.3.2.2, "SPI Control Register 2 (SPICR2)

### 11.4.3.1 Clock Phase and Polarity Controls

Using two bits in the SPI control register 1, 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.

### 11.4.3.2 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.

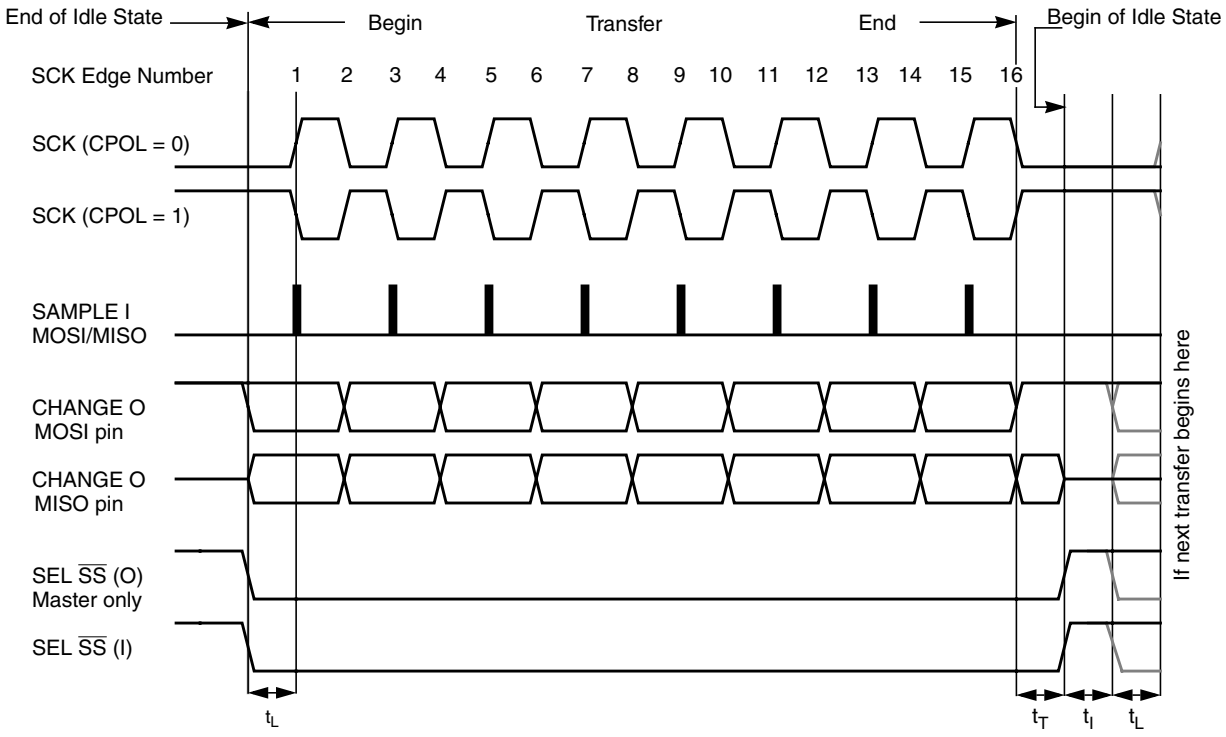
Data reception is double buffered. Data is shifted serially into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

After $2n$[1] (last) SCK edges:

- 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 11-12 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.

1. n depends on the selected transfer width, please refer to Section 11.3.2.2, "SPI Control Register 2 (SPICR2)



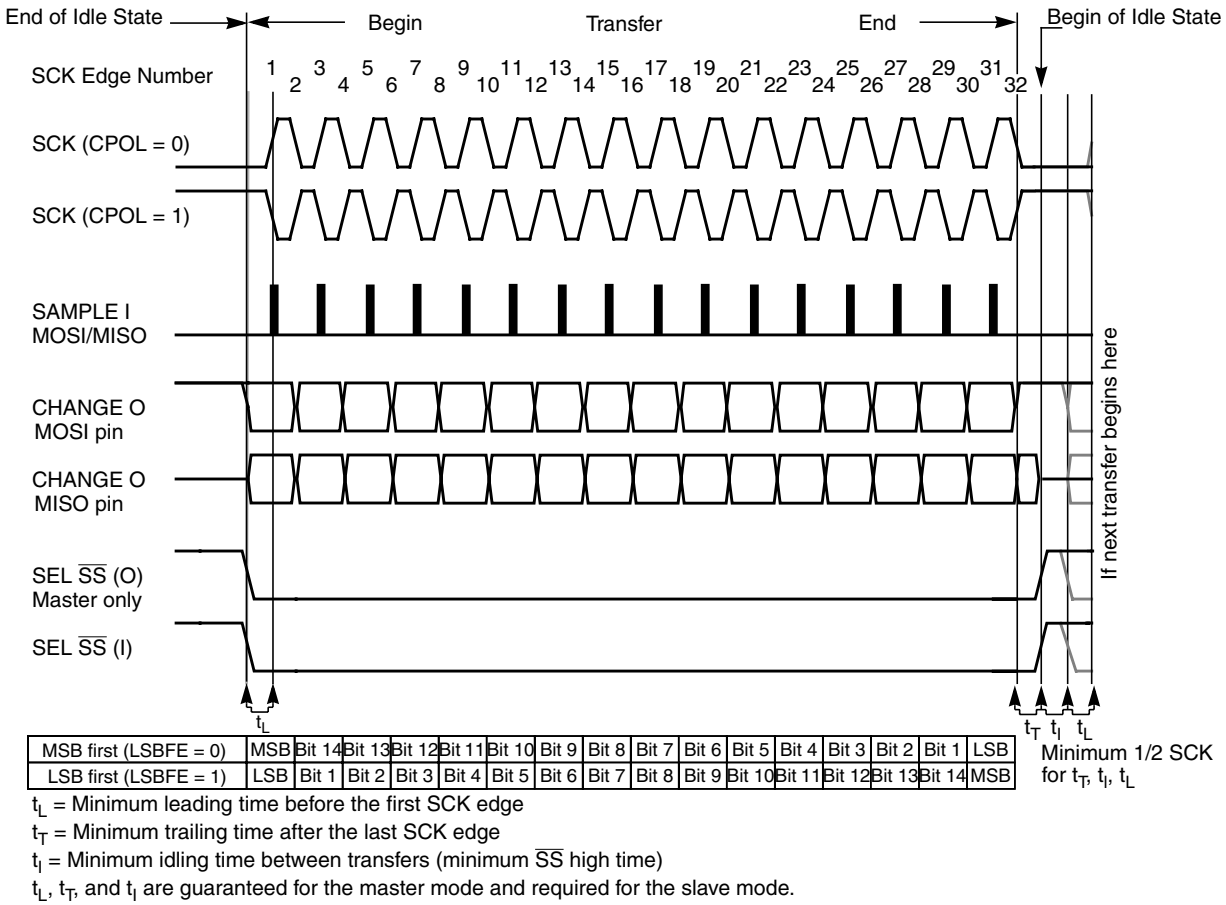

$t_L$ = Minimum leading time before the first SCK edge

$t_T$ = Minimum trailing time after the last SCK edge

$t_I$ = Minimum idling time between transfers (minimum $\overline{SS}$ high time)

$t_L$, $t_T$, and $t_I$ are guaranteed for the master mode and required for the slave mode.

**Figure 11-12. SPI Clock Format 0 (CPHA = 0), with 8-bit Transfer Width selected (XFRW = 0)**




$t_L$ = Minimum leading time before the first SCK edge
$t_T$ = Minimum trailing time after the last SCK edge
$t_I$ = Minimum idling time between transfers (minimum $\overline{SS}$ high time)
$t_L$, $t_T$, and $t_I$ are guaranteed for the master mode and required for the slave mode.

**Figure 11-13. SPI Clock Format 0 (CPHA = 0), with 16-Bit Transfer Width selected (XFRW = 1)**

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 data 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.

### 11.4.3.3 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 n[1]-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.

1. n depends on the selected transfer width, please refer to Section 11.3.2.2, “SPI Control Register 2 (SPICR2)

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.

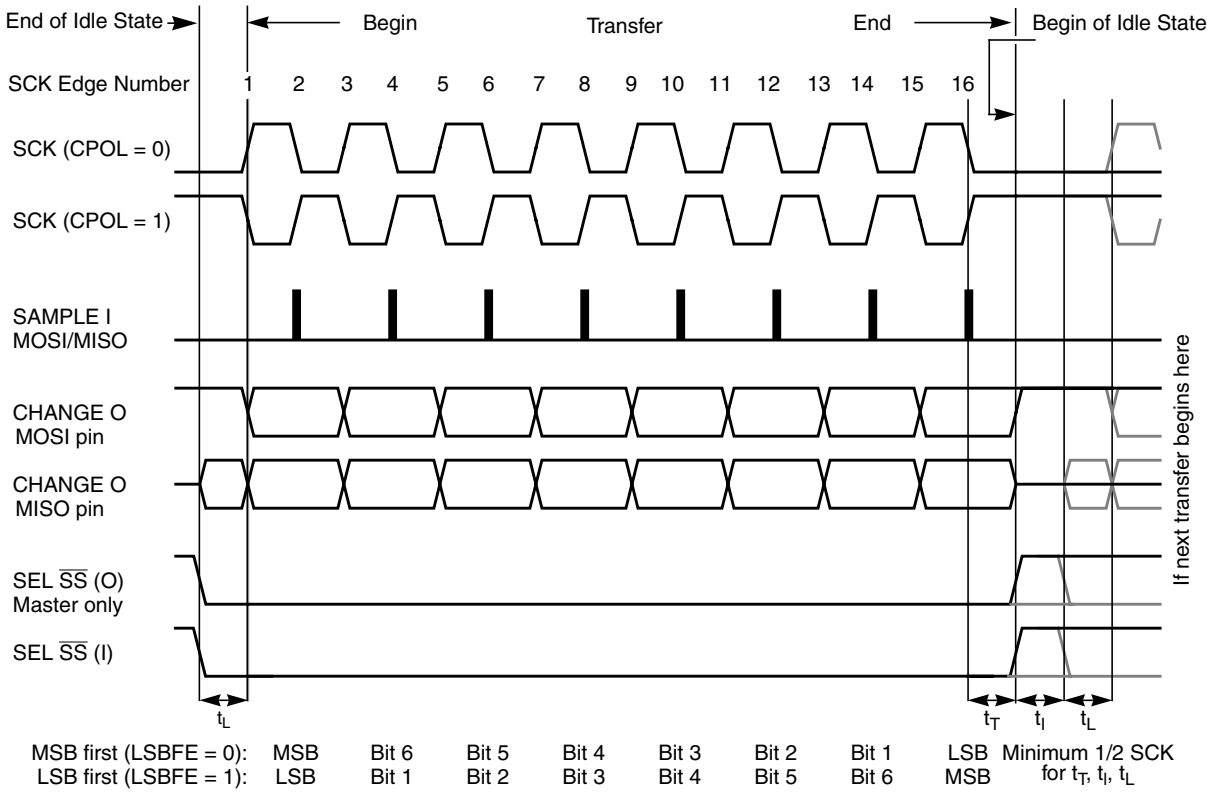
This process continues for a total of $n^1$ edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

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 $2n^1$ SCK edges:

- 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 11-14 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.




$t_L$ = 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 11-14. SPI Clock Format 1 (CPHA = 1), with 8-Bit Transfer Width selected (XFRW = 0)**

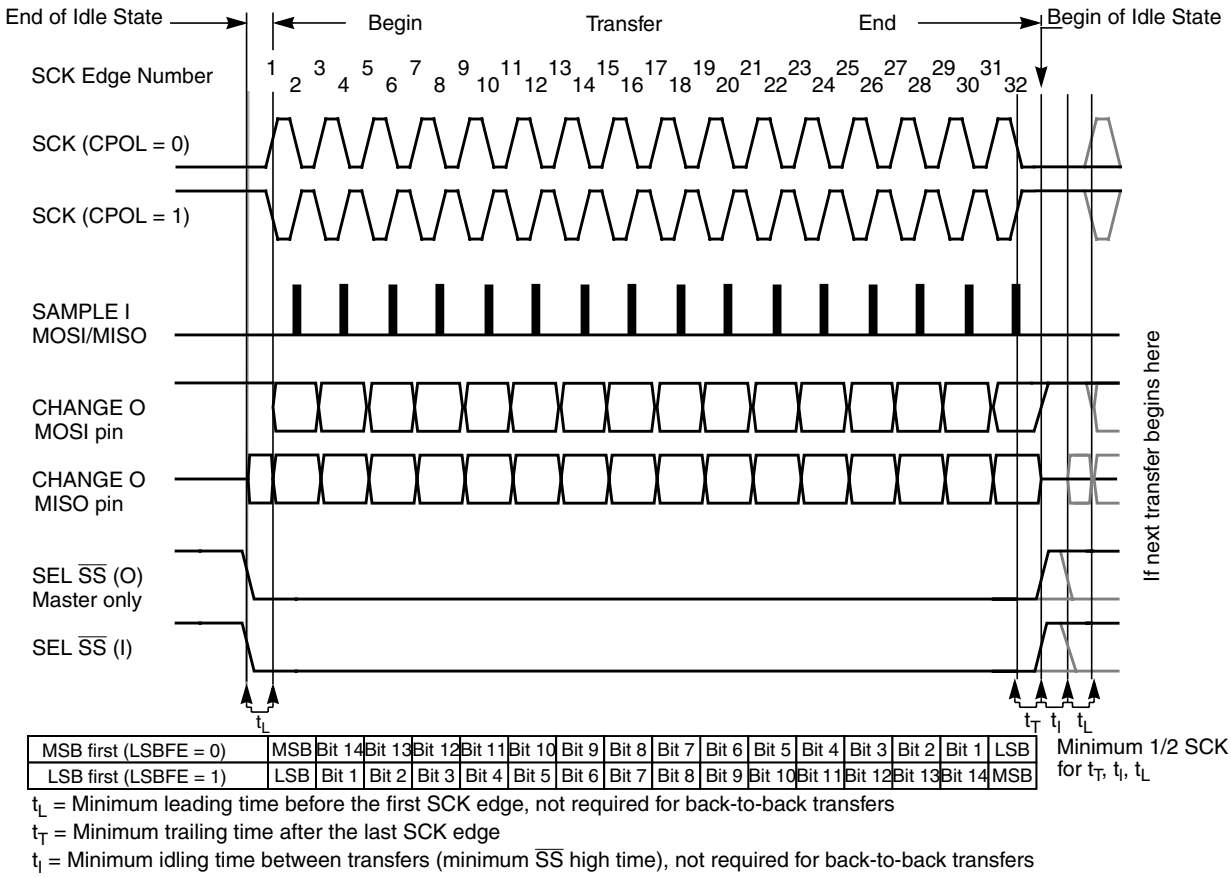



**Figure 11-15. SPI Clock Format 1 (CPHA = 1), with 16-Bit Transfer Width selected (XFRW = 1)**

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 new data is available in the SPI data register, this data is sent 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.

## 11.4.4 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 Equation 11-3.

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 11-3}$$

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 11-7 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 master mode and a serial transfer is taking place. In the other cases, the divider is disabled to decrease $I_{DD}$ current.

**NOTE**

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

## 11.4.5 Special Features

### 11.4.5.1 $\overline{SS}$ 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 11-3.

The mode fault feature is disabled while $\overline{SS}$ output is enabled.

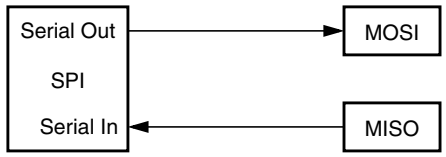
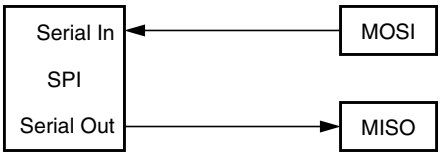
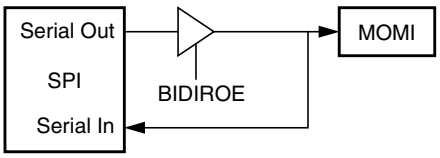
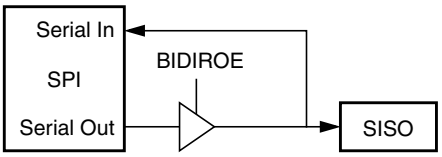
**NOTE**

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.

### 11.4.5.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI control register 2 (see Table 11-11). 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 11-11. 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 must be considered, if the MISO pin is used for another purpose.

## 11.4.6 Error Conditions

The SPI has one error condition:

- Mode fault error

### 11.4.6.1 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.

**NOTE**

If a mode fault error occurs and a received data byte is pending in the receive shift register, this data byte will be lost.

## 11.4.7 Low Power Mode Options

### 11.4.7.1 SPI 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.

### 11.4.7.2 SPI 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. In slave mode, a received byte pending in the receive shift register will be lost when entering wait or stop mode. An SPIF flag and SPIDR copy is generated only 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.

#### 11.4.7.3 SPI 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.

#### 11.4.7.4 Reset

The reset values of registers and signals are described in Section 11.3, “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 data last received from the master before the reset.
- Reading from the SPIDR after reset will always read zeros.

#### 11.4.7.5 Interrupts

The SPI only originates interrupt requests when SPI is enabled (SPE bit in SPICR1 set). The following is a description of how the SPI 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.

##### 11.4.7.5.1 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 11-3). 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 11.3.2.4, “SPI Status Register (SPISR)”.

#### 11.4.7.5.2 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 11.3.2.4, “SPI Status Register (SPISR)”.

#### 11.4.7.5.3 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 11.3.2.4, “SPI Status Register (SPISR)”.

# Chapter 12
# Timer Module (TIM16B4CV3)

**Table 12-1.**

| | | | |
|---|---|---|---|
| V03.00 | Jan. 28, 2009 | | Initial version |
| V03.01 | Aug. 26, 2009 | 12.1.2/12-368<br>Figure 1-4./1-8<br>1.3.2.15/1-18<br>12.3.2.2/12-371,<br>1.3.2.3/1-8,<br>1.3.2.4/1-9,<br>12.4.3/12-383 | - Correct typo: TSCR ->TSCR1;<br>- Correct typo: ECTxxx->TIMxxx<br>- Correct reference: Figure 1-25 -> Figure 12-22<br>- Add description, “a counter overflow when TTOV[7] is set”, to be the condition of channel 7 override event.<br>- Phrase the description of OC7M to make it more explicit |
| V03.02 | Apri,12,2010 | 12.3.2.6/12-374<br>12.3.2.9/12-376<br>12.4.3/12-383 | -Add Table 1-10<br>-update TCRE bit description<br>-add Figure 1-31 |
| V03.03 | Jan,14,2013 | | -single source generate different channel guide |

## 12.1 Introduction

The basic scalable timer consists of a 16-bit, software-programmable counter driven by a flexible programmable prescaler.

This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform.

This timer could contain up to 4 input capture/output compare channels . 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.

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.

### 12.1.1 Features

The TIM16B4CV3 includes these distinctive features:

- Up to 4 channels available. (refer to device specification for exact number)
- All channels have same input capture/output compare functionality.
- Clock prescaling.
- 16-bit counter.

## 12.1.2 Modes of Operation

Stop: Timer is off because clocks are stopped.

Freeze: Timer counter keeps on running, unless TSFRZ in TSCR1 is set to 1.

Wait: Counters keeps on running, unless TSWAI in TSCR1 is set to 1.

Normal: Timer counter keep on running, unless TEN in TSCR1 is cleared to 0.

## 12.1.3 Block Diagrams

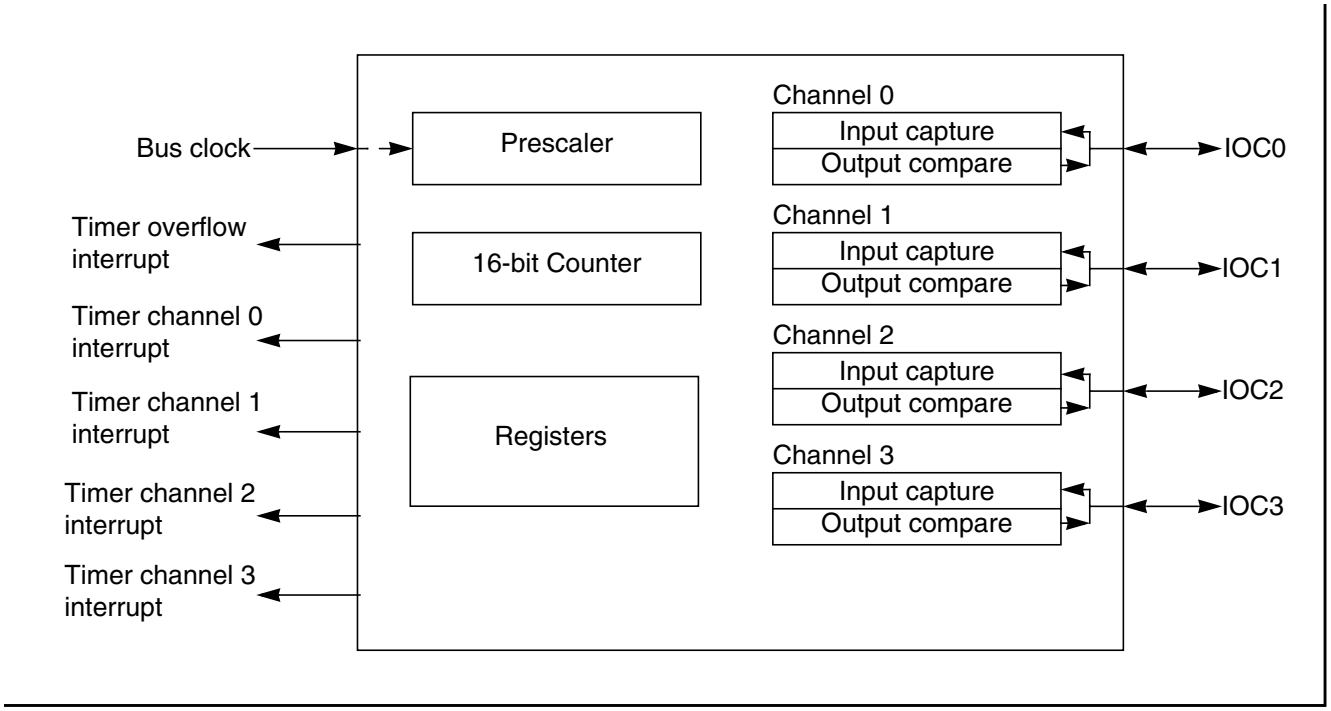



**Figure 12-1. TIM16B4CV3 Block Diagram**

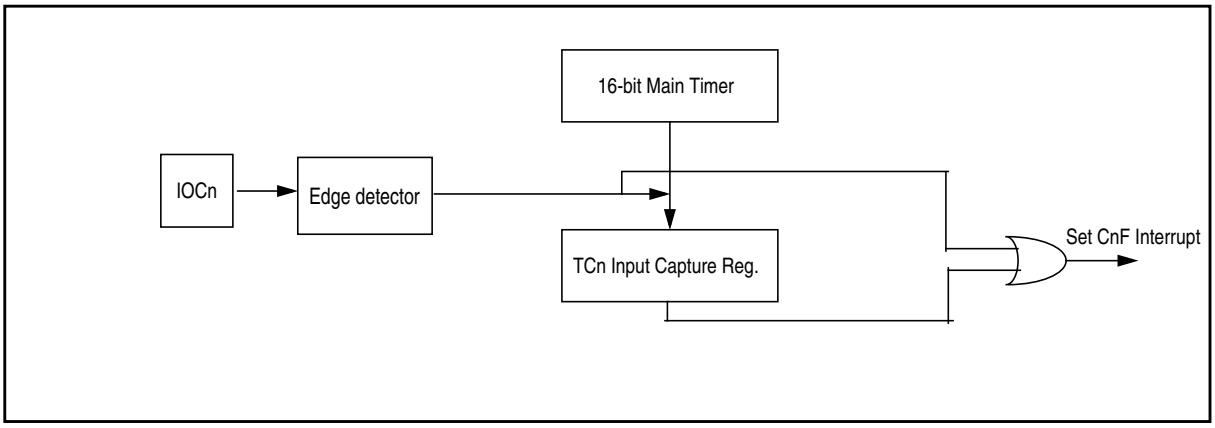



**Figure 12-2. Interrupt Flag Setting**

## 12.2 External Signal Description

The TIM16B4CV3 module has a selected number of external pins. Refer to device specification for exact number.

### 12.2.1 IOC3 - IOC0 — Input Capture and Output Compare Channel 3-0

Those pins serve as input capture or output compare for TIM16B4CV3 channel .

**NOTE**

For the description of interrupts see Section 12.6, “Interrupts”.

## 12.3 Memory Map and Register Definition

This section provides a detailed description of all memory and registers.

### 12.3.1 Module Memory Map

The memory map for the TIM16B4CV3 module is given below in Figure 12-3. 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 TIM16B4CV3 module and the address offset for each register.

### 12.3.2 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.

Only bits related to implemented channels are valid.

| Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0000 TIOS | R | RESERVED | RESERVED | RESERVED | RESERVED | IOS3 | IOS2 | IOS1 | IOS0 |
| | W | | | | | | | | |
| 0x0001 CFORC | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | RESERVED | RESERVED | RESERVED | RESERVED | FOC3 | FOC2 | FOC1 | FOC0 |
| 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 TSCR1 | R | TEN | TSWAI | TSFRZ | TFFCA | PRNT | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0007 TTOV | R | RESERVED | RESERVED | RESERVED | RESERVED | TOV3 | TOV2 | TOV1 | TOV0 |
| | W | | | | | | | | |
| 0x0008 TCTL1 | R | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED |
| | W | | | | | | | | |
| 0x0009 TCTL2 | R | OM3 | OL3 | OM2 | OL2 | OM1 | OL1 | OM0 | OL0 |
| | W | | | | | | | | |
| 0x000A TCTL3 | R | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED |
| | W | | | | | | | | |
| 0x000B TCTL4 | R | EDG3B | EDG3A | EDG2B | EDG2A | EDG1B | EDG1A | EDG0B | EDG0A |
| | W | | | | | | | | |
| 0x000C TIE | R | RESERVED | RESERVED | RESERVED | RESERVED | C3I | C2I | C1I | C0I |
| | W | | | | | | | | |
| 0x000D TSCR2 | R | TOI | 0 | 0 | 0 | RESERVED | PR2 | PR1 | PR0 |
| | W | | | | | | | | |
| 0x000E TFLG1 | R | RESERVED | RESERVED | RESERVED | RESERVED | C3F | C2F | C1F | C0F |
| | W | | | | | | | | |
| 0x000F TFLG2 | R | TOF | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0010–0x001F TCxH–TCxL[1] | 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 | | | | | | | | |
| 0x0024–0x002B Reserved | R | | | | | | | | |
| | W | | | | | | | | |
| 0x002C OCPD | R | RESERVED | RESERVED | RESERVED | RESERVED | OCPD3 | OCPD2 | OCPD1 | OCPD0 |
| | W | | | | | | | | |
| 0x002D Reserved | R | | | | | | | | |
| 0x002E PTPSR | R | PTPS7 | PTPS6 | PTPS5 | PTPS4 | PTPS3 | PTPS2 | PTPS1 | PTPS0 |
| | W | | | | | | | | |
| 0x002F Reserved | R | | | | | | | | |
| | W | | | | | | | | |

**Figure 12-3. TIM16B4CV3 Register Summary**

[1] The register is available only if corresponding channel exists.

### 12.3.2.1 Timer Input Capture/Output Compare Select (TIOS)

Module Base + 0x0000

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | RESERVED | RESERVED | RESERVED | RESERVED | IOS3 | IOS2 | IOS1 | IOS0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-4. Timer Input Capture/Output Compare Select (TIOS)**

Read: Anytime

Write: Anytime

**Table 12-2. TIOS Field Descriptions**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

| Field | Description |
|---|---|
| 3:0<br>IOS[3:0] | **Input Capture or Output Compare Channel Configuration**<br>0 The corresponding implemented channel acts as an input capture.<br>1 The corresponding implemented channel acts as an output compare. |

### 12.3.2.2 Timer Compare Force Register (CFORC)

Module Base + 0x0001

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | RESERVED | RESERVED | RESERVED | RESERVED | FOC3 | FOC2 | FOC1 | FOC0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-5. Timer Compare Force Register (CFORC)**

Read: Anytime but will always return 0x0000 (1 state is transient)

Write: Anytime

**Table 12-3. CFORC Field Descriptions**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

| Field | Description |
|---|---|
| 3:0<br>FOC[3:0] | **Note: Force Output Compare Action for Channel 3:0** — 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. 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. |

### 12.3.2.3 Timer Count Register (TCNT)

Module Base + 0x0004

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 9 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | TCNT15 | TCNT14 | TCNT13 | TCNT12 | TCNT11 | TCNT10 | TCNT9 | TCNT8 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-6. Timer Count Register High (TCNTH)**

Module Base + 0x0005

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | TCNT7 | TCNT6 | TCNT5 | TCNT4 | TCNT3 | TCNT2 | TCNT1 | TCNT0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-7. 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 .

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.

### 12.3.2.4 Timer System Control Register 1 (TSCR1)

Module Base + 0x0006

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | TEN | TSWAI | TSFRZ | TFFCA | PRNT | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 12-8. Timer System Control Register 1 (TSCR1)**

Read: Anytime

Write: Anytime

**Table 12-4. TSCR1 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>TEN | **Timer Enable**<br>0 Disables the main timer, including the counter. Can be used for reducing power consumption.<br>1 Allows the timer to function normally.<br>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<br>TSWAI | **Timer Module Stops While in Wait**<br>0 Allows the timer module to continue running during wait.<br>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.<br>TSWAI also affects pulse accumulator. |
| 5<br>TSFRZ | **Timer Stops While in Freeze Mode**<br>0 Allows the timer counter to continue running while in freeze mode.<br>1 Disables the timer counter whenever the MCU is in freeze mode. This is useful for emulation.<br>TSFRZ does not stop the pulse accumulator. |
| 4<br>TFFCA | **Timer Fast Flag Clear All**<br>0 Allows the timer flag clearing to function normally.<br>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. 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. |
| 3<br>PRNT | **Precision Timer**<br>0 Enables legacy timer. PR0, PR1, and PR2 bits of the TSCR2 register are used for timer counter prescaler selection.<br>1 Enables precision timer. All bits of the PTPSR register are used for Precision Timer Prescaler Selection, and all bits.<br>This bit is writable only once out of reset. |

## 12.3.2.5 Timer Toggle On Overflow Register 1 (TTOV)

Module Base + 0x0007

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | RESERVED | RESERVED | RESERVED | RESERVED | TOV3 | TOV2 | TOV1 | TOV0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-9. Timer Toggle On Overflow Register 1 (TTOV)**

Read: Anytime

Write: Anytime

**Table 12-5. TTOV Field Descriptions**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

| Field | Description |
|---|---|
| 3:0<br>TOV[3:0] | **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<br>0 Toggle output compare pin on overflow feature disabled.<br>1 Toggle output compare pin on overflow feature enabled. |

### 12.3.2.6 Timer Control Register 1/Timer Control Register 2 (TCTL1/TCTL2)

Module Base + 0x0008

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-10. Timer Control Register 1 (TCTL1)**

Module Base + 0x0009

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | OM3 | OL3 | OM2 | OL2 | OM1 | OL1 | OM0 | OL0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-11. Timer Control Register 2 (TCTL2)**

Read: Anytime

Write: Anytime

**Table 12-6. TCTL1/TCTL2 Field Descriptions**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero

| Field | Description |
|---|---|
| 3:0<br>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.<br>**Note:** For an output line to be driven by an OCx the OCPDx must be cleared. |
| 3:0<br>OLx | **Output Level** — These fourpairs 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.<br>**Note:** For an output line to be driven by an OCx the OCPDx must be cleared. |

**Table 12-7. Compare Result Output Action**

| OMx | OLx | Action |
|---|---|---|
| 0 | 0 | No output compare<br>action on the timer output signal |
| 0 | 1 | Toggle OCx output line |
| 1 | 0 | Clear OCx output line to zero |
| 1 | 1 | Set OCx output line to one |

### 12.3.2.7 Timer Control Register 3/Timer Control Register 4 (TCTL3 and TCTL4)

Module Base + 0x000A

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-12. Timer Control Register 3 (TCTL3)**

Module Base + 0x000B

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | EDG3B | EDG3A | EDG2B | EDG2A | EDG1B | EDG1A | EDG0B | EDG0A |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-13. Timer Control Register 4 (TCTL4)**

Read: Anytime

Write: Anytime.

**Table 12-8. TCTL3/TCTL4 Field Descriptions**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

| Field | Description |
|---|---|
| 3:0<br>EDGnB<br>EDGnA | **Input Capture Edge Control** — These four pairs of control bits configure the input capture edge detector circuits. |

**Table 12-9. 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 |

**Table 12-9. Edge Detector Circuit Configuration**

| EDGnB | EDGnA | Configuration |
|---|---|---|
| 1 | 1 | Capture on any edge (rising or falling) |

### 12.3.2.8 Timer Interrupt Enable Register (TIE)

Module Base + 0x000C

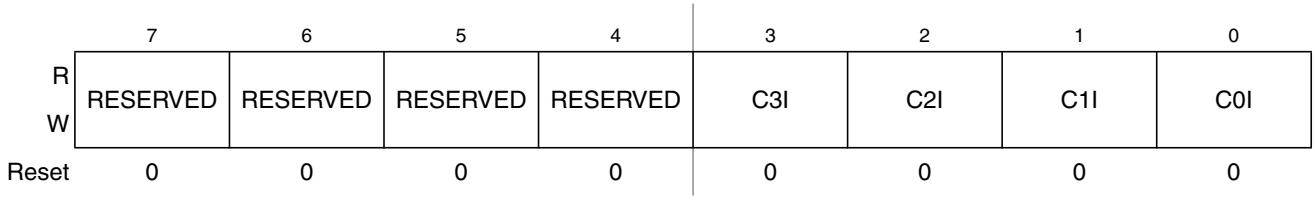



**Figure 12-14. Timer Interrupt Enable Register (TIE)**

Read: Anytime

Write: Anytime.

**Table 12-10. TIE Field Descriptions**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero

| Field | Description |
|---|---|
| 3:0<br>C3I: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. |

### 12.3.2.9 Timer System Control Register 2 (TSCR2)

Module Base + 0x000D

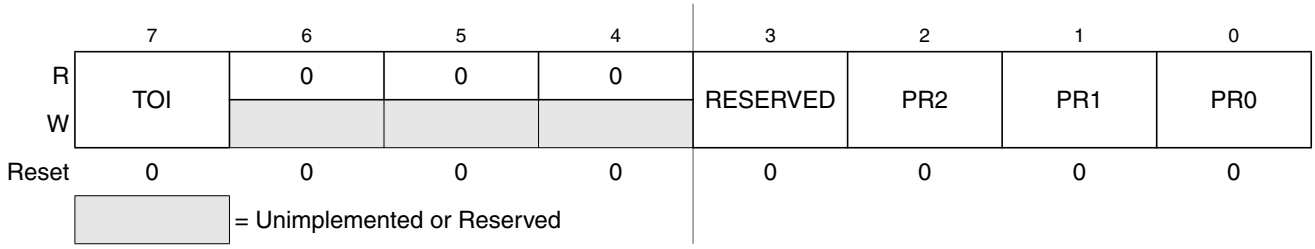



**Figure 12-15. Timer System Control Register 2 (TSCR2)**

Read: Anytime

Write: Anytime.

**Table 12-11. TSCR2 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>TOI | **Timer Overflow Interrupt Enable**<br>0 Interrupt inhibited.<br>1 Hardware interrupt requested when TOF flag set. |
| 2:0<br>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 12-12. |

**Table 12-12. 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.

### 12.3.2.10 Main Timer Interrupt Flag 1 (TFLG1)

Module Base + 0x000E

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | RESERVED | RESERVED | RESERVED | RESERVED | C3F | C2F | C1F | C0F |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-16. 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 12-13. TRLG1 Field Descriptions**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

| Field | Description |
|---|---|
| 3:0<br>C[3:0]F | **Input Capture/Output Compare Channel "x" Flag** — These flags are set when an input capture or output compare event occurs. Clearing requires writing a one to the corresponding flag bit while TEN is set to one.<br><br>**Note:** 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. |

## 12.3.2.11 Main Timer Interrupt Flag 2 (TFLG2)

Module Base + 0x000F

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | TOF | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Unimplemented or Reserved

**Figure 12-17. 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 while TEN bit of TSCR1 .

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 12-14. TRLG2 Field Descriptions**

| Field | Description |
|---|---|
| 7<br>TOF | **Timer Overflow Flag** — Set when 16-bit free-running timer overflows from 0xFFFF to 0x0000. Clearing this bit requires writing a one to bit 7 of TFLG2 register while the TEN bit of TSCR1 is set to one . |

### 12.3.2.12 Timer Input Capture/Output Compare Registers High and Low 0–3(TCxH and TCxL)

Module Base + 0x0010 = TC0H
0x0012 = TC1H
0x0014=TC2H
0x0016=TC3H
0x0018=RESERVD
0x001A=RESERVD
0x001C=RESERVD
0x001E=RESERVD

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 12-18. Timer Input Capture/Output Compare Register x High (TCxH)**

Module Base + 0x0011 = TC0L
0x0013 = TC1L
0x0015 =TC2L
0x0017=TC3L
0x0019 =RESERVD
0x001B=RESERVD
0x001D=RESERVD
0x001F=RESERVD

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>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 12-19. Timer Input Capture/Output Compare Register x Low (TCxL)**

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

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 take place before low byte otherwise it will give a different result.

### 12.3.2.13 Output Compare Pin Disconnect Register(OCPD)

Module Base + 0x002C

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | RESERVED | RESERVED | RESERVED | RESERVED | OCPD3 | OCPD2 | OCPD1 | OCPD0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-20. Output Compare Pin Disconnect Register (OCPD)**

Read: Anytime

Write: Anytime

All bits reset to zero.

**Table 12-15. OCPD Field Description**

**Note:** Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

| Field | Description |
|---|---|
| 3:0<br>OCPD[3:0] | **Output Compare Pin Disconnect Bits**<br>0 Enables the timer channel port. Output Compare action will occur on the channel pin. These bits do not affect the input capture .<br>1 Disables the timer channel port. Output Compare action will not occur on the channel pin, but the output compare flag still become set. |

### 12.3.2.14 Precision Timer Prescaler Select Register (PTPSR)

Module Base + 0x002E

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | PTPS7 | PTPS6 | PTPS5 | PTPS4 | PTPS3 | PTPS2 | PTPS1 | PTPS0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 12-21. Precision Timer Prescaler Select Register (PTPSR)**

Read: Anytime

Write: Anytime

All bits reset to zero.

**Table 12-16. PTPSR Field Descriptions**

| Field | Description |
|---|---|
| 7:0<br>PTPS[7:0] | **Precision Timer Prescaler Select Bits** — These eight bits specify the division rate of the main Timer prescaler. These are effective only when the PRNT bit of TSCR1 is set to 1. Table 12-17 shows some selection examples in this case.<br>The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero. |

The Prescaler can be calculated as follows depending on logical value of the PTPS[7:0] and PRNT bit:

PRNT = 1 : Prescaler = PTPS[7:0] + 1

**Table 12-17. Precision Timer Prescaler Selection Examples when PRNT = 1**

| PTPS7 | PTPS6 | PTPS5 | PTPS4 | PTPS3 | PTPS2 | PTPS1 | PTPS0 | Prescale Factor |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 2 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 4 |
| - | - | - | - | - | - | - | - | - |
| - | - | - | - | - | - | - | - | - |
| - | - | - | - | - | - | - | - | - |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 20 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 21 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 22 |
| - | - | - | - | - | - | - | - | - |
| - | - | - | - | - | - | - | - | - |
| - | - | - | - | - | - | - | - | - |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 253 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 254 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 255 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 256 |

## 12.4 Functional Description

This section provides a complete functional description of the timer TIM16B4CV3 block. Please refer to the detailed timer block diagram in Figure 12-22 as necessary.

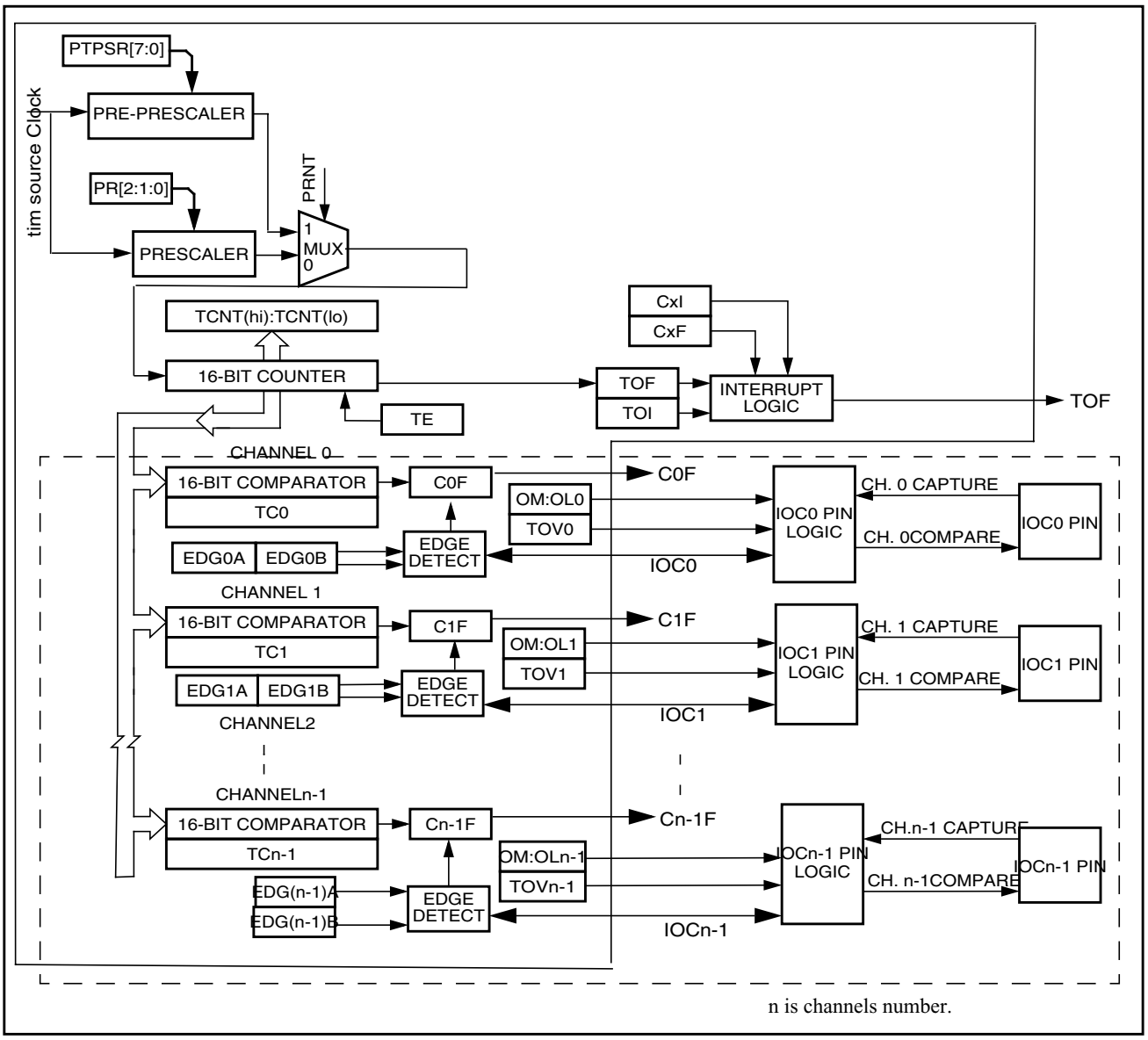



**Figure 12-22. Detailed Timer Block Diagram**

## 12.4.1 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).

The prescaler divides the Bus clock by a prescalar value. Prescaler select bits PR[2:0] of in timer system control register 2 (TSCR2) are set to define a prescalar value that generates a divide by 1, 2, 4, 8, 16, 32, 64 and 128 when the PRNT bit in TSCR1 is disabled.

By enabling the PRNT bit of the TSCR1 register, the performance of the timer can be enhanced. In this case, it is possible to set additional prescaler settings for the main timer counter in the present timer by using PTPSR[7:0] bits of PTPSR register generating divide by 1, 2, 3, 4,....20, 21, 22, 23,.....255, or 256.

### 12.4.2 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. Timer module must stay enabled (TEN bit of TSCR1 register must be set to one) while clearing CxF (writing one to CxF).

### 12.4.3 Output Compare

Setting the I/O select bit, IOSx, configures channel x when available 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 if the corresponding OCPDx bit is set to zero. An output compare on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests. Timer module must stay enabled (TEN bit of TSCR1 register must be set to one) while clearing CxF (writing one to CxF).

The output mode and level bits, OMx and OLx, select set, clear, toggle on output compare. Clearing both OMx and OLx results in no output compare action on the output compare channel pin.

Setting a force output compare bit, FOCx, causes an output compare on channel x. A forced output compare does not set the channel flag.

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.

#### 12.4.3.1 OC Channel Initialization

The internal register whose output drives OCx can be programmed before the timer drives OCx. The desired state can be programmed to this internal register by writing a one to CFORCx bit with TIOSx, OCPDx and TEN bits set to one.

Set OCx: Write a 1 to FOCx while TEN=1, IOSx=1, OMx=1, OLx=1 and OCPDx=1

Clear OCx: Write a 1 to FOCx while TEN=1, IOSx=1, OMx=1, OLx=0 and OCPDx=1

Setting OCPDx to zero allows the internal register to drive the programmed state to OCx. This allows a glitch free switch over of port from general purpose I/O to timer output once the OCPDx bit is set to zero.

## 12.5 Resets

The reset state of each individual bit is listed within Section 12.3, “Memory Map and Register Definition” which details the registers and their bit fields

## 12.6 Interrupts

This section describes interrupts originated by the TIM16B4CV3 block. Table 12-18 lists the interrupts generated by the TIM16B4CV3 to communicate with the MCU.

**Table 12-18.** TIM16B4CV3 **Interrupts**

| Interrupt | Offset | Vector | Priority | Source | Description |
|---|---|---|---|---|---|
| C[3:0]F | — | — | — | Timer Channel 3–0 | Active high timer channel interrupts 3–0 |
| TOF | — | — | — | Timer Overflow | Timer Overflow interrupt |

The TIM16B4CV3 could use up to 5 interrupt vectors. The interrupt vector offsets and interrupt numbers are chip dependent.

### 12.6.1 Channel [3:0] Interrupt (C[3:0]F)

This active high outputs will be asserted by the module to request a timer channel 7 – 0 interrupt. The TIM block only generates the interrupt and does not service it. Only bits related to implemented channels are valid.

### 12.6.2 Timer Overflow Interrupt (TOF)

This active high output will be asserted by the module to request a timer overflow interrupt. The TIM block only generates the interrupt and does not service it.

# Chapter 13
# High-Side Drivers - HSDRV (S12HSDRVV2)

**Table 13-1. Revision History Table**

| Rev. No. (Item No.) | Date (Submitted By) | Sections Affected | Substantial Change(s) |
|---|---|---|---|
| V1.00 | 10 December 2010 | All | - Initial |
| V2.00 | 07 Sep 2012 | All | - Added description and register bits for over-current masking feature |
| V2.02 | 05 August 2013 | All | - Removed open-load detection feature |

## 13.1 Introduction

The HSDRV module provides two high-side drivers typically used to drive LED or resistive loads

### 13.1.1 Features

The HSDRV module includes two independent high-side drivers with common high voltage supply. Each driver has the following features:

- Selectable gate control of high-side switches: HSDR[1:0] register bits or PWM or timer channels.
- Over-current shutdown for the drivers, while they are enabled, comprising:
  - Interrupt flag generation
  - Driver shutdown
  - Optional masking window

### 13.1.2 Modes of Operation

The HSDRV module behaves as follows in the system power modes:

1. MCU run mode

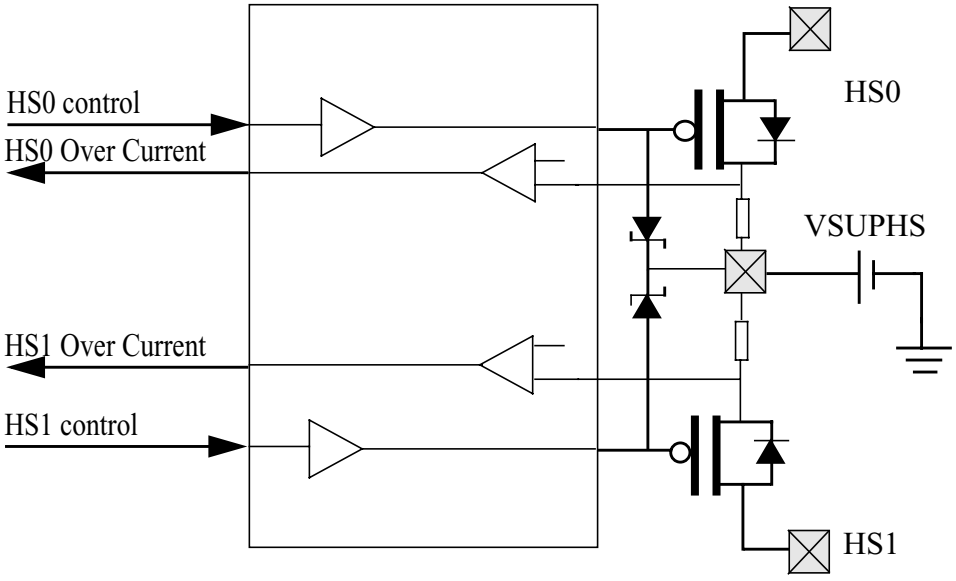
   The activation of the HSE0 or HSE1 bits enable the related high-side driver. The driver is controlled by the selected source.
2. MCU stop mode

During stop mode operation the high-side drivers are shut down, i.e. the high-side drivers are disabled and their drivers are turned off. The bits in the data register which control the drivers (HSDRx) are cleared automatically. After returning from stop mode the drivers are re-enabled and the state of the HSE bits are automatically set. If the data register bits (HSDRx) were chosen as source in PIM module, then the respective high-side driver stays turned off until the software sets the associated bit in the data register (HSDRx). When the timer or PWM were chosen as source, the respective high-side driver is controlled by the timer or PWM without further handling. When it is required that the driver stays turned off after the stop mode for this case (PWM or timer), the software must take the appropriate action to turn off the driver before entering stop mode.

### 13.1.3 Block Diagram

Figure 13-1 shows a block diagram of the HSDRV module. The module consists of a control and an output stage. Internal functions can be routed to control the high-side drivers. See PIM chapter for routing options.

**Figure 13-1. HSDRV Block Diagram**



## 13.2 External Signal Description

Table 13-2 shows the external pins associated with the HSDRV module.

**Table 13-2. HSDRV Signal Properties**

| Name | Function | Reset State |
|---|---|---|
| HS0 | High-side driver output 0 | disabled (off) |
| HS1 | High-side driver output 1 | disabled (off) |
| VSUPHS | High Voltage Power Supply for both high side drivers | disabled (off) |

### 13.2.1 HS0, HS1— High Side Driver Pins

Outputs of the two high-side drivers are intended to drive LEDs or resistive loads.

### 13.2.2 VSUPHS — High Side Driver Power Pin

Power supply for both high-side drivers.

This pin must be connected to the main power supply with the appropriate reverse battery protection network.

## 13.3 Memory Map and Register Definition

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

### 13.3.1 Module Memory Map

A summary of registers associated with the HSDRV module is shown in Table 13-3. Detailed descriptions of the registers and bits are given in the following sections.

**NOTE**

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

**Table 13-3. Register Summary**

| Address Offset<br>Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0000<br>HSDR | R | 0 | 0 | 0 | 0 | 0 | 0 | HSDR1 | HSDR0 |
| | W | | | | | | | | |
| 0x0001<br>HSCR | R | 0 | 0 | HSOCME1 | HSOCME0 | 0 | 0 | HSE1 | HSE0 |
| | W | | | | | | | | |
| 0x0002<br>Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |

**Table 13-3. Register Summary**

| Address Offset / Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0003 Reserved | R | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| | W | | | | | | | | |
| 0x0004 Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0005 Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0006 HSIE | R | HSOCIE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0007 HSIF | R | 0 | 0 | 0 | 0 | 0 | 0 | HSOCIF1 | HSOCIF0 |
| | W | | | | | | | | |

## 13.3.2 Register Definition

## 13.3.3 Port HS Data Register (HSDR)

Module Base + 0x0000 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | HSDR1 | HSDR0 |
| W | | | | | | | | |
| Altern. Read Function | — | — | — | — | — | — | OC[2] | OC[2] |
| | — | — | — | — | — | — | PWM[2] | PWM[2] |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 13-2. Port HS Data Register (HSDR)**

1 Read: Anytime The data source (HSDRx or alternate function) depends on the HSE control bit settings.
Write: Anytime

2 See PIM chapter for detailed routing description.

**Table 13-4. PTHS Register Field Descriptions**

| Field | Description |
|---|---|
| 1-0 HSDRx | **Port HS Data Bits**—Data registers or routed timer outputs or routed PWM outputs<br>These register bits can be used to control the high-side drivers if selected as control source. See PIM section for routing details.<br>If the associated HSEx bit is set to 0, a read returns the value of the Port HS Data Register (HSDRx).<br>If the associated HSEx bit is set to 1, a read returns the value of the selected control source for the driver.<br>When entering in STOP mode the Port HS Data Register (HSDRx) is cleared.<br><br>0 High-side driver is turned off<br>1 High-side driver is turned on<br><br>**NOTE**<br>After enabling the high-side driver with the HSEx bit in HSCR register, the user must wait a minimum settling time $t_{HS_settling}$ before turning on the high-side driver. |

## 13.3.4 HSDRV Configuration Register (HSCR)

Module Base + 0x0001 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | HSOCME1 | HSOCME0 | 0 | 0 | HSE1 | HSE0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 13-3. HSDRV Configuration Register (HSCR)**

[1] Read: Anytime
Write: Anytime, except HSOCME (see description)

**Table 13-5. HSCR Register Field Descriptions**

| Field | Description |
|---|---|
| 5-4 HSOCMEx | **HSDRV Over-Current Mask Enable**<br>These bits enable the masking of the over-current shutdown for $t_{HSOCM}$ on the related high-side driver, after switching the driver on. These bits are only writable when the related high-side driver is disabled (HSEx=0)<br><br>0 over-current masking window is disabled<br>1 over-current masking window is enabled |
| 1-0 HSEx | **HSDRV Enable**<br>These bits control the bias of the related high-side driver circuit.<br><br>0 High-side driver is disabled<br>1 High-side driver is enabled<br><br>**NOTE**<br>After enabling the high-side driver (write 1 to HSEx) a settling time $t_{HS_settling}$ is required before the high-side driver is allowed to be turned on (e.g. by writing HSDRx bits). |

## 13.3.5 Reserved Register

Module Base + 0x0003 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| Reset | x | x | x | x | x | x | x | x |

= Unimplemented

**Figure 13-4. Reserved Register**

[1] Read: Anytime
Write: Only in special mode

**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 mode can alter the module's functionality.

**Table 13-6. Reserved Register Field Descriptions**

| Field | Description |
|---|---|
| 7-0<br>Reserved | These reserved bits are used for test purposes. Writing to these bits can alter the module functionality. |

## 13.3.6 HSDRV Interrupt Enable Register (HSIE)

Module Base + 0x0006 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | HSOCIE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 13-5. HSDRV Interrupt Enable Register (HSIE)**

[1] Read: Anytime
Write: Anytime

**Table 13-7. HSIE Register Field Descriptions**

| Field | Description |
|---|---|
| 7<br>HSOCIE | **HSDRV Over-Current Interrupt Enable**<br>0 Interrupt request is disabled<br>1Interrupt is requested whenever a HSOCIFx flag is set |

## 13.3.7 HSDRV Interrupt Flag Register (HSIF)

Module Base + 0x0007 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | HSOCIF1 | HSOCIF0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 13-6. HSDRV Interrupt Flag Register (HSIF)**

[1] Read: Anytime
Write: Write 1 to clear, writing 0 has no effect

**Table 13-8. HSIF Register Field Descriptions**

| Field | Description |
|---|---|
| 1-0<br>HSOCIFx | **HSDRV Over-Current Interrupt Flag**<br>These flags are set to 1 when an over-current event occurs on the related high-side driver ($\lvert I_{HS} \rvert > \lvert I_{OCTHSX} \rvert$). While set the related high-side driver is turned off.<br>Once these flags are cleared, the related driver is controlled again by the source selected in PIM module.<br><br>0 No over-current event occurred since last clearing of flag<br>1 An over-current event occurred since last clearing of flag |

## 13.4 Functional Description

### 13.4.1 General

The HSDRV module provides two high-side drivers able to drive LED or resistive loads. The driver can be controlled directly through register bits or alternatively by dedicated timer or PWM channels. See PIM chapter for routing details.

### 13.4.2 Over-Current Shutdown

Each high-side driver has an over-current shutdown feature with a current threshold of $I_{OCTHSX}$.

If an over-current is detected the related interrupt flag (HSOCIF1 or HSOCIF0) is set in the HSDRV Interrupt Flag Register (HSIF). As long as the over-current interrupt flag remains set, the related high-side driver is turned off to protect the circuit. Clearing the related over-current interrupt flag returns back the control to the selected source in the PIM module.

The over-current detection and driver shutdown can be masked for an initial $T_{HSOCM}$ after switching the driver on. This can be achieved by writing the related HSOCME register bit in the HSCR register to 1. The HSOCME bits are only writable while the related driver is disabled (HSE=0).

### 13.4.3 Interrupts

This section describes the interrupt generated by HSDRV module. The interrupt is only available in MCU run mode. Entering and exiting MCU stop mode has no effect on the interrupt flags.

The HSDRV interrupt vector is named in Table 13-9. Vector addresses and interrupt priorities are defined at MCU level.

#### 13.4.3.1 HSDRV Over Current Interrupt (HSOCI)

**Table 13-9. HSDRV Interrupt Sources**

| Module Interrupt Source | Module Internal Interrupt Source | Local Enable |
|---|---|---|
| HSDRV Interrupt (HSI) | HSDRV Over-Current Interrupt (HSOCI) | HSOCIE = 1 |

If an over-current is detected the related interrupt flag HSOCIFx asserts. Depending on the setting of the HSDRV Error Interrupt Enable (HSOCIE) bit an interrupt is requested.

# Chapter 14
# Low-Side Drivers - LSDRV (S12LSDRV1)

**Table 14-1. Revision History Table**

| Rev. No. (Item No.) | Date (Submitted By) | Sections Affected | Substantial Change(s) |
|---|---|---|---|
| V1.01 | 22 February 2011 | All | - Added clarification to open-load mechanism in over-current conditions |
| V1.02 | 12 April 2011 | All | - improved clarification to open-load mechanism in over-current conditions<br>- corrected typos |
| V1.03 | 3 April 2011 | Register Descriptions for LSDR and LSCR | - added Note on considering settling time $t_{LS_settling}$ to LSDR and LSCR register description<br>- added Note on how to disable the low-side driver to LSDR register description |
| V1.04 | 29 January 2013 | All | - Cleaning |

## 14.1 Introduction

The LSDRV module provides two low-side drivers typically used to drive inductive loads (relays).

### 14.1.1 Features

The LSDRV module includes two independent low side drivers with common current sink. Each driver has the following features:

- Selectable driver control of low-side switches: LSDRx register bits, PWM or timer channels. See PIM chapter for routing options.
- Open-load detection while enabled
  - While driver off: selectable high-load resistance open-load detection
- Over-current protection with shutdown and interrupt while enabled
- Active clamp to protect the device against over-voltage when the power transistor that is driving an inductive load (relay) is turned off.

### 14.1.2 Modes of Operation

The LSDRV module behaves as follows in the system operating modes:

1. MCU run mode

   The activation of the LSE0 or LSE1 bits enable the related low-side driver. The driver is controlled by the selected source in the Port Integration Module (see PIM chapter).

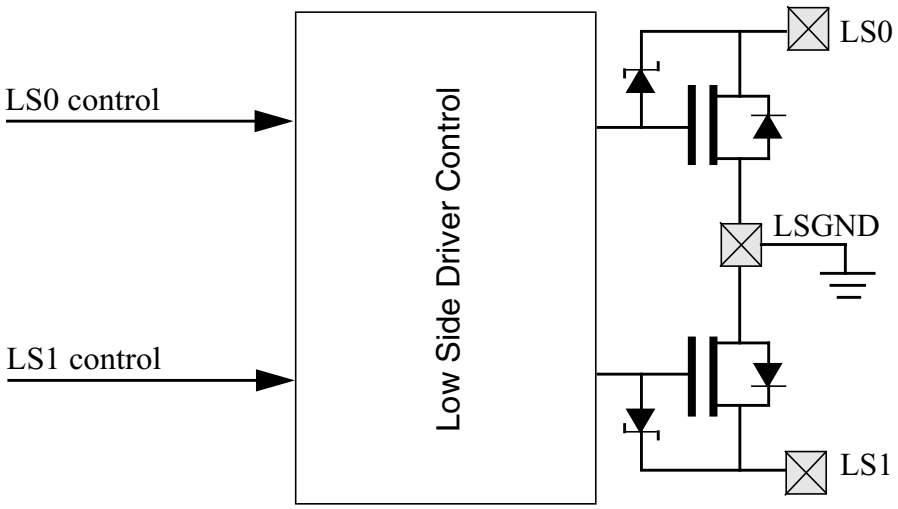
2. MCU stop mode

   During stop mode operation the low-side drivers are shut down, i.e. the low-side drivers are disabled and their drivers are turned off. The bits in the data register which control the drivers (LSDRx) are cleared automatically. After returning from stop mode the drivers are re-enabled. If the data register bits (LSDRx) were chosen as source in PIM module, then the respective low-side driver stays turned off until the software sets the associated bit in the data register (LSDRx). When the timer or PWM were chosen as source, the respective low-side driver is controlled by the timer or PWM without further handling. When it is required that the driver stays turned off after the stop mode for this case (PWM or timer), the software must take the appropriate action to turn off the driver before entering stop mode.

### 14.1.3 Block Diagram

Figure 14-1 shows a block diagram of the LSDRV module. The module consists of a control and an output stage. Internal functions can be routed to control the low-side drivers. See PIM chapter for routing options.

**Figure 14-1. LSDRV Block Diagram**



## 14.2 External Signal Description

Table 14-2 shows the external pins associated with the LSDRV module.

**Table 14-2. LSDRV Signal Properties**

| Name | Function | Reset State |
|---|---|---|
| LS0 | Low-side driver output 0 | disabled (off) |
| LS1 | Low-side driver output 1 | disabled (off) |
| LSGND | Low-side driver ground pin | — |

### 14.2.1 LS0, LS1— Low Side Driver Pins

Outputs of the two low-side drivers intended to drive inductive loads (relays).

### 14.2.2 LSGND — Low Side Driver Ground Pin

Common current sink for both low-side driver pins. This pin should be connected on-board to the common ground.

## 14.3 Memory Map and Register Definition

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

### 14.3.1 Module Memory Map

A summary of registers associated with the LSDRV module is shown in Table 14-3. Detailed descriptions of the registers and bits are given in the following sections.

**NOTE**

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

**Table 14-3. Register Summary**

| Address Offset<br>Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0000<br>LSDR | R | 0 | 0 | 0 | 0 | 0 | 0 | LSDR1 | LSDR0 |
| | W | | | | | | | | |
| 0x0001<br>LSCR | R | 0 | 0 | 0 | 0 | LSOLE1 | LSOLE0 | LSE1 | LSE0 |
| | W | | | | | | | | |
| 0x0002<br>Reserved | R | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| | W | | | | | | | | |

**Table 14-3. Register Summary**

| Address Offset<br>Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0003<br>Reserved | R<br>W | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| 0x0004<br>Reserved | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0005<br>LSSR | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | LSOL1 | LSOL0 |
| 0x0006<br>LSIE | R<br>W | LSOCIE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0007<br>LSIF | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | LSOCIF1 | LSOCIF0 |

## 14.3.2 Register Definition

## 14.3.3 Port LS Data Register (LSDR)

Module Base + 0x0000 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | LSDR1 | LSDR0 |
| W | | | | | | | | |
| Altern. Read Function | 0 | 0 | 0 | 0 | 0 | 0 | OC[2] / PWM[2] | OC[2] / PWM[2] |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 14-2. Port LS Data Register (LSDR)**

1 Read: Anytime. The data source (LSDRx or alternate function) depends on the LSE control bit settings.
Write: Anytime

2 See PIM chapter for detailed routing description.

**Table 14-4. LSDR Register Field Descriptions**

| Field | Description |
|---|---|
| 1-0 LSDRx | **Port LS Data Bits**—Data registers or routed timer outputs or routed PWM outputs<br>These register bits can be used to control the low-side drivers if selected as control source. See PIM section for routing details.<br>If the associated LSEx bit is set to 0, a read returns the value of the Port LS Data Register (LSDRx).<br>If the associated LSEx bit is set to 1, a read returns the value of the selected control source for the driver.<br>When entering in STOP mode the Port LS Data Register (LSDR) is cleared.<br><br>0 Low-side driver is turned off<br>1 Low-side driver is turned on<br><br>**NOTE**<br>After enabling the low-side driver with the LSEx bit in LSCR register, the user must wait a minimum settling time $t_{LS_settling}$ before turning on the low-side driver.<br><br>**NOTE**<br>The low-side driver should be turned off (e.g. LDSRx=0 or OC=0 or PWM=0) and the load should be de-energized before going into Stop Mode or disabling the low-side driver with the LSEx bits. |

## 14.3.4 LSDRV Configuration Register (LSCR)

Module Base + 0x0001 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | LSOLE1 | LSOLE0 | LSE1 | LSE0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 14-3. LSDRV Configuration Register (LSCR)**

[1] Read: Anytime
Write: Anytime

**Table 14-5. LSCR Register Field Descriptions**

| Field | Description |
|---|---|
| 3-2<br>LSOLEx | **LSDRV High-Load Resistance Open-Load Detection Enable**<br>These bits enable the measurement function to detect an open-load condition on the related low-side driver operating on high-load resistance loads. If the low-side driver is enabled and is not being driven by the selected source, then the high-load resistance detection circuit is activated when this bit is set to ‘1’.<br><br>0 high-load resistance open-load detection is disabled<br>1 high-load resistance open-load detection is enabled |
| 1-0<br>LSEx | **LSDRV Enable**<br>These bits control the bias of the related low-side driver circuit.<br><br>0 Low-side driver is disabled.<br>1 Low-side driver is enabled.<br><br>**NOTE**<br>After enabling the low-side driver (write “1” to LSEx) a settling time $t_{LS_settling}$ is required before the low-side driver is allowed to be turned on (e.g. by writing LSDRx bits). |

## 14.3.5 LSDRV Status Register (LSSR)

Module Base + 0x0005 Access: User read[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | LSOL1 | LSOL0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 14-4. LSDRV Status Register (LSSR)**

[1] Read: Anytime
Write: No Write

**Table 14-6. LSSR - Register Field Descriptions**

| Field | Description |
|---|---|
| 1-0<br>LSOLx | **LSDRV Open-Load Status Bits**<br>These bits reflect the open-load condition status on each driver related pin.<br>This open-load monitoring will only be active if the detection function is enabled (bits LSOLEx) and the corresponding low-side driver is enabled and turned off.<br>A delay of $t_{HLROLDT}$ must be granted after enabling the high-load resistance open-load detection function in order to read valid data.<br><br>0 Open-load condition $I_{LS} < I_{HLROLDC}$<br>1 Open-load condition $I_{LS} \geq I_{HLROLDC}$ |

## 14.3.6 LSDRV Interrupt Enable Register (LSIE)

Module Base + 0x0006 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | LSOCIE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 14-5. LSDRV Interrupt Enable Register (LSIE)**

[1] Read: Anytime
Write: Anytime

**Table 14-7. LSIE Register Field Descriptions**

| Field | Description |
|---|---|
| 7<br>LSOCIE | **LSDRV Error Interrupt Enable**<br>0 Interrupt request is disabled<br>1 Interrupt will be requested whenever a LSOCIFx flag is set |

## 14.3.7 LSDRV Interrupt Flag Register (LSIF)

Module Base + 0x0007 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | LSOCIF1 | LSOCIF0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 14-6. LSDRV Interrupt Flag Register (LSIF)**

[1] Read: Anytime
Write: Write 1 to clear, writing 0 has no effect

**Table 14-8. LSIF Register Field Descriptions**

| Field | Description |
|---|---|
| 1-0<br>LSOCIFx | **LSDRV Over-Current Interrupt Flag**<br>These flags are set to 1 when an over-current event occurs on the related low-side driver ($I_{LS} > I_{LIMLSX}$). While set the related low-side driver is turned off.<br>Once these flags are cleared, the related driver is again driven by the source selected in PIM module.<br><br>0 No over-current event occurred since last clearing of flag<br>1 An over-current event occurred since last clearing of flag |

## 14.4 Functional Description

### 14.4.1 General

The LSDRV module provides two low-side drivers able to drive inductive loads (relays). The driver can be controlled directly through register bits or alternatively by dedicated timer or PWM channels. See PIM section for routing details.

Both drivers feature an open-load and over-current detection described in the following sub-sections. In addition to this an active clamp (for driving relays) is protecting each driver stage. The active clamp will turn on a low-side FET if the voltage on a pin exceeds $V_{CLAMP}$ when the driver is turned off.

### 14.4.2 Open-Load Detection

A “High-load resistance Open Load Detection” can be enabled for each driver by setting the corresponding LSOLEx bit (refer to Section 14.3.4, “LSDRV Configuration Register (LSCR)”. This detection will only be executed when the driver is enabled and it is not being driven (LSDRx = 0). That is because the measurement point is between the load and the driver, and the current should not go through the driver. To detect an open-load condition the voltage will be observed at the output from the driver. Then if the driving pin LSx stays at low voltage which is approximately LSGND, there is no load for the corresponding low-side driver.

An open-load condition is flagged with bits LSOL0 and LSOL1 in the LSDRV Status Register (LSSR).

### 14.4.3 Over-Current Detection

Each low-side driver has an over-current detection while enabled with a current threshold of $I_{LIMLSX}$.

If over-current is detected the related interrupt flag (LSOCIF1 or LSOCIF0) is set in the LSDRV Interrupt Flag Register (LSIF). As long as the over-current interrupt flag remains set the related low-side driver is turned off to protect the circuit.Clearing the related over-current interrupt flag returns back the control of the driver to the selected source in the PIM module.

**NOTE**

The open-load detection is only active if the selected source (e.g. PWM, Timer, LSDRx) for the low-side driver is turned off.

### 14.4.4 Interrupts

This section describes the interrupt generated by LSDRV module. The interrupt is only available in MCU run mode. Entering and exiting MCU stop mode has no effect on the interrupt flags.

The LSDRV interrupt vector is named in Table 14-9. Vector addresses and interrupt priorities are defined at MCU level.

**Table 14-9. LSDRV Interrupt Sources**

| Module Interrupt Source | Module Internal Interrupt Source | Local Enable |
|---|---|---|
| LSDRV Interrupt (LSI) | LSDRV Over-Current Interrupt (LSOCI) | LSOCIE=1 |

#### 14.4.4.1 LSDRV Over Current Interrupt (LSOCI)

If a low-side driver over-current event is detected the related interrupt flag LSOCIFx asserts. Depending on the setting of the LSDRV Error Interrupt Enable (LSOCIE) bit an interrupt is requested.

# Chapter 15
# LIN Physical Layer (S12LINPHYV2)

**Table 15-1. Revision History Table**

| Rev. No. (Item No.) | Date (Submitted By) | Sections Affected | Substantial Change(s) |
|---|---|---|---|
| V02.11 | 19 Sep 2013 | All | - Removed preliminary note.<br>- Fixed grammar and spelling throughout the document. |
| V02.12 | 20 Sep 2013 | Standby Mode | - Clarified Standby mode behavior. |
| V02.13 | 8 Oct 2013 | All | - More grammar, spelling, and formating fixes throughout the document. |

## 15.1 Introduction

The LIN (Local Interconnect Network) bus pin provides a physical layer for single-wire communication in automotive applications. The LIN Physical Layer is designed to meet the LIN Physical Layer 2.2 specification from LIN consortium.

### 15.1.1 Features

The LIN Physical Layer module includes the following distinctive features:

- Compliant with LIN Physical Layer 2.2 specification.
- Compliant with the SAE J2602-2 LIN standard.
- Standby mode with glitch-filtered wake-up.
- Slew rate selection optimized for the baud rates: 10.4 kbit/s, 20 kbit/s and Fast Mode (up to 250 kbit/s).
- Switchable 34 kΩ/330 kΩ pullup resistors (in shutdown mode, 330 kΩ only)
- Current limitation for LIN Bus pin falling edge.
- Overcurrent protection.
- LIN TxD-dominant timeout feature monitoring the LPTxD signal.
- Automatic transmitter shutdown in case of an overcurrent or TxD-dominant timeout.
- Fulfills the OEM “Hardware Requirements for LIN (CAN and FlexRay) Interfaces in Automotive Applications” v1.3.

The LIN transmitter is a low-side MOSFET with current limitation and overcurrent transmitter shutdown. A selectable internal pullup resistor with a serial diode structure is integrated, so no external pullup components are required for the application in a slave node. To be used as a master node, an external

resistor of 1 kΩ must be placed in parallel between VLINSUP and the LIN Bus pin, with a diode between VLINSUP and the resistor. The fall time from recessive to dominant and the rise time from dominant to recessive is selectable and controlled to guarantee communication quality and reduce EMC emissions. The symmetry between both slopes is guaranteed.

## 15.1.2 Modes of Operation

The LIN Physical Layer can operate in the following four modes:

1. **Shutdown Mode**

   The LIN Physical Layer is fully disabled. No wake-up functionality is available. The internal pullup resistor is replaced by a high ohmic one (330 kΩ) to maintain the LIN Bus pin in the recessive state. All registers are accessible.

2. **Normal Mode**

   The full functionality is available. Both receiver and transmitter are enabled.

3. **Receive Only Mode**

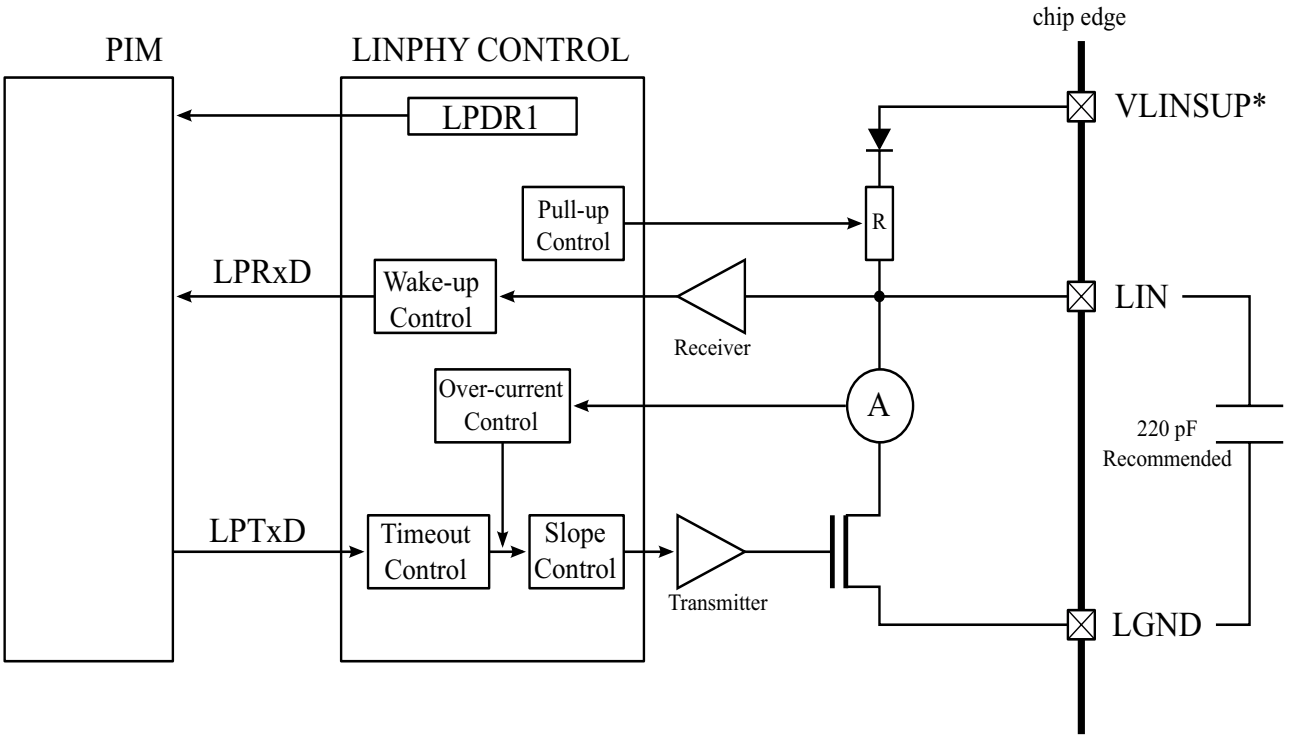
   The transmitter is disabled and the receiver is running in full performance mode.

4. **Standby Mode**

   The transmitter of the LIN Physical Layer is disabled. If the wake-up feature is enabled, the internal pullup resistor can be selected (330 kΩ or 34 kΩ). The receiver enters a low power mode and optionally it can pass wake-up events to the Serial Communication Interface (SCI). If the wake-up feature is enabled and if the LIN Bus pin is driven with a dominant level longer than $t_{WUFR}$ followed by a rising edge, the LIN Physical Layer sends a wake-up pulse to the SCI, which requests a wake-up interrupt. (This feature is only available if the LIN Physical Layer is routed to the SCI).

## 15.1.3 Block Diagram

Figure 15-1 shows the block diagram of the LIN Physical Layer. The module consists of a receiver with wake-up control, a transmitter with slope and timeout control, a current sensor with overcurrent protection as well as a registers control block.




*The VLINSUP supply mapping is described in device level documentation

**Figure 15-1. LIN Physical Layer Block Diagram**

**NOTE**

The external 220 pF capacitance between LIN and LGND is strongly recommended for correct operation.

## 15.2 External Signal Description

This section lists and describes the signals that connect off chip as well as internal supply nodes and special signals.

### 15.2.1 LIN — LIN Bus Pin

This pad is connected to the single-wire LIN data bus.

### 15.2.2 LGND — LIN Ground Pin

This pin is the device LIN ground connection. It is used to sink currents related to the LIN Bus pin. A de-coupling capacitor external to the device (typically 220 pF, X7R ceramic) between LIN and LGND can further improve the quality of this ground and filter noise.

### 15.2.3 VLINSUP — Positive Power Supply

External power supply to the chip. The VLINSUP supply mapping is described in device level documentation.

### 15.2.4 LPTxD — LIN Transmit Pin

This pin can be routed to the SCI, LPDR1 register bit, an external pin, or other options. Please refer to the PIM chapter of the device specification for the available routing options.

This input is only used in normal mode; in other modes the value of this pin is ignored.

### 15.2.5 LPRxD — LIN Receive Pin

This pin can be routed to the SCI, an external pin, or other options. Please refer to the PIM chapter of the device specification for the available routing options.

In standby mode this output is disabled, and sends only a short pulse in case the wake-up functionality is enabled and a valid wake-up pulse was received in the LIN Bus.

## 15.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the LIN Physical Layer.

### 15.3.1 Module Memory Map

A summary of the registers associated with the LIN Physical Layer module is shown in Table 15-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

**NOTE**

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

| Address Offset<br>Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0000<br>LPDR | R | 0 | 0 | 0 | 0 | 0 | 0 | LPDR1 | LPDR0 |
| | W | | | | | | | | |
| 0x0001<br>LPCR | R | 0 | 0 | 0 | 0 | LPE | RXONLY | LPWUE | LPPUE |
| | W | | | | | | | | |
| 0x0002<br>Reserved | R | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| | W | | | | | | | | |
| 0x0003<br>LPSLRM | R | LPDTDIS | 0 | 0 | 0 | 0 | 0 | LPSLR1 | LPSLR0 |
| | W | | | | | | | | |
| 0x0004<br>Reserved | R | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| | W | | | | | | | | |
| 0x0005<br>LPSR | R | LPDT | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0006<br>LPIE | R | LPDTIE | LPOCIE | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0007<br>LPIF | R | LPDTIF | LPOCIF | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |

**Figure 15-2. Register Summary**

## 15.3.2 Register Descriptions

This section describes all the LIN Physical Layer registers and their individual bits.

### 15.3.2.1 Port LP Data Register (LPDR)

Module Base + Address 0x0000 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | LPDR1 | LPDR0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

= Unimplemented

**Figure 15-3. Port LP Data Register (LPDR)**

[1] Read: Anytime
Write: Anytime

**Table 15-2. LPDR Field Description**

| Field | Description |
|---|---|
| 1<br>LPDR1 | **Port LP Data Bit 1** — The LIN Physical Layer LPTxD input (see Figure 15-1) can be directly controlled by this register bit. The routing of the LPTxD input is done in the Port Inetrgation Module (PIM). Please refer to the PIM chapter of the device Reference Manual for more info. |
| 0<br>LPDR0 | **Port LP Data Bit 0** — Read-only bit. The LIN Physical Layer LPRxD output state can be read at any time. |

### 15.3.2.2 LIN Control Register (LPCR)

Module Base + Address 0x0001 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | LPE | RXONLY | LPWUE | LPPUE |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 15-4. LIN Control Register (LPCR)**

[1] Read: Anytime
Write: Anytime,

**Table 15-3. LPCR Field Description**

| Field | Description |
|---|---|
| 3<br>LPE | **LIN Enable Bit** — If set, this bit enables the LIN Physical Layer.<br>0 The LIN Physical Layer is in shutdown mode. None of the LIN Physical Layer functions are available, except that the bus line is held in its recessive state by a high ohmic (330kΩ) resistor. All registers are normally accessible.<br>1 The LIN Physical Layer is not in shutdown mode. |
| 2<br>RXONLY | **Receive Only Mode bit** — This bit controls RXONLY mode.<br>0 The LIN Physical Layer is not in receive only mode.<br>1 The LIN Physical Layer is in receive only mode. |
| 1<br>LPWUE | **LIN Wake-Up Enable** — This bit controls the wake-up feature in standby mode.<br>0 In standby mode the wake-up feature is disabled.<br>1 In standby mode the wake-up feature is enabled. |
| 0<br>LPPUE | **LIN Pullup Resistor Enable** — Selects pullup resistor.<br>0 The pullup resistor is high ohmic (330 kΩ).<br>1 The 34 kΩ pullup is switched on (except if LPE=0 or when in standby mode with LPWUE=0). |

### 15.3.2.3 Reserved Register

Module Base + Address 0x0002 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| Reset | x | x | x | x | x | x | x | x |

= Unimplemented

**Figure 15-5. LIN Test register**

[1] Read: Anytime
Write: Only in special mode

**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 mode can alter the module's functionality.

**Table 15-4. Reserved Register Field Description**

| Field | Description |
|---|---|
| 7-0<br>Reserved | These reserved bits are used for test purposes. Writing to these bits can alter the module functionality. |

### 15.3.2.4 LIN Slew Rate Mode Register (LPSLRM)

Module Base + Address 0x0003 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | LPDTDIS | 0 | 0 | 0 | 0 | 0 | LPSLR1 | LPSLR0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 15-6. LIN Slew Rate Mode Register (LPSLRM)**

[1] Read: Anytime
Write: Only in shutdown mode (LPE=0)

**Table 15-5. LPSLRM Field Description**

| Field | Description |
|---|---|
| 7<br>LPDTDIS | **TxD-dominant timeout disable Bit** — This bit disables the TxD-dominant timeout feature. Disabling this feature is only recommended for using the LIN Physical Layer for other applications than LIN protocol. It is only writable in shutdown mode (LPE=0).<br>0 TxD-dominant timeout feature is enabled.<br>1 TxD-dominant timeout feature is disabled. |
| 1-0<br>LPSLR[1:0] | **Slew-Rate Bits** — Please see section 15.4.2 for details on how the slew rate control works. These bits are only writable in shutdown mode (LPE=0).<br>00 Normal Slew Rate (optimized for 20 kbit/s).<br>01 Slow Slew Rate (optimized for 10.4 kbit/s).<br>10 Fast Mode Slew Rate (up to 250 kbit/s). This mode is not compliant with the LIN Protocol (LIN electrical characteristics like duty cycles, reference levels, etc. are not fulfilled). It is only meant to be used for fast data transmission. Please refer to section 15.4.2.2 for more details on fast mode.Please note that an external pullup resistor stronger than 1 kΩ might be necessary for the range 100 kbit/s to 250 kbit/s.<br>11 Reserved . |

## 15.3.2.5 Reserved Register

Module Base + Address 0x0004 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| Reset | x | x | x | x | x | x | x | x |

= Unimplemented

**Figure 15-7. Reserved Register**

[1] Read: Anytime
Write: Only in special mode

**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 mode can alter the module's functionality.

**Table 15-6. Reserved Register Field Description**

| Field | Description |
|---|---|
| 7-0<br>Reserved | These reserved bits are used for test purposes. Writing to these bits can alter the module functionality. |

### 15.3.2.6 LIN Status Register (LPSR)

Module Base + Address 0x0005 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | LPDT | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 15-8. LIN Status Register (LPSR)**

[1] Read: Anytime
Write: Never, writes to this register have no effect

**Table 15-7. LPSR Field Description**

| Field | Description |
|---|---|
| 7<br>LPDT | **LIN Transmitter TxD-dominant timeout Status Bit** — This read-only bit signals that the LPTxD pin is still dominant after a TxD-dominant timeout. As long as the LPTxD is dominant after the timeout the LIN transmitter is shut down and the LPTDIF is set again after attempting to clear it.<br>0 If there was a TxD-dominant timeout, LPTxD has ceased to be dominant after the timeout.<br>1 LPTxD is still dominant after a TxD-dominant timeout. |

### 15.3.2.7 LIN Interrupt Enable Register (LPIE)

Module Base + Address 0x0006 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | LPDTIE | LPOCIE | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 15-9. LIN Interrupt Enable Register (LPIE)**

[1] Read: Anytime
Write: Anytime

**Table 15-8. LPIE Field Description**

| Field | Description |
|---|---|
| 7<br>LPDTIE | **LIN transmitter TxD-dominant timeout Interrupt Enable** —<br>0 Interrupt request is disabled.<br>1 Interrupt is requested if LPDTIF bit is set. |
| 6<br>LPOCIE | **LIN transmitter Overcurrent Interrupt Enable** —<br>0 Interrupt request is disabled.<br>1 Interrupt is requested if LPOCIF bit is set. |

### 15.3.2.8 LIN Interrupt Flags Register (LPIF)

Module Base + Address 0x0007 Access: User read/write[1]

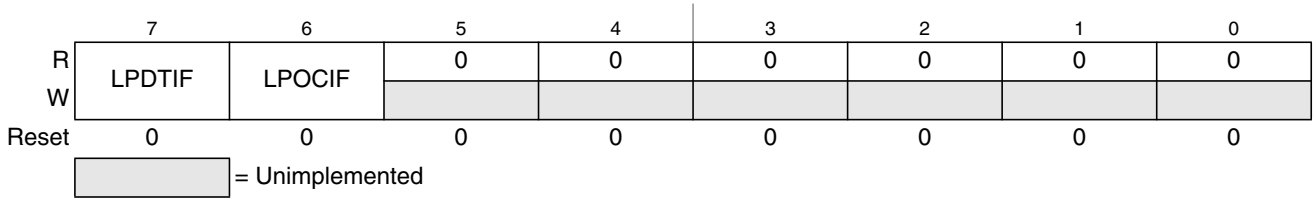



**Figure 15-10. LIN Interrupt Flags Register (LPIF)**

[1] Read: Anytime
Write: Writing ‘1’ clears the flags, writing a ‘0’ has no effect

**Table 15-9. LPIF Field Description**

| Field | Description |
|---|---|
| 7<br>LPDTIF | **LIN Transmitter TxD-dominant timeout Interrupt Flag** — LPDTIF is set to 1 when LPTxD is still dominant (0) after $t_{TDLIM}$ of the falling edge of LPTxD. For protection, the transmitter is disabled. This flag can only be cleared by writing a 1. Writing a 0 has no effect. Please make sure that LPDTIF=1 before trying to clear it. Clearing LPDTIF is not allowed if LPDTIF=0 already. If the LPTxD is still dominant after clearing the flag, the transmitter stays disabled and this flag is set again (see 15.4.4.2 TxD-dominant timeout Interrupt).<br>If interrupt requests are enabled (LPDTIE= 1), LPDTIF causes an interrupt request.<br>0 No TxD-dominant timeout has occurred.<br>1 A TxD-dominant timeout has occurred. |
| 6<br>LPOCIF | **LIN Transmitter Overcurrent Interrupt Flag** — LPOCIF is set to 1 when an overcurrent event happens. For protection, the transmitter is disabled. This flag can only be cleared by writing a 1. Writing a 0 has no effect. Please make sure that LPOCIF=1 before trying to clear it. Clearing LPOCIF is not allowed if LPOCIF=0 already. If the overcurrent is still present or LPTxD is dominant after clearing the flag, the transmitter stays disabled and this flag is set again (see15.4.4.1 Overcurrent Interrupt).<br>If interrupt requests are enabled (LPOCIE= 1), LPOCIF causes an interrupt request.<br>0 No overcurrent event has occurred.<br>1 Overcurrent event has occurred. |

## 15.4 Functional Description

### 15.4.1 General

The LIN Physical Layer module implements the physical layer of the LIN interface. This physical layer can be driven by the SCI (Serial Communication Interface) module or directly through the LPDR register.

### 15.4.2 Slew Rate and LIN Mode Selection

The slew rate can be selected for Electromagnetic Compatibility (EMC) optimized operation at 10.4 kbit/s and 20 kbit/s as well as at fast baud rate (up to 250 kbit/s) for test and programming. The slew rate can be chosen with the bits LPSLR[1:0] in the LIN Slew Rate Mode Register (LPSLRM). The default slew rate corresponds to 20 kbit/s.

The LIN Physical Layer can also be configured to be used for non-LIN applications (for example, to transmit a PWM pulse) by disabling the TxD-dominant timeout (LPDTDIS=1).

Changing the slew rate (LPSLRM Register) during transmission is not allowed in order to avoid unwanted effects. To change the register, the LIN Physical Layer must first be disabled (LPE=0). Once it is updated the LIN Physical Layer can be enabled again.

**NOTE**

For 20 kbit/s and Fast Mode communication speeds, the corresponding slew rate ***MUST*** be set; otherwise, the communication is not guaranteed (violation of the specified LIN duty cycles). For 10.4 kbit/s, the 20 kbit/s slew rate ***can*** be set but the EMC performance is worse. The up to 250 kbit/s slew rate must be chosen ***ONLY*** for fast mode, not for any of the 10.4 kbit/s or 20 kbit/s LIN compliant communication speeds.

#### 15.4.2.1 10.4 kbit/s and 20 kbit/s

When the slew rate is chosen for 10.4 kbit/s or 20 kbit/s communication, a control loop is activated within the module to make the rise and fall times of the LIN bus independent from VLINSUP and the load on the bus.

#### 15.4.2.2 Fast Mode (not LIN compliant)

Choosing this slew rate allows baud rates up to 250 kbit/s by having much steeper edges (please refer to electricals). As for the 10.4 kbit/s and 20 kbit/s modes, the slope control loop is also engaged. This mode is used for fast communication only, and the LIN electricals are not supported (for example, the LIN duty cycles).

**A stronger external pullup resistor might be necessary to sustain communication speeds up to 250 kbit/s.** The LIN signal (and therefore the receive LPRxD signal) might not be symmetrical for high baud rates with high loads on the bus.

Please note that if the bit time is smaller than the parameter $t_{OCLIM}$ (please refer to electricals), then no overcurrent is reported nor does an overcurrent shutdown occur. However, the current limitation is always engaged in case of a failure.

## 15.4.3 Modes

Figure 15-11 shows the possible mode transitions depending on control bits, stop mode, and error conditions.

### 15.4.3.1 Shutdown Mode

The LIN Physical Layer is fully disabled. No wake-up functionality is available. The internal pullup resistor is high ohmic only (330 kΩ) to maintain the LIN Bus pin in the recessive state. LPTxD is not monitored in this mode for a TxD-dominant timeout. All the registers are accessible.

Setting LPE causes the module to leave the shutdown mode and to enter the normal mode or receive only mode (if RXONLY bit is set).

Clearing LPE causes the module to leave the normal or receive only modes and go back to shutdown mode.

### 15.4.3.2 Normal Mode

The full functionality is available. Both receiver and transmitter are enabled. The internal pullup resistor can be chosen to be high ohmic (330 kΩ) if LPPUE = 0, or LIN compliant (34 kΩ) if LPPUE = 1.

If RXONLY is set, the module leaves normal mode to enter receive only mode.

If the MCU enters stop mode, the LIN Physical Layer enters standby mode.

### 15.4.3.3 Receive Only Mode

Entering this mode disables the transmitter and immediately stops any on-going transmission. LPTxD is not monitored in this mode for a TxD-dominant timeout.

The receiver is running in full performance mode in all cases.

To return to normal mode, the RXONLY bit must be cleared.

If the device enters stop mode, the module leaves receive only mode to enter standby mode.

### 15.4.3.4 Standby Mode with Wake-Up Feature

The transmitter of the LIN Physical Layer is disabled and the receiver enters a low power mode.

**NOTE**

Before entering standby mode, ensure no transmissions are ongoing.

If LPWUE is not set, no wake up feature is available and the standby mode has the same electrical properties as the shutdown mode. This allows a low-power consumption of the device in stop mode if the wake-up feature is not needed.

If LPWUE is set the receiver is able to pass wake-up events to the SCI (Serial Communication Interface). If the LIN Physical Layer receives a dominant level longer than $t_{WUFR}$ followed by a rising edge, it sends a pulse to the SCI which can generate a wake-up interrupt.

Once the device exits stop mode, the LIN Physical Layer returns to normal or receive only mode depending on the status of the RXONLY bit.

**NOTE**

Since the wake-up interrupt is requested by the SCI, the wake-up feature is not available if the SCI is not used.

The internal pullup resistor is selectable only if LPWUE = 1 (wake-up enabled). If LPWUE = 0, the internal pullup resistor is not selectable and remains at 330 kΩ regardless of the state of the LPPUE bit.

If LPWUE = 1, selecting the 330 kΩ pullup resistor (LPPUE = 0) reduces the current consumption in standby mode.

**NOTE**

When using the LIN wake-up feature in combination with other non-LIN device wake-up features (like a periodic time interrupt), some care must be taken.

If the device leaves stop mode while the LIN bus is dominant, the LIN Physical Layer returns to normal or receive only mode and the LIN bus is re-routed to the RXD pin of the SCI and triggers the edge detection interrupt (if the interrupt's priority of the hardware that awakes the MCU is less than the priority of the SCI interrupt, then the SCI interrupt will execute first). It is up to the software to decide what to do in this case because the LIN Physical Layer can not guarantee it was a valid wake-up pulse.

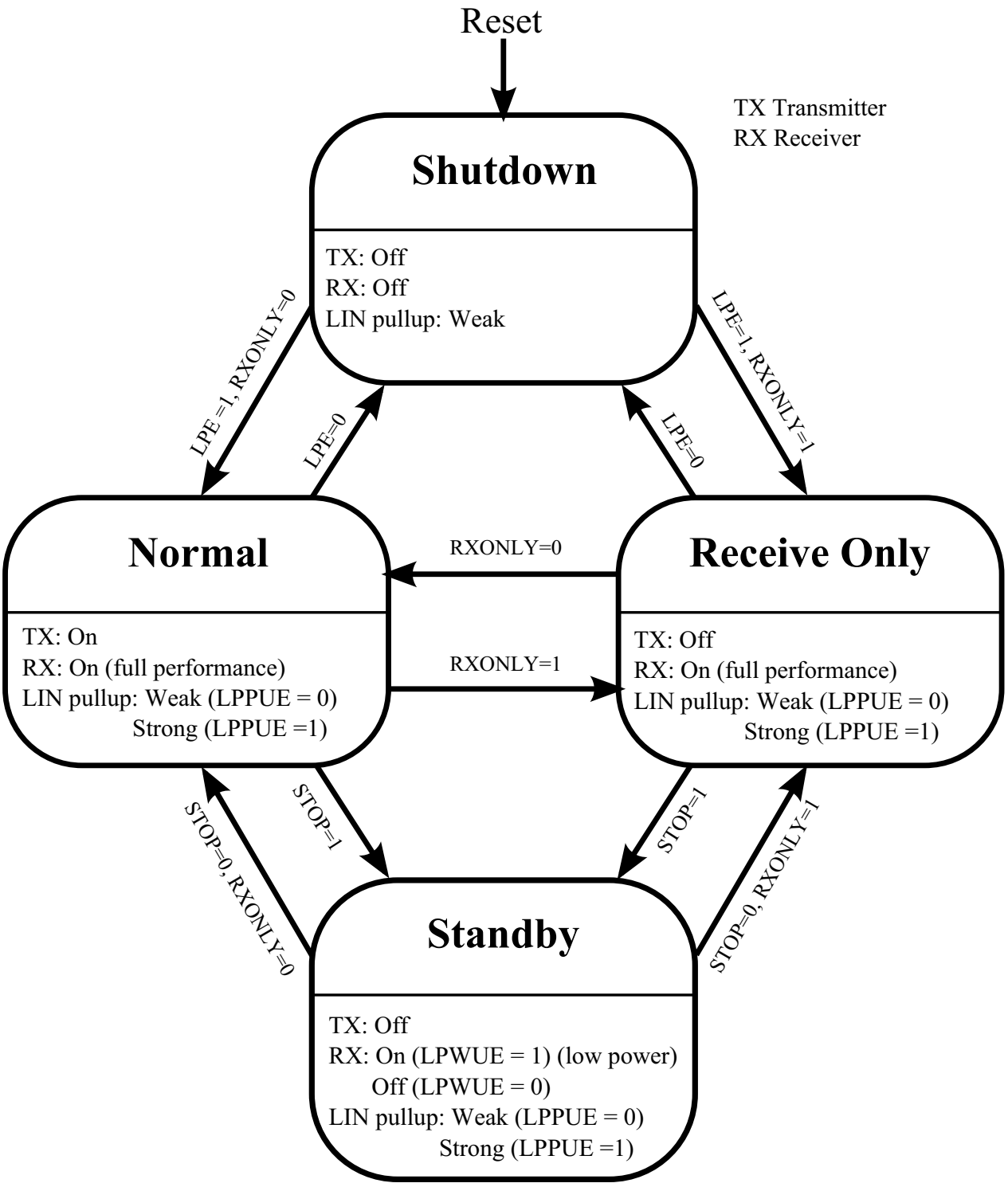



**Figure 15-11. LIN Physical Layer Mode Transitions**

### 15.4.4 Interrupts

The interrupt vector requested by the LIN Physical Layer is listed in Table 15-10. Vector address and interrupt priority is defined at the MCU level.

The module internal interrupt sources are combined into a single interrupt request at the device level.

**Table 15-10. Interrupt Vectors**

| Module Interrupt Source | Module Internal Interrupt Source | Local Enable |
|---|---|---|
| LIN Interrupt (LPI) | LIN Txd-Dominant Timeout Interrupt (LPDTIF) | LPDTIE = 1 |
| | LIN Overcurrent Interrupt (LPOCIF) | LPOCIE = 1 |

#### 15.4.4.1 Overcurrent Interrupt

The transmitter is protected against overcurrent. In case of an overcurrent condition occurring within a time frame called $t_{OCLIM}$ starting from LPTxD falling edge, the current through the transmitter is limited (the transmitter is not shut down). The masking of an overcurrent event within the time frame $t_{OCLIM}$ is meant to avoid “false” overcurrent conditions that can happen during the discharging of the LIN bus. If an overcurrent event occurs out of this time frame, the transmitter is disabled and the LPOCIF flag is set.

In order to re-enable the transmitter again, the following prerequisites must be met:

1) Overcurrent condition is over

2) LPTxD is recessive or the LIN Physical Layer is in shutdown or receive only mode for a minimum of a transmit bit time.

To re-enable the transmitter then, the LPOCIF flag must be cleared (by writing a 1).

**NOTE**

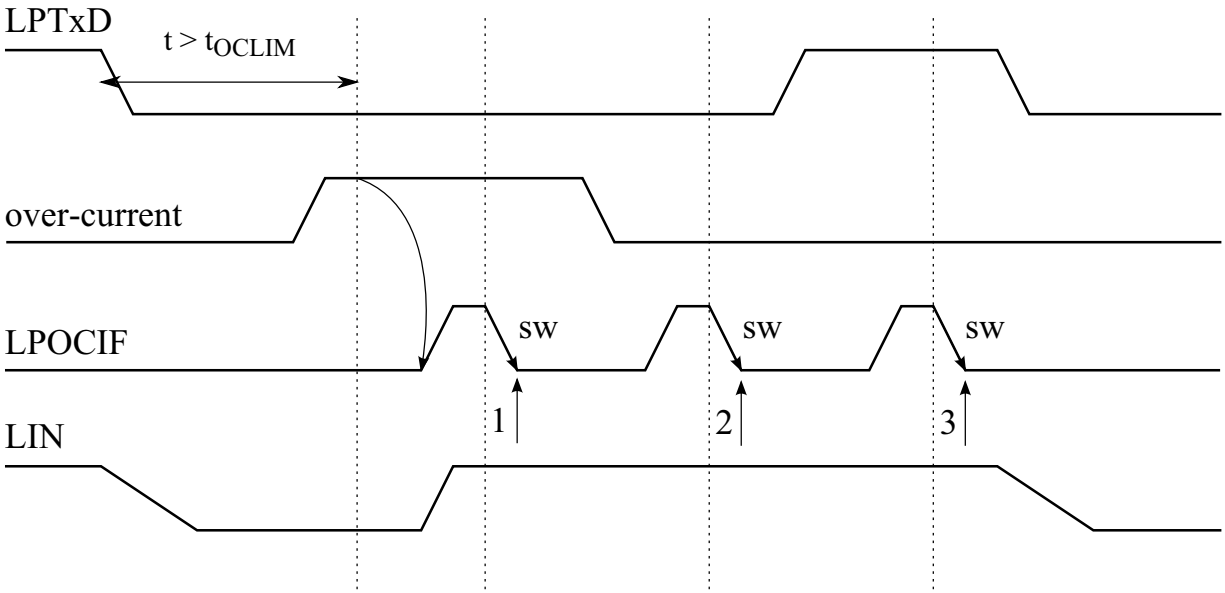
Please make sure that LPOCIF=1 before trying to clear it. It is not allowed to try to clear LPOCIF if LPOCIF=0 already.

After clearing LPOCIF, if the overcurrent condition is still present or the LPTxD pin is dominant while being in normal mode, the transmitter remains disabled and the LPOCIF flag is set again after a time to indicate that the attempt to re-enable has failed. This time is equal to:

- minimum 1 IRC period (1 us) + 2 bus periods
- maximum 2 IRC periods (2 us) + 3 bus periods

If the bit LPOCIE is set in the LPIE register, an interrupt is requested.

Figure 15-12 shows the different scenarios for overcurrent interrupt handling.




1: Flag cleared, transmitter re-enable not successful because over-current is still present
2: Flag cleared, transmitter re-enable not successful because LPTxD is dominant
3: Flag cleared, transmitter re-enable successful

**Figure 15-12. Overcurrent interrupt handling**

### 15.4.4.2 TxD-dominant timeout Interrupt

To protect the LIN bus from a network lock-up, the LIN Physical Layer implements a TxD-dominant timeout mechanism. When the LPTxD signal has been dominant for more than $t_{DTLIM}$ the transmitter is disabled and the LPDT status flag and the LPDTIF interrupt flag are set.

In order to re-enable the transmitter again, the following prerequisites must be met:

1) TxD-dominant condition is over (LPDT=0)

2) LPTxD is recessive or the LIN Physical Layer is in shutdown or receive only mode for a minimum of a transmit bit time

To re-enable the transmitter then, the LPDTIF flag must be cleared (by writing a 1).

**NOTE**

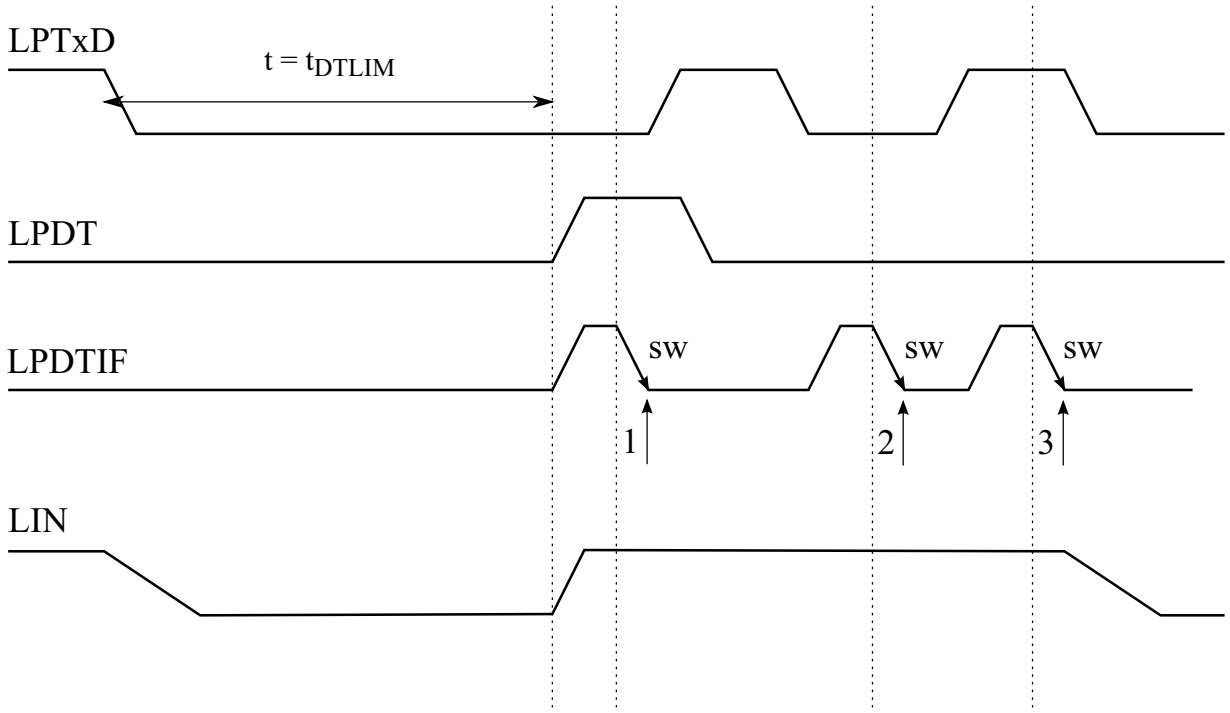
Please make sure that LPDTIF=1 before trying to clear it. It is not allowed to try to clear LPDTIF if LPDTIF=0 already.

After clearing LPDTIF, if the TxD-dominant timeout condition is still present or the LPTxD pin is dominant while being in normal mode, the transmitter remains disabled and the LPDTIF flag is set after a time again to indicate that the attempt to re-enable has failed. This time is equal to:

- minimum 1 IRC period (1 us) + 2 bus periods
- maximum 2 IRC periods (2 us) + 3 bus periods

If the bit LPDTIE is set in the LPIE register, an interrupt is requested.

Figure 15-13 shows the different scenarios of TxD-dominant timeout interrupt handling.




1: Flag cleared, transmitter re-enable not successful because TxD-dominant timeout condition is still present
2: Flag cleared, transmitter re-enable not successful because LPTxD is dominant
3: Flag cleared, transmitter re-enable successful

**Figure 15-13. TxD-dominant timeout interrupt handling**

## 15.5 Application Information

### 15.5.1 Module Initialization

The following steps should be used to configure the module before starting the transmission:

1. Set the slew rate in the LPSLRM register to the desired transmission baud rate.
2. When using the LIN Physical Layer for other purposes than LIN transmission, de-activate the dominant timeout feature in the LPSLRM register if needed.
3. In most cases, the internal pullup should be enabled in the LPCR register.
4. Route the desired source in the PIM module to the LIN Physical Layer.
5. Select the transmit mode (Receive only mode or Normal mode) in the LPCR register.
6. If the SCI is selected as source, activate the wake-up feature in the LPCR register if needed for the application (SCI active edge interrupt must also be enabled).
7. Enable the LIN Physical Layer in the LPCR register.
8. Wait for a minimum of a transmit bit.
9. Begin transmission if needed.

**NOTE**

It is not allowed to try to clear LPOCIF or LPDTIF if they are already cleared. Before trying to clear an error flag, always make sure that it is already set.

### 15.5.2 Interrupt handling in Interrupt Service Routine (ISR)

Both interrupts (TxD-dominant timeout and overcurrent) represent a failure in transmission. To avoid more disturbances on the transmission line, the transmitter is de-activated in both cases. The interrupt subroutine must take care of clearing the error condition and starting the routine that re-enables the transmission. For that purpose, the following steps are recommended:

1. First, the cause of the interrupt must be cleared:
   — The overcurrent will be gone after the transmitter has been disabled.
   — The TxD-dominant timeout condition will be gone once the selected source for LPTxD has turned recessive.
2. Clear the corresponding enable bit (LPDTIE or LPOCIE) to avoid entering the ISR again until the flags are cleared.
3. Notify the application of the error condition (LIN Error handler) and leave the ISR.

In the LIN Error handler, the following sequence is recommended:

1. Disable the LIN Physical Layer (LPCR) while re-configuring the transmission.
   — If the receiver must remain enabled, set the LIN Physical Layer into receive only mode instead.
2. Do all required configurations (SCI, etc.) to re-enable the transmission.
3. Wait for a transmit bit (this is needed to successfully re-enable the transmitter).

4. Clear the error flag.
5. Enable the interrupts again (LPDTIE and LPOCIE).
6. Enable the LIN Physical Layer or leave the receive only mode (LPCR register).
7. Wait for a minimum of a transmit bit before beginning transmission again.

If there is a problem re-enabling the transmitter, then the error flag will be set again during step 3 and the ISR will be called again.

# Chapter 16
# Supply Voltage Sensor - (BATSV2)

**Table 16-1. Revision History Table**

| Rev. No. (Item No.) | Data | Sections Affected | Substantial Change(s) |
|---|---|---|---|
| V01.00 | 15 Dec 2010 | all | Initial Version |
| V02.00 | 16 Mar 2011 | 16.3.2.1<br>16.4.2.1 | - added BVLS[1] to support four voltage level<br>- moved BVHS to register bit 6 |

## 16.1 Introduction

The BATS module provides the functionality to measure the voltage of the battery supply pin VSENSE or of the chip supply pin VSUP.

### 16.1.1 Features

Either One of the voltage present on the VSENSE or VSUP pin can be routed via an internal divider to the internal Analog to Digital Converter. Independent of the routing to the Analog to Digital Converter, it is possible to route one of these voltages to a comparator to generate a low or a high voltage interrupt to alert the MCU.

### 16.1.2 Modes of Operation

The BATS module behaves as follows in the system power modes:

1. Run mode

   The activation of the VSENSE Level Sense Enable (BSESE=1) or ADC connection Enable (BSEAE=1) closes the path from the VSENSE pin through the resistor chain to ground and enables the associated features if selected.

   The activation of the VSUP Level Sense Enable (BSUSE=1) or ADC connection Enable (BSUAE=1) closes the path from VSUP pin through the resistor chain to ground and enables the associated features if selected.

   BSESE takes precedence over BSUSE. BSEAE takes precedence over BSUAE.

2. Stop mode

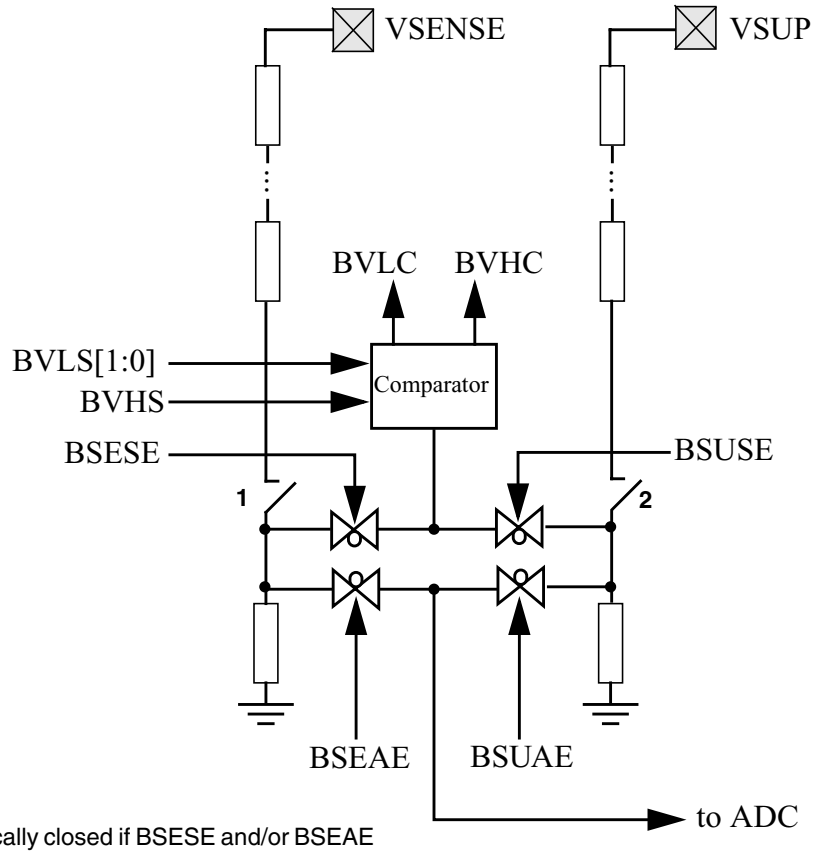
   During stop mode operation the path from the VSENSE pin through the resistor chain to ground is opened and the low voltage sense features are disabled.

During stop mode operation the path from the VSUP pin through the resistor chain to ground is opened and the low voltage sense features are disabled.
The content of the configuration register is unchanged.

### 16.1.3 Block Diagram

Figure 16-1 shows a block diagram of the BATS module. See device guide for connectivity to ADC channel.

**Figure 16-1. BATS Block Diagram**




1 automatically closed if BSESE and/or BSEAE is active, open during Stop mode
2 automatically closed if BSUSE and/or BSUAE is active, open during Stop mode

## 16.2 External Signal Description

This section lists the name and description of all external ports.

### 16.2.1 VSENSE — Supply (Battery) Voltage Sense Pin

This pin can be connected to the supply (Battery) line for voltage measurements. The voltage present at this input is scaled down by an internal voltage divider, and can be routed to the internal ADC or to a

comparator via an analog multiplexer. The pin itself is protected against reverse battery connections. To protect the pin from external fast transients an external resistor ($R_{VSENSE_R}$) is needed for protection.

### 16.2.2 VSUP — Voltage Supply Pin

This pin is the chip supply. It can be internally connected for voltage measurement. The voltage present at this input is scaled down by an internal voltage divider, and can be routed to the internal ADC or to a comparator via an analog multiplexer.

## 16.3 Memory Map and Register Definition

This section provides the detailed information of all registers for the BATS module.

### 16.3.1 Register Summary

Figure 16-2 shows the summary of all implemented registers inside the BATS module.

**NOTE**

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

| Address Offset<br>Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x0000<br>BATE | R | 0 | BVHS | BVLS[1:0] | | BSUAE | BSUSE | BSEAE | BSESE |
| | W | | | | | | | | |
| 0x0001<br>BATSR | R | 0 | 0 | 0 | 0 | 0 | 0 | BVHC | BVLC |
| | W | | | | | | | | |
| 0x0002<br>BATIE | R | 0 | 0 | 0 | 0 | 0 | 0 | BVHIE | BVLIE |
| | W | | | | | | | | |
| 0x0003<br>BATIF | R | 0 | 0 | 0 | 0 | 0 | 0 | BVHIF | BVLIF |
| | W | | | | | | | | |
| 0x0004 - 0x0005<br>Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| 0x0006 - 0x0007<br>Reserved | R | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| | W | | | | | | | | |

= Unimplemented

**Figure 16-2. BATS Register Summary**

## 16.3.2 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. Unused bits read back zero.

### 16.3.2.1 BATS Module Enable Register (BATE)

Module Base + 0x0000 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | BVHS | BVLS[1:0] | | BSUAE | BSUSE | BSEAE | BSESE |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

= Unimplemented

**Figure 16-3. BATS Module Enable Register (BATE)**

[1] Read: Anytime
Write: Anytime

**Table 16-2. BATE Field Description**

| Field | Description |
|---|---|
| 6<br>BVHS | **BATS Voltage High Select** — This bit selects the trigger level for the Voltage Level High Condition (BVHC).<br><br>0 Voltage level $V_{HBI1}$ is selected<br>1 Voltage level $V_{HBI2}$ is selected |
| 5:4<br>BVLS[1:0] | **BATS Voltage Low Select** — This bit selects the trigger level for the Voltage Level Low Condition (BVLC).<br><br>00 Voltage level $V_{LBI1}$ is selected<br>01 Voltage level $V_{LBI2}$ is selected<br>10 Voltage level $V_{LBI3}$ is selected<br>11 Voltage level $V_{LBI4}$ is selected |
| 3<br>BSUAE | **BATS VSUP ADC Connection Enable** — This bit connects the VSUP pin through the resistor chain to ground and connects the ADC channel to the divided down voltage. This bit can be set only if the BSEAE bit is cleared.<br><br>0 ADC Channel is disconnected<br>1 ADC Channel is connected |
| 2<br>BSUSE | **BATS VSUP Level Sense Enable** — This bit connects the VSUP pin through the resistor chain to ground and enables the Voltage Level Sense features measuring BVLC and BVHC. This bit can be set only if the BSESE bit is cleared.<br><br>0 Level Sense features disabled<br>1 Level Sense features enabled |
| 1<br>BSEAE | **BATS VSENSE ADC Connection Enable** — This bit connects the VSENSE pin through the resistor chain to ground and connects the ADC channel to divided down voltage. Setting this bit will clear bit BSUAE .<br><br>0 ADC Channel is disconnected<br>1 ADC Channel is connected |
| 0<br>BSESE | **BATS VSENSE Level Sense Enable** — This bit connects the VSENSE pin through the resistor chain to ground and enables the Voltage Level Sense features measuring BVLC and BVHC.Setting this bit will clear bit BSUSE<br><br>0 Level Sense features disabled<br>1 Level Sense features enabled |

## NOTE

When opening the resistors path to ground by changing BSESE, BSEAE or BSUSE, BSUAE then for a time $T_{EN_UNC}$ + two bus cycles the measured value is invalid. This is to let internal nodes be charged to correct value. BVHIE, BVLIE might be cleared for this time period to avoid false interrupts.

### 16.3.2.2 BATS Module Status Register (BATSR)

Module Base + 0x0001 Access: User read only[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | BVHC | BVLC |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

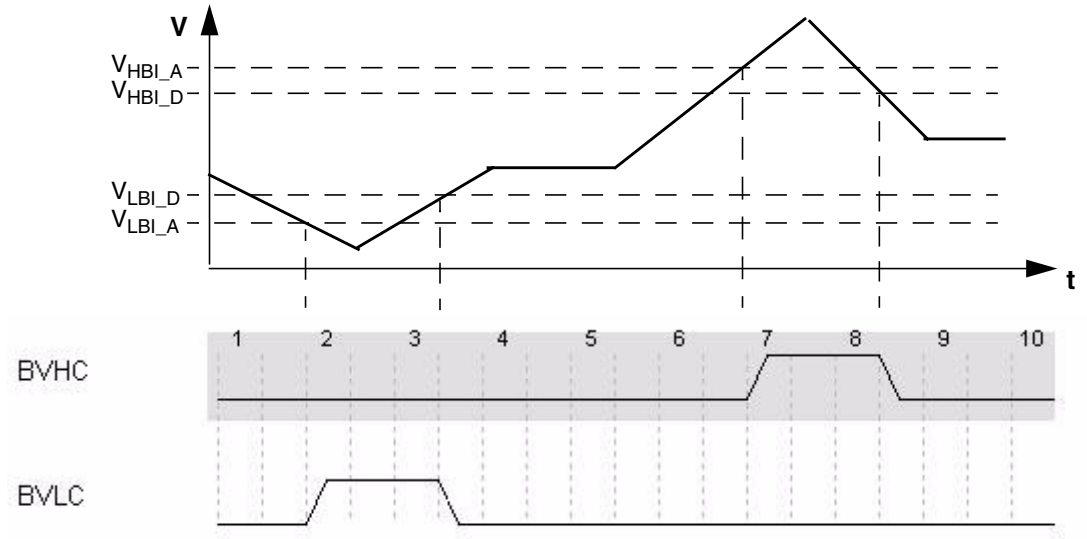
**Figure 16-4. BATS Module Status Register (BATSR)**

[1] Read: Anytime
Write: Never

**Table 16-3. BATSR - Register Field Descriptions**

| Field | Description |
|---|---|
| 1<br>BVHC | **BATS Voltage Sense High Condition Bit** — This status bit indicates that a high voltage at VSENSE or VSUP, depending on selection, is present.<br><br>0 $V_{measured} < V_{HBI_A}$ (rising edge) or $V_{measured} < V_{HBI_D}$ (falling edge)<br>1 $V_{measured} \geq V_{HBI_A}$ (rising edge) or $V_{measured} \geq V_{HBI_D}$ (falling edge) |
| 0<br>BVLC | **BATS Voltage Sense Low Condition Bit** — This status bit indicates that a low voltage at VSENSE or VSUP, depending on selection, is present.<br><br>0 $V_{measured} \geq V_{LBI_A}$ (falling edge) or $V_{measured} \geq V_{LBI_D}$ (rising edge)<br>1 $V_{measured} < V_{LBI_A}$ (falling edge) or $V_{measured} < V_{LBI_D}$ (rising edge) |

**Figure 16-5. BATS Voltage Sensing**



### 16.3.2.3 BATS Interrupt Enable Register (BATIE)

Module Base + 0x0002 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | BVHIE | BVLIE |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 16-6. BATS Interrupt Enable Register (BATIE)**

[1] Read: Anytime
Write: Anytime

**Table 16-4. BATIE Register Field Descriptions**

| Field | Description |
|---|---|
| 1<br>BVHIE | **BATS Interrupt Enable High** — Enables High Voltage Interrupt **.**<br><br>0 No interrupt will be requested whenever BVHIF flag is set .<br>1 Interrupt will be requested whenever BVHIF flag is set |
| 0<br>BVLIE | **BATS Interrupt Enable Low** — Enables Low Voltage Interrupt .<br><br>0 No interrupt will be requested whenever BVLIF flag is set .<br>1 Interrupt will be requested whenever BVLIF flag is set . |

### 16.3.2.4 BATS Interrupt Flag Register (BATIF)

Module Base + 0x0003 Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | BVHIF | BVLIF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 16-7. BATS Interrupt Flag Register (BATIF)**

[1] Read: Anytime
Write: Anytime, write 1 to clear

**Table 16-5. BATIF Register Field Descriptions**

| Field | Description |
|---|---|
| 1<br>BVHIF | **BATS Interrupt Flag High Detect** — The flag is set to 1 when BVHC status bit changes.<br><br>0 No change of the BVHC status bit since the last clearing of the flag.<br>1 BVHC status bit has changed since the last clearing of the flag. |
| 0<br>BVLIF | **BATS Interrupt Flag Low Detect** — The flag is set to 1 when BVLC status bit changes.<br><br>0 No change of the BVLC status bit since the last clearing of the flag.<br>1 BVLC status bit has changed since the last clearing of the flag. |

### 16.3.2.5 Reserved Register

Module Base + 0x0006
Module Base + 0x0007

Access: User read/write[1]

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |
| Reset | x | x | x | x | x | x | x | x |

**Figure 16-8. Reserved Register**

[1] Read: Anytime
Write: Only in special mode

**NOTE**

These reserved registers are designed for factory test purposes only and are not intended for general user access. Writing to these registers when in special mode can alter the module's functionality.

## 16.4 Functional Description

### 16.4.1 General

The BATS module allows measuring voltages on the VSENSE and VSUP pins. The VSENSE pin is implemented to allow measurement of the supply Line (Battery) Voltage $V_{BAT}$ directly. By bypassing the device supply capacitor and the external reversed battery protection diode this pin allows to detect under/over voltage conditions without delay. A series resistor ($R_{VSENSE_R}$) is required to protect the VSENSE pin from fast transients.

The voltage at the VSENSE or VSUP pin can be routed via an internal voltage divider to an internal Analog to Digital Converter Channel. Also the BATS module can be configured to generate a low and high voltage interrupt based on VSENSE or VSUP. The trigger level of the high and low interrupt are selectable.

In a typical application, the module could be used as follows: The voltage at VSENSE is observed via usage of the interrupt feature (BSESE=1, BVHIE=1), while the VSUP pin voltage is routed to the ADC to allow regular measurement (BSUAE=1).

## 16.4.2 Interrupts

This section describes the interrupt generated by the BATS module. The interrupt is only available in CPU run mode. Entering and exiting CPU stop mode has no effect on the interrupt flags.

To make sure the interrupt generation works properly the bus clock frequency must be higher than the Voltage Warning Low Pass Filter frequency ($f_{VWLP_filter}$).

The comparator outputs BVLC and BVHC are forced to zero if the comparator is disabled (configuration bits BSESE and BSUSE are cleared). If the software disables the comparator during a high or low Voltage condition (BVHC or BVLC active), then an additional interrupt is generated. To avoid this behavior the software must disable the interrupt generation before disabling the comparator.

The BATS interrupt vector is named in Table 16-6. Vector addresses and interrupt priorities are defined at MCU level.

The module internal interrupt sources are combined into one module interrupt signal.

**Table 16-6. BATS Interrupt Sources**

| Module Interrupt Source | Module Internal Interrupt Source | Local Enable |
|---|---|---|
| BATS Interrupt (BATI) | BATS Voltage Low Condition Interrupt (BVLI) | BVLIE = 1 |
| | BATS Voltage High Condition Interrupt (BVHI) | BVHIE = 1 |

### 16.4.2.1 BATS Voltage Low Condition Interrupt (BVLI)

To use the Voltage Low Interrupt the Level Sensing must be enabled (BSESE =1 or BSUSE =1).

If measured when

a) $V_{LBI1}$ selected with BVLS[1:0] = 0x0
at selected pin $V_{measure} < V_{LBI1_A}$ (falling edge) or $V_{measure} < V_{LBI1_D}$ (rising edge)

or when

b) $V_{LBI2}$ selected with BVLS[1:0] = 0x1
at selected pin $V_{measure} < V_{LBI2_A}$ (falling edge) or $V_{measure} < V_{LBI2_D}$ (rising edge)

or when

c) $V_{LBI3}$ selected with BVLS[1:0] = 0x2
at selected pin $V_{measure} < V_{LBI3_A}$ (falling edge) or $V_{measure} < V_{LBI3_D}$ (rising edge)

or when

d) $V_{LBI4}$ selected with BVLS[1:0] = 0x3
at selected pin $V_{measure} < V_{LBI4_A}$ (falling edge) or $V_{measure} < V_{LBI4_D}$ (rising edge)

then BVLC is set. BVLC status bit indicates that a low voltage at the selected pin is present. The Low Voltage Interrupt flag (BVLIF) is set to 1 when the Voltage Low Condition (BVLC) changes state . The

Interrupt flag BVLIF can only be cleared by writing a 1. If the interrupt is enabled by bit BVLIE the module requests an interrupt to MCU (BATI).

#### 16.4.2.2 BATS Voltage High Condition Interrupt (BVHI)

To use the Voltage High Interrupt the Level Sensing must be enabled (BSESE =1 or BSUSE).

If measured when

a) $V_{HBI1}$ selected with BVHS = 0
at selected pin $V_{measure} \geq V_{HBI1_A}$ (rising edge) or $V_{measure} \geq V_{HBI1_D}$ (falling edge)

or when

a) $V_{HBI2}$ selected with BVHS = 1
at selected pin $V_{measure} \geq V_{HBI2_A}$ (rising edge) or $V_{measure} \geq V_{HBI2_D}$ (falling edge)

then BVHC is set. BVHC status bit indicates that a high voltage at the selected pin is present. The High Voltage Interrupt flag (BVHIF) is set to 1 when a Voltage High Condition (BVHC) changes state. The Interrupt flag BVHIF can only be cleared by writing a 1. If the interrupt is enabled by bit BVHIE the module requests an interrupt to MCU (BATI).

# Chapter 17
# 64 KByte Flash Module (S12FTMRG64K512V1)

**Table 17-1. Revision History**

| Revision Number | Revision Date | Sections Affected | Description of Changes |
|---|---|---|---|
| V01.00 | 17 Jun 2010 | 17.4.6.1/17-34<br>17.4.6.2/17-35<br>17.4.6.3/17-35<br>17.4.6.14/17-45 | Clarify Erase Verify Commands Descriptions related to the bits MGSTAT[1:0] of the register FSTAT. |
| V01.01 | 31 aug 2010 | 17.4.6.2/17-35<br>17.4.6.12/17-42<br>17.4.6.13/17-44 | Updated description of the commands RD1BLK, MLOADU and MLOADF |
| V01.02 | 31 Jan 2011 | 17.3.2.9/17-18 | Updated description of protection on Section 17.3.2.9 P-Flash Protection Register (FPROT) |
| V01.03 | 04 Oct 2013 | 17.3.2.9/17-18<br>17.3.2.10/17-22 | Updated notes regarding restrictions to change Protection in Special Single Chip Mode (SS) |

## 17.1 Introduction

The FTMRG64K512 module implements the following:

- 64Kbytes of P-Flash (Program Flash) memory
- 64bytes of EEPROM memory

The Flash memory is ideal for single-supply applications allowing for field reprogramming without requiring external high voltage sources for program or erase operations. The Flash module includes a memory controller that executes commands to modify Flash memory contents. The user interface to the memory controller consists of the indexed Flash Common Command Object (FCCOB) register which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register can be written to with a new command.

**CAUTION**

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

The Flash memory may be read as bytes and aligned words. Read access time is one bus cycle for bytes and aligned words. For misaligned words access, the CPU has to perform twice the byte read access command. For Flash memory, an erased bit reads 1 and a programmed bit reads 0.

It is possible to read from P-Flash memory while some commands are executing on EEPROM memory. It is not possible to read from EEPROM memory while a command is executing on P-Flash memory. Simultaneous P-Flash and EEPROM operations are discussed in Section 17.4.5 Allowed Simultaneous P-Flash and EEPROM Operations.

Both P-Flash and EEPROM memories are implemented with Error Correction Codes (ECC) that can resolve single bit faults and detect double bit faults. For P-Flash memory, the ECC implementation requires that programming be done on an aligned 8 byte basis (a Flash phrase). Since P-Flash memory is always read by half-phrase, only one single bit fault in an aligned 4 byte half-phrase containing the byte or word accessed will be corrected.

### 17.1.1 Glossary

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

**EEPROM Memory** — The EEPROM memory constitutes the nonvolatile memory store for data.

**EEPROM Sector** — The EEPROM sector is the smallest portion of the EEPROM memory that can be erased. The EEPROM sector consists of 4 bytes.

**NVM Command Mode** — An NVM mode using the CPU to setup the FCCOB register to pass parameters required for Flash command execution.

**Phrase** — An aligned group of four 16-bit words within the P-Flash memory. Each phrase includes two sets of aligned double words with each set including 7 ECC bits for single bit fault correction and double bit fault detection within each double word.

**P-Flash Memory** — The P-Flash memory constitutes the main nonvolatile memory store for applications.

**P-Flash Sector** — The P-Flash sector is the smallest portion of the P-Flash memory that can be erased. Each P-Flash sector contains 512 bytes.

**Program IFR** — Nonvolatile information register located in the P-Flash block that contains the Version ID, and the Program Once field.

## 17.1.2 Features

### 17.1.2.1 P-Flash Features

- 64 Kbytes of P-Flash memory composed of one 64 Kbyte Flash block divided into 128 sectors of 512 bytes
- Single bit fault correction and double bit fault detection within a 32-bit double word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and phrase program operation
- Ability to read the P-Flash memory while programming a word in the EEPROM memory
- Flexible protection scheme to prevent accidental program or erase of P-Flash memory

### 17.1.2.2 EEPROM Features

- 512 bytes of EEPROM memory composed of one 512 byte Flash block divided into 128 sectors of 4 bytes
- Single bit fault correction and double bit fault detection within a word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and word program operation
- Protection scheme to prevent accidental program or erase of EEPROM memory
- Ability to program up to four words in a burst sequence

### 17.1.2.3 Other Flash Module Features

- No external high-voltage power supply required for Flash memory program and erase operations
- Interrupt generation on Flash command completion and Flash error detection
- Security mechanism to prevent unauthorized access to the Flash memory

### 17.1.3 Block Diagram

The block diagram of the Flash module is shown in Figure 17-1.

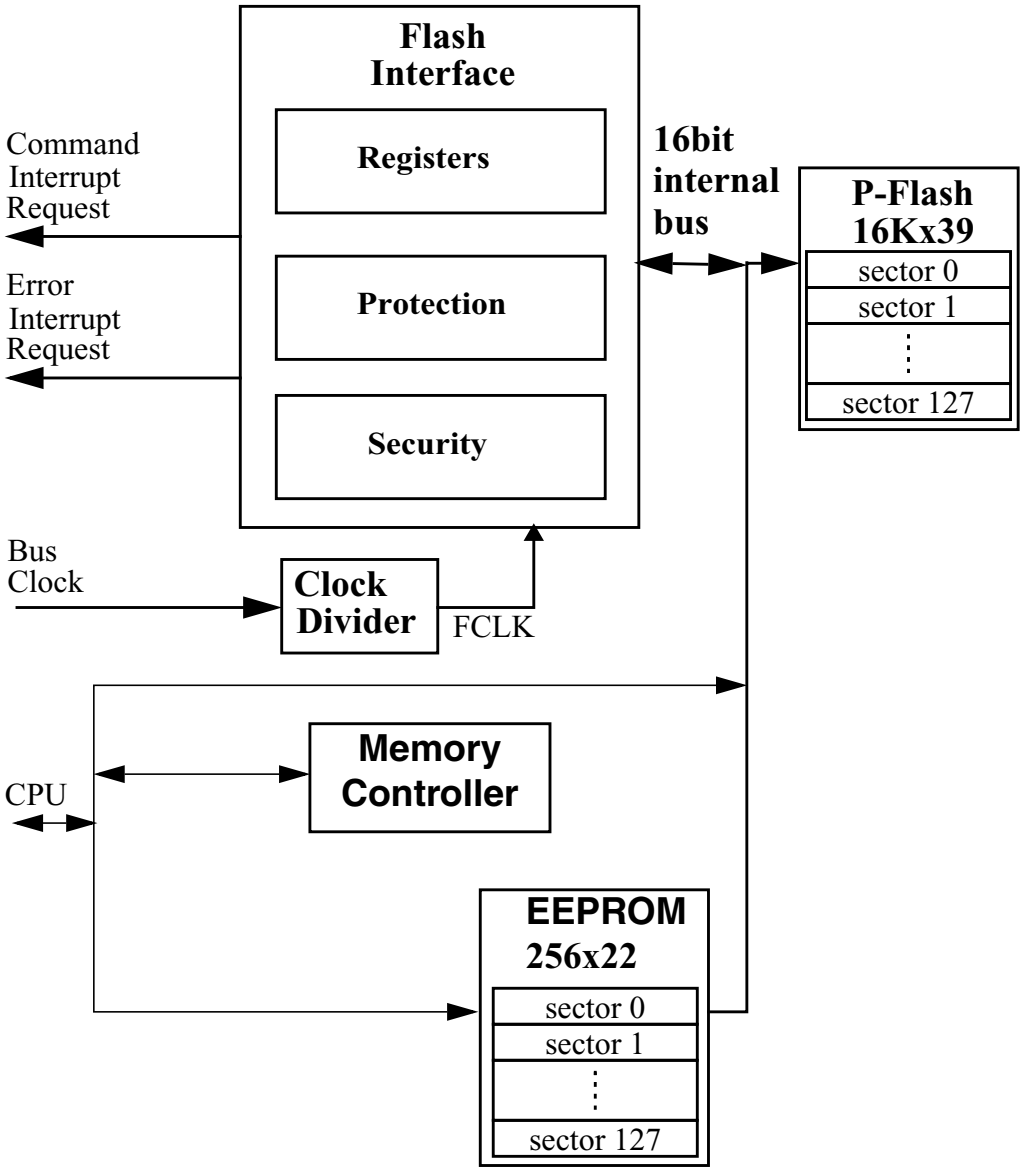



**Figure 17-1. FTMRG64K512 Block Diagram**

## 17.2 External Signal Description

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

## 17.3 Memory Map and Registers

This section describes the memory map and registers for the Flash module. Read data from unimplemented memory space in the Flash module is undefined. Write access to unimplemented or reserved memory space in the Flash module will be ignored by the Flash module.

**CAUTION**

Writing to the Flash registers while a Flash command is executing (that is indicated when the value of flag CCIF reads as '0') is not allowed. If such action is attempted the write operation will not change the register value.

Writing to the Flash registers is allowed when the Flash is not busy executing commands (CCIF = 1) and during initialization right after reset, despite the value of flag CCIF in that case (refer to Section 17.6 Initialization for a complete description of the reset sequence).

.

**Table 17-2. FTMRG Memory Map**

| Global Address (in Bytes) | Size (Bytes) | Description |
|---|---|---|
| 0x0_0000 - 0x0_03FF | 1,024 | Register Space |
| 0x0_0400 – 0x0_05FF | 512 | EEPROM Memory |
| 0x0_4000 – 0x0_7FFF | 16,284 | NVMRES[1]=1 : NVM Resource area (see Figure 17-2) |

[1] See NVMRES description in Section 17.4.3 Internal NVM resource (NVMRES)

### 17.3.1 Module Memory Map

The S12 architecture places the P-Flash memory between global addresses 0x3_0000 and 0x3_FFFF as shown in Table 17-3.The P-Flash memory map is shown in Figure .

The FPROT register, described in Section 17.3.2.9 P-Flash Protection Register (FPROT), can be set to

**Table 17-3. P-Flash Memory Addressing**

| Global Address | Size (Bytes) | Description |
|---|---|---|
| 0x3_0000 – 0x3_FFFF | 64 K | P-Flash Block<br>Contains Flash Configuration Field<br>(see Table 17-4) |

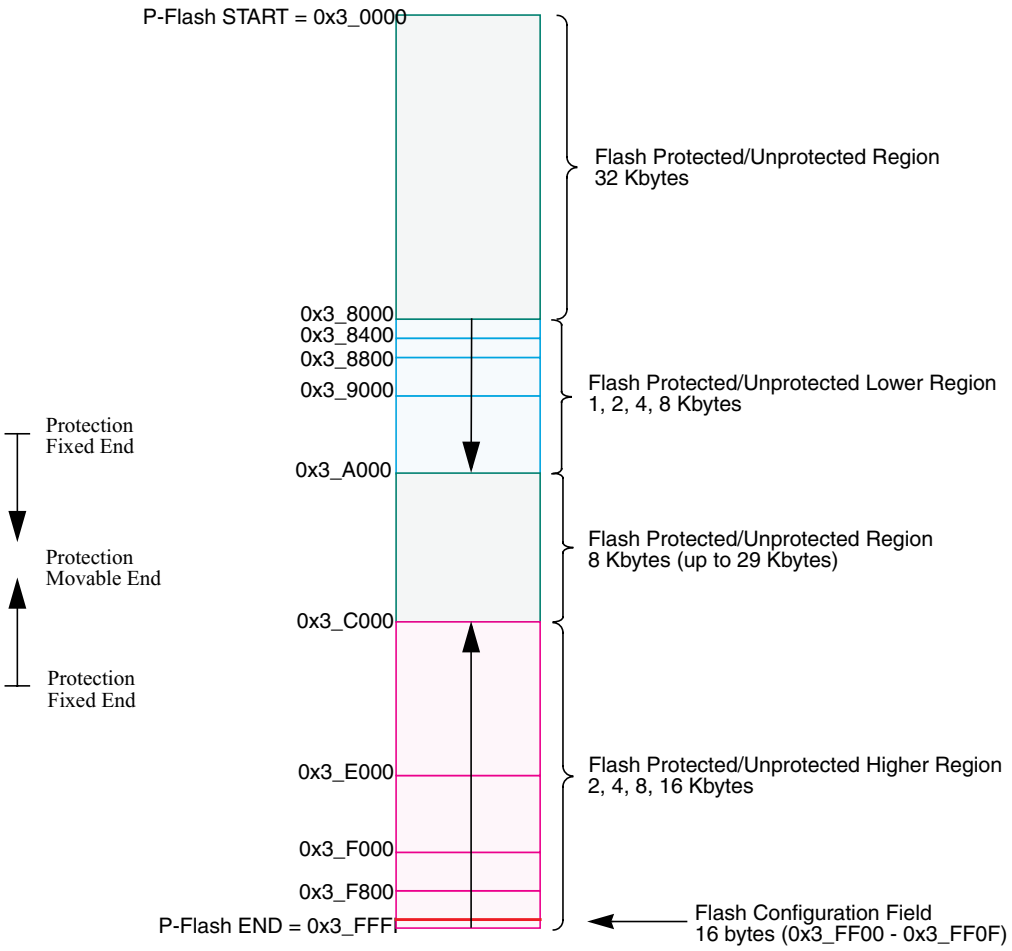
protect regions in the Flash memory from accidental program or erase. Three separate memory regions, one growing upward from global address 0x3_8000 in the Flash memory (called the lower region), one growing downward from global address 0x3_FFFF in the Flash memory (called the higher region), and the remaining addresses in the Flash memory, can be activated for protectio. Two separate memory regions,

one growing downward from global address 0x3_FFFF in the Flash memory (called the higher region), and the remaining addresses in the Flash memory, can be activated for protectionThe Flash memory addresses covered by these protectable regions are shown in the P-Flash memory map. The higher address region is mainly targeted to hold the boot loader code since it covers the vector space. Default protection settings as well as security information that allows the MCU to restrict access to the Flash module are stored in the Flash configuration field as described in Table 17-4.

**Table 17-4. Flash Configuration Field**

| Global Address | Size (Bytes) | Description |
|---|---|---|
| 0x3_FF00-0x3_FF07 | 8 | Backdoor Comparison Key<br>Refer to Section 17.4.6.11, “Verify Backdoor Access Key Command,” and Section 17.5.1, “Unsecuring the MCU using Backdoor Key Access” |
| 0x3_FF08-0x3_FF0B[1] | 4 | Reserved |
| 0x3_FF0C[1] | 1 | P-Flash Protection byte.<br>Refer to Section 17.3.2.9, “P-Flash Protection Register (FPROT)” |
| 0x3_FF0D[1] | 1 | EEPROM Protection byte.<br>Refer to Section 17.3.2.10, “EEPROM Protection Register (EEPROT)” |
| 0x3_FF0E[1] | 1 | Flash Nonvolatile byte<br>Refer to Section 17.3.2.16, “Flash Option Register (FOPT)” |
| 0x3_FF0F[1] | 1 | Flash Security byte<br>Refer to Section 17.3.2.2, “Flash Security Register (FSEC)” |

1 0x3FF08-0x3_FF0F form a Flash phrase and must be programmed in a single command write sequence. Each byte in the 0x3_FF08 - 0x3_FF0B reserved field should be programmed to 0xFF.


P-Flash START = 0x3_0000
Flash Protected/Unprotected Region
32 Kbytes
0x3_8000
0x3_8400
0x3_8800
0x3_9000
Flash Protected/Unprotected Lower Region
1, 2, 4, 8 Kbytes
Protection
Fixed End
0x3_A000
Flash Protected/Unprotected Region
8 Kbytes (up to 29 Kbytes)
Protection
Movable End
0x3_C000
Protection
Fixed End
0x3_E000
Flash Protected/Unprotected Higher Region
2, 4, 8, 16 Kbytes
0x3_F000
0x3_F800
P-Flash END = 0x3_FFFF
Flash Configuration Field
16 bytes (0x3_FF00 - 0x3_FF0F)

### P-Flash Memory Map

**Table 17-5. Program IFR Fields**

| Global Address | Size (Bytes) | Field Description |
|---|---|---|
| 0x0_4000 – 0x0_4007 | 8 | Reserved |
| 0x0_4008 – 0x0_40B5 | 174 | Reserved |
| 0x0_40B6 – 0x0_40B7 | 2 | Version ID[1] |
| 0x0_40B8 – 0x0_40BF | 8 | Reserved |
| 0x0_40C0 – 0x0_40FF | 64 | Program Once Field<br>Refer to Section 17.4.6.6, “Program Once Command” |

1 Used to track firmware patch versions, see Section 17.4.2 IFR Version ID Word

**Table 17-6. Memory Controller Resource Fields (NVMRES[1]=1)**

| Global Address | Size (Bytes) | Description |
|---|---|---|
| 0x0_4000 – 0x040FF | 256 | P-Flash IFR (see Table 17-5) |
| 0x0_4100 – 0x0_41FF | 256 | Reserved. |
| 0x0_4200 – 0x0_57FF | | Reserved |
| 0x0_5800 – 0x0_59FF | 512 | Reserved |
| 0x0_5A00 – 0x0_5FFF | 1,536 | Reserved |
| 0x0_6000 – 0x0_6BFF | 3,072 | Reserved |
| 0x0_6C00 – 0x0_7FFF | 5,120 | Reserved |

1 NVMRES - See Section 17.4.3 Internal NVM resource (NVMRES) for NVMRES (NVM Resource) detail.

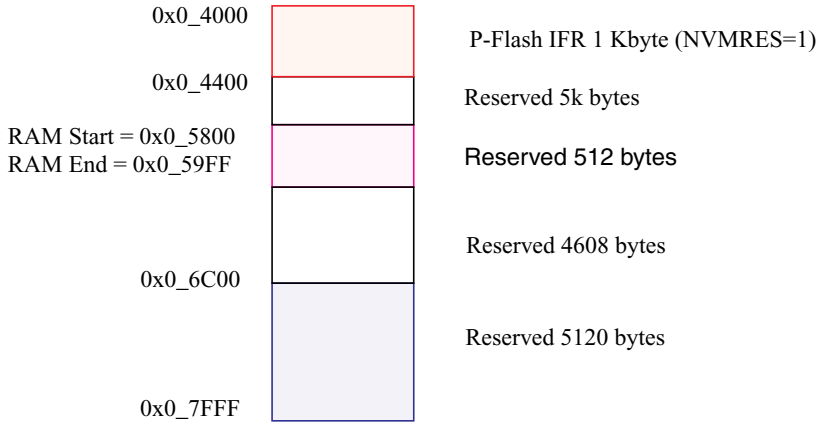



**Figure 17-2. Memory Controller Resource Memory Map (NVMRES=1)**

## 17.3.2 Register Descriptions

The Flash module contains a set of 20 control and status registers located between Flash module base + 0x0000 and 0x0013.

In the case of the writable registers, the write accesses are forbidden during Fash command execution (for more detail, see Caution note in Section 17.3 Memory Map and Registers).

A summary of the Flash module registers is given in Figure 17-3 with detailed descriptions in the following subsections.

| Address & Name | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FCLKDIV | R | FDIVLD | FDIVLCK | FDIV5 | FDIV4 | FDIV3 | FDIV2 | FDIV1 | FDIV0 |
| | W | | | | | | | | |
| FSEC | R | KEYEN1 | KEYEN0 | RNV5 | RNV4 | RNV3 | RNV2 | SEC1 | SEC0 |
| | W | | | | | | | | |
| FCCOBIX | R | 0 | 0 | 0 | 0 | 0 | CCOBIX2 | CCOBIX1 | CCOBIX0 |
| | W | | | | | | | | |

**Figure 17-3. FTMRG64K512 Register Summary**

| Address & Name | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FRSV0 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| FCNFG | R | CCIE | 0 | 0 | IGNSF | 0 | 0 | FDFD | FSFD |
| | W | | | | | | | | |
| FERCNFG | R | 0 | 0 | 0 | 0 | 0 | 0 | DFDIE | SFDIE |
| | W | | | | | | | | |
| FSTAT | R | CCIF | 0 | ACCERR | FPVIOL | MGBUSY | RSVD | MGSTAT1 | MGSTAT0 |
| | W | | | | | | | | |
| FERSTAT | R | 0 | 0 | 0 | 0 | 0 | 0 | DFDIF | SFDIF |
| | W | | | | | | | | |
| FPROT | R | FPOPEN | RNV6 | FPHDIS | FPHS1 | FPHS0 | FPLDIS | FPLS1 | FPLS0 |
| | W | | | | | | | | |
| EEPROT | R | DPOPEN | 0 | 0 | 0 | DPS3 | DPS2 | DPS1 | DPS0 |
| | W | | | | | | | | |
| FCCOBHI | R | CCOB15 | CCOB14 | CCOB13 | CCOB12 | CCOB11 | CCOB10 | CCOB9 | CCOB8 |
| | W | | | | | | | | |
| FCCOBLO | R | CCOB7 | CCOB6 | CCOB5 | CCOB4 | CCOB3 | CCOB2 | CCOB1 | CCOB0 |
| | W | | | | | | | | |
| FRSV1 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| FRSV2 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| FRSV3 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| FRSV4 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| FOPT | R | NV7 | NV6 | NV5 | NV4 | NV3 | NV2 | NV1 | NV0 |
| | W | | | | | | | | |

**Figure 17-3. FTMRG64K512 Register Summary (continued)**

| Address & Name | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FRSV5 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| FRSV6 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |
| FRSV7 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | | | | | | | | |

= Unimplemented or Reserved

**Figure 17-3. FTMRG64K512 Register Summary (continued)**

### 17.3.2.1 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 | FDIVLCK | FDIV[5:0] | | | | | |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 17-4. Flash Clock Divider Register (FCLKDIV)**

All bits in the FCLKDIV register are readable, bit 7 is not writable, bit 6 is write-once-hi and controls the writability of the FDIV field in normal mode. In special mode, bits 6-0 are writable any number of times but bit 7 remains unwritable.

**CAUTION**

The FCLKDIV register should never be written while a Flash command is executing (CCIF=0).

**Table 17-7. FCLKDIV Field Descriptions**

| Field | Description |
|---|---|
| 7<br>FDIVLD | **Clock Divider Loaded**<br>0 FCLKDIV register has not been written since the last reset<br>1 FCLKDIV register has been written since the last reset |

**Table 17-7. FCLKDIV Field Descriptions (continued)**

| Field | Description |
|---|---|
| 6<br>FDIVLCK | **Clock Divider Locked**<br>0 FDIV field is open for writing<br>1 FDIV value is locked and cannot be changed. Once the lock bit is set high, only reset can clear this bit and restore writability to the FDIV field in normal mode. |
| 5–0<br>FDIV[5:0] | **Clock Divider Bits** — FDIV[5:0] must be set to effectively divide BUSCLK down to 1 MHz to control timed events during Flash program and erase algorithms. Table 17-8 shows recommended values for FDIV[5:0] based on the BUSCLK frequency. Please refer to Section 17.4.4, “Flash Command Operations,” for more information. |

**Table 17-8. FDIV values for various BUSCLK Frequencies**

| BUSCLK Frequency (MHz) | | FDIV[5:0] | BUSCLK Frequency (MHz) | | FDIV[5:0] |
|---|---|---|---|---|---|
| MIN[1] | MAX[2] | | MIN[1] | MAX[2] | |
| 1.0 | 1.6 | 0x00 | 16.6 | 17.6 | 0x10 |
| 1.6 | 2.6 | 0x01 | 17.6 | 18.6 | 0x11 |
| 2.6 | 3.6 | 0x02 | 18.6 | 19.6 | 0x12 |
| 3.6 | 4.6 | 0x03 | 19.6 | 20.6 | 0x13 |
| 4.6 | 5.6 | 0x04 | 20.6 | 21.6 | 0x14 |
| 5.6 | 6.6 | 0x05 | 21.6 | 22.6 | 0x15 |
| 6.6 | 7.6 | 0x06 | 22.6 | 23.6 | 0x16 |
| 7.6 | 8.6 | 0x07 | 23.6 | 24.6 | 0x17 |
| 8.6 | 9.6 | 0x08 | 24.6 | 25.6 | 0x18 |
| 9.6 | 10.6 | 0x09 | | | |
| 10.6 | 11.6 | 0x0A | | | |
| 11.6 | 12.6 | 0x0B | | | |
| 12.6 | 13.6 | 0x0C | | | |
| 13.6 | 14.6 | 0x0D | | | |
| 14.6 | 15.6 | 0x0E | | | |
| 15.6 | 16.6 | 0x0F | | | |

1 BUSCLK is Greater Than this value.

2 BUSCLK is Less Than or Equal to this value.

### 17.3.2.2 Flash Security Register (FSEC)

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

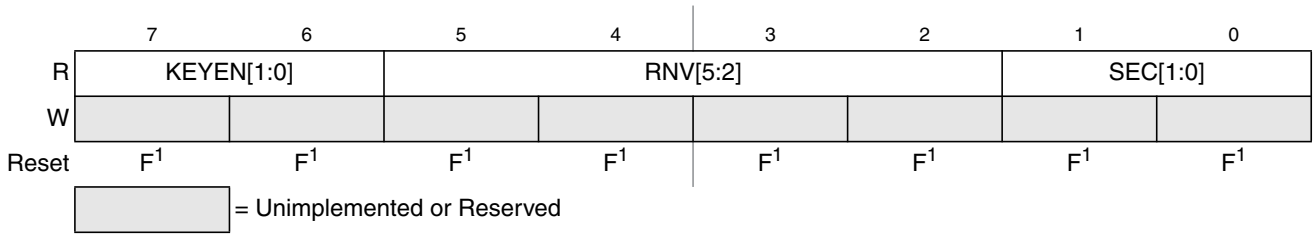



**Figure 17-5. Flash Security Register (FSEC)**

[1] Loaded from IFR Flash configuration field, during reset sequence.

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

During the reset sequence, the FSEC register is loaded with the contents of the Flash security byte in the Flash configuration field at global address 0x3_FF0F located in P-Flash memory (see Table 17-4) as indicated by reset condition F in Figure 17-5. If a double bit fault is detected while reading the P-Flash phrase containing the Flash security byte during the reset sequence, all bits in the FSEC register will be set to leave the Flash module in a secured state with backdoor key access disabled.

**Table 17-9. FSEC Field Descriptions**

| Field | Description |
|---|---|
| 7–6<br>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 17-10. |
| 5–2<br>RNV[5:2] | **Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements. |
| 1–0<br>SEC[1:0] | **Flash Security Bits** — The SEC[1:0] bits define the security state of the MCU as shown in Table 17-11. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10. |

**Table 17-10. Flash KEYEN States**

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

[1] Preferred KEYEN state to disable backdoor key access.

**Table 17-11. Flash Security States**

| SEC[1:0] | Status of Security |
|---|---|
| 00 | SECURED |
| 01 | SECURED[1] |
| 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 17.5 Security.

### 17.3.2.3 Flash CCOB Index Register (FCCOBIX)

The FCCOBIX register is used to index the FCCOB register for Flash memory operations.

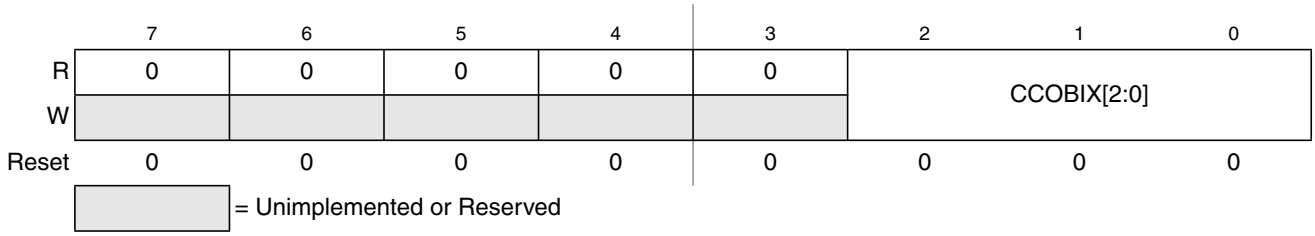



**Figure 17-6. FCCOB Index Register (FCCOBIX)**

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

**Table 17-12. FCCOBIX Field Descriptions**

| Field | Description |
|---|---|
| 2–0 CCOBIX[1:0] | **Common Command Register Index**— The CCOBIX bits are used to select which word of the FCCOB register array is being read or written to. See 17.3.2.11 Flash Common Command Object Register (FCCOB)," for more details. |

### 17.3.2.4 Flash Reserved0 Register (FRSV0)

This Flash register is reserved for factory testing.

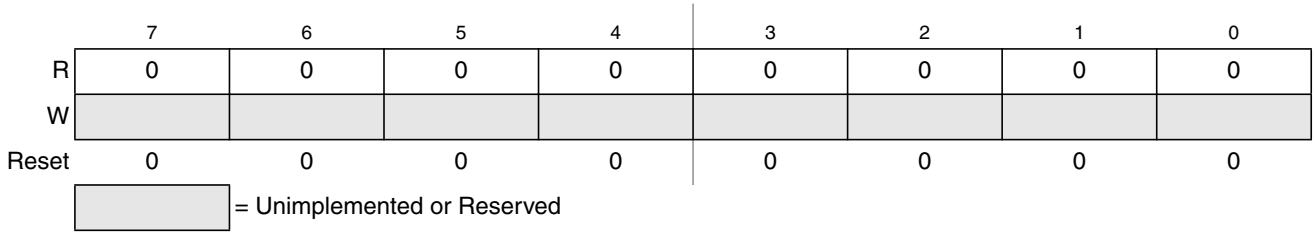



**Figure 17-7. Flash Reserved0 Register (FRSV0)**

All bits in the FRSV0 register read 0 and are not writable.

### 17.3.2.5 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash command complete interrupt and forces ECC faults on Flash array read access from the CPU.

**Figure 17-8. Flash Configuration Register (FCNFG)**

CCIE, IGNSF, FDFD, and FSFD bits are readable and writable while remaining bits read 0 and are not writable.

**Table 17-13. FCNFG Field Descriptions**

| Field | Description |
|---|---|
| 7<br>CCIE | **Command Complete Interrupt Enable** — The CCIE bit controls interrupt generation when a Flash command has completed.<br>0 Command complete interrupt disabled<br>1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set (see Section 17.3.2.7 Flash Status Register (FSTAT)) |
| 4<br>IGNSF | **Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see Section 17.3.2.8 Flash Error Status Register (FERSTAT)).<br>0 All single bit faults detected during array reads are reported<br>1 Single bit faults detected during array reads are not reported and the single bit fault interrupt will not be generated |
| 1<br>FDFD | **Force Double Bit Fault Detect** — The FDFD bit allows the user to simulate a double bit fault during Flash array read operations and check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD.<br>0 Flash array read operations will set the DFDIF flag in the FERSTAT register only if a double bit fault is detected<br>1 Any Flash array read operation will force the DFDIF flag in the FERSTAT register to be set (see Section 17.3.2.7 Flash Status Register (FSTAT)) and an interrupt will be generated as long as the DFDIE interrupt enable in the FERCNFG register is set (see Section 17.3.2.6 Flash Error Configuration Register (FERCNFG)) |
| 0<br>FSFD | **Force Single Bit Fault Detect** — The FSFD bit allows the user to simulate a single bit fault during Flash array read operations and check the associated interrupt routine. The FSFD bit is cleared by writing a 0 to FSFD.<br>0 Flash array read operations will set the SFDIF flag in the FERSTAT register only if a single bit fault is detected<br>1 Flash array read operation will force the SFDIF flag in the FERSTAT register to be set (see Section 17.3.2.7 Flash Status Register (FSTAT)) and an interrupt will be generated as long as the SFDIE interrupt enable in the FERCNFG register is set (see Section 17.3.2.6 Flash Error Configuration Register (FERCNFG)) |

## 17.3.2.6 Flash Error Configuration Register (FERCNFG)

The FERCNFG register enables the Flash error interrupts for the FERSTAT flags.

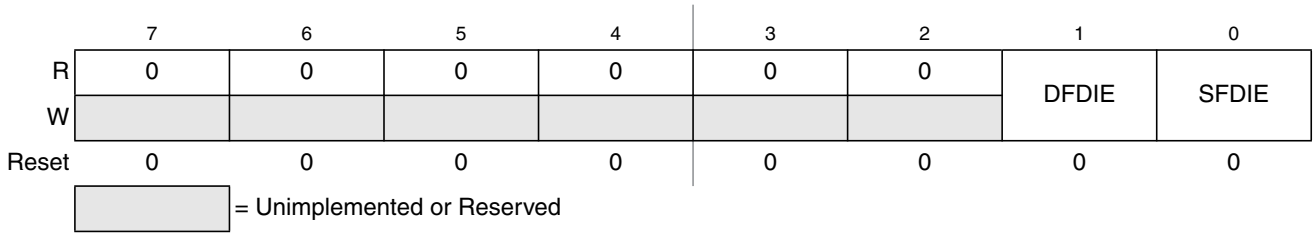



**Figure 17-9. Flash Error Configuration Register (FERCNFG)**

All assigned bits in the FERCNFG register are readable and writable.

**Table 17-14. FERCNFG Field Descriptions**

| Field | Description |
|---|---|
| 1<br>DFDIE | **Double Bit Fault Detect Interrupt Enable** — The DFDIE bit controls interrupt generation when a double bit fault is detected during a Flash block read operation.<br>0 DFDIF interrupt disabled<br>1 An interrupt will be requested whenever the DFDIF flag is set (see Section 17.3.2.8 Flash Error Status Register (FERSTAT)) |
| 0<br>SFDIE | **Single Bit Fault Detect Interrupt Enable** — The SFDIE bit controls interrupt generation when a single bit fault is detected during a Flash block read operation.<br>0 SFDIF interrupt disabled whenever the SFDIF flag is set (see Section 17.3.2.8 Flash Error Status Register (FERSTAT))<br>1 An interrupt will be requested whenever the SFDIF flag is set (see Section 17.3.2.8 Flash Error Status Register (FERSTAT)) |

## 17.3.2.7 Flash Status Register (FSTAT)

The FSTAT register reports the operational status of the Flash module.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | CCIF | 0 | ACCERR | FPVIOL | MGBUSY | RSVD | MGSTAT[1:0] | |
| W | | | | | | | | |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0[1] | 0[1] |

= Unimplemented or Reserved

**Figure 17-10. Flash Status Register (FSTAT)**

1 Reset value can deviate from the value shown if a double bit fault is detected during the reset sequence (see Section 17.6 Initialization).

CCIF, ACCERR, and FPVIOL bits are readable and writable, MGBUSY and MGSTAT bits are readable but not writable, while remaining bits read 0 and are not writable.

**Table 17-15. FSTAT Field Descriptions**

| Field | Description |
|---|---|
| 7<br>CCIF | **Command Complete Interrupt Flag** — The CCIF flag indicates that a Flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command and CCIF will stay low until command completion or command violation.<br>0 Flash command in progress<br>1 Flash command has completed |
| 5<br>ACCERR | **Flash Access Error Flag** — The ACCERR bit indicates an illegal access has occurred to the Flash memory caused by either a violation of the command write sequence (see Section 17.4.4.2 Command Write Sequence) or issuing an illegal Flash command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR bit has no effect on ACCERR.<br>0 No access error detected<br>1 Access error detected |

**Table 17-15. FSTAT Field Descriptions (continued)**

| Field | Description |
|---|---|
| 4<br>FPVIOL | **Flash Protection Violation Flag** —The FPVIOL bit indicates an attempt was made to program or erase an address in a protected area of P-Flash or EEPROM memory during a command write sequence. The FPVIOL bit is cleared by writing a 1 to FPVIOL. Writing a 0 to the FPVIOL bit has no effect on FPVIOL. While FPVIOL is set, it is not possible to launch a command or start a command write sequence.<br>0 No protection violation detected<br>1 Protection violation detected |
| 3<br>MGBUSY | **Memory Controller Busy Flag** — The MGBUSY flag reflects the active state of the Memory Controller.<br>0 Memory Controller is idle<br>1 Memory Controller is busy executing a Flash command (CCIF = 0) |
| 2<br>RSVD | **Reserved Bit** — This bit is reserved and always reads 0. |
| 1–0<br>MGSTAT[1:0] | **Memory Controller Command Completion Status Flag** — One or more MGSTAT flag bits are set if an error is detected during execution of a Flash command or during the Flash reset sequence. See Section 17.4.6, “Flash Command Description,” and Section 17.6, “Initialization” for details. |

### 17.3.2.8 Flash Error Status Register (FERSTAT)

The FERSTAT register reflects the error status of internal Flash operations.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | DFDIF | SFDIF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 17-11. Flash Error Status Register (FERSTAT)**

All flags in the FERSTAT register are readable and only writable to clear the flag.

**Table 17-16. FERSTAT Field Descriptions**

| Field | Description |
|---|---|
| 1<br>DFDIF | **Double Bit Fault Detect Interrupt Flag** — The setting of the DFDIF flag indicates that a double bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation returning invalid data was attempted on a Flash block that was under a Flash command operation.[1] The DFDIF flag is cleared by writing a 1 to DFDIF. Writing a 0 to DFDIF has no effect on DFDIF.[2]<br>0 No double bit fault detected<br>1 Double bit fault detected or a Flash array read operation returning invalid data was attempted while command running |
| 0<br>SFDIF | **Single Bit Fault Detect Interrupt Flag** — With the IGNSF bit in the FCNFG register clear, the SFDIF flag indicates that a single bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation returning invalid data was attempted on a Flash block that was under a Flash command operation.[1] The SFDIF flag is cleared by writing a 1 to SFDIF. Writing a 0 to SFDIF has no effect on SFDIF.<br>0 No single bit fault detected<br>1 Single bit fault detected and corrected or a Flash array read operation returning invalid data was attempted while command running |

1 The single bit fault and double bit fault flags are mutually exclusive for parity errors (an ECC fault occurrence can be either single fault or double fault but never both). A simultaneous access collision (Flash array read operation returning invalid data attempted while command running) is indicated when both SFDIF and DFDIF flags are high.

2 There is a one cycle delay in storing the ECC DFDIF and SFDIF fault flags in this register. At least one NOP is required after a flash memory read before checking FERSTAT for the occurrence of ECC errors.

### 17.3.2.9 P-Flash Protection Register (FPROT)

The FPROT register defines which P-Flash sectors are protected against program and erase operations.

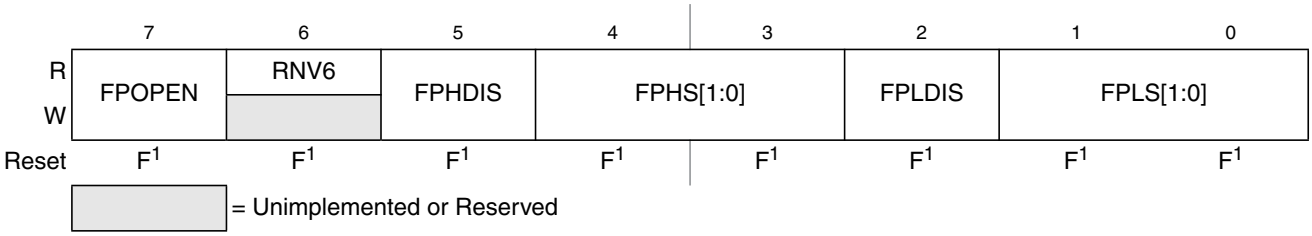



**Figure 17-12. Flash Protection Register (FPROT)**

1 Loaded from IFR Flash configuration field, during reset sequence.

The (unreserved) bits of the FPROT register are writable in Normal Single Chip Mode with the restriction that the size of the protected region can only be increased (see Section 17.3.2.9.1, “P-Flash Protection Restrictions,” and Table 17-21).All (unreserved) bits of the FPROT register are writable without restriction in Special Single Chip Mode .

During the reset sequence, the FPROT register is loaded with the contents of the P-Flash protection byte in the Flash configuration field at global address 0x3_FF0C located in P-Flash memory (see Table 17-4) as indicated by reset condition ‘F’ in Figure 17-12. To change the P-Flash protection that will be loaded during the reset sequence, the upper sector of the P-Flash memory must be unprotected, then the P-Flash protection byte must be reprogrammed. If a double bit fault is detected while reading the P-Flash phrase containing the P-Flash protection byte during the reset sequence, the FPOPEN bit will be cleared and remaining bits in the FPROT register will be set to leave the P-Flash memory fully protected.

Trying to alter data in any protected area in the P-Flash memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. The block erase of a P-Flash block is not possible if any of the P-Flash sectors contained in the same P-Flash block are protected.

**Table 17-17. FPROT Field Descriptions**

| Field | Description |
|---|---|
| 7<br>FPOPEN | **Flash Protection Operation Enable** — The FPOPEN bit determines the protection function for program or erase operations as shown in Table 17-18 for the P-Flash block.<br>0 When FPOPEN is clear, the FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS and FPLS bits<br>1 When FPOPEN is set, the FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS and FPLS bits |
| 6<br>RNV[6] | **Reserved Nonvolatile Bit** — The RNV bit should remain in the erased state for future enhancements. |

**Table 17-17. FPROT Field Descriptions (continued)**

| Field | Description |
|---|---|
| 5<br>FPHDIS | **Flash Protection Higher Address Range Disable** — The FPHDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory ending with global address 0x3_FFFF.<br>0 Protection/Unprotection enabled<br>1 Protection/Unprotection disabled |
| 4–3<br>FPHS[1:0] | **Flash Protection Higher Address Size** — The FPHS bits determine the size of the protected/unprotected area in P-Flash memory as shown inTable 17-19. The FPHS bits can only be written to while the FPHDIS bit is set. |
| 2<br>FPLDIS | **Flash Protection Lower Address Range Disable** — The FPLDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory beginning with global address 0x3_8000.<br>0 Protection/Unprotection enabled<br>1 Protection/Unprotection disabled |
| 1–0<br>FPLS[1:0] | **Flash Protection Lower Address Size** — The FPLS bits determine the size of the protected/unprotected area in P-Flash memory as shown in Table 17-20. The FPLS bits can only be written to while the FPLDIS bit is set. |

**Table 17-18. P-Flash Protection Function**

| FPOPEN | FPHDIS | FPLDIS | Function[1] |
|---|---|---|---|
| 1 | 1 | 1 | No P-Flash 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 P-Flash Memory 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 Table 17-19 and Table 17-20.

**Table 17-19. P-Flash Protection Higher Address Range**

| FPHS[1:0] | Global Address Range | Protected Size |
|---|---|---|
| 00 | 0x3_F800–0x3_FFFF | 2 Kbytes |
| 01 | 0x3_F000–0x3_FFFF | 4 Kbytes |
| 10 | 0x3_E000–0x3_FFFF | 8 Kbytes |
| 11 | 0x3_C000–0x3_FFFF | 16 Kbytes |

**Table 17-20. P-Flash Protection Lower Address Range**

| FPLS[1:0] | Global Address Range | Protected Size |
|---|---|---|
| 00 | 0x3_8000–0x3_83FF | 1 Kbyte |
| 01 | 0x3_8000–0x3_87FF | 2 Kbytes |
| 10 | 0x3_8000–0x3_8FFF | 4 Kbytes |
| 11 | 0x3_8000–0x3_9FFF | 8 Kbytes |

All possible P-Flash protection scenarios are shown in Figure 17-13 . Although the protection scheme is loaded from the Flash memory at global address 0x3_FF0C during the reset sequence, it can be changed by the user. The P-Flash protection scheme can be used by applications requiring reprogramming in Normal Single Chip Mode while providing as much protection as possible if reprogramming is not required.

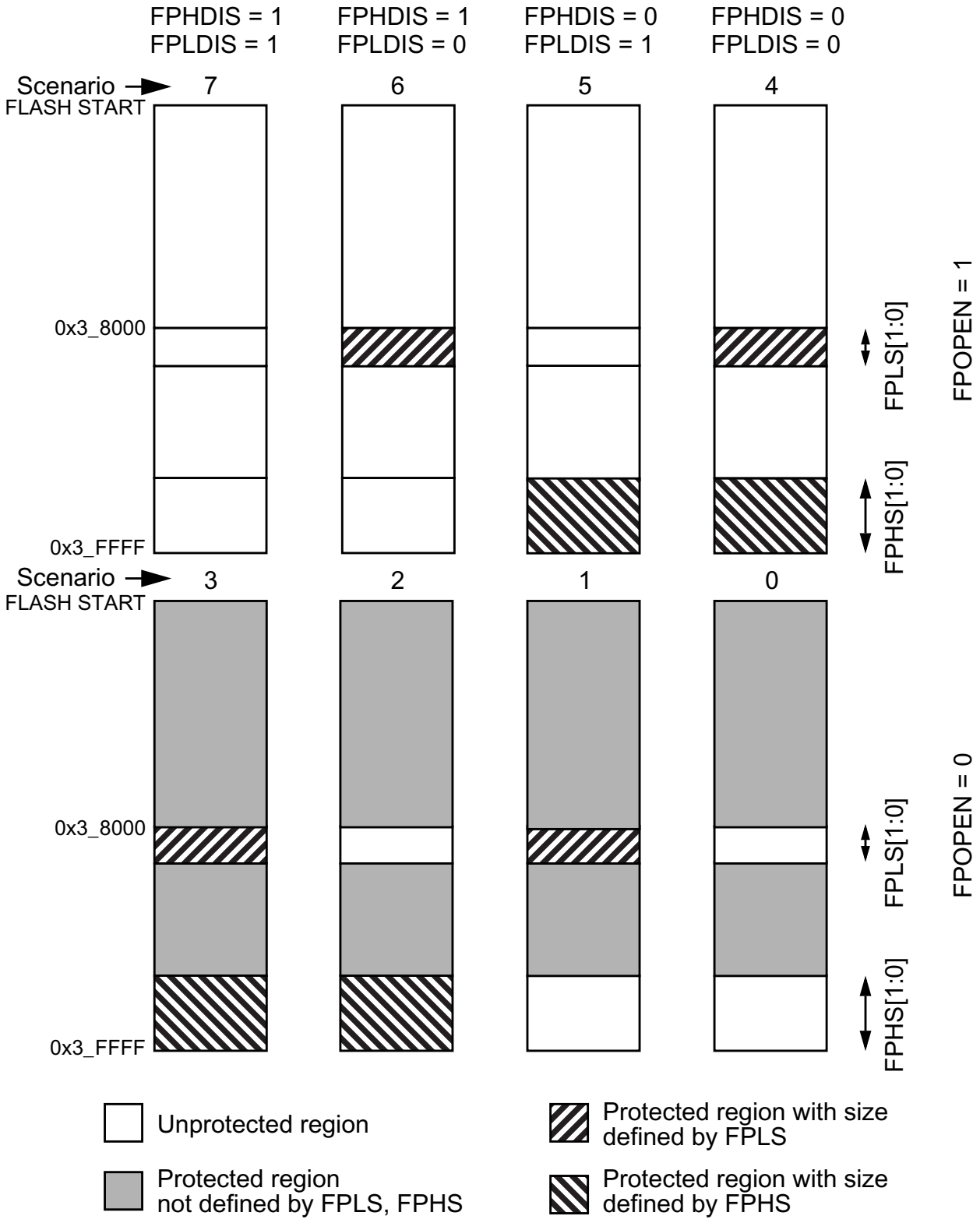



**Figure 17-13. P-Flash Protection Scenarios**

#### 17.3.2.9.1 P-Flash Protection Restrictions

In Normal Single Chip Mode the general guideline is that P-Flash protection can only be added and not removed. Table 17-21 specifies all valid transitions between P-Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS bit descriptions for additional restrictions.

**Table 17-21. P-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, see Figure 17-13 for a definition of the scenarios.

### 17.3.2.10 EEPROM Protection Register (EEPROT)

The EEPROT register defines which EEPROM sectors are protected against program and erase operations.

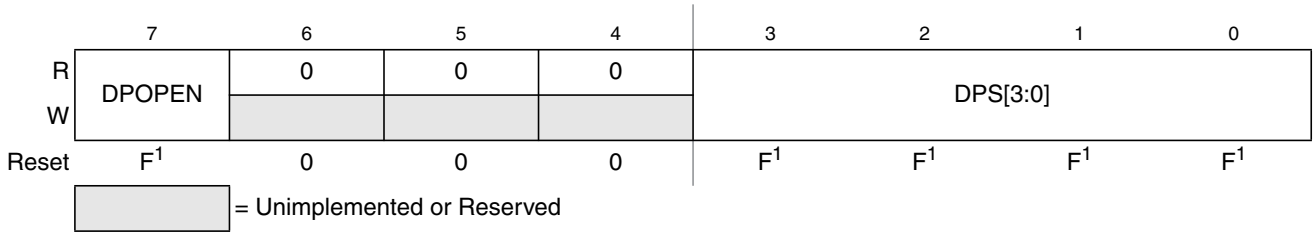



**Figure 17-14. EEPROM Protection Register (EEPROT)**

[1] Loaded from IFR Flash configuration field, during reset sequence.

The (unreserved) bits of the EEPROT register are writable in Normal Single Chip Mode with the restriction that protection can be added but not removed. Writes in Normal Single Chip Mode must increase the DPS value and the DPOPEN bit can only be written from 1 (protection disabled) to 0 (protection enabled). If the DPOPEN bit is set, the state of the DPS bits is irrelevant.All DPOPEN/DPS bit registers are writable without restriction in Special Single Chip Mode.

During the reset sequence, fields DPOPEN and DPS of the EEPROT register are loaded with the contents of the EEPROM protection byte in the Flash configuration field at global address 0x3_FF0D located in P-Flash memory (see Table 17-4) as indicated by reset condition F in . To change the EEPROM protection

that will be loaded during the reset sequence, the P-Flash sector containing the EEPROM protection byte must be unprotected, then the EEPROM protection byte must be programmed. If a double bit fault is detected while reading the P-Flash phrase containing the EEPROM protection byte during the reset sequence, the DPOPEN bit will be cleared and DPS bits will be set to leave the EEPROM memory fully protected.

Trying to alter data in any protected area in the EEPROM memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. Block erase of the EEPROM memory is not possible if any of the EEPROM sectors are protected.

**Table 17-22. EEPROT Field Descriptions**

| Field | Description |
|---|---|
| 7<br>DPOPEN | **EEPROM Protection Control**<br>0 Enables EEPROM memory protection from program and erase with protected address range defined by DPS bits<br>1 Disables EEPROM memory protection from program and erase |
| 3–0<br>DPS[3:0] | **EEPROM Protection Size** — The DPS[3:0] bits determine the size of the protected area in the EEPROM memory as shown inTable 17-23 . |

**Table 17-23. EEPROM Protection Address Range**

| DPS[3:0] | Global Address Range | Protected Size |
|---|---|---|
| 0000 | 0x0_0400 – 0x0_041F | 32 bytes |
| 0001 | 0x0_0400 – 0x0_043F | 64 bytes |
| 0010 | 0x0_0400 – 0x0_045F | 96 bytes |
| 0011 | 0x0_0400 – 0x0_047F | 128 bytes |
| 0100 | 0x0_0400 – 0x0_049F | 160 bytes |
| 0101 | 0x0_0400 – 0x0_04BF | 192 bytes |
| The Protection Size goes on enlarging in step of 32 bytes, for each DPS value increasing of one.<br>.<br>.<br>. | | |
| 1111 | 0x0_0400 – 0x0_05FF | 512 bytes |

### 17.3.2.11 Flash Common Command Object Register (FCCOB)

The FCCOB is an array of six words addressed via the CCOBIX index found in the FCCOBIX register. Byte wide reads and writes are allowed to the FCCOB register.

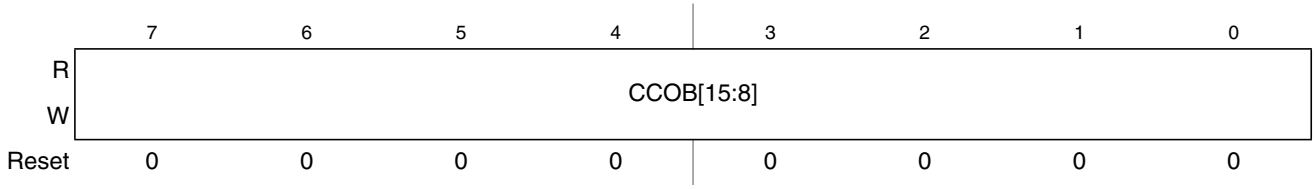



**Figure 17-15. Flash Common Command Object High Register (FCCOBHI)**

**Figure 17-16. Flash Common Command Object Low Register (FCCOBLO)**

### 17.3.2.11.1 FCCOB - NVM Command Mode

NVM command mode uses the indexed FCCOB register to provide a command code and its relevant parameters to the Memory Controller. The user first sets up all required FCCOB fields and then initiates the command's execution by writing a 1 to the CCIF bit in the FSTAT register (a 1 written by the user clears the CCIF command completion flag to 0). When the user clears the CCIF bit in the FSTAT register all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (as evidenced by the Memory Controller returning CCIF to 1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in NVM command mode is shown in Table 17-24. The return values are available for reading after the CCIF flag in the FSTAT register has been returned to 1 by the Memory Controller. Writes to the unimplemented parameter fields (CCOBIX = 110 and CCOBIX = 111) are ignored with reads from these fields returning 0x0000.

Table 17-24 shows the generic Flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific Flash command. For details on the FCCOB settings required by each command, see the Flash command descriptions in Section 17.4.6 Flash Command Description.

**Table 17-24. FCCOB - NVM Command Mode (Typical Usage)**

| CCOBIX[2:0] | Byte | FCCOB Parameter Fields (NVM Command Mode) |
|---|---|---|
| 000 | HI | FCMD[7:0] defining Flash command |
| | LO | 6'h0, Global address [17:16] |
| 001 | HI | Global address [15:8] |
| | LO | Global address [7:0] |
| 010 | HI | Data 0 [15:8] |
| | LO | Data 0 [7:0] |
| 011 | HI | Data 1 [15:8] |
| | LO | Data 1 [7:0] |

**Table 17-24. FCCOB - NVM Command Mode (Typical Usage)**

| CCOBIX[2:0] | Byte | FCCOB Parameter Fields (NVM Command Mode) |
|---|---|---|
| 100 | HI | Data 2 [15:8] |
| | LO | Data 2 [7:0] |
| 101 | HI | Data 3 [15:8] |
| | LO | Data 3 [7:0] |

### 17.3.2.12 Flash Reserved1 Register (FRSV1)

This Flash register is reserved for factory testing.

| | 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 17-17. Flash Reserved1 Register (FRSV1)**

All bits in the FRSV1 register read 0 and are not writable.

### 17.3.2.13 Flash Reserved2 Register (FRSV2)

This Flash register is reserved for factory testing.

| | 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 17-18. Flash Reserved2 Register (FRSV2)**

All bits in the FRSV2 register read 0 and are not writable.

### 17.3.2.14 Flash Reserved3 Register (FRSV3)

This Flash register is reserved for factory testing.

| | 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 17-19. Flash Reserved3 Register (FRSV3)**

All bits in the FRSV3 register read 0 and are not writable.

### 17.3.2.15 Flash Reserved4 Register (FRSV4)

This Flash register is reserved for factory testing.

| | 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 17-20. Flash Reserved4 Register (FRSV4)**

All bits in the FRSV4 register read 0 and are not writable.

### 17.3.2.16 Flash Option Register (FOPT)

The FOPT register is the Flash option register.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | NV[7:0] | | | | | | | |
| W | | | | | | | | |
| Reset | F[1] | F[1] | F[1] | F[1] | F[1] | F[1] | F[1] | F[1] |

= Unimplemented or Reserved

**Figure 17-21. Flash Option Register (FOPT)**

1 Loaded from IFR Flash configuration field, during reset sequence.

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

During the reset sequence, the FOPT register is loaded from the Flash nonvolatile byte in the Flash configuration field at global address 0x3_FF0E located in P-Flash memory (see Table 17-4) as indicated by reset condition F in Figure 17-21. If a double bit fault is detected while reading the P-Flash phrase containing the Flash nonvolatile byte during the reset sequence, all bits in the FOPT register will be set.

**Table 17-25. FOPT Field Descriptions**

| Field | Description |
|---|---|
| 7–0<br>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. |

### 17.3.2.17 Flash Reserved5 Register (FRSV5)

This Flash register is reserved for factory testing.

| | 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 17-22. Flash Reserved5 Register (FRSV5)**

All bits in the FRSV5 register read 0 and are not writable.

### 17.3.2.18 Flash Reserved6 Register (FRSV6)

This Flash register is reserved for factory testing.

| | 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 17-23. Flash Reserved6 Register (FRSV6)**

All bits in the FRSV6 register read 0 and are not writable.

### 17.3.2.19 Flash Reserved7 Register (FRSV7)

This Flash register is reserved for factory testing.

| | 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 17-24. Flash Reserved7 Register (FRSV7)**

All bits in the FRSV7 register read 0 and are not writable.

## 17.4 Functional Description

### 17.4.1 Modes of Operation

The FTMRG64K512 module provides the modes of operation normal and special . The operating mode is determined by module-level inputs and affects the FCLKDIV, FCNFG, FPROT and EEPROT registers (see Table 17-27).

### 17.4.2 IFR Version ID Word

The version ID word is stored in the IFR at address 0x0_40B6. The contents of the word are defined in Table 17-26.

**Table 17-26. IFR Version ID Fields**

| [15:4] | [3:0] |
|---|---|
| Reserved | VERNUM |

- VERNUM: Version number. The first version is number 0b_0001 with both 0b_0000 and 0b_1111 meaning ‘none’.

### 17.4.3 Internal NVM resource (NVMRES)

IFR is an internal NVM resource readable by CPU , when NVMRES is active. The IFR fields are shown in Table 17-5.

The NVMRES global address map is shown in Table 17-6.

### 17.4.4 Flash Command Operations

Flash command operations are used to modify Flash memory contents.

The next sections describe:

- How to write the FCLKDIV register that is used to generate a time base (FCLK) derived from BUSCLK for Flash program and erase command operations
- The command write sequence used to set Flash command parameters and launch execution
- Valid Flash commands available for execution, according to MCU functional mode and MCU security state.

#### 17.4.4.1 Writing the FCLKDIV Register

Prior to issuing any Flash program or erase command after a reset, the user is required to write the FCLKDIV register to divide BUSCLK down to a target FCLK of 1 MHz. Table 17-8 shows recommended values for the FDIV field based on BUSCLK frequency.

**NOTE**

Programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 0.8 MHz. Setting FDIV too high can destroy the Flash memory due to overstress. Setting FDIV too low can result in incomplete programming or erasure of the Flash memory cells.

When 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. If the FCLKDIV register has not been written, any Flash program or erase command loaded during a command write sequence will not execute and the ACCERR bit in the FSTAT register will set.

#### 17.4.4.2 Command Write Sequence

The Memory Controller will launch all valid Flash commands entered using a command write sequence.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be clear (see Section 17.3.2.7 Flash Status Register (FSTAT)) and the CCIF flag should be tested to determine the status of the current command write sequence. If CCIF is 0, the previous command write sequence is still active, a new command write sequence cannot be started, and all writes to the FCCOB register are ignored.

##### 17.4.4.2.1 Define FCCOB Contents

The FCCOB parameter fields must be loaded with all required parameters for the Flash command being executed. Access to the FCCOB parameter fields is controlled via the CCOBIX bits in the FCCOBIX register (see Section 17.3.2.3 Flash CCOB Index Register (FCCOBIX)).

The contents of the FCCOB parameter fields are transferred to the Memory Controller when the user clears the CCIF command completion flag in the FSTAT register (writing 1 clears the CCIF to 0). The CCIF flag will remain clear until the Flash command has completed. Upon completion, the Memory Controller will return CCIF to 1 and the FCCOB register will be used to communicate any results. The flow for a generic command write sequence is shown in Figure 17-25.

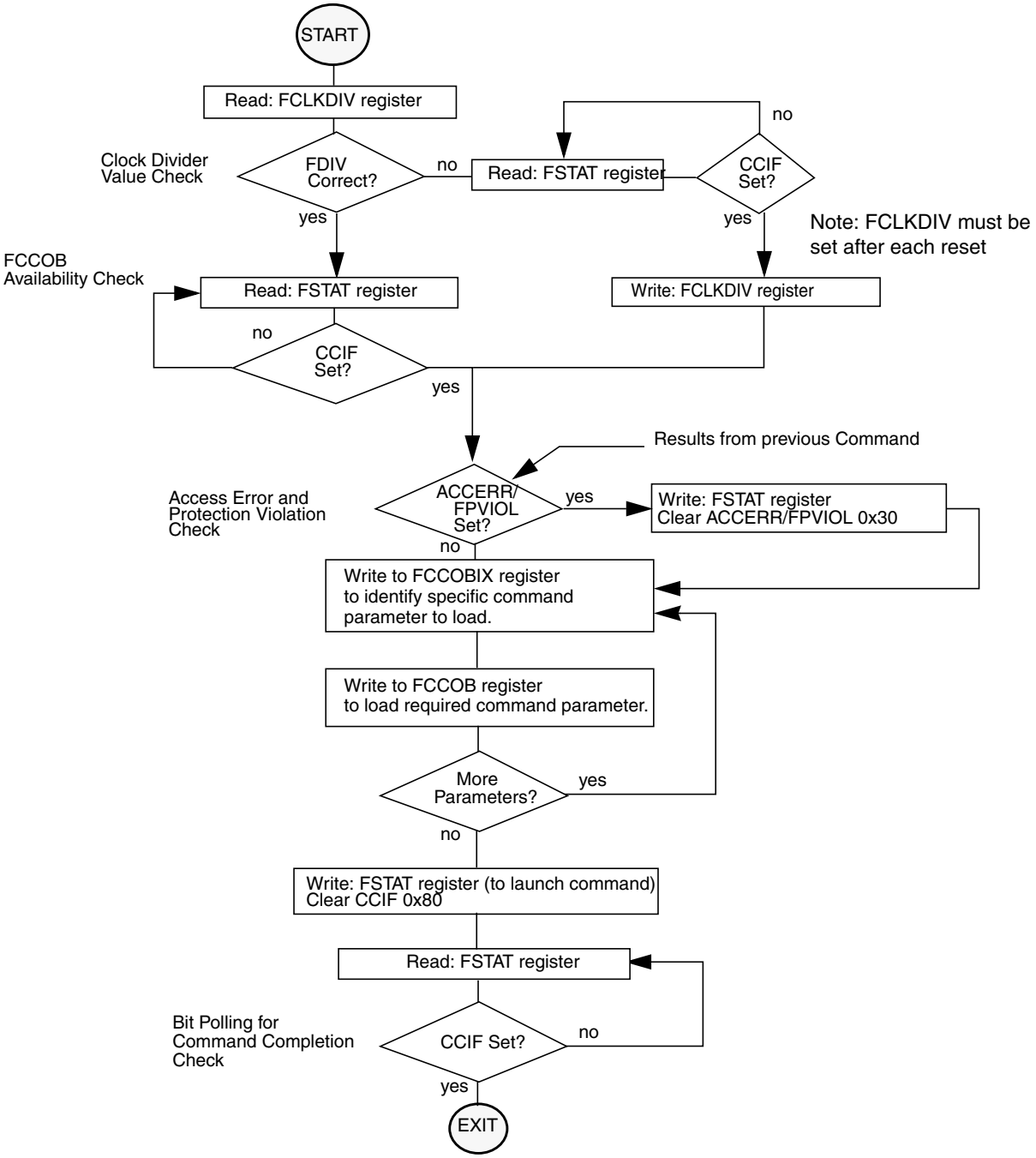



**Figure 17-25. Generic Flash Command Write Sequence Flowchart**

### 17.4.4.3 Valid Flash Module Commands

Table 17-27 present the valid Flash commands, as enabled by the combination of the functional MCU mode (Normal SingleChip NS, Special Singlechip SS) with the MCU security state (Unsecured, Secured).

Special Singlechip mode is selected by input mmc_ss_mode_ts2 asserted. MCU Secured state is selected by input mmc_secure input asserted.

**Table 17-27. Flash Commands by Mode and Security State**

| FCMD | Command | Unsecured | | Secured | |
|---|---|---|---|---|---|
| | | NS[1] | SS[2] | NS[3] | SS[4] |
| 0x01 | Erase Verify All Blocks | * | * | * | * |
| 0x02 | Erase Verify Block | * | * | * | * |
| 0x03 | Erase Verify P-Flash Section | * | * | * | |
| 0x04 | Read Once | * | * | * | |
| 0x06 | Program P-Flash | * | * | * | |
| 0x07 | Program Once | * | * | * | |
| 0x08 | Erase All Blocks | | * | | * |
| 0x09 | Erase Flash Block | * | * | * | |
| 0x0A | Erase P-Flash Sector | * | * | * | |
| 0x0B | Unsecure Flash | | * | | * |
| 0x0C | Verify Backdoor Access Key | * | | * | |
| 0x0D | Set User Margin Level | * | * | * | |
| 0x0E | Set Field Margin Level | | * | | |
| 0x10 | Erase Verify EEPROM Section | * | * | * | |
| 0x11 | Program EEPROM | * | * | * | |
| 0x12 | Erase EEPROM Sector | * | * | * | |

1 Unsecured Normal Single Chip mode
2 Unsecured Special Single Chip mode.
3 Secured Normal Single Chip mode.
4 Secured Special Single Chip mode.

### 17.4.4.4 P-Flash Commands

Table 17-28 summarizes the valid P-Flash commands along with the effects of the commands on the P-Flash block and other resources within the Flash module.

**Table 17-28. P-Flash Commands**

| FCMD | Command | Function on P-Flash Memory |
|---|---|---|
| 0x01 | Erase Verify All Blocks | Verify that all P-Flash (and EEPROM) blocks are erased. |
| 0x02 | Erase Verify Block | Verify that a P-Flash block is erased. |
| 0x03 | Erase Verify P-Flash Section | Verify that a given number of words starting at the address provided are erased. |
| 0x04 | Read Once | Read a dedicated 64 byte field in the nonvolatile information register in P-Flash block that was previously programmed using the Program Once command. |
| 0x06 | Program P-Flash | Program a phrase in a P-Flash block. |
| 0x07 | Program Once | Program a dedicated 64 byte field in the nonvolatile information register in P-Flash block that is allowed to be programmed only once. |
| 0x08 | Erase All Blocks | Erase all P-Flash (and EEPROM) blocks.<br>An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the DPOPEN bit in the EEPROT register are set prior to launching the command. |
| 0x09 | Erase Flash Block | Erase a P-Flash (or EEPROM) block.<br>An erase of the full P-Flash block is only possible when FPLDIS, FPHDIS and FPOPEN bits in the FPROT register are set prior to launching the command. |
| 0x0A | Erase P-Flash Sector | Erase all bytes in a P-Flash sector. |
| 0x0B | Unsecure Flash | Supports a method of releasing MCU security by erasing all P-Flash (and EEPROM) blocks and verifying that all P-Flash (and EEPROM) blocks are erased. |
| 0x0C | Verify Backdoor Access Key | Supports a method of releasing MCU security by verifying a set of security keys. |
| 0x0D | Set User Margin Level | Specifies a user margin read level for all P-Flash blocks. |
| 0x0E | Set Field Margin Level | Specifies a field margin read level for all P-Flash blocks (special modes only). |

### 17.4.4.5 EEPROM Commands

Table 17-29 summarizes the valid EEPROM commands along with the effects of the commands on the EEPROM block.

**Table 17-29. EEPROM Commands**

| FCMD | Command | Function on EEPROM Memory |
|---|---|---|
| 0x01 | Erase Verify All Blocks | Verify that all EEPROM (and P-Flash) blocks are erased. |
| 0x02 | Erase Verify Block | Verify that the EEPROM block is erased. |

**Table 17-29. EEPROM Commands**

| FCMD | Command | Function on EEPROM Memory |
|---|---|---|
| 0x08 | Erase All Blocks | Erase all EEPROM (and P-Flash) blocks.<br>An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the DPOPEN bit in the EEPROT register are set prior to launching the command. |
| 0x09 | Erase Flash Block | Erase a EEPROM (or P-Flash) block.<br>An erase of the full EEPROM block is only possible when DPOPEN bit in the EEPROT register is set prior to launching the command. |
| 0x0B | Unsecure Flash | Supports a method of releasing MCU security by erasing all EEPROM (and P-Flash) blocks and verifying that all EEPROM (and P-Flash) blocks are erased. |
| 0x0D | Set User Margin Level | Specifies a user margin read level for the EEPROM block. |
| 0x0E | Set Field Margin Level | Specifies a field margin read level for the EEPROM block (special modes only). |
| 0x10 | Erase Verify EEPROM Section | Verify that a given number of words starting at the address provided are erased. |
| 0x11 | Program EEPROM | Program up to four words in the EEPROM block. |
| 0x12 | Erase EEPROM Sector | Erase all bytes in a sector of the EEPROM block. |

## 17.4.5 Allowed Simultaneous P-Flash and EEPROM Operations

Only the operations marked ‘OK’ in Table 17-30 are permitted to be run simultaneously on the Program Flash and EEPROM blocks. Some operations cannot be executed simultaneously because certain hardware resources are shared by the two memories. The priority has been placed on permitting Program Flash reads while program and erase operations execute on the EEPROM, providing read (P-Flash) while write (EEPROM) functionality.

**Table 17-30. Allowed P-Flash and EEPROM Simultaneous Operations**

| | EEPROM | | | | |
|---|---|---|---|---|---|
| **Program Flash** | **Read** | **Margin Read[1]** | **Program** | **Sector Erase** | **Mass Erase[2]** |
| **Read** | | OK | OK | OK | |
| **Margin Read[1]** | | | | | |
| **Program** | | | | | |
| **Sector Erase** | | | | | |
| **Mass Erase[2]** | | | | | OK |

1 A ‘Margin Read’ is any read after executing the margin setting commands ‘Set User Margin Level’ or ‘Set Field Margin Level’ with anything but the ‘normal’ level specified. See the Note on margin settings in Section 17.4.6.12 Set User Margin Level Command and Section 17.4.6.13 Set Field Margin Level Command.

[2] The ‘Mass Erase’ operations are commands ‘Erase All Blocks’ and ‘Erase Flash Block’

## 17.4.6 Flash Command Description

This section provides details of all available Flash commands launched by a command write sequence. The ACCERR bit in the FSTAT register will be set during the command write sequence if any of the following illegal steps are performed, causing the command not to be processed by the Memory Controller:

- Starting any command write sequence that programs or erases Flash memory before initializing the FCLKDIV register
- Writing an invalid command as part of the command write sequence
- For additional possible errors, refer to the error handling table provided for each command

If a Flash block is read during execution of an algorithm (CCIF = 0) on that same block, the read operation will return invalid data if both flags SFDIF and DFDIF are set. If the SFDIF or DFDIF flags were not previously set when the invalid read operation occurred, both the SFDIF and DFDIF flags will be set.

If the ACCERR or FPVIOL bits are set in the FSTAT register, the user must clear these bits before starting any command write sequence (see Section 17.3.2.7 Flash Status Register (FSTAT)).

**CAUTION**

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

### 17.4.6.1 Erase Verify All Blocks Command

The Erase Verify All Blocks command will verify that all P-Flash and EEPROM blocks have been erased.

**Table 17-31. Erase Verify All Blocks Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x01 | Not required |

Upon clearing CCIF to launch the Erase Verify All Blocks command, the Memory Controller will verify that the entire Flash memory space is erased. The CCIF flag will set after the Erase Verify All Blocks operation has completed. If all blocks are not erased, it means blank check failed, both MGSTAT bits will be set.

**Table 17-32. Erase Verify All Blocks Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 000 at command launch |
| | FPVIOL | None |
| | MGSTAT1 | Set if any errors have been encountered during the reador if blank check failed . |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the read or if blank check failed. |

### 17.4.6.2 Erase Verify Block Command

The Erase Verify Block command allows the user to verify that an entire P-Flash or EEPROM block has been erased. The FCCOB FlashBlockSelectionCode[1:0]bits determine which block must be verified.

**Table 17-33. Erase Verify Block Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x02 | Flash block selection code [1:0]. See Table 17-34 |

**Table 17-34. Flash block selection code description**

| Selection code[1:0] | Flash block to be verified |
|---|---|
| 00 | EEPROM |
| 01 | Invalid (ACCERR) |
| 10 | Invalid (ACCERR) |
| 11 | P-Flash |

Upon clearing CCIF to launch the Erase Verify Block command, the Memory Controller will verify that the selected P-Flash or EEPROM block is erased. The CCIF flag will set after the Erase Verify Block operation has completed.If the block is not erased, it means blank check failed, both MGSTAT bits will be set.

**Table 17-35. Erase Verify Block Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 000 at command launch |
| | | Set if an invalid FlashBlockSelectionCode[1:0] is supplied |
| | FPVIOL | None |
| | MGSTAT1 | Set if any errors have been encountered during the read or if blank check failed. |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the read or if blank check failed. |

### 17.4.6.3 Erase Verify P-Flash Section Command

The Erase Verify P-Flash Section command will verify that a section of code in the P-Flash memory is erased. The Erase Verify P-Flash Section command defines the starting point of the code to be verified and the number of phrases.

**Table 17-36. Erase Verify P-Flash Section Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x03 | Global address [17:16] of a P-Flash block |
| 001 | Global address [15:0] of the first phrase to be verified | |
| 010 | Number of phrases to be verified | |

Upon clearing CCIF to launch the Erase Verify P-Flash Section command, the Memory Controller will verify the selected section of Flash memory is erased. The CCIF flag will set after the Erase Verify P-Flash Section operation has completed. If the section is not erased, it means blank check failed, both MGSTAT bits will be set.

**Table 17-37. Erase Verify P-Flash Section Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 010 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid global address [17:0] is supplied see ) |
| | | Set if a misaligned phrase address is supplied (global address [2:0] != 000) |
| | | Set if the requested section crosses a the P-Flash address boundary |
| | FPVIOL | None |
| | MGSTAT1 | Set if any errors have been encountered during the read or if blank check failed. |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the read or if blank check failed. |

### 17.4.6.4 Read Once Command

The Read Once command provides read access to a reserved 64 byte field (8 phrases) located in the nonvolatile information register of P-Flash. The Read Once field is programmed using the Program Once command described in Section 17.4.6.6 Program Once Command. The Read Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

**Table 17-38. Read Once Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x04 | Not Required |
| 001 | Read Once phrase index (0x0000 - 0x0007) | |
| 010 | Read Once word 0 value | |

**Table 17-38. Read Once Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters |
|---|---|
| 011 | Read Once word 1 value |
| 100 | Read Once word 2 value |
| 101 | Read Once word 3 value |

Upon clearing CCIF to launch the Read Once command, a Read Once phrase is fetched and stored in the FCCOB indexed register. The CCIF flag will set after the Read Once operation has completed. Valid phrase index values for the Read Once command range from 0x0000 to 0x0007. During execution of the Read Once command, any attempt to read addresses within P-Flash block will return invalid data.

**Table 17-39. Read Once Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 001 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid phrase index is supplied |
| | FPVIOL | None |
| | MGSTAT1 | Set if any errors have been encountered during the read |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the read |

### 17.4.6.5 Program P-Flash Command

The Program P-Flash operation will program a previously erased phrase in the P-Flash memory using an embedded algorithm.

**CAUTION**

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

**Table 17-40. Program P-Flash Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x06 | Global address [17:16] to identify P-Flash block |
| 001 | Global address [15:0] of phrase location to be programmed[1] | |
| 010 | Word 0 program value | |
| 011 | Word 1 program value | |
| 100 | Word 2 program value | |
| 101 | Word 3 program value | |

1 Global address [2:0] must be 000

Upon clearing CCIF to launch the Program P-Flash command, the Memory Controller will program the data words to the supplied global address and will then proceed to verify the data words read back as expected. The CCIF flag will set after the Program P-Flash operation has completed.

**Table 17-41. Program P-Flash Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 101 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid global address [17:0] is supplied see ) |
| | | Set if a misaligned phrase address is supplied (global address [2:0] != 000) |
| | FPVIOL | Set if the global address [17:0] points to a protected area |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

### 17.4.6.6 Program Once Command

The Program Once command restricts programming to a reserved 64 byte field (8 phrases) in the nonvolatile information register located in P-Flash. The Program Once reserved field can be read using the Read Once command as described in Section 17.4.6.4 Read Once Command. The Program Once command must only be issued once since the nonvolatile information register in P-Flash cannot be erased. The Program Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

**Table 17-42. Program Once Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x07 | Not Required |
| 001 | Program Once phrase index (0x0000 - 0x0007) | |
| 010 | Program Once word 0 value | |
| 011 | Program Once word 1 value | |
| 100 | Program Once word 2 value | |
| 101 | Program Once word 3 value | |

Upon clearing CCIF to launch the Program Once command, the Memory Controller first verifies that the selected phrase is erased. If erased, then the selected phrase will be programmed and then verified with read back. The CCIF flag will remain clear, setting only after the Program Once operation has completed.

The reserved nonvolatile information register accessed by the Program Once command cannot be erased and any attempt to program one of these phrases a second time will not be allowed. Valid phrase index values for the Program Once command range from 0x0000 to 0x0007. During execution of the Program Once command, any attempt to read addresses within P-Flash will return invalid data.

**Table 17-43. Program Once Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 101 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid phrase index is supplied |
| | | Set if the requested phrase has already been programmed[1] |
| | FPVIOL | None |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

1 If a Program Once phrase is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command will be allowed to execute again on that same phrase.

### 17.4.6.7 Erase All Blocks Command

The Erase All Blocks operation will erase the entire P-Flash and EEPROM memory space.

**Table 17-44. Erase All Blocks Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x08 | Not required |

Upon clearing CCIF to launch the Erase All Blocks command, the Memory Controller will erase the entire Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag will set after the Erase All Blocks operation has completed.

**Table 17-45. Erase All Blocks Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 000 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | FPVIOL | Set if any area of the P-Flash or EEPROM memory is protected |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

### 17.4.6.8 Erase Flash Block Command

The Erase Flash Block operation will erase all addresses in a P-Flash or EEPROM block.

**Table 17-46. Erase Flash Block Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x09 | Global address [17:16] to identify Flash block |
| 001 | Global address [15:0] in Flash block to be erased | |

Upon clearing CCIF to launch the Erase Flash Block command, the Memory Controller will erase the selected Flash block and verify that it is erased. The CCIF flag will set after the Erase Flash Block operation has completed.

**Table 17-47. Erase Flash Block Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 001 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid global address [17:16] is supplied |
| | | Set if the supplied P-Flash address is not phrase-aligned or if the EEPROM address is not word-aligned |
| | FPVIOL | Set if an area of the selected Flash block is protected |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

### 17.4.6.9 Erase P-Flash Sector Command

The Erase P-Flash Sector operation will erase all addresses in a P-Flash sector.

**Table 17-48. Erase P-Flash Sector Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x0A | Global address [17:16] to identify P-Flash block to be erased |
| 001 | Global address [15:0] anywhere within the sector to be erased. Refer to Section 17.1.2.1 P-Flash Features for the P-Flash sector size. | |

Upon clearing CCIF to launch the Erase P-Flash Sector command, the Memory Controller will erase the selected Flash sector and then verify that it is erased. The CCIF flag will be set after the Erase P-Flash Sector operation has completed.

**Table 17-49. Erase P-Flash Sector Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 001 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid global address [17:16] is supplied see ) |
| | | Set if a misaligned phrase address is supplied (global address [2:0] != 000) |
| | FPVIOL | Set if the selected P-Flash sector is protected |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

### 17.4.6.10 Unsecure Flash Command

The Unsecure Flash command will erase the entire P-Flash and EEPROM memory space and, if the erase is successful, will release security.

**Table 17-50. Unsecure Flash Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x0B | Not required |

Upon clearing CCIF to launch the Unsecure Flash command, the Memory Controller will erase the entire P-Flash and EEPROM memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. If the erase verify is not successful, the Unsecure Flash operation sets MGSTAT1 and terminates without changing the security state. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag is set after the Unsecure Flash operation has completed.

**Table 17-51. Unsecure Flash Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 000 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | FPVIOL | Set if any area of the P-Flash or EEPROM memory is protected |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

### 17.4.6.11 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command will only execute if it is enabled by the KEYEN bits in the FSEC register (see Table 17-10). The Verify Backdoor Access Key command releases security if user-supplied keys match those stored in the Flash security bytes of the Flash configuration field (see

Table 17-4). The Verify Backdoor Access Key command must not be executed from the Flash block containing the backdoor comparison key to avoid code runaway.

**Table 17-52. Verify Backdoor Access Key Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x0C | Not required |
| 001 | Key 0 | |
| 010 | Key 1 | |
| 011 | Key 2 | |
| 100 | Key 3 | |

Upon clearing CCIF to launch the Verify Backdoor Access Key command, the Memory Controller will check the FSEC KEYEN bits to verify that this command is enabled. If not enabled, the Memory Controller sets the ACCERR bit in the FSTAT register and terminates. If the command is enabled, the Memory Controller compares the key provided in FCCOB to the backdoor comparison key in the Flash configuration field with Key 0 compared to 0x3_FF00, etc. If the backdoor keys match, security will be released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are aborted (set ACCERR) until a reset occurs. The CCIF flag is set after the Verify Backdoor Access Key operation has completed.

**Table 17-53. Verify Backdoor Access Key Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 100 at command launch |
| | | Set if an incorrect backdoor key is supplied |
| | | Set if backdoor key access has not been enabled (KEYEN[1:0] != 10, see Section 17.3.2.2 Flash Security Register (FSEC)) |
| | | Set if the backdoor key has mismatched since the last reset |
| | FPVIOL | None |
| | MGSTAT1 | None |
| | MGSTAT0 | None |

### 17.4.6.12 Set User Margin Level Command

The Set User Margin Level command causes the Memory Controller to set the margin level for future read operations of the P-Flash or EEPROM block.

**Table 17-54. Set User Margin Level Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x0D | Flash block selection code [1:0]. See Table 17-34 |
| 001 | Margin level setting. | |

Upon clearing CCIF to launch the Set User Margin Level command, the Memory Controller will set the user margin level for the targeted block and then set the CCIF flag.

**NOTE**

When the EEPROM block is targeted, the EEPROM user margin levels are applied only to the EEPROM reads. However, when the P-Flash block is targeted, the P-Flash user margin levels are applied to both P-Flash and EEPROM reads. It is not possible to apply user margin levels to the P-Flash block only.

Valid margin level settings for the Set User Margin Level command are defined in Table 17-55.

**Table 17-55. Valid Set User Margin Level Settings**

| CCOB (CCOBIX=001) | Level Description |
|---|---|
| 0x0000 | Return to Normal Level |
| 0x0001 | User Margin-1 Level[1] |
| 0x0002 | User Margin-0 Level[2] |

[1] Read margin to the erased state

[2] Read margin to the programmed state

**Table 17-56. Set User Margin Level Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 001 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid FlashBlockSelectionCode[1:0] is supplied (See Table 17-34) |
| | | Set if an invalid margin level setting is supplied |
| | FPVIOL | None |
| | MGSTAT1 | None |
| | MGSTAT0 | None |

**NOTE**

User margin levels can be used to check that Flash memory contents have adequate margin for normal level read operations. If unexpected results are encountered when checking Flash memory contents at user margin levels, a potential loss of information has been detected.

### 17.4.6.13 Set Field Margin Level Command

The Set Field Margin Level command, valid in special modes only, causes the Memory Controller to set the margin level specified for future read operations of the P-Flash or EEPROM block.

Upon clearing CCIF to launch the Set Field Margin Level command, the Memory Controller will set the

**Table 17-57. Set Field Margin Level Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x0E | Flash block selection code [1:0]. See Table 17-34 |
| 001 | Margin level setting. | |

field margin level for the targeted block and then set the CCIF flag.

**NOTE**

When the EEPROM block is targeted, the EEPROM field margin levels are applied only to the EEPROM reads. However, when the P-Flash block is targeted, the P-Flash field margin levels are applied to both P-Flash and EEPROM reads. It is not possible to apply field margin levels to the P-Flash block only.

Valid margin level settings for the Set Field Margin Level command are defined in Table 17-58.

**Table 17-58. Valid Set Field Margin Level Settings**

| CCOB (CCOBIX=001) | Level Description |
|---|---|
| 0x0000 | Return to Normal Level |
| 0x0001 | User Margin-1 Level[1] |
| 0x0002 | User Margin-0 Level[2] |
| 0x0003 | Field Margin-1 Level[1] |
| 0x0004 | Field Margin-0 Level[2] |

1 Read margin to the erased state

2 Read margin to the programmed state

**Table 17-59. Set Field Margin Level Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 001 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid FlashBlockSelectionCode[1:0] is supplied (See Table 17-34) |
| | | Set if an invalid margin level setting is supplied |
| | FPVIOL | None |
| | MGSTAT1 | None |
| | MGSTAT0 | None |

**CAUTION**

Field margin levels must only be used during verify of the initial factory programming.

**NOTE**

Field margin levels can be used to check that Flash memory contents have adequate margin for data retention at the normal level setting. If unexpected results are encountered when checking Flash memory contents at field margin levels, the Flash memory contents should be erased and reprogrammed.

### 17.4.6.14 Erase Verify EEPROM Section Command

The Erase Verify EEPROM Section command will verify that a section of code in the EEPROM is erased. The Erase Verify EEPROM Section command defines the starting point of the data to be verified and the number of words.

**Table 17-60. Erase Verify EEPROM Section Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x10 | Global address [17:16] to identify the EEPROM block |
| 001 | Global address [15:0] of the first word to be verified | |
| 010 | Number of words to be verified | |

Upon clearing CCIF to launch the Erase Verify EEPROM Section command, the Memory Controller will verify the selected section of EEPROM memory is erased. The CCIF flag will set after the Erase Verify EEPROM Section operation has completed. If the section is not erased, it means blank check failed, both MGSTAT bits will be set.

**Table 17-61. Erase Verify EEPROM Section Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 010 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid global address [17:0] is supplied |
| | | Set if a misaligned word address is supplied (global address [0] != 0) |
| | | Set if the requested section breaches the end of the EEPROM block |
| | FPVIOL | None |
| | MGSTAT1 | Set if any errors have been encountered during the read or if blank check failed. |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the read or if blank check failed. |

### 17.4.6.15 Program EEPROM Command

The Program EEPROM operation programs one to four previously erased words in the EEPROM block. The Program EEPROM operation will confirm that the targeted location(s) were successfully programmed upon 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.

**Table 17-62. Program EEPROM Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x11 | Global address [17:16] to identify the EEPROM block |
| 001 | Global address [15:0] of word to be programmed | |
| 010 | Word 0 program value | |
| 011 | Word 1 program value, if desired | |
| 100 | Word 2 program value, if desired | |
| 101 | Word 3 program value, if desired | |

Upon clearing CCIF to launch the Program EEPROM command, the user-supplied words will be transferred to the Memory Controller and be programmed if the area is unprotected. The CCOBIX index value at Program EEPROM command launch determines how many words will be programmed in the EEPROM block. The CCIF flag is set when the operation has completed.

**Table 17-63. Program EEPROM Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] < 010 at command launch |
| | | Set if CCOBIX[2:0] > 101 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid global address [17:0] is supplied |
| | | Set if a misaligned word address is supplied (global address [0] != 0) |
| | | Set if the requested group of words breaches the end of the EEPROM block |
| | FPVIOL | Set if the selected area of the EEPROM memory is protected |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

### 17.4.6.16 Erase EEPROM Sector Command

The Erase EEPROM Sector operation will erase all addresses in a sector of the EEPROM block.

**Table 17-64. Erase EEPROM Sector Command FCCOB Requirements**

| CCOBIX[2:0] | FCCOB Parameters | |
|---|---|---|
| 000 | 0x12 | Global address [17:16] to identify EEPROM block |
| 001 | Global address [15:0] anywhere within the sector to be erased. See Section 17.1.2.2 EEPROM Features for EEPROM sector size. | |

Upon clearing CCIF to launch the Erase EEPROM Sector command, the Memory Controller will erase the selected Flash sector and verify that it is erased. The CCIF flag will set after the Erase EEPROM Sector operation has completed.

**Table 17-65. Erase EEPROM Sector Command Error Handling**

| Register | Error Bit | Error Condition |
|---|---|---|
| FSTAT | ACCERR | Set if CCOBIX[2:0] != 001 at command launch |
| | | Set if command not available in current mode (see Table 17-27) |
| | | Set if an invalid global address [17:0] is suppliedsee ) |
| | | Set if a misaligned word address is supplied (global address [0] != 0) |
| | FPVIOL | Set if the selected area of the EEPROM memory is protected |
| | MGSTAT1 | Set if any errors have been encountered during the verify operation |
| | MGSTAT0 | Set if any non-correctable errors have been encountered during the verify operation |

## 17.4.7 Interrupts

The Flash module can generate an interrupt when a Flash command operation has completed or when a Flash command operation has detected an ECC fault.

**Table 17-66. Flash Interrupt Sources**

| Interrupt Source | Interrupt Flag | Local Enable | Global (CCR) Mask |
|---|---|---|---|
| Flash Command Complete | CCIF (FSTAT register) | CCIE (FCNFG register) | I Bit |
| ECC Double Bit Fault on Flash Read | DFDIF (FERSTAT register) | DFDIE (FERCNFG register) | I Bit |
| ECC Single Bit Fault on Flash Read | SFDIF (FERSTAT register) | SFDIE (FERCNFG register) | I Bit |

**NOTE**

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

#### 17.4.7.1 Description of Flash Interrupt Operation

The Flash module uses the CCIF flag in combination with the CCIE interrupt enable bit to generate the Flash command interrupt request. The Flash module uses the DFDIF and SFDIF flags in combination with the DFDIE and SFDIE interrupt enable bits to generate the Flash error interrupt request. For a detailed description of the register bits involved, refer to Section 17.3.2.5, “Flash Configuration Register (FCNFG)”, Section 17.3.2.6, “Flash Error Configuration Register (FERCNFG)”, Section 17.3.2.7, “Flash Status Register (FSTAT)”, and Section 17.3.2.8, “Flash Error Status Register (FERSTAT)”.

The logic used for generating the Flash module interrupts is shown in Figure 17-26.

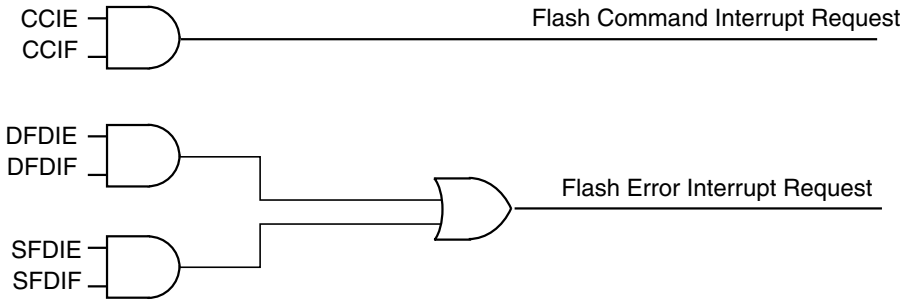



**Figure 17-26. Flash Module Interrupts Implementation**

### 17.4.8 Wait Mode

The Flash module is not affected if the MCU enters wait mode. The Flash module can recover the MCU from wait via the CCIF interrupt (see Section 17.4.7, “Interrupts”).

### 17.4.9 Stop Mode

If a Flash command is active (CCIF = 0) when the MCU requests stop mode, the current Flash operation will be completed before the MCU is allowed to enter stop mode.

## 17.5 Security

The Flash module provides security information to the MCU. The Flash security state is defined by the SEC bits of the FSEC register (see Table 17-11). During reset, the Flash module initializes the FSEC register using data read from the security byte of the Flash configuration field at global address 0x3_FF0F.

The security state out of reset can be permanently changed by programming the security byte assuming that the MCU is starting from a mode where the necessary P-Flash erase and program commands are available and that the upper region of the P-Flash is unprotected. If the Flash security byte is successfully programmed, its new value will take affect after the next MCU reset.

The following subsections describe these security-related subjects:

- Unsecuring the MCU using Backdoor Key Access
- Unsecuring the MCU in Special Single Chip Mode using BDM
- Mode and Security Effects on Flash Command Availability

### 17.5.1 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 0x3_FF00-0x3_FF07). If the KEYEN[1:0] bits are in the enabled state (see Section 17.3.2.2 Flash Security Register (FSEC)), the Verify Backdoor Access Key command (see Section 17.4.6.11 Verify Backdoor Access Key Command) allows the user to present four prospective keys for comparison to the keys stored in the Flash memory via the Memory Controller. If the keys presented in the Verify Backdoor Access Key command match the backdoor keys stored in the Flash memory, the SEC bits in the FSEC register (see Table 17-11) will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, P-Flash memory and EEPROM memory will not be available for read access and will return invalid data.

The user code stored in the P-Flash memory must have a method of receiving the backdoor keys 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 17.3.2.2 Flash Security Register (FSEC)), the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Section 17.4.6.11 Verify Backdoor Access Key Command
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 10

The Verify Backdoor Access Key command is monitored by the Memory Controller and an illegal key will prohibit future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command. The security as defined in the Flash security byte (0x3_FF0F) is not changed by using the Verify Backdoor Access Key command sequence. The backdoor keys stored in addresses 0x3_FF00-0x3_FF07 are unaffected by the Verify Backdoor Access Key command sequence. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the Flash protection register, FPROT.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the Flash security byte can be erased and the Flash security byte can be reprogrammed 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 0x3_FF00-0x3_FF07 in the Flash configuration field.

### 17.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM

A secured MCU can be unsecured in special single chip mode by using the following method to erase the P-Flash and EEPROM memory:

1. Reset the MCU into special single chip mode
2. Delay while the BDM executes the Erase Verify All Blocks command write sequence to check if the P-Flash and EEPROM memories are erased
3. Send BDM commands to disable protection in the P-Flash and EEPROM memory
4. Execute the Erase All Blocks command write sequence to erase the P-Flash and EEPROM memory. Alternatively the Unsecure Flash command can be executed, if so the steps 5 and 6 below are skeeped.
5. After the CCIF flag sets to indicate that the Erase All Blocks operation has completed, reset the MCU into special single chip mode
6. Delay while the BDM executes the Erase Verify All Blocks command write sequence to verify that the P-Flash and EEPROM memory are erased

If the P-Flash and EEPROM memory are verified as erased, the MCU will be unsecured. All BDM commands will now be enabled and the Flash security byte may be programmed to the unsecure state by continuing with the following steps:

7. Send BDM commands to execute the Program P-Flash command write sequence to program the Flash security byte to the unsecured state
8. Reset the MCU

### 17.5.3 Mode and Security Effects on Flash Command Availability

The availability of Flash module commands depends on the MCU operating mode and security state as shown in Table 17-27.

## 17.6 Initialization

On each system reset the flash module executes an initialization sequence which establishes initial values for the Flash Block Configuration Parameters, the FPROT and EEPROT protection registers, and the FOPT and FSEC registers. The initialization routine reverts to built-in default values that leave the module in a fully protected and secured state if errors are encountered during execution of the reset sequence. If a double bit fault is detected during the reset sequence, both MGSTAT bits in the FSTAT register will be set.

CCIF is cleared throughout the initialization sequence. The Flash module holds off all CPU access for a portion of the initialization sequence. Flash reads are allowed once the hold is removed. Completion of the initialization sequence is marked by setting CCIF high which enables user commands.

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.

# Appendix A
# MCU Electrical Specifications

## A.1 General

This supplement contains the most accurate electrical information for the MC9S12VR-Family available at the time of publication.

This introduction is intended to give an overview on several common topics like power supply, current injection etc.

### A.1.1 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.

**NOTE**

This classification is shown in the column labeled “C” in the parameter tables where appropriate.

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.

T: Those parameters are achieved by design characterization on a small sample size from typical devices under typical conditions unless otherwise noted. All values shown in the typical column are within this category.

D: Those parameters are derived mainly from simulations.

**Table A-1. Power Supplies**

| Mnemonic | Nominal Voltage | Description |
|---|---|---|
| VSS | 0V | Ground pin for 1.8V core supply voltage generated by on chip voltage regulator |
| VDDX1[1] | 5.0 V | 5V power supply output for I/O drivers generated by on chip voltage regulator |
| VSSX1[2] | 0V | Ground pin for I/O drivers |
| VDDX2 | 5.0 V | 5V power supply output for I/O drivers generated by on chip voltage regulator |
| VSSX2 | 0V | Ground pin for I/O drivers |
| VDDA[3] | 5.0 V | External power supply for the analog-to-digital converter and for the reference circuit of the internal voltage regulator |
| VSSA | 0V | Ground pin for VDDA analog supply |
| LGND | 0V | Ground pin for LIN physical |
| LSGND | 0V | Ground pin for low-side driver |
| VSUP | 12V/18V | External power supply for voltage regulator |
| VSUPHS | 12V/18V | External power supply for high-side driver |

1 All VDDX pins are internally connected by metal

2 All VSSX pins are internally connected by metal

3 VDDA, VDDX and VSSA, VSSX are connected by diodes for ESD protection

## A.1.2 Pins

There are four groups of functional pins.

### A.1.2.1 I/O Pins

The I/O pins have a level in the range of 3.13V to 5.5V. This class of pins is comprised of all port I/O pins, the analog inputs, BKGD and the $\overline{\text{RESET}}$ pins. Some functionality may be disabled.

### A.1.2.2 High Voltage Pins

LS[1:0], HS[1:0], PL[3:0], VSENSE have a nominal 12V level.

### A.1.2.3 Oscillator

The pins EXTAL, XTAL dedicated to the oscillator have a nominal 1.8V level.

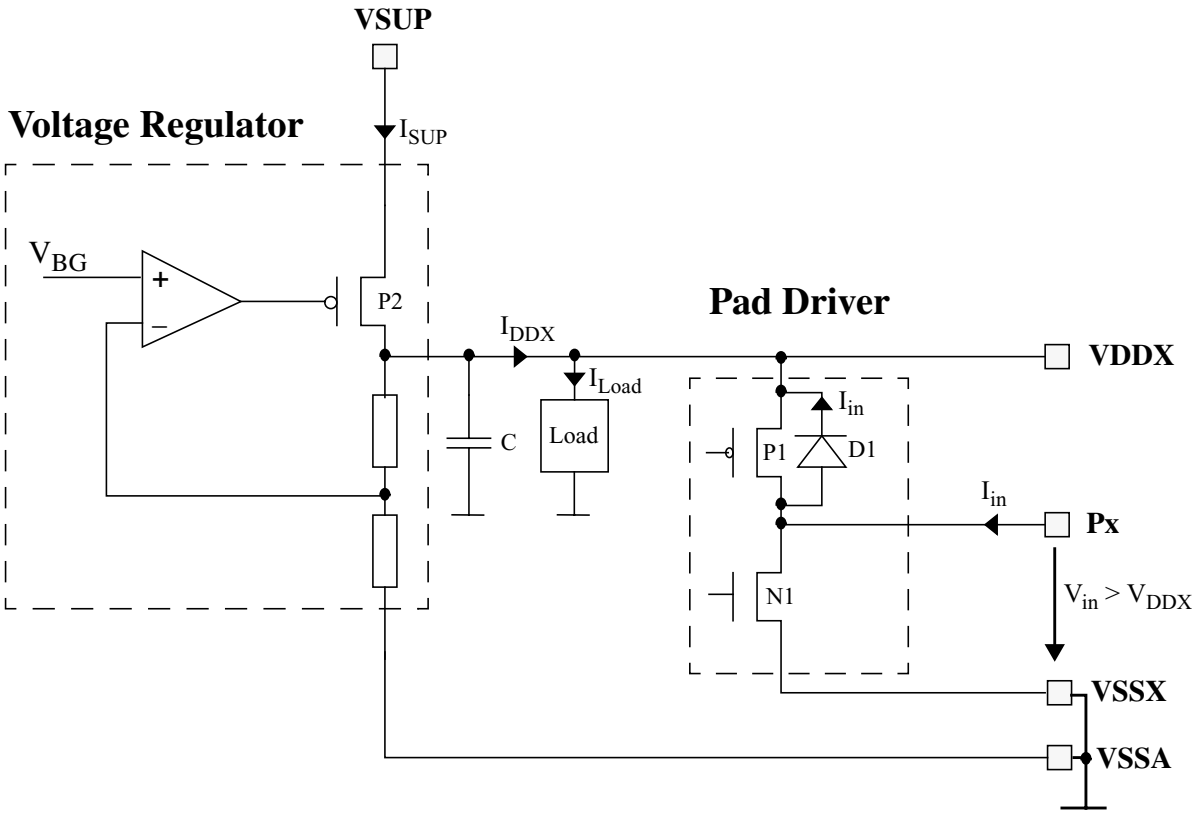
### A.1.2.4 TEST

This pin is used for production testing only. The TEST pin must be tied to ground in all applications.

## A.1.3 Current Injection

Power supply must maintain regulation within operating $V_{DDX}$ or $V_{DD}$ range during instantaneous and operating maximum current conditions. **Figure A-1.** shows a 5V GPIO pad driver and the on chip voltage regulator with VDDX output. It shows also the power & gound pins VSUP, VDDX, VSSX and VSSA. Px represents any 5V GPIO pin. Assume Px is configured as an input. The pad driver transistors P1 and N1 are switched off (high impedance). If the voltage $V_{in}$ on Px is greated than $V_{DDX}$ a positive injection current $I_{in}$ will flow through diode D1 into VDDX node. If this injection current $I_{in}$ is greater than $I_{Load}$, the internal power supply VDDX may go out of regulation. Ensure external $V_{DDX}$ 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.

**Figure A-1. Current Injection on GPIO Port if $V_{in} > V_{DDX}$**




## A.1.4 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.

**Table A-2. Absolute Maximum Ratings[1]**

| Num | Rating | Symbol | Min | Max | Unit |
|---|---|---|---|---|---|
| 1 | Voltage regulator supply voltage | $V_{SUP}$ | -0.3 | 42 | V |
| 2 | LINPHY supply voltage | $V_{LINSUP}$ | -32 | 42 | V |
| 3 | High side driver supply voltage | $V_{SUPHS}$ | -0.3 | 42 | V |
| 4 | Battery sensor input voltage VSENSE pin | $V_{VSENSE_M}$ | -27 | 42 | V |
| 5 | Voltage difference $V_{DDX}$ to $V_{DDA}$[2] | $\Delta_{VDDX}$ | –0.3 | 0.3 | V |
| 6 | Voltage difference $V_{SSX}$ to $V_{SSA}$ | $\Delta_{VSSX}$ | –0.3 | 0.3 | V |
| 7 | Digital I/O input voltage sources | $V_{IN}$ | –0.3 | 6.0 | V |
| 8 | HVI PL[3:0] input voltage (with external resistor $R_{EXT_HVI}$ = 10kΩ) | $V_{Lx}$ | -27 | 42 | V |
| 9 | High-side driver HS[1:0] | $V_{PHS0/1}$ | 0 | $V_{SUPHS}$ + 0.3 | V |
| 10 | Low-side driver LS[1:0] | $V_{PLS0/1}$ | 0 | 40 | V |
| 11 | EXTAL, XTAL[3] | $V_{ILV}$ | –0.3 | 2.16 | V |
| 12 | Instantaneous maximum current<br>Single pin limit for all digital I/O pins[4] | $I_D$ | –25 | +25 | mA |
| 13 | Instantaneous maximum current on PP2 / EVDD | $I_{EVDD}$ | -80 | +25 | mA |
| 14 | Instantaneous maximum current<br>Single pin limit for EXTAL, XTAL | $I_{DL}$ | –25 | +25 | mA |
| 15 | Storage temperature range | $T_{stg}$ | –65 | 155 | °C |

1 Beyond absolute maximum ratings device might be damaged.

2 VDDX and VDDA must be shorted

3 EXTAL and XTAL are shared with PE0 and PE1 5V GPIO's

4 All digital I/O pins are internally clamped to $V_{SSX}$ and $V_{DDX}$, or $V_{SSA}$ and $V_{DDA}$.

## A.1.5 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) and the Charged-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-3. ESD and Latch-up Test Conditions**

| Model | Spec | Description | Symbol | Value | Unit |
|---|---|---|---|---|---|
| Human Body | JESD22-A114 | Series Resistance | R | 1500 | Ω |
| | | Storage Capacitance | C | 100 | pF |
| | | Number of Pulse per pin<br>positive<br>negative | | <br>1<br>1 | |
| Charged-Device | JESD22-C101 | Series Resistance | R | 0 | Ω |
| | | Storage Capacitance | C | 4 | pF |
| Latch-up for 5V GPIO's | | Minimum Input Voltage Limit | | -2.5 | V |
| | | Maximum Input Voltage Limit | | +7.5 | V |
| Latch-up for LS/HS/HVI/V SENSE/LIN | | Minimum Input Voltage Limit | | -7 | V |
| | | Maximum Input Voltage Limit | | +21 | V |

**Table A-4. ESD Protection and Latch-up Characteristics**

| Num | C | Rating | Symbol | Min | Max | Unit |
|---|---|---|---|---|---|---|
| 1 | C | HBM: LIN to LGND | $V_{HBM}$ | +/- 6 | - | kV |
| 2 | C | HBM: VSENSE, HVI[3:0] to GND | | +/- 4 | | kV |
| 3 | C | HBM: HS1, HS2 to GND | | +/- 4 | | kV |
| 4 | C | HBM: LS0, LS1 to GND | | +/- 2 | | kV |
| 5 | C | HBM: Pin to Pin (all Pins LS0, LS1 excluded) | | +/- 2 | | kV |
| 6 | C | HBM: Pin to Pin (all Pins LS0, LS1 included) | | +/- 1.5 | | kV |
| 7 | C | CDM : Corner Pins | $V_{CDM}$ | +/-750 | - | V |
| 8 | C | CDM: All other Pins | $V_{CDM}$ | +/-500 | | V |
| 9 | C | Direct Contact Discharge IEC61000-4-2 with and with out 220pF capacitor (R=330, C=150pF):<br>LIN vs LGND | $V_{ESDIEC}$ | +/-6 | - | kV |

**Table A-4. ESD Protection and Latch-up Characteristics**

| | | | | | | |
|---|---|---|---|---|---|---|
| 10 | C | Latch-up Current of 5V GPIO's at T=125°C<br>positive<br>negative | $I_{LAT}$ | +100<br>-100 | - | mA |
| 11 | C | Latch-up Current for LS[1:0], HS[1:0], VSENSE, LIN & HVI[3:0] at T=125°C<br>positive<br>negative | $I_{LAT}$ | +100<br>-100 | - | mA |

## A.1.6 Operating Conditions

This section describes the operating conditions of the device. Unless otherwise noted those conditions apply to all the following data.

**NOTE**

Please refer to the temperature rating of the device with regards to the ambient temperature $T_A$ and the junction temperature $T_J$. For power dissipation calculations refer to Section A.1.7, "Power Dissipation and Thermal Characteristics".

**Table A-5. Operating Conditions**

| Num | Rating | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| 1 | Voltage regulator and LINPHY supply voltage | $V_{SUP}$/ $V_{LINUP}$ | 3.5 | 12 | 40[1] | V |
| 2 | High side driver supply voltage | $V_{SUPHS}$ | 7 | 12 | 40[1] | V |
| 3 | Voltage difference $V_{DDX}$ to $V_{DDA}$ | $\Delta_{VDDX}$ | -0.1 | — | 0.1 | V |
| 4 | Voltage difference $V_{SSX}$ to $V_{SSA}$ | $\Delta_{VSSX}$ | -0.1 | — | 0.1 | V |
| 5 | Oscillator | $f_{osc}$ | 4 | — | 20 | MHz |
| 6 | Bus frequency | $f_{bus}$ | see Footnote[2] | — | 25 | MHz |
| 7 | Operating junction temperature range<br>Operating ambient temperature range[3] | $T_J$<br>$T_A$ | –40<br>–40 | —<br>— | 150<br>125 | °C |

1 Normal operating range is 6V - 18V. Continous operation at 40V is not allowed. Only Transient Conditions (Load Dump) single pulse $t_{max}$<400ms

2 Minimum bus frequency for ADC module refer to Table C-1., "ATD Operating Characteristics and for Flash Module refer to Table M-1., "NVM Timing Characteristics

3 Please refer to Section A.1.7, "Power Dissipation and Thermal Characteristics" for more details about the relation between ambient temperature $T_A$ and device junction temperature $T_J$.

**NOTE**

Operation is guaranteed when powering down until low voltage reset assertion.

## A.1.7 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]

$\Theta_{JA}$ = Package Thermal Resistance, [°C/W]

The total power dissipation $P_D$ can be calculated from the equation below. Table A-6 below lists the power dissipation components . **Figure A-2.** gives an overview of the supply currents.

$$P_D = P_{INT} + P_{HS} + P_{LS} + P_{LIN} + P_{SENSE} + P_{HVI} - P_{EVDD} - P_{GPIO}$$

**Table A-6. Power Dissipation Components**

| Power Component | Description |
|---|---|
| $P_{INT} = V_{SUP}\ I_{SUP}$ | Internal Power for LQFP 48 Package with seperate VSUP and VSUPHS pins. |
| $P_{INT} = V_{SUP}\ (I_{SUP} - I_{PHS0/1})$ | Internal Power for LQFP 32 Package with single VSUP pin which is double bonded to VSUP pad and VSUPHS pad. |
| $P_{HS} = I_{PHS0/1}^2\ R_{DSONHS0/1}$ | Power dissipation of High-side drivers |
| $P_{LS} = I_{PLS0/1}^2\ R_{DSONLS0/1}$ | Power dissipation of Low-side drivers |
| $P_{LIN} = V_{LIN}\ I_{LIN}$ | Power dissipation of LINPHY |
| $P_{SENSE} = V_{SENSE}\ I_{SENSE}$ | Power dissipation of Battery Sensor |
| $P_{HVI} = V_{HVI}\ I_{HVI}$ | Power dissipation of High Voltage Inputs |
| $P_{EVDD} = V_{DDX}\ I_{EVDD}$ | Power dissipation of external load driven by EVDD. (see **Figure A-2.**) This component is included in $P_{INT}$ and is subtracted from overall MCU power dissipation $P_D$ |
| $P_{GPIO} = V_{I/O}\ I_{I/O}$ | Power dissipation of external load driven by GPIO Port.(see **Figure A-2.**) Assuming the load is connected between GPIO and ground. This power component is included in $P_{INT}$ and is subtracted from overall MCU power dissipation $P_D$ |

**Figure A-2. Supply Currents Overview**

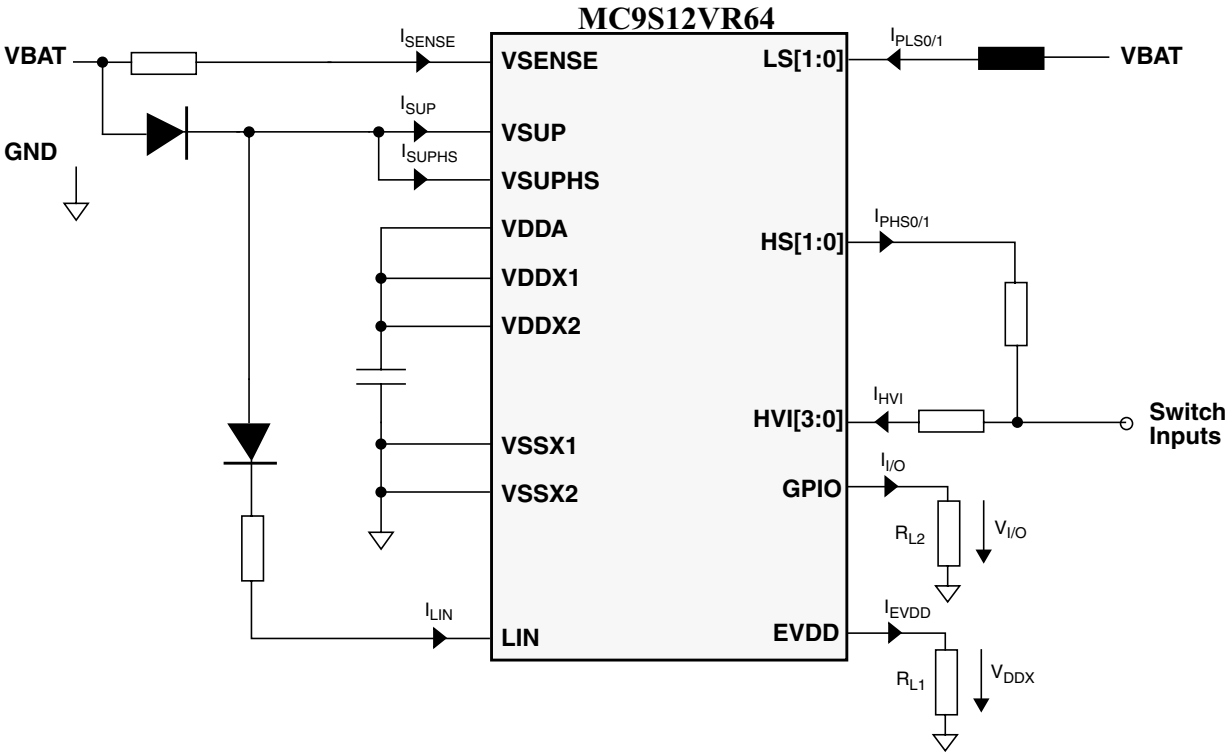


**Table A-7. Thermal Package Characteristics[1]**

| Num | C | Rating | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| LQFP 32 | | | | | | | |
| 1 | D | Thermal resistance LQFP 32, single sided PCB[1] | $\theta_{JA}$ | — | 85 | — | °C/W |
| 2 | D | Thermal resistance LQFP 32, double sided PCB with 2 internal planes[2] | $\theta_{JA}$ | — | 56 | — | °C/W |
| 3 | D | Junction to Board LQFP 32 | $\theta_{JB}$ | — | 33 | — | °C/W |
| 4 | D | Junction to Case LQFP 32[4] | $\theta_{JC}$ | — | 23 | — | °C/W |
| 5 | D | Junction to Case (Bottom) LQFP 32[5] | $\Psi_{JT}$ | — | 5 | — | °C/W |
| LQFP 48 | | | | | | | |
| 6 | D | Thermal resistance LQFP 48, single sided PCB[3] | $\theta_{JA}$ | — | 80 | — | °C/W |
| 7 | D | Thermal resistance LQFP 48, double sided PCB with 2 internal planes[4] | $\theta_{JA}$ | — | 56 | — | °C/W |
| 8 | D | Junction to Board LQFP 48 | $\theta_{JB}$ | — | 34 | — | °C/W |
| 9 | D | Junction to Case LQFP 48[4] | $\theta_{JC}$ | — | 23 | — | °C/W |
| 10 | D | Junction to Case (Bottom) LQFP 48[5] | $\Psi_{JT}$ | — | 5 | — | °C/W |

1 Junction to ambient thermal resistance, $\theta_{JA}$ was simulated to be equivalent to the JEDEC specification JESD51-2 in a horizontal configuration in natural convection.

2 Junction to ambient thermal resistance, $\theta_{JA}$ was simulated to be equivalent to the JEDEC specification JESD51-7 in a horizontal configuration in natural convection.

3 Junction to ambient thermal resistance, $\theta_{JA}$ was simulated to be equivalent to the JEDEC specification JESD51-2 in a horizontal configuration in natural convection.

4 Junction to ambient thermal resistance, $\theta_{JA}$ was simulated to be equivalent to the JEDEC specification JESD51-7 in a horizontal configuration in natural convection.

1. The values for thermal resistance are achieved by package simulations

## A.1.8 I/O Characteristics

This section describes the characteristics of I/O pins

**Table A-8. 5-V I/O Characteristics**

| Conditions are 4.5 V < $V_{DDX}$< 5.5 V junction temperature from –40°C to +150°C, unless otherwise noted<br>I/O Characteristics for all I/O pins except EXTAL, XTAL,TEST, PL, HS[1:0], LS[1:0], LIN and supply pins. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | P | Input high voltage | $V_{IH}$ | 0.65*$V_{DDX}$ | — | — | V |
| 2 | T | Input high voltage | $V_{IH}$ | — | — | $V_{DDX}$+0.3 | V |
| 3 | P | Input low voltage | $V_{IL}$ | — | — | 0.35*$V_{DDX}$ | V |
| 4 | T | Input low voltage | $V_{IL}$ | $V_{SSX}$–0.3 | — | — | V |
| 5 | C | Input hysteresis | $V_{HYS}$ | | 250 | — | mV |
| 6 | P | Input leakage current (pins in high impedance input mode)[1] $V_{in}$ = $V_{DDX}$ or $V_{SSX}$ | $I_{in}$ | -1 | | 1 | µA |
| 7 | P | Output high voltage (pins in output mode)<br>$I_{OH}$ = –4 mA for PP[5:3], PS, PT, PAD | $V_{OH}$ | $V_{DDX}$ – 0.8 | — | — | V |
| 8 | P | Output high voltage (pins in output mode) PP[1:0]<br>Partial Drive $I_{OH}$ = –2 mA<br>Full Drive IOH = –10mA | $V_{OH}$ | $V_{DDX}$ – 0.8 | | | V |
| 9 | P | Output high voltage (pins in output mode)<br>PP[2]/EVDD<br>Partial Drive $I_{OH}$ = –2 mA<br>Full Drive IOH = –20mA | $V_{OH}$ | $V_{DDX}$ – 0.8 | | | V |
| 10 | P | Output low voltage (pins in output mode)<br>$I_{OL}$ = +4mA for PP[5:3], PS, PT, PAD | $V_{OL}$ | — | — | 0.8 | V |
| 11 | P | Output low voltage (pins in output mode) PP[1:0]<br>Partial drive $I_{OL}$ = +2mA<br>Full drive $I_{OL}$ = +10mA | $V_{OL}$ | | | 0.8 | V |
| 12 | P | Output low voltage (pins in output mode) for<br>PP[2]/EVDD<br>Partial drive $I_{OL}$ = +2mA<br>Full drive $I_{OL}$ = +20mA | $V_{OL}$ | | | 0.8 | V |
| 13 | P | Over-current Detect Threshold PP[2]/EVDD | $I_{OCD}$ | 20 | | 55 | mA |
| 14 | P | Internal pull up resistor on $\overline{\text{RESET}}$ pin<br>$V_{IH}$ min > input voltage > $V_{IL}$ max | $R_{PUL}$ | 3.8 | 5 | 10.5 | kΩ |
| 15 | P | Internal pull up current<br>$V_{IH}$ min > input voltage > $V_{IL}$ max | $I_{PUL}$ | -10 | — | -130 | µA |
| 16 | P | Internal pull down current<br>$V_{IH}$ min > input voltage > $V_{IL}$ max | $I_{PDH}$ | 10 | — | 130 | µA |
| 17 | D | Input capacitance | $C_{in}$ | — | 7 | — | pF |
| 18 | T | Injection current[2]<br>Single pin limit<br>Total device Limit, sum of all injected currents | <br>$I_{ICS}$<br>$I_{ICP}$ | <br>–2.5<br>–25 | — | <br>2.5<br>25 | mA |

[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.1.3, “Current Injection” for more details

## A.1.9 Supply Currents

This section describes the current consumption characteristics of the device as well as the conditions for the measurements.

### A.1.9.1 Measurement Conditions

Current is measured on VSUP & VSUPHS pins. VDDX is connected to VDDA. It does not include the current to drive external loads. Unless otherwise noted the currents are measured in special single chip mode and the CPU code is executed from RAM. For Run and Wait current measurements PLL is on and the reference clock is the IRC1M trimmed to 1MHz. The bus frequency is 25MHz and the CPU frequency is 50MHz. Table A-9, Table A-10 and Table A-11 show the configuration of the CPMU module and the peripherals for Run, Wait and Stop current measurement.

**Table A-9. CPMU Configuration for Pseudo Stop Current Measurement**

| CPMU REGISTER | Bit settings/Conditions |
|---|---|
| CPMUCLKS | PLLSEL=0, PSTP=1, CSAD=0<br>PRE=PCE=RTIOSCSEL=COPOSCSEL=1 |
| CPMUOSC | OSCE=1, External Square wave on EXTAL $f_{EXTAL}$=4MHz,<br>$V_{IH}$= 1.8V, $V_{IL}$=0V |
| CPMURTI | RTDEC=0, RTR[6:4]=111, RTR[3:0]=1111; |
| CPMUCOP | WCOP=1, CR[2:0]=111 |

**Table A-10. CPMU Configuration for Run/Wait and Full Stop Current Measurement**

| CPMU REGISTER | Bit settings/Conditions |
|---|---|
| CPMUSYNR | VCOFRQ[1:0]=01,SYNDIV[5:0] = 24 |
| CPMUPOSTDIV | POSTDIV[4:0]=0 |
| CPMUCLKS | PLLSEL=1 |
| CPMUOSC | OSCE=0,<br>Reference clock for PLL is $f_{ref}$=$f_{irc1m}$ trimmed to 1MHz |
| API settings for STOP current measurement | |

**Table A-10. CPMU Configuration for Run/Wait and Full Stop Current Measurement**

| CPMU REGISTER | Bit settings/Conditions |
|---|---|
| CPMUAPICTL | APIEA=0, APIFE=1, APIE=0 |
| CPMUAPITR | trimmed to >=10Khz |
| CPMUAPIRH/RL | set to $FFFF |

**Table A-11. Peripheral Configurations for Run & Wait Current Measurement**

| Peripheral | Configuration |
|---|---|
| SCI | continuously transmit data (0x55) at speed of 19200 baud |
| SPI | configured to master mode, continuously transmit data (0x55) at 1Mbit/s |
| PWM | configured to toggle its pins at the rate of 40kHz |
| ADC | the peripheral is configured to operate at its maximum specified frequency and to continuously convert voltages on all input channels in sequence. |
| DBG | the module is enabled and the comparators are configured to trigger in outside range.The range covers all the code executed by the core. |
| TIM | the peripheral is configured to output compare mode, pulse accumulator and modulus counter enabled. |
| COP & RTI | enabled |
| HSDRV 1 & 2 | module is enabled but output driver disabled |
| LSDRV 1 & 2 | module is enabled but output driver disabled |
| BATS | enabled |
| LINPHY | connected to SCI and continuously transmit data (0x55) at speed of 19200 baud |

**Table A-12. Run and Wait Current Characteristics**

| Conditions are: $V_{SUP}$=$V_{SUPHS}$=18V, $T_A$=105°C, see Table A-10 and Table A-9 | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | P | Run Current | $I_{SUPR}$ | | 15 | 22 | mA |
| 2 | P | Wait Current | $I_{SUPW}$ | | 10 | 15 | mA |

### Table A-13. Stop Current Characteristics

| Conditions are: $V_{SUP}$=$V_{SUPHS}$=12V API see CPMU Configuration for Pseudo Stop Current MeasurementTable A-9. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| Stop Current all modules off | | | | | | | |
| 1 | P | $T_A = T_J = -40°C$[1] | $I_{SUPS}$ | | 29 | 60 | µA |
| 2 | P | $T_A = T_J = 150°C$[1] | $I_{SUPS}$ | | 140 | 600 | µA |
| 3 | C | $T_A = T_J = 25°C$[1] | $I_{SUPS}$ | | 33 | 65 | µA |
| 4 | C | $T_A = T_J = 105°C$[1] | $I_{SUPS}$ | | 55 | 90 | µA |
| Stop Current API enabled & LINPHY in standby (see 15.4.3.4 Standby Mode with wake-up feature) | | | | | | | |
| 5 | C | $T_A = T_J = 25°C$[1] | $I_{SUPS}$ | | 50 | 80 | µA |

[1] If MCU is in STOP long enough then $T_A = T_J$ . Die self heating due to stop current can be ignored.

### Table A-14. Pseudo Stop Current Characteristics

| Conditions are: $V_{SUP}$=$V_{SUPHS}$=12V, API see CPMU Configuration for Pseudo Stop Current MeasurementTable A-9., COP & RTI enabled | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | C | $T_A$= 25°C | $I_{SUPPS}$ | | 358 | 480 | µA |

# Appendix B
# VREG Electrical Specifications

**Table B-1. Voltage Regulator Electrical Characteristics**

| -40°C <= $T_J$ <= 150°C unless noted otherwise, VDDA and VDDX must be shorted on the application board. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Characteristic** | **Symbol** | **Min** | **Typical** | **Max** | **Unit** |
| 1 | P | Input Voltages | $V_{SUP}$ | 3.5 | — | 40 | V |
| 4 | P | Output Voltage VDDX<br>Full Performance Mode $V_{SUP}$ > 6V<br>Full Performance Mode 5.5V < $V_{SUP}$ <=6V<br>Full Performance Mode 3.5V <= $V_{SUP}$ <=5.5V<br>Reduced Performance Mode (stopmode)<br>$V_{SUP}$ > =3.5V | $V_{DDX}$ | <br>4.75<br>4.50<br>3.13<br><br>2.5 | <br>5.0<br>5.0<br>—<br><br>5.5 | <br>5.25<br>5.25<br>5.25<br><br>5.75 | <br>V<br>V<br>V<br><br>V |
| 5 | P | Load Current VDDX[1] [2,3]<br>Full Performance Mode $V_{SUP}$ > 6V<br>Full Performance Mode 3.5V <= $V_{SUP}$ <=6V<br>Reduced Performance Mode (stopmode) | $I_{DDX}$ | <br>0<br>0<br>0 | <br>—<br>—<br>— | <br>70<br>25<br>5 | <br>mA<br>mA<br>mA |
| 6 | P | Low Voltage Interrupt Assert Level[4]<br>Low Voltage Interrupt Deassert Level | $V_{LVIA}$<br>$V_{LVID}$ | 4.04<br>4.19 | 4.23<br>4.38 | 4.40<br>4.49 | V<br>V |
| 7a | P | VDDX Low Voltage Reset deassert[5] | $V_{LVRXD}$ | — | — | 3.13 | V |
| 7b | P | VDDX Low Voltage Reset assert | $V_{LVRXA}$ | 2.97 | 3.02 | — | V |
| 8 | C | Trimmed ACLK output frequency | $f_{ACLK}$ | — | 20 | — | kHz |
| 9 | C | Trimmed ACLK internal clock $\Delta f / f_{nominal}$ [6] | $df_{ACLK}$ | - 6% | — | + 6% | — |
| 10 | D | The first period after enabling the counter by APIFE might be reduced by API start up delay | $t_{sdel}$ | — | — | 100 | µs |
| 11 | T | Temperature Sensor Slope | $dV_{HT}$ | 5.05 | 5.25 | 5.45 | mV/°C |
| 12 | T | Temperature Sensor Output Voltage ($T_J$=150°C) | $V_{HT}$ | — | 2.4 | — | V |
| 13 | T | High Temperature Interrupt Assert[7]<br>High Temperature Interrupt Deassert | $T_{HTIA}$<br>$T_{HTID}$ | 120<br>110 | 132<br>122 | 144<br>134 | °C<br>°C |

**Table B-1. Voltage Regulator Electrical Characteristics**

| -40°C <= $T_J$ <= 150°C unless noted otherwise, VDDA and VDDX must be shorted on the application board. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Characteristic** | **Symbol** | **Min** | **Typical** | **Max** | **Unit** |
| 14 | P | Bandgap output voltage | $V_{BG}$ | 1.13 | 1.22 | 1.32 | V |
| 15 | C | $V_{BG}$ Voltage Distribution over input voltage $V_{SUP}$<br>3.5V <= $V_{SUP}$ <= 18V, $T_A$ = 125°C | $\Delta_{VBGV}$ | -5 | | 5 | mV |
| 16 | C | $V_{BG}$ Voltage Distribution over ambient temperature $T_A$<br>$V_{SUP}$ = 12V, -40°C <= $T_A$ <= 125°C | $\Delta_{VBGV}$ | -20 | | 20 | mV |
| 17 | D | Recovery time from STOP | $t_{STP_REC}$ | — | 23 | — | µs |

[1]For the given maximum load currents and $V_{SUP}$ input voltages, the MCU will stay out of reset.
[2]Please note that the core current is derived from VDDX
[3]further limitation may apply due to maximum allowable $T_J$
[4]LVI is monitored on the VDDA supply domain
[5]LVRX is monitored on the VDDX supply domain only active during full performance mode. During reduced performance mode (stopmode) voltage supervision is solely performed by the POR block monitoring core VDD.
[6]The ACLK trimming must be set that the minimum period equals to 0.2ms
[7]VREGHTTR=$88

**NOTE**

The LVR monitors the voltages VDD, VDDF and VDDX. If the voltage drops on these supplies to a level which could prohibit the correct function (e.g. code execution) of the microcontroller, the LVR triggers.

**Table B-2. Recommended Capacitor Values**

| **Num** | **Characteristic** | **Symbol** | **Typical[1]** | **Unit** |
|---|---|---|---|---|
| 1 | VDDX capacitor[2] | $C_{VDDX}$ | 100-220 | nF |
| 2 | VDDA capacitor[3] | $C_{VDDA}$ | 100-220 | nF |
| 3 | Stability capacitor[4,5] | $C_{VDD5}$ | 4.7 or 10 | µF |

[1]Values are nominal component values.
[2]X7R ceramics
[3]X7R ceramics
[4]Can be placed anywhere on the 5V supply node (VDDA, VDDX)
[5]4.7µF X7R ceramics or 10µF tantalum

# Appendix C
# ATD Electrical Specifications

This section describes the characteristics of the analog-to-digital converter.

## C.1 ATD Operating Characteristics

The Table C-1 and Table C-2 show 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 C-1. ATD Operating Characteristics**

| Supply voltage 3.13 V < $V_{DDA}$ < 5.5 V, -40°C < $T_J$ < 150°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | C | ATD Clock Frequency (derived from bus clock via the prescaler bus) | $f_{ATDCLk}$ | 0.25 | | 8.0 | MHz |
| 2 | D | ATD Conversion Period[1]<br>10 bit resolution:<br>8 bit resolution: | <br>$N_{CONV10}$<br>$N_{CONV8}$ | <br>19<br>17 | | <br>41<br>39 | ATD clock Cycles |

[1] The minimum time assumes a sample time of 4 ATD clock cycles. The maximum time assumes a sample time of 24 ATD clock cycles and the discharge feature (SMP_DIS) enabled, which adds 2 ATD clock cycles.

## C.2 Factors Influencing Accuracy

Source resistance, source capacitance and current injection have an influence on the accuracy of the ATD. A further factor is that PortAD pins that are configured as output drivers switching.

### C.2.1 Port AD Output Drivers Switching

PortAD output drivers switching can adversely affect the ATD accuracy whilst converting the analog voltage on other PortAD pins because the output drivers are supplied from the VDDA/VSSA ATD supply pins. Although internal design measures are implemented to minimize the affect of output driver noise, it is recommended to configure PortAD pins as outputs only for low frequency, low load outputs. The impact on ATD accuracy is load dependent and not specified. The values specified are valid under condition that no PortAD output drivers switch during conversion.

## C.2.2 Source Resistance

Due to the input pin leakage current as specified 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 (10-bit resolution) 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 of up to 10Kohm are allowed.

## C.2.3 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 (10-bit resilution), then the external filter capacitor, $C_f \geq 1024 * (C_{INS}–C_{INN})$.

## C.2.4 Current Injection

There are two cases to consider.

1. A current is injected into the channel being converted. The channel being stressed has conversion values of $3FF (in 10-bit mode) for analog inputs greater than $V_{RH}$ and $000 for values less than $V_{RL}$ unless the current is higher than specified as disruptive condition.
2. 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 C-2. ATD Electrical Characteristics**

| Supply voltage 3.13 V < $V_{DDA}$ < 5.5 V, -40°C < $T_J$ < 150°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | C | Max input source resistance[1] | $R_S$ | — | — | 1 | kΩ |
| 2 | D | Total input capacitance Non sampling<br>Total input capacitance Sampling | $C_{INN}$<br>$C_{INS}$ | —<br>— | —<br>— | 10<br>16 | pF |
| 3 | D | Input internal Resistance | $R_{INA}$ | - | 5 | 15 | kΩ |
| 4 | C | Disruptive analog input current | $I_{NA}$ | -2.5 | — | 2.5 | mA |
| 5 | C | Coupling ratio positive current injection | $K_p$ | — | — | 1E-4 | A/A |
| 6 | C | Coupling ratio negative current injection | $K_n$ | — | — | 5E-3 | A/A |

[1] 1 Refer to C.2.2 for further information concerning source resistance

## C.3 ATD Accuracy

**Table C-3.** and **Table C-4.** specifies the ATD conversion performance excluding any errors due to current injection, input capacitance and source resistance.

### C.3.1 ATD Accuracy Definitions

For the following definitions see also Figure C-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:

$$INL(n) = \sum_{i=1}^{n} DNL(i) = \frac{V_n - V_0}{1LSB} - n$$

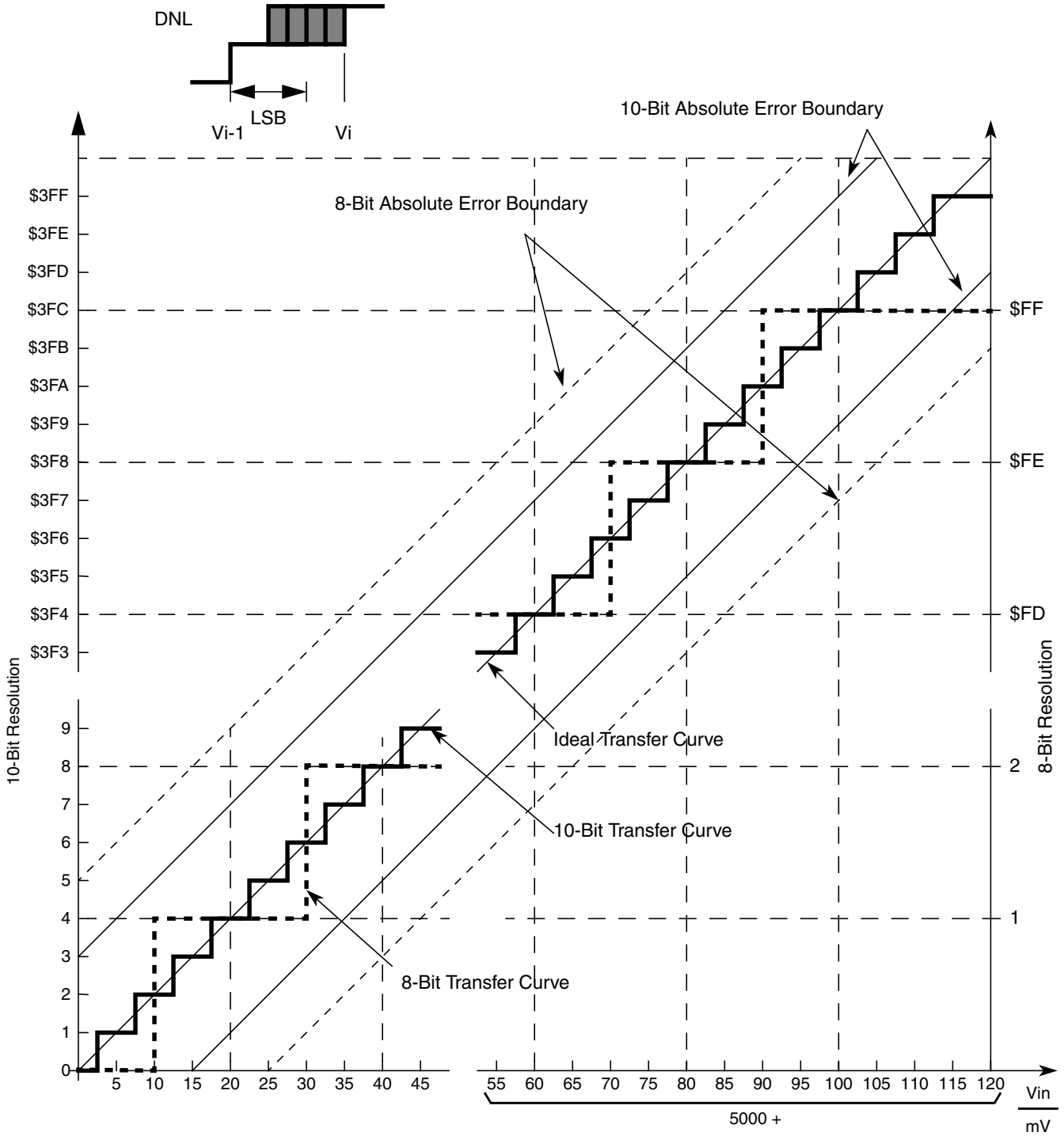



**Figure C-1. ATD Accuracy Definitions**

## NOTE

Figure A-1 shows only definitions, for specification values refer to Table A-3 and Table A-4.

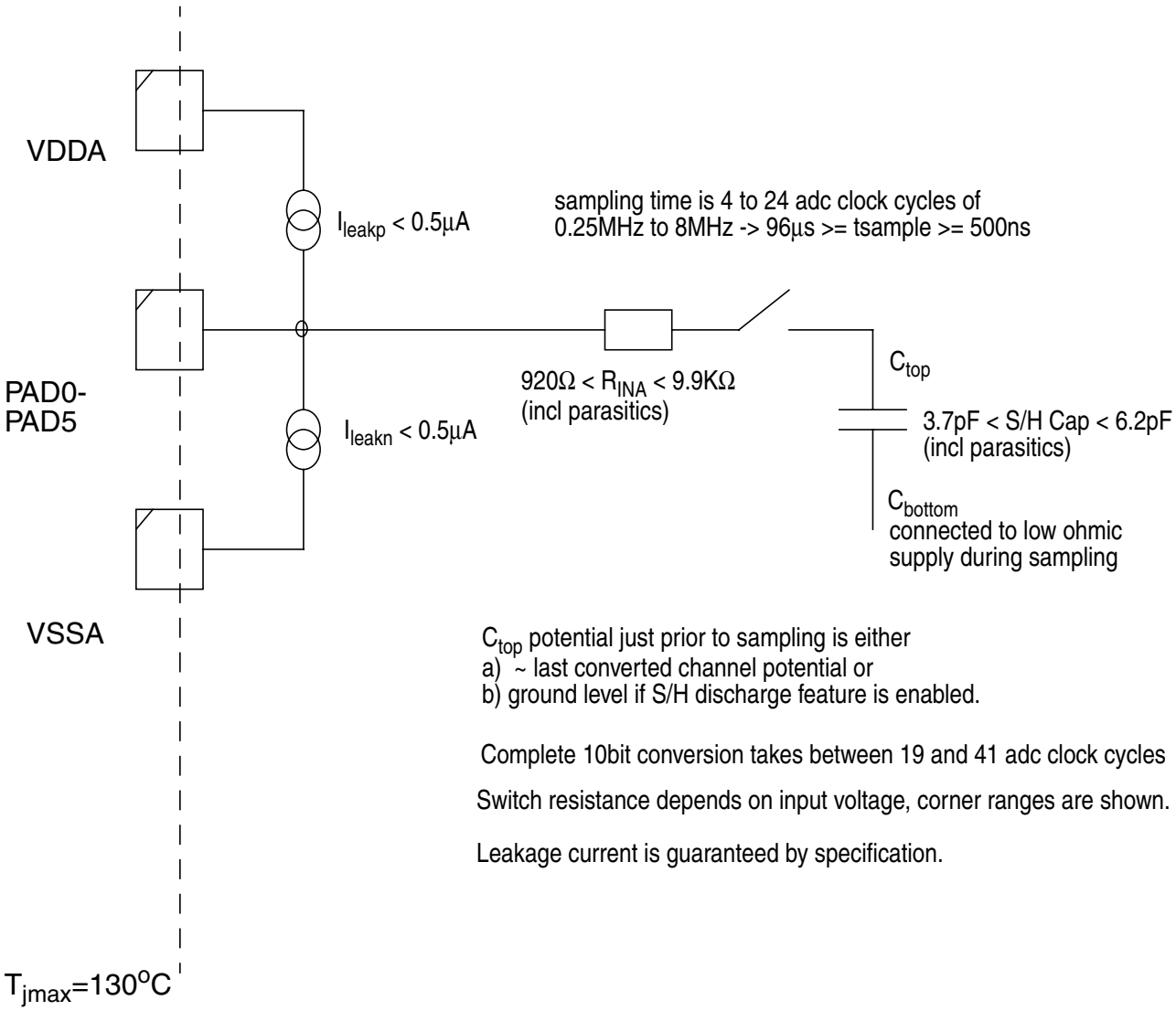
**Table C-3. ATD Conversion Performance 5V range**

Supply voltage $V_{DDA}$ =5.12 V, -40°C < $T_J$ < 150°C. $V_{REF}$ = $V_{RH}$ - $V_{RL}$ = 5.12V. $f_{ATDCLK}$ = 8.0MHz
The values are tested to be valid with no PortAD output drivers switching simultaneous with conversions.

| Num | C | Rating | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|---|
| 1 | P | Resolution | 10-Bit | LSB | | 5 | | mV |
| 2 | P | Differential Nonlinearity | 10-Bit | DNL | -1 | ±0.5 | 1 | counts |
| 3 | P | Integral Nonlinearity | 10-Bit | INL | -2 | ±1 | 2 | counts |
| 4 | P | Absolute Error | 10-Bit | AE | -3 | ±2 | 3 | counts |
| 5 | C | Resolution | 8-Bit | LSB | | 20 | | mV |
| 6 | C | Differential Nonlinearity | 8-Bit | DNL | -0.5 | ±0.3 | 0.5 | counts |
| 7 | C | Integral Nonlinearity | 8-Bit | INL | -1 | ±0.5 | 1 | counts |
| 8 | C | Absolute Error | 8-Bit | AE | -1.5 | ±1 | 1.5 | counts |

## C.3.2 ATD Analog Input Parasitics

**Figure C-2. ATD Input Parasitics**



# Appendix D
# HSDRV Electrical Specifications

This section provides electrical parametric and ratings for the HSDRV.

## D.1 Operating Characteristics

**Table D-1. Operating Characteristics - HSDRV**

| -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | P | High Voltage Supply for the high-side drivers. | $V_{SUPHS}$ | 7 | – | 42 | V |
| 2 | T | VSUP_HS in case of being connected to VDDX | $V_{SUPHS_X}$ | 4.5 | – | 5.5 | V |

1 $T_J$: Junction Temperature

2 $T_A$: Ambient Temperature

## D.2 Static Characteristics

**Table D-2. Static Characteristics - HSDRV**

| Characteristics noted under conditions 7V ≤ VSUPHS ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | P | Output Drain-to-Source On Resistance<br>$T_J$ = 150˚C, $I_{PHS0/1}$ = 50 mA | $R_{DS(ON)}$ | – | – | 18.0 | Ω |
| 2 | P | Over-current Threshold.<br>The threshold is valid for each HS-driver output.<br><br>**Note:**<br>**The high-side driver is NOT intended to switch capacitive loads. A significant capacitive load on HS0/1 would induce a current when the high-side driver gate is turned on. This current will be sensed by the over-current circuitry and eventually lead to an immediate over-current shut down.** | $I_{OCTHSX}$ | 100 | 120 | 150 | mA |
| 3 | T | Nominal Current for continuous operation.<br>This value is valid for each HS-driver output. | $I_{NOMHSX}$ | – | – | 50 | mA |

### Table D-2. Static Characteristics - HSDRV

| Characteristics noted under conditions 7V ≤ VSUPHS ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 4 | T | Leakage Current -40˚C < $T_J$ < 80˚C<br>Leakage Current -40˚C < $T_J$< 150˚C<br><br>(0V < $V_{PHS0/1}$ < $V_{SUP_HS}$) | $I_{LEAK_L}$<br>$I_{LEAK_H}$ | -1<br>-10 | –<br>– | 1<br>10 | µ A<br>µ A |

[1] $T_J$: Junction Temperature

[2] $T_A$: Ambient Temperature

### Table D-3. Static Characteristics - HSDRV - VSUP_HS connected to VDDX

| Characteristics noted under conditions 4.5V ≤ VSUPHS ≤ 5.5V , -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_J$ = 25˚C[1] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | T | Output Drain-to-Source On Resistance<br>$I_{PHS0/1}$ = -50 mA | $R_{DS(ON)}$ | – | 18 | – | Ω |
| 2 | T | Over-current Threshold.<br>The threshold is valid for each HS-driver output.<br><br>**Note:**<br>**The high-side driver is NOT intended to switch capacitive loads. A significant capacitive load on HS0/1 would induce a current when the high-side driver gate is turned on. This current will be sensed by the over-current circuitry and eventually lead to an immediate over-current shut down. In such cases of capacitive loads you can leverage the over current masking feature or handle it by software.** | $I_{OCTHSX}$ | 100 | 120 | 150 | mA |
| 3 | T | Nominal Current for continuous operation.<br>This value is valid for each HS-driver output. | $I_{NOMHSX}$ | – | – | 50 | mA |
| 4 | T | Leakage Current -40˚C < $T_J$ < 80˚C<br>Leakage Current -40˚C < $T_J$< 150˚C<br>(0V < $V_{HS0/1}$ < $V_{SUP_HS}$) | $I_{LEAK_L}$<br>$I_{LEAK_H}$ | -1<br>-10 | –<br>– | 1<br>10 | µ A<br>µ A |

[1] $T_J$: Junction Temperature

## D.3 Dynamic Characteristics

**Table D-4. Dynamic Characteristics - HSDRV**

| Characteristics noted under conditions 7V ≤ VSUPHS ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | T | High-Side Driver Operating Frequency | $f_{HS}$ | – | – | 10 | kHz |
| 2 | D | Settling time after the high-side driver is enabled (write HSEx Bits) | $t_{HS_settling}$ | 1 | – | – | µs |
| 3 | T | Over-Current Shutdown Masking Time(IRC trimmed at 1 MHz) | $t_{HSOCM}$ | 10 | - | 11 | µs |

1 $T_J$: Junction Temperature

2 $T_A$: Ambient Temperature

**Table D-5. Dynamic Characteristics - HSDRV - VSUP_HS connected to VDDX**

| Characteristics noted under conditions 4.5V ≤ VSUPHS ≤ 5.5V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_J$ = 25˚C[1] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | T | High-Side Driver Operating Frequency | $f_{HS}$ | – | – | 10 | kHz |
| 2 | D | Settling time after the high-side driver is enabled (write HSEx Bits) | $t_{HS_settling}$ | 1 | – | – | µs |
| 3 | T | Over-Current Shutdown Masking Time(IRC trimmed at 1 MHz) | $t_{HSOCM}$ | 10 | - | 11 | µs |

1 $T_J$: Junction Temperature

# Appendix E
# PLL Electrical Specifications

## E.1 Reset, Oscillator and PLL

### E.1.1 Phase Locked Loop

#### E.1.1.1 Jitter Information

With each transition of the feedback clock, the deviation from the reference clock is measured and the input voltage to the VCO is adjusted accordingly.The adjustment is done continuously with no abrupt changes in the VCOCLK 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 E-1.**.

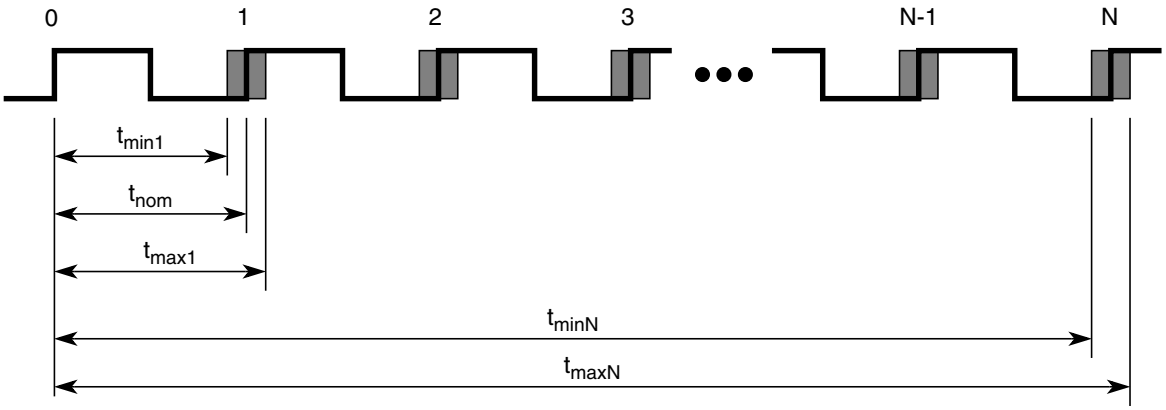



**Figure E-1. Jitter Definitions**

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:

$$J(N) = \max\left(\left|1 - \frac{t_{max}(N)}{N \cdot t_{nom}}\right|, \left|1 - \frac{t_{min}(N)}{N \cdot t_{nom}}\right|\right)$$

The following equation is a good fit for the maximum jitter:

$$J(N) = \frac{j_1}{\sqrt{N(POSTDIV + 1)}}$$

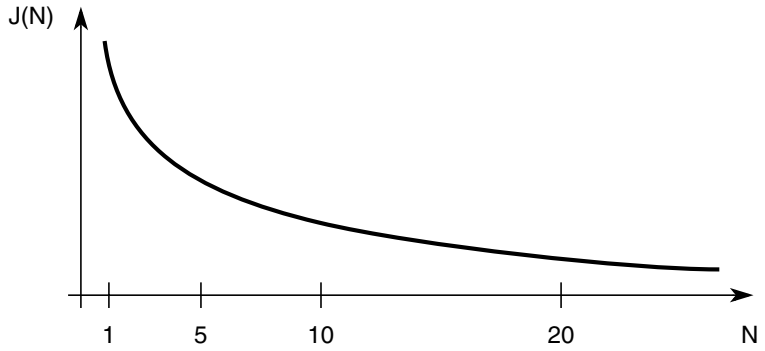



**Figure E-2. Maximum Bus Clock Jitter Approximation (N=Number of Bus Cycles)**

**NOTE**

On timers and serial modules a prescaler will eliminate the effect of the jitter to a large extent.

**Table E-1. PLL Characteristics**

| Conditions are 4.5V < $V_{DDX}$ < 5.5V, -40°C <= $T_J$ <= 150°C , unless otherwise noted | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | D | VCO frequency during system reset | $f_{VCORST}$ | 8 | | 32 | MHz |
| 2 | C | VCO locking range | $f_{VCO}$ | 32 | | 50 | MHz |
| 3 | C | Reference Clock | $f_{REF}$ | 1 | | | MHz |
| 4 | D | Lock Detection | $\|\Delta_{Lock}\|$ | 0 | | 1.5 | %[1] |
| 5 | D | Un-Lock Detection | $\|\Delta_{unl}\|$ | 0.5 | | 2.5 | %[1] |
| 6 | C | Time to lock | $t_{lock}$ | | | $150 + 256/f_{REF}$ | µs |
| 7 | C | Jitter fit parameter 1[2]<br>IRC as reference clock source | $j_1$ | | | 1.4 | % |
| 8 | C | Jitter fit parameter 1[3]<br>XOSCLCP as reference clock source | $j_1$ | | | 1.0 | % |

1 % deviation from target frequency

2 $f_{REF}$ = 1MHz (IRC), $f_{BUS}$ = 25MHz equivalent $f_{PLL}$=50MHz, CPMUSYNR=0x58, CPMUREFDIV=0x00,CPMUPOSTDIV=0x00

3 $f_{REF}$ = 4MHz (XOSCLCP), $f_{BUS}$ = 24MHz equivalent $f_{PLL}$=48MHz, CPMUSYNR=0x05,CPMUREFDIV=0x40,CPMUPOSTDIV=0x00

# Appendix F
# IRC Electrical Specifications

**Table F-1. IRC electrical characteristics**

| Num | C | Rating | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| 1 | P | Junction Temperature - 40 to 150 Celsius<br>Internal Reference Frequency, factory trimmed | $f_{IRC1M_TRIM}$ | 0.987 | 1 | 1.013 | MHz |

# Appendix G
# LINPHY Electrical Specifications

## G.1 Maximum Ratings

**Table G-1. Maximum ratings of the LINPHY**

| Characteristics noted under conditions 7V<= $V_{LINSUP}$ <=18V unless otherwise noted[1] [2]. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C under nominal conditions unless otherwise noted. | | | | | |
|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Value** | **Unit** |
| 1 | P | DC voltage on LIN | $V_{LIN}$ | -32 to +42 | V |
| 2 | D | Continuous current on LIN | $I_{LIN}$ | 200[3] | mA |

[1]For 3.5V<= $V_{LINSUP}$ <7V, the LINPHY is still working but with degraded parametrics.
[2]The $V_{LINSUP}$ voltage is provided by the VLINSUP supply. This supply mapping is described in device level documentation.
[3]The current on the LIN pin is internally limited. Therefore, it should not be possible to reach the 200mA anyway.

## G.2 Static Electrical Characteristics

**Table G-2. Static electrical characteristics of the LINPHY**

| Characteristics noted under conditions 7V <= $V_{LINSUP}$ <= 18V unless otherwise noted[1] [2]. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | C | $V_{LINSUP}$ range for LIN compliant electrical characteristics | $V_{LINSUP_LIN}$ | 7[1] | 12 | 18 | V |
| 2 | | Current consumption, recessive state ($V_{LINSUP}$ = 12V, $V_{DDX}$ = 5V, $V_{DDA}$ = 5V, $V_{DD}$ = 1.8V, Tj = 25 C) | | | | | |
| | D | on chip VLINSUP | | | 10 | | µA |
| | D | on VDDX | | | 634 | | µA |
| | D | on VDDA | | | 28 | | µA |
| | D | on VDD | | | 0 | | µA |
| 3 | | Current consumption, dominant state ($V_{LINSUP}$=12V, $V_{DDX}$ = 5V, $V_{DDA}$ = 5V, $V_{DD}$ = 1.8V, Tj = 25 C) | | | | | |
| | D | on chip VLINSUP | | | 328 | | µA |
| | D | on VDDX | | | 1100 | | µA |
| | D | on VDDA | | | 28 | | µA |
| | D | on VDD | | | 0 | | µA |
| 4 | | Current consumption, standby ($V_{LINSUP}$ = 12V, $V_{DDX}$ = 5V, $V_{DDA}$ = 5V, $V_{DD}$ = 1.8V, Tj = 25 C) | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | D | on chip VLINSUP<br>Internal pull-up disabled<br>Internal pull-up enabled | | | 7.1<br>10 | | µA |
| | D | on VDDX<br>Internal pull-up disabled<br>Internal pull-up enabled | | | 3.3<br>6 | | µA |
| | D | on VDDA | | | 0 | | µA |
| | D | on VDD | | | 0 | | µA |
| 5a | P | Current limitation into the LIN pin in dominant state[3], for $V_{LIN} = V_{LINSUP_LIN_MAX}$, -40 C < TJ < 150 C | $I_{LIN_LIM}$ | 40 | | 200 | mA |
| 6a | P | Input leakage current in dominant state, driver off, internal pull-up on ($V_{LIN}$ = 0V, $V_{LINSUP}$ = 12V), -40 C < TJ < 150 C | $I_{LIN_PAS_dom}$ | -1 | | | mA |
| 7a | P | Input leakage current in recessive state, driver off, -40 C < TJ < 150 C (8V<$V_{LINSUP}$<18V, 8V<$V_{LIN}$<18V, $V_{LIN}$ > $V_{LINSUP}$) | $I_{LIN_PAS_rec}$ | | | 20 | µA |
| 8a | T | Input leakage current when ground disconnected -40 C < TJ < 150 C ($GND_{Device}$ = $V_{LINSUP}$, 0V<$V_{LIN}$<18V, $V_{LINSUP}$ = 12V) | $I_{LIN_NO_GND}$ | -1 | | 1 | mA |
| 9a | T | Input leakage current when battery disconnected -40 C < TJ < 150 C ($V_{LINSUP}$ = $GND_{Device}$, 0<$V_{LIN}$<18V) | $I_{LIN_NO_BAT}$ | | | 30 | µA |
| 10a | P | Receiver dominant state, -40 C < TJ < 150 C | $V_{LINdom}$ | | | 0.4 | $V_{LINSUP}$ |
| 11a | P | Receiver recessive state, -40 C < TJ < 150 C | $V_{LINrec}$ | 0.6 | | | $V_{LINSUP}$ |
| 12a | P | $V_{LIN_CNT}$ =($V_{th_dom}$+ $V_{th_rec}$)/2 , -40 C < TJ < 150 C | $V_{LIN_CNT}$ | 0.475 | 0.5 | 0.525 | $V_{LINSUP}$ |
| 13a | P | $V_{HYS}$ = $V_{th_rec}$ -$V_{th_dom}$ , -40 C < TJ < 150 C | $V_{HYS}$ | | | 0.175 | $V_{LINSUP}$ |
| 14 | - | Maximum capacitance allowed on slave node including external components | $C_{slave}$ | | 220 | 250 | pF |
| 15a | T<br>D | Capacitance of the LIN pin, -40 C < TJ < 150 Ç<br>0V<=$V_{LIN}$<=$V_{LINSUP_LIN_MAX}$ | $C_{LIN}$ | | 20 | <br>45 | pF<br>pF |
| 16a | P | Internal pull-up (slave), -40 C < TJ < 150 C | $R_{slave}$ | 27 | 34 | 40 | kΩ |

[1]For 3.5V<= $V_{LINSUP}$ <7V, the LINPHY is still working but with degraded parametrics.
[2]The $V_{LINSUP}$ voltage is provided by the VLINSUP supply. This supply mapping is described in device level documentation.
[3]At temperatures above 25C the current may be naturally limited by the driver, in this case the limitation circuit is not engaged and the flag is not set.

# G.3 Dynamic Electrical Characteristics

**Table G-3. Dynamic electrical characteristics of the LINPHY**

| Characteristics noted under conditions 7V<= $V_{LINSUP}$ <=18V unless otherwise noted[1]. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1a | P | Minimum duration of wake-up pulse generating a wake-up interrupt<br>-40 C < TJ < 150 C | $t_{WUFR}$ | 56 | 72 | 120 | µs |
| 2a | P | TxD-dominant timeout, -40 C < TJ < 150C | $t_{DTLIM}$ | 16388 | | 16389 | µs |
| 3a | P | Propagation delay of receiver, -40 C < TJ < 150C | $t_{rx_pd}$ | | | 6 | µs |
| 4a | P | Symmetry of receiver propagation delay rising edge w.r.t. falling edge<br>-40 C < TJ < 150C | $t_{rx_sym}$ | -2 | | 2 | µs |
| LIN PHYSICAL LAYER: DRIVER CHARACTERISTICS FOR NOMINAL SLEW RATE - 20.0KBIT/S | | | | | | | |
| 5 | T | Rising/falling edge time (min to max / max to min) | $t_{rise}$ | | 6.5 | | µs |
| 6a | T | Over-current masking window (IRC trimmed at 1MHz)<br>-40 C < TJ < 150C | $t_{OCLIM}$ | 15 | | 16 | µs |
| 7a | P | Duty cycle 1<br>$T_{HRec(max)}$ = 0.744 x $V_{LINSUP}$<br>$T_{HDom(max)}$ = 0.581 x $V_{LINSUP}$<br>$V_{LINSUP}$ = 7.0V...18V<br>$t_{Bit}$ = 50us<br>D1 = $t_{Bus_rec(min)}$ / (2 x $t_{Bit}$)<br>-40 C < TJ < 150C | D1 | 0.396 | | | |
| 8a | P | Duty cycle 2<br>$T_{HRec(min)}$ = 0.422 x $V_{LINSUP}$<br>$T_{HDom(min)}$ = 0.284 x $V_{LINSUP}$<br>$V_{LINSUP}$ = 7.6V...18V<br>$t_{Bit}$ = 50us<br>D2 = $t_{Bus_rec(max)}$ / (2 x $t_{Bit}$)<br>-40 C < TJ < 150C | D2 | | | 0.581 | |
| LIN PHYSICAL LAYER: DRIVER CHARACTERISTICS FOR SLOW SLEW RATE - 10.4KBIT/S | | | | | | | |
| 9 | T | Rising/falling edge time (min to max / max to min) | $t_{rise}$ | | 13 | | µs |
| 10a | T | Over-current masking window (IRC trimmed at 1MHz)<br>-40 C < TJ < 150C | $t_{OCLIM}$ | 31 | | 32 | µs |
| 11a | P | Duty cycle 3<br>$T_{HRec(max)}$ = 0.778 x $V_{LINSUP}$<br>$T_{HDom(max)}$ = 0.616 x $V_{LINSUP}$<br>$V_{LINSUP}$ = 7.0V...18V<br>$t_{Bit}$ = 96us<br>D3 = $t_{Bus_rec(min)}$ / (2 x $t_{Bit}$)<br>-40 C < TJ < 150C | D3 | 0.417 | | | |

| Characteristics noted under conditions 7V<= $V_{LINSUP}$ <=18V unless otherwise noted[1]. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 12a | P | Duty cycle 4<br>$T_{HRec(min)}$ = 0.389 x $V_{LINSUP}$<br>$T_{HDom(min)}$ = 0.251 x $V_{LINSUP}$<br>$V_{LINSUP}$ = 7.6V...18V<br>$t_{Bit}$ = 96us<br>D4 = $t_{Bus_rec(max)}$ / (2 x $t_{Bit}$)<br>-40 C < TJ < 150C | D4 | | | 0.590 | |
| LIN PHYSICAL LAYER: DRIVER CHARACTERISTICS FOR FAST MODE SLEW RATE - 100KBIT/S UP TO 250KBIT/S | | | | | | | |
| 13 | T | Rising/falling edge time (min to max / max to min) | $t_{rise}$ | | 0.5 | | µs |
| 14a | T | Over-current masking window (IRC trimmed at 1MHz)<br>-40 C < TJ < 150C | $t_{OCLIM}$ | 5 | | 6 | µs |

[1]For 3.5V<= $V_{LINSUP}$ <7V, the LINPHY is still working but with degraded parametrics.

# Appendix H
# LSDRV Electrical Specifications

This section provides electrical parametric and ratings for the LSDRV.

## H.1 Static Characteristics

**Table H-1. Static Characteristics - LSDRV**

| Characteristics noted under conditions 6V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | P | VSUP range for LSDRV compliant electrical characteristics | $V_{SUP}$ | 6 | 12 | 18 | V |
| 2 | C | VSUP range within which the device is working without LSDRV compliant electrical characteristics | $V_{SUP}$ | | 3.5 to 6 and 18 to 27 | | V |
| 3 | P | Output Drain-to-Source On Resistance<br>$T_J$ = 25˚C, $I_{PLS0/1}$ = 150 mA<br>$T_J$ = 150˚C, $I_{PLS0/1}$ = 150 mA | $R_{DS(ON)}$ | <br>–<br>– | <br>2.3<br>– | <br>–<br>4.5 | Ω |
| 4 | P | Output Over-Current Threshold<br>The threashold is valid for each LS-driver output.<br><br>**Note:**<br>**The low-side driver is NOT intended to switch capacitive loads. A significant capacitive load on PLS0/1 would induce a current when the low-side driver gate is turned on. This current will be sensed by the over-current circuitry and eventually lead to an immediate over-current shut down.** | $I_{LIMLSX}$ | 160 | 270 | 350 | mA |
| 5 | D | Nominal Current for continuous operation.<br>This value is valid for each LS-driver output. | $I_{NOMLSX}$ | – | – | 150 | mA |
| 5 | D | Settling time after the low-side driver is enabled (write LSEx Bits) | $t_{LS_settling}$ | 1 | – | | µs |
| 7 | P | High-Load Resistance Open-Load Detection Current (if low-side driver is enabled and gate turned off) | $I_{HLROLDC}$ | 30 | 40 | 50 | µA |
| 8 | C | Leakage Current -40˚C < $T_J$ < 80˚C<br>Open Load Detection disabled. | $I_{LEAK_L}$ | – | – | 1 | µA<br>µA |

**Table H-1. Static Characteristics - LSDRV**

| Characteristics noted under conditions 6V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 9 | P | Leakage Current -40˚C < $T_J$< 150˚C<br>Open Load Detection disabled. | $I_{LEAK_H}$ | – | – | 10 | µA<br>µA |
| 10 | P | Active Output Voltage Clamp ($I_{PLS0/1}$ = 150 mA) | $V_{CLAMP}$ | 40 | 44 | – | V |

[1] $T_J$: Junction Temperature

[2] $T_A$: Ambient Temperature

# H.2 Dynamic Characteristics

**Table H-2. Dynamic Characteristics - LSDRV**

| Characteristics noted under conditions 6V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | T | Low-Side Driver Operating Frequency | $f_{LS}$ | – | – | 10 | kHz |
| 2 | T | Inductive Load on each LS-driver output | $L_{PLS0/1}$ | – | – | 450 | mH |

[1] $T_J$: Junction Temperature

[2] $T_A$: Ambient Temperature

# Appendix I
# BATS Electrical Specifications

This section describe the electrical characteristics of the Supply Voltage Sense module.

## I.1 Maximum Ratings

**Table I-1. Maximum ratings of the Supply Voltage Sense - (BATS).**

| Characteristics noted under conditions 5.5V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | D | VSENSE Max Rating | $V_{VSENSE_M}$ | -27 | – | 42 | V |

[1] $T_J$: Junction Temperature

[2] $T_A$: Ambient Temperature

# I.2 Static Electrical Characteristics

**Table I-2. Static Electrical Characteristics - Supply Voltage Sense - (BATS).**

Characteristics noted under conditions 5.5V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted.
All parameters in this table assume a in series connected $R_{VSENSE_R}$ at VSENSE pin unless otherwise noted and are valid on input voltage of $R_{VSENSE_R}$ and not on VSENSE pin.

| Num | C | Ratings | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| 1 | P | Low Voltage Warning (LBI 1)<br>Assert (Measured on selected pin, falling edge)<br>Deassert (Measured on selected pin, rising edge)<br>Hysteresis (measured on selected pin) | <br>$V_{LBI1_A}$<br>$V_{LBI1_D}$<br>$V_{LBI1_H}$ | <br>5<br>–<br>– | <br>5.5<br>–<br>0.4 | <br>6<br>6.5<br>– | <br>V<br>V<br>V |
| 2 | P | Low Voltage Warning (LBI 2)<br>Assert (Measured on selected pin, falling edge)<br>Deassert (Measured on selected pin, rising edge)<br>Hysteresis (measured on selected pin) | <br>$V_{LBI2_A}$<br>$V_{LBI2_D}$<br>$V_{LBI2_H}$ | <br>6<br>–<br>– | <br>6.75<br>–<br>0.4 | <br>7<br>7.75<br>– | <br>V<br>V<br>V |
| 3 | P | Low Voltage Warning (LBI 3)<br>Assert (Measured on selected pin, falling edge)<br>Deassert (Measured on selected pin, rising edge)<br>Hysteresis (measured on selected pin) | <br>$V_{LBI3_A}$<br>$V_{LBI3_D}$<br>$V_{LBI3_H}$ | <br>7<br>–<br>– | <br>7.75<br>–<br>0.4 | <br>8.5<br>9<br>– | <br>V<br>V<br>V |
| 4 | P | Low Voltage Warning (LBI 4)<br>Assert (Measured on selected pin, falling edge)<br>Deassert (Measured on selected pin, rising edge)<br>Hysteresis (measured on selected pin) | <br>$V_{LBI4_A}$<br>$V_{LBI4_D}$<br>$V_{LBI4_H}$ | <br>8<br>–<br>– | <br>9<br>–<br>0.4 | <br>10<br>10.5<br>– | <br>V<br>V<br>V |
| 5 | P | High Voltage Warning (HBI 1)<br>Assert (Measured on selected pin, rising edge)<br>Deassert (Measured on selected pin, falling edge)<br>Hysteresis (measured on selected pin) | <br>$V_{HBI1_A}$<br>$V_{HBI1_D}$<br>$V_{HBI1_H}$ | <br>15<br>14.5<br>– | <br>16.5<br>–<br>1.0 | <br>18<br>–<br>– | <br>V<br>V<br>V |
| 6 | P | High Voltage Warning (HBI 2)<br>Assert (Measured on selected pin, rising edge)<br>Deassert (Measured on selected pin, falling edge)<br>Hysteresis (measured on selected pin) | <br>$V_{HBI2_A}$<br>$V_{HBI2_D}$<br>$V_{HBI2_H}$ | <br>25<br>24<br>– | <br>27.5<br>–<br>1.0 | <br>29<br>–<br>– | <br>V<br>V<br>V |
| 7 | T | Pin Input Divider Ratio<br>$Ratio_{VSENSE} = V_{SENSE} / V_{ADC}$[3]<br>$Ratio_{VSUP} = V_{SUP} / V_{ADC}$<br><br>5.5V < VSENSE < 29 V; 5.5V < VSUP < 29 V | <br>$Ratio_{VSENSE}$<br>$Ratio_{VSUP}$ | <br>–<br>– | <br>9<br>9 | <br>–<br>– | <br>–<br>– |
| 8 | C | Analog Input Matching<br><br>Absolute Error on $V_{ADC}$<br>- compared to $V_{SENSE}$ / $Ratio_{VSENSE}$<br>- compared to $V_{SUP}$ / $Ratio_{VSUP}$ | $AI_{Matching}$ | – | +-2% | +-5% | – |

**Table I-2. Static Electrical Characteristics - Supply Voltage Sense - (BATS).**

| Characteristics noted under conditions 5.5V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted.<br>All parameters in this table assume a in series connected $R_{VSENSE_R}$ at VSENSE pin unless otherwise noted and are valid on input voltage of $R_{VSENSE_R}$ and not on VSENSE pin. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 9 | D | VSENSE Series Resistor<br><br>Required to be placed externally at VSENSE pin. | $R_{VSENSE_R}$ | 9.5 | 10 | 10.5 | kΩ |
| 10 | D | VSENSE Impedance<br><br>If path to ground is enabled.<br>Value at VSENSE pin. $R_{VSENSE_R}$ is excluded. | $R_{VSEN_IMP}$ | – | 350 | – | kΩ |
| 11 | D | VSENSE Input Capacitance | $C_{VSEN_IN}$ | – | 8 | – | pF |

1 $T_J$: Junction Temperature

2 $T_A$: Ambient Temperature

3 $V_{ADC}$: Voltage accessible at the ATD input channel

# I.3 Dynamic Electrical Characteristics

**Table I-3. Dynamic Electrical Characteristics - Supply Voltage Sense - (BATS).**

| Characteristics noted under conditions 5.5V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A$ = 25˚C[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | D | Enable Stabilisation Time | $T_{EN_UNC}$ | – | 1 | – | µs |
| 2 | D | Voltage Warning Low Pass Filter | $f_{VWLP_filter}$ | – | 0.5 | – | MHz |

1 $T_J$: Junction Temperature

2 $T_A$: Ambient Temperature

# Appendix J
# PIM Electrical Specifications

## J.1 High-Voltage Inputs (HVI) Electrical Characteristics

**Table J-1. Static Electrical Characteristics - High Voltage Input Pins - Port L**

| Characteristics are $5.5V \le V_{SUP} \le 18$ V, $-40^{\circ}C \le T_J \le 150^{\circ}C$[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A = 25^{\circ}C$[2] under nominal conditions unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Ratings** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 2 | P | Digital Input Threshold<br>• $V_{SUP} > 6.5V$<br>• $5.5V \le V_{SUP} \le 6.5V$ | $V_{TH_HVI}$ | <br>2.8<br>2.0 | <br>3.5<br>2.5 | <br>4.5<br>3.8 | <br>V<br>V |
| 3 | D | Input Hysteresis | $V_{HYS_HVI}$ | _ | 250 | _ | mV |
| 4 | T | Pin Input Divider Ratio with external series $R_{EXT_HVI}$<br>Ratio = $V_{HVI}$ / $V_{Internal(ADC)}$ | <br>$Ratio_{L_HVI}$<br>$Ratio_{H_HVI}$ | <br>–<br>– | <br>2<br>6 | <br>–<br>– | |
| 5 | C | Analog Input Matching<br><br>Absolute Error on $V_{ADC}$<br>• Compared to $V_{HVI}$ / $Ratio_{L_HVI}$ ($1V < V_{HVI} < 7V$)<br>• Compared to $V_{HVI}$ / $Ratio_{H_HVI}$ ($3V < V_{HVI} < 21V$)<br>• Direct Mode (PTADIRL=1) ($0.5V < V_{HVI} < 3.5V$) | <br><br><br>$AIM_{L_HVI}$<br>$AIM_{H_HVI}$<br>$AIM_{D_HVI}$ | <br><br><br>–<br>–<br>– | <br><br><br>± 2<br>± 2<br>± 2 | <br><br><br>± 5<br>± 5<br>± 5 | <br><br><br>%<br>%<br>% |
| 6 | D | High Voltage Input Series Resistor<br>**Note:** Always required externally at HVI pins. | $R_{EXT_HVI}$ | – | 10 | – | kΩ |
| 7 | D | Enable Uncertainty Time | $t_{UNC_HVI}$ | – | 1 | – | µs |
| 8 | D | Input capacitance | $C_{IN_HVI}$ | – | 8 | – | pF |

1 $T_J$: Junction Temperature

2 $T_A$: Ambient Temperature

**Table J-2. Absolute Maximum Ratings - High Voltage Input Pins - Port L**

| Num | C | Ratings | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| 1 | D | $V_{HVI}$ Voltage Range | $V_{HVI}$ | -27 | – | 42 | V |

# J.2 Pin Interrupt Characteristics

**Table J-3. Pin Interrupt Characteristics**

| Characteristics are 5.5V ≤ VSUP ≤ 18 V, -40˚C ≤ $T_J$ ≤ 150˚C[1] junction temperature from –40°C to +150°C unless otherwise noted. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | P | Port L, P, AD interrupt input pulse filtered (STOP)[2] | $t_{P_MASK}$ | — | — | 3 | µs |
| 2 | P | Port L, P, AD interrupt input pulse passed (STOP)[2] | $t_{P_PASS}$ | 10 | — | — | µs |
| 3 | D | Port L, P, AD interrupt input pulse filtered ($\overline{STOP}$) in number of bus clock cycles of period $1/f_{bus}$ | $n_{P_MASK}$ | — | — | 3 | |
| 4 | D | Port L, P, AD interrupt input pulse passed ($\overline{STOP}$) in number of bus clock cycles of period $1/f_{bus}$ | $n_{P_PASS}$ | 4 | — | — | |
| 5 | D | $\overline{IRQ}$ pulse width, edge-sensitive mode ($\overline{STOP}$) in number of bus clock cycles of period $1/f_{bus}$ | $n_{IRQ}$ | 1 | — | — | |

1 $T_J$: Junction Temperature

2 Parameter only applies in stop or pseudo stop mode.

# Appendix K
# SPI Electrical Specifications

This section provides electrical parametrics and ratings for the SPI.

In **Table K-1.** the measurement conditions are listed.

**Table K-1. Measurement Conditions**

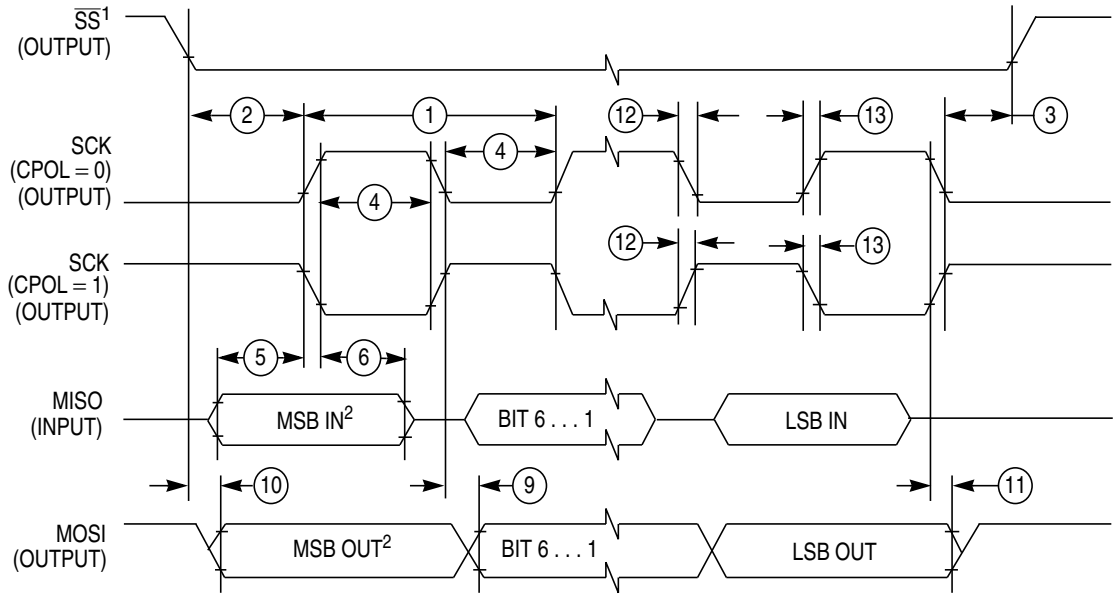
| Description | Value | Unit |
|---|---|---|
| Drive mode | full drive mode | — |
| Load capacitance $C_{LOAD}$[1], on all outputs | 50 | pF |
| Thresholds for delay measurement points | (35% / 65%) VDDX | V |

[1]Timing specified for equal load on all SPI output pins. Avoid asymmetric load.

## K.1 Timing

### K.1.1 Master Mode

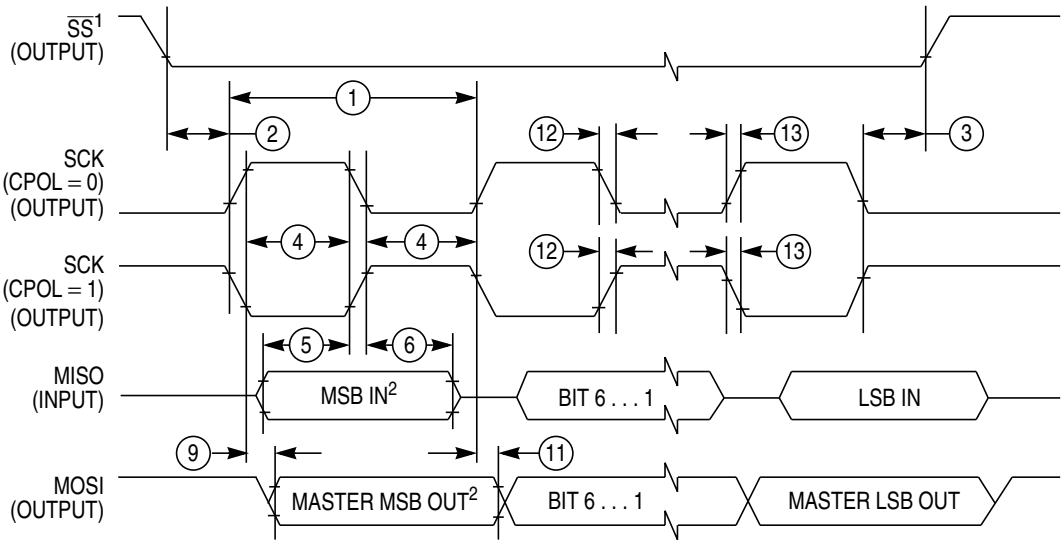
In **Figure K-1.** the timing diagram for master mode with transmission format CPHA=0 is depicted.




1. If enabled.
2. LSBFE = 0. For LSBFE = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

**Figure K-1. SPI Master Timing (CPHA=0)**

In **Figure K-2.** the timing diagram for master mode with transmission format CPHA=1 is depicted.




1. If enabled.
2. LSBFE = 0. For LSBFE = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

**Figure K-2. SPI Master Timing (CPHA=1)**

In **Figure K-2.** the timing characteristics for master mode are listed.

**Table K-2. SPI Master Mode Timing Characteristics**

| Num | C | Characteristic | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| 1 | D | SCK Frequency | $f_{sck}$ | 1/2048 | — | 1/2[1] | $f_{bus}$ |
| 1 | D | SCK Period | $t_{sck}$ | 2[1] | — | 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 | $t_{vsck}$ | — | — | 15 | ns |
| 10 | D | Data Valid after $\overline{SS}$ fall (CPHA=0) | $t_{vss}$ | — | — | 15 | ns |
| 11 | D | Data Hold Time (Outputs) | $t_{ho}$ | 0 | — | — | ns |
| 12 | D | Rise and Fall Time Inputs | $t_{rfi}$ | — | — | 8 | ns |
| 13 | D | Rise and Fall Time Outputs | $t_{rfo}$ | — | — | 8 | ns |

[1]pls. see **Figure K-3.**

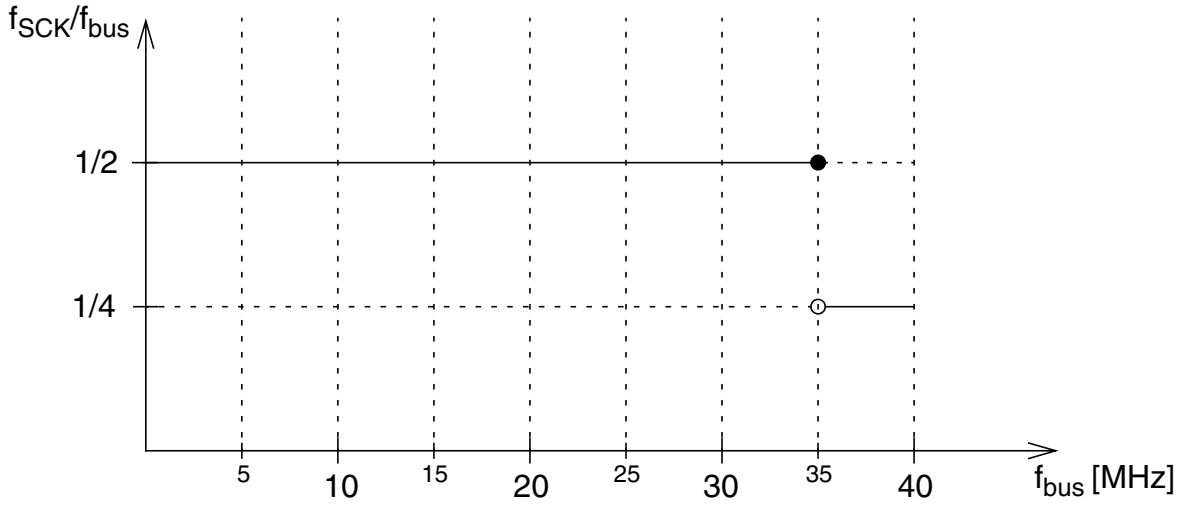



**Figure K-3. Derating of maximum $f_{SCK}$ to $f_{bus}$ ratio in Master Mode**

In Master Mode the allowed maximum $f_{SCK}$ to $f_{bus}$ ratio (= minimum Baud Rate Divisor, pls. see SPI Block Guide) derates with increasing $f_{bus}$, please see **Figure K-3.**.

## K.1.2 Slave Mode

In **Figure K-4.** the timing diagram for slave mode with transmission format CPHA=0 is depicted.

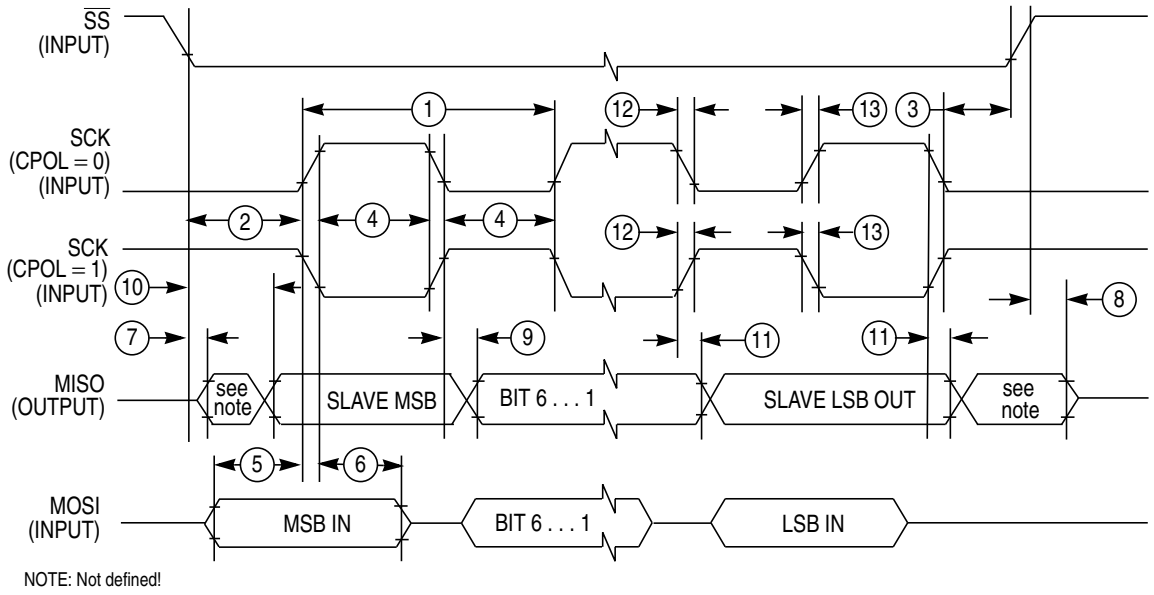



**Figure K-4. SPI Slave Timing (CPHA=0)**

In **Figure K-5.** the timing diagram for slave mode with transmission format CPHA=1 is depicted.

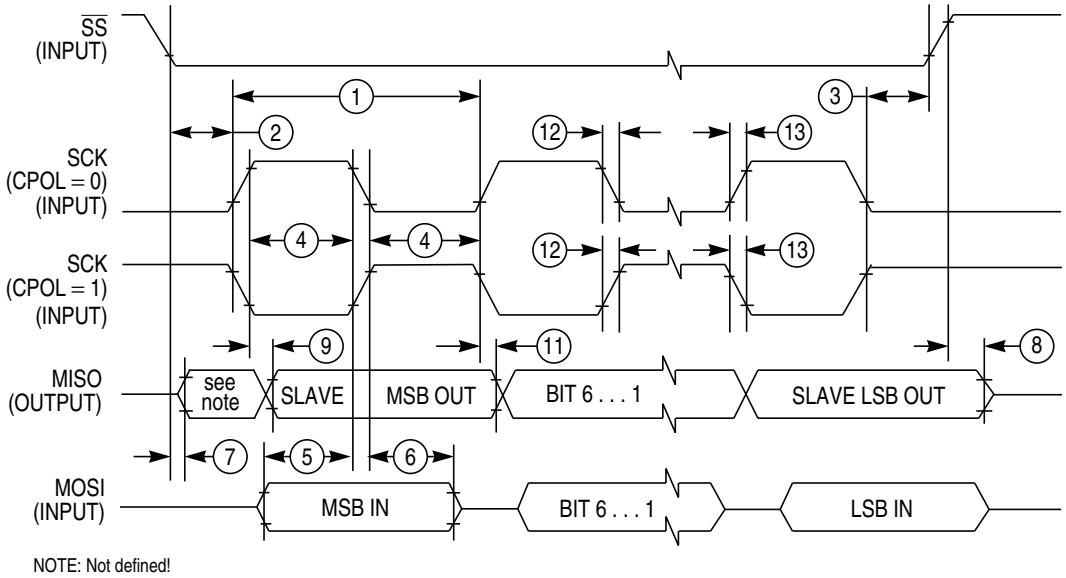



**Figure K-5. SPI Slave Timing (CPHA=1)**

In **Table K-3.** the timing characteristics for slave mode are listed.

**Table K-3. SPI Slave Mode Timing Characteristics**

| Num | C | Characteristic | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| 1 | D | SCK Frequency | $f_{sck}$ | DC | — | 1/4 | $f_{bus}$ |
| 1 | D | 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 | $t_{vsck}$ | — | — | $28 + 0.5 \cdot t_{bus}$ [1] | ns |
| 10 | D | Data Valid after $\overline{SS}$ fall | $t_{vss}$ | — | — | $28 + 0.5 \cdot t_{bus}$ [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] $0.5t_{bus}$ added due to internal synchronization delay

# Appendix L
# XOSCLCP Electrical Specifications

**Table L-1. XOSCLCP Characteristics**

| Conditions are shown in Table A-5 unless otherwise noted | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Num** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| 1 | C | Nominal crystal or resonator frequency | $f_{OSC}$ | 4.0 | | 20 | MHz |
| 2 | P | Startup Current | $i_{OSC}$ | 100 | | | µA |
| 3a | C | Oscillator start-up time (4MHz)[1] | $t_{UPOSC}$ | — | 2 | 10 | ms |
| 3b | C | Oscillator start-up time (8MHz)[1] | $t_{UPOSC}$ | — | 1.6 | 8 | ms |
| 3c | C | Oscillator start-up time (16MHz)[1] | $t_{UPOSC}$ | — | 1 | 5 | ms |
| 4 | P | Clock Monitor Failure Assert Frequency | $f_{CMFA}$ | 200 | 450 | 1200 | kHz |
| 5 | D | Input Capacitance (EXTAL, XTAL pins) | $C_{IN}$ | | 7 | | pF |
| 6 | C | EXTAL Pin Input Hysteresis | $V_{HYS,EXTAL}$ | — | 120 | — | mV |
| 7 | C | EXTAL Pin oscillation amplitude (loop controlled Pierce) | $V_{PP,EXTAL}$ | — | 0.9 | — | V |
| 8 | D | EXTAL Pin oscillation required amplitude[2] | $V_{PP,EXTAL}$ | 0.8 | — | 1.5 | V |

[1] These values apply for carefully designed PCB layouts with capacitors that match the crystal/resonator requirements.

[2]Needs to be measured at room temperature on the application board using a probe with very low (<=5pF) input capacitance.

# Appendix M
# FTMRG Electrical Specifications

## M.1 Timing Parameters

The time base for all NVM program or erase operations is derived from the bus clock using the FCLKDIV register. The frequency of this derived clock must be set within the limits specified as $f_{NVMOP}$. The NVM module does not have any means to monitor the frequency and will not prevent program or erase operation at frequencies above or below the specified minimum. When attempting to program or erase the NVM module at a lower frequency, a full program or erase transition is not assured.

All timing parameters are a function of the bus clock frequency, $f_{NVMBUS}$. All program and erase times are also a function of the NVM operating frequency, $f_{NVMOP}$. A summary of key timing parameters can be found in **Table M-1.** and **Table M-2.**

**Table M-1. Operating frequency**

| | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Bus Frequency | $f_{NVMBUS}$ | 1 | 25 | 25 | MHz |
| Operating Frequency | $f_{NVMOP}$ | 0,8 | 1 | 1.05 | MHz |

**Table M-2. NVM Timing Characteristics**

| # | C | Command | $f_{NVMOP}$ cycle | $f_{NVMBUS}$ cycle | Symbol | Min[1] | Typ[2] | Max[3] | Lfmax[4] | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | D | Erase Verify All Blocks | 0 | 17169 | $t_{RD1ALL}$ | 0.69 | 0.69 | 1.37 | 34.34 | ms |
| 2 | D | Erase Verify Block (Pflash) | 0 | 16924 | $t_{RD1BLK_P}$ | 0.68 | 0.68 | 1.35 | 33.85 | ms |
| 3 | D | Erase Verify Block (EEPROM) | 0 | 744 | $t_{RD1BLK_D}$ | 0.03 | 0.03 | 0.06 | 1.49 | ms |
| 4 | D | Erase Verify P-Flash Section | 0 | 476 | $t_{RD1SEC}$ | 19.04 | 19.04 | 38.08 | 952.00 | us |
| 5 | D | Read Once | 0 | 445 | $t_{RDONCE}$ | 17.80 | 17.80 | 17.80 | 445.00 | us |
| 6 | D | Program P-Flash (4 Word) | 164 | 2896 | $t_{PGM_4}$ | 0.27 | 0.28 | 0.63 | 11.79 | ms |
| 7 | D | Program Once | 164 | 2859 | $t_{PGMONCE}$ | 0.27 | 0.28 | 0.28 | 3.06 | ms |
| 8 | D | Erase All Blocks | 100066 | 17505 | $t_{ERSALL}$ | 96.00 | 100.77 | 101.47 | 160.09 | ms |

**Table M-2. NVM Timing Characteristics**

| # | C | Command | $f_{NVMOP}$ cycle | $f_{NVMBUS}$ cycle | Symbol | Min[1] | Typ[2] | Max[3] | Lfmax[4] | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| 9 | D | Erase Flash Block (Pflash) | 100060 | 17150 | $t_{ERSBLK_P}$ | 95.98 | 100.75 | 101.43 | 159.38 | ms |
| 10 | D | Erase Flash Block (EEPROM) | 100060 | 1033 | $t_{ERSBLK_D}$ | 95.34 | 100.10 | 100.14 | 127.14 | ms |
| 11 | D | Erase P-Flash Sector | 20015 | 858 | $t_{ERSPG}$ | 19.10 | 20.05 | 20.08 | 26.73 | ms |
| 12 | D | Unsecure Flash | 100066 | 17570 | $t_{UNSECU}$ | 96.00 | 100.77 | 101.47 | 160.22 | ms |
| 13 | D | Verify Backdoor Access Key | 0 | 481 | $t_{VFYKEY}$ | 19.24 | 19.24 | 19.24 | 481.00 | us |
| 14 | D | Set User Margin Level | 0 | 399 | $t_{MLOADU}$ | 15.96 | 15.96 | 15.96 | 399.00 | us |
| 15 | D | Set Factory Margin Level | 0 | 408 | $t_{MLOADF}$ | 16.32 | 16.32 | 16.32 | 408.00 | us |
| 16 | D | Erase Verify EEPROM Section | 0 | 546 | $t_{DRD1SEC}$ | 0.02 | 0.02 | 0.04 | 1.09 | ms |
| 17 | D | Program EEPROM (1 Word) | 68 | 1552 | $t_{DPGM_1}$ | 0.13 | 0.13 | 0.32 | 6.29 | ms |
| 18 | D | Program EEPROM (2 Word) | 136 | 2486 | $t_{DPGM_2}$ | 0.23 | 0.24 | 0.53 | 10.11 | ms |
| 19 | D | Program EEPROM (3 Word) | 204 | 3420 | $t_{DPGM_3}$ | 0.33 | 0.34 | 0.75 | 13.94 | ms |
| 20 | D | Program EEPROM (4 Word) | 272 | 4354 | $t_{DPGM_4}$ | 0.43 | 0.45 | 0.97 | 17.76 | ms |
| 21 | D | Erase EEPROM Sector | 5015 | 746 | $t_{DERSPG}$ | 4.81 | 5.04 | 20.57 | 37.76 | ms |

1 Minimum times are based on maximum $f_{NVMOP}$ and maximum $f_{NVMBUS}$

2 Typical times are based on typical $f_{NVMOP}$ and typical $f_{NVMBUS}$

3 Maximum times are based on typical $f_{NVMOP}$ and typical $f_{NVMBUS}$ plus aging

4 Worst times are based on minimum $f_{NVMOP}$ and minimum $f_{NVMBUS}$

Table M-3. Timing Characteristics

## M.1.1 NVM Reliability Parameters

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 data retention and program/erase cycling failure rates are specified at the operating conditions noted. The program/erase cycle count on the sector is incremented every time a sector or mass erase event is executed.

**Table M-4. NVM Reliability Characteristics**

| Conditions are shown in Table A-5 unless otherwise noted | | | | | | | |
|---|---|---|---|---|---|---|---|
| **NUM** | **C** | **Rating** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| | | **Program Flash Arrays** | | | | | |
| 1 | C | Data retention at an average junction temperature of $T_{Javg}$ = 85°C[1] after up to 10,000 program/erase cycles | $t_{NVMRET}$ | 20 | 100[2] | — | Years |
| 2 | C | Program Flash number of program/erase cycles (-40°C ≤ Tj ≤ 150°C) | $n_{FLPE}$ | 10K | 100K[3] | — | Cycles |
| | | **EEPROM Array** | | | | | |
| 3 | C | Data retention at an average junction temperature of $T_{Javg}$ = 85°C[1] after up to 100,000 program/erase cycles | $t_{NVMRET}$ | 5 | 100[2] | — | Years |
| 4 | C | Data retention at an average junction temperature of $T_{Javg}$ = 85°C[1] after up to 10,000 program/erase cycles | $t_{NVMRET}$ | 10 | 100[2] | — | Years |
| 5 | C | Data retention at an average junction temperature of $T_{Javg}$ = 85°C[1] after less than 100 program/erase cycles | $t_{NVMRET}$ | 20 | 100[2] | — | Years |
| 6 | C | EEPROM number of program/erase cycles (-40°C ≤ Tj ≤ 150°C) | $n_{FLPE}$ | 100K | 500K[3] | — | Cycles |

1 $T_{Javg}$ does not exceed 85°C in a typical temperature profile over the lifetime of a consumer, industrial or automotive application.

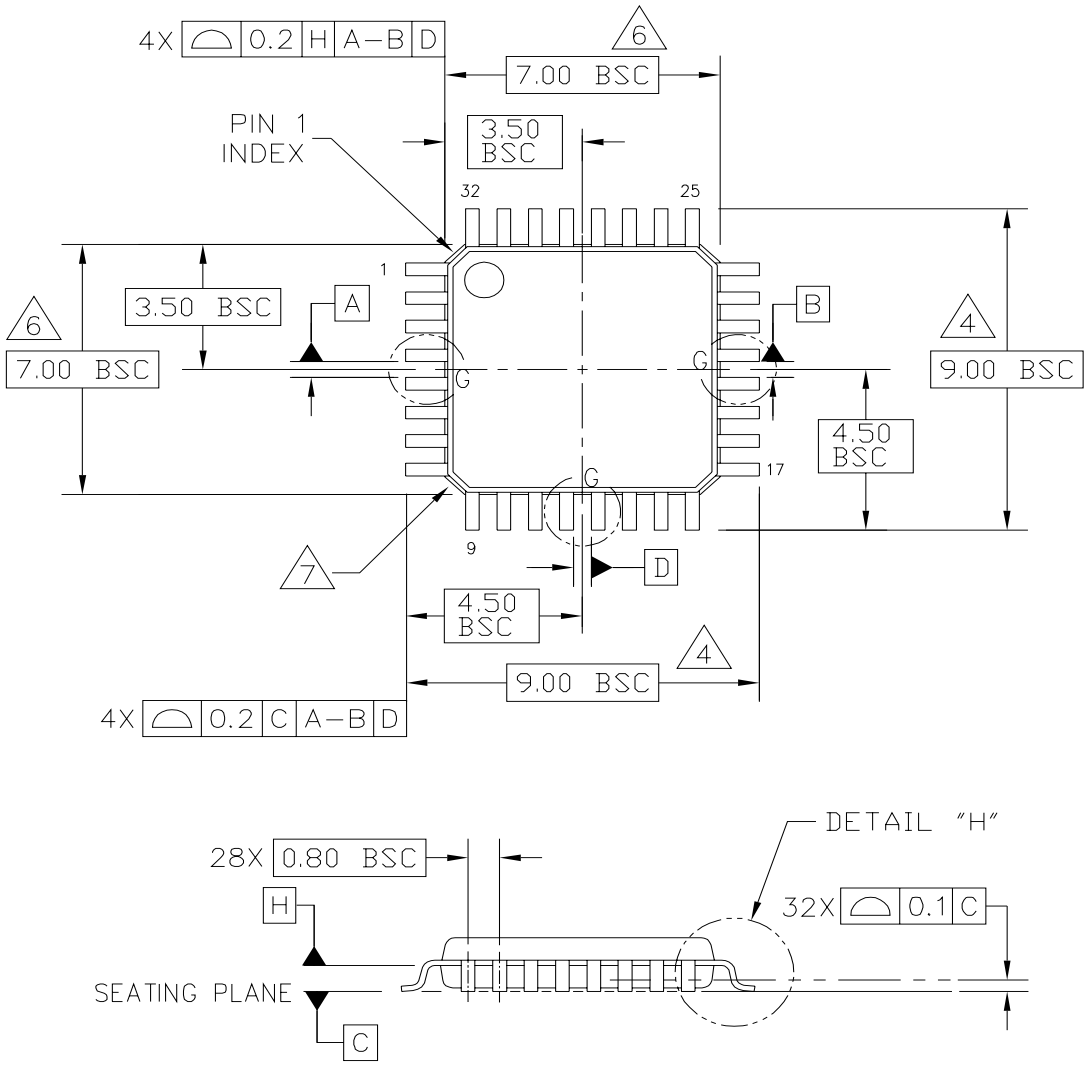
2 Typical data retention values are based on intrinsic capability of the technology measured at high temperature and de-rated to 25°C using the Arrhenius equation. For additional information on how Freescale defines Typical Data Retention, please refer to Engineering Bulletin EB618

3 Spec table quotes typical endurance evaluated at 25°C for this product family. For additional information on how Freescale defines Typical Endurance, please refer to Engineering Bulletin EB619.

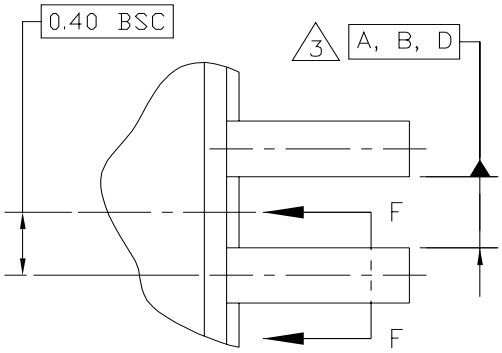
## M.1.2 NVM Factory Shipping Condition

Devices are shipped from the factory with flash and EEPROM in the erased state. Data retention specifications stated in **Table M-4.** begin at time of this erase operation. For additional information on how Freescale defines Typical Data Retention, please refer to Engineering Bulletin EB618.

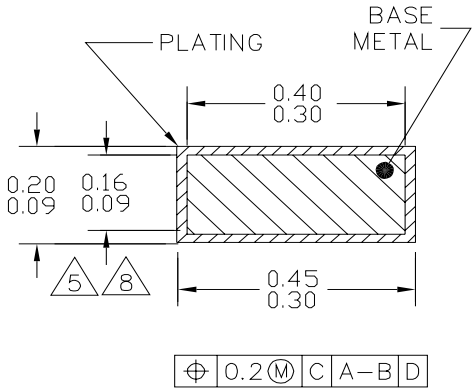
# Appendix N
# Package Information



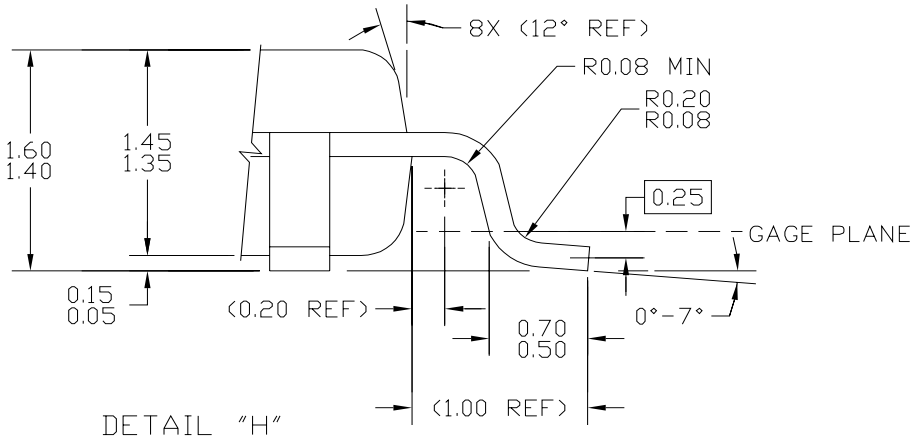

| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|---|
| TITLE: LOW PROFILE QUAD FLAT PACK (LQFP) 32 LEAD, 0.8 PITCH (7 X 7 X 1.4) | DOCUMENT NO: 98ASH70029A | REV: D |
| | CASE NUMBER: 873A-03 | 19 MAY 2005 |
| | STANDARD: JEDEC MS-026 BBA | |




DETAIL G




SECTION F–F
ROTATED 90°CW
32 PLACES




DETAIL "H"

| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|---|
| TITLE: LOW PROFILE QUAD FLAT PACK (LQFP) 32 LEAD, 0.8 PITCH (7 X 7 X 1.4) | DOCUMENT NO: 98ASH70029A | REV: D |
| | CASE NUMBER: 873A-03 | 19 MAY 2005 |
| | STANDARD: JEDEC MS-026 BBA | |

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5–1994.

3. DATUMS A, B, AND D TO BE DETERMINED AT DATUM PLANE H.

4. DIMENSIONS TO BE DETERMINED AT SEATING PLANE DATUM C.

5. DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM DIMENSION BY MORE THAN 0.08 MM. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION: 0.07 MM.

6. DIMENSIONS DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 MM PER SIDE. DIMENSIONS ARE MAXIMUM PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH.

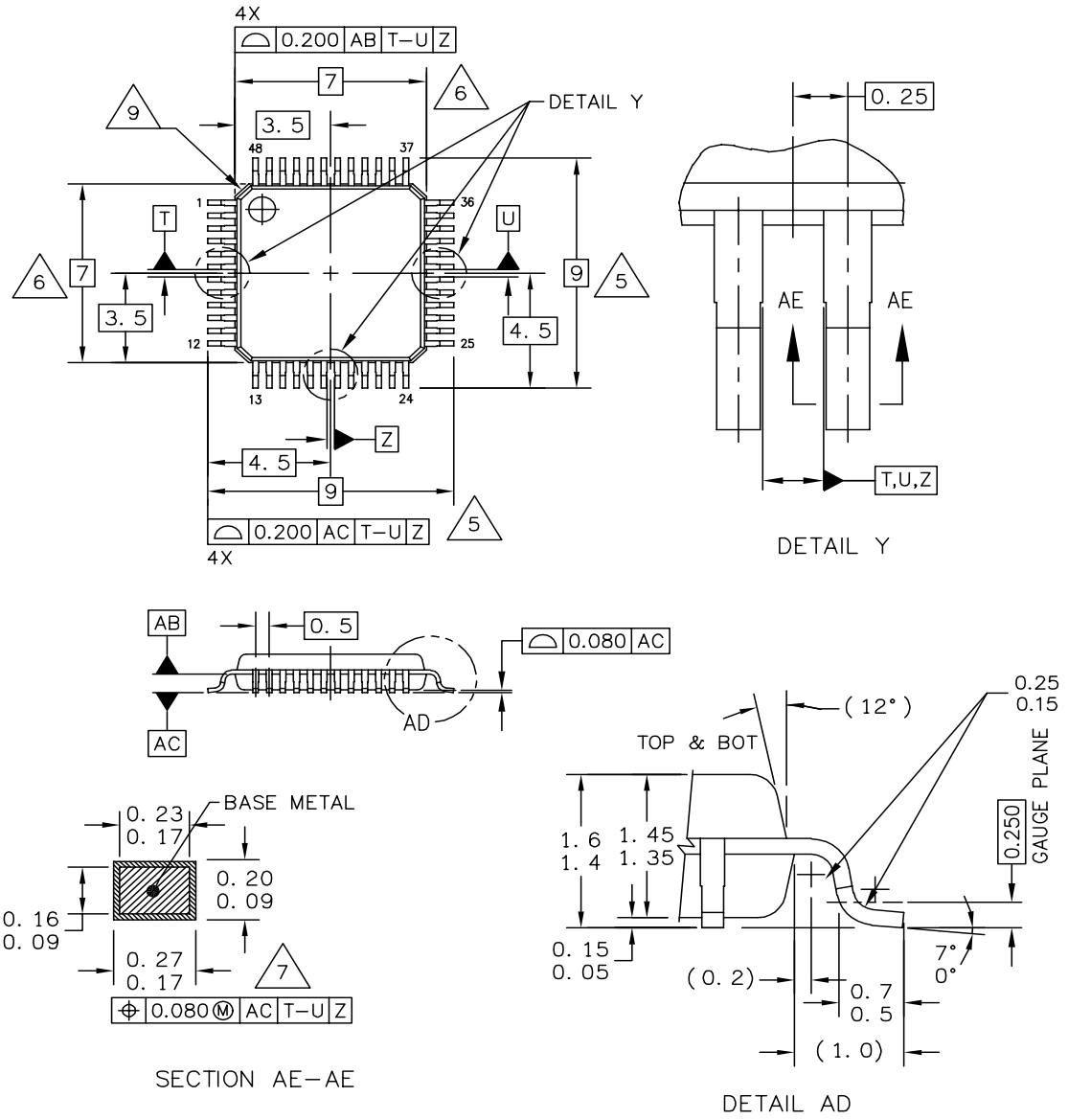
7. EXACT SHAPE OF EACH CORNER IS OPTIONAL.

8. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.1 MM AND 0.25 MM FROM THE LEAD TIP.

© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

| MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|

| TITLE: | DOCUMENT NO: 98ASH70029A | REV: D |
|---|---|---|
| LOW PROFILE QUAD FLAT PACK (LQFP)<br>32 LEAD, 0.8 PITCH (7 X 7 X 1.4) | CASE NUMBER: 873A-03 | 19 MAY 2005 |
| | STANDARD: JEDEC MS-026 BBA | |




| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|---|
| TITLE: LQFP, 48 LEAD, 0.50 PITCH (7.0 X 7.0 X 1.4) | DOCUMENT NO: 98ASH00962A | REV: G |
| | CASE NUMBER: 932-03 | 14 APR 2005 |
| | STANDARD: JEDEC MS-026-BBC | |

NOTES:

1. DIMENSIONS AND TOLERANCING PER ASME Y14.5M–1994.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE AB IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS T, U, AND Z TO BE DETERMINED AT DATUM PLANE AB.
5. DIMENSIONS TO BE DETERMINED AT SEATING PLANE AC.
6. DIMENSIONS DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.250 PER SIDE. DIMENSIONS DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE AB.
7. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.350.
8. MINIMUM SOLDER PLATE THICKNESS SHALL BE 0.0076.
9. EXACT SHAPE OF EACH CORNER IS OPTIONAL.

| © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. | MECHANICAL OUTLINE | PRINT VERSION NOT TO SCALE |
|---|---|---|
| TITLE: LQFP, 48 LEAD, 0.50 PITCH (7.0 X 7.0 X 1.4) | DOCUMENT NO: 98ASH00962A | REV: G |
| | CASE NUMBER: 932-03 | 14 APR 2005 |
| | STANDARD: JEDEC MS-026-BBC | |

# Appendix O
# Ordering Information

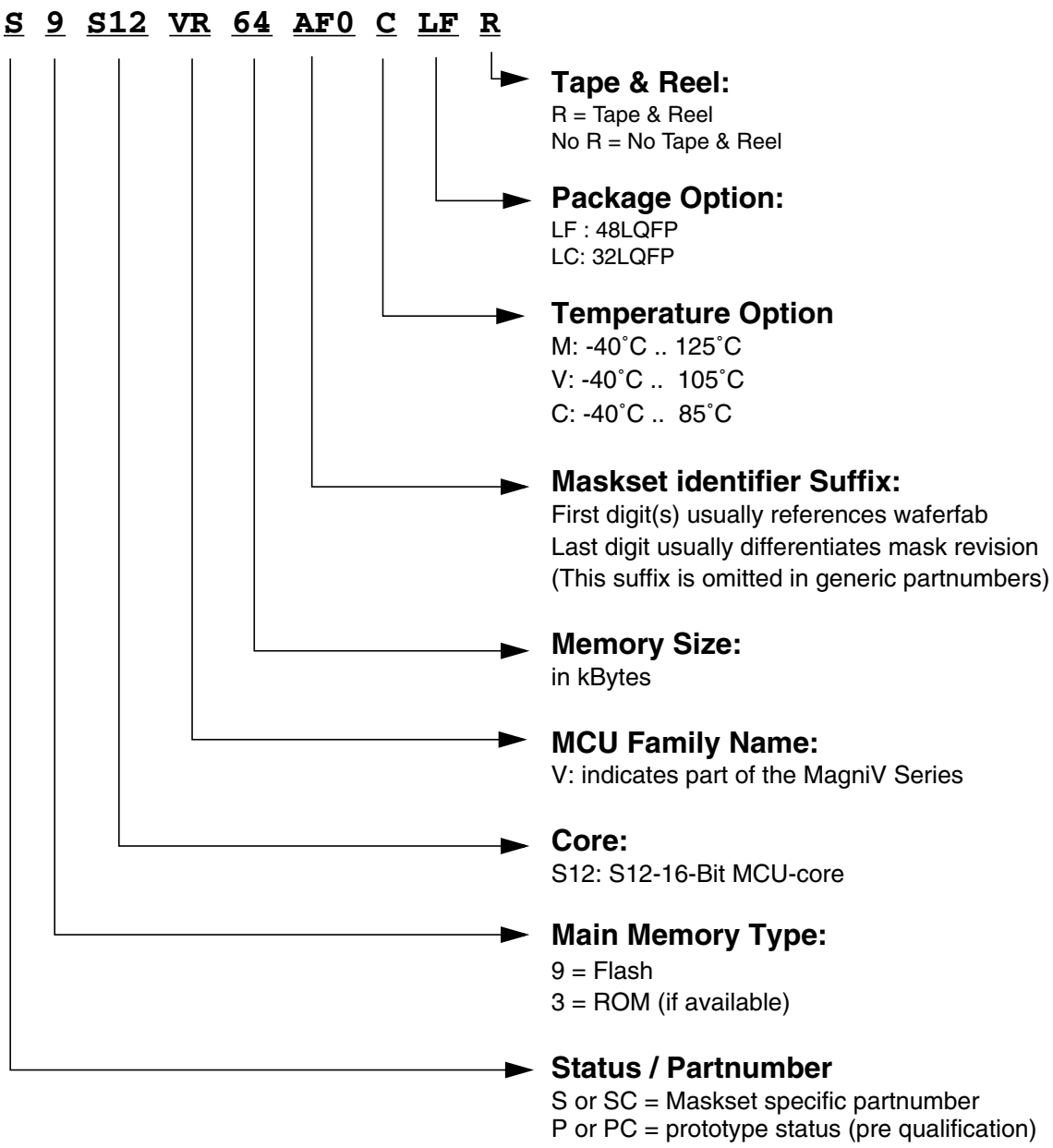
The following figure provides an ordering partnumber example for the devices covered by this data book. There are two options when ordering a device. Customers must choose between ordering either the mask-specific partnumber or the generic / mask-independent partnumber. Ordering the mask-specific partnumber enables the customer to specify which particular maskset they will receive whereas ordering the generic maskset means that FSL will ship the currently preferred maskset (which may change over time).

In either case, the marking on the device will always show the generic / mask-independent partnumber and the mask set number.

**NOTE**

**The mask identifier suffix and the Tape & Reel suffix are always both omitted from the partnumber which is actually marked on the device.**

For specific partnumbers to order, please contact your local sales office. The below figure illustrates the structure of a typical mask-specific ordering number for the MC9S12VR64 devices.


S 9 S12 VR 64 AF0 C LF R
Tape & Reel:
R = Tape & Reel
No R = No Tape & Reel
Package Option:
LF : 48LQFP
LC: 32LQFP
Temperature Option
M: -40˚C .. 125˚C
V: -40˚C .. 105˚C
C: -40˚C .. 85˚C
Maskset identifier Suffix:
First digit(s) usually references waferfab
Last digit usually differentiates mask revision
(This suffix is omitted in generic partnumbers)
Memory Size:
in kBytes
MCU Family Name:
V: indicates part of the MagniV Series
Core:
S12: S12-16-Bit MCU-core
Main Memory Type:
9 = Flash
3 = ROM (if available)
Status / Partnumber
S or SC = Maskset specific partnumber
P or PC = prototype status (pre qualification)

# Appendix P
# Detailed Register Address Map

## P.1 Detailed Register Map

The following tables show the detailed register map of the MC9S12VR64.

### 0x0000-0x0009 Port Integration Module (PIM) Map 1 of 4

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | W | | | | | | | | |
| 0x0000 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0001 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0002 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0003 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0004 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0005 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0006 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0007 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0008 | PORTE | R | 0 | 0 | 0 | 0 | 0 | 0 | PE1 | PE0 |
| | | W | | | | | | | | |
| 0x0009 | DDRE | R | 0 | 0 | 0 | 0 | 0 | 0 | DDRE1 | DDRE0 |
| | | W | | | | | | | | |

### 0x000A-0x000B Module Mapping Conrol (MMC) Map 1 of 2

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x000A | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000B | MODE | R | MODC | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x000C-0x000D Port Integration Module (PIM) Map 2 of 4

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x000C | PUCR | R | 0 | BKPUE | 0 | PDPEE | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000D | Reserved | R | 0 | 0 | 0 | | 0 | 0 | | |
| | | W | | | | | | | | |

## 0x000E-0x000F Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x000E | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x000F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0010-0x0017 Module Mapping Control (MMC) Map 2 of 2

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0010 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0011 | DIRECT | R | DP15 | DP14 | DP13 | DP12 | DP11 | DP10 | DP9 | DP8 |
| | | W | | | | | | | | |
| 0x0012 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0013 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | NVMRES |
| | | W | | | | | | | | |
| 0x0014 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0015 | PPAGE | R | 0 | 0 | 0 | 0 | PIX3 | PIX2 | PIX1 | PIX0 |
| | | W | | | | | | | | |
| 0x0016 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0017 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0018-0x0019 Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0018 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0019 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x001A-0x001B Part ID Registers

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x001A | PARTIDH | R | PARTIDH | | | | | | | |
| | | W | | | | | | | | |
| 0x001B | PARTIDL | R | PARTIDL | | | | | | | |
| | | W | | | | | | | | |

## 0x001C-0x001F Port Intergartion Module (PIM) Map 3 of 4

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x001C | ECLKCTL | R | NECLK | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x001D | PIMMISC | R | OCPE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x001E | IRQCR | R | IRQE | IRQEN | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x001F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0020-0x002F Debug Module (S12SDBG) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0020 | DBGC1 | R | ARM | 0 | 0 | BDM | DBGBRK | 0 | COMRV | |
| | | W | | TRIG | | | | | | |
| 0x0021 | DBGSR | R | TBF | 0 | 0 | 0 | 0 | SSF2 | SSF1 | SSF0 |
| | | W | | | | | | | | |
| 0x0022 | DBGTCR | R | 0 | TSOURCE | 0 | 0 | TRCMOD | | 0 | TALIGN |
| | | W | | | | | | | | |
| 0x0023 | DBGC2 | R | 0 | 0 | 0 | 0 | 0 | 0 | ABCM | |
| | | W | | | | | | | | |
| 0x0024 | DBGTBH | R | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x0025 | DBGTBL | R | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x0026 | DBGCNT | R | TBF | 0 | CNT | | | | | |
| | | W | | | | | | | | |
| 0x0027 | DBGSCRX | R | 0 | 0 | 0 | 0 | SC3 | SC2 | SC1 | SC0 |
| | | W | | | | | | | | |
| 0x0027 | DBGMFR | R | 0 | 0 | 0 | 0 | 0 | MC2 | MC1 | MC0 |
| | | W | | | | | | | | |
| 0x0028[1] | DBGACTL | R | SZE | SZ | TAG | BRK | RW | RWE | NDB | COMPE |
| | | W | | | | | | | | |
| 0x0028[2] | DBGBCTL | R | SZE | SZ | TAG | BRK | RW | RWE | 0 | COMPE |
| | | W | | | | | | | | |

## 0x0020-0x002F Debug Module (S12SDBG) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0028[3] | DBGCCTL | R | 0 | 0 | TAG | BRK | RW | RWE | 0 | COMPE |
| | | W | | | | | | | | |
| 0x0029 | DBGXAH | R | 0 | 0 | 0 | 0 | 0 | 0 | Bit17 | Bit 16 |
| | | W | | | | | | | | |
| 0x002A | DBGXAM | R | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x002B | DBGXAL | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x002C | DBGADH | R | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x002D | DBGADL | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x002E | DBGADHM | R | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x002F | DBGADLM | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | | | | | | | | |

1 This represents the contents if the Comparator A or C control register is blended into this address

2 This represents the contents if the Comparator B or D control register is blended into this address

3 This represents the contents if the Comparator B or D control register is blended into this address

## 0x0030-0x0033 Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0030 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0031 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0032 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0033 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0034-0x003F Clock Reset and Power Management (CPMU) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0034 | CPMUSYNR | R | VCOFRQ[1:0] | | SYNDIV[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x0035 | CPMUREFDIV | R | REFFRQ[1:0] | | 0 | 0 | REFDIV[3:0] | | | |
| | | W | | | | | | | | |
| 0x0036 | CPMUPOSTDIV | R | 0 | 0 | 0 | POSTDIV[4:0] | | | | |
| | | W | | | | | | | | |
| 0x0037 | CPMUFLG | R | RTIF | PORF | LVRF | LOCKIF | LOCK | ILAF | OSCIF | UPOSC |
| | | W | | | | | | | | |

## 0x0034-0x003F Clock Reset and Power Management (CPMU) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0038 | CPMUINT | R | RTIE | 0 | 0 | LOCKIE | 0 | 0 | OSCIE | 0 |
| | | W | | | | | | | | |
| 0x0039 | CPMUCLKS | R | PLLSEL | PSTP | 0 | COPOSC SEL1 | PRE | PCE | RTIOSCS EL | COPOSC SEL0 |
| | | W | | | | | | | | |
| 0x003A | CPMUPLL | R | 0 | 0 | FM1 | FM0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x003B | CPMURTI | R | RTDEC | RTR6 | RTR5 | RTR4 | RTR3 | RTR2 | RTR1 | RTR0 |
| | | W | | | | | | | | |
| 0x003C | CPMUCOP | R | WCOP | RSBCK | 0 | 0 | 0 | CR2 | CR1 | CR0 |
| | | W | | | WRTMASK | | | | | |
| 0x003D | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | Reserved For Factory Test | | | | | | | |
| 0x003E | Reserved | R | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| | | W | Reserved For Factory Test | | | | | | | |
| 0x003F | CPMU ARMCOP | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |

## 0x0040-0x006F Timer Module (TIM) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0040 | TIOS | R | 0 | 0 | 0 | 0 | IOS3 | IOS2 | IOS1 | IOS0 |
| | | W | | | | | | | | |
| 0x0041 | CFORC | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | | | FOC3 | FOC2 | FOC1 | FOC0 |
| 0x0042-0x0043 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0044 | TCNTH | R | Bit 15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x0045 | TCNTL | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x0046 | TSCR1 | R | TEN | TSWAI | TSFRZ | TFFCA | PRNT | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0047 | TTOV | R | 0 | 0 | 0 | 0 | TOV3 | TOV2 | TOV1 | TOV0 |
| | | W | | | | | | | | |
| 0x0048 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0049 | TCTL2 | R | OM3 | OL3 | OM2 | OL2 | OM1 | OL1 | OM0 | OL0 |
| | | W | | | | | | | | |
| 0x004A | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x004B | TCTL4 | R | EDG3B | EDG3A | EDG2B | EDG2A | EDG1B | EDG1A | EDG0B | EDG0A |
| | | W | | | | | | | | |
| 0x004C | TIE | R | 0 | 0 | 0 | 0 | C3I | C2I | C1I | C0I |
| | | W | | | | | | | | |

## 0x0040-0x006F Timer Module (TIM) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x004D | TSCR2 | R | TOI | 0 | 0 | 0 | TCRE | PR2 | PR1 | PR0 |
| | | W | | | | | | | | |
| 0x004E | TFLG1 | R | 0 | 0 | 0 | 0 | C3F | C2F | C1F | C0F |
| | | W | | | | | | | | |
| 0x004F | TFLG2 | R | TOF | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0050 | TC0H | R | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x0051 | TC0L | R | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x0052 | TC1H | R | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x0053 | TC1L | R | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x0054 | TC2H | R | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x0055 | TC2L | R | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x0056 | TC3H | R | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| | | W | | | | | | | | |
| 0x0057 | TC3L | R | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x0068–0x006B | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x006C | OCPD | R | 0 | 0 | 0 | 0 | OCPD3 | OCPD2 | OCPD1 | OCPD0 |
| | | W | | | | | | | | |
| 0x006D | Reserved | R | | | | | | | | |
| | | W | | | | | | | | |
| 0x006E | PTPSR | R | PTPS7 | PTPS6 | PTPS5 | PTPS4 | PTPS3 | PTPS2 | PTPS1 | PTPS0 |
| | | W | | | | | | | | |
| 0x006F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0070-0x009F Analog to Digital Converter 10-Bit 6-Channel (ATD) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0070 | ATDCTL0 | R | 0 | 0 | 0 | 0 | WRAP3 | WRAP2 | WRAP1 | WRAP0 |
| | | W | | | | | | | | |
| 0x0071 | ATDCTL1 | R | ETRIG SEL | SRES1 | SRES0 | SMP_DIS | ETRIG CH3 | ETRIG CH2 | ETRIG CH1 | ETRIG CH0 |
| | | W | | | | | | | | |
| 0x0072 | ATDCTL2 | R | 0 | AFFC | ICLKSTP | ETRIGLE | ETRIGP | ETRIGE | ASCIE | ACMPIE |
| | | W | | | | | | | | |
| 0x0073 | ATDCTL3 | R | DJM | S8C | S4C | S2C | S1C | FIFO | FRZ1 | FRZ0 |
| | | W | | | | | | | | |

## 0x0070-0x009F Analog to Digital Converter 10-Bit 6-Channel (ATD) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0074 | ATDCTL4 | R | SMP2 | SMP1 | SMP0 | PRS4 | PRS3 | PRS2 | PRS1 | PRS0 |
| | | W | | | | | | | | |
| 0x0075 | ATDCTL5 | R | 0 | SC | SCAN | MULT | CD | CC | CB | CA |
| | | W | | | | | | | | |
| 0x0076 | ATDSTAT0 | R | SCF | 0 | ETORF | FIFOR | CC3 | CC2 | CC1 | CC0 |
| | | W | | | | | | | | |
| 0x0077 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0078 | ATDCMPEH | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0079 | ATDCMPEL | R | 0] | 0 | CMPE[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x007A | ATDSTAT2H | R | | | | | | | | |
| | | W | | | | | | | | |
| 0x007B | ATDSTAT2L | R | 0 | 0 | CCF[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x007C | ATDDIENH | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x007D | ATDDIENL | R | 0 | 0 | IEN[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x007E | ATDCMPHTH | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x007F | ATDCMPHTL | R | 0 | 0 | CMPHT[5:0] | | | | | |
| | | W | | | | | | | | |
| 0x0080 | ATDDR0H | R | Bit15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit8 |
| | | W | | | | | | | | |
| 0x0081 | ATDDR0L | R | Bit7 | Bit6 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0082 | ATDDR1H | R | Bit15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit8 |
| | | W | | | | | | | | |
| 0x0083 | ATDDR1L | R | Bit7 | Bit6 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0084 | ATDDR2H | R | Bit15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit8 |
| | | W | | | | | | | | |
| 0x0085 | ATDDR2L | R | Bit7 | Bit6 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0086 | ATDDR3H | R | Bit15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit8 |
| | | W | | | | | | | | |
| 0x0087 | ATDDR3L | R | Bit7 | Bit6 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0088 | ATDDR4H | R | Bit15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit8 |
| | | W | | | | | | | | |
| 0x0089 | ATDDR4L | R | Bit7 | Bit6 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0070-0x009F Analog to Digital Converter 10-Bit 6-Channel (ATD) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x008A | ATDDR5H | R | Bit15 | 14 | 13 | 12 | 11 | 10 | 9 | Bit8 |
| | | W | | | | | | | | |
| 0x008B | ATDDR5L | R | Bit7 | Bit6 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x008C-0x009F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x00A0-0x00C7 Pulse Width Modulator 6-Channels (PWM) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00A0 | PWME | R | PWME7 | PWME6 | PWME5 | PWME4 | PWME3 | PWME2 | PWME1 | PWME0 |
| | | W | | | | | | | | |
| 0x00A1 | PWMPOL | R | PPOL7 | PPOL6 | PPOL5 | PPOL4 | PPOL3 | PPOL2 | PPOL1 | PPOL0 |
| | | W | | | | | | | | |
| 0x00A2 | PWMCLK | R | PCLK7 | PCLK6 | PCLK5 | PCLK4 | PCLK3 | PCLK2 | PCLK1 | PCLK0 |
| | | W | | | | | | | | |
| 0x00A3 | PWMPRCLK | R | 0 | PCKB2 | PCKB1 | PCKB0 | 0 | PCKA2 | PCKA1 | PCKA0 |
| | | W | | | | | | | | |
| 0x00A4 | PWMCAE | R | CAE7 | CAE6 | CAE5 | CAE4 | CAE3 | CAE2 | CAE1 | CAE0 |
| | | W | | | | | | | | |
| 0x00A5 | PWMCTL | R | CON67 | CON45 | CON23 | CON01 | PSWAI | PFRZ | 0 | 0 |
| | | W | | | | | | | | |
| 0x00A6 | PWMCLKAB | R | PCLKAB7 | PCLKAB6 | PCLKAB5 | PCLKAB4 | PCLKAB3 | PCLKAB2 | PCLKAB1 | PCLKAB0 |
| | | W | | | | | | | | |
| 0x00A7 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x00A8 | PWMSCLA | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x00A9 | PWMSCLB | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | | | | | | | | |
| 0x00AA | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x00AB | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x00AC | PWMCNT0 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x00AD | PWMCNT1 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x00AE | PWMCNT2 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x00AF | PWMCNT3 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x00B0 | PWMCNT4 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 0x00A0-0x00C7 Pulse Width Modulator 6-Channels (PWM) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00B1 | PWMCNT5 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x00B2 | PWMCNT6 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x00B3 | PWMCNT7 | R | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| | | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x00B4 | PWMPER0 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00B5 | PWMPER1 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00B6 | PWMPER2 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00B7 | PWMPER3 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00B8 | PWMPER4 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00B9 | PWMPER5 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00BA | PWMPER6 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00BB | PWMPER7 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00BC | PWMDTY0 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00BD | PWMDTY1 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00BE | PWMDTY2 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00BF | PWMDTY3 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00C0 | PWMDTY4 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00C1 | PWMDTY5 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00C2 | PWMDTY6 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00C3 | PWMDTY7 | R<br>W | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
| 0x00C4-<br>0x00C7 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x00C8-0x00CF Serial Communication Interface (SCI0) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00C8 | SCI0BDH[1] | R | SBR15 | SBR14 | SBR13 | SBR12 | SBR11 | SBR10 | SBR9 | SBR8 |
| | | W | | | | | | | | |
| 0x00C9 | SCI0BDL[1] | R | SBR7 | SBR6 | SBR5 | SBR4 | SBR3 | SBR2 | SBR1 | SBR0 |
| | | W | | | | | | | | |
| 0x00CA | SCI0CR1[1] | R | LOOPS | SCISWAI | RSRC | M | WAKE | ILT | PE | PT |
| | | W | | | | | | | | |
| 0x00C8 | SCI0ASR1[2] | R | RXEDGIF | 0 | 0 | 0 | 0 | BERRV | BERRIF | BKDIF |
| | | W | | | | | | | | |
| 0x00C9 | SCI0ACR1[2] | R | RXEDGIE | 0 | 0 | 0 | 0 | 0 | BERRIE | BKDIE |
| | | W | | | | | | | | |
| 0x00CA | SCI0ACR2[2] | R | IREN | TNP1 | TNP0 | 0 | 0 | BERRM1 | BERRM0 | BKDFE |
| | | W | | | | | | | | |
| 0x00CB | SCI0CR2 | R | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| | | W | | | | | | | | |
| 0x00CC | SCI0SR1 | R | TDRE | TC | RDRF | IDLE | OR | NF | FE | PF |
| | | W | | | | | | | | |
| 0x00CD | SCI0SR2 | R | AMAP | 0 | 0 | TXPOL | RXPOL | BRK13 | TXDIR | RAF |
| | | W | | | | | | | | |
| 0x00CE | SCI0DRH | R | R8 | T8 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x00CF | SCI0DRL | R | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| | | W | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |

[1] Those registers are accessible if the AMAP bit in the SCI0SR2 register is set to zero

[2] Those registers are accessible if the AMAP bit in the SCI0SR2 register is set to one

## 0x00D0-0x00D7 Serial Communication Interface (SCI1) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00D0 | SCI1BDH[1] | R | SBR15 | SBR14 | SBR13 | SBR12 | SBR11 | SBR10 | SBR9 | SBR8 |
| | | W | | | | | | | | |
| 0x00D1 | SCI1BDL[1] | R | SBR7 | SBR6 | SBR5 | SBR4 | SBR3 | SBR2 | SBR1 | SBR0 |
| | | W | | | | | | | | |
| 0x00D2 | SCI1CR1[1] | R | LOOPS | SCISWAI | RSRC | M | WAKE | ILT | PE | PT |
| | | W | | | | | | | | |
| 0x00D0 | SCI1ASR1[2] | R | RXEDGIF | 0 | 0 | 0 | 0 | BERRV | BERRIF | BKDIF |
| | | W | | | | | | | | |
| 0x00D1 | SCI1ACR1[2] | R | RXEDGIE | 0 | 0 | 0 | 0 | 0 | BERRIE | BKDIE |
| | | W | | | | | | | | |
| 0x00D2 | SCI1ACR2[2] | R | IREN | TNP1 | TNP0 | 0 | 0 | BERRM1 | BERRM0 | BKDFE |
| | | W | | | | | | | | |
| 0x00D3 | SCI1CR2 | R | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| | | W | | | | | | | | |
| 0x00D4 | SCI1SR1 | R | TDRE | TC | RDRF | IDLE | OR | NF | FE | PF |
| | | W | | | | | | | | |

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00D5 | SCI1SR2 | R | AMAP | 0 | 0 | TXPOL | RXPOL | BRK13 | TXDIR | RAF |
| | | W | | | | | | | | |
| 0x00D6 | SCI1DRH | R | R8 | T8 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x00D7 | SCI1DRL | R | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| | | W | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |

[1] Those registers are accessible if the AMAP bit in the SCI0SR2 register is set to zero

[2] Those registers are accessible if the AMAP bit in the SCI0SR2 register is set to one

## 0x00D8-0x00DF Serial Peripheral Interface (SPI) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00D8 | SPICR1 | R | SPIE | SPE | SPTIE | MSTR | CPOL | CPHA | SSOE | LSBFE |
| | | W | | | | | | | | |
| 0x00D9 | SPICR2 | R | 0 | XFRW | 0 | MODFEN | BIDIROE | 0 | SPISWAI | SPC0 |
| | | W | | | | | | | | |
| 0x00DA | SPIBR | R | 0 | SPPR2 | SPPR1 | SPPR0 | 0 | SPR2 | SPR1 | SPR0 |
| | | W | | | | | | | | |
| 0x00DB | SPISR | R | SPIF | 0 | SPTEF | MODF | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x00DC | SPIDRH | R | R15 | R14 | R13 | R12 | R11 | R10 | R9 | R8 |
| | | W | T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 |
| 0x00DD | SPI0DRL | R | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| | | W | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |
| 0x00DE | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x00DF | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x00E0-0x00FF Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00E0-0x00FF | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0100-0x0113 NVM Contol Register (FTMRG) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0100 | FCLKDIV | R | FDIVLD | FDIVLCK | FDIV5 | FDIV4 | FDIV3 | FDIV2 | FDIV1 | FDIV0 |
| | | W | | | | | | | | |
| 0x0101 | FSEC | R | KEYEN1 | KEYEN0 | RNV5 | RNV4 | RNV3 | RNV2 | SEC1 | SEC0 |
| | | W | | | | | | | | |

## 0x0100-0x0113 NVM Contol Register (FTMRG) Map

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0102 | FCCOBIX | R | 0 | 0 | 0 | 0 | 0 | CCOBIX2 | CCOBIX1 | CCOBIX0 |
| | | W | | | | | | | | |
| 0x0103 | FRSV0 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0104 | FCNFG | R | CCIE | 0 | 0 | IGNSF | 0 | 0 | FDFD | FSFD |
| | | W | | | | | | | | |
| 0x0105 | FERCNFG | R | 0 | 0 | 0 | 0 | 0 | 0 | DFDIE | SFDIE |
| | | W | | | | | | | | |
| 0x0106 | FSTAT | R | CCIF | 0 | ACCERR | FPVIOL | MGBUSY | RSVD | MGSTAT1 | MGSTAT0 |
| | | W | | | | | | | | |
| 0x0107 | FERSTAT | R | 0 | 0 | 0 | 0 | 0 | 0 | DFDIF | SFDIF |
| | | W | | | | | | | | |
| 0x0108 | FPROT | R | FPOPEN | RNV6 | FPHDIS | FPHS1 | FPHS0 | FPLDIS | FPLS1 | FPLS0 |
| | | W | | | | | | | | |
| 0x0109 | DFPROT | R | DPOPEN | 0 | 0 | 0 | DPS3 | DPS2 | DPS1 | DPS0 |
| | | W | | | | | | | | |
| 0x010A | FCCOBHI | R | CCOB15 | CCOB14 | CCOB13 | CCOB12 | CCOB11 | CCOB10 | CCOB9 | CCOB8 |
| | | W | | | | | | | | |
| 0x010B | FCCOBLO | R | CCOB7 | CCOB6 | CCOB5 | CCOB4 | CCOB3 | CCOB2 | CCOB1 | CCOB0 |
| | | W | | | | | | | | |
| 0x010C | FRSV1 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x010D | FRSV2 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x010E | FRSC3 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x010F | FRSV4 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0110 | FOPT | R | NV7 | NV6 | NV5 | NV4 | NV3 | NV2 | NV1 | NV0 |
| | | W | | | | | | | | |
| 0x0111 | FRSV5 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0112 | FRSV6 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0113 | FRSV7 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0114-0x011F Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0114-0x011F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0120 Interrupt Vector Base Register

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0120 | IVBR | R<br>W | IVB_ADDR[7:0] | | | | | | | |

## 0x0121-0x013F Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0121-<br>0x013F | Reserved | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 0x0140-0x0147 High Side Drivers (HSDRV)

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0140 | HSDR | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | HSDR1 | HSDR0 |
| 0x0141 | HSCR | R<br>W | 0 | 0 | HSOCME1 | HSOCME0 | 0 | 0 | HSE1 | HSE0 |
| 0x0142 | Reserved | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0143 | Reserved | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0144 | Reserved | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0145 | Reserved | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0146 | HSIE | R<br>W | HSOCIE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0147 | HSIF | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | HSOCIF1 | HSOCIF0 |

## 0x0148-0x014F Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0148-<br>0x014F | Reserved | R<br>W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 0x0150-0x0157 Low Side Drivers (LSDRV)

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| 0x0150 | LSDR | R | 0 | 0 | 0 | 0 | 0 | 0 | LSDR1 | LSDR0 |
| | | W | | | | | | | | |
| 0x0151 | LSCR | R | 0 | 0 | 0 | 0 | LSOLE1 | LSOLE0 | LSE1 | LSE0 |
| | | W | | | | | | | | |
| 0x0152 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0153 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0154 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0155 | LSSR | R | 0 | 0 | 0 | 0 | 0 | 0 | LSOL1 | LSOL0 |
| | | W | | | | | | | | |
| 0x0156 | LSIE | R | LSOCIE | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0157 | LSIF | R | 0 | 0 | 0 | 0 | 0 | 0 | LSOCIF1 | LSOCIF0 |
| | | W | | | | | | | | |

## 0x0158-0x015F Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x058-<br>0x015F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0160-0x0167 LIN Physical Layer (LINPHY)

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0160 | LPDR | R | 0 | 0 | 0 | 0 | 0 | 0 | LPDR1 | LPDR0 |
| | | W | | | | | | | | |
| 0x0161 | LPCR | R | 0 | 0 | 0 | 0 | LPE | RXONLY | LPWUE | LPPUE |
| | | W | | | | | | | | |
| 0x0162 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0163 | LPSLRM | R | LPDTDIS | 0 | 0 | 0 | 0 | 0 | LPSLR1 | LPSLR0 |
| | | W | | | | | | | | |
| 0x0164 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0160-0x0167 LIN Physical Layer (LINPHY)

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0165 | LPSR | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LPOC |
| | | W | | | | | | | | |
| 0x0166 | LPIE | R | LPDTIE | LPOCIE | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0167 | LPIF | R | LPDTIF | LPOCIF | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0168-0x016F Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0168-0x016F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0170-0x0177 Supply Voltage Sense (BATS)

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0170 | BATE | R | 0 | 0 | BVHS | BVLS | BSUAE | BSUSE | 0BSEAE | BSESE |
| | | W | | | | | | | | |
| 0x0171 | BATSR | R | 0 | 0 | 0 | 0 | 0 | 0 | BVHC | BVLC |
| | | W | | | | | | | | |
| 0x0172 | BATIE | R | 0 | 0 | 0 | 0 | 0 | 0 | BVHIE | BVLIE |
| | | W | | | | | | | | |
| 0x0173 | BATIF | R | 0 | 0 | 0 | 0 | 0 | 0 | BVHIF | BVLIF |
| | | W | | | | | | | | |
| 0x0174 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0175 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0176 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0177 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0178-023F Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0178-0x023F | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0240 -0x027F Port Integration Module (PIM) Map 4 of 4

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0240 | PTT | R | 0 | 0 | 0 | 0 | PTT3 | PTT2 | PTT1 | PTT0 |
| | | W | | | | | | | | |
| 0x0241 | PTIT | R | 0 | 0 | 0 | 0 | PTIT3 | PTIT2 | PTIT1 | PTIT0 |
| | | W | | | | | | | | |
| 0x0242 | DDRT | R | 0 | 0 | 0 | 0 | DDRT3 | DDRT2 | DDRT1 | DDRT0 |
| | | W | | | | | | | | |
| 0x0243 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0244 | PERT | R | 0 | 0 | 0 | 0 | PERT3 | PERT2 | PERT1 | PERT0 |
| | | W | | | | | | | | |
| 0x0245 | PPST | R | 0 | 0 | 0 | 0 | PPST3 | PPST2 | PPST1 | PPST0 |
| | | W | | | | | | | | |
| 0x0246 | PTTRR0 | R | PTTRR07 | PTTRR06 | PTTRR05 | PTTRR04 | PTTRR03 | PTTRR02 | PTTRR01 | PTTRR00 |
| | | W | | | | | | | | |
| 0x0247 | PTTRR1 | R | 0 | 0 | PTTRR15 | PTTRR14 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0240 -0x027F Port Integration Module (PIM) Map 4 of 4

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0248 | PTS | R | 0 | 0 | PTS5 | PTS4 | PTS3 | PTS2 | PTS1 | PTS0 |
| | | W | | | | | | | | |
| 0x0249 | PTIS | R | 0 | 0 | PTIS5 | PTIS4 | PTIS3 | PTIS2 | PTIS1 | PTIS0 |
| | | W | | | | | | | | |
| 0x024A | DDRS | R | 0 | 0 | DDRS5 | DDRS4 | DDRS3 | DDRS2 | DDRS1 | DDRS0 |
| | | W | | | | | | | | |
| 0x024B | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x024C | PERS | R | 0 | 0 | PERS5 | PERS4 | PERS3 | PERS2 | PERS1 | PERS0 |
| | | W | | | | | | | | |
| 0x024D | PPSS | R | 0 | 0 | PPSS5 | PPSS4 | PPSS3 | PPSS2 | PPSS1 | PPSS0 |
| | | W | | | | | | | | |
| 0x024E | WOMS | R | 0 | 0 | WOMS5 | WOMS4 | WOMS3 | WOMS2 | WOMS1 | WOMS0 |
| | | W | | | | | | | | |
| 0x024F | MODRR2 | R | PTSRR7 | 0 | PTSRR5 | PTSRR4 | PTSRR3 | PTSRR2 | PTSRR1 | PTSRR0 |
| | | W | | | | | | | | |
| 0x0250-0x0257 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0258 | PTP | R | 0 | 0 | PTP5 | PTP4 | PTP3 | PTP2 | PTP1 | PTP0 |
| | | W | | | | | | | | |
| 0x0259 | PTIP | R | 0 | 0 | PTIP5 | PTIP4 | PTIP3 | PTIP2 | PTIP1 | PTIP0 |
| | | W | | | | | | | | |
| 0x025A | DDRP | R | 0 | 0 | DDRP5 | DDRP4 | DDRP3 | DDRP2 | DDRP1 | DDRP0 |
| | | W | | | | | | | | |
| 0x025B | RDRP | R | 0 | 0 | 0 | 0 | 0 | RDRP2 | RDRP1 | RDRP0 |
| | | W | | | | | | | | |
| 0x025C | PERP | R | 0 | 0 | PERP5 | PERP4 | PERP3 | PERP2 | PERP1 | PERP0 |
| | | W | | | | | | | | |
| 0x025D | PPSP | R | 0 | 0 | PPSP5 | PPSP4 | PPSP3 | PPSP2 | PPSP1 | PPSS0 |
| | | W | | | | | | | | |
| 0x025E | PIEP | R | OCIE | 0 | PIEP5 | PIEP4 | PIEP3 | PIEP2 | PIEP1 | PIEP0 |
| | | W | | | | | | | | |
| 0x025F | PIFP | R | OCIF | 0 | PIFP5 | PIFP4 | PIFP3 | PIFP2 | PIFP1 | PIFP0 |
| | | W | | | | | | | | |

## 0x0240 -0x027F Port Integration Module (PIM) Map 4 of 4

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0260-0x0268 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0269 | PTIL | R | 0 | 0 | 0 | 0 | PTIL3 | PTIL2 | PTIL1 | PTIL0 |
| | | W | | | | | | | | |
| 0x026A | DDRL | R | 0 | 0 | 0 | 0 | DDRL3 | DDRL2 | DDRL1 | DDRL0 |
| | | W | | | | | | | | |
| 0x026B | PTAL | R | 0 | 0 | 0 | 0 | PTAENL | 0 | PTAL1 | PTAL0 |
| | | W | | | | | | | | |
| 0x026C | PIRL | R | 0 | 0 | 0 | 0 | PIRL3 | PIRL2 | PIRL1 | PIRL0 |
| | | W | | | | | | | | |
| 0x026D | PPSL | R | 0 | 0 | 0 | 0 | PPSL3 | PPSL2 | PPSL1 | PPSL0 |
| | | W | | | | | | | | |
| 0x026E | PIEL | R | 0 | 0 | 0 | 0 | PIEL3 | PIEL2 | PIEL1 | PIEL0 |
| | | W | | | | | | | | |
| 0x026F | PIFL | R | 0 | 0 | 0 | 0 | PIFL3 | PIFL2 | PIFL1 | PIFL0 |
| | | W | | | | | | | | |
| 0x0270 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0271 | PT1AD | R | 0 | 0 | PT1AD05 | PT1AD04 | PT1AD03 | PT1AD02 | PT1AD01 | PT1AD00 |
| | | W | | | | | | | | |
| 0x0272 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0273 | PTI1AD | R | 0 | 0 | PTI1AD5 | PTI1AD4 | PTI1AD3 | PTI1AD2 | PTI1AD1 | PTI1AD0 |
| | | W | | | | | | | | |
| 0x0274 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0275 | DDR1AD | R | 0 | 0 | DDR1AD5 | DDR1AD4 | DDR1AD3 | DDR1AD2 | DDR1AD1 | DDR1AD0 |
| | | W | | | | | | | | |
| 0x0276-0x0278 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x0279 | PER1AD | R | 0 | 0 | PER1AD5 | PER1AD4 | PER1AD3 | PER1AD2 | PER1AD1 | PER1AD0 |
| | | W | | | | | | | | |
| 0x027A | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x027B | PPS1AD | R | 0 | 0 | PPS1AD5 | PPS1AD4 | PPS1AD3 | PPS1AD2 | PPS1AD1 | PPS1AD0 |
| | | W | | | | | | | | |
| 0x027C | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x027D | PIE1AD | R | 0 | 0 | PIE1AD5 | PIE1AD4 | PIE1AD3 | PIE1AD2 | PIE1AD1 | PIE1AD0 |
| | | W | | | | | | | | |
| 0x027E | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x027F | PIF1AD | R | 0 | 0 | PIF1AD5 | PIF1AD4 | PIF1AD3 | PIF1AD2 | PIF1AD1 | PIF1AD0 |
| | | W | | | | | | | | |

## 0x0280-0x02EF Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0280-0x02EF | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x02F0-0x02FF Clock and Power Management Unit (CPMU) Map 2 of 2

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02F0 | CPMUHTCL | R | 0 | 0 | VSEL | 0 | HTE | HTDS | HTIE | HTIF |
| | | W | | | | | | | | |
| 0x02F1 | CPMULVCTL | R | 0 | 0 | 0 | 0 | 0 | LVDS | LVIE | LVIF |
| | | W | | | | | | | | |
| 0x02F2 | CPMUAPICTL | R | APICLK | 0 | 0 | APIES | APIEA | APIFE | APIE | APIF |
| | | W | | | | | | | | |
| 0x02F3 | CPMUACLKTR | R | ACLKTR5 | ACLKTR4 | ACLKTR3 | ACLKTR2 | ACLKTR1 | ACLKTR0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x02F4 | CPMUAPIRH | R | APIR15 | APIR14 | APIR13 | APIR12 | APIR11 | APIR10 | APIR9 | APIR8 |
| | | W | | | | | | | | |
| 0x02F5 | CPMUAPIRL | R | APIR7 | APIR6 | APIR5 | APIR4 | APIR3 | APIR2 | APIR1 | APIR0 |
| | | W | | | | | | | | |
| 0x02F6 | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x02F7 | CPMUHTTR | R | HTOE | 0 | 0 | 0 | HTTR3 | HTTR2 | HTTR1 | HTTR0 |
| | | W | | | | | | | | |
| 0x02F8 | CPMU IRCTRIMH | R | TCTRIM[3:0] | | | | | 0 | IRCTRIM[9:8] | |
| | | W | | | | | | | | |
| 0x02F9 | CPMU IRCTRIML | R | IRCTRIM[7:0] | | | | | | | |
| | | W | | | | | | | | |
| 0x02FA | CPMUOSC | R | OSCE | 0 | 0 | Reserved | | | | |
| | | W | | | | | | | | |
| 0x02FB | CPMUPROT | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PROT |
| | | W | | | | | | | | |
| 0x02FC | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x02FD | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x02FE | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |
| 0x02FF | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## 0x0300-0x03FF Reserved

| Address | Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0300-0x03FF | Reserved | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | W | | | | | | | | |

## *How to Reach Us:*

**USA/Europe/Locations not listed:**
Freescale Semiconductor Literature Distribution
P.O. Box 5405, Denver, Colorado 80217
1-800-521-6274 or 480-768-2130

**Japan:**
Freescale Semiconductor Japan Ltd.
SPS, Technical Information Center
3-20-1, Minami-Azabu
Minato-ku
Tokyo 106-8573, Japan
81-3-3440-3569

**Asia/Pacific:**
Freescale Semiconductor H.K. Ltd.
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T. Hong Kong
852-26668334

*Learn More:*
For more information about Freescale Semiconductor products, please visit
**http://www.freescale.com**

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

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

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.
© Freescale Semiconductor, Inc. 2013

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View S9S12VR48AF0MLCR on WIN SOURCE
- NXP / Nexperia Information

## Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management